- Predstavljanje podataka u računaru-

Vrste podataka

- Numerički podaci:
  - Celi neoznačeni brojevi
  - Celi označeni brojevi
  - Realni brojevi u fiksnom zarezu
  - Realni brojevi u pokretnom zarezu
- Nenumerički podaci:
  - Tekst
  - Slika
  - Audio i video zapis

Predstavljanje numerickih podataka u binarnom brojnom sistemu

Predstavljanje celih neoznačenih brojeva

- Celi neoznačeni brojevi se u računaru pamte u binarnom brojnom sistemu.
- Kada je zapis broja kraći od veličine registra (jedne memorijske reči), on se dopunjuje nevažećim nulama sa leve strane.

Predstavljanje celih neoznačenih brojeva

- Primer: Predstavti broj 25 u 32-bitnom registru računara.
  - Korak 1: prevesti broj u binarni brojni sistem

$$(25)_{10} = (11001)_2$$

  - Korak 2: prikazati sadržaj registra

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

Predstavljanje celih označenih brojeva

- Prosto kodiranje znaka
- Nepotpuni komplement
- Potpuni komplement
- Pomeraj

Prosto kodiranje znaka

- Pozicija najveće težine se koristi za predstavljanje znaka broja
  - 0 na poziciji najveće težine označava pozitivan broj
  - 1 na poziciji najveće težine označava negativan broj

Prosto kodiranje znaka

- Primer: Predstaviti brojeve 25 i -25 u 32-bitnom registru računara ako se za predstavljanje znaka koristi prosto kodiranje.

$$(25)_{10} = (11001)_2$$

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-25:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

Nepotpuni komplement (Komplement najveće cifre)

- Brojevi se transformišu po sledećoj formuli:

$$A = \begin{cases} A, & A \geq 0 \\ q^n - 1 - |A|, & A < 0 \end{cases}$$

gde je:

q – osnova brojnog sistema,

n – ukupan broj pozicija predvidjen za predstavljanje broja

Komplement jedinice

- Ukoliko se koristi binarni brojni sistem, nepotpuni komplement se naziva
  - Jedinični komplement ili
  - Komplement jedinice.

Postupak nalaženja komplementa jedinice

- Broj se predstavi u binarnom brojnom sistemu i na poziciju najveće težine se upiše nula.
- Ukoliko je broj negativan, komplementira se svaki bit u zapisu.
  - Komplementirati – dopuniti do najveće cifre
  - Nula se zemenjuje jedinicom, a jedinica nulom

Komplement jedinice

- <u>Primer:</u> Predstaviti brojeve 25 i -25 u 32-bitnom registru računara ako se za predstavljanje znaka koristi komplement jedinice.

$$(25)_{10} = (11001)_2$$

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
-25:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0

Znak broja u nepotpunom komplementu

- Znak broja se pamti na poziciji najveće težine
  - 0 – broj je pozitivan
  - 1 – broj je negativan

Nedostaci jediničnog komplementa

- Postoje dva načina za prestavljanje nule:

  00...0 i

  11...1

- Pri izvršavanju operacija sabiranja i oduzimanja, prenos sa pozicije najveće težine se dodaje na poziciju najmanje težine.

Potpuni komplement (Komplement osnove)

- Brojevi se transformišu po sledećoj formuli:

$$A = \begin{cases} A, & A \geq 0 \\ q^n - |A|, & A < 0 \end{cases}$$

gde je:

q – osnova brojnog sistema,

n – ukupan broj pozicija predvidjen za predstavljanje broja

Komplement dvojke

- Ukoliko se koristi binarni brojni sistem, potpuni komplement se naziva
  - Dvojični komplement ili
  - Komplement dvojke.

Komplement dvojke

- <u>Primer:</u> Predstavti brojeve 25 i -25 u 32-bitnom registru računara ako se za predstavljanje znaka koristi komplement dvojke.

$$(25)_{10} = (11001)_2$$

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-25:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1

Pomeraj

- Brojevi se transformišu po sledećoj formuli:

$$A = A+p$$

gde je:

p – pomeraj čija je vrednost

obično q^{n-1}

q – osnova brojnog sistema,

n – ukupan broj pozicija predvidjen

za predstavljanje broja

Pomeraj

- Primer: Predstavti brojeve 25 i -25 u 32-bitnom registru računara ako se za predstavljanje znaka koristi pomeraj.

