2. Brojevni sustavi i kodovi

Pregled tema

- Tipovi i prikaz podataka
- Brojevni sustavi
- Zbrajanje i odbijanje prirodnih binarnih brojeva
- Prikaz brojeva u modulu
- Binarna aritmetika
- Binarno kodiranje znamenki i simbola
- Kodovi za otkrivanje i ispravljanje pogrešaka

Tipovi i prikaz podataka

- prikaz podataka u digitalnom obliku
~ niz bitova, *bitovni vektor*
- značenja bitovnog vektora:
  - broj
  - znak/simbol
  - specijalni znakovi:
upravljački, *instrukcije*, …

Tipovi i prikaz podataka

- *bitovni vektor* ~ "tipiziran":
  - pripada nekom *tipu podataka* (engl. data type)
  - nametanje *discipline manipuliranja* s podacima
- osnovni tipovi podataka:
  - brojevi: prirodni, cijeli, realni, …
  - znak/simbol: pojedine abecede (~ *znakovni kodovi*)
  - specijalni znakovi ~ posebno značenje:
    logičke varijable
- značenje bitovnog vektora
  ~ utvrđeno *interpretacijom, kontekstom obrade*

Tipovi i prikaz podataka

- zapis podataka (~ zapis bitovnog vektora):
  utvrđeni oblik = *format*
  - organizacija niza bitova (grupe bitova ~ *polja*)
  - značenje pojedinih bitova/grupa bitova
- najjednostavniji zapis:
  prirodni binarni brojevi
  - vrijednost bita u broju = pozicija bita u binarnom vektoru
- posve općenito:
  pridruživanje značenja binarnom vektoru = *kôd*
  - broj
  - nešto drugo (~ simbol)

Pozicijski brojevni sustavi

- pozicija znamenke određuje njenu težinu
  - faktor kojim se znamenka množi
- težina - potencija *baze* brojevnog sustava
- dekadski sustav:

$$234 = 2 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0$$

- baza sustava može općenito biti bilo koji cijeli broj

Pozicijski brojevni sustavi - II dio

- Prikaz *n*-znamenkastih *cijelih* brojeva:

$$N_B = a_{n-1} \cdot B^{n-1} + a_{n-2} \cdot B^{n-2} + \cdots + a_1 \cdot B^1 + a_0 \cdot B^0$$
$$= \sum_{i=0}^{n-1} a_i \cdot B^i$$
$$= a_{n-1} a_{n-2} \ldots a_1 a_0$$

B: *baza* ili *korijen* brojevnog sustava

a_i: koeficijent uz *i*-tu potenciju (težinu);
a_i = {0, 1, …., B-1}, i = 0, 1, …., n-1

$\Rightarrow$ znamenke

Prikaz razlomljenih brojeva

- isti princip, potencije baze koje odgovaraju znamenkama iza zareza - negativne
- kod pretvorbe - posebno pretvoriti cjelobrojni a posebno razlomljeni dio broja

$$
\begin{aligned}
n_B &= a_{-1} \cdot B^{-1} + a_{-2} \cdot B^{-2} + \cdots + a_{-(m-1)} \cdot B^{-m+1} + a_{-m} \cdot B^{-m} \\
&= \sum_{i=-m}^{-1} a_i \cdot B^i \\
&= 0, a_{-1} a_{-2} \ldots a_{-(m-1)} a_{-m}
\end{aligned}
$$

Miješani ili racionalni brojevi

- prikaz s *fiksnim zarezom* [fixed-point notation]

$$
\begin{aligned}
N &= N_B + n_B \\
&= \sum_{i=-m}^{n-1} a_i \cdot B^i \\
&= a_{n-1} a_{n-2} \ldots a_1 a_0 , a_{-1} a_{-2} \ldots a_{-(m-1)} a_{-m}
\end{aligned}
$$

Neki brojevni sustavi

baza B	brojevni sustav	znamenke sustava (B)
2	binarni	0,1
3	ternarni	0,1,2
8	oktalni	0,1,2,3,4,5,6,7
10	dekadski	0,1,2,3,4,5,6,7,8,9
16	heksadekadski	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

dekadski	binarni	oktalni	heksadekadski
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Pretvorba brojeva u različitim sustavima

- pretvorba *cijelog* dekadskog broja u neki drugi sustav $\Rightarrow$ sukcesivno dijeljenje bazom tog sustava
  - ostaci dijeljenja s bazom - znamenke
  - ostatak prvog dijeljenja - najmanje značajna znamenka

$$N_{10} = a_{n-1}a_{n-2}\cdots a_1 a_0$$
$$N_2 = b_{s-1}b_{s-2}\cdots b_1 b_0$$

$$\begin{aligned} N_{10} &= b_{s-1}\cdot 2^{s-1} + b_{s-2}\cdot 2^{s-2} + \cdots + b_1\cdot 2^1 + b_0\cdot 2^0 \\ &= 2\cdot\left(b_{s-1}\cdot 2^{s-2} + b_{s-2}\cdot 2^{s-3} + \cdots + b_1\cdot 2^0\right) + b_0 \\ &= 2\cdot A_1 + b_0 \end{aligned}$$

Pretvorba dekadskog broja u binarni

Primjer

$$
\begin{array}{rl}
345:2=172 & 1 \\
172:2=86 & 0 \\
86:2=43 & 0 \\
43:2=21 & 1 \\
21:2=10 & 1 \\
10:2=5 & 0 \\
5:2=2 & 1 \\
2:2=1 & 0 \\
1:2=0 & 1
\end{array}
$$

$$\Rightarrow \quad 345_{10} = 101011001_2$$

Pretvorba dekadskog broja u ternarni

Primjer

$345 : 3 = 115$	0
$115 : 3 = 38$	1
$38 : 3 = 12$	2
$12 : 3 = 4$	0
$4 : 3 = 1$	1
$1 : 3 = 0$	1

$$\Rightarrow \quad 345_{10} = 110210_3$$

Pretvorba dekadskog broja u heksadekadski

Primjer

$$\begin{array}{rl} 345:16=21 & 9 \\ 21:16=1 & 5 \\ 1:16=0 & 1 \end{array} \uparrow$$

$$\Rightarrow \qquad 345_{10}=159_{16}$$

Pretvorba necijelog dekadskog broja

- pretvorba razlomljenog dijela dekadskog broja u sustav s nekom drugom bazom - uzastopnim *množenjem* s bazom sustava

Primjer

Pretvoriti dekadski broj 0,625 u binarni sustav.

$$
\begin{array}{llll}
0{,}625 * 2 & = 1{,}25 = 1 + 0{,}25 & \rightarrow & 1 \\
0{,}25 * 2 & = 0{,}5 \quad = 0 + 0{,}5 & \rightarrow & 0 \\
0{,}5 * 2 & = 1 \quad\ \ = 1 + 0 & \rightarrow & 1
\end{array}
\Bigg\downarrow
$$

$$\Rightarrow 0{,}625_{10} = 0{,}101_2$$

- ne mora uvijek završiti konačnim brojem znamenaka !!

Pretvorba u dekadski sustav

- izravno - odrediti dekadski zapis svake potencije baze izvornog sustava, pomnožiti vrijednost svake znamenke s odgovarajućom težinom, sumirati

Primjer

Pretvoriti binarni broj 10010,101 u dekadski sustav.

$$
\begin{aligned}
10010{,}101_2 &= 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 0*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3} \\
&= 1*16 + 1*2 + 1*0{,}5 + 1*0{,}125 = 18{,}625
\end{aligned}
$$

$$\Rightarrow 10010{,}101_2 = 18{,}625_{10}$$

Rekurzivno množenje i pribrajanje

- umjesto računanja potencija baze i množenjem sa znamenkama i pribrajanja - posmak za 1 mjesto i pribrajanje, za svaku znamenku

$$
\begin{array}{ccccccccc}
 & 2^{s-1} & 2^{s-2} & 2^{s-3} & 2^{s-4} & 2^{s-5} & \cdots & 2^{1} & 2^{0} \\
 & b_{s-1} & b_{s-2} & b_{s-3} & b_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
\hline
= & \multicolumn{2}{c}{(2 \cdot b_{s-1} + b_{s-2})} & b_{s-3} & b_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
 & & s_{s-2} & b_{s-3} & b_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
= & & \multicolumn{2}{c}{(2 \cdot s_{s-2} + b_{s-3})} & b_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
 & & & s_{s-3} & b_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
= & & & \multicolumn{2}{c}{(2 \cdot s_{s-3} + b_{s-4})} & b_{s-5} & \cdots & b_{1} & b_{0} \\
 & & & & s_{s-4} & b_{s-5} & \cdots & b_{1} & b_{0} \\
 & & & & \multicolumn{2}{c}{(2 \cdot s_{s-4} + b_{s-5})} & \cdots & b_{1} & b_{0} \\
 & & & \cdots & & & \cdots & b_{1} & b_{0}
\end{array}
$$

Rekurzivno množenje i pribrajanje

- osnovni korak: $s_{s-1} = a_{s-1}$
- korak rekurzije: $s_{i-1} = 2 \cdot s_i + a_{i-1}$

$$s_{s-1} = a_{s-1}$$

$$s_{s-2} = 2 \cdot s_{s-1} + a_{s-2}$$

$$= 2 \cdot a_{s-1} + a_{s-2}$$

$$s_{s-3} = 2 \cdot s_{s-2} + a_{s-3}$$

$$= 2^2 \cdot a_{s-1} + 2^1 \cdot a_{s-2} + a_{s-3}$$

$$s_{s-s} = 2^{s-1} \cdot a_{s-1} + \cdots + 2^{s-s} \cdot a_{s-s}$$

$$= \sum_{0}^{s-1} a_i \cdot 2^i$$

1	1	0	1
-	2	6	12
1	3	6	13

Rekurzivno množenje i pribrajanje

Primjer

Metodom rekurzivnog množenja i pribrajanja pretvoriti broj 10011101 u dekadski sustav.

$$(((1*2*2*2 + 1)*2 + 1)*2 + 1)*2*2 + 1 =$$
$$((9*2 + 1)*2 + 1)*2*2 + 1 = (19*2 + 1)*2*2 + 1 =$$
$$39*2*2 + 1 = 157$$

- postupak vrijedi za cijele brojeve
- razlomljeni dio: rekurzivno *dijeljenje* i pribrajanje

Usporedba brojevnih sustava

- Povećanjem baze sustava smanjuje se broj brojnih mjesta

Baza sustava	Broj 11_{10}
2	1011
3	201
8	13
10	11
>16	B

Optimalna baza brojevnog sustava

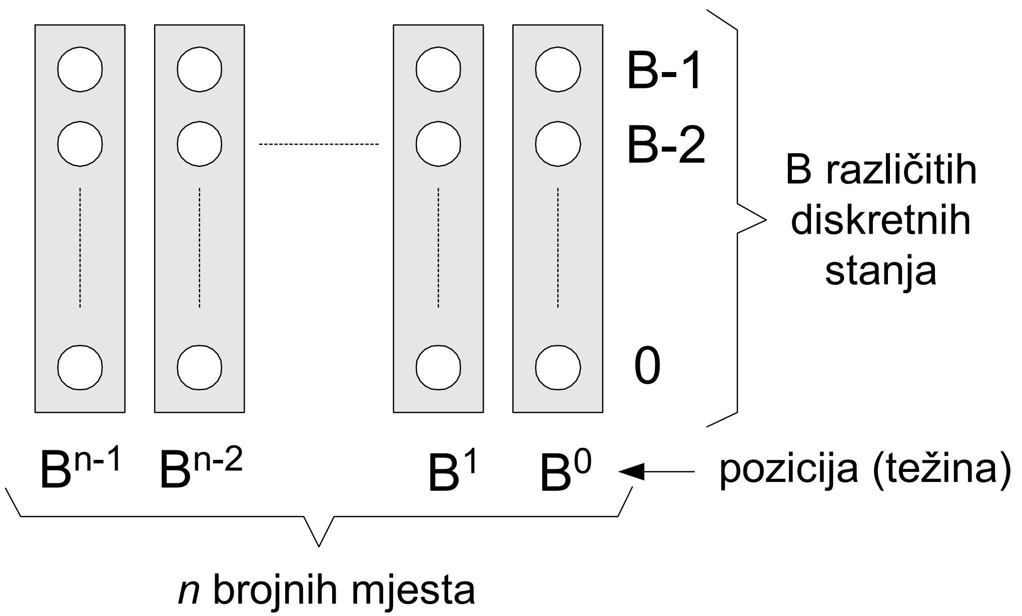
- prikladnost brojevnog sustava za fizičku realizaciju ne odgovara njegovoj prikladnosti za čovjekovu upotrebu
- prikaz znamenki elektroničkim sklopom:
  ➔ toliko različitih diskretnih stanja koliko ima znamenki

