Архитектура ЭВМ

Архитектура ЭВМ — концептуальная структура вычислительной машины, определяющая проведение обработки информации и включающая методы преобразования информации в данные и принципы взаимодействия технических средств и программного обеспечения.

Архитектура – это наиболее общие принципы построения ЭВМ, реализующие программное управление работой и взаимодействием основных ее функциональных узлов.

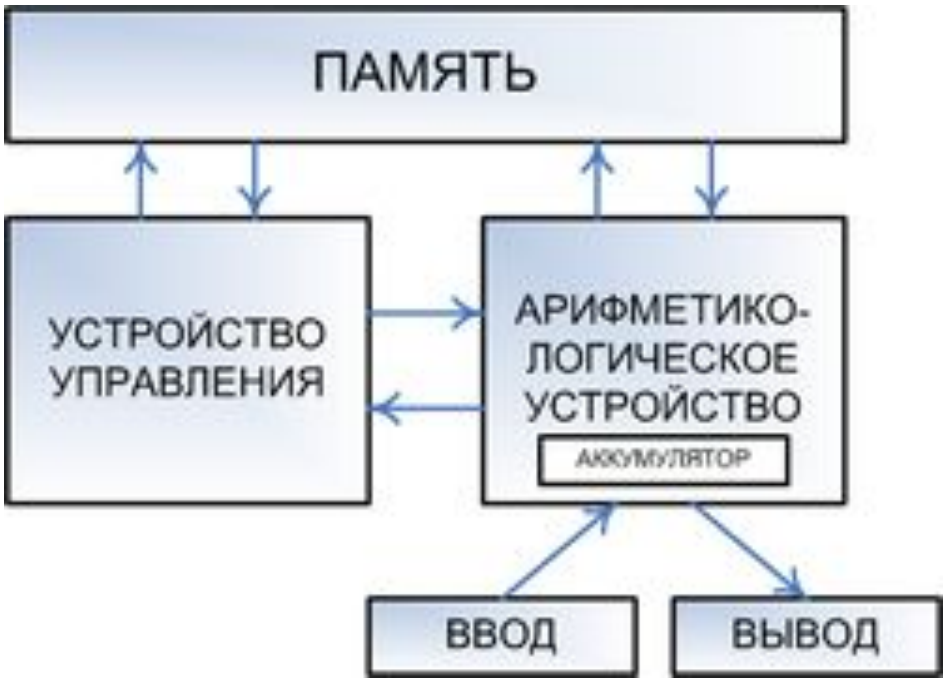
Архитектура фон Неймана

Первые компьютерные системы имели заданный набор программ

Изменение встроенной программы требовало практически полной их переделки, что требовало огромного объёма ручной работы по подготовке новой документации, перекоммутации и перестройки блоков и устройств и т. п.

Архитектура фон Неймана

В 1946 г. Был предложен принцип совместного хранения программ и данных в памяти компьютера. При этом память физически отделялась от процессора.



Архитектура фон Неймана

Принципы фон Неймана:

1. **Принцип двоичного кодирования.**
2. **Принцип однородности памяти.**
3. **Принцип адресуемости памяти.**
4. **Принцип последовательного программного управления.**
5. **Принцип жесткости архитектуры**

Архитектура фон Неймана

Принцип двоичного кодирования.

Для представления данных и команд используется двоичная система счисления.

Этим обеспечивалась простота технической реализации, простота выполнения арифметических и логических операций

Архитектура фон Неймана

Принцип однородности памяти.

Программы и данные хранятся в одной и той же памяти. Над командами можно выполнять такие же действия, как и над данными.

Это позволяет легко изменять программы для ЭВМ.

Архитектура фон Неймана

Принцип адресуемости памяти.

Структурно основная память состоит из пронумерованных ячеек, процессору в произвольный момент времени доступна любая ячейка.

Архитектура фон Неймана

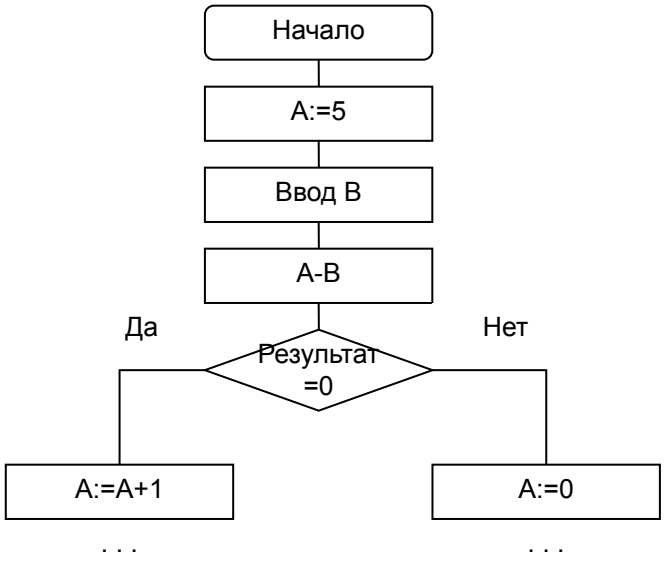
Принцип последовательного программного управления.

