

循序渐进，学习开发一个 RISC-V 上的操作系统



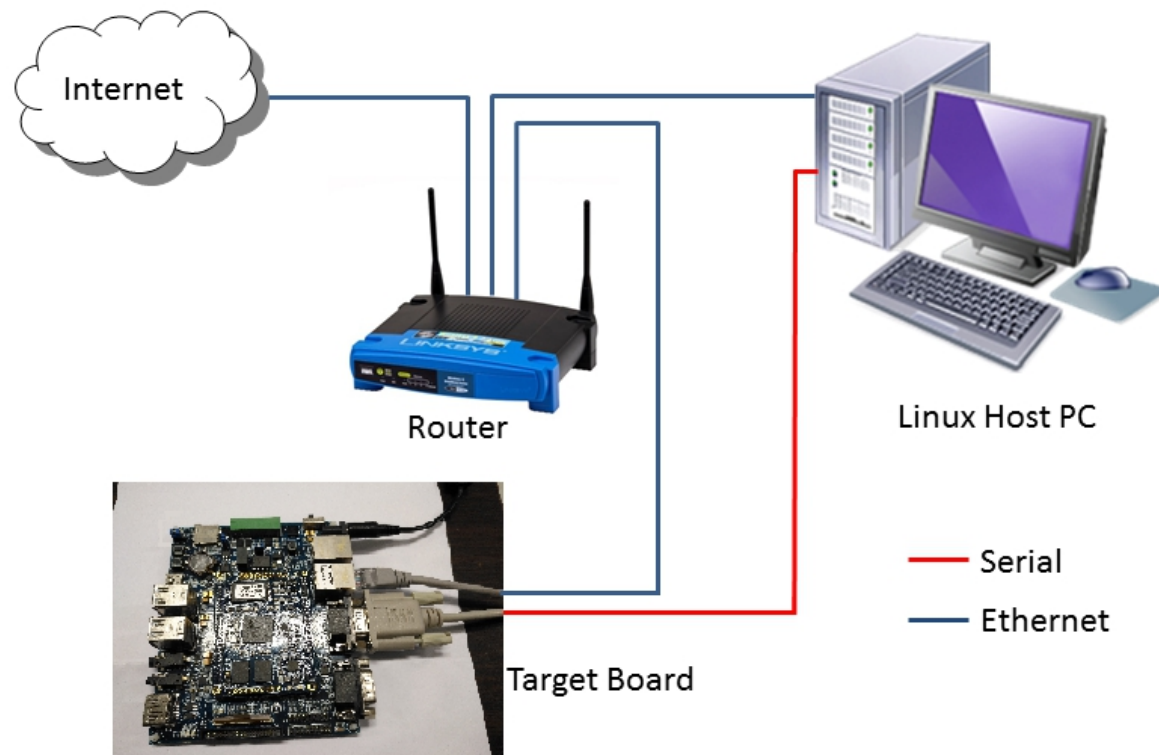
第 4 章 嵌入式开发介绍

汪辰

- 什么是嵌入式开发
- 交叉编译
- 调试器 GDB
- 模拟器 QEMU
- 项目构造工具 Make

- 什么是嵌入式开发
- 交叉编译
- 调试器 GDB
- 模拟器 QEMU
- 项目构造工具 Make

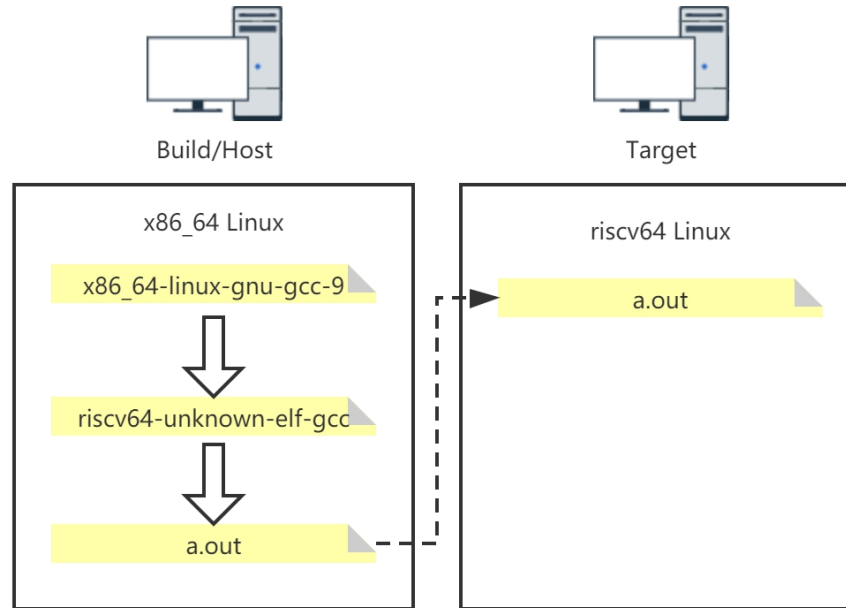
嵌入开发是一种比较综合性的技术，它不单指纯粹的软件开发技术，也不单是一种硬件配置技术；它是在特定的硬件环境下针对某款硬件进行开发，是一种系统级别的与硬件结合比较紧密的软件开发技术。



- 什么是嵌入式开发
- 交叉编译
- 调试器 GDB
- 模拟器 QEMU
- 项目构造工具 Make

- 参与编译和运行的机器根据其角色可以分成以下三类：
 - 构建 (build) 系统：生成编译器可执行程序的计算
机。
 - 主机 (host) 系统：运行编译器可执行程序，编
译链接应用程序的计算机系统。
 - 目标 (target) 系统：运行应用程序的计算机系
统。
- 根据 build/host/target 的不同组合我们可以得到
