MULTI Compiler 2015.1 Release Notes for Embedded V850 and RH850

Green Hills Software
30 West Sola Street
Santa Barbara, California 93101
USA
Tel: 805-965-6044
Fax: 805-965-6343
www.ghs.com

LEGAL NOTICES AND DISCLAIMERS

PubID: release_notes_v800-544152
Branch: http://toolsvc/branches/release-branch-70-bto
Date: October 21, 2015

Contents

Chapter 1

Changes in Version 2015.1

Contents

System Requirements

The MULTI IDE may have different system requirements than the Compiler. For information about IDE requirements, see the release notes for the MULTI IDE.

Windows

Running a full installation on Windows requires:

- A modern, x86 processor.
- One of the following:
  - Windows XP Service Pack 2 or greater (IA-32, 32-bit mode only).

    

    Note

    Support for Windows XP is deprecated and will be removed in a future release.

  - Windows Vista (64- or 32-bit mode).
  - Windows 7 (64- or 32-bit mode).
  - Windows 8 (64- or 32-bit mode).
  - Windows 10 (64- or 32-bit mode).
- 4 GB of RAM (16 GB is recommended).
- 2 GB of free disk space per installation (100 GB of free space and a solid state drive are recommended).
- A monitor running at a display resolution of 1024x768 or higher. (Two high-resolution monitors are recommended.)
- A DVD-ROM drive or access to the Green Hills FTP site. If neither is available, contact Green Hills Support.
- An Ethernet connection.
- The ability to write to the Windows System directories. On most Windows systems, you must also be a member of the Administrators group. End users must have full access to installed files.

Linux

Running a full installation on Linux requires:

- A modern, multi-core x86 processor.
- One of the following:
  - Ubuntu 14.04.1 LTS (64- or 32-bit mode).
  - Ubuntu 12.04.5 LTS (64- or 32-bit mode).
  - Ubuntu 10.04 (64- or 32-bit mode).
  - CentOS 7.x (64- or 32-bit mode).
  - CentOS 6.x (64- or 32-bit mode).
- 32-bit compatibility libraries are required when running on 64-bit systems. If you are using Ubuntu 14, run:

  ```
  sudo dpkg --add-architecture i386
  sudo apt-get install libc6:i386 libncurses5:i386
  sudo apt-get install libstdc++6:i386 libx11-6:i386 lib32z1
  ```

  If you are using Ubuntu 12 or earlier, run:

  ```
  sudo apt-get install ia32-libs
  ```

- 4 GB of RAM (16 GB is recommended).
- 2 GB of free disk space per installation (100 GB of free space and a solid state drive are recommended).
- A monitor running at a resolution of 1024x768 or higher. (Two high-resolution monitors are recommended.)
- A mounted DVD-ROM drive or access to the Green Hills FTP site. If neither is available, contact Green Hills Support.
- An Ethernet connection.
- Write permissions to the installation directory.

Solaris

Running a full installation on Solaris requires:

- A sun4u SPARC-based workstation.
- One of the following:
  - Solaris 10 (64- or 32-bit mode).
  - Solaris 9 (64- or 32-bit mode).
  - Solaris 8 (64- or 32-bit mode).

Note

Native development of 64-bit applications is not supported on Solaris.

- 256 MB of physical memory (1 GB is recommended).
- 2 GB of free disk space per installation.
- A graphical display running the X Windowing System.
- A mounted DVD-ROM drive or access to the Green Hills FTP site. If neither is available, contact Green Hills Support.
- An Ethernet connection.
- Write permissions to the installation directory.

Note

Support for all Solaris-hosted toolchain components is deprecated and may be removed in a future release.

Product Compatibility

Green Hills Compiler 2015.1 is officially supported with:

- MULTI version 6.1.6.
- MULTI version 7.x.
- ThreadX 5.5.x or newer.
- INTEGRITY-MZ version 15.0.1.
- INTEGRITY version 5.x.
- INTEGRITY version 10.0.2.
- INTEGRITY version 11.0.2, 11.0.4, 11.2.4, and 11.4.4.

- u-velOSity version 2.2.9 and 2.6.2.

For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Warning

Using Compiler 2015.1 with MULTI 6 or later and a Green Hills Debug Probe requires Green Hills Debug Probe software version 4.4.4 or newer. Do not install Green Hills Debug Probe software version 4.2.x or earlier over your Compiler 2015.1 installation if you intend to use it with MULTI 6 or later.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Project Build Configurations

Compiler 2015.1 makes it much easier to set up multiple build configurations of the same project. In previous releases, you might have created multiple Top Projects—each of which included shared source—to manually set up multiple build configurations. Now you can easily create your own build configurations within a single project structure. For more information, see the description of **defineConfig** in "Group 1 Directives" in Appendix C, "Green Hills Project File Format" in *MULTI: Building Applications for Embedded V850 and RH850*.

Note

This feature is only available with MULTI 7.x and newer.

Automated Setup of Multicore Targets

Two new options, **-ghsmc_file** and **-ghsmc_core_count**, enable the automated setup of multicore targets. Note that these options are only available for MULTI 7 or newer. For more information, see "Multicore Configuration File" in Chapter 3,

"Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*. See also "Multicore Core Count" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Using Interprocedural Optimizations with Makefiles

Interprocedural optimizations are now officially supported for programs built using makefiles. See "Interprocedural Optimizations" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*, and see Example 1.5. Using Interprocedural Optimizations With Makefiles in *MULTI: Building Applications for Embedded V850 and RH850*.

Duplicate Symbol Detection in Linker and Archiver

The linker now has an option to detect symbols that are present in more than one library being linked. While legal, and often useful, symbols being defined in multiple libraries can also be the source of subtle problems. The options **-link_reject_duplicate_lib_syms**, **-link_allow_duplicate_ghs_syms**, **-link_allow_duplicate_linkonce_syms**, and **-duplicate_whitelist** are added to control the behavior of the linker. For more information, see "Report an Error When Multiple Libraries on a Link Line Contain a Definition of the Same Symbol" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

The archiver now has an option to detect multiple definitions of a symbol when creating a library. The options **-reject_duplicates**, **-allow_cxx_duplicates**, and **-allow_linkonce_duplicates** are added to control the behavior of the archiver. For more information, see "Error on Duplicate Symbols in an Archive" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Preserving Temporary Preprocessor Files

Two new options, **-keepcppfiles** and **-keepasmfiles** can be used to keep the preprocessor output from the compiler and the assembly file, respectively. See "Temporary Preprocessor Output" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* and "Temporary

Assembly Output" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Support for Stripping Debug Information From Relocatable Objects and Libraries

A new option, **-stripreloc** has been added to the **gstrip** utility. When this option is passed, **gstrip** can be run on relocatable object files and libraries, in addition to executable files. When stripping relocatable object files and libraries, **gstrip** will preserve any symbols necessary for linking. See "The gstrip Utility Program" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Section Sorting in Linker

The linker now supports the new `SORT` and `SORT_BY_NAME` commands in linker directives files to allow sections included with wildcards to be sorted by name. See "Using Wildcards in Section Inclusion Commands" in Chapter 9, "The elxr Linker" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Assigning Variables to SDA and ZDA From a Text File

Two new options, **-sda_file** and **-zda_file**, enable assigning variables to the Small Data Area or the Zero Data Area by listing them in a control file. See "Small Data Area File" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* and "Zero Data Area File" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Generating Both Numerically and Alphabetically Sorted Symbols in Map File

A new option, **-Man** is now supported by the linker to produce both a list of symbols sorted alphabetically (as with the **-Ma** option) and numerically (as with the **-Mn** option). See "Map File Sorting" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Annotating ELF Files with Toolchain Version Information

A new option, **-version_info** is now supported to add information about the compiler and assembler version and command line options to the `.comment` section of the output file. See "Write Version Info To .comment section" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Link-time Global Variable Type Checking

Two new options, **-globalcheck=[full|normal|none]** and **-globalcheck_qualifiers**, are now supported to perform link-time checks of global variable types. When **-globalcheck** is enabled, the compiler generates information about global variable data-types at their definition and uses. The linker checks that the types are compatible with respect to size and signedness. With **-globalcheck_qualifiers**, the qualifiers (such as `const` and `volatile`) are also checked. See "Link-Time Global Variable Type Checking" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* and "Link-Time Global Variable Type Qualifiers" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Support for COMMON Variables in the Zero Data Area on Additional Targets

Support for `COMMON` variables in the Zero Data Area is now available on SH, TriCore, and Power Architecture targets. See "Allow Common Variables in ZDA" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Aligned Data Sections

A new option, **-split_data_sections_by_alignment**, is now supported to allocate variables into aligned data sections. See "Aligned Data Sections" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Individual Function and Variable Sections

Individual function and variable sections are now supported for assigning functions (including static functions), global variables and static variables (including function scope local static variables) in default sections or custom sections to their own individual sections by the compiler. These individual sections can then be placed at arbitrary addresses via linker directives file or be merged back to the sections they derived from implicitly at link time. For more information, see "Individual Function and Variable Sections" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850*.

64-Bit Linker and dblink on Windows and Linux Hosts

The `elxr` linker and **dblink** debug information linker are now supported in 64-bit mode on 64-bit Windows and Linux hosts. For more information, see "Use 64-bit Toolchain Components" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Options to Modify Message Severity Supported for asm and elxr

`asm` and `elxr` now support setting discretionary diagnostic messages to different levels of severity. For more information, see "Toolchain Messages" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Specify the Output Name of a Dependency File

A new option, **-MF**, has been added to specify the output name of a dependency file. This name will be used for files created by the **-MD** or **-MMD** options. For more information, see "Generating Dependency and Header File Information" in Chapter 1, "The Compiler Driver" in *MULTI: Building Applications for Embedded V850 and RH850*.

Additional Documentation of Diagnostic Messages for Assemblers and elxr

Additional documentation of diagnostic messages has been added to the *MULTI: Toolchain Error Messages* book.

Ability to Specify Diagnostics by Symbolic Tag Names

Several constructs that previously required a diagnostic to be specified by number now accept symbolic tag names as well. These constructs include:

- The **--diag_[suppress|remark|warning|error]** compiler options. (See "Varying Message Severity" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.)
- The `ghs nowarning` `#pragma` directive. (See "Green Hills Extension Pragma Directives" in Chapter 15, "Macros, Pragma Directives, and Intrinsics" in *MULTI: Building Applications for Embedded V850 and RH850*.)

For example, these two options are now equivalent:

- **--diag_error=9**
- **--diag_error=nested_comment**

See the *MULTI: Toolchain Error Messages* book for the new symbolic tag names.

