Оглавление

От автора

Первые микропроцессоры появились на свет в 1971 году. В то время их мощности хватало только для обработки 4 бит за операцию, и максимум на что они годились, так это на использование в качестве калькуляторов и других простейших устройств. Однако само их появление было технологическим прорывом, поскольку использованная в них идея реализации нужных функций путём разработки программного обеспечения без создания новой аппаратуры дала импульс к снижению затрат на разработку оборудования, применению электроники во множестве различных устройствах, минитюаризации, энергосбережению и росту функциональности устройств, в которых они применялись. Микропроцессоры продолжали развиваться в направлении увеличения размера данных, обрабатываемых одной командой, и уменьшения времени выполнения одной команды. По мере этого сфера их применения всё более расширялась, а это, в свою очередь, давало импульс к повышению их производительности. Разрядность вычислений увеличивалась с 8 до 16, 32 и даже до 64 бит, а вместе с тем были созданы такие архитектуры обработки, как SIMD (Single-Instruction Multiple-Data — один поток команд и много потоков данных), которая позволяет разбивать длинное битовое поле и обрабатывать его в качестве нескольких более коротких значений, суперскалярная архитектура, архитектура VLIW (Very Long Instruction Word — сверхдлинное командное слово), в которых параллельно работают несколько вычислительных блоков. Архитектура центрального процессора (ЦП) тоже менялась: сначала была CISC-архитектура (Complex Instruction Set Computer — процессор с комплексной системой команд), основным средством разработки которой был язык ассемблера, потом на смену пришла RISC (Reduced Instruction Set Computer — процессор с сокращённым набором команд), в которой благодаря увеличению производительности и объёма памяти команды выполняются на конвейере, а для разработки используются главным образом такие языки, как C и C++.

SuperH является одним из распространённых в Японии семейств 32-битных микропроцессоров, в котором вопреки общепринятым правилам построения RISC-архитектур были использованы команды 16-битной длины. В это семейство входят, во-первых, микроконтроллеры, способные на аппаратном уровне с высокой скоростью обрабатывать множественные прерывания. Микроконтроллеры объединены в подсемейства SH-1, SH-2, SH2-DSP, SH-2E, SH-2A. Во-вторых, в это семейство входят микропроцессоры общего назначения, содержащие блок управления памятью (MMU), предоставляющий механизм управления памятью для реализации операционных систем (ОС) широкого применения. Микропроцессоры этого типа объединены в подсемейства SH-3, SH3-DSP, SH-4, SH-4A,

SH4AL-DSP. В данной книге будут описаны в основном микропроцессоры SuperH общего назначения.

В отличие от однокристальных микроконтроллеров в микропроцессорах используются MMU, кэш-память, а также внешняя память, например динамическое ОЗУ (DRAM). Наверное, есть много инженеров, которым не приходилось использовать кэш-память и синхронную DRAM, хотя они и слышали о них. В данной книге будут описаны принципы их работы и то, как они влияют на производительность системы.

Если прекрасное «железо» не подкреплено удобными средствами разработки программ, то не стоит ожидать популярности этого «железа» у разработчиков. Для семейства SuperH написано множество компиляторов, распространяемых на бесплатной основе в комплекте с исходными кодами (на базе лицензии GNU), в том числе и компанией Renesas, а также несколько ОС реального времени и ОС широкого применения, в том числе и на базе Linux. В этой книге не только описывается архитектура ЦП, но и приводятся некоторые справочные сведения о вышеупомянутом программном обеспечении.

Автор надеется, что с помощью этой книги читатели не только получат представление о микропроцессорах SuperH, но и научатся их использовать. За счёт расширяемости архитектуры и лучшей, чем у традиционных однокристальных микропроцессоров, производительности микропроцессоры SuperH позволят разработчикам наилучшим образом реализовать все задумки.

ГЛАВА 1

АРХИТЕКТУРА ЦЕНТРАЛЬНОГО ПРОЦЕССОРА

Семейство SuperH — это группа встраиваемых микропроцессоров с оригинальной RISC-архитектурой, разработанных компанией Renesas Technology. При их разработке стремились добиться (**Рис. 1.1**):

- высокой производительности;
- компактности;
- повышенной эффективности.

Эти микроконтроллеры применяются в экономичных кондиционерах, видеорекордерах с жёстким диском, мобильных телефонах, системах автомобильной навигации, игровых приставках и других изделиях. В настоящее время в семействе SuperH есть две группы микропроцессоров: микропроцессоры первой группы — SH-1, SH-2, SH2-DSP, SH-2E, SH-2A — оптимизированы для использования в качестве контроллеров, а второй группы — SH-3, SH3-DSP, SH-4, SH-4A, SH4AL-DSP — в качестве процессоров общего назначения.

