

Тема 4. Стандарти за кодиране на информацията

Една от най-важните дейности на човечеството и основа, върху която то съществува, е обменът на информация.

Информацията се съдържа в съобщенията, разглеждани като редици от знаци или символи. При предаването на съобщения – на разстояние за кратко време, възниква проблемът за осъществяване на предаването. Този проблем е стоял пред хората от незапомнени времена и се е решавал в една или друга степен, с едни или други средства: чрез дима или светлината от запалени огньове, чрез моряшките флагове, чрез пощата, телефона и телеграфа. Да разгледаме по-подробно работата на телеграфа. Предназначеното за предаване съобщение – буквен или цифров текст, се предава на телеграфния предавател, където на всеки знак от съобщението се съпоставят определени комбинации от електрически сигнали. Те се изпращат по телеграфната линия (електрически проводник или радиовръзка) до приемателния пункт и там телеграфният приемник обратно възстановява знаците на предаването съобщение.

Изобщо, когато знаците на едно съобщение се заменят с други знаци или техни съчетания, това действие се нарича кодиране, а съответствието, по което едните знаци се заменят с други – код.

Кодирането в телеграфията е едно от най-известните и се осъществява във форми с различни кодове.

Един много известен код носи името на американския художник и изобретател Семюел Морз и е разработен от него през 1837 – 1838 г. Приспособен е за различни азбуки и се употребява и днес. В кода на Морз на всяка буква от азбуката, на всяка цифра или препинателен знак се съпоставят съчетания от два вида електрически сигнали – дълъг сигнал или тире и къс сигнал или точка, и за първите пет наши букви е прието съответствието:

А.-; Б-...; В.--; Г --.; Д-..

Тъй като броят на точките и тиретата за всички букви не е еднакъв, за да се различават буквите, между съответните им съчетания трябва да има по-голяма пауза, отколкото между елементите на съчетанията. Това е един от недостатъците на кода на Морз.

По-съвършен код предлага през седемдесетте години на миналия век френският изобретател Жан Бодо. В кода на Бодо буквите или другите символи получават номера от 1 до 32 и на всеки номер се съпоставят по пет символа от два вида, например + и -, които се изпращат по телеграфната линия с два вида електрически сигнали през равни интервали от време. Това става най-лесно с излъчване на импулс в даден момент – за единия символ, и липса на импулс в този момент - за другия. Ясно е, че тези два импулса е най-добре да означим съответно с цифрите 1 и 0. Така по естествен път чрез кода на Бодо се въвежда двоичната бройна система. Петсимволните комбинации се записват с цифрите 0 и 1 и отговарят точно на числата от 0 до 31 включително, записани в позиционна двоична бройна система с петцифрен запис, а именно

0=00000

1=00001

2=00010
3=00011
.....
31=11111

В наше време телеграфът се прилага за изпращане не само на кратки съобщения от рода на телеграмите, но и за изпращане съдържанието на цели вестници или книги, за тяхното едновременно печатане на отдалечени места. Нещо повече, по телеграфа се изпращат и снимки. Това става по следния начин: Снимката се разглежда като съставена от точки или петна с малки размери. Всяка точка има цвят с определена интензивност. За простота да разгледама черно-белите снимки. Черният цвят на една точка може да бъде от едва забележим до силно черен. При грубо предаване всяка точка можем да приемем или като бяла, или като черна с цифрите 0 и 1. Черния цвят по-точно можем да опишем с четири градации или нива и на всяко ниво да съпоставяме двуцифрени комбинации: 00, 01, 10 и 11.

В практиката на телеграфното предаване на снимки черният цвят се разделя много често на 16 или 32 нива. При по-големи изисквания се определят 64 нива и тогава на всяко ниво се съпоставя по едно шестцифрено двоично число, което се предава по линията като съчетание от импулси. На приемателния пункт по тези съчетания се възстановява предавания цвят за дадената точка. Така се предават снимки от космическите апарати.

Своеобразно двоично кодиране намира приложение и при записването на звука. Така например известните фирми “Филипс” и “Сони” преминават към производство на нов вид “грамофонни плочи” – така наречените компактни дискове. Те имат диаметър 120 мм и дебелина 1,2 мм. И докато при обикновените грамофонни плочи в каналите, по които се движи иглата, има издатини във вид на непрекъсната крива линия, която точно съответства на звуковите трептения, тук има само еднакви малки издатини и равни участъци. Издатините отговарят на цифрата 1, а равните участъци - на цифрата 0. Чрез различни кодови комбинации от 1 и 0, т.е. от издатини и равни участъци, се съпоставят на тези комбинации различни звуци. Един лазерен лъч разчита цифрите 1 и 0, а специално техническо устройство осъществява съответствието на кодовите комбинации със звуците, в резултат на което се издава и чува записаният звук – мелодия, говор и др.

Записването на числата в някаква бройна система е също вид кодиране. Следователно бройните системи са кодове на числата.

Широко разпространение са добили кодовете, при които числата се записват в дадена бройна система, а съответните цифри се кодират в друга бройна система. Така например в много сметачни машини числата се представят в десетична система, обаче десетичните цифри се записват в двоична система и така получената система често се нарича двоично-десетична.

Кодът “2 от 5” е намерил приложение в някои електронноизчислителни машини. Той е забележителен и с нещо друго. Постоянният брой на единиците и нулите в 5-цифровите комбинации може да послужи за откриване на грешки при предаването и записването на информация. Кодове с това свойство се наричат кодове, откриващи грешки. В такива кодове се използва само част от всички възможни кодови комбинации. Например кодирането в двоична бройна система, както и във всяка бройна система не е кодиране, откриващо грешки. Тук каквато и комбинация от нули и единици да запишем, тя ще съответства на някое число.

