VTŠ: Osnovi računarske tehnike

Binarna aritmetika

mr. Veličković Zoran

Februar, 2010.

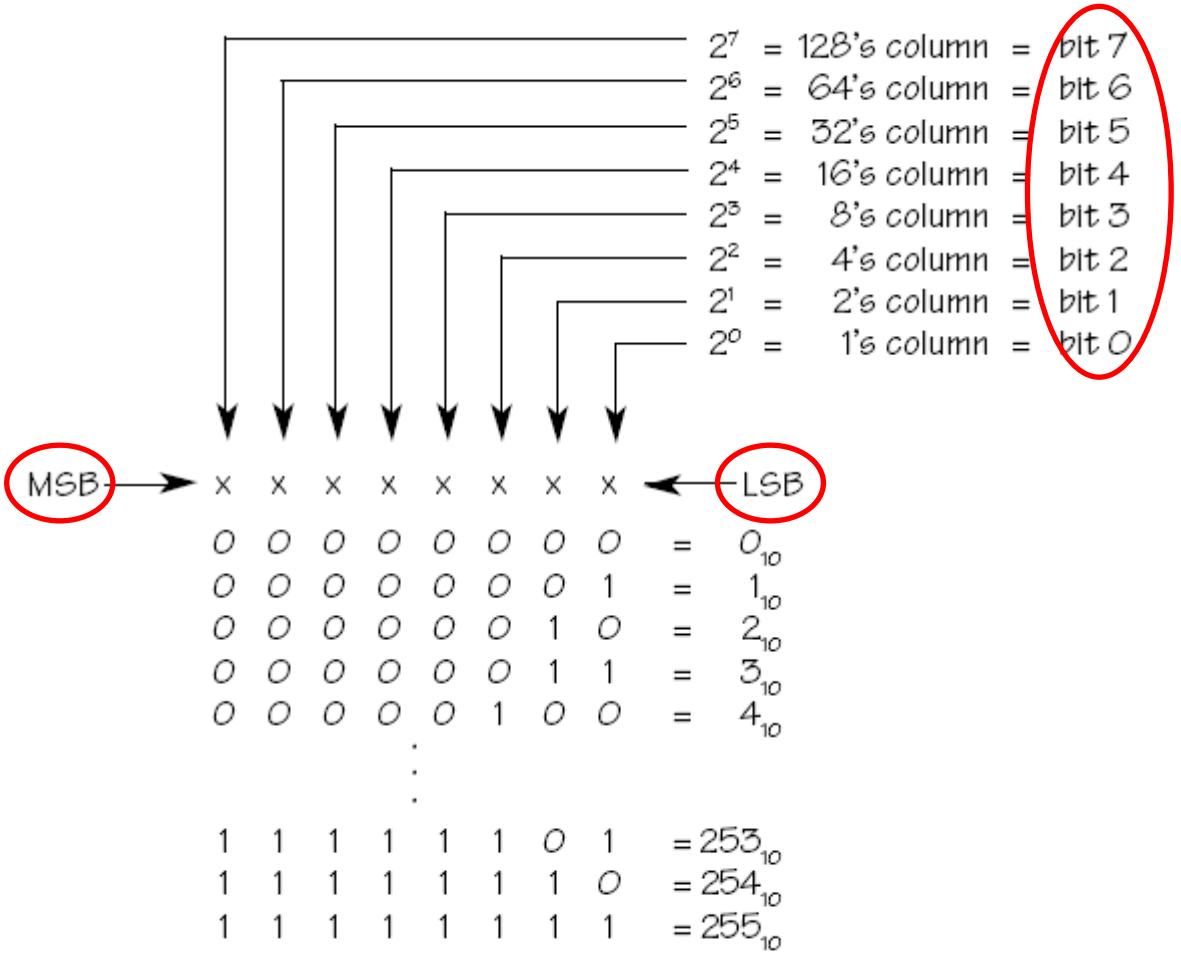
Binarna aritmetika

- Slično kao i kod aritmetike na papiru, i decimalni i binarni brojevi mogu biti bilo koje veličine, ograničenja su samo u veličini papira i dugotrajnosti olovke!
- U manipulaciji podataka u računaru se koriste fizičke logičke kapije i veze, tako da je njima određena širina podataka.
- Tako je maksimalna vrednost broja reprezentovanog u računaru odeđena brojem bitova koji predstavljaju neki broj.

