Бройни системи за класически архитектури на цифрови компютри

Кирил Иванов

Април 2022 година

Предговор

В това изложение са събрани най-важните за практиката факти за позиционните бройни системи, подобни на десетичната. Главната цел е тази тематика да бъде представена пределно нагледно, ясно и едновременно достатъчно пълно, така че да може да служи за стабилна основа на бъдещо самообучение на читателите, както в посока на задълбочаване в математическата теория, така и за прецизно практическо овладяване на разнообразните области на компютърния свят.

От гледна точка на компютърната информатика тази тематика е най-полезна за разбирането на архитектурите на хардуера, включително на машинните езици, и за програмирането на ниво, близко до машинното. Обаче и в езиците за програмиране от високо ниво, и въобще в разнообразните аспекти на работата със софтуер често се срещат редица особености, които може да бъдат разбрани само при познаване на основните зависимости, описани по-надолу (а такова разбиране днес е много желателно почти за всеки). Например включването на тип decimal в езика С# и предпочитането на decimal за бизнес изчисления е следствие от получаването на безкрайна периодична дроб при двоичното представяне на голям брой числа с краен десетичен запис. Има и много други подобни особености в компютърната информатика.

Естествено, описаният материал може да бъде използван и извън каквато и да било връзка с компютри. Например той е достъпен за средните училищни класове и може да послужи като база за интересни математически задачи (главно за допълнителни занятия, извън обхвата на актуалните днес учебни програми).

Този текст е предназначен преди всичко за упражнения, в смисъл, че е съсредоточен върху практиката, но той може да служи и за основа или илюстриране на абстрактни теоретични построения.

Необичайното оцветяване цели максимално нагледно поднасяне на съдържанието.

К. Иванов

Съдържание

100 opsicarrae	
1. Бройни системи от типа на Арабската	2
2. Някои свойства на ПБС от типа на Арабската	4
3. Преобразуване на периодична дроб в рационален израз с крайни записи	6
4. Сравняване на памети с близка сложност, базирани на различни ПБС	7
5. Преобразуване на краен непериодичен запис от s-ична в 10-ична ПБС	9
6. Преобразуване на краен непериодичен запис от 10-ична в s-ична ПБС	13
7. Преобразуване на краен непериодичен запис от и към основи степени на едно и също s	20
8. Сумиране в ПБС с основа ѕ	22
9. Изваждане в ПБС с основа s	24
10. Умножаване в ПБС с основа s	26
11. Делене в ПБС с основа s	28
12. Преобразуване от една в друга ПБС на записи, съдържащи период	. 29

1. Бройни системи от типа на Арабската

Бройна система (по-надолу ще пишем само БС) наричаме система от знакове и правила, чрез които може да се представят числа с две основни цели: първо и най-важно – да има удобна възможност за извършване на основните аритметични действия – събиране, изваждане, умножение и деление и второ – да може да се представят всякакви числа.

Първите използвани от хората БС са служили само за представяне на относително малки естествени числа. В най-удобните БС, каквато е Арабската, сложността и на изчисленията, и на представянето на числата нараства малко и плавно с увеличението на големината на числата или с приближаването им към нулата.

Говорим именно за *представяне* на числа, защото знаковете може да имат всякаква природа – визуален образ, звук, електрически ток, магнитно поле, светлинен лъч, наличие или липса на перфорация върху пластина и т. н.

Цифра наричаме знак от БС, който означава някаква стойност. Обикновено болшинството знакове на БС са цифри, но има и други. Например "+", "-", ", ", "(" и ")" не са цифри, но участвуват в записите на числата в дробта $\frac{+9,6(7)_{(16)}}{-38,2(54)_{(9)}}.$

Непозиционна бройна система наричаме тази, в която стойността на цифрата винаги участвува по един и същ начин във формирането, означаването на представеното число, независимо къде, в коя позиция се намира самата цифра в това представяне. Например в някои от най-ранните известни днес БС всяка резка върху пръчка добавя по една единица към записаното върху пръчката число.

Позиционна бройна система (ПБС) наричаме тази, в която стойността на една и съща цифра може да участвува по различни начини във формирането, означаването на представеното число, в зависимост от различните места на цифрата в представянето. Например в записа 101 (сто и едно) първата единица се умножава по сто, а втората – по едно.

По-надолу ще работим само с визуални изображения, означения на числа (и ще ги наричаме записи на числата).

Това изложение разглежда само ПБС, аналогични на привичната за нас Арабска ПБС, защото именно върху такива ПБС се градят класическите архитектури на цифрови компютри.

При тези ПБС се използват записи от вида:

(1)
$$\overline{a_n \dots a_1 a_0, c_1 \dots c_k \dots}_{(s)}$$

където:

- може да има още знак плюс или минус, например $+34{,}01_{(5)}$ и -0,f $9_{(16)}$;
- ullet може да има период в дробната част, означен с кръгли скоби, например $\ 210,\!21\,(304)_{(8)}$;
- хоризонталната линия означава, че под нея поне една цифра е заменена с някакво означение (във формулата 1 всички цифри са означени с индексирани букви);
- запетаята отделя цялата от дробната част;
- с индекс *S* в кръгли скоби в края на записа е означена **основата на ПБС**, т. е. естествено число, по-голямо от единица, което участвува в дефиницията на записаното число, както е показано във формулите *2* и *3* (или еквивалентните им формули *4* и *5*);
- възможните стойности на цифрите са числата 0, 1, 2, ..., s-1;
- при основа до 10 се използват цифрите 0, 1, ..., 9;
- при основи от 11 до 36 за цифри се използват латинските букви със стойност 10 на цифрата a или A , 11 на b или B , 12 на c или C , ..., 35 на z или Z .

Числото, записвано чрез горния запис, се дефинира с формулите

(2)
$$\overline{a_n...a_1 a_0}_{(s)} \stackrel{\text{def}}{=} a_n.s^n + ... + a_1.s^1 + a_0.s^0$$
 за цялата част и

(3)
$$\overline{0, c_1 c_2 \dots c_k \dots}_{(s)} \stackrel{\text{def}}{=} \frac{c_1}{s^1} + \frac{c_2}{s^2} + \dots + \frac{c_k}{s^k} + \dots$$
 за дробната част.

Понякога, например при изчисления с калкулатор или алгоритъм, е по-удобно да се използват еквивалентни на горните формули

(4)
$$\overline{a_n...a_1 a_0}_{(s)} = \left(\left(... \left(a_n.s + a_{n-1} \right).s + ... + a_2 \right).s + a_1 \right).s + a_0$$
 за цялата част и

$$c_{k-2} + \frac{c_{k-1} + \frac{c_k + \dots}{s}}{s}$$

$$c_1$$
 (5) $c_1 c_2 \dots c_{k-2} c_{k-1} c_k \dots c_s$ $c_2 + \frac{c_2 + c_2 + c_2 + c_2 + \frac{c_2 + \frac{c_2 + c_2 + c_2$

Когато числото се записва с една цифра, основата на ПБС може да се пропуска, защото смисълът на записа е еднозначен.

Пример 1

$$2504_{(6)} = 2.6^3 + 5.6^2 + 0.6^1 + 4.6^0 = ((2.6+5).6+0).6+4$$

Пример 2

$$0,215_{(7)} = \frac{2}{7^1} + \frac{1}{7^2} + \frac{5}{7^3} = \frac{2 + \frac{1 + \frac{5}{7}}{7}}{7}$$

ПБС от описания вид с основа S наричаме S -ична ПБС.

Арабска ПБС наричаме 10-ичната ПБС.

Прочитането на запис в такава ПБС с основа, различна от 10, става знак по знак. Думите сто, осемдесет, хиляда, петнадесет и т. н. означават (по подразбиране), че записът е в десетична ПБС (Арабската). При четенето на цифри букви в математиката в България е прието да се използват латинските им названия, т. е. "а", "бе", "це", "де" и т. н., вместо английските названия на същите букви "ей", "би", "си", "ди" и т. н.

Например:

 $-21.2(01)_{(3)}$ прочитаме минус две едно запетайка две и нула едно в период троично или минус две едно цяло и две и нула едно в период троично.

4f , $8c(2a)_{(16)}$ прочитаме четири еф запетайка осем це и две а в период шестнадесетично или четири еф цяло осем це и две а в период шестнадесетично. В този запис няма нужда от хоризонтална черта, защото буквите f, c и a са самите цифри, а не буквени означения на цифри.

 $21099_{(10)}$ прочитаме двадесет и една хиляди деветдесет и девет.

