СОДЕРЖАНИЕ

[Введение 4](#_Toc388192438)

[Троичная система счисления 5](#_Toc388192439)

[История троичных ЭВМ 7](#_Toc388192447)

[Аппаратная реализация 12](#_Toc388192448)

[Перспективы 13](#_Toc388192453)

[Заключение 14](#_Toc388192462)

[Список использованной литературы 15](#_Toc388192463)

# 

# Введение

# Троичная система счисления

# Трёхзначная логика  — один из видов многозначной логики. Она отличается от двухзначной тем, что кроме значений «истина» и «ложь» существует третье, которое понимается как «не определено». При этом сохраняется совместимость с двухзначной логикой – логические операции с «известными» значениями дают те же результаты.

# Логике, оперирующей тремя значениями, соответствует троичная симметричная система счисления. К этой системе впервые обратился итальянский математик Фибоначчи для решения своей «задачи о гирях».

# Сущность задачи состоит в следующем, при какой системе гирь, имея их по одной, можно взвесить всевозможные грузы от 0 до максимального груза, чтобы значение максимального груза было бы наибольшим среди всех возможных вариаций? Известны следующие решения этой задачи:

# 1) когда гири позволено класть на свободную чашу весов;  2) когда гири позволяется класть на обе чаши весов.

# В первом случае "оптимальная система гирь" сводится к двоичной системе гирь: 1, 2, 4, 8, 16, ..., а появляющийся при этом "оптимальный" алгоритм или способ измерения рождает двоичную систему счисления, лежащую в основе современных компьютеров. Во втором – наилучшей является троичная система гирь: 1, 3, 9, 27, 81, ..., а возникающий при этом способ измерения рождает троичную симметричную систему счисления.

# Троичная система счисления существует в двух вариантах: симметричная, имеющая в наборе отрицательные цифры (−1, 0, +1), и несимметричная (0, 1, 2).

Как отмечает создатель уникального троичного компьютера Н.П. Брусенцов, главное

преимущество троичного представления чисел перед принятым в современных компьютерах двоичным состоит не в иллюзорной экономичности троичного кода. В данном случае речь идет об утверждении, высказанном и доказанном в свое время одним из основателей информатики Джоном фон Нейманом.

Это теорема о представлении некоторого числа n минимальным набором символов в определенной системе счисления. Её основанием является число e = ,2 718281828...,

основание натуральных логарифмов. С математической точки зрения доказательство сводится к поиску экстремума функции . На рис. 1 приведен график этой функции для n = 8.



Рис. 1 Функция, характеризующая компактность систем счисления по основанию x.

# На практике это утверждение воспринимается как интересный факт и используется для оценки "идеальности" системы счисления. Если рассматривать представление чисел только в этом аспекте, то действительно 3 находится ближе к е, чем 2. Это, тем не менее, не является тем решающим критерием, по которому использование троичной симметричной системы счисления более удобно в реальных условиях. Для того чтобы понять в чем удобство упомянутого представления нужно сравнить его с двоичным. Далее приводятся основные характеристики, определяющие практическую ценность троичного кода и трехзначной логики.

* имеет место естественное представление чисел со знаком, т.е. не нужно пользоваться искусственными приемами типа прямого, обратного или дополнительного кода
* знак числа определяется знаком старшей ненулевой цифры и не нужно использовать специальный знаковый бит, как в двоичной системе
* просто производится сравнение чисел по величине, при этом не нужно обращать

внимание на знак числа

* в соответствии с этим команда ветвления по знаку в троичной машине занимает в два раза меньше времени, чем в двоичной
* усечение длины числа равносильно правильному округлению; способы округления,

используемые в двоичных машинах, не обеспечивают этого

* троичный сумматор осуществляет вычитание при инвертировании одного из слагаемых, откуда следует, что троичный счетчик автоматически является реверсивным
* в трехвходовом троичном сумматоре перенос в следующий разряд возникает в 8 ситуациях из 27, а в двоичном сумматоре - в 4 из 8. В четырехвходовом сумматоре

перенос также происходит только в соседний разряд.

* таблицы умножения и деления почти так же просты, как и в двоичной системе,

умножение на -1 инвертирует множимое

* трехуровневый сигнал более устойчив к воздействию помех в линиях передачи. Это означает что специальные методы избыточного кодирования троичной информации проще, нежели двоичной

# История троичных ЭВМ

Первую вычислительную машину с троичной системой счисления построил английский изобретатель-самоучка Томас Фаулер в 1840 году. Его машина была механической и полностью деревянной.