Neoznačen binarni broj

- Neoznačen (unsigned) binarni brojevi mogu se koristiti za prikazivanje samo **pozitivnih** vrednosti. (Na sledećem slajdu prikazan je opseg brojeva koji se mogu reprezentovati sa 8 bitova.
- ''**x**'' (''*****'') karakter označava jedan bit (bilo 0 bilo 1)
- Na krajnjoj desnoj strani se nalazi bit najmanje težine (*least significant bit -LSB), indeksira se kao **bit 0**.*
- *Na levoj strani se nalazi najteži bit (most significant bit - MSB), indeksira se kao **bit 7**.*
- Svaki individualni bit može uzeti vrednost 1 ili 0, tako da grupa od 8 bitova može predstaviti $\mathbf{2^8}$ jedinstvenih kombinacija 0 i 1.
- U decimalnoj notaciji to se predstavlja 0_{10} do $+255_{10}$

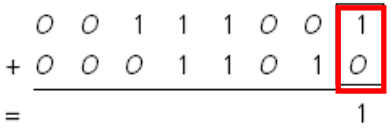
Neoznačen binarni broj (2)



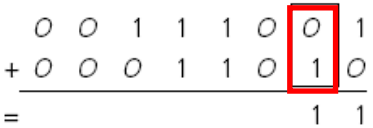
Binarno sabiranje (1)

- Dva binarna broja se sabiraju na sličan način kao i kod decimalne notacije.
- Prvo se sabiraju dva najmanje značajna LSB bita i ako je potrebno formiraju prenos *za sabiranje u sledećem koraku*.
- Piostupak se ponavlja sve dok se ne dođe do MSB bita.
- Pogledajmo primer.

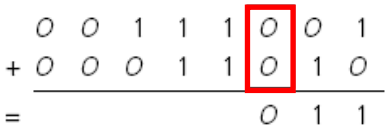
Binarno sabiranje (2)



