Signálové procesory

Reprezentácia čísiel a numerické operácie v DSP

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY
KATEDRA RÁDIOELEKTRONIKY
Laboratórium DSP a mikroradičov

:: Zobrazenie čísiel v mikroprocesorovej technike

- Celé nezáporné čísla - čísla bez znamienka
  - dvojková sústava
  - iné používané sústavy - šestnástková a osmičková
- Všetky celé čísla - čísla so znamienkom
  - priamy kód
  - jednotkový doplnok
  - dvojkový doplnok
  - kód posunutej nuly (kód "excess N")
- Reálne čísla (v skutočnosti podmnožina racionálnych čísiel)
  - s pevnou čiarkou
  - s pohyblivou čiarkou

:: Celé čísla so znamienkom - priamy kód

- oddelíme jeden bit pre znamienko. V N bitoch teda použijeme
  - *N-1 bitov na absolútnu hodnotu čísla*
  - *1 bit na znamienko: 0 znamená plus, 1 znamená mínus*

$$5 \approx 00000101$$

$$-5 \approx 10000101$$

- toto zobrazenie sa v praxi používa zriedka, lebo:
  - *sčítačka, ktorá vie sčitovať čísla bez znamienka sa nedá použiť na takéto čísla → je nevyhnutný zložitejší obvod*
  - *existuje v ňom kladná aj záporná nula*

:: Celé čísla so znamienkom - jednotkový doplnok

Binárna hodnota	Jednotkový doplnok	Neznamienková interpretácia
00000000	+0	0
00000001	1	1
...	...	...
01111101	125	125
01111110	126	126
01111111	127	127
10000000	−127	128
10000001	−126	129
10000010	−125	130
...	...	...
11111110	−1	254
11111111	−0	255

- jednotkový doplnok (nájdenie zápornej reprezentácie daného čísla) vytvoríme bitovou negáciou daného čísla
- takýto systém reprezentácie záporných čísiel zavádza dve nuly:
- kladnú +0 = 00000000
- a zápornú -0 = 11111111
- aritmetika v takomto systéme pracuje podľa očakávania, ale pri výpočtoch musíme vziať do úvahy prenos do vyššieho rádu

Využívali ho staršie výpočtové systémy.

V súčasnosti sa využíva v algoritmoch kontrolných súm sieťových protokolov.

'"binárne	dekadicky"'
11111110	-1
+ 00000010	+2
.....	...
1 00000000	0 <-- nie je správne
1	+1 <-- pripočitame prenos
.....	...
00000001	1 <-- správny výsledok

Zaveďme operáciu $\oplus$ ako sčítanie modulo 5, teda a $\oplus$ b=(a+b) mod 5, v takejto aritmetike potom platí napríklad:

$1 \oplus 1=2$, $1 \oplus 2=3$, $1 \oplus 3=4$, $1 \oplus 4=0$, $2 \oplus 2=4$, $2 \oplus 3=0$, $0 \oplus 0=0$

Z červeno označených rovností vidíme, že pri takejto operácii sa číslo 4 správa ako číslo opačné k 1, teda **4 sa správa ako -1 a podobne 3 sa správa ako -2 a 0 je opačná sama k sebe.**

Vo všeobecnosti číslo (5-x) mod 5 sa chová vzhľadom k operácii $\oplus$ ako číslo opačné k x.

