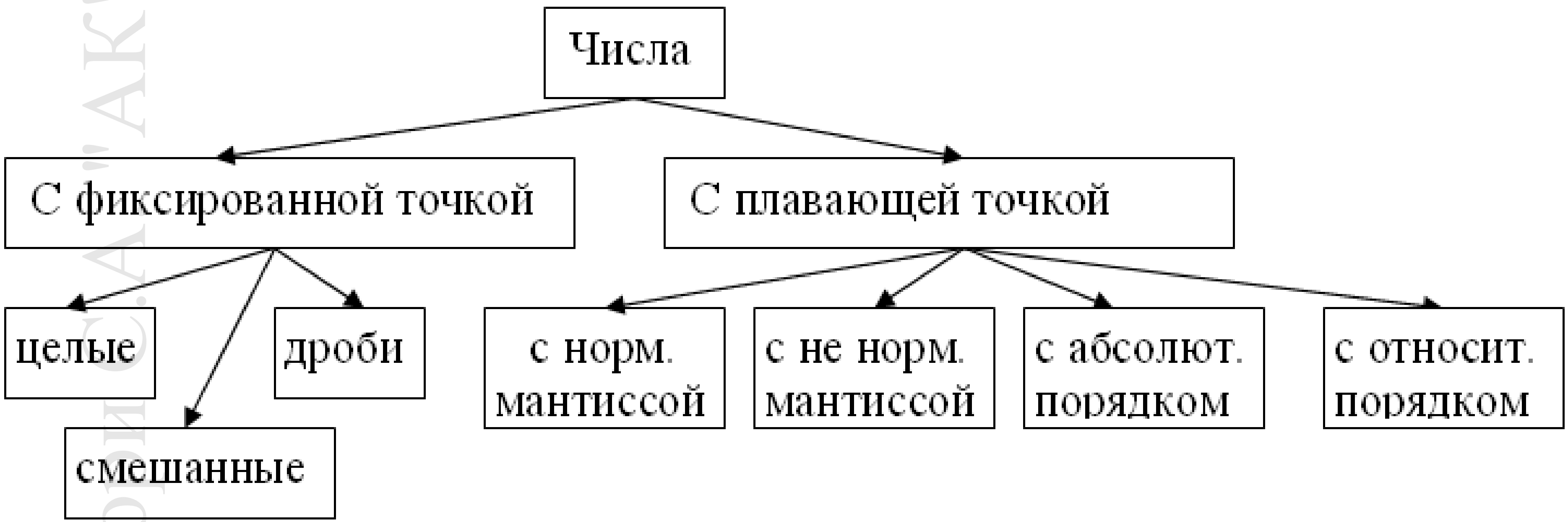
ПРЕДСТАВЛЕНИЕ И ОСНОВНЫЕ ОПЕРАЦИИ НАД ЧИСЛАМИ В КОМПЬЮТЕРЕ

I. Способы представления (форматы) чисел в компьютере

Известны следующие основные способы представления числовой информации в компьютере:



Представление чисел любым способом характеризуется двумя основными параметрами: ***точностью*** и ***диапазоном***.

Они в свою очередь определяются ***разрядной сеткой компьютера*** – количеством разрядов (двоичных цифр, бит), отведенных для представления (записи, хранения, обработки) числа.

•Разрядная сетка компьютера при использовании стандартных наборов команд (обработка чисел) ***ограничена***.

•Количество разрядов, используемых в компьютере для хранения чисел в разных представлениях (форматах), может быть ***различно***.

•Алгоритмы обработки чисел в любом представлении *примерно* ***одинаковы***.

Рассмотрим более подробно представление чисел в каждом из форматов.

Числа с фиксированной точкой

Точка – разделитель **целой** и **дробной** части числа:

- позиция точки *известна*;
- точка *никогда не хранится* (не выделяется дополнительный разряд для ее хранения), она – ***подразумевается*** (учитывается алгоритмами обработки и интерпретации чисел).

Дале будем считать, что все числа (**z**) – *n*-разрядные, представленные в *двоичной* с.с..

- Знак числа всегда хранится **в старшем бите** представления числа**:**

  0 – неотрицательное

  1 – отрицательное

Целые числа:

n-1	*n-2*		*2*	*1*	*0*	
Зн	**цифровые разряды**					**.**

Точка фиксирована после младшего разряда (единиц).
Разряды обычно нумеруются от *0* до *n-1* (всего *n*).

Знак (Зн) обычно кодируется следующим образом: 0 – “+”, 1 – “–”

Основные характеристики:

$|z| < 2^{n-1}$, если условие нарушается – переполнение разрядной сетки.

Диапазон: $\lfloor -(2^{n-1}-1); +(2^{n-1}-1) \rfloor$

Точность: $\delta = 2^{-1}$

Дроби:

0	*1*		*n-2*	*n-1*
Зн	**. цифровые разряды**			

Точка фиксирована после знакового разряда (подразумевается; целая часть числа – 0 – также не хранится).

Основные характеристики:

$|z| < 1$, если нет – переполнение разрядной сетки.

Диапазон: $\left[-(1-2^{-(n-1)});+(1-2^{-(n-1)})\right]$

Точность: $\delta = 2^{-n}$

Смешанные:

0	*1*	*2* *m*	*m+1*	*n-1*
Зн	**Целая часть .**		**Дробная часть**	

Точка фиксирована между целой и дробной частью.
Используются редко (обычно в специальных случаях).
Основные характеристики: *самостоятельно*

<u>Замечание</u>: числа в любом из рассмотренных представлений могут быть беззнаковыми (считается, что число ≥0). Тогда знаковый разряд используется как дополнительный цифровой – диапазон расширяется в два раза, точность увеличивается в два раза.

2. <u>Числа с плавающей точкой:</u>

Формат используется для представления действительных чисел. При этом число представляется в следующем виде:

$$Z = \pm M \cdot d^{\pm \Pi},$$ где

M – мантисса числа, $|M| < 1$ - дробь,

П – порядок числа, $|\Pi| \geq 0$ - целое число,

d – основание порядка (обычно 2^r),

M, П – обычно двоичные числа.

