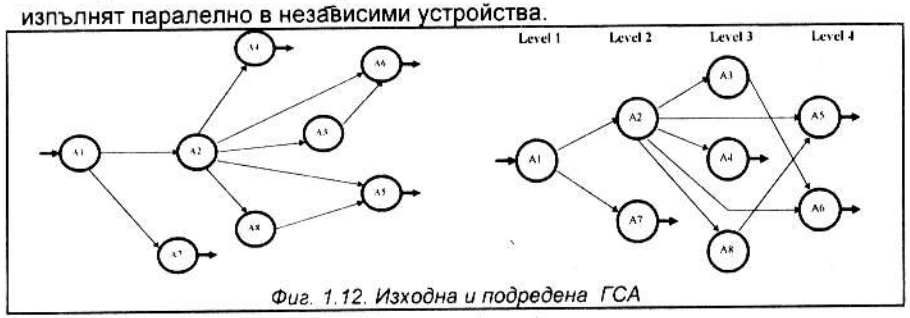
♦ Организацията на изчислителната обработка на ***макро-ниво*** се свързва с приложния аспект. В общия случай, компютърната обработка е съвкупност от процедури, изпълнявани над потоци от данни. Процедурите се реализират на базата на отделни алгоритми, чието обединение се разглежда като *пълен алгоритъм* на компютърната обработка. За неговото описание е необходимо да са известни отделните алгоритми A_i и условията за тяхното активиране.

Представянето на множеството от задачи (отделни алгоритми) може да стане чрез насочен граф - възможните преходи между отделните задачи (върховете в графа) се описват чрез матрица на връзките (наличните ребра между върховете). Такова формализирано описание се нарича *графова схема на алгоритъм (ГСА)*. При детерминирания подход за изследване на компютърната обработка се дефинира булева матрица на връзките $L=\{l_{ij}\}$, където $l_{ij}=1$ (ако има информационна връзка) и $l_{ij}=0$ (ако няма връзка). Тази матрица може да се използва за преобразуване на изходната ГСА в *подредена ГСА*, в която информационно-независими върхове от един ранг са обединени в общ слой (фиг. 1.12). Задачите от даден слой могат да се изпълнят паралелно в независими устройства.



Фиг. 1.12. Изходна и подредена ГСА

Подреждането на върховете от графа става по правилата:

- върховете от първия слой нямат предшественици, а върховете от последния слой нямат наследници;

- върховете, включени в общ слой, нямат свързващи ребра.

Слоевете в подредената ГСА са формирани съгласно процедурата:

1. Всички начални върхове (без предшественици) се номерират първи и се включват в слой (1).

2. Пореден връх се номерира (подлежи на включване към текущ слой), ако всички негови предшественици са вече номерирани (включени в предишни слоеве).

3. Върховете, които подлежат на номериране, са непосредствени наследници на вече номерирани върхове.

Подредената ГСА позволява да се изчислят дължините и броя на всички възможни пътища в графа, представящи възможните информационни процеси в компютърната обработка.

Глава втора
Организация на компютърната обработка

2.1. Понятие за машинна аритметика

2.1.1. Представяне на информацията в КС

Информацията, представяна в КС, може да се разглежда като числова (за представяне на числови данни) и символна (за представяне на текстова информация, включваща букви, цифри и специални знаци). При представянето се използва азбука A={N,S}, която включва две подмножества: на цифрите N={0,1, ..., p-1} и на разрешените символи $S=\{s_1, ..., s_k\}$. Мощността на азбуката $|A|=p+k$ зависи от приетия механизъм за кодиране (код, дължина на думата, начин) и е свързана с т.нар. кодово пространство (съвкупност от всички възможни кодови комбинации).

Базовите информационни единици в КС са следните:

- *бит* – информация за елементарно събитие с равновероятностен изход {0, 1};
- *байт* – минимална адресируема единица (8 бита), която е основна клетка на паметта;
- *поле* – група от *m* байта, които се адресират едновременно чрез адреса на първия (левия) байт;
- *машинна дума* – структурна единица за всяка КС, определяща формата на данните.

