Celi brojevi i celobrojna aritmetika

Celi brojevi

U računaru se celi brojevi zapisuju u n-bitnoj reči u obliku binarnog broja $a_{n-1}a_{n-2}...a_1a_0$.

Binarna cifra	a_{n-1}	a_{n-2}			a_1	a_0
Pozicija	n-1	n-2	...	...	1	0

Slika 1: Zapis celog broja sa n binarnih cifara

Zapis neoznačenih brojeva

Broj čiji zapis ne sadrži znak se naziva *neoznačen*. Zapis neoznačenih celih brojeva je identičan njihovoj reprezentaciji u binarnom brojčanom sistemu.

Ako je A neoznačen ceo (dekadni) broj zapisan u obliku binarnog broja $a_{n-1}a_{n-2}...a_1a_0$ tada važi $A \in [0, 2^n - 1]$.

Ako se niz od n uzastopnih binarnih cifara $a_{n-1}a_{n-2}...a_1a_0$ interpretira kao neoznačeni ceo broj A, tada je njegova dekadna vrednost

$$A = \sum_{i=0}^{n-1} a_i 2^i$$

Binarni zapis	Dekadna vrednost
0000000000000000	0
0000000000000001	1
0000000000001111	15
0000000000100000	32
0001000000001111	4111
1000000000000000	32768
1111111111111111	65535
11111111111111111111111111111111	4294967295

Tabela 1: Primeri zapisa nekih neoznačenih brojeva

Zapis označenih brojeva

1. Zapis pomoću znaka i apsolutne vrednosti. U ovom zapisu cifra najveće težine označava znak broja, dok ostale cifre predstavljaju apsolutnu vrednost broja. Uobičajeno je da najmanja cifra brojčanog sistema označava pozitivne, a najveća negativne brojeve.

2. Zapis uz korišćenje *komplementa* broja. Komplement se takodje koristi i za konverziju iz pozitivnih u negativne brojeve i obrnuto. Neka je X_N pozitivan broj zapisan u brojčanom sistemu sa osnovom N pomoću n cifara, pri čemu je znak broja zapisan na mestu najveće težine pomoću najmanje cifre brojčanog sistema. Broj $-X_N$ se može zapisati pomoću:

   - $N-1$-og komplementa koji se dobija tako što se svaka cifra u zapisu broja X_N oduzme od $N-1$,
   - N-tog komplementa ili komplementa u odnosu na osnovu sistema (*radix* komplement) koji se dobija tako što se na zapis broja u $N-1$-om komplementu doda 1 na poziciju najmanje težine. *Radix* komplement broja X_N se može izračunati i direktno, bez odredjivanja $N-1$-og komplementa, kao razlika $N^n - X$. Pri tome se oduzimanje izvodi u brojčanom sistemu sa osnovom N.

3. Zapisom uz *dodavanje uvećanja* koji se dobija dodavanjem uvećanja k na N-ti komplement broja. Pojava eventualnih prenosa pri sabiranju se ignoriše. Ovaj način zapisa se još naziva i zapis u *kodu vi ak k*.

Primeri zapisa pozitivnih i negativnih brojeva u različitim brojčanim sistemima su prikazani u tabeli 2.

Broj	Znak i apsolutna vrednost	$N-1$-vi komplement	N-ti komplement	Uvećanje za $(6)_{10}$
$(+127)_{10}$	0127	0127	0127	0133
$(-127)_{10}$	9127	9872	9873	9879
$(+64)_{8}$	064	064	064	072
$(-64)_{8}$	764	713	714	722
$(+AB)_{16}$	0AB	0AB	0AB	0B1
$(-AB)_{16}$	FAB	F54	F55	F5B
$(+101)_{2}$	0101	0101	0101	1011
$(-101)_{2}$	1101	1010	1011	0001

Tabela 2: Zapis označenih celih brojeva u različitim brojčanim sistemima

Zapis označenih brojeva u računarskom sistemu

- Znak broja se predstavlja binarnom cifrom a_{n-1}
- Pravilo: $a_{n-1} = 0$ označava pozitivne, a $a_{n-1} = 1$ označava negativne brojeve.
- u binarnom sistemu n-ti komplement se naziva *potpuni* komplement (drugi komplement), a $n-1$-vi komplement se naziva *nepotpuni* komplement (prvi komplement),

Znak i apsolutna vrednost

U n-bitnoj reči krajnje levi bit označava znak, a ostalih $n-1$ bitova apsolutnu vrednost broja.

Vrednost broja A je

$$A_{ZA} = (-1)^{a_{n-1}} \sum_{i=0}^{n-2} 2^i a_i$$

Ako se niz od n uzastopnih binarnih cifara $a_{n-1}a_{n-2}...a_1a_0$ interpretira kao ceo broj A zapisan pomoću znaka i apsolutne vrednosti, tada je njegova dekadna vrednost $A \in [-2^{n-1}+1, 2^{n-1}-1]$.

Nepotpuni komplement

Broj A u nepotpunom komplementu se zapisuje na sledeći način:

- Krajnje levi bit označava znak broja
- Ostalih $n-1$ bitova se zapisuje:
    1. za pozitivne brojeve kao apsolutna vrednost broja, i
    2. za negativne brojeve kada se u zapisu apsolutne vrednosti broja A (bez znaka broja) svaka cifra zameni njenim komplementom do najveće cifre brojnog sistema.

Vrednost broja $A = a_{n-1}a_{n-2}...a_1a_0$ zapisanog u binarnom sistemu u nepotpunom komplementu je u dekadnom sistemu data pomoću zbira

$$A_{NK} = (-2^{n-1}+1)a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

Ako se niz od n uzastopnih binarnih cifara $a_{n-1}a_{n-2}...a_1a_0$ interpretira kao ceo broj A zapisan u nepotpunom komplementu, za njegovu dekadnu vrednost važi $A \in [-2^{n-1}+1, 2^{n-1}-1]$.

Potpuni komplement

- Krajnje levi bit u n-bitnoj reči označava znak broja, a ostalih $n-1$ bitova označavaju vrednost broja.
- Vrednost broja A se zapisuje na sledeći način:
  - za pozitivne brojeve kao apsolutna vrednost tog broja, i
  - za negativne brojeve kao broj koji se dobija kada se na zapis broja A (bez znaka broja) u nepotunom komplementu doda jedinica na mesto najmanje težine.

Vrednost broja A zapisanog u binarnom sistemu u potpunom komplementu je u dekadnom sistemu

$$A_{PK} = -2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

Ako se niz od n uzastopnih binarnih cifara $a_{n-1}a_{n-2}...a_1a_0$ interpretira kao ceo broj A zapisan u potpunom komplementu, tada za njegovu dekadnu vrednost važi $A \in [-2^{n-1}, 2^{n-1}-1]$.