 $3451_{(7)}\,$ прочитаме три четири пет едно седмично.

Параметричния запис $-\overline{xycx}_{(9)}$ прочитаме: минус хикс игрек це хикс деветично.

2. Някои свойства на ПБС от типа на Арабската

По-надолу са изброени някои прости следствия от дефинициите на записа на число в описваните ПБС. Тези свойства се използват много често в компютърните архитектури, а понякога са определящи за избора на отделни решения.

Във всяка s -ична ПБС и за всяко естествено число k са верни следните твърдения:

(6)
$$s^k = 1 \underbrace{0 \dots 0}_{k} (s)$$
.

(7)
$$\frac{1}{s^k} = 0, \underbrace{0...0}_{k-1} 1_{(s)}.$$

(8)
$$\overline{a_{k-1}...a_0}_{(s)} < 1 \underbrace{0...0}_{k}_{(s)} \le \overline{1d_{k-1}...d_0}_{(s)}$$
,

каквито и да са цифрите a_{k-1} , . . . , a_0 , d_{k-1} , . . . , d_0

От свойствата (8) и (6) следва, че в k двоични разряда може да се запишат естествените числа от интервала $\begin{bmatrix} 0; 2^k - 1 \end{bmatrix}$, а тези числа са равни на своите машинни кодове, чрез които обикновено се представяния във формат на цели числа без знак. Затова във всички масово разпространени езици който и да било целочислен тип без знак в k разряда има възможни стойности целите числа от интервала $\begin{bmatrix} 0; 2^k - 1 \end{bmatrix}$.

(9)
$$0, \underbrace{0, 0, 0, 0}_{k} c \dots c_{(s)} < 0, \underbrace{0, 0, 0}_{k-1} 1_{(s)} \le 0, c_{1} \dots c_{k-1} 1_{(s-1)} c_{k+1} \dots c_{(s)}$$

каквито и да са цифрите $C, C_0, \ldots, C_{k-1}, C_{k+1}$

(10)
$$0 \le \overline{a_{k-1}...a_0}_{(s)} < s^k$$
, каквито и да са цифрите $a_{k-1},...,a_0$.

(11)
$$(n>k \land d_n\neq 0) \Rightarrow \overline{d_n...d_0}_{(s)} > \overline{c_k...c_0}_{(s)}$$

каквито и да са цифрите $d_n, ..., d_0, a_{n-1}, ..., a_0$

(12)
$$(n>k \land c_{k+1}\neq 0) \Rightarrow \overline{0, \underbrace{0...0}_{k} c_{k+1}...}_{(s)} > \overline{0, \underbrace{0...0}_{n} d_{n+1}...}_{(s)}$$

за всякакви цифри c_{k+1} , d_{n+1}

Едно и също цяло число може да се запише с толкова по-малко цифри, колкото по-голяма е основата на ПБС, т. е.

(13)
$$\left(\overline{a_k...a_0}_{(s)} = \overline{d_n...d_0}_{(t)} \land a_k \neq 0 \land d_n \neq 0 \land s < t\right) \Rightarrow k \geq n$$

Ако две различни положителни числа са записани в една и съща *S* -ичната ПБС с еднакъв брой цифри, евентуално с добавяне на незначещи нули, и дробната запетайка е на една и съща позиция спрямо първата цифра, тогава по-голямото от числата може да бъде определено чрез лексикографско сравняване (точно както се сравняват низове). Т. е., сравняват се последователно отляво надясно цифрите от едни и същи позиции и първата намерена двойка различни цифри определя кое число е по-голямо. Това може да се изрази с формули по следния начин:

$$(14) \quad X = \overline{a_n \dots a_k d_x \dots, \dots}_{(s)} \land Y = \overline{a_n \dots a_k d_y \dots, \dots}_{(s)} \land d_x > d_y \quad \Rightarrow \quad X > Y$$

(когато има разлика в цялата част) и

(15)
$$X = \overline{a_n ... a_0, c_1 ... c_k d_x ...}_{(s)} \land Y = \overline{a_n ... a_0, c_1 ... c_k d_y ...}_{(s)} \land d_x > d_y \Rightarrow X > Y$$

(когато първата, най-лявата разлика е в дробната част).

Лексикографското сравняване често се използва в хардуера, защото е по-лесно от числовото. Например на хардуерно ниво, в аритметичния блок на процесора, така се сравняват абсолютните стойности на числа от типа double на езика C++.

Всяко число от интервала $[0\,;s^k-1]$ може да се представи във вида $\overline{a_{k-1}\dots a_0}_{(s)}$ и всяко число от вида $\overline{a_{k-1}\dots a_0}_{(s)}$ принадлежи на интервала $[0\,;s^k-1]$, т. е.

(16)
$$\left\{\overline{a_{k-1}...a_0}_{(s)}:0\leq a_{k-1}\leq s-1,...,0\leq a_0\leq s-1\right\}\equiv \left[0;s^k-1\right].$$

Броят на целите неотрицателни $\,k\,$ -цифрени $\,S\,$ -ични числа е точно $\,S^{\,k}\,$. Т. е.

(17)
$$\left| \left\{ \overline{a_{k-1} ... a_0}_{(s)} : 0 \le a_{k-1} \le s-1, ..., 0 \le a_0 \le s-1 \right\} \right| = s^k$$

Затова в двоичните компютри в k разряда може да се представят най-много $\left[0;2^k-1\right]$ различни стойности.

(18)
$$\overline{a_{n-1}...a_0}_{(s)}.s^k = \overline{a_{n-1}...a_0}\underbrace{0...0}_{k}_{(s)}.$$

(19)
$$\overline{a_n...a_k a_{k-1}...a_0} |_{(s)} = \overline{a_n...a_k} |_{(s)}.s^k + \overline{a_{k-1}...a_0} |_{(s)}.$$

(20)
$$\frac{a_{n-1}...a_k a_{k-1}...a_0}{s^k} = \overline{a_{n-1}...a_k, a_{k-1}...a_0}_{(s)}.$$

(21)
$$\overline{0, a_1 \dots a_k c_1 \dots c_{k+n} \dots}_{(s)} \cdot s^k = \overline{a_1 \dots a_k, c_1 \dots c_{k+n} \dots}_{(s)}$$

(22)
$$\overline{0, c_1...c_k c_{k+1}...c_n}_{(s)} = \overline{0, c_1...c_k}_{(s)} + \frac{\overline{0, c_{k+1}...c_n}_{(s)}}{s^k}.$$

Ако ξ е най-голямата цифра в s -ичната ПБС, т. е. $\xi_{(s)} = s - 1$, тогава

(23)
$$\underbrace{\xi ... \xi}_{k} (s) = s^{k} - 1$$
 (следствие от това, че $\underbrace{\xi ... \xi}_{k} (s)$ е най-голямото k -цифрено s -ично число

и че
$$1\underbrace{0...0}_{k}(s) = s^{k}$$
 е най-малкото $(k+1)$ -цифрено s -ично число) и

(24)
$$0, \underbrace{\xi ... \xi}_{k} (s) = \frac{s^{k} - 1}{s^{k}}$$
 (следствие от формули 20 и 23).

3. Преобразуване на периодична дроб в рационален израз с крайни записи

Във всяка s -ична ПБС и за всяко естествено число k е вярно равенството

(25)
$$\overline{0,(c_1...c_k)}_{(s)} = \frac{c_1...c_{k}_{(s)}}{s^k-1}.$$

Доказателство

$$\overline{0,(c_{1}...c_{k})}_{(s)} = \overline{0,c_{1}...c_{k}}_{(s)} + \overline{0,0}...0_{(c_{1}...c_{k})}_{(s)} = \overline{0,c_{1}...c_{k}}_{(s)} + \frac{0,(c_{1}...c_{k})_{(s)}}{s^{k}}.$$

$$\Rightarrow \overline{0,(c_{1}...c_{k})}_{(s)} \cdot \left(1 - \frac{1}{s^{k}}\right) = \overline{0,c_{1}...c_{k}}_{(s)}.$$

$$\Rightarrow \overline{0,(c_{1}...c_{k})}_{(s)} = \frac{s^{k}.\overline{0,c_{1}...c_{k}}_{(s)}}{s^{k}-1} = \frac{\overline{c_{1}...c_{k}}_{(s)}}{s^{k}-1}.$$

Пример 3

$$0,(613)_{(9)} = \frac{613_{(9)}}{9^3 - 1} = \frac{6 \cdot 9^2 + 1 \cdot 9 + 3}{9^3 - 1} = \frac{486 + 9 + 3}{729 - 1} = \frac{498}{728} = \frac{249}{364}$$

