Логические основы информатики

Высказыванием называется любое предложение естественного языка, которое мы можем оценить как *истинное* или *ложное*. Например, любое повествовательное предложение русского (или английского) языка является высказыванием, а вопросительное – нет. Рассмотрим несколько примеров высказываний.

Пример 1.

1. «Дважды два - четыре».
2. «Завтра произойдет сражение».
3. «**Если** число оканчивается на пять, **то** оно делится на пять».
4. «Студент сдает экзамен **и** идет дождь»
5. «Летом тепло **или** трижды три - девять»

Первое высказывание, очевидно является истинным. Истинность второго высказывания зависит от того, что произойдет завтра. Третье, четвертое и пятое состоят из двух частей: третье – содержит конструкцию «если…,то…»; части четвертого и пятого связанны союзами «и» и «или», соответственно. Истинность этих высказываний зависит от истинности или ложности частей.

Высказывания мы будем обозначать большими или малыми латинскими буквами, возможно с индексами. Используя логические связки, из *простых высказываний* (то есть, не содержащих логических связок) можно получать новые, сложные высказывания. Истинность или ложность получаемых высказываний определяется истинностью или ложностью составляющих их простых высказываний. При этом содержание высказываний не анализируется. Так высказывание «Летом тепло или трижды три - девять» будет истинным, так как входящее в него высказывание «трижды три - девять» истинно при любой погоде.

Основными логическими связками являются: *конъюнкция* (логическое «и»), *дизъюнкция* (логическое «или»), *импликация* («если..., то…»), *тождество* и *отрицание*. Истинностные значения высказываний, построенных с использованием *логических связок* определяются из *таблиц истинности*. В таблицах истинности в левых столбцах указываются все возможные значения простых высказываний, входящих в высказывание; в крайнем правом столбце указываются соответствующие значения высказывания; значение «*истина*» обозначается буквой «и», значение «*ложь*» - буквой «л».

Отрицание обозначается символами «¬» или «$\bar{}$». Отрицание высказывания А, которое записывается как ¬А или $\bar{A}$ (читается «не А»), определится из таблицы (Таб.1):

А	¬А
и	л
л	и

Таб.1

Из таблицы видно, что если высказывание А *истинно*, то высказывание ¬А, являющееся *отрицанием* высказывания А *ложно* и, наоборот, если А *ложно*, то ¬А *истинно*.

Конъюнкция обозначается символом «∧». Для высказываний A и B конъюнкция запишется как A∧B (читается: «*A и B*»). Конъюнкция определяется таблицей истинности (Таб.2):

A	B	A ∧ B
и	и	и
и	л	л
л	и	л
л	л	л

Таб.2

Из таблицы видно, что высказывание, являющееся конъюнкцией двух высказываний истинно тогда и только тогда, когда истинны оба высказывания, определяющие конъюнкцию. Если же хотя бы одно из высказываний, определяющих конъюнкцию ложно, то конъюнкция ложна.

Пример 2.

Пусть C = «сегодня зимний день и эта книга интересна». A = «сегодня зимний день», B = «эта книга интересна». Тогда C = A∧B. Это высказывание истинно если и только если истинны высказывания A и B, то есть истинно «сегодня зимний день» и «эта книга интересна».

Дизъюнкция обозначается символом «∨». Дизъюнкция высказываний A и B запишется как A∨B (читается: «*A или B* ») и определяется таблицей истинности(Таб.3):

A	B	A ∨ B
и	и	и
и	л	и
л	и	и
л	л	л

Таб.3

Из таблицы видно, что высказывание, являющееся дизъюнкцией двух высказываний истинно, когда истинно хотя бы одно из высказываний, определяющих дизъюнкцию. Дизъюнкция является ложной тогда и только тогда, когда ложны оба высказывания, определяющие дизъюнкцию.

Пример 3.

Пусть D высказывание «он сдаст сессию или эта формула не верна»; A - «он сдаст сессию»; B - «эта формула не верна». Тогда D = A ∨ B. Это высказывание ложно, если и только если ложно высказывание «он сдаст сессию» и «эта формула не верна».