Support for %= Format String in GNU Extended Assembly

The compiler now accepts the `%=` special format string in GNU Extended Assembly. The string expands to a number that is unique to each specific instantiation of an `asm` statement. For more information, see Chapter 19, "GNU Extended Assembly" in *MULTI: Building Applications for Embedded V850 and RH850*.

Changes in Behavior

Interfacing with the Function Entry/Exit Feature

`#pragma ghs function_entry_exit` is now `#pragma ghs entry_exit_history`. In addition, the following now use the term "`history`" in place of "`log`":

- **-gen_entry_exit_log**
- **-no_gen_entry_exit_log**
- **-gen_entry_exit_arg_log**
- **-no_gen_entry_exit_arg_log**
- `attribute ((entry_exit_log))`

For example, **-gen_entry_exit_log** is now **-gen_entry_exit_history**.

Diagnostic Messages When Building INTEGRITY 5.x and 10.x

In some rare cases, building INTEGRITY 5.x or 10.x from source may generate additional diagnostic messages. To disable these messages, see "Varying Message Severity" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* or contact Green Hills support.

bcmp and bcopy now treat their size arguments as unsigned

The library functions `bcmp()` and `bcopy()` now treat a size argument larger than `INT_MAX` as representing a large size, rather than a negative one. This will cause a long run time due to iteration over large arrays. Previously, both functions would do nothing if their size arguments were larger than `INT_MAX`.

Removed and Deprecated Features

The following feature has been deprecated:

- Support for the **--prelink_objects** option. As an alternative for build systems that use **clearmake**, please see the **--link_once_templates** option.

Chapter 2

Changes in Version 2014.1

Contents

Product Compatibility

Green Hills Compiler 2014.1 is officially supported with INTEGRITY versions 5.x, 10.0.2, 11.0.2, 11.0.4, and 11.2.2. For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Green Hills Compiler 2014.1 is officially supported with ThreadX 5.5.x and newer.

Warning

Using Compiler 2014.1 with MULTI 6 or later and a Green Hills Debug Probe requires Green Hills Debug Probe software version 4.4.4 or newer. Do not install Green Hills Debug Probe software version 4.2.x or earlier over your Compiler 2015.1 installation if you intend to use it with MULTI 6 or later.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Cross-Project Implicit Dependency Analysis

Implicit dependency analysis is now supported across Top Projects. As a result, building a Top Project that depends on libraries or objects in another Top Project now builds the defined dependencies. Previously, you had to remember to build these dependencies yourself. This feature is especially useful for INTEGRITY users whose applications depend on source code in the INTEGRITY project tree. For more information, see “Implicit Dependency Analysis” in Chapter 11, “Utility Programs” in *MULTI: Building Applications for Embedded V850 and RH850*.

New Undefined Behavior Checking

The compiler now checks for undefined behavior resulting from improperly sequenced expressions. By default, these diagnostics are issued as warnings, but they may be issued as errors via the **--diag_error=2017,2018** option. Alternatively,

these diagnostics may be suppressed altogether via the **--diag_suppress=2017,2018** option.

For more information, see "Sequence Related Undefined Behavior Checking" in Chapter 12, "Green Hills C" in *MULTI: Building Applications for Embedded V850 and RH850*.

Range-Based For Loops

This release adds support for C++11-style range-based for loops. For more information, see "Range-Based For Loops" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

auto Type Keyword

This release allows you to use the `auto` keyword as a type specifier, as opposed to a storage class, for C++11-style type deduction. For more information, see "auto Type Keyword Support" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Improved Inlining in C++

The heuristics used by the compiler to determine whether or not to inline functions in C++ have been updated. Depending upon the circumstances, this may result in slight improvements in both size and speed in C++ programs.

Additionally, when using GNU C++ compatibility mode, function templates are now considered for inlining under a broader range of compiler options. Previously, function templates were only considered for inlining in GNU C++ compatibility mode when the function qualified for automatic inlining or when either intermodule inlining or **--max_inlining** was enabled for the compilation.

For more information, see:

- "Automatic Inlining" in Chapter 4, "Optimizing Your Programs" in *MULTI: Building Applications for Embedded V850 and RH850*,
- "Intermodule Inlining" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*, and

- "C++ Inlining Level" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Ability to Check for Use of Floating Point

The compiler is now able to give a warning or error for the use of floating point. Furthermore, this diagnostic can be given for the use of double precision and long double precision floating type, or for all floating point types. Previously, the only checking the compiler offered was to give a fatal error for all floating point types. For more information, see "Do not allow types double or long double" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Calls to pow(x, y) Now Optimized in Certain Cases

If the optimization strategy **-Ospace**, **-Ogeneral**, or **-Ospeed** is enabled, and **-ffunctions** is enabled, calls to `pow(x, y)` will be optimized if either `x` or `y` has one of the following constant integer values:

- If `x` has constant value `2` or `10`, `pow` is converted to an equivalent call to `exp2` or `exp`.
- If `y` has constant value `2`, `3`, ... `8`, `pow` is converted to a series of multiplies.

If the optimization strategy **-Ospace**, **-Ogeneral**, or **-Ospeed** is enabled, and the optimization is not disabled by the new option **-Ono_inline_constant_math**, calls to `exp2(x)` (where `x` is a constant integer) and calls to `sqrt(x)` (where `x` is a finite, positive constant) will be replaced with an equivalent constant value.

Support for FPSIMD in RH850

This release adds support for the Floating-Point SIMD coprocessor (FPSIMD) in RH850 devices. For more information, see "Floating-Point SIMD" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Inlining Constant Math Functions

Calls to certain math functions with constant parameters are now optimized into a constant expression. For more information, see "Inlining Constant Math Functions" in Chapter 4, "Optimizing Your Programs" in *MULTI: Building Applications for Embedded V850 and RH850*.

Changes in Behavior

Changes in the Behavior of Some #pragma Directives

The following `#pragma` directives now have no effect when used inside a function. Additionally, using any of these `#pragma` directives inside a function will trigger a compiler warning:

- `#pragma ghs extra_stack`
- `#pragma ghs max_stack`
- `#pragma ghs max_instances`
- `#pragma alignfunc`

Previously, when used inside a function, these `#pragma` directives silently affected the next function declared (not the enclosing function).

index() and rindex() Prototypes Now Hidden

The prototypes for functions `index()` and `rindex()`, which are non-standard, are no longer visible unless:

- The `__BSD_INDEX_PROTOTYPE` symbol is defined, or
- The operating system is INTEGRITY and `_POSIX_SOURCE` is not defined.

Change in Default Behavior for Virtual Function Table Size

The **--large_vtbl_offsets** option (see "Virtual Function Table Offset Size" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*) is now enabled by default. Code built with this option

is not compatible with INTEGRITY 11 or earlier, with code built using Green Hills Compiler 2012.5 or earlier, or with code built using the **--no_large_vtbl_offsets** option. It is also not compatible with versions of MULTI that shipped with prior releases, such as MULTI 6.1.4.

The **--no_large_vtbl_offsets** option is now deprecated.

New C++ Restrictions Regarding --large_vtbl_offsets

When using **--large_vtbl_offsets** (either explicitly or by default), you must use a C++ library with the same exception handling support as the compiled code. In general, the driver will select an appropriate library for you, but if you explicitly select a library (for example, with **--stdl** or **--stdle**), you must ensure that it matches your selected level of exception handling.

If you use an invalid combination of options according to the above rules (for example, by specifying **--stdl** and **--exceptions**), the driver will issue a diagnostic similar to the one below:

```
Error: Option "--stdl" requires option "--no_exceptions"
```

See “Specifying C++ Libraries” in Chapter 13, “Green Hills C++” in *MULTI: Building Applications for Embedded V850 and RH850*, and see “C++ Exception Handling” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*.

Furthermore, the **--large_vtbl_offsets** option now requires that the **--link_once_templates** option is also used. Note that while **--link_once_templates** is also enabled by default, manually disabling it with the **--no_link_once_templates** option will imply **--no_large_vtbl_offsets**. Explicitly specifying both **--no_link_once_templates** and **--large_vtbl_offsets** is an invalid combination.

See “Virtual Function Table Offset Size” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*, and see “Link-Once Template Instantiation” in Chapter 13, “Green Hills C++” in *MULTI: Building Applications for Embedded V850 and RH850*.

Link-Once Template Instantiation Now Enabled by Default

The **--link_once_templates** option is now enabled by default.

Note

For targets that support large virtual function table offsets, disabling link-once template instantiation (via the **--no_link_once_templates** option) will imply **--no_large_vtbl_offsets**.

Limited Support for C-Like Structure Packing in C++

The compiler now has more explicit support for C-like structure packing in C++. In general, only C-like usage of packed structures (or classes) is supported. Most C++ specific features (for example, references, pointers to members, etc.) are not supported in combination with structure packing, nor were they before.

Packing of non-POD (Plain Old Data) classes is not supported unless the **-misalign_pack** option is enabled.

In conjunction with the above restriction, a warning will be issued if you explicitly attempt to pack a non-POD. Additionally, a warning will be issued if the current structure packing level, which is determined by the **-pack=*n*** command line option and any active `#pragma pack` directives, would have otherwise had an effect on a non-POD (that is, only if one of the fields of a non-POD would have had its alignment changed by the current structure packing level).

Note

A “Plain Old Data” class is a class that has no constructors, no destructors, no private non-static data members, no protected non-static data members, no base classes, no virtual functions, and no non-POD members.

-gs No Longer Implies -gtws

The option **-gs** no longer implies **-gtws**. This improves code size and speed.

Instantiate Extern Inline Now Enabled by Default

The **--instantiate_extern_inline** option is now enabled by default.

While **--instantiate_extern_inline** usually results in smaller program sizes, the results depend upon the code being compiled. In general, **--instantiate_extern_inline** leads to better program size when the majority of non-static inline functions have more than one call site (across the entire program). However, when the majority of non-static inline functions have exactly one call site, this option may lead to worsened program size. This can be mitigated (without disabling **--instantiate_extern_inline**) by marking inline functions as static when they are only used in one module.

sqrt() and sqrtf() Now Return NAN for Negative Input

Library functions `sqrt()` and `sqrtf()` now return a quiet `NAN` instead of `0.0` when the input is less than zero.

Removed and Deprecated Features

The following features are no longer available:

- The **regnum** debug server command
- The **rg** debug server command

The following feature has been deprecated:

- Support for Exported Templates (**--export**).

Chapter 3

Changes in Version 2013.5

Contents

Product Compatibility

Green Hills Compiler 2013.5 is officially supported with INTEGRITY versions 5.x, 10.0.2, 11.0.2, 11.0.4, and 11.2.2. For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Green Hills Compiler 2013.5 is officially supported with ThreadX 5.5.x and newer.