Представяне на информацията – превръщане на данните (числови, символни) чрез съответна кодираща система и разполагане на кодираните цифри и символи в полета с фиксирана дължина (определят т.нар. формат на данните).

Машинната обработка е свързана с най-ниското ниво на компютърната обработка в процесора и формира т.нар. машинна аритметика – аритметични основи за представяне, преобразуване и обработване на данните. Основните проблеми, свързани с поддържане на машинната аритметика (представяне и обработка на данни), са следните:

- приложение на бройни системи за представяне на числата в КС и извършване на аритметични действия с тях;
- използване на машинни кодове за представяне на преобразуваните двоични числа със знак и опериране с тях;

- задаване на формати за представяне на различни по тип данни (цели, реални, десетични, символни);

- приложение на контролиращи и коригиращи кодове при представяне и обработка на данните;

- организация на машинната обработка в процесора на КС, свързана с осигуряване на изпълнението на операциите и тяхното управление.

2.1.2. Бройни системи и превръщане на числата

Бройни системи

Бройната система (БС) е съвкупност от азбука и правила за представяне на произволно число чрез елементите на азбуката. Основните видове са:

а) *Позиционна БС* (ПБС) – стойността на всеки разряд на числото се определя от: абсолютната стойност на цифрата; мястото в записа; основата "p" на БС.

б) *Непозиционна БС* (НБС) – стойността на отделните разряди не зависят от мястото им в записа на числото, т.е. стойността на числото се определя от номиналната стойност на цифрите (примери: Римската БС, НБС с остатъчни класове и др.).

Общото ***представяне на число в ПБС*** може да се даде с следните три записа, които са еквивалентни:

/1/ $N_p = a_n.p^n + a_{n-1}.p^{n-1} + ... + a_1.p^1 + a_0.p^0 + a_{-1}.p^{-1} + ... + a_{-m}.p^{-m}$

/2/ $N_p = \sum_{j=-m}^{n} a_j.p_j$;

/3/ $N_p = a_n\, a_{n-1} \ldots a_1\, a_0\, ,\, a_{-1} \ldots a_{-m}$.

В зависимост от основата "p" в КС намират приложение двоична (p=2 и A={0,1}), осмична (p=8 и A={0,1, ..., 7}), десетична (p=10 и A={0,1, ..., 9}) и шестнадесетична (p=16 и A={0,1, ...,9, A ,B, C, D, E, F}) ПБС. Примери са показани в табл.2.1. Трябва да се отбележи, че осмичната и шестнадесетичната ПБС са свързани с двоичната чрез $p=2^k$ (табл. 2.2).

Таблица 2.1

ПБС	Число	Разгърнат запис (основите са в десетична ПБС)	Десетична стойност
Двоична	11100,1	$1.2^4+1.2^3+1.2^2+0.2^1+0.2^0+1.2^{-1}$	28,5
Осмична	34,4	$3.8^1+4.8^0+4.8^{-1}$	28,5
Десетична	28,5	$2.10^1+8.10^0+5.10^{-1}$	28,5
Шестнадесетична	1C,8	$1.16^1+C.16^0+8.16^{-1}$	28,5

Таблица 2.2

k	1	2	3	4	5	6	7	8	9	10
$p=2^k$	2	4	8	16	32	64	128	256	512	1024

Превръщане на числата

Използването на различни ПБС налага преобразуване на числата от една изходна ПБС с основа "p" и с азбука $A_p=\{a_1, ..., a_p\}$ в друга ПБС с основа "q" и с азбука $A_q=\{b_1, ..., b_q\}$. Преобразуването на числата от произволна ПБС в десетична ПБС може да стане чрез т.нар. алгоритъм на заместването, основаващ се на представения по-горе запис /1/. Например $1011_2 = 1.2^3+0.2^2+1.2^1+1.2^0 = 11_{10}$. За общия случай на преход между произволни ПБС се прилагат алгоритми отделно за цялата и за дробната части от записа на числата, представени по-долу.