Пример 4

$$0,(002)_{(7)} = \frac{2}{7^3 - 1} = \frac{2}{342} = \frac{1}{171}$$

Пример 5

$$0,(0001)_{(2)} = \frac{1}{2^4 - 1} = \frac{1}{15} = 0,00(6)_{(10)}$$

От формули *21, 22* и *25* следва

(26)
$$\overline{0, f_1...f_j(c_1...c_k)}_{(s)} = \frac{\overline{f_1...f_j(s)}.(s^k-1) + \overline{c_1...c_k(s)}}{s^j.(s^k-1)}.$$

Доказателство

$$\frac{\overline{0, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)}}{s^{j}} = \frac{\overline{f_{1}...f_{j}(s)}}{s^{j}} + \frac{\overline{0, (c_{1}...c_{k})}_{(s)}}{s^{j}}$$

$$= \frac{\overline{f_{1}...f_{j}(s)}}{s^{j}} + \frac{\overline{c_{1}...c_{k}(s)}}{s^{j}(s^{k}-1)} = \frac{\overline{f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)}$$

Пример 6

$$0,0001(4)_{(5)} = \frac{1.(5^{1}-1)+4}{5^{4}.(5^{1}-1)} = \frac{4+4}{5^{4}.4} = \frac{2}{5^{4}} = \frac{2^{5}}{5^{4}.2^{4}} = \frac{32}{10^{4}} = 0,0032$$

От формули 18 и 26 следва

(27)
$$\overline{a_n...a_0, f_1...f_j(c_1...c_k)}_{(s)} = \frac{\overline{a_n...a_0 f_1...f_j(s)}.(s^k-1) + \overline{c_1...c_k(s)}}{s^j.(s^k-1)}.$$

Пример 7

$$1,0(12)_{(3)} = \frac{10_{(3)} \cdot (3^2 - 1) + 12_{(3)}}{3 \cdot (3^2 - 1)} = \frac{3 \cdot 8 + 5}{3 \cdot 8} = \frac{29}{24} = 1\frac{5}{24}$$

4. Сравняване на памети с близка сложност, базирани на различни ПБС

За съхраняване и разпознаване на стойност в един разряд на компютърната памет трябва да се поддържат и разпознават различни устойчиви състояния за всички различни цифри, които може да съдържа разрядът, т. е. толкова на брой състояния, колкото е основата на ПБС, върху която е базирана паметта. Следователно с нарастването на основата на ПБС расте и сложността на паметта.

Също така, очевидно, компютърната памет се усложнява и с увеличаването на броя на разрядите.

Затова, като *относителна мярка за сложност* на компютърната памет, може да се приеме произведението на броя на разрядите и основата на ПБС, върху която е базирана паметта. Тогава за практиката има значение въпросът: Измежду паметите с приблизително еднаква сложност, при коя ПБС може да се съхранят в паметта най-голям брой различни стойности?

Ако означим с S основата на ПБС и с n броя на разрядите в паметта, то от формула (17) следва, че в тази памет може да се съхраняват S^n различни числа. Тогава горният въпрос може да се формулира така: При фиксирано произведение S.n, при кое S е най-голяма степента S^n ?

Следващата таблица показва сравнението по такъв критерий на няколко памети със сложност 30:

s.n=30							
S	2	3	5	6	10	15	30
$\Rightarrow n$	15	10	6	5	3	2	1
$\Rightarrow s^n$	32768	59049	15625	7776	1000	225	30

А ето и капацитетите на памети със сложност 48:

s.n=48									
S 2 3 4 6				8	12	16	24	48	
$\Rightarrow n$	24	16	12	8	6	4	3	2	1
$\Rightarrow s^n$	16777216	43046721	16777216	1679616	262144	20736	4096	576	48

Може да се докаже, че при фиксирано s.n степента s^n нараства с приближаването на s^n към Неперовата единица $e=2,7\dots$ Следователно при основа 3 на ПБС се получава най-голям капацитет на компютърната памет по горния критерий.

Нещо повече, съотношението на капацитетите на памети, базирани върху основи 3 и 2 нараства значително при увеличаването на сложността. Това може да покажем така:

Нека означим с C(s;N) броя на различните числа (на редиците от цифри), които може да бъдат записани в памет, базирана на основа s и със сложност N .

Следователно $C(s;s.n)=s^n$.

Тогава
$$R = \frac{C(3; 2.3.k)}{C(2; 2.3.k)} = \frac{3^{2.k}}{2^{3.k}} = \left(\frac{9}{8}\right)^k$$
.

Очевидно е, че отношението на капацитетите на троична и двоична памети с еднаква сложност нараства с увеличаването на сложността, при това, много бързо.

Например ето някои съответствия между k , сложността на паметта и отношението R :

k	10	20	60	100	200		
сложност	60	120	360	600	1200		
R >	3	10	1172	130392	17002175293		

Въпреки това, масово разпространените днес компютри са базирани на 2-ична ПБС, защото получаваната при това функционалност все още е задоволителна, а при основа 2 много се опростява хардуерът и технологията на производството му, а се подобряват и някои от експлоатационните им характеристики. Например в сложните схеми, дори при преминаване от основа 3 към основа 2, намалява топлинното отделяне при функционирането им. Много съществено се оказва и това, че съвременните 2-ични памети, въпреки че имат капацитет, много по-малък от този на 3-ичните памети, все пак са достатъчни за обичайните потребности на отделните потребители и на стопанските приложения.

Все пак, обективно погледнато, при голям брой n на разрядите, произведението S.n, би било точна мярка за сложност на паметта, само когато схемата се усложнява съизмеримо и при добавянето на един разряд, и при реализирането на още една възможна стойност на цифра. В практиката има различия между усложняването в двата случая. Освен това, основата на ПБС твърде много влияе и върху сложността на схемите, обработващи числата. Важно значение имат и броят и разнообразието на схемите. В крайна сметка, сложността на цялата компютърна конфигурация трябва да бъде оценявана комплексно и оценката е относителна.

По съвсем аналогични съображения — за да се избегне загубата на точност в много голям брой изчисления с дроби — всички съвременни аритметични блокове на процесори за универсалните компютри, а също и някои (и съвременни, и вече излезли от употреба) езици за програмиране, поддържат формати за представяне на числа в паметта с точно съхраняване на стойностите на десетичните цифри. Аритметиката с такива формати изцяло следва правилата на 10-ичната ПБС и избягва закръглянията, които биха били предизвикани при преминаването от 10-ичен към 2-ичен запис.

За тази цел на ниско ниво обикновено се използват формати със съхраняване на редици от стойности на десетични цифри, но с тях се работи ограничено. Например в масово разпространените процесори от серията х86 с формат във вид на редица от десетични цифри има аритметика само с една или две цифри, а представяне на числа има само за цели числа в 1 или в 10 байта. В процесорите на масово разпространените универсални компютри няма машинни команди за пълноценна аритметика с такива представяния на дробни числа. Дори за 10-байтовия формат за цели числа изчисленията привличат преобразуване в друг формат (с плаваща запетая).

Пълната поддръжка на аритметика с формати във вид на редици от стойности на десетични цифри се осигурява в някои по-специални архитектури (свързвани с представата за повишена изчислителна мощ).

В някои типове на числа от езиците от високо ниво (от компилаторите за тях) се поддържат особени формати, за представяне на дробни стойности, в които по подходящ начин се съхраняват точно стойностите на десетичните цифри от допустимите десетични записи на кодираните числа. Подобен числов тип е decimal в С#. В него например за числото 1,23 се съхраняват поотделно стойността 123 и позицията (-2) на запетаята. Съответно за 0,0123 ще се съхраняват стойностите 123 и -4, а за 12,3 ще се запомнят в паметта 123 и -1.

5. Преобразуване на краен непериодичен запис от s-ична в 10-ична ПБС

Универсалният алгоритъм за това е да се заместят стойностите на цифрите и на основата *S* съответно във формулите 2, 3, 4 или 5, в тези от тях, с които е най-удобно да се работи, и да се извършат действията.

Пример 8

Търсим $1001_{(2)} = ?_{(10)}$:

$$1001_{(2)} = 1.2^3 + 0.2^2 + 0.2^1 + 1.2^0 = 9$$
или
 $1001_{(2)} = ((1.2+0).2+0).2+1 = 9.$

Пример 9

Търсим $1001_{(3)} = ?_{(10)}$:

$$1001_{(3)} = 1.3^3 + 0.3^2 + 0.3^1 + 1.3^0 = 28_{(10)}$$
 или $1001_{(3)} = ((1.3+0).3+0).3+1 = 28_{(10)}$.

