АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ ЦИФРОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Гринюк Д.А. (ауд.3-1)

hryniukda@gmail.com

hryniuk.ucoz.com

Книги

- Цифровая схемотехника и архитектура компьютера (второе издание). Дэвид М. Хэррис и Сара Л. Хэррис
- Паттерсон Д., Хеннеси Д. - Архитектура компьютеров и проектирование компьютерных систем, 4-е изд (Классика Computer Science) – 2012.
- Жмакин А.П. Архитектура ЭВМ. 2-е изд. СПб.: БХВ-Петербург, 2010.
- Орлов С.А., Цилькер Б.Я. Организация ЭВМ и систем. 2-е изд. СПб.: Питер, 2011.
- А. С. Кобайло. Арифметико-логические основы цифровых вычислительных машин и архитектура компьютера (Конспект лекций)
- Таненбаум Э. Архитектура компьютера

Рабочая программа

- **Предметами изучения** данной дисциплины являются арифметические и логические данные, методы представления и обработки данных, законы и положения алгебры логики.
- **Объектом** изучения являются информационные логические элементы, узлы и устройства ЦВМ, микропроцессоры и микропроцессорные системы.
- **Задачи** преподавания дисциплины – освоение студентами:
- – арифметических основ построения операционных узлов ЭВМ на основе правил двоичной арифметики;
- – логических основ вычислительной техники и методологии анализа и синтеза цифровых узлов и устройств на основе изучения алгебры логики;
- – схемотехнических и архитектурных особенностей построения компьютеров.

В результате освоения курса студенты должны:

- ***знать:***
- – системы счисления и методы перехода между ними;
- – способы представления информации в ЭВМ;
- – правила выполнения арифметических операций над алгебраическими числами;
- – законы и правила алгебры логики;
- – методы минимизации логических выражений;
- – методы синтеза логических схем по логическим выражениям в различных базисах;
- – устройство процессора, назначение его основных блоков и входов/выходов, механизмы управления обработкой команд;
- – систему команд, форматы команд и данных, их размещение в памяти компьютера;
- – организацию памяти компьютеров, назначение сегментов, организацию стека и буферов ввода-вывода;
- ***уметь:***
- – синтезировать операционные устройства и устройства управления с жесткой логикой в различных базисах;
- – синтезировать микропрограммные цифровые автоматы;
- – характеризовать различные варианты организации и устройства компьютеров и вычислительных систем;

- – анализировать класс решаемых задач и возможности применения конкретной архитектуры компьютера;
- – проектировать алгоритмы выполнения операций;
- – разрабатывать параллельные алгоритмы построения вычислительных систем;
- ***владеть:***
- – навыками синтеза операционных и управляющих узлов ЭВМ в различных базисах (Буля, Пирса, Шеффера);
- – навыками синтеза микропрограммных автоматов.
- – способами построения архитектур компьютеров;
- – системой команд их форматами и способами размещения в памяти;
- способами построения основных логических устройств компьютера;
- методикой логического проектирование основных вычислительных узлов компьютера;
- основами программирования на Ассемблере;
- топологиями построения вычислительных систем.

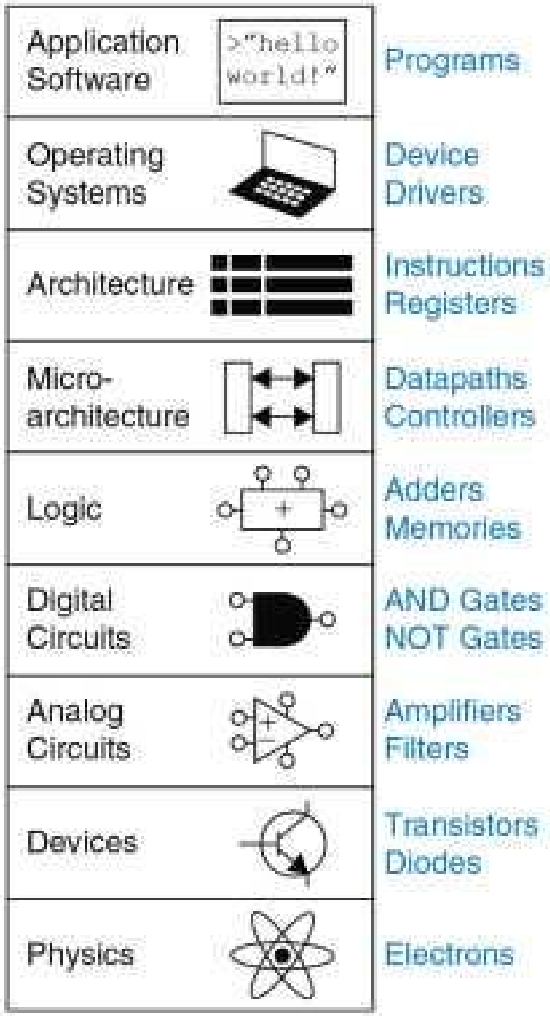
- Прогресс полупроводниковых технологий и инструментальных средств проектирования цифровых систем в 1990-2000-е годы вывел на первый план языки описания аппаратуры System Verilog и VHDL, которые практически вытеснили традиционное схемотехническое проектирование электронных устройств, включая блоки микропроцессоров. Создание сложных систем на кристалле, объединяющих несколько различных типов микропроцессоров, стало возможным только при использовании средств проектирования, моделирования и верификации ультра-больших интегральных схем, поставляемых фирмами Cadence, Synopsys и Mentor Graphics.
- Кроме того, появились доступные по цене средства моделирования и макетирования в виде конструкторских плат, использующих FPGA (Field Programmable Gate Array), называемых у нас ПЛИС
- (Программируемая логическая интегральная схема). Такие платы могут быть приобретены любым учебным центром или даже частным лицом. Используя языки описания аппаратуры и предоставленные производителем FPGA инструментальные средства от вышеупомянутых фирм, любой грамотный студент в состоянии самостоятельно спроектировать и построить сложную цифровую систему, включая микропроцессор.

- За последние тридцать лет микропроцессоры буквально изменили наш мир до неузнаваемости. Сегодняшний ноутбук обладает большей вычислительной мощностью, чем большой компьютер недавнего прошлого, занимавший целую комнату. Внутри современного автомобиля представительского класса можно обнаружить около пятидесяти микропроцессоров. Именно прогресс в области микропроцессорной техники сделал возможным появление сотовых телефонов и Интернета, значительно продвинул вперед медицину и радикально изменил тактику и стратегию современной войны. Объем продаж мировой полупроводниковой промышленности вырос с 21 миллиарда долларов в 1985 году до 300 миллиардов долларов в 2011 году, причем микропроцессоры составили львиную долю этих продаж. И мы убеждены, что микропроцессоры важны не только с технической, экономической и социальной точек зрения, но и стали одним из самых увлекательных изобретений в истории человечества. Когда вы закончите изучение курса, вы будет знать, как спроектировать и построить ваш собственный микропроцессор, а навыки, полученные на этом пути, пригодятся вам для разработки и многих других цифровых систем.
- Предполагаем, что у вас уже есть базовые знания по теории электричества, некоторый опыт программирования и искреннее желание понять, что происходит под капотом компьютера. В первом семестре основное внимание уделяется разработке цифровых систем, то есть систем, которые используют для своей работы два уровня напряжения, представляющих единицу и нуль. Мы начнем с простейших цифровых логических элементов - вентилей (digital logic gates), которые принимают определенную комбинацию единиц и нулей на входе и трансформируют ее в другую комбинацию единиц и нулей на выходе. После этого мы с вами научимся объединять эти простейшие логические элементы в более сложные модули, такие как сумматоры и блоки памяти.

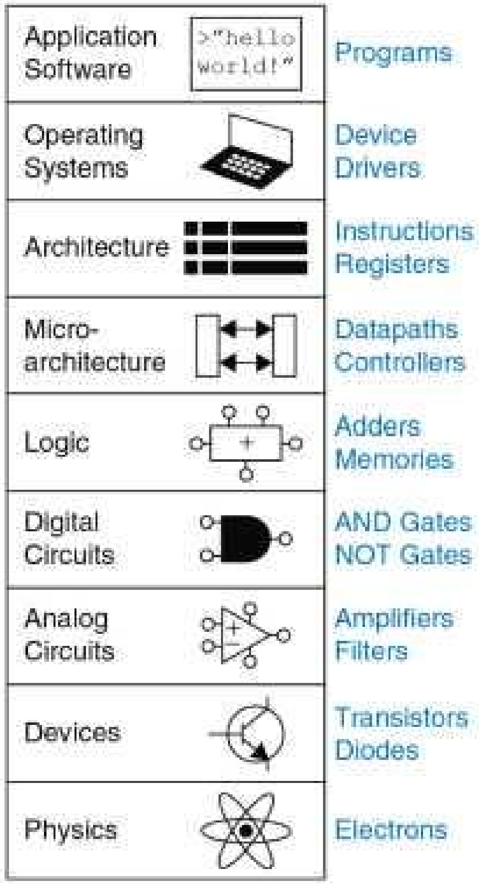
- Затем во втором семестре вы должны будете перейдем к программированию на языке ассемблера - родном языке микропроцессора. И в завершение, из кирпичиков логических элементов мы с вами соберем полноценный микропроцессор, способный выполнять ваши программы, написанные на языке ассемблера.
- ПРАВИЛА
- Огромным преимуществом цифровых систем над аналоговыми является то, что необходимые для их построения блоки чрезвычайно просты, поскольку оперируют не непрерывными сигналами, а единицами и нулями.
- Построение цифровой системы не требует запутанных математических расчетов или глубоких знаний в области физики. Вместо этого, задача, стоящая перед разработчиком цифровых устройств, заключается в том, чтобы собрать сложную работающую систему из этих простых блоков.
- Возможно, микропроцессор станет первой спроектированной вами системой, настолько сложной, что ее невозможно целиком удержать в голове. Именно поэтому одной из тем, проходящих красной нитью через эту книгу, является искусство управления сложностью системы.

