Release Notes: LLVM FOR RENESAS RL78 17.0.1.202406

28th of June, 2024

CyberThor Studios Ltd. is releasing the LLVM for Renesas RL78 17.0.1.202406, a cross compiler tool for Renesas RL78 micro-controllers.

SALIENT FEATURES

The LLVM for Renesas RL78 17.0.1.202406 toolchain is based on:

- ❖ LLVM 17.0.1 [released]
- ❖ Compiler-rt 17.0.1 [released]
- ❖ Libcxx 17.0.1 [released]
- ❖ Libcxx-abi 17.0.1 [released]
- ❖ Newlib 4.1.0 [released]
- ❖ GDB 12.1 [released]

LLVM RL78 comes with significant performance improvements (both code size and speed) compared to GCC RL78. It also comes with support for the latest language standards: full support for C17 and C++17 and partial support for C++20, C++23, C++2C.

The latest patches are applied to the LLVM sources.

ABOUT LLVM FOR RENESAS RL78 17.0.1.202406

Release Version:	LLVM for Renesas RL78 17.0.1.202406
Release Date:	28th of June, 2024
Platforms Supported:	Ubuntu 20.04 or later (or compatible distribution) Windows 7 or later macOS Ventura 13.4 on Apple M1
Language:	C, C++
Targets:	All RL78 devices
Object File Format:	ELF

This toolchain is the successor of GCC RL78 toolchain and it is meant as a direct replacement to GCC RL78.
This section describes the fixes made in the LLVM for Renesas RL78 17.0.1.202406 release.

1. **[Improvement] Various codegen optimizations**

Code speed and size optimizations were made in the compiler.

2. **[Improvement] Far address space handling in C++**

Far address spaces, for both data and functions, are now supported in C++.
For details and limitations please consult the user manual (-mfar-code/-mfar-rom).

3. **[Change] Default -fno-finite-loops compile option**

In order to avoid replacing trivially infinite loops in C++ when using standards newer than C++03,
the -fno-finite-loops option was added as a default compile option.
If users wish to disable this, the -ffinite-loops option can be passed to the compiler.

4. **[Bug fix] Incorrect call stack information**

When debugging, the call stack information was incorrect.
This was due to a bug in the debug information generation that is fixed in the current release.

5. **[Bug fix] -flto option ignored**

Due to the -flto option resulting in linking errors, previous versions silently ignored it.
Current release fixes the linking errors and re-enables this option.
Please note that the -flto option and .KEEP linker script directive are not compatible.
For details please consult the user manual (Linker script compatibility).

6. **[Bug fix] Initial call of malloc**

A bug where initial malloc calls for certain sizes caused it to always return NULL was fixed.

7. **[Bug fix] __UBIT macro not defined**

When using -funsigned-bitfields, the __UBIT macro was not defined.
This was fixed in the current release.

Notes:

The LLVM for Renesas RL78 toolchain on the Windows platform requires the presence of the x86 Microsoft Visual C++ Redistributable package for Visual Studio 2015-2022.
This can be downloaded and installed from the official site:
https://aka.ms/vs/17/release/vc_redist.x86.exe

This installer does not provide an option to integrate the LLVM RL78 toolchain with e2 studio, as the e2 studio IDE will automatically detect the LLVM RL78 toolchain installation on start-up for integration. Alternatively, you may use the 'Toolchain Management' feature in e2 studio to achieve this.

For details on e2 studio please visit the following link below:
https://www.renesas.com/eu/en/software-tool/e-studio

There is no support in this installer to integrate the toolchain with the HEW IDE.

The following is a list of known issues for the tools we include for the LLVM for Renesas RL78 17.0.1.202406 toolchain:

1. **Assembly parsing issue of callt instruction operand.**

The assembly parser cannot handle any complex expressions as operands for callt instruction, only constant values are allowed at the moment.

2. **Missing assembly listing support (-a[cdhlns] option in GNU AS).**

There's no equivalent in the LLVM for the -a[cdhlns] GNU AS option.
The alternative solution is to use llvm-objdump to obtain source code interleaved with assembly (-S, --source option).

3. **Binding of references to packed fields.**

Creating references to struct members which are declared as packed, e.q. __attribute__ ((packed)), is incorrect as it can cause unaligned access issues. The compiler should return an error in this case however this is not currently the case. This will be fixed in a future release.

