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 C# 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

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 s........ 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 $\dfrac{+9{,}6(7)_{(16)}}{-38{,}2(54)_{(9)}}$.

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

където:

- може да има още знак плюс или минус, например $+34{,}01_{(5)}$ и $-0{,}f\ 9_{(16)}$;
- 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) $\overline{a_n \ldots a_1 a_0}_{\,(s)} \overset{\text{def}}{=} a_n . s^n + \ldots + a_1 . s^1 + a_0 . s^0$ за цялата част и

(3) $\overline{0,c_1c_2\ldots c_k\ldots}_{(s)} \stackrel{\text{def}}{=} \frac{c_1}{s^1}+\frac{c_2}{s^2}+\ldots+\frac{c_k}{s^k}+\ldots$ за дробната част .

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

(4) $\overline{a_n\ldots a_1a_0}_{(s)}=\Big(\Big(\ldots\big(a_n.s+a_{n-1}\big).s+\ldots+a_2\Big).s+a_1\Big).s+a_0$ за цялата част и

$$\cfrac{c_{k-2}+\cfrac{c_{k-1}+\cfrac{c_k+\ldots}{s}}{s}}{}$$

(5) $$\overline{0,c_1c_2\ldots c_{k-2}c_{k-1}c_k\ldots}_{(s)}=\cfrac{c_1+\cfrac{c_2+\cfrac{\ldots}{s}}{s}}{s}$$

за дробната част.

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

Пример 1

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

Пример 2

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

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.

$4\mathrm{f}\,{,}8c(2\mathrm{a})_{(16)}$ 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) $s^k = 1\underbrace{0\ldots0}_{k}{}_{(s)}$.

(7) $\dfrac{1}{s^k} = 0,\underbrace{0\ldots0}_{k-1}1_{(s)}$.

(8) $\overline{a_{k-1}\ldots a_0}_{(s)} < 1\underbrace{0\ldots0}_{k}{}_{(s)} \leq \overline{1d_{k-1}\ldots d_0}_{(s)}$,

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

От свойствата (8) и (6) следва, че в k двоични разряда може да се запишат естествените числа от интервала $\left[0;2^k-1\right]$, 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 $\left[0;2^k-1\right]$.

(9) $\overline{0,\underbrace{0\ldots0}_{k}c\ldots}_{(s)} < 0,\underbrace{0\ldots0}_{k-1}1_{(s)} \leq \overline{0,c_1\ldots c_{k-1}1c_{k+1}\ldots}_{(s)}$,

каквито и да са цифрите $c,c_0,\ldots,c_{k-1},c_{k+1}$.

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

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

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

(12) $(n>k \ \wedge \ c_{k+1} \neq 0\ ) \Rightarrow \overline{0,\underbrace{0\ldots0}_{k}c_{k+1}\ldots}_{(s)} > \overline{0,\underbrace{0\ldots0}_{n}d_{n+1}\ldots}_{(s)}$

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

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) $$X=\overline{a_n \ldots a_k d_x \ldots , \ldots}_{(s)} \wedge Y=\overline{a_n \ldots a_k d_y \ldots , \ldots}_{(s)} \wedge d_x > d_y \Rightarrow X > Y$$

(когато има разлика в цялата част) и

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

(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 $\overline{a_{k-1} \ldots a_0}_{(s)}$ и всяко число от вида $\overline{a_{k-1} \ldots a_0}_{(s)}$ принадлежи на интервала $[0;s^k-1]$, т. е.

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

Броят на целите неотрицателни k-цифрени s-ични числа е точно s^k. Т. е.

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

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

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

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

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

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

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

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

(23) $\underbrace{\xi\ldots\xi}_{k}{}_{(s)}=s^k-1$ (следствие от това, че $\underbrace{\xi\ldots\xi}_{k}{}_{(s)}$ е най-голямото k -цифрено s -ично число и че $1\underbrace{0\ldots0}_{k}{}_{(s)}=s^k$ е най-малкото $(k+1)$ -цифрено s -ично число) и

(24) $0,\underbrace{\xi\ldots\xi}_{k}{}_{(s)}=\dfrac{s^k-1}{s^k}$ (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) $\overline{0,(c_1\ldots c_k)}_{(s)}=\dfrac{\overline{c_1\ldots c_k}_{(s)}}{s^k-1}$.

Доказателство

$$\overline{0,(c_1\ldots c_k)}_{(s)}=\overline{0,c_1\ldots c_k}_{(s)}+\overline{0,\underbrace{0\ldots0}_{k}(c_1\ldots c_k)}_{(s)}=\overline{0,c_1\ldots c_k}_{(s)}+\frac{\overline{0,(c_1\ldots c_k)}_{(s)}}{s^k}.$$

$$\Rightarrow\ \overline{0,(c_1\ldots c_k)}_{(s)}.\left(1-\frac{1}{s^k}\right)=\overline{0,c_1\ldots c_k}_{(s)}.$$

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

Пример 3

$$0,(613)_{(9)}=\frac{613_{(9)}}{9^3-1}=\frac{6.9^2+1.9+3}{9^3-1}=\frac{486+9+3}{729-1}=\frac{498}{728}=\frac{249}{364}$$

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

Доказателство

$$\overline{0,f_1\dots f_j(c_1\dots c_k)}_{(s)}=\frac{\overline{f_1\dots f_j}_{(s)}}{s^j}+\frac{\overline{0,(c_1\dots c_k)}_{(s)}}{s^j}$$

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

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

Пример 7

$$1{,}0(12)_{(3)}=\frac{10_{(3)}.(3^2-1)+12_{(3)}}{3.(3^2-1)}=\frac{3.8+5}{3.8}=\frac{29}{24}=1\frac{5}{24}$$

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 към Неперовата единица $e = 2{,}7\ldots$. 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 $R = \dfrac{C(3;2.3.k)}{C(2;2.3.k)} = \dfrac{3^{2.k}}{2^{3.k}} = \left(\dfrac{9}{8}\right)^k$.

