Практическо упражнение No 6

КОДИРАНЕ НА ЧИСЛАТА В КОМПЮТЪРА

1. Цел на упражнението:

Целта на упражнението е студентите да задълбочат познанията си по начините за кодиране на числата в компютрите и за извършване на аритметични операции с тях.

2. Теоретична част:

Чрез кодирането на числата в компютрите се цели следното:

- кодиране на знака на числото;
- представяне на отрицателните числа чрез положителни;
- създаване на условия за просто и бързо изпълнение на аритметичните операции и по-точно за свеждане на всички аритметични операции с числа до аритметично събиране на техните кодове - това се налага, т.к. суматорът, който е основна съставна част на АЛУ, може да извършва само тази операция;
- формулиране на критерий (признак) за препълване на разрядната мрежа.

Използуват се три основни кода: прав, обратен и допълнителен. Във всеки един от тях се предвижда по един допълнителен разряд за кода на знака на числото (“+”→ “0”, “-”→ “1”), който се получава автоматично.

По-долу ще бъде разгледано кодирането на правилни двоични дроби. Кодирането на цели числа става по аналогичен начин.

2.1. Прав код.

а. Правило за кодиране.

При $a \geq 0$ $[a]_{ПК} = a$.

При $a \leq 0$ $[a]_{ПК} = 1+|a| = 1 - a$.

Пример:

a = +0,1001

└ разряд за цялата част на числото

$[a]_{ПК}$ = 0,1001

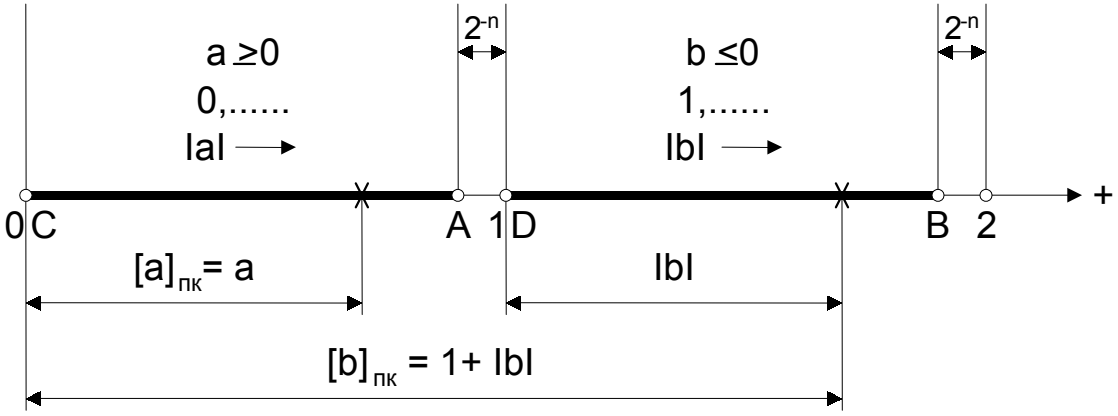
└ знаков разряд на правия код на числото

b = - 0,0011

$[b]_{ПК}$ = 1,0011

В този вид числата се съхраняват в ОП.

б. Геометрична интерпретация (фиг.1).



Фиг.1. Геометрично представяне на ПК

A - ПК на максималното положително число;
B - ПК на максималното по модул отрицателно число;
C - ПК на “положителната нула”;
D - ПК на “отрицателната нула”.

Забележка: Появата на понятията “положителна и отрицателна нула” е следствие от показаното по-горе правило за получаване на правия код на числата.

в. Извършване на аритметични операции с числа в ПК.

Събирането (a+b) на числа с еднакви знаци в ПК се извършва като се съберат мантисите на кодовете и на получената сума се присвои кодът на знака на събираемите.

Пример:

$[-0,1001]_{ПК}$ =	1,	1001
$+[-0,0011]_{ПК}$ =	1,	+ 0011
	1,	1100

Изваждането (a-b) на числа с еднакви знаци, т.е. алгебрическото събиране [a+(-b)] на числа с различни знаци в ПК се извършва като първо се определи кое от числата е по-голямо по модул, след което от мантисата на неговия код се извади мантисата на кода на другото число и на получената разлика се присвои кодът на знака на по-голямото по модул число. Очевидно е, че в този случай е необходима схема за сравнение и субтрактор. Следователно ПК не е удобен за извършване на тази операция.