В обеих группах представлены высокопроизводительные микропроцессоры, старшие модели которых отличаются от младших не только более высокими частотами, но и улучшенными характеристиками, оптимизированной системой команд, а также наличием встроенного FPU (устройства для выполнения операций с плавающей точкой) и DSP (процессора цифровой обработки сигналов).

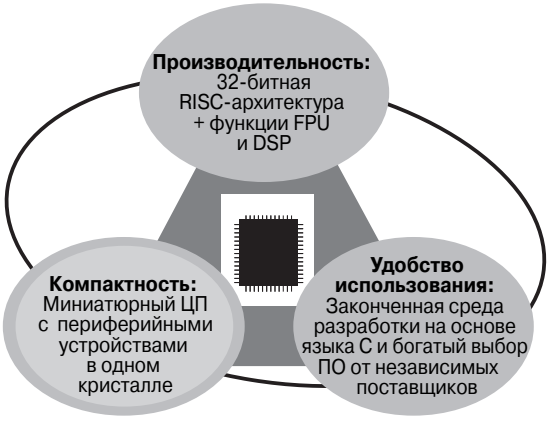


Рис. 1.1. Особенности семейства SuperH

Благодаря встраиванию в микроконтроллер периферийных узлов, рассчитанных на конкретную область применения, разработчики получили прекрасную

возможность создавать законченные однокристальные устройства. Кроме того, вместе с микроконтроллерами разработчики получают интегрированную среду разработки *High-Performance Embedded Workshop* (HEW), позволяющую проделать в единой программной оболочке всю работу — от подключения к средствам разработки верхнего уровня до использования редакторов, компиляторов, сред моделирования, эмуляторов, анализа результатов испытаний и подключения к средствам управления версиями.

1.1. Общие сведения

Центральный процессор семейства SuperH представляет собой устройство с 16-битной RISC-архитектурой.

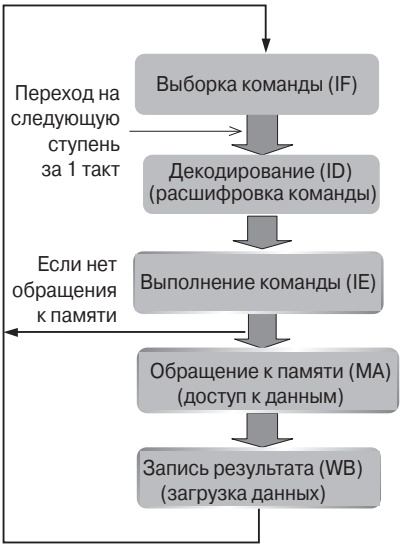
При разработке процессорного ядра — устройства для выполнения команд, которое является центральной частью микроконтроллера, — одна из стоящих перед разработчиком задач заключается в том, чтобы выполнить требуемую обработку как можно быстрее, другая задача — уменьшить потребляемую при этом электроэнергию.

Вся история повышения производительности процессоров сводится к следующему:

1. Увеличение разрядности данных, обрабатываемых за 1 такт.
2. Увеличение тактовой частоты.
3. Увеличение количества команд, выполняемых за 1 такт.

В 70-х и первой половине 80-х годов XX века, когда техника оптимизации компиляторов была ещё незрелой, основной архитектурой системы команд была CISC (**Рис. 1.2**). Целью при этом было выполнение одного оператора языка высокого уровня (в том числе и производящего сложные вычисления) одной машинной командой. Однако это породило следующую проблему: при попытке реализовать слишком сложные команды происходило усложнение схемы, возникали неполадки, замедляющие этап разработки процессора. Кроме того, в процессе анализа эффективности CISC-архитектуры выяснилось, что частота использования сложных операторов в системе мала, а замена одного такого оператора на несколько более простых инструкций почти не сказывается на производительности, в то время как к наибольшему повышению производительности приводит увеличение скорости выполнения часто используемых команд. Другими словами, стало ясно, что разработанные с огромными усилиями схемы не дают повышения производительности, адекватного увеличению занимаемой ими площади кристалла. К тому времени начала совершенствоваться и техника оптимизации компиляторов.