Таким образом, представление в этом формате числа Z:

0	*1*		*m*	*m+1*	*m+2*		*m+p+1*
Зн М	**М**			**Зн П**	**П**		
1 .	m			1	p		.

2.1. С нормализованной мантиссой:

Считается, что мантисса нормализована, если выполняется следующее условие:

$$\frac{1}{d} \le \left| M_{норм} \right| < 1$$

Диапазон: $\frac{1}{d} \cdot d^{-(2^P-1)} \le |z| \le (1-2^{-m}) \cdot d^{2^P-1}$

Точность: $\delta = d^{П} 2^{-m}$

2.2. С ненормализованной мантиссой:

$$0 \le \left| M_{не\_норм} \right| < \frac{1}{d}$$

Диапазон: $2^{-m} \cdot d^{-(2^P-1)} \le |z| \le (\frac{1}{d} - 2^{-m}) \cdot d^{2^P-1}$

Точность: $\delta = d^{П} 2^{-m}$

Плавающая точка

Пример:

X = 5.3, d = 10, десятичное хранение в памяти

$5.3 = 0.53 * 10^1$

M = 0.53, П = 1 нормализованная M

$5.3 = 0.053 * 10^2$

M = 0.053, П = 2 ненормализованная M

Плавающая точка

Пример:

X = 3, d = 2, двоичная с.с. для хранения в памяти

$3_{10} = 11._2 = \quad 0.11_2 * 2^2 = \quad 0.11_2 * 2^{10_2}$

М = 0.**11**$_2$, **П** = **10**$_2$ нормализованная М

$3_{10} = 11._2 = \quad 0.011_2 * 2^3 = \quad 0.011_2 * 2^{11_2}$

М = 0.**011**$_2$, **П** = **11**$_2$ ненормализованная М

Что лучше??

Плавающая точка – нормализованная мантисса VS ненормализованная

Пусть под Мантиссу отведено **2** разряда. Тогда:

$X = \mathbf{3}_{10} = 11._2 \;=\; 0.\mathbf{11}_2 * 2^{\mathbf{2}}$

$M = 0.11_2,\ П = 10_2$ нормализованная М

$\mathbf{M = .}\boxed{\mathbf{1}}\boxed{\mathbf{1}}$ => $\mathbf{X} = 0.11_2 * 2^2 = \mathbf{3}$

$X = 3_{10} = 11._2 \;=\; 0.\mathbf{011}_2 * 2^{\mathbf{3}}$

$M = 0.011_2,\ П = 11_2$ ненормализованная М

$\mathbf{M = .}\boxed{\mathbf{0}}\boxed{\mathbf{1}}\boxed{\mathbf{1}}$ => $\mathbf{X} = 0.01_2 * 2^3 = \mathbf{2}$

$X = 3_{10} = 11._2 \;=\; 0.\mathbf{0011}_2 * 2^{\mathbf{4}}$

$M = 0.0011_2,\ П = 100_2$ ненормализованная М

$\mathbf{M = .}\boxed{\mathbf{0}}\boxed{\mathbf{0}}\boxed{\mathbf{1}}\boxed{\mathbf{1}}$ => $\mathbf{X} = 0.00_2 * 2^4 = \mathbf{0}$

Замечание:Так как у чисел с нормальной мантиссой старшие цифровые разряды мантиссы не могут быть заняты незначащими нулями (все заняты значащими цифрами), то точность представления в этом формате выше, чем в формате с ненормализованной мантиссой (разряды M используются максимально эффективно). Это обусловливает применения этого формата в качестве основного формата для представления, хранения и обработки чисел с плавающей точкой (запятой) в ЭВМ.

2.3. С абсолютным (несмещенным) порядком:

Порядок представляет собой целое число со знаком (отсчитывается от 0, может быть положительным и отрицательным).

2.4. С относительным (смещенным) порядком:

Порядок представляет собой беззнаковое положительное целое число (минимальный порядок принят за 0).

Представление порядка

Пусть под Порядок отведено **4** двоичных разряда. Тогда:

Несмещенный (абсолютный) порядок (со знаком):

ЗнП	**X**	**X**	**X**

.

Минимальный = -111. = 1111 = - 7

Ноль = + 000. = 0000 = 0

Максимальный = +111. = 0111 = +7

Смещенный (относительный) порядок (без знаковый):

X	**X**	**X**	**X**

.

Минимальный = 0000 => 0 = -8 (смещение = -8)

Ноль = 1000 => 8 = 0 (**смещение** = -8)

Максимальный = 1111 => 15 = +7 (смещение = -8)

