Лекция 5
РАЗВИТИЕ АРХИТЕКТУРЫ МИКРОПРОЦЕССОРОВ

Микропроцессоры – это программно управляемая сверхбольшая интегральная схема (или сборка схем), управляющая работой компьютера и выполняющая большую часть обработки информации.

Микропроцессоры, появившись в 1971 г. как «начинка» калькулятора, где они выполняли базовые арифметические операции с двоично- десятичными числами, через два десятилетия стали основой всех вычислительных машин.

Из основных дат развития микропроцессоров отметим следующие:

•самый первый микропроцессор SLF – 1968 г.;
•первый массовый микропроцессор Intel 4004 – 15 ноября 1971 г.;
•первые микропроцессоры для компьютеров i8080 и MC6800 – 1974;
•основные процессоры Intel 8086 – 1978 г., 286 – 1982, 386 – 1986, Pentium Pro – 1995, Pentium 4 – 2000, Pentium M – 2003, Core – 2006;
•разработка RISC-процессоров – около 1980 г.;
•первый двухядерный процессор – 2001 г. (Power4).

§5.1. Основные архитектурные решения, применяемые в микропроцессорах

•Принстонская/гарвардская архитектуры
•Конвейерная архитектура
•Суперскалярная архитектура
•полный и урезанный наборы команд (CISC/RISC-процессоры)
•Многоядерность
•Кэширование
•Векторность

Большинство архитектурных методов, воплощенных в микропроцессорах, было перенесено из процессоров «больших» ЭВМ, которые в 70-е гг. опережали в своем развитии на десятилетия.

Гарвардская архитектура, предусматривающая *раздельный доступ к инструкциям и данным*, была разработана в 1930-е гг. в Гарварде Г. Эйкеном, создавшим позднее Harvard Mark I. Ввод инструкций в Mark I осуществлялся с перфоленты, данных – набором регистров. *Раздельный ввод данных и команд позволял ускорить работу, но удорожал схему*.

Принстонская архитектура, предусматривающая *хранение программ в общей памяти с данными*, была разработана в 1940-е гг. в Пенсильванском и Принстонском университетах. По имени одного из руководителей работы она также называется «фон Неймановской». Совместное размещение инструкций и данных *повысило гибкость вычислительных систем в плане обработки данных*. Кроме того, такая архитектура была *проще в реализации*, поэтому она нашла повсеместное применение.

В 1970-е гг. вновь обратились к гарвардской архитектуре, она стала применяться в микроконтроллерах. Затем, вследствие возникновения т.н. «бутылочного горлышка фон Неймана», её элементы стали появляться и в микропроцессорах – например, в виде раздельной кэш-памяти для команд и данных или в виде запрета на исполнение данных как инструкций.

Конвейер для инструкций (pipelining) был впервые применен в компьютерах ILLIAC II (1962 г.) и IBM 7030 Stretch (1961 г.), позже С. Крей в суперкомпьютерах XMP применил его для операции многократного умножения и сложения (1982 г.). *Конвейерная архитектура позволяет повысить быстродействие за счет начала обработки команды до окончания обработки предыдущей.* Обычно для выполнения каждой команды требуется осуществить некоторое количество однотипных операций, например: выборка кода операции из памяти, дешифрация команды, адресация операнда и его выборка из памяти, выполнение команды, запись результата в память.

Каждой из этих операций сопоставляют одну ступень конвейера. После освобождения от выполнения элементарной операции одной команды ступень конвейера сразу приступают к работе над следующей командой. Так процессор обрабатывает одновременно несколько команд, находящихся на разных стадиях выполнения. Быстродействие, в самом оптимистичном случае, пропорционально длине конвейера.

Ряд факторов снижает эффективность конвейерной архитектуры – это простой конвейера, когда некоторые ступени не используются, ожидание, если следующая команда использует результат предыдущей, и очистка конвейера при выполнении перехода. Для повышения эффективности конвейера команды делают максимально схожего формата, применяют внеочередное выполнение команд и предсказание переходов. Последний способ особенно развит в современных процессорах, некоторые из которых имеют более 30 ступеней.

Суперскалярность – это способность параллельного выполнения нескольких машинных инструкций, которая обеспечивается работой нескольких декодирующих блоков, нагружающих множество исполнительных блоков.

В аппаратуру процессора закладываются средства, позволяющие одновременно выполнять две или более скалярные операции, т. е. команды обработки пары чисел. Суперскалярная архитектура базируется на **многофункциональном параллелизме** и позволяет увеличить производительность компьютера пропорционально числу одновременно выполняемых операций.

Реализация суперскалярной обработки заключается в чисто аппаратном механизме выборки из буфера инструкций несвязанных команд и параллельном запуске их на исполнение.