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 x86 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 C#. 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) $$\underbrace{\overline{d\ \dots}}_{n\ цифри}{}_{(s)}=d.s^{n-1}+\dots.$$

Пример 10

Търсим $1001_{(5)}=?_{(10)}$:

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

или

$$1001_{(5)}=((1.5+0).5+0).5+1=126_{(10)}.$$

Пример 11

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

$$211202_{(3)}=((((2.3+1).3+1).3+2).3+0).3+2$$

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

Пример 13

Търсим $f9ac_{(16)}=?_{(10)}$:

$$f9ac_{(16)}=15.16^3+9.16^2+10.16^1+12.16^0$$

$$=15.4096+9.256+10.16+12.1=61440+2304+160+12=63916_{(10)}$$

или

$$f9ac_{(3)}=((15.16+9).16+10).16+12=(249.16+10).16+12=3994.16+12=63916.$$

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

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

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

Пример 14

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

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

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

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

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

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

Пример 18

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

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

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

$$-14323,001_{(5)}=-\left(\left(\left(\left(1.5+4\right).5+3\right).5+2\right).5+3+\frac{0+\frac{0+\frac{1}{5}}{5}}{5}\right)=-\left(\left(\left(9.5+3\right).5+2\right).5+3+\frac{0+\frac{1}{25}}{5}\right)$$

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

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

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

Пример 22

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.

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

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.

Пример 23

Търсим $29_{(10)}=?_{(2)}$:

$$\begin{array}{rcr} 29 & = & 14.2+1\\ 14 & = & 7.2+0\\ 7 & = & 3.2+1\\ 3 & = & 1.2+1\\ 1 & = & 0.2+1 \end{array}$$

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

$$\Rightarrow\quad 29_{(10)}=11101_{(2)}$$

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

$$\begin{array}{r|l} \multicolumn{2}{c}{29:2}\\ \hline 14 & 1\\ 7 & 0\\ 3 & 1\\ 1 & 1\\ 0 & 1 \end{array}$$

цяла част от частното на 29 и 2

остатък от делението на 28 с 2

цяла част от частното на 14 и 2

остатък от делението на 14 с 2

$$\Rightarrow\quad 29_{(10)}=11101_{(2)}$$

Пример 24

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

$$\begin{array}{r|l} \multicolumn{2}{c}{29:3} \\ \hline 9 & 2 \\ 3 & 0 \\ 1 & 0 \\ 0 & 1 \end{array} \Rightarrow 29_{(10)}=1002_{(3)}; \quad \begin{array}{r|l} \multicolumn{2}{c}{29:5} \\ \hline 5 & 4 \\ 1 & 0 \\ 0 & 1 \end{array} \Rightarrow 29_{(10)}=104_{(5)}; \quad \begin{array}{r|l} \multicolumn{2}{c}{29:7} \\ \hline 4 & 1 \\ 0 & 4 \end{array} \Rightarrow 29_{(10)}=41_{(7)}.$$

Пример 25

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

$$\begin{array}{r|l} \multicolumn{2}{c}{735:2} \\ \hline 367 & 1 \\ 183 & 1 \\ 91 & 1 \\ 45 & 1 \\ 22 & 1 \\ 11 & 0 \\ 5 & 1 \\ 2 & 1 \\ 1 & 0 \\ 0 & 1 \end{array} \Rightarrow -735_{(10)}=-1011011111_{(2)}; \quad \begin{array}{r|l} \multicolumn{2}{c}{735:3} \\ \hline 245 & 0 \\ 81 & 2 \\ 27 & 0 \\ 9 & 0 \\ 3 & 0 \\ 1 & 0 \\ 0 & 1 \end{array} \Rightarrow -735_{(10)}=-1000020_{(3)};$$

$$\begin{array}{r|l} \multicolumn{2}{c}{735:7} \\ \hline 105 & 0 \\ 15 & 0 \\ 2 & 1 \\ 0 & 2 \end{array} \Rightarrow -735_{(10)}=-2100_{(7)}.$$

Пример 26

Търсим $347_{(10)}=?_{(8)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{347:8} \\ \hline 43 & 3 \\ 5 & 3 \\ 0 & 5 \end{array} \Rightarrow 347_{(10)}=533_{(8)}.$$

Пример 27

Търсим $347_{(10)} = ?_{(16)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{347:16} \\ \hline 21 & 11 \\ 1 & 5 \\ 0 & 1 \end{array} \quad \Rightarrow \quad 347_{(10)} = 15b_{(16)}$$ (16-ичната цифра b има стойност 11).

