Toolchain Projects

This page gives an overview of upstream projects. If you miss information or find mistakes, please edit.

- **ABIs, APIs, and other conventions**
  - Calling conventions
  - Default library path
- **Binutils**
  - RISC-V maintainers
  - Releases
  - RISC-V status
- **GCC**
  - RISC-V maintainers
  - GCC RISC-V community meetings
  - Releases
  - RISC-V status
- **GDB**
  - RISC-V maintainers
  - Releases
  - RISC-V status
- **Glibc**
  - RISC-V maintainers
  - Releases
  - RISC-V status
- **LLVM**
  - RISC-V maintainer
  - LLVM RISC-V meetings
  - Releases
  - RISC-V status
- **Newlib**
  - RISC-V maintainer
  - Releases
  - RISC-V status

ABIs, APIs, and other conventions

The main document for RISC-V ABI/ELF-related information is the RISC-V ELF psABI document, which can be found here: [https://github.com/riscv/riscv-elf-psabi-doc/blob/master/riscv-elf.md](https://github.com/riscv/riscv-elf-psabi-doc/blob/master/riscv-elf.md)

An overview of other RISC-V ABI/API related documents can be found here: [https://github.com/riscv/riscv-elf-psabi-doc/blob/master/README.md](https://github.com/riscv/riscv-elf-psabi-doc/blob/master/README.md)

The following RISC-V ABIs are currently defined:

- **32-bit**
  - ILP32, ILP32F, ILP32D, ILP32E
- **64-bit**
  - LP64, LP64F, LP64D, LP64Q

Calling conventions

There are two calling conventions for GP registers in the RISC-V ecosystem:


Default library path

The rich set of APIs has the consequence, that glibc's default library path includes a subdirectory for the actual ABI (e.g. "/usr/lib64/lp64d" for LP64D).

These default paths are defined here: [https://sourceware.org/git/?p=glibc.git;a=blob;f=sysdeps/unix/sysv/linux/riscv/configure.ac](https://sourceware.org/git/?p=glibc.git;a=blob;f=sysdeps/unix/sysv/linux/riscv/configure.ac)

Binutils

The GNU Binutils are a collection of binary tools (GNU linker, GNU assembler, many other excellent tools such as gprof).

- Binutils Homepage
- Source Release Page

RISC-V maintainers
Releases

**Rule of thumb:** Binutils (GNU linker, GNU assembler, tons of other excellent tools) releases twice per year (mid July and mid January).

- Binutils 2.35 schedule:
  - Binutils 2.35 (2020-07-24)
  - Binutils 2.35.1 (2020-09-19)
  - Binutils 2.35.2 (2021-01-30)
- Binutils 2.36 schedule:
  - Binutils 2.36 (2021-01-24)

RISC-V status

- RV32GC = RV32IMAFDC is implemented
- RV64GC = RV64IMAFDC is implemented
- Unratified extension support is kept in staging branches here: [https://github.com/riscv/riscv-binutils-gdb](https://github.com/riscv/riscv-binutils-gdb)

GCC

The GNU Compiler Collection (GCC) includes front ends for C, C++, Objective-C, Fortran, Ada, Go, and D, as well as libraries for these languages (e.g. libstdc).

- GCC Homepage
- Source Release Page

RISC-V maintainers

- Andrew Waterman (SiFive)
- Palmer Dabbelt (Google)
- Jim Wilson (SiFive)
- Kito Cheng (SiFive)

GCC RISC-V community meetings

- Schedule: biweekly, Thursday 7 am (PST), starting on Mar 11, 2021
- Organized by Wei Wu (PLCT)
- Public announcement: [https://www.mail-archive.com/gcc@gcc.gnu.org/msg94197.html](https://www.mail-archive.com/gcc@gcc.gnu.org/msg94197.html)

Releases

**Rule of thumb:** GCC closes the merge window for the next release in mid-November (once per year).

- GCC 11 schedule
  - GCC 11 Stage 1 (starts 2020-04-30)
  - GCC 11 Stage 3 (starts 2020-11-16)
  - GCC 11 Stage 4 (starts 2021-01-17)
  - GCC 11.1 release (not defined; possibly May/June)

After stage 3 has started, new functionality may not be introduced.