ИСКУССТВО УПРАВЛЕНИЯ СЛОЖНОСТЬЮ

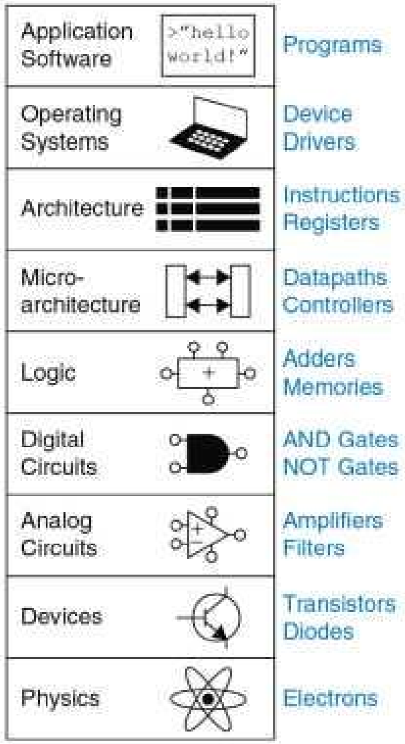
- Одной из характеристик, отличающих профессионального инженера-электронщика или программиста от дилетанта, является систематический подход к управлению сложностью многоуровневой системы. Современные цифровые системы построены из миллионов и миллиардов транзисторов. Человеческий мозг не в состоянии предсказать поведение подобных систем путем составления уравнений, описывающих движение каждого электрона в каждом транзисторе системы, и последующего решения этой системы уравнений. Для того, чтобы разработать удачный микропроцессор и не утонуть при этом в море избыточной информации, необходимо научиться управлять сложностью разрабатываемой системы.
- Критически важный принцип управления сложностью системы - ***абстракция,*** подразумевающая исключение из рассмотрения тех элементов, которые в данном конкретном случае несущественны для понимания работы этой системы. Любую систему можно рассматривать с различных уровней абстракции. Политику, участвующему в выборах, например, нет нужды учитывать все детали окружающего его мира, ему достаточно абстрактной иерархической модели страны, состоящей из населенных пунктов, областей и федеральных округов.



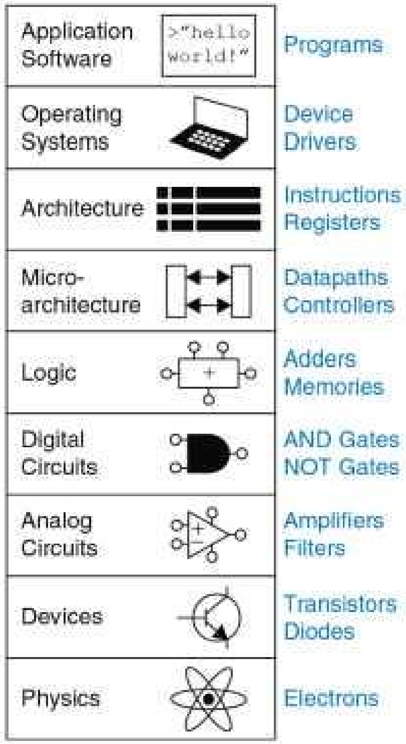
Уровни абстракции электронной вычислительной системы



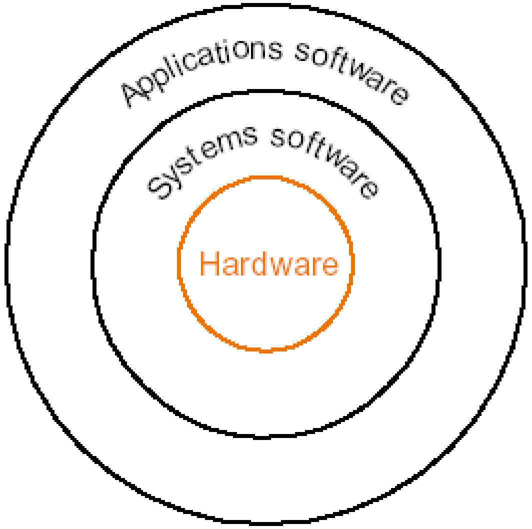
- На самом низком уровне абстракции находится физика, изучающая движение электронов. Поведение электронов описывается квантовой механикой и системой уравнений Максвелла.
- Рассматриваемая нами современная электронная система состоит из полупроводниковых устройств *(devices),* таких как транзисторы (а когда- то это были электронные лампы). Каждое такое устройство имеет четко определенные точки соединения с другими подобными устройствами. Эти точки мы будем называть *контактами* (в англоязычной литературе используется термин *terminal).* Любое электронное устройство может быть представлено абстрактной математической моделью, описывающей изменяющуюся во времени взаимозависимость тока и напряжения. Такие же изменения тока и напряжения можно наблюдать на экране осциллографа, если подключить осциллограф к контактамреального устройства. Данный подход означает, что, если рассматривать систему на уровне устройств, функции которых однозначно определены, то можно не учитывать поведение электронов внутри отдельных устройств этой системы.

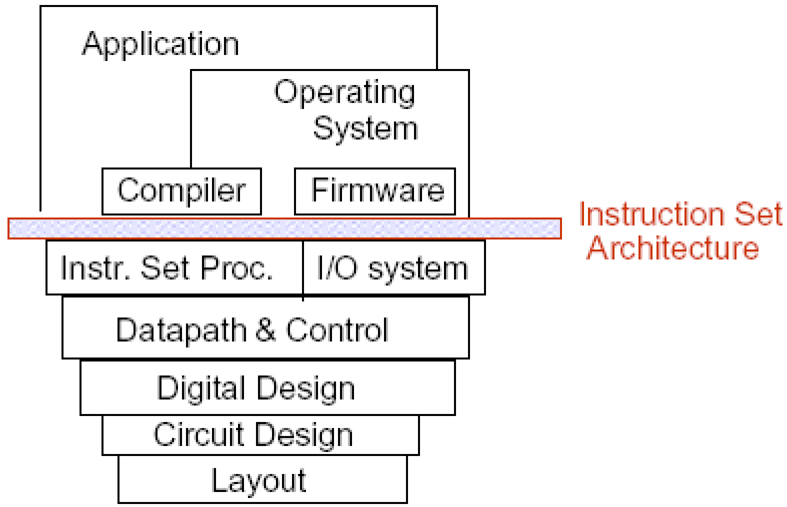


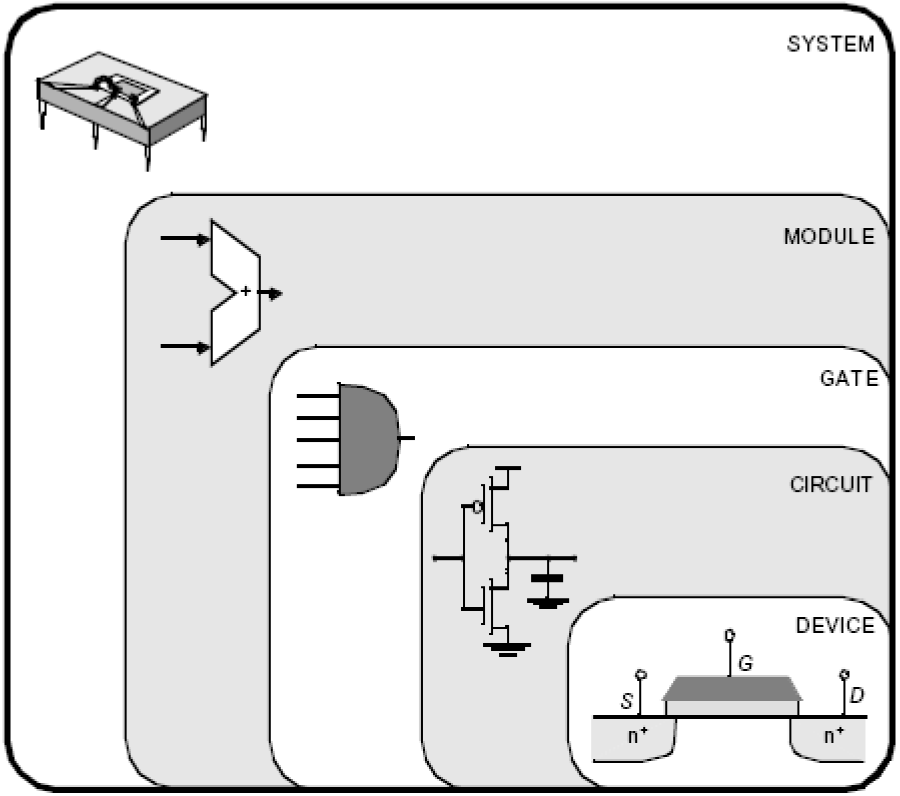
- Следующий уровень абстракции - это *аналоговые схемы (analog circuits),* в которых полупроводниковые устройства соединены таким образом, чтобы они образовывали функциональные компоненты, такие как усилители, например. Напряжение на входе и на выходе аналоговой цепи изменяется в непрерывном диапазоне.
- В отличие от аналоговых цепей, *цифровые схемы (digital circuits),* такие как логические вентили, используют два строго ограниченных дискретных уровня напряжения. Один из этих дискретных уровней - это логический нуль, другой - логическая единица. В разделах этой книги, посвященных разработке цифровых схем и устройств, мы будем использовать простейшие цифровые схемы для построения сложных цифровых модулей, таких как сумматоры и блоки памяти.