Импликация обозначается символом «→»: A→B (читается: «*Если A, то B*») и определяется таблицей истинности:

A	B	A → B
и	и	и
и	л	л
л	и	и
л	л	и

Таб.4

Первый член в импликации(посылка) A называется *антецедентом*, второй член (заключение) B *консеквентом*.

Пример 4.

Рассмотрим высказывание D = «Если натуральное число оканчивается на 2, то это число - четное». A = «натуральное число оканчивается на 2».B = «это число - четное». D = A → B.

Основные логические операции можно наглядно интерпретировать в теории множеств. Рассмотрим некоторое универсальное множество *I*. Оно будет содержать все множества, о которых мы говорим, в качестве подмножеств. Если $A, B, C \subset I$, то высказываниям* A, B, C будут сопоставляться соответствующие множества и изображаться в виде кругов – кругов Эйлера. Так, отрицанию соответствует дополнение, дизъюнкции – объединение, конъюнкции – пересечение кругов, импликации – включение одного множества в другое.

Отрицание :

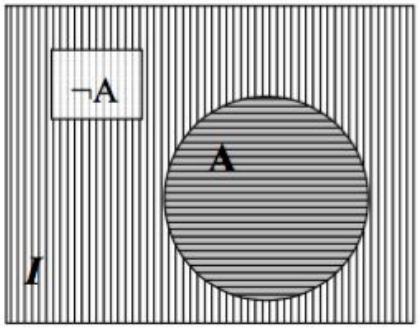


Рис. 1

Дизъюнкция:

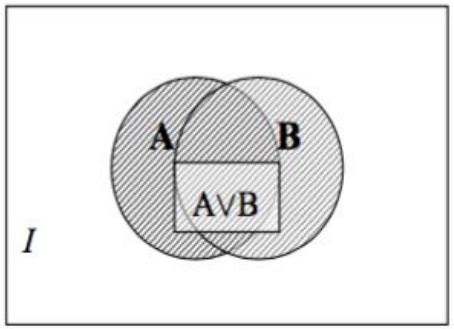


Рис. 2

Конъюнкция:

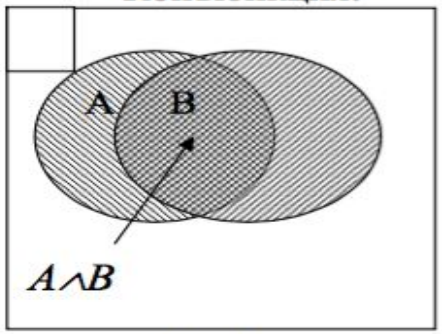


Рис.3

Импликация:

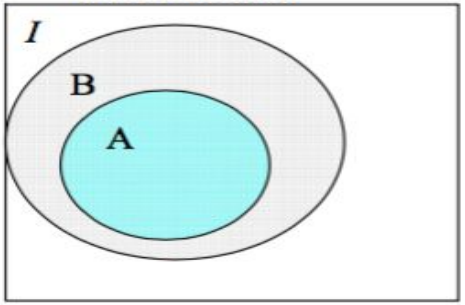


Рис.4

Очень важным является понятие *формулы*.

Определение

1). Простые высказывания являются формулами;

2). Если A и B-формулы, то $(A \vee B)$, $(A \wedge B)$, $(A \to B)$, $\neg A$ - формулы.

3). Формулами являются те, и только те выражения, которые определяются пунктами 1) и 2).

Таким образом, в качестве функционального выражения высказывания мы будем использовать понятие формулы. Значение формулы определяется таблицей истинности. Для логических связок определяется следующий порядок приоритетов:

1. «$\neg$»;
2. «$\wedge$», «$\vee$»;
3. «$\to$».

Если возникает возможность неоднозначного прочтения формулы, скобки сохраняются, если – нет, то лишние скобки снимаются.

Пример 5.

Требуется построить таблицу истинности для следующей формулы: $(A \wedge \neg B) \vee (\neg A \wedge B)$ (Таб 6)*.