Обаче не е трудно от двоичен код, който няма свойството да открива грешки,

да получим код, откриващ грешки. Това може да стане например така: Към всяка кодова комбинация прибавяме (приписваме) допълнително по една двоична цифра, която да бъде 0, ако броят на предишните единици е четен, и 1, ако той е нечетен. Такъв код със сигурност ще открива една грешка.

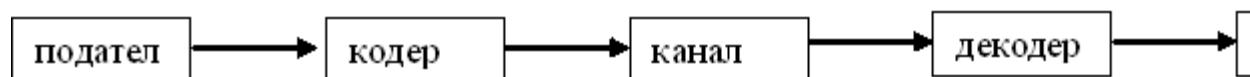
За по-сигурно откриване на грешки се създават кодове с повече проверовъчни цифри при определени правила. Има богата теория на кодовете, откриващи грешки, но за целите на предаването на съобщения не е достатъчно един код да може само да открива грешки. Техническите средства за предаване на съобщения не са свършени – те допускат грешки (понякога 0 записват като 1 или 1 – като 0), особено когато се иска предаването да бъде с голяма скорост. Необходимо е кодът да има способността и да поправя допуснатите грешки. Най-простият начин за поправяне на грешките е чрез многократното повтаряне на кодовите комбинации. Това обаче силно намалява скоростта на предаване на съобщенията и затова се предпочитат кодове, съобразени с изискването за бързина и точност.

По-удобни кодове за откриване и поправяне на грешки се съставят чрез подходящо добавяне на проверовъчни цифри към основните кодови комбинации и се използват свойства, известни от теорията на числата, алгебрата и др. Създадена е богата теория на тези кодове, която продължава да се развива.

Да обобщим:

Кодирането се прилага при предаване на съобщения, които разглеждаме като поредици от знаци – букви, цифри или други символи. Множеството от различните знаци наричаме азбука. Съобщенията се получават с последователно записване на знаци от азбуката. Този или това, което произвежда съобщение, ще наричаме подател или източник на съобщението. Съобщението се изпраща по среда, наречена канал, до даден получател или адресант. За да бъде изпратено, съобщението се кодира чрез т.нар. кодер(възможно е да се извършат няколко последователни кодираня). След изпращането му по канала то в обратен ред се кодира – декодира се чрез декодер, за да се възстанови първоначалната му форма и се възприеме от получателя. Всичко това може да се изрази схематично така:

подател . л
кодер
канал
декодер
получател



До получателя следва да постъпи същото съобщение, същата поредица от знаци, каквато е изпратил подателят. На практика това не винаги е така – допускат се грешки. Използването на кодове, откриващи и поправящи грешки, силно намалява броя на грешките, но не може да се осигури напълно безпогрешно предаване, а и не винаги е необходимо да се предадат точни всички знаци на съобщението. В много случаи получателят се интересува не толкова от всички отделни знаци на съобщението, а от информацията, която то му носи.

Измерване на информацията

1 бит (bit) - Произлиза от латинското *binary digit* (двоична цифра).

се нарича количеството информация, определящо кое от две равновероятни събития се е случило. Това е най-малката единица за количество информация. Един бит е например информацията, определяща дали при хвърляне на зар се е паднал четен или нечетен брой точки. Названието бит произлиза от английското словосъчетание *binary digit* (двоична цифра).

Често един бит се определя и като елемент от памет, който може да има само две състояния, т.е. като място (клетка), в която може да се съхранява 1 бит информация.

По – голямата единица за информация е 1 **byte**(байт), равен на 8 бита.

Количество информация, определящо кое от $2^8=256$ равновероятни събития се е случило се нарича БАЙТ.

Защо именно 8 ? Понеже първите компютри са били осем разрядни и са могли да работят едновременно само с 8 бита. В голяма част от съвременните компютри за съхранение на символи се задела 8 - разрядна клетка (*байт*). В 1 байт може да се запишат 256 различни двоични числа – това позволява да се закодират 256 различни символи. Кода на символите се задава с помощта на таблица (за всеки код се указва съответстващия му символ). = 8 bit или байта е информация за изхода на едно от $2^8 = 256$ равновероятни събития.

Производни единици за измерване на информация

ЕДИНИЦА	К /Кило/	М /Мега/	Г /Гига/
Десетичен множител	10^3	10^6	10^9
Двоичен множител	2^{10} /1 024/	2^{20} /1 024/ ²	2^{30} /1 024/ ³
Наименование /бита/	Килобит Кбит	Мегабит Мбит	Гигабит бит
Наименование /байта/	Килобайт КВ =1 024 В	Мегабайт МВ = 1 024 КВ	Гигабайт ГВ =1 024 МВ

$$1 \text{ KB} = 2^{10} \text{ В} \approx 10^3 \text{ В}$$

$$1 \text{ MB} = 2^{20} \text{ В} \approx 10^6 \text{ В}$$

$$1 \text{ GB} = 2^{30} \text{ В} \approx 10^9 \text{ В}$$

$$1 \text{ TB} = 2^{40} \text{ В} \approx 10^{12} \text{ В.}$$

Азбука се нарича множество от краен брой знакове.

Пример Азбуки

1. Българската азбука.

2. Знаковете за записване на десетични числа.
- 3 Нотните знакове.
- 4 Множеството от два елемента - 0 и 1.

Код се нарича множеството от думи (наричани още кодови комбинации), образувани от знаковете на дадена азбука. Терминът код произлиза от латинската дума *codex* (закон, сборник правила)

Кодиране се нарича процесът на представяне на информация чрез определен Код. Декодиране е обратният на кодирането процес.

Пример Кодиране в различни азбуки

Представянията на числата в различни бройни системи са примери за кодиране в различни азбуки, а съответните алгоритми за преминаване от една бройна система в друга задават начини за кодиране и декодиране: 10100_2 е код на 20. Азбуката на оригиналното съобщение се състои от всички десетични цифри, а азбуката на вторичното - от двоичните 0 и 1.