如下的编译方式分类：
 - 本地 (native) 编译：build == host == target
 - 交叉 (cross) 编译：build == host != target

➤ 交叉 (cross) 编译: $\text{build} == \text{host} \neq \text{target}$



➤ GNU 交叉编译工具链 (Toolchain)

- 命名格式: $\text{arch-vendor-os1-[os2-]XXX}$
- 例子:
 - x86_64-linux-gnu-gcc
 - riscv64-unknown-elf-gcc
 - riscv64-unknown-elf-objdump

- 什么是嵌入式开发
- 交叉编译
- **调试器 GDB**
- 模拟器 QEMU
- 项目构造工具 Make



➤ <https://www.gnu.org/software/gdb/>

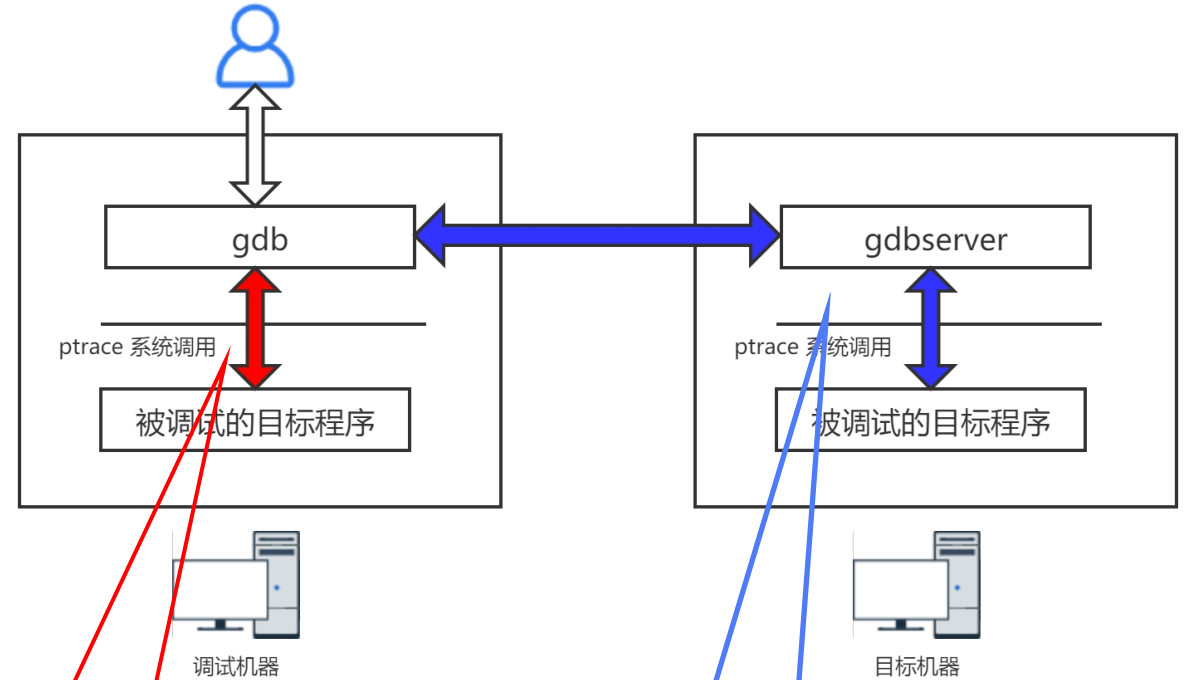
➤ GDB (GDB: The GNU Project Debugger), GNU 项目调试器, 用于查看另一个程序在执行过程中正在执行的操作, 或该程序崩溃时正在执行的操作。