Warning

Using Compiler 2013.5 with MULTI 6 or later and a Green Hills Debug Probe requires Green Hills Debug Probe software version 4.4.4 or newer. Do not install Green Hills Debug Probe software version 4.2.x or earlier over your Compiler 2015.1 installation if you intend to use it with MULTI 6 or later.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Static Assertions

This release adds a C11/C++11 style `static_assert` in permissive C and C++ language dialects, that allow you to statically check conditions at compile time with a standard mechanism. For more information, see “Static Assertions” in Chapter 15, “Macros, Pragma Directives, and Intrinsics” in *MULTI: Building Applications for Embedded V850 and RH850*.

Specify a Compiler Inside a Project File

You can now specify the location of the Compiler installation directory that should be used to build your project from the Project Manager using the **gbuildDir** option inside a project (**.gpj**) file as a Group 1 option. For example:

```
#!gbuild
gbuildDir=c:\ghs\comp_201354
```

```
primaryTarget=ppc_standalone.tgt
[Project]
...
```

This option only affects projects built from the Project Manager; it is ignored by the **gbuild** utility program.

With MULTI 6 and earlier, this option silently defaults to the Compiler installation directory that the MULTI IDE is linked with in the event that a bad path is supplied.

For more information, see "Group 1 Directives" in Appendix C, "Green Hills Project File Format" in *MULTI: Building Applications for Embedded V850 and RH850*.

Ability to Specify the Size of wchar_t

You can now select the size of the `wchar_t` type using the **-wchar_u16** and **-wchar_s32** options to conserve memory when necessary. For more information, see "WChar Size" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Stack Smashing Protection

The new **-stack_protector** option provides protection against stack smashing attacks by using a *stack canary* — a random number on the stack — that is compared to an expected value in function epilogues. For more information, see "Check for Stack Smashing Attacks" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Updated gstack Utility Program for Better C++ Analysis

The **gstack** utility program has been updated to provide better support for analyzing function calls involving C++ virtual functions, startup code, and libraries. For more information about **gstack**, see "The gstack Utility Program" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850*.

New Attribute and Intrinsic to Remove Specific Functions from Profiling Reports

The new `ghs_noprofile` attribute and `__ghs_noprofile_func()` intrinsic allow you to mark functions so that any code paths they appear in are not considered when MULTI builds reports from coverage data gathered with **-coverage=**.

If you are using MULTI 6.1.4, you must enable the **-Xprofileignorefatalblocks** switch to use this feature. With this version of MULTI, the following caveats apply:

- The **Coverage** tab will always display `0` for **% Covered**, but the **Block Detailed** tab will continue to display count information for the block.
- Coverage line coloring in the Debugger is reversed for functions with the `ghs_noprofile` attribute.
- The `ghs_noprofile` attribute is ignored by **TimeMachine** profiling.

Changes in Behavior

GNU Compatibility Updated for Better Compatibility with Version 4.3

The GNU C and C++ dialects have been updated to provide closer compatibility with the 4.3 version of the GNU compiler. To revert to a version of these dialects that is more closely compatible with version 3.3, use **--gnu_version=30300** in conjunction with the **-gcc** or **--g++** language dialect option. Make sure that you have also selected a GNU dialect before specifying this option.

Additionally, dependent name processing is now on by default when using the GNU C++ dialect, even when emulating version 3.3 of the GNU compiler. To disable dependent name processing, use **--no_dep_name**.

Furthermore, the parsing of function template definitions is done lazily in the GNU C++ dialect. Even though the parsing of a template in its generic form (without argument substitution) is typically done to resolve non-dependent names, this parse is only done for templates that are actually instantiated, and the parse itself is deferred until the entire input file has been processed. This may result in the apparent suppression of errors in unused function templates. This may be disabled with the **--no_defer_parse_function_templates** option.

For more information, see:

- "GNU C/C++ Extensions" in Chapter 12, "Green Hills C" in *MULTI: Building Applications for Embedded V850 and RH850*
- "Dependent Name Lookup" in Chapter 13, "Green Hills C++" in *MULTI: Building Applications for Embedded V850 and RH850*
- "Deferral of Function Template Parsing" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*

Chapter 4

Changes in Version 2013.1

Contents

Product Compatibility

Green Hills Compiler 2013.1 is officially supported with INTEGRITY versions 5.x, 10.0.2, and 11.0.4. For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Green Hills Compiler 2013.1 is officially supported with ThreadX 5.5.x and newer.

Warning

Do not install Green Hills Debug Probe software version 4.2 or earlier over your compiler installation.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Relaxed Size Restrictions

The compiler no longer restricts arrays and structures to 256 megabytes in size. It now allows uninitialized arrays and structures declared as up to 2 gigabytes on 32-bit targets, and over 2 gigabytes on 64-bit targets.

There may be some issues with debugging information for large arrays and structures, such as warnings issued from **dblink**, problems viewing these objects in the MULTI Debugger, or problems when these objects are involved in command line procedure calls. In addition, large programs may not load into the simulator if they consume too much RAM on the host.

protrans Supports Converting Section Dumps to .pro Files

If you use one of the **-coverage=** options to collect profiling information from your program and dump that program into a file on your host computer, you can now use **protrans** to convert that file into a **.pro** file using the **-cc**, **-cd**, or **-cf** option. For more information, see “The protrans Utility” in Chapter 11, “Utility Programs” in *MULTI: Building Applications for Embedded V850 and RH850*.

Changes in Behavior

Some C and C++ Libraries Have New Names Based on Floating Point Support

Some C and C++ libraries have been renamed based on the level of floating point support they provide. For more information, see "Libraries" in Chapter 16, "Libraries and Header Files" in *MULTI: Building Applications for Embedded V850 and RH850*.

Removed and Deprecated Features

The following features have been deprecated:

- Call count and call graph profiling (**-p** and **-pg**).
- Run-mode debug over a freeze-mode debug connection with hardware targets running INTEGRITY 5. Run-mode connections are still supported over Ethernet and serial ports.
- Socket emulation over freeze-mode host I/O with hardware targets running INTEGRITY 5. Socket emulation is still supported over run-mode debug, or you can use the GHnet v2 TCP/IP stack on the target.

Chapter 5

Changes in Version 2012.5

Contents

In the Green Hills Compiler 2012.5 release, we focused on improving the **gstack** utility program, improving profiling data collection, and increasing C++ performance.

Product Compatibility

Green Hills Compiler 2012.5 is officially supported with INTEGRITY versions 5.x, 10.0.2, and 11.0.4. For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Green Hills Compiler 2012.5 is officially supported with ThreadX 5.5.x and newer.

Warning

Do not install Green Hills Debug Probe software version 4.2 or earlier over your compiler installation.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Updated gstack Utility Program

The **gstack** stack analysis program has been greatly improved with the ability to analyze recursive functions, function pointers, and interrupts. Its output has also been improved to provide detailed information to guide you in annotating your program for the most accurate results. For more information, see "The gstack Utility Program" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850*.

New Block Profiling System

The **-coverage=** option provides a faster method for measuring both performance and code coverage than the legacy **-a** method. If you only want to test for coverage and not execution counts, **-coverage=flag** provides a significantly more efficient

method than **-a**. For more information, see "Enabling Instrumented Coverage or Performance Profiling" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850*.

New ax Modifier

The **ax** librarian now includes the **u** modifier for compatibility with the GNU archiver.

New Linker Options to Control Linker p_align Behavior

The new `-max_p_align`, `-any_p_align`, and `-no_p_align` options allow you to specify how the linker determines the maximum value of the `p_align` field in ELF program headers. For more information, see "Linker-Specific Options" in Chapter 9, "The elxr Linker" in *MULTI: Building Applications for Embedded V850 and RH850*.

New Stack Check Attribute

The new C/C++ `stackcheck` attribute allows you to enable stack checking on a per-function basis. For more information, see "`stackcheck`" in "Attributes" in Chapter 12, "Green Hills C" in *MULTI: Building Applications for Embedded V850 and RH850*.

Changes in Behavior

Clean Builds Recommended

To mitigate any problems that may occur due to small changes in behavior from previous versions, we recommend that you clean and rebuild your programs from scratch when you build them for the first time with the upgraded compiler.

Error Compiling ind_initmem.c

When porting a customized **libstartup** from an earlier version to Green Hills Compiler 2012.5, one of the following messages may be given when compiling **ind_initmem.c**:

```
#error This file should only be included from another file in the same directory
#error Please include <stddef.h> instead of ghs_null.h
```

To fix this problem, edit **ind_initmem.c** to replace:

```
#include <ghs_null.h>
```

with:

```
#include <stddef.h>
```

For more information, see "C and C++ Header Files" in Chapter 16, "Libraries and Header Files" in *MULTI: Building Applications for Embedded V850 and RH850*.

Vector's Clear Method No Longer Deallocates

Prior to Green Hills Compiler 2012.5, vector's `clear()` method deallocated memory, resetting the capacity to zero, even if the `reserve()` method had been called. In Green Hills Compiler 2012.5, `clear()` no longer deallocates memory.

To deallocate a vector's memory, call the `swap()` method instead. This technique is portable across versions of the Green Hills Compiler and other STL implementations:

```
std::vector<T> v;
...
std::vector<T>().swap(v);

std::vector<T> v;
...
std::vector<T>(v).swap(v);
```

This change to `clear()` can be reverted by defining the `__GHS_VECTOR_CLEAR_DEALLOCATES` macro to 1. This macro might be removed in a future version of the compiler.

Implicit Inclusion is No Longer Enabled By Default

Implicit inclusion is no longer enabled by default. If you require this feature, pass the **--implicit_include** option to the driver.

Eclipse Plug-in Option Type Changes

In the Green Hills Eclipse plug-in, the types of a few options have changed. If you have existing Green Hills Eclipse Managed Make projects that you created using an older version of the Green Hills Eclipse plug-in and you encounter options that do not work well (for example, they display erroneous or garbled text values in the C/C++ Build Settings dialog) we recommend creating a new project using the 2012.5 version of the plug-in and re-importing your sources into the new project, rather than reusing the old project with the new plug-in.

ghide No Longer Hides COMMON Symbols

The **ghide** utility no longer hides `COMMON` symbols. For more information about **ghide**, see "The ghide Utility Program" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850*.

Removed and Deprecated Features

The following features have been deprecated:

- The **Legacy Coverage Profiling** (**-a**) driver option.

Chapter 6

Changes in Version 2012.1

Contents

In the Green Hills Compiler 2012.1 release, we focused on improving build performance, enhancing existing optimizations, and making architectural changes that allow you to upgrade the MULTI IDE and Compiler separately. This version of the compiler also adds GHS Standard Mode and coding standard profiles, two new features that help you improve code quality.