Пример:

$[-0,0011]_{ПК}$	→	$[-0,0011]_{ПК}$ =	1,0011	0,	1001
$-[-0,1001]_{ПК}$		$+[+0,1001]_{ПК}$ =	+0,1001	1,	-0011
изваждане		алгебрическо събиране		0,	0110

ПК е удобен за извършване на операциите умножение и деление, тъй като мантисите на кодовете и на положителните и на отрицателните числа са равни на мантисите на самите числа, а това означава, че при извършване на тези операции с ПК на числата могат да се използват алгоритми, аналогични на тези, по които става умножението, респ. делението на самите числа. В ПК тези операции се извършват по следния начин:

• умножават се или се разделят кодовете на мантисите - така се получава кодът на мантисата на резултата;

• сумират се по модул 2 кодовете на знаците - така се получава кодът на знака на резултата;

Знак на a		Знак на b		Знак на a.b ; a/b	
+	0	+	0	+	0
+	0	-	1	-	1
-	1	+	0	-	1
-	1	-	1	+	0

• така получените кодове се обединяват в код на резултата.

Пример:

$[+0,1001]_{ПК} = 0,1001$ 0,1001 0+1 (mod 2) = 1

$x[-0,0011]_{ПК} = 1,0011$ x 0,0011

0,00011011

резултат: 1,00011011

2.2. Обратен код.

а. Правило за кодиране.

При $a \geq 0$ $[a]_{ОК} = a$.

При $a \leq 0$ $[a]_{ОК} = 2-2^{-n} - |a| = 2-2^{-n}+a$.

$2 - 2^{-n} = 1, 1\ 1 \ldots 1\ 1$

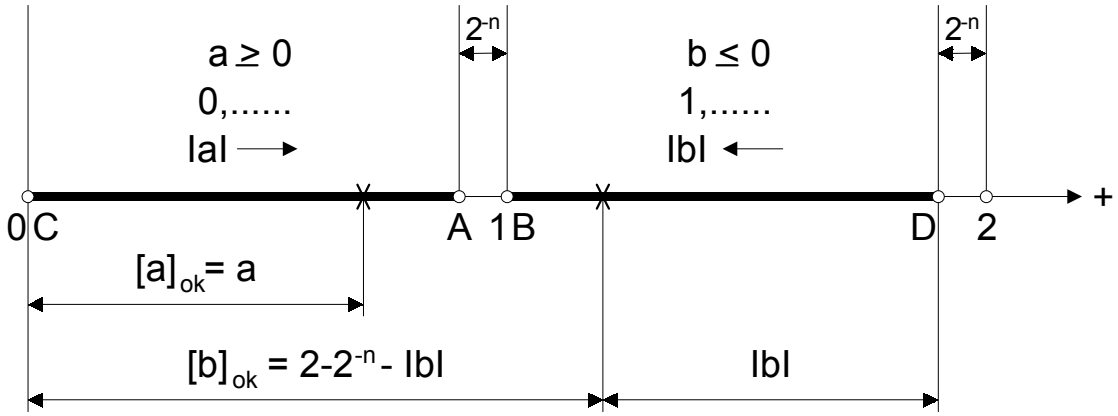
$-|a| = 0, a_1\ a_2 \ldots a_{n-1}\ a_n$ $a_i = 0$ или 1

$[a]_{ОК} = 1, \bar{a}_1\ \bar{a}_2 \ldots \bar{a}_{n-1}\ \bar{a}_n$ $1 - a_i = \bar{a}_i$

Следователно ОК на отрицателни двоични дроби се получава като в знаковия разряд се запише "1", а всички останали разряди се инвертират.

Примери:
a = +0,1001
└ разряд за цялата част на числото
$[a]_{ОК}$ = 0,1001
└ знаков разряд на обратния код на числото
b = - 0,0011
$[b]_{ОК}$ = 1,1100

б. Геометрична интерпретация (фиг.2).



Фиг.2. Геометрично представяне на ОК

A - ОК на максималното положително число;
B - ОК на максималното по модул отрицателно число;
C - ОК на “положителната нула”;
D - ОК на “отрицателната нула”.

в. Извършване на аритметични операции с числа в ОК.

В ОК е удобно изпълнението на операцията алгебрическо събиране на числата, тъй като тя се свежда до аритметическо събиране на техните кодове - операция, която може да бъде извършена с обикновен суматор. При това автоматически се получава и кодът на знака на резултата.

В ОК е удобно и изпълнението на операцията изваждане, т.к. тя лесно се свежда до алгебрическо събиране → a - b = a + (-b).

