

# Građa računala

2. Von Neumannov model računala i  
SISD arhitektura. Principi kodiranja  
podataka u računalu.



Preddiplomski izvanredni stručni studij  
Informacijske tehnologije

# Zapis podataka u računalu

- Osnovna namjena računala je pohranjivanje, pretraživanje i izvođenje operacija nad podacima (brojevi, znakovi, signali)
- Zahtjevi brojevnog sustava za zapis brojeva:
  - jednostavno izvršavanje logičkih i aritmetičkih operacija
  - jednostavna realizacija pomoću elektroničkih elemenata
  - dovoljno velik kapacitet i preciznost zapisa
  - moguć zapis signala pomoću brojevnog sustava

# Zapis podataka u računalu

- Zbog tehnoloških razloga podaci se u digitalnim računalima pohranjuju u binarnom obliku (0,1)
- **Bistabil**
  - Osnovna ćelija za pamćenje, može se nalaziti u jednom ili dva stabilna stanja (0 ili 1), odnosno može zapamtiti jednu binarnu znamenku
- **Registar**
  - Skup bistabila, n-bitovni registar može pohraniti n-bitovni podatak, odnosno može poprimiti jedno od  $2^n$  stanja

# Zapis podataka u računalu

- **BIT**

- jedno binarno mjesto se zove bit  $\{0,1\}$
- iz engleskog binary digit (bit)
- najmanja količina informacije

- **BYTE**

- 8 bitova = 1 byte
- iz engleskog binary term(byte)

- **RIJEČ**

- zapisi u računalima riječi dužine 1,2,4,8 bajta, tj. 8, 16, 32, 64 bita

# Brojevni sustavi

- pozicijski brojevni sustavi: vrijednost znamenke u zapisu ovisi o njenom položaju (npr. rimski brojevni sustav nije pozicijski)
- dekadski brojevni sustav: znamenke su iz skupa  $\{0,1,2,3,4,5,6,7,8,9\}$  a baza brojnog sustava je  $B=10$
- Npr.

$$\begin{aligned} 126,73 &= 1 \times 10^2 \\ &\quad + 2 \times 10^1 \\ &\quad + 6 \times 10^0 \\ &\quad + 7 \times 10^{-1} \\ &\quad + 3 \times 10^{-2} \end{aligned}$$

# Brojevnii sustavi

- broj znamenki brojevnog sustava određuje bazu sustava
- $z = \sum_{j=0}^{N-1} b_j B^j$
- Znamenke brojnog sustava:  $0 \leq b_j < B$
- Baza brojnog sustava:  $B$

# Brojevnii sustavi – decimalni brojevi

- $z = \sum_{j=-M}^{N-1} b_j B^j$
- Znamenke brojnog sustava:  $0 \leq b_j < B$
- Baza brojnog sustava:  $B$

# Brojevnii sustavi

- $B=2$   $\{0,1\}$  -**binarni** brojevni sustav
- $B=8$   $\{0,1,2,3,4,5,6,7\}$  – **oktalni** brojevni sustav
- $B=10$   $\{0,1,2,3,4,5,6,7,8,9\}$  – **dekadski** brojevni sustav
- $B=16$   $\{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$  – **heksadekadski**
  - često se koristi
  - blizak je binarnom sustavu, a zapis bitno kraći



# Brojevnii sustavi

- 4 BITNI zapis

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

- 8 BITNI zapis

- Binarni

od  $0000\ 0000_2$  do  $1111\ 1111_2$

- Heksadekadski

od  $00_{16}$  do  $FF_{16}$

- Dekadski

od  $0_{10}$  do  $255_{10}$

# Binarni brojevni sustav

$$\begin{aligned}(1011,011)_2 = & 1 \times 2^3 \\ & + 0 \times 2^2 \\ & + 1 \times 2^1 \\ & + 1 \times 2^0 \\ & + 0 \times 2^{-1} \\ & + 1 \times 2^{-2} \\ & + 1 \times 2^{-3}\end{aligned}$$