Кодиране се използва от дълбока древност за засекрещаване на определена информация. Свидетелства за съставянето на кодирани (шифровани) съобщения привежда гръцкият историк Херодот (V в. пр. н. е.). С изобретяване на тайни шифри са се занимавали редица исторически личности (Юлий Цезар, Франсоа Виет и др.) и организации - от древността до днешни дни.

Представяне на десетичните числа в цифровите устройства

Двоичен код

Състоянието на логическите елементи има само две логически стойности 0 и 1. Поради това всички числа, с които работят цифровите схеми, са в двоична бройна система - двоични числа. Ние сме свикнали да работим с десетични числа, т.е. числата с основа 10. Двоичната бройна система използва, както беше казано, двоични числа, т.е. числата с основа 2. Как се записват числа в двоична система? За да отговорим на този въпрос, трябва първо да си припомним, как те се записват в десетична система. Ако искаме да запишем числото двеста петдесет и осем, какво правим? Ние мислено определяме теглови коефициенти за единиците (10^0), десетиците (10^1), стотиците (10^2). Задаваме си въпрос: "Колко единици има в това число?" Отговаряме: "Осем". След това питаем: "А колко десетици?" Отговаряме: "Пет". "Стотици?" Отговаряме: "Две". По този начин ние извършваме следните операции: $2 \cdot 10^2 + 5 \cdot 10^1 + 8 \cdot 10^0$.

10^2	10^1	10^0
2	5	8

След това записваме числото в познатия ни запис: 258. Всичко това го извършваме автоматично.

По същия начин се постъпва и при двоичната система. Първо се определят теглови коефициенти: 2^0 , 2^1 , 2^2 , 2^3 и т.н. следващата таблица показва представяне на десетични числа, записани в двоична система.

--	--	--	--	--	--	--

Число в десетична система	2^4	2^3	2^2	2^1	2^0
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0

Нека да разгледаме числото 4. Как може да бъде записано то, ако се използват само теглови коефициенти 2^0 , 2^1 , 2^2 , 2^3 ? Знаем, че $4 = 2^2$, значи в колоната с теглото 2^2 се записва единица, а във всички останали нули. В резултат се получава двоичен запис 00100. За числото десет ще имаме $10 = 8 + 2 = 2^3 + 2^1$, следователно в колоните с теглата 2^3 и 2^1 се записват единици, а във всички останали - нули, получаваме двоичното число 01010.

Подобен вид операции се извършват и при преминаване в други бройни системи.

И така, за да се запише едно десетично число в двоична система е необходимо да се замени това число със съответен двоичен набор (набор от 0 и 1). Такъв набор, за който може да се каже, че между него и едно десетично число съществува еднозначно и обратимо съответствие, се нарича **код** на десетично число. Процесът на заместване на десетичните числа с техните кодове се нарича **кодирание**.

Всеки, който се занимава с цифрова електроника, трябва да може да преобразува числа от двоична в десетична бройна система и обратно.

Двоично - десетичен код

Двоичното число 111001110 за повечето хора няма особен смисъл. Съществува един друг код, при който също се използват двоични цифри 0 и 1, но който се възприема по-лесно от хората. Този код се нарича двоично-десетичен код 8421, или BCD код. Следващата таблица показва преобразуването на десетичното число 692 в код 8421 BCD.

--	--	--	--	--

	Хиляди	Стотици	Десетици	Единици
Десетично число	-	6	9	2
1 етап на преобразуване	-	$6 = 2^2 + 2^1$	$9 = 2^3 + 2^0$	$2 = 2^1$
2 етап на преобразуване	-	0110	1001	0010
Резултат	692 $\Leftrightarrow$ 0110 1001 0010			

От таблицата се вижда, че всяка десетична цифра е представена с четири двоични цифри. Лявата група (0110) заема позицията на стотиците, средната (1001) - на десетиците, а дясната (0010) на единиците. В следващата таблица е показано, как се преобразуват две десетични числа в 8421 BCD код.

Таблица

	Хиляди	Стотици	Десетици	Единици
Десетично число	6	7	9	5
Число в код 8421 BCB	0110	0111	1001	0101
Десетично число	4	5	1	3
Число в код 8421 BCB	0100	0101	0001	0011

В кода не се използват двоичните числа 1010, 1011, 1100, 1101, 1110, 1111, за него те са забранени. Сега трябва да знаете, защо така се номерират полета в картите на Карно.

Код с излишък 3

Един друг код от тип BCO е код с излишък 3. За да преобразуваме едно десетично число в код с излишък 3, трябва да прибавим 3 към всяка цифра на десетичното число и след това да преобразуваме тази цифра в двоичен код. В следващата таблица е показан начинът за преобразуване на числото 386 в код с излишък 3. Виждаме, че числото 386 се преобразува в 0110 1011 1001.

	Хиляди	Стотици	Десетици	Единици
Десетично число	-	3	8	6

1 етап на преобразуване на десетичното число в код с излишък 3	-	$3 + 3 = 6$	$8 + 3 = 11$	$6 + 3 = 9$
2 етап на преобразуване на десетичното число в код с излишък 3	-	0110	1011	1001

Код на Грей

Кодът на Грей се появил първоначално като код в механични шифратори за ъгъл на завъртане на вал. Понастоящем намира приложение при бързите съвременни аналогово-цифрови преобразуватели. Той притежава свойството *при преминаване от произволно състояние в следващото се променя само един разряд*. Това довежда до повишаване на шумоустойчивостта. Кодът на Грей може да съдържа произволен брой на разряди. Той започва с нулево състояние на всички разряди, а след това за получаването на всеки следващ разряд се инвертира възможно най-младши разряд, изменението на който довежда до образуването на новото състояние.