(а) Преобразуване на цели числа

Нека е зададено число $N_p=a_n.p^n+a_{n-1}.p^{n-1}+...+a_1.p^1+a_0.p^0$ в ПБС с основа 'p', което трябва да се представи в друга ПБС с основа 'q' като числото $N_q=b_m.q^m+b_{m-1}.q^{m-1}+...+b_1.q^1+b_0.q^0$. Новият запис може да се преобразува във вида: $N_q=(b_m.q^{m-1}+b_{m-1}.q^{m-2}+...+b_1).q+b_0 = N^{(1)}.q+b_0$. Като се отчете, че $N^{(1)}.q$ се дели на q и винаги $b_0<q$, то стойностите $N^{(1)}$ и b_0 могат да бъдат определени в резултат на целочисленото делене $N_q/q=N^{(1)}+b_0/q$ (b_0 е остатък). Изхождайки от постановката, че по стойност двете числа са еквивалентни, алгоритъмът се описва чрез следните стъпки:

<u>Стъпка (1):</u> Първата (най-младша) цифра b_0 от търсеното число се определя от получения остатък (заграденото в скоби) при деленето:

$$N_p \approx N_q \Rightarrow \frac{N_p}{q} \approx \frac{N_q}{q} = b_m q^{m-1} + ... + b_1 q^0 + \left[\frac{b_0}{q}\right]$$

Цялата част (след равенството) се означава с N_p^1 и участва в следващата стъпка.

<u>Стъпка (2):</u> Следващата (по-старша) цифра b_1 се определя като остатък от ново делене на получената от стъпка (1) цяла част:

$$\frac{N_p^1}{q} = b_m q^{m-2} + ... + b_2 q^0 + \left[\frac{b_1}{q}\right]$$

Получената нова цяла част N_p^2 продължава участие в следващата стъпка.

Аналогично се формират следващите разряди, като общият запис е:

<u>Стъпка (к+1)</u>: Текуща цифра b_k се определя от деленето:

$$\frac{N_p^k}{q} = N_p^{k+1} + \left[\frac{b_k}{q}\right]$$

като условие за край на алгоритъма е получаване на цяла част $N_p^{k+1} = 0$.

Пример: $18_{10} \rightarrow N_2$ (p=10; q=2; A_q={0,1})

18:2 = 9 + остатък(0) $\Rightarrow b_0 = 0$

9:2 = 4 + остатък(1) $\Rightarrow b_1 = 1$

4:2 = 2 + остатък(0) $\Rightarrow b_2 = 0$

2:2 = 1 + остатък(0) $\Rightarrow b_3 = 0$

1:2 = 0 + остатък(1) $\Rightarrow b_4 = 1$ (изпълнено условие за край на алгоритъма)

Резултат от преобразуването : $N_2 = b_4b_3b_2b_1b_0 = 1\ 0\ 0\ 1\ 0$

(б) Преобразуване на дробни числа

Преходът към десетична ПБС може да стане по указания по-горе алгоритъм на заместването.

Пример: $N_2 = 0{,}1101 \rightarrow N_{10} = 1.2^{-1} + 1.2^{-2} + 0.2^{-3} + 1.2^{-4}$

$N_{10} = 1/2 + 1/4 + 1/16 = 0{,}5 + 0{,}25 + 0{,}0625 = 0{,}8125_{10}$

При преход между произволни ПБС се прилага алгоритъм, който може да се опише така:

(1) Изходната дроб се умножава по основата q на новата ПБС.

(2) Отделя се цялата част от резултата, която се приема за поредната цифра от търсения (нов) запис на числото.

(3) Ако е достигната необходимата точност на представянето преобразуването се прекратява. В противен случай се започва от (1).

