ЛАБОРАТОРНАЯ РАБОТА №4
ВВЕДЕНИЕ В АРХИТЕКТУРУ ARM

Цели работы

1. Знакомство с программной архитектурой ARM.
2. Анализ ассемблерного листинга программы для архитектуры ARM.

1. КРАТКОЕ ОПИСАНИЕ АРХИТЕКТУРЫ ARM

Архитектура ARM (Advanced RISC Machine) является широко распространенной (в особенности на рынке мобильных и встраиваемых устройств) архитектурой класса RISC (Reduced Instruction Set Computer), который является альтернативой класса архитектур CISC (Complex Instruction Set Computer – к нему, в частности, принадлежит семейство архитектур x86).

RISC-процессоры возникли позже CISC-процессоров, и в своей организации более соответствуют современному положению вещей. CISC-процессоры создавались в расчёте на быструю (относительно процессора) и очень ограниченную по объему память и написание программ на ассемблере. Соответственно, часто используемые команды кодируются по возможности короткими кодами, имеются сложные команды, позволяющие кратко описывать сложные действия, а в качестве операндов наравне используются и регистры и память. Позднее стало ясно, что память дешевеет, увеличивается в объеме, но замедляется относительно скорости процессора. При этом развиваются технологии компиляции, и программирование переходит на языки более высокого уровня, чем ассемблер. В соответствии с этими тенденциями были разработаны RISC-процессоры.

RISC-архитектуры имеют следующие основные особенности. Во-первых, упрощен набор инструкций. Команды просты в декодировании – они имеют одинаковый простой формат и кодируются машинным словом фиксированной длины. RISC-команды просты в исполнении, т.е. они

исполняются быстро, и, преимущественно, одинаково по времени – отсутствуют команды, которые бы выполнялись много тактов. Например, часто отсутствуют команды целочисленного деления или вычисления квадратного корня. Они выражаются через более простые команды компилятором. За счёт такого упрощения команд улучшается их конвейеризация и суперскалярные свойства процессора.

Во-вторых, большой (по сравнению с CISC) регистровый файл. Регистры, как правило, нумеруются, а не именуются. Все арифметические команды выполняются именно над регистрами, а не над оперативной памятью. Для работы с памятью же существует всего две команды – чтение (`load`) и запись (`store`), позволяющие считывать значения ячеек памяти в регистр и записать значения из регистров в память. Память, таким образом, не используется для хранения временных значений (что является обычным делом в CISC) системах.

В-третьих, RISC-архитектуры не рассчитаны на удобство программирования вручную, на языке ассемблера. Ставка сделана на компиляторы, которые сгенерируют машинный код и проведут его оптимизацию автоматически. RISC-код является менее ёмким, чем CISC-код, он занимает больше места и обычно менее удобен для восприятия человеком.

В целом, RISC-архитектура является более удобной для реализации в процессоре, чем CISC-архитектура. Поэтому современные CISC-процессоры, как правило, реализованы на базе RISC-ядра – сложные CISC-команды расщепляются на микрокоманды, и далее процессор функционирует уже как RISC. CISC-оболочка, по большому счёту, нужна лишь для поддержки бинарной совместимости с существующим кодом.

Регистры. Программисту доступны 16 32-битных регистров общего назначения (r0 – r15) и 32-битный регистр состояния (CPSR).

Регистры общего назначения используются для хранения произвольных целочисленных данных. При использовании в большинстве команд в качестве операндов эти регистры полностью взаимозаменяемы, но

некоторые команды используют только конкретные регистры. Регистр r15 используется как счетчик команд (program counter, instruction pointer). Регистр r14 используется для сохранения адреса возврата в результате исполнения специальной команды условного перехода. Регистр r13 обычно используется как указатель вершины стека (stack pointer), хотя и не поддержан специально программной архитектурой.

Регистр состояния (CPSR – current program status register) хранит однобитовые флаги:

- знаковый флаг (N – Negative),
- флаг нуля (Z – Zero),
- флаг переноса (C – Carry),
- флаг переполнения (V – oVerflow),

а также другие поля, отражающие состояние процессора.

Команды. Можно выделить следующие основные группы команд:

- Команды преобразования данных: сложение, вычитание, умножение, умножение со сложением, побитовые логические операции, сравнение.
- Команды передачи данных: копирование данных между регистрами, чтение и запись данных в памяти. Для операций с памятью поддерживается автоматическое увеличение регистра-указателя до или после обращения в память. Кроме того, поддерживаются многорегистровые операции с памятью, использующие сразу несколько регистров общего назначения.
- Команды передачи управления: команды безусловного и условного перехода. В архитектуре поддерживается условное исполнение любой команды в зависимости от значений флагов. Кроме того, существует быстрый способ вызова подпрограммы с использованием специальной команды перехода, сохраняющей адрес следующей за ней программы в регистре r14.

2. ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Изучить основы программной архитектуры ARM.
2. Для программы на языке Си (из лабораторной работы 1) сгенерировать ассемблерные листинги для архитектуры ARM, используя различные уровни комплексной оптимизации.
3. Проанализировать полученные листинги и сделать следующее:
    - Сопоставьте команды языка Си с машинными командами.
    - Определить размещение переменных языка Си в программах на ассемблере (в каких регистрах, в каких ячейках памяти).
    - Описать и объяснить оптимизационные преобразования, выполненные компилятором.
    - Продемонстрировать использование ключевых особенностей архитектуры ARM на конкретных участках ассемблерного кода.
4. Составить отчет по лабораторной работе. Отчет должен содержать следующее.
    - Титульный лист.
    - Цель лабораторной работы.
    - Полный компилируемый листинг реализованной программы и команды для ее компиляции.
    - Листинг на ассемблере с описаниями назначения команд с точки зрения реализации алгоритма выбранного варианта.
    - Вывод по результатам лабораторной работы.

3. ВАРИАНТЫ ЗАДАНИЙ

Варианты заданий взять из лабораторной работы №1.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите отличия архитектур CISC и RISC. Продемонстрируйте их на примере архитектур x86/x86-64 и ARM.
2. Что такое условные команды? Приведите примеры условных команд в архитектуре ARM.
3. Назовите особенности команд обращения в память в архитектуре ARM.