II. Отображение двоичных чисел со знаком в компьютере

Для хранения и обработки **чисел со знаком**

(«+» и «-») в любом из рассмотренных выше форматов

(с фиксированной или плавающей точкой) в компьютере

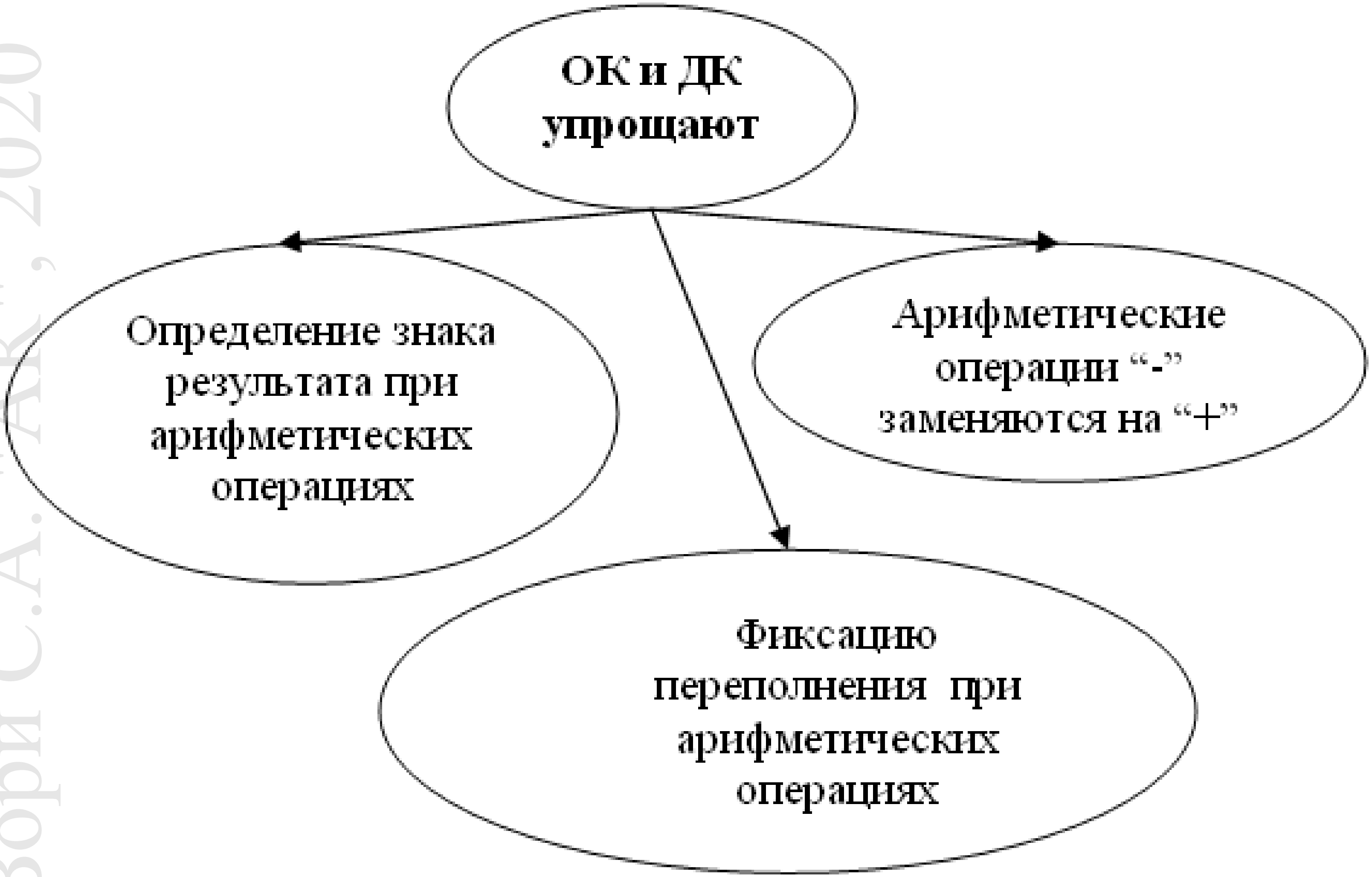
используется три основных представления:

в ***прямом** **коде (ПК)***,

в ***обратном** **коде (ОК)*** и

в ***дополнительном** **коде (ДК)***.

- Они различаются ТОЛЬКО **способом записи цифр** для положительных и отрицательных чисел.


ОК и ДК упрощают
Определение знака результата при арифметических операциях
Арифметические операции “-” заменяются на “+”
Фиксацию переполнения при арифметических операциях

Рассмотрим формы записи чисел в ПК, ОК и ДК на примере дробей.

Представление в <u>ПК</u>

$$Z = \begin{cases} \overset{знак}{0}.|z|, z \geq 0 \\ \overset{знак}{1}.|z|, z < 0 \end{cases}$$

Представление в <u>ОК</u>

$$Z = \begin{cases} 0.|z|, z \geq 0 \\ 1.|\bar{z}|, z < 0 \end{cases}$$

Представление в <u>ДК</u>

$$Z = \begin{cases} 0.|z|, z \geq 0 \\ 1.|\bar{z}| + 1_{мл.р.}, z < 0 \end{cases}$$

(фактически нужно прибавить единицу к младшему разряду изображения отрицательного числа в ОК)

Прямой код

- Прямой код (ПК) – значащие разряды записываются для отрицательных чисел так же, как и для положительных

Пример (длина разрядной сетки равна 8)

$115_{10} = 01110011_{ПК}$

$-115_{10} = 11110011_{ПК}$

Прямой код

- Наиболее удобен для человека
- Имеет 2 нуля: +0 и -0
- Недостаток: трудно алгоритмически складывать числа разного знака (или вычитать числа одного знака)
  – необходим анализ знака перед действием и отдельные схемы сложения и вычитания
  – решение - использовать другие представления (обратный или дополнительный код)

Обратный код

- Обратный код (ОК)
  - для положительных чисел совпадает с ПК
  - для отрицательных все разряды **инвертируются**

Пример

$$114_{10} = 01110010_{ОК}$$

$$-114_{10} = 10001101_{ОК}$$

Дополнительный код

- Дополнительный код (ДК)
  - для положительных чисел совпадает с ПК и ОК
  - для отрицательных все разряды **инвертируются**, затем к младшему **прибавляется 1**

Пример

$114_{10} = 01110010_{ДК}$

$-114_{10} = 10001101_{ОК} = 10001110_{ДК}$

Для дробей - аналогично

Пример.

$Z = \frac{3}{16}$	$Z = -\frac{3}{16}$
ПК: 0.0011	ПК: 1.0011
ОК: 0.0011	ОК: 1.1100
ДК: 0.0011	ДК: 1.1101

Обратные правила перевода аналогичны.

Сравнение кодов

	L бит		8 бит		16 бит		нули
	мин.	макс.	мин.	макс.	мин.	макс.	
без знака	0	2^L-1	0	255	0	6555	0
ПК, ОК	$-2^{L-1}-1$	$2^{L-1}-1$	-127	127	-32767	-32767	+0, -0
ДК	-2^{L-1}		-128		-32768		0

В современных компьютерах используется **ДК**!!