Позиционен код

В цифровите устройства често се използва позиционен код, в който десетичното число се определя от позицията на 1 в серията от 0 (или от позицията на 0 в серията от 1). Например, ако серията се състои от десет позиции, то 0010000000 представлява числото 7, а 0000001000 представлява числото 3. Очевидно е, че да се представят големи числа в този код е трудно.

В следващата таблица са показани десетични числа, записани с двоичен код, с кода на Грей и с позиционен (униполярен код).

Десетично число	Двоичен код	Код на Грей	Позиционен код
0	0000	0000	0000000001
1	0001	0001	0000000010
2	0010	0011	0000000100
3	0011	0010	0000001000
4	0100	0110	0000010000
5	0101	0111	0000100000
6	0110	0101	0001000000

7	0111	0100	0010000000
8	1000	1100	0100000000
9	1001	1101	1000000000

Шестнадесетичен (хексагонален) код

В електронните изчислителни машини често се използва цифров код с основа 16 - хексагонален код. В този код има 16 символа: от 0 до 9 и първите букви от латинската азбука A, B, C, D, E, F (табл. 4.6).

Десетично число	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Хексагонален код	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Например, числото 2006 може да бъде представено като сума

$$2006 = 7 * 16^2 + B * 16^1 + 6 * 16^0 = 7B6$$

$$2006 = 7 * 256 + 13 * 16 + 6 * 1 = 1792 + 208 + 6$$

Код на Юлий Цезар

Кодът на Юлий Цезар се състои в това, че всеки знак в оригиналното съобщение се замества от втория следващ го знак на съответната азбука (предполага се, че редът на знаковете в тази азбука е фиксиран и предварително известен на кореспондентите). При това предпоследният знак се замества с първия, а последният - с втория знак на азбуката. Например за кирилицата буква А в оригиналното съобщение се замества с В, Б - с Г, ..., Ю - с А, а Я - с Б. При този начин на кодиране е очевидно, че декодирането възстановява еднозначно оригиналното съобщение.

Съобщението „QUO VADIS" (на български КЪДЕ ОТИВАШ?), записано с буквите на латинската азбука, се кодира като „SWQ YCFKU".

В днешно Време информацията се кодира по четири причини:

- за да се записва кратко (сгъстено, компактно, компресирано) с цел да не заема много място при съхраняване и за по-бързо и по-евтино предаване на разстояние;
- за да се улесни извършването на различни операции при обработката ѝ;
- за да се засекрети достъпът до информацията (в този случай кодирането обикновено се нарича шифриране или криптиране);
- да се защити от промяна при предаване и съхраняване.

Каквато и да е кодовата азбука, всеки неин знак може да се номерира с

число, записано в двоична бройна система. Следователно кодовите думи, образувани от азбуката, също ще представляват редици от нули и единици. Всяка редица от кодови думи също ще е редица от нули и единици. Това важно обстоятелство е основно за информатиката.

Защита на информацията (данните) при предаване под формата на съобщение се осигурява, като съдържанието на съобщението се подчини на точно определени правила. Тогава нарушаването на правилата свидетелства за изкривяване на оригиналното съобщение. Правилата могат да се подбират така, че освен установяване на събитие за изкривяване на съобщението, да позволяват локализиране на грешките и дори възстановяване на оригиналното съобщение. Съответно се говори за кодове, откриващи грешки и кодове, коригиращи грешки.

A S C I I код

ASCII (American Standard Code for Information Interchange) кодова таблица

Всички данни се представят в компютъра като числа. Това се отнася за буквите, цифрите и другите знаци. На всеки знак трябва да съответства едно единствено число, за да може представянето на знака да бъде уникално. При това въпросното число трябва да бъде едно и също за всички компютри, за да има съвместимост на данните помежду им. За тази цел са създадени така наречените кодови таблици.

Буквите, цифрите и другите знаци представляват графични изображения, с които ние си служим за обмен на текстова информация. Кодовата таблица е съответствие между тези графични изображения и числа, които представляват пореден номер в таблицата. Когато набираме от клавиатурата думата „пролет“, Компютърът я запомня като поредица от шест числа, които стоят срещу съответните букви в таблицата. При по-нататъшната обработка на тази дума, тя е във вид на поредица от шест числа. Когато я извежда на екрана или на принтера, компютърът съпоставя на шестте числа съответните графични изображения от таблицата и ние отново можем да прочетем думата „пролет“.

Добилата най-голяма популярност и приета за стандарт при микрокомпютрите е ASCII кодовата таблица. В тази таблица на всеки знак съответства точно определено число. С това число съответният знак се представя в компютъра. Буквата „P“ например се представя с числото 70, запетаята с числото 44 и т. н.

Стандартната ASCII таблица има 128 елемента. В нея знаците са представени

с поредни числа от 0 до 127. Тя се дели на две части с различно предназначение. Първите 32 знака - от 0 до 31 се използват и интерпретират различно от различните периферни устройства. Те рядко се изписват на изходните устройства като графично изображение. Например 12 (PF) за преминаване на нова страница; 10 (LF) - за нов ред 13 (CR) - за връщане в началото на реда; 27 (ESC) - за начало на поредица от управляващи символи и т.н.

Означенията на първите 32 символа от ASCII кодовата таблица и функциите, които те изпълняват, са показани в следващата таблица.

