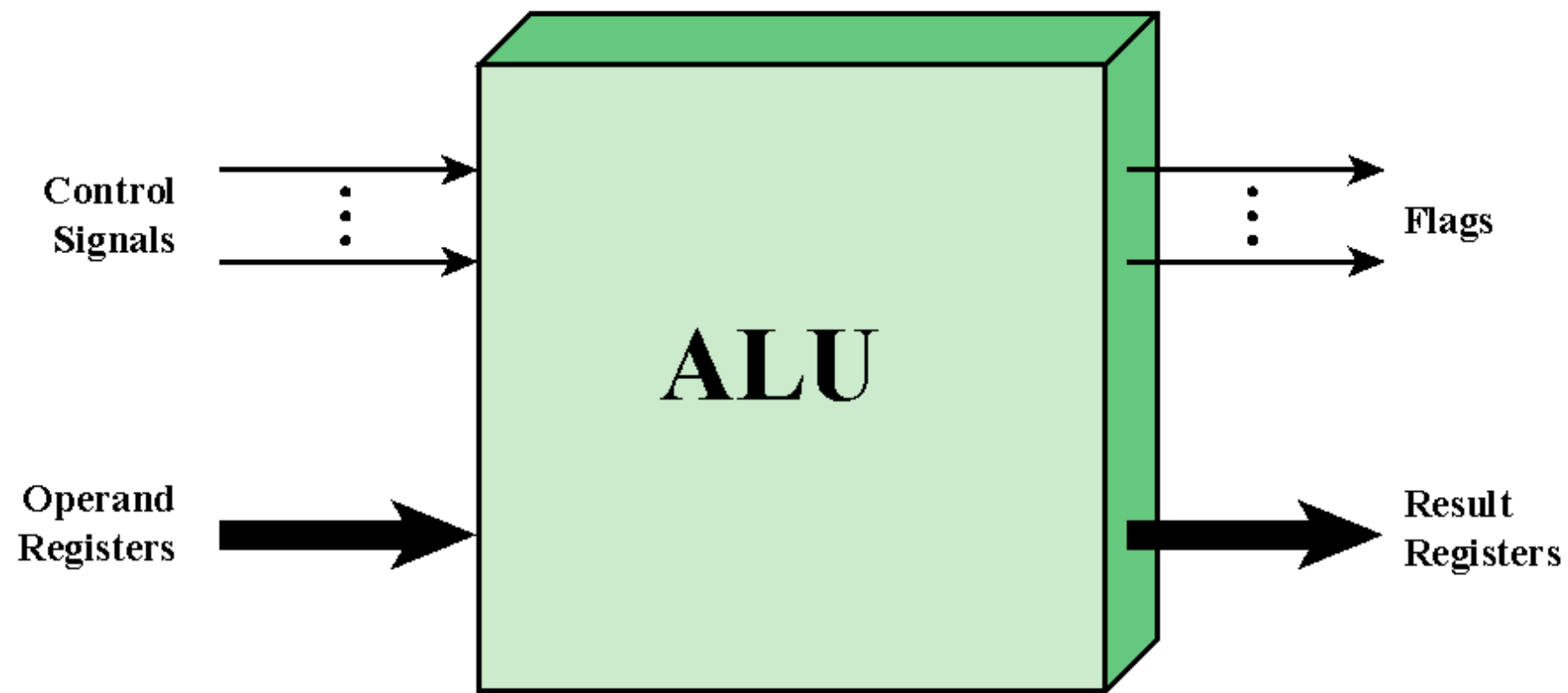


# Увод у организацију и архитектуру рачунара 1

Александар Картељ  
kartelj@matf.bg.ac.rs

# Аритметичко логичка јединица



# Рачунарска аритметика

Цели бројеви

# Подсећање на различите типове записа

- Знак и апсолутна вредност
  - Први бит означава знак
  - Позитивна и негативна нула
- Непотпуни комплемент
  - Први бит такође означава знак
  - Негација број се добија комплентирањем свих цифара
  - Позитивна и негативна нула
- Потпуни комплемент
  - Први бит означава знак
  - Негација се добија додавањем јединце на непотпуни комплемент
  - Само једна нула

$$A_{ZA} = (-1)^{a_{n-1}} \sum_{i=0}^{n-2} 2^i a_i$$

$$A_{NK} = (-2^{n-1} + 1)a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

$$A_{PK} = -2^{n-1} a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

# Примери записа

Dekadna vrednost	Znak i apsolutna vrednost	Nepotpuni kompliment	Potpuni komplement
+127	01111111	01111111	01111111
+64	01000000	01000000	01000000
+3	00000011	00000011	00000011
+2	00000010	00000010	00000010
+1	00000001	00000001	00000001
+0	00000000	00000000	00000000
-0	10000000	11111111	---
-1	10000001	11111110	11111111
-2	10000010	11111101	11111110
-3	10000011	11111100	11111101
-64	11000000	10111111	11000000
-127	11111111	10000000	10000001