- Микроархитектурный уровень абстракции, или просто *микроархитектура (microarchitecture),* связывает логический и архитектурный уровни абстракции. Архитектурный уровень абстракции, или *архитектура (architecture)*, описывает компьютер с точки зрения программиста. Например, архитектура Intel x86, используемая
- микропроцессорами большинства персональных компьютеров *(ПК),* определяется набором инструкций и регистров (памяти для временного хранения переменных), доступным для использования программистом. Микроархитектура - это соединение простейших цифровых элементов в логические блоки, предназначенные для выполнения команд, определенных какой-то конкретной архитектурой. Отдельно взятая архитектура может быть реализована с использованием различных вариантов микроархитектур с разным соотношением цены, производительности и потребляемой энергии, и такое соотношение зачастую выбирается как баланс между этими тремя факторами. Процессоры Intel Core i7, Intel 80486 и AMD Athlon, например, используют одну и ту же архитектуру x86, но реализованную с использованием трех разных микроархитектурных решений.







В компьютере, как и в любой другой сложной системе, существуют различные уровни на которых эта система работает.

Level 7 Application Programs

Level 6 OS 5 BIOS or Drivers SOFT

Level 4 Hardware registers ------------------

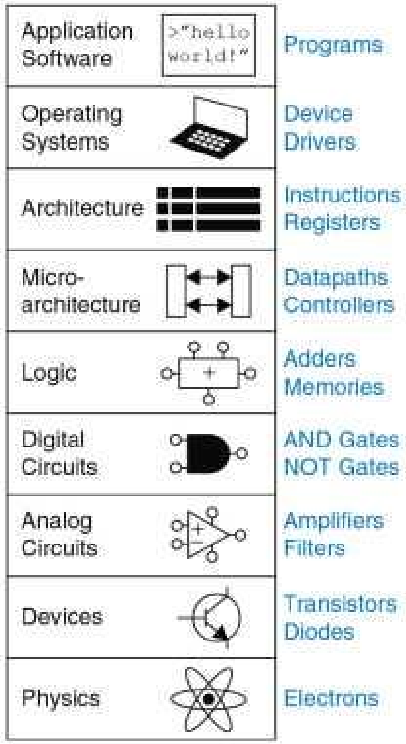
Level 3 Functional HARD

Level 2 Logical ---------------- RTL Transfers

Level 1 Physical ------- Topological Gate

Electrical Transistors

Technological

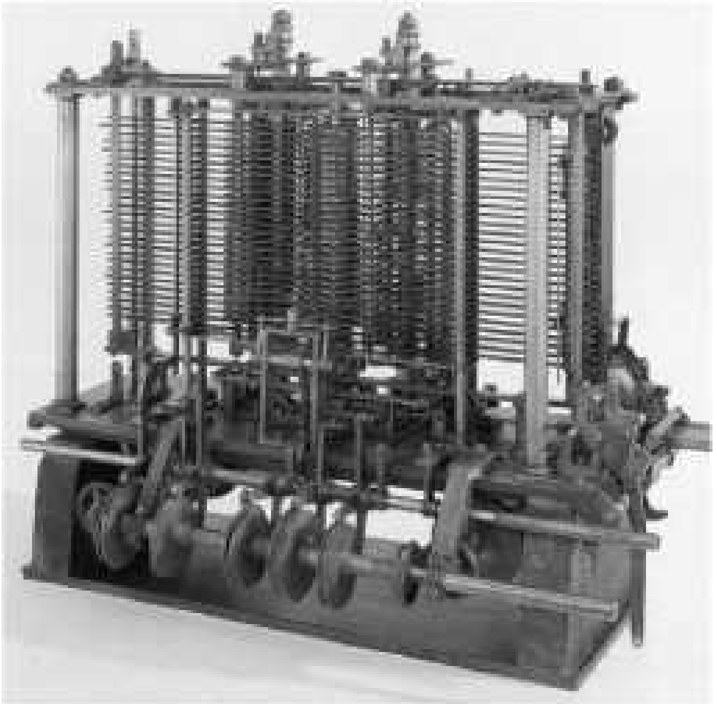


- Теперь мы перемещаемся в область программного обеспечения. *Операционная система (operating system)* управляет операциями нижнего уровня, такими как доступ к жесткому диску или управление памятью. И, наконец, программное обеспечение использует ресурсы операционной системы для решения конкретных задач пользователя.
- Именно принцип *абстрагирования от маловажных деталей* позволяет вашей бабушке общаться с внуками в Интернете, не задумываясь о квантовых колебаниях электронов или организации памяти компьютера.
-

- Предмет курса - уровни абстракции от цифровых схем до компьютерной архитектуры. Работая на каком-либо из этих уровней абстракции, полезно знать кое-что и об уровнях абстракции, непосредственно сопряженных с тем уровнем, где вы находитесь. Программист, например, не сможет полностью оптимизировать код без понимания архитектуры процессора, который будет выполнять эту программу. Инженер-электронщик, разрабатывающий какой-либо блок микросхемы, не сможет найти компромисс между быстродействием и уровнем потребления энергии транзисторами, ничего не зная о той цифровой схеме, где этот блок будет использоваться. Будем надеемся, что к тому времени, когда закончится курс, вы сможете выбрать уровень абстракции, необходимый для успешного выполнения любой стоящей перед вами задачи, и оценить влияние ваших инженерных решений на другие уровни абстракции в разрабатываемой вами системе.

- **Конструкторская дисциплина**
- *Конструкторская Дисциплина* - это преднамеренное ограничение самим конструктором выбора возможных вариантов разработки, что позволяет работать продуктивнее на более высоком уровне абстракции. Использование взаимозаменяемых частей - это, вероятно, самый хорошо знакомый всем нам пример практического применения конструкторской дисциплины.
- Соблюдение конструкторской дисциплины в виде максимального использования цифровых схем играет очень важную роль для понимания архитектуры ЭВМ. В цифровых схемах используются дискретные значения напряжения, в то время как в аналоговых схемах напряжение изменяется непрерывно. Таким образом, цифровые схемы, которые можно рассматривать как подмножество аналоговых цепей, в некотором смысле уступают по своим характеристикам более широкому классу аналоговых цепей. Однако цифровые цепи гораздо проще проектировать. Ограничивая использование аналоговых схем и по возможности заменяя их цифровыми, мы можем легко объединять отдельные компоненты в сложные системы, которые, в конечном итоге, для большинства приложений превзойдут по своим параметрам системы, построенные на аналоговых цепях. Примером тому могут служить цифровые телевизоры, компакт-диски (CD) и мобильные телефоны, которые уже практически полностью вытеснили своих аналоговых предшественников.

- **Три базовых принципа**
- В дополнение к абстрагированию от несущественных деталей и конструкторской дисциплине разработчики электронных систем используют еще три базовых принципа для управления сложностью системы: иерархичность, модульность конструкции и регулярность. Эти принципы применительно как к программному обеспечению, так и к аппаратной части компьютерных систем.
- *Иерархичность* - принцип иерархичности предполагает разделение системы на отдельные модули, а затем последующее разделение каждого такого модуля на фрагменты до уровня, позволяющего легко понять поведение каждого конкретного фрагмента.
- *Модульность* - принцип модульности требует, чтобы каждый модуль в системе имел четко определенную функциональность и набор интерфейсов и мог быть легко и без непредвиденных побочных эффектов соединен с другими модулями системы.
- *Регулярность* - принцип регулярности требует соблюдения единообразия при проектировании отдельных модулей системы. Стандартные модули общего назначения, например, такие как блоки питания, могут использоваться многократно, во много раз снижая количество модулей, необходимых для разработки новой системы в виде дискретно меняющихся переменных с конечным числом строго определённых значений.



- Одной из наиболее ранних цифровых систем стала Аналитическая Машина Чарльза Бэббиджа, которая использовала переменные с десятью дискретными значениями. Начиная с 1834 года и до 1871 года1 Бэббидж разрабатывал и пытался построить этот механический компьютер. Шестеренки Аналитической Машины могли находится в одном из десяти фиксированных положений, а каждое такое положение было промаркировано от 0 до 9 подобно механическому счетчику пробега автомобиля. **Рис.** показывает, как выглядел прототип Аналитической Машины. Каждый ряд шестеренок такой машины обрабатывал одну цифру. В своем механическом компьютере Бэббидж использовал 25 рядов шестеренок таким образом, чтобы машина обеспечивала вычисления с точностью до 25-го знака.

