RISC-V: практические примеры

Оглавление

Введение

В этом разделе вы получите практический опыт программирования на языке ассемблера RISC-V для операционной системы Linux. Даже если вы никогда раньше не программировали на ассемблере, в этой главе вы получите всю необходимую информацию, чтобы разработать свою первую программу "Hello World".

В этой главе вы узнаете:

- как эмулировать простую систему Linux с помощью QEMU;
- как разработать простую программу "Hello World" на 64-битном языке ассемблера RISC-V;
- как компилировать и запускать приложение RISC-V в эмуляторе.

Обзор языка ассемблера RISC-V

Необходимая документация

В главе 2 спецификации RISC-V Unprivileged Specification подробно рассматривается набор команд RV32I Base Integer Instruction Set, включая программистскую модельи объяснение форматов команд. Хотя эта информация не является обязательной для данного курса, она полезна для понимания того, как в RISC-V выполняются команды.

Для программирования на языке ассемблера можно использовать справочную документацию ABI и руководство ASM, чтобы найти ответы на вопросы, которые могут возникнуть в процессе работы::

- спецификации RISC-V[1];
- документация ABI[1];
- руководство по ASM[2].

Ни один из этих документов не является обязательным для изучения в данной главе, но к ним можно обратиться, если возникнут вопросы, на которые здесь нет ответов.

Обзор языка ассемблера

Эта глава будет представлять собой поверхностный обзор инструкций ассемблера RISC-V и лишь некоторые из них будут рассмотрены на практике. Мы надеемся, что данный материал даст вам инструментарий , необходимый для продолжения вашего дальнейшего изучения программирования на языке ассемблера. Если Ваша цель понять основы и далее разрабатывать приложения на языке более высокого уровня, этот курс, скорее всего, даст вам большую часть необходимой информации.

RISC-V – это архитектура с сокращенным набором команд , и поэтому в ней не так много команд ассемблера, которые нужно изучать. В этом курсе мы используем только 3 инструкции: LA (загрузка абсолютного адреса), ADDI (сложение с константой) и ECALL. Команда ECALL используется для выполнения служебного запроса к среде выполнения. В разрабатываемом приложении “Hello

[1] Specifications: [Электронный ресурс] // RISC-V. URL: https://riscv.org/technical/specifications/.

[2] RISC-V Assembly Programmer’s Manual: [Электронный ресурс] // GitHub. URL: https://github.com/riscv-non-isa/riscv-asmmanual/blob/master/riscv-asm.md.

World” будут использоваться только два вызова, один для записи и один для выхода из цикла программы.

Полный список инструкций содержится в спецификации RISC-V Unprivileged Specification в главе 24 "RV32/64G Instruction Set Listings". Если вы хотите узнать больше о программировании на языке ассемблера, существует множество книг и курсов. Многие из них перечислены на веб-сайте RISC-V [1].

Начало работы с QEMU

Компиляция двоичных файлов

Инструкции по компиляции двоичных файлов находятся в документе RISC-V – Руководство по началу работы [2].

Создание пользовательской системы RISC-V

Если вы уже имеете опыт компиляции ядра Linux, QEMU и таких программных пакетов, как BusyBox, возможно, вы захотите сделать еще один шаг вперед. Существует система сборки для создания корневых файловых систем на базе Linux и их эмуляции под названием Yocto Project. В RISC-V есть “слой”, который можно использовать для создания полностью собственного дистрибутива Linux. Для получения более подробной информации обратитесь к репозиторию meta-riscv [3] на GitHub.

RISC-V Hello World

Обзор окружения

Ниже приведен код приложение “Hello World”:

```
-----code-----
# Simple RISC-V Hello World

.global _start

_start: addi  a0, x0, 1
la    a1, helloworld
addi  a2, x0, 13     addi
a7, x0, 64     ecall
     addi    a0, x0, 0
addi    a7, x0, 93
ecall  .data
helloworld:    .ascii "Hello World!\n"  -----end
code-----
```

Существует два способа компиляции этого кода: либо с помощью GCC, либо вызывая команды AS & LD напрямую из консоли:

```
-----code-----
# GCC
riscv64-linux-gnu-gcc -o rv-hello rv-hello.s -nostdlib -static
# AS & LD
riscv64-linux-gnu-as -march=rv64imac -o rv-hello.o rv-hello.s riscv64-linux-gnu-ld
-o rv-hello rv-hello.o
-----end code-----
```

Список источников

1. RISC-V Learn: [Электронный ресурс] // RISC-V. URL: https://riscv.org/learn/.
2. Running 64- and 32-bit RISC-V Linux on QEMU: [Электронный ресурс] // RISC-V Getting Started Guide. URL: https://risc-v-getting-started-guide.readthedocs.io/en/latest/linux-qemu.html.
3. meta-riscv: [Электронный ресурс] // GitHub. URL: https://github.com/riscv/meta-riscv.