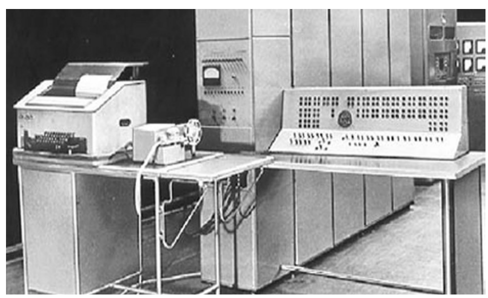
Фалина Ирина Николаевна
Кандидат педагогических наук, доцент кафедры информатики СУНЦ МГУ им. М.В. Ломоносова

Компьютер и фибоначчиева система счисления

Спросите любого старшеклассника: «Какие системы счисления используются для кодирования информации в компьютерах и других цифровых устройствах?». Большинство на этот вопрос ответят: «Двоичная система счисления». И совсем немногие знают, что кроме двоичной в компьютерах можно использовать другие системы счисления.

В компьютерах и иных цифровых устройствах для ввода-вывода часто используется двоично-десятичная система счисления. В 60-х годах XX столетия в МГУ им. М.В. Ломоносова была создана троичная ЭВМ «Сетунь», запущенная потом в серию. Для кодирования информации в этой машине использовалась троичная уравновешенная система счисления: вместо бита использовался трит, вместо привычной двоичной логики – троичная. Разработчик этой ЭВМ Николай Петрович Брусенцов и сейчас работает на факультете вычислительной математики и кибернетики МГУ им. М.В. Ломоносова.

Вопрос любознательному читателю: зачем учёные и инженеры тратят силы и время на создание компьютеров, работающих не на двоичной системе счисления?



ЭВМ «Сетунь»

В каждой области науки и техники существуют фундаментальные идеи или принципы, которые определяют её содержание и развитие. В компьютерной науке роль таких фундаментальных идей сыграли принципы, сформулированные независимо друг от друга двумя крупнейшими учёными XX века – американским математиком и физиком Джоном фон Нейманом и советским инженером и учёным Сергеем Лебедевым.

Центральное место среди принципов Неймана – Лебедева, определяющих архитектуру ЭВМ, занимает предложение об использовании двоичной системы счисления. Это предложение было обусловлено рядом обстоятельств: простота выполнения арифметических операций в двоичной системе счисления; её «оптимальное» согласованием с булевой логикой; простота технической реализации двоичного элемента памяти (триггера).

Однако на определённом этапе развития компьютерной техники выявилось, что использование классической двоичной системы счисления для представления информации в компьютере имеет существенные недостатки.

Первым из них является так называемая *проблема представления отрицательных чисел.*

Второй недостаток двоичной системы счисления получил название *нулевой избыточности.*

Попытки преодолеть эти и другие недостатки двоичной системы счисления стимулировали использование в компьютерах других систем счисления и развитие собственно теории систем счисления.

1. О некоторых недостатках двоичной системы счисления

В десятичной системе счисления для представления отрицательных чисел надо использовать знак числа. А как же иначе, спросите вы? То, что привычно и хорошо для человека, ведущего вычисления на бумаге, не всегда хорошо для компьютера. Для того чтобы сложить два целых числа с разными знаками, и человек, и компьютер, вообще говоря, должны выполнить следующие операции:

1) взять модули слагаемых;
2) сравнить эти модули;
3) запомнить знак большего по модулю слагаемого;
4) из большего модуля вычесть меньший модуль;
5) получившийся результат записать со знаком из п. 3.

Многовато будет для простой и часто используемой операции! Для увеличения скорости работы процессора математики предложили все отрицательные целые числа записывать в дополнительном коде [2, 5], и тогда сложение двух целых чисел стало выполняться за одну операцию (т. е. без анализа знаков слагаемых).

А можно ли записывать целые отрицательные числа без знака, но и без дополнительного кода? Можно! Например, в любой P-ичной уравновешенной системе счисления все целые отрицательные числа записываются без знака [1, 3].