- В отличие от машины Бэббиджа большинство электронных компьютеров использует двоичный (бинарный) код. В случае двоичного кода высокое напряжение - это единица, а низкое напряжение - нуль, поскольку гораздо легче оперировать двумя уровнями напряжения, чем десятью.
- Объем информации D, передаваемый одной дискретной переменной, которая может находиться в *N* различных состояниях, измеряется в единицах, называемых *битами,* и вычисляется по следующей формуле:
  - $D = \log_2 N$ bits (1.0)
- Двоичная переменная передает log22 = 1 - один бит информации. Теперь вам, вероятно, понятно, почему единица информации называется битом. *Bit (бит)* - это сокращение от английского *binary digit,* что дословно переводится как *двоичный разряд.* Каждая шестеренка в машине Бэббиджа содержит log2l0 = 3,322 бит информации, поскольку она может находиться в одном из 23,322 = 10 уникальных положений. Теоретически непрерывный сигнал может передавать бесконечное количество информации, поскольку может принимать неограниченное число значений. На практике, однако, шум и ошибки измерения ограничивают информацию, передаваемую большинством непрерывных сигналов, диапазоном от 10 бит до 16 бит. Если же измерение уровня сигнала должно быть произведено очень быстро, то объём передаваемой информации будет еще ниже (в случае 10 бит, например, это будет только 8 бит).
- Предмет этой книги - цифровые схемы, использующие двоичные переменные нуль и единицу. Джордж Буль разработал систему логики, использующую двоичные переменные, и эту систему сегодня называют его именем - *Булева логика.* Булевы переменные могут принимать значения *ИСТИНА (TRUE)* или *ЛОЖЬ (FALSE).* В электронных компьютерах положительное напряжение обычно представляет единицу, а нулевое напряжение представляет нуль. В этой книге мы будем использовать понятия единица (1), ИСТИНА (TRUE) и ВЫСОКОЕ (HIGH) как синонимы. Аналогичным образом мы будем использовать нуль (0), ЛОЖЬ (FALSE), и НИЗКОЕ (LOW) как взаимозаменяемые термины.

- Преимущества *цифровой абстракции* заключаются в том, что разработчик цифровой системы может сосредоточиться исключительно на единицах и нулях, полностью игнорируя, каким образом булевы переменные представлены на физическом уровне. Разработчика не волнует, представлены ли нули и единицы определенными значениями напряжения, вращающимися шестернями или уровнем гидравлической жидкости. Программист может продуктивно работать, не располагая детальной информацией об аппаратном обеспечении компьютера. Однако, понимание того, как работает это аппаратное обеспечение, позволяет программисту гораздо лучше оптимизировать программу для конкретного компьютера.
- Как вы могли видеть выше, один-единственный бит не может передать большого количества информации. Поэтому в следующем разделе мы рассмотрим вопрос о том, каким образом набор битов можно использовать для представления десятичных чисел. В последующих главах мы также покажем, как группы битов могут представлять буквы и даже целую программу.

- Подавляющее большинство современных вычислительных машин построено по принципу архитектуры фон Неймана. В функциональном устройстве компьютера можно выделить следующие основные блоки:
- устройства ввода-вывода (УВВ),
- память,
- центральный процессор.
- Все они взаимодействуют между собой через системную шину. Устройства ввода принимают закодированную информацию от операторов, электромеханических устройств (клавиатура, мышь) или от других компьютеров сети. Полученная информация либо сохраняется в памяти компьютера для дальнейшего использования, либо немедленно используется АЛУ для выполнения необходимых операций. Последовательность шагов обработки определяется хранящейся в памяти программой. Полученные результаты обратно отправляются получателю посредством устройств вывода. Все эти действия координируются устройством управления.

- **Основные принципы построения ЭВМ**
- Основные принципы построения ЭВМ были сформулированы американским ученым Джоном фон Нейманом в 40-х годах 20 века:
- Принцип двоичности. Для представления данных и команд используется двоичная система счисления.
- Принцип программного управления. Программа состоит из набора команд, которые выполняются процессором друг за другом в определённой последовательности.
- Принцип однородности памяти. Как программы (команды), так и данные хранятся в одной и той же памяти (и кодируются в одной и той же системе счисления, чаще всего – двоичной). Над командами можно выполнять такие же действия, как и над данными.
- Принцип адресуемости памяти. Структурно основная память состоит из пронумерованных ячеек, процессору в произвольный момент времени доступна любая ячейка.
- Принцип последовательного программного управления. Все команды располагаются в памяти и выполняются последовательно, одна после завершения другой.
- Принцип условного перехода. Команды из программы не всегда выполняются одна за другой. Возможно присутствие в программе команд условного перехода, которые изменяют последовательность выполнения команд в зависимости от значений данных. (Сам принцип был сформулирован задолго до фон Неймана Адой Лавлейс и Чарльзом Бэббиджем, однако он логически включен в фоннеймановский набор как дополняющий предыдущий принцип).

Структура классической ЭВМ

СИСТЕМЫ СЧИСЛЕНИЯ

- Мы все привыкли работать с десятичными числами. Однако, в цифровых системах, построенных на единицах и нулях, использование двоичных или шестнадцатеричных чисел зачастую более удобно.
- В своей повседневной деятельности человек использует различные системы счисления, к которым относятся десятичная система счисления, римская система, система исчисления времени и т. д. Все их можно подразделить на позиционные и непозиционные.
- В непозиционных системах счисления «доля» цифры или ее вес в количественном измерении записанного числа не зависит от местоположения данной цифры в записи этого числа. Типичным примером такой системы счисления является римская. В ней используются цифры:

I	V	X	L	C	D	M	– римские цифры
1	5	10	50	100	500	1000	– их десятичные эквиваленты

- При количественной оценке числа его значение определяется как сумма значений цифр, составляющих запись числа, кроме пар, состоящих из цифры меньшего веса, предшествующей цифре большего веса, значение которой определяется как разность веса большей и меньшей цифр.

```
                 Системы счисления
                 /               \
          Позиционные        Непозиционные
        /      |      \
Традиционные Смешанные  Нетрадиционные
  P-ричные    P-Q-            |
                        Сбалансированные
```

- Позиционные - вклад цифры зависит от ее позиции Непозиционные - вклад чифры зависит от самой цифры (как правило непозиционные системы - алфавитные).
- Позиционная система счисления - система счисления, в которой числовое значение каждой цифры зависит от номера ее позиции в последовательности цифр, представляющих число. Непозиционная система счисления - система счисления в котором цифре всегда соответствует определенное значение не зависимо от местоположения в записи числа.

- $a_0 + a_1 P + a_2 P^2 + \ldots + a_{n1} P^{n1} = b_0 + b_1 Q + b_2 Q^2 + \ldots + b_{n2} Q^{n2}$.

- ***Смешанной*** называется система счисления, в которой числа, заданные в некоторой системе счисления с основанием P изображаются с помощью цифр другой системы счисления с основанием Q, где Q<P. В такой системе P называется ***старшим основанием***, Q — ***младшим основанием***, а сама система счисления называется ***Q-P-ичной***.

  В смешанной системе счисления во избежании разночтения для изображения каждой P-ичной цифры отводится одинаковое количество Q-ичных разрядов, достаточное для представления любой P-ичной цифры.

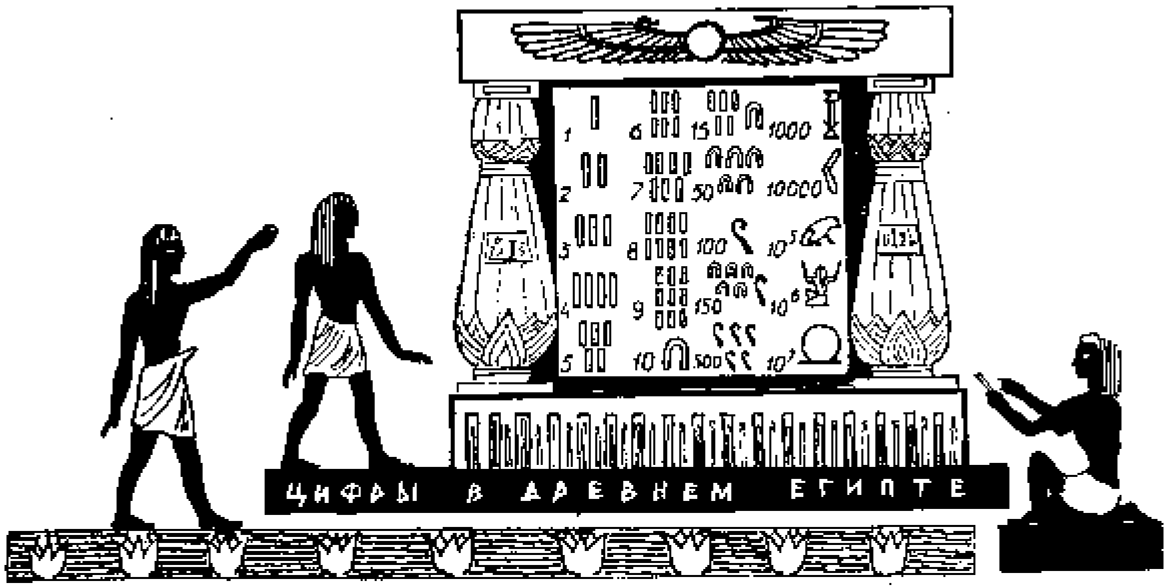
- **Двоично-десятичная система счисления**
- Примером смешанной системы счисления является ***двоично-десятичная система***. В двоично-десятичной системе счисления для изображения каждой десятичной цифры отводится 4 двоичных разряда, поскольку максимальная десятичная цифра 9 кодируется как 10012. Например,
- 925_{10} = $1001\ 0010\ 0101_{2\text{-}10}$.
- Здесь последовательные четверки (тетрады) двоичных разрядов изображают цифры 9, 2 и 5 десятичной записи соответственно.
- Хотя в двоично-десятичной записи используются только цифры 0 и 1, эта запись отличается от двоичного изображения данного числа. Например, двоичный код 1001 0010 0101 соответствует десятичному числу 2341, а не 925.
- В случае если $P=Q^l$ (l – целое положительное число), запись любого числа в смешанной системе счисления тождественно совпадает с изображением этого числа в системе счисления с основанием Q. Примерами такой смешанной системы счисления являются двоично-восьмеричная и двоично-шестнадцатеричная.
- Например,
- $A2_{16} = 1010\ 0010_2 = 1010\ 0010_{2\text{-}16}$

