Лекция 3. Представление информации в ЭВМ

Любая информация в ЭВМ, включая текст, аудио и видео, представляется в виде последовательности нулей и единиц. Используя определённые правила, двоичные разряды формируются в числа, которые, в свою очередь, преобразуются в нужную для человека форму (текст, звук, изображение и т.п.). Важным этапом в процессе изучения принципов работы ЭВМ является понимание того, каким же образом внутри них хранится информация.

3.1 Краткая история возникновения чисел, алгебры и систем счисления

Потребность человечества в числах определялась необходимостью счёта и измерения, возникавшей в непосредственной практической деятельности. Затем число становится основным понятием математики, и дальнейшее развитие этого понятия определяется потребностями этой науки.

Первоначально числа обозначались чёрточками на материале, служащем для записи (папирус, глиняные таблички и т.д.). Числа соотносились с конкретными предметами, например, три камня, четыре палочки и т.п. Арифметические операции являлись действиями по объединению нескольких совокупностей предметов в одну или разделения одной совокупности на части. Лишь в многовековом опыте сложилось представление об отвлечённом характере этих действий, о независимости количественного результата действия от природы предметов, составляющих совокупности, о том, что, например, два предмета и три предмета составят пять предметов независимо от природы этих предметов. Тогда стали разрабатывать правила действий, изучать их свойства, создавать методы для решения задач, т.е. начинается развитие науки о числах – арифметики.

Со временем потребность человека в арифметических операциях возрастала. Для их выполнения были разработаны различные приспособления, одними из которых являются электронные вычислительные машины.

3.2 Системы счисления

Для записи чисел человечеством придуманы различные правила, называемые *системами счисления*. По этим правилам любое число представляется в виде набора специальных символов – *цифр* (от араб. сифр – нуль, буквально – пустой; арабы этим словом называли знак отсутствия разряда в числе). Получение количественного эквивалента числа осуществляется по *алгоритму замещения*, согласно которому сначала цифры заменяются их количественными эквивалентами, а затем эквивалент числа получается путём арифметических операций над эквивалентами цифр.

В зависимости от того, меняет ли свое количественное значение цифра при разном положении в числе, системы счисления можно классифицировать как *непозиционные* и *позиционные* системы.

В *системах счисления первого типа (непозиционных)* число образуется из цифр, значение которых не изменяется при различном положении цифр в числе. Примером таких систем служит *римская* система счисления. В ней в качестве цифр для составления чисел используются буквы латинского алфавита. I

означает единицу, V – пять, X – десять, L – пятьдесят, C – сто, D – пятьсот, M – тысячу. Для получения числа требуется просто просуммировать количественные эквиваленты входящих в него цифр, с учётом того, что если младшая цифра идет перед старшей цифрой, то она входит в сумму с отрицательным знаком. Например, DLXXVII = пятьсот + пятьдесят + десять + десять + пять + один + один = пятьсот семьдесят семь. Или CDXXIX = минус сто + пятьсот + десять + десять + минус один + десять = четыреста двадцать девять.

В *позиционных системах счисления* количественное значение цифры определяется её позицией в числе. Номер позиции называется *разрядом*. Число цифр, используемых для представления чисел, называется *основанием*. Количественное значение числа в позиционной системе счисления, состоящего из n цифр $\{a_i\}$, $i \in \overline{0,n-1}$ (т.е. числа, имеющего вид $a_{n-1}a_{n-2}\ldots a_1a_0$), может быть получено следующим образом:

$$A_p = a_{n-1}p^{n-1} + a_{n-2}p^{n-2} + \text{K} + a_1p^1 + a_0p^0, \qquad (1)$$

число n называется *разрядностью* и определяет максимальный эквивалент, который можно получить для такого числа:

$$A_p^{max} = p^n.$$

В силу того, что ЭВМ строятся на базе логических схем, которые могут иметь только два состояния – включено и выключено, то все числа в них представлены в *двоичной системе счисления*, которая по своей сути является позиционной. Набор цифр в этой системе состоит из 0 и 1 (основание равно 2). Например, число в двоичной системе может иметь вид $10100111_{(2)}$. Количественный эквивалент такого числа равен ста шестидесяти семи ($167_{(10)}$).