Стандартна ASCII кодова таблица на символите от 0 до 31		
N	Означение	Функция
0	NUL	Null character
1	SOH	Start of Header
2	STX	Start of Text
3	ETX	End of Text
4	EOT	End of Transmission
5	ENQ	Enquire
6	ACK	Acknowledge transmission
7	BEL	Bell
8	BS	Backspace
9	HT	Horizontal Tab
10	LF	Line Feed
11	VT	Vertical Tab
12	FF	Form Feed
13	CR	Carriage Return
14	SO	Shift Out
15	SI	Shift
16	DLE	Data Link Escape
17	DC1	Device Control 1
18	DC2	Device Control 2
19	DC3	Device Control 3
20	DC4	Device Control 4
21	NAK	Negative acknowledge
22	SYN	Synchronization
23	ETB	End of Transmission Block
24	CAN	Cancel
25	EM	End of Medium
26	SUB	Substitute
27	ESC	Escape
28	FS	File Separator
29	GS	Group Separator
30	RS	Record Separator
31	US	Unit Separator

Стойностите от 32 до 127 представят големите и малките букви от английската азбука, цифрите, шпацията и други знаци (+, -, ., ;, ' ? ! @ \$ %):

Стандартна ASCII кодова таблица на символите от 32 до 127					
32	48 0	64 @	80 P	96 ‘	112 p
33 !	49 1	65 A	81 Q	97 a	113 q
34 “	50 2	66 B	82 R	98 b	114 r
35 #	51 3	67 C	83 S	99 c	115 s
36 \$	52 4	68 D	84 T	100 d	116 t
37 %	53 5	69 E	85 U	101 e	117 u
38 &	54 6	70 F	86 V	102 f	118 V
39 .	55 7	71 G	87 W	103 g	119 w
40 (	56 8	72 H	88 X	104 h	120 x
41)	57 9	73 I	89 Y	105 i	121 y
42 *	58 :	74 J	90 Z	106 j	122 z
43 +	59 ;	75 K	91 [	107 k	123 {
44 ,	60 <	76 L	92 \	108 l	124 I
45 -	61 =	77 M	93]	109 m	125 }
46 .	62 >	78 N	94 ^	110 n	126 ~
47 /	63 ?	79 O	95 _	111 o	127

С появта на персоналния компютър на фирмата IBM - IBM PC(Personal Computer) през 1981 г., ASCII таблицата се допълва до 256 знака. Допълнителните символи от 128 до 255 съставят така наречената разширена ASCII кодова таблица. IBM включиха в нея всички букви от западноевропейските азбуки, които не фигурират в английската азбука. Включени са също така и 48 специални символа за изобразяване на таблици и рамки, няколко гръцки букви, парични знаци и най-често използвани математически символи.

За разлика от стандартната ASCII-Кодова таблица (0-127), Където знаците и техните позиции са фиксирани и са еднакви за всички компютърни системи, разширената ASCII-таблица има множество от варианти. Българската азбука се наместа в разширената ASCII-кодова таблица, възприета от производителите на персонални компютри в Правец.

Американски стандартен код за обмяна на информация е символно множество и символно кодиране базирано на латинската азбука. ASCII кодовете представят текст в компютрите и други комуникационни и управляващи устройства работещи с текст.

ASCII дефинира кодове за 33 неизобразими контролни символа (предимно

определящи обработката на текста) плюс следните 95 изобразими символа (започвайки със символа интервал):

ASCII определя еднозначно съответствие между двоични кодове и писмени знаци (глифи), правейки възможно обмяната на текстова информация между отделни цифрови устройства, както и нейното съхраняване в тези устройства. Важно е да се отбележи, че ASCII определя съответствие между кода и *семантичната* стойност на глифа, а не негова конкретна реализация. Компютърните шрифтове са тези които определят реализациите на глифите. Да вземем за пример главната латинска буква А - нейният двоичен код е 01000001 или 65 в десетичен код. В един обикновен текстов файл в ASCII код тя ще бъде представена именно с този код - 65. Но за да се изобрази на екрана е нужен шрифт който от своя страна съпоставя на този код някакво изображение. Това изображение е различно при отделните шрифтове, но смисълът който то предава остава същия - главна латинска буква А.

ASCII е 7-битов код, което означава, че може да представи 128 символа. Често той е допълван до 8 бита добавяйки една или повече допълнителни азбуки (освен латинската). По времето на своето създаване за изход на текст са се използвали предимно печатащи устройства затова в ASCII се съдържат и контролни кодове уточняващи начина по който да се изпечатат символите.

Разбиране на кодирането на текста

Това, което на екрана виждате като текст, всъщност се съхранява като цифрови стойности в текстовия файл. Вашият компютър превежда цифровите стойности в знаци. Той прави това, използвайки кодиращ стандарт.

Кодиращият стандарт представлява схема на номериране, която приписва на всеки знак в набор от знаци една цифрова стойност. Наборът от знаци може да включва знаци от азбуката, числа и други символи. Различните езици обикновено се състоят от различни набори от знаци, така че съществуват много различни стандарти за кодиране, представляващи наборите от знаци, използвани в различните езици.

Различни стандарти за кодиране за различните азбуки

Стандартът за кодиране, който се съхранява с текстов файл, предоставя информацията, нужна на вашия компютър, за да изобрази текста на екрана. Например в кодирането на кирилица (Windows) знакът Й има числова стойност 201. Когато отворите файл, съдържащ този знак на компютър, използващ кодирането на кирилица (Windows), компютърът чете числовата стойност 201 и на екрана показва Й.

Ако обаче отворите същия файл на компютър, който използва друго кодиране, компютърът показва такъв знак, който отговаря на числовата стойност 201 в стандарта за кодиране, използван по подразбиране от съответния компютър. Ако например вашият компютър използва стандарта за кодиране Western European (Windows), знакът в оригиналния файл, базиран на кирилица, ще се изобрази като Ё, а не Й, тъй като в кодирането Western European (Windows) стойността на 201 отговаря на Ё.