➤ 被调试的程序可能与 GDB 在同一台计算机上执行, 也可能在另一台计算机 (远程) 上或者在模拟器上执行。

➤ GDB 支持调试多种语言: 譬如: Assembly, C, Go, Rust,

本地调试

远程调试



- 重新编译程序并在编译选项中加入 “-g”
\$ gcc -g test.c
- 运行 gdb 和程序
\$ gdb a.out
- 设置断点
(gdb) b 6
- 运行程序
(gdb) r
- 程序暂停在断点处，执行查看
(gdb) p xxx
- 继续、单步或者恢复程序运行
(gdb) s/n/c

- 什么是嵌入式开发
- 交叉编译
- 调试器 GDB
- 模拟器 QEMU
- 项目构造工具 Make

➤ <https://www.qemu.org/>



- QEMU 是一套由 (Fabrice Bellard) 编写的以 GPL 许可证分发源码的计算机系统模拟软件，在 GNU/Linux 平台上使用广泛。
- 支持多种体系架构。譬如：IA-32 (x86), AMD 64, MIPS 32/64, RISC-V 32/64 等等。
- QEMU 有两种主要运作模式：
 - User mode: 直接运行应用程序。
 - System mode。模拟整个计算机系统，包括中央处理器及其他周边设备。

➤ 安装

- Ubuntu 上 apt install
- 源码编译安装

➤ `qemu-system-riscv32 ... -kernel ./test.elf`

➤ `qemu-system-riscv32 ... -kernel ./test.elf -s -S`

- `-s`: “`-gdb tcp::1234`” 的缩写，启动 gdbserver 并在 1234 端口号上监听客户端
- `-S`: 在启动时停止CPU (只有到在客户端键入'c' 才会开始执行)

- 什么是嵌入式开发
- 交叉编译
- 调试器 GDB
- 模拟器 QEMU
- 项目构造工具 Make



➤ <https://www.gnu.org/software/make/>

➤ **make 是什么**

- make是一种自动化工程管理工作。

➤ **Makefile 是什么**

- 配合 make，用于描述构建工程过程中所管理的对象以及如何构造工程的过程。

➤ **make 如何找到 Makefile**

- 隐式查找：当前目录下按顺序找寻文件名为“GNUmakefile”、“makefile”、“Makefile”的文件
- 显式查找：-f

- **Makefile 由一条或者多条规则 (rule) 组成**
- **每条规则由三要素构成**
 - **target:** 目标，可以是 obj 文件也可以是可执行文件
 - **prerequisites:** 生成 target 所需要的依赖
 - **command:** 为了生成 target 需要执行的命令，可以有多条
- **一个简单的 Makefile 规则如下:**