A	B	$(A \wedge \neg B) \vee (\neg A \wedge B)$		
И	И	л	**Л**	л
И	Л	и	**И**	и
Л	И	л	**И**	и
Л	Л	л	**Л**	л

Таб.6

Так же можно доказать, что $(A \to B) \equiv (\neg A \vee B)$ (упражнение). Таким образом получится три пары тождественно равных формул:

1. $\neg(A \vee B) \equiv (\neg A \wedge \neg B)$;
2. $\neg(A \wedge B) \equiv (\neg A \vee \neg B)$;
3. $(A \to B) \equiv (\neg A \vee B)$.

Эти равенства показывают, что для того чтобы построить все формулы достаточно иметь конъюнкцию и отрицание или дизъюнкцию и отрицание, или импликацию и отрицание. Однако такое представление не всегда будет простым.

Свойства

1. $\neg(A \vee B) \equiv (\neg A \wedge \neg B)$;
2. $\neg(A \wedge B) \equiv (\neg A \vee \neg B)$;
3. $(A \rightarrow B) \equiv (\neg A \vee B)$.
4. $A \vee \textbf{и} \equiv \textbf{и}$, $A \vee \textbf{л} \equiv \textbf{л}$;
5. $A \wedge \textbf{и} \equiv A$, $A \wedge \textbf{л} \equiv \textbf{л}$;
6. $A \vee \neg A \equiv \textbf{и}$, $A \wedge \neg A \equiv \textbf{л}$;
7. $A \equiv \neg\neg A$;
8. $A \rightarrow A \equiv \textbf{и}$;
9. $A \vee B \equiv B \vee A$, $A \wedge B \equiv B \wedge A$;
10. $(A \vee B) \vee C \equiv A \vee (B \vee C)$,
11. $(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$.

Релейно-контактные схемы

Электрическая цепь состоит из проводника и переключателей. Переключатели могут находиться в двух состояниях: замкнут и разомкнут. Сопоставим переключателям высказывания. Если высказывание P = и, то соответствующий переключатель замкнут, то есть проводит ток и если P= л, то переключатель разомкнут, то есть ток не проходит(см рис 1).

Рис.1

Конъюнкция P∧Q реализуется цепью с последовательным расположением переключателей (рис.2): ток проходит через цепь если и только если оба переключателя замкнуты, P∧Q = **и** ⇔ P = **и** и Q = **и**. Дизъюнкция P∨Q реализуется цепью с параллельным расположением переключателей (рис. 3): ток проходит через цепь если хотя бы один переключатель замкнут, P∨Q = **и** ⇔ P = **и** или Q = **и**.

Рис. 2

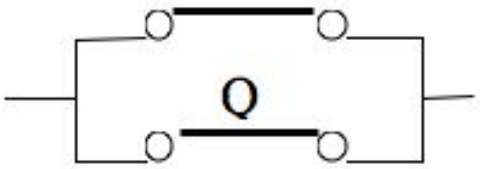


Рис.3

Остается описать реализацию отрицания. Переключатели в цепи взаимодействуют, тогда может быть так, что цепь соответствующая высказываниязыванию Р замкнута или разомкнута когда другой преключатель, соответствующий высказыванию Р* разомкнут или замкнут соответственно (Рис.4), что дает реализацию отрицание Р, то есть $P^* = \overline{P}$:

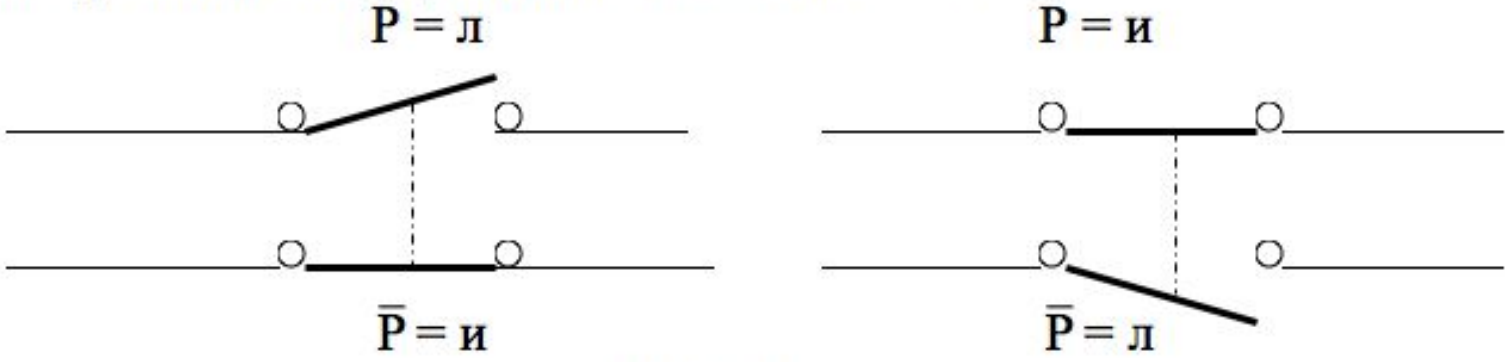
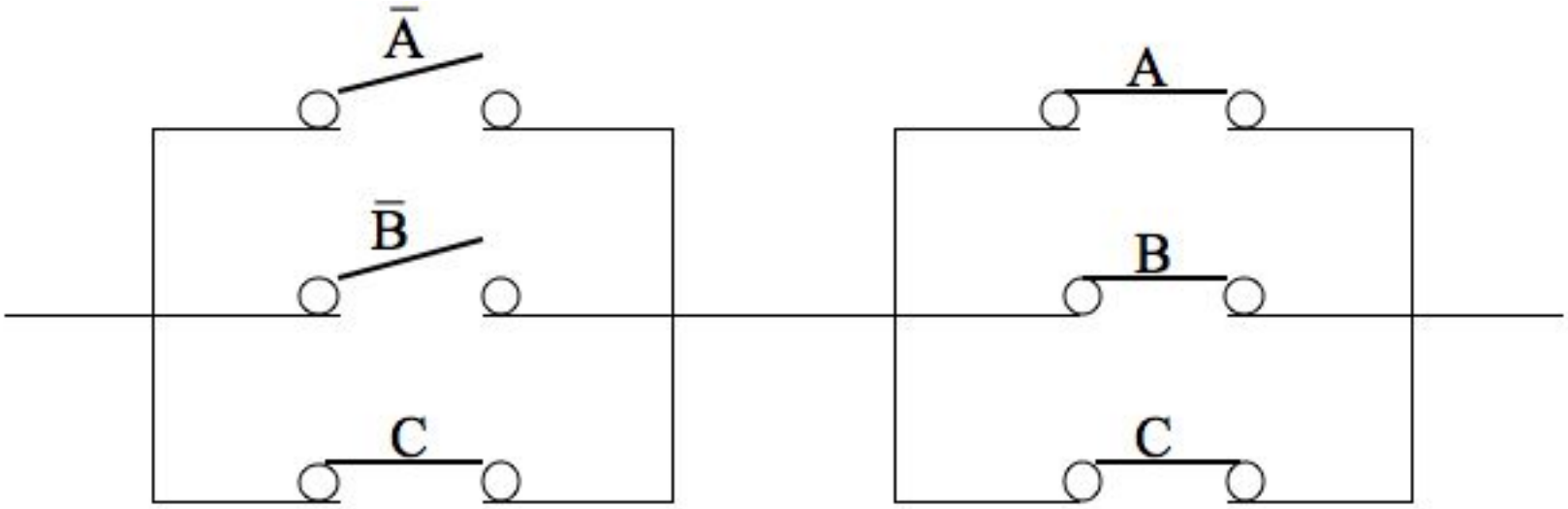


Рис. 4

Составить релейно-контактную схему для формулы

$$\left(A \to \left(\bar{B} \vee C\right)\right) \wedge \left(C \vee \left(\bar{A} \to B\right)\right).$$

Используя свойства представим формулу в виде: $\left(\bar{A} \vee \bar{B} \vee C\right) \wedge \left(A \vee B \vee C\right)$.



Можно рассматривать формулы высказываний как функции, определенные на множестве двоичных последовательностей со значениями 0 и 1, с заменой «и» на «1» и «л» на «0».