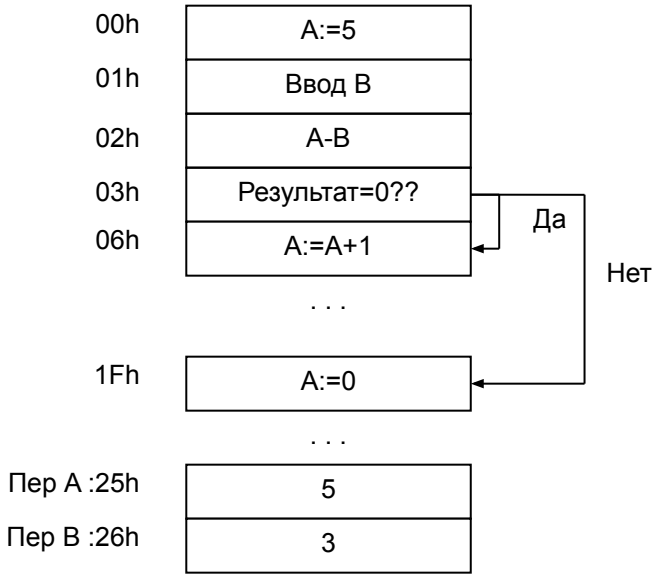
Все команды располагаются в памяти и выполняются последовательно, одна после завершения другой, в последовательности, определяемой программой.

В зависимости от результатов работы предыдущей команды линейность может нарушаться (команды условного перехода).

Архитектура фон Неймана

Принцип последовательного программного управления.





Архитектура фон Неймана

Принцип жесткости архитектуры

Неизменяемость в процессе работы топологии, архитектуры, списка команд.

Позволяет повторное использование программ, использование одних и тех же программ на разных ЭВМ.

Системы счисления

Десятичная – наиболее удобна для понимания человеком.

Двоичная – наиболее просто реализуется в ЭВМ.

Восьмеричная – удобна, если используются числа, имеющие количество двоичных разрядов, кратное трем.

Шестнадцатиричная – кратное четырем.

Системы счисления

Пример:

Права на файл в ОС UNIX

1-й бит – чтение

2-й бит – запись

3-й бит – выполнение

7(111) – все права

5(101) – чтение и выполнение

4(100) – чтение

0(000) – нет прав

Системы счисления

Пример:

Адреса памяти в ОС Windows

10010010011110001111000011101010 (2)

9278F0EA (16)

2457399530 (10)

Представление данных

Целые беззнаковые типы

Все разряды ячейки отводятся для представления числа

1 байт=8 бит – 0..255

2 байта=16 бит – 0..65535

4 байта=32 бита – 0.. 4294967295

300 - 00000001 00101100

1000000 - 00000000 00001111 01000010 01000000

Представление данных

Целые со знаком

- **Прямой код числа. Старший (левый) бит отводится под знак.**

  1 байт – -127..127

  2 байта – -32767.. 32767

  4 байта – -2147483647.. 2147483647

-3 10000011

+3 00000011

Представление данных

Прямой код в ЭВМ не используется из-за громоздкости операции сложения/вычитания

10011111		**01000101**
+	**===**	**-**
01000101		**00011111**

Представление данных

Целые со знаком

2.Дополнительный код.

1 байт – -128..127

2 байта – -32768.. 32767

4 байта – -2147483648.. 2147483647

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

- Положительное число. Записывается так же, как в прямом коде.

5 00000101

3 00000011

128 0000000010000000

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

2. Отрицательное число.
    1. Записывается по модулю в прямом коде.
    2. Все биты инвертируются – нули заменяются единицами и наоборот.
    3. К полученному прибавляется 1

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

Примеры:

-35

1) 00100011
2) 11011100
3) **11011101**

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

Примеры:

-127

1) 01111111
2) 10000000
3) **10000001**

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

Примеры:

-128

1) 10000000
2) 01111111
3) **10000000**

Представление данных

Целые со знаком

Алгоритм преобразования в дополнительный код.

Примеры:

-1

1) 00000001
2) 11111110
3) **11111111**

Представление данных

Целые со знаком

Сложение чисел в дополнительном коде.

35+(-1) 00100011

\+

11111111

00100010

Представление данных

Целые со знаком

Сложение чисел в дополнительном коде.

-35+(-35) 11011101

\+

11011101

10111010 (-70 в доп. коде)

Представление данных

Целые со знаком

Перенос и переполнение

Получено	Перенос в знаковый разряд	Перенос за пределы разрядной сетки	Переполнение
00+00=00	-	-	-
00+01=01	-	-	-
00+11=11	-	-	-
00+10=10	-	-	-
01+01=10	+	-	!
01+11=100	+	+	-
01+10=11	-	-	-
11+11=110	+	+	-
11+10=101	-	+	!
10+10=100	-	+	!

Представление данных

Целые со знаком

Сложение чисел в дополнительном коде.

127+1 01111111

\+

<u>00000001</u>

10000000 (переполнение)

Представление данных

Целые со знаком

Сложение чисел в дополнительном коде.

(-128)+(-1) 10000000

\+

11111111

01111111 (переполнение)