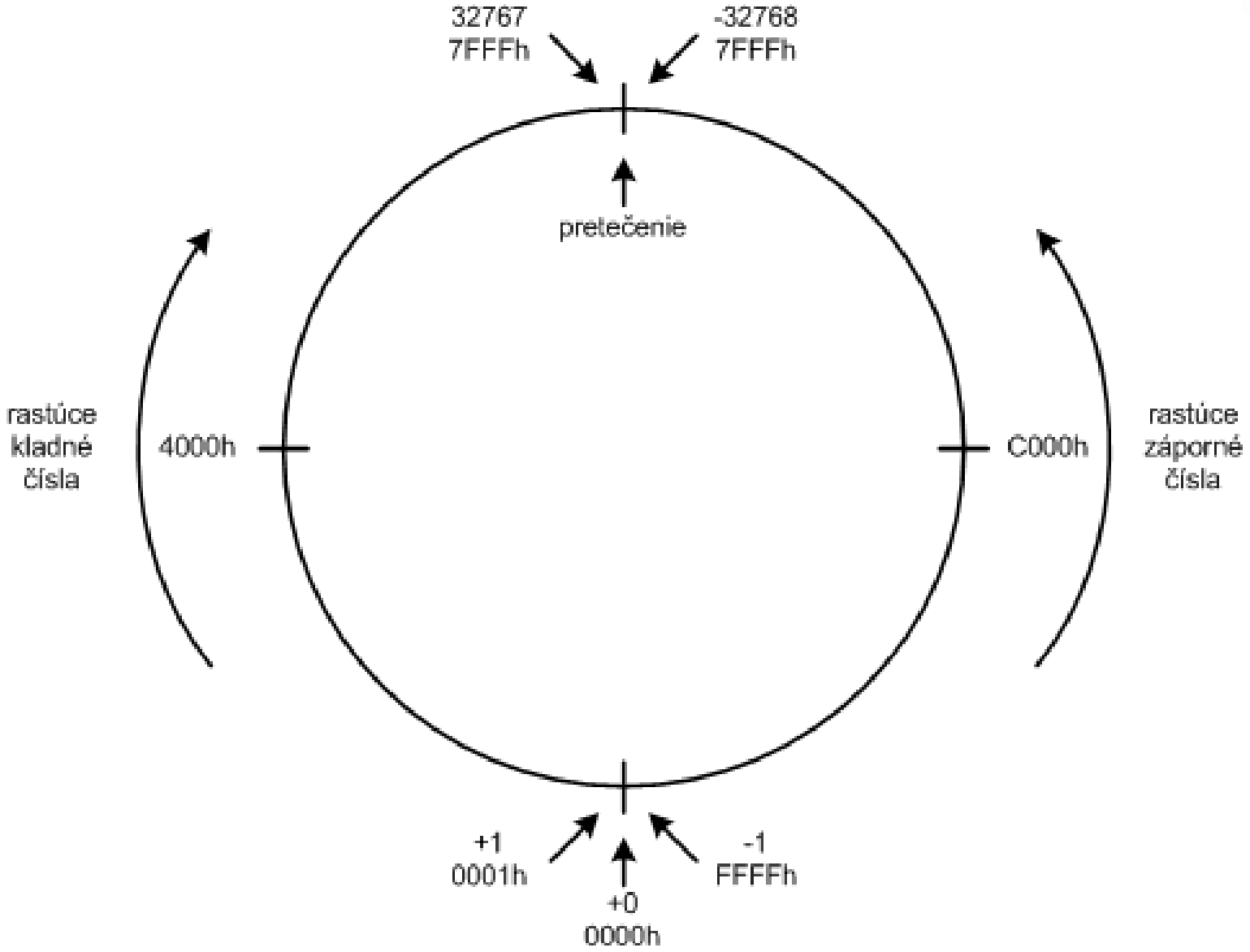
N-bitová sčítačka implementovaná v CPU procesora počíta na N dvojkových miest – **jedná sa teda o aritmetiku modulo 2^N.**

Ak teda zobrazíme -1 ako 2^n-1, -2 ako 2^n-2 atď., tak nemusíme vôbec meniť sčítačku - bude vedieť sčitovať kladné aj záporné čísla.

:: Celé čísla so znamienkom - dvojkový doplnkový kód

znamienkový bit (MSb)									
0	1	1	1	1	1	1	1	=	127
0	1	1	1	1	1	1	0	=	126
0	0	0	0	0	0	1	0	=	2
0	0	0	0	0	0	0	1	=	1
0	0	0	0	0	0	0	0	=	0
1	1	1	1	1	1	1	1	=	−1
1	1	1	1	1	1	1	0	=	−2
1	0	0	0	0	0	0	1	=	−127
1	0	0	0	0	0	0	0	=	−128

:: Celé čísla so znamienkom - dvojkový doplnkový kód



:: Celé čísla so znamienkom - dvojkový doplnkový kód

Príklad: ako je zobrazené číslo -10 v dvojkovom doplnkovom kóde s dĺžkou 8 bitov?