# Операције над целим бројевима

- Мењање дужине записа
- Промена знака
- Прекорачење
- Сабирање и одузимање
- Множење
- Дељење

# Мењање дужине записа

- Нека тренутна дужина записа износи  $n$
- Ако је циљна дужина записа  $m$ , онда имамо три случаја:
  1.  $m < n$  – није дозвољена операција, јер се могу изгубити информације
  2.  $m = n$  – нема промене дужине у овом случају
  3.  $m > n$  – валидна промена дужине

# Мењање дужине записа знак и апсолутна вредност

- Бит за знак се помери на позицију највеће тежине, а остала места попуне нулама

Dekadna vrednost	8-bitna reč	16-bitna reč	Zapis
+127	01111111	0000000001111111	znak i aps. vred.
+5	00000101	0000000000000101	znak i aps. vred.
+0	00000000	0000000000000000	znak i aps. vred.
-0	10000000	1000000000000000	znak i aps. vred.
-5	10000101	1000000000000101	znak i aps. vred.
-127	11111111	1000000001111111	znak i aps. vred.



# Мењање дужине записа непотпуни и потпуни комплемент

- Број са највише позиције у краћем запису се дописује на све новододате позиције

Dekadna vrednost	8-bitna reč	16-bitna reč	Zapis
+5	00000101	00000000000000101	nepotpuni komp.
-5	11111010	1111111111111010	nepotpuni komp.
+9	00001001	00000000000001001	potpuni kompl.
-9	11110111	1111111111110111	potpuni kompl.

# Промена знака – знак и апсолутна вредност

- Комплементира се бит за знак броја

Dekadna vrednost	Binarni zapis
+9	00001001
+5	00000101
-5	10000101
-9	10001001

# Промена знака – непотпуни комплемент

- Комплементирање сваке цифре у запису

<u>Dekadna vrednost</u>	<u>Binarni zapis</u>
+9	00001001
+5	00000101
-5	11111010
-9	11110110

# Промена знака – потпуни комплемент

- Два корака:
  1. Комплементирање сваке цифре укључујући и место за знак
  2. Добијени број се сабере са 1 и игнорише евентуално прекорачење

+9	=	00001001	potpuni komplement
		11110110	1. korak
		+00000001	2. korak
-9		11110111	rezultat
-5	=	11111011	potpuni komplement
		00000100	1. korak
		+00000001	2. korak
+5	=	00000101	rezultat

# Промена знака – потпуни комплемент (2)

0	=	00000000	potpuni komplement
		11111111	1. korak
		+00000001	2. korak
0	=	1 00000000	rezultat, pri čemu se prekoračenje igno- riše

-128	=	10000000	potpuni komplement
		01111111	1. korak
		+00000001	2. korak
-128	=	10000000	rezultat koji predstavlja isti broj

# Прекорачење

- При сабирању бројева  $A$  и  $B$  који су записани са  $n$  цифара, може се добити број  $C$  за чији тачан запис је потребна  $n+1$  цифра
- Ова ситуација се назива прекорачење
- На бројеве  $A$  и  $B$  се додаје једна цифра вишка на највишу позицију
  - Код сабирања се прекорачење препознаје тако што збир има различит знак у односу на операнде – односно различиту цифру вишка
- Генерише се специјални (overflow) сигнал у ALU
- Може се десити и код множења, ако није одвојено довољно простора за производ

# Прекорачење код потпуног комплемента

- Код потпуног комплемента је могуће одредити прекорачење и без додавања цифре вишка
- Логичка формула за детекцију прекорачења је:

$$P = (a_{n-1} \wedge b_{n-1} \wedge \neg c_{n-1}) \vee (\neg a_{n-1} \wedge \neg b_{n-1} \wedge c_{n-1})$$

- Проверити је на четворобитним бројевима (3 и 5) или (-4 и -7).
- Проверити да ли ради у запису знак и апсолутна вредност.

# Сабирање и одузимање неозначених бројева

- Записи се поравнају у складу са тежинама цифара
- Само се врши бинарно сабирање сваке две поравнате цифре
- Могућ је пренос у износу од 1 на вишу позицију
- Пренос на позицију најмање тежине је подразумевано 0

a) 14 + 10

$$\begin{array}{r} 14 = 00001110 \\ 10 = 00001010 \\ \hline 24 = 00011000 \end{array}$$

b) 252 + 5

$$\begin{array}{r} 252 = \boxed{0} 11111100 \\ 5 = \boxed{0} 00000101 \\ \hline *** = \boxed{1} 00000001 \end{array}$$