- n=16
$$\begin{array}{rcl} -2^{15} & \leq x \leq & +2^{15}-1, \quad \text{odnosno} \\ -32768 & \leq x \leq & +32767 \end{array}$$
- n=32
$$\begin{array}{rcl} -2^{31} & \leq x \leq & +2^{31}-1, \quad \text{odnosno} \\ -2147483648 & \leq x \leq & +2147483647 \end{array}$$
- n=64
$$\begin{array}{rcl} -2^{63} & \leq x \leq & +2^{63}-1, \quad \text{odnosno} \\ -9223372036854775808 & \leq x \leq & +9223372036854775807 \end{array}$$

Prevodjenje izmedju zapisa celih brojeva u dekadnom sistemu i njihovog zapisa u potpunom komplementu – pojednostavljen način

1. Tabela sa vrednostima binarnih pozicija

7	6	5	4	3	2	1	0	binarna pozicija
-128	64	32	16	8	4	2	1	vrednost pozicije

2. Pri prevodjenju iz potpunog komplementa u dekadnu vrednost sabiraju se sve vrednosti koje odgovaraju pozicijama na kojima je 1.

-128	64	32	16	8	4	2	1	vrednost pozicije
1	0	1	1	0	1	0	1	binarni zapis broja
-128	0	+32	+16	0	+4	0	+1	$= -75_{10}$

3. Pri prevodjenje iz dekadnog zapisa u potpuni komplement brojevi se zapisuju kao zbir vrednosti pozicija u binarnom zapisu.

-128	64	32	16	8	4	2	1	vrednost pozicije
0	+64	0	0	+8	0	+2	+1	$= +75_{10}$
0	1	0	0	1	0	1	1	binarni zapis broja

Zapis uz dodavanje uvećanja

U ovom zapisu se broj predstavlja kao zbir njegovog potpunog komplementa i vrednosti *k* koja je poznata pod nazivom *uvećanje* ili *vi ak*.

Na primer, zapis brojeva $(+12)_{10}$ i $(-12)_{10}$ u 8 bita se pomoću uvećanja 128 formira tako što se sabere 128 sa originalnim brojem i dobijeni zbir se zapiše kao neoznačen ceo broj. Tako je

$(128)_{10}+(12)_{10}=(140)_{10} \longrightarrow (10001100)_2$
$(128)_{10}+(-12)_{10}=(116)_{10} \longrightarrow (01110100)_2$

- Vrednost uvećanja nema numerički značaj, već je njena jedina funkcija pomeranje reprezentacije broja u potpunom komplementu.
- Vrednost uvećanja se obično bira tako da ima istu masku bitova kao i najmanji negativan broj. Tada su brojevi u ovom zapisu sortirani, ako se posmatraju kao neoznačeni celi brojevi.

Dekadna vrednost	Znak i apsolutna vrednost	Nepotpuni komplment	Potpuni komplement	Uvećanje 128
+127	01111111	01111111	01111111	11111111
+64	01000000	01000000	01000000	11000000
+32	00100000	00100000	00100000	10100000
+16	00010000	00010000	00010000	10010000
+15	00001111	00001111	00001111	10001111
+10	00001010	00001010	00001010	10001010
+9	00001001	00001001	00001001	10001001
+8	00001000	00001000	00001000	10001000
+7	00000111	00000111	00000111	10000111
+6	00000110	00000110	00000110	10000110
+5	00000101	00000101	00000101	10000101
+4	00000100	00000100	00000100	10000100
+3	00000011	00000011	00000011	10000011
+2	00000010	00000010	00000010	10000010
+1	00000001	00000001	00000001	10000001
+0	00000000	00000000	00000000	10000000
-0	10000000	11111111	---	---
-1	10000001	11111110	11111111	01111111
-2	10000010	11111101	11111110	01111110
-3	10000011	11111100	11111101	01111101
-4	10000100	11111011	11111100	01111100
-5	10000101	11111010	11111011	01111011
-6	10000110	11111001	11111010	01111010
-7	10000111	11111000	11111001	01111001
-8	10001000	11110111	11111000	01111000
-9	10001001	11110110	11110111	01110111
-10	10001010	11110101	11110110	01110110
-15	10001111	11110000	11110001	01110001
-16	10010000	11101111	11110000	01110000
-32	10100000	11011111	11100000	01100000
-64	11000000	10111111	11000000	01000000
-127	11111111	10000000	10000001	00000001
-128	---	---	10000000	00000000

Tabela 3: Zapis označenih celih brojeva u obliku znaka i apsolutne vrednosti, nepotpunog komplementa, potpunog komplementa i uvećanja od 128 u binarnoj reči dužine 8

Konverzija izmedju zapisa različitih dužina

Neka je ceo broj $A = a_{n-1}a_{n-2}...a_1a_0$ zapisan u binarnoj reči dužine n i neka ga treba upisati u binarnu reč dužine m.

- Ako je $m < n$ upisivanje nije moguće izvesti korektno zbog mogućeg gubitka značajnih cifara.
- Ako je $m = n$ konverzija ne postoji.
- Ako je $m > n$ tada način konverzije zavisi od načina zapisa celog broja.

1. Znak i apsolutna vrednost: bit za znak se pomeri na mesto najveće težine i ostala mesta se popune nulama. Na primer, za $n = 8, m = 16$

Dekadna vrednost	8-bitna reč	16-bitna reč	Zapis
+127	01111111	0000000001111111	znak i aps. vred.
+5	00000101	0000000000000101	znak i aps. vred.
+0	00000000	0000000000000000	znak i aps. vred.
-0	10000000	1000000000000000	znak i aps. vred.
-5	10000101	1000000000000101	znak i aps. vred.
-127	11111111	1000000001111111	znak i aps. vred.

2. Nepotpuni i potpuni komplement: upisuje se a_{n-1} na sve pozicije i u zapisu broja, gde važi $n \leq i < m$. Na primer, za $n = 8, m = 16$

Dekadna vrednost	8-bitna reč	16-bitna reč	Zapis
+5	00000101	0000000000000101	nepotpuni komp.
-5	11111010	1111111111111010	nepotpuni komp.
+9	00001001	0000000000001001	potpuni kompl.
-9	11110111	1111111111110111	potpuni kompl.