Product Compatibility

Green Hills Compiler 2012.1 is officially supported with INTEGRITY versions 5.x, 10.0.2, and 11.0.4. For information about compatibility with newer versions of INTEGRITY, see the INTEGRITY release notes.

Green Hills Compiler 2012.1 is officially supported with ThreadX 5.5.x and newer.

Warning

Do not install Green Hills Debug Probe software version 4.2 or earlier over your compiler installation.

Note

If your MULTI IDE is newer than the compiler, see the IDE release notes for up-to-date product compatibility.

New Features and Enhancements

Separate Releases Allow Greater Flexibility When Upgrading

The Compiler and IDE products now release independently of each other, allowing you to upgrade them separately. This makes it possible to freeze on one version of the Compiler while upgrading the version of the IDE, or vice versa. This release structure also allows Green Hills Software to release compilers with new CPU support on a faster schedule.

Faster Builds

Build speeds have increased significantly in this release of the MULTI Compiler. Build speed on Windows hosts has increased, and parallel build speed has improved

up to 50%. In addition, **gbuild** now builds in parallel by default, so users that did not explicitly pass **-parallel=** in previous versions of MULTI may see much larger improvements. In one real-world case, a customer reported a 4.5x increase in build speed.

Define Macros in Any Project File

You can now define macros in any project file, not just your Top Project.

New Coding Standard Features

The Green Hills Compiler now ships with GHS Standard Mode, an internally-designed collection of diagnostics that help you avoid common pitfalls while developing programs. We use this standard internally because its rules are pragmatic to apply to existing projects and helpful for improving code quality.

The compiler also provides coding standard profiles, a feature that allows you to create your own coding standards by packaging diagnostic severity levels into a single file to distribute among all developers in your organization. For more information, see Chapter 14, "Coding Standards" in *MULTI: Building Applications for Embedded V850 and RH850*.

UTF-8 Support

The Green Hills Compiler now supports multi-byte characters encoded in UTF-8. For more information, see "Host and Target Character Encoding" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Changes in Behavior

New Licenses Required

The MULTI licensing system was updated in this release. If you are upgrading from an older release, you must obtain new licenses. For more information, please contact Green Hills support.

Clean Builds Recommended

To mitigate any problems that may occur due to small changes in behavior from previous versions, we recommend that you clean and rebuild your programs from scratch when you build them for the first time with the upgraded compiler.

gbuild Defaults to Parallel Builds

gbuild now builds programs in parallel by default, and automatically determines the number of builds to run at once based on the number of CPUs on your host machine. You do not need to specify **parallel=*n*** to **gbuild** unless you want to limit the number of CPUs used for building programs.

Different malloc() Implementation Used By Default in Stand-Alone Projects

The compiler uses a new `malloc()` implementation by default for all Stand-Alone projects. This implementation generally provides better performance, but has a slightly larger library footprint. To use the old implementation instead, pass the **-no_fast_malloc** option.

__MULTI_DIR__ Customization File Macro Changes

If you use the predefined `__MULTI_DIR__` macro in your customization files, replace each instance with `__TOOLS_DIR__`, which points to your MULTI Compiler installation directory. There is no longer a macro defined for the MULTI IDE installation directory.

Pointers Are Now Unsigned by Default

Pointers are now unsigned by default for all architectures, and shipping libraries are built with **--unsigned_pointer**. If you require signed pointers, pass the **--signed_pointer** option, but keep in mind that operations that rely on memory spanning `0` may not work consistently between user code and the libraries.

memmove Moved to libind.a

The `memmove()` function has been moved from **libansi.a** to **libind.a**. If your build fails because `memmove()` cannot be located, include **libind.a**.

Removed and Deprecated Features

The following features are no longer available:

- Windows 2000 support
- Elan licenses
- The **gbldconvert** utility that converts **.bld** files to **.gpj** files. If you are migrating from MULTI 4 or earlier and need to convert your **.bld** files, contact Green Hills Support for more information.

The following features have been deprecated:

- **gproxy**
- **grun**
- P&E Lightning card support in **peserv**
- The `truncate()` function in **libsys.a**.
- **--cxx_include_directory** (use the **-I** driver option instead).

Chapter 7

Changes in Version 5.4

Contents

New Features and Enhancements

UTF-8 Support

The compiler now supports UTF-8 encoding for extended characters, in addition to EUC-JP and Shift-JS. For more information, see “Extended Character Support” in the *MULTI: Building Applications* book.

Updated MULTI Eclipse Plug-In

The MULTI Eclipse plug-in now supports Eclipse 3.7.x and CDT 8.0.x. For more information, see the documentation about the MULTI Eclipse plug-in in the *MULTI: Building Applications* book.

Changes in Behavior

Removed Features

Driver No Longer Accepts .bld Files

The driver and Project Manager no longer build projects in the legacy **.bld** format, and the **gbldconvert** utility that converts files from **.bld** to the newer **.gpj** format has been deprecated. Support for converting **.bld** files will be removed in the next release.

Chapter 8

Changes in Version 5.3

Contents

Product Compatibility

The MULTI 5.3 compiler has been tested for compatibility and is officially supported with:

For information about the compatibility of the MULTI 5.3 compiler with older versions of these products, contact your Green Hills sales representative. For information about compatibility with newer versions of these products, see the release notes for the appropriate product.

New Features and Enhancements

New -Omoredebug Optimization Strategy

The new **-Omoredebug** optimization strategy has been added in this release and is enabled by default for new projects. This strategy enables only optimizations that do not affect debugging ability. Additionally, the compiler intentionally generates code so that:

- Variable lifetimes are extended to let you view variable values at any point in the function, and
- Every statement has a breakdot.

Note that in some cases, **-Omoredebug** may cause your program's stack size to increase.

Updated Compiler Front-End

MULTI Compiler 5.3 is based on a newer version of the EDG C++ front-end. The new front-end includes many bug fixes, improvements in standard compliance, and improvements in GNU compatibility. The default severity levels of some diagnostics may have changed.

Updated Fast Flash Programmer

The fast flash programmer includes several enhancements:

- faster SREC handling
- delta flashing capabilities (only re-flash changed sectors)
- improved logging
- support for new flash parts

For more information about which flash parts are supported, contact your Green Hills sales representative.

General Improvements

The following features have seen major improvements:

- Wholeprogram optimizations
- Interprocedural optimizations

Changes in Behavior

New Warnings Enabled by Default

The **--prototype_warnings** option is now set by default. This option may generate warnings that indicate bugs in your code.

Changes to Line Continuations in GNU C Mode

In versions of the MULTI compiler earlier than 5.2, a backslash (\) followed by a space was never parsed as a line continuation. As of 5.2, in the GNU C or C++ dialect, the compiler parses backslash as a line continuation character whether or not it is followed by a space.

If you compile your code using the GNU C or C++ dialect and it has a line that contains a backslash followed by a whitespace, it might create an unintentional line continuation. The following example shows one of these cases (the space is indicated by an underscore, because it does not display in print):

```
#include <stdio.h>
int main(void) {
    int i = 42; // use space after backslash, as in C:\_
```

```
    i = 0; // ignored in GNU
    if (i != 0) {
        printf("i = %d, should be 0\n", i);
    }
}
```

When the compiler encounters one of these cases, it issues diagnostic 1400. If you need to hide this diagnostic, pass the option **--diag_suppress=1400** to the driver.

Changes in std::string

Due to changes in `std::string`, the C++ libraries included with this release are incompatible with the C++ libraries shipped with previous versions of MULTI.

Options Enabled by Default

The following options are now enabled by default in MULTI 5:

- **--alternative_tokens**
- **--no_implicit_extern_c_type_conversion**
- **--no_guiding_decls**
- **--readonly_typeinfo**
- **--readonly_virtual_tables**

The following options are enabled by default unless you pass **--guiding_decls**:

- **--no_old_specializations**
- **--no_implicit_typename**

Map File Format Changes

The format of map files generated by the **elxr** linker has changed in version 5.3. The map file now lists the size of each symbol in addition to its address (in the format *`address+size`*).

To force the linker to produce map files in the old format, pass the following option to the driver:

`-lnk=-old_mapfile`

This option is not supported in conjunction with the **-Mu** or **-Ml** linker options.

Loops with an Iteration Variable Shorter than int May Execute Slower than in Previous Releases

If you have enabled speed optimizations and your program has a loop that uses an iteration variable of a type smaller than `int`, that loop may execute slower than it did with previous versions of MULTI. To work around this issue, use an iteration variable of type `int` or larger, or `int_fast`*N*`_t` or `uint_fast`*N*`_t`, defined in **stdint.h**.

__alt_init Hook Removed From Stand-Alone Run-time Library

The call to `__alt_init` from the stand-alone run-time routine `__ghs_ind_crt1` has been removed as of the 5.2 release. We recommend renaming `__alt_init` to `__ghs_board_devices_init`, which is called at approximately the same point in the startup sequence. A second approach is to add the following C code to your project:

```
#if defined(__GHS_VERSION_NUMBER) && (__GHS_VERSION_NUMBER >= 520)
void __alt_init(void);
void __ghs_board_devices_init(void) {
    __alt_init();
}
#endif
```

If neither of these approaches work for you, you can customize **libsys.a** and modify `__ghs_ind_crt1` in **ind_crt1.c** to call `__alt_init` prior to calling `main()`. For more information, see "Creating a Project with a Customizable Run-Time Environment" in "Libraries and Header Files" in the *MULTI: Building Applications* book.

Removed Features

The following features are no longer included in MULTI, but may still be available for purchase separately. For more information, contact your Green Hills sales representative.

- Elan licensing tools
- HP-UX host support
- The following driver options:
  - **--array_new_and_delete**
  - **--common_implicit_initialization/--no_common_implicit_initialization**
  - **--distinct_template_signatures**
  - **--early_tiebreaker/--late_tiebreaker**
  - **--enum_overloading/--no_enum_overloading**
  - **--explicit**
  - **--extern_inline/--no_extern_inline**
  - **--friend_injection/--no_friend_injection**
  - **--include_once/--include_never**
  - **--initpipointers**
  - **--memory_description_file**
  - **--new_for_init/-old_for_init**
  - **-Owholeprogram_libs**
  - **--relative_xof_path/--no_relative_xof_path**
  - **--short_lifetime_temps/--long_lifetime_temps**
  - **--std_qualifier_deduction**
  - **--typename**
  - **-use_vp_as_dbg_source_root/-no_use_vp_as_dbg_source_root**

Deprecated Features and Discontinued Support

Deprecated K&R C Support

Support for the K&R C dialect (**-k+r**) is deprecated and will be removed in the next MULTI release. Certain old-style features, such as K&R-style parameter declarations, will continue to be supported outside of the K&R C dialect.