Суперскалярная аппаратура динамически строит план вычислений на основе последовательного кода программ. Хотя такой подход и увеличивает сложность физической реализации, скалярный процессор создает план, используя преимущества тех факторов, которые могут быть определены только во время выполнения.

Первым суперскалярным компьютером считается CDC 6600, разработанный С. Креем в 1964 г., а первыми суперскалярными массовыми микропроцессорами – SuperSPARC (1992) и Pentium (1993).

Практически все современные процессоры являются такими, они оснащены несколькими блоками выборки, декодирования, арифметических операций и пр. В архитектуре VLIW суперскалярность закладывается в инструкции на стадии компиляции.

CISC (Complex Instruction Set Computing) – концепция проектирования процессоров, характеризующаяся полным набором команд. Объединение нескольких машинных операций в одной команде позволило непосредственно поддерживать языки высокого уровня, увеличить плотность кода и уменьшить обращения к памяти. В начале 1960-х это позволило добиться большой экономии на памяти и большей производительности при программировании на ассемблере, т.к. языки высокого уровня, такие как Алгол или Fortran, были не всегда доступны.

Для CISC-процессоров характерно *большое количество машинных команд, некоторые из которых функционально аналогичны операторам высокоуровневых языков программирования*, большое количество методов адресации и большое количество поддерживаемых форматов команд различной разрядности.

На первых этапах реализация в компьютерах большого количества инструкций была оправдана, т.к. языки высокого уровня только начали развиваться, и это было удобно для программистов. *Использование сложных инструкций позволяло сократить размеры программ, а значит уменьшить требования к памяти и время выполнения.* А использование микропрограмм смягчало проблему пропускной способности памяти.

К началу восьмидесятых годов классические CISC полностью исчерпали себя, расширять набор инструкций больше не имело смысла. Процессор VAX поддерживал инструкцию «провести интерполяцию полиномом», а процессоры Motorola 68k поддерживали до двенадцати режимов адресации. Из-за высокой сложности процессоров оказалось трудно наращивать их тактовую частоту, а инструкции сложно было не только выполнять, но и декодировать

Первым представителем CISC среди компьютеров считается System/360, среди микропроцессоров это Intel 8086 (1978) и Motorola 68000 (1979).

RISC (Reduced Instruction Set Computing) – вычисления с сокращённым набором команд. Концепция разработана независимо в IBM (1975) и университете Беркли (1980) с целью преодоления недостатков микропроцессоров CISC. Архитектура основана на статистическом анализе используемых команд и операндов, развитой регистровой архитектуре, *отказе от микрокода и отказе от использования сложных команд в пользу нескольких простых*.

Идея создания RISC процессоров пришла после того, как в 1970-х годах исследователи из IBM обнаружили, что многие функциональные особенности процессоров игнорируются программистами. Этот эффект лишь отчасти объяснялся сложностью компиляторов, которые могли использовать лишь часть из набора команд процессора, и тем, что некоторые сложные операции выполнялись медленнее, чем те же действия, выполняемые набором простых команд.

Разработчики *свели к минимуму набор инструкций* и количество режимов адресации памяти, создав простой и удобный для декодирования регулярный (фиксированной длины?) машинный код. В частности, из инструкций, работающих с памятью, оставлены только две – загрузки и выгрузки (архитектура Load/Store), а все сложные инструкции разбиты на ортогональные.

Удалены инструкции вроде вычисления синуса, косинуса или квадратного корня (их можно реализовать «вручную»). В нескольких ранних экспериментальных моделях предпринимались даже попытки отказаться от аппаратного умножения и деления.

Второе важное усовершенствование RISC-процессоров, целиком вытекающее из Load/Store-архитектуры, – *увеличение числа регистров общего назначения (до нескольких десятков).* Эти регистры равноправны, что *позволяет сохранять большую часть промежуточных данных именно в них, а не в стеке или оперативной памяти*. Все математические операции проводятся с данными, расположенными в регистрах, а при переключении между задачами вместо стека используется более быстрый механизм регистровых окон. Почти любой RISC-процессор обладает куда большим набором регистров, чем даже самый продвинутый CISC.

Достижения архитектуры, а именно *резко уменьшившаяся сложность процессора* и сопутствующее *увеличение тактовой частоты и ускорение исполнения инструкций*, *уравновешивались возросшими размерами программ (на 30%) и сильно упавшей их вычислительной плотностью* (средним количеством вычислений на единицу длины машинного кода). Но в то время, когда начались разработки RISC- архитектуры, в процессорах появился конвейер, реализация которого в рамках RISC архитектуры оказалась намного проще, чем в CISC. Это и вывело архитектуру в лидеры по производительности.