Optimalna baza brojevnog sustava

- *n*-znamenkasti broj u sustavu s bazom B

  N: broj mogućih n-znamenkastih brojeva u brojevnom sustavu s bazom B: "kapacitet" *n* pozicija

  v: ukupni broj različitih diskretnih stanja



Optimalna baza brojevnog sustava

$$N = B^n$$

$$v = n \cdot B$$

$$\ln N = n \cdot \ln B$$

$$v = \frac{\ln N}{\ln B} \cdot B$$

$$\frac{dv}{dB} = \ln N \cdot \frac{d}{dB}\left(\frac{B}{\ln B}\right)$$

$$= \ln N \cdot \frac{\ln B - B \cdot \frac{1}{B}}{\left(\ln B\right)^2}$$

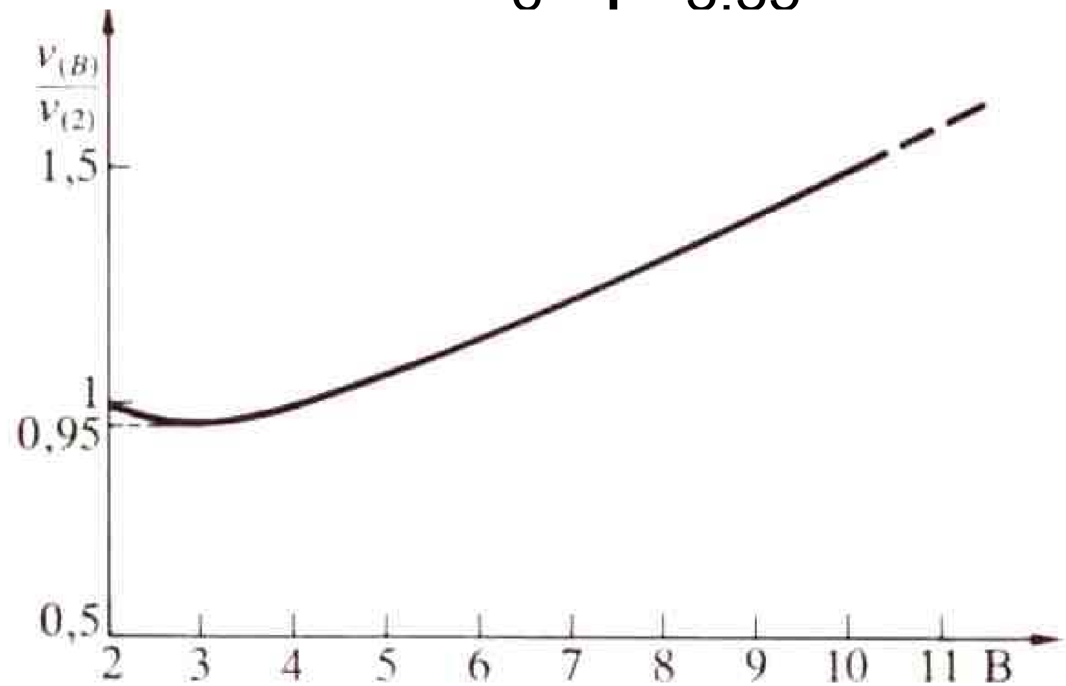
$$= \ln N \cdot \frac{\ln B - 1}{\left(\ln B\right)^2}$$

$$\frac{dv}{dB} = 0 \quad \Rightarrow \quad \ln B = 1 \quad \Rightarrow \quad B = e = 2,7183$$

Optimalna baza brojevnog sustava

- "najekonomičnija" baza: B = 3
  ➔ ternarni brojevni sustav:
  najbliži teorijskom minimumu
- binarni brojevni sustav:
  ➔ lakša realizacija:
  tehnički bolji,
  a samo 5% lošiji
  od ternarnog

B	B/lnB
2	2,89
e	e (2,718)
3	2,73
4	2,89
5	3,11
6	3,35



Oktalni i heksadekadski sustav

- pozicijski brojevni sustavi, baza 8 odnosno 16
- baza je potencija broja 2 ⇒ jednostavna pretvorba u binarni sustav
- veća baza ⇒ manji broj znamenaka za zapis broja
- oktalni sustav - znamenke 0-7 predstavljaju se s 3 bita

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Oktalni i heksadekadski sustav

Primjer

Pretvoriti broj 101111011001100_2 u oktalni sustav.

101	111	011	001	100
5	7	3	1	4

$$101111011001100_2 = 57314_8$$

Primjer

Pretvoriti broj 765432_8 u binarni sustav.

7	6	5	4	3	2
111	110	101	100	011	010

$$765432_8 = 111110101100011010_2$$

Heksadekadski sustav

- baza sustava 16, znamenke 0 - "15", tj. 0-9, A, B,..., F
- svaka znamenka predstavljena s 4 bita
- jednostavna pretvorba, vrlo raširen brojevni sustav kao sažeti zapis binarnog
- 2 heksa znamenke ~ 1 oktet

0	0000	A	1010
1	0001	B	1011
...		C	1100
7	0111	D	1101
8	1000	E	1110
9	1001	F	1111

Heksadekadski sustav

Primjer

Pretvoriti broj 01011110001110011100_2 u heksadekadski sustav.

0101	1110	0011	1001	1100
5	E	3	9	C

$$01011110001110011100_2 = 5E39C_{16}$$

Primjer

Pretvoriti broj $76A4C2_{16}$ u binarni sustav.

7	6	A	4	C	2
0111	0110	1010	0100	1100	0010

$$76A4C2_{16} = 011101101010010011000010_2$$

Binarna aritmetika

- binarna aritmetika
  ~ aritmetičke operacije u binarnom sustavu
  (zbrajanje, odbijanje, množenje, ...)
  - specifičnosti u odnosu na dekadsku aritmetiku
  - binarno zbrajanje
    ~ osnovna operacija u digitalnim sustavima (računalima)

Binarna aritmetika

- binarno zbrajanje
  - najjednostavnije
    ~ zbrajanje *dviju* binarnih znamenki:
    suma *mod 2* : operator ⊕

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +1 \\ \hline 10 \end{array}$$

C: prijenos

S: suma

b \ a	0	1
0	0	1
1	1	10

  - rezultat: 2_{10} = **1**0_2
    ~ pojava *prijenosa* (engl. carry) na višu bitovnu poziciju
  - oznake:
    S : suma, zbroj ; C : prijenos

Binarna aritmetika

- binarno zbrajanje dvaju binarnih *brojeva* :
  - općenito *n*-bitni binarni *brojevi*
  - prijenos pribrojiti višoj bitovnoj poziciji
    ~ zbrajanje *triju* binarnih znamenki

$$
\begin{array}{llccc l}
 & & 3 & 7 & 8 & \\
 & + & & 2 & 7 & \\
\hline
1. & & 3 & 9 & 5 & :S \\
 & + & & 1 & & :C \\
\hline
2. & & 3 & 0 & 5 & :S \\
 & + & 1 & & & :C \\
\hline
 & & 4 & 0 & 5 &
\end{array}
$$

$$
\begin{array}{c ccccccccc l}
 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & \\
\oplus & & & & & 1 & 1 & 0 & 1 & 1 & \\
\hline
 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & S_1 \\
\oplus & & & & 1 & 1 & & 1 & & & C_1 \\
\hline
 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & S_2 \\
\oplus & & & 1 & & & & & & & C_2 \\
\hline
 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & S_3 \\
\oplus & & 1 & & & & & & & & C_4 \\
\hline
 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 &
\end{array}
$$

Binarna aritmetika

- binarno zbrajanje dvaju binarnih *brojeva* :
  - *n*-bitni binarni *brojevi*
    ~ općenito promatrati *i*-ti bit

    $S_i = A_i \oplus B_i \oplus C_{i-1}$

    $C_i = ?$
  - posebna tablica zbrajanja

    Primjer : prethodni

$$\begin{array}{r} 1\;0\;1\;1\;1\;1\;0\;1\;0 \\ +\qquad\quad 1\;1\;0\;1\;1 \\ \hline 1\;1\;0\;0\;1\;0\;1\;0\;1 \end{array}$$

?

A_i	B_i	C_{i-1}	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Binarna aritmetika

- binarno odbijanje dvaju binarnih *znamenki* :
  - diferencija = minuend – suptrahend

minuend	0	1	1	0
suptrahend	−0	−0	−1	−1
	0	1	0	1 1

C: posudba

D: diferencija

b \ a	0	1
0	0	1
1	11	0

Binarna aritmetika

- binarno odbijanje dvaju binarnih *brojeva* :
  - *n*-bitni binarni *brojevi*
    ~ općenito promatrati *i*-ti bit
  - diferencija = suma !!!

    $D_i = A_i \oplus B_i \oplus C_{i-1}$

    $C_i = ?$
  - posebna tablica odbijanja
  - stvarna izvedba
    ~ pribrajanje komplementa broja
    (vidi kasnije)

A_i	B_i	C_{i-1}	D_i	C_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Prikaz brojeva u modulu

- digitalni sustavi (računala):
  - pohranjivanje brojeva u *registrima*

D_{n-1}	D_{n-2}	D_{n-3}	......	D_2	D_1	D_0

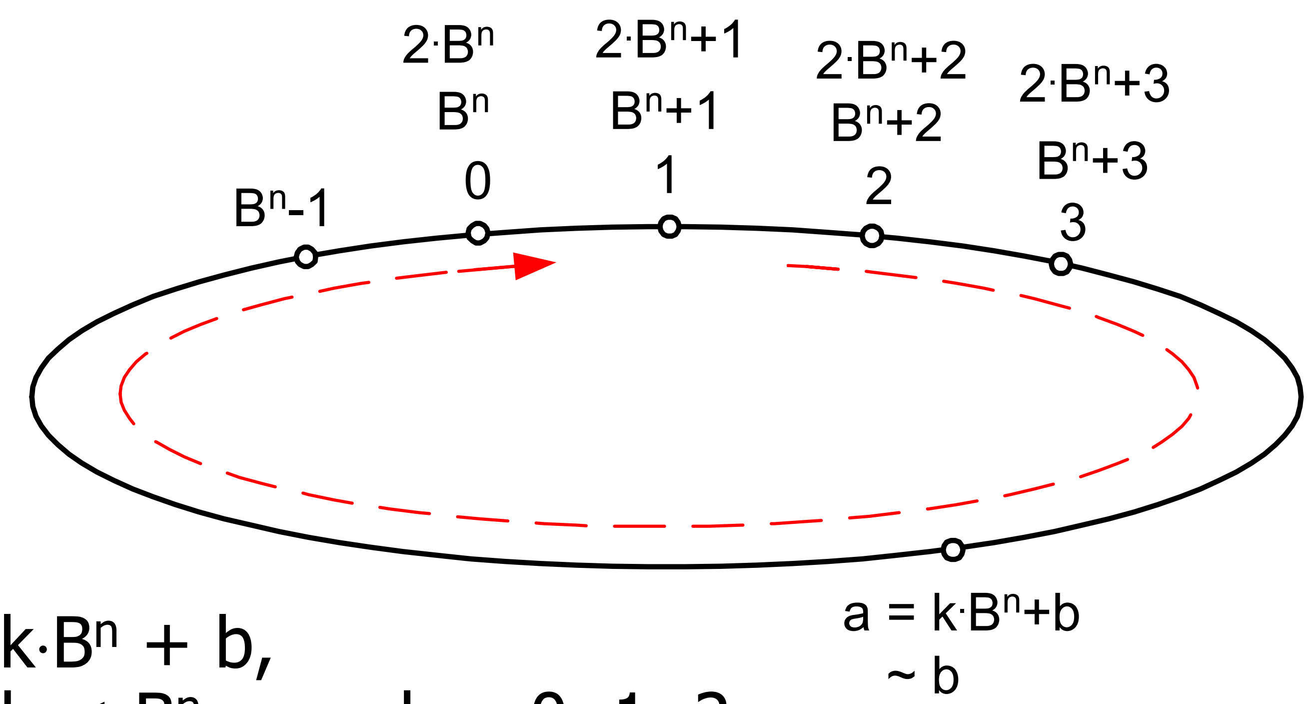
  - ograničeni broj mjesta
    ~ *n*-znamenkasti brojevi
  - broj mogućih *n*-znamenkastih brojeva
    kod baze B:

    $B^n = m$: *modul* ~ broj stanja registra,
    "kapacitet" registra od n mjesta

    $W = B^n - 1$: najveći *n*-znamenkasti broj

Prikaz brojeva u modulu

- prikaz *n*-znamenkastih brojeva:
  - ograničenje na brojeve $< m = B^n$
  - grafički prikaz ~"brojna kružnica"



- uočiti: $a = k \cdot B^n + b,$
  $b < B^n = m,\ k = 0, 1, 2, \ldots$

  $b = a \pmod{m}$

Prikaz brojeva u modulu

- prikaz *n*-znamenkastih brojeva:
  - interpretacija relacije

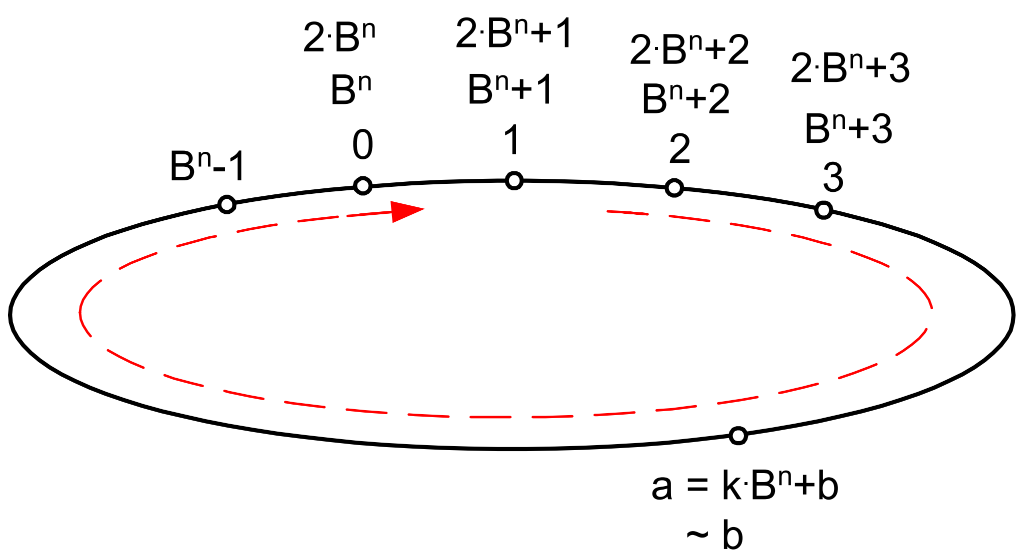
    b = a (mod m)

    "b je ostatak dijeljenja broja a s modulom m"