Очевидно, что для человека выполнение арифметических операций с двоичными числами затруднительно и неестественно. Поэтому для ввода и вывода чисел используются другие системы счисления. Наибольшее распространение получили восьмеричная, десятичная и шестнадцатеричная системы счисления и представление целых чисел в двоично-десятичной форме (англ. Binary Coded Decimal, BCD).

В *десятичной системе счисления* набор цифр включает 0, 1, 2, 3, 4, 5, 6, 7, 8 и 9. Например, число $10_{(10)}$ имеет количественный эквивалент десять ($1\cdot10^1 + 0\cdot10^0$). Эта система используется нами в повседневной жизни. Такое распространение она получила из-за того, что первоначально счёт предметов производился с использованием пальцев на человеческих руках. Как известно, у обычного человека на руках ровно десять пальцев.

В *восьмеричной системе счисления* набор цифр включает 0, 1, 2, 3, 4, 5, 6 и 7. Например, число $77_{(8)}$ имеет количественный эквивалент шестьдесят три ($7\cdot8^1 + 7\cdot8^0$), а $10_{(8)}$ равно восьми ($1\cdot8^1 + 0\cdot8^0$).

В *шестнадцатеричной системе счисления* набор цифр включает арабские цифры от 0 до 9 и 6 букв латинского алфавита – A (десять), B (одиннадцать), C (двенадцать), D (тринадцать), E (четырнадцать), F (пятнадцать). Например, число $FF_{(16)}$ имеет количественный эквивалент двести пятьдесят пять ($15\cdot16^1 + 15\cdot16^0$), а $100_{(16)}$ равно двумстам пятидесяти шести ($1\cdot16^2 + 0\cdot16^1 + 0\cdot16^0$).

Двоично-десятичные числа – это специальный вид представления числовой информации, в основу которого положен принцип кодирования каждой десятичной цифры группой из четырёх бит. При этом каждый байт содержит одну или две цифры. Первый способ называется *неупакованное число*, второй – *упакованное*. Например, число 3456 может быть записано как неупакованное в виде $00000011\ 00000100\ 00000101\ 00000110_{(2)}$, или как упакованное – $0011\ 0100\ 0101\ 0110_{(2)}$.

3.3 Перевод чисел из одной системы счисления в другую

Для выполнения арифметических операций над числами они должны быть представлены в одной системе счисления.

Перевод чисел из *любой системы счисления в десятичную систему* осуществляется простым получением их количественных эквивалентов и записи в виде десятичного числа.

Перевод чисел из *десятичной системы в любую другую* осуществляется путём деления исходного числа на основание требуемой системы и записи остатков от деления в обратном порядке. Например, смотрите рисунок 17.

Перевод числа из шестнадцатеричной системы в двоичную систему может быть осуществлён путём представления каждой его цифры в виде двоичной тетрады и последовательной записи этих тетрад. Например, число $FF_{(16)}$ представляется как $1111\ 1111_{(2)}$. А число $3A_{(16)}$ – как $0011\ 1010_{(2)}$. Соответственно, перевод числа из двоичной системы в шестнадцатеричную систему осуществляется путём деления его на тетрады и представления каждой тетрады в виде шестнадцатеричной цифры. Например, $10111001_{(2)}$ представляется как $1011\ 1001_{(2)}$ и в шестнадцатеричной системе имеет вид $B9_{(16)}$.