$$= (8 + 0 + 2 + 1 + 0 + 1/4 + 1/8)_{10}$$

$$= (11,375)_{10}$$

# Kapacitet binarnog broja

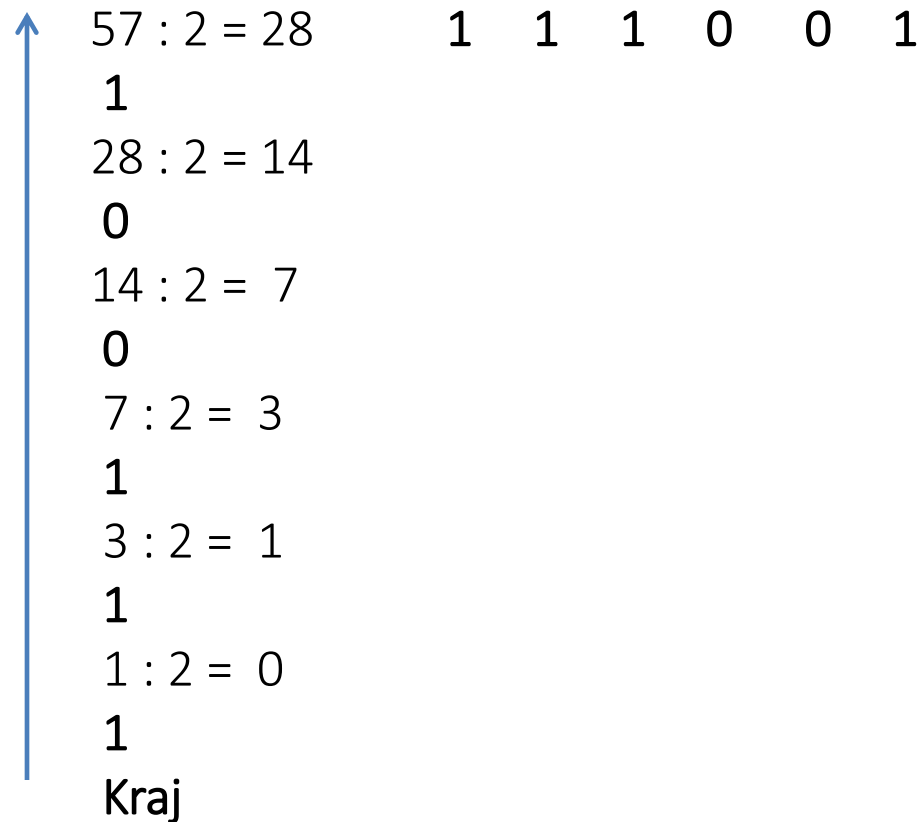
- Brojevi se pohranjuju u registrima širine  $n$  bitova
- Tipične vrijednosti:  $n = 8$  ili  $16$  ili  $32$  ili  $64$  bita
- Najveći cijeli nepredznačni broj (engl. *integer*) koji se može prikazati  $n$ -bitnim binarnim zapisom je  $2^n - 1$
- Npr. s  **$n=8$  bitova** to je:  **$2^8 - 1 = 256 - 1 = 255$**
- Ukupan broj različitih brojeva koji se mogu prikazati je  $2^n$ :
- Primjer:
  - $2^8 = 256$
  - $2^{16} = 65.536$
  - $2^{32} = 4.294.967.296$

# Zapis binarnih brojeva

- Kilobajt **kB** = 1024 B =  $2^8 \cong 10^3$
- Megabajt **MB** = 1024 kB =  $2^{20} \cong 10^6 = 1.048.576$  B
- Gigabajt **GB** = 1024 MB =  $2^{30} \cong 10^9$
- Tetrabajt **TB** = 1024 GB =  $2^{40} \cong 10^{12}$
- Peta, Eksa ...

# Prevođenje zapisa brojeva iz dekadskog u binarni zapis

- Binarni broj tvore ostaci dijeljenja s 2, odozdo prema gore



57 : 2 = 28	1	1	1	0	0	1
1						
28 : 2 = 14						
0						
14 : 2 = 7						
0						
7 : 2 = 3						
1						
3 : 2 = 1						
1						
1 : 2 = 0						
1						
Kraj						

# Prevođenje zapisa brojeva iz decimalnog dekadskog u binarni zapis

- Binarni broj tvore prve znamenke rezultata množenja s 2, od vrha prema dolje

$$0,375 \times 2 = 0,75$$

$$(0,375)_{10} = (0,011)_2$$

0

$$0,75 \times 2 = 1,5$$

1

$$0,5 \times 2 = 1,0$$

1

$$0 \times 2 = 0$$


Kraj



# Prevođenje zapisa brojeva iz dekadskog u heksadekadski zapis

- Heksadekadski broj tvore ostaci dijeljenja s 16, odozdo prema gore

$314156/16= 19634$	ostatak	12 (C)
$19634/16= 1227$	ostatak	2 (2)
$1227/16 = 76$	ostatak	11 (B)
$76/16= 4$	ostatak	12 (C)
$4/16= 0$	ostatak	4 (4)



$$(31415610)_{10}=(4CB2C)_{16}$$

# Prevođenje zapisa brojeva iz dekadskog u heksadekaderski zapis

- Neki dekadski brojevi ne mogu se točno prevesti u binarni zapis  $\Rightarrow$  greška zaokruživanja!
- npr.  $(0,1)_{10} = (0,00011001100110\dots)_2$
- točnost zapisa ovisi o broju bitova s kojima se zapisuju brojevi u računalu



# Zbrajanje binarnih brojeva

- potrebna je tablica zbrajanja
- za binarni brojni sustav:

$0+0 = 0$  i prijenos 0

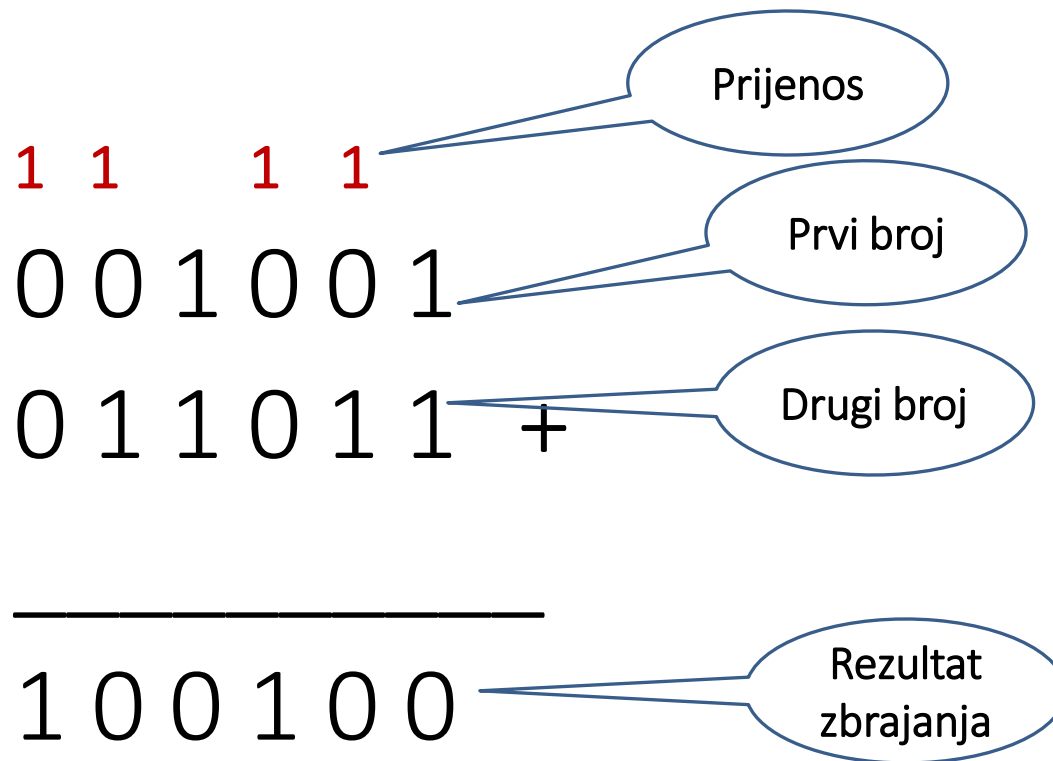
$0+1 = 1$  i prijenos 0

$1+0 = 1$  i prijenos 0

$1+1 = 0$  i prijenos 1

# Zbrajanje binarnih brojeva

- Primjer zbrajanja binarnih brojeva  
001001 i 011011



# Pozitivni i negativni binarni brojevi

- negativne binarne brojeve zapisujemo pomoću njihovog dvojnog komplementa
- **dvojni komplement** se računa kao zbroj jediničnog komplementa i 1
- **jedinični komplement** se računa zamjenom vrijednosti za 0 i 1 u zadanom registru za apsolutnu vrijednost zadanog negativnog binarnog broja

# Pozitivni i negativni binarni brojevi

- npr. za dekadski broj - 5 koji je zapisan u 4-bitnom registru:  $(5)_{10} = (0101)_2$
- jedinični komplement (JK) =  $(1010)_2$
- dvojni komplement (DK) =  $JK + (0001)_2$

$$DK = 1010 + 0001 = 1011$$

$(-5)_{10} = (1011)_2 \Rightarrow$  Prikaz negativnog dekadskog broja -5 zapisanog u 4-bitnom registru !

# Pozitivni i negativni binarni brojevi

- npr. u registru s 3 bita prikazuje se slijedeći raspon pozitivnih i negativnih brojeva

Dekadski broj	Binarni broj
0	000
1	001
2	010
3	011
-4	100
-3	101
-2	110
-1	111

## Pozitivni i negativni binarni brojevi

- **Primjer: Prikažite broj -12 u registru s 8 bita**
- Broj 12 predstavljen u 8-bitnom registru glasi: 0 0 0 0 1 1 0 0
- Zamjenimo 0 i 1 (JK) i dobivamo:  
1 1 1 1 0 0 1 1
- Na kraju dodamo 1 (DK) i dobivamo :  
1 1 1 1 0 0 1 1
- Izračunata vrijednost predstavlja -12, zapisan u metodi dvojnog komplementa

# Raspon vrijednosti binarnih brojeva

- Za registar veličine  $n$  bita, vrijedi:
- raspon brojeva kada se prikazuju cijeli brojevi s predznakom:

$$-2^{n-1}, \dots, -1, 0, 1, \dots, 2^{n-1}-1$$

- raspon brojeva kada se prikazuju samo pozitivni cijeli brojevi:

$$0, \dots, 2^n-1$$

# Raspon vrijednosti binarnih brojeva

- **Zadatak:** *Odredite najveći i najmanji pozitivni cijeli broj koji se može smjestiti u registar sa 4, 8 i 16 bita.*
- Općenito, za registar za  $n$  bita, najveći pozitivni cijeli broj koji se može pohraniti je:  $2^n - 1$
- Za registar sa 4 bita:  $2^4 - 1 = 15$
- Za registar sa 8 bita:  $2^8 - 1 = 255$
- Za registar sa 16 bita:  $2^{16} - 1 = 65\,535$



# Raspon vrijednosti binarnih brojeva

- **Zadatak:** *Odredite najveći i najmanji cijeli broj s predznakom koji se može smjestiti u registar sa 4, 8 i 16 bita, pod pretpostavkom da za prikaz negativnog broja koristite metodu dvojnog komplementa.*
- Općenito, za registar s n bita u kojem je prvi bit predznak, važe slijedeći izrazi:
- najmanji negativni broj koji se može prikazati :  
$$-2^{n-1}$$
- najveći pozitivni broj koji se može prikazati:  
$$2^{n-1}-1$$

# Raspon vrijednosti binarnih brojeva

- najmanji negativni broj koji se može prikazati s predznakom:
  - za  $n=4$  = -8
  - za  $n=8$  = -128
  - za  $n=16$  = -32768
- najveći pozitivni broj koji se može prikazati u registru s predznakom:
  - za  $n=4$  = 7
  - za  $n=8$  = 127
  - za  $n=16$  = 32767