III. Базовые операции над двоичными числами в компьютере

1. Логические операции

Производятся поразрядно (побитно), результат в текущем разряде ***не влияет*** *на другие разряды результата.*

НЕ – логическое отрицание, ¬

$$c = \overline{a}$$

Таблица истинности:

a		c
0		1
1		0

ИЛИ – логическое сложение, $\vee$

$$c = a \vee b$$

Таблица истинности:

a	b		C
0	0		0
0	1		1
1	0		1
1	1		1

И – логическое умножение, $\wedge$

$$c = a \wedge b$$

Таблица истинности:

a	b		c
0	0		0
0	1		0
1	0		0
1	1		1

Исключающее или – сумма по модулю 2, $\oplus$

$$c = a \oplus b$$

Таблица истинности:

a	b		c
0	0		0
0	1		1
1	0		1
1	1		0

2. Арифметические операции

Операция **сложения** является базовой арифметической операцией (фактически используется в алгоритмах выполнения любых других арифметических операций и вычисления сложных функций).

- Сложение в текущем разряде происходит *с учётом результата сложения в предыдущем разряде* - входного переноса ***Cin*** (который является выходным переносом ***Cout*** от сложения в предыдущем разряде) и влияет соответственно на следующий разряд результата.
- Сложение начинается ***с младших разрядов*** слагаемых и выполняется ***последовательно***.

$S = a + b$

a	**b**	$\mathbf{C_{in}}$		$\mathbf{C_{out}}$	**S**
0	0	0		0	0
0	0	1		0	1
0	1	0		0	1
0	1	1		1	0
1	0	0		0	1
1	0	1		1	0
1	1	0		1	0
1	1	1		1	1

IV. Основные алгоритмы выполнения арифметики в компьютере

Основные алгоритмы выполнения операции сложения (вычитания) в компьютере

Сложение (вычитание) чисел

- В связи с тем, что **формальные правила сложения** чисел ***одинаковы*** для целых и дробных чисел, в дальнейшем (в примерах) будем рассматривать числа в форме двоичных дробей.
- При выполнении операций нужно учитывать, что результат выполнения АО над числами с ФЗ может *выйти за границы диапазона* (разрядной сетки).

Если модуль результата превышает верхнюю границу диапазона, то старшие разряды числа теряются и в разрядной сетке записывается неверный результат!

Такая ситуация называется **переполнением** ***разрядной сетки***.

При переполнении производится обязательная ***фиксация переполнения*** и формируется соответствующее сообщение, результат ***отбраковывается***.

Переполнение

- Переполнение – получение в результате арифметической операции результата, выходящего за *диапазон* представления чисел используемого формата
  - может возникнуть при сложении чисел одного знака, но не разных!
  - его *нужно* обнаруживать во время вычислений, т.к. он приводит к неверному результату!

Переполнение

- 2 метода обнаружения переполнения (эквивалентных):
  - **если операнды имеют одинаковый знак, а знак результата отличается от них**
    (знаковый (старший) разряд результата испорчен «непоместившийся» в отведенное количество разрядов старшей цифрой результата)
  - **если был перенос только в знаковый разряд или только из знакового разряда**
    (несовпадение переносов из- и в- знаковый разряд)

Переполнение

- **Пример**: 78 + 64 = 142 , 8 бит разрядная сетка (диапазон представления чисел = +/-127!!)

```
01111110          - перенос

 01001110
+
 00111110
 ________
 10001100  – переполнение, т.к.:
```

- Не совпадают переносы из знакового разряда и в него
- сумма двух положительных чисел отрицательна

1. Сложение чисел в ПК

Алгебраическое сложение чисел в ПК выполняется на двоичном сумматоре для **модулей чисел**, то есть над цифровой частью. **Знаки чисел обрабатываются отдельно**.

С целью упрощения (унификации) операция выполняется для всех разрядов числа, включая знаковый. Однако, значение, получаемое в знаковом разряде, игнорируется.

При вычитании в ПК знак вычитаемого можно заменить на противоположный, а затем выполнить операцию сложения, т.е.:

$$\mathbf{A - B = A + (-B)}$$

Таким образом, операцию в ПК можно представить следующим образом:

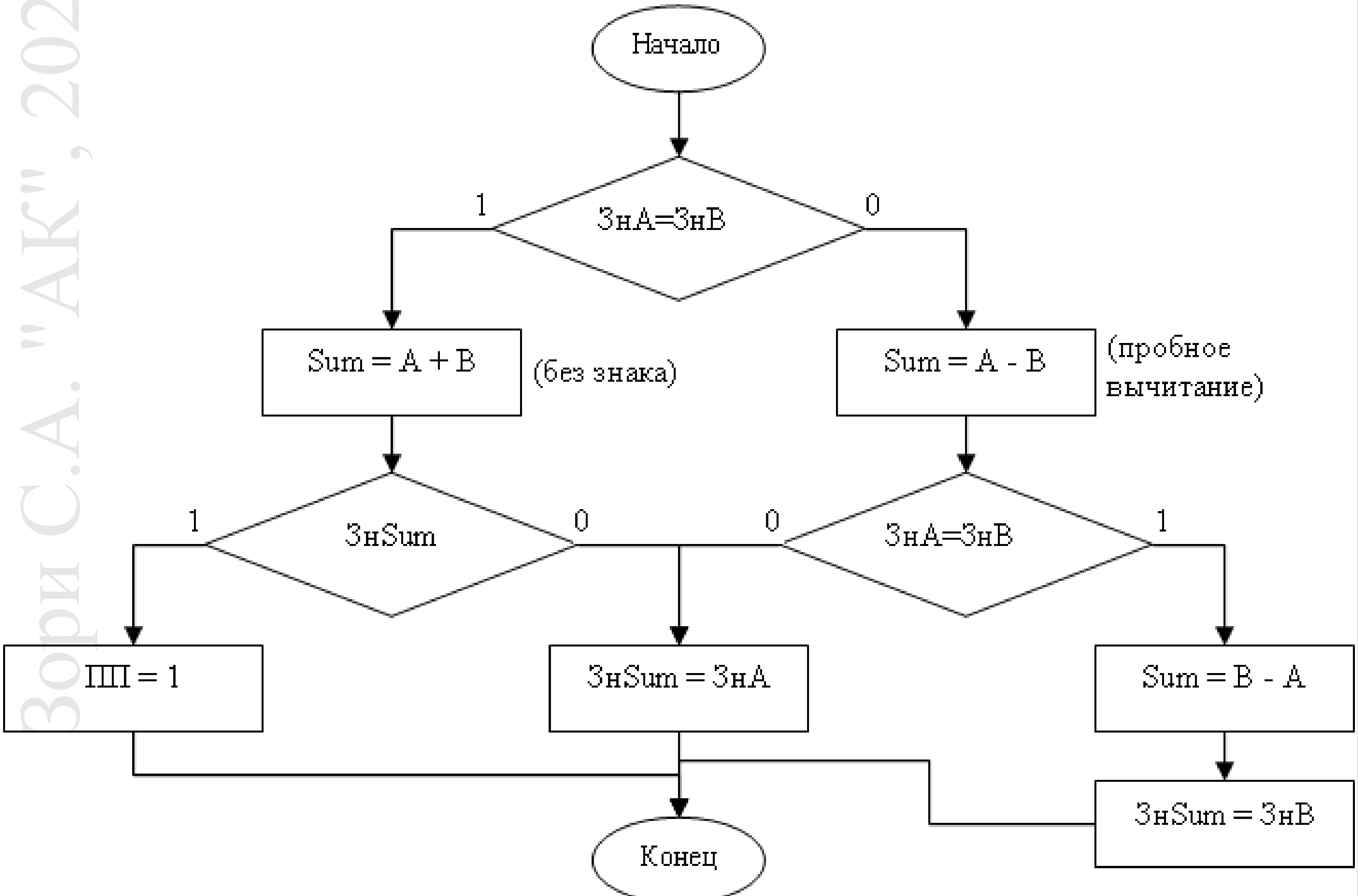
Sum = A + B

$A_{пк}$

ЗнА	$\cdot\lvert A\rvert$

$B_{пк}$

ЗнВ	$\cdot\lvert B\rvert$



2. Сложение чисел в ОК и ДК

Алгебраическое сложение чисел в ДК и ОК выполняется *<u>включая знаковые разряды</u>*.

При сложении в ДК *перенос из знакового разряда* ***игнорируется***.

В ОК выполняется ***циклическое сложение***, т.е. перенос из знакового разряда еще раз суммируется с младшим разрядом цифровой части.

Операция ***вычитания*** чисел в ОК и ДК заменяется сложением: $A – B = A + (-B)$

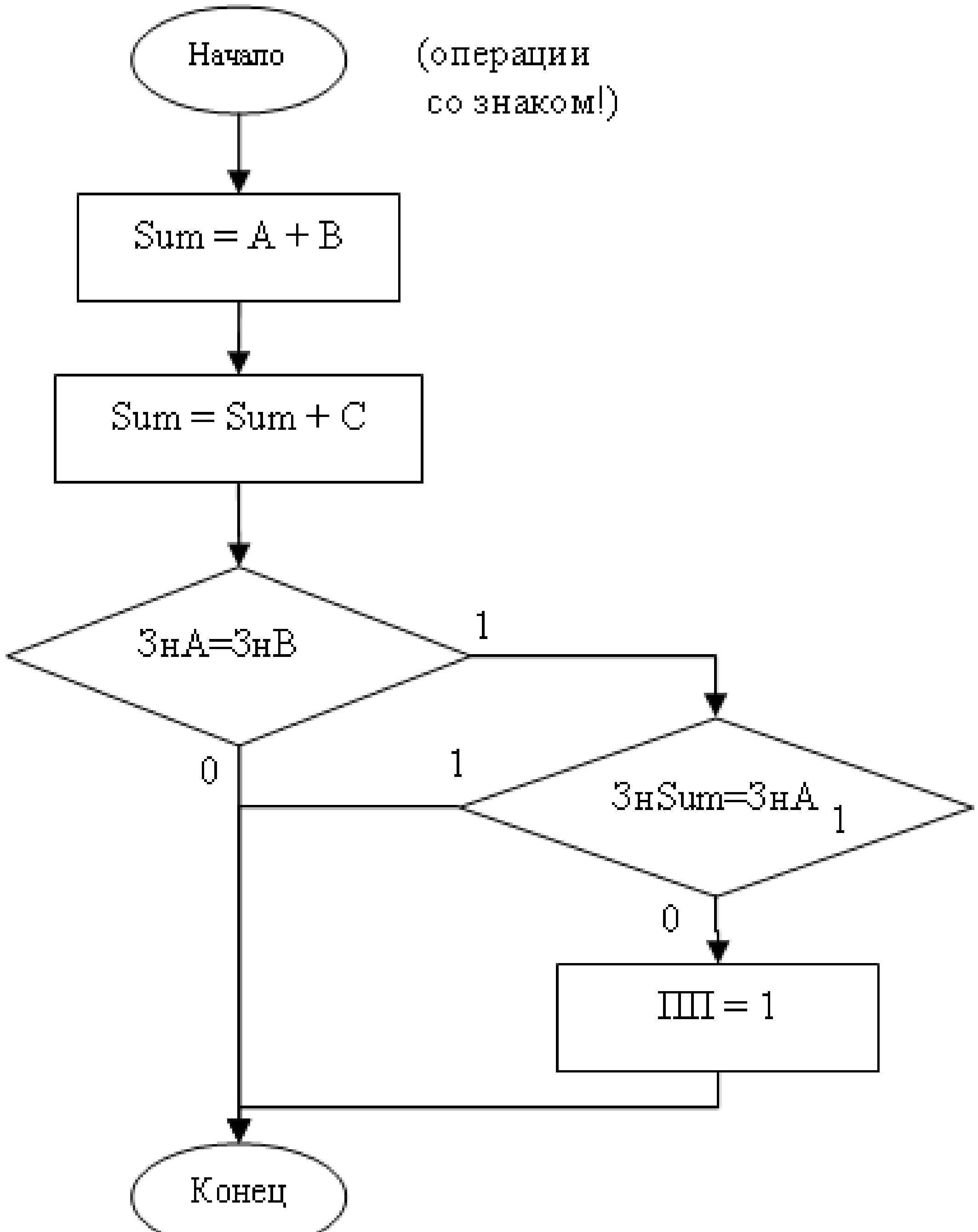
Для фиксации переполнения производится анализ **знаков** операндов и результата. *Переполнение возможно только при сложении чисел с одинаковыми знаками*.