Перевод числа из восьмеричной системы в двоичную систему осуществляется аналогично переводу числа из шестнадцатеричной системы, только цифры заменяются не тетрадами, а двоичными триадами. Например, $77_{(8)}$ будет представлено как $111\ 111_{(2)}$, а число $10_{(8)}$ – как $001\ 000_{(2)}$. Двоичное число при переводе в восьмеричную систему счисления сначала делится на триады, а затем каждая триада представляется в виде восьмеричной цифры. Например, $11100111_{(2)}$ делится на $011\ 100\ 111_{(2)}$ и получается $347_{(8)}$.

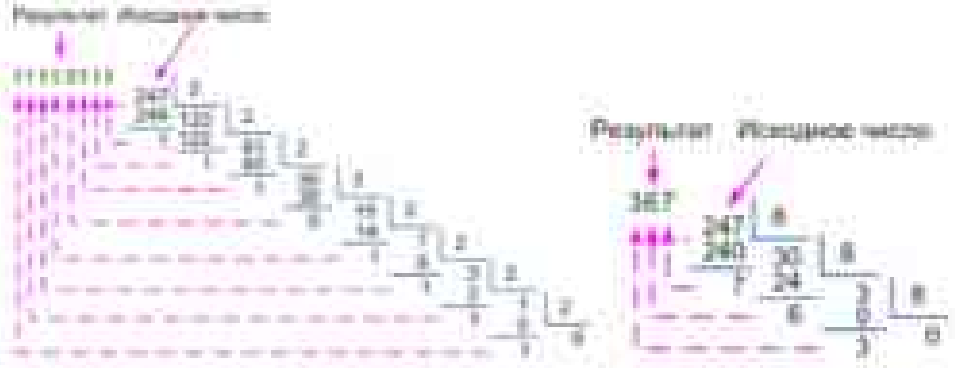

Рисунок 12 - Перевод числа 247 из десятичной системы счисления в двоичную (слева) и восьмеричную (справа) системы

Перевод чисел из шестнадцатеричной системы счисления в восьмеричную систему и наоборот осуществляется в два этапа. Сначала исходное число

переводится в двоичную или десятичную систему, а затем из двоичной или десятичной системы число переводится в требуемую систему счисления. Например, число $FF_{(16)}$ в двоичной системе имеет вид $1111\ 1111_{(2)}$, и в восьмеричной системе оно примет вид $377_{(8)}$.

3.4 Представление отрицательных целых и вещественных чисел в ЭВМ

Отрицательные целые числа в ЭВМ представляются в специальном виде, называемом *дополнительным кодом*, который позволяет при выполнении операций исключить отличия отрицательных чисел от положительных.

Дополнительный код отрицательного числа представляет собой результат инвертирования (замены в числе нулей на единицы и наоборот) каждого бита двоичного числа (модуля отрицательного числа) и прибавления к нему единицы.

Обратное преобразование числа из дополнительного кода в обычный вид осуществляется аналогичным образом (сначала инвертируется, затем прибавляется 1).

Например, рассмотрим число $-185_{(10)}$. Его модуль в двоичном виде имеет вид $-10111001_{(2)}$. Чтобы его перевести в дополнительный код, надо произвести его инвертирование. Причём инвертируется вся ячейка памяти, отведённая под число (будем считать, что мы работаем с 16-разрядными ячейками). В результате получится число $1111111101000110_{(2)}$. Добавляя к нему единицу, получаем число в дополнительном коде $1111111101000111_{(2)}$.

Если требуется определить, является ли число отрицательным, то необходимо проанализировать его старший разряд. Если он равен 1, то число отрицательное, если 0 – то положительное. Если считать, что результат предыдущего примера только положительный, то в десятичном виде он равен 65351. Именно поэтому изменяется диапазон отрицательных чисел, которые можно записать в n-разрядную ячейку. Например, в 8-разрядную ячейку можно записать целые положительные числа из диапазона от $0_{(10)}$ до $255_{(10)}$, а целые отрицательные из диапазона $-128_{(10)}$ до $127_{(10)}$.

Вещественные числа в ЭВМ могут быть представлены в форме с фиксированной или плавающей запятой.