# Сабирање и одузимање знак и апсолутна вредност

- Ако су бројеви истог знака, онда је и резултат сабирања тог истог знака
- Ако су бројеви различитог знака, знак ће бити једнак сабирку који има већу апсолутну вредност
- Апсолутна вредност је разлика веће и мање апс. вр. бројева
- Одузимање се своди на сабирање уз промену знака операнду

a) +14 + 10

$$\begin{array}{r} +14 = 0|0001110 \\ +10 = 0|0001010 \\ \hline 24 = 0|0011000 \end{array}$$

b) +127 + 3

$$\begin{array}{r} +127 = \boxed{0} 0|1111111 \\ +3 = \boxed{0} 0|0000011 \\ \hline *** = \boxed{1} 0|0000010 \end{array}$$



# Сабирање и одузимање непотпуни комплемент (2)

- Одузимање се своди на сабирање са негираним операндом.

a)  $+14 + 10$

I korak:

$$A=+14 = 00001110$$

$$B=+10 = 00001010$$

$$\hline C' = 0|00011000$$

II korak:

$$C'' = 00011000$$

0

$$\hline C=+24 = 00011000$$

b)  $+127 - (+10) = +127 + (-10)$

I korak:

$$A=+127 = 01111111$$

$$B=-10 = 11110101$$

$$\hline C' = 1|01110100$$

II korak:

$$C'' = 01110100$$

1

$$\hline C=+117 = 01110101$$

c)  $+100 + 65$

I korak:

$$A=+100 = 01100100$$

$$B=+65 = 01000001$$

$$\hline C' = 0|10100101$$

II korak:

$$C'' = 10100101$$

0

$$\hline C=*** = 10100101$$

# Сабирање и одузимање ПОТПУНИ КОМПЛЕМЕНТ

- Најпре се рачуна међурезултат као и збир неозначених бројева

$$\begin{array}{r} A = \quad a_{n-1} \quad a_{n-2} \quad \dots \quad a_1 \quad a_0 \\ B = \quad b_{n-1} \quad b_{n-2} \quad \dots \quad b_1 \quad b_0 \\ \hline C' = c'_n \quad c'_{n-1} \quad c'_{n-2} \quad \dots \quad c'_1 \quad c'_0 \end{array}$$

- Коначан резултат се добија простим уклањањем преноса са највише позиције и провером прекорачења
- Одузимање се, као и раније, своди на сабирање са промењеним знаком операнда

# Сабирање и одузимање потпуни комплемент (2)

a)  $+14 + 10$

$$\begin{array}{rcl} A=+14 & = & 00001110 \\ B=+10 & = & 00001010 \\ \hline C' & = & 0|00011000 \\ \hline C=+24 & = & 00011000 \end{array}$$

c)  $+127 - 10$

$$\begin{array}{rcl} A=+127 & = & 01111111 \\ B=-10 & = & 11110110 \\ \hline C' & = & 1|01110101 \\ \hline C=+117 & = & 01110101 \end{array}$$

b)  $+100 + 65$

$$\begin{array}{rcl} A=+100 & = & 01100100 \\ B=+65 & = & 01000001 \\ \hline C' & = & 0|10100101 \\ \hline C=*** & = & 10100101 \end{array}$$

d)  $-100 - (+65) = -100 + (-65)$

$$\begin{array}{rcl} A=-100 & = & 10011100 \\ B=-65 & = & 10111111 \\ \hline C' & = & 1|01011011 \\ \hline C=*** & = & 01011011 \end{array}$$

# Множење неозначених бројева

- Стандардно множење са потписивањем делимичних збирова

<u>00001110</u> x <u>00001001</u>		14 × 9 (činioci)
00001110		
00000000		
00000000		Delimični
00001110		
00000000		proizvodi
00000000		
00000000		
00000000		
<u>00000000</u>		
0000000001111110		= 126 (rezultat)

# Множење неозначених бројева (2)

- Множење неким степеном двојке одговара померању регистра

00000000000001110 × 00001001

00000000000001110

00000000000000000

00000000000000000

00000000011100000

00000000000000000

00000000000000000

00000000000000000

00000000000000000

0000000001111110

00000000000001110 × 1 × 2<sup>0</sup>

00000000000001110 × 0 × 2<sup>1</sup>

00000000000001110 × 0 × 2<sup>2</sup>

00000000000001110 × 1 × 2<sup>3</sup>

00000000000001110 × 0 × 2<sup>4</sup>

00000000000001110 × 0 × 2<sup>5</sup>

00000000000001110 × 0 × 2<sup>6</sup>

00000000000001110 × 0 × 2<sup>7</sup>

# Множење неозначених бројева (3)

- Побољшања:
  1. Проблем је памћење делимичних резултата
    - Боље је имати само један међурезултат (активни резултат)
  2. Потребно је рачунати међузбирове само за позиције које одговарају бинарним јединицама
- Хардверску имплементацију у домену дигиталне логике
  - Овде дајемо само идеју



# Идеја хардверског множења неозн. бр.

- Потребна су три регистра А, М и Р и један једнобитни регистар С
  1. Множеник се уписује у М, а множилац у Р, А и С су на нули
  2. Бит множиоца са помера ка већој позицији (иницијално на најнижој)
    - Уколико је вредност бита 1, врши се сабирање М и А, иначе ништа
  3. Померање садржаја регистра С, А и Р удесно (сва три се посматрају спојено)
  4. Ако нису обрађени сви битови множиоца, враћамо се на корак 2.
  5. Завршено, вредност производа је уписана у регистрима А и Р (посматрају се као један регистар)