$$(25)_{10} = (11001)_2$$

25: 11001+10000000000000000000000000000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

-25: -11001+10000000000000000000000000000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1

Znak broja kod pomeraja

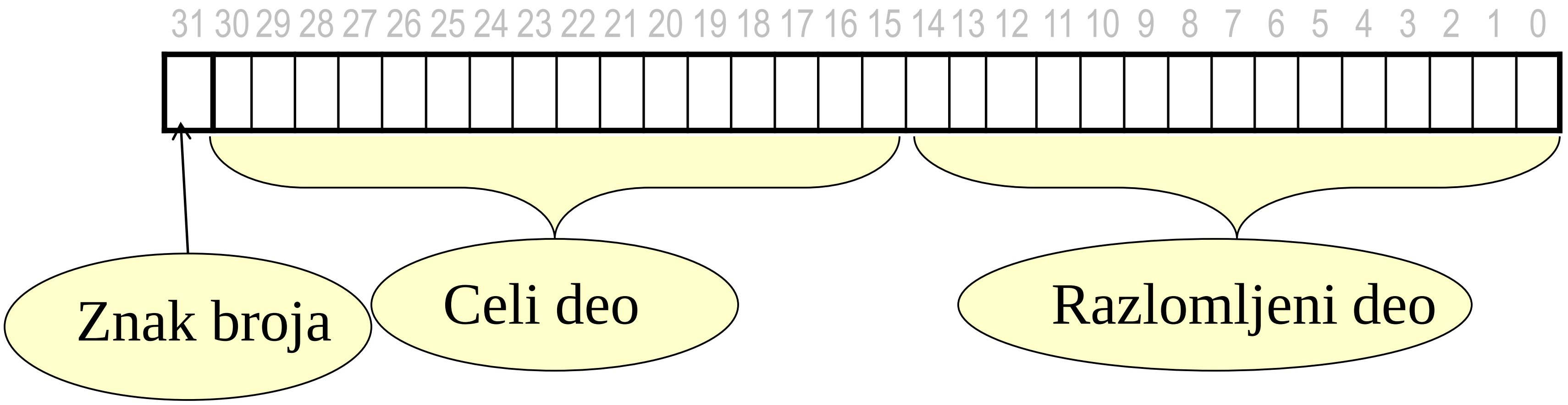
- Znak broja se pamti na poziciji najveće težine
  - 1 – broj je pozitivan
  - 0 – broj je negativan

Opseg celih brojeva koji se mogu predstaviti u računaru

- Kod 32-bitnih računara: od -2^{31} do $2^{31}-1$.
- Kod 64-bitnih računara: od -2^{63} do $2^{63}-1$.

Fiksni zarez

- Fiksira se broj pozicija koje se koriste za prestavljanje celog deo broja (n) i broj pozicija za predstavljanje razlomljenog dela broja (m).
- Za n=16 i m=15 sadržaj 32-bitnog registra bio bi:



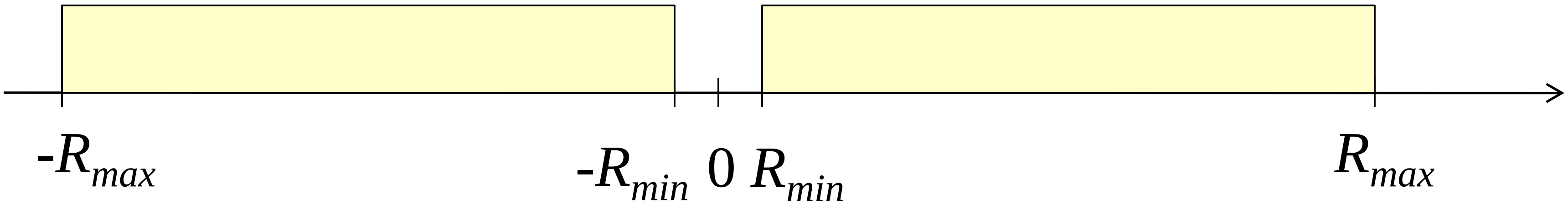
Fiksni zarez

- Primer: Predstaviti broj 27.125 u 32-bitnom registru ukoliko se predstavljanje celog dela broja koristi 16, a za predstavljanje razlomljenog dela broja 15 pozicija.

$$(27.125)_{10} = (11011.001)_2$$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Opseg brojeva koji se mogu predstaviti fiksnim zarezom



$R_{min} = 2^{-m}$ - minimalna vrednost i korak diskretizacije

$R_{max} = 2^{n} - 2^{-m}$ - maksimalna vrednost

Eksponencijalni zapis broja

- U praksi, neke veličine imaju mnogo malu ili mnogo veću vrednost od onih koje se fiksnim zarezom mogu predstaviti.