Dokaz korektnosti pravila sa operacije u potpunom komplementu
(slično važi i za nepotpuni komplement uz dodatni korak za eliminaciju jedinica)

Neka je $A = a_{n-1}a_{n-2}...a_1a_0$ broj zapisan u potpunom komplementu. Tada važi

$$A = -2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

Neka je $A = a_{m-1}a_{m-2}...a_1a_0$. zapisan pomoću m-bitne reprezentacije. Tada takodje važi

$$A = -2^{m-1}a_{m-1} + \sum_{i=0}^{m-2} 2^i a_i$$

Ove dve dobijene vrednosti treba da budu jednake.

1. Ako je A pozitivan broj jednakost je očigledna jer je vrednost znaka broja = 0 pa su i sve vrednosti a_i koje su identične znaku takodje 0.

2. Ako je A nula jednakost je očigledna.

3. Ako je A negativan broj:

$$\begin{aligned}
-2^{m-1}a_{m-1} + \sum_{i=0}^{m-2} 2^i a_i &= -2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i \\
-2^{m-1} + \sum_{i=0}^{m-2} 2^i a_i &= -2^{n-1} + \sum_{i=0}^{n-2} 2^i a_i \\
-2^{m-1} + \sum_{i=n-1}^{m-2} 2^i a_i &= -2^{n-1} \\
2^{n-1} + \sum_{i=n-1}^{m-2} 2^i a_i &= 2^{m-1} \\
1 + \sum_{i=0}^{n-2} 2^i + \sum_{i=n-1}^{m-2} 2^i a_i &= 1 + \sum_{i=0}^{m-2} 2^i \\
\sum_{i=n-1}^{m-2} 2^i a_i &= \sum_{i=n-1}^{m-2} 2^i
\end{aligned}$$

Ova jednakost važi jer su ispunjeni uslovi:

- $n-1$ bitova manje težine nije promenilo vrednost pri konverziji;
- Svi bitovi izmedju $n-1$ i $m-2$ pozicije treba da budu jedinice, odnosno jednake znaku, što važi jer je $a_{m-1}=a_{n-1}=1$.

Celobrojna aritmetika

Promena znaka

- Znaka i apsolutna vrednost – komplementira se bit za znak broja

Dekadna vrednost	Binarni zapis
+9	00001001
+5	00000101
-5	10000101
-9	10001001

- Nepotpuni komplement – komplementiranje svake cifre u binarnom zapisu broja, uključujući i mesto za znak.

Dekadna vrednost	Binarni zapis
+9	00001001
+5	00000101
-5	11111010
-9	11110110

- U zapisu pomoću potpunog komplementa promena znaka broja se vrši u dva koraka:
  1. U prvom se izvrši komplementiranje svake cifre do najveće cifre brojnog sistema (u binarnom sistemu do jedinice), uključujući i mesto za znak.
  2. U drugom se dobijeni broj sabere sa jedinicom, pri čemu se sabiranje obavlja po pravilima za sabiranje neoznačenih brojeva.

+9	=	00001001	potpuni komplement
		11110110	1. korak
		+00000001	2. korak
-9		11110111	rezultat
-5	=	11111011	potpuni komplement
		00000100	1. korak
		+00000001	2. korak
+5	=	00000101	rezultat
0	=	00000000	potpuni komplement
		11111111	1. korak
		+00000001	2. korak
0	=	1|00000000	rezultat, pri čemu se prekoračenje ignoriše

-128	=	10000000	potpuni komplement
		01111111	1. korak
		+00000001	2. korak
-128	=	10000000	rezultat koji predstavlja isti broj

Dokaz korektnosti ovakvog načina zapisa promene znaka

Neka je $A = a_{m-1}a_{m-2}...a_1a_0$ broj predstavljen u potpunom komplementu. Njegova vrednost je

$$A = -2^{m-1}a_{m-1} + \sum_{i=0}^{m-2} 2^i a_i$$

Po prethodnim pravilima za promenu znaka ($\overline{a_i}$ označava komplement od a_i):

$$B = -A = -2^{m-1}\overline{a_{m-1}} + \sum_{i=0}^{m-2} 2^i \overline{a_i} + 1$$

Korišćenjem činjenice $a_i + \overline{a_i} = 1$ dokazuje se da je $A + B = 0$:

$$\begin{aligned}
A + B &= -2^{m-1}a_{m-1} + \sum_{i=0}^{m-2} 2^i a_i + 1 - 2^{m-1}\overline{a_{m-1}} + \sum_{i=0}^{m-2} 2^i \overline{a_i} \\
&= 1 - 2^{m-1}(a_{m-1} + \overline{a_{m-1}}) + \sum_{i=0}^{m-2} 2^i (a_i + \overline{a_i}) \\
&= 1 - 2^{m-1} + \sum_{i=0}^{m-2} 2^i \\
&= 1 - 2^{m-1} + 2^{m-1} - 1 \\
&= 0
\end{aligned}$$

Sabiranje i oduzimanje

Prekoračenje

Ako se kao rezultat operacije sabiranja brojeva A i B koji su zapisani sa po n cifara dobije broj C za čiji je tačan zapis potrebna $n+1$ cifra tada se kaže da je došlo do *prekoračenja* pri izvodjenju operacije.

U računaru se prekoračenje otkriva upotrebom *modifikovanog oblika broja*.

$$A = \boxed{a_n} a_{n-1}a_{n-2}...a_1a_0$$
$$B = \boxed{b_n} b_{n-1}b_{n-2}...b_1b_0$$
$$C = \boxed{c_n} c_{n-1}c_{n-2}...c_1b_0$$

Pravilo za otkrivanje prekoračnja: ako se sabiraju dva broja istog znaka (bilo oni pozitivni ili negativni) prekoračenje se javlja ako i samo ako rezultat sabiranja ima suprotan znak.

Sabiranje i oduzimanje neoznačenih brojeva

Vrši se u skladu sa pravilima za sabiranje i oduzimanje u binarnom sistemu.

Primeri:

a) 14 + 10

14	=	00001110
10	=	00001010
24	=	00011000

b) 252 + 5

252	=	$\boxed{0}$ 11111100
5	=	$\boxed{0}$ 00000101
***	=	$\boxed{1}$ 00000001

Sabiranje i oduzimanje brojeva zapisanih u obliku znak i apsolutna vrednost

Pravila za odredjivanje zbira brojeva A i B:

1. Ukoliko su brojevi A i B istog znaka, isti znak ima i rezultat sabiranja. Apsolutna vrednost zbira se dobija sabiranjem apsolutnih vrednosti sabiraka kao neoznačenih brojeva. Ako se u tom sabiranju javi prekoračenje, tada se prekoračenje javlja i u konačnom zbiru.

2. Ukoliko su brojevi A i B različitog znaka, znak rezultata je isti kao i znak sabirka koji ima veću apsolutnu vrednost. Apsolutna vrednost zbira je razlika apsolutnih vrednosti sabiraka pri čemu se oduzima manja apsolutna vrednost od veće.