# Идеја хардверског множења неозн. бр. (2)

<i>M</i>	<i>C</i>	<i>A</i>	<i>P</i>	Komentar	
00001110	0	00000000	00001001	Početno stanje	M=14, P=9, C=0, A=0
00001110	0	00001110	00001001	$A = A + M$	Prvi ciklus
00001110	0	00000111	00000100	Pomeranje udesno	
00001110	0	00000111	00000100	Bez akcije	Drugi ciklus
00001110	0	00000011	10000010	Pomeranje udesno	
00001110	0	00000011	10000010	Bez akcije	Treći ciklus
00001110	0	00000001	11000001	Pomeranje udesno	
00001110	0	00001111	11000001	$A = A + M$	Četvrti ciklus
00001110	0	00000111	11100000	Pomeranje udesno	
00001110	0	00000111	11100000	Bez akcije	Peti ciklus
00001110	0	00000011	11110000	Pomeranje udesno	
00001110	0	00000011	11110000	Bez akcije	Šesti ciklus
00001110	0	00000001	11111000	Pomeranje udesno	
00001110	0	00000001	11111000	Bez akcije	Sedmi ciklus
00001110	0	00000000	11111100	Pomeranje udesno	
00001110	0	00000000	11111100	Bez akcije	Osmi ciklus
00001110	0	00000000	01111110	Pomeranje udesno	
		00000000	01111110	Rezultat	$14 \times 9 = 126$

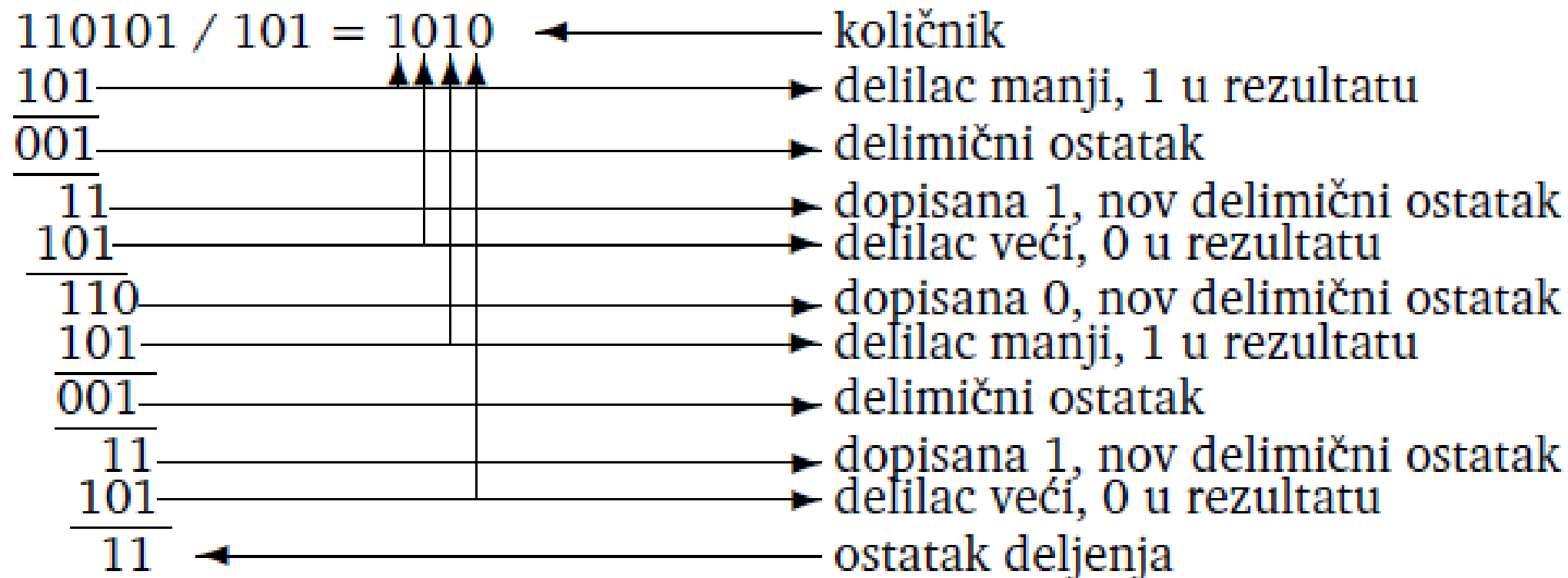
# Множење означених бројева у потпуном комплементу – Бутов алгоритам

