Predstavljanje podataka u računaru

Brojni sistemi

Brojevi se mogu predstaviti u bilo kojoj bazi - osnovi (dekadni 10)

- Simboli brojnog sistema osnove B su 0, 1, 2, …, B –1
- Dekadni (decimalni) sistem 0, 1, 2, .., 9; binarni (osnova 2) 0, 1
- Vrijednost i-te cifre d je "$d * B^i$" gdje i počinje od 0 i povećava se sdesna ulijevo

2 1 0	i - pozicija		poziciona notacija
3 7 5	d - cifra		
$5 * 10^0$	=	5	Trista sedamdeset pet
$7 * 10^1$	=	70	
$3 * 10^2$	=	300	

Matematički zapis

- $a_n a_{n-1} \ldots a_2 a_1 a_0 = a_n\, b^n + a_{n-1}\, b^{n-1} + \ldots + a_2\, b^2 + a_1\, b^1 + a_0\, b^0$

gdje su: b – baza (osnova) brojnog sistema, i

a_i – cifre brojnog sistema čija vrijednost može biti od nule do baze -1 ($a_i \in \{0,1,\ldots,b-1\}$).

$$23\boxed{2}5 = 2\times10^3 + 3\times10^2 + \boxed{2}\times10^1 + 5\times10^0$$

7	6	5	4	3	2	1	0	stepen pozicije
0	0	0	1	0	1	0	1	binarni broj

Konverzija iz binarnog u dekadni

Konvertovati 1011_2 u decimalni broj

3 2 1 0   i

1 0 1 1 d

$= (1 * 2^0) + (1 * 2^1) + (0 * 2^2) + (1 * 2^3)$

$= 1 + 2 + 0 + 8$

$= 11_{10}$

Ovaj proces može se primjeniti za konverziju bilo kog sistema u dekadni, samo umjesto 2 stavimo odgovarajuću osnovu

Konverzija iz dekadnog u binarni

Korak 1: podijeliti sa 2 i sačuvati ostatak

Korak 2: dok god količnik nije nula, dijeliti novi količnik sa 2 i sačuvati ostatak

Korak 3: kada je količnik nula, binarno predstavljanje je lista ostataka u obrnutom redosljedu

Konvertovati 13_{10} u binarni

Operacija	***Količnik***	***Ostatak***
13 / 2	6	1
6 / 2	3	0
3 / 2	1	1
1 / 2	0	1

$13_{10} = 1101_2$

Ostali brojni sistemi

- Oktalni (osnova 8)

  Simboli (0, 1, 2, 3, 4, 5, 6, 7)
- Problem sa previše dugim binarnim brojevima
- Heksadecimalni (osnova 16)

  Simboli (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)
- Byte = 8 bits = 2 hex cifre (1 hex cifra je 4 bita)