3. Oduzimanje $A - B$ se svodi na sabiranje uz promenu znaka drugom operandu.

Primeri:

a) +14 + 10

+14	=	0|0001110
+10	=	0|0001010
24	=	0|0011000

b) +127 + 3

+127	=	$\boxed{0}$	0|1111111
+3	=	$\boxed{0}$	0|0000011
***	=	$\boxed{1}$	0|0000010

Sabiranje i oduzimanje brojeva u nepotpunom komplementu

- Neka su brojevi $A = a_{n-1}a_{n-2}...a_1a_0$ i $B = b_{n-1}b_{n-2}...b_1b_0$ zapisani u nepotpunom komplementu u reči dužine n.

- Izračunavanje njihovog zbira se vrši u dva koraka:

  1. Označimo medjurezultat koji se dobija sabiranjem A i B sa C':

  $$\begin{array}{rcccccc} A = & & a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \\ B = & & b_{n-1} & b_{n-2} & \dots & b_1 & b_0 \\ \hline C' = & c'_n & c'_{n-1} & c'_{n-2} & \dots & c'_1 & c'_0 \end{array}$$

     Sabiranje se vrši kao sabiranje neoznačenih brojeva bez kontrole prekoračenja; c'_n predstavlja prenos pri sabiranju sa pozicije za znak.

  2. Neka je C'' broj koji se dobija uklanjanjem c'_n iz medjurezultata C'. Konačan rezultat $C = A + B$ se dobija tako što se prenos c'_n sabere sa C'' uz kontrolu prekoračenja.

  $$\begin{array}{ccr} C'' & = & c'_{n-1}c'_{n-2}...c'_1c'_0 \\ & & c'_n \\ \hline C & = & c_{n-1}c_{n-2}...c_1c_0 \end{array}$$

- Oduzimanje $C = A - B$ se svodi na sabiranje uz promenu znaka drugom operandu: $C = A + (-B)$.

Primeri:

a) `+14 + 10`

I korak:

```
A=+14    =    00001110
B=+10    =    00001010
--------------------------
C'       =  0|00011000
```

II korak:

```
C''      =  00011000
                   0
--------------------------
C=+24    =  00011000
```

b) `+127-(+10) = +127+(-10)`

I korak:

```
A=+127   =    01111111
B=-10    =    11110101
--------------------------
C'       =  1|01110100
```

II korak:

```
C''      =    01110100
                     1
--------------------------
C=+117   =    01110101
```

c) `+100 + 65`

I korak:

```
A=+100   =    01100100
B=+65    =    01000001
--------------------------
C'       =  0|10100101
```

II korak:

```
C''      =    10100101
                     0
--------------------------
C=***    =    10100101
```

Prekoračenje jer se sabiranjem dva pozitivna dobija negativan broj.

d) `-100-(+65) = -100+(-65)`

I korak:

```
A=-100   =    10011011
B=-65    =    10111110
--------------------------
C'       =  1|01011001
```

II korak:

```
C''      =    01011001
                     1
--------------------------
C=***    =    01011010
```

Prekoračenje jer se sabiranjem dva negativna dobija pozitivan broj.

Sabiranje i oduzimanje brojeva u potpunom komplementu

Pravilo:

Neka su brojevi $A = a_{n-1}a_{n-2}...a_1a_0$ i $B = b_{n-1}b_{n-2}...b_1b_0$ zapisani u potpunom komplementu u reči dužine n.

1. Izračunavanje njihovog zbira se vrši u dva koraka:

   (a) Označimo medjurezultat koji se dobija sabiranjem A i B sa C':

   $$\begin{array}{r|cccccc} A = & & a_{n-1} & a_{n-2} & \ldots & a_1 & a_0 \\ B = & & b_{n-1} & b_{n-2} & \ldots & b_1 & b_0 \\ \hline C' = & c'_n & c'_{n-1} & c'_{n-2} & \ldots & c'_1 & c'_0 \end{array}$$

   Sabiranje se vrši kao sabiranje neoznačenih brojeva bez kontrole prekoračenja; c'_n predstavlja prenos pri sabiranju sa pozicije za znak.

   (b) Konačan rezultat $C = A + B$ se dobija uklanjanjem c'_n iz medjurezultata C' i proverom pojave prekoračenja.

2. Oduzimanje $C = A - B$ se svodi na sabiranje uz promenu znaka drugom operandu: $C = A + (-B)$.

Primeri:

a) `+14 + 10`

A=+14	=	00001110
B=+10	=	00001010
C'	=	0|00011000
C=+24	=	00011000

b) `+100 + 65`

A=+100	=	01100100
B=+65	=	01000001
C'=	=	0|10100101
C=***	=	10100101

Prekoračenje jer se sabiranjem dva pozitivna dobija negativan broj.

c) `+127 -10`

A=+127	=	01111111
B=-10	=	11110110
C'=	=	1|01110101
C=+117	=	01110101

d) `-100-(+65)=-100+(-65)`

A=-100	=	10011100
B=-65	=	10111111
C'=	=	1|01011011
C=***	=	01011011

Prekoračenje jer se sabiranjem dva negativna dobija pozitivan broj.

Množenje

Množenje neoznačenih brojeva

```
        00001110 x 00001001        |   14 × 9 (činioci)
        00001110                   |
       00000000                    |
      00000000                     |  Delimični
     00001110                      |
    00000000                       |  proizvodi
   00000000                        |
  00000000                         |
 00000000                          |
0000000001111110                   |  = 126 (rezultat)
```

Slika 2: Množenje neoznačenih binarnih brojeva

0000000000001110 $\times$ 00001001	
0000000000001110	$0000000000001110 \times 1 \times 2^0$
0000000000000000	$0000000000001110 \times 0 \times 2^1$
0000000000000000	$0000000000001110 \times 0 \times 2^2$
0000000001110000	$0000000000001110 \times 1 \times 2^3$
0000000000000000	$0000000000001110 \times 0 \times 2^4$
0000000000000000	$0000000000001110 \times 0 \times 2^5$
0000000000000000	$0000000000001110 \times 0 \times 2^6$
0000000000000000	$0000000000001110 \times 0 \times 2^7$
0000000001111110	

Moguća poboljšanja:

1. Formira se medjuzbir umesto čuvanja svih delimičnih proizvoda
2. Izračunavaju se samo oni medjuzbirovi koji odgovaraju binarnim jedinicama množioca.