Вижда се, че при представянето на записа на числото в израз със степени на основата на ПБС всяка стойност на цифра се умножава по степен с такъв степенен показател, колкото е броя на цифрите в записа след умножаваната. Например първата цифра на цялата част винаги се умножава по степен на основата на ПБС със степенен показател, с единица по-малък от броя на цифрите в цялата част. Това може да се изрази с формула по следния начин:

(28)
$$\underbrace{\overline{d \dots}}_{n \text{ } uu\phi pu} (s) = d \cdot s^{n-1} + \dots$$

Пример 10

Търсим $1001_{(5)} = ?_{(10)}$:

$$1001_{(5)} = 1.5^3 + 0.5^2 + 0.5^1 + 1.5^0 = 126_{(10)}$$
или
 $1001_{(5)} = ((1.5+0).5+0).5 + 1 = 126_{(10)}$.

Търсим $1322_{(4)} = ?_{(10)}$:

$$1322_{(4)} = 1.4^3 + 3.4^2 + 2.4^1 + 2.4^0 = 64 + 48 + 8 + 2 = 122_{(10)}$$
 или $1322_{(4)} = ((1.4+3).4+2).4 + 2 = (7.4+2).4 + 2 = 30.4 + 2 = 122_{(10)}$.

Пример 12

Търсим $211202_{(3)} = ?_{(10)}$:

$$211202_{(3)} = 2.3^{5} + 1.3^{4} + 1.3^{3} + 2.3^{2} + 0.3^{1} + 2.3^{0} = 486 + 81 + 27 + 18 + 0 + 2 = 614_{(10)}$$
или
$$211202_{(3)} = \left(\left(\left(2.3 + 1\right).3 + 1\right).3 + 2\right).3 + 0\right).3 + 2$$

$$= \left(\left((7.3 + 1).3 + 2\right).3 + 0\right).3 + 2 = \left((22.3 + 2).3 + 0\right).3 + 2 = (68.3 + 0).3 + 2 = 204.3 + 2 = 614_{(10)}\right)$$

Пример 13

Търсим $f9ac_{(16)} = ?_{(10)}$:

$$\begin{split} f9ac_{(16)} &= 15 \cdot 16^3 + 9 \cdot 16^2 + 10 \cdot 16^1 + 12 \cdot 16^0 \\ &= 15 \cdot 4096 + 9 \cdot 256 + 10 \cdot 16 + 12 \cdot 1 = 61440 + 2304 + 160 + 12 = 63916_{(10)} \\ \text{или} \\ f9ac_{(3)} &= \left(\left(15 \cdot 16 + 9\right) \cdot 16 + 10\right) \cdot 16 + 12 = \left(249 \cdot 16 + 10\right) \cdot 16 + 12 = 3994 \cdot 16 + 12 = 63916 \cdot 16 + 12 =$$

При преобразуване на дроби от 2-ична или от 5-ична към 10-ична система е удобно, вместо делене, да се използват зависимостите

(29)
$$\frac{A}{2^k} = \frac{A.5^k}{2^k.5^k} = \frac{A.5^k}{10^k} = \overline{0, c_1...c_k}_{(10)}$$
, където $A.5^k = \overline{c_1...c_k}_{(10)}$, и

(30)
$$\frac{A}{5^k} = \frac{A \cdot 2^k}{5^k \cdot 2^k} = \frac{A \cdot 2^k}{10^k} = \overline{0, c_1 \dots c_k}$$
 (10), където $A \cdot 2^k = \overline{c_1 \dots c_k}$ (10), като в някои случаи $c_1 = c_2 = \dots = c_{k-j} = 0$ за някакво j .

Търсим $0,101_{(2)}=?_{(10)}$:

$$0,101_{(2)} = \frac{1}{2^1} + \frac{0}{2^2} + \frac{1}{2^3} = \frac{5}{8} = 0,625_{(10)}$$

или

$$0,101_{(2)} = \frac{1 + \frac{0 + \frac{1}{2}}{2}}{2} = \frac{5}{8} = 0,625_{(10)}.$$

Пример 15

Търсим $0,101_{(5)} = ?_{(10)}$:

$$0,101_{(5)} = \frac{1}{5^1} + \frac{0}{5^2} + \frac{1}{5^3} = \frac{26}{125} = 0,184_{(10)}$$

или

$$0,101_{(5)} = \frac{1 + \frac{0 + \frac{1}{5}}{5}}{5} = \frac{26}{125} = 0,184_{(10)}.$$

Пример 16

Търсим $0,1_{(9)}=?_{(10)}$:

$$0,1_{(9)} = \frac{1}{9} = 0,(1)_{(10)}$$
.

Пример 17

Търсим $0,72_{(8)} = ?_{(10)}$:

$$0,72_{(8)} = \frac{7}{8^1} + \frac{2}{8^2} = \frac{58}{64} = 0,90625_{(10)}$$

или

$$0,72_{(8)} = \frac{7 + \frac{2}{8}}{8} = \frac{\frac{56 + 2}{8}}{8} = \frac{58}{64} = 0,90625_{(10)}.$$

Търсим $-101,11_{(2)}=?_{(10)}$:

$$-101,11_{(2)} = -\left(1.2^2 + 0.2^1 + 1.2^0 + \frac{1}{2^1} + \frac{1}{2^2}\right) = -\left(4 + 0 + 1 + \frac{1}{2} + \frac{1}{4}\right) = -\left(5 + \frac{3}{4}\right) = -5,75_{(10)}$$

или

$$-101,11_{(2)} = -\left((1.2+0).2+1+\frac{1+\frac{1}{2}}{2}\right) = -\left(4+1+\frac{\frac{3}{2}}{2}\right) = -\left(5+\frac{3}{4}\right) = -5,75_{(10)}.$$

Пример 19

Търсим $0, fa_{(16)} = ?_{(10)}$:

$$0, fa_{(16)} = \frac{15}{16^1} + \frac{10}{16^2} = \frac{240 + 10}{256} = \frac{250}{256} = 0,9765625_{(10)}$$

или

$$0, fa_{(16)} = \frac{15 + \frac{10}{16}}{16} = \frac{\frac{240 + 10}{16}}{16} = \frac{250}{256} = 0,9765625_{(10)}.$$

Пример 20

Търсим $-14323,001_{(5)} = ?_{(10)}$:

$$-14323,001_{(5)} = -\left(1.5^4 + 4.5^3 + 3.5^2 + 2.5^1 + 3.5^0 + \frac{1}{5^3}\right) = -\left(625 + 500 + 75 + 10 + 3 + \frac{1}{125}\right) = -1213,008_{(10)}$$
 или
$$-14323,001_{(5)} = -\left(\left((1.5 + 4).5 + 3\right).5 + 2\right).5 + 3 + \frac{0 + \frac{0}{5}}{5}\right) = -\left((9.5 + 3).5 + 2\right).5 + 3 + \frac{0 + \frac{1}{25}}{5}\right)$$

$$= -\left((48.5 + 2).5 + 3 + \frac{1}{125}\right) = -\left(242.5 + 3 + \frac{1}{125}\right) = -\left(1213 + \frac{1}{125}\right) = -1213,008_{(10)}.$$

Пример 21

Търсим $-1,22_{(3)}=?_{(10)}$:

$$-1,22_{(3)} = -\left(1.3^{0} + \frac{2}{3^{1}} + \frac{2}{3^{2}}\right) = -\left(1 + \frac{8}{9}\right) = -1,(8)_{(10)}$$

или

$$-1,22_{(3)} = -\left(1 + \frac{2 + \frac{2}{3}}{3}\right) = -\left(1 + \frac{8}{9}\right) = -1,(8)_{(10)}$$

Търсим $243,15_{(6)} = ?_{(10)}$:

$$243,15_{(6)} = 2.6^2 + 4.6^1 + 3.6^0 + \frac{1}{6^1} + \frac{5}{6^2} = 99 + \frac{11}{36} = 99,30(5)_{(10)}$$

или 243,15₍₆₎=
$$(2.6+4).6+3+\frac{1+\frac{5}{6}}{6}=99+\frac{11}{36}=99,30(5)_{(10)}$$
.

6. Преобразуване на краен непериодичен запис от 10-ична в s-ична ПБС

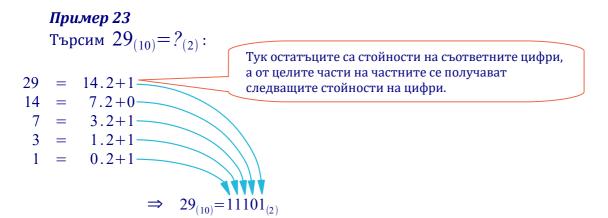
Начините за тези преобразувания следват от желанието да правим всички изчисления само в 10-ична ПБС.