Пример 28

Търсим $-249_{(10)} = ?_{(16)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{249:16} \\ \hline 15 & 9 \\ 0 & 15 \end{array} \quad \Rightarrow \quad -249_{(10)} = -f\,9_{(16)}$$ (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) $$\overline{0,c_1c_2\ldots c_n\ldots}_{(s)}.s = \overline{c_1,c_2\ldots c_n\ldots}_{(s)}.$$

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.

$$\begin{aligned} 2.0{,}3125 &= 0{,}625 \\ 2.0{,}625 &= 1{,}25 \\ 2.0{,}25 &= 0{,}5 \\ 2.0{,}5 &= 1{,}0 \end{aligned} \qquad \Rightarrow \quad 0{,}3125_{(10)} = 0{,}0101_{(2)}$$

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

цяла част от произведението на 0,3125 и 2

дробна част от произведението на 0,3125 и 2

$$\begin{array}{r|l} \multicolumn{2}{c}{2.0{,}3125} \\ \hline 0 & 625 \\ 1 & 25 \\ 0 & 5 \\ 1 & 0 \end{array}$$

цяла част от произведението на 0,25 и 2

дробна част от произведението на 0,25 и 2

$$\Rightarrow \quad 0{,}3125_{(10)} = 0{,}0101_{(2)}$$

Пример 30

Търсим $0{,}1875_{(10)}=?_{(2)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{2\,.\,0{,}1875} \\ \hline 0 & 375 \\ 0 & 75 \\ 1 & 5 \\ 1 & 0 \end{array} \quad \Rightarrow \quad 0{,}1875_{(10)}=0{,}0011_{(2)}\,.$$

Пример 31

Търсим $0{,}1376_{(10)}=?_{(5)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{5\,.\,0{,}1376} \\ \hline 0 & 688 \\ 3 & 44 \\ 2 & 2 \\ 1 & 0 \end{array} \quad \Rightarrow \quad 0{,}1376_{(10)}=0{,}0321_{(5)}\,.$$

Пример 32

Търсим $0{,}0390625_{(10)}=?_{(16)}$:

$$\begin{array}{r|l} \multicolumn{2}{c}{16\,.\,0{,}0390625} \\ \hline 0 & 625 \\ 10 & 0 \end{array} \quad \Rightarrow \quad 0{,}0390625_{(10)}=0{,}0\,a_{(16)}$$ (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 $0{,}35_{(10)}=?_{(2)}$:

една и съща дробна част, която води до една и съща последователност от цифри

$$\begin{array}{r|l} \multicolumn{2}{c}{2\,.\,0{,}35} \\ \hline 0 & 7 \\ 1 & 4 \\ 0 & 8 \\ 1 & 6 \\ 1 & 2 \\ 0 & 4 \end{array} \qquad \Rightarrow \quad 0{,}35_{(10)}=0{,}01\,(0110)_{(2)}$$

повтаряща се последователност от цифри

Пример 34

Търсим $0{,}1_{(10)}=?_{(3)}$:

$$\begin{array}{r|l} \multicolumn{2}{l}{3\,.\,0{,}1} \\ \hline 0 & 3 \\ 0 & 9 \\ 2 & 7 \\ 2 & 1 \end{array} \quad \Rightarrow \quad 0{,}1_{(10)}=0,(0022)_{(3)}\,.$$

Пример 35

Търсим $-0{,}14_{(10)}=?_{(5)}$:

$$\begin{array}{r|l} \multicolumn{2}{l}{5\,.\,0{,}14} \\ \hline 0 & 7 \\ 3 & 5 \\ 2 & 5 \end{array} \quad \Rightarrow \quad -0{,}14_{(10)}=-0{,}03(2)_{(5)}\,.$$

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 $-0{,}8_{(10)}=?_{(7)}$:

$$\begin{array}{r|l} \multicolumn{2}{l}{7\,.\,0{,}8} \\ \hline 5 & 6 \\ 4 & 2 \\ 1 & 4 \\ 2 & 8 \end{array} \quad \Rightarrow \quad -0{,}8_{(10)}=-0,(5412)_{(7)}\,.$$

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 $-517{,}325_{(10)}=?_{(4)}$:

$$\begin{array}{r|l} \multicolumn{2}{l}{517:4} \\ \hline 129 & 1 \\ 32 & 1 \\ 8 & 0 \\ 2 & 0 \\ 0 & 2 \end{array} \quad \text{и} \quad \begin{array}{r|l} \multicolumn{2}{l}{4\,.\,0{,}325} \\ \hline 1 & 3 \\ 1 & 2 \\ 0 & 8 \\ 3 & 2 \end{array} \quad \Rightarrow \quad -517{,}325_{(10)}=-20011{,}11(03)_{(4)}\,.$$

Пример 38

Търсим $-517{,}325_{(10)} = ?_{(2)}$:

$$
\begin{array}{r|l}
\multicolumn{2}{c}{517:2} \\
\hline
258 & 1 \\
129 & 0 \\
64 & 1 \\
32 & 0 \\
16 & 0 \\
8 & 0 \\
4 & 0 \\
2 & 0 \\
1 & 0 \\
0 & 1
\end{array}
\quad \text{и} \quad
\begin{array}{r|l}
\multicolumn{2}{c}{2\,.\,0{,}325} \\
\hline
0 & 65 \\
1 & 3 \\
0 & 6 \\
1 & 2 \\
0 & 4 \\
0 & 8 \\
1 & 6
\end{array}
\quad \Rightarrow \quad -517{,}325_{(10)} = -1000000101{,}010(1001)_{(2)} \ .
$$

Пример 39

Търсим $-325{,}325_{(10)} = ?_{(5)}$:

$$
\begin{array}{r|l}
\multicolumn{2}{c}{325:5} \\
\hline
65 & 0 \\
13 & 0 \\
2 & 3 \\
0 & 2
\end{array}
\quad \text{и} \quad
\begin{array}{r|l}
\multicolumn{2}{c}{5\,.\,0{,}325} \\
\hline
1 & 625 \\
3 & 125 \\
0 & 625
\end{array}
\quad \Rightarrow \quad -325{,}325_{(10)} = -2300{,}1(30)_{(5)} \ .
$$