Тогда основные логические операции определятся из таблицы:

x	y	$\bar{x}$	$x \vee y$	$x \wedge y$	$x \to y$	$x \oplus y$	$x \mid y$	$x \downarrow y$
0	0	1	0	0	1	0	1	0
0	1	1	1	0	1	1	1	0
1	0	0	1	0	0	1	1	0
1	1	0	1	1	1	0	0	1

$x \oplus y$ - *сложение по модулю 2;*

$x \mid y$ – *штрих Шеффера;* $x \mid y \equiv \overline{x \wedge y}$;

$x \downarrow y$ – *стрелка Пирса;* $x \downarrow y \equiv \overline{x \vee y}$.

Пример таблицы истинности

x	y	z	$\overline{y \wedge x}$	$\bar{y} \vee z$	$\overline{y \wedge x} \rightarrow (\bar{y} \vee z)$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	1	0	1	1

Используя формулы преобразования:

$$\bar{\bar{x}} \equiv x;$$
$$\overline{x \vee y} \equiv \bar{x} \wedge \bar{y};$$
$$\overline{x \wedge y} \equiv \bar{x} \vee \bar{y};$$
$$x \to y \equiv \bar{x} \vee y\ .$$

Можно все формулы записать либо с использованием *дизъюнкции* и *отрицания* или *конъюнкции* и *отрицания*.

Пример.

$$(x \to (\bar{y} \vee z)) \wedge (z \vee (\bar{x} \to y)) \equiv (\bar{x} \vee (\bar{y} \vee z)) \wedge (z \vee (\bar{\bar{x}} \vee y)) \equiv \overline{\overline{\bar{x} \vee (\bar{y} \vee z)} \vee \overline{z \vee (x \vee y)}}$$

Для штриха Шеффера такие формулы преобразования:

$$\bar{x} \equiv x|x;$$

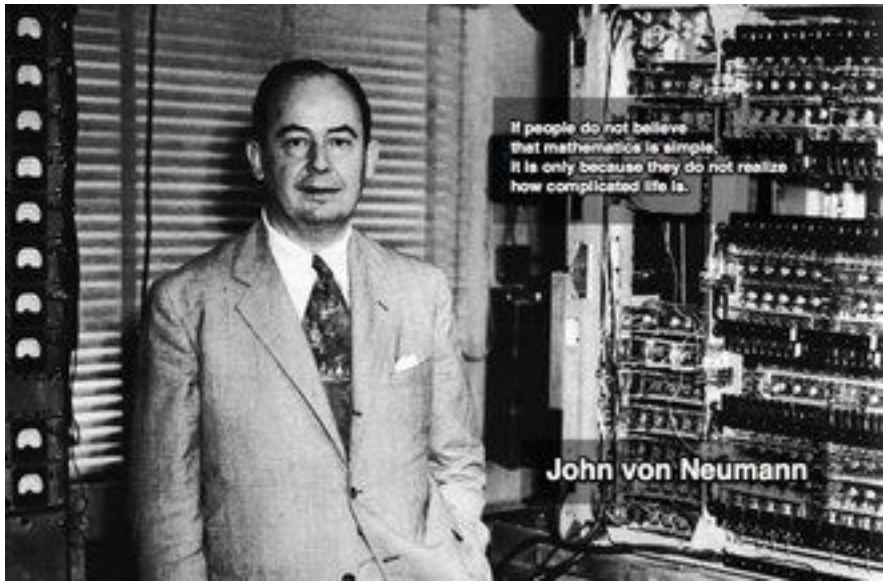
$$x \wedge y \equiv \overline{x|y} \equiv (x|y)|(x|y);$$

$$x \vee y \equiv \overline{\bar{x} \wedge \bar{y}} \equiv (x|x)\big|(y|y); x \to y \equiv \bar{x} \vee y \equiv (x|x) \vee y \equiv \big((x|x)\big|(x|x)\big)\big|(y|y).$$

Таким образом можно представить любую формулу с использованием только штриха Шеффера. Аналогичное утверждение справедливо относительно стрелки Пирса.