Математическият запис на алгоритъма за преобразуването:

$$R_p = \sum_{j=1}^{n} a_{-j}.p^{-j} \quad \rightarrow \quad R_q = \sum_{j=1}^{m} b_{-j}.q^{-j}$$

се описва чрез следните стъпки:

Стъпка (1): От еквивалентността на двата записа се извършва:

$$R_p \approx R_q \Rightarrow R_p \times q \approx R_q \times q = b_{-1} + \left\lfloor b_{-2}.q^{-1} + \ldots + b_{-m}.q^{-(m-1)} \right\rfloor,$$

където заграденото в скоби е дробната част R^1, участваща в следващата стъпка, а цялата част определя първата търсена цифра b_{-1} на новия запис.

Стъпка (2): Аналогично чрез

$$R_p^1 \times q = b_{-2} + \left\lfloor b_{-3}.q^{-2} + \ldots + b_{-m}.q^{-(m-2)} \right\rfloor$$

се определят следващата цифра b_{-2} (от цялата част) и дробната част R^2 за следващата стъпка. Аналогично се формират следващите разряди.

Стъпка (к+1): Общият запис за определяне на цифра $b_{-(k+1)}$ е:

$$R_p^k \times q = b_{-(k+1)} + R_p^{k+1},$$

като условието за край на алгоритъма е получаване на нулева дробна част или по предварително зададен брой разряди за дробната част.

Пример: $0{,}9_{10} \rightarrow N_2$

0,9 x 2 = 1,8 = 1(цяла част определяща b_{-1}) + 0,8 (дробна част)

0,8 x 2 = 1,6 = 1 + 0,6 ($b_{-2} = 1$)

0,6 x 2 = 1,2 = 1 + 0,2 ($b_{-3} = 1$)

0,2 x 2 = 0,4 = 0 + 0,4 ($b_{-4} = 0$)

0,4 x 2 = 0,8 = 0 + 0,8 ($b_{-5} = 0$) - край по зададен брой разряди 5.

Резултат от преобразуването : $N_2 = 0{,}b_{-1}b_{-2}b_{-3}b_{-4}b_{-5} = 0{,}11100$

(в) Превръщане на смесени числа

Извършва се чрез независимо прилагане на двата алгоритъма за двете части – цяла и дробна, след което получените резултати се комплектоват, разделени със запетайка.

(г) Превръщане на числа в ПБС с основа $p=2^k$

Използва се позиционно заместване, като *k* определя размера на необходимия брой разряди за това. При осмична ПБС се прилагат триади (k=3), а при шестнадесетична – тетради (k=4). Заместването се основава на записа на числата в съответните ПБС (Таблица 2.3).

Таблица 2.3

$ПБС_{10}$	$ПБС_2$	$ПБС_8$	$ПБС_{16}$	Примери:
0	0 000	0	0	
1	0 001	1	1	$N_{16} \rightarrow N_2$
2	0 010	2	2	A5C,4B→1010 0101 1100 , 0100 1011
3	0 011	3	3	
4	0 100	4	4	$N_2 \rightarrow N_{16}$
5	0 101	5	5	1110 0100, 1001 1100 → E4,9C
6	0 110	6	6	
7	0 111	7	7	
8	1 000	10	8	$N_8 \rightarrow N_2$
9	1 001	11	9	53,27 → 101 011 , 010 111
10	1 010	12	A	
11	1 011	13	B	$N_2 \rightarrow N_8$
12	1 100	14	C	011 100 , 001 110 → 34,16
13	1 101	15	D	
14	1 110	16	E	
15	1 111	17	F	

Аритметични действия

Изпълнението на аритметичните операции в различните ПБС става по еднакви правила, но отчитайки съответната азбука и основа. В КС се прилага машинна аритметика, основаваща се на двоична ПБС, като най-използвана операция е двоичното сумиране. Известно е, че при поразрядно сумиране на две цифри (разряда) се получава резултат, определящ сумата

и преноса към следващ разряд. Поразрядните двоични операции сумиране, умножение и изваждане са представени в табл.2.4.

Таблица 2.4.