Римские цифры

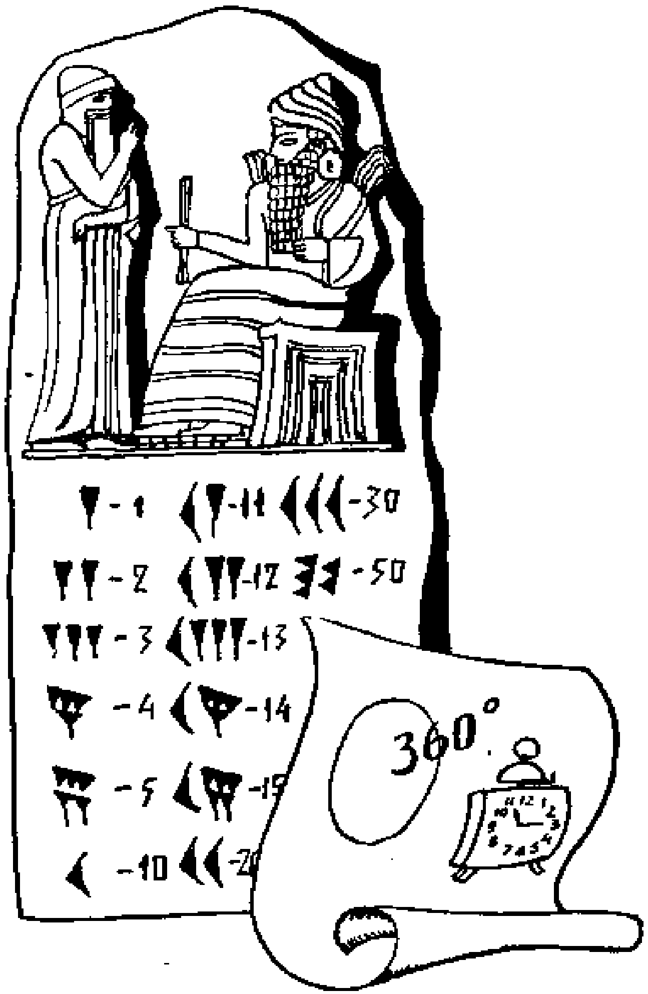
- I 1
- V 5
- X 10
- L 50
- C 100
- D 500
- M 1000

- XC 90
- XD 490
- MCMLXXIV 1974
- MMMM
- 4000
- MMMMMMMMMXM
- 9990
- MMMMMMMMMM
- 10000

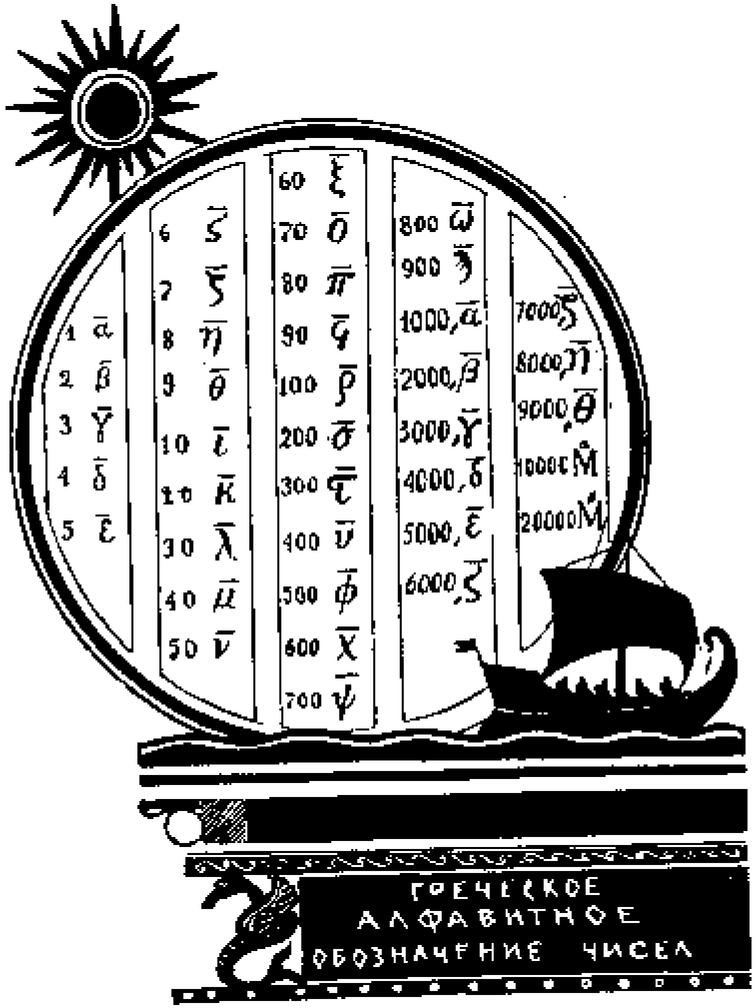
- Непозиционная система с дополнительными правилами. Правила: Значение числа равно
- 1) сумме значений идущих подряд одинаковых цифр
- 2) разности двух цифр если слева от большей стоит меньшая (причем левая чифра может быть меньше только на порядок) т.е. для V и X - I для L и C - X для D и M - C
- 3) сумма значений цифр и групп не вошедших в (1) и (2)



- **Вавилонская**
- Наследие от вавилона: 24 часа 60 минут 60 секунд
- 360 градусов 60 минут 60 секунд



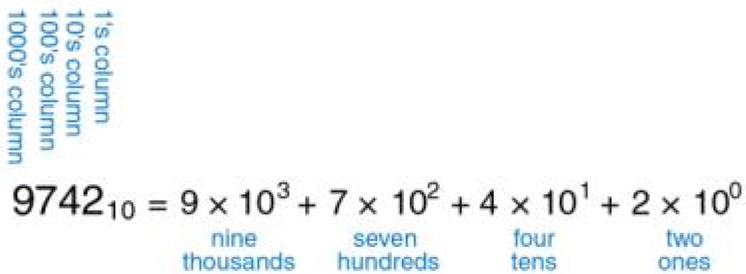
Клинописная запись чисел древних вавилонян.



- *Пример*
- Значение числа MMMCMLIX определяется как сумма:
- 1000 + 1000 + 1000 + (1000 – 100) + 50 + (10 – 1),
- что соответствует десятичному эквиваленту 3959.
- Количественная оценка числа, записанного в позиционной системе счисления, определяется как сумма произведений значения цифр, составляющих запись числа, умноженных на вес позиции, в которой располагается цифра.
- Примером такой системы счисления является широко используемая десятичная система счисления.

Десятичная система счисления

- Еще в начальной школе нас всех научили считать и выполнять различные арифметические операции в *десятичной (decimal)* системе счисления. Такая система использует десять арабских цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 - столько же, сколько у нас пальцев на руках. Числа больше 9 записываются в виде строки цифр. Причем, цифра, находящаяся в каждой последующей позиции такой строки, начиная с крайней правой цифры, имеет «вес», в десять раз превышающий «вес» цифры, находящейся в предыдущей позиции. Именно поэтому десятичную систему счисления называют системой по основанию *(base)* 10. Справа налево «вес» каждой позиции увеличивается следующим образом: 1, 10, 100, 1000 и т.д. Позицию, которую цифра занимает в строке десятичного числа, называют разрядом или декадой.
- Чтобы избежать недоразумений при одновременной работе с более чем одной системой счисления, основание системы обычно указывается путем добавления цифры позади и чуть ниже основного числа:
- 9742_{10}. **Рис.** показывает, для примера, как десятичное число 9742_{10} может быть записано в виде суммы цифр, составляющих это число, умноженных на «вес» разряда, соответствующего каждой конкретной цифре.
- n-разрядное десятичное число может представлять одну из 10^n цифровых комбинаций: 0, 1, 2, 3, ... 10^n - 1. Это называется диапазоном n-разрядного числа. Десятичное число, состоящее из трех цифр (разрядов), например, представляет одну из 1000 возможных цифровых комбинаций в диапазоне от 0 до 999.