Hardverski se ovaj algoritam implementira preko serijskog množioca koji koristi tri registra A, M i P, kao i jednobitni registar C koji sadrži prenos pri sabiranju. Algoritam se može opisati na sledeći način:

1. Na početku množenja se množenik upisuje u registar M, množilac u registar P, dok se u registre A i C upisuje 0.
2. U svakom koraku množenja bit množioca na mestu najmanje težine odredjuje da li će u tom koraku množenik biti sabran sa tekućom vrednošću proizvoda:
   (a) Ako je vrednost bita 1 sabiranje se vrši.
   (b) Ako je vrednost bita 0 ne vrši se nikakva akcija.
3. Vrši se logičko pomeranje udesno sadržaja registara C, A i P, pri čemu se sva tri posmatraju kao jedinstven registar.
4. Korak 3 se ponavlja u ciklusu sve dok se ne obrade svi bitovi u množiocu.
5. Vrednost proizvoda je upisana u registrima A i P, posmatranim kao jedan registar.

M	C	A	P	Komentar	
00001110	0	00000000	00001001	Početno stanje	M=14, P=9, C=0, A=0
00001110	0	00001110	00001001	$A = A + M$	Prvi ciklus
00001110	0	00000111	00000100	Pomeranje udesno	
00001110	0	00000111	00000100	Bez akcije	Drugi ciklus
00001110	0	00000011	10000010	Pomeranje udesno	
00001110	0	00000011	10000010	Bez akcije	Treći ciklus
00001110	0	00000001	11000001	Pomeranje udesno	
00001110	0	00001111	11000001	$A = A + M$	Četvrti ciklus
00001110	0	00000111	11100000	Pomeranje udesno	
00001110	0	00000111	11100000	Bez akcije	Peti ciklus
00001110	0	00000011	11110000	Pomeranje udesno	
00001110	0	00000011	11110000	Bez akcije	Šesti ciklus
00001110	0	00000001	11111000	Pomeranje udesno	
00001110	0	00000001	11111000	Bez akcije	Sedmi ciklus
00001110	0	00000000	11111100	Pomeranje udesno	
00001110	0	00000000	11111100	Bez akcije	Osmi ciklus
00001110	0	00000000	01111110	Pomeranje udesno	
		00000000	01111110	Rezultat	14×9 = 126

Tabela 4: Implementacija hardverskog množenja neoznačenih brojeva

Množenje brojeva u potpunom komplementu

```
A. 14×9 = 126
00001110 × 00001001
        00001110
     00001110
0000000001111110
```

```
B. -14×9 = +2178
11110010 × 00001001
        11110010
     11110010
0000100010000010
```

```
C. 14×(-9) = +3458
00001110 × 11110111
        00001110
       00001110
      00001110
    00001110
   00001110
  00001110
 00001110
0000110110000010
```

```
D. -14×(-9)= -27006
11110010 × 11110111
        11110010
       11110010
      11110010
    11110010
   11110010
  11110010
 11110010
1110100101111110
```

Slika 3: Nekorektan način množenja brojeva u potpunom komplementu

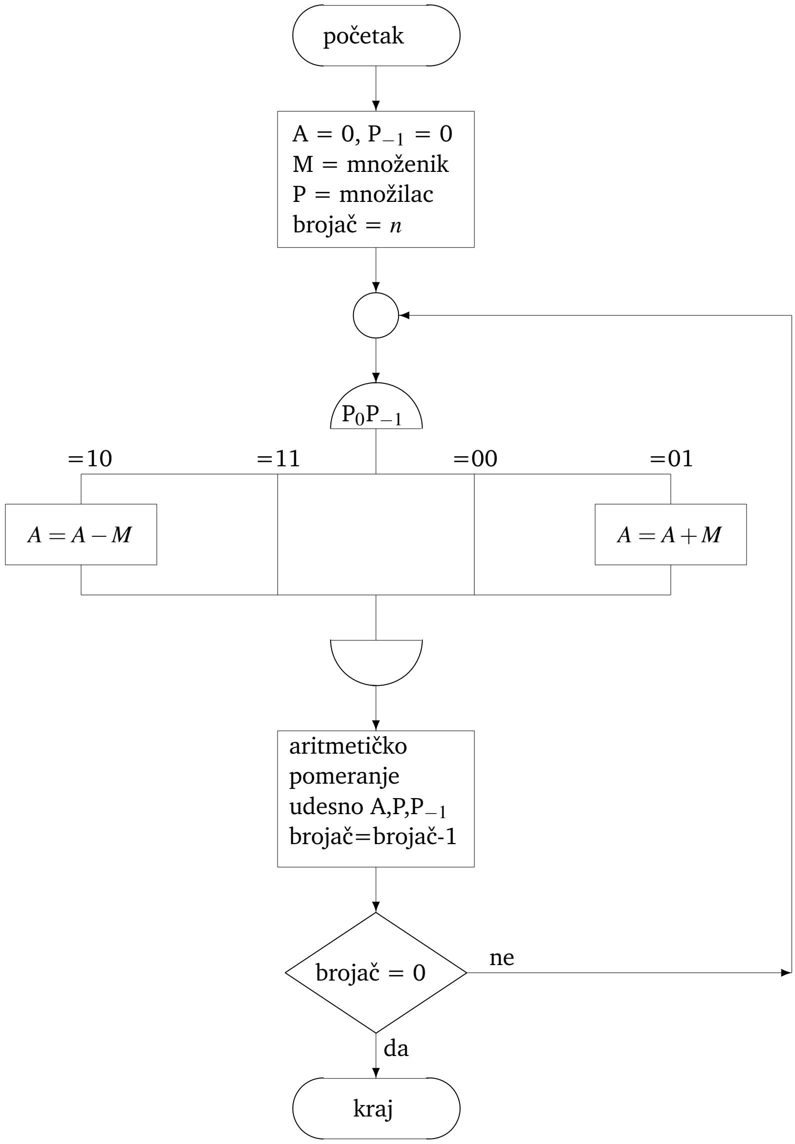
Kada je množenik negativan korektan rezultat se dobija proširivanjem delimičnih proizvoda do dužine reči u koju se upisuje proizvod:

```
11110010 × 00001001   -14×9
1111111111110010
1111111110010000
1111111110000010     = -126
```

Ovaj postupak ne daje korektan rezultat ako je množilac negativan:

```
00001001 × 11110010   9×-14
0000000000000000
0000000000010010
0000000000000000
0000000000000000
0000000010010000
0000000100100000
0000001001000000
0000010010000000
0000100010000010     = +2178
```

Zbog toga se množenje celih brojeva u potpunom komplementu najčešće vrši prema Butovom algoritmu.