Представление числа *в форме с фиксированной запятой*, которую иногда называют также *естественной формой*, включает в себя знак числа и его модуль в q-ичном коде. Здесь q – это *основание системы* или *база*. В ЭВМ чаще всего используется двоичная система, однако существуют ещё 8- и 16-ричные формы. Числам с фиксированной запятой соответствует запись вида $a_{n-1}a_{n-2}\text{K}\,a_1a_0a_{-1}a_{-2}\text{K}\,a_{-m+1}a_{-m}$. При этом положение запятой никак не фиксируется, а лишь подразумевается в процессе выполнения арифметических операций. Точность числа в формате с фиксированной запятой определяется числом разрядов, отводимых под дробную часть.

Получить количественный эквивалент числа с фиксированной запятой можно по формуле:

$$A_{(q)} = a_{n-1}p^{n-1} + a_{n-2}p^{n-2} + \text{K} + a_1p^1 + a_0p^0 + a_{-1}p^{-1} + a_{-2}p^{-2} + \text{K} + a_{-m}p^{-m},$$

Перевод из двоичной системы счисления в шестнадцатеричную осуществляется аналогично целым числам – деление на тетрады и представление каждой тетрады в виде шестнадцатеричной цифры. Деление на тетрады осуществляется от запятой (влево для целой части и вправо для дробной).

Перевод числа с фиксированной запятой из десятичной системы в двоичную осуществляется в два этапа. На первом этапе переводится целая часть обычным образом. На втором этапе переводится дробная часть умножением её на 2 и выделением целой части на каждом шаге (рис. 13). Умножение производят до тех пор, пока не будет достигнута требуемая точность (будет получено требуемое количество разрядов после запятой) или при очередном умножении получается дробная часть, равная нулю.

Вещественные числа в формате с плавающей запятой представляются в виде двух групп цифр – мантиссы и порядка. Число представляется в виде произведения $X = \pm m q^{\pm p}$, где m – мантисса числа X, q – основание системы счисления, p – порядок числа. Форму записи чисел с плавающей запятой также называют *нормальной*. Точность числа в формате с плавающей запятой зависит от числа разрядов, отводимых под мантиссу и порядок.

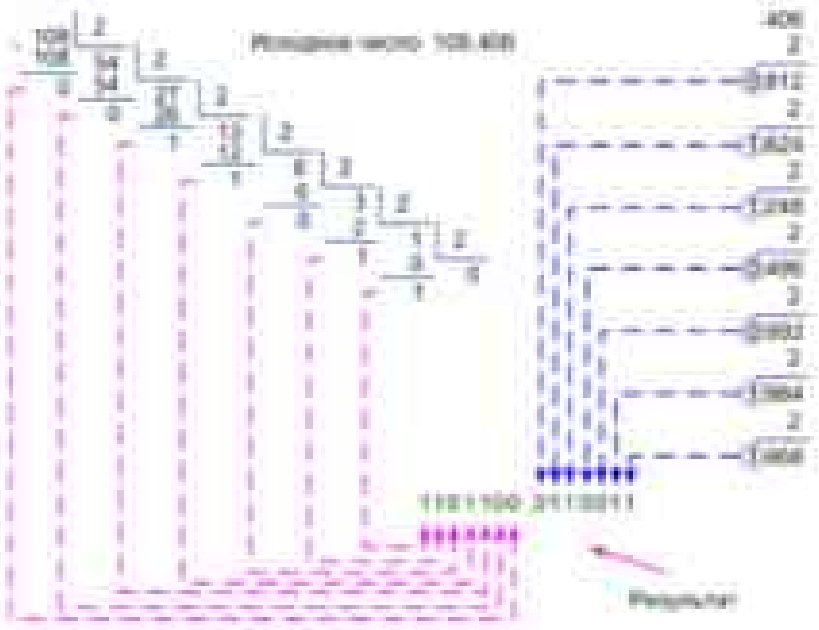

Рисунок 13 - Перевод числа 108,406 из десятичной системы в двоичную систему счисления

Для представления чисел с плавающей запятой в ЭВМ был разработан стандарт IEEE 754, согласно которому выделяют два типа чисел: 32-битовое с 8-ю разрядами под порядок и 64-битовое с 11-ю разрядами под порядок. Первый тип называется одинарным или простым, второй – двойным или двойной точностью.