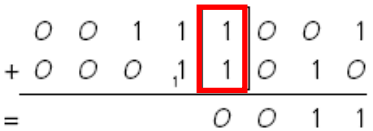
(a) Bit 0, $1 + 0 = 1_2$



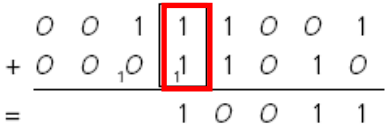
(b) Bit 1, $0 + 1 = 1_2$



(c) Bit 2, $0 + 0 = 0_2$

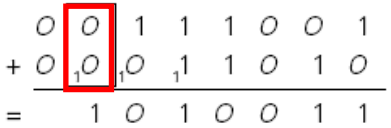


(d) Bit 3, $1 + 1 = 10_2$

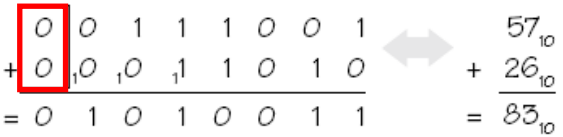


(e) Bit 4, $1 + 1 + \text{carry\_in} = 11_2$

(f) Bit 5, $1 + 0 + \text{carry\_in} = 10_2$



(g) Bit 6, $0 + 0 + \text{carry\_in} = 1_2$

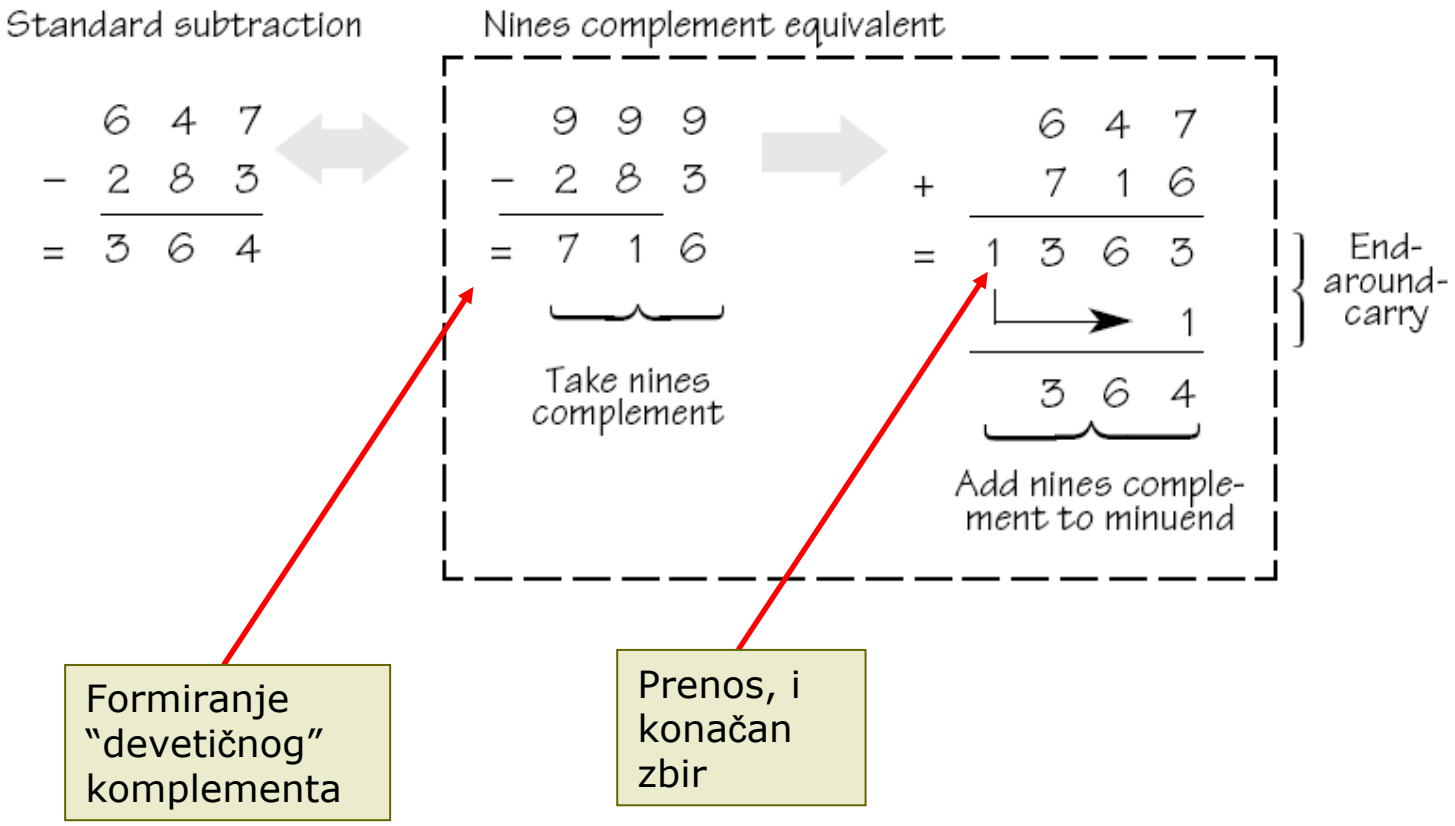


