**Реферат на тему:** «Классификация архитектур ЭВМ»

**студента** Бондика Сергея **группы** 21-1

# Содержание

[**Содержание 2**](#_Toc177937320)

[**Введение 2**](#_Toc177937321)

[**Классификация архитектур 2**](#_Toc177937322)

[**По типу применяемого процессора 3**](#_Toc177937323)

[**CISC 3**](#_Toc177937324)

[**RISC 3**](#_Toc177937325)

[**MISC 4**](#_Toc177937326)

[**VLIW 4**](#_Toc177937327)

[**По принципу разделения памяти 6**](#_Toc177937328)

[**Гарвардская архитектура 6**](#_Toc177937329)

[**Фон Неймовская архитектура 6**](#_Toc177937330)

# Введение

**Архитектура компьютера** – это его устройство и принципы взаимодействия его основных элементов – логических узлов, среди которых основными являются:

* *процессор*
* *внутренняя память (основная и оперативная)*
* *внешняя память*
* *устройства ввода-вывода информации (периферийные)*

# Классификация архитектур

*По типу применяемого процессора:*

1. CISC — архитектура с полным набором команд;
2. RISC — архитектура с сокращённым набором команд;
3. MISC — архитектура с минимальным набором команд;
4. VLIW — архитектура с длинной машинной командой, в которой указывается параллельность выполнения вычислений.

*По принципу разделения памяти:*

1. Гарвардская архитектура — разделение памяти программ и памяти данных;
2. Фон Неймановская архитектура — совместное хранение программ и данных.

# По типу применяемого процессора

CISC (англ. complex instruction set computing или complexinstruction set computer) — тип процессорной архитектуры, который характеризуется следующими свойствами:

1. Длина команды произвольна (в отличие от RISC-архитектуры, в которой длина команды зафиксирована, например, 32 бита).
2. Арифметические действия кодируются в одной команде.
3. Небольшое число регистров, каждый из которых выполняет строго определённую функцию.

**Типичными представителями CISC-архитектуры являются:**

* *процессоры на основе команд x86,*
* *процессоры Motorola MC680x0,*
* *процессоры Zilog Z80,*
* *процессоры мейнфреймов zSeries.*

RISC- архитектура предполагает реализацию в ЭВМ сокращенного набора простейших, но часто употребляемых команд. Это позволяет упростить аппаратные средства процессора и получить возможность повысить его быстродействие. При использовании RISC-архитектуры выбор набора команд и структуры процессора направлены на то, чтобы команды набора выполнялись за один машинный цикл процессора. Выполнение более сложных, но редко встречаемых операций обеспечивают подпрограммы.

В RISC-ЭВМ машинным циклом называется время, в течение которого производится выборка двух операндов из регистров, выполнение операции в АЛУ и запоминание результатов в регистре. Большинство команд в RISC являются быстрыми командами типа "регистр-регистр" и выполняются без обращения к ОП. Для того, чтобы это было возможно, процессор должен содержать достаточно большое число общих регистров.

Таким образом, ЭВМ RISC-архитектуры имеют ряд характерных особенностей:

* сокращенный набор команд (обычно не более 50-100);
* небольшое число (обычно 2-3) простых способов адресации;
* небольшое число простых форматов команд с фиксированными размерами и функциональным назначением полей.

Все это упрощает УУ процессора и позволяет обходиться без микропрограм­много уровня управления и управляющей памяти, т.е. УУ может быть выполнено на быстродействующей жесткой логике.

MISC (Minimal Instruction Set Computer) — архитектура для проектирования процессора, которая отличается наилучшей эффективностью и простотой в сравнении с CISC и RISC.

Может содержать в себе блок RISC, обрабатывающий от 10 базовых команд (+, —, /, \*, if, else & etc), из которых формируются более сложные операции над значениями методом ветвления полученных результатов в ПЗУ.

С точки зрения быстродействия, время выполнения инструкции, скорость записи и передачи данных в память сократилось бы в разы, так как не нужно было бы ожидать, пока заполнится и очистится конвейер, а выполнять всё «потоково» без задержек.

Причиной, по которой данная архитектура не стала популярной в компьютерных технологиях, является сложность написания программ под различные процессоры.