4. **Inline assembly missing constraints and modifiers support for parameters.**

Clang supports, in the case of RL78, inline assembly constructs without parameters, e.q. asm (AssembleTemplate).
The extended syntax available in GCC, asm (AssemblerTemplate : OutputOperands : InputOperands : Clobbers : GotoLabels), is not currently supported. This will be fixed in the next release, however the constraints will not be the same as in GCC.

5. **Newlib format specifiers**

The pre-built newlib library included with the toolchain is built without using the -enable-newlib-io-c99-formats configure flag.
Without this flag, handling for some format specifiers will not be included in the resulting library.
Using libgen, users can build their own newlib, specifying the -D_WANT_IO_C99_FORMATS=1 option to include the extra format specifiers, at the cost of code size.

6. **PLT usage for function calls over 64K**

For each near call, where the callee was allocated over the 0xFFFF boundary, the linker will create an entry for the callee (if it doesn't exist yet) in the Procedure Linkage Table(PLT).
The entry will consist of a BR !!<Callee> instruction and the address written to the call instruction will be the address of the PLT entry.

This indirection allows developers to increase the address range they can allocate their code, without the usage of _far functions.

Example:
test.c:

```
  void bar() __attribute__((section(".far_section")));
   void foo() {
      bar();
   }
```

linkerscript.ld:

```
    ...
     .far_section 0x111D8 : AT(0x111D8)
     {
         . = ALIGN(2);
         *(.far_section)
         . = ALIGN(2);
     } >ROM
    ...
```

will result in the following ELF file:

```
    ...
000000d8 .lowtext:
     d8: ec d8 11 01                         br !!_bar
    ...
00003004 _foo:
    3004: fd d8 00                           call !.lowtext
    3007: d7                                 ret
    ...
000111d8 _bar:
```

7. **Incompatibility of -save-temps and -frenesas-extensions options**

Using `-save-temps` and `-frenesas-extensions` together will result in an error.

8. **clang: "-save-temps -g" option leads to "warning: inconsistent use of MD5 checksums"**

See https://github.com/llvm/llvm-project/issues/56378

9. **Other issues, non-specific to RL78**

LLVM tries to be a complete replacement. As such there are still a couple of missing features from GCC which will be implemented in future releases. In particular, the following issues should be noted:

lld: Information printed using --print-gc-sections is not as nice as when using the GNU ld. Bug 46783

llvm-ar: Errors when printing multiple members with the same name. Bug 42521

llvm-dwarfdump: Does not print section attribute flags yet. [Bug 38488](#)

llvm-nm: Unable to understand symbols built with gcc-lto [Bug 41437](#)

llvm-nm: Needs support for --line-numbers to llvm-nm [Bug 40001](#)

llvm-objcopy: Unknown argument '--change-section-address'. [Bug 45217](#)

llvm-objcopy: Objcopy zero-size section might cause huge binaries. [Bug 46299](#)

llvm-objdump: Prints wrong line number info for obj file compiled with -ffunction-sections. [Bug 40703](#)

llvm-objdump: Wrong behavior for non-relocatable objects when suing llvm-objdump with -r option. Bug [Bug 41901](#)

llvm-readobj: Make GNU style symbol printing invalid symbol section indexes match GNU readelf [Bug 43850](#)

llvm-readelf: Relocation addends printed style does not match GNU readelf [Bug 45235](#)

llvm-string: Short option with argument grouping not GNU compatible [Bug 42942](#)

llvm-string: Allow "-<integer>" as an alias for "-n <integer>" [Bug 42964](#)

llvm-symbolizer: Shows incorrect source line info if --gc-sections used [Bug 41124](#)

llvm-symbolizer: llvm-addr2line does not exit when passed a non-existent file [Bug 42754](#)

10. Other issues

For better understanding regarding the status of the toolchain please visit
https://github.com/llvm/llvm-project/issues

FREE SUPPORT FOR LLVM FOR RENESAS RL78 17.0.1.202406

For free technical support, please register at
https://llvm-gcc-renesas.com

For your feedback and suggestions, please visit
https://llvm-gcc-renesas.com/help/contact-us/