Лабораторная №4. Создание основных компонентов образа. Ядро Linux.

Цель работы

Изучить что такое ядро ОС и за что оно отвечает, научиться изменять конфигурационный файл для сборки ядра и собрать ядро с заданной конфигурацией.

Подготовительный материал

Ядро и его сборка

Ядро Linux — это самая главная и центральная часть операционной системы Linux.

Если представить ОС в виде слоёв, то ядро находится в самом центре, между оборудованием компьютера (процессор, память, диски) и всеми остальными программами.

Обозначим основные задачи:

- управление аппаратными ресурсами
- обеспечение безопасности и разграничение прав;
- обеспечение работы сетевых протоколов;
- управление файловыми системами, обеспечение чтения и записи данных на диски.

Стоит отметить, что само по себе ядро - это не полноценная ОС. Комбинация ядра Linux и набора программ из проекта GNU (компиляторы, библиотеки, системные утилиты) образует ту самую операционную систему, которую мы называем Linux.

Сборка ядра Linux из исходных кодов — это процесс компиляции и компоновки исходного кода ядра в исполняемые файлы (ядро и загружаемые модули) на основе предоставленной вами конфигурации.

Подготовка к сборке ядра

Предварительно, установим необходимые пакеты:

```
# apt-get install wget build-essential bc flex bison u-boot-tool ncurses-devel
openssl libssl-devel dtc
```

Архив с исходным кодом получим в папке */usr/src/kernel/sources/kernel-source-6.16.tar*, установив соответствующий пакет.

Для этого переключимся на репозиторий Sisyphus и выкачаем из него архивы с пакетами.

```
# apt-repo rm all

# apt-repo set sisyphus

# apt-get update
```

А затем установим нужный пакет

```
# apt-get install kernel-source-6.16
```

Распакуем в удобное место архив и перейдем в его корневую папку.

Далее, нам будет необходимо сконфигрировать файл .config, который собирает в себя все возможные параметры для сборки ядра.

Существует несколько способов создания такого файла. Воспользуемся классическим интерфейсом для make. Выполним

```
$ make menuconfig
```

Откроется интерактивный интерфейс для тонкой настройки, который позволяет представить древовидную структуру всех опций ядра (драйверы устройств, поддержка файловых систем, сетевые функции и т.д.) и выбирать нужные.

В menuconfig все навигируется стрелками. Выбор опций осуществляется с помощью Enter, включение (Y), выключение (N), установка опции как модуль (M) осуществляется с помощью соответствующих клавиш. Кроме того может пригодиться поиск нужных опций конфигурации и их зависимостей (/).

По завершении выбора нужных настроек все сохраняется в файл конфигурации .config.

Пример изменения конфигурации

Приведем **пример** включения параметра PREEMPT_RT, отвечающего за включение в ядро модуля поддержки мягкого реального времени.

Для начала, можно в файл .config включить стандартную конфигурацию ядра, которая заполнит все параметры и теоретически позволяет запуститься ядру на любом устройстве указанной архитектуры. Сделать это можно так.

```
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu- defconfig
```

Эта команда готовит исходный код ядра Linux к сборке для архитектуры RISC-V, используя кросс-компиляцию. Рассмотрим подробнее, что мы передали в качестве аргументов и что будет

выполнено в результате.

ARCH=riscv указывает целевую архитектуру процессора, для которой мы собираем ядро. Исходный код ядра Linux поддерживает десятки архитектур (x86_64, arm, arm64, powerpc, riscv и т.д.). Код, специфичный для каждой архитектуры, лежит в разных поддиректориях (например, arch/x86/, arch/arm/, arch/riscv/). Переменная ARCH говорит утилите make и системе сборки, к какой из этих директорий обращаться для поиска правильных исходников, конфигураций и правил сборки.

В данном случае мы говорим системе — "собирай не для моей родной архитектуры (скорее всего, x86_64), а для RISC-V".

CROSS_COMPILE=riscv64-linux-gnu- задает префикс для набора инструментов компилятора (toolchain), который используется для кросс-компиляции.

Для этого нужен специальный компилятор — кросс-компилятор. Его бинарные файлы обычно имеют префикс, указывающий на целевую архитектуру.

riscv64-linux-gnu- — это и есть тот самый префикс. Система сборки будет вызывать не просто gcc, а riscv64-linux-gnu-gcc; не ld, а riscv64-linux-gnu-ld и так далее. Об установке нужного кросс-компилятора прочитать можно здесь.

defconfig — это цель (target) для make. Она генерирует базовый конфигурационный файл (.config) для выбранной архитектуры.

В директории arch//configs/ (в нашем случае arch/riscv/configs/) лежат несколько заранее подготовленных конфигурационных файлов. defconfig — это обычно конфигурация по умолчанию для данной архитектуры. Она включает все необходимые опции для базовой работоспособности ядра на большинстве устройств с этой архитектурой, но без специфичных драйверов для какого-то конкретного железа.