    Primjeri :

    23 mod 17 = 6

    35 mod 16 = 3



Modulo-aritmetika

- umjesto jednakosti, relacija ***kongruencije***, $\equiv$
- npr. za m = 10:

$$1 \equiv 1 \equiv 11 \equiv -9 \equiv 21 \equiv -19 \equiv \dots$$

- općenito: $a \equiv a + k \cdot 10, \quad k = \dots -2, -1, 0, 1, 2, \dots$
- primjeri zbrajanja i oduzimanja mod 10:

$$4 + 5 \equiv 9 \equiv 19 \equiv -1 \equiv \dots \qquad 5 - 4 \equiv 1 \equiv 11 \equiv -9 \equiv \dots$$

$$5 + 5 \equiv 0 \equiv 10 \equiv -10 \equiv \dots \qquad 5 - 5 \equiv 0 \equiv 10 \equiv -10 \equiv \dots$$

$$6 + 5 \equiv 1 \equiv 11 \equiv -9 \equiv \dots \qquad 5 - 6 \equiv 9 \equiv 19 \equiv -1 \equiv \dots$$

Modulo-aritmetika

- zapis: radi jasnoće na kraju izraza se piše $(\mathrm{mod}\ m)$
- npr. $5 \equiv 15 \pmod{10}$
- algebarski izrazi, npr:

$$a \equiv b + 2 \pmod{10}$$

- jednadžbu zadovoljavaju:

$$a = b + 2,\ b - 8,\ b + 12,\ b - 18,\ \ldots$$

Komplementi brojeva

- komplementi brojeva:
  - u odnosu na modul brojevnog sustava $m = B^n$
    ~ u odnosu na broj mjesta n za prikaz brojeva u registru
  - u odnosu na najveći n-znamenkasti broj $W = B^n - 1$
- značaj komplementa brojeva:
  - pojednostavljivanje obavljanja aritmetičkih operacija
  - npr. korištenje istog sklopovlja
    za obavljanje zbrajanja i odbijanja

Komplement

- svaki broj a, $0 \le a < m$, ima svoj **komplement**: $\overline{a}$
  - definicijska relacija:

$$a + \overline{a} = m$$

- srodan pojmu *suprotnog* broja ($-a$):

$$a + \left(-a\right) = 0$$

$$a + \overline{a} \equiv 0 \pmod{m}$$

Komplement

- korist od komplementa:
  oduzimanje pretvara u zbrajanje!

$$a - b = a - b + 0 \equiv a - b + \left(b + \overline{b}\right) = a + \overline{b}$$

$$a - b \equiv a + \overline{b}$$

- omogućuje korištenje istog sklopovlja
  za zbrajanje i oduzimanje

Komplement

- ***B*-komplement**: naziv za već prikazani komplement u odnosu na $m=B^n$:

$$\overline{a}_B = m - a = B^n - a$$

  - (n je maksimalan broj znamenki koji se koristi)

- B = 10: *10-komplement*

  n= 2: $\overline{(35)}_{10} = 10^2 - 35 = 65$

  n= 3: $\overline{(35)}_{10} = 10^3 - 35 = 965$

- B = 2: *2-komplement*

$$\overline{(010101)}_2 = 2^6 - 010101 = 1000000 - 010101 = 101011$$

Komplementi brojeva

- praktični algoritam za dobivanje 2-komplementa:

  "Počev od najmanje značajnog bita broja,
  invertirati svaki bit nakon prve 1."

  Primjer:

  00010110 $\rightarrow$ **11101**010

  00100101 $\rightarrow$ **1101101**1

Komplementi brojeva

- (B-1)-komplement
  ~ komplement u odnosu na W

  $$\overline{N} \equiv B^n - N - 1 = \overline{N_B} - 1 = W - N$$

  - B = 10: *9-komplement*

    n = 2: $\overline{(35)} = 10^2 - 35 - 10^0 = 64 = \left(10^2 - 10^0\right) - 35 = 99 - 35$

    n = 3: $\overline{(35)} = 10^3 - 35 - 10^0 = 964 = \left(10^3 - 10^0\right) - 35 = 999 - 35$

  - B = 2: 1-komplement

$$\overline{(010101)} = 2^6 - 010101 - 1 = 111111 - 010101 = 101010$$

Komplementi brojeva

- dobivanje (B-1)-komplementa:
  - svaku znamenku broja odbiti od W = B – 1
  - dobivanje 1-komplementa
    ~ komplementiranje pojedinih bitova:
    vrlo jednostavna sklopovska izvedba!
- dobivanje 2-komplementa iz 1-komplementa:

$$\overline{B_2} = \overline{B_1} + 1$$

- u odnosu na B-komplement je kod (B-1)-komplementa znamenka najmanje težine umanjena za 1

Oduzimanje komplementom

- uzmimo da naše sklopovlje podržava:
  - zbrajanje
  - inverziju svih bitova u broju
- **problem**: izračunati $D = M - S$
- možemo izračunati komplemente:
  - 1-komplement = inverzija svih bitova
  - 2-komplement = 1-komplement + 1
- u nastavku: oduzimanje komplementom u bilo kojoj bazi B

Oduzimanje B-komplementom

- Oduzimanje B-komplementom: sklopovlje izračunava $M + \bar{S}_B$

$$M + \bar{S}_B = M + \left(B^n - S\right) = \left(M - S\right) + B^n = D + B^n$$

$$D = \left(M + \bar{S}_B\right) - B^n$$

$$D \equiv M + \bar{S}_B$$

- u sklopovskoj izvedbi razlikujemo dva slučaja:

  1. $M > S \;\Rightarrow\; D > 0$
  2. $M < S \;\Rightarrow\; D < 0$

Odbijanje pomoću komplementa

- odbijanje pomoću komplementa
  ~ *pribrajanje* minuendu *komplementa suptrahenda*
  - općeniti algoritam za brojevni sustav s bazom B
  - odbijanje pomoću (B-1)-komplementa
  - odbijanje pomoću B-komplementa
- dva slučaja:
  - $M > S \Rightarrow D > 0$
  - $M < S \Rightarrow D < 0$

Oduzimanje B-komplementom

- slučaj 1: $M > S \;\Rightarrow\; D > 0$

  $$M + \bar{S}_B = M + B^n - S = D + B^n = D + W + 1 > W$$

- rezultat je veći od najvećeg prikazivog broja, W
  - dolazi do **preljeva**
  - u registru je rezultat kojem nedostaje najviša znamenka
    - njena težina: B^n
  - u registru je dakle $M + \bar{S}_B - B^n \equiv M + \bar{S}_B$
    - preljev narušava jednakost, ali ne i kongruenciju!
  - sadržaj registra je upravo traženi rezultat:

$$\left(M + \bar{S}_B\right) - B^n = \left(D + B^n\right) - B^n = D$$

Oduzimanje *B*-komplementom

- primjer: B=2, n=8 (8-bitno binarno računalo)

$$W = B^n - 1 = 2^8 - 1 = 256 - 1 = 255$$

- izračunati 3–2, dakle M=3, S=2

$$\overline{S}_B = B^n - S = 256 - 2 = 254$$

$$M + \overline{S}_B = 3 + 254 = 257 > W$$

preljev!

$$257 \equiv 1$$

broj koji dobiva sklopovlje

Oduzimanje *B*-komplementom

- registri: A=3, B=2

$$A: 00000011 \qquad B: 00000010 \qquad \overline{B}_2 = 11111110$$

$$A + \overline{B}_2: \quad \begin{array}{r} \boxed{00000011} \\ + \boxed{11111110} \\ \hline 1\ \boxed{00000001} \end{array}$$

traženi rezultat

ne stane u registar – preljev!

složenost posla:
2x zbrajanje
1x inverzija bitova

Oduzimanje B-komplementom

- slučaj 2: $M < S \;\Rightarrow\; D < 0$

$$M + \bar{S}_B = D + B^n = D + W + 1 \leq W$$

- nema preljeva, sklopovlje dobiva $M + \bar{S}_B$
- oduzimanje B^n od rezultata: uzeti mu komplement i tome staviti predznak minus:

$$\begin{aligned} D &= \left(M + \bar{S}_B\right) - B^n \\ &= -\left(B^n - \left(M + \bar{S}_B\right)\right) \\ &= -\overline{\left(M + \bar{S}_B\right)}_B \end{aligned}$$

Oduzimanje *B*-komplementom

- izračunati 2–3, dakle M=2, S=3

$$\overline{S}_B = B^n - S = 256 - 3 = 253$$

$$M + \overline{S}_B = 2 + 253 = 255 \leq W$$

nema preljeva, sklopovlje dobiva upravo 255

$$D \equiv 255$$

$$D = 255 - B^n = 255 - 256 = -1$$

$$D = -\overline{(255)}_2 = -(1)$$

Oduzimanje B-komplementom

- registri: A=2, B=3

$A: 00000010 \quad B: 00000011 \quad \overline{B}_2 = 11111101$

$$
\begin{array}{lr}
A+\overline{B}_2: & \boxed{00000010} \\
+ & \boxed{11111101} \\
\hline
 & \boxed{11111111}
\end{array}
\longleftarrow \text{novi sadržaj registra A}
$$

$$-\overline{(11111111)}_2 = -00000001$$

traženi rezultat

složenost posla:
3x zbrajanje
2x inverzija bitova

Odbijanje pomoću komplementa

- *algoritam* oduzimanja B-komplementom:
  - pribrojiti minuendu komplement suptrahenda
  - ako se pojavi preljev, to je rezultat
  - ako nema preljeva, još jednom komplementirati te promijeniti predznak

Oduzimanje (B-1)-komplementom

- oduzimanje (B-1)-komplementom: sklopovski izračunavamo $M + \overline{S}_{B-1}$

$$M + \overline{S}_{B-1} = M + \left(W - S\right) = \left(M - S\right) + W = D + W$$

$$D = \left(M + \overline{S}_{B-1}\right) - W = M + \overline{S}_{B-1} - \left(B^n - 1\right) = M + \overline{S}_{B-1} - B^n + 1$$

$$D \equiv M + \overline{S}_{B-1} + 1$$

Oduzimanje (B-1)-komplementom

- slučaj 1: $M > S \;\Rightarrow\; D > 0$

$$M + \overline{S}_{B-1} = D + W > W$$

- dolazi do preljeva u izračunu, registar sadrži

$$M + \overline{S}_{B-1} - B^n \equiv M + \overline{S}_{B-1}$$

- dobivanje rezultata:

$$D = M + \overline{S}_{B-1} - W = M + \overline{S}_{B-1} - B^n + B^n - W$$
$$= \left(M + \overline{S}_{B-1} - B^n\right) + 1$$

- dakle, registru treba dodati 1
  - "**povratni preljev**" (eng. *end-around carry*) jer bit prijenosa dodajemo sadržaju registra

Oduzimanje (B-1)-komplementom

- izračunati 3–2, dakle M=3, S=2

$$\overline{S}_{B-1} = W - S = 255 - 2 = 253$$

$$M + \overline{S}_{B-1} = 3 + 253 = 256 > W \Rightarrow \text{preljev!}$$

$$D = 256 - 256 + 1 = 0 + 1 = 1$$

Oduzimanje (B-1)-komplementom

- registri: A=3, B=2
- strojna naredba: **`SUB A,B`**

A: 00000011   B: 00000010 $\overline{B}_1 = 11111101$

$A + \overline{B}_1$:

$$\begin{array}{lr} & 00000011 \\ + & 11111101 \\ \hline \text{preljev} \quad 1 & 00000000 \end{array}$$

povratni prijenos:

$$\begin{array}{lr} & 00000000 \\ + & 00000001 \\ \hline \text{rezultat} \rightarrow & 00000001 \end{array}$$

složenost posla:
2x zbrajanje
1x inverzija bitova

Oduzimanje (B-1)-komplementom

- slučaj 2: $M < S \;\Rightarrow\; D < 0$

$$M + \overline{S}_{B-1} = D + W < W$$

- *nema* preljeva
- dobivanje rezultata kao i kod B-komplementa, samo radimo (B-1)-komplement:

$$D = \left(M + \overline{S}_{B-1}\right) - W = -\left(W - \left(M + \overline{S}_{B-1}\right)\right) = -\overline{\left(M + \overline{S}_{B-1}\right)}_{B-1}$$

Oduzimanje (B-1)-komplementom

- izračunati 2–3, dakle M=2, S=3

$$\overline{S}_{B-1} = W - S = 255 - 3 = 252$$

$$M + \overline{S}_{B-1} = 2 + 252 = 254 < W \Rightarrow \text{nema preljeva!}$$

$$D = 254 - 255 = -1$$

$$D = -\overline{(254)}_{B-1} = -(1)$$

Oduzimanje (B-1)-komplementom

- registri: A=2, B=3
- strojna naredba: **`SUB A,B`**

$A: 00000010 \quad B: 00000011 \quad \overline{B}_1 = 11111100$

$$A + \overline{B}_1 : \quad \begin{array}{r} |\overline{\underline{00000010}}| \\ + |\overline{11111100}| \\ \hline |\overline{\underline{11111110}}| \end{array}$$