Кодови таблици и шрифтове

Повечето операционни системи и приложни програми за персонални компютри записват текста като последователност от байтове (един байт

представлява 8 бита - 256 възможни комбинации). Доскоро, най-разпространен бе стандарта, при който един символ се кодира с един байт (за РС компютрите това е ANSI стандарта). Използването на този стандарт е ограничение, тъй като могат да се използват само 256 символа, което е недостатъчно при работа с много езици. Решението на проблема е в използването на няколко кодови таблици от по 256 символа.

В съвременните операционни системи и програми се налага тенденцията за използване на два байта за кодиране на един символ (два байта представляват 16 бита - 65536 възможни комбинации). Това дава възможност да има отделни кодове за буквите от почти всички световни азбуки, като по този начин се преодоляват трудностите при поддръжката на много кодови таблици. Най-разпространения стандарт за двубайтово кодиране на символите е **Unicode**.

Стандарт ANSI

При този стандарт, един символ от текста се представя с един байт. Тъй като кодовете от 0 до 31 обикновено са резервирани, общо 224 различни символа могат да се използват в документите.

Реално можете да ползвате много повече от 224 символа, като избирате различни шрифтове. Например, когато изберете шрифта Wingdings в Word Pad, разполагате с много голям набор от икони и малко символи, които съответстват на буквите, цифрите и специалните знаци в "обикновените" буквени шрифтове.

Обяснението на този факт се крие в термина кодова таблица. Когато натиснете клавиш и по този начин въведете в една текстообработваща програма някоя буква, програмата записва кода на буквата и информация за шрифта на буквата. Но в различните шрифтове един и същ код не означава един и същ символ! Например в Arial, Courier New и Times New Roman кодът 65 представлява главната латинска буква „А". Същият този код в Wingdings представлява иконата „ръка с два вдигнати пръста". Кодовата таблица е именно това съответствие между кодовете на символите и самите символи.

Илюстрация на разликата между символа с код 65 в шрифтовете Arial и Windings

Arial	WinDings
65 = A	65 = 

Важно е да разберете, че кодовата таблица не определя същинското графично изображение на символа. По-скоро, тя определя, че „символът с код 65 е главна латинска буква А". По този начин шрифтовете Arial и Times New Roman ползват една и съща кодова таблица (US, Western Europe), като и двата шрифта съдържат буквата А на позиция 65. Въпреки това, графичното изображение на „А" в Arial е доста по-различно от това на „А" в Times New Roman.

Буквите от кирилицата (които могат да достигнат до 66, като броим отделно малките и големите букви) не са включени в кодовата таблица US, Western Europe. Следователно, потребителите трябва да ползват такива шрифтове, които са построени по друга, съдържаща символите от кирилицата, кодова таблица. Такава

таблица е 1251 (Eastern Europe/Cyrillic), която съдържа всички букви, използвани в английския език, както и всички букви от българския, руския и т.н.

Всеки шрифт е построен по някаква кодова таблица. Ако желаете да пишете букви от латиницата, Вие можете да ползвате всеки шрифт с кодова таблица US, Western Europe или Eastern Europe/Cyrillic, тъй като всички те съдържат латински букви. Ако желаете да пишете на кирилица, трябва да изберете шрифт подреден по Eastern Europe/Cyrillic, тъй като US, Western Europe не съдържа буквите от А до Я.

Когато желаете да пишете текст на кирилица, трябва да изберете шрифт, съдържащ кирилица (напр. Timok или Hebar) от командата Fonts... Ако желаете да пишете на латиница, можете да ползвате както Timok и Hebar, така и стандартните шрифтове за Windows (напр. Ariel и Times New Roman).

Стандарт UNICODE

При този стандарт един символ от текста се представя с два байта, т.е. за почти всички букви от световните азбуки има отделни кодове в диапазона от 0 до 65535.

В стандарта ANSI, тъй като има ограничение от 256 символа, се получава препокриване на кодовете за буквите от различните азбуки. Например, буквата „Д“ от кирилската кодова таблица има код 196. В гръцката кодова таблица на код 196 отговаря буквата „делта“, а в западноевропейската кодова таблица на същия код отговаря буквата „А с две точки отгоре“. Тъй като кодът на символите е един и същ, кой от трите символа ще бъде изобразен зависи от кодовата таблица на избрания шрифт. Ако шрифтът е кирилски, ще бъде изобразена кирилската буква „Д“, ако шрифтът е гръцки, ще бъде изобразена гръцката буква „делта“ и т. н. Изборът на правилен шрифт обикновено затруднява обработката на документи, а много често е и причина за грешки.

В стандарта Unicode всяка една буква от най-разпространените азбуки има отделен код. В разглеждания пример, буквата „Д“ има код 1044, гръцката буква „делта“ има код 916, а „буквата А с две точки отгоре“ има код 192. По този начин, ако в даден документ бъде избран Unicode шрифт, ще е възможна работа с много различни символи без да е необходима смяна на шрифта.

Unicode: Един стандарт за кодиране за много азбуки

За да избегнете проблеми с кодирането и декодирането на текстови файлове, можете да записвате файловете с кодиране Unicode. Unicode съдържа повечето набори от знаци, съдържащи се в езиците, използвани от компютърните потребители.

Тъй като Word е базиран на Unicode, Word автоматично записва файловете, кодирани като Unicode. Можете да отваряте и четете файлове, кодирани с Unicode, на вашата компютърна система на английски език, независимо от езика на текста. И подобно, когато използвате вашата система на английски език, за да запишете файлове, кодирани като Unicode, файлът може да включва знаци, които не се намират в западноевропейските азбуки, като например гръцки, арабски, японски знаци или знаци на кирилица.

Уникод или *Универсален набор от символи* (на английски, **Unicode**) е разработен да реши проблемите с наличието на много съществуващи еднобайтови набори от символи (кодировки), използвани при писане на текст на различни езици.