B_{16} is 1011_2

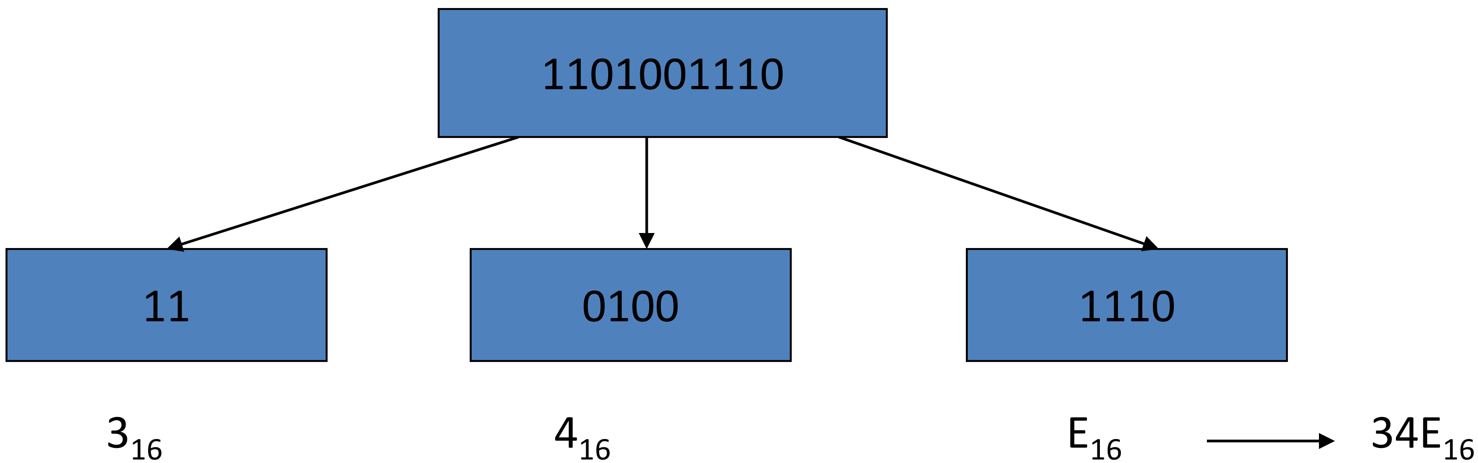
$B5_{16} = ?_2$ $\qquad$ $B5_{16} = 10110101_2$

5_{16} is 0101_2

Konverzija iz binarnog u hex

Konvertovati 1101001110_2 u hex

Grupišimo cifre u grupe po 4 sdesna ulijevo



Decimalni	Binarni	Heksadecimalni
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

$2^0 = 1$

$2^1 = 2$

$2^2 = 4$

$2^3 = 8$

$2^4 = 16$

$2^5 = 32$

$2^6 = 64$

$2^7 = 128$

$2^8 = 256$

$2^9 = 512$

$2^{10} = 1024$

$2^{11} = 2048$

$2^{12} = 4096$

$2^{13} = 8192$

Razlomci

- Koristi se decimalna tačka kao u decimalnom sistemu
- Desno od decimalne tačke pozicije se numerišu sa -1, -2 , ...

2 1 0 -1 -2 -3	pozicija
1 0 1. 1 0 1	cifra

$$101.101 = (1 * 2^0) + (0 * 2^1) + (1*2^2) + (1*2^{-1}) + (0* 2^{-2}) + (1 * 2^{-3})$$

$$= 1 + 4 + ½ + 1/8 = 5\frac{5}{8}$$

Razlomci

Broj $3\frac{5}{16}$ konvertovati u binarni

- Konvertovati prvo cjelobrojni dio, pa zatim razlomljeni dio

3_{10} je 11_2

$$\frac{5}{16} = \frac{1}{16} + \frac{4}{16} = \frac{1}{16} + \frac{1}{4} = 0.0101_2$$

$$3\frac{5}{16} = 11.0101_2$$

Konverzija razlomka u binarni

Korak 1: pomnožiti sa 2 i sačuvati ostatak

Korak 2: dok god proizvod nije jedan, množiti novi priozvod sa 2 i sačuvati ostatak

Korak 3: kada je proizvod jedan, binarno predstavljanje je lista ostataka

Konvertovati 5/16 =0.3125 u binarni

Operacija	***Proizvod***	***Ostatak***
(5/16)*2 ili 0,3125*2	10/16=5/8=0,625	0
(10/16)*2 ili 0.625*2	20/16=5/4=1,25	1
(1/4)*2 ili 0.25*2	2/4=1/2=0,5	0
(1/2)*2 ili 0,5*2	2/2=1	1

$(5/16)_{10} = 0.3125_{10} = 0.0101_2$

Predstavljanje podataka

- Skup cijelih brojeva u matematici i skup cijelih brojeva na računaru se razlikuju
- Skup realnih brojeva u matematici i skup realnih brojeva na računaru se razlikuju
- Kako predstavljamo cijele brojeve, realne brojeve, karaktere, slike i zvuk u računaru?
- Tehnike kompresije podataka