В этот момент и была предложена архитектура RISC, родившаяся в процессе поиска способов ускорения выполнения только часто используемых или абсолютно необходимых для работы машинных инструкций. В основе этой архитектуры лежит идея достижения высокой производительности путём выполнения команды за один такт. Полностью отработать команду — от выборки из памяти до завершения выполнения — за один такт сложно, поэтому обработка производится в конвейерном режиме. Например, если использовать трёхступенчатый конвейер, то на



а) Основные действия ЦП (на примере SuperH)

б) Архитектуры CISC и RISC

Рис. 1.2. Функционирование микроконтроллера

первой ступени выборки команды за один такт производится только считывание команды из памяти и никакие другие операции, например декодирование, при этом не производятся. На второй ступени, ступени декодирования, производится только расшифровка команды. На третьей ступени производится выполнение команды. За счёт конвейеризации обработки для завершения первой инструкции потребуется 3 такта, однако вторая команда в момент завершения первой будет уже на стадии декодирования, поэтому её можно будет завершить за 1 такт. Таким образом, все команды, начиная со второй, завершаются за 1 такт, и условие достижения скорости в один такт на инструкцию выполняется. Для выполнения этого условия процессор должен соответствовать по меньшей мере двум требованиям:

1. Память и ширина шины команд должны позволять производить выборку за один такт.
2. Выполнение должно завершаться за один такт.

Другими словами, процессор с шириной шины в 32 бита должен иметь длину команды не более 32 бит. Кроме того, он должен реализовать только команды, выполняющие элементарные действия, которые можно завершить за один такт. Именно это и является сутью архитектуры RISC.

Пусть, например, процессор имеет разрядность 32 бита, тогда он, скорее всего, будет использовать 32-битное пространство адресов размером 4 ГБ. В этом случае код операции и адрес невозможно одновременно разместить в команде длиной 32 бита, и тогда доступ к памяти по абсолютным адресам будет невозможен. Поэтому указание адресов для пересылки данных производится косвенно, с помощью внутренних регистров процессора. Кроме того, в командах вычисления тоже используются только внутренние регистры. Следовательно, так как в этом случае понадобятся регистры и для указания адреса, и для вычислительных операций над данными, то, чем больше будет внутренних регистров, тем лучше. Именно поэтому многие устройства имеют более 16 внутренних регистров. Правда, если таких регистров слишком много, то на сохранение и восстановление контекста при переключении задач уходит много времени, поэтому выбор числа внутренних регистров необходимо производить с учётом приложений, в которых данное устройство будет использоваться.

Многие процессоры с архитектурой RISC, работающие с данными с разрядностью 32 бита, имеют длину команды 32 бита, хотя на самом деле эта длина определяется шириной поля указания регистра в коде команды. Так, если число внутренних регистров равно 32, то один операнд займёт 5 бит. Для 3 операндов в этом случае понадобится 15 бит, поэтому, с учётом длины кода операции, вполне естественно выбрать длину команды равной 32 битам. Однако это не означает, что все 32 бита кода являются значимыми. Кроме того, некоторые команды выполняют вычисления всего с 2 операндами, из-за чего доля значимых битов в коде хранящихся в памяти команд вновь уменьшается.

Если бы команды имели переменную длину, как в архитектуре CISC, то была бы возможность повысить плотность кода путём назначения для часто используемых команд коротких кодов операций, однако в архитектуре RISC это невозможно (**Рис. 1.3**).

В RISC-микроконтроллерах SuperH высокая плотность кода обеспечивается за счёт использования 16-битных команд, что стало возможным из-за того, что число внутренних регистров ограничено шестнадцатью и задействовано всего 2 операнда в команде.

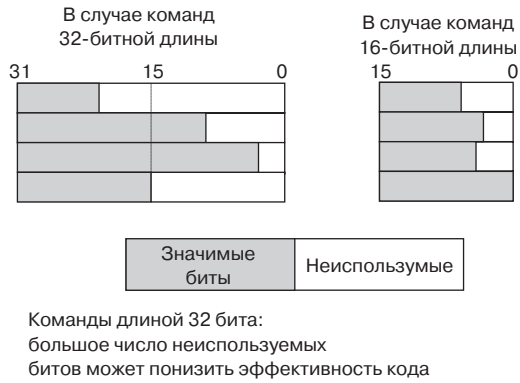


Рис. 1.3. Разница в плотности кода в зависимости от длины команды

Длина команды в 16 бит даёт ещё одно преимущество, связанное со структурой шины. На самом деле если выполняется одна команда за такт, некоторым командам нужно будет использовать дополнительный цикл шины (например, для пересылки данных) и, если шина только одна, в это время произвести выборку команды будет невозможно. В подобном случае приходится использовать так называемую *гарвардскую архитектуру*, в которой шина и память данных отделены от шины и памяти команд. Однако при длине команды 16 бит можно производить одновременную выборку двух команд по 32-битной шине и без использования гарвардской архитектуры. Так как при этом один раз из двух пропускается цикл шины выборки команды, то даже в негарвардской (фоннеймановской) архитектуре можно обеспечить условие достижения скорости в 1 такт на команду. Это даёт очень большой эффект в смысле энергосбережения, поэтому другие производители микроконтроллеров тоже используют эту идею.