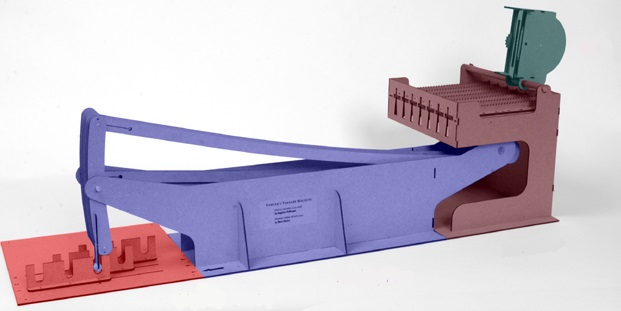
Томас Фаулер работал банковским служащим и по роду деятельности был вынужден производить сложные вычисления. Чтобы облегчить и ускорить свою работу, он сделал таблицы для счета степенями двойки и тройки, а позже опубликовал эти таблицы в виде брошюры. Затем он пошел дальше, решив полностью автоматизировать расчеты по таблицам, и построил счетную машину. Английская патентная система того времени была несовершенна, предыдущее изобретение Фаулера (термосифон для систем парового отопления) было скопировано с минимальными изменениями и запатентовано множеством недобросовестных «изобретателей», поэтому, опасаясь, что его идею снова могут украсть, он решил изготовить машину в единственном экземпляре и – из дерева. Так как дерево – материал ненадежный, для обеспечения достаточной точности вычислений Фаулеру пришлось сделать машину весьма громоздкой, около 2 м в длину. Впрочем, как писал сам изобретатель в сопроводительной записке, отправляя машину в Лондонский королевский колледж, «если бы ее можно было изготовить из металла, она бы оказалась не больше пишущей машинки».

Рис. 1. **Машина Фаулера.**

Машина состояла из четырех частей. В зависимости от выполняемой операции первая часть (на рисунке выделена красным) представляет множимое или частное, вторая (синим) — это множитель или делитель, третья (бордовым) — произведение или делимое, четвертая — аппарат переноса, который упрощает ответ до его простейшей формы после выполнения основных операций и может быть перемещен для этого к множителю или делителю для работы с ними.

Множитель — это рама, которая может перемещаться перпендикулярно стержням множимого и произведения, и расположена между их плоскостями таким образом, что её край может быть совмещён скользящим движением с каждым стержнем множимого поочередно. Множитель состоит из набора стержней, при этом каждый из стержней имеет зубец с каждого края и зубец, которым множитель взаимодействует со стержнями множимого. Один набор зубцов размещён так, что опирается на раму, которая может вращаться вокруг оси. Каждый стержень может соприкасаться с рамой выше, ниже или в районе оси вращения. Зубцы, находящиеся на оси, не получают движения, остальные же придут в движение в ту или иную сторону в зависимости от того, с какой стороны зубец касается рамы. Перпендикулярные зубцы с другого конца могут совершать движение вперёд–назад или оставаться неподвижными, в свою очередь они тем самым взаимодействуют со стержнями, которые составляют раму произведения, которая схожа с рамой множимого и имеет деталь, соединяющую её с множителем.

При умножении рама множителя устанавливается так, чтобы край множителя совпал с первым стержнем множимого. С края закреплен зубец, который зацепляется со стержнем множимого, над которым он оказался, перемещая его в том или ином направлении согласно тому, в какую сторону может поворачиваться рама множителя. Затем вращающуюся раму нужно перемещать таким образом, чтобы стержень множимого попал в ноль. Так перемножается разряд множимого с множителем. Затем воздействие перпендикулярных зубцов отображает результат на раме произведения. Рама множителя сдвигается до тех пор, пока зубец не окажется над следующим стержнем множимого, после этого зубец сдвигается так, чтобы сдвинуть очередной стержень множимого в положение нуля. Следующий разряд множимого будет умножен на множитель, а рама произведения получит и сложит его. Операция повторяется до тех пор, пока не кончатся разряды множимого.

Результат умножения будет представлен не в простейшей форме: он должен содержать только +1 или −1, а полученный промежуточный ответ может содержать на любом стержне +2, −2, +3 или −3 и т.д. Для корректировки результата используется механизм переноса, который представляет собой простое устройство, которое, как и множитель, может совершать поперечное перемещение, и может быть помещено на любой паре соседних стержней. Аппарат переноса выдвигает левый из двух стержней на единицу и втягивает стержень находящийся справа на 3 единицы или наоборот. Даже без наличия обязательного навыка работы с механизмом переноса серьезная ошибка невозможна, так как за каждую операцию можно перенести нижнюю колонку только на 3 единицы, в то время как следующая колонка изменится в противоположном направлении на одну единицу.