Сумиране			*Изваждане*			*Умножение*		
+	0	1	-	0	1	x	0	1
0	0	1	0	0	1	0	0	0
1	0	10	1	-1	0	1	0	1

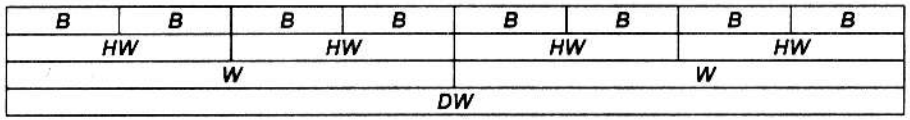
Примери:

Сумиране	*Изваждане*	*Умножение*
1010	10101	10101
+10011	- 100	x 100
11101	10001	00000
		+ 00000
		10101
		1010100

2.1.3. Машинна дума и машинни кодове

Машинна дума

В КС информацията се кодира чрез двоичната азбука, което е свързано с използване на двупозиционни структурни елементи за нейното електронно представяне. Основната организационна единица в компютъра е *машинната дума*, която представлява подредена последователност от двоични разряди с фиксирана дължина n, а всеки разряд може да бъде представен чрез такъв структурен елемент. Въвеждането на машинна дума с дължина n ограничава разрядността при представяне на информацията и определя не повече от 2^n двоични комбинации. За представяне на различни данни в компютъра са въведени и производни на машинната дума. Едно примерно съотношение е представено на фиг.2.1, където: B - байт; HW - полудума; W - дума; DW - двойна дума.



Фиг. 2.1. Машинна дума и производни форми в КС

Представяне на числата

Дължината n на машинната дума определя диапазона на представимите числа - обхват от минималната до максималната стойност, които могат да бъдат записани в n разряда.

Очевидно, за цяло число без знак това са стойностите от 0 до 2^n-1. В общия случай едно *число без знак* N се кодира в доична ПБС като: $N_2=b_k.2^k+b_{k-1}.2^{k-1}+...+b_1.2^1+b_0.2^0$, където $b_i \in \{0,1\}$; за $i=0 \div k$. При представяне на това число в n-разрядна машинна дума са възможни ситуациите:

- $k \le n$ - позволява точно представяне на N_2 в разрядната решетка на машинната дума;
- $k > n$ - дължината на машинната дума е недостатъчна и част от двоичния код ще се загуби; например, при ситуацията:

$N_2=b_k.2^k+b_{k-1}.2^{k-1}+....+b_n.2^n+b_{n-1}.2^{n-1}+...+b_1.2^1+b_0.2^0=N''.2^n+N'$,

частта $N''.2^n=(b_k.2^{k-n}+....+b_n).2^n$ от записа на двоичното число ще се загуби, което ще въведе грешка при представянето.

При представяне на *числа със знак* се налага въвеждане на допълнителен разряд за кодиране на знака. За произволна ПБС с основа p и дължина на представяне n разряда, параметърът $M=p^n$ се нарича модул на системата и определя броя на представимите числа. При използване на знак, множеството на тези числа ще се раздели на две равни части, всяка с мощност M/2, за положителните и за отрицателните числа. Тогава за двоична ПБС:

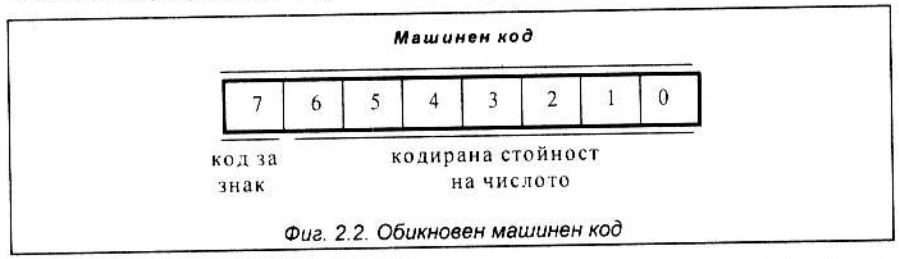
- положителните числа (N≥0) ще се представят чрез диапазона [0, 2^{n-1}-1], понеже N<M/2;
- отрицателните числа (N<0) ще се представят чрез останалите комбинации, т.е. диапазона [2^{n-1}, 2^n-1], което представлява допълнението на числото $N_{доп}$=M-N.