Пример 40

Търсим $-75143{,}45_{(10)}$:

$$
\begin{array}{r|l}
\multicolumn{2}{l}{\underline{75143:2}} \\
37571 & 1 \\
18785 & 1 \\
9392 & 1 \\
4696 & 0 \\
2348 & 0 \\
1174 & 0 \\
587 & 0 \\
293 & 1 \\
146 & 1 \\
73 & 0 \\
36 & 1 \\
18 & 0 \\
9 & 0 \\
4 & 1 \\
2 & 0 \\
1 & 0 \\
0 & 1
\end{array}
$$

$$\Rightarrow \quad -75143_{(10)} = -10010010110000111_{(2)}$$

$$
\begin{array}{r|l}
\multicolumn{2}{l}{\underline{2\,.\,0{,}45}} \\
0 & 9 \\
1 & 8 \\
1 & 6 \\
1 & 2 \\
0 & 4 \\
0 & 8 \\
\vdots & \vdots
\end{array}
$$

$$\Rightarrow \quad 0{,}45_{(10)} = 0{,}01\,(1100)_{(2)}$$

$$\Rightarrow \quad -75143{,}45_{(10)} = -10010010110000111{,}01\,(1100)_{(2)}$$

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

Формула *10*, $0\le\overline{a_{k-1}\ldots a_0}_{(s)}<s^k$, показва, че всяка цифра при основа s^k на ПБС може да се запише във вида $\overline{a_{k-1}\ldots a_0}_{(s)}$ и всяко число от вида $\overline{a_{k-1}\ldots a_0}_{(s)}$ 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 в 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 се разделя на групи по 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-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 $1111011000011011 0,001001111_{(2)}=?_{(4)}=?_{(8)}=?_{(16)}$:

$$\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c|c|c|l}
0\;\;1 & 1\;\;1 & 1\;\;0 & 1\;\;1 & 0\;\;0 & 0\;\;0 & 1\;\;1 & 0\;\;1 & 1\;\;0 & , & 0\;\;0 & 1\;\;0 & 0\;\;1 & 1\;\;1 & 1\;\;0 \;{}_{(2)} \\
1 & 3 & 2 & 3 & 0 & 0 & 3 & 1 & 2 & , & 0 & 2 & 1 & 3 & 2 \;{}_{(4)}
\end{array}$$

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

$$\begin{array}{c|c|c|c|c|c|c|c|c|l}
0\;\;1\;\;1 & 1\;\;1\;\;0 & 1\;\;1\;\;0 & 0\;\;0\;\;0 & 1\;\;1\;\;0 & 1\;\;1\;\;0 & , & 0\;\;0\;\;1 & 0\;\;0\;\;1 & 1\;\;1\;\;1 \;{}_{(2)} \\
3 & 6 & 6 & 0 & 6 & 6 & , & 1 & 1 & 7 \;{}_{(8)}
\end{array}$$

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

$$\begin{array}{c|c|c|c|c|c|c|c|l}
0\;\;0\;\;0\;\;1 & 1\;\;1\;\;1\;\;0 & 1\;\;1\;\;0\;\;0 & 0\;\;0\;\;1\;\;1 & 0\;\;1\;\;1\;\;0 & , & 0\;\;0\;\;1\;\;0 & 0\;\;1\;\;1\;\;1 & 1\;\;0\;\;0\;\;0 \;{}_{(2)} \\
1 & e & c & 3 & 6 & , & 2 & 7 & 8 \;{}_{(16)}
\end{array}$$

$\Rightarrow\; 11110110000110110,001001111_{(2)}=1\mathrm{ec}36,278_{(16)}$ $\left(e_{(16)}=15_{(10)}\text{ и } c_{(16)}=12_{(10)}\right)$.

Пример 42

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

$$\begin{array}{c|c|c|c|c|c|c|c|c|c|c|l}
0\;\;2 & 1\;\;1 & 0\;\;2 & 0\;\;1 & 2\;\;0 & 1\;\;0 & , & 0\;\;1 & 2\;\;2 & 1\;\;0 & 2\;\;1 & 2\;\;0 \;{}_{(3)} \\
2 & 4 & 2 & 1 & 6 & 3 & , & 1 & 8 & 3 & 7 & 6 \;{}_{(9)}
\end{array}$$

$\Rightarrow\; -21102012010,012210212_{(3)}=-242163,18376_{(9)}$

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.

Пример 43

Търсим $a10feac97d8,240e_{(16)}=?_{(2)}$:

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

a	1	0	f	e	a	c	9	7	d	8	,	2	4	0	e	$_{(16)}$
1010	0001	0000	1111	1110	1010	1100	1001	0111	1101	1000	,	0010	0100	0000	1110	$_{(2)}$

$$\Rightarrow a10feac97d8,240e_{(16)}=$$

$$=10100001000011111110101011001001011111011000,00100100000111_{(2)}$$

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 $718043,65226_{(9)}=?_{(3)}$:

7	1	8	0	4	3	,	6	5	2	2	6	$_{(9)}$
2 1	0 1	2 2	0 0	1 1	1 0	,	2 0	1 2	0 2	0 2	2 0	$_{(3)}$

$$\Rightarrow \quad -718043,65226_{(9)}=-210122001110,201202022_{(3)}.$$

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 $b4fa,7c08_{(16)}=?_{(8)}$:

	b	4	f	a	,	7	c	0	8 $_{(16)}$
0 0	1 0 1 1	0 1 0 0	1 1 1 1	1 0 1 0	,	0 1 1 1	1 1 0 0	0 0 0 0	1 0 0 0 $_{(2)}$