Деление чисел осуществляется противоположным способом.

О практическом использовании троичной системы счисления забыли более чем на сто лет. Следующими, кто вернулся к этой идее, были инженеры с кафедры вычислительной математики механико-математического факультета МГУ.

В 1955 году в лаборатории вычислительных систем Московского государственного университета началась разработка троичной вычислительной машины, основанной на троичной системе счисления и троичной логике, относившейся тогда к классу системно малых.

Идея разработки принадлежит Н. П. Брусенцову, и возникла спонтанно, после того, как он увидел, что в разрабатываемой ЭВМ не совсем рационально используются элементы памяти, кроме того из-за несовершенства конструкции их приходилось устанавливать значительно больше, чем нужно.

Предполагалось создать малогабаритную, недорогую, непритязательную в обслуживании и простую в использовании машину для учебных заведений, исследовательских лабораторий, конструкторских бюро, а также для управления технологическими процессорами.

Ввиду неудовлетворительной надежности типичных в то время цифровых элементов с электронными лампами и недоступности транзисторов решили разработать быстродействующие магнитные элементы с использованием миниатюрных ферритовых тороидов и полупроводниковых диодов. Построенные по принципу управляемых трансформаторов импульсов тока эти элементы оказались эффективным средством реализации пороговой логики, в частности, ее трехзначной версии с положительными и отрицательными весами, которая обеспечивала большее по сравнению с двоичной быстродействие, существенную экономию оборудования и лучшую надежность. Поэтому стали разрабатывать троичный компьютер.

Минимальной адресуемой единицей памяти «Сетуни» стал трайт, равный шести тритам и принимающий значения от -364 до 364. Работа с диапазоном отрицательных значений — особенность, отличающая трайт от двоичного байта, значения которого распространяются от 0 до 255. С помощью трайта можно закодировать все заглавные и строчные символы русского и латинского алфавитов, необходимые математические и служебные символы.

Набор команд "Сетуни" состоял всего из двадцати четырёх операций, три из которых были зарезервированы и никогда не использовались. Под код операции отводилось три трита. Шеститритовая адресная часть операции содержала: адрес, указание длины операнда и трит индексации (сложить, вычесть или не индексировать). Шесть тритов адреса позволяли адресовать сто шестьдесят два девятитритных слова, разбитых на три страницы памяти.

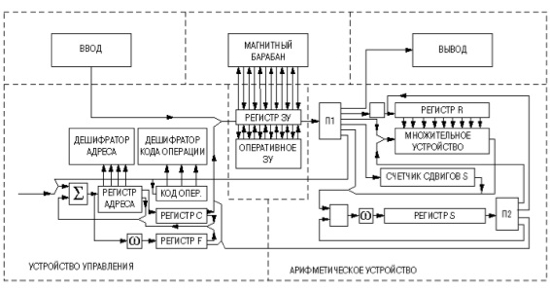


Рис. 2. Блок-схема компьютера «Сетунь».

Аппаратная реализация "Сетуни" была весьма простой. Структурной единицей компьютера стала ячейка, представляющая собой феррит-диодный магнитный усилитель, собранный на гетинаксовой основе. Генератор тактовой частоты задавал такт работы ячеек в двести герц. Ячейки компоновались в функциональные блоки: сумматоры, дешифраторы троичного кода, регистры сдвига. С помощью тридцати контактного разъёма каждый блок стыковался с другими блоками "Сетуни", формируя базовые компоненты ЭВМ: арифметическое устройство и устройство управления.

Память в "Сетуни", подобно современным гибридным системам хранения данных, была двухступенчатой: ферритовый куб ёмкостью в одну страницу постранично обменивался с традиционным для того времени запоминающим устройством - магнитным барабаном.

"Сетунь" показала необыкновенную для того времени безотказность и устойчивость функционирования в широком диапазоне температур при значительных изменениях напряжения электрической сети, была признана несложной в изготовлении и техническом обслуживании, пригодной в широком спектре применений.

Троичный компьютер оказался исключительно благоприятным для пользователей. Простота и экономность программирования в машинном коде (ассемблер разрабатывать не стали) позволили в сжатые сроки создать серию интерпретаторов, в том числе польской инверсной (постфиксной) записи, позволявших эффективно и надежно программировать самые различные применения от научно-технических расчетов и автоматизированной обработки экспериментальных данных до управления производственными процессами и программированного обучения.

Идеи, заложенные в архитектуру первого троичного компьютера и реализованные в "Сетуни", оказались настолько удачными, что в 1967 году было принято решение выпустить её модифицированную версию.