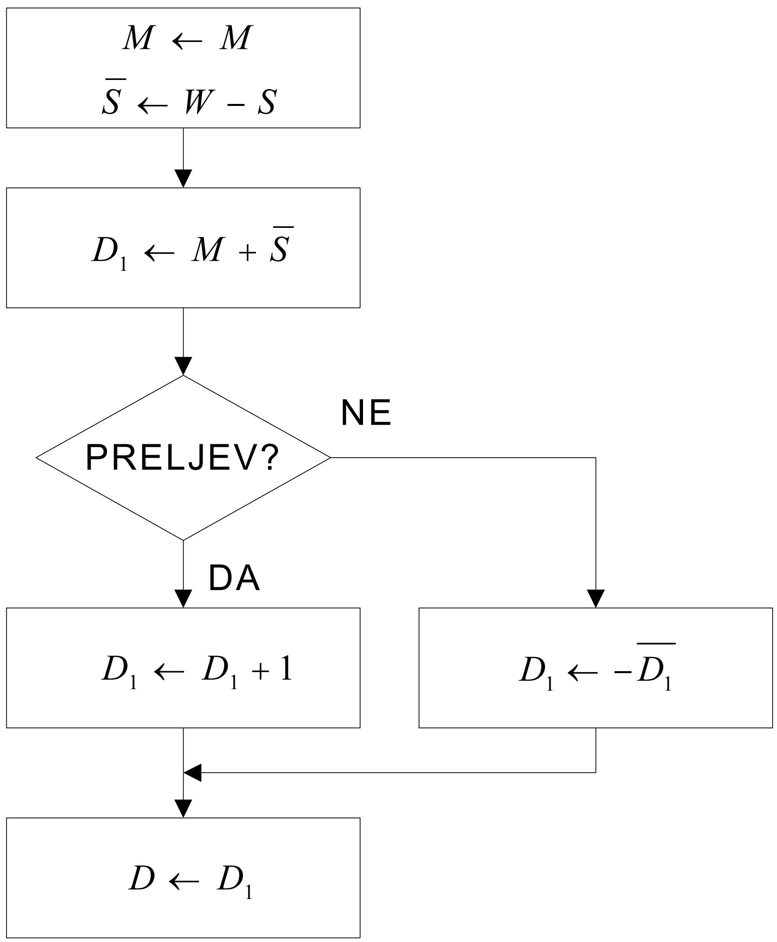
$$-\overline{(11111110)}_1 = -00000001$$

traženi rezultat

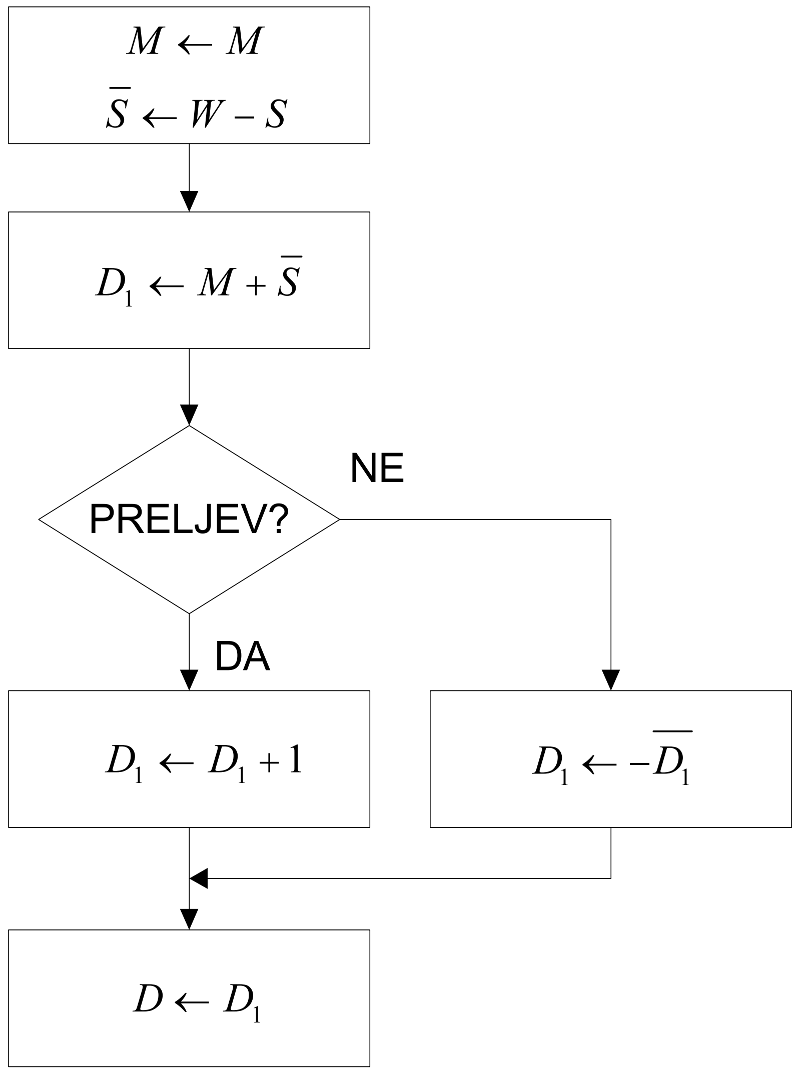
složenost posla:
1x zbrajanje
2x inverzija bitova

Oduzimanje pomoću komplementa

- algoritam oduzimanja (B-1)-komplementom:
  - pribrojiti minuendu komplement suptrahenda
  - ako se pojavi preljev, dodati 1 ~ povratni prijenos
  - ako nema preljeva, još jednom komplementirati te promijeniti predznak



Oduzimanja (B-1)-komplementom



Oduzimanje pomoću komplementa

- oduzimanje (B-1)-komplementom:

Primjer: oduzimanje 1-komplementom

$$
\begin{array}{lcccccc}
M=11 & \equiv & & 1 & 0 & 1 & 1 \\
S=9 & \equiv & - & 1 & 0 & 0 & 1 \\
\hline
 & & & 1 & 0 & 1 & 1 \\
 & & + & 0 & 1 & 1 & 0 \\
\hline
\text{povratni} & & \boxed{1} & 0 & 0 & 0 & 1 \\
\text{prijenos} & & + & & & & \boxed{1} \\
\hline
 & & & 0 & 0 & 1 & 0
\end{array}
\qquad
\begin{array}{lcccccc}
M=9 & \equiv & & 1 & 0 & 0 & 1 \\
S=11 & \equiv & - & 1 & 0 & 1 & 1 \\
\hline
 & & & 1 & 0 & 0 & 1 \\
 & & + & 0 & 1 & 0 & 0 \\
\hline
 & & & 1 & 1 & 0 & 1 \\
\hline
 & & - & 0 & 0 & 1 & 0
\end{array}
$$

Operacije nad brojevima s predznakom

- zapis brojeva s predznakom:
  - veličina broja ~ iznos
  - *predznak*
    ~ još jedan bit: najznačajniji (najlijeviji) bit
  - tipično:
    0 : "+"
    1 : "-"

D_n	D_{n-1}	D_{n-2}	......	D_2	D_1	D_0

↑ bit predznaka

- prikaz negativnih brojeva:
  - predznak i veličina
  - predznak i 2-komplement
  - predznak i 1-komplement

Prikaz brojeva s predznakom

- prikaz brojeva predznakom i veličinom :
  - odvojeno manipuliranje predznaka i veličine
  - relativno složeno izvođenje računskih operacija
  - problem "negativne nule"

Primjer: prikaz jednim oktetom

$$+63 = 0011\ 1111 \qquad -63 = 1011\ 1111$$

$$+114 = 0111\ 0010 \qquad -114 = 1111\ 0010$$

$$+0 = 0000\ 0000 \qquad -0 = 1000\ 0000$$

Prikaz brojeva s predznakom

- prikaz brojeva predznakom i 2-komplementom :
  - pozitivni brojevi: predznak i veličina
  - negativni brojevi: predznak i 2-komplement
  - komplementiranje predznaka i veličine zajedno
  - nema problema "negativne nule"
    ~ nula je jedinstvena !

  Primjer: prikaz jednim oktetom

$$+63 = 0011\ 1111 \qquad -63 = 1100\ 0001$$

$$+114 = 0111\ 0010 \qquad -114 = 1000\ 1110$$

$$+0 = 0000\ 0000 \qquad -0 = 0000\ 0000$$

Prikaz brojeva s predznakom

- prikaz brojeva predznakom i 1-komplementom :
  - slično prikazu predznakom i 2-komplementom
    ~ komplementiranje predznaka i veličine zajedno
  - problem "negativne nule"

Primjer: prikaz jednim oktetom

$$+63 = 0011\ 1111 \qquad -63 = 1100\ 0000$$

$$+114 = 0111\ 0010 \qquad -114 = 1000\ 1101$$

$$+0 = 0000\ 0000 \qquad -0 = 1111\ 1111$$

Usporedba 1 i 2 komplementa

- prikaz predznakom i 2-komplementom praktičniji!
  - nema "negativne nule"
  - asimetrični raspon pozitivnih i negativnih brojeva
    ~ nula je "pozitivna"

broj	predznak i veličina	2-komplement	1-komplement
-8	-	1000	-
-7	1111	1001	1000
-6	1110	1010	1001
-5	1101	1011	1010
-4	1100	1100	1011
3	1011	1101	1100
-2	1010	1110	1101
-1	1001	1111	1110
0	0000 ili 1000	0000	0000 ili 1111
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	0101	0101
6	0110	0110	0110
7	0111	0111	0111

Zbrajanje u 2-komplementu

- pojava aritmetičkog preljeva (engl. arithmetic overflow):
  - pribrojnici istog predznaka (+ ili –),
    a predznak rezultata se razlikuje (– ili +)
  - suma premašuje broj mjesta veličine (n-1)
- potreba detekcije aritmetičkog preljeva

Primjer: 7 + 4

	0111	+7
+	0100	+4
	1011	-5

POGREŠNO

	00111	+7
+	00100	+4
	01011	+1
		1

Oduzimanje u 2-komplementu

- od suptrahenda dobiti 2-komplement ~ promjena predznaka!
  - 2-komplement suptrahenda pribrojiti minuendu

Primjer:

$$
\begin{array}{rr}
 & 1011 \\
- & 1001 \\
\hline
 & 1011 \\
+ & 0111 \\
\hline
\color{red}{1} & 0010
\end{array}
\qquad\qquad
\begin{array}{rr}
 & 1001 \\
- & 1011 \\
\hline
 & 1001 \\
+ & 0101 \\
\hline
 & 1110
\end{array}
$$

preljev se zanemaruje

nema preljeva! (rezultat je negativan: 1 na najznačajnijem mjestu)

Množenje

- binarno množenje
  ~ prema pravilima za dekadsko množenje:

$$
\begin{array}{cccccccccc}
 & & & \multicolumn{3}{c}{\text{multiplikand}} & & \multicolumn{3}{c}{\text{multiplikator}} \\
 & & & 1 & 1 & 0 & \times & 0 & 1 & 0 \\
\cline{3-10}
 & & & 0 & 0 & 0 & & \multicolumn{3}{l}{\text{parcijalni}} \\
 & & 1 & 1 & 0 & & & \multicolumn{3}{l}{\text{produkti}} \\
+ & 0 & 0 & 0 & & & & & & \\
\cline{1-6}
 & 0 & 1 & 1 & 0 & 0 & & \multicolumn{3}{l}{\text{produkt}}
\end{array}
$$

Množenje

- mogućnosti ostvarivanja binarnog množenja:
  - uzastopna zbrajanja
  - parcijalna množenja s 2 (~ "posmak") i zbrajanje

$M = m_3 m_2 m_1 m_0$ → multiplikand

$N = n_3 n_2 n_1 n_0$ → multiplikator

$$M \times N = M \cdot \left(n_3 \cdot 2^3 + n_2 \cdot 2^2 + n_1 \cdot 2^1 + n_0 \cdot 2^0\right)$$

$$= M \cdot n_3 \cdot 2^3 + M \cdot n_2 \cdot 2^2 + M \cdot n_1 \cdot 2^1 + M \cdot n_0 \cdot 2^0$$

$$= \sum_{i=0}^{3} M \cdot n_i \cdot 2^i$$

  - efikasnije primjenom Hornerove sheme:

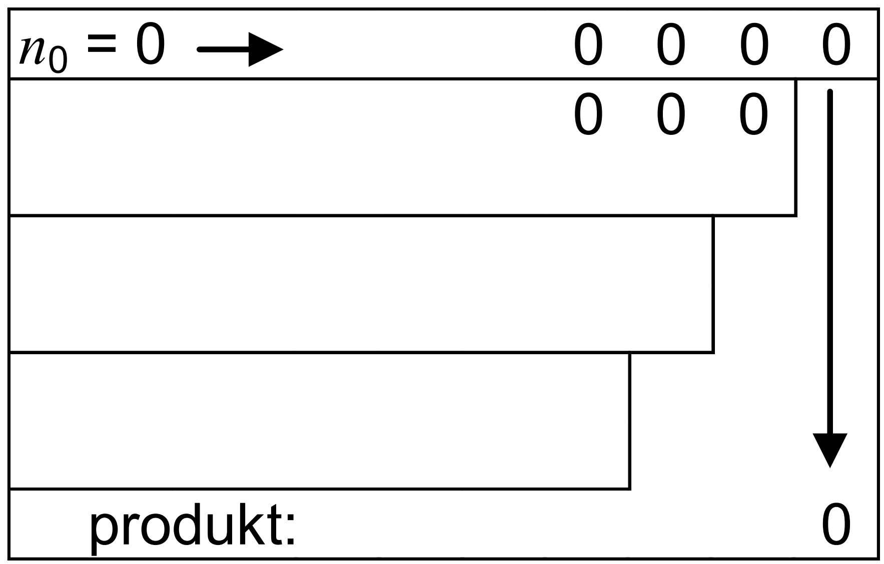
$$M \times N = \left(\left(M \cdot n_3 \cdot 2 + M \cdot n_2\right) \cdot 2 + M \cdot n_1\right) \cdot 2 + M \cdot n_0,\ n_i \in \{0,1\}$$