- Primer: $1.0 \cdot 10^{25}$ ili $2.57 \cdot 10^{-33}$

Eksponencijalni zapis broja

- Broj predstavljen u brojnom sistemu sa osnovom b se može zapisati u eksponencijalnom obliku na sledeći način:

$$m \cdot b^E$$

gde je:

m – mantisa,

E - eksponent

Novi standard za zapis normalizovane binarne mantise

- Pošto je prva cifra u normalizovanoj binarnoj mantisi obavezno 1, po novom standardu, normalizovani zapis binarne mantise je:

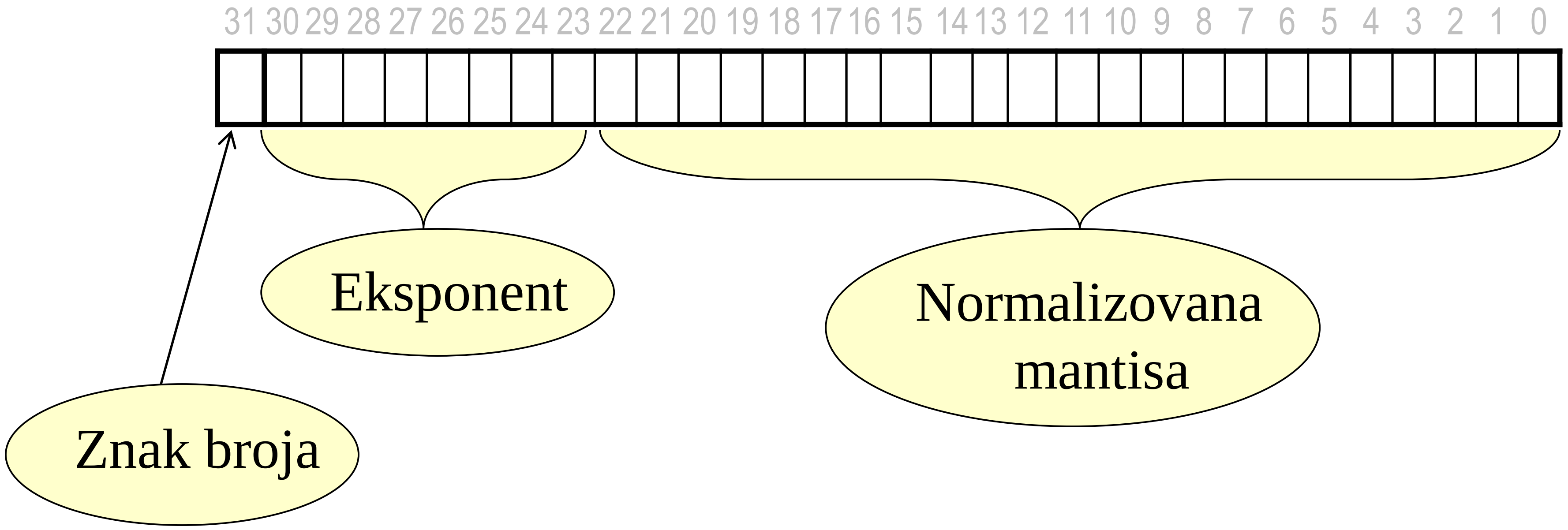
$$1.b_1 \ldots b_n$$

pri čemu se jedinica ispred decimalne tačke ne pamti (ona se podrazumeva).

Pokretni zarez

- Pokretni zarez broja se bazira na eksponencijalnom zapisu binarnog broja.
- Pokretni zarez broja ima 3 elementa:
  - Znak broja,
  - Eksponent (koji se pamti sa pomerajem),
  - Normalizovanu mantisu.

Pokretni zarez



Pokretni zarez

- <u>Primer:</u> Predstaviti broj 27.125 u 32-bitnom registru ukoliko se za predstavljanje broja koristi pokretni zarez.