- Потребна су четири регистра,  $A$ ,  $M$ ,  $P$  и једнобитни  $P_{-1}$ 
  1. Иницијализација као код претходног алгоритма
  2. Пореде се бит најмање тежине у  $P$  и вредност  $P_{-1}$ 
    - Ако су различити 01, онда се множеник и  $A$  саберу
    - Ако су различити 10, тада се множеник одузме од  $A$
    - Ако су једнаки (00 или 11), онда нема акције
  3. Померају се спојени регистри  $A$ ,  $P$  и  $P_{-1}$  удесно
  4. Ако нису обрађени сви битови множиоца, враћамо се на 2.
  5. Резултат је уписан у спојене регистре  $A$  и  $P$

# Множење означених бројева у потпуном комплементу – Бутов алгоритам (2)

<i>M</i>	<i>A</i>	<i>P</i>	<i>P</i> <sub>-1</sub>	Komentar	
11110010	00000000	11110111	0	Početno stanje	$M = -14, P = -9$
11110010	00001110	11110111	0	$A = A - M$	Prvi ciklus
11110010	00000111	01111011	1	Pomeranje udesno	
11110010	00000111	01111011	1	Bez akcije	Drugi ciklus
11110010	00000011	10111101	1	Pomeranje udesno	
11110010	00000011	10111101	1	Bez akcije	Treći ciklus
11110010	00000001	11011110	1	Pomeranje udesno	
11110010	11110011	11011110	1	$A = A + M$	Četvrti ciklus
11110010	11111001	11101111	0	Pomeranje udesno	
11110010	00000111	11101111	0	$A = A - M$	Peti ciklus
11110010	00000011	11110111	1	Pomeranje udesno	
11110010	00000011	11110111	1	Bez akcije	Šesti ciklus
11110010	00000001	11111011	1	Pomeranje udesno	
11110010	00000001	11111011	1	Bez akcije	Sedmi ciklus
11110010	00000000	11111101	1	Pomeranje udesno	
11110010	00000000	11111101	1	Bez akcije	Osmi ciklus
11110010	00000000	01111110	1	Pomeranje udesno	
		01111110	1	Rezultat	$-14 \times -9 = 126$

# Дељење неозначених бројева



## Дељење неозначених бројева (2)

- Три регистра  $P$ ,  $A$  и  $M$ , и бројач иницијално постављен на број битова у регистрима
- У  $P$  се уписује дељеник, у  $M$  делилац, а у  $A$  нула
  1.  $A$  и  $P$  се померају улево за један бит
  2. Од  $A$  се одузима  $M$
  3. Ако је  $A$  веће или једнако од  $0$ , у бит најмање тежине  $P$  се уписује  $1$ , а иначе  $0$ . Садржај  $M$  се сабира са  $A$  ради враћања претходног стања.
  4. Вредност бројача се смањује за  $1$  и ако је бројач већи од  $0$ , врамо се на корак  $1$
  5. Завршено, количник је уписан у регистру  $P$ , а остатак у  $A$

# Дељење неозначених бројева (3)

- Три регистра  $P$ ,  $A$  и  $M$ , и бројач иницијално постављен на број битова у регистрима
- У  $P$  се уписује дељеник, у  $M$  делилац, а у  $A$  нула
  1.  $A$  и  $P$  се померају улево за један бит
  2. Од  $A$  се одузима  $M$
  3. Ако је  $A$  веће или једнако од  $0$ , у бит најмање тежине  $P$  се уписује  $1$ , а иначе  $0$ . Садржај  $M$  се сабира са  $A$  ради враћања претходног стања.
  4. Вредност бројача се смањује за  $1$  и ако је бројач већи од  $0$ , врамо се на корак  $1$
  5. Завршено, количник је уписан у регистру  $P$ , а остатак у  $A$



# Дељење неозначених бројева (4)

<i>M</i>	<i>A</i>	<i>P</i>	Komentar
Primer 1: 24/9			
00001001	00000000	00011000	Početno stanje, 24/9
00001001	00000000	00110000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000000	00110000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00000000	01100000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000000	01100000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00000000	11000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000000	11000000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00000001	10000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000001	10000000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00000011	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000011	00000000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00000110	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000110	00000000	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
00001001	00001100	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000011	00000001	$P_0 \leftarrow 1$
00001001	00000110	00000000	Pomeranje ulevo
	00001001		Oduzimanje $A \leftarrow A - M$
00001001	00000110	00000010	$P_0 \leftarrow 0$ , restauracija sadržaja <i>A</i>
	00000110	00000010	Količnik = 2, ostatak = 6



# Рачунарска аритметика

Бинарно кодирани декадни бројеви

# Бинарно кодирани декадни бројеви (BCD)

- Мотивација: омогућавање тачног записа мешовитих бројева
  - Видели смо раније примере када неки разломљени број не можемо тачно запамтити
- Сваку цифру чувамо засебно са најмање 4 бинарне цифре
- Кодирање декадних цифара мора бити једнозначно
- Пожељне особине за извођење аритметичких операција:
  - Највећој цифри (9) придружити највећу бинарну вредност
  - (Не)парним декадним одговарају (не)парни бинарни
  - Комплементарност кода – ако су  $a+b=9$ , онда и бинарне кодне репрезентације треба да буду комплементарне

