Московский физико-технический институт
(государственный университет)
Физтех-школа радиотехники и компьютерных технологий
Кафедра радиоэлектроники и прикладной информатики

Анализ эффективности системы кросс-сборки crossfs дистрибутива ОС «Эльбрус»

Выпускная квалификационная работа
(бакалаврская работа)

Выполнила студентка 418 группы Беланова А.К.
Научный руководитель: Шалаев М.А.

Москва
2018

Определения

- Кросс-сборка — сборка пакетов для дистрибутива на целевой платформе (Эльбрус, SPARC), выполняемая на инструментальной платформе (x86)
- Кросс-компилятор — компилятор, который запускается на инструментальной платформе и собирает бинарные модули для целевой
- Эффективность системы кросс-сборки — способность выполнять сборку с наименьшими затратами времени

Система кросс-сборки crossfs

crossfs — переносимое программное окружение, разрабатываемое АО «МЦСТ», позволяющее собирать пакеты для платформ семейства Linux различных аппаратных архитектур; основано на системе скриптов any

crossfs поддерживает тулчейны gcc и lcc

> **any** — свободно распространяемая система скриптов для получения бинарных программных пакетов под любую POSIX-систему
>
> **lcc** — разрабатываемый АО «МЦСТ» компилятор для архитектур SPARC, Эльбрус и используемый для кросс-компиляции

crossfs используется для сборки дистрибутива для вычислительных комплексов на базе процессоров семейств Эльбрус (e2k, e3s, e2s, e8c, e8c2, e1c+), SPARC (e90s, e90), Intel (x86, x86_64)

Цель работы

Разработка и внедрение инструментария для анализа эффективности в систему кросс-сборки crossfs

Требования

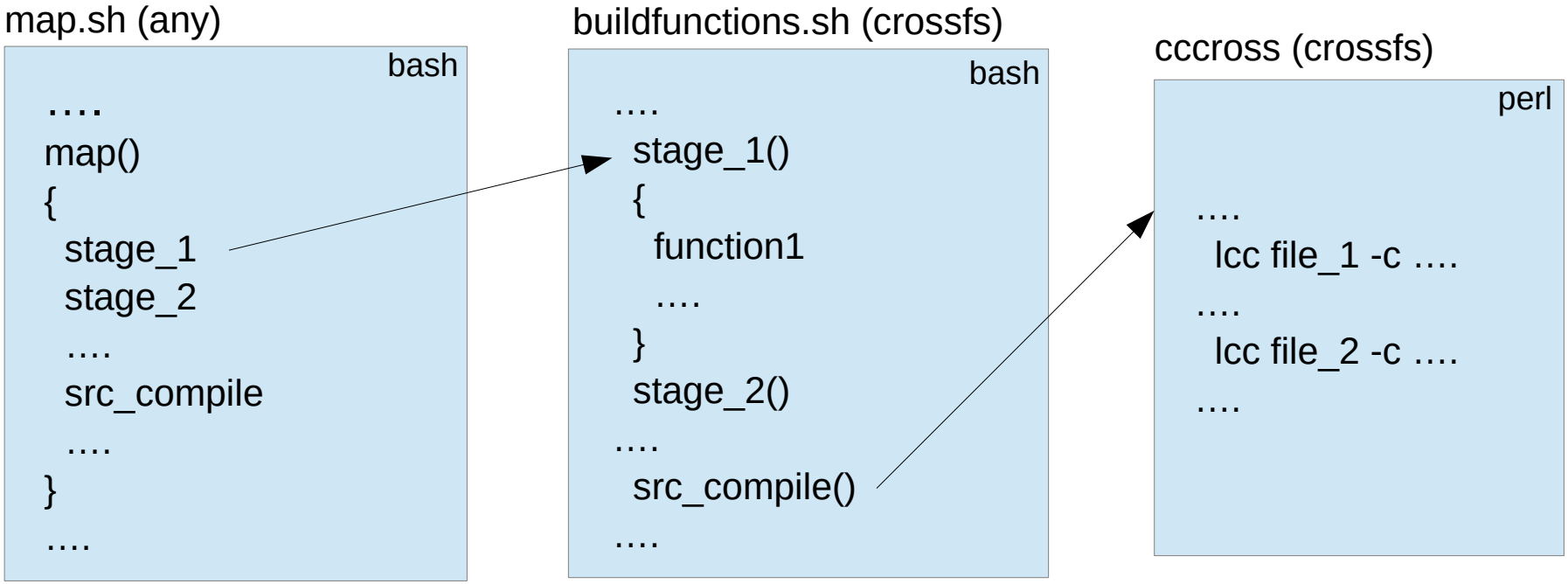
- Автоматический анализ времени отдельных стадий кросс-сборки
- Детальный анализ времени стадии компиляции