Deprecated -gnu Option

The **-gnu** driver option is deprecated. To get the same behavior as **-gnu**, pass both the **-gcc** and **--g++** options instead.

Chapter 9

Changes in Version 5.0.6

Contents

New Features and Enhancements

Improved Code Generation

MULTI 5 comes complete with Green Hills Software's world class compilers, including new interprocedural optimization and link-time optimization. MULTI 5 boasts compilers which generate some of the industry's smallest and fastest code. Additionally, you can use MULTI 5 to enforce programming guidelines such as the MISRA 2004 coding standards.

Improved Dependency Analysis

Building programs with complicated dependencies (such as an INTEGRITY kernel) has historically required that the user identify libraries that might have changed since the last build in order to avoid link-time errors. MULTI's Improved Dependency Analysis solves this problem by automatically building out of date libraries if the current program is dependent on those libraries. This capability extends to prebuilt libraries or other libraries linked in with an option such as **-l** or **-kernel**.

DoubleCheck Static Source Code Analyzer

The DoubleCheck static source code analyzer reviews code, searching for problems which may lead to bugs in the final product. DoubleCheck is fully integrated with the MULTI IDE. It executes automatically when you build your application. Any problems discovered by DoubleCheck are reported to you alongside syntax errors reported by the compiler. This creates an opportunity for you to immediately fix bugs, saving many hours of debugging time later in your development process.

Fast Flash Programmer

The MULTI IDE's Fast Flash Programmer solves several problems which occur in firmware development. It significantly cuts down on the time and effort required to repair corrupted boot code, fix bugs in firmware and program production boards. The Fast Flash Programmer uses little or no RAM and runs with greater speed than its predecessor. Additionally, the Fast Flash programmer provides effective feedback

when hardware causes a programming failure. Operation of the Fast Flash Programmer can be scripted using the MULTI command interface for greater flexibility and power.

Changes in Behavior

Changes to Driver Option Defaults

Builder Option	MULTI 4.2.4 Default	MULTI 5.0.6 Default
Alternative C++ Tokens	Off (**--no_alternative_tokens**)	On (**--alternative_tokens**)
Consistent Code Without Debug Information	Option does not exist	Off (**-noconsistentcode**)
Constant Propagation	Off (**-Onoconstprop**)	Depends on optimization strategy
DoubleCheck Level	Option does not exist	None (**-double_check.level=none**)
DoubleCheck Output that Stops Builds	Option does not exist	Off (**-double_check.stop_build=off**)
Guiding Declarations of Template Functions	On (**--guiding_decls**)	Off (**--no_guiding_decls**)
Link-Once Template Instantiation	Option does not exist	Off (**--no_link_once_templates**)
-Olimit= Without Debug Information	Option does not exist	Off (**-noglimits**)
Simplify C Print Functions	Option does not exist	On (**-Oprintfuncs**)
Stack Limit Checking	Option does not exist	Off (**-no_stack_check**)
Section Overlap Checking	Warnings (**-overlap_warn**)	Errors on Non-Zero Overlap (**-nooverlap**)
Support for Implicit Extern C Type Conversion	On (**–implicit_extern_c_type_conversion**)	Off (**–no_implicit_Extern_c_type_conversion**)
Support for Old-Style Specializations	On (**--old_specializations**)	Off (**--no_old_specializations**)
Support for Implicit Typenames	On (**--implicit_typename**)	Off (**--no_implicit_typename**)
Treat Doubles as Singles	Option does not exist	Off (**-nofloatsingle**)

Options Enabled by Standard C++ Dialects

When you select either of the Green Hills Standard C++ dialects (`--std` or `--STD`), MULTI enables various options in order to conform more closely to the C++ standard as defined in International Standard ISO/IEC 14882:1998. MULTI 5 enables a slightly different set of options from MULTI 4:

Options enabled by `--std` and `--STD`:

- `--alternative_tokens`
- `--no_implicit_extern_c_type_conversion`
- `--no_old_specializations`
- `--no_implicit_typename`
- `--no_guiding_decls`

Additional options enabled by `--STD`:

- `--dep_name`

Libind.a Splitup

Some of the contents of the libind.a library from previous releases have been split out into a new library. The libmath.a library is new in MULTI 5, and contains math routines.

Building System Libraries from 4.x Distributions

The MULTI 4 System Libraries and headers make use of **stdio.h** field names that are not preceded by an underscore. In MULTI 5, the names are all defined with a leading underscore.

If you are using MULTI 5 to build a project with system libraries (libsys.a) from a MULTI 4 distribution, you must pass **-D__GHS_OLD_STDIO_FIELD_NAMES** so that MULTI 5's **stdio.h** defines the old names as well.

Global Const Variables Placed in Read-Only Data

In prior versions of MULTI, global `const` variables without an initializer, such as:

```
const int i;
```

were output as common variables (see "Allocation of Uninitialized Global Variables" in Chapter 3, "Builder and Driver Options" in the *MULTI: Building Applications* book). This could cause inconsistencies when using special data areas or when referencing the variable with position-independent code (PIC) or position-independent data (PID).

In MULTI 5, these variables are output as if they have been initialized to zero. The example above would be output in the same way as:

```
const int i = 0;
```

As a result, these variables are typically placed in read-only data.

Unbundled Features

Unbundled features are no longer included in MULTI, but are still available for purchase separately.

Unbundled in 5.0.6

The following features have been unbundled from MULTI:

- FORTRAN compiler
- CodeBalance
- Distributed Build System
- Remote serial connections
- TraceEdge and In-Memory TimeMachine
- Pre-MULTI-4 debug server protocol compatibility
- EST ICE integration
- PMON ROM monitor integration
- IDT ROM monitor integration

- Green Hills ROM monitor integration
- HP-UX host support

Deprecated Features and Discontinued Support

Deprecated Options

The following options have been deprecated in this release:

Builder Option	Driver Options
ANSI C and Standard C++ Extensions	**--discretionary_errors**, **--discretionary_warnings**, **-pedantic-errors**, **-pedantic**
Auto-Instantiation of Templates	**--auto_instantiation**, **-template=auto**, **--no_auto_instantiation**, **-template=noauto**
.c Files are C++	**-nodotciscxx**, **-dotciscxx**
Ignore #include Directives	**--no_include_never**, **--include_never**, **--include_once**, **--no_include_once**
Link Map Method	**-default_lnk**, **-no_default_lnk**, **-default_i**, **-no_default_i**
Minimum Structure Alignment	**--struct_min_alignment**
New-Style 'for-init' Code	**--no_for_init_diff_warning**, **--for_init_diff_warning**
PCH Messages	**--pch_messages**, **--no_pch_messages**, **--no_pch_silent**, **--pch_silent**
Precompiled Header File	**--no_pch**, **--use_pch**, **--pch**, **--create_pch**
Signedness of Enum Type	**--unsigned_enum_fields**, **--signed_enum_fields**
Unroll Loops Up to 8 Times	**-Ounroll8**, **-Onounroll8**
Use ThreadX Demo Library	**-no_use_demo_library**, **-use_demo_library**
Use Before 'Set'	**--no_use_before_set_warnings**, **--use_before_set_warnings**
Use Debug Library List (.dli) Files	**-no_use_dli_files**, **-use_dli_files**

The ghexfile Utility is No Longer Available

The **ghexfile** utility is no longer available. If you used **ghexfile** in a previous version of MULTI, reconfigure your projects to use **gsrec** instead. **gsrec** includes functionality that was formerly included in the **ghexfile** utility. For more information, see "The gsrec Utility Program" in "Utility Programs" in the *MULTI: Building Applications* book.

Manual Template Instantiation Mode Options No Longer Supported

The following **Manual Template Instantiation Mode** options are no longer supported:

- **-tall**
- **-tlocal**
- **-tused**
- **-tnone**

Programming flash from grun

Programming flash memory from grun is no longer supported in MULTI 5. Similar functionality can be obtained by scripting the flash burn command in the debugger.

Chapter 10

Known Issues

Contents

Installation and Licensing

Green Hills Debug Probe CD Compatibility

Some previous revisions of the Green Hills Debug Probes software (shipped with the Green Hills Probe, SuperTrace Probe, and Slingshot) are not compatible with MULTI 5. If your Green Hills Debug Probe CD is not marked "Compatible with MULTI 5," do not install it with MULTI 5. Contact Green Hills customer support for a replacement CD.

Name of Install Directory Cannot Contain Spaces

The MULTI Compiler cannot be run from a directory whose name contains spaces.

Host Support

UNC Paths Are Not Supported in the MULTI Debugger

On Windows hosts with MULTI 6.1.4 or older, creating new projects or debugging projects over a UNC path is not supported. In general, we recommend mapping UNC paths to network drives before trying to access them in the MULTI IDE.

UNC Paths to Virtual Machines Are Not Supported

When remotely accessing a MULTI or Green Hills Compiler installation on a Windows virtual machine using a UNC path, you may get errors that the application could not be found or that it could not obtain a license. To work around this issue, map the path to a network drive and use the drive letter instead of the UNC path.

Target Support

Some u-velOSity Kernel Libraries Do Not Build with Default Options

Some kernel libraries for u-velOSity 2.2.9 do not build with the default option set. If you cannot build the u-velOSity kernel libraries:

- In the following files, remove `70` from the list of rules specified with `--saferc`. If your target is a V850, V850E, or V850F, remove `70` and `3`.
  - **src/kernel/libuv.gpj**
  - **src/kernel/libuv_debug.gpj**
  - **src/kernel/libuv_min.gpj**

Toolchain Issues

Building Previous Releases of INTEGRITY or u-velOSity May Produce New Diagnostic Messages

When building u-velOSity 2.2.9 or earlier, INTEGRITY 5, or INTEGRITY 10, the compiler may output new errors or warnings. To avoid compatibility problems, set the **-act_like** option in your Top Project to `5.0`.

If you encounter other warnings, you may need to further adjust the compiler diagnostics options.

No Function Prototypes For INTEGRITY 5 strdup() and strnlen()

There are no function prototypes available for `strdup()` and `strnlen()` on INTEGRITY 5.

gstack Provides Warnings for Green Hills Library Functions

Some functions in the Green Hills libraries are not annotated for **gstack**. If you link your program with these libraries, you may receive warnings about these Green Hills library functions when using **gstack**.

gstack Provides Warnings for INTEGRITY

INTEGRITY has not been modified to take advantage of new **gstack** features. If you use **gstack** with an INTEGRITY program, you may receive warnings about INTEGRITY functions.