# Zapis realnih brojeva

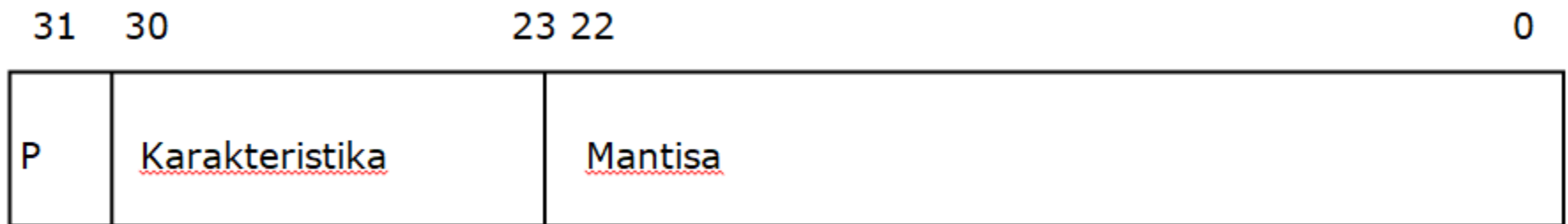
- U prikazu brojeva je moguće koristiti eksponencijalni oblik, npr. dekadski broj  $123,456 = 0,123456 * 10^3$
- decimalni broj  $R$  u dekadskom sustavu se općenito može zapisati kao:

$$R = M * 10^E$$

- mantisa broja  $M = 0,123456$
- eksponent  $E = 3$

# Zapis realnih brojeva

- IEEE (Institute of Electrical and Electronics Engineers) standard 754 za prikaz realnih brojeva u standardnoj točnosti:



- P predznak (  $P=1$  negativan,  $P=0$  pozitivan)
- Karakteristika: binarni eksponent + 127 (da se izbjegne prikaz negativnog eksponenta)
- Mantisa je normalizirana (samo jedan bit ispred binarne točke)

# Zapis realnih brojeva

- **Primjer:** Prikazati dekadski broj 5.75 u IEEE 754 standardu za prikaz realnih brojeva u standardnoj točnosti
- **$5.75_{10} = 101.11_2 * 2^0 = 1.0111_2 * 2^2$**
- Kako se normalizacijom svakog binarnog broja (osim nule) postiže oblik 1.xxxxx, vodeća jedinica ne pohranjuje se u računalu i naziva se skrivenim bitom. Time se štedi jedan bit što povećava točnost.

# Zapis realnih brojeva

- Predznak = 0 (pozitivan broj)
- Binarni eksponent = 2
- $K = 2 + 127 = 129 = (1000\ 0001)_2$
- Mantisa (cijela): 1.0111
- Mantisa (bez skrivenog bita): 0111
- Rezultat:

0 10000001 011100000000000000000000

0100 0000 1011 1000 0000 0000 0000 0000

**4 0 B 8 0 0 0 0**

(heksadekadni)

# Raspon i točnost realnih brojeva

- Za slučaj realnog broja standardne točnosti karakteristika (8 bita) se može nalaziti u intervalu  $[0,255]$
- $K = 0$  rezervirana je za prikaz nule
- $K = 255$  rezervirana je za prikaz beskonačno velikog broja
- Kako je  $BE = K - 127$ , BE se može kretati u intervalu:  $[-126,127]$ .
- Najmanji pozitivni broj različit od nule koji se može prikazati je:

$$1.0_2 * 2^{-126} \approx 1.175494350822 * 10^{-38}$$

- a najveći je:

$$1.1111111111111111111111111111111_2 * 2^{127} = 3.402823669209 * 10^{38}$$

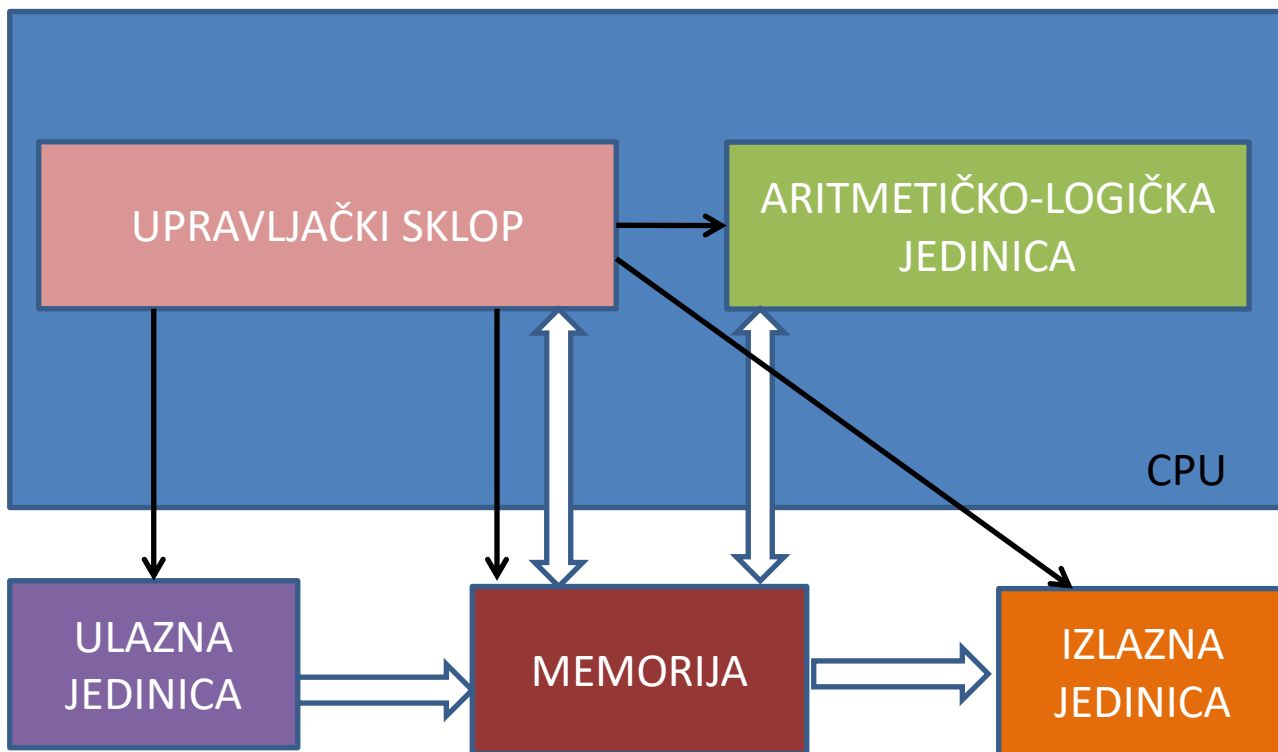
# Model von Neumannova računala


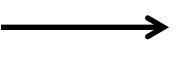
- Ideja:
  - zajedničko pohranjivanje podataka i programa u memoriji računala
  - sljedeći korak programa ovisi o prijašnjem
  - John von Neumann
  - (28/12/1903 . – 08/02/1957.)
- predložena arhitektura za EDVAC(Electronic Discrete Variable Automatic Computer) 1945. postaje poznata pod imenom von Neumanova arhitektura računala





# Model von Neumannova računala



- Ulazno/izlazni tok 
- Upravljački signal 

# Model von Neumannova računala

- računalno se sastoji od četiri osnovne funkcijske jedinice:
  - aritmetičko-logičke
  - upravljačke
  - memorijske
  - ulazno-izlazne jedinice
- funkcijske jedinice računala povezane su tokom podataka, instrukcijskim tokom i tokom upravljačkih signala
- većinu upravljačkih signala generira upravljačka jedinica na temelju tumačenja instrukcije

# Ulazna jedinica

- služi za unos podataka iz vanjskog svijeta u memoriju računala: tipkovnica, miš, mikrofon, kamera ....
- na prethodnoj slici modela von Neumannova računala je vidljivo kako nema direktnog toka podataka između ulazne jedinice i memorije – tzv. „usko grlo von Neumannova modela”
- današnja računala ostvaruju direktan pristup ulazne jedinice memoriji računala – tzv. DMA prijenos ((engl. DMA - Direct Memory Access)
- prijenosom podataka na tom putu upravlja poseban DMA upravljački sklop pa je omogućen istodobni prijenos podataka i obrada u aritmetičko- logičkoj jedinici.

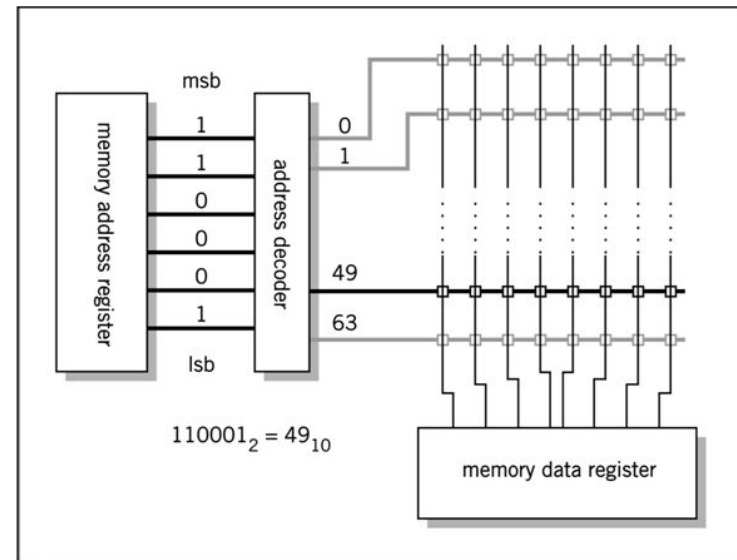
# Izlazna jedinica

- služi za prikaz obrađenih podataka:
  - monitor
  - pisač
  - ploter
  - zvučnik ...