0 0 1	0 1 1	0 1 0	0 1 1	1 1 1	0 1 0	,	0 1 1	1 1 1	0 0 0	0 0 0	1 0 0 $_{(2)}$
1	3	2	3	7	2	,	3	7	0	0	4 $_{(8)}$

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

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

Нека

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

$$\overline{\boldsymbol{y}}_{\left(s^k\right)}=\overline{\boldsymbol{y}_{k-1}\ldots\boldsymbol{y}_0}_{\left(s\right)}=\boldsymbol{y}_{k-1}.\boldsymbol{s}^{k-1}+\ldots+\boldsymbol{y}_0.\boldsymbol{s}^0.$$

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

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

$$=\ldots+\left(x_{k-1}.s^{k-1}+\ldots+x_0.s^0\right).(s^k)^u+\ldots+\left(y_{k-1}.s^{k-1}+\ldots+y_0.s^0\right).(s^k)^w+\ldots$$

$$=\ldots+\left(x_{k-1}.s^{k-1}+\ldots+x_0.s^0\right).\left(s^{k.u}\right)+\ldots+\left(y_{k-1}.s^{k-1}+\ldots+y_0.s^0\right).\left(s^{k.w}\right)+\ldots$$

$$=\ldots+\left(x_{k-1}.s^{u.k+(k-1)}+\ldots+x_0.s^{u.k+0}\right)+\ldots+\left(y_{k-1}.s^{w.k+(k-1)}+\ldots+y_0.s^{w.k+0}\right)+\ldots$$

$$=\overline{\ldots x_{k-1}\ldots x_0\ldots y_{k-1}\ldots y_0\ldots}_{(s)}$$

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

8. Сумиране в ПБС с основа s

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:

$$\begin{array}{cr} & 27\\ & 8\\ + & 3\\ & 15\\ & 19\\ \hline & 72\\ & \underset{3}{\leftarrow}\ \end{array}$$

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:

$$\begin{array}{cr} & 24\\ & 2\\ + & 5\\ & 6\\ & 13\\ \hline & 56\\ & \underset{2}{\leftarrow}\ \end{array}$$

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.

Пример 48

Сумиране при *основа 2* на ПБС:

```
            1  0  0  0  0  0  0  0
            1  0  0  0  0  0  0  0
 +          1  0  0  0  0  0  0  0
            1  0  0  0  0  0  0  0
      1  0  1  1  1  0  1  1  0  0
            1  0  1  0  1  0  1  0
 ---------------------------------
   1  0  1  1  0  0  1  0  1  1  0
        ←  ←  ←  ←     ←
        1  3  1  1     1
```

Сумата на цифрите от тази колонка и преноса към нея е 7.
7=**3**.2+**1**.
Следователно, **1** е цифрата в резултата,
а **3** е преноса към следващата наляво колона.

Пример 49

Сумиране при *основа 3* на ПБС:

```
      2  1  2  1  1  0  1  1  0  1  ,  2
 +          1  0  2  2  1  0  2  2  ,  0  1
         1  2  1  2  1  1  1  2  2  ,  0  1
 ------------------------------------------
   1  0  1  0  1  0  1  1  0  2  2  ,  2  2
     ←  ←  ←  ←  ←  ←  ←  ←  ←  ←
     1  1  2  1  2  1  1  1  1  1
```

Пример 50

Сумиране при *основа 2* на ПБС:

```
         1  1  0  1  1  0  1  1  1  0  ,  1
 +                1  0  0  0  1  1  1  ,  1  1  0  0  0  1
      1  1  0  0  0  0  1  1  1  1  1  ,  1  1  1
                     1  0  0  0  1  1  ,  1  0  1
 ---------------------------------------------------------
   1  0  0  1  1  1  1  1  1  0  0  1  ,  1  1  0  0  0  1
     ←  ←        ←  ←  ←  ←  ←  ←  ←    ←   ←  ←
     1  1        1  1  1  2  3  3  2    2   1  1
```

Пример 51

Сумиране при *основа 2* на ПБС:

```
      1  1  0  1  1  1  1  0  0  1  ,  0  1  0  1  0  0  1
 +             1  1  1  0  0  1  1  ,  0  0  0  1
                  1  0  1  1  1  1  ,  0  0  0  1
 ---------------------------------------------------------
   1  0  0  0  0  0  1  1  0  1  1  ,  0  1  1  1  0  0  1
     ←  ←  ←  ←  ←  ←  ←  ←  ←  ←                 ←
     1  1  1  2  2  1  1  1  1  1                 1
```

Пример 52

Сумиране при *основа 16* на ПБС:

$$
\begin{array}{cccccccc}
 & & f & a & 3 & , & 9 & c \\
+ & 8 & e & 6 & 4 & , & 2 & \\
 & & f & 0 & 1 & , & 4 & 1 \\
\hline
 & a & d & 0 & 8 & , & e & d \\
 & & \overleftarrow{2} & \overleftarrow{1} & & & &
\end{array}
$$

защото $c_{(16)}+0+1=12_{(10)}+0+1=13_{(10)}=d_{(16)}$

защото $9+2+4=15_{(10)}=e_{(16)}$

защото $3+4+1=8$

защото $a_{(16)}+6+0=10_{(10)}+6=16_{(10)}=1.16_{(10)}+0$

защото $f_{(16)}+e_{(16)}+f_{(16)}+1=15_{(10)}+14_{(10)}+15_{(10)}+1=45_{(10)}=2.16_{(10)}+13_{(10)}=2.16_{(10)}+d_{(16)}$

защото $8+2=10_{(10)}=a_{(16)}$

9. Изваждане в ПБС с основа s

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:

от тази позиция **се взема** заем 1

в тази позиция **се получава** заем 10, защото основата на ПБС е 10

$$
\begin{array}{ccc}
 & \dot{6} & 2 \\
- & 1 & 9 \\
\hline
 & 4 & 3
\end{array}
$$

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.