Množenje i dijeljenje

$$M = 1011_2 \equiv 11_{10}$$

$$N = 1010_2 \equiv 10_{10}$$

$$P = M \times N = 1101110_2 \equiv 110_{10}$$

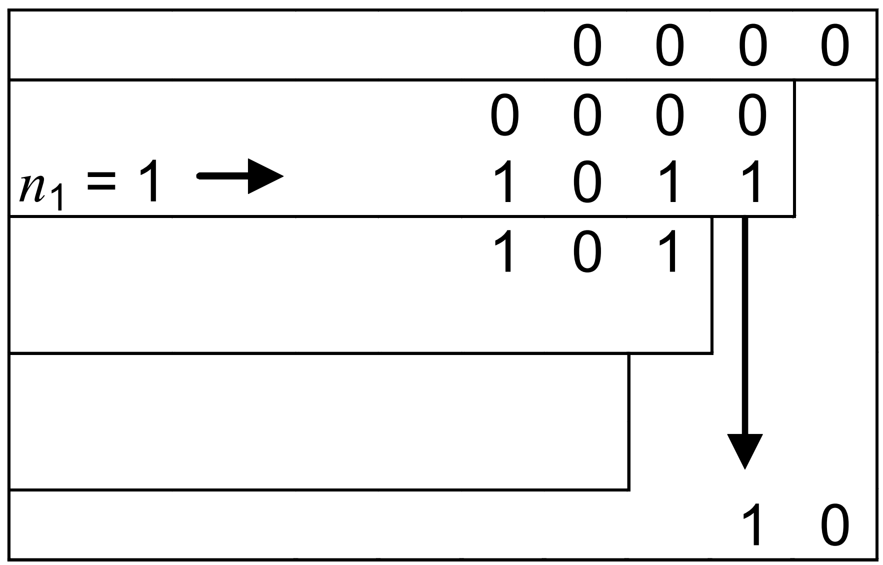


Množenje i dijeljenje

$$M = 1011_2 \equiv 11_{10}$$

$$N = 1010_2 \equiv 10_{10}$$

$$P = M \times N = 1101110_2 \equiv 110_{10}$$

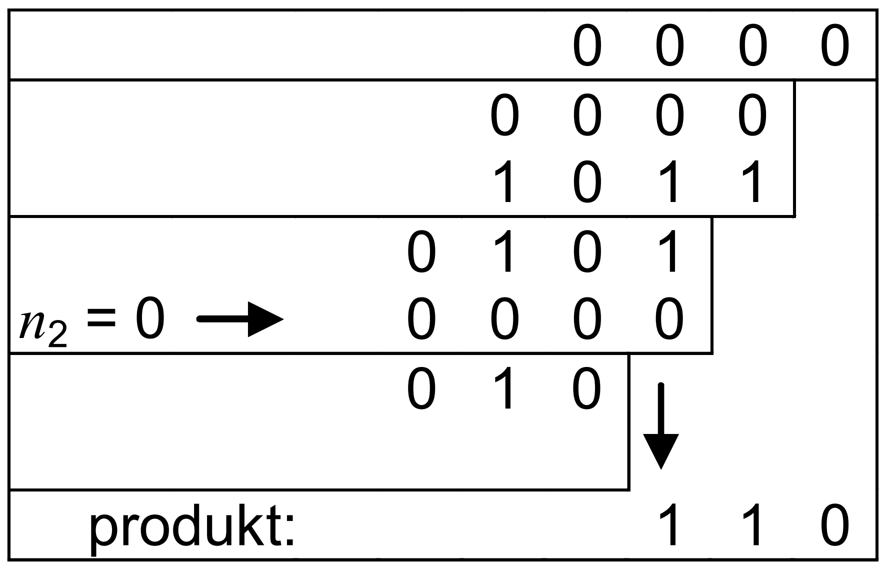


Množenje i dijeljenje

$$M = 1011_2 \equiv 11_{10}$$

$$\underline{N = 1010_2 \equiv 10_{10}}$$

$$P = M \times N = 1101110_2 \equiv 110_{10}$$



Množenje i dijeljenje

$$M = 1011_2 \equiv 11_{10}$$

$$\underline{N = 1010_2 \equiv 10_{10}}$$

$$P = M \times N = 1101110_2 \equiv 110_{10}$$

				0	0	0	0
			0	0	0	0	
			1	0	1	1	
		0	1	0	1		
		0	0	0	0		
	0	0	1	0			
n_3 = 1 $\longrightarrow$	1	0	1	1			
produkt:	1	1	0	1	1	1	0

Množenje i dijeljenje

- Primjer:

$$M = 1011_2 \equiv 11_{10}$$

$$N = 1001_2 \equiv 9_{10}$$

$$P = M \times N = 1100011_2 \equiv 99_{10}$$

$n_0 = 1 \rightarrow$				1	0	1	1
			0	1	0	1	
$n_1 = 0 \rightarrow$			0	0	0	0	
		0	0	1	0		
$n_2 = 0 \rightarrow$		0	0	0	0		
	0	0	0	1			
$n_3 = 1 \rightarrow$	1	0	1	1			
produkt:	1	1	0	0	0	1	1

Množenje i dijeljenje

- binarno množenje:

Primjer : množenje 4-bitnih brojeva

$$M = 1011_2 \equiv 11_{10}$$

$$N = 1001_2 \equiv 9_{10}$$

$$P = M \times N = 01100011_2 \equiv 99_{10}$$

$$P = M \times N = \left(\left(M \cdot n_3 \cdot 2 + M \cdot n_2\right) \cdot 2 + M \cdot n_1\right) \cdot 2 + M \cdot n_0 , n_i \in \{0,1\}$$

$n_0 = 1$ →				1	0	1	1
				1	0	1	
$n_1 = 0$ →			0	0	0	0	
			0	1	0		
$n_2 = 0$ →		0	0	0	0		
		0	0	1			
$n_3 = 1$ →	1	0	1	1			
produkt:	1	1	0	0	0	1	1

Dijeljenje

- binarno dijeljenje
  ~ svodi se na *uzastopno oduzimanje*
- primjer: **57 : 5 = 11, ostatak 2**
  - 1. korak: djelitelj potpišemo lijevo ispod djeljenika i oduzmemo
  - iza jednakosti stavljamo 1

$$
\begin{array}{rcccccccccc}
 & 1 & 1 & 1 & 0 & 0 & 1 & : & 1\,0\,1 & = & 1 \\
- & 1 & 0 & 1 & & & & & & & \\
\hline
 & & 1 & 0 & & & & & & &
\end{array}
$$

Dijeljenje

- 2. dopišemo sljedeću znamenku djeljenika
  - sad imamo 100, broj manji od djelitelja
  - iza jednakosti dodajemo nulu
  - ne radimo oduzimanje

$$
\begin{array}{rcccccccccccc}
 & 1 & 1 & 1 & 0 & 0 & 1 & : & 1 & 0 & 1 & = & 1 \; 0 \\
- & 1 & 0 & 1 & & & & & & & & & \\
\hline
 & & 1 & 0 & 0 & & & & & & & &
\end{array}
$$

Dijeljenje

- 3. dopišemo još jednu znamenku djeljenika
  - sad imamo 1000, veći od djelitelja
  - iza jednakosti dodajemo jedinicu
  - provodimo oduzimanje

$$
\begin{array}{rrrrrrcrrrcrrr}
 & 1 & 1 & 1 & 0 & 0 & 1 & : & 1 & 0 & 1 & = & 1 & 0 & 1 \\
- & 1 & 0 & 1 & & & & & & & & & & & \\
\hline
 & & 1 & 0 & 0 & 0 & & & & & & & & & \\
 & & - & 1 & 0 & 1 & & & & & & & & & \\
\hline
 & & & & 1 & 1 & & & & & & & & &
\end{array}
$$

Dijeljenje

- i tako dalje

$$
\begin{array}{ccccccc l}
 & 1 & 1 & 1 & 0 & 0 & 1 & : 101 = 1011 \\
- & 1 & 0 & 1 & & & & \\
\hline
 & & 1 & 0 & 0 & & & \\
\hline
 & & 1 & 0 & 0 & 0 & & \\
 & - & & 1 & 0 & 1 & & \\
\hline
 & & & & 1 & 1 & 1 & \\
 & & & - & 1 & 0 & 1 & \\
\hline
 & & & & & 1 & 0 & \longleftarrow \text{ostatak}
\end{array}
$$

Dijeljenje

- primjer: **38 : 5 = 7, ostatak 3**

$$
\begin{array}{rrrrrrrrrrrrr}
 & 1 & 0 & 0 & 1 & 1 & 0 & : & 1 & 0 & 1 & = & 1\;1\;1 \\
- & & 1 & 0 & 1 & & & & & & & & \\
\hline
 & & 1 & 0 & 0 & 1 & & & & & & & \\
 & - & & 1 & 0 & 1 & & & & & & & \\
\hline
 & & & 1 & 0 & 0 & 0 & & & & & & \\
 & & - & & 1 & 0 & 1 & & & & & & \\
\hline
 & & & 0 & 0 & 1 & 1 & & & & & &
\end{array}
$$

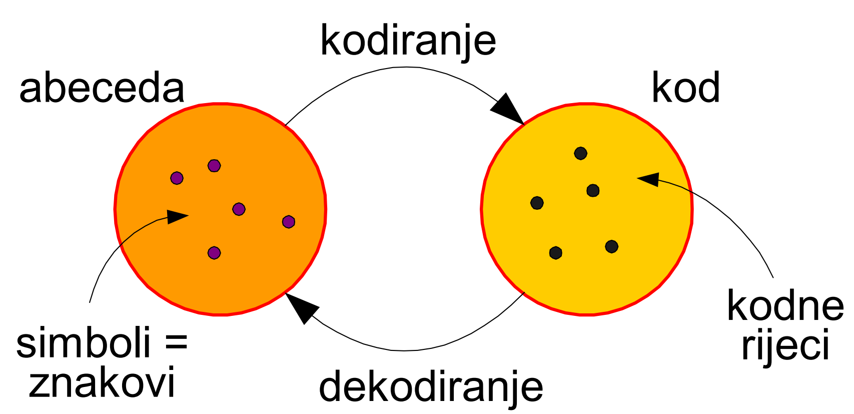
Dijeljenje

- binarno dijeljenje
  ~ svodi se na *uzastopno odbijanje*

$$
\begin{array}{ccccccccccccccc}
 & 1 & 1 & 1 & 0 & 0 & 1 & : & 1 & 0 & 1 & = & 1 & 0 & 1 & 1 \\
- & 1 & 0 & 1 & & & & & & & & & & & & \\
\hline
 & & 1 & 0 & 0 & & & & & & & & & & & \\
\hline
 & & 1 & 0 & 0 & 0 & & & & & & & & & & \\
 & & - & 1 & 0 & 1 & & & & & & & & & & \\
\hline
 & & & & 1 & 1 & 1 & & & & & & & & & \\
 & & & - & 1 & 0 & 1 & & & & & & & & & \\
\hline
 & & & & & 1 & 0 & & & & & & & & &
\end{array}
$$

Binarno kodiranje znamenki i simbola

- princip kodiranja binarnim riječima:
  - izražavanje simbola/znakova u *binarnom* obliku, radi dalje obrade digitalnim sklopom
    ~ binarno *kodiranje*
  - *kôd* : grupa simbola kojoj se dogovorno daje značenje
  - *kodna riječ* : niz bitova kojem se pridaje neko značenje
  - *abeceda* : skup svih simbola prikazanih kodnim riječima
  - *znakovi* : elementi abecede



Binarno kodiranje znamenki i simbola

- princip kodiranja binarnim riječima:
  - broj simbola = broj različitih prikaza
    $\rightarrow$ broj bitova kodnih riječi

$$K \text{ simbola: } n \geq \text{ld } K \text{ [bit]}, \text{ ld } x = \log_2 x$$

$$2^n \geq K$$

  - n bitova: $N = 2^n$ mogućih kombinacija

pridruživanje kodne riječi prvom simbolu	N načina
pridruživanje kodne riječi drugom simbolu	N-1 način
pridruživanje kodne riječi trećem simbolu	N-2 načina
…	…
pridruživanje kodne riječi K-tom simbolu	N-(K-1)

$$N \cdot (N-1) \cdot (N-2) \cdot \cdots \cdot (N-(K-1)) = \frac{N!}{(N-K)!} = V_N^{(K)}$$

Binarno kodiranje znamenki i simbola

- dekadski kodovi
  ~ binarni prikaz dekadskih znamenki
  - $n \geq 4$ bita; $2^3 < 10 < 2^4$
  - $n = 4$ bita
    ~ 16 kombinacija
  - broj 4-bitnih kodova
    ~ mogući broj kodiranja: $V_{16}^{(10)} = \frac{16!}{6!} = 29{,}059 \cdot 10^9$
  - odabrati kodove s povoljnim svojstvima!