# Memorija

- svaki podatak u memoriji ima svoju jednoznačnu adresu
- u memoriju se pohrajuju podaci i programi
- postupci pisanja i čitanja preko dva registra:
  - MAR (engl. Memory Address Register)
  - MDR (engl. Memory Data Register)



# Memorija - čitanje

- zapisuje se adresa podatka koji se čita u registar MAR
- generiraju se upravljački signali na liniji za čitanje, koji omogućavaju da se sadržaj memorijske lokacije zapisane u MAR prenese u registar MDR
- nakon operacije čitanja podatak se nalazi u MDR
- vrši se prijenos podatka iz registra MDR u ciljni registar

# Memorija - pisanje

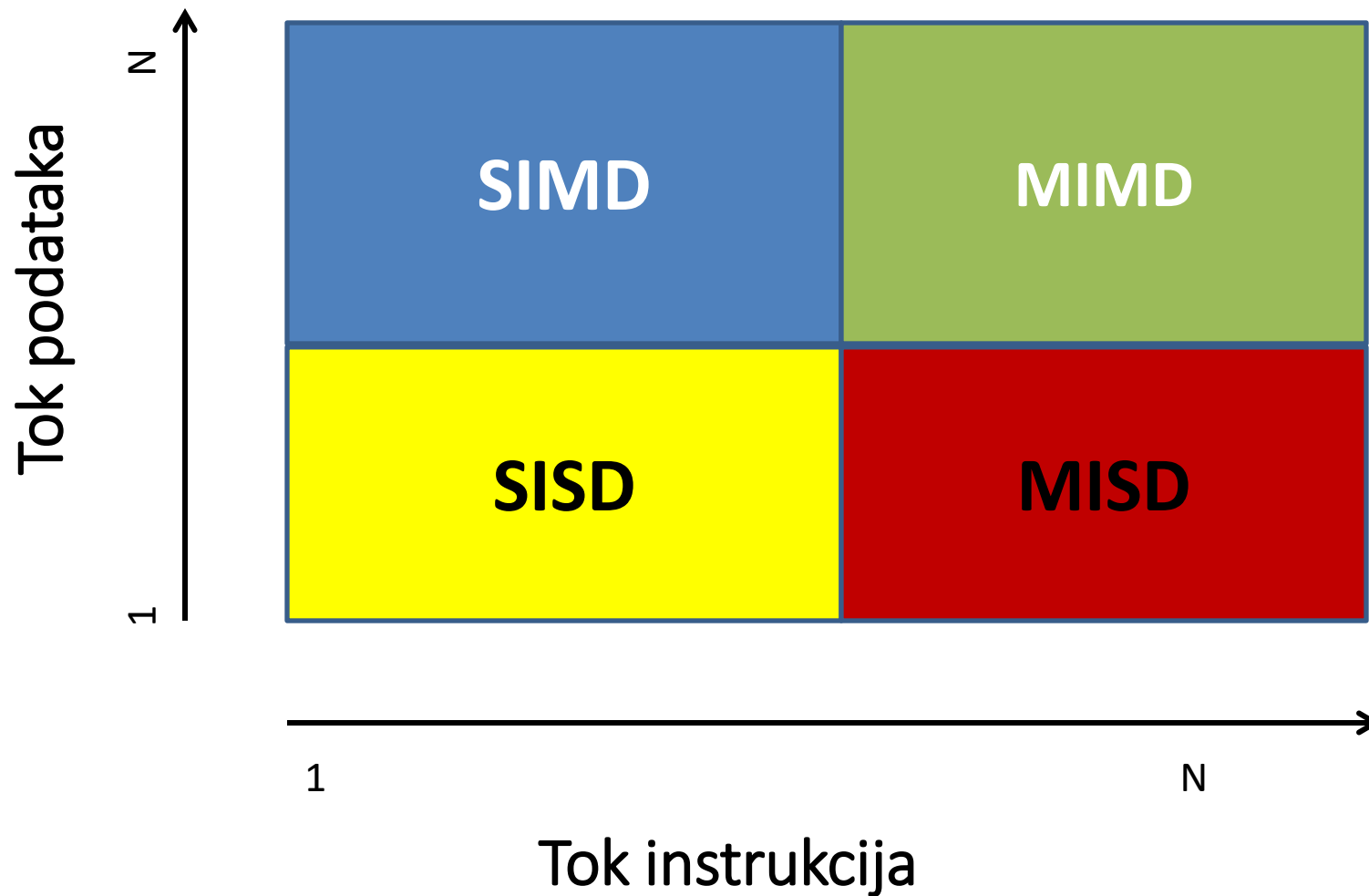
- zapisuje se adresa na koju želimo pohraniti podatak u MAR
- podatak se prenosi u registar MDR
- generiraju se upravljački signali, koji vrše prijenos podataka iz MDR u memoriju na adresu zapisanu u registru MAR
- nakon operacije pisanja podatak se nalazi u memoriji

# Flynnova klasifikacija

- tijekom izvođenja programa možemo govoriti o toku podataka (DS) i instrukcijskom toku (IS) koji se uspostavljaju između funkcijskih jedinica stroja
- tipovi arhitekture prema Flynnovoj klasifikaciji mogu se predložiti u dvodimenzionalnom prostoru koji je određen brojem tokova podataka i brojem instrukcijskih tokova



# Flynnova klasifikacija



# Flynnova klasifikacija

- **SISD** (engl. *Single Instruction Stream Single Data Stream*) - računalo s jednostrukim instrukcijskim tokom i jednostrukim tokom podataka
- arhitektura SISD predstavlja arhitekturu sekvencijalnog računala temeljenog na von Neumannovom modelu

# Flynnova klasifikacija

- **SIMD** (engl. *Single Instruction Stream Multiple Data Stream*) - računalo s jednostrukim instrukcijskim tokom i višestrukim tokom podataka
- u ovu se kategoriju svrstavaju paralelna računala (nazivaju se i matrična računala) koja se obično sastoje od velikog broja procesora ili procesnih elemenata koji istodobno izvršavaju istu instrukciju na različitim podacima