Общие принципы организации и работы компьютеров

В основу построения подавляющего большинства компьютеров положены следующие общие принципы, сформулированные в 1945 г. американским ученым **Джоном фон Нейманом(1903-1957)**.



1. Принцип программного управления. Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.

Выборка программы из памяти осуществляется с помощью **счетчика команд**. Этот регистр процессора **последовательно увеличивает хранимый в нем адрес очередной команды на длину команды**.

А так как команды программы расположены в памяти друг за другом, то тем самым организуется выборка цепочки команд из последовательно расположенных ячеек памяти. Если же нужно после выполнения команды перейти не к следующей, а к какой-то другой, используются команды **условного** или **безусловного переходов**, которые **заносят в счетчик команд номер ячейки памяти, содержащей следующую команду**. Выборка команд из памяти прекращается после достижения и выполнения команды *“стоп”*.

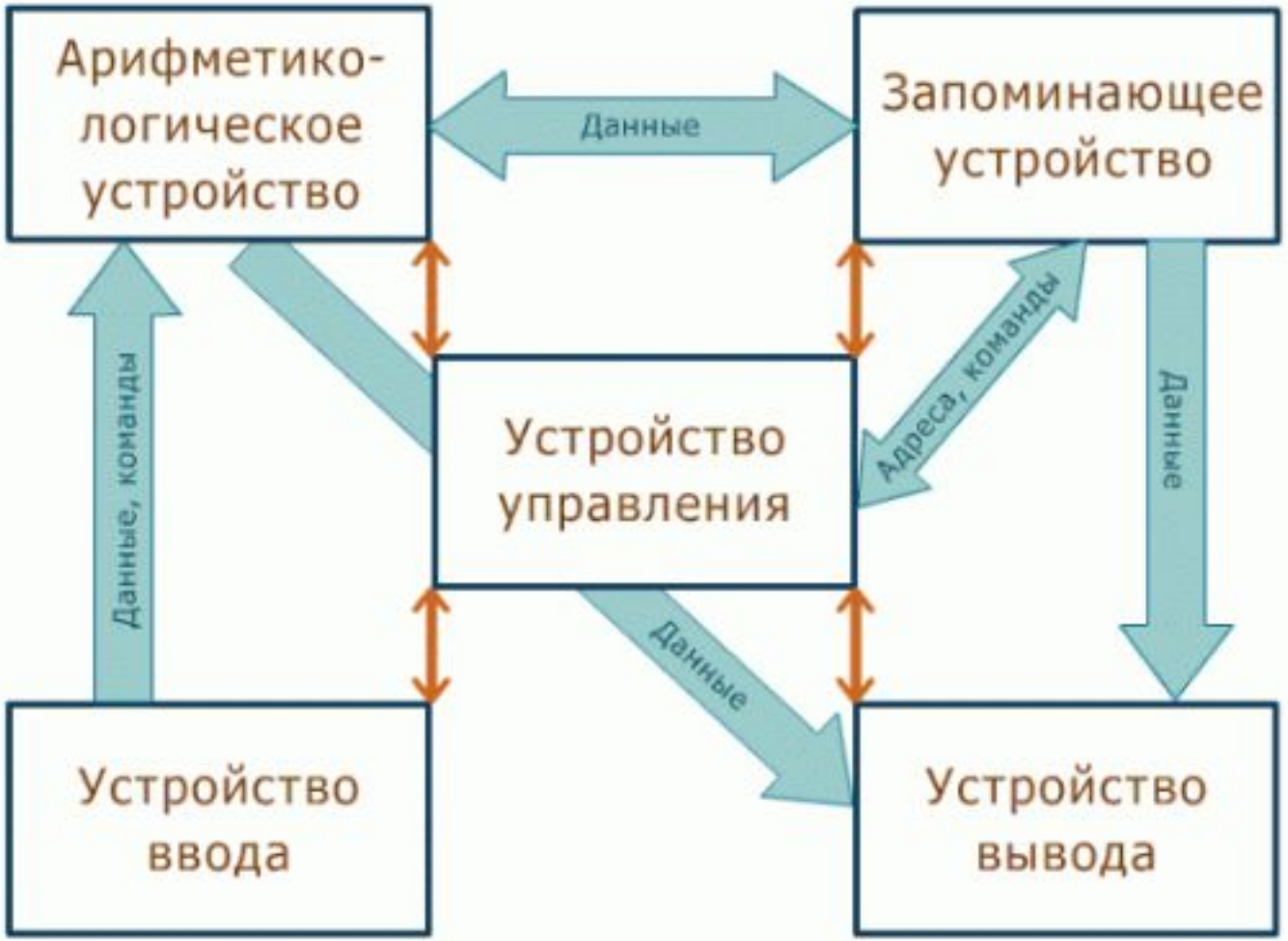
Таким образом, **процессор исполняет программу автоматически, без вмешательства человека**.