3.5 Представление символьной информации в ЭВМ

Каждому символу однозначно сопоставляется некоторая двоичная последовательность, называемая его **кодом**. Совокупность возможных символов и их кодов образуют **таблицу кодировки**.

В настоящее время применяется огромное количество различных кодировок символов. Общим (хотя и не обязательным) для всех систем кодирования является **весовой принцип**, согласно которому коды цифр (т.е. двоичные числа) увеличиваются по мере увеличения цифры, а коды латинских букв увеличиваются в алфавитном порядке. Например, код символа '1' на единицу меньше кода символа '2', а код символа 'b' на единицу меньше кода символа 'c'. Требований к порядку расположения специальных символов (символов национальных языков, знаков препинания, математических символов и т.п.) обычно не предъявляется.

3.5.1 Кодировочные таблицы с фиксированной длиной кода

Самыми первыми и наиболее распространёнными являются кодировки, представляющие символы восьмиразрядными двоичными числами. При этом в таблице может содержаться не более 256 кодов различных символов. Примерами таких таблиц являются:

- расширенный двоично-кодированный код (англ. Extended Binary Coded Decimal Interchange Code, EBCDIC). Также он известен под названием ДКОИ. Кодировка EBCDIC используется в ЭВМ, производимых фирмой IBM;
- американский стандарт кода для представления информации (англ. American Standard Code for Information Interchange, ASCII), разработанный Институтом стандартизации США (ANSI).

Таблица ASCII делится на две части: базовую и расширенную. Базовая таблица закрепляет значение кодов от 0 до 127 (т.е. использует только 7 бит), а расширенная относится к символам с номерами от 128 до 255. Изначально существовала только базовая часть таблицы ASCII, а старший (восьмой) бит использовался для контроля чётности (т.е. принимал единичное значение, если в 7-битной части чётное число единиц). Основная таблица описывает 128 символов, из которых (рис. 14):

- первые 32 кода отданы производителям аппаратурных средств (в первую очередь производителям печатающих устройств). В этой области размещаются так называемые управляющие коды, которым не соответствуют никакие символы языков, и эти коды не выводятся ни на экран, ни на устройства печати, но ими можно управлять тем, как производится вывод прочих данных;
- коды с 32 по 127 описывают символы английского алфавита, знаков препинания, цифр, арифметических действий и некоторых вспомогательных символов.

Стандарт ASCII с 8 битами не определяет содержание верхней половины таблицы кодировки, которая используется разработчиками разных стран для представления символов соответствующих языков. Однако, для того чтобы обеспечить единообразие правил представления символов этой части кодовой таблицы, Международная организация по стандартизации (ISO) взяла ответственность по определению семейства стандартов, известных как семейство ISO 8859-X. Это семейство представляет собой совокупность 8-битных кодировок, где

младшая половина каждой кодировки (символы с кодами 0–127) соответствует ASCII, а старшая половина определяет символы для различных языков. В зависимости от использования кодов 128–255 различают следующие варианты стандарта ISO 8859 (табл. 1).