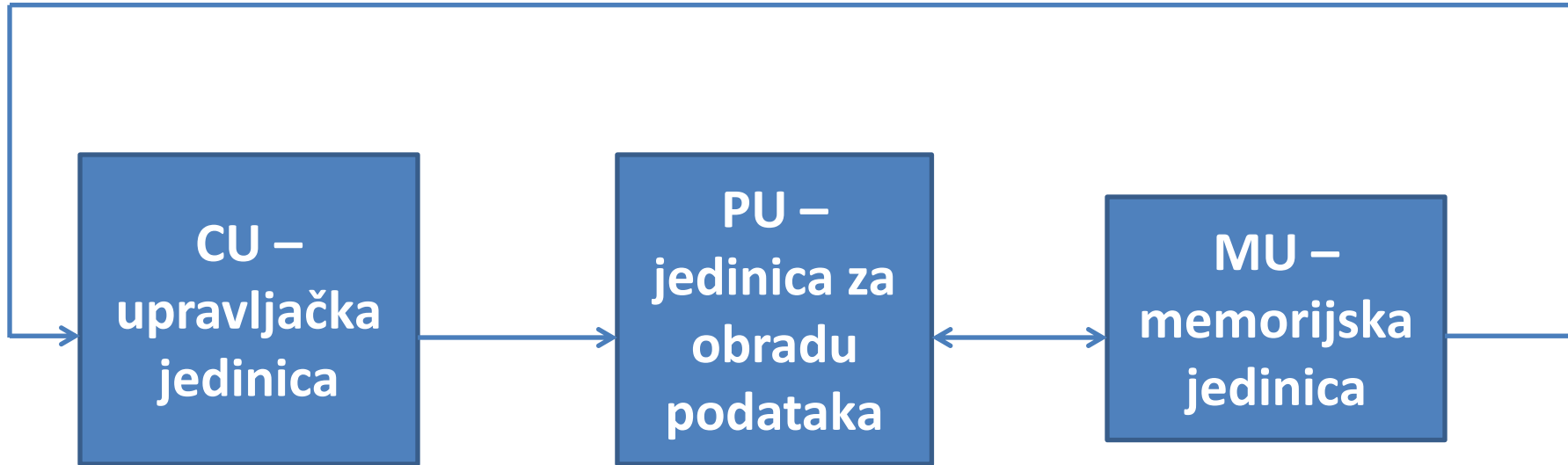
# Flynnova klasifikacija

- **MIMD** (engl. *Multiple Instruction Stream Multiple Data Stream*) - računalno s višestrukim instrukcijskim tokom i višestrukim tokom podataka
- multiprocesorski sustavi, odnosno paralelni računarski sustavi s dva i više procesora približno jednakih performansi, pri čemu svaki od njih ima pristup zajedničkoj memorijskoj jedinici i svi dijele ulazno-izlazne jedinice, a pritom djeluju pod jednim operacijskim sustavom

# Flynnova klasifikacija

- **MISD** (engl. *Multiple Instruction Stream Single Data Stream*) - računalo s jednostrukim instrukcijskim tokom i višestrukim tokom podataka.
- Teorijski strogo gledano, računala ovog tipa arhitekture ne mogu se fizički realizirati jer je nemoguće ostvariti da se istodobno više različitih instrukcija izvršava na istim podacima
- Dogovorno u ovu kategoriju uvrštavamo protočna (engl. *pipeline*) računala i računarske sustave koji se temelje na sistoličkim poljima

# Shematski prikaz arhitekture tipa SISD



# Shematski prikaz arhitekture tipa SISD

- jedan instrukcijski tok (IS) i jedan tok podataka (DS) izvire iz memorijske jedinice (MU)
- instrukcije se dovode do upravljačke jedinice (CU) gdje se dekodiraju
- dekodirani instrukcijski tok i tok podataka "susreću" se u jedinici za obradu PU u kojoj se tok podataka preoblikuje
- preoblikovani podaci se ponovo dovode u memorijsku jedinicu koja predstavlja izvor ponora toka podataka

# Binarni kôdovi – BCD kôd

- BCD kôd za prikaz jedne znamenke koristi 4 bita, pri čemu se znamenka zapisuje kao binarni ekvivalent njene vrijednosti
- Pretvorimo znamenke iz dekadskih u binarne vrijednosti u grupama po 4 bita, npr. za dekadski broj 3720:

3	7	2	0
0011	0111	0010	0000



# Sigurnosni kodovi za kontrolu pariteta

- U prijenosu signala se u praksi uvijek događaju pogreške, najčešće uzrokovane smetnjama na prijenosnim sustavima
- Da bi se signali zaštitili, a time i informacija koju prenosimo, koristimo se kodovima koji omogućuju detekciju greške
- Redudanciju (zalihost) koda definiramo izrazom:  
$$R = \text{broj zaštitnih bitova} / \text{ukupan broj bitova}$$