Уникод (универсална кодировка) е набор от символи, създаден с цел е да съдържа всеки символ, използван при писане, на всички азбуки (включително и много мъртви езици) и други символи, използвани в математиката и инженерните науки.

Всеки знак или символ отговаря на определен номер в кодовата таблица, който се ползва примерно за изобразяване на символа в даден шрифт. Тъй като символите са няколко десетки хиляди и не е възможно да се представят в един байт, по специални правила (UTF) номерата на символите се трансформират в един, два, три или четири байта, за да могат да се ползват от компютрите.

- Обикновената латинска азбука, цифрите и основните препинателни знаци се изобразяват с един байт.

- Повечето диакритични латински букви, кирилицата, гръцката, арменската, еврейската, арабската и др. азбуки и някои знаци се представят с два байта.

- Други азбуки (африкански, азиатски, американски) и някои специални знаци се кодират с три или четири байта.

Всеки един от досегашните набори от символи, кодирани с един байт, може без загуба да бъде преобразуван към Уникод, при едно условие: да сме сигурни точно от кой набор (в коя кодировка) са символите. Обратното преобразуване, от Уникод към някоя от еднобайтовите кодировки, е еднозначно и безспорно за тези знаци, които съществуват в целевата кодировка. Но текстовете в универсалната кодировка могат да съдържат всякакви знаци и тогава не е възможно да се преобразуват коректно в някоя еднобайтова кодировка.

Например текст само на кирилица или само на чешка латиница (с диакритични знаци) се преобразуват без проблем от съответната кодировка към Уникод и обратно. Обаче текст в Уникод и с кирилски букви, и с латински диакритични знаци (букви с ударения и лигатури), не може да се преобразува изцяло: или кирилицата ще се загуби, или буквите с диакритични знаци, в зависимост от целевата кодировка.

Относно дължината на текстовете, поради кодирането на всеки знак с един до четири байта (вместо само с един), обикновено текстове в Уникод са с по-голяма дължина от същите текстове в старите еднобайтови кодировки.

Unicode – това е уникален код за всеки знак, независимо от компютърната платформа, независимо от програмата, независимо от езика.

В самата си същност, компютрите работят само с числа. За да запомнят буквите и другите знаци компютрите им присвояват определено число или код. Преди въвеждането на Unicode по света имаше стотици различни системи за кодиране, които им присвояваха тези кодове. Нито една от тези системи не бе достатъчно добра за описване знаците дори на един език: например Европейския Съюз само изисква няколко различни кодирания за да поддържа всичките си езици. Дори за един език, като например Английски едно кодиране не бе адекватно за всички букви, знаци за пунктуация, технически символи и т.н.

Освен това отделните системи за кодиране имаха конфликти една с друга. Така например две кодирания може да използват един и същи номер за *два различни* знака, или пък да се да използват два различни номера за *един и същи* знак. Всеки отделен компютър (и по-специално сървърите) трябва да поддържа много различни кодирания; освен това когато данни се прехвърлят между различни кодирания или платформи винаги има риск от проблеми.

Unicode промени всичко това !

Unicode осигурява уникален номер за всеки знак, независимо от платформата, независимо от програмата, независимо от езика. Стандартът Unicode бе приет от такива технологични лидери, като компаниите Apple, HP, IBM, Microsoft, Oracle, SAP, Sun, Sybase, Unisys и много други. Unicode е задължителен за новите стандарти като XML, java, ECMAScript (JavaScript). LDAP, CORBA 3.0, WML и т.н. и е официално приет като стандарт ISO/IEC 10646. Той се поддържа в много операционни системи, всички модерни браузъри и много други програмни продукти. Появата на стандарта Unicode, както и наличието на редица инструментариуми за поддръжката му е един от най-значителните глобални технологични пробиви в последно време.

Внедряването на Unicode в приложения от типа клиент-сървър или сложни приложения с много модули, както и уеб сайтове предлага значителни икономии, в сравнение с използването на старите кодови таблици. Unicode дава възможност на отделни продукти или уеб сайтове да бъдат планирани и пускани практически без преработка. Пренасянето на данни през различни операционни системи без загуба или повреда вече е реалност.

H.264, MPEG-4 или **AVC** (съкращението означава *напреднало видеокодиране*), е стандарт за цифров кодек за видеосигнали, който е създаден за постигане на много високо компресиране на данни. Той е създаден от ITU-T Video Coding Experts Group (VCEG) заедно с ISO/IEC Moving Picture Experts Group (MPEG) като продукт на колективното партньорство познато също като Joint Video Team (JVT). ITU-T **H.264** стандарта и ISO/IEC **MPEG-4** стандарта (накратко, ISO/IEC 14496-10) са създадени заедно, така че те са технически идентични. Последната проектна работа по първата версия на стандарта е била завършена през май 2003 г.

DES (Data Encryption Standard - Стандарт за кодиране на данни) - Стандартизиран метод за кодиране, широко използван в Интернет

Кодиране на цифровите изображения. Основни принципи за компресиране на данни. Компресиране на неподвижни изображения. Алгоритъм JPEG.

Компресиране на неподвижни изображения (JPEG)

Първите приложения, целящи намаляване на количеството информация, необходима за кодирането на неподвижни картини, се появиха през 80-те години и имаха като основна цел значително намаляване на размера на графичните и фотографските файлове от гледна точка на тяхното съхранение или транспортиране. През 1990г. ISO (Международната организация по стандартизация) създаде международна работна група, наречена **JPEG** (Joint Photographic Expert Group), която имаше за задача да разработи международен стандарт за компресиране на неподвижни картини с различна разделителна способност във формат Y Cr Ch или RGB. Полученият международен стандарт (широко известен като JPEG) беше публикуван през 1993г. като ръководство за компресиране на неподвижни картини. Тук няма да описваме в подробности този стандарт, тъй като това не е целта на тази книга, но ще бъдат представени неговите основни стъпки, тъй като те в голяма степен дават представа за начина, по който работи и MPEG.

