5. Fejezet : Lebegőpontos számok

The Architecture of Computer Hardware and Systems Software:
An Information Technology Approach

3. kiadás, Irv Englander

John Wiley and Sons ©2003

Wilson Wong, Bentley College
Linda Senne, Bentley College

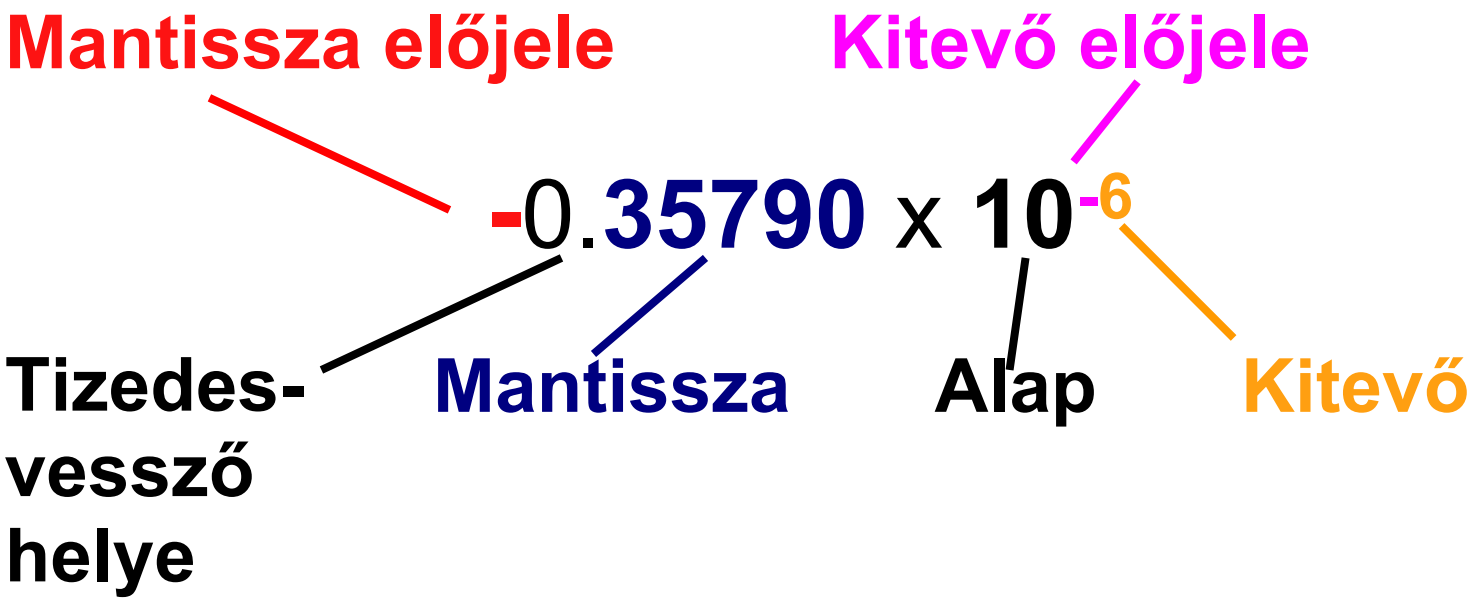
Lebegőpontos számok

- Valós számok
- Akkor használjuk a számítógépeknél, ha
  - Kívül esik a számítógép integer határán (túl kicsi vagy túl nagy)
  - Tizedes törtet tartalmaz

Exponenciális jelölés

- Hívják még *tudományos jelölésnek*
  - 12345
  - 12345×10^{0}
  - 0.12345×10^{5}
  - 123450000×10^{-4}
- 4 alkotóelem kell egy számhoz
  1. Előjel (pl.: ”+”)
  2. Érték vagy *mantissza* (pl.: 12345)
  3. Kitevő előjele (pl.: ”+” a 10^{5} -en)
  4. Kitevő értéke (pl.: 5)
- Plusz
  5. Kitevő alapja (pl.: 10)
  6. Tizedesvessző helye (vagy más bázisé) alappont

Szabályok összegzése



Formai leírás

- Előre definiált forma, általában 8 bit
  - Megnövelt értékhatár (két jegyű kitevő) kisebb pontosság árán (ötjegyű mantissza)