Второй недостаток двоичной системы особенно неприятен при её использовании для кодирования и передачи информации. *Нулевая избыточность* двоичного представления означает, что в системе счисления отсутствует механизм обнаружения ошибок, которые, к сожалению, неизбежно возникают в компьютерных системах.

Суть этой проблемы состоит в следующем. Пусть в процессе передачи или хранения информации, представленной, например, двоичным кодом 10011010, под влиянием внешних или внутренних факторов произошло искажение информации, и она перешла в кодовую комбинацию 11010010 (искажённые разряды выделены). Поскольку комбинация 11010010 (как и любой другой двоичный код) является «разрешённой» в двоичной системе счисления, то без дополнительных действий невозможно определить, произошло искажение информации или нет. Для решения этой проблемы можно, например, для каждого байта (8 разрядов двоичного числа) считать количество единиц или для группы байтов считать контрольную сумму и т. д. В любом случае должны быть использованы специальные методы избыточного кодирования, что замедляет работу компьютера и требует дополнительной памяти.

В условиях, когда человечество всё сильнее зависит от надёжности работы компьютерных систем (управление ракетами, самолётами, атомными реакторами, банковскими системами), вопрос об эффективных механизмах обнаружения ошибок выдвигается на передний план. Ясно, что для компьютеров, основанных на двоичной системе счисления, не всегда можно эффективно решать эту проблему.

В 80-х годах XX столетия группа учёных под руководством профессора Стахова Алексея Петровича из Таганрогского радиотехнического института создала опытный экземпляр помехоустойчивого процессора [4]. Этот процессор мог сам контролировать возникающие в его работе сбои. Для кодирования информации была выбрана фибоначчиева система счисления. Именно её использование позволило построить такой удивительный процессор, названный Фибоначчи-процессор, или Ф-процессор. И хотя успешная попытка построения помехоустойчивого процессора на основе фибоначчиевой системы счисления носила скорее теоретический, чем практический интерес, изучение этой замечательной системы счисления заслуживает внимания.

Наша статья посвящена знакомству с фибоначчиевой системой счисления, правилами записи чисел и выполнения арифметических операций в этой системе счисления. И, наконец, мы покажем, что фибоначчиева система счисления обладает таким важным для двоичного мира свойством, как избыточность.

Прежде чем приступить к изучению фибоначчиевой системы счисления, договоримся о некоторых обозначениях. Вместо длинного (но красивого) названия «фибоначчиева система счисления» будем использовать сокращение ФСС. Для указания, что число записано в ФСС, будем использовать в нижнем индексе сокращение *fib*. Например,

$$10000101_{fib} = 38_{10}.$$

2. Понятие избыточности информационной системы

Известно, что каналы, по которым передаётся информация, практически никогда не бывают идеальными (каналами без помех). Помехи в каналах возникают по разным причинам, но результат воздействия их на передаваемую информацию всегда один – информация теряется (искажается).

Под *потерей* информации мы будем понимать *искажение* информации при сохранении её объёма.

Например, по каналам связи был

передан код доступа к серверу (слово «марс») в виде двоичного кода 10101100 10100000 11100000 1110000. Длина переданного кода (т. е. объём информации) равна четырём байтам, или 32 битам. Сервер получил этот код в виде 10101100 10100100 11100000 11100001.

Проанализируем ситуацию:

1) формально передача завершена успешно, т. к. полученный код является допустимой комбинацией в двоичной системе счисления;

2) длина принятого кода равна длине отправленного кода (при успешной передаче нельзя физически потерять один или несколько битов, можно только изменить содержимое битов);

3) произошло изменение (искажение) содержимого одного бита: с точки зрения пользователя информация при передаче пропала (слово «мдрс» не равно слову «марс»).

В результате пользователю в доступе к серверу отказано!

При передаче информации всегда важно знать ответы на два вопроса:

- произошло ли искажение (потеря) информации?
- если произошло искажение, то в каких битах (разрядах)?

Для ответа на эти вопросы необходимо, чтобы передаваемые коды обладали свойством *избыточности*. Или, что то же самое, символьная система, при помощи которой кодируется информация, должна обладать свойством избыточности.