# Примери кодирања

Dekadna cifra	Binarni kod						
	8421	2421	5421	753-6	84-2-1	višak 3	ciklički
0	0000	0000	0000	0000	0000	0011	0001
1	0001	0001	0001	1001	0111	0100	0101
2	0010	0010	0010	0111	0110	0101	0111
3	0011	0011	0011	0010	0101	0110	1111
4	0100	0100	0100	1011	0100	0111	1110
5	0101	1011	1000	0100	1011	1000	1100
6	0110	1100	1001	1101	1010	1001	1000
7	0111	1101	1010	1000	1001	1010	1001
8	1000	1110	1011	0110	1000	1011	1011
9	1001	1111	1100	1111	1111	1100	0011

# Грејов код

- Грејов код поставља услов да бинарне репрезентације узастопних декадних бројева имају разлику на само једном биту (практично?)
- Постоји више начина да се конструише Грејов код
  - Функција кодирања није јединствена
- Често употребљавана Грејова функција кодирања је:

$$G(n, i) =_n i \oplus_n [i/2]$$

- Где је  $n$  – дужина записа, а  $i$  је број који представљамо, док је  $_n i$  одговарајућа бинарна репрезентација броја  $i$
- Заокружени знак плус је ексклузивна дисјункција

## Грејов код (2)

Хексадекадна cifra	Бинарна vrednost	Грејов kod	Хексадекадна cifra	Бинарна vrednost	Грејов kod
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	A	1010	1111
3	0011	0010	B	1011	1110
4	0100	0110	C	1100	1010
5	0101	0111	D	1101	1011
6	0110	0101	E	1110	1001
7	0111	0100	F	1111	1000

# Означени бинарно кодирани декадни бр.

## 1. Знак и апсолутна вредност

- За запис знака се оставља додатна декадна цифра
- Које ће вредности представљати знакове зависи од имплементације

## 2. 10-ти комплемент

- Негација броја се добија комплентирањем према основи 10
- То значи да ће нула означавати позитивне, а деветка негативне бројеве
- Најчешће се користи запис знак и апсолутна вредност

# Чен-Хо кодирање

- Кодира три декадне цифре у 10 бита (оптимизовано)
- 20% ефикасније од BCD записа
- Дели цифре на мале (0-7) и велике (8,9)
  - За мале довољна три бита, а за велике један за међусобну разлику
- Примери комбинација:
  - Све три цифре мале:  $3 + 3 + 3$  за цифре + 1 бит да означи комбинацију
  - Две цифре мале:  $3 + 3 + 1$  за цифре + 3 бита за комбинацију
  - Једна цифра мала:  $3 + 1 + 1$  за цифре + 5 битова за комбинацију
  - Све цифре велике:  $1 + 1 + 1$  за цифре + 5 за комбинацију (2 су вишак)

# Чен-Хо кодирање (2)

1024 стања	Бинарно кодирање										Децимални запис			
	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	d2	d1	d0	Вредности
50.0% (512)	0	a	b	c	d	e	f	g	h	i	0abc	0def	0ghi	(0–7) (0–7) (0–7)
37.5% (384)	1	0	0	c	d	e	f	g	h	i	100c	0def	0ghi	(8–9) (0–7) (0–7)
	1	0	1	c	a	b	f	g	h	i	0abc	100f	0ghi	(0–7) (8–9) (0–7)
	1	1	0	c	d	e	f	a	b	i	0abc	0def	100i	(0–7) (0–7) (8–9)
9.375% (96)	1	1	1	c	0	0	f	a	b	i	0abc	100f	100i	(0–7) (8–9) (8–9)
	1	1	1	c	0	1	f	d	e	i	100c	0def	100i	(8–9) (0–7) (8–9)
	1	1	1	c	1	0	f	g	h	i	100c	100f	0ghi	(8–9) (8–9) (0–7)
3.125% (8 од 32)	1	1	1	c	1	1	f	0	0	i	100c	100f	100i	(8–9) (8–9) (8–9)



# DPD

- Densely Packed Decimal
- Слично као Чен-Хо, али користи другачије уређење битова
- Кодира три декадне цифре у 10 бита
- Иста идеја да се велике цифре представљају једним битом, а мале са три бита
- Предности у односу на Чен-Хо:
  - 1 дек. цифра се може кодирати у 4 бита, а 2. дек. цифре у 7 бита, нпр. за 38 цифара Чен-Хо користи 130 бита (13 триплета), док DPD користи 127 бита
  - Кодирање 1 или 2 дек. цифре је десно поравнато, док су остали битови 0, због тога се проширење записа лако извршава