(h) Bit 7, $0 + 0 = 0_2$

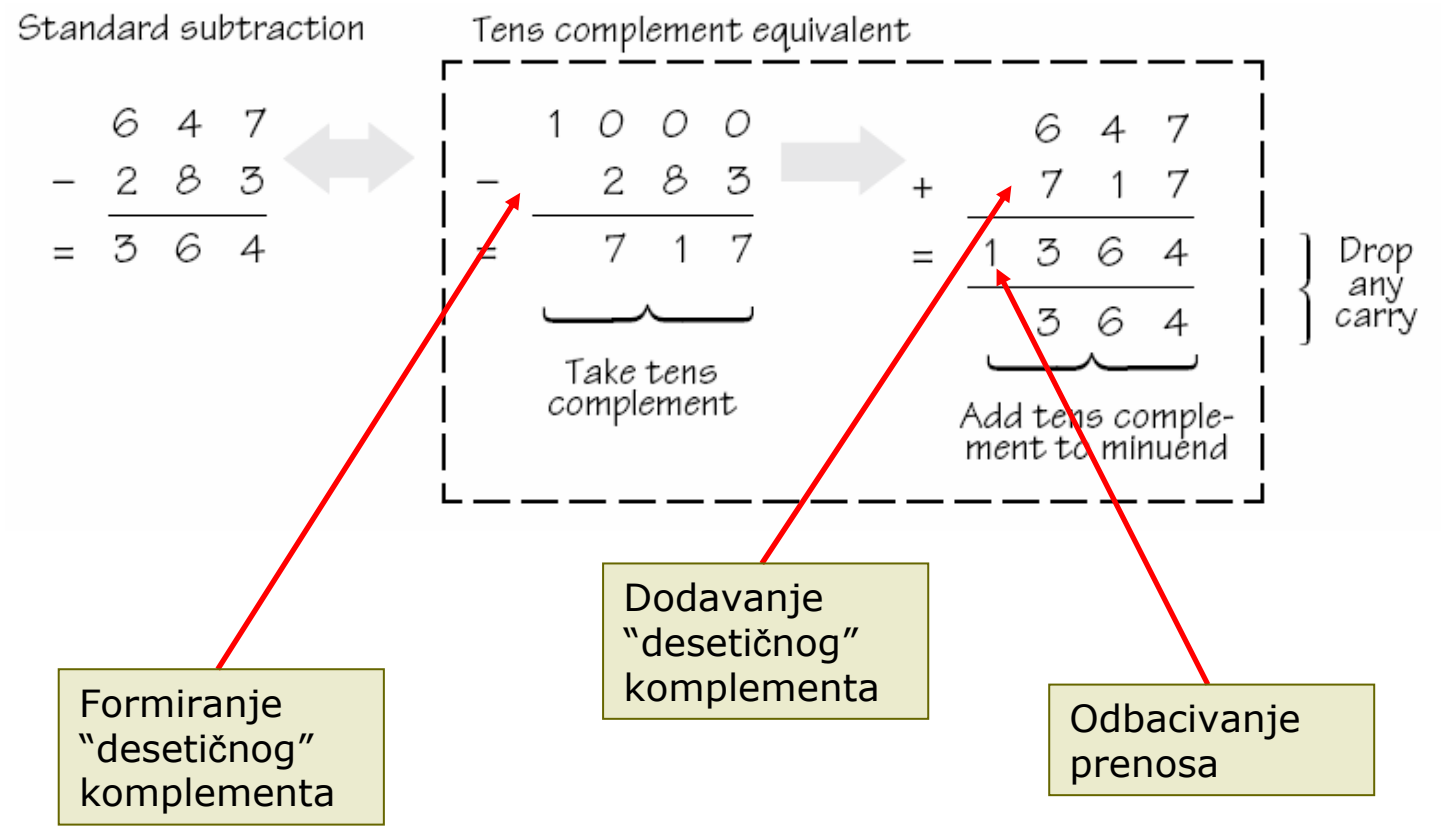
Binarno oduzimanje

- Binarno oduzimanje se može obaviti slično kao kod decimalne notacije.
- Međutim, kod računara se koristi takozvana komplementarna tehnika.
- Da bi se ova tehnika realizovala kod decimalne notacije treba formirati *tens* i *nines* complement.
- Kod binarnog oduzimanja ekvivalenti u *dvojični* i *jedinični* komplement.
- Pogledajmo sledeće primere.