- Десятичная *система счисления* является также системой с равномерно распределенными весами, которые характеризуются тем, что соотношение весов двух любых соседних разрядов имеет для такой системы одинаковое значение. Это соотношение называется основанием системы счисления, которое далее будем обозначать как «q».
- Общая запись числа в системе с равномерно распределенными весами имеет вид
- $N_q = A_n A_{n-1} \ldots A_2 A_1 A_0$. (1.1)
- Значение такого числа определяется как
- $\mathrm{N}_q = A_n \times q_n + A_{n-1} \times q_{n-1} + A_{n-2} \times q_{n-1} + \ldots A_2 \times q_2 + A_1 \times q_1 + A_0 \times q_0,$ (1.2)
- где A_i – цифра записи числа, удовлетворяющая условию
- $0 \leq A_i \leq (q-1)$,
- где q – основание системы счисления.
- При q =10 A изменяется в диапазоне от 0 до 9.
- Запись числа N в виде (1.1) называется кодированной, а запись в форме (1.2) – расширенной.
- Помимо q = 10 (десятичная система счисления), возможны другие значения для основания системы счисления:
  - **двоичная система счисления;**
  - **восьмеричная система счисления;**
  - **шестнадцатеричная система счисления** и т. д.

- В различных системах счисления в качестве цифр используются обозначения соответствующих цифр десятичной системы счисления – 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, а в случае, когда десятичных цифр «не хватает» (для систем счисления с основанием *q*, большим чем 10), для цифр, превышающих 9, вводятся дополнительные обозначения, например, для *q* = 16, это будут обозначения A, B, C, D, E, F, которые соответствуют шестнадцатеричным цифрам (десятичные эквиваленты их равны, соответственно 10, 11, 12, 13, 14, 15).
- В связи с тем, что в дальнейшем изложении будут использоваться различные системы счисления, примем такое обозначение:
- N_q – число *N*, представленное в системе счисления с основанием *q*.
- ***Примеры записи чисел в различных системах счисления***:
- $N_2 = 10011011 = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$;
- $N_8 = 471025 = 4 \times 8^5 + 7 \times 8^4 + 1 \times 8^3 + 0 \times 8^2 + 2 \times 8^1 + 5 \times 8^0$;
- $\mathrm{N}_{16} = 84FE4A = 8 \times 16^5 + 4 \times 16^4 + F \times 16^3 + E \times 16^2 + 4 \times 16^1 + A \times 16^0$;
- $N10 = 35491 = 3 \times 10^4 + 5 \times 10^3 + 4 \times 10^2 + 9 \times 10^1 + 1 \times 10^0$.

- На основании вышеизложенного можно заключить, что запись одного и того же числа в различных системах счисления будет тем длиннее, чем меньше основание системы счисления. Например,
- $N = 2063_{10} = 100000001111_2 = 4017_8 = 80F_{16}$.
- При работе с различными системами счисления полезно помнить соотношения, приведенные в табл. 1.1. и 1.2.

Соответствие показателя степени двоичного числа значению десятичного числа

n	0	1	2	3	4	5	6	7	8	9	10	11
2^n	1	2	4	8	16	32	64	128	256	512	1024	2048

Соответствие символов различных систем счисления

№ п/п	$q = 2$	$q = 8$	$q = 16$	$q = 10$	Десятичный эквивалент	Двоичный эквивалент
1	0	0	0	0	0	0000
2	1	1	1	1	1	0001
3	–	2	2	2	2	0010
4	–	3	3	3	3	0011
5	–	4	4	4	4	0100
6	–	5	5	5	5	0101
7	–	6	6	6	6	0110
8	–	7	7	7	7	0111
9	–	*10*	8	8	8	1000
10	–	*11*	9	9	9	1001
11	–	*12*	A	*10*	10	1010
12	–	*13*	B	*11*	11	1011
13	–	*14*	C	*12*	12	1100
14	–	*15*	D	*13*	13	1101
15	–	*16*	E	*14*	14	1110
16	–	*17*	F	*15*	15	1111

Двоичная система счисления

- Человек в своей практической деятельности наиболее часто использует *десятичную* систему счисления. *Двоичная система счисления* является удобной для обработки информации в ЭВМ. Промежуточное место между ними занимает *двоично-десятичная система* счисления, которая, в принципе, является десятичной, но отдельные десятичные цифры в ней записываются в виде набора двоичных разрядов. Существуют разные двоично-десятичные системы.
- Например, десятичное число 804714_{10} в двоично-десятичной системе представляется в виде $1000\ 0000\ 0100\ 0111\ 0001\ 0100_2$.
- В дальнейшем для сокращения будем использоваться название «двоично-десятичная система»(BDC), имея в виду двоично-десятичную систему 8, 4, 2, 1. Цифры в обозначении разновидности двоично-десятичной системы указывают веса соответствующих двоичных разрядов.

Таблица двоичных чисел и их десятичный эквивалент

1-битные двоичные числа	2-битные двоичные числа	3-битные двоичные числа	4-битные двоичные числа	Десятичные эквиваленты
0	00	000	0000	0
1	01	001	0001	1
	10	010	0010	2
	11	011	0011	3
		100	0100	4
		101	0101	5
		110	0110	6
		111	0111	7
			1000	8
			1001	9
			1010	10
			1011	11
			1100	12
			1101	13
			1110	14
			1111	15

Перевод чисел из одной системы счисления в другую

- Наличие различных систем счисления предполагает использование способов перевода записи числа из одной системы в другую. Для этой цели применяются следующие методы преобразований:
- преобразования с использованием весов разрядов в исходной и в искомой записи числа;
- деления (умножения) на новое основание;
- с использованием особого соотношения заданной и искомой систем счисления.
- ***Метод преобразования с использованием весов разрядов***
- Метод преобразования с использованием весов разрядов записи числа в исходной и в искомой системах предполагает применение расширенной записи числа (1.2) в некоторой системе счисления.
- Метод имеет две разновидности в зависимости от того, какая система счисления (исходная или искомая) является более привычной. Если более привычной является искомая система, то на основании расширенной записи исходного числа подсчитываются значения ее отдельных разрядов в новой системе счисления. Далее полученные значения суммируются. Например, при преобразовании целого двоичного числа
  - $N_2 = 110011010$
- в десятичную систему счисления исходное число представляется в расширенной записи
  - $N = 2^8 + 2^7 + 2^4 + 2^3 + 2^1$
- и рассчитывается вес отдельных (ненулевых) двоичных разрядов в десятичной системе счисления:
- 256, 128, 16, 8, 2.

- Затем искомая запись числа определяется как сумма весов всех ненулевых разрядов записи числа в заданной системе счисления:
- 256 + 128 + 16 + 8 + 2 = 410.
- При преобразовании правильных дробей в принципе используется тот же подход, но при расчете весов отдельных разрядов берутся отрицательные степени основания счисления.

- *Пример*
- Найти двоичный эквивалент десятичного числа:
- 43610 = ____?2.
- *Решение*
- Первый (старший) разряд, имеющий значение 1 в искомой двоичной записи числа, будет разряд весом $2^8 = 256$. С помощью остальных (младших) разрядов искомой записи числа необходимо представить значение 180 (180 – остаток, полученный как 436 – 256).
- Второй разряд с весом $2^7 = 128$ будет иметь в искомой двоичной записи числа значение 1. С помощью остальных (более младших) разрядов искомой записи числа необходимо представить значение 52 (остаток, полученный как 180 – 128).
- Третий разряд с весом $2^6 = 64$ будет иметь в искомой двоичной записи числа значение 0.
- Четвертый разряд с весом $2^5 = 32$ будет иметь в искомой двоичной записи числа значение 1, а остаток – 20.
- Пятый разряд с весом $2^4 = 16$ будет иметь в искомой двоичной записи числа значение 1, а остаток – 4.
- Шестой разряд с весом $2^3 = 8$ будет иметь в искомой двоичной записи числа значение 0.
- Седьмой разряд с весом $2^2 = 4$ будет иметь в искомой двоичной записи числа значение 1, а остаток – 0.
- Восьмой разряд с весом $2^1 = 2$ будет иметь в искомой двоичной записи числа значение 0.
- Девятый разряд с весом $2^0 = 1$ будет иметь в искомой двоичной записи числа значение 0.
- Таким образом
- $436_{10} = 110110100_2$.

- Найти двоичный эквивалент числа $0{,}7_{10} = \underline{?}_{2}$.
- *Решение*
- Предварительный результат ищется с точностью до пяти двоичных разрядов, причем пятый разряд используется только для округления при переходе к четырехразрядному окончательному результату.
- Первый (старший) разрядом весом $2^{-1} = 0{,}5$ искомой двоичной записи числа будет иметь значение 1. С помощью остальных (младших) разрядов искомой записи числа необходимо представить значение 0,2 (0,2 – остаток, полученный как 0,7 – 0,5 = 0,2).
- Второй (старший) разряд с весом $2^{-2} = 0{,}25$ в искомой двоичной записи числа будет иметь значение 0.
- Третий разрядом с весом $2^{-3} = 0{,}13$ в искомой двоичной записи числа будет иметь значение 1. С помощью остальных (более младших) разрядов искомой записи числа необходимо представить значение 0,07 (0,07 – остаток, полученный как 0,2 - 0,13).
- Четвертый разрядом с весом $2^{-4} = 0{,}06$ в искомой двоичной записи числа будет иметь значение 1, а остаток 0,01.
- Пятый разряд с весом $2^{-5} = 0{,}03$ искомой двоичной записи числа будет иметь значение 0.
- Таким образом, десятичное число 0,710 = 0,101102.
- После округления имеет место 0,710 = 0,10112.