# DPD (2)

DPD кодирање										Децимални запис			
b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	d2	d1	d0	Вредности
a	b	c	d	e	f	0	g	h	i	0abc	0def	0ghi	(0-7) (0-7) (0-7)
a	b	c	d	e	f	1	0	0	i	0abc	0def	100i	(0-7) (0-7) <u>(8-9)</u>
a	b	c	g	h	f	1	0	1	i	0abc	100f	0ghi	(0-7) <u>(8-9)</u> (0-7)
g	h	c	d	e	f	1	1	0	i	100c	0def	0ghi	<u>(8-9)</u> (0-7) (0-7)
g	h	c	0	0	f	1	1	1	i	100c	100f	0ghi	<u>(8-9)</u> <u>(8-9)</u> (0-7)
d	e	c	0	1	f	1	1	1	i	100c	0def	100i	<u>(8-9)</u> (0-7) <u>(8-9)</u>
a	b	c	1	0	f	1	1	1	i	0abc	100f	100i	(0-7) <u>(8-9)</u> <u>(8-9)</u>
x	x	c	1	1	f	1	1	1	i	100c	100f	100i	<u>(8-9)</u> <u>(8-9)</u> <u>(8-9)</u>

# Упоредна табела различитих кодирања

Број	BCD	Чен-Хо	DPD
5	0000 0000 0101	000 000 0101	000 000 0101
9	0000 0000 1001	110 000 0001	000 000 1001
55	0000 0101 0101	000 010 1101	000 101 0101
79	0000 0111 1001	110 011 1001	000 111 1001
80	0000 1000 0000	101 000 0000	000 000 1010
99	0000 1001 1001	111 000 1001	000 101 1111
555	0101 0101 0101	010 110 1101	101 101 0101
999	1001 1001 1001	111 111 1001	001 111 1111

# Рачунарска аритметика

Реални бројеви



# Стандард IEEE754-2008

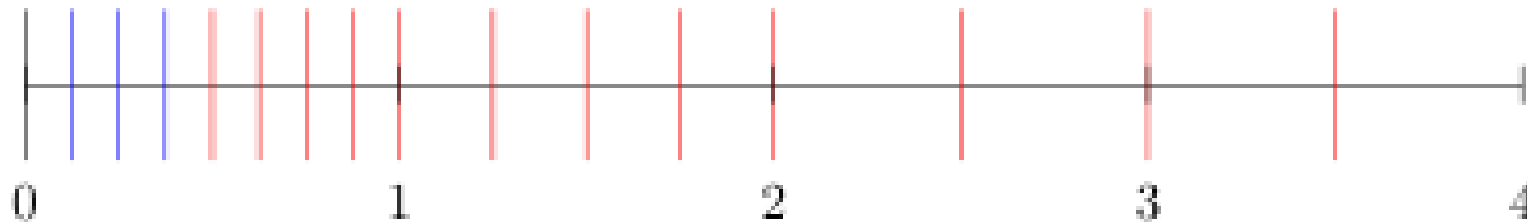
- Прописује репрезентације реалних бројева у покретном зарезу
- Основе  $B=2$  и  $B=10$
- Прописује операције:
  - Сабирања и дузимања
  - Множења и дељења
  - Спојеног вишеструког сабирања
  - Кореновања
  - Рачунања остатка при дељењу
  - Поређења два броја
  - Конверзије из (и у) целобројну вредност
  - Конверзије између различитих формата у покретном зарезу, ...

# IEEE754-2008 - репрезентација

- Број се представља на један од три начина:
  - Уређена тројка (знак, експонент, значајни део броја)  
 $(-1)^{\text{знак}} \times B^{\text{експонент}} \times [\text{значајни део}]$
  - $+\infty$ ,  $-\infty$
  - qNaN (тихи NaN) и sNaN (сигнални NaN)
- Основе су 2 или 10
- Величина значајног дела је одређена параметром  $p$

# IEEE754-2008 – опсег вредности

- Максимална вредност експонента је  $e_{max}$ , а минимална  $e_{min}=1-e_{max}$
- Бројеви у интервалу  $[V^{e_{min}}, V^{e_{max}} \times (V-V^{1-p})]$  се називају нормални
- Бројеви мањи по апсолутној вредности од  $V^{e_{min}}$  називају се субнормални
  - Значајне цифре имају водеће нуле, нпр. број  $2^{-10}$  ако је  $e_{min}=-9$ ,  $V=2$





# IEEE754-2008 – класе података

- Класе података прописане стандардом:
  - Нормални бројеви
  - Субнормални бројеви
  - NaN
  - Бесконачно
  - Означена нула