Рисунок 14 - Таблицы символов

Таблица 1 - Варианты стандарта ISO 8559-X

Стандарт	Характеристика
ISO 8859-0	Новый европейский стандарт (так называемый Latin-0)
ISO 8859-1	Языки Западной Европы и Латинской Америки (Latin-1)
ISO 8859-2	Языки стран Центральной и Восточной Европы
ISO 8859-3	Языки стран Южной Европы, мальтийский и эсперанто
ISO 8859-4	Языки стран Северной Европы
ISO 8859-5	Языки славянских стран с символами кириллицы
ISO 8859-6	Арабский язык
ISO 8859-7	Современный греческий язык
ISO 8859-8	Языки иврит и идиш
ISO 8859-9	Турецкий язык
ISO 8859-10	Языки стран Северной Европы (лапландский, исландский)
ISO 8859-11	Тайский язык
ISO 8859-13	Языки балтийских стран
ISO 8859-14	Кельтский язык
ISO 8859-15	Комбинированная таблица для европейских языков
ISO 8859-16	Специфические символы для языков: албанского, хорватского, английского, финского, французского, немецкого, венгерского, ирландского, итальянского, польского, румынского и словенского

Несмотря на то, что в ISO разработали стандарты для представления языков многих стран, разработчики программного обеспечения предпочитают пользоваться другими (собственными) кодировочными таблицами.

Одной из альтернатив стандарту ISO 8859-5 стала кодовая таблица KOI8, разработчики которой поместили символы русской кириллицы в верхней части расширенной ASCII-таблицы таким образом, что позиции кириллических символов соответствуют их фонетическим аналогам в английском алфавите в нижней части таблицы. Это означает, что если в тексте, написанном в KOI8, убрать восьмой бит каждого символа, то текст останется читаемым, хотя и выглядеть будет забавно. Например, слово «привет» будет выглядеть как «privet». Такая кодировка широко используется (например, при передаче SMS-сообщений). Следует отметить, что KOI8-R подходит только для русских текстов, и как следствие был создан украинский вариант KOI8-U.

Компания Microsoft в своих операционных системах MS-DOS и MS Windows использует свои кодовые страницы, называемые OEM (Original Equipment Manufacturer) (табл. 2):

Таблица 2 - Наиболее распространенные страницы OEM

CP437	США, страны Западной Европы и Латинской Америки
CP708	Арабские страны
CP737	Греция
CP866	Российская кодировка для MS-DOS
CP932	Япония
CP936	Китай
CP1251	Российская кодировка для MS Windows

К сожалению, стандарты ISO, KOI8 и OEM хотя и предназначены для одного и того же (кодирования символов национальных языков), но не согласованы между собой. Поэтому при передаче информации необходимо чётко оговаривать, с использованием какой таблицы она закодирована.

3.5.2. Универсальная кодовая таблица и системы кодирования символов

Стандарты кодирования символов с фиксированной длиной кода (ASCII и все аналогичные ему стандарты) в силу 8-битной кодировки имеет существенные ограничения по числу символов, которые могут быть закодированы. Огромное разнообразие таких кодовых таблиц привело к хаусу и полной неразберихе, которая дала понять, что необходимо как-то унифицировать кодирование символов в ЭВМ.

По этой причине в 1989 году международная организация по стандартизации (ISO) начала разрабатывать стандарт ISO/IEC 10646, получивший название Универсальной системы кодирования символов (Universal Coded Character Set[1]). В этом стандарте записываются все существующие символы и знаки и каждому

[1] Список Unicode символов - https://symbl.cc/ru/unicode-table/

из них присваивается некоторый универсальный код. Кодовое пространство разбито на 17 плоскостей (англ. planes) по 2^{16} (65 536) символов. Итого стандарт позволяет закодировать 1 114 112 символов (пока этого достаточно ☺). Плоскость с номером 0 называется базовой (Basic Multilingual Plane, BMP) и содержит символы наиболее употребительных письменностей.

Для описания кода символа стали использовать общепринятую нотацию с префиксом U+ и с указанием кода символа в шестнадцатеричной форме. При этом, для символов базовой плоскости формат записи кода символа будет выглядеть как U+0030 (символ цифры 0) или U+0412 (заглавная буква В) и т.п. Для кодовой плоскости 1 запись кода принимает вид от U+10000 до U+FFFFF, и т.д. Система кодирования развивается бурно и на сегодня уже выпущено более 20 версий этого стандарта (на конец 2023 года версия стандарта – 15.1.0).