VLIW - архитектуру можно рассматривать как статическую суперскалярную архитектуру. Имеется в виду, что распараллеливание кода производится на этапе компиляции, а не динамически во время исполнения. То, что в выполняемой сверх-длинной команде исключена возможность конфликтов, позволяет предельно упростить аппаратуру VLIW-процессора и, как следствие, добиться более высокого быстродействия.

В качестве простых команд, образующих сверх длинную, обычно используются команды RISC-типа. Максимальное число полей в сверхдлинной команде равно числу вычислительных устройств и обычно колеблется в диапазоне от 3 до 20. Все вычислительные устройства имеют доступ к данным, хранящимся в едином много портовом регистровом файле. Отсутствие сложных аппаратных механизмов, характерных для супер скалярных процессоров (предсказание переходов, внеочередное исполнение и т. д.), дает значительный выигрыш в быстродействии и возможность более эффективно использовать площадь кристалла. Подавляющее большинство цифровых сигнальных процессоров и мультимедийных процессоров с производительностью более 1 млрд операций/с базируется на VLIW-архитектуре. Серьезная проблема VLIW - усложнение регистрового файла и связей этого файла с вычислительными устройствами.

Преимущества. Использование компилятора позволяет устранить зависимости между командами до того, как они будут реально выполняться, в отличие от супер скалярных процессоров, где такие зависимости приходится обнаруживать и устранять "на лету". Отсутствие зависимостей между командами в коде, сформированном компилятором, ведет к упрощению аппаратных средств процессора и за счет этого к существенному подъему его быстродействия. Наличие множества функциональных блоков дает возможность выполнять несколько команд параллельно.

Недостатки. Требуется новое поколение компиляторов, способных проанализировать программу, найти в ней независимые команды, связать такие команды в строки длиной от 256 до 1024 бит, обеспечить их параллельное выполнение. Компилятор должен учитывать конкретные детали аппаратных средств. При определенных ситуациях программа оказывается недостаточно гибкой.

# По принципу разделения памяти

Гарвардская архитектура в информатике —  это архитектура процессора, которая использует для повышения производительности две различные памяти (память команд и память данных), а также раздельные шины адреса и данных для доступа к ним.

Благодаря независимости шин, команда и её операнд могут быть считаны одновременно за один машинный такт. Это уменьшает число тактов, требуемых для выполнения машинной команды, и вдвое увеличивает пропускную способность памяти.

Недостаток гарвардской архитектуры — необходимость большего числа ножек (выводов) у микропроцессора. Поэтому она используется главным образом во встраиваемых процессорах и микроконтроллерах, где один из типов памяти (или оба) является внутренним.

Фон Неймовская архитектура

Идея программного управления вычислительным процессом была существенно развита немецким математиком **Джоном фон Нейманом** (эмигрировавшим из Европы в США во время войны). Он обратился к проблеме вычислительных машин в 1943 году, а в 1945 году сформулировал принцип хранимой в памяти программы и ряд других необходимых подходов и методов для создания ЭВМ.

Его научный доклад об идеях и принципах построения программно-управляемых электронных машин представлял по сути дела теоретическую базу принципиально нового класса вычислительной техники. Данная концепция приобрела мировую известность и одобрение, прежде всего за то, что там были изложены конкретные пути создания стратегически требуемых вычислителей, сочетающих необходимое быстродействие, простоту изготовления, реальную работоспособность и приемлемую надежность.

Именно по предложенной в докладе методологии осуществлялись разработки и создание большинства первых ЭВМ. За это концепция ученого получила обобщающее название - «фон-неймановская» архитектура.

Следует отметить, что основные идеи такой архитектуры, несмотря на их первоначальную простоту, тривиальность, принципиальные ограничения (которые, кстати говоря, выявились на качественном уровне гораздо позднее - напомним «провал» пятого поколения), имели самое широкое распространение в области разработки ЭВМ. Продолжалось это на протяжении четырех поколений ЭВМ.

Лишь только в современный период начали осуществляться идеи создания машин, которые по своему построению отличаются от данной архитектуры (например, нейрокомпьютеры и другие архитектуры, о чем мы поговорим ниже). Рассмотрим суть «фон-неймановской» концепции.