В настоящее время многие архитектуры процессоров являются RISC-подобными, к примеру, ARM, DEC Alpha, SPARC, AVR, MIPS, POWER и PowerPC. Процессоры архитектуры x86, начиная с Intel486DX, являются CISC-процессорами с RISC-ядром, преобразующими CISC-инструкции в набор внутренних инструкций RISC.

Архитектура RISC позволила существенно упростить структуру процессора и получить большую производительность, чем CISC, поэтому с 1990-х гг. почти все процессоры являются RISC или RISC-подобными (Power), либо это CISC-процессоры с RISC-ядром (x86).

Многоядерность подразумевает использование несколько процессорных ядер в одном корпусе (на одном или нескольких кристаллах). Первой двухядерность применила IBM в процессорах Power4 (2001). На данный момент массово доступны процессоры с несколькими ядрами. В 2006 Intel продемонстрировала прототип 80-ядерного процессора.

Кэширование – это использование быстродействующей памяти (кэш-памяти) для хранения копий блоков информации из устройств памяти, вероятность обращения к которым в ближайшее время велика. Идея кэширования связана с иерархией запоминающих устройств, описанной еще фон Нейманом, но непосредственно идея была осуществлена позднее, в 1960-х. Сам термин «кэш-память» появился в 1967 г. как обозначение высокоскоростного

буфера в компьютерах System/360. *С появлением микропроцессоров кэширование оперативной памяти на кристалле процессора стало одним из самых простых архитектурных способов повышения производительности.*

В **векторных процессорах** операндами команд могут выступать упорядоченные массивы данных – векторы. Отличается от скалярных процессоров, которые могут работать только с одним операндом в единицу времени. Идея векторной обработки появилась в начале 1960-х в корпорации Westinghouse Electric, планировавшей существенно *увеличить математическую производительность путем использования множества простых математических сопроцессоров.* Сопроцессоры должны были запускаться одной командой, поступающей на центральный процессор, и обрабатывать собственные данные. Затем идею попытались реализовать в проекте многопроцессорного ILLIAC IV (1966–1976), который сочли провалившимся, хотя компьютер оказался самым быстрым в мире.

Первыми успешными реализациями архитектуры считаются TI Advanced Scientific Computer (1973) и CDC STAR-100 (1974), но известность пришла с выпуском суперкомпьютера Cray-1 (1976). Поначалу векторные процессоры были основой суперкомпьютеров, но в 1990-е гг. они стали вытесняться массовыми процессорами, которые, в свою очередь, стали получать векторные расширения (такие как MMX и SSE). Позднее персональные компьютеры обзавелись векторными процессорами в составе графических ускорителей и видеокарт.

Для иллюстрации разницы в работе векторного и скалярного процессора, рассмотрим простой пример попарного сложения двух наборов по 10 чисел. При "обычном" программировании используется цикл, который берёт пары чисел последовательно, и складывает их:

```
повторить цикл 10 раз
  прочитать следующую инструкцию и декодировать
  получить первое слагаемое
  получить второе слагаемое
  сложить
  сохранить результат
конец цикла
```

Для векторного процессора алгоритм будет значительно отличаться:

```
прочитать следующую инструкцию и декодировать
получить 10 первых слагаемых
получить 10 вторых слагаемых
сложить
сохранить результат
```

Таким образом, математические операции выполняются гораздо быстрее, основным ограничивающим фактором становится время, необходимое для извлечения данных из памяти.

RISC-процессоры характеризуются следующими особенностями:

1. Из них удалены сложные (типа двоичного умножения) и редко используемые инструкции.
2. Все инструкции имеют одну длину. При этом уменьшается сложность устройства управления процессора и увеличивается скорость дешифрации команд.
3. Отсутствуют инструкции, работающие с памятью напрямую (типа команд "память - память", "регистр - память"). Возможна только загрузка данных из памяти в регистр и наоборот, из регистра в память. Соответственно на порядок увеличивается число регистров.
4. Отсутствуют операции работы со стеком.
5. Возможно использования конвейера и параллельных вычислений. АЛУ, например, одновременно может работать с 2-мя 32-х разрядными, 4-мя 16-ти разрядными, и 8-мью 8-ми разрядными числами. Смысл же конвейера – в накоплении последовательно выполняемых команд программы (т.н. линейных участков) в буфере для их ускоренного дешифрования и выполнения.
6. Почти все операции осуществляются за один такт микропроцессора.
7. Благодаря этим нововведениям тактовая частота RISC-процессоров (при прочих равных условиях) выше.

Более того, в RISC-микропроцессорах появилась возможность работы разных его составляющих на разных тактовых частотах. Например, из-за того, что содержимое памяти обычно дублируется в кэше, частоту работы АЛУ, регистров и дешифратора команд можно повысить, а частоту синхронизации пересылки между кэшем и памятью, предвыборки команд можно уменьшить. Поэтому при указании тактовой частоты процессора выбирают его максимальную частоту.