Metóda č. 1: -10 bude v 8 bitoch zobrazené ako 2^8 - 10 = 256 – 10 = 246.
Preveďme teda 246 do dvojkovej sústavy: $\mathbf{246_{10}=11110110_2=F6_{16}}$

Skúška správnosti, 10-10:

	0	0	0	0	1	0	1	0
+	1	1	1	1	0	1	1	0
	1	0	0	0	0	0	0	0

Metóda č. 2: číslo zmeníme na opačné tak, že v jeho dvojkovom zápise vykonáme bitovú negáciu (jednotkový doplnok) a potom k číslu pripočítame 1

10_{10}=	0	0	0	0	1	0	1	0
	1	1	1	1	0	1	0	1
+	0	0	0	0	0	0	0	1
	1	1	1	1	0	1	1	0

bitová negácia

pripočítame 1

Je to najpoužívanejší kód pre kódovanie celých čísiel v súčasných procesoroch!

:: Celé čísla so znamienkom - dvojkový doplnkový kód

- na sčítanie a odčítanie možno použiť rovnaký hardvér ako pre čísla bez znamienka
- pre násobenie a delenie ale treba vedieť, či sú čísla bez znamienka alebo so znamienkom (výsledok násobenia sa vždy zapisuje do dvojnásobného počtu bitov):
  - neznamienkovo: 05*FF=5*255=1275=04FB
  - znamienkovo: 05*FF=5*(-1)=-5=FFFB
- aj pri porovnaniach treba vedieť, či ide o čísla so znamienkom alebo bez znamienka:
  - neznamienkovo: 05 < FF lebo 5 < 255
  - znamienkovo: 05 > FF lebo 5 > -1

:: Celé čísla so znamienkom - dvojkový doplnkový kód

Pri sčítaní a odčítaní pracuje dvojkový doplnok spoľahlivo.
Ako je to však pri násobení?

Motivačný príklad: Vypočítajme súčin čísiel +4 a -3!

	0	1	0	0	+4
x	1	1	0	1	-3

:: Celé čísla so znamienkom - dvojkový doplnkový kód

Pri sčítaní a odčítaní pracuje dvojkový doplnok spoľahlivo.
Ako je to však pri násobení?

Motivačný príklad: Vypočítajme súčin čísiel +4 a -3!

			0	1	0	0	+4	
		x	1	1	0	1	-3	
			0	1	0	0		
		0	0	0	0			
	0	1	0	0				
1	1	0	0				dvojkový doplnok k +4	
1	1	1	0	1	0	0	-12	

:: Celé čísla so znamienkom - dvojkový doplnkový kód

Pri sčítaní a odčítaní pracuje dvojkový doplnok spoľahlivo.
Ako je to však pri násobení?

Motivačný príklad: Vypočítajme súčin čísiel +4 a -3!

	0 1 0 0	+4
x	1 1 0 1	-3
	0 1 0 0	
	0 0 0 0	
	0 1 0 0	
1 1 0 0		dvojkový doplnok k +4
1 1 1 0 1 0 0		-12

Čo sa stane ak výsledok uložíme do 8-bitového akumulátora?

0 1 1 1 0 1 0 0

Potrebujeme znamienkové rozšírenie!

1 1 1 1 0 1 0 0

:: Celé čísla so znamienkom - dvojkový doplnkový kód

Pri sčítaní a odčítaní pracuje dvojkový doplnok spoľahlivo.
Ako je to však pri násobení?

Ďalší motivačný príklad: Vypočítajme súčin čísiel -3 a -3!

	1	1	0	1	-3
x	1	1	0	1	-3
	1	1	0	1	
0	0	0	0		
1	1	0	1		
0	0	1	1		dvojkový doplnok k -3
1	0	1	1	0	0	1	89 – nesprávne

:: Celé čísla so znamienkom - dvojkový doplnkový kód

Pri sčítaní a odčítaní pracuje dvojkový doplnok spoľahlivo.
Ako je to však pri násobení?