Slika 4: Booth-ov algoritam za množenje brojeva u potpunom komplementu

U množenju se koriste četiri registra A, M, P i P_{-1} pri čemu P_{-1} sadrži samo jedan bit. Rad Butovog algoritma se odvija na sledeći način:

1. Moženik i množilac se upisuju u registre M i P, a u registre A i P_{-1} se upisuju 0. Takodje, postavlja se vrednost brojača koji odredjuje broj ponavljanja koraka u telu algoritma na n koje odgovara dužini registara u kojima se nalaze množenik i množilac.

2. U svakom koraku se porede vrednosti u bitu najmanje težine P_0 registra P i uvedenog dodatka P_{-1}:

   - Ako se bitovi razlikuju, i kombinacija je ’01’ tada se množenik sabere sa sadržajem registra A.
   - Ako se bitovi razlikuju, i kombinacija je ’10’ tada se množenik oduzme od sadržaja registru A.
   - Ako su te dve vrednosti jednake (’11’ ili ’00’) ne vrši se nikakva akcija.

   Posle svakog od ova tri slučaja vrši se aritmetičko pomeranje udesno za jednu poziciju sadržaja A, P i P_{-1} pri čemu se oni posmatraju kao jedan registar.

3. Prethodni korak se ponavlja u ciklusu sve dok ne budu obradjeni svi bitovi u množiocu.

4. Rezultat množenja je upisan u registre A i P, posmatrane kao jedna reč.

M	A	P	P_{-1}	Komentar	
00001110	00000000	00001001	0	Početno stanje	M=14, P=9
00001110	11110010	00001001	0	$A = A - M$	Prvi ciklus
00001110	11111001	00000100	1	Pomeranje udesno	
00001110	00000111	00000100	1	$A = A + M$	Drugi ciklus
00001110	00000011	10000010	0	Pomeranje udesno	
00001110	00000011	10000010	0	Bez akcije	Treći ciklus
00001110	00000001	11000001	0	Pomeranje udesno	
00001110	11110011	11000001	0	$A = A - M$	Četvrti ciklus
00001110	11111001	11100000	1	Pomeranje udesno	
00001110	00000111	11100000	1	$A = A + M$	Peti ciklus
00001110	00000011	11110000	0	Pomeranje udesno	
00001110	00000011	11110000	0	Bez akcije	Šesti ciklus
00001110	00000001	11111000	0	Pomeranje udesno	
00001110	00000001	11111000	0	Bez akcije	Sedmi ciklus
00001110	00000000	11111100	0	Pomeranje udesno	
00001110	00000000	11111100	0	Bez akcije	Osmi ciklus
00001110	00000000	01111110	0	Pomeranje udesno	
		01111110		Rezultat	14×9 = 126

M	A	P	P_{-1}	Komentar	
11110010	00000000	11110111	0	Početno stanje	M=-14, P=-9
11110010	00001110	11110111	0	$A = A - M$	Prvi ciklus
11110010	00000111	01111011	1	Pomeranje udesno	
11110010	00000111	01111011	1	Bez akcije	Drugi ciklus
11110010	00000011	10111101	1	Pomeranje udesno	
11110010	00000011	10111101	1	Bez akcije	Treći ciklus
11110010	00000001	11011110	1	Pomeranje udesno	
11110010	11110011	11011110	1	$A = A + M$	Četvrti ciklus
11110010	11111001	11101111	0	Pomeranje udesno	
11110010	00000111	11101111	0	$A = A - M$	Peti ciklus
11110010	00000011	11110111	1	Pomeranje udesno	
11110010	00000011	11110111	1	Bez akcije	Šesti ciklus
11110010	00000001	11111011	1	Pomeranje udesno	
11110010	00000001	11111011	1	Bez akcije	Sedmi ciklus
11110010	00000000	11111101	1	Pomeranje udesno	
11110010	00000000	11111101	1	Bez akcije	Osmi ciklus
11110010	00000000	01111110	1	Pomeranje udesno	
		01111110	1	Rezultat	-14×-9 = 126

Tabela 5: Butov algoritam primenjen na množenje brojeva 14×9 i -14×-9

Optimizacija Butovog algoritma

Butov algoritam omogućuje optimizaciju procesa izračunavanja koja se sastoji u primeni samo jednog sabiranja i oduzimanja za svaki niz uzastopnih 1.

- Neka je množilac A pozitivan i neka se sastoji od niza uzastopnih jedinica pre i posle koga se nalaze binarne nule. Broj operacija može da se smanji na dve korišćenjen jednakosti

$$2^n + 2^{n-1} + \ldots + 2^{n-k} = 2^{n+1} - 2^{n-k} \qquad (1)$$

  Na primer, Neka je npr. $A = 62$, i neka je sa M označen množenik. Bez korišćenja optimizacije proizvod se izračunava kao

$$\begin{aligned} M * (00111110) &= M * (2^5 + 2^4 + 2^3 + 2^2 + 2^1) \\ &= M * (32 + 16 + 8 + 4 + 2) \\ &= M * 62 \end{aligned}$$

  Na osnovu jednakosti 1 važi

$$\begin{aligned} M * (00111110) &= M * (2^6 - 2^1) \\ &= M * (64 - 2) \\ &= M * 62 \end{aligned}$$

  Odavde sledi da se proizvod bilo kog broja sa 62 može izračunati pomoću jednog sabiranja množenika sa sadržajem registra A i jednog oduzimanja množenika od sadržaja registra A. Ovaj postupak se može proširiti na bilo koji broj uzastopnih nizova jedinica u množiocu, pri čemu se i pojavljivanje samo jedne jedinice smatra pojavljivanjem niza. Tako se npr. množenje sa $+118$ može realizovati kao

$$\begin{aligned} M * (01110110) &= M * (2^6 + 2^5 + 2^4 + 2^2 + 2^1) \\ &= M * (2^7 - 2^4 + 2^3 - 2^1) \end{aligned}$$

  Butov algoritam u skladu sa ovom šemom, predvidja oduzimanje množenika kad god se pojavi prva jednica u novom nizu (slučaj pojave '10' u P_0P_{-1}) i sabiranja pri dolasku na kraj takvog niza (slučaj '01' u P_0P_{-1}).