Метод деления (умножения) на новое основание

- Метод деления (умножения) имеет две разновидности соответственно для преобразования целых и дробных чисел.
- ***Преобразование целых чисел.***
- Задачу представления числа *N*, заданного в системе q_1, в системе счисления с основанием q_2 можно рассматривать как задачу поиска коэффициентов полинома, представляющего собой расширенную запись числа *N* в системе счисления *q*2:
- $N_{q1} = a_0 + a_1 \times q_2^1 + a_2 \times q_2^2 + ... + a_{n-2} \times q_2^{n-2} +$
- $+ a_{n-1} \times q_2^{n-1} + a_n \times q_2^n = N_{q2}$.

Введем скобочную форму для выражения (1.3):

$$N_{q1} = N_{q2} = a_0 + q_2\underbrace{(a_1 + q_2\underbrace{(a_2 + q_2(a_3 + . + q_2\underbrace{(a_{n-1} + q_2\underbrace{(a_n)}_{N_n})}_{N_{(n-1)}}\ldots)}_{N_2})}_{N_1,};$$

Обозначим выражение в первой скобке как N_1, выражение во второй скобке как N_2, в третьей – как N_3 и т. д., выражение в $(n–1)$-й скобке – как $N_{(n–1)}$, выражение в n-й скобке – как N_n. Теперь, основываясь на выражении (1.3), можно утверждать, что при делении N_{q1}/q_2 будет получена целая часть частного $\mathrm{int}(N_{q1}/q_2)$ и остаток $\mathrm{rest}(N_{q1}/q_2)$.

Это можно записать:

$N_{q1}/q_2 \rightarrow \mathrm{int}(N_{q1}/q_2)$ — целая часть частного N_1, и остаток $\mathrm{rest}(N_{q1}/q_2)$, равный a_0.

Аналогично для остальных скобок:

$N_1/q_2 \rightarrow \mathrm{int}(N_1/q_2)$ равное N_2 и остаток $\mathrm{rest}(N_1/q_2)$, равный a_1;

$N_2/q_2 \rightarrow \mathrm{int}(N_2/q_2)$ равное N_3, и остаток $\mathrm{rest}(N_2/q_2)$, равный a_2;

$N_{(n-2)}/q\ 2 \rightarrow \mathrm{int}(N_{(n-2)}/q_1)$ равное $N_{(n-1)}$, остаток $\mathrm{rest}(N(n = 2)/q_1)$, равный $a_{(n-2)}$;

$N_{(n-1)}/q_2 \rightarrow \mathrm{int}(N_{(n-1)}/q_2) = N_n = a_n$ и остаток $\mathrm{rest}(N_{(n-1)}/q_2)$, равный $a_{(n-1)}$, при этом $N_n < q_2$.

Отсюда вытекает правило формирования коэффициентов полинома (1.3) или разрядов записи заданного числа N в системе счисления с основанием q_2:

– необходимо разделить исходное число N_{q1} на новое основание q_2, при этом получив целое частное и остаток;

– полученный остаток снова необходимо разделить на q_2, процесс деления продолжается до тех пор, пока частное будет не меньше нового основания q_2. Если очередное сформированное частное будет меньше, чем q_2, то процесс формирования записи заданного числа в новой системе с основанием q_2 считается законченным, а в качестве искомых разрядов новой записи числа используются результаты выполненных операций деления следующим образом:

– в качестве старшего разряда берется значение последнего частного, для остальных разрядов используются значения остатков в порядке, обратном порядку их получения.

Пример

Найти запись в двоичной форме десятичного числа $N_{10} = 436$.

Решение

Делим сначала исходное число N_{10}, а затем получаемые частные на значение нового основания 2 до получения частного со значением, меньше чем 2:

436/2 → int(436/2) = 218 и rest (436/2) = 0;

218/2 → int(218/2) = 109 и rest (218/2) = 0;

109/2 →int(109/2) = 54 и rest (109/2) = 1;

54/2 → int(54/2) = 27и rest (54/2) = 0;

27/2 → int(27/2) = 13 и rest (27/2) = 1;

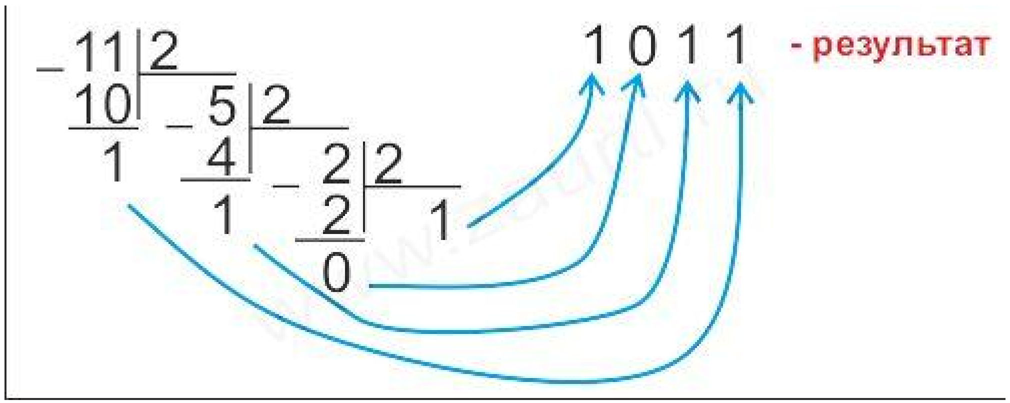
13/2→int(13/2) = 6 и rest (13/2) = 1;

6/2 → int(6/2) = 3 и rest (6/2) = 0;

3/2 → int(3/2) = 1 и rest (3/2) = 1.

Таким образом: 436 = 11 0110100.

- Другой вариант записи



Преобразование дробных чисел

Задача представления дробного числа M_{q1}, заданного в системе q_1, в системе счисления с основанием q_2, можно рассматривать как задачу поиска коэффициентов полинома, представляющего собой расширенную запись числа M в системе счисления q_2:

$$B_1 \cdot q_2^{-1} + B_2 \cdot q_2^{-2} + B_3 \cdot q_2^{-3} + ... + B_{n-2} \cdot q_2^{-(n-2)} + B_{n-1} \cdot q_2^{-(n-1)} + B_n \cdot q_2^{-n} = M_q \quad (1.4)$$

Введем скобочную форму для выражения (1.4). Обозначим выражение в первой скобке как M_1, выражение во второй – как M_2, в третьей скобке – как M_3 и т. д., выражение в $(n - 1)$-й скобке как M_{n-1}, выражение в n-й скобке – как M_n:

$$M_{q2} = M_{q1} = q_2^{-1}(B_1 + q_2^{-1}(B_2 + q_2^{-1}(B_3 + \ldots + q_2^{-1}(B_{n-1} + q_2^{-1}(B_n))\ldots))).$$

Число M_{q1} – правильная дробь, поэтому при умножении $M_{q1} \cdot q_2$ будет получено произведение, в общем случае состоящее из целой части $\mathrm{int}(M_{q1} \cdot q_2)$ и дробной части DF $(M_{q1} \cdot \mathrm{q}_2)$. Использование введенных обозначений позволяет записать:

$$M_{q1} \cdot q_2 = (\mathrm{int}(M_{q1} \cdot q_2) = B_1) + (\mathrm{DF}(M_{q1} \cdot q_2) = M_1),$$

аналогично для остальных скобок будем иметь следующее:

$$M_1 \cdot q_2 = (\mathrm{int}(M_1 \cdot q_2) = B_2) + (\mathrm{DF}(M_1 \cdot q_2) = M_2);$$

$$M_2 \cdot q_2 = (\mathrm{int}(M_2 \cdot q_2) = B_3) + (\mathrm{DF}(M_2 \cdot q_2) = M_3);$$

$$M_3 \cdot q_2 = (\mathrm{int}(M_3 \cdot q_2) = B_4) + (\mathrm{DF}(M_3 \cdot q_2) = M_4);$$

$$M_{n-2} \cdot q_2 = (\mathrm{int}(M_{n-2} \cdot q_2) = B_{n-1}) + (\mathrm{DF}(M_{n-2} \cdot q_2) = M_{n-1});$$

$$M_{n-1} \cdot q_2 = (\mathrm{int}(M_{n-1} \cdot q_2) = B_n) + (\mathrm{DF}(M_{n-1} \cdot q_2) = M_n);$$

$$M_n \cdot q_2 = (\mathrm{int}(M_n \cdot q_2) = B_{n+1}) + (\mathrm{DF}(M_n \cdot q_2) = M_{n+1}).$$

Отсюда вытекает следующее правило формирования коэффициентов полинома, которые одновременно являются разрядами записи заданного числа M в системе счисления с основанием q_2:

– определяется количество разрядов «n» в записи числа M_{q2} в новой системе счисления;

– исходное число M_{q1} умножается на q_2, при этом будет получено смешанное число;

– дробная часть полученного произведения снова умножается на q_2 и т. д.; процесс умножения повторяется n +1 раз. В качестве искомых разрядов новой записи числа используются результаты выполненных операции деления следующим образом:

– в качестве первого старшего разряда искомой записи числа в новом основании берется значение целой части первого произведения, в качестве второго старшего разряда искомой записи числа в новом основании берется значение целой части второго произведения и т. д.