Наряду с аппаратными улучшениями (увеличение объёма оперативной памяти, реализация системы прерываний, уменьшение потребляемой мощности и размеров ЭВМ), важнейшим нововведением стала реализация двух стековой архитектуры.

Рис. 3. Реализации первого варианта «Сетуни» (1958 год) и экземпляр, демонстрировавшийся на ВДНХ в 1961 году.

Выпущенный в 1970 году вариант обновлённого троичного компьютера получил название "Сетунь-70".

Стремление к реализации более продуманного и компактного представления программ привело разработчиков "Сетуни-70" к идее отказа от традиционного машинного кода и использования в качестве машинного языка обратной польской записи ([ПОЛИЗ](http://ru.wikipedia.org/wiki/%D0%9F%D0%9E%D0%9B%D0%98%D0%97)). Стандартные машинные команды, состоящие из кода операции и адреса операнда, были заменены на трайты операций и операндов. Применение обратной польской записи и обусловило стековую архитектуру "Сетуни-70". В первом стеке адресные трайты управляют передачей данных из оперативной памяти в стек, а операционные трайты - преобразованием данных и возвращением результата из стека в оперативную память. При этом, как и положено, при стековой обработке, эти операции выполняются над данными в вершине стека и нижележащими данными.

Пятьдесят четыре операции были реализованы аппаратно. Из них половина была операциями общего пользования, а оставшиеся двадцать семь были служебными и не могли выполняться в пользовательском режиме. Кроме аппаратных операций "Сетунь-70" поддерживала работу с двадцатью семью макрооперациями - подпрограммами, создаваемыми пользователем и по мере надобности вызываемыми из оперативной памяти. Для работы с макрооперациями использовался второй (системный) стек ЭВМ.

Подобная реализация архитектуры "Сетуни-70" была неслучайной. Применение стеков и разработка операций в нотации ПОЛИЗ предполагали внедрение в процесс разработки программ идей структурированного программирования. Структурированный подход существенно экономил время на разработку и отладку сложных программ, разбивая их на ряд структурных единиц, с каждой из которых можно было работать как с независимым объектом. Специально для реализации этой идеи разрабатывалась среда ДССП (Диалоговая система структурированного программирования).

Обмен информацией между троичным компьютером Сетунь и всеми остальными двоичными компьютерами вычислительного центра был затруднен. Эту проблему решили в Сетунь-70. Она, перед печатью на [перфоленту](http://chernykh.net/content/view/71/127/), преобразовывала все данные в двоичный код, и умела читать перфоленты других двоичных компьютеров.

Несмотря на то, что ЭВМ серии «Сетунь» и «Сетунь-70» хорошо себя зарекомендовали высокой производительностью, надёжностью, судьба их весьма печальна. Чиновники всячески препятствовали серийному производству ЭВМ, хотя потребность в них была очень высока.

Было выпущено всего 50 машин первой серии, которые бесперебойно работали много лет без всякого сервисного обслуживания в разных климатических зонах. ЭВМ «Сетунь», которая находилась в ВЦ МГУ в полностью исправном состоянии с полным пакетом программного обеспечения, была распилена и выброшена на свалку после 17 лет бесперебойной работы.

А что касается «Сетунь-70», то она была создана в единственном экземпляре. После создания «Сетунь-70» лабораторию закрыли, и дальнейшие разработки были прекращены.

В 1973 г. в SUNY (State University of New York), Буффало, США Frieder'ом был создан [Ternac](http://ru.wikipedia.org/wiki/Ternac) — эмулятор троичной арифметики на двоичном компьютере. В Ternac'е были предусмотрены два формата чисел: с фиксированной точкой и с плавающей точкой. Слова с фиксированной точкой были длиной 24 трита, слова с плавающей точкой были длиной 48 тритов, 42 трита для мантиссы и 6 тритов для экспоненты.

Реализация эмулятора TERNAC была предпринята для того, чтобы узнать, возможна ли реализация не двоичной структуры на двоичном компьютере, и каков расход памяти и времени в такой реализации. Как проверка возможности, эта попытка была успешной. Первая версия этой реализации доказала, что скорость и расход памяти были того же порядка, как и при двоичных вычислениях.

Что касается нынешнего времени, достоверных фактов использования или разработки современных троичных ЭВМ не обнаружено – имеется определённый интерес со стороны компаний, например, IBM, но отсутствует активное участие или освещение этого участия в прессе.