Правило: При алгебрическо събиране на двоични числа с използуване на ОК положителните събираеми се представят в ПК, а отрицателните в ОК и се извършва аритметическо събиране на тези кодове, като тази операция се разпростира и над знаковите разряди. При възникване на пренос от знаковия разряд, единицата на преноса се прибавя към младшия разряд на сумата на кодовете. В резултат се получава алгебрическата сума на числата в ПК, ако същата е ≥ 0 или в ОК, ако е < 0. (Модулът на сумата трябва да бъде по-малък от 1).

Доказателство:

Възможни са четири различни случая:

1-ви случай: a > 0, b > 0; (a + b) > 0;

2-ри случай: a > 0, b < 0; a > |b|; (a + b) > 0;

3-ти случай: a > 0, b < 0; a < |b|; (a + b) < 0;

4-ти случай: a < 0, b < 0; (a + b) < 0

Доказателството ще бъде направено само за 2-рия случай като най-комплициран. Валидността на правилото в останалите три случая се доказва по аналогичен начин.

Нека: a > 0; b < 0; a > |b|; (a + b) > 0.

Според правилото, ако $[a]_{ПК} = a$, $[b]_{ОК} = 2\text{-}2^{-n} + b$, то като се съберат тези кодове и се отчете евентуално възникналият пренос, трябва да се получи $[a + b]_{ПК} = a + b$.

В действителност се получава:

$[a]_{ПК} + [b]_{ОК} = a + 2\text{-}2^{-n} + b = a + b + 2\text{-}2^{-n} = [a + b]_{ПК} + 2\text{-}2^{-n}$

$2\text{-}2^{-n}$ = 1,11...1...11

\+

$[a + b]_{ПК}$ = 0,.....1.....

\+

→ 1

Следователно в този случай от знаковия разряд винаги възниква пренос, който според правилото се прибавя към младшия разряд на сумата на кодовете, след което се получава :

$[a + b]_{ПК} + 2\text{-}2^{-n} + 2^{-n} = [a + b]_{ПК} + 2.$

Тъй като двойката (10,0..) излиза от разрядната мрежа в ляво и се губи, то след прибавянето на преноса се получава $[a + b]_{ПК}$ както и трябва да бъде.

Пример:

a = +0,1001; b = -0,0011; a+b = +0,0110

Съгласно правилото:

$[a]_{ПК} = 0,1001$; $[b]_{ОК}=1,1100$; $[a+b]_{ПК}= 0,0110$

В действителност:

0,1001

\+

1,1100

0,0101

\+

→ 1

0,0110

ОК не е удобен за изпълнение на операциите умножение и деление, т.к. мантисите на ОК на отрицателните числа се различават от мантисите на самите числа.

2.3. Допълнителен код.

а. Правило за кодиране.

При $a \geq 0$ $[a]_{ДК} = a$.

При $a \leq 0$ $[a]_{ДК} = 2 - |a| = 2+a$.

На практика ДК на отрицателни двоични дроби се получава като към младшия разряд на ОК се прибави единица:

$$\begin{array}{l} [a]_{ДК} = 2 + a \\ + \\ 2-2^{-n}+a = [a]_{ОК} \end{array}$$

$$[a]_{ДК} = [a]_{ОК} + 2^{-n}$$

Примери:

a = +0,1001

└ разряд за цялата част на числото

$[a]_{ДК}$ = 0,1001

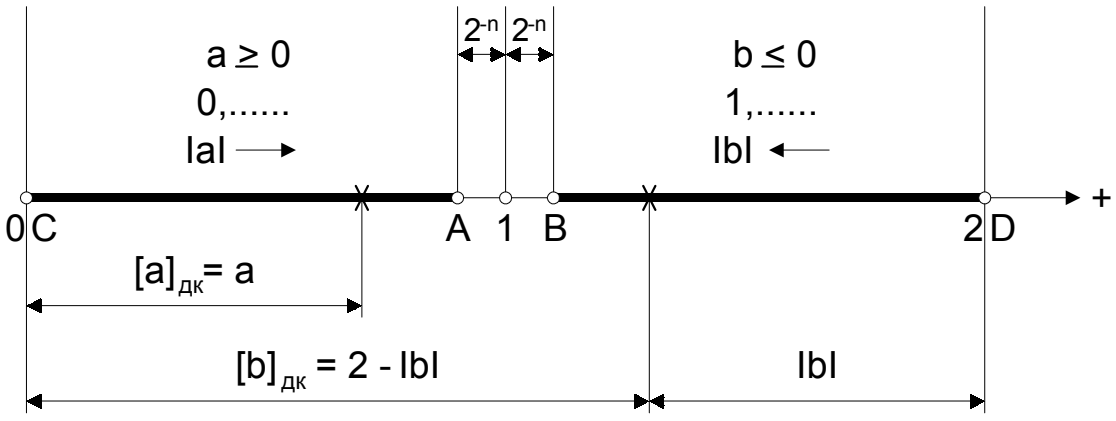
└ знаков разряд на допълнителния код на числото