Алгоритм кросс-сборки

1. Подготовка окружения сборки (обработка и подготовка зависимостей)
2. Подготовка исходного кода пакета (получение исходных файлов из репозитория и применение патчей)
3. Основной этап сборки (конфигурация, компиляция и связывание, установка)
4. Формирование бинарного архива для установки на вычислительный комплекс

Алгоритм реализован в методе map() скрипта map.sh системы скриптов any

Скрипты системы crossfs



buildfunctions.sh — скрипт системы crossfs, реализующий отдельные стадии кросс-сборки

cccross — скрипт системы crossfs, осуществляющий преобразование, адаптацию к режиму кросс-сборки и исполнение команд компиляции

Последовательность стадий кросс-сборки в методе map()

pkg_pretend — ранняя проверка сборки с доступом только для чтения

pkg_setup — ранняя подготовка сборки с разрешением записывать файлы

pkg_context — обработка пакетных зависимостей и подготовка контекста для сборки

Подготовка окружения сборки

src_fetch — получение из репозитория исходных текстов программы

src_prepare — применение патчей на исходные тексты пакета

Подготовка исходного кода пакета

Последовательность стадий кросс-сборки в методе map()

src_config — конфигурация исходных текстов пакета

src_compile — компиляция и связывание исходного кода

pkg_rminstall — удаление файлов, установленных при прошлой сборке

src_install — копирование результатов сборки в выделенную директорию с нужной иерархией каталогов

Основной этап сборки

pkg_install — установка дополнительных файлов, описанных политикой сборки, для развертывания на целевой системе

pkg_config — подготовка файлов для установочных архивов

pkg_root — открытие доступа другим пакетам к файлам в сборочном контексте

pkg_pack — создание бинарных установочных пакетов

Формирование бинарного архива для установки

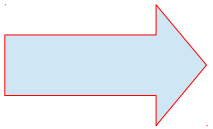
Реализация анализа времени исполнения стадий кросс-сборки

map.sh (any)

bash

```
….
map()
{
  stage_1
  stage_2
  ….
  src_compile
  ….
}
….
```

Исполнение метода map()



timemap.sh (any)

bash

```
….
map()
{
  time_start
  stage_1
  time_end
  time_start
  stage_2
  time_end
  ….
  time_start
  src_compile
  time_end
  ….
}
….
```

Переопределение и исполнение метода map()

time_start — получение времени старта исполнения стадии с помощью date

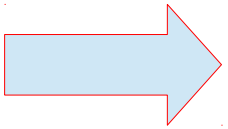
time_end — аналогичное получение времени окончания исполнения стадии, вычисление длительности исполнения, сохранение информации в трассировочном файле

Реализация детального анализа времени стадий

Каждая стадия метода map() вызывает функции из buildfunctions.sh (количество функций в скрипте — 72). Скрипт был модифицирован:

buildfunctions.sh (crossfs)