Цифрите на търсения запис се получават една след друга, като първа се получава тази до дробната запетая, а последна — цифрата, която е най-далеч от дробната запетая.

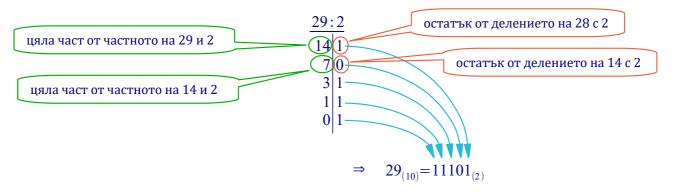
Универсалният алгоритъм за преобразуване на **цяло число** от 10-ична към друга ПБС е следствие от формула 20 при k=1 , т. е.

(31)
$$\frac{\overline{a_{n-1}...a_1a_0}_{(s)}}{s} = \overline{a_{n-1}...a_1, a_0}_{(s)}.$$

Това равенство е вярно в каквато и ПБС да са записани числата. Следователно може в 10-ична ПБС да се раздели числото с основата, към която се преминава, и остатъкът от делението е стойността на последната цифра в записа при новата основа, а от цялата част на частното може да се получат останалите търсени цифри.



Такива изчисления се подреждат много по-удобно в таблица от следния вид:



Търсим $29_{(10)} = ?_{(3)} = ?_{(5)} = ?_{(7)}$:

Пример 25

Търсим $-735_{(10)} = ?_{(2)} = ?_{(3)} = ?_{(7)}$:

$$\frac{735:2}{367 \mid 1} \Rightarrow -735_{(10)} = -1011011111_{(2)}; \frac{735:3}{245 \mid 0} \Rightarrow -735_{(10)} = -1000020_{(3)};$$

$$\frac{735:2}{367 \mid 1} \Rightarrow -735_{(10)} = -1000020_{(3)};$$

$$\frac{81 \mid 2}{27 \mid 0} \Rightarrow 0$$

$$\frac{9 \mid 0}{45 \mid 1} \Rightarrow 0 \mid 1$$

$$\frac{11 \mid 0}{5 \mid 1} \Rightarrow 0 \mid 1$$

Пример 26

Търсим $347_{(10)} = ?_{(8)}$:

$$\begin{array}{c|c}
347:8 \\
43 & 3 \\
5 & 3 \\
0 & 5
\end{array} \Rightarrow 347_{(10)} = 533_{(8)}.$$

Търсим $347_{(10)} = ?_{(16)}$:

Пример 28

Търсим $-249_{(10)} = ?_{(16)}$:

$$\frac{249:16}{15|9}$$
 \Rightarrow $-249_{(10)} = -f 9_{(16)}$ (16-ичната цифра f има стойност 15).

Универсалният алгоритъм за преобразуване на **правилна дроб** от 10-ична към друга ПБС е следствие от формула 21 при k=1 , т. е.

(32)
$$\overline{0}, c_1 c_2 \dots c_n \dots (s). s = \overline{c_1}, c_2 \dots c_n \dots (s).$$

Това равенство е вярно в каквато и ПБС да са записани числата. Следователно може в 10-ична ПБС да се умножи числото с основата, към която се преминава, и цялата част от произведението е стойността на първата цифра след дробната запетая в записа при новата основа, а от дробната част на произведението може да се получат останалите търсени цифри.

Пример 29

Търсим $0,3125_{(10)} = ?_{(2)}$:

Тук целите части са стойности на съответните цифри, а от дробните части се получават следващите стойности на цифри.

$$2.0,3125 = 0,625$$

 $2.0,625 = 1,25$ $\Rightarrow 0,3125_{(10)} = 0,0101_{(2)}$

2.0,25 = 0,5

2.0,5 = 1,0

Такива изчисления се подреждат много по-удобно в таблица от следния вид:

Търсим $0,1875_{(10)} = ?_{(2)}$:

$$\frac{2.0,1875}{0|375} \Rightarrow 0,1875_{(10)} = 0,0011_{(2)}.$$

$$0|75$$

$$1|5$$

$$1|0$$

Пример 31

Търсим $0,1376_{(10)} = ?_{(5)}$:

$$\frac{5.0,1376}{0|688} \Rightarrow 0,1376_{(10)} = 0,0321_{(5)}.$$

$$\frac{3|44}{2|2}$$

$$1|0$$

Пример 32

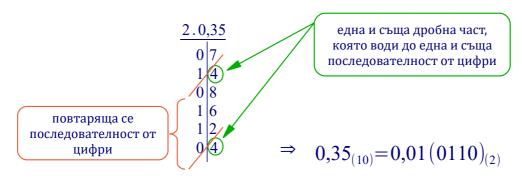
Търсим $0.0390625_{(10)} = ?_{(16)}$:

$$\frac{16.0,0390625}{0\,|\,625}$$
 \Rightarrow 0,0390625 $_{(10)}$ =0,0 $a_{(16)}$ (16-ичната цифра a има стойност 10).

Когато се преобразува крайна дроб от една в друга ПБС, е възможно да се получи безкрайна периодична дроб. Достигането на края на период се разпознава по това, че два пъти се получава една и съща дробна част, от която се пораждат една и съща последователност от цифри.

Пример 33

Търсим $0,35_{(10)} = ?_{(2)}$:



Пример 34

Търсим
$$0,1_{(10)}=?_{(3)}:$$

$$\frac{3.0,1}{0|3} \Rightarrow 0,1_{(10)}=0,(0022)_{(3)}.$$
0 9
2 7
2 1

Търсим $-0.14_{(10)} = ?_{(5)}$:

$$\begin{array}{ccc} \frac{5.0,14}{0|7} & \Rightarrow & -0.14_{(10)} = -0.03(2)_{(5)} \ . \\ \frac{3}{2} & 5 \end{array}$$

При преобразуването на крайна 10-ична дроб към 2-ична основа също може да се получи период. В такъв случай, както беше казано на страница 7, съхраняването на краен брой двоични цифри в компютърната памет ще изисква закръгляне, т. е. губи се точност. Обаче при въвеждането от клавиатурата е необходимо да се работи с 10-ични записи, защото те са привични за нас. Затова, ако паметта е 2-ична, когато трябва да се избегнат закръглянията, се използва специално представяне на числата в паметта с точно представяне (по подходящ начин) на стойностите на десетичните цифри, а отделно в паметта се описва местоположението на дробната запетая. Съответно изчисленията с такива представяния се правят само по правилата на 10-ичната ПБС.

Пример 36
 Търсим
$$-0.8_{(10)} = ?_{(7)}$$
:
$$\frac{7.0.8}{5 \mid 6} \Rightarrow -0.8_{(10)} = -0, (5412)_{(7)}.$$
1 | 4 | 2 | 8

Тъй като дробната част на числото се записва като дробна част във всяка ПБС и цялата част също се записва като цяла част във всяка ПБС, то двете части се преобразуват поотделно.

Пример 37

Търсим $-517,325_{(10)} = ?_{(4)}$:

Търсим $-517,325_{(10)}=?_{(2)}$:

$$\frac{517:2}{258 \mid 1} \text{ M} \frac{2.0,325}{0 \mid 65} \Rightarrow -517,325_{(10)} = -1000000101,010(1001)_{(2)}.$$

$$\frac{517:2}{258 \mid 1} \quad 0 \mid 65$$

$$\frac{64 \mid 1}{32 \mid 0} \quad 0 \mid 65$$

$$\frac{32 \mid 0}{16 \mid 0} \quad 0 \mid 4$$

$$\frac{8 \mid 0}{4 \mid 0} \quad 0 \mid 8$$

$$\frac{4 \mid 0}{100} \quad 0 \mid 1$$

$$\frac{100}{0 \mid 1} \quad 0 \mid 1$$

Пример 39

Търсим $-325,325_{(10)} = ?_{(5)}$:

$$\frac{325:5}{65 \mid 0} \text{ M} \frac{5.0,325}{1 \mid 625} \Rightarrow -325,325_{(10)} = -2300,1(30)_{(5)}.$$

$$\frac{325:5}{13 \mid 0} \text{ M} \frac{5.0,325}{1 \mid 625} \Rightarrow -325,325_{(10)} = -2300,1(30)_{(5)}.$$

```
Търсим -75143,45_{(10)}:
```

$$\begin{array}{c|c}
2.0,45 \\
\hline
0 & 9 \\
1 & 8 \\
1 & 6 \\
1 & 2 \\
0 & 4 \\
0 & 8 \\
\vdots & \vdots \\
\Rightarrow -75143,45_{(10)} = -10010010110000111,01(1100)_{(2)}
\end{array}$$

7. Преобразуване на краен непериодичен запис от и към основи степени на едно и също s

Формула 10, $0 \le \overline{a_{k-1} \dots a_0}_{(s)} < s^k$, показва, че всяка цифра при основа s^k на ПБС може да се запише във вида $\overline{a_{k-1} \dots a_0}_{(s)}$ и всяко число от вида $\overline{a_{k-1} \dots a_0}_{(s)}$ е стойност на цифра при основа s^k на ПБС. От тази зависимост следват прости начини за преобразуване на запис от ПБС с основа s^k и обратно.