b = - 0,0011

$[b]_{ДК}$ = 1,1101

Преобразуването от ДК в ПК при числа ≤ 0 става чрез инвертиране на разрядите на мантисата и прибавяне на единица към най-младшия разряд.

б. Геометрична интерпретация (фиг.3).



Фиг.3. Геометрично представяне на ДК

A - ДК на максималното положително число;
B - ДК на максималното по модул отрицателно число;
C - ДК на “положителната нула”;
D - ДК на “отрицателната нула”.

в. Извършване на аритметични операции с числа в ДК.

В ДК е удобно изпълнението на операцията алгебрическо събиране на числата, тъй като тя се свежда до аритметическо събиране на техните кодове - операция, която може да бъде извършена с обикновен суматор. При това автоматически се получава и кодът на знака на резултата.

В ДК е удобно и изпълнението на операцията изваждане, т.к. тя лесно се свежда до алгебрическо събиране $\rightarrow$ a - b = a + (-b).

Правило: При алгебрическо събиране на двоични числа с използуване на ДК положителните събираеми се представят в ПК, а отрицателните в ДК и се извършва аритметическо събиране на тези кодове, като тази операция се разпростира и над знаковите разряди. При възникване на пренос от знаковия разряд, той не се отчита. В резултат се получава алгебрическата сума на числата в ПК, ако същата е ≥ 0 или в ДК, ако е < 0. (Модулът на сумата трябва да бъде по-малък от 1).

Доказателство:
Възможни са четири различни случая:
1-ви случай: a > 0, b > 0; (a + b) > 0;
2-ри случай: a > 0; b < 0; a > |b|; (a + b) > 0;
3-ти случай: a > 0; b < 0; a < |b|; (a + b) < 0;
4-ти случай: a < 0, b < 0; (a + b) < 0
Доказателството ще бъде направено само за 2-рия случай като най-комплициран. Валидността на правилото в останалите три случая се доказва по аналогичен начин.
Нека: a > 0; b < 0; a > |b|; (a + b) > 0.
Според правилото, ако $[a]_{ПК} = a$, $[b]_{ДК} = 2+b$, то като се съберат тези кодове, трябва да се получи $[a + b]_{ПК} = a + b$.
В действителност се получава:
$[a]_{ПК} + [b]_{ДК} = a + 2 + b = a + b + 2 = [a + b]_{ПК} + 2$.
Тъй като двойката (10,0...) излиза от разрядната мрежа в ляво и се губи, то се получава $[a + b]_{ПК}$, както и трябва да бъде.

ДК не е удобен за изпълнение на операциите умножение и деление, но независимо от това, в някои машини се използува именно този код по причини, които ще бъдат разяснени по-късно.

Ако $a \geq 0$, то $[a]_{ПК} = [a]_{ОК} = [a]_{ДК}$. Затова често се казва, че ОК и ДК се използуват за представяне само на отрицателни числа.

Чрез разгледаните до тук кодове, се постигат първите три цели. Но нито един от тях не позволява да се открие кога се получава препълване на разрядната мрежа. Този проблем е решен, чрез въвеждането на така наречените модифицирани кодове.

2.4. Модифицирани кодове.

Тези кодове се отличават от вече разгледаните по това, че при тях за кодиране на знака на числото се използуват два разряда (“+” $\rightarrow$ “00”, “-”$\rightarrow$“11”).

а. Правило за кодиране.

При $a \leq 0$

$[a]_{МПК} = [a]_{МОК} = [a]_{МДК} = a$ 00,...

При $a \leq 0$

$[a]_{МПК} = 3 + |a|$ 11,...

$[a]_{МОК} = 4 - 2^{-n} - |a|$ 11,...

$[a]_{МДК} = 4 - |a|$ 11,...