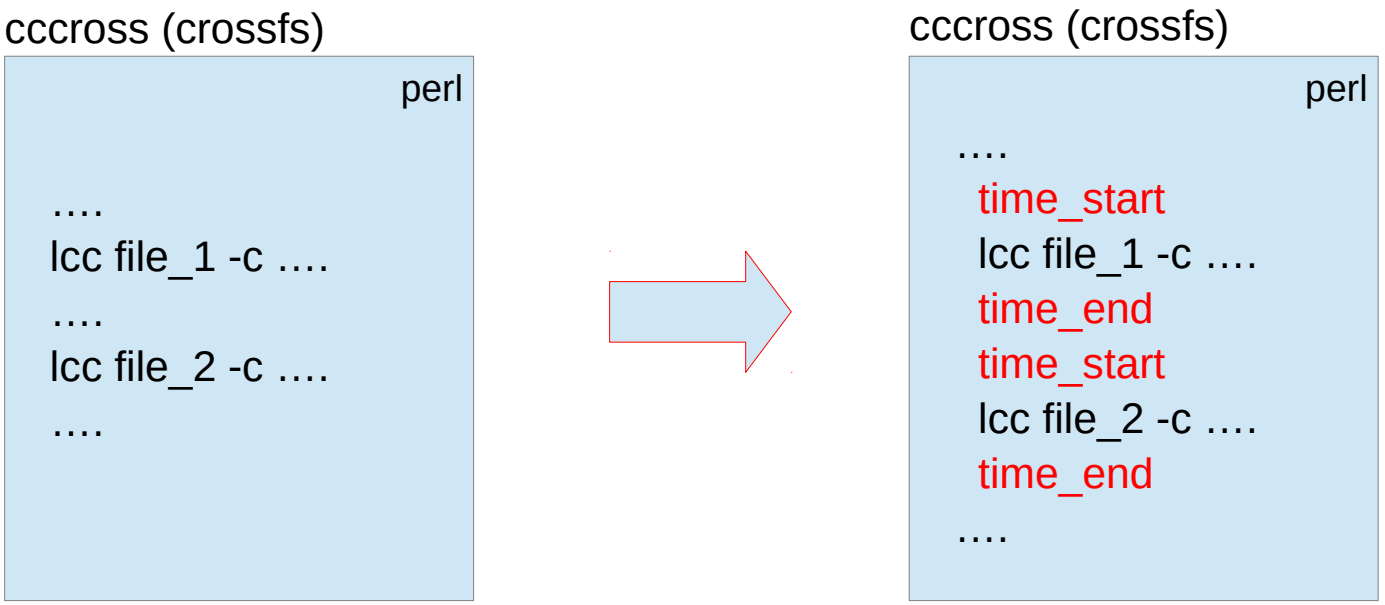
```
                                    bash
....
  stage_1()
  {
    function1
    ....
  }
  stage_2()
....
  src_compile()
....
```



buildfunctions.sh (crossfs)

```
                                    bash
....
  stage_1()
  {
    time_start
    function1
    time_end
    ....
  }
  stage_2()
....
  src_compile()
....
```

Реализация детального анализа времени стадии компиляции



time_start — получение времени старта компиляции команды с помощью gettimeofday модуля Time::HiRes для точности до 6-го знака после запятой
time_end — аналогичное получение времени окончания компиляции команды, вычисление длительности компиляции команды, сохранение информации в трассировочном файле

Новый подход к модификации системы кросс-сборки crossfs

Пример применения инструментария анализа эффективности системы

При использовании инструментария было обнаружено, что исполнение стадии pkg_context занимает много времени. В ходе исследования кода стадии было выяснено, что можно получить уменьшение времени исполнения pkg_context за счёт единоразовой установки путей.

Уменьшение времени исполнения стадии pkg_context:

Пакеты	Дистрибутив devel-2.5.8, сек	Дистрибутив devel-3.0, сек
opencv-3.1.0	104	51
mysql-5.7.17	67	62
mesa-17.0.0	22	14

Результаты

- Разработан инструментарий для оценки времени отдельных стадий сборки пакетов дистрибутива
- Инструментарий внедрен в систему кросс-сборки дистрибутива ОС «Эльбрус»
- Внедрение инструмента позволило обнаружить неэффективную реализацию стадии **pkg_context**