Пример 54

Изваждане при *основа 4* на ПБС:

от тази позиция **се взема** заем 1

в тази позиция **се получава** заем 4, защото основата на ПБС е 4

$$
\begin{array}{r}
\_\ \dot{3}\ 1 \\
1\ 2 \\
\hline
1\ 3
\end{array}
$$

Пример 55

Изваждане при *основа 2* на ПБС:

от тази позиция **се взема** заем 1

в тези позиции **първо се получава** заем 2, защото основата на ПБС е 2, а **после се взема** заем 1

в тази позиция **се получава** заем 2, защото основата на ПБС е 2

$$
\begin{array}{r}
\_\ \dot{1}\ \dot{0}\ \dot{0}\ \dot{0}\ 0\ ,\ 1\ 1 \\
1\ 0\ 1\ ,\ 1\ \\
\hline
1\ 0\ 1\ 1\ ,\ 0\ 1
\end{array}
$$

Пример 56

Изваждане при *основа 2* на ПБС:

$$
\begin{array}{r}
\_\ 1\ \dot{1}\ \dot{1}\ 0\ \dot{1}\ \dot{0}\ \dot{0}\ \dot{1}\ \dot{1}\ \dot{0}\ ,\ 0\ \dot{1}\ 0\ 1\ 0\ 1 \\
1\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ ,\ 1\ 0\ 1\ 1\ \ \\
\hline
1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ ,\ 1\ 0\ 1\ 0\ 0\ 1
\end{array}
.
$$

Пример 57

Изваждане при *основа 4* на ПБС:

$$
\begin{array}{r}
\_\ \dot{3}\ \dot{1}\ \dot{0}\ \dot{0}\ \dot{0}\ 2\ \dot{2}\ \dot{1}\ 0\ 3\ ,\ 3\ 0\ \dot{2}\ 0\ 0\ 1 \\
1\ 1\ 2\ 3\ 3\ 1\ 2\ 3\ 2\ ,\ 1\ 0\ 1\ 1\ \ \\
\hline
2\ 3\ 2\ 1\ 0\ 3\ 0\ 2\ 1\ 1\ ,\ 2\ 0\ 0\ 3\ 0\ 1
\end{array}
.
$$

Пример 58

Изваждане при *основа 16* на ПБС:

$$
\begin{array}{r}
\_\ 3\ \dot{f}\ 1\ \dot{a}\ \dot{9}\ c\ ,\ \dot{a}\ \dot{8}\ 0\ 1\ \dot{7} \\
1\ e\ 8\ b\ f\ ,\ 9\ a\ 3\ 0\ 0\ 1 \\
\hline
3\ d\ 3\ 1\ d\ d\ ,\ 0\ d\ d\ 1\ 6\ f
\end{array}
$$

$(0+16_{(10)})-1=15_{(10)}=f_{(16)}$

$(0+16_{(10)})-3=13_{(10)}=d_{(16)}$

$(7+16_{(10)})-a_{(16)}=23_{(10)}-10_{(10)}=13_{(10)}=d_{(16)}$

$(c_{(16)}+16_{(10)})-f_{(16)}=(12_{(10)}+16_{(10)})-15_{(10)}=28_{(10)}-15_{(10)}=13_{(10)}=d_{(16)}$

$(8+16_{(10)})-b_{(16)}=24_{(10)}-11_{(10)}=13_{(10)}=d_{(16)}$

$14_{(10)}$ $(1+16_{(10)})-e_{(16)}=17_{(10)}-14_{(10)}=3$

10. Умножаване в ПБС с основа s

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:

$$\begin{array}{r} 8\ 5\ 0\ 4\ 9\ 3\ 2\ ,\ 1\ 7\ \cdot\ 6 \\ \hline 5\ 1\ 0\ 2\ 9\ 5\ 9\ 3\ ,\ 0\ 2 \end{array}.$$

$$\underset{5}{\leftarrow}\ \underset{3}{\leftarrow}\quad \underset{2}{\leftarrow}\ \underset{5}{\leftarrow}\ \underset{1}{\leftarrow}\ \underset{1}{\leftarrow}\quad \underset{1}{\leftarrow}\ \underset{4}{\leftarrow}$$

Пример 60

Умножение с цифра при *основа 5* на ПБС:

$$\begin{array}{r} 4\ 2\ 3\ 0\ 1\ 2\ 4\ 0\ 2\ 0\ 1\ ,\ 2\ 1\ 4\ 0\ 2\ \cdot\ 3 \\ \hline 2\ 3\ 2\ 4\ 0\ 4\ 3\ 2\ 1\ 1\ 0\ 4\ ,\ 2\ 0\ 2\ 1\ 1 \end{array}.$$

$$\underset{2}{\leftarrow}\ \underset{1}{\leftarrow}\ \underset{1}{\leftarrow}\quad \underset{1}{\leftarrow}\ \underset{2}{\leftarrow}\quad \underset{1}{\leftarrow}\quad \underset{1}{\leftarrow}\ \underset{1}{\leftarrow}\ \underset{2}{\leftarrow}\quad \underset{1}{\leftarrow}$$

защото $2.3=6=1.5+1$

защото $2.3+2=8=1.5+3$

защото $4.3+1=13_{(10)}=2.5+3$

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:

$$\begin{array}{c} 1011011110{,}110\ \cdot\ 1 \\ \hline 1011011110{,}110 \end{array} \quad \text{и} \quad \begin{array}{c} 1011011110{,}110\ \cdot\ 0 \\ \hline 0 \end{array}.$$