Ďalší motivačný príklad: Vypočítajme súčin čísiel -3 a -3!

				1	1	0	1	-3	
			x	1	1	0	1	-3	
znamienkové rozšírenie	1	1	1	1	1	0	1		
	0	0	0	0	0	0			
	1	1	1	0	1				
	0	0	1	1				dvojkový doplnok k -3	
1	0	0	0	0	1	0	0	1	9 – správne

:: Celé čísla so znamienkom - kód posunutej nuly

- číslo x zobrazíme ako x+S kde S je pevne dané číslo
- napríklad pre 8 bitov a S=127 dostaneme takéto kódovanie (nazývané aj „excess 127“):

číslo	zobrazené	dvojkovo
-127	0	00000000
-126	1	00000001
...		
-1	126	01111110
0	127	01111111
1	128	10000000
2	129	10000001
...		
127	254	11111110
128	255	11111111

- výhodou je, že pre porovnávanie zakódovaných čísiel netreba poznať ich kódovanie (menšie čísla sa zobrazia do menších, väčšie do väčších)
- používa sa pre špeciálne účely (napr. exponent čísla v pohyblivej čiarke)

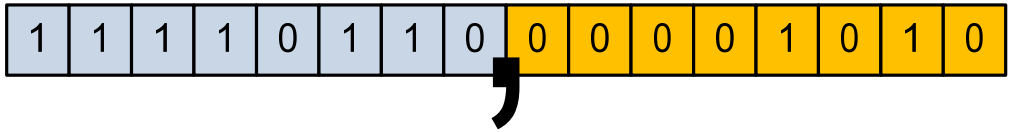
:: Reálne čísla - zobrazenie s pevnou čiarkou

- v dvojkovej sústave sa dajú zobraziť aj necelé čísla rovnako ako v desiatkovej sústave:

  $12.35_{10} = 1*10^{1}+2*10^{0}+3*10^{-1}+5*10^{-2}$

  $1001.101_{2}=2^{3}+2^{0}+2^{-1}+2^{-3}=8+1+0.5+0.125=9.625$

- v N bitoch teda môžeme zobraziť čísla s pevnou čiarkou tak, že prvých K cifier zľava je „pred desatinnou čiarkou“ a zvyšok je „za ňou“ (K<=N); napríklad ak N=16 a K=8:



- násobenie čísiel v rozsahu <0;1> nevedie k pretečeniu, preto sa tento formát často používa pri reprezentácii koeficientov a vzoriek pri číslicovom spracovaní signálov
- takéto čísla sa sčitujú a odčitujú rovnako ako celé čísla, ale násobenie a delenie je iné
- majú malý rozsah zobraziteľných hodnôt, preto sú nevhodné na vedecké výpočty
- majú menej problémov so zaokrúhľovaním, preto sa niekedy používajú na finančné výpočty, kde malý rozsah až tak neprekáža

:: Reálne čísla - zobrazenie s pevnou čiarkou

Q - formát

- Q formát je formát zobrazovania čísiel s pevnou rádovou čiarkou zavedený spoločnosťou Texas Instruments. V tomto formáte je zadaný počet zlomkových bitov a voliteľne aj počet celočíselných bitov.
- napr. číslo vo formáte Q15 má 15 zlomkových bitov a číslo Q1.14 má jeden celočíslený bit a 14 zlomkových.
- Q formát je často používaný v signálových procesoroch, ktoré nedisponujú hardvérovou podporou pre prácu s plávajúcou rádovou čiarkou.
- tento spôsob reprezentácie zabezpečuje konštantné rozlíšenie v celom rozsahu zobrazovaných čísiel.