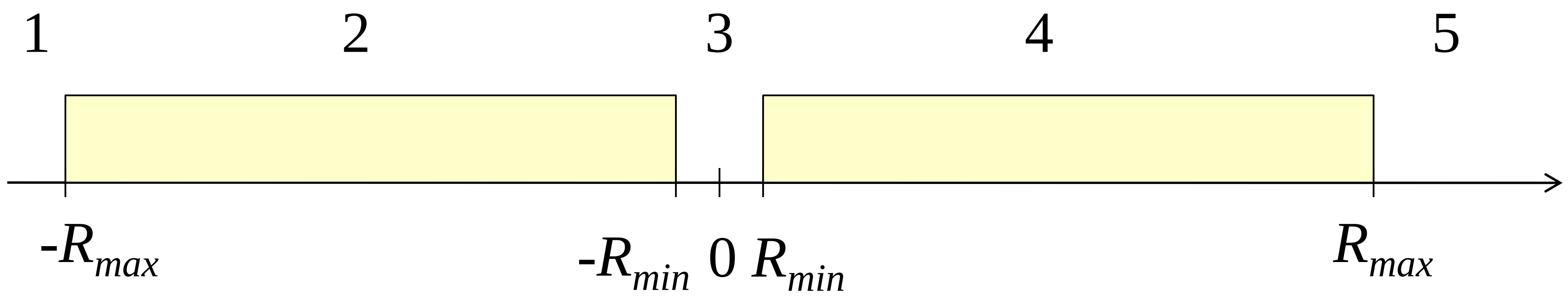
  $(27.125)_{10}=(11011.001)_2=(1.1011001\cdot 2^4)_2$

  Eksponent se pamti sa pomerajem:

  $e=(127+4)_{10}=(10000011)_2$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Opseg brojeva koji se mogu predstaviti pokretnom zarezom



$R_{min} = 0.12 \cdot 10^{-37}$ - minimalna vrednost

$R_{max} = 0.34 \cdot 10^{39}$ - maksimalna vrednost

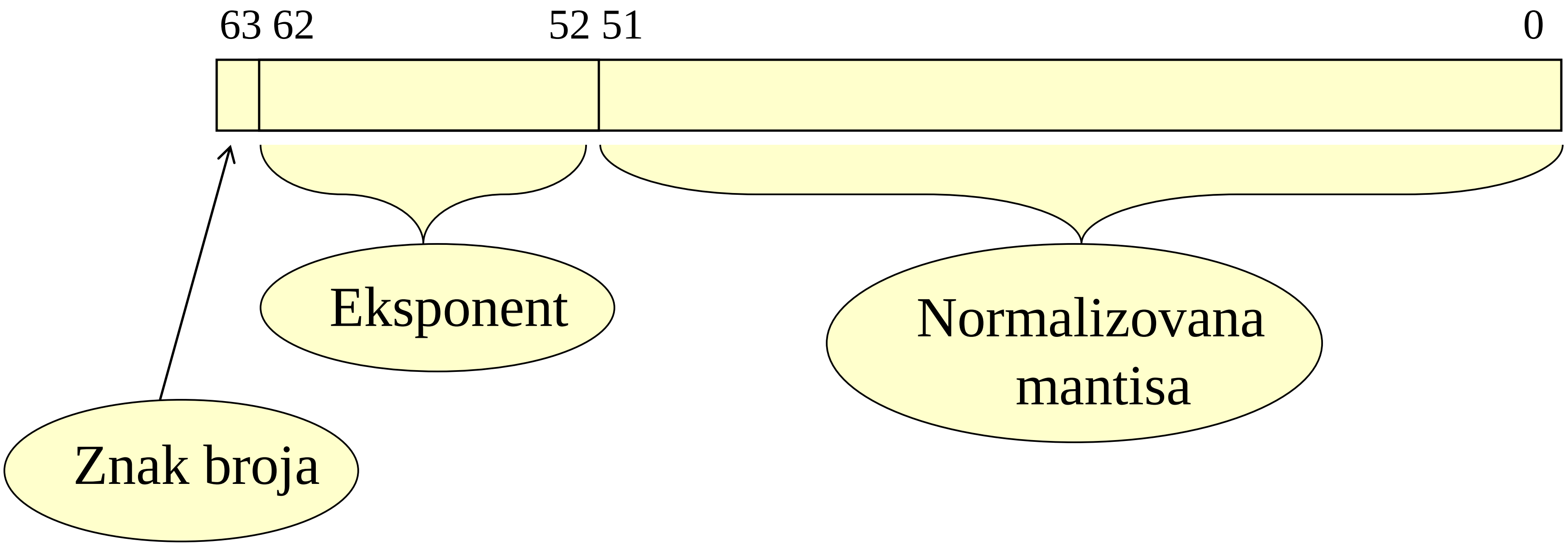
Obeležene oblasti:

1 - Oblast negativnog prekoračenja (-∞)

5 - Oblast pozititivnog prekoračenja (+ ∞)

3 – Oblast mašinske nule (=0)

Pokretni zarez dvostruke tačnosti



$R_{min} = 0.22 \cdot 10^{-307}$ - minimalna vrednost

$R_{max} = 0.18 \cdot 10^{309}$ - maksimalna vrednost

Binarni kodovi za predstavljanje numeričkih podataka

Zašto binarni kodovi?

- Normalni postupak obrade podataka u računaru obuhvata:
  - Prevodjenje ulaznih podataka iz dekadnog brojnog sistema u binarni,
  - Obradu,
  - Prevod rezultata iz binarnog brojnog sistema u dekadni i prikaz korisniku

Zašto binarni kodovi?

- Prevod iz dekadnog brojnog sistema u binarni i obrnuto se ne može uvek izvršiti bez greške,
- Ako je obrada koja se vrši mala, više vremena se troši na prevodjenje podataka nego na samu obradu,
- Ljudi su navikli na rad sa dekadnim brojnim sistemom pa je normalno zadržati primenu ovog sistema kad god je to moguće.

Šta je to BCD kod?

- Binarno-kodirani dekadni brojni sistem
- Skraćenica BCD – Binary Coded Decimal
- Svaka dekadna cifra u broju se nezavisno kodira nizom binarnih cifara.
- Minimalna dužina kodne reci za kodiranje 10 različitih dekadnih cifara 4 je $\lceil \log_2 10 \rceil$.

Težinski BCD kodovi

- Kod težinskih kodova svaka pozicija u kodnoj reči ima svoju težinu (slično težinama u pozicionim brojnim sistemima).
- Najkorišćeniji je takozvani "prirodni" BCD kod – težinski kod sa težinama "8421" (težine su stepeni dvojke)

Težinski BCD kodovi

d	8421	2421	5421	5211	4221	3321
0	0000	0000	0000	0000	0000	0000
1	0001	0001	0001	0001	0001	0001
2	0010	0010	0010	0011	0010	0010
3	0011	0011	0011	0101	0011	0011
4	0100	0100	0100	0111	1000	0101
5	0101	1011	1000	1000	0111	1010
6	0110	1100	1001	1001	1100	1100
7	0111	1101	1010	1011	1101	1101
8	1000	1110	1011	1101	1110	1110
9	1001	1111	1100	1111	1111	1111

Prirodni BCD kod - primer

- Predstaviti broj 27.125 u prirodnom BCD kodu.

0010 0111. 0001 0010 0101

Komplementarni BCD kodovi

- Kodovi kod kojih su kodne reči za svake dve cifre čiji je zbir jednak 9 medjusobno komplementarni
  - Svaki od njih se može dobiti iz onog drugog komplementiranjem svake binarne pozicije posebno

Komplementarni BCD kodovi

d	"visak 3"	2421
0	0011	0000
1	0100	0001
2	0101	0010
3	0110	0011
4	0111	0100
5	1000	1011
6	1001	1100
7	1010	1101
8	1011	1110
9	1100	1111

Još neki bitni BCD kodovi

- Grejov kod
  - Koristi se kod analogno-digitalnih pretvarača i ulazno-izlaznih uredjaja.
  - Kodovi za dve susadne dekadne cifre se uvek razlikuju samo na jednoj poziciji
- Hafmenov kod
  - Kod sa otkrivanjem i izpravljanjem gresaka. Pozicije u kodu su AB8C421 gde su cifre A, B i C kontrolni bitovi.

Grejov i Hafmenov kod

d	Grejov kod	Hafmenov kod
0	0000	0000000
1	0001	1101001
2	0011	0101010
3	0010	1000011
4	0110	1001100
5	0111	0100101
6	0101	1100110
7	0100	0001111
8	1100	1110000
9	1000	0011001