б. Признак за препълване на разрядната мрежа при използване на модифицирани кодове е различието на цифрите в двата знакови разряда, т.е. получаването на 01,... или 10,... . При това левият знаков разряд показва знака на резултата. При 01,... резултатът е положителен и е ≥ 1, а при 10,... - резултатът е отрицателен и по модул е ≥ 1.

Пример:

По принцип препълване на разрядната мрежа, т.е. резултат по модул ≥ 1 се получава при събиране на числа с еднакви знаци или при изваждане на числа с различни знаци при условие, че $|a|+|b| \geq 1$.

7/16 + 11/16 = 18/16

Ако се използува обикновен ПК, се получава:

  0,0111
+ 0,1011
  ———————
  1,0010 - тази сума ще бъде възприета от машината като (-2/16).

Ако се използува модифициран ПК, се получава:

  00,0111
+ 00,1011
  ———————
  01,0010 - тази сума ще бъде възприета като ≥ 1 и ще бъде вдигнат флагът OV.

В резюме може да се отбележи следното:

Когато в **машина с ФЗ** се получи **препълване на разрядната мрежа**, т.е. 01,... или 10,... то се вдига автоматично флагът OV. Някои машини автоматично спират работата и сигнализират на оператора.

Когато в **машина с ПЗ** се получи **нарушаване на нормализацията,** т.е. 2^P.01,...; 2^P.10,...; 2^P.00,0...; 2^P.11,1... (отрицателните числа са в ОК или ДК), автоматично се извършва нормализация на резултата:

2^P.01,... ; 2^P.10,... - нарушаване на нормализацията на 1 разряд в ляво - числата се нормализират чрез изместване на мантисата на един разряд в дясно, т.е. до получаване съответно на 00,1.. или 11,0.., а към порядъка се прибавя 1.

2^P.00,0....01... ; 2^P.11,1....10... - нарушаване на нормализацията на m разряд в дясно - числата се нормализират чрез изместване на мантисата на m - разряда в ляво, т.е. до получаване на 00,1.. или 11,0.., а от порядъка се изважда m. (При използуване на ДК това е вярно само, ако след нулата има още поне една единица. В противен случай последната единица от групата m единици е значеща цифра и нарушението е на (m-1) разряда в дясно.)

2.5. Прав, обратен и допълнителен код на правилни десетични двоично-кодирани дроби.

а. При използуване на код 8421.

При $A \geq 0$ $[A]_{ПК}=A$.
При $A \leq 0$ $[A]_{ПК}=1 + |A| =1 - A$.

При $A \geq 0$ $[A]_{ОК}=A$.
При $A \leq 0$ $[A]_{ОК}=2 - 10^{-n} - |A| = 2 - 10^{-n} + A$.

ОК на отрицателни десетични дроби кодирани чрез кода 8421 се получава като в знаковия разряд се запише "1", всички двоични разряди на мантисата се инвертират и към всяка тетрада се прибави корекция $+(1010)_2$ без да се отчита възникващият при това пренос (от тетрада към тетрада).

Пример:

$A = (-0,278)_{10} =(-0,0010\ 0111\ 1000)_{10/2}$

$$\begin{array}{rl} & 1,1101\ 1000\ 0111 \\ + & \ \ 1010\ 1010\ 1010 \ \text{- корекция} \\ \hline [A]_{ОК} = & 1,0111\ 0010\ 0001 \end{array}$$

При $A \geq 0$ $[A]_{ДК}=A$.
При $A \leq 0$ $[A]_{ДК}=2 - |A| =2 + A =[A]_{ОК} + 2^{-4n}$ + корекция (+6) към младшата тетрада, ако след прибавяне на 2^{-4n} се получи код на число > 9.

Забележка: По-горе с A е означен двоично-десетичният код на някаква правилна десетична дроб, а с n - броят на десетичните разряди.

б. При използуване на код с излишък 3.

При $A \geq 0$ $[A]_{ПК}=A$

При $A \leq 0$ $[A]_{ПК}=1+|A| = 1 - A$

При $A \geq 0$ $[A]_{ОК}=A$

При $A \leq 0$ $[A]_{ОК}=2 - 10^{-n} - |A| = 2 - 10^{-n} + A$

ОК на отрицателни десетични дроби кодирани чрез кода с излишък 3 се получава като в знаковия разряд се запише "1", а всички двоични разряди на мантисата се инвертират.

Пример:

$A = (-0,278)_{10} = (-0,0101\ 1010\ 1011)_{10/2}$

$[A]_{ОК} = 1,1010\ 0101\ 0100$

При $A \geq 0$ $[A]_{ДК}=A$

При $A \leq 0$ $[A]_{ДК}=2 - |A|=2 + A=[A]_{ОК} + 2^{-4n}$

При този начин за формиране на кодовете правилата за алгебрическо събиране остават същите, но се налага внасянето на съответни корекции.