- Если в этом случае *знак результата* противоположен знакам операндов, то имеет место переполнение.

Второй способ определения переполнения состоит в анализе значений переносов в знаковый разряд и из него.

- Если они *различны* - то переполнение.

В ОК


Начало
(операции
со знаком!)
Sum = A + B
Sum = Sum + C
ЗнА=ЗнВ
1
0
1
ЗнSum=ЗнА
1
0
ПП = 1
Конец

Сложение в ОК

- Пример: 10 - 114

$10_{10} = 00001010_{ОК}$

$-114_{10} = 11110010_{ПК} = 10001101_{ОК}$

1 - перенос

$$
\begin{array}{r}
 00001010 \\
+\ 10001101 \\
\hline
 10010111_{ОК}
\end{array}
= 11101000_{ПК} = -104_{10}
$$

Сложение в ОК

- Пример: -10 - 114

$-10_{10} = 10001010_{ПК} = 11110101_{ОК}$

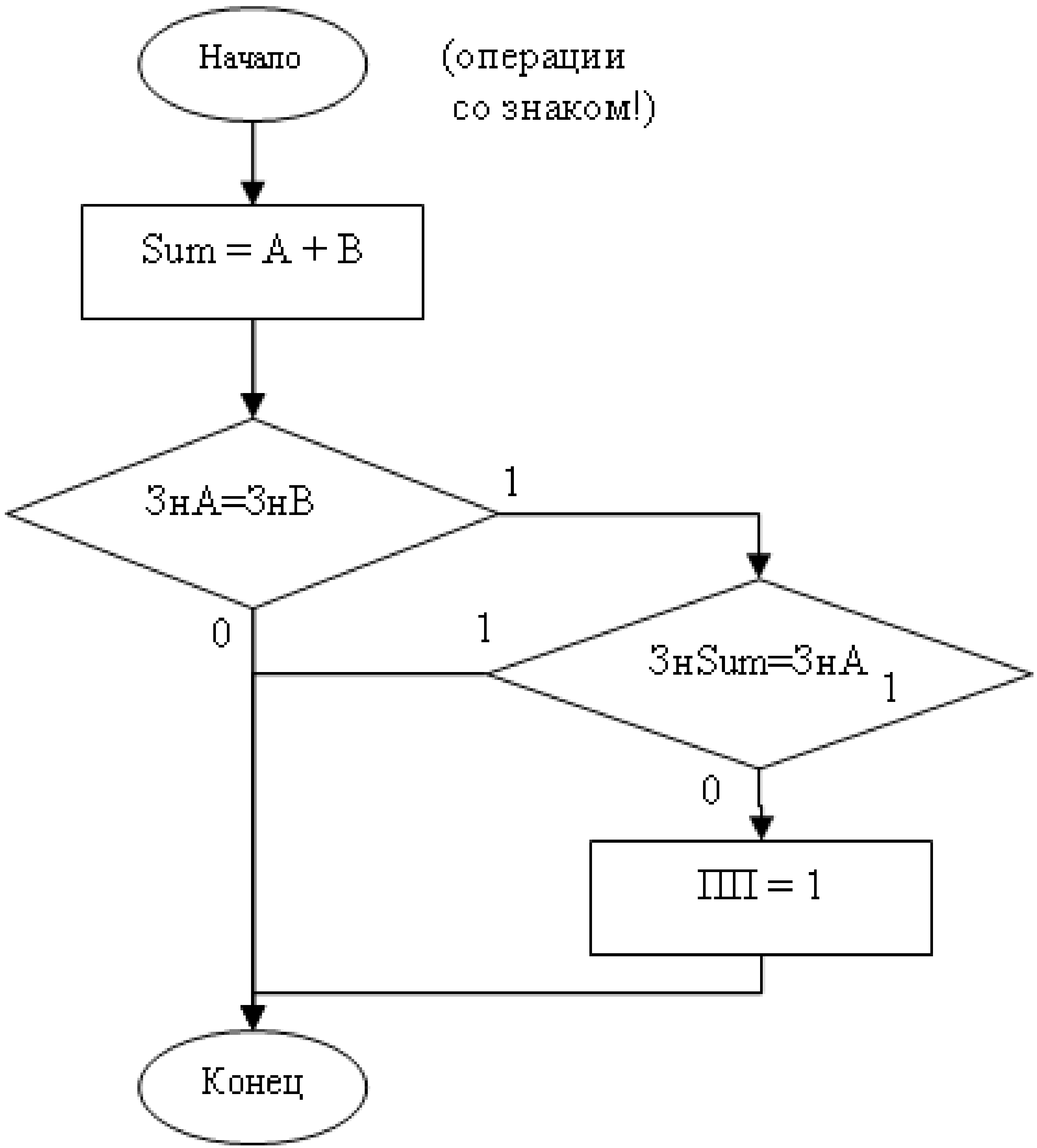
$-114_{10} = 11110010_{ПК} = 10001101_{ОК}$

```
111111 1          - перенос

  11110101
+
  10001101
  --------
  10000010
+
  00000001   - перенос из знак. разряда
  --------
  10000011
```

$10000011_{ОК} = 11111100_{ПК} = -124_{10}$

В ДК


Начало
(операции
со знаком!)
Sum = A + B
ЗнА=ЗнВ
1
0
1
ЗнSum=ЗнА
1
0
ПП = 1
Конец

Пример

- Вычислить (a – b) в ДК, a = -0.45_{10}, b = 0.37_{10}, длина разрядной сетки равна 6