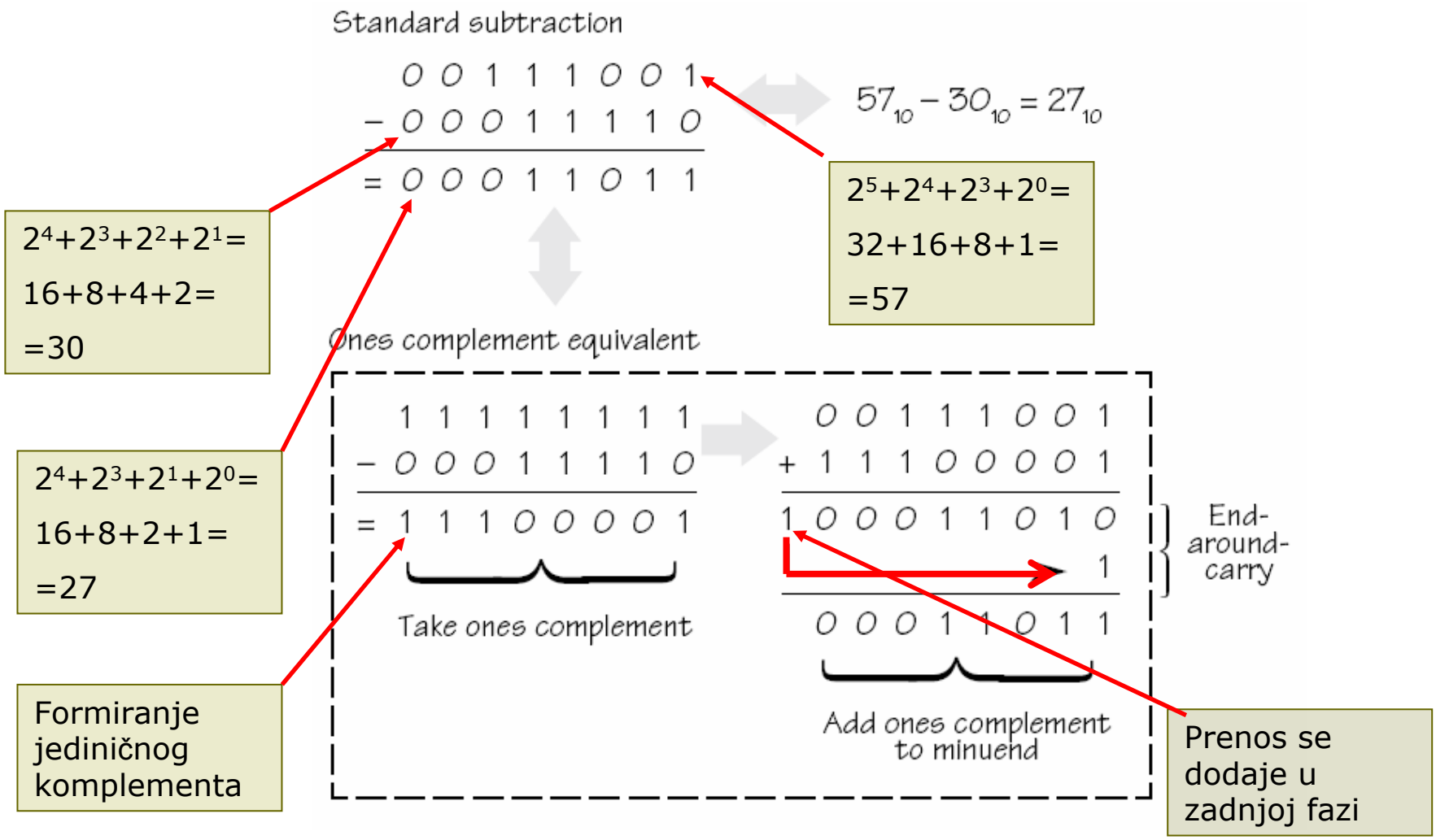
Nines komplement$_{10}$



Tens komplement$_{10}$



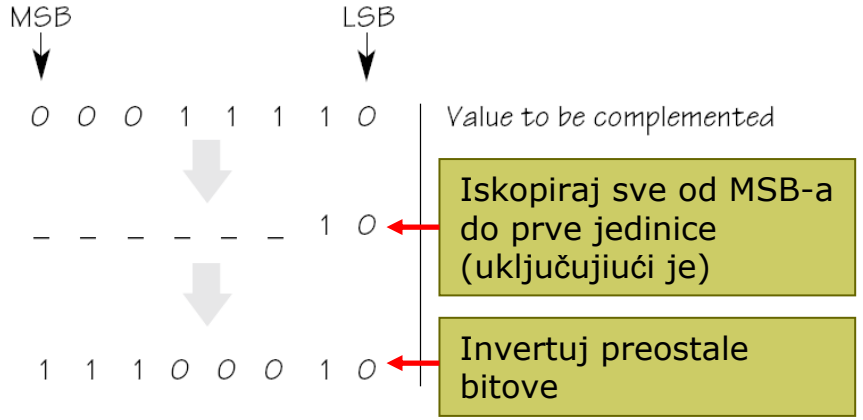
Jedinični komplement$_2$



Dvojični komplement$_2$

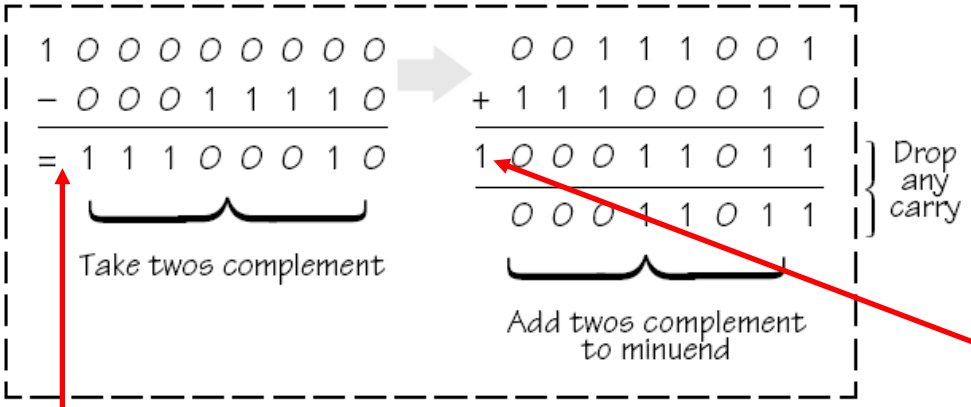
Standardno oduzimanje

```
  0 0 1 1 1 0 0 1
- 0 0 0 1 1 1 1 0
-----------------
= 0 0 0 1 1 0 1 1
```