Найти запись в двоичной форме десятичного числа M_{10} = 0,7.

Решение

Определяем количество разрядов числа M_2. Так как исходная запись числа содержит один десятичный разряд, то запись данного числа в двоичном основании должна содержать четыре разряда. Учитывая округление, ищется предварительный двоичный эквивалент с пятью разрядами.

Умножаем исходное число M_{10}, а затем дробные части последовательно получаемых произведений на новое основание 2. Выполняется пять таких операций умножения, в результате получаем:

$0{,}7 \cdot 2 = 1{,}4$ ($\mathrm{int}(0{,}7 \cdot 2) = 1$ и $\mathrm{DF}\ (0{,}7 \cdot 2) = 0{,}4$);

$0{,}4 \cdot 2 = 0{,}8$ ($\mathrm{int}(0{,}4 \cdot 2) = 0$ и $\mathrm{DF}(\mathrm{rest}\ (0{,}4 \cdot 2) = 0{,}8$);

$0{,}8 \cdot 2 = 1{,}6$ ($\mathrm{int}(0{,}8 \cdot 2) = 1$ и $\mathrm{DF}(\mathrm{rest}\ (0{,}8 \cdot 2) = 0{,}6$);

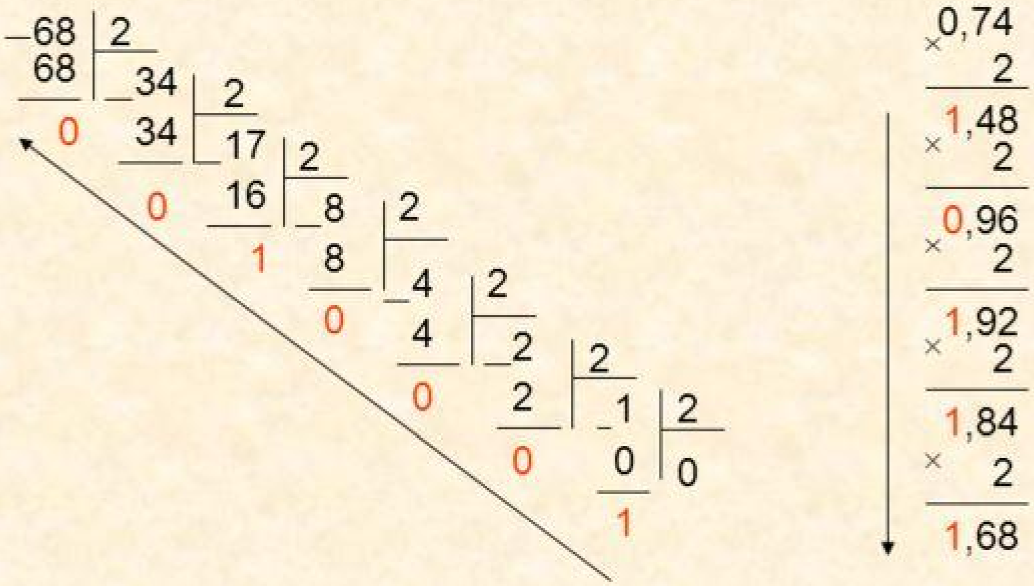
$0{,}6 \cdot 2 =1{,}2$ ($\mathrm{int}(0{,}6 \cdot 2) = 1$ и $\mathrm{DF}(\mathrm{rest}\ (0{,}6 \cdot 2) = 0{,}2$);

$0{,}2 \cdot 2 = 0{,}4$ ($\mathrm{int}(0{,}2 \cdot 2) = 0$ и $\mathrm{DF}(\mathrm{rest}\ (0{,}2 \cdot 2) = 0{,}4$).

Таким образом, 0,7 = 0,10110, а окончательный результат перехода в двоичную систему будет $0{,}7_{10} = 0{,}1011_2$.

Перевод вещественных чисел из десятичной системы счисления

Пример. Перевести число 68,74 из десятичной системы в счисления в двоичную, восьмеричную и шестнадцатеричную



$68{,}74_{10} = 1000100{,}10111_2$

1. 3. Метод с использованием особого соотношения оснований заданной и искомой систем счисления

Данный метод применим в тех случаях, когда исходное q_1 и новое q_2 основания могут быть связаны через целую степень, т.е. когда выполняются условия: $q_1{}^m = q_2$ (*условие 1*) или $q_2{}^m = q_1$ (*условие 2*).

Если имеет место *условие 2*, то для заданного в системе с основанием q_1 числа $N_{q1} = a_n\, a_{n\text{-}1}\, a_{n\text{-}2} \ldots\, a_1 a_0$ запись его в системе с новом основании q_2 определяется следующим образом:

– каждому разряду a_i исходной записи числа ставится в соответствие его m-разрядный эквивалент в системе счисления с основанием q_2;

– искомая запись всего заданного числа формируется за счет объединения всех полученных m-разрядных групп.

Если имеет место *условие 1*, то запись заданного числа

$N = a_n a_{n-1} a_{n-2} \ldots a_1 a_0$

в системе с новом основании q_2 формируется следующим образом:

–исходная запись числа разбивается на группы по m разрядов, двигаясь от точки вправо и влево (недостающие разряды в крайних группах (слева и справа) дополняются нулями;

–каждой полученной группе ставится в соответствие цифра новой системы счисления;

–искомая запись заданного числа в новой системе счисления образуется из цифр, соответствующих группам, на которые была разбита исходная запись.

Пример

Найти двоичный эквивалент восьмеричного числа 67401.64_8.

Решение

Основания исходной и новой систем счисления можно выразить через целую степень:

$2^3 = 8$.

Поэтому применяем третий метод для случая перехода из системы с большим основанием в систему с меньшим основанием. Ставим в соответствие каждой цифре исходной записи числа трехразрядный двоичный код (*триаду*):

6	7	4	0	1	6	4
110	111	100	000	001	110	100

Формируем окончательный результат посредством объединения полученных трехразрядных двоичных чисел в единый двоичный эквивалент:

$67401.64_8 = 110111100000001.110100.$

Пример

Найти шестнадцатеричный эквивалент двоичного числа

$N = 1110010110110.111011001_2.$

Решение

Основания исходной и новой систем счисления можно выразить через целую степень:

$2^4 = 16.$

Поэтому применяем третий метод для случая перехода из системы с меньшим основанием в систему с большим основанием. Разбиваем исходную запись числа на группы по четыре разряда (*тетрады*) вправо и влево от точки, в крайних левой и правой группах недостающие разряды заполняем нулями и каждой полученной группе из четырех разрядов ставим в соответствие цифру шестнадцатеричной системы счисления

0011	1001	0111	0110	.	1110	1100	1000
3	9	7	6		E	C	8

Формируем окончательный результат посредством объединения полученных цифр в единый шестнадцатеричный эквивалент

$1110010111 0110.111011001_2 = 3976.EC8_{16}$.

Пример

Найти шестнадцатеричный эквивалент числа 67401.64_8, представленного в восьмеричной системе счисления.

Решение

Основания исходной q_1 и новой q_2 систем счисления не могут быть связаны через целую степень, поэтому напрямую третий метод перехода неприменим. Однако существует система с двоичным основанием, для которой допустим третий метод перехода и восьмеричную (исходную для данного примера), и в шестнадцатеричную (новую систему для данного примера) системы счисления, т. к.

$2^3 = 8$ и $2^4 = 16$.

Поэтому в данном случае для решения поставленной задачи целесообразно использовать два быстрых перехода из восьмеричной системы счисления в двоичную (промежуточную), а затем из двоичной системы счисления в шестнадцатеричную третьим методом. Это будет гораздо быстрее, чем использовать для заданного преобразования второй или третий метод.

Таким образом, поставленная задача решается в следующем порядке:

$67401.64_8 = 110\ 111\ 100\ 000\ 001\ .\ 110\ 100_2$;

$110\ 111\ 100\ 000\ 001\ .\ 110\ 100_2 = 0110\ 1111\ 0000\ 0001.\ 1101\ 0000_2 = 6\ F\ 0\ 1.\ D\ 0_{16}$, т.е.:

$67401.64_8 = 6F01.D0_{16}$.

Пример

Найти двоичный эквивалент числа 6740_{10}.

Решение

Основания исходной q_1 и новой q_2 систем счисления не могут быть связаны через целую степень, поэтому третий метод перехода неприменим. В принципе здесь целесообразно использовать второй метод – метод деления на новое основание. Однако в этом случае потребуется большое количество операций деления на два. Для сокращения количества операций деления может оказаться целесообразным решить эту задачу за счет перехода с использованием второго метода в промежуточную шестнадцатеричную систему счисления, а затем, используя третий метод, быстро перейти в заданную двоичную систему счисления:

– выполняем переход в промежуточную систему счисления:

6740/16= 421 (остаток 4);

421/16=26 (остаток 5);

26/16 = 1 (остаток 10);

– в промежуточной системе счисления имеем:

$6740_{10} = 1A54_{16}$,

– выполняем переход из промежуточной системы счисления в заданную:

$1A54_{16} = 0001\ 1010\ 0101\ 0100_2$.

Как видно из вышеприведенного, для заданного перехода потребовалось выполнить только 3 операции деления на «16» вместо 13-ти операций деления на 2.