Forma

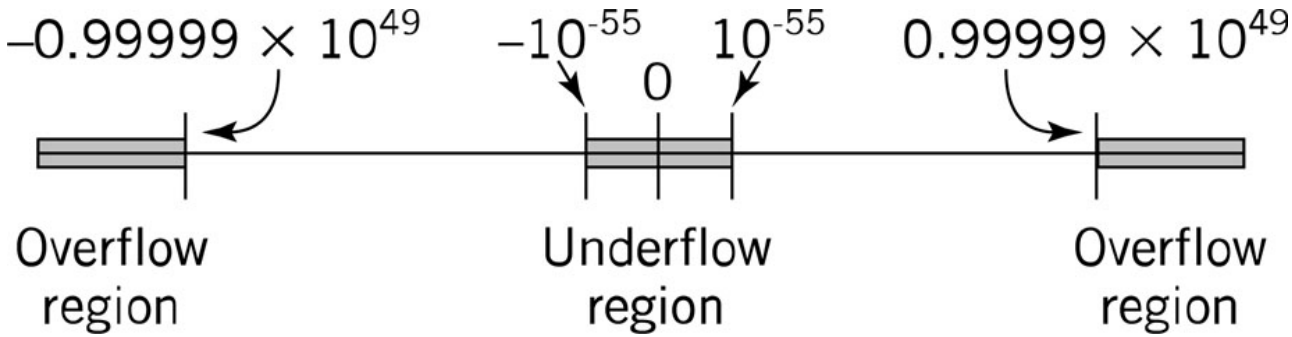
- Mantissza: előjel karakter az előjel-érték formában
- Tizedespont a mantissza elején helyezkedik el
- Excess-N jelölés: komplemens ábrázolás
  - Vegyük az eltolást a középértéknek, ahol N a középérték

Ábrázolás	0	49	50	99
Ábrázolt kitevő	-50	-1	0	49

– Növekvő érték +

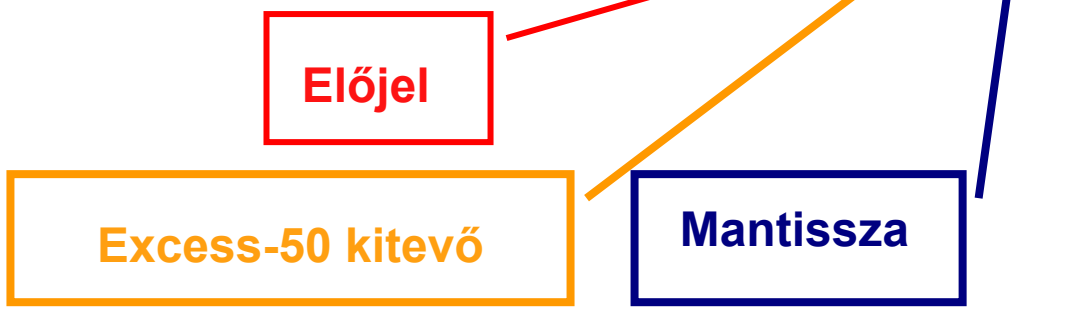
Túlcsordulás és alulcsordulás

- Akkor fordul elő, ha az adott számábrázolást tekintve túl kicsi vagy túl nagy számokat szeretnénk kezelni



Példa 1: 246.8035

1. Adjunk kitevőt 246.8035×10^0
2. Pozícionáljuk a tizedesvesszőt a normál alakhoz $.2468035 \times 10^3$
3. Vágjuk le 5 jegyűre $.24680 \times 10^3$
4. Szám átalakítása 05324680



Példa 2: 1255 x 10^{-3}

1. Már exponenciális formájú	1255 x 10^{-3}
2. Pozícionáljuk a tizedesvesszőt	.1255 x 10^{+1}
3. Normál alakú	
4. Adjunk hozzá 0-t az 5 jegyhez	0.1255 x 10^{+1}
5. Szám átalakítása	05112550

Példa 3: - 0.00000075

1. Exponenciális jelölés — $- 0.00000075 \times 10^{0}$
2. Tizedespont a helyén
3. Normál alakra hozás — $- 0.75 \times 10^{-6}$
4. 0 hozzáadása az 5 jegyhez — $- 0.75000 \times 10^{-6}$
5. Szám átalakítása — 54475000