0	45	0	37
0	9	0	74
1	8	1	48
1	6	0	96
1	2	1	92
0	4	1	84

$a = -\ 0.01110_2 = 101110_{ПК} = 110010_{ДК}$

$b = 0.01011_2 = 001011_{ДК}$

$-\ b = 110101_{ДК}$

Пример

$$\begin{array}{r} 11 \\ +\;110010 \\ 110101 \\ \hline 100111 \end{array}$$

$100111_{ДК} = 111001_{ПК} = -0.11001_2$

$(a-b) = -(2^{-1} + 2^{-2} + 2^{-5}) \approx -0.78$

$-0.45 - 0.37 = -0.82$

Примеры сложения чисел в ОК

A	B	$[A]_{ОК}$	$[B]_{ОК}$	Пример	$[S]_{ОК}$	S
Сложение положительных чисел						
+0.101	+0.001	0.101	0.001	0.101 (A) +0.001 (B) 0←0.110 (S) + 0 (перенос) 0.110 (S)	0.110	+0.110
Сложение отрицательных чисел						
-0.011	-0.001	1.100	1.110	1.100 (A) +1.110 (B) 1←1.010 (S) + 1 (перенос) 1.011 (S)	1.011	-0.100
Сложение чисел с переполнением						
+0.011	+0.101	0.011	0.101	0.011 (A) +0.101 (B) 0←1.000 (S) + 0 (перенос) 1.000 (переполн.)	1.000	-

Примеры вычитания чисел в ДК

A	B	$[A]_{ДК}$	$[B]_{ДК}$	Пример	$[S]_{ДК}$	S
Вычитание положительных чисел						
+0.101	+0.010	0.101	0.010	0.101 (A) +1.110 (-B) 0.011 (S)	0.011	+0.011
Вычитание отрицательных чисел						
-0.111	-0.011	1.001	1.101	1.001 (A) +0.011 (-B) 1.100 (S)	1.100	-0.100
Вычитание чисел с переполнением						
+0.011	-0.110	0.011	1.010	0.011 (A) +0.110 (-B) 1.001 (S) (переполн.)	-	-
-0.100	0.101	1.100	0.010	1.100 (A) +1.011 (-B) 0.111 (S) (переполн.)	-	-

3. Использование модифицированных ОК и ДК кодов

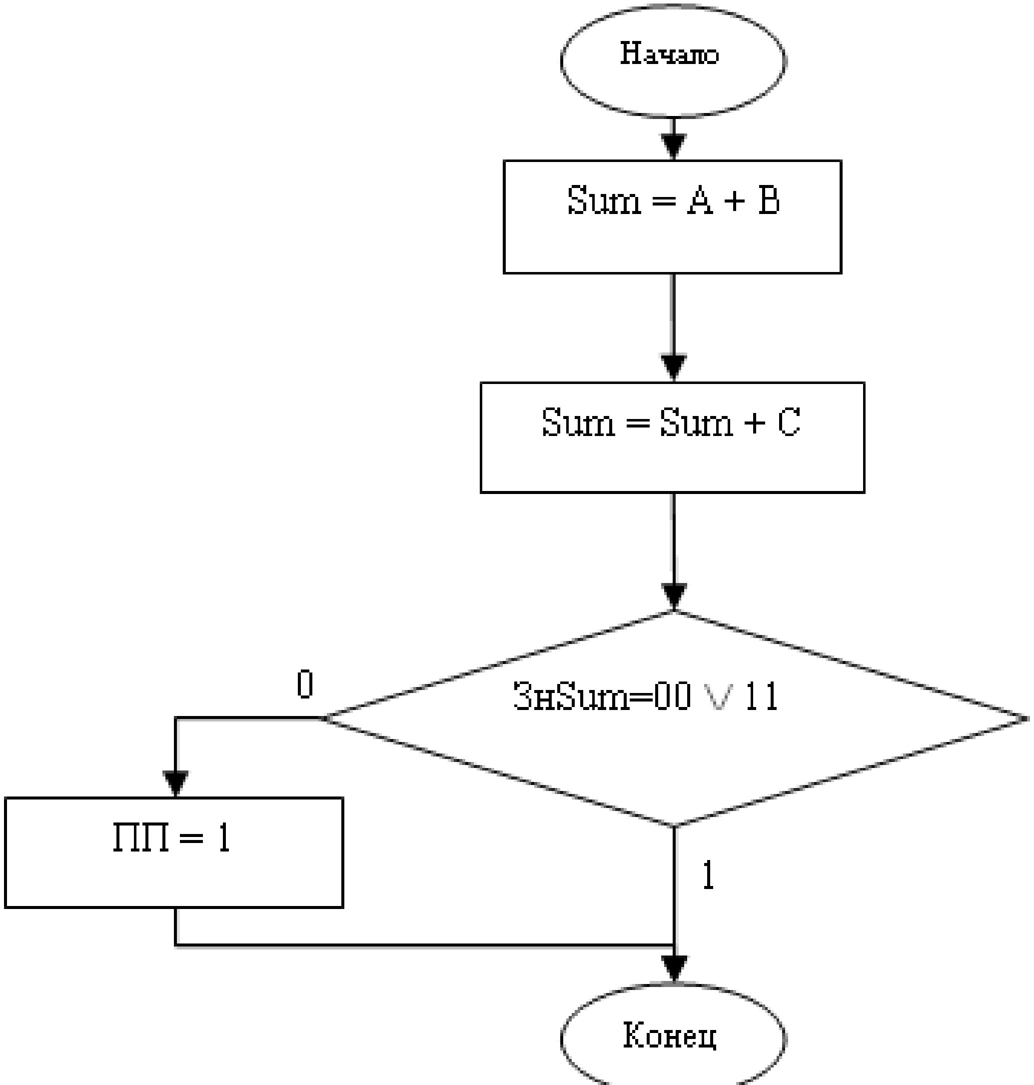
В некоторых случаях для представления и операциями над числами используются ***модифицированные ОК и ДК*** (***МОК*** и ***МДК***) с двумя знаковыми разрядами (00 – «+», 11 – «-»). При сложении чисел наличие дополнительного знакового разряда позволяет достаточно просто распознавать переполнение - комбинации цифр 01 и 10 в знаковых разрядах результата указывают на переполнение.