Машинни кодове

За представяне на числа със знак в КС са въведени *машинни кодове*, които позволяват едновременна обработка на всички разряди от записа (знакови и числови). По този начин алгебричните действия над числата се заместват с алгебрични действия над техните машинни кодове.

Машинният код на дадено число N_2 е също двоичен запис, разполаган в разрядната решетка на машинната дума с дължина n. За този случай, абсолютният максимум на представимо число е M/2=2^{n-1}. За кодиране на знака се използва допълнително поле от 1 бит (обикновен код) или 2 бита (модифициран код), като знак "+" се кодира с нулева стойност, а знак "-" с единична стойност. Използват се три основни кода - прав (ПК), обратен (ОК) и допълнителен (ДК). На фиг.2.2 е показана примерна 8-битова машинна дума при представяне на числа чрез обикновен машинен код. В тази машинна дума може да се запише двоичната информация $X=x_7x_6x_5x_4x_3x_2x_1x_0$, където $x_i \in \{0,1\}$ е двоичен разряд с индекс (номер на бит) $0 \le i \le 7$. За представяне на знака се отделя левият бит x_7, а останалите определят числовата стойност. За тази ситуация дефинициите на трите обикновени

кода са представени в табл.2.5, където x_i' е инверсната стойност на съответния разряд, а x_i'' е допълнението до 2 (т.е. $x_i'' = x_i' + 1$).



Фиг. 2.2. Обикновен машинен код

Таблица 2.5

Число	*Прав код*	*Обратен код*	*Допълнителен код*
Положително	$0x_6x_5x_4x_3x_2x_1x_0$	$0x_6x_5x_4x_3x_2x_1x_0$	$0x_6x_5x_4x_3x_2x_1x_0$
Отрицателно	$1x_6x_5x_4x_3x_2x_1x_0$	$1x'_6x'_5x'_4x'_3x'_2x'_1x'_0$	$1\ x'_6x'_5x'_4x'_3x'_2x'_1x_0''$
+1011010	01011010	01011010	01011010
-1011010	11011010	10100101	10100110

Като уточним, че за 8-битова машинна дума модулът на системата е $M=2^8=100000000$, тогава от израза за допълнението $X_{доп}=M-X$ следва $X+X_{дк}=100000000$ и от $X_{доп}=X_{ок}+1$ следва $X+X_{ок}=1111111$. От тук следва, че преходът от прав в обратен код е поразрядна операция и не изисква значително време, за разлика от получаване на допълнителен код, при която е възможно разпространяване на преноса към старшите разряди. Необходимо е да се отбележи, че нулевата стойност има еднакви кодове, независимо от знака, т.е. $[+0]_{пк} = [-0]_{дк} = [-0]_{ок} = 00000000$.

При модифицираните машинни кодове се използват същите правила за формиране на числовите разряди, като за знаковите са в сила: '00' - положителен знак; '11' - отрицателен знак; '01' и '10' - грешен резултат поради препълване на разрядната решетка.

Препълване на разрядната решетка

Целта при машинните кодове (МК) е аритметичните операции над числата със знак да се заменят със съответните операции над кодовете, т.е. $X+Y=Z \Rightarrow [X]_{мк} + [Y]_{мк} = [Z]_{мк}$, при което е необходимо $[X+Y]_{мк} = [X]_{мк} + [Y]_{мк}$.

За осъществяване на това изискване при сумиране на машинни кодове са въведени апаратно реализирани приоми:

- при сумиране в ПК знаковите разряди се обработват отделно от останалите;
- при сумиране в ОК евентуален пренос от знаковите разряди се прибавя към най-младшия разряд на резултата;
- в ДК евентуален пренос от знаковите разряди не се отчита.