Formiranje dvoičnog komplementa, jedan od načina

Oduzimanje dvojičnim komplementom



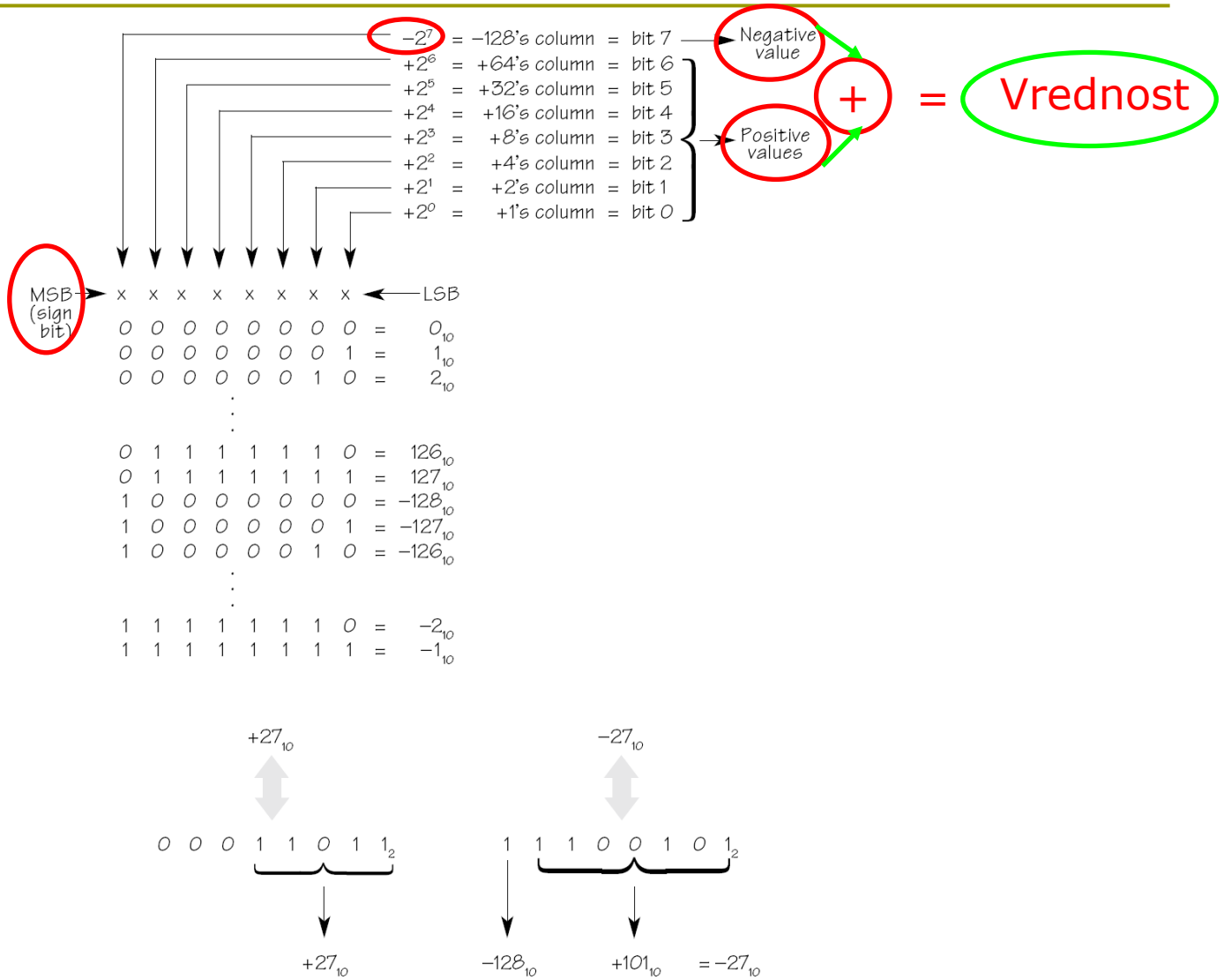
Formiranje dvoičnog komplementa

Prenos se odbacuje u zadnjoj fazi

Označeni binarni brojevi (1)

- Označeni brojevi se koriste za predstavljanje **pozitivnih** i **negativnih** veličina.
- Kod decimalne notacije se koristi **sign-magnitude** način za predstavu negativnih brojeva. Primer **+**27 i **-**27.
- Iz razloga efikasnosti u računarima se ne koristi ovaj pristup!
- Naime, u računarima se koristi **sign-bit** način predstave negativnih brojeva. Vrednost MSB bita određuje znak binarnog broja.
- Pogledajte sledeću tabelu u kojoj je predstavljeno korišćenje bita za označavanje znaka (sign bita)