- Neka je množilac A negativan broj. U skladu sa pravilima za zapis u potpunom komplementu zapis ovog broja počinje sa 1. Neka se nula prvi put pojavljuje sleva na k-toj poziciji:

$$A = 1111...10a_{k-1}a_{k-2}...a_1a_0 \tag{2}$$

Vrednost broja A je

$$A = -2^{n-1} + 2^{n-2} + ... + 2^{k+1} + a_{k-1}2^{k-1} + ... + a_0 2^0 \tag{3}$$

Na osnovu jednakosti 1 važi

$$2^{n-2} + 2^{n-3} + ... + 2^{k+1} = 2^{n-1} - 2^{k+1}$$

odnosno

$$-2^{n-1} + 2^{n-2} + 2^{n-3} + ... + 2^{k+1} = -2^{k+1}$$

Zamenjujući ovu jednakost u jednakosti 3 kojom je predstavljena vrednost broja A dobija se

$$A = -2^{k+1} + a_{k-1}2^{k-1} + ... + a_0 2^0 \tag{4}$$

Uzimajući u obzir vrednost broja prikazanu u jednakosti 2 jasno je da je za sve bitove počevši od a_0 do a_k (koji je jednak 0) optimizacija Butovog algoritma korektna, jer se dobijaju svi sabirci u jednakosti 4 sem vodeće vrednosti -2^{k+1}. Vodeća vrednost se dobija jer se pri dolasku do naredne jedinice (na mestu $k+1$), u skladu sa Butovim algoritmom, vrši se oduzimanje -2^{k+1} (jer se javlja kombinacija ’10’) čime se dobija ista vrednost kao u jednakosti 4.

Kao ilustracija neka posluži množenje sa -62 koje se može realizovati kao

$$M * (11000010) \quad = \quad M * (-2^7 + 2^6 + 2^1)$$

Prema jednakosti 4

$$M * (11000010) \quad = \quad M * (-2^6 + 2^1)$$

Direktnom primenom optimizovanog Butovog algoritma (oduzimanje pri pojavi kombinacije ’10’, sabiranje pri pojavi kombinacije ’01’) dobija se ista vrednost:

$$\begin{aligned} M * (11000010) &= M * (-2^6 + 2^2 - 2^1) \\ &= M * (-2^6 + 2^1) \end{aligned}$$

Modifikovani Butov algoritam

Postoje slučajevi kada ni osnovni ni optimizovan iButov algoritam ne daju najbolje rešenje. Na primer, sledeće množenje se realizuje preko 8 a potrebno je samo 4 sabiranja:

M	A	P	P_{-1}	Komentar	
00001110	00000000	01010101	0	Početno stanje	M=14, P=85
00001110	11110010	01010101	0	$A = A - M$	Prvi ciklus
00001110	11111001	00101010	1	Pomeranje udesno	
00001110	00000111	00101010	1	$A = A + M$	Drugi ciklus
00001110	00000011	10010101	0	Pomeranje udesno	
00001110	11110101	10010101	0	$A = A - M$	Treći ciklus
00001110	11111010	11001010	1	Pomeranje udesno	
00001110	00001000	11001010	1	$A = A + M$	Četvrti ciklus
00001110	00000100	01100101	0	Pomeranje udesno	
00001110	11110110	01100101	0	$A = A - M$	Peti ciklus
00001110	11111011	00110010	1	Pomeranje udesno	
00001110	00001001	00110010	1	$A = A + M$	Šesti ciklus
00001110	00000100	10011001	0	Pomeranje udesno	
00001110	11110110	10011001	0	$A = A - M$	Sedmi ciklus
00001110	11111011	01001100	1	Pomeranje udesno	
00001110	00001001	01001100	1	$A = A + M$	Osmi ciklus
00001110	00000100	10100110	0	Pomeranje udesno	
	00000100	10100110		Rezultat	14×85 = 1190

Tabela 6: Butov algoritam primenjen na množenje brojeva 14 i 85

U ovakvim slučajevima se koristi *modifikovani Butov algoritam*:

1. Formira se *Butov kodirani množilac* tako što se pretraži originalni množilac zdesna ulevo i upiše –1 na svakoj poziciji gde je 1 na početku niske, +1 kada se naidje na prvu sledeću 0, dok se na ostalim mestima upiše 0. Iz kodiranog množioca se vidi uspešnost optimizacije (pojava -1 znači da se primenjuje oduzimanje, +1 primenjuje se sabiranje, a 0 da nema akcije).

   Primer: za množioce +9, -9 i +85 kodirani množioci su:

0	0	0	0	1	0	0	1	Množilac $(+9)_{10}$
0	0	0	+1	-1	0	+1	-1	Kodirani množilac
1	1	1	1	0	1	1	1	Množilac $(-9)_{10}$
0	0	0	-1	+1	0	0	-1	Kodirani množilac
0	1	0	1	0	1	0	1	Množilac $(+85)_{10}$
+1	-1	+1	-1	+1	-1	+1	-1	Kodirani množilac

2. Izdvojiti parove oblika (a_{2k+1}, b_{2k}) iz kodiranog množioca, gde su u indeksu označene pozicije na kojima se nalaze vrednosti a i b, pri čemu važi $k \in [0, n/2-1]$. Težina a_{2k+1} je dvostruko veća od težine b_{2k}. Na osnovu težine se odredjuje zajednička vrednost svakog para.

   Na primer, zajednička vrednost para +1 –1 = +2 –1 = 1, dok je vrednost para 0 –1 = 0 – 1 = –1. Vrednosti parova kodiranog množioca su prikazane u narednoj tabeli, pri čemu se uzima da je vrednost bita množioca na poziciji -1 (za $k=0$) jednaka 0.

Butov par $(2k+1, 2k)$	Vrednost para	Bitovi množioca $(2k+1, 2k, 2k-1)$
0, 0	0	000 ili 111
0,-1	-1	110
0,+1	+1	001
+1, 0	+2	011
+1,-1	+1	010
-1, 0	-2	100
-1,+1	-1	101
+1,+1	**	***
-1,-1	**	***

3. Za svaki par koji se pojavi u kodiranom množiocu izvrši se pomeranje množenika za $2k$ mesta ulevo. Tako dobijena vrednost se množi sa vrednošću para i dodaje proizvodu. U implementaciji algoritma se odredjivanje Butovog kodiranog množioca i vrednosti njegovih parova spaja u jedan korak na osnovu trobitne kombinacije prikazane u prethodnoj tabeli.