Cijeli brojevi

Potpuni komplement (Two's Complement)

Invertovati bitove
0 postaje 1
1 postaje 0

- Za pozitivne brojeve, samo naći bina sa nulom kao prvim bitom
- Za negativne brojeve, naći komplement pozitivne vrijednosti i dodati 1

3 u potpunom komplementu je 011

-3 u potpunom komplementu je:

Invertovati bitove   011 postaje 100

Dodati 1   100 + 1 = 101

Cijeli brojevi

Šta je 1010 u potpunom komplementu?

Negativan broj jer je krajnji lijevi bit 1

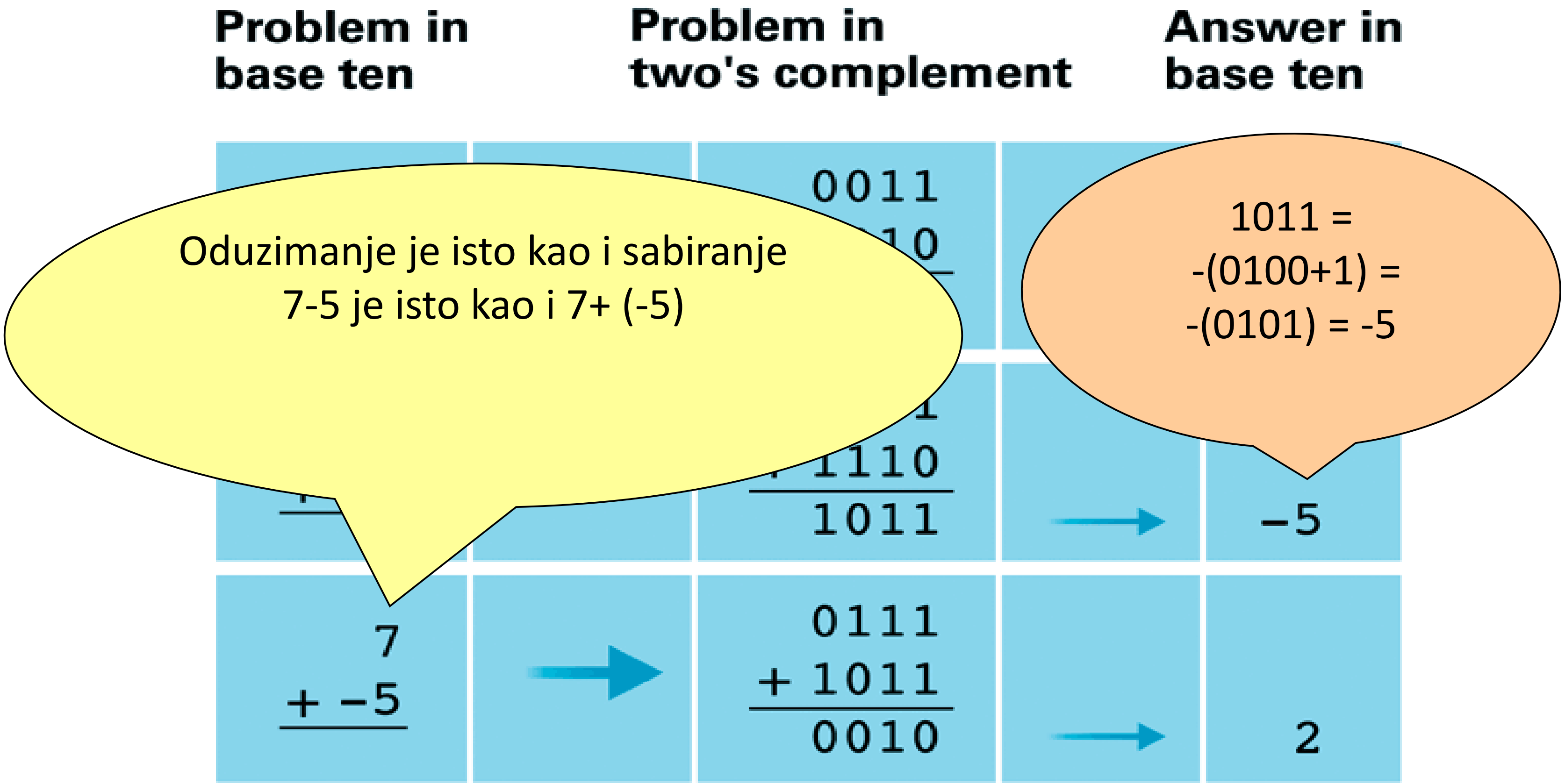
Invertovati bitove   1010 postaje 0101

Dodaj 1   0101 + 1 = 0110 (+6)

Originalni broj je -6

Negativni brojevi

Sabiranje u potpunom komplementu



Opseg cijelih brojeva

7	6	5	4	3	2	1	0
0	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Pozitivan broj

21 =	0	0	0	1	0	1	0	1
	1	1	1	0	1	0	1	0
+								1
-21 =	1	1	1	0	1	0	1	1

Negativan broj

Za 8 bita: od -128 do 127 (od -2^7 do 2^7-1)

Za 16 bita: od -2^{15} do $2^{15}-1$

Za 32 bita: od -2^{31} do $2^{31}-1$

Overflow

Sabiranje 5 + 4 u čevorobitnoj notaciji potpunog komplementa

5	0101
+4	0100
9	1001

→ **Overflow (prekoračenje)**. Rezultat je negativna vrijednsot

–(0110+1) = -(0111) = -7

Postoje ograničenja na veličinu vrijednosti koja se može predstaviti.

Obično se cijeli brojevi (integers) predstavljaju sa 32 bita.

Cijeli brojevi u računaru

Excess notacija

Excess 4 notacija

Niz bitova	**Vrijednost**	**Neoznačeni**
111	3	7
110	2	6
101	1	5
100	0	4
011	-1	3
010	-2	2