Чтобы ещё более повысить производительность, необходимо за 1 такт выполнять несколько команд. Для этого как по отдельности, так и в комбинации друг с другом можно использовать описанные ниже способы:

1. Суперскалярная архитектура.
2. Архитектура VLIW (архитектура с командными словами сверхбольшой длины).
3. Архитектура SIMD (архитектура с одним потоком команд и многими потоками данных).

Суперскалярная архитектура — это способ организации вычислений, при котором несколько команд выполняются несколькими арифметико-логическими устройствами. В такой архитектуре возможна, например, одновременная работа блока целочисленных вычислений и блока вычислений с плавающей точкой, поэтому если группа команд включает оба этих типа вычислений, то эти команды могут выполняться одновременно.

В семействе SuperH есть процессоры, в которых используется суперскалярная архитектура с 2 конвейерами.

В *архитектуре VLIW* увеличена длина одной команды, в состав которой включены разнообразные операции (выполняемые отдельными схемами микропроцессора), поэтому эту архитектуру также можно, наверное, считать расширением CISC.

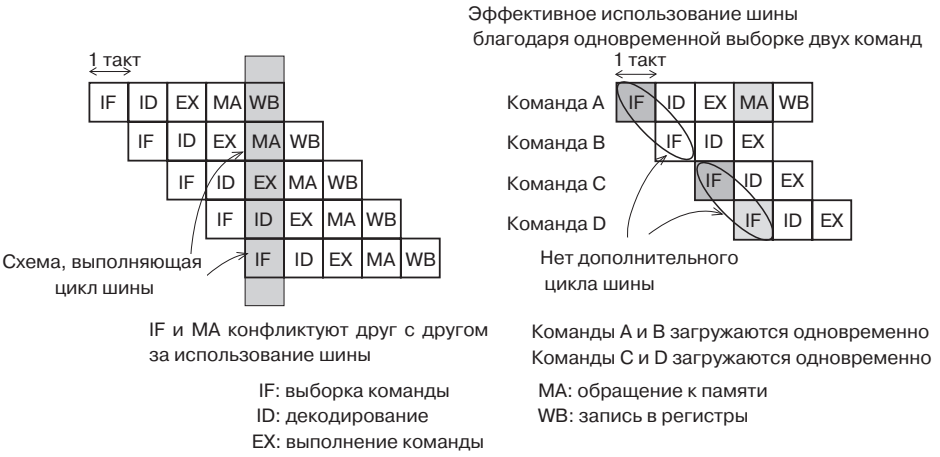


Рис. 1.4. Эффект от длины команды в 16 бит

В *архитектуре SIMD* с помощью одной команды обрабатываются множественные данные, благодаря чему возможно увеличение скорости определённых вычислительных операций. Например, 32-битный регистр используется как четыре 8-битных значения и одной командой производится их сложение с соответствующими значениями в другом 32-битном регистре, а в результате можно увеличить скорость в 4 раза по сравнению с обработкой 4 командами (**Рис. 1.5**). В семействе SuperH тоже есть процессоры, использующие SIMD.

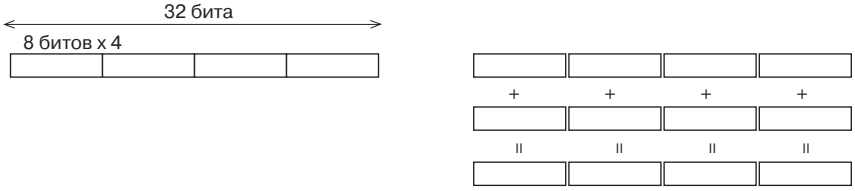


Рис. 1.5. Пример SIMD

Кроме того, если осуществляется предварительное (до исполнения) считывание команд, то в случае выполнения операции перехода приходится отбрасывать их и производить новое считывание команд по адресу перехода, т. е. потребуется дополнительное время обработки (это справедливо как для RISC, так и для CISC). Данное явление называется *штраф перехода*. Наверное, многие читатели часто используют управляющие конструкции типа if, while, switch, for. Все они являются командами перехода (считается, что переходы составляют 20...30% всех инструк-