Na primer, pri množenju 14×85:

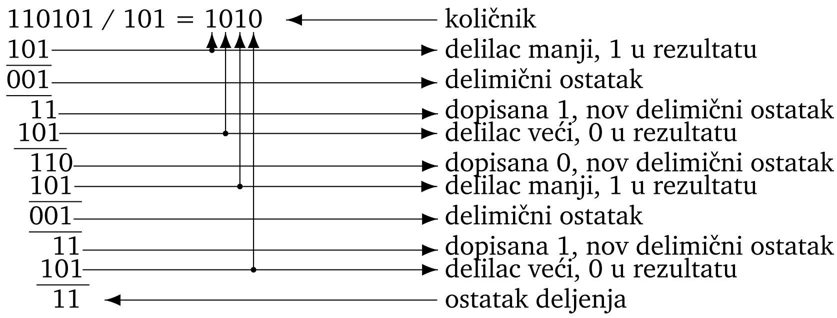
0	0	0	0	1	1	1	0	Množenik $(+14)_{10}$
0	1	0	1	0	1	0	1	Množilac $(+85)_{10}$
+1	-1	+1	-1	+1	-1	+1	-1	Kodirani množilac

sva četiri para su oblika (+1, -1) i imaju vrednost 1 (sve trobitne kombinacije oblika 010). Proizvod se dobija sabiranjem

0000000000001110	$k = 0$, množilac pomeren za 0 mesta ulevo
0000000000111000	$k = 1$, množilac pomeren za 2 mesta ulevo
0000000011100000	$k = 2$, množilac pomeren za 4 mesta ulevo
0000001110000000	$k = 3$, množilac pomeren za 6 mesta ulevo
0000010010100110	proizvod, 1190_{10}

Deljenje

Deljenje neoznačenih brojeva



Slika 5: Primer deljenja neoznačenih celih brojeva

Hardverska implementacija koristi tri registra P,A i M i brojač čija je inicijalna vrednost jednaka broju bitova u registrima. Inicijalno se u registar P upisuje deljenik, u registar M delilac, a u registar A nula. Deljenje se vrši na sledeći način:

1. U prvom koraku se pomera se sadržaj registara A i P (posmatranih kao jedna binarna reč) ulevo za jedan bit.

2. Sadržaj registra M se oduzima od sadržaja registra A i dobijena vrednost se upisuje u registar A.

3. Ispituje se da li je vrednost u registru $A \geq 0$, tj. da li može da se izvrši oduzimanje i dobije delimični ostatak. Ako može, tada se u bit najmanje težine registra P upisuje 1. U suprotnom se u bit najmanje težine registra P upisuje 0 i sadržaj registra M se sabira sa sadržajem registra A radi restauracije prethodnog sadržaja registra A.

4. Vrednost brojača se smanjuje za 1; ukoliko je vrednost brojača veća od nule, izvršavanje se vraća na korak 1.

5. Na kraju deljenja količnik se nalazi u registru P, ostatak u registru A.

Deljenje označenih brojeva zapisanih u potpunom komplementu

Prethodni algoritam se može proširiti tako da važi i za brojeve zapisane u potpunom komplementu:

1. Na početku se u registre A i P upisuje deljenik posmatran kao broj u potpunom komplementu dužine $2n$. Delilac se upisuje u registar M.

2. Sadržaj registara A i P se pomera (aritmetičkim pomeranjem) za jedno mesto u levo. Ako M i A imaju isti znak tada se izvršava operacija oduzimanja: $A = A - M$; u suprotnom se izvršava sabiranje $A = A + M$. Izvršena operacija se smatra uspešnom ako je znak registra A nepromenjen posle njenog izvršavanja.

3.
   - Ako je operacija uspešna ili $(A = 0 \wedge P = 0)$ tada se bit najmanje težine registra P postavlja na 1.
   - U suprotnom (ako je operacija neuspešna i $(A \neq 0 \vee P \neq 0)$) bit najmanje težine registra P se postavlja na 0 i restaurira se prethodna vrednost registra A.

4. Prethodna dva koraka se ponavljaju n puta, gde je n dužina registra P. Na kraju procesa registar A sadrži ostatak. Ako je znak deljenika i delioca isti tada je vrednost količnika zapisana u registru P. U suprotnom, za vrednost količnika treba uzeti vrednost u registru P sa promenjenim znakom.

M	A	P	Komentar
Primer 1: 24/9			
00001001	00000000	00011000	Početno stanje, 24/9
00001001	00000000	00110000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000000	00110000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00000000	01100000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000000	01100000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00000000	11000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000000	11000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00000001	10000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000001	10000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00000011	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000011	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00000110	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000110	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	00001100	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000011	00000001	$P_0 \longleftarrow 1$
00001001	00000110	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \longleftarrow A - M$
00001001	00000110	00000010	$P_0 \longleftarrow 0$, restauracija sadržaja A
	00000110	00000010	Količnik = 2, ostatak = 6

M	A	P	Komentar
Primer 2: 24/(-9)			
11110111	00000000	00011000	Početno stanje, 24/(-9)
11110111	00000000	00110000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000000	00110000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00000000	01100000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000000	01100000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00000000	11000000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000000	11000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00000001	10000000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000001	10000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00000011	00000000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000011	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00000110	00000000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000110	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	00001100	00000000	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000011	00000001	$P_0 \longleftarrow 1$
11110111	00000110	00000010	Pomeranje ulevo
	11110111		Sabiranje $A \longleftarrow A + M$
11110111	00000110	00000010	$P_0 \longleftarrow 0$, restauracija sadržaja A
	00000110	00000010	Znak deljenika i delioca nije isti $\Rightarrow$ $\Rightarrow$ količnik = - P = -2, ostatak = 6

M	A	P	Komentar
Primer 3: -24/9			
00001001	11111111	11101000	Početno stanje, -24/9
00001001	11111111	11010000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111111	11010000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11111111	10100000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111111	10100000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11111111	01000000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111111	01000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11111110	10000000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111110	10000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11111101	00000000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111101	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11111010	00000000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111010	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
00001001	11110100	00000000	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111101	00000001	$P_0 \longleftarrow 1$
00001001	11111010	00000010	Pomeranje ulevo
	00001001		Sabiranje $A \longleftarrow A+M$
00001001	11111010	00000010	$P_0 \longleftarrow 0$, restauracija sadržaja A
	11111010	00000010	Znak deljenika i delioca nije isti $\Rightarrow$ $\Rightarrow$ količnik = - P = -2, ostatak = -6

M	A	P	Komentar
Primer 4: -24/(-9)			
11110111	11111111	11101000	Početno stanje, -24/(-9)
11110111	11111111	11010000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111111	11010000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11111111	10100000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111111	10100000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11111111	01000000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111111	01000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11111110	10000000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111110	10000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11111101	00000000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111101	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11111010	00000000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111010	00000000	$P_0 \longleftarrow 0$, restauracija sadržaja A
11110111	11110100	00000000	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111101	00000001	$P_0 \longleftarrow 1$
11110111	11111010	00000010	Pomeranje ulevo
	11110111		Oduzimanje $A \longleftarrow A - M$
11110111	11111010	00000010	$P_0 \longleftarrow 0$, restauracija sadržaja A
	11111010	00000010	Količnik = 2, ostatak = -6