2. Принцип однородности памяти. Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке памяти — число, текст или команда. Над командами можно выполнять такие же действия, как и над данными. Это открывает целый ряд возможностей. Например, программа в процессе своего выполнения также может подвергаться переработке, что позволяет задавать в самой программе правила получения некоторых ее частей (так в программе организуется выполнение циклов и подпрограмм). Более того, команды одной программы могут быть получены как результаты исполнения другой программы. На этом принципе основаны методы трансляции — перевода текста программы с языка программирования высокого уровня на язык конкретной машины.

3. Принцип адресности. Структурно основная память состоит из перенумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка. Отсюда следует возможность давать имена областям памяти, так, чтобы к запомненным в них значениям можно было впоследствии обращаться или менять их в процессе выполнения программ с использованием присвоенных имен.

Компьютеры, построенные на этих принципах, относятся к типу фон-неймановских. Но существуют компьютеры, принципиально отличающиеся от фон-неймановских. Для них, например, может не выполняться принцип программного управления, т.е. они могут работать без “счетчика команд”, указывающего текущую выполняемую команду программы. Для обращения к какой-либо переменной, хранящейся в памяти, этим компьютерам не обязательно давать ей имя. Такие компьютеры называются не-фон-неймановскими.

Схема вычислительной машины фон Неймана



Машина фон Неймана состоит из запоминающего устройства (памяти) - ЗУ, арифметико-логического устройства - АЛУ, устройства управления – УУ, а также устройств ввода и вывода.

Программы и данные вводятся в память из устройства ввода через арифметико-логическое устройство. Все команды программы записываются в соседние ячейки памяти, а данные для обработки могут содержаться в произвольных ячейках. У любой программы последняя команда должна быть командой завершения работы.

Команда состоит из указания, какую операцию следует выполнить (из возможных операций на данном «железе») и адресов ячеек памяти, где хранятся данные, над которыми следует выполнить указанную операцию, а также адреса ячейки, куда следует записать результат (если его требуется сохранить в ЗУ).

Арифметико-логическое устройство выполняет указанные командами операции над указанными данными.

Из арифметико-логического устройства результаты выводятся в память или устройство вывода. Принципиальное различие между ЗУ и устройством вывода заключается в том, что в ЗУ данные хранятся в виде, удобном для обработки компьютером, а на устройства вывода (принтер, монитор и др.) поступают так, как удобно человеку.

УУ управляет всеми частями компьютера. От управляющего устройства на другие устройства поступают сигналы «что делать», а от других устройств УУ получает информацию об их состоянии.

Управляющее устройство содержит специальный регистр (ячейку), который называется «счетчик команд». После загрузки программы и данных в память в счетчик команд записывается адрес первой команды программы. УУ считывает из памяти содержимое ячейки памяти, адрес которой находится в счетчике команд, и помещает его в специальное устройство — «Регистр команд». УУ определяет операцию команды, «отмечает» в памяти данные, адреса которых указаны в команде, и контролирует выполнение команды. Операцию выполняет АЛУ или аппаратные средства компьютера.

В результате выполнения любой команды счетчик команд изменяется на единицу и, следовательно, указывает на следующую команду программы. Когда требуется выполнить команду, не следующую по порядку за текущей, а отстоящую от данной на какое-то количество адресов, то специальная команда перехода содержит адрес ячейки, куда требуется передать управление.