:: Reálne čísla - zobrazenie s pevnou čiarkou

Q - formát

- najvýznamnejší bit je vždy znamienkový, pretože Q-formát využíva dvojkový doplnok.
- Q formát teda vždy vyžaduje $m+n+1$ bitov pre úplné zobrazenie znamienkového čísla s pevnou rádovou čiarkou.
- pre daný Qm.n formát, platí:
  - rozsah je $[-2^m, 2^m - 2^{-n}]$
  - rozlíšenie je 2^{-n}
- napr. pre číslo vo formáte Q14.1:
  - potrebujeme 14+1+1 = 16 bitov
  - jeho rozsah je $[-2^{14}, 2^{14} - 2^{-1}]$ = [-16384.0, +16383.5] = [0x8000, 0x8001 … 0xFFFF, 0x0000, 0x0001 … 0x7FFE, 0x7FFF]
  - jeho rozlíšenie je 2^{-1} = 0.5

Q - formát

Príklad 1

Zobrazte 2.375 v tvare binárneho čísla s pevnou čiarkou v 8 bitoch, kde 4 bity sú pred čiarkou a 4 za ňou.

$2.375_{10} = 2+0.25+0.125 = 2+1/4+1/8 = 2^1+2^{-2}+2^{-3} = 10.011_2$, zapísané do 8 bitov: 00100110

Aký je to Q formát?