Пример 62

Умножение на числа при *основа 3* на ПБС:

$$\begin{array}{r} 1\ 2\ 2\ 1\ 0\ 1\ 2\ \ \cdot\ \ 2\ 1\ 2\ 0\ 1 \\ \hline 1\ 2\ 2\ 1\ 0\ 1\ 2 \\ 0 \\ +\quad 1\ 0\ 2\ 1\ 2\ 1\ 0\ 1 \\ 1\ 2\ 2\ 1\ 0\ 1\ 2 \\ 1\ 0\ 2\ 1\ 2\ 1\ 0\ 1 \\ \hline 1\ 1\ 2\ 2\ 2\ 0\ 0\ 0\ 0\ 1\ 1\ 2 \end{array}$$

$$\underset{1}{\leftarrow}\ \underset{1}{\leftarrow}\ \underset{2}{\leftarrow}\ \underset{2}{\leftarrow}\ \underset{2}{\leftarrow}\ \underset{2}{\leftarrow}\ \underset{1}{\leftarrow}$$

тези два реда се поучават от $1221012_{(3)}.2=10212101_{(3)}$

, което е точно $1409_{(10)}.208_{(10)}=293072_{(10)}$.

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:

$$
\begin{array}{rr}
 & 11010011011011 \cdot 10111 \\
\hline
 & 11010011011011 \\
 & 11010011011011 \\
+ & 11010011011011 \\
 & 0 \\
 & 11010011011011 \\
\hline
 & 1001011111111010101 \\
 & \small 1\;1\;1\;2\;1\;1\;1\;1\;1\;2\;2\;2\;2\;2\;1\;1\;1
\end{array}.
$$

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:

$$
\begin{array}{r}
10,\overbrace{0001}^{\boxed{\text{4 цифри}}} \cdot 0,\overbrace{00001}^{\boxed{\text{5 цифри}}} \\
\hline
0,\underbrace{000100001}_{\boxed{\text{9=4+5 цифри}}}
\end{array}.
$$

Пример 65

Умножение на числа при *основа 2* на ПБС:

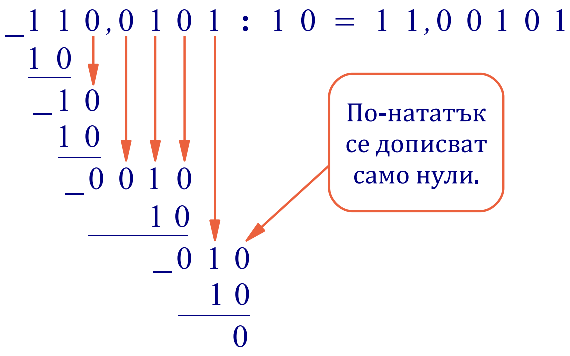
$$
\begin{array}{rr}
 & 11,\overbrace{1001}^{\boxed{\text{4 цифри}}} \cdot 0,\overbrace{011}^{\boxed{\text{3 цифри}}} \\
\hline
 & 111001 \\
+ & 111001 \\
 & 0 \\
 & 0 \\
\hline
 & 1,\underbrace{0101011}_{\boxed{\text{7=4+3 цифри}}} \\
 & \small 1\;1\;1
\end{array}.
$$

11. Делене в ПБС с основа s

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

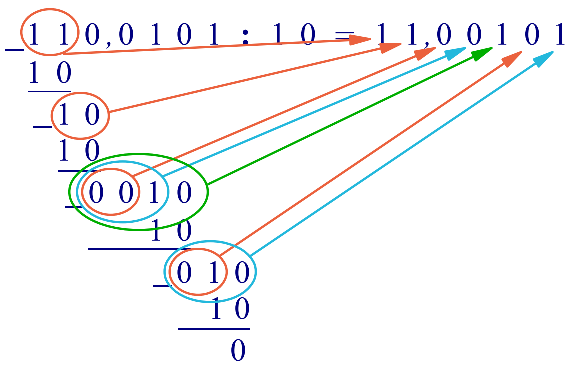
Пример 66

Делене при *основа 2* на ПБС:



Тук стрелките показват участието на цифрите на делимото.

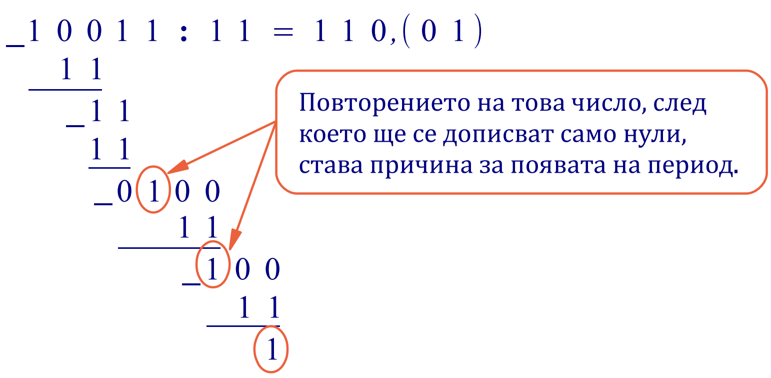
Ето същия пример, като овалите и стрелките показват точно от кое число се получава всяка цифра на частното:



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

Пример 67

Делене при *основа 2* на ПБС:



Това е точно $19:3=6,(3)$.