Универсалният алгоритъм за преобразуване от основа S към основа S^k на ПБС е следния: Записът при основа S^k се разделя на групи по K^k цифри, започвайки от дробната запетая наляво и надясно, и всяка от получените групи се замества с точно тази цифра при основа S^k , чиято стойност е записана с групата цифри в S^k -ична ПБС. При това, за да се получат пълни групи в началото и в края на записа може да се дописват нули (те са незначещи, т. е. записаното число е едно и също и с тях, и без тях).

Пример 41

Търсим 11110110000110110,001001111 $_{(2)} = ?_{(4)} = ?_{(8)} = ?_{(16)}$:

 $\Rightarrow 11110110000110110,001001111_{(2)} = 132300312,02132_{(4)}$

 $\Rightarrow 11110110000110110,001001111_{(2)} = 366066,117_{(8)}$

 $\Rightarrow 11110110000110110,001001111_{(2)} = 1ec36,278_{(16)} \ \ (e_{(16)} = 15_{(10)} \ \ \textit{if} \ \ c_{(16)} = 12_{(10)}).$

Пример 42

Търсим $-21102012010,012210212_{(3)} = ?_{(9)}$:

 \Rightarrow -21102012010,012210212₍₃₎=-242163,18376₍₉₎

Съвсем аналогично, универсалният алгоритъм за преобразуване от основа S^k към основа S на ПБС е следния: Всяка цифра при основа S^k се замества k -цифрения S -ичен запис на нейната стойност. Тук е важно да бъде заместена с точно k на брой S -ични, иначе получавания запис би бил за друго число, различно от даденото.

Търсим a10 feac 97 d8, $240 e_{(16)} = ?_{(2)}$:

(Нека напомним, че $\,a\!=\!10_{(10)}$, $\,b\!=\!11_{(10)}$, $\,c\!=\!12_{(10)}$, $\,d\!=\!13_{(10)}$, $\,e\!=\!14_{(10)}$ и $\,f\!=\!15_{(10)}$.)

$$\Rightarrow a10 feac 97 d8,240 e_{(16)} =$$

В такива случаи е особено очевидно, колко много изчисления се спестяват, когато се избягва преобразуването от основа 16 на ПБС към основа 10 и от основа 10 към основа 2.

Пример 44

Търсим 718043,65226₍₉₎=?₍₃₎:

$$\Rightarrow$$
 -718043,65226₍₉₎=-210122001110,201202022₍₃₎

Универсалният алгоритъм за преобразуване от основа S^n към основа S^k на ПБС предвижда преминаване през запис в ПБС с основа S .

Пример 45

Търсим b4fa,7 $c08_{(16)}$ =?₍₈₎:

$$\begin{bmatrix} b & 4 & f & a \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 3 & 2 & 3 & 7 & 2 \end{bmatrix}, \begin{bmatrix} 7 & c & 0 & 8_{(16)} \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 3 & 7 & 2 & 3 & 7 & 0 \end{bmatrix}_{(2)}$$

$$\Rightarrow b4fa,7c08_{(16)}=132372,37004_{(8)}$$

Строгото математическо доказателство на описаните алгоритми съществено се опира на факта, че всеки две позиции в записа в ПБС с основа s съответствуват на непресичащи се редици от по s позиции в записа в ПБС с основа s, а именно:

Нека

$$\overline{x}_{(s^k)} = \overline{x_{k-1}...x_0}_{(s)} = x_{k-1}.s^{k-1} + ... + x_0.s^0$$
 и

$$\overline{y}_{(s^k)} = \overline{y_{k-1}...y_0}_{(s)} = y_{k-1}.s^{k-1} + ... + y_0.s^0$$

Тогава за всяко число от вида $\overline{\ldots x \ldots y \ldots}$ $\binom{k}{s^k}$ е вярно:

$$\overline{\ldots x \ldots y \ldots} \Big|_{(s^{k})} = \ldots + x \cdot (s^{k})^{u} + \ldots + y \cdot (s^{k})^{w} + \ldots$$

$$= \ldots + (x_{k-1} \cdot s^{k-1} + \ldots + x_{0} \cdot s^{0}) \cdot (s^{k})^{u} + \ldots + (y_{k-1} \cdot s^{k-1} + \ldots + y_{0} \cdot s^{0}) \cdot (s^{k})^{w} + \ldots$$

$$= \ldots + (x_{k-1} \cdot s^{k-1} + \ldots + x_{0} \cdot s^{0}) \cdot (s^{k \cdot u}) + \ldots + (y_{k-1} \cdot s^{k-1} + \ldots + y_{0} \cdot s^{0}) \cdot (s^{k \cdot w}) + \ldots$$

$$= \ldots + (x_{k-1} \cdot s^{u \cdot k} + (k-1) + \ldots + x_{0} \cdot s^{u \cdot k} + 0) + \ldots + (y_{k-1} \cdot s^{w \cdot k} + (k-1) + \ldots + y_{0} \cdot s^{w \cdot k} + 0) + \ldots$$

$$= \ldots x_{k-1} \ldots x_{0} \ldots y_{k-1} \ldots y_{0} \ldots (s)$$

Последното преобразование ще бъде коректно, защото от w < u следва, че $w \cdot k + (k-1) < u \cdot k + 0$, а от това пък следва, че няма общо число в интервалите $[w \cdot k + 0; w \cdot k + (k-1)]$ и $[u \cdot k + 0; u \cdot k + (k-1)]$.

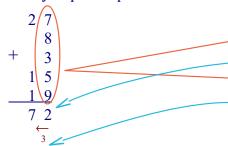
8. Сумиране в ПБС с основа ѕ

Основните аритметични действия се извършват при която и да било основа s на ПБС съвсем аналогично на алгоритмите в 10-ичната ПБС.

При сумиране, точно както в 10-ичната ПБС, събираемите се подравняват по дробната запетая и цифрите на сумата се получават една по една отдясно наляво. Единствената разлика между 10-ична ПБС и s-ична ПБС се проявява при възникването на пренос, но има и точна аналогия, която илюстрират следващите примери.

Пример 46

Сумиране при основа 10 на ПБС:



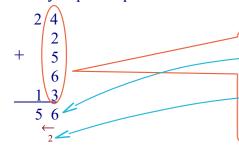
Сумата на цифрите от тази колонка е 32, което е повече от основата 10 на ПБС. Следователно 32 се дели с основата 10.

32=3.10+2. Затова **2**, т. е. остатъкът от делението на 32 с основата на ПБС, е стойността на **цифрата**, която се записва под колонката в крайната сума. Съответно **3**, т. е. цялата част от делението на 32 с основата на ПБС, е **преноса** към следващата наляво колонка от цифри на събираемите.

В този и следващите примери всяка двойка оцветени в червено *стрелка* под сумата и *число под стрелката* показват преноса, който възниква от колонката над началото на стрелката към следващата отляво колонка.

Пример 47

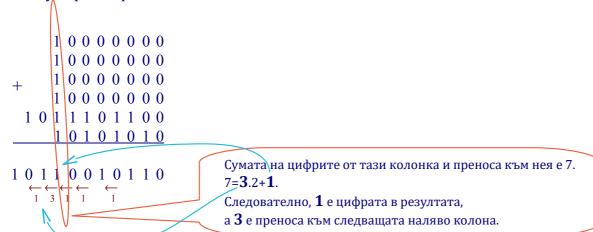
Сумиране при основа 7 на ПБС:



Сумата на цифрите от тази колонка е 20, което е повече от основата 7 на ПБС. Следователно 20 се дели с основата 7.