Независимо от това, има ситуации, при които се получава некоректен резултат поради т.нар. *препълване*, дължащо се на ограничения диапазон на представимите числа в машинната дума. Обикновено това е свързано с разпространяване на преноса. Различават се два вида:

а) съгласуван пренос - получава се при едновременното му възникване от старшия разряд на числото към знаковия и от знаковия разряд извън разрядната решетка;

б) несъгласуван пренос - пренася се само до знаковия разряд или се поражда само от знаковия разряд извън разрядната решетка.

В първия случай се получава положително препълване (при сумиране на две положителни числа), а във втория случай - отрицателно препълване (при сумиране на две отрицателни числа).

Примери (сумиране):

а) без възникване на пренос (коректен резултат)

$[+122]_{пк}$ $X_{пк} =$ 01111010
$[+5]_{пк}$ $Y_{пк} =$ 00000101
$[+127]_{пк}$ $[X+Y]_{пк} =$ 01111111

б) съгласуван пренос (коректен резултат)

$[-122]_{дк}$ $X_{дк} =$ 10000110
$[-5]_{дк}$ $Y_{дк} =$ 11111011
$[-127]_{дк}$ $[X+Y]_{дк} =$ 110000001

в) несъгласуван пренос от старшия разряд при положително препълване (грешен резултат)

$[+122]_{пк}$ $X_{пк} =$ 01111010
$[+6]_{пк}$ $Y_{пк} =$ 00000110
$[+128]_{пк}$ $[X+Y]_{пк} =$ 10000000

г) несъгласуван пренос от знаковия разряд при отрицателно препълване (грешен резултат)

$[-122]_{дк}$ $X_{дк} =$ 10000110
$[-7]_{дк}$ $Y_{дк} =$ 11111001
$[-129]_{дк}$ $[X+Y]_{дк} =$ 101111111

Възникване на препълване е аварийна ситуация и тя може да бъде предвидена чрез предварителен анализ на стойностите на двете числа и вида на техните знаци.

2.1.4. Формати за представяне на числови и символни данни

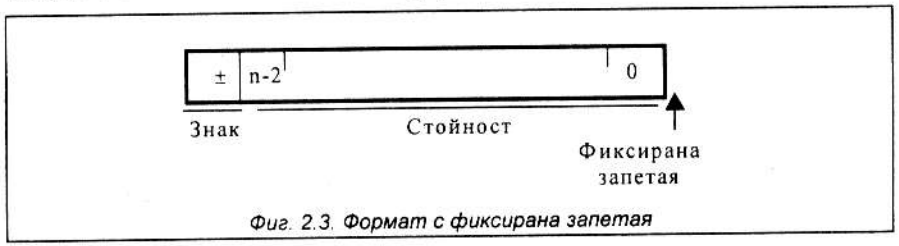
Форматът определя правилата, по които се записват определен тип данни в машинната дума или нейни производни. За представяне на данните се прилагат следните формати.

Представяне на числа с фиксирана запетая

Представянето се извършва в доична ПБС, като съгласно стандарта на IEEE знакът се записва в най-левия бит, а отрицателните числа се представят в допълнителен код. Мястото на запетаята винаги е апаратно фиксирано. На фиг. 2.3 е представен формат за представяне на числата като цели (X*), за което те се обработват с мащабен коефициент 2^{-k} на базата на следния израз:

$$X = \pm 2^{-k} [x_{m}.2^{m+k} + \ldots + x_{-k}.2^{0}] = \pm (2^{-k}).X^{*},$$

където $n\text{-}2=m+k$ и $n=m+k+2$. Такъв формат определя диапазон $1 \le X^{*} \le 2^{n-1}\text{-}1$.



Фиг. 2.3. Формат с фиксирана запетая

Възможно е представяне с фиксирана запетая като дробни числа – мястото на запетаята е между разряди с номера '0' и '1'.