Átalakítás példák

$05324567 = 0.24567 \times 10^{3} = 245.67$

$54810000 = -0.10000 \times 10^{-2} = -0.0010000$

$55555555 = -0.55555 \times 10^{5} = -55555$

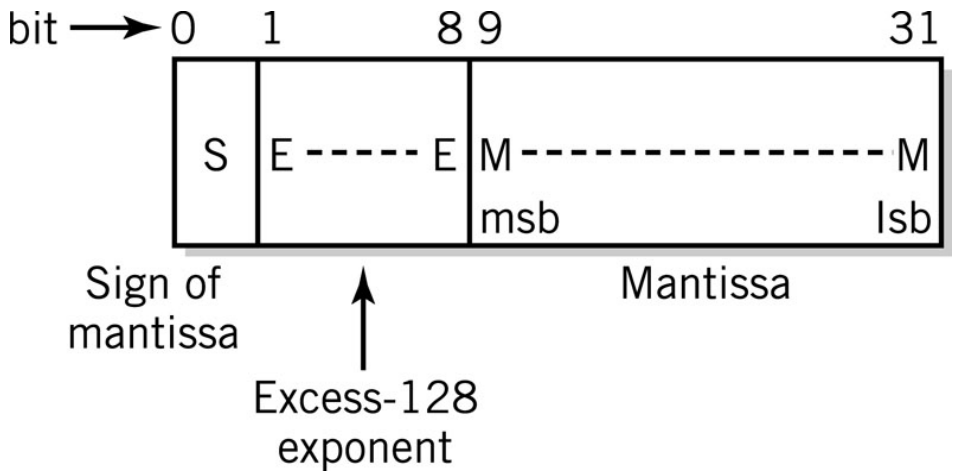
$04925000 = 0.25000 \times 10^{-1} = 0.025000$

Normál alakra hozás

- Toljuk el balra a számokat növelve a kitevőt, míg a kezdő nullák el nem tűnnek
- Decimális számok konvertálása általános formába
  1. Lássuk el számmal a kitevőt (0 ha még nincs meghatározva)
  2. Növeljük/csökkentsük a kitevő értékét a tizedes pont eltolásával a helyes irányba
  3. Csökkentsük a kitevőt, hogy eltüntessük a nullákat a mantisszából
  4. Javítsunk a pontosságon 0-kat írva vagy elhagyva/kerekítve a kevésbé fontos számjegyeknél

Lebegőpont a számítógépben

- Tipikus lebegőpontos forma
  - 32 biten a mérete ~10^{-38}-tól 10^{+38}-ig terjed
  - 8 bites kitevő = 256 szint
    - Excess-128 jelölés
  - 23/24 bites mantissza: körülbelül 7 decimális jegy pontosságú



Lebegőpont a számítógépben

Excess-128 kitevő

mantissza előjele		mantissza
0	1000 0001	1100 1100 0000 0000 0000 000
		+1.1001 1000 0000 0000 00 azaz: +0.M x 2^{+1}
1	1000 0100	1000 0111 1000 0000 0000 000
		-1000.0111 1000 0000 0000 000 azaz: -0.M x 2^{+4}
1	0111 1110	1010 1010 1010 1010 10101 101
		-0.0010 1010 1010 1010 1010 1 azaz: -0.M x 2^{-2}

IEEE 754 szabvány

Pontosság	Egyszerű (32 bit)	Dupla (64 bit)
Előjel	1 bit	1 bit
Kitevő	8 bit	11 bit
Jelölés	Excess-127	Excess-1023
Mutató alap	2	2
Terjedelem	2^{-126}-tól 2^{127}-ig	2^{-1022}-től 2^{1023}-ig
Mantissza	23	52
Decimális pontosság	≈ 7	≈ 15
Értékhatár	≈ 10^{-45}-tól 10^{38} -ig	≈ 10^{-300}-tól 10^{300} -ig

IEEE 754 szabvány

- 32 bit-es lebegő pontos érték definíció

Kitevő	mantissza	Érték
0	±0	0
0	Nem 0	$\pm 2^{-126} \times 0.M$
1^{-254}	Bármi	$\pm 2^{(E-127)} \times 1.M$
255	±0	±∞
255	Nem 0	Speciális eset