必须是一个
TAB

```
target...:prerequisites...  
    command...  
    ...
```

图1

```
hello: hello.c  
    gcc hello.c -o hello
```

图2

➤ Makefile 中的其他元素介绍

- 缺省规则

```
.DEFAULT_GOAL := all  
all :
```

- 伪规则

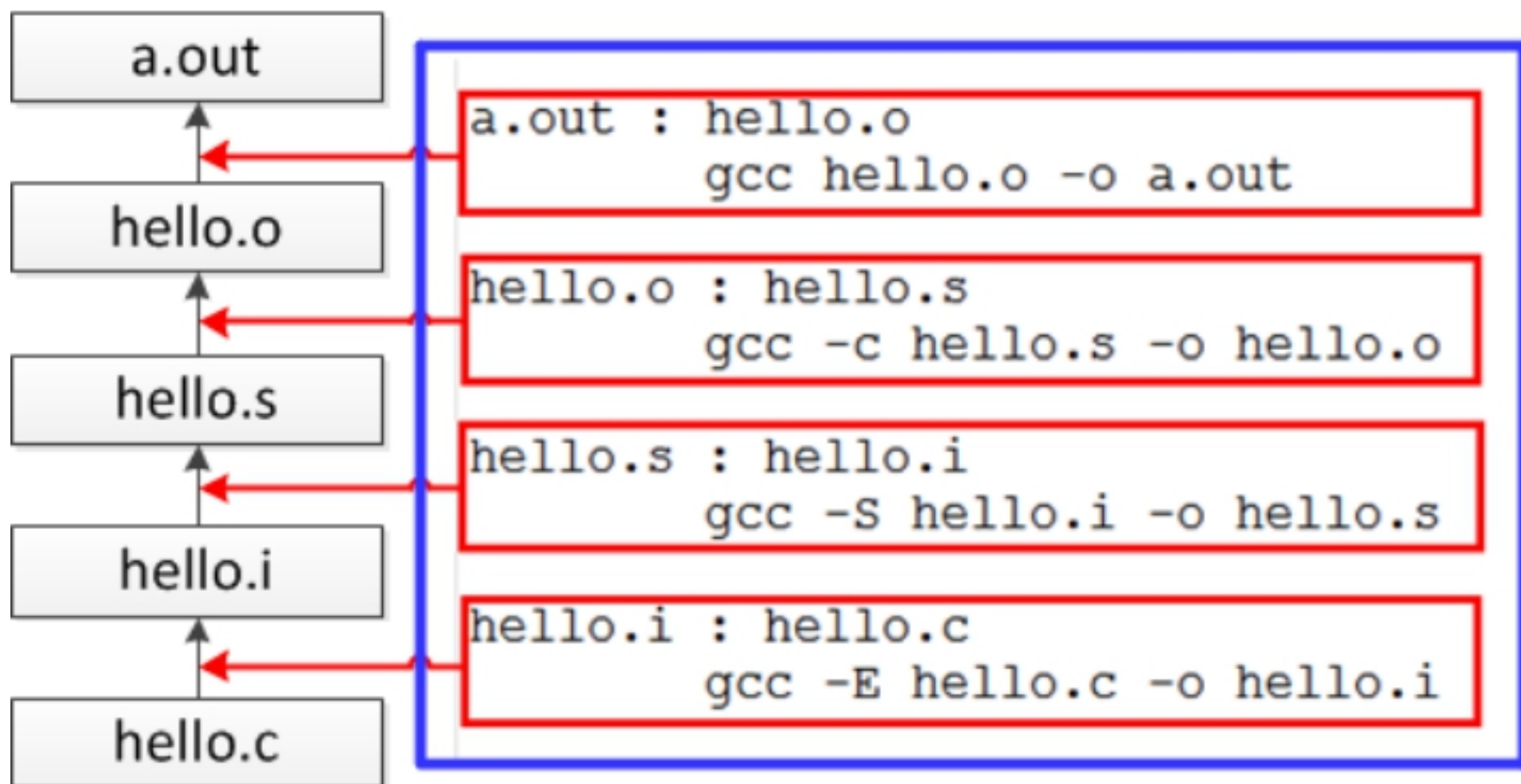
```
.PHONY : clean  
clean:  
    rm -f *.o
```

- 注释: 行注释, 以“#”开头

- Makefile 中规则的执行顺序







谢谢

欢迎交流合作