# Реални бројеви у декадној основи

## 1. Софтверски

- Запис и операције се реализују софтверски
- Неефикасно

## 2. Хардверски

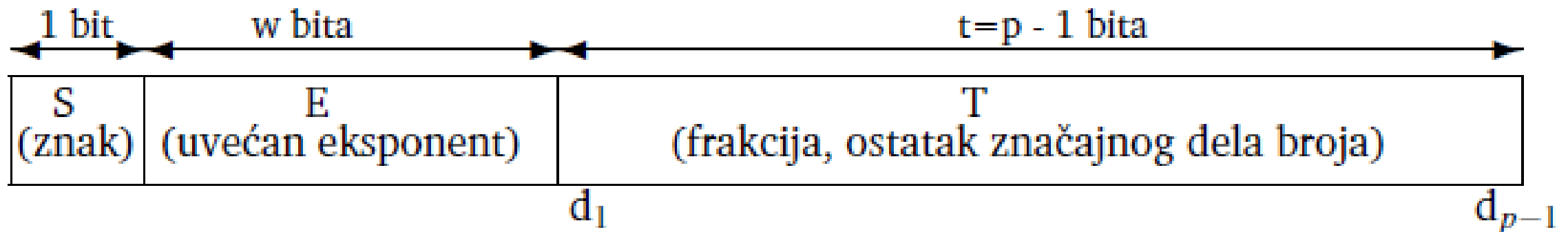
- Помоћу VCD записа
  - Фиксни зарез
  - Сложено извођење операција (прескочили смо овај део)
  - Знатно спорије од рада у бинарној основи
- Помоћу записа у покретном зарезу (IEEE754-2008)
  - Мали број процесора подржава овакав запис

# IEEE754-2008 – преглед формата записа

Parametri	Formati				
	Sa binarnom osnovom ( $\beta=2$ )			Sa dekadnom osnovom ( $\beta=10$ )	
	binary32	binary64	binary128	decimal64	decimal128
p	24	53	113	16	34
emax	+127	+1023	+16383	+384	+6144

# IEEE754-2008 binary32

- Експонент може бити записан или као означен или као неозначен
- Ако је неозначен, онда се права вредност добија одузимањем  $2^w-1=127$
- $d_0$  се не записује, јер је имплицитно 1



# IEEE754-2008 binary32

	Znak	EkspONENT	Frakcija
+15 =	0	10000010	111000000000000000000000
-15 =	1	10000010	111000000000000000000000
+1/64 =	0	01111001	000000000000000000000000
+0 =	0	00000000	000000000000000000000000
-0 =	1	00000000	000000000000000000000000
$(1 - 2^{-24}) \times 2^{+128}$ =	0	11111110	111111111111111111111111
$+1 \times 2^{-126}$ =	0	00000001	000000000000000000000000
$+1 \times 2^{-149}$ =	0	00000000	000000000000000000000001

Provera vrednosti zapisa broja +15:

- Znak = +
- EkspONENT = 3 (=130-127)
- 1,frakcija =  $(1,111)_2 = 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$
- Vrednost = Znak 1,frakcija \*  $2^{\text{ekspONENT}}$  =  $+(+1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}) * 2^3$   
=  $+1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 4 + 2 + 1 = +15$

# IEEE754-2008 decimal64

Знак	Комбинација	Део експонента	Значајне цифре
1 бит	5 битова	8 битова	50 битова
s	mmmm	xxxxxxxx	cc

- Из битова комбинације се позајмљује у одређеним случајевима 2 бита за експонент {00,01,10} (експонент се записује као неозначен)
- Највећи експонент је дакле  $(101111111)_2 = (767)_{10}$
- Вредност броја се рачуна по формули:

$$(-1)^{\text{знак}} \times 10^{\text{експонент}-383} \times [\text{значајни део}]$$

# IEEE754-2008 decimal64 (2)

Комбинација	Додатни битови експонента	Додатни значајни битови	Значење
00mmm	00	0xxx	Регуларно
01mmm	01	0xxx	Регуларно
10mmm	10	0xxx	Регуларно
1100m	00	100x	Регуларно
1101m	01	100x	Регуларно
1110m	10	100x	Регуларно
11110	—	—	$\pm\infty$
11111	—	—	NaN

# IEEE754-2008 сабирање и одузимање

- Ако су:  $x = x_s \times \beta^{x_e}, y = y_s \times \beta^{y_e}$
- Онда се операције сабирања и одузимања у општем случају раде:

$$x \pm y = \begin{cases} (x_s \times \beta^{x_e - y_e} \pm y_s) \times \beta^{y_e} & \text{ако } x_e \leq y_e \\ (x_s \pm y_s \times \beta^{y_e - x_e}) \times \beta^{x_e} & \text{ако } y_e < x_e \end{cases}$$

- Треба водити рачуна о специјалним вредностима (бесконачности)
- Сабирање значајних делова се врши према правилима за сабирање целих бројева у запису знак и апсолутна вредност
- Покушава се нормализација броја (тражи експонент тако да  $d_0$  буде имплицитно 1)



# IEEE754-2008 множење и дељење

- Формула:

$$x * y = (x_s * y_s) \times \beta^{x_e + y_e}$$

$$x / y = (x_s / y_s) \times \beta^{x_e - y_e}$$

- И овде се води рачуна о специјалним вредностима
- Множење значајних делова се врши према правилима множења целих бројева у запису знак и апсолутна вредност
- Покушава се нормализација броја