В информатике под избыточностью понимают следующее: код обладает свойством *избыточности*, если количество бит в нём больше, чем это необходимо для однозначного декодирования (восстановления) исходной информации при условии, что информация передана верно.

Информация кодируется не только в компьютерах. Мы можем считать, что наша письменная речь – это информация, закодированная русскими буквами. Представьте, что вам в руки попадает записка с частично стёртыми буквами:

Вы__т со_т__т__ в н_ч_ле н__бр_.

Подумав, вы сможете восстановить текст этой записки, причём однозначно. А ведь из 27 букв записки 12 букв были уничтожены! Отметим, что для однозначного восстановления потерянной информации важно знать, на каких местах стояли стёртые буквы.

Вы смогли восстановить текст записки потому, что русский язык обладает *избыточностью*. В русском языке существуют слова, однозначно прочитываемые в случае «потери» некоторых букв. Например, С_НТ_БРЬ, Д_Р_ВО.

Имея текст на русском языке с частично «потерянными» буквами, человек, хорошо владеющий русским языком, сможет однозначно восстановить его. Например, вы без труда прочитаете предложение с пропущенными буквами: «Дм_т_ий Ива_ов__ Менд_ле_в – в_л_ки_ рус_кий х_мик». Однако если это предложение будет читать иностранец, едва знающий русский язык и русскую историю, то он не сможет понять, что написано в данном предложении. Мы, носители русского языка, можем с лёгкостью восстановить окончания, пропущенные буквы в слогах, и можем подобрать подходящие слова (из тех, что нам известны). А иностранцу просто не с чем сравнивать получаемую информацию! Таким образом, для носителя языка обычный связный текст на его родном языке содержит избыточную информацию – её можно удалить, но смысл текста (для носителя языка) сохранится.

Избыточностью обладает не только русский язык, но и любой другой язык. Выдающийся американский инженер и математик, основатель теории информации Клод Шеннон проделал такой эксперимент. В литературном английском тексте были удалены 50% букв. Пробелы между словами оставались неизменными, буквы удалялись, конечно, не случайным образом. Например, в английских словах за буквой *q* всегда следует только буква *u*, поэтому «потерянная» буква *u* может быть всегда восстановлена. После «искажения» текст давали читать носителям языка, обладающим достаточно высоким культурным уровнем. Поразительно, но практически все участники этого эксперимента смогли восстановить искажённый текст полностью!

Итак, для обнаружения и восстановления потерянной информации необходимо, чтобы символьная система, в которой происходит кодирование информации, обладала избыточностью. К сожалению, двоичная система счисления не обладает избыточностью, поэтому говорят, что она обладает *нулевой избыточностью*.

О нулевой избыточности двоичной системы счисления инженеры и учёные знают, поэтому для предотвращения потерь информации при передаче были придуманы *избыточные коды* (коды с избыточностью). Существуют избыточные коды с обнаружением (они только обнаруживают ошибку) и коды с исправлением (эти коды обнаруживают место ошибки и исправляют её).

В разных каналах передачи информации возникают разные помехи. Поэтому инженеры и учёные вынуждены разрабатывать различные по своей структуре и избыточности коды. Так, при записи информации на лазерные диски применяют коды с избыточностью по длине в 25%. Примерно с такой же избыточностью применяют коды в системах цифрового спутникового ТВ.

Понятно, что использование избыточных кодов приводит к снижению скорости работы компьютеров, увеличению объёма передаваемых сообщений и, соответственно, снижению скорости передачи информации. Известно, и мы это покажем, что фибоначчиева система счисления обладает избыточностью.

3. Фибоначчиева система счисления

Прежде чем мы приступим к изучению фибоначчиевой системы счисления, напомним основные термины и определения, касающиеся систем счисления в целом.

Все известные системы счисления делятся на позиционные и непозиционные.

Символы, при помощи которых записываются числа, называются *цифрами*, а их совокупность – *алфавитом* системы счисления. Количество цифр, составляющих алфавит, называется его *размерностью*.