Exported Templates With Inline Functions May Produce Incorrect Code

If your code contains the following:

- A C++ `extern inline` function that contains a class definition that contains a static function definition that contains a static variable
- C++ export templates

the compiler may produce incorrect code.

ThreadX Build Failure When Customizing Startup and System Libraries

If using ThreadX and your project includes customizable **Startup** and **System** library source code, you may get build errors like "[elxr] (error) symbol ___ghsLock multiply defined in: ...libsys.a(ind_lock.o) ...tx.a(tx_ghs.o)". This combination is not supported.

Non-flat Directory Structure in Multi-core Builds Causes Warnings in gcores

If a full path is provided in the output filename of a Core or a SharedMemory, the directory component of the output filename will be truncated and a warning will be issued. Those truncated items will be generated in the project's root directory and will not be cleaned properly by the command **gbuild -clean**.

For example, given the following **.gpj** corresponding to a SharedMemory subproject file:

```
#!gbuild
[SharedMemory]
        -o shared/shared_imp.so
shared.c
```

The output of that build will be placed in **shared_imp.so** instead of in **shared/shared_imp.so** and the following warning will be issued:

```
gcores: warning: The path shared/shared_imp.so has
been flattened to shared_imp.so.
```

Output Filename in Multi-Core Builds with extension different to .mca Causes Warnings in gcores

If the output filename (specified with the option **-o**) of a MultiCoreArchive has no extension, or the extension is different to **.mca**, the extension will automatically be set to **.mca**. That changed output will not be cleaned properly by the command **gbuild -clean**.

For example, if the option **-o demo.mcb** is provided in a MultiCoreArchive project file, the output filename will be changed to **demo.mca** and the following warning will be issued:

```
gcores: warning: The output filename demo.mcb has
been changed to demo.mca
```

Use of -a, -p, or -pg in INTEGRITY KernelSpace Requires -Onomemfuncs

If **-a**, **-p**, or **-pg** are used with INTEGRITY KernelSpace code, **-Onomemfuncs** should be used at the same time.

Building u-velOSity with Compiler 2014.1 or Later

If you are building C++ applications for u-velOSity 2.2.9 or earlier with Compiler 2014.1 or later, they may not link when using default compile options. You may see the following link-time error (note that filenames may vary):

```
[elxr] (error) Possible C++ virtual function table format
incompatibility between modules libuv_debug.a(ind_exit.o) and foo.o
```

To resolve this problem, take one of the following actions:

- If you have the full version of u-velOSity, rebuild your u-velOSity libraries with Compiler 2014.1 or later. For information about rebuilding these libraries, see the *u-velOSity User's Guide*.
- If you are using an evaluation version of u-velOSity, make sure to use the **--no_large_vtbl_offsets** option when compiling all of your application's files.

Target Connection Issues

Flash Driver Library libflash.a Limited to Four CFI Drivers

The flash driver library distributed with the MULTI IDE offers support for Common Flash Interface (CFI)-compatible flash devices. Drivers for these devices are configured at run time. Previous versions of **libflash.a** supported only targets with a single CFI flash device. MULTI 5 now includes support for up to four CFI flash devices. Calls to `FLASH_loadDriver()` on a fifth CFI flash device will return an error code. Multiple calls to `FLASH_loadDriver()` on a single flash base address will return the same CFI driver, even if the hardware is changed or remapped in memory. These limitations do not affect non-CFI drivers. You can force a non-CFI driver by setting the `disableCFI` flag in `FLASH_loadDriver_2()`.

Debugging Issues

Auto Trace Collection Reduces Performance

Automatically enabled trace collection reduces the performance of simulated targets by a significant amount. This is especially noticeable with INTEGRITY simulators on Solaris.

To work around this issue, disable automatic trace collection. For instructions, see the documentation about enabling and disabling trace collection in the *MULTI: Debugging* book.

Target Agent May Cause Failures When Programming Flash

The MULTI **Fast Flash Programmer** dialog contains an input field for the RAM base address. During a program operation, the flash programmer downloads target agents to this address. If the memory used by the target agent conflicts with reserved memory sections, programming may fail. Error messages printed by the flash programmer may indicate that either a hardware exception was hit or the target agent stalled. Check the link map of your executable or the memory map of the board to confirm that the RAM base address entered in the flash dialog is located after the end of any reserved sections.

References Not Available for Inlined Functions

Issuing the **browseref** or **xref** command or right-clicking a symbol (a function, type, member, variable, etc.) and selecting **Browse References** allows you to view all references of a symbol. However, references are not stored, and consequently not displayed, for inlined functions.

Workaround: Disable inlining by compiling your program with **--no_inlining**. Note that functions defined in header files may be implicitly inlined by the compiler if this option is not used.

Cannot Debug Executables in Eclipse Projects with a Space in the Name

You cannot debug executables in Eclipse projects with a space in the name. Do not put a space in the name of an Eclipse project.

File System Issues

Two Dots (..) Not Supported After Symbolic Links in Paths

MULTI and other Green Hills tools do not support paths that use two dots (`..`) to indicate the parent directory of a symbolic link. For example, if `symlink` is a symbolic link, the following paths are not supported:

```
/projects/symlink/..
/projects/symlink/foo/../../bar
```

NFS May Cause Poor Performance

This release note applies to Linux/Solaris hosts only.

MULTI tools may hang if they are accessing files over an NFS server that is slow or is not responding. In certain cases, it may not be possible to terminate them, even when using `kill -9`. These problems are caused by a fundamental limitation in the design and implementation of NFS.

Workaround: To help prevent these problems, eliminate references to broken file system mount points in your user configuration directory (**~/.ghs/***) — in particular in the **integrity.dist** and/or **uvelosity.dist** files located there.

Miscellaneous

Links to Connecting Book

The MULTI documentation set contains links to the *MULTI: Configuring Connections* book. These links are disabled for MULTI and compiler versions designated for native targets.

Chapter 11

V850 and RH850 Release Notes

Contents

V850 and RH850 Toolchain

Changes and Known Issues in 2015.1

Coprocessor Register Usage in the Assembler

Coprocessor registers, such as FPU system registers, are now rejected by the assembler unless the appropriate coprocessor is enabled when building.

Interrupt routines and nofloat and -no_callt

Interrupt routines written in C using `#pragma ghs nofloat interrupt` will now avoid using the FPU registers in this release. Interrupt routines written in C compiled with the **-no_callt** option will now avoid using the CALLT-related registers in this release if they are also compiled with the **-ignore_callt_state_in_interrupts** option.

RTESERV2 Support is Deprecated

Support for the **rteserv2** debug server is now deprecated. While the debug server is still provided with this release, it is provided on an "as is" basis. Customers wishing to make inquiries about future support needs for **rteserv2** should contact their local Green Hills sales office or contact Green Hills support.

MCTL.MA Support in Startup Libraries and Simulator

The default RH850 startup code included with the libraries (**crt0.800**) will now initialize the MCTL.MA bit to one (1) if the program is linked with **-misalign_pack**. Furthermore, the simulator will now simulate this bit to determine if the simulated RH850 processor should support a misaligned memory access or not. Code built with previous versions of the tools and expecting misaligned memory access support may require changes to enable MCTL.MA to properly simulate in this release.

64-bit Misaligned Memory Accesses

Previous releases did not account for the fact that 64-bit load and store instructions on RH850 devices do not support misaligned memory accesses for alignments below 32-bits, even when MCTL.MA is enabled. This release alters the behavior of the **-misalign_pack** option in the compiler to now properly account for this limitation of the RH850 hardware.

FE-level Interrupt Support in Compiler

The compiler now supports creation of FE-level interrupt routines. For more information, please see “Interrupt Routines” in Chapter 2, “Developing for V850 and RH850” in *MULTI: Building Applications for Embedded V850 and RH850*.

Use of CALLT Instruction for Function Prologue

The compiler will now make use of the `CALLT` instruction when support for `CALLT` is enabled (See “Epilogues and Prologues via callt” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*) and the optimization strategy is set to optimize for size. As the library routines called use the `PREPARE` and `DISPOSE` instructions, support for these must also be enabled (See “Epilogues and Prologues via prepare and dispose” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*). Furthermore, inline prologue must be disabled (See “Function Prologues” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*).

Placing read-only Small Data in the Zero Data Area

The compiler now supports the **-rosda_is_zda** option to put data that would normally be allocated to the `.rosdata` section into `.rozdata`. (See “Place all read-only SDA items in ZDA instead” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850*).

RH850G3MH and RH850G3KH support

Support for the RH850G3MH and RH850G3KH devices are now included in this release. This includes two new CPU options (**-cpu=rh850g3mh** and **-cpu=rh850g3kh** respectively) for the command line drivers **ccrh850** and **cxrh850**, the assembler **ease850** and the simulator **simrh850** (See "V850 and RH850 Processor Variants" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850*).

-EVA_load_nops is Removed

The **-EVA_load_nops** option to control a software workaround for a hardware bug on the uPD703191Eva ICE core is no longer supported.

Connecting to Hardware via a Remote Server

The **850eserv2** debug server can only directly connect to hardware on Windows. In this release it is now possible to connect indirectly from a non-Windows host, such as Linux, by using **850eserv2_server** on Windows and **850eserv2** on the other host. For more information, see "Connecting Through a Remote Server (850eserv2_server)" in Chapter 3, "Renesas V850/V850E ICE (850eserv2) Connections" in *MULTI: Configuring Connections for V850 and RH850 Targets*.

New Project Wizard support for iev850e is Removed

The New Project Wizard no longer supports the iev850e board (`-bsp=iev850e`). Compiler and debugger support for the v850e processor is unchanged.

Trace can restart when using Stop Trace triggers

Stop Trace triggers on hardware are implemented using section trace, which requires both a starting address and ending address. The current PC is used for the starting address when setting a Stop Trace trigger. Trace will stop as expected when the ending address is hit, but if the starting address is hit again after that point (if it is in a loop, for example) then trace will be reenabled.

-asm_far_jumps is Removed

The **-asm_far_jumps** option to the assembler is no longer supported.

Passing structures to functions

Structures that are passed to functions by value may be in registers (if available) or on the stack. A structure with 64-bit alignment will be passed at a 64-bit aligned address. When passed in registers, such a structure will start in an even register, which may result in a register being skipped. Since how the structure is passed is dependent on the alignment of the type, it can change with the use of structure packing options and pragma directives.

