GNU Toolchain

2019-09-27

1 About

The GNU toolchain is a collection of programming tools. A version for RISC-V is available at `https://github.com/riscv/riscv-gnu-toolchain`.

The objective of this document is to present the relevant tools for compiling and debugging software, and to offer quick guides and examples about using them.

2 Platforms

2.1 Target Triplet

A target triplet is used to identify systems. It allows build system to know if the code will run.

Some examples are :

- x86_64-linux-gnu
- x86_64-unknown-freebsd
- i686-pc-linux-gnu
- arm-linux-gnueabihf
- riscv32-unknown-elf

Target triplets have the following structure: **machine - vendor - operating system**

The **vendor** field may be left out. In that case, it is assumed to be the default (unknown). As such, x86_64-unknown-freebsd and x86_64-freebsd are the same.

The **operating system** field can be composed of two words. For example, x86_64-linux-gnu is actually:

- machine: x86_64
- vendor: unknown (left out)
- os: linux-gnu

2.2 Build, Host, Target

The **build platform** is the one on which the compilation tools are executed.

The **host platform** is the one on which the code will eventually run.

The **target platform** of a compiler is the one the compiler generates code for.

The GNU tools state the platforms they are configured for. The target triplet can be included in the filename of the binaries:

- aarch64-linux-gnu-gcc
- riscv32-unknown-elf-gdb
- riscv64-unknown-elf-objdump

You may also find it with the -v option:

```
$ gcc -v
[...]
Target: x86_64-linux-gnu
Configured with: [...] --build=x86_64-linux-gnu --host=x86_64-linux-gnu --
    target=x86_64-linux-gnu
[...]
```

Or the -dumpmachine option:

```
$ gcc -dumpmachine
x86_64-linux-gnu
```

2.3 RISC-V extensions

The RISC-V instruction set architecture can use extensions. There is the core ISA, standard extensions, and custom extensions.

Each extension brings its own instructions. Consequently, the toolchain must be configured with the correct architecture, or you might encounter errors. This is especially true with custom non-standard extensions.

You can find the target architecture:

```
$ riscv32-unknown-elf-gcc -v
[...]
Target: riscv32-unknown-elf
Configured with: [...] --with-arch=rv32i [...]
[...]
```

3 GCC

option list : https://gcc.gnu.org/onlinedocs/gcc/Option-Summary.html

for debugging : -Og -g

linker

4 GDB

gdb binary target remote host:port load

breakpoints continue layout src, asm, (???) list step

5 objdump

-S -D

6 objcopy