Система счисления называется *позиционной*, если количественный эквивалент цифры зависит от её положения в записи числа.

В привычной нам десятичной системе значение числа образуется следующим образом: значение цифры умножается на «вес» соответствующего разряда (1, 10, 100, ...). Например, $6407 = 6 \cdot 1000 + 4 \cdot 100 + 0 \cdot 10 + 7 \cdot 1$.

Последовательность чисел, каждое из которых задаёт «вес» соответствующего разряда, называется *базисом* позиционной системы счисления. Для десятичной системы это 1, 10, 100, ... Если базис позиционной системы образуют члены геометрической прогрессии, а значения цифр есть целые неотрицательные числа, то такие системы счисления называют *традиционными*. Так, базисы десятичной, двоичной и восьмеричной систем счисления образуют геометрические прогрессии со знаменателями 10, 2 и 8 соответственно.

Но среди позиционных систем счисления есть и такие, базис которых – не геометрическая прогрессия. К таким необычным позиционным системам счисления относится и *фибоначчиева система*.

Числа Фибоначчи – элементы числовой последовательности 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946,..., в которой каждое последующее число, начиная с третьего, равно сумме двух предыдущих чисел.

Формально числа Фибоначчи, определяются так: $F_0 = 1$, $F_1 = 1$, $F_n = F_{n-2} + F_{n-1}$, $n \geq 2$. Последовательность, известная у нас как числа Фибоначчи, использовалась в древней Индии задолго до того, как стала известна в Европе. На Западе эта последовательность была исследована Леонардо Пизанским, известным как Фибоначчи (1170–1250). Благодаря книге Фибоначчи «Liber Abaci» Европа познакомилась с индоарабской системой чисел, которая позднее

Леонардо Пизанский Фибоначчи

вытеснила традиционные для того времени римские числа. Фибоначчиева система счисления относится к позиционным системам. Алфавитом фибоначчиевой системы являются цифры 0 и 1, а её базисом – последовательность чисел Фибоначчи 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

Обратите внимание, что $F_0 = 1$ в базис не включается.

Перечислим некоторые числа в двоичной и фибоначчиевой системах счисления.

Десятичное число	Запись числа в двоичной системе счисления	Запись числа в фибоначчиевой системе счисления
1	1	1
2	10	10
3	11	100, или 11
4	100	101
5	101	1000, или 110
10	1010	10010, или 1110
20	10100	101010
50	110010	10100100, или 10011100, или 10100011, или 10011011

Фибоначчиева система является разновидностью двоичной системы – её алфавит составляют цифры 0 и 1. Эту неклассическую двоичную систему счисления можно использовать для кодирования информации в компьютере, так как элементная база современной компьютерной техники ориентирована на обработку двоичных последовательностей.

4. Алгоритмы перевода целых чисел из ФСС в десятичную систему и обратно

В P-ичных системах счисления базис является геометрической прогрессией. Вклад в значение числа цифры a, стоящей в k-ом разряде, равен $a \cdot P^{k-1}$, где P – основание системы счисления. Часто говорят, что «вес» k-ого разряда равен P^{k-1}.

Система счисления	**Основание**	**Количество цифр**	**Цифры**	**Базис**
Десятичная	10	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	1, 10, 10^2, 10^3, 10^4 и т. д.
Двоичная	2	2	0, 1	1, 2, 2^2, 2^3, 2^4 и т. д
Троичная	3	3	0, 1, 2	1, 3, 3^2, 3^3, 3^4 и т. д
Восьмеричная	8	8	0, 1, 2, 3, 4, 5, 6, 7	1, 8, 8^2, 8^3, 8^4 и т. д
Шестнадцатеричная	16	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	1, 16, 16^2, 16^3, 16^4 и т. д.

В ФСС вес каждого разряда числа также определяется базисом этой системы. Для удобства дальнейшей работы выпишем веса первых 10 разрядов ФСС (нумерацию разрядов ведём справа налево, начиная с первого). Такая нумерация разрядов удобна, поскольку в качестве веса k-го разряда используется k-ое число Фибоначчи.