Параллельно разработкой универсальной кодовой таблицей не утихал спор на тему «а как же представлять коды символов в памяти ЭВМ». Очевидным стал тот факт, что кодирование символов с фиксированной длиной кода на практике продемонстрировало свою неэффективность. В результате появились правила форматирования кодов символов (англ. Unicode Transformation Format). Совокупность универсальной таблицы символов и правил форматирования кодов символов стали называть одним словом – **Unicode**.

Наибольшую популярность приобрели правила UTF-8. Согласно этим правилам, универсальный код символа в зависимости от его значения представляется последовательностью из 1, 2, 3 или 4 байтов. При этом, в старших разрядах первого байта кода содержится подсказка, из скольки байт состоит эта последовательность, а все остальные байты последовательности в своих двух старших разрядах содержат префикс 10.

Количество требуемых байт определяется по следующему правилу:

Диапазон кодов универсальных символов	Количество байт	Префикс первого байта
00000000 -0000007F	1	0xxxxxxx
00000080-000007FF	2	110xxxxx
00000800-0000FFFF	3	1110xxxx
00010000-0010FFFF	4	11110xxx

Такая система правил позволяет получить совместимость с базовой таблицей ASCII (коды символов из этой таблицы в USC находятся в самом начале и их коды совпадают). Под символы кириллицы в UCS выделены области знаков с кодами от U+0400 до U+052F, от U+2DE0 до U+2DFF, от U+A640 до U+A69F.

Рассмотрим пример, как кодируются символы в UTF-8. Например, символ Ю (большая кириллическая буква Ю) имеет код U+042E. Код символа попадает во второй диапазон, поэтому будет записываться двумя байтами (рис. 15).

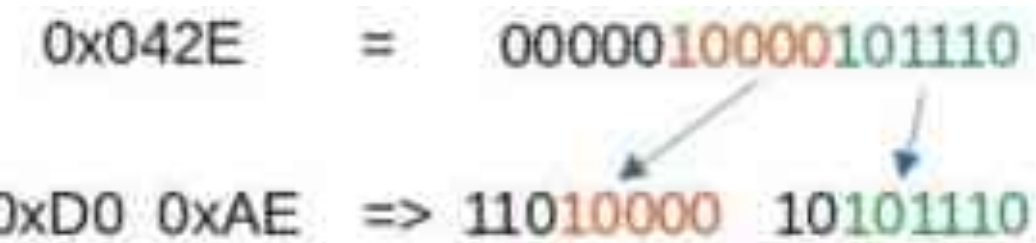

Рисунок 15 - Пример кодирования универсального символа в UTF-8.

Для того, чтобы понять, какая система правил форматирования символов использовалась для создания текстового файла, в самом начале файла записывается «магическая последовательность», указывающая на соответствующую UTF. Для UTF-8 такой последовательностью являются три байта со значениями: 0xEF 0xBB 0xBF.

3.6 Вывод символьной информации. Шрифты

При кодировании символьной информации в ЭВМ никак не описывается, как будет выглядеть каждый символ на экране или на бумаге. А только лишь определяются соглашения вида «если это число X, то будем понимать его как символ «буква а», а если это число Z, то будем понимать его как символ «запятая» и т.д. Для того чтобы описать, как должны выглядеть символы на экране или на бумаге, используются дополнительные правила – **шрифты**, в которых для каждого числа однозначно определяется вид соответствующего символа.

Шрифты бывают *растровые* и *векторные*. В первом случае в памяти ЭВМ хранится образ символов (растр), который при необходимости выбирается из неё и выводится пользователю. Растр (рис. 16) – это матрица определённого размера, в которой в тех ячейках, которые должны быть закрашены, помещается 1, а в остальных – 0. Во втором случае в памяти ЭВМ хранятся команды, которые надо выполнить устройству отображения, чтобы вывести требуемый символ.