20=2.7+6. Затова **6**, т. е. остатъкът от делението на 20 с основата на ПБС, е стойността на **цифрата**, която се записва под колонката в крайната сума. Съответно **2**, т. е. цялата част от делението на 20 с основата на ПБС, е **преноса** към следващата наляво колонка от цифри на събираемите.

Сумиране при основа 2 на ПБС:

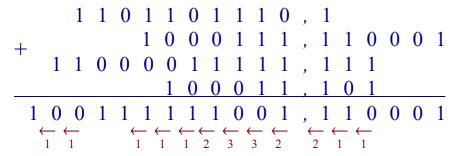


Пример 49

Сумиране при основа 3 на ПБС:

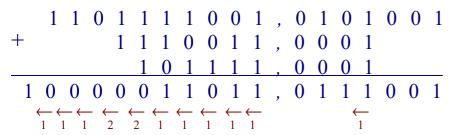
Пример 50

Сумиране при основа 2 на ПБС:

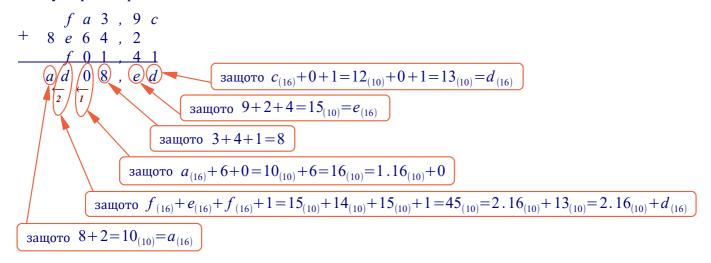


Пример 51

Сумиране при основа 2 на ПБС:



Сумиране при основа 16 на ПБС:



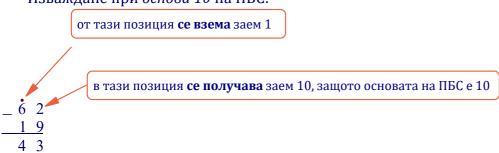
9. Изваждане в ПБС с основа ѕ

При изваждане, точно както в 10-ичната ПБС, умаляемото и умалителя се подравняват по дробната запетая и цифрите на разликата се получават една по една отдясно наляво. Единственото различие между 10-ична ПБС и *S* -ична ПБС се появява, когато е необходим заем, но при всички ПБС има и точна аналогия, която илюстрират следващите примери.

Главното правило при изваждането е, че единицата, вземана в заем, преминавайки към съседната позиция надясно, се умножава по основата на ПБС, т. е. във всяка ПБС се взема заем единица, но при основа S на ПБС винаги се получава заем S .

Пример 53

Изваждане при основа 10 на ПБС:



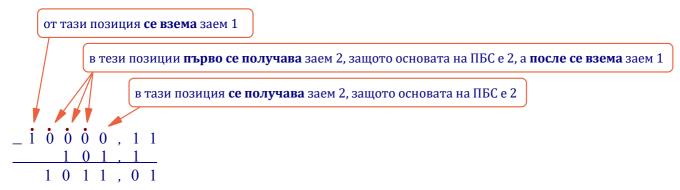
В този и следващите примери червената точка над цифра показва, че от тази цифра се взема заем една единица и към цифрата отдясно на тази с точката се прибавя като получаван заем толкова, колкото е основата на ПБС.

Изваждане при основа 4 на ПБС:



Пример 55

Изваждане при основа 2 на ПБС:



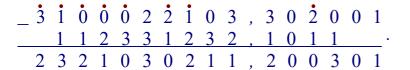
Пример 56

Изваждане при основа 2 на ПБС:

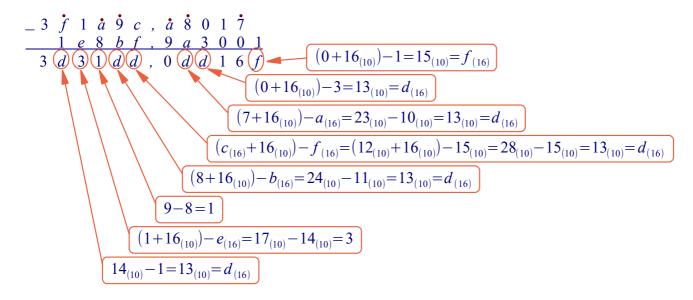


Пример 57

Изваждане при основа 4 на ПБС:



Изваждане при основа 16 на ПБС:



10. Умножаване в ПБС с основа ѕ

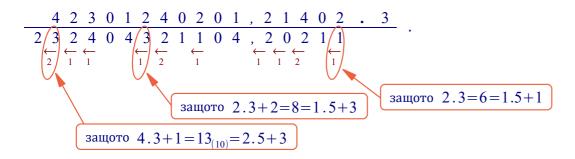
Точно както при 10-ичната ПБС, умножението на числа се свежда до умножение на число с цифра и събиране.

Пример 59

Умножение с цифра при основа 10 на ПБС:

Пример 60

Умножение с цифра при основа 5 на ПБС:



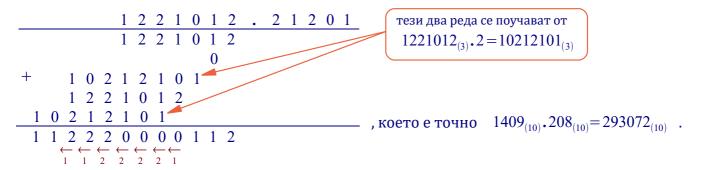
В 2-ична ПБС произведението на число с цифра или е самото число, или е нула.

Пример 61

Умножение с цифра при основа 2 на ПБС:

$$\frac{1011011110,110 \cdot 1}{1011011110,110}$$
 и $\frac{1011011110,110 \cdot 0}{0}$.

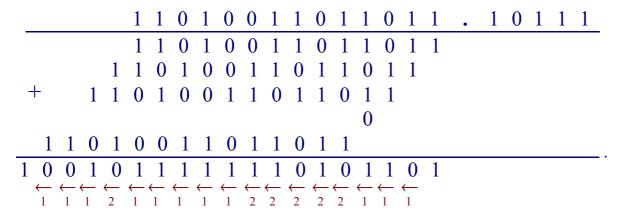
Умножение на числа при основа 3 на ПБС:



В 2-ична ПБС произведението на числа се свежда до преписване с подравняване и сумиране.

Пример 63

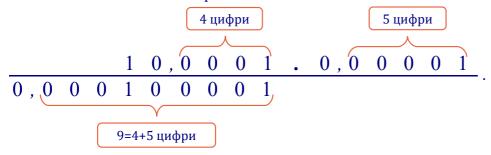
Умножение на числа при основа 2 на ПБС:



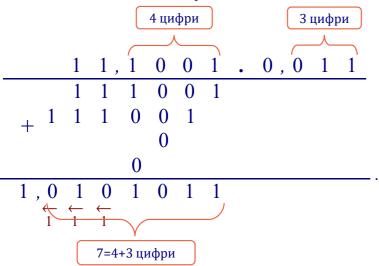
При умножение на числа с дробни части, точно както в 10-ична ПБС, с цифрите се работи както при цели числа, а дробната запетая се поставя в произведението така, че след нея да има такъв брой цифри, колкото е сумата от броевете цифри след дробните запетаи в множителите.

Пример 64

Умножение на числа при основа 2 на ПБС:



Умножение на числа при основа 2 на ПБС:

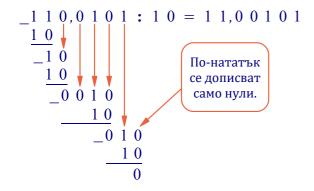


11. Делене в ПБС с основа ѕ

Точно както при 10-ичната ПБС, деленето на числа се свежда до сравняване, умножение на число с цифра и изваждане.

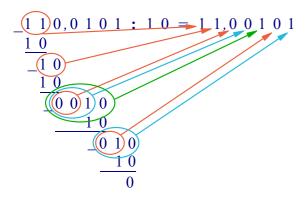
Пример 66

Делене при основа 2 на ПБС:



Тук стрелките показват участието на цифрите на делимото.

Ето същия пример, като овалите и стрелките показват точно от кое число се получава всяка цифра на частното:



Точно както при 10-ичната ПБС, при делене на крайни записи може да се получи период.