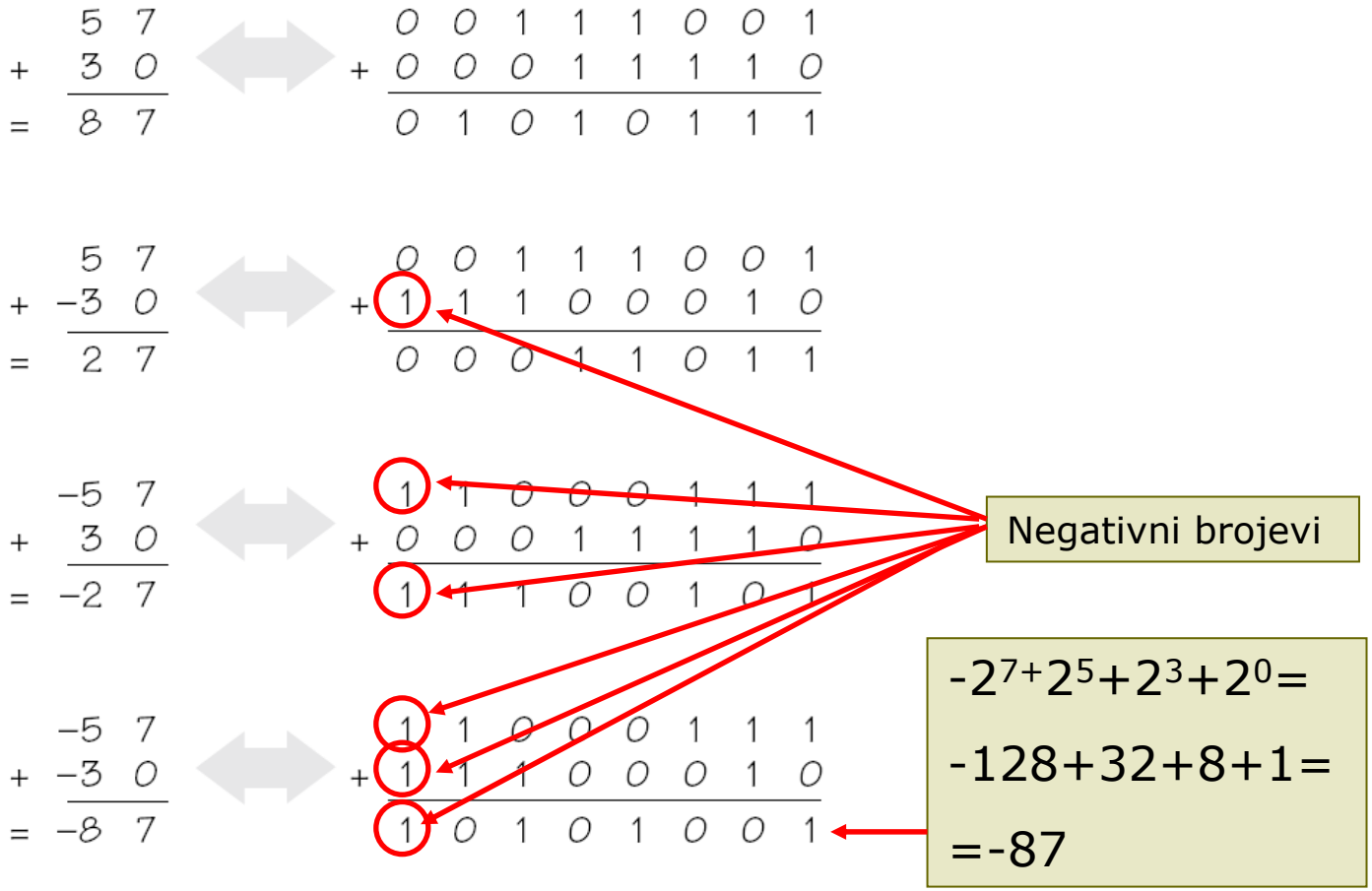
Označeni binarni brojevi (2)



Oduzimanje i označeni brojevi

Decimalno: signed-magnitude

Binarno: signed-binarno



Prednosti signed-binary predstave

- Brojevi mogu biti uvek dodati direktno bez obzira da li oni predstavljaju pozitivne ili negativne brojeve.
- Aritmetička operacije:

  $$a + b,\ \ a + (-b),\ \ (-a) + b,\ \text{i}\ \ (-a) + (-b)$$

  se obavljaju uvek na **isti način**, jednostavno sabiranjem ovih dveju veličina.
- Ovo ubrzava rad računara i može se realizovati sa manjim brojem logičkih kapija.
- Dakle, računari ne zahtevaju različite blokove za sabiranje i oduzimanje, već samo zahtevaju sabirač i dvojični komplementator (koji se realizuje sa par logičkih kola).
- Realizaciju operacija binarnog oduzimanja i binarnog sabiranja učićemo kasnije.