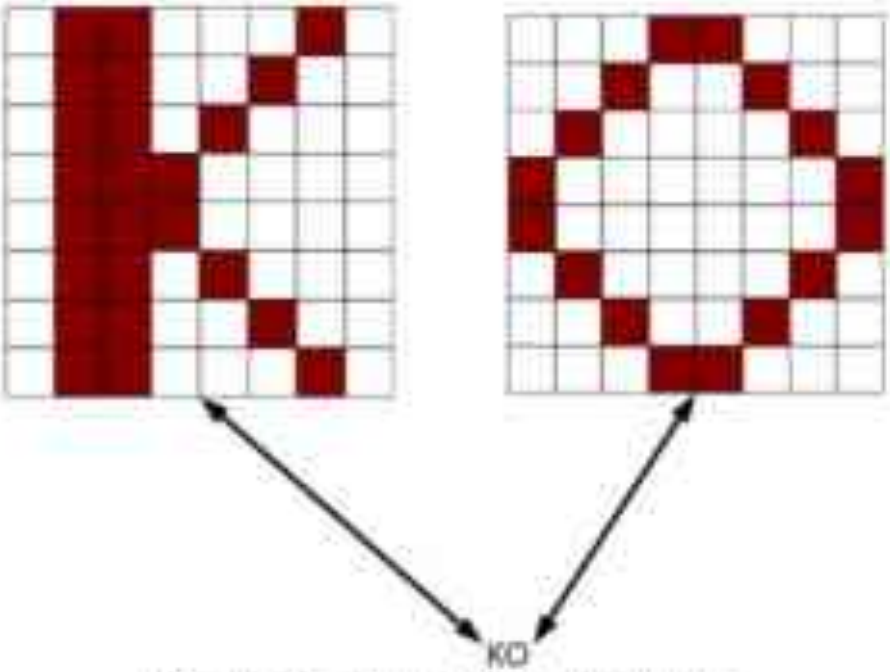

Рисунок 16 - Пример растрового шрифта

В текстовом режиме вся область для вывода информации поделена на ячейки, называемые **знакоместом**. В каждую ячейку может быть выведен только один символ.

Контрольные вопросы

1. Что такое система счисления? Назовите отличия позиционных систем счисления от непозиционных. Приведите примеры систем счисления обоих видов.
2. Как перевести число из двоичной системы счисления в десятичную и наоборот? Приведите примеры.
3. Как перевести число из восьмеричной системы счисления в десятичную и наоборот? Приведите примеры.
4. Как перевести число из шестнадцатеричной системы счисления в десятичную и наоборот? Приведите примеры.
5. Как перевести число из шестнадцатеричной системы в двоичную и наоборот? Приведите примеры.
6. Как перевести число из восьмеричной системы в двоичную и наоборот? Приведите примеры.
7. Как перевести число из шестнадцатеричной системы в восьмеричную и наоборот? Приведите примеры.
8. Как представляются отрицательные числа в ЭВМ? Что представляет собой дополнительный код? Приведите примеры перевода отрицательных десятичных чисел в двоичную систему счисления.
9. Расскажите о представлении вещественных чисел в ЭВМ. В чем отличие представления таких чисел в форме с фиксированной и плавающей запятой?
10. Как представляется символьная информация в ЭВМ? Какие виды кодировок символов вы знаете?
11. Объясните отличия стандарта ASCII и ISO 8859-X.
12. Какие кодовые страницы используются в операционных системах компании Microsoft?
13. Что такое шрифт? Какие виды шрифтов вы знаете?
14. Какие типы данных используются в языке программирования Си для хранения переменных?
15. Что такое двоичный флаг? Каково его назначение?
16. Какие поразрядные операции существуют в языке программирования Си?
17. Что такое маска и маскирование?