# Sigurnosni kôdovi za kontrolu pariteta

- Podatak se smješta u 8 bita, a 9-ti bit je bit za otkrivanje i korekciju greške.
- **Neparan paritet:** prilikom zapisa podatka u registar, bit pariteta se koristi tako da se postavlja na 1 ako je broj jedinica u bloku podatka neparan
- **Paran paritet:** kontrolni bit se postavlja na 1 ako je broj jedinica u registru podatka paran

$$R = 1/9 = 0,111$$

# Hammingov kôd

- Hamming je prvi izveo linearne kodove  $(n,k)$  koji korigiraju pojedinačnu pogrešku, pri čemu je:
  - $n$  = ukupan broj bitova u kodnoj riječi
  - $k$  = broj bitova informacije
  - $m = n - k$  = broj kontrolnih bitova
- Vrijedi pravilo da mora postojati odnos:
$$2m \geq k + m + 1$$
- Kontrolni bitovi se postavljaju na pozicijama koje su potencije broja 2

# Hammingov kôd

- **Primjer:** *Koliko je bitova potrebno za prijenos 12 bita informacije Hammingovim kodom ?*
- Ako sa C označimo kontrolne, a sa P podatkovne:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
$2^0$	$2^1$		$2^2$				$2^3$								$2^4$	
C	C	P	C	P	P	P	C	P	P	P	P	P	P	P	C	P

potrebno je poslati ukupno 17 bitova da bismo prenijeli 12 bitova informacije, upotrebom Hammingova koda.

## EXCESS-3 kôd

- Kao i BCD kod za kodiranje jedne dekadске znamenke koristi 4 bita, pri čemu se znamenka zapisuje kao binarni ekvivalent njene vrijednosti uvećane za 3
- Primjer: dekadski broj 3720 u Excess-3 kodu

3	7	2	0
0110	1010	0101	0011

# Bikvinarni kôd

- Za kodiranje dekadskih znamenki se primjenjuje sljedeća tablica :

broj	bikvinarni kod
	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

# ASCII kôd

- zapis svih slova i znakova pomoću binarnih brojeva
- za kodiranje znakova ASCII kôd koristi 7 bitova
- Unicode–zapis sa 16 bitova –65000 različitih znakova- uveden 1991.

Binary	Oct	Dec	Hex	Glyph	Binary	Oct	Dec	Hex	Glyph	Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20		100 0000	100	64	40	@	110 0000	140	96	60	`
010 0001	041	33	21	!	100 0001	101	65	41	A	110 0001	141	97	61	a
010 0010	042	34	22	"	100 0010	102	66	42	B	110 0010	142	98	62	b
010 0011	043	35	23	#	100 0011	103	67	43	C	110 0011	143	99	63	c
010 0100	044	36	24	\$	100 0100	104	68	44	D	110 0100	144	100	64	d
010 0101	045	37	25	%	100 0101	105	69	45	E	110 0101	145	101	65	e
010 0110	046	38	26	&	100 0110	106	70	46	F	110 0110	146	102	66	f
010 0111	047	39	27	'	100 0111	107	71	47	G	110 0111	147	103	67	g
010 1000	050	40	28	(	100 1000	110	72	48	H	110 1000	150	104	68	h
010 1001	051	41	29	)	100 1001	111	73	49	I	110 1001	151	105	69	i
010 1010	052	42	2A	*	100 1010	112	74	4A	J	110 1010	152	106	6A	j
010 1011	053	43	2B	+	100 1011	113	75	4B	K	110 1011	153	107	6B	k
010 1100	054	44	2C	,	100 1100	114	76	4C	L	110 1100	154	108	6C	l
010 1101	055	45	2D	-	100 1101	115	77	4D	M	110 1101	155	109	6D	m
010 1110	056	46	2E	.	100 1110	116	78	4E	N	110 1110	156	110	6E	n
010 1111	057	47	2F	/	100 1111	117	79	4F	O	110 1111	157	111	6F	o
011 0000	060	48	30	0	101 0000	120	80	50	P	111 0000	160	112	70	p
011 0001	061	49	31	1	101 0001	121	81	51	Q	111 0001	161	113	71	q
011 0010	062	50	32	2	101 0010	122	82	52	R	111 0010	162	114	72	r
011 0011	063	51	33	3	101 0011	123	83	53	S	111 0011	163	115	73	s
011 0100	064	52	34	4	101 0100	124	84	54	T	111 0100	164	116	74	t
011 0101	065	53	35	5	101 0101	125	85	55	U	111 0101	165	117	75	u
011 0110	066	54	36	6	101 0110	126	86	56	V	111 0110	166	118	76	v
011 0111	067	55	37	7	101 0111	127	87	57	W	111 0111	167	119	77	w
011 1000	070	56	38	8	101 1000	130	88	58	X	111 1000	170	120	78	x
011 1001	071	57	39	9	101 1001	131	89	59	Y	111 1001	171	121	79	y
011 1010	072	58	3A	:	101 1010	132	90	5A	Z	111 1010	172	122	7A	z
011 1011	073	59	3B	;	101 1011	133	91	5B	[	111 1011	173	123	7B	{
011 1100	074	60	3C	<	101 1100	134	92	5C	\	111 1100	174	124	7C	
011 1101	075	61	3D	=	101 1101	135	93	5D	]	111 1101	175	125	7D	}
011 1110	076	62	3E	>	101 1110	136	94	5E	^	111 1110	176	126	7E	~
011 1111	077	63	3F	?	101 1111	137	95	5F	_					

# ASCII kôd

- Najvažnije ASCII vrijednosti:

0	– znak NULL ('\0')
32	– praznina (' ')
48 – 57	– znamenke '0'-'9'
65 – 90	– velika slova 'A' do 'Z'
97 – 122	– mala slova 'a' do 'z'