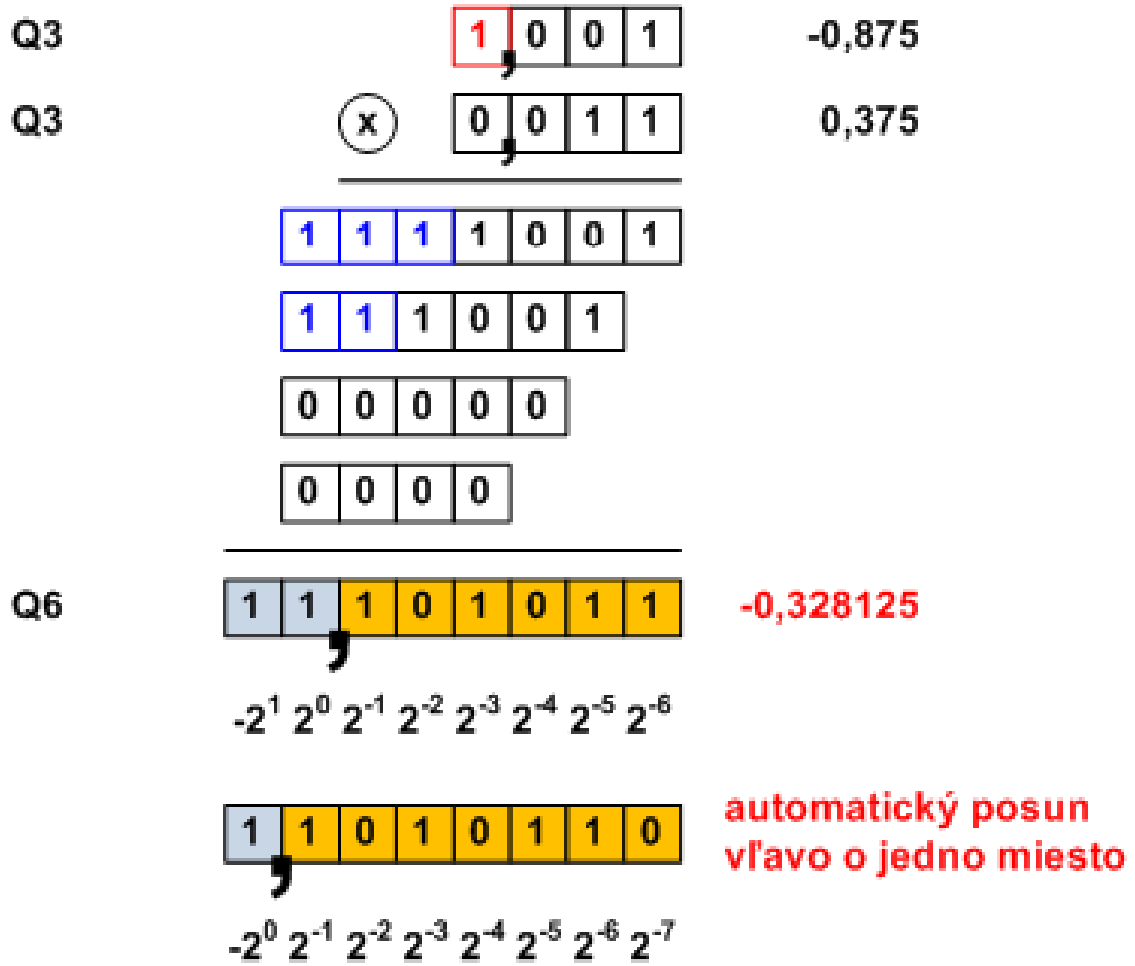
Príklad 2

Aké číslo s pevnou desatinnou čiarkou je zobrazené v 8 bitovom zápise 01011010 s 3 bitmi pred desatinnou čiarkou a 5 bitmi za ňou?

01011010 : $10.110_{12}=2^1+2^{-1}+2^{-2}+2^{-4}=2+1/2+1/4+1/16=$
$2+0.5+0.25+0.0625=\mathbf{2.8125_{10}}$

Q - formát

Násobenie dvoch čísiel vo formáte Q3 s automatickým posunom



Q - formát

Konverzia čísla typu Float na Qm.n – formát

1. číslo vynásobíme konštantou 2^n
2. výsledok zaokrúhlime na najbližšie celé číslo
3. výsledok prevedieme do binárnej sústavy

Konverzia čísla v Qm.n - formáte na číslo typu Float

1. binárne číslo prevedieme na celé číslo v desiatkovej sústave
2. výsledok násobíme konštantou 2^{-n}

:: Reálne čísla - zobrazenie s pohyblivou čiarkou

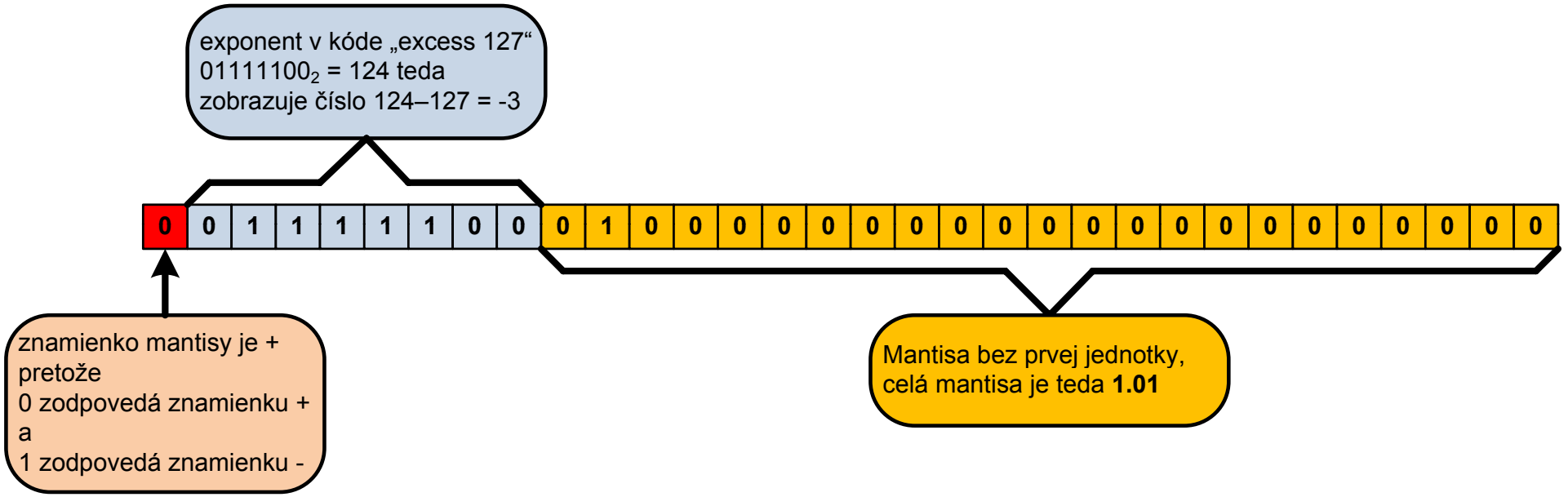
- hmotnosť slnka je asi $2*10^{33}$g, hmotnosť elektrónu je asi $9*10^{-28}$g. Rozdiel medzi nimi je 61 desiatkových rádov, čo je asi 202 dvojkových rádov.
- fyzikálny výpočet s oboma týmito číslami v pevnej rádovej čiarke si vyžaduje aspoň 202 bitové čísla. Pritom ale takú veľkú presnosť vôbec nevyužijeme lebo obe čísla poznáme s presnosťou na cca 5 cifier.
- preto sa pri vedeckých výpočtoch používa zápis čísla v pohyblivej (plávajúcej) rádovej čiarke, nazývaný tiež semilogaritmický tvar; v angloamerickej literatúre sa takáto aritmetika označuje *floating-point arithmetic*
- napr. číslo 2308 sa dá zapísať v tomto tvare ako $23.08*10^2$ alebo $0.002308*10^6$. Prednosť má vždy zápis $2.308*10^3$, ktorý má pred desatinnou bodkou práve jednu nenulovú cifru. Nazývame ho normalizovaný tvar čísla.
- v mikroprocesorovej technike sa používa základ 2, teda čísla sa zapisujú v tvare $a*2^b$, t.j. treba uložiť dve čísla a aj b, obe môžu byť kladné aj záporné.