Пример 68

Делене при *основа 2* на ПБС:

```
_1 1 1 1,0 0 1 : 1 0 1 = 1 1,0 0 0 (0 0 1 1)
 1 0 1
  _1 0 1
   1 0 1
     _0 0 0 1 0 0 0
            1 0 1
           _1 1 0
            1 0 1
             _1 0 0 0
               1 0 1
               _1 1 0
                1 0 1
                    1
```

Повторението на това число, след което ще се дописват само нули, става причина за появата на период.

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 участват само s -ични числа, и извършване на изчисленията в s -ична ПБС.

Пример 69

Търсим $110,(01)_{(2)} = ?_{(10)}$:

От формула *27* , т. е. от

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

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

Пример 70

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

От формула *27*, т. е. от

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

се получава

$$11{,}000(0011)_{(2)}=\frac{11000_{(2)}.(2^4-1)+0011_{(2)}}{2^3.(2^4-1)}=\frac{24_{(10)}.15_{(10)}+3}{8.15_{(10)}}=\frac{363_{(10)}}{120_{(10)}}=3{,}025_{(10)}.$$

Пример 71

Търсим $0,(6)_{(10)}=?_{(7)}$:

От формула *27* , т. е. от

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

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

От деленето в 7-ична ПБС се получава

$$\begin{array}{l} -\ 6,0_{(7)}\ :\ 1\ 2_{(7)}\ =\ 0,(\,4\,)_{(7)} \\ \quad\ \ 5\ \ 1 \\ \hline \qquad\ 6 \end{array}.$$

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

Този резултат може да се провери чрез обратното преобразование:

От формула *27*, т. е. от

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

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

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

$$3\,.\,0,00(85)_{(10)}=0,02\underset{\overleftarrow{2}}{(}\underset{\overleftarrow{1}}{5}\;\underset{\overleftarrow{2}}{7})_{(10)}\ .$$

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

Също така се използва, че:

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

Пример 73

Търсим $0,(4)_{(10)}=?_{(2)}$:

$$\begin{array}{r|l} \multicolumn{2}{l}{2\,.\,0,\cancel{(4)}} \\ \hline 0 & (8) \\ 1 & (7) \\ 1 & (5) \\ 1 & (1) \\ 0 & (2) \\ 0 & \cancel{(4)} \end{array} \quad \Rightarrow \quad 0,(4)_{(10)}=0,(011100)_{(2)}.$$

Проверка чрез първия подход:

От формула *27*, т. е. от

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

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

Пример 74

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

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

$$10_{(10)}=22_{(4)};\quad \begin{array}{r} \underline{0,002(30)_{(4)}.22_{(4)}} \\ +\;\; 11(21) \\ 112(12) \\ \hline 0,123(33)_{(4)} \end{array};\quad \begin{array}{r} \underline{0,123(33)_{(4)}.22_{(4)}} \\ +\;\; 313(33) \\ 3\ 133(33) \\ \hline 10,113(33)_{(4)} \end{array};\quad \begin{array}{r} \underline{0,113(33)_{(4)}.22_{(4)}} \\ +\;\; 233(33) \\ 2\ 333(33) \\ \hline 3,233(33)_{(4)} \end{array};$$

$$\begin{array}{r} \underline{0,233(33)_{(4)}.22_{(4)}} \\ +\;\; 1\ 133(33) \\ 11\ 333(33) \\ \hline 13,133(33)_{(4)} \end{array};\quad \begin{array}{r} \underline{0,133(33)_{(4)}.22_{(4)}} \\ +\;\; 333(33) \\ 3\ 333(33) \\ \hline 10,333(33)_{(4)} \end{array};\quad \begin{array}{r} \underline{0,333(33)_{(4)}.22_{(4)}} \\ +\;\; 1\ 333(33) \\ 13\ 333(33) \\ \hline 21,333(33)_{(4)} \end{array}.$$

(Тук се използва формула *33*.)

$$\underline{22_{(4)}.0,002(30)_{(4)}}$$

$$\Rightarrow\quad \begin{array}{r|l}
0_{(4)} & 123(33)_{(4)} \\
10_{(4)} & 113(33)_{(4)} \\
3_{(4)} & 233(33)_{(4)} \\
13_{(4)} & 133(33)_{(4)} \\
\cancel{10_{(4)}} & 333(33)_{(4)} \\
\cancel{21_{(4)}} & 333(33)_{(4)}
\end{array}.$$

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

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

Проверка чрез първия подход:

От формула *27* , т. е. от

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

се получава

$$-11201,002(30)_{(4)}=-\frac{11201002_{(4)}.(4^2-1)+30_{(4)}}{4^3.(4^2-1)}=$$

$$=-\frac{22594_{(10)}.15_{(10)}+12_{(10)}}{64_{(10)}.15_{(10)}}=-\frac{338922_{(10)}}{960_{(10)}}=-353,04375_{(10)}.$$

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

$$\begin{array}{ccc|ccc|ccc|ccc|ccc|ccc|cccl}
 & 6 & , & & 3 & & & 5 & & & 4 & & (& 1 & & & 7 & & & 2 &) & _{(8)} \\
1 & 1 & 0, & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & (0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0) & _{(2)}
\end{array}$$

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

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

Пример 76

Търсим $0{,}102(221)_{(3)}=?_{(9)}$:

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

$$\begin{array}{cc|c|c|c|c|cc}
0 & , & 1\ \ 0 & 2\ \ 2 & (\ 2\ \ 1 & 2\ \ 2 & 1\ \ 2 &)_{(3)} \\
0 & , & 3 & 8 & (\ \ 7 & 8 & 5 &)_{(9)}
\end{array}$$

$$\Rightarrow\ 0{,}102(221)_{(3)}=0{,}38(785)_{(9)}.$$