# Аппаратная реализация

# Согласно работе Александра Кушнерова «Троичная цифровая техника. Ретроспектива и современность», аппаратную реализацию элементов трехзначной логики можно рассматривать с двух позиций. Первая – моделирование троичных функций с помощью кодирования в двоичном виде (binary coded ternary, BCT representation). В этом представлении существует несколько вариантов кодировки, но в любом случае на два бита приходится один трит и ещё один уровень остается неопределенным. Его можно использовать для обозначения разделительной запятой в числах или индикации высокоимпедансного состояния на выходе. Двоичное моделирование обусловлено отсутствием доступных трехуровневых элементов и использованием вместо них стандартных бинарных. Общий подход к синтезу BCT схем рассмотрен в работах "Логические методы анализа и синтеза схем" Д.А. Поспелова и «Algorithms for binary coded balanced and ordinary ternary operations» G. Frieder'а and C. Luk'а и состоит в следующем. Составляется таблица истинности для трехзначной функции в соответствии с выбранной кодировкой. На основе этой таблицы для каждого выходного разряда составляются карты Карно для минимизации моделирующей функции. По результатам минимизации производится синтез бинарных схем.

# Второй подход к реализации трехуровневых устройств заключается в использовании управляемых электронных ключей при симметричном питании.

# Обязательными для построения большинства цифровых устройств являются последовательностные схемы. В их составе особое место занимают всевозможные триггеры. Они различаются между собой по выполняемым функциям и способам управления. В общем случае триггер можно рассматривать как устройство, содержащее запоминающий элемент и комбинационную схему управления. Троичный триггер также как и двоичный можно реализовать на соответствующих элементах И-НЕ (ИЛИ-НЕ), с положительными обратными связями.

# Очень перспективными в качестве элементной базы троичной логики являются полупроводниковые негатроны. Это название присвоено большой группе приборов, имеющих в определенном режиме отрицательное значение дифференциального сопротивления или проводимости.

# Перспективы

# В последнее десятилетие возникла необходимость в поиске новых компьютерных технологий, и некоторые из этих технологий лежат в области троичности.

# Одно из направлений – разработки в области асинхронных процессоров.

# Компания Theseus Logic предлагает использовать «расширенную двоичную» (фактически – троичную) логику, где помимо обычных значений «истина» и «ложь» есть отдельный сигнал «NULL», который используется для самосинхронизации процессов. В этом же направлении работают еще несколько исследовательских групп.

# Такие компании, как IBM, Motorola и Texas Instruments, ведут исследования, в рамках которых можно реализовать цифровые интегральные схемы, работающие с тремя и более уровнями сигнала.

Другое направление - поиск альтернативных способов увеличения производительности процессоров.

# Согласно закону Мура, количество транзисторов в микропроцессорах удваивается каждые полтора года. Корпорация Intel и другие компании, занимающиеся выпуском полупроводниковых устройств, уже вплотную подошли к нанометрическим масштабам; гига- и даже терагерцевыми частотами компьютерных систем уже никого особо не удивишь и не испугаешь, равно как и мега-, гига- и терабайтовыми вместилищами данных.

# Сейчас отрасль стоит на сложном распутье.

# С одной стороны, за законом Мура угнаться становится все сложнее: «втискивать» все новые и новые двоичные кремниевые транзисторы на платы становится труднее — и так масштабы уже нанометрические. По мере дальнейшего уменьшения размеров транзисторов разработчики столкнутся с целым рядом технических сложностей.

# С другой стороны, все больше людей в мире понимает, что в пользовательском секторе «гнать гигагерцы» — уже практически бессмысленно. Вычислительные мощности пользовательских компьютеров дошли до той ступени, когда дальнейшее их увеличение просто лишено какой-либо перспективы.

Переход от однородных кремниевых структур к гетеропереходным проводникам, состоящим из слоев различных сред и способным генерировать несколько уровней сигнала вместо привычных «есть» и «нет», – это возможность повысить интенсивность обработки информации без увеличения количества элементов. При этом от двухзначной логики придется перейти к многозначным – трехзначной, четырехзначной и т.д.

Еще одно из возможных направлений развития вычислительной техники будущего – создание так называемых квантовых компьютеров.

Группа исследователей из Австралии предполагает, что троичные данные будут не просто подспорьем, но практической необходимостью в перспективной области квантовых вычислительных систем.

С пришествием квантовых компьютеров троичные вычисления получили новую жизнь. Расширение квантовых компьютеров неизбежно. Группа исследователей под руководством Б.П. Ланьона из Квислендского университета предложила новую методику для ускоренного расширения квантовых компьютеров с использованием хорошо исследованной области троичных вычислений.

# Заключение

# Список использованной литературы