:: Reálne čísla - zobrazenie s pohyblivou čiarkou

- IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Std 754-1985), tiež IEC 60559:1989, Binary floating-point arithmetic for microprocessor systems (pôvodné označenie IEC 559:1989)
  - najrozšírenejší tvar čísla v pohyblivej rádovej čiarke. Väčšina procesorov používa práve tento tvar.
- Single precision (32 bitov):
  - 1 bitové znamienko mantisy, 8 bitový exponent v kóde "excess 127" a 23 bitová absolútna hodnota mantisy s pevnou desatinnou čiarkou pred prvou cifrou
- Double precision (64 bitov):
  - 1 bitové znamienko mantisy, 11 bitový exponent v kóde "excess 1023" a 52 bitová absolútna hodnota mantisy
- normalizované číslo má pred desatinnou bodkou nenulovú cifru, v dvojkovej sústave jediná nenulová cifra je 1. Keď vieme, že pred desatinnou bodkou je vždy 1, tak si ju nemusíme pamätať a ušetríme jeden bit mantisy.

:: Reálne čísla - zobrazenie s pohyblivou čiarkou

Príklad zobrazenia čísla vo formáte IEEE 754-1985, single precision



$$= +1.01_2 * 2^{-3} = 1.25 * 2^{-3} = 1.25 * 0.125 = \mathbf{0.15625}$$

- najmenšia a najväčšia hodnota exponentu sa nepoužíva pre normalizované čísla. Napr. v single precision môže byť exponent len od -126 po +127, hodnoty exponentu -127 (kód 0) a +128 (kód FF) sú rezervované pre špeciálne čísla:
  - denormalizované číslo má exponent -127 (kód 0) a k mantise nie je pridávaná prvá jednotka - tak možno zobraziť mantisu až do 2^{-149} za cenu znižujúcej sa presnosti
  - nula má exponent -127 (kód 0) a mantisu 0, znamienko môže byť + aj –
  - plus nekonečno má znamienko +, exponent 128 (kód FF), mantisu 0
  - mínus nekonečno má znamienko -, exponent 128 (kód FF) a mantisu 0
  - Not a Number (NaN) má znamienko ľubovoľné, exponent 128 (kód FF) a mantisu nenulovú
- špeciálne hodnoty sa používajú aj pri výpočtoch, napríklad 1/0 je plus nekonečno, nekonečno deleno nekonečno je NaN a pod.

Pozn.: exponenty sú uvádzané pre single precision, pre double precision sú -1023 a 1024

Zhrnutie špeciálnych formátov IEEE 754-1985

Normalized	±	0 < Exp < Max	Any bit pattern
Denormalized	±	0	Any nonzero bit pattern
Zero	±	0	0
Infinity	±	1 1 1…1	0
Not a number	±	1 1 1…1	Any nonzero bit pattern

Sign bit