Dekadski kodovi

- svojstva dekadskih kodova:
  - aditivnost
    ~ veza između kodne riječi
    i prikazane dekadske znamenke
  - samokomplementarnost
    (engl. self-complementing)
    ~ veza kodnih riječi po parovima

Dekadski kodovi

- težinski kod:
  - zbroj težina = vrijednost prikazane znamenke

$$N = \sum_{i=0}^{n-1} a_i \cdot w_i + D$$

N	:	dekadski ekvivalent
w_i	:	i-ta težina
a_i	:	koeficijent za i-tu težinu
D	:	konstanta pomaka

  - 17 težinskih kodova s pozitivnim težinama, 71 s jednom ili dvije negativne težine

Dekadski kodovi

- samokomplementirajući kod:
  ~ 9-komplement dekadskog broja
  zamjenom 0 i 1 u kodnoj riječi
  - korisno kod binarno-dekadske aritmetike
  - težinski je kod samokomplementirajući ako:

$$\sum_i w_i = 9$$

Dekadski kodovi

- kod 8421,
  BCD (engl. Binary Coded Decimal)
  - prvih 10 binarnih brojeva
  - težine: 8, 4, 2, 1
  - neupotrijebljene kombinacije:
    1001÷1111

	2^3	2^2	2^1	2^0
	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
	1	0	1	0
	1	0	1	1
	1	1	0	0
	1	1	0	1
	1	1	1	0
	1	1	1	1

Dekadski kodovi

- kod 2421 (Aikenov kod)
  - težinski kod
    ~ težine: 2, 4, 2, 1
  - samokomplementirajući kod:
    0-9, 1-8, 2-7, 3-6, 4-5
  - prvih i zadnjih pet
    4-bitnih brojeva
  - neupotrijebljene kombinacije:
    0101÷1010

	2	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
	0	1	0	1
	0	1	1	0
	0	1	1	1
	1	0	0	0
	1	0	0	1
	1	0	1	0
5	1	0	1	1
6	1	1	0	0
7	1	1	0	1
8	1	1	1	0
9	1	1	1	1

Dekadski kodovi

- kod XS-3 (Stibitzov kod)
  - kod 8421,
    s "prekoračenjem" (ekscesom) od 3
  - uz D = 3
    ~ težinski kod
  - ne postoji 0000:
    detekcije prekida kod prijenosa
  - neupotrijebljene kombinacije:
    0000÷0010, 1101÷1111
  - simetrična tablica koda
    ~ samokomplementirajući kod!

	2^3 8	2^2 4	2^1 2	2^0 1
	0	0	0	0
	0	0	0	1
	0	0	1	0
0	0	0	1	1
1	0	1	0	0
2	0	1	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0
8	1	0	1	1
9	1	1	0	0
	1	1	0	1
	1	1	1	0
	1	1	1	1

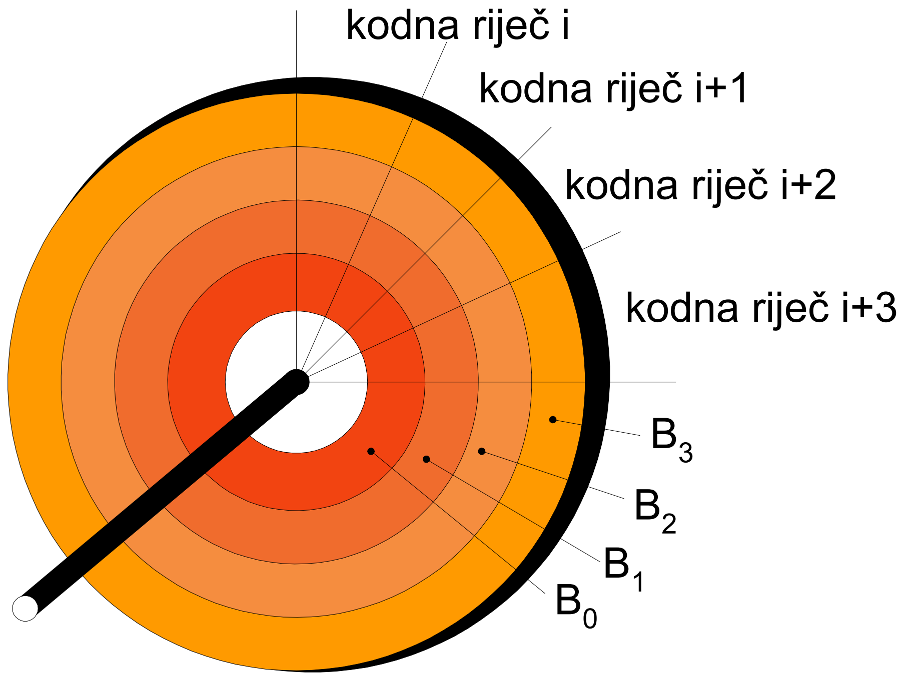
Dekadski kodovi

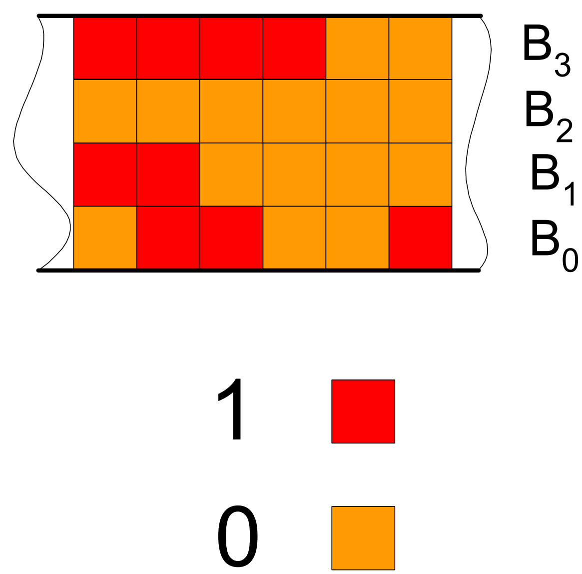
- bikvinarni kod
  - težinski 7-bitni kod (2+5=7)
  - kodne riječi s dvije 1:
    - otkrivanje pogrešaka
    - ne ako je pogreška samokompenzirajuća
  - velika zalihost: ~ 10 od 128 mogućih kombinacija

	5	0	4	3	2	1	0
0	0	1	0	0	0	0	1
1	0	1	0	0	0	1	0
2	0	1	0	0	1	0	0
3	0	1	0	1	0	0	0
4	0	1	1	0	0	0	0
5	1	0	0	0	0	0	1
6	1	0	0	0	0	1	0
7	1	0	0	0	1	0	0
8	1	0	0	1	0	0	0
9	1	0	1	0	0	0	0

Grayev kod

- kod s *minimalnom* promjenom
  - susjedne kodne riječi
    ~ razlika u samo 1 bitu
  - ograničavanje pogreški pri slijednoj promjeni
    npr. direktno očitavanje položaja





Grayev kod

- svojstva Grayevog koda:
  - susjedne kodne riječi
    ~ razlika u samo jednom bitu
    ("jedinična distanca")
  - izgradnja koda:
    ~ zrcaljenje u jednom
    bitovnom mjestu:
    reflektirani kod
  - netežinski kod
  - binarni, ali i "dekadski"
    ~ XS-3 Grayev kod

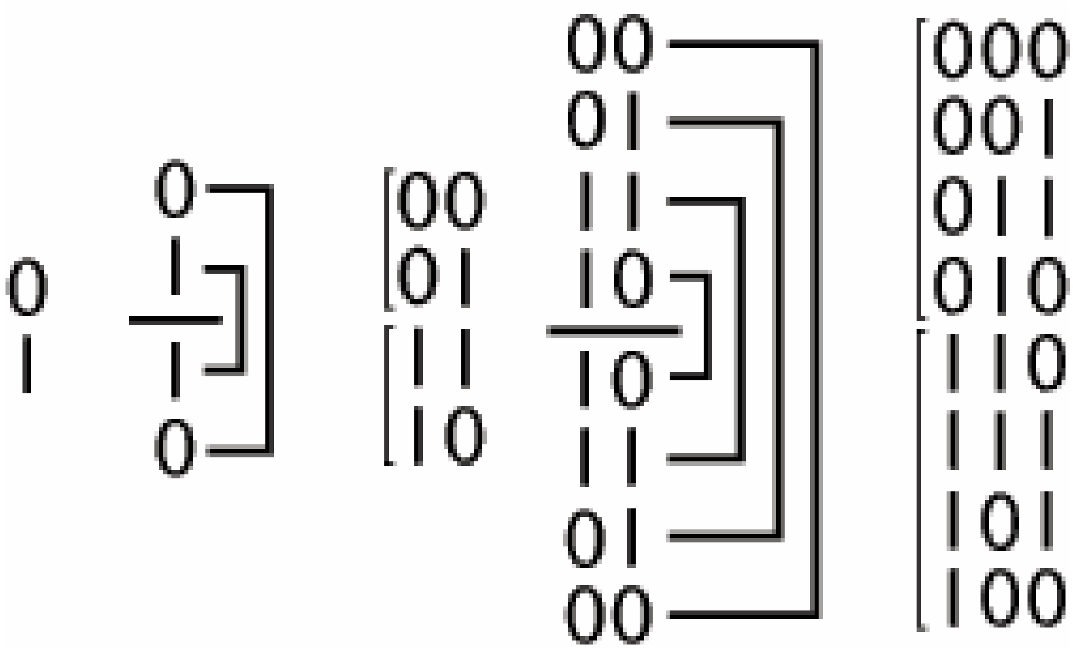
dekadski Grayev kod

binarni Grayev kod

	0	0	0	0
	0	0	0	1
	0	0	1	1
0	0	0	1	0
1	0	1	1	0
2	0	1	1	1
3	0	1	0	1
4	0	1	0	0
5	1	1	0	0
6	1	1	0	1
7	1	1	1	1
8	1	1	1	0
9	1	0	1	0
	1	0	1	1
	1	0	0	1
	1	0	0	0

Grayev kod

- izgradnja koda:
  ~ *zrcaljenje* u jednom bitovnom mjestu:
  reflektirani kod



Znakovni kodovi

- prikaz skupa znakova:
  - prikaz slova i znamenki:
    - "grafički"
      ~ "alfa-numerički" znakovi, interpunkcije, simboli, ...
    - *upravljački* znakovi
- standardizirani znakovni kodovi:
  npr. 7-bitni (128 kombinacija) ASCII:
  ISO IS 646, ITU-T/CCITT No. 5

Znakovni kodovi

- kod ASCII (engl. American Standard Code for Information Interchange):

  ' ': 20H, CR : 08H, LF : 0AH

  '0' – '9': 30-39H

  'A' - 'Z' : 41-5AH

  'a' – 'z' : 61-7AH

npr.

A = 100 0001 = 41H

a = 110 0001 = 61H

* = 010 1010 = 1AH

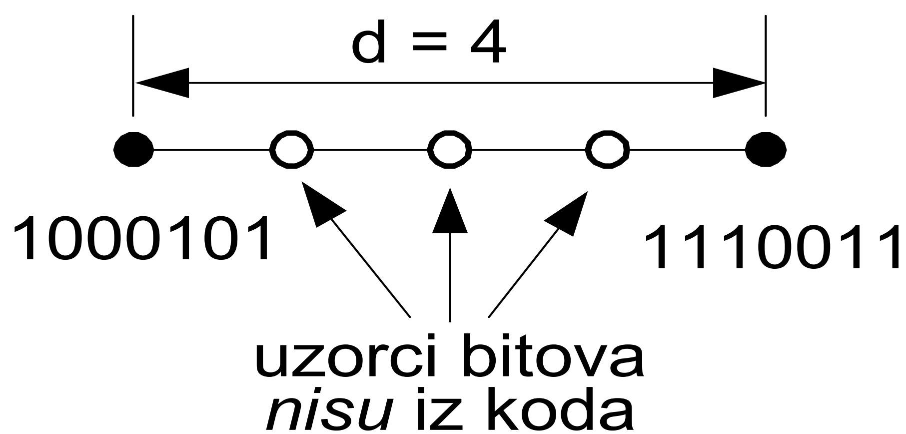
					0	0	0	0	1	1	1	1
					0	0	1	1	0	0	1	1
					0	1	0	1	0	1	0	1
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	3	P	3	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#/£	3	C	S	c	s
0	1	0	0	4	EOT	DC4	¤/$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	3	k	3
1	1	0	0	12	FF	IS4	,	<	L	3	l	3
1	1	0	1	13	CR	IS3	-	=	M	3	m	3
1	1	1	0	14	SO	IS2	.	>	N	3	n	3
1	1	1	1	15	SI	IS1	/	?	O	_	o	DEL

Kodovi za zaštitu podataka

- prijenos podataka
  ~ utjecaj smetnji: moguća pojava pogreške
  - pogreška
    ~ neželjena promjena jednog/više bitova u kodnoj riječi
    - jednostruka pogreška
      ~ promjena vrijednosti jednog bita
      $(0 \rightarrow 1$ ili $1 \rightarrow 0)$
    - višestruka pogreška ~ više bitova
  - rezultat
    ~ neispravna, ali i ispravna kodna riječ !
  - dobivena kodna riječ ispravna
    ~ nemoguće otkriti da je došlo do pogreške!!!

Kodovi za zaštitu podataka

- princip otkrivanja (i ispravljanja) pogrešaka
  ~ razlika kodnih riječi u ≥ 1 bita
- distanca kodnih riječi (R. W. Hamming)
  ~ "udaljenost" dviju kodnih riječi:
  - najmanji broj bitova u kojima se dvije kodne riječi razlikuju
  - broj bitova koje treba promijeniti
    da se jedna kodna riječ pretvori u drugu
    ~ pogreška ostaje neotkrivena !!!