Átalakítás: 10-es és 2-es alap

- Két lépés
  - A számok egész és tört részét külön kell átalakítani, beágyazott tizedes vagy bináris ponttal
  - Az átalakítás végrehajtása előtt az exponenciális alakban lévő számokat vissza kell alakítani egyszerű decimális vagy bináris kevert számmá vagy törtté

Átalakítás: 10-es és 2-es alap

- 253.75_{10} átalakítása bináris lebegőpontos formába
  - Szám szorzása 100-zal — 25375
  - Binárissá alakítás — 110 0011 0001 1111 vagy 1.1000 1100 0111 11 x 2^{14}
  - IEEE ábrázolás — 0 1000 1101 10 0011 0001 1111

Előjel

Excess-127 kitevő = 127 + 14

mantissza

- Az eredeti decimális értéket visszakapjuk, ha elvégzünk egy 100_{10}-zal való osztást.

Excess 128-as illetve 127-es alak

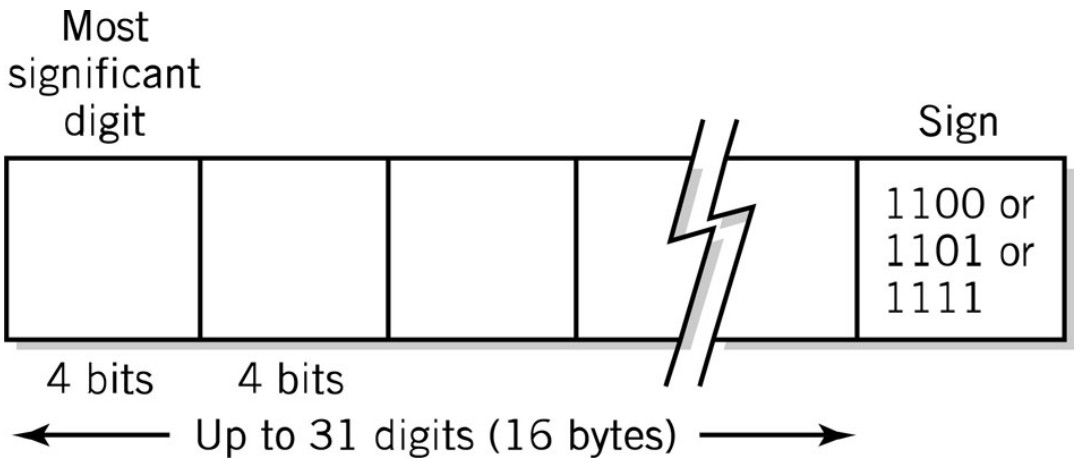
- 128-as esetén:
  - valódi kitevő = szimbolikus kitevő - 128
- 127-es esetén:
  - valódi kitevő = szimbolikus kitevő - 128+1 azaz:
  - valódi kitevő = szimbolikus kitevő - 127

Excess 128-as illetve 127-es alak

- Például, ha a pozitív szimbolikus kitevő:
  - 1000 0001 akkor a:
  - vk = 129 - 128 = +1 => (0.M * 2^{+1}) 128-as alak esetén,
  - vk = 129 - 127 = +2 => (1.M * 2^{+2}) 127-es alak esetén.
- Például, ha a negatív szimbolikus kitevő:
  - 0111 1110 akkor a:
  - vk = 126 - 128 = -2 => (0.M * 2^{-2}) 128-as alak esetén,
  - vk = 126 - 127 = -1 => (1.M * 2^{-1}) 127-es alak esetén.

Tömörített decimális forma

- Dollárok és centek ábrázolása valós számokkal
- A vállalat-orientált nyelvek támogatják, mint a COBOL
- IBM 370/390 és Compaq Alpha



Programozási megfontolás

- Integer előnyei
  - A számítógép könnyebben használja
  - Precízebb
  - Gyorsabb
  - Kisebb tárhely és idő igény
- A legtöbb magas szintű nyelv kettő vagy több formát szolgáltat
  - Rövid egész (16 bit)
  - Hosszú egész (64 bit)

Programozási megfontolás

- Valós számok
  - Változónak vagy konstansnak van tört része
  - A nagyon kicsi vagy nagyon nagy értékű számok kívül esnek az integer ábrázolási határán
  - A program lehet, hogy nem lesz elég precíz a feladat végrehajtásához
  - Tömörített decimális számábrázolás vonzóbb a vállalati alkalmazásoknál