Support for Additional SDA Registers

The toolchain now supports using up to five additional SDA registers. This is controlled via the **-additional_sda_reg**=*n* and **-additional_sda_common** options. See "Specifying Additional Registers for Special Data Areas" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Support for RAM initialization at Startup

The toolchain now supports initializing RAM at startup. This is controlled via the **-init_ram_at_startup** and **-ram_init_val**=*value* options. See "Run-Time RAM Initialization and RAM Initialization Table" in Chapter 9, "The elxr Linker" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Changes and Known Issues in 2014.1

Memory Clearing Optimization Control

An optimization that changes `memset()` filled with zeros to optimized calls to `memclr()` can now be controlled using the **-Omemclr** and **-Onomemclr** options. For more information, see "Convert C Memset to Memclr" in Chapter 3, "Builder

and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

Check Incompatible RH850 ABI flags

-check_rh850_abi_flags is now turned ON by default for checking incompatible RH850 ABI flags in the linker. This is a change of behavior from 2013.5 where the feature is turned off by default. To turn off the check, pass **-no_check_rh850_abi_flags**.

-cpu=v850e3v5 now implies RH850G3M

Selecting V850E3V5 as the target CPU will now assume the target is an RH850G3M device. This is a change of behavior from 2013.5 where the target was assumed to be an RH850G3K device.

Software Trace Load Address Logging Support

The RH850 software trace support has been extended to support automatically logging the addresses of loads, excluding those known to be from the stack. Please refer to "Logging Load Addresses" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850* for more details.

New Project Wizard Support for RH850/E1x-fcc1

The New Project Wizard now has support for the RH850/E1x-fcc1 board. The corresponding command line option is `-bsp=rh850_e1x_fcc1`. This board has two cores, and the linker directives file can be used for either core, as well as for memory shared by the two cores. The memory layout of the directives file is controlled by the preprocessor macros `__ghs_GC_core_1__`, `__ghs_GC_core_2__`, and `__ghs_GC_shared__`, which specify a layout for core 1, core 2, and shared memory, respectively. If none of the preprocessor macros are defined, the memory layout maximizes the memory available for the first core.

Both cores of this board must be running in order for either to advance, even if code is only built and downloaded for a single core. In order to facilitate debugging in this case, a new option is supported in 850eserv2. The options `-force_coreid=0`

or `-force_coreid=1` can be used to make the board act like a single-core board (using just the first or second core, respectively).

IECUBE2 Support for RH850/E1x-fcc1

The 850eserv2 debug server in this release can be used to connect to an IECUBE2 with an RH850/E1x-fcc1 emulation pod. The RH850/E1x-fcc1 New Project Wizard entry (`-bsp=rh850_e1x_fcc1`) can be used with this configuration. The connection options for 850eserv2 must be edited to use IECUBE as the debug type. The corresponding command line will be similar to the following:

```
connect 850eserv2 -rh850 -iecube \
-df=c:/ghs/comp_201413/rh850/dr7f701z07.dvf \
-dclock=20000,0,swoff -id ffffffffffffffffffffffffffffffff \
-fastflashload
```

New Project Wizard Support for RH850/P1x

The New Project Wizard now has support for the RH850/P1x board. The corresponding command line option is `-bsp=rh850_p1x`. There is a known hardware problem with this board that can make it sometimes fail to connect with 850eserv2. For best results, the board should be powered with an external power supply, as opposed to through the E1 probe. When the board fails to connect, MULTI displays a message similar to the following:

```
RSU info write error 0xca5: fatal err (could not connect to target)
Please check target connection and main clock specified with -dclock.
```

This connection failure is intermittent, and may succeed if tried again.

Library change for __mcopy

The `__mcopy` support routine has been moved from `libarch.a` to `libstartup.a`. Under some circumstances, this may result in a linker error similar to the following when building a project that was created with previous versions of the tools and that rebuild `libstartup.a` from source:

```
[elxr] (error) unresolved symbols: 1
 ___mcopy       from libfmalloc.a(ccmalloc.o)
```

To fix this issue, replace the copy of `ind_mcpy.800` in the project with an updated one from the new release.

Data trace triggers on RH850

Qualifying trace based on data read/write events is now supported on RH850. However, unlike on other boards, all PC trace data will be generated when qualifying trace on data events. Only the data trace will be qualified based on the trigger. To achieve a result equivalent to that on other boards, disable `'Trace PC'` and enable `'Trace Data Access PC'` in the trace options.

Renesas RH850/E1x-fcc1 Data Trace

The data trace packets that are generated for the `PREPARE` instruction by the RH850/E1x-fcc1 hardware are sometimes not correct. When multiple registers are stored to the stack with this instruction, the trace information does not always include packets for all of the stores, and may have the wrong value for some registers. This incorrect trace data can interfere with register reconstruction in TimeMachine.

-no_prepare_dispose and -callt

When **-no_prepare_dispose** and **-callt** are both enabled (the default for V850E), the compiler now will not use the `CALLT` instruction function prologues and epilogues that use the `PREPARE`/`DISPOSE` instruction for space-saving optimization. This may impact the code size for **-Osize** when **-no_prepare_dispose** and **-callt** are enabled.

Changes and Known Issues in 2013.5

gsrec now supports multi-core archives

gsrec now supports operating on multi-core archives created by the **gcores** utility to produce HEX and S-Record files. Please see "GCores Generate Additional Output" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

AUTOSAR and OSEK Operating System Awareness

Provides operating systems awareness (OSA) for ORTI-enabled operating systems such as many AUTOSAR and OSEK operating systems. See its documentation in "AUTOSAR and OSEK Operating System Awareness" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Interrupt routines and nofloat and -no_callt

Interrupt routines written in C using `#pragma ghs nofloat interrupt` or **-no_callt** build option should be built with the **-inline_prologue** build option. Otherwise library routines may be used to save and restore registers, and these routines will save and restore system registers relating to the FPU and to the call table unconditionally.

Software Trace Logging Support

Support for the software trace functionality of the RH850 family of processors has been added in this release. Please refer to "Software Trace Logging Support" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850* for more details.

Preventing the use of double-precision floating point

Two new options have been added to give users greater control over restricting the usage of double-precision floating point in their programs. Please refer to "Warn for types double or long double" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* and "Do not allow types double or long double" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more details.

gcores -target option deprecated

For users of the **gcores** utility, the **-target** option has been deprecated. Using it will result in an error or warning. Users should instead use the new **-driver** option. For more information about this new option, please refer to"GCores Driver" in Chapter 3,

"Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850*.

RH850G3x loss of target control with some trace options

For RH850 hardware with 850eserv2, if the "Real-Time Trace Mode" option is disabled and either the "When Trace Buffer Fills Since Last Halt" option is set to "Stop Trace" or a Delay Trace trigger is set and "When Delay Trigger is Hit" is set to "Stop Trace at Delay Trigger", control of the target may be lost. This is due to a limitation of the current hardware. If the problem comes up, disconnect from 850eserv2, power cycle the target, and reconnect to 850eserv2.

Name change from RH850/F1A to RH850/F1L

The New Project Wizard now refers to the RH850/F1L board, not the RH850/F1A. The corresponding command line option is `-bsp=rh850_f1l`.

Changes and Known Issues in 2013.1

RH850G3x support

Support for the RH850G3K, RH850G3M and RH850G3H devices is now included in this release. The previous RH850 setting available in 2012.5 (**-cpu=rh850**) is now treated as an RH850G3K (same as **-cpu=rh850g3k**). When connecting to the simulator or other debug server targeting the RH850G3H or RH850G3K, the debugger may still report them as the RH850G3M. This does not affect support for these devices, and will change in a future release to reflect the correct CPU. When using the **ccrh850** driver, if no CPU is specified on the command line, then the default setting will be RH850G3K. Please note that whenever the CPU is specified as RH850G3K, software floating-point will be enabled by default (**-fsoft**). To enable hardware floating-point instead, please specify **-fhard** or **-fsingle** as appropriate for your floating-point unit. Please refer to "Floating-Point Mode" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more details on these FPU settings.

RH850G3x trace support

The **850eserv2** debug server included in this release supports the collection of trace from trace-enabled hardware using the Renesas E1 emulator. For multi-core configurations, the debug server currently has the limitation that any trace configuration setting made will affect all cores. Per-core trace setting support is planned for a future release. Multi-core trace support is also included in this release, however, it is provided in preliminary form only, and should be considered alpha release quality.

RH850G3x loss of target control with some trace options

For RH850 hardware with 850eserv2, if the "Real-Time Trace Mode" option is disabled and either the "When Trace Buffer Fills Since Last Halt" option is set to "Stop Trace" or a Delay Trace trigger is set and "When Delay Trigger is Hit" is set to "Stop Trace at Delay Trigger", control of the target may be lost, and attempting to halt would give errors such as "user system err (cannot break)". This is due to a limitation of the current version of the Renesas EXEC library shipped with this release. If the problem comes up, disconnect from 850eserv2, power cycle the target, and reconnect to 850eserv2. This will be resolved in a future release of the Renesas EXEC library. Users experiencing this problem should contact support@ghs.com to see if an updated Renesas EXEC library patch is available.

PUSHSP/POPSP support

The options **-push_pop** and **-no_push_pop** have been added to control the compiler's use of the `PUSHSP` and `POPSP` instructions. See “Use pushsp and popsp instructions” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

PREPARE/DISPOSE support

The option **-prepare_dispose** is now on by default whenever the option **-no_callt** is used. This represents a change from previous default behavior. See “Epilogues and Prologues via prepare and dispose” in Chapter 3, “Builder and Driver Options” in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

SIMD support in interrupt routines

The compiler in this release does not generate code to save or restore the SIMD coprocessor state for interrupt routines. Interrupt routines should either not modify the SIMD coprocessor state, or should use assembly code to manually save its state beforehand.

FPU version 3 support

The version 3 FPU supported on most RH850 devices is now supported and enabled by default when compiling for RH850 devices. Users who are using an early-generation RH850 device may need to enable the legacy version 2 FPU support. Consult your device manual for more information. For information about how to enable the legacy FPU support, please see "Use FPU 3.0 instructions (RH850 and later)" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Coprocessor initialization

The compiler library startup code for RH850 processors will now initialize the PSW.CU0 and PSW.CU1 register fields to enable these coprocessors on implementations which support them.