Kodovi za zaštitu podataka

- računanje distance kodnih riječi
  - broj različitih bitovnih mjesta dviju kodnih riječi:

    $c = a \oplus b$ po bitovima

    d = aritmetička suma "1" u c
  - formalno:

    $c = a \oplus b = (a_{n-1} \oplus b_{n-1}, a_{n-2} \oplus b_{n-2}, \ldots, a_0 \oplus b_0)$

    $d = |c| = |a \oplus b|$

    $|x|$: težina kodne riječi (engl. weight) , broj jedinica u kodnoj riječi

Kodovi za zaštitu podataka

- minimalna distanca koda d_{min}
  ~ najmanji razmak između dvije kodne riječi
  - npr. kod 8421: $d_{min} = 1$
  - bikvinarni kod: $d_{min} = 2$
  - Grayev kod: $d_{min} = d = 1$
- kod pruža zaštitu od t pogrešaka

  $t=d_{min}-1$

  $d_{min} \geq (t+1)$

  Primjer: $d_{min} = 2$

  ~ otkrivanje jednostruke pogreške

Kodovi za zaštitu podataka

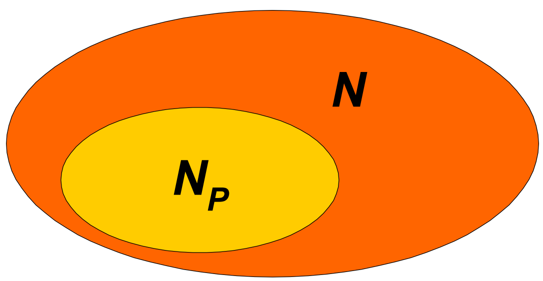
- kodovi s $d_{min} > 1$
  ~ postoji zalihost (redundancija), R:
  snaga zaštite, višak informacije

n : duljina kodne riječi
$k<n$: broj informacijskih bitova
$r=n-k$: broj zaštitnih bitova

$$R = \frac{r}{n} \qquad R = 1 - \frac{ldN_p}{ldN} \qquad (ldX = \log_2 X)$$

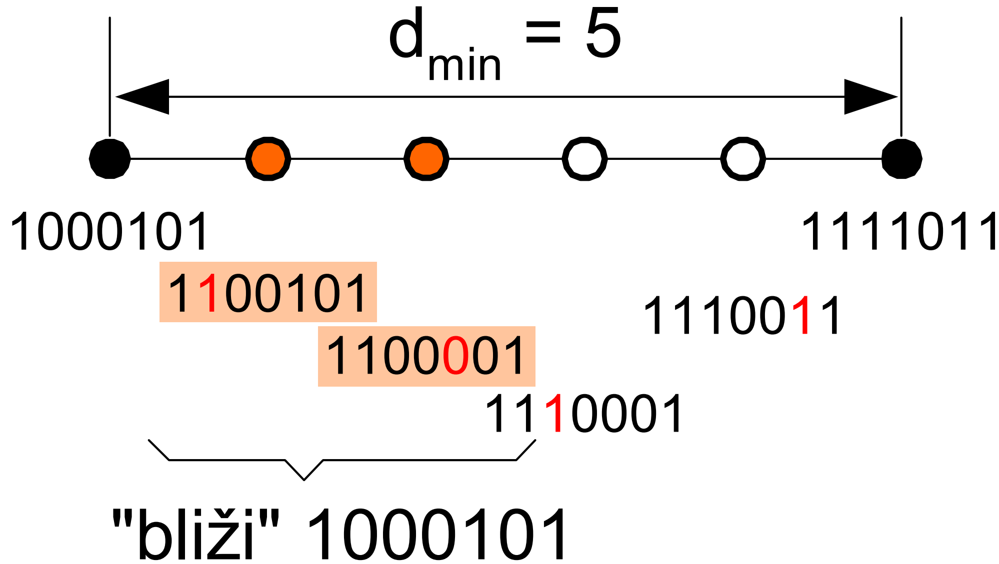
ukupni broj kodnih riječi $\quad N_p = 2^k < 2^n = N \quad$ ukupan broj mogućih kombinacija od n bitova

- veći broj bitova od minimalno potrebnih za prikaz informacije
  npr. bikvinarni kod
- kodna riječ = bitovi + zaštitni bitovi
- sistematski kodovi
  ~ zaštitni bitovi nakon informacijskih



Kodovi za zaštitu podataka

- dvije skupine zaštitnih kodova:
  - s mogućnošću otkrivanja pogrešaka
    ~ EDC (engl. Error Detecting Codes):
    $d_{min} \geq t + 1$ za otkrivanje t pogrešaka
  - s mogućnošću ispravljanja pogrešaka
    ~ ECC (engl. Error Correcting Codes):
    $d_{min} \geq 2 \cdot t + 1$ za ispravljanje t pogrešaka



Kodovi za zaštitu podataka

- geometrijski prikaz kodnih riječi/koda ~ kubusi u n-dimenzijskom prostoru
  - 0-kubus ~ točka
  - 1-kubus ~ dužina
  - 2-kubus ~ kvadrat
  - 3-kubus ~ kocka
  - n-kubus ~ "hiperkocka"

Kodovi za zaštitu podataka

- geometrijski prikaz kodnih riječi/koda

Primjer:

n-kubus $\rightarrow$ 3-kubus

1. za 2^n uzoraka: $d_{min} = 1$

2. za {100, 011}: $d_{min} = 3$

otkriva 2 pogreške:

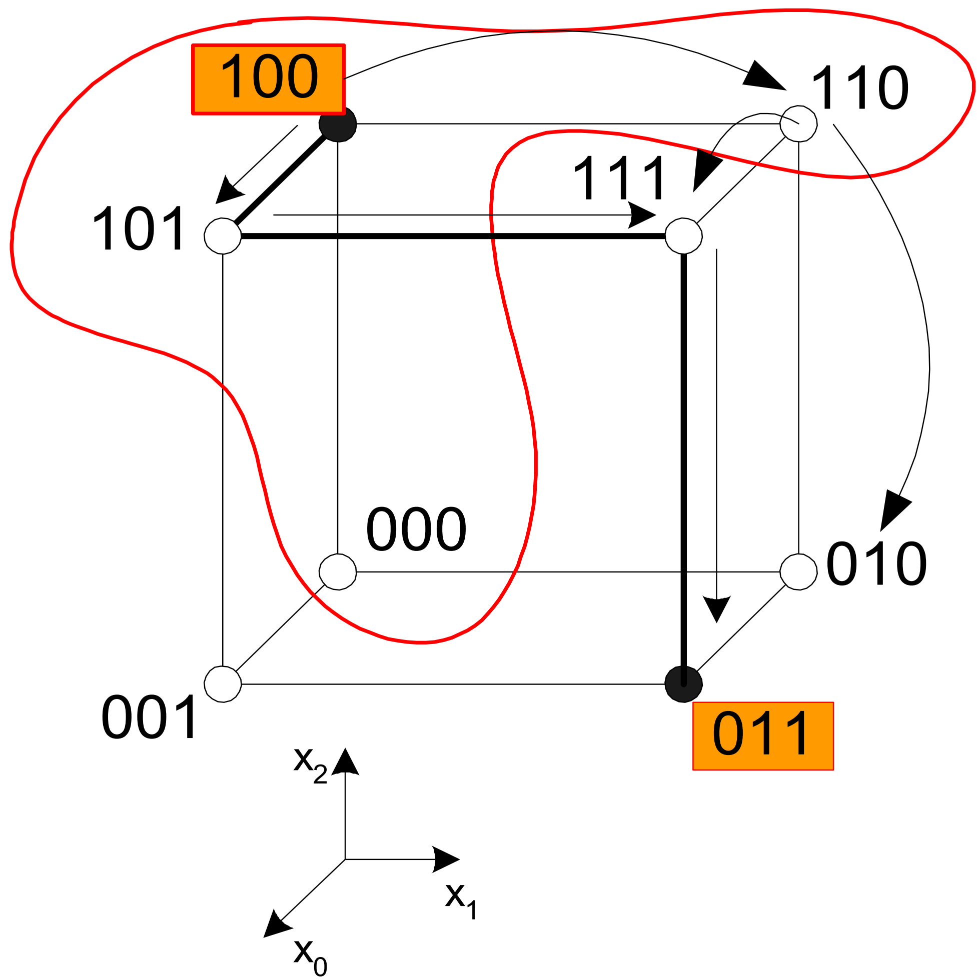
010, 111, 001

110, 101, 000

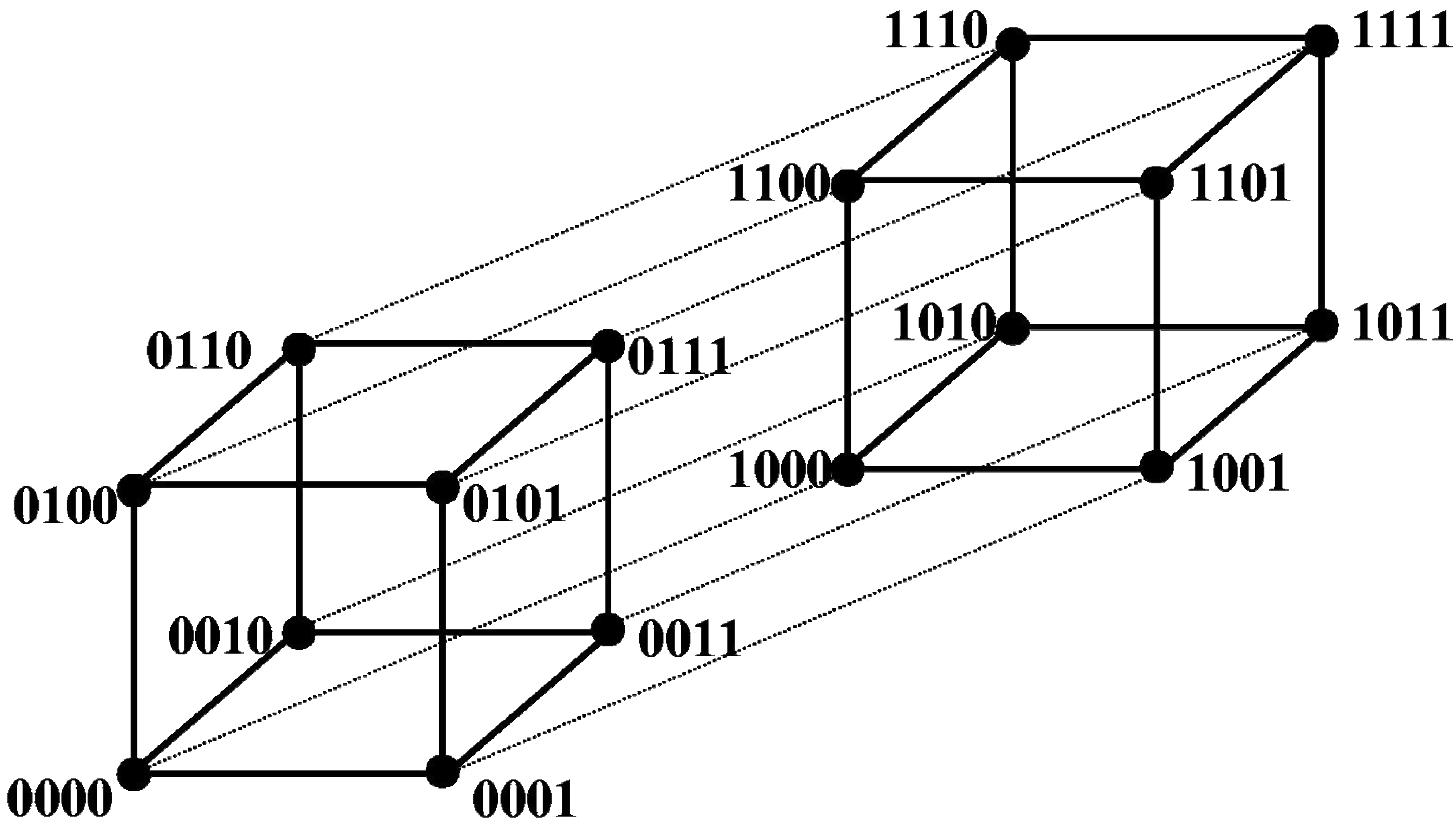
ispravlja 1 pogrešku:

110, 101, 000

001, 111, 010



Izgradnja 4 kubusa



Kodovi za zaštitu podataka

- paritet ~ najjednostavniji način zaštite
  - dodati paritetni bit
    ~ tipično osmi bit riječi iz ASCII koda:

    $$p\ b_6 b_5 b_4\ b_3 b_2 b_1 b_0$$

  - nova kodna riječ mora imati paran/neparan broj jedinica
    ~ paran/neparan paritet

ZNAK		PARITET	
		PARNI	NEPARNI
A	100 0001	0 100 0001	1 100 0001
a	110 0001	1 110 0001	0 110 0001
*	010 1010	1 010 1010	0 010 1010

  - "vertikalna" (okomita) paritetna zaštita
    (engl. Vertical Redundancy Check, VRC)
    ~ otkrivanje neparnih pogrešaka

Kodovi za zaštitu podataka

- višestruko ispitivanje pariteta :
  - zahtjev: povećati moć zaštite!
    - veći broj paritetnih ispitivanja
      ~ "nezavisna" (ortogonalna)
    - veći broj zaštitnih bitova
      ~ veća zalihost
  - više mogućnosti:
    - dvodimenzijski kod
    - Hammingov kod

Kodovi za zaštitu podataka

- dvodimenzijski kod
  ~ 2D matrica informacijskih bitova
  "pravokutni" kod
- uzdužna i poprečna paritetna zaštita:
  - kodna riječ ← paritetni bit
  - cijelom bloku kodnih riječi ←
    paritetna riječ, BCC (engl. Block Check Character)
    ~ "horizontalna" (uzdužna) paritetna zaštita
    (engl. Longitudinal Redundancy Check, LRC)
  - ispravljanje jednostruke pogreške

Kodovi za zaštitu podataka

- vodoravna i okomita paritetna zaštita:

  Primjer: kodne riječi iz ASCII koda

A	a	*	*BCC*
0	1	1	x
1	1	0	0
0	1	1	0
0	0	0	0
0	0	1	1
0	0	0	0
0	0	1	1
1	1	0	0

  - zalihost:

    R = 1/m + 1/n

    R = 0,125 + m-1 (ASCII)

  - komentar:

    ~ prevelika zalihost za relativno ograničenu moć zaštite!

Kodovi za zaštitu podataka

- uzdužna i poprečna paritetna zaštita:

  Primjer : kodne riječi iz ASCII koda

  - zalihost:
    - sa BCC

$$R = \frac{m+n+1}{(m+1)(n+1)}$$

    - bez BCC

$$R = \frac{m+n}{(m+1)(n+1)-1}$$

A	a	*	*BCC*
0	1	1	x
1	1	0	0
0	1	1	0
0	0	0	0
0	0	1	1
0	0	0	0
0	0	1	1
1	1	0	0

  - komentar:

    ~ prevelika zalihost za relativno ograničenu moć zaštite!