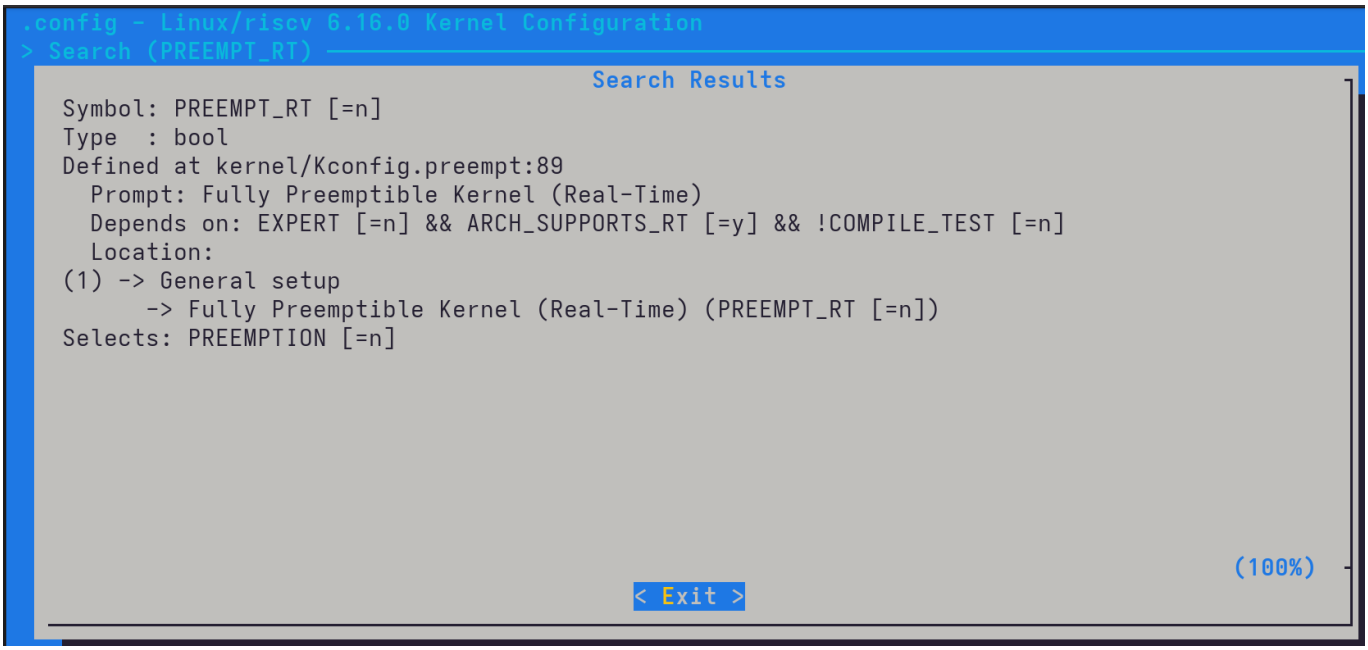
Выполнение этой цели скопирует конфиг из arch/riscv/configs/defconfig в корневую директорию с исходным кодом ядра и сохранит его как файл .config, что и станет отправной точкой для нашей настройки.

Произведем донастройку получившейся конфигурации.

```
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu- menuconfig
```

P.S указание архитектуры здесь необходимо для того, чтобы система сборки обратилась к директории arch/riscv/ для получения всех архитектурно-зависимых настроек, списков плат и драйверов.

Итак, мы в графическом меню, и нам необходимо включить параметр *PREEMPT_RT* для того, чтобы добавить соответствующий модуль ядра в конфигурацию. Жмем (/) и пишем название параметра в появившееся поле ввода.



Получив такой вывод можно сделать вывод о том,где в дереве расположен параметр и какие параметры с какими значениями нужно установить для возможности включения целевого параметра.

Важно отметить, что в окне поиска(в которое можно попасть нажав клавишу (/)) есть возможность быстро перейти к параметру нажатием на клавиатуре (числа) расположенного слева от пути до его расположения. Если параметр в данный момент недоступен для изменения по причине неудоволетворенных зависимостей от других параметров, будет осуществлен переход по той части пути, которая доступна на данный момент.

В нашем случае необходимо чтобы параметр *EXPERT* был включен(он выключен), параметр *ARCH_SUPPORTS_RT* был включен(уже), а параметр *COMPILE_TEST* был отключен(тоже уже так).

Включив параметр *EXPERT* и найдя по указанному пути расположение параметра *PREEMPT_RT* мы получим желаемое.

По завершении указанных действий, можно произвести попытку запуска сборки командой.

```
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu- -j$(nproc)
```

Обобщенный порядок действий для нашего случая можно описать так:

```
$ cd ~/директория/сборки/

# загрузили в .config конфиг по умолчанию
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu- defconfig

# сделали специфичные изменения
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu menuconfig

# запустили сборку ядра
```

```
$ make ARCH=riscv CROSS_COMPILE=riscv64-linux-gnu- -j$(nproc)
```

После окончания сборки по пути *arch/riscv/boot/* можно будет найти бинарный файл готового ядра, а также его сжатый вариант с расширением .gz. Кроме того в других подкаталогах будут сгенерированы внешние модули ядра.

Задание

Ознакомившись с подготовительным материалом к лабораторным №4 решить следующие подзадачи:

- Подготовить или убедиться в наличии необходимых для сборки ядра зависимостей
- Сформировать конфигурационный файл для ядра под riscv-архитектура со стандартными параметрами конфигурации
- Внести изменения в полученный конфигурационный файл:
    - По примеру из подготовительных материалов добавить поддержку патча PREEMPT_RT в конфигурацию
    - В разделе SoC selection в качестве целевого варианта выбрать семейство на базе Allwinner D1
    - Включить поддержку wifi-модуля RTW88_8723DS
- Зафиксировать разницу между полученной и стандартной конфигурацией(например, с помощью утилиты diff)
- Добавить любой параметр в конфигурацию, убедившись в его уникальности внутри подгруппы
- Скомпилировать ядро с полученной конфигурацией
- **Сохранить конфигурационный файл и скопмилированное ядро** для дальнейших лабораторных работ
- **Продемонстрировать работу преподавателю**
- Сформировать отчет о выполнении поставленных задач .doc и **выслать на почту преподавателя до обозначенного срока**