001	-3	1
000	-4	0

- Svi pozitivni počinju sa 1
- Svi negativni počinju sa 0
- 0 je predstavljena kao 100
- Neoznačeni brojevi su za 4 veći od vrijednosti koju predstavljaju bitovi, pa je zato ime excess 4
- Zašto 4?

  $(2^{\#\text{of bits} - 1}) = 2^{3-1} = 4$

- Najmanji negativan broj je 000
- Najveći pozitivan broj je 111

Cijeli brojevi u računaru

Šta je 101 u excess 4 notaciji?

101 je niz bitova koji predstavlja 5

101 u excess 4 notaciji je (5-4) = 1

Kako se 3 reprezentuje u excess 4 notaciji?

Excess 4 znači da su nam potrebna (4-1) = 3 bita za predstavljanje

Dodajemo 4 na datu vrijednost 3+4 = 7

Predstavljamo 7 kao 3-bitni binaran broj = 111 3 u excess 4 je 111

Excess-127

8 bit excess-127

Binary value	Excess-127 interpretation	Unsigned interpretation
00000000	-127	0
00000001	-126	1
...	...	...
01111111	0	127
10000000	1	128
...	...	...
11111111	+128	255

The IEEE floating-point standard defines the exponent field of a single-precision (32-bit) number as an 8-bit Excess-127 field. The double-precision (64-bit) exponent field is an 11-bit Excess-1023 field.

Cijeli brojevi - zaključak

- U računaru se mogu uskladištiti samo brojevi unutar opsega.
- Uskladišteni brojevi se predstavljaju u računaru tačno.
- Rezultat arimetičke operacije sa cijelim brojevima je cio broj.
- Ako je rezultat neke operacije izvan opsega cijelih brojeva dolazi do prekoračenja opsega (Integer Overflow) i prekida rada programa.