Пример 67

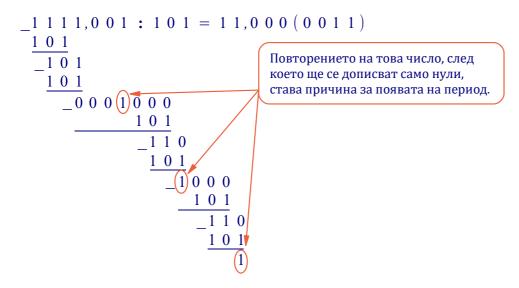
Делене при основа 2 на ПБС:

```
-1\ 0\ 0\ 1\ 1: \ 1\ 1 = 1\ 1\ 0, (\ 0\ 1\ )
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 0\ 0
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-1\ 1
-
```

Това е точно 19:3=6,(3).

Пример 68

Делене при основа 2 на ПБС:



Това е точно 15,125:5=3,025.

12. Преобразуване от една в друга ПБС на записи, съдържащи период

При наличие на период в дадения запис, получаването на записа на същото число при друга основа на ПБС може да стане по поне два относително лесни начина:

Първо, даденият запис, съдържащ период, който трябва да се преобразува в S-ичен вид, може да се преобразува в рационален израз, после да се заместят всички числа от израза с техните S-ични записи и да се извършат действията в S-ична ПБС. Този подход е особено удобен за преминаване от друга към 10-ична ПБС, защото изчисленията ще се провеждат в привичната 10-ична ПБС, но може да се прилага и в посока от всяка към всяка основа.

Второ, към дадения запис, съдържащ период, който трябва да се преобразува в S-ичен вид, може да се приложат описаните по-горе алгоритми за преобразуване на записа, като изчисленията с период се опират на съответните математически правила (или на подходящи разсъждения). Този подход е съществено по-удобен при преминаване от основа 10 към друга основа, защото за десетичните изчисления по-лесно се възприема аритметиката с периодични дроби, но по аналогия може да се подходи и за преминаване от друга основа към основа 10.

Следващите три примера илюстрират първия подход, т. е. — чрез получаване на рационален израз, в който участвуват само *S* -ични числа, и извършване на изчисленията в *S* -ична ПБС.

Пример 69

Търсим 110,
$$(01)_{(2)} = ?_{(10)}$$
:

От формула 27, т. e. от

$$\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)},$$

се получава
$$110,(01)_{(2)} = \frac{110_{(2)}.(2^2-1)+01_{(2)}}{2^0.(2^2-1)} = \frac{6.3+1}{1.3} = \frac{19_{(10)}}{3} = 6,(3)_{(10)}.$$

Пример 70

Търсим $11,000(0011)_{(2)} = ?_{(10)}$:

От формула *27*, т. е. от

$$\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)},$$

се получава

$$11,000(0011)_{(2)} = \frac{11000_{(2)} \cdot (2^4 - 1) + 0011_{(2)}}{2^3 \cdot (2^4 - 1)} = \frac{24_{(10)} \cdot 15_{(10)} + 3}{8 \cdot 15_{(10)}} = \frac{363_{(10)}}{120_{(10)}} = 3,025_{(10)}.$$

Пример 71

Търсим $0,(6)_{(10)} = ?_{(7)}$:

От формула 27, т. е. от

$$\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)},$$

се получава
$$0,(6)_{(10)} = \frac{0_{(10)}.(10^1-1)+6_{(10)}}{10^0.(10^1-1)} = \frac{6_{(10)}}{9_{(10)}} = \frac{6_{(7)}}{12_{(7)}}.$$

От деленето в 7-ична ПБС се получава
$$-\frac{6,0}{5}\frac{1}{6}:12_{(7)}=0,(4)_{(7)}$$
 .

$$\Rightarrow 0,(6)_{(10)} = 0,(4)_{(7)}.$$

Този резултат може да се провери чрез обратното преобразование:

От формула 27, т. е. от

$$\frac{\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}}{s^{j}.(s^{k}-1)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)},$$

се получава
$$0, (4)_{(7)} = \frac{0_{(7)}.(7^1-1)+4_{(7)}}{7^0.(7^1-1)} = \frac{4_{(7)}}{6_{(7)}} = \frac{4_{(10)}}{6_{(10)}} = \frac{2_{(10)}}{3_{(10)}} = 0, (6)_{(10)}.$$

Примери 73 и 74 илюстрират втория подход, т. е. — чрез аритметика с периодични дроби.

В тези примери съществено се използва правилото, че при умножаване на период възникващият пренос наляво от периода се прибавя и отдясно към самия период.

Пример 72

$$3.0.0.0 (8.5)_{(10)} = 0.0.2 (5.7)_{(10)}$$

тези два преноса са предизвикани от 3.8+1=25=2.10+5

Също така се използва, че:

(33)
$$\overline{a_n...a_0, f_1...f_j(c_1...c_kc_{k+1}...c_{k+h})}_{(s)} = \overline{a_n...a_0, f_1...f_jc_1...c_k(c_{k+1}...c_{k+h}c_1...c_k)}_{(s)}.$$

Пример 73

Търсим $0,(4)_{(10)}=?_{(2)}$:

$$\begin{array}{c|c} 2.0, (4) \\ \hline 0 \mid (8) \\ 1 \mid (7) \\ 1 \mid (5) \\ 1 \mid (1) \\ 0 \mid (2) \\ 0 \mid (4) \end{array} \Rightarrow 0, (4)_{(10)} = 0, (011100)_{(2)}.$$

Проверка чрез първия подход:

От формула 27, т. е. от

$$\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}(s)}}{s^{j}.(s^{k}-1)},$$

се получава
$$0,(011100)_{(2)} = \frac{0.(2^6-1)+011100_{(2)}}{2^0.(2^6-1)} = \frac{28_{(10)}}{63_{(10)}} = 0,(4)_{(10)}.$$

Пример 74

Търсим
$$-11201,002(30)_{(4)} = ?_{(10)}$$
:

$$(((1.4+1).4+2).4+0).4+1=353_{(10)}.$$

$$10_{(10)} = 22_{(4)}; \\ \frac{0,002 (30)_{(4)}.22_{(4)}}{+11 (21)}; \\ \frac{112 (12)}{0,123 (33)_{(4)}}; \\ \frac{313 (33)}{10,113 (33)_{(4)}}; \\ \frac{233 (33)}{3,233 (33)_{(4)}}$$

(Тук се използва формула 33.)

$$\Rightarrow \begin{array}{c|c} 22_{(4)}.0,002(30)_{(4)} \\ \hline 0_{(4)} & 123(33)_{(4)} \\ 10_{(4)} & 113(33)_{(4)} \\ 233(33)_{(4)} & 233(33)_{(4)} \\ 10_{(4)} & 133(33)_{(4)} \\ 10_{(4)} & 333(33)_{(4)} \\ 21_{(4)} & 333(33)_{(4)} \end{array}$$

$$10_{(4)} = 4_{(10)}$$
; $13_{(4)} = 7_{(10)}$; $21_{(4)} = 9_{(10)}$;

$$\Rightarrow$$
 -11201,002(30)₍₄₎=-353,04374(9)₍₁₀₎=-353,04375₍₁₀₎.

Проверка чрез първия подход:

От формула 27, т. e. от

$$\overline{a_{n}...a_{0}, f_{1}...f_{j}(c_{1}...c_{k})}_{(s)} = \frac{\overline{a_{n}...a_{0}f_{1}...f_{j}}_{(s)}.(s^{k}-1) + \overline{c_{1}...c_{k}}_{(s)}}{s^{j}.(s^{k}-1)},$$

се получава

$$\begin{split} &-11201,002(30)_{(4)} = -\frac{11201002_{(4)}.(4^2-1)+30_{(4)}}{4^3.(4^2-1)} = \\ &= -\frac{22594_{(10)}.15_{(10)}+12_{(10)}}{64_{(10)}.15_{(10)}} = -\frac{338922_{(10)}}{960_{(10)}} = -353,04375_{(10)} \,. \end{split}$$

При преобразуване от основа S^k към основа S всяка цифра от периода се заменя, както и извън периода. Понякога след това е възможно да се намалят цифрите пред периода.

Пример 75

Търсим
$$6,354(172)_{(8)} = ?_{(2)}$$
:

$$\Rightarrow$$
 6,354(172)₍₈₎=110,011101100(001111010)₍₂₎=110,01110110(000111101)₍₂₎

При преобразуване от основа s към основа s първо трябва числото да се представи по еквивалентен начин така, че периодът да се разделя точно на цяло число групи от по s цифри.

Пример 76

Търсим $0,102(221)_{(3)} = ?_{(9)}$:

$$0,102(221)_{(3)} = 0,1022(212)_{(3)} = 0,1022(212212)_{(3)}$$

$$\Rightarrow 0.102(221)_{(3)} = 0.38(785)_{(9)}$$