3. Задачи за изпълнение:

3.1. Да се представят в ПК, ОК и ДК числата:
+0,10111001
- 0,00011110

3.2. Като се използва модифициран ОК да се пресметне ($\frac{11}{16} - \frac{3}{16}$). В какво състояние ще се установи флагът OV в регистъра на признаците на резултата?

3.3. Като се използва модифициран ДК да се пресметне ($-\frac{15}{16} - \frac{7}{16}$). В какво състояние ще се установи флагът OV в регистъра на признаците на резултата?

3.4. Да се нормализират при необходимост числата:
2^p.01,10011011
2^p.00,00011010
2^p.10,11101001
2^p.11,11010101
2^p.00,11110001

Забележка: За представяне на отрицателните числа е използван ДК.

3.5. Да се изследва работата на програмния модел на блока за нормализиране на числата (фиг.4).

3.6. Да се представи в ПК, ОК и ДК числото -0,561 (-0,937) с използване на код 8421 (код с излишък 3).

4. Контролни въпроси:

4.1. Какво е правилото за получаване на ПК? За извършване на какви аритметични операции е удобен този код?

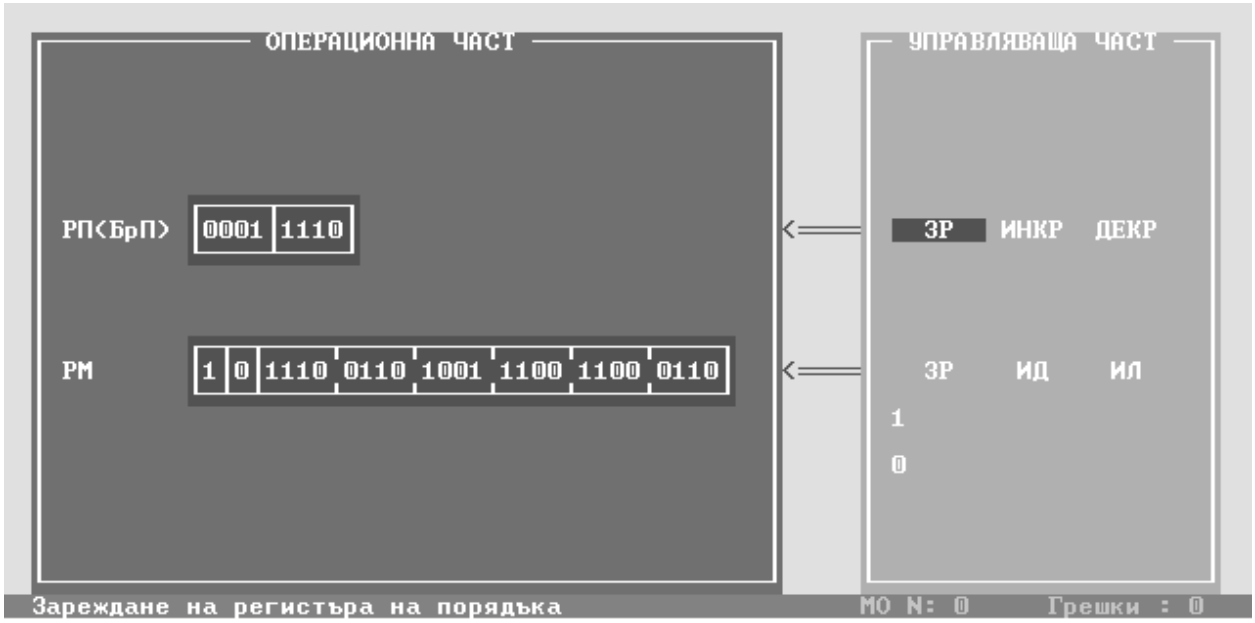
4.2. Какво е правилото за получаване на ОК? За извършване на какви аритметични операции е удобен този код?

4.3. Какво е правилото за получаване на ДК? За извършване на какви аритметични операции е удобен този код?

4.4. С каква цел се въвеждат модифицираните кодове?

4.5. Как реагират машините с ФЗ при препълване на разрядната мрежа?

4.6. Как реагират машините с ПЗ при нарушаване на нормализацията?



Фиг.4. Програмен модел на блока за нормализиране на числата

BACK