Kodovi za zaštitu podataka

- Hammingovi kodovi:
  - sustavni mehanizam za izgradnju
    niza kodova za ispravljanje pogrešaka
    ~ R.W. Hamming, 1950.
  - princip:
    ~ višestruko (nezavisno) paritetno ispitivanje
  - bolja efikasnost kodiranja
    ~ manja zalihost
    usp. dvodimenzijski kod
  - naročito popularan Hammingov kod
    za ispravljanje jednostruke pogreške
    ~ tipična primjena: memorijski sklopovi

Kodovi za zaštitu podataka

- Hammingovi kodovi :
  - nezavisna paritetna ispitivanja
    ~ ne mogu se dobiti kombinacijom preostalih
  - princip izgradnje kodne riječi:
    - "nezavisna" (ortogonalna) ispitivanja
    - "nezavisni" (ortogonalni) smještaj zaštitnih bitova
  - "nezavisna" (ortogonalna) ispitivanja:
    - svaki zaštitni bit "pokriva" (= štiti) drugi podskup bitova podatka
    - svaki bit podatka zaštićen s više zaštitnih bitova

Kodovi za zaštitu podataka

- Hammingovi kodovi:
  - odnos zaštitnih i informacijskih bitova :

    $2^r \geq k + r + 1$, $n = k + r$

    r: broj zaštitnih bitova
    k: broj informacijskih bitova
    n: duljina kodne riječi
  - obrazloženje:
    - jednostruka pogreška na jednom od n mjesta
    - bez pogrešaka

Kodovi za zaštitu podataka

- Hammingovi kodovi:
  - zaštitni bitovi su na mjestima 2^i (1, 2, 4, 8, …)
  - ostali bitovi su informacijski
- primjer: kod (11,7)
  - ukupno 11 bitova, od čega 7 nose podatke
  - korisno za zaštitu ASCII-znakova

1	**2**	3	**4**	5	6	7	**8**	9	10	11
P1	**P2**	d1	**P3**	d2	d3	d4	**P4**	d5	d6	d7

Kodovi za zaštitu podataka

- zaštitni bitovi računaju paritet počevši od sebe:
  - Pozicija 1: provjera 1 bita, preskače 1 bit, provjera 1, ...
  - Pozicija 2: prov. 2 bita, pres. 2 bita, prov. 2 bita, ...
  - Pozicija 4: prov. 4 bita, pres. 4 bita, prov. 4 bita, ...
  - ...
- raspored odgovornosti bitova:

	1	**2**	3	**4**	5	6	7	**8**	9	10	11
P1:	P1	P2		P3				P4			
P2:	P1	P2		P3				P4			
P3:	P1	P2		P3				P4			
P4:	P1	P2		P3				P4			

Kodovi za zaštitu podataka

- Hammingovi kodovi:
  - odnos zaštitnih i informacijskih bitova :

$$2^r \geq k + r + 1 \,, \quad n = k + r$$

BROJ INFORMACIJSKIH BITOVA (≤)	BROJ ZAŠTITNIH BITOVA	DULJINA KODNE RIJEČI
1	2	3
4	3	7
11	4	15
26	5	31
57	6	63
120	7	127

Kodovi za zaštitu podataka

- izgradnja Hammingovog koda za ispravljanje jednostruke pogreške :
  - zaštitni bitovi na mjesta koja se ne mogu dobiti kombinacijama drugih zaštitnih bitova: 2^i
  - zaštitni bitovi "pokrivaju" svoju poziciju ~ sve pozicije čiji redni broj sadrži 2^i

POZICIJA	nema pogreške	1	2	3	4	5	6	7
C_2	0	0	0	0	1	1	1	1
C_1	0	0	1	1	0	0	1	1
C_0	0	1	0	1	0	1	0	1
		C_0	C_1	k_1	C_2	k_2	k_3	k_4

  - zaštitni bitovi: C_2 C_1 C_0

Kodovi za zaštitu podataka

- Hammingov kod za ispravljanje jednostruke pogreške :
  - izračunavanje zaštitnih bitova, za parni paritet

POZICIJA	nema pogreške	1	2	3	4	5	6	7
C_2	0	0	0	0	1	1	1	1
C_1	0	0	1	1	0	0	1	1
C_0	0	1	0	1	0	1	0	1
		C_0	C_1	k_1	C_2	k_2	k_3	k_4

$$C_0 \oplus k_1 \oplus k_2 \oplus k_4 = 0 \quad \rightarrow \quad C_0 = k_1 \oplus k_2 \oplus k_4$$

$$C_1 \oplus k_1 \oplus k_3 \oplus k_4 = 0 \quad \rightarrow \quad C_1 = k_1 \oplus k_3 \oplus k_4$$

$$C_2 \oplus k_2 \oplus k_3 \oplus k_4 = 0 \quad \rightarrow \quad C_2 = k_2 \oplus k_3 \oplus k_4$$

Kodovi za zaštitu podataka

- Hammingov kod za ispravljanje jednostruke pogreške :
  - izračunavanje zaštitnih bitova, za neparni paritet

POZICIJA	nema pogreške	1	2	3	4	5	6	7
C_2	0	0	0	0	1	1	1	1
C_1	0	0	1	1	0	0	1	1
C_0	0	1	0	1	0	1	0	1
		C_0	C_1	k_1	C_2	k_2	k_3	k_4

$$C_0 \oplus k_1 \oplus k_2 \oplus k_4 \oplus 1 = 0 \rightarrow C_0 = k_1 \oplus k_2 \oplus k_4 \oplus 1$$

$$C_1 \oplus k_1 \oplus k_3 \oplus k_4 \oplus 1 = 0 \rightarrow C_1 = k_1 \oplus k_3 \oplus k_4 \oplus 1$$

$$C_2 \oplus k_2 \oplus k_3 \oplus k_4 \oplus 1 = 0 \rightarrow C_2 = k_2 \oplus k_3 \oplus k_4 \oplus 1$$

Kodovi za zaštitu podataka

- Hammingov kod za ispravljanje jednostruke pogreške :
  - Primjer: zaštita ASCII znaka A (41H)

$$k = n - r = 7 \Rightarrow r = 4$$

1	2	3	4	5	6	7	8	9	10	11
C_0	C_1	k_1	C_2	k_2	k_3	k_4	C_3	k_5	k_6	k_7

k_1	k_2	k_3	k_4	k_5	k_6	k_7
1	0	0	0	0	0	1

$C_0 = k_1 \oplus k_2 \oplus k_4 \oplus k_5 \oplus k_7 \rightarrow C_0 = 0$

$C_1 = k_1 \oplus k_3 \oplus k_4 \oplus k_6 \oplus k_7 \rightarrow C_1 = 0$

$C_2 = k_2 \oplus k_3 \oplus k_4 \rightarrow C_2 = 0$

$C_3 = k_5 \oplus k_6 \oplus k_7 \rightarrow C_3 = 1$

$\Rightarrow$ 0 0 1 0 0 0 0 1 0 0 1

		k_1		k_2	k_3	k_4		k_5	k_6	k_7
0	1	1	0	0	0	0	1	0	0	1
C_0	C_1		C_2				C_3			

$X = C_3\, C_2\, C_1\, C_0 = 1\ 0\ 1\ 0$

$Y = C_3'\, C_2'\, C_1'\, C_0' = 1\ 0\ 0\ 0$

mjesto pogreške: $X \oplus Y = 0010_2 = 2_{10}$

Kodovi za zaštitu podataka

- Hammingov kod za ispravljanje jednostruke pogreške:
  - sindrom
    ~ uzorak zaštitnih bitova
    koji ukazuje na mjesto pojave pogreške

Primjer:

sindrom = 2 ~ drugi bit kodne riječi je pogrešan!

		k_1		k_2	k_3	k_4		k_5	k_6	k_7
0	1	1	0	0	0	0	1	0	0	1
c_0	c_1		c_2				c_3			

sindrom = 0
~ nema pogreške

Hammingov kod – primjer

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																		
Provjera:8																		
Provjera:4																		
Provjera:2																		
Provjera:1																		

Hammingov kod - zaštitni bitovi

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4														
Provjera:2		C_2																
Provjera:1	C_1																	

Hammingov kod – zaštitni bit C_1

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4														
Provjera:2		C_2																
Provjera:1	C_1		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_1

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4														
Provjera:2		C_2																
Provjera:1	0		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_2

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4														
Provjera:2		C_2	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_2

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4														
Provjera:2		0	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_4

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				C_4	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_4

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8										
Provjera:4				1	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_8

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								C_8	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_8

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}		
Provjera:8								1	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_{16}

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																C_{16}	0	
Provjera:8								**1**	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštitni bit C_{16}

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ			1		0	1	1		0	0	0	1	1	0	1		0	
Provjera:16																**0**	0	
Provjera:8								**1**	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – zaštićena poruka

Podatak = $P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}$ = 101100011010
Zaštita = $C_1C_2C_4C_8C_{16}$ = 00110

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**1**	0	1	1	**1**	0	0	0	1	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**1**	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod - otkrivanje i ispravljanje pogreške

Pretpostavimo pogrešku na 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**1**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**1**	0	0	0		1	0	1			
Provjera:4				**1**	0	1	1						1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod - otkrivanje i ispravljanje pogreške

Pretpostavimo pogrešku na 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**1**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**0**	0	0	0	0	1	0	1			**X**
Provjera:4				**0**	0	1	1					0	1	0	1			**X**
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod - otkrivanje i ispravljanje pogreške

Pretpostavimo pogrešku na 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**1**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	0
Provjera:8								**0**	0	0	0	0	1	0	1			**1**
Provjera:4				**0**	0	1	1					0	1	0	1			**1**
Provjera:2		**0**	1			1	1			0	0			0	1			0
Provjera:1	**0**		1		0		1		0		0		1		1		0	0

Niz zaštitnih bitova iz dobivene poruke: 01100
Vrijednost zaštitnih bitova dobivena provjerom: 00000

Mjesto pogreške određuje *sindrom*: 01100 XOR 00000 = 01100 = $12_{(10)}$

Hammingov kod – pogreška na zaštitnom bitu

Pretpostavimo pogrešku na 4. bitu, tj. zaštitnom bitu C_4.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	1	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**1**	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – pogreška na zaštitnom bitu

Pretpostavimo pogrešku na 4. bitu, tj. zaštitnom bitu C_4.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	1	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**1**	0	0	0	1	1	0	1			
Provjera:4				**1**	0	1	1					1	1	0	1			X
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – pogreška na zaštitnom bitu

Pretpostavimo pogrešku na 4. bitu, tj. zaštitnom bitu C_4.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	1	1	0	1	**0**	0	
Provjera:16																**0**	0	0
Provjera:8								**1**	0	0	0	1	1	0	1			0
Provjera:4				**1**	0	1	1					1	1	0	1			**1**
Provjera:2		**0**	1			1	1			0	0			0	1			0
Provjera:1	**0**		1		0		1		0		0		1		1		0	0

Pogreška je na bitu $00100_2 = 4_{10}$

Hammingov kod – dvije pogreške

Pretpostavimo pogrešku na 4. bitu (tj. zaštitnom bitu C_4) i 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8									0	0	0	0	1	0	1			
Provjera:4					0	1	1					0	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – dvije pogreške

Pretpostavimo pogrešku na 4. bitu (tj. zaštitnom bitu C_4) i 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**0**	0	0	0	0	1	0	1			
Provjera:4				**0**	0	1	1					0	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – dvije pogreške

Pretpostavimo pogrešku na 4. bitu (tj. zaštitnom bitu C_4) i 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	
Provjera:8								**0**	0	0	0	0	1	0	1			
Provjera:4				**0**	0	1	1					0	1	0	1			
Provjera:2		**0**	1			1	1			0	0			0	1			
Provjera:1	**0**		1		0		1		0		0		1		1		0	

Hammingov kod – dvije pogreške

Pretpostavimo pogrešku na 4. bitu (tj. zaštitnom bitu C_4) i 12. bitu.

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ	C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																**0**	0	0
Provjera:8								**0**	0	0	0	0	1	0	1			**1**
Provjera:4				**0**	0	1	1					0	1	0	1			0
Provjera:2		**0**	1			1	1			0	0			0	1			0
Provjera:1	**0**		1		0		1		0		0		1		1		0	0

Pogreška na bitu 8. bitu ($01000_2 = 8_{10}$) !?

Hammingov kod – otkrivanje dvostruke pogreške

Dodavanjem paritetnog bita (na poziciji 0) moguće je otkriti prisustvo dvostruke pogreške (ili ispraviti jednu pogrešku).

Pozicija	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Rezultat
Binarno		00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	
Riječ		C_1	C_2	P_1	C_4	P_2	P_3	P_4	C_8	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	C_{16}	P_{12}	
Riječ	**0**	**0**	**0**	1	**0**	0	1	1	**1**	0	0	0	0	1	0	1	**0**	0	
Provjera:16																	**0**	0	0
Provjera:8									**0**	0	0	0	0	1	0	1			**1**
Provjera:4					**0**	0	1	1					0	1	0	1			0
Provjera:2			**0**	1			1	1			0	0			0	1			0
Provjera:1		**0**		1		0		1		0		0		1		1		0	0

Jednostruka pogreška → paritetni bit mora biti pogrešan

U primjeru je:
- paritetni bit ispravan
- a provjera 8 javlja grešku
→ dvostruka pogreška

Kodovi za zaštitu podataka

- Hammingov kod za ispravljanje jednostruke pogreške:
  - distanca d = 3
  - kod za ispravljanje "nezavisnih pogrešaka"
    ~ rezultat djelovanja "bijelog šuma"
  - efikasan kod, jer je R mali