Представяне на числа с плаваща запетая

За представяне на числата се използва преобразуването:

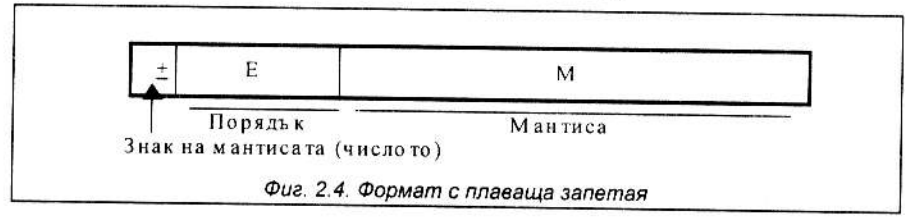
$$A_{p} = \langle знак \rangle\ a_{m}.p^{m} + a_{m-1}.p^{m-1} + \ldots + a_{-k}.p^{-k} =$$
$$= \langle знак \rangle\ (a_{m}.p^{-1} + a_{m-1}.p^{-2} + \ldots + a_{-k}.p^{-k-m-1}).\ p^{m+1} = \langle знак \rangle\ M.p^{E}$$

Този запис се нарича полулогаритмична форма, защото числото се представя чрез мантисата $\pm M = \pm 0,a_{m}a_{m-1} \ldots a_{-k}$, която е дробно число и получения при преобразуването порядък E, който е цяло (положително или отрицателно) число. Приема се, че за дадена КС основата 'p' на използваната ПБС е известна и не е необходимо изричното указване. В случая мястото на запетаята зависи от конкретната стойност на E и с неговата промяна тя се движи (плава) в една от двете посоки, например:

$7{,}56 = 75{,}6.10^{-1} = 756.\ 10^{-2} = 0{,}756.\ 10^{+1} = 0{,}0756.\ 10^{+2}$

Илюстрация на формата е показана на фиг. 2.4. За еднозначност на представянето се изисква мантисата да бъде нормализирано дробно число

(първата цифра след запетаята да бъде значеща). За представяне на отрицателен порядък се използва допълнителен код.



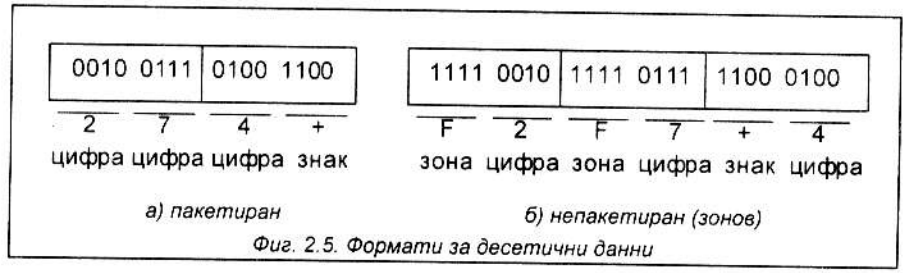
Фиг. 2.4. Формат с плаваща запетая

Представяне на десетични числа

За представяне на десетични числа се използва двоично-десетичен код BCD (Binary Coded Decimal), чрез който всяка десетична цифра d_i от числото се кодира с нейния двоичен еквивалент $b_3b_2b_1b_0$ (виж табл. 2.3). За кодиране на знака се използват кодовите комбинации: $1100_{(2)} = C_{(16)}$ (за знак '+') и $1101_{(2)} = D_{(16)}$ (за знак '-'). Използват се два формата (фиг. 2.5):

а) пакетиран - в един байт се разполагат кодовете на две десетични цифри (използват се при обработка в процесора);

б) непакетиран – кода на всяка десетична цифра се разполага в отделен байт с допълваща зона $1111_{(2)} = F_{(16)}$ (използва се за обмен с периферията).



Фиг. 2.5. Формати за десетични данни

Представяне на символни данни

Поддържането на символни данни в КС е свързано със символна таблица, която описва пълната азбука и използваните цифри. Разположението на символа в таблицата (ред и колона) определя формирането на съответния код за неговото представяне. Обикновено този код е с дължина един байт, а символните кодове се различават по кодовата таблица. Най-разпространени са осембитовите кодове EBCDIC (Extended Binary Coded Decimal Interchange Code) и ASCII-8 (American Standard Code for Information Interchange), както и седембитовия ASCII-7, използващ осмия бит като служебен.