Алгоритмы в МОК и МДК ***ничем другим не отличаются*** от одноименных в ОК и ДК.

В МОК

$$Z = \begin{cases} 00.|Z|, Z \geq 0 \\ \mathbf{11}.|\overline{Z}|, Z < 0 \end{cases}$$

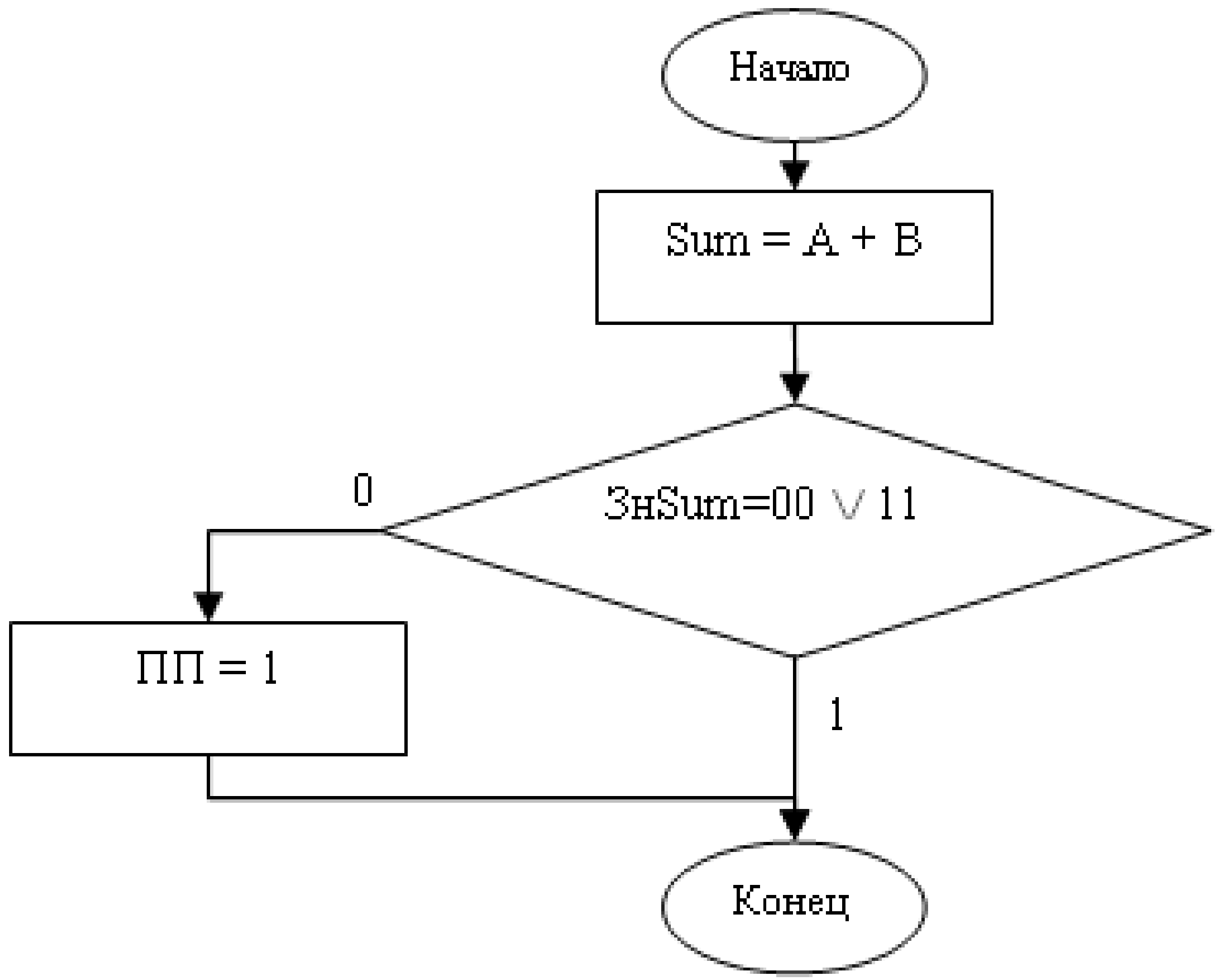
Sum = A + B



В МДК

$$Z = \begin{cases} 00.|Z|, Z \ge 0 \\ \mathbf{11}.|\overline{Z}| + \mathbf{1}, Z < 0 \end{cases}$$

$$Sum = A + B$$



Примеры вычитания чисел в модифицированном ОК

A	B	$[A]м_{ОК}$	$[B]м_{ОК}$	Пример	$[S]м_{ОК}$	S
Вычитание положительных чисел						
+0.101	+0.010	00.101	00.010	00.101 (A) + 11.101 (-B) 1←00.010 (S) + 1 (перенос) 00.011 (S)	00.011	+0.011
Вычитание отрицательных чисел						
-0.111	-0.011	11.000	11.100	11.000 (A) + 00.011 (-B) 0←11.011 (S) + 0 (перенос) 11.011 (S)	11.011	-0.100
Вычитание чисел с переполнением						
+0.011	-0.110	00.011	11.001	00.011 (A) +00.110 (-B) 0←01.001 (S) + 0 (перенос) 01.001 (S) (переполн.)	-	-

Примеры вычитания чисел в модифицированном ДК

A	B	$[A]_{мдк}$	$[B]_{мдк}$	Пример	$[S]_{мдк}$	S
Вычитание положительных чисел						
+0.101	+0.010	00.101	00.010	00.101 (A) + 11.110 (-B) 00.011 (S)	00.011	+0.011
Вычитание отрицательных чисел						
-0.111	-0.011	11.001	11.101	11.001(A) + 00.011 (-B) 11.100 (S)	11.100	-0.100
Вычитание чисел с переполнением						
+0.011	-0.110	00.011	11.010	00.011 (A) +00.110 (-B) 01.001 (S) (переполн.)	-	-
-0.100	0.101	11.100	00.101	11.100 (A) +11.011 (-B) 10.111 (S) (переполн.)	-	-