Legacy simulator for V850 and RH850 (sim850)

The legacy **sim850** simulator has been discontinued. It has been replaced with the new multi-core simulator for V850 and RH850, **simrh850**. A **sim850** executable is still included in this release for compatibility with project connections from previous releases. However, it simply invokes **simrh850** with the appropriate options. Users should transition existing projects from previous releases to use the new simulator. The new simulator, in addition to full multi-core and shared memory support, is also significantly faster at simulation than the legacy simulator. For more information about using the new simulator, please refer to Chapter 5, "Multi-core Simulator for V850 and RH850 (simrh850) Connections" in *MULTI: Configuring Connections for V850 and RH850 Targets*.

simrh850 coprocessor simulation

The **simrh850** simulator now simulates the coprocessor fields of the PSW register available on RH850 processors, specifically the PSW.CU0 and PSW.CU1 fields. By default, the simulator will rely on information provided by the compiler in the ELF file to set determine whether or not it will simulate the appropriate coprocessor as enabled or disabled. If the coprocessors are disabled and an associated coprocessor instruction is encountered, the simulator will simulate a coprocessor exception and print and error. Users can forcefully disable the coprocessors in the simulator using the **-fpu=none** and **-no_rh850_simd** options. Please refer to Chapter 5, "Multi-core Simulator for V850 and RH850 (simrh850) Connections" in *MULTI: Configuring Connections for V850 and RH850 Targets* for more details.

simrh850 hyper-threading simulation

The **simrh850** simulator now reflects recent changes to the RH850 specification and resets the HTSCCTL.RND register field to zero. In previous releases it was reset to `0x3F`. Users of previous releases should take care to ensure they initialize this register appropriately.

Extended GNU asm support in V850 and RH850

The option **--gnu_asm** is supported and enabled by default. See Chapter 19, "GNU Extended Assembly" in *MULTI: Building Applications for Embedded V850 and RH850* for more information.

Renesas RH850 E1x connections

The New Project Wizard does not support the multi-core E1x board. To connect to the board, invoke 850eserv2 with options similar to the following:

```
connect 850eserv2 -rh850 -e1lpd4=11000 \
-df=c:/ghs/comp_201315/rh850/dr7f701z006ebg.dvf \
-dclock=4000,0,swoff -id ffffffffffffffffffffffffffffffff \
-fastflashload
```

Renesas RH850 Data Trace

Trace packets for loads and stores can sometimes be dropped or issued out-of-order by the hardware, resulting in data trace packets that cannot be correlated with the appropriate instructions in the trace list.

Stop Execution on Trace Buffer Full for Renesas RH850 E1x

When the Buffer Full trace option is set to Stop Execution on the multi-core E1x board, the behavior is unreliable. Sometimes both cores halt with a full trace buffer message at the same time, but other times only a single core will display the full trace buffer message. Retrieving trace in the latter case can lead to problems with decoding the data.

Trace Delay Triggers Across System Calls

When a trace delay trigger is set to disable trace for a certain condition, such as execution of a specific address, trace may be re-enabled upon execution of a system call (for example, during printf). This may lead to the trace list containing extra instructions or overflowing.

Errors when re-enabling trace on multi-core E1x board

On the multi-core E1x board, when trace is manually disabled and re-enabled repeatedly, it is possible to get an error from 850eserv2 like the following:

```
Target: BRS condition write error 0x404: param err (event num overflow)
```

Changes and Known Issues in 2012.5

RH850 Support

Full support for the RH850 processor family is included in this release. This includes new CPU options (**-cpu=rh850**), new command line drivers (**ccrh850** and **cxrh850**) and a new simulator (**simrh850**). The new simulator supports multi-core simulation of the V850 and RH850 processors, as well as simulation of hyper-threading on RH850 processors. See Part I. Using the MULTI Compiler in *MULTI: Building*

Applications for Embedded V850 and RH850 and Chapter 5, "Multi-core Simulator for V850 and RH850 (simrh850) Connections" in *MULTI: Configuring Connections for V850 and RH850 Targets* for more details.

Multi-core Support

Multi-core building and debugging is supported in this release. This includes the new utility program **gcores**, the new driver options: **-exportall** and **-exported_absolutes_only**, the new project files types: MultiCoreArchive, Core and SharedMemory. See "The gcores Utility Program" in Chapter 11, "Utility Programs" in *MULTI: Building Applications for Embedded V850 and RH850* for more information about building for multiple cores. Multi-core debugging is supported in the new MULTI provided in this release, including as well a new configuration file type **.ghsmc**. Please see "Multi-Core Debugging" in Chapter 25, "Freeze-Mode Debugging and OS-Awareness" in *MULTI: Debugging* for more detailed information.

V850 and RH850 Target Connections

Renesas E1 Emulator support in 850eserv2

E1 Emulator support in 850eserv2 is included in this release. See Chapter 3, "Renesas V850/V850E ICE (850eserv2) Connections" in *MULTI: Configuring Connections for V850 and RH850 Targets* for more details.

Renesas V850/V850E ICE connections (850eserv2)

In the Connection Editor, selecting the box labeled "Write 0's into BSS section" has no effect.

When initially connecting to 850eserv2, if you abort the connection attempt, the target hardware may need to be power-cycled before you can connect to it again.

Limitation in Retrieve Trace when Buffer Fills Option

The trace option “Retrieve trace when buffer fills” has a limitation for V850 targets that trace might not be stopped immediately when the buffer fills. This means that some trace from the beginning of the buffer may be missing.

Some unsupported trace trigger conditions shown for IECUBE2

For trace on IECUBE2, the **Set Triggers** window shows some trigger conditions that are not supported by the hardware. Selecting a condition that involves data reads or writes such as **Data Written** will result in an error message like “Trace event write error 0xc60: param err (invalid event condition)”.

Problem When Using Section Sub-Switch Off to Enable Trace

For trace with the Midas ICE, e.g., for the KIT-V850E2/MN4, using the Section Sub-Switch Off trace options to enable PC or data trace does not work properly. Some instructions/data will not appear while Section Sub-Switch is Off.

SuperTrace Probe with Renesas IE Cube Deprecated

Support for using a SuperTrace Probe to collect trace from a Renesas IE Cube is deprecated in this release and may be removed from a future release.

V850 and RH850 Debugging

Spawn Window Script

A new Python script allows you to spawn a Debugger window for a target listed in the current Debugger window. If the new Debugger window is spawned for the currently selected target, the script attempts to select a different target in the target list.

You can specify the script with or without a target name. If specified without a target name, it is applied to the currently selected target.

You can either use the script as a script or you can import and run it. To run it as a script from the Debugger command pane, enter:

```
py -f compiler_installation/python/ghs_examples/spawn/spawn.py
```

or

```
py -f compiler_installation/python/ghs_examples/spawn/spawn.py \
       target_name
```

To import and run it from a script, from the **Py** pane, the **Py Window**, or the **MULTI-Python GUI**, use:

```
sys.path.append("compiler_installation/python/ghs_examples/spawn");
       import spawn;
       spawn.spawn();
```

or

```
spawn.spawn("target_name");
```

Changes and Known Issues in 2012.1

Improvements to V850E CALLT usage

In previous releases of the tools the compiler would, when compiled with -callt (the default), optionally use the CALLT instruction for certain prologue and epilogue space-saving optimizations. The use of this instruction, and the method with which it is implemented, has changed. This creates some important noteworthy differences in this release that customers making use of this feature should be aware of. Please see "Generate a backwards-compatible (MULTI 5.1.x and earlier) call table in the linker" in Chapter 3, "Builder and Driver Options" in *MULTI: Building Applications for Embedded V850 and RH850* and "Function Call Pragma Directives" in Chapter 2, "Developing for V850 and RH850" in *MULTI: Building Applications for Embedded V850 and RH850* for more details.

V850 Global Registers (-globalreg)

The V850 global registers (those declared with the “register” keyword) are always r20-r24; previous release notes indicated additional registers may be used, but that is no longer the case.

V850E1F Signed Zero Treatment

A new driver option has been added to the tools as of MULTI 5 which provides some control over the treatment of floating point zero values. This option is -precise_signed_zero_compare and its corresponding antithesis -no_precise_signed_zero_compare. The behavior of the compiler for the V850E1F architecture in older versions of the compiler does not correspond to the default value of this new option. Users wishing to revert to previous behavior should use -no_precise_signed_zero_compare. Users should consult their manual before enabling this option so that they are fully aware of the consequences of its use.

V850 TDA Support

-notda is now the default compilation mode with V850. The linker will give an error when a user attempts to link a module compiled with -notda together with one that uses the TDA.

Users migrating from older releases who want to use TDA should compile with -single_tda

EP Register Usage

-notda is now the default compilation mode with V850. See the note above on V850 TDA Support.

Additionally, -allocate_ep is also now the default compilation mode with V850. In this mode, EP is allocated to as a temporary register.

Users migrating from older releases who want to disable TDA and limit the use of the EP register should compile with -no_allocate_ep.

Force Save/Restore of R4/R5 for Interrupt Service Routines

It is now possible to force the save and subsequent restore of the small data area (SDA) and read-only small data area (ROSDA) base registers (r4 and r5) during the prologue and epilogue of interrupt service routines. A new command line driver option **-v850_isr_save_r4r5** enables this behavior. This behavior is off by default.

Earlier releases of the V850 tools would save and restore these registers under certain circumstances. As that was a violation of the ABI, and resulted in inefficient code generation, that behavior was removed. However, customers who relied on this behavior can use this new option to retain this previous behavior. Most customers will not need to use this option.

V850E and Later C/C++ Interrupt Routines Compiled with -no_callt

In previous releases, a library compatibility issue existed when compiling a C/C++ interrupt routine with the **-no_callt** driver option, the **-Osize** size optimization, and the default **-no_inline_prologue** driver option enabled. This issue was addressed in MULTI 5, but requires that C/C++ interrupt routines built with these options and targeting V850E and later cores be recompiled to ensure compatibility with libraries going forward.

For reference, C/C++ interrupt routines are those routines declared with the `__interrupt` keyword or the `#pragma ghs interrupt` pragma.

V850E -no_v850e_mul_errata Option

When targeting the V850E processor (**-cpu=v850e**), the compiler does not by default emit instructions of the following forms:

```
mul reg1, reg2, reg3
   mulu reg1, reg2, reg3
   mul imm1, reg2, reg3
   mulu imm1, reg2, reg3
```

where *reg3* is `r0`, or *reg2* is the same as *reg3*. This is to work around an erratum with the V850ES implementation.

The assembler and linker flag any occurrences of the above instructions, but do not attempt to change them.

To disable both the checks in the assembler and linker, and the compiler's avoidance of these instructions, specify the **-no_v850e_mul_errata** driver option.

MULTI 6.1.6 Compatibility

This Compiler release should be used with the version of MULTI provided on this CD , the SH CD or the TriCore CD. If you have MULTI 6.1.6 installed from a previous version of V800 , SH or TriCore Compiler 2014.1, ARM, Power, Coldfire, Solaris native or x86 platforms Compiler 2014.1, you can install a patch to MULTI to make it compatible with this release, using the following steps:

- Completely exit the MULTI IDE before applying the patch.
- Change into the **multi-v6.1.6-patch** directory on the CD.
- Run **install.bat** (on Windows) or **install.sh** (on Linux or Solaris).
- When prompted for an Installation Directory, enter your existing MULTI installation directory.