The upstream release schedule can be found [here](https://github.com/riscv/riscv-gcc).

The upstream release timeline can be found [here](https://github.com/riscv/riscv-gcc).

RISC-V status

RV32G and RV64G are mostly implemented. However, there is still some optimization potential.

- RV32GC = RV32IMAFDC is implemented
- RV64GC = RV64IMAFDC is implemented
- RV32E is supported
- Supported calling conventions: ilp32, ilp32f, ilp32d, lp64, lp64f, and lp64d
- GCC supports -msave-restore
- Unratified extension support is kept in staging branches here: [https://github.com/riscv/riscv-gcc](https://github.com/riscv/riscv-gcc)

---

**GDB**

GDB is the GNU Project debugger.

- [GDB Homepage](#)
- [Source Release Page](#)

**RISC-V maintainers**

- Andrew Burgess (Embecosm)
- Palmer Dabbelt (Google)

**Releases**

GDB major releases are approximately annually. There are typically one or two minor releases each year. This is the typical schedule:

- major release branch/pre-release approximately 1 month before release
- first minor release (“re-spin”) approximately 3 months after major release

At the time of writing the most recent release was 10.1, released on 2020-10-24. Dates for branching (and hence release) of GDB 11 have yet to be announced.

**RISC-V status**

Debugging works on top of **PTRACE** syscalls. HW-Breakpoint or HW-Watchpoint support is missing.

---

**Glibc**

Glibc is the GNU C library.

- [Glibc Homepage](#)
- [Source Release Page](#)

**RISC-V maintainers**

- Palmer Dabbelt (Google)
- Andrew Waterman (SiFive)
- DJ Delorie (Red Hat)
- Darius Rad(Bluespec)

**Releases**

**Rule of thumb:** Glibc releases twice per year (February and August).

Previous releases:

- glibc 2.32 (2020-08-05)
- glibc 2.33 (2021-02-01)
- glibc 2.34 (2021-08-01)

**RISC-V status**

- The following ABIs are supported:
  - ILP32, ILP32D, LP64, LP64D

---

**LLVM**

The LLVM Project is a collection of modular and reusable compiler and toolchain technologies.

- [LLVM Homepage](#)
- [Source Release Page](#)
RISC-V maintainers

- Alex Bradbury (lowRISC)

LLVM RISC-V meetings

- Schedule: biweekly, Thursday 8 am (PST), since Sept 2019
- Organized by Alex Bradbury (lowRISC) and Ana Pazos (Qualcomm)

Releases

- LLVM 12 schedule
  - LLVM 12.0.0 RC1 (branching) (2021-01-26)
  - LLVM 12.0.0 RC2 (2021-02-23)
  - LLVM 12.0.0 final (2021-03-02)

The upstream release page can be found [here](https://lists.llvm.org/pipermail/llvm-dev/2021-February/148345.html).

RISC-V status

- RV32GC = RV32IMAFDC is implemented
- RV64GC = RV64IMAFDC is implemented
- RV32E is not supported
- Some non-ratified extensions have been merged mainline (they need the flag -menable-experimental-extensions to enable them):
  - Bitmanip v0.93
  - b, zba, zbb, zbc, zbe, zbf, zbm, zbp, zbr, zbs, zbt, zbpredicate
  - Vector v0.10
  - z, zvamo, zvssseg
  - Example command for building: `clang --target=riscv64-unknown-elf -march=rv64gcv0p10 -menable-experimental-extensions`
  - Floating-point v0.1
  - zfh
- LLVM supports -msave-restore
- RISC-V specific command-line options: [https://clang.llvm.org/docs/ClangCommandLineReference.html#riscv](https://clang.llvm.org/docs/Clang CommandLine Reference.html#riscv)

Newlib

Newlib is a C standard library implementation intended for use on embedded systems.

- [Newlib Homepage](https://newlib.net)
- [Source Release Page](https://newlib.net/Releases)

RISC-V maintainer

- Kito Cheng (SiFive)

Releases

**Rule of thumb:** Newlib releases once per year.

Last releases:

- Newlib 3.3.0 (2020-01-22)
- Newlib 4.0.0 (2020-11-17)
- Newlib 4.1.0 (2020-12-18)

RISC-V status

RV32 and RV64 are supported. Still, there is optimization and completeness potential.