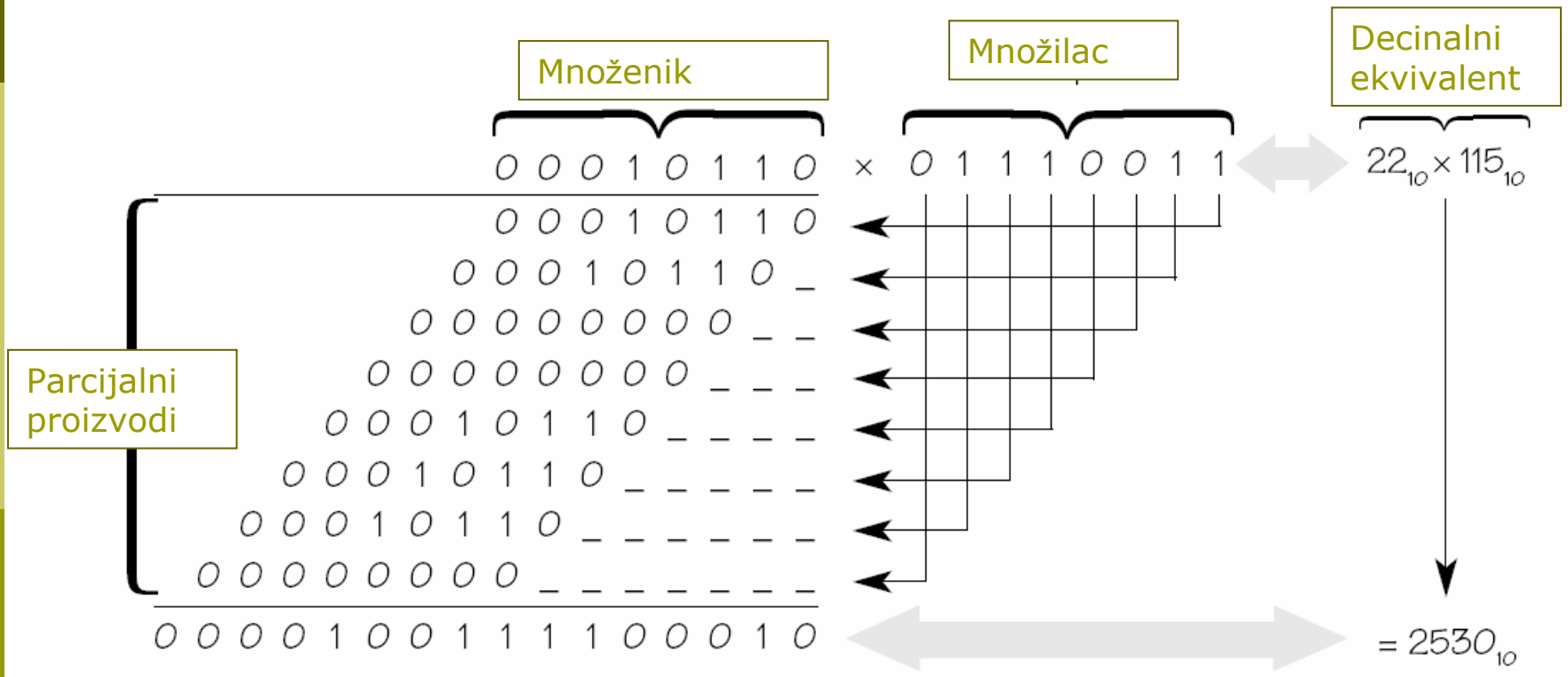
Binarno množenje

- Jedan od načina realizacije množenja je prevođenje u **višestruko sabiranje**:

$$6 \times 4 = 6 + 6 + 6 + 6 = 24.$$

- Iako je to jednostavno za računare, u slučaju dugih nizova brojeva to može biti vremenski zahtevan proces.
- Još jedan način za realizaciju množenja je tehnika "**pomeri i saberi**" (shift-and-add).
- Pogledajmo sledeći primer ove tehnike:

Tehnika “pomeri i saberi”



$2^{11}+2^8+2^7+2^6+2^5+2^1=$

$2048+256+128+64+32+2$

$=2530.$

$2*10^3+5*10^2+3*10^1=$

$2000+500+30=$

$=2530.$