разряд	10	9	8	7	6	5	4	3	2	1
вес	**89**	**55**	**34**	**21**	**13**	**8**	**5**	**3**	**2**	**1**

Пример 1. Пусть нам дано число $A_{fib} = 101010_{fib}$, записанное в фибоначчиевой системе счисления. Чему равно это число в десятичной системе счисления?

Чтобы ответить на этот вопрос, запишем цифры числа в разрядную сетку, затем умножим каждую цифру на вес разряда и сложим полученные числа. Такая процедура называется *разложением по базису*. Так как цифрами фибоначчиевой системы счисления являются 0 и 1, то нам достаточно сложить веса тех разрядов, где стоят единицы.

разряд	10	9	8	7	6	5	4	3	2	1
вес	**89**	**55**	**34**	**21**	**13**	**8**	**5**	**3**	**2**	**1**
					1	0	1	0	1	0

Получим: $A_{fib} = 101010_{fib} = 1\cdot13+0\cdot8+1\cdot5+0\cdot3+1\cdot2+0\cdot1 = 13+5+2 = 20_{10}$.

Алгоритм перевода целых чисел из фибоначчиевой системы счисления в десятичную.

1. Напишем над каждой цифрой в фибоначчиевой записи числа вес соответствующего разряда, начиная с младшей цифры числа.

2. Сложим все числа, стоящие над единицами. Полученное число будет десятичным эквивалентом фибоначчиева числа.

Пример 2. Решим обратную задачу. Запишем в фибоначчиевой системе счисления десятичные числа 10_{10}, 25_{10} и 100_{10}.

$$10_{10} = 1110_{fib} = 1\cdot5+1\cdot3+1\cdot2+0\cdot1$$

(выполнили разложение по базису).

Однако это же число 10_{10} можно записать в фибоначчиевой системе счисления и по-другому:

$$10_{10} = 10010_{fib} = 1\cdot8+0\cdot5+0\cdot3+1\cdot2+0\cdot1.$$

Аналогично, число 25_{10} можно также записать двумя способами:

$$25_{10} = 1000101_{fib} = 110101_{fib}.$$

А вот число 100_{10} можно записать уже 6-ю способами:

$$100_{10} = 1000010011_{fib} =$$
$$= 1000010100_{fib} =$$
$$= 110010011_{fib} = 110010100_{fib} =$$
$$= 101110011_{fib} = 101110100_{fib}.$$

Алгоритм перевода целых чисел из десятичной системы счисления в ФСС.

1. Найдём максимальное число F_k из базиса фибоначчиевой системы, которое умещается в наше число A.

2. Положим $A = A - F_k$. Искомая запись числа A в фибоначчиевой системе будет содержать k цифр. Сформируем «заготовку» искомого фибоначчиева числа в виде $1\underbrace{0\ldots0}_{k-1}$.

3. Если полученное значение A равно 0, переход на п. 9.

4. Иначе положим $k = k-1$.

5. Найдём такое F_p $(p \le k)$, которое умещается в число A.

6. В сформированную заготовку на p-ое место поставим 1.

7. Положим $k = k-1$. Вычислим разность $A = A - F_p$.

8. Если $A \ne 0$, то переход на п. 5.

9. Сформированная «заготовка» является искомым числом в ФСС. Конец алгоритма.

Продолжение следует.

Литература

1. *Андреева Е., Фалина И.* Системы счисления и компьютерная арифметика. – М.: БИНОМ. Лаборатория знаний, 2004.

2. *Андреева Е., Босова Л., Фалина И.* Математические основы информатики. – М.: БИНОМ. Лаборатория знаний, 2007.

3. *Касаткин Н.В.* Новое о системах счисления. – Киев.: Вища школа, 1982.

4. Помехоустойчивые коды: Фибоначчи-компьютер. – Сб. статей, серия «Радиоэлектроника и связь» – М.: Знание, 1989.

5. *Филиппов К.С.* Научили железку считать… //Потенциал. – 2011. – №5.