Трябва да се отбележи, че компресирането по JPEG може да бъде **със загуба на информация** или **без загуба на информация** (обратима), в зависимост от избраното приложение и желанния коефициент на компресиране. Повечето приложения, в зависимост от съдържанието на картината, използват метода със загуба на информация, който позволява да се постигнат коефициенти на компресиране, по-големи от 10 без забележимо влошаване на качеството на картината. Тук ще представим компресиране по JPEG със загуба на информация, тъй като за кодирането на I (вътрешни) картини MPEG използва същия процес. JPEG компресирането без загуба на информация използва едно по-различно предсказуемо кодиране, което не е базирано на DCT и поради това няма да бъде разгледано в тази книга.

MPEG MPEG е съкращение от Motion Picture Experts Group. Същата организация стояща зад MPEG-1 (използвано във VideoCDs), MPEG-2 (използва се в DVD и SuperVCDs и др. високопоточни устройства) и MPEG-4 (ASF, DivX, WMV, средно-ниско поточни системи) стандарти. Отговорни са за MP3 (MPEG Audio Layer 3) и AAC аудио стандартите .

Основните мултимедийни видео формати са 3: “AVI (Audio Video Interleave)”, “MPEG (Motion Picture Experts Group)” и “QUICKTIME”.

“AVI” е стандартният формат поддържан от “Windows”. Създаден от Майкрософт, той е разработен с цел съхраняване на аудио и видео информация. Максималната големина на картината е 320x240 с 30 кадъра в секунда, което е крайно недостатъчно за пълноекранно видео, но тъй като няма специфични изисквания към системата, е предпочитан от компаниите за внедряване на филмчета в продуктите им, предназначени за “Windows”.

“QUICKTIME” е формат за видео и анимация, разработен от “Apple”. Той е базисен видео формат за “Macintosh” системи. Възпроизвежда се и на “Windows” системи, но е нужен специален драйвър. Освен че поддържа повечето известни методи за компресия, стандартът предлага завидно аудио и видео качество и е напълно съвместим с всички по-известни уеб браузъри, което го прави предпочитан за Интернет приложения.

“MPEG” форматът предлага голям набор от възможности за компресия с различни скорости на предаване. Той е стандартизиран за видео компактдискове (“VCD”) и “DVD”.

име	битове на отчет	компресия	създател
“ AVI”	24	“RLE”, аудио, видео, некомпресирано	“Microsoft”
“QUICKTIME”	24	“RLE”, “JPEG” и други	“Apple”
“MPEG”	24 – 4:2:0	дискретно косинус преобразуване и вътрешноблоково кодиране с компенсация на движението	“Motion Picture Expert Group”

Видео форматите съдържат в себе си кодирано видео, аудио и допълнителна информация. За разлика от аудио форматите тук кодерите, които може да се използват са доста повече. Допълнителната информация може да е за синхронизация, субтитри и т.н.

Формати за кодиране на аудио информация - MP3 and OGG Vorbis

Компресиране на информацията.

В много приложения за предаване на данни, когато цената на линията, която заплащаме зависи от времето на използване или от количеството пренасяна информация е целесъобразно да използваме компресия на данните. Компресията представлява замяната на някои символи или битови поредици с други - по-къси символи или битови поредици. Като резултат предаваният низ става по-малък, следователно се предава по-бързо. Например ако предаваме данни с модем със скорост 14400 bps и постигаме компресия 50% е все едно че използваме модем със скорост 28800 bps без компресия. На практика можем да приложим различни компресионни алгоритми, всеки от които е насочен към ефективно компресиране на различен тип данни. Някои модеми имат вградени няколко компресиращи алгоритъма и в зависимост от типа на данните, които се предават в момента използват най-подходящия алгоритъм.

Представяне на цветовете в цифров вид

С компютърна графика ние се срещаме на всяка крачка. И не само в компютрите. Практически всяка произволна съвременна книга може да се направи на компютър. Нека погледнем екрана на компютъра през увеличително стъкло. В зависимост от марката и модела на техниката ние виждаме или множество разноцветни правоъгълници, или множество разноцветни кръгчета. И едните, и другите се групират по три броя, при това един цвят, но различни оттенъци. Те се наричат ПИКСЕЛИ (от английски PICTURE's ELEMENT). Пикселите биват само три цвята - зелен, син и червен. Другите цветове се образуват с помоща на смесване на цветовете.

Нека разгледаме най – простия случай – всяко късче пиксел може или да свети (1), или да не свети (0). Тогава ние получаваме следния набор от цветове :

Червен	Зелен	Син	наименование	цвят
0	0	0	Черен	
0	1	0	Зелен	
0	0	1	Син	
1	0	0	Червен	
0	1	1	Тюркоаз	
1	1	0	Жълт	
1	0	1	Малиново	
1	1	1	Бял	

Представяне на цветовете в цифров вид

И така, всяко графическо изображение на екрана може да се закодира с помощта на числа, като се знае колко такива има във всеки пиксел/точка/ за червен, зелен или син цвят.

Кодиране на звуци

По своята природа звуците се явяват непрекъснати сигнали. За кодиране на звук е необходимо този непрекъснат сигнал да се превърне в последователност от нули и единици (дискретизация). При работа със стереозвуци това се изпълнява отделно и независимо за левия и десния канал. На качеството на възпроизвеждане на закодирания звук влияят основно два параметъра : честотата на дискретизация и нейното разрешение – размера на клетките, отвежданото за запис значение на амплитудата.

[предишна тема](#)

[следваща тема](#)