

ISU

Programování na strojové úrovni

stránka předmětu:

<https://www.fit.vutbr.cz/study/courses/ISU/>

Přednášky

1. Úvod, číselné soustavy, reprezentace čísel, binární aritmetika.
2. Základní funkce procesoru, strojový jazyk, jazyk symbolických instrukcí, assembler.
3. Architektura procesoru - registry, typy operandů, formát instrukcí, adresování paměti, přerušení.
4. Architektura procesoru - přenosy, aritmetické a logické instrukce.
5. Architektura procesoru - posuny a rotace, předávání řízení.
6. Architektura procesoru - další instrukce.
7. Půlsemestrální test.

8. Zásady programování ve strojovém jazyku, základní řídicí konstrukce.
9. Funkce, standardní předávání řízení a parametrů.
10. Programové moduly, knihovny, služby operačního systému.
11. Koprocessor FPU - architektura, reprezentace reálných čísel, instrukční sada.
12. Koprocessor FPU - instrukční sada, programování a ukázky použití.
13. Překladač jazyka symbolických instrukcí - pseudoinstrukce, direktivy, výrazy, operátory, operandy a makra.

1. Úvod, číselné soustavy, reprezentace čísel, binární aritmetika

Číselné soustavy

z ... základ číselné soustavy (přirozené číslo > 1)

a_i ... číslice soustavy (celé číslo: $0 \leq a_i < z$)

Soustava	základ	čísllice
dvojková	2	0 1
trojková	3	0 1 2
...		
osmičková	8	0 1 2 3 4 5 6 7
devítková	9	0 1 2 3 4 5 6 7 8
desítková	10	0 1 2 3 4 5 6 7 8 9
...		
šestnáctková	16	0 1 2 3 4 5 6 7 8 9 A B C D E F
...		

$$\begin{aligned}\check{c}íslo &= a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + K + a_0z^0 + a_{-1}z^{-1} + a_{-2}z^{-2} + K = \\ &= (a_{n-1}a_{n-2}a_0.a_{-1}a_{-2}K)_z\end{aligned}$$

$$\check{c}íslo = a_{n-1}a_{n-2}K a_0.a_{-1}a_{-2}K \quad (\text{pro implicitní základ } z)$$

Příklad (desítkové číslo 13.75):

$$1 \cdot 10^1 + 3 \cdot 10^0 + 7 \cdot 10^{-1} + 5 \cdot 10^{-2} = (13.75)_{10} = 13.75$$

$$1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} = (1101.11)_2 = 1101.11$$

$$1 \cdot 8^1 + 5 \cdot 8^0 + 6 \cdot 8^{-1} = (15.6)_8 = 15.6$$

$$D \cdot 16^0 + C \cdot 16^{-1} = (D.C)_{16} = D.C$$

Příklad (některá čísla v různých soustavách):

$$(0)_2 = (0)_8 = (0)_{10} = (0)_{16}$$

$$(1)_2 = (1)_8 = (1)_{10} = (1)_{16}$$

$$(10)_2 = (2)_{10}$$

$$(10)_8 = (8)_{10}$$

$$(10)_{16} = (16)_{10}$$

Je dobré znát nazpaměť:

Desítková	dvojková	šestnáctková-hexadecimální
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11

Převod čísla z desítkové soustavy do soustavy o základu z

Nechť značí:

n počet číslic celé části čísla (je dáno převodem),

m počet číslic necelé části čísla (musí být zadáno předem),

$a[i]$ i -tou číslici čísla

vše v soustavě o základu z .

Dále necht'

$\text{číslo}_{\text{celé}} \leftarrow \text{celá část převáděného čísla } \text{číslo}$

$\text{číslo}_{\text{necelé}} \leftarrow \text{číslo} - \text{číslo}_{\text{celé}}$

a necht' div a mod jsou operátory celočíselného dělení:

div ... výsledkem operace je celočíselný podíl

mod ... výsledkem operace je zbytek po celočíselném dělení

Převod celé části čísla

```

i ← 0 ;
dokud (číslocelé ≠ 0) opakuj
    { a[i] ← číslocelé mod z ;
      číslocelé ← číslocelé div z ;
      i ← i+1 ; }
n ← i;

```

Převod necelé části čísla

```

pro i ← −1 do −m opakuj
    { pom ← číslonecelé * z ;
      a[i] ← cela_cast(pom) ;
      číslonecelé ← pom - a[i] ;
      i ← i − 1 ; }

```

Pak

$$(\text{číslo})_{10} = (a[n-1]a[n-2].....a[0].a[-1]a[-2].....a[-m])_z$$

a) převod celé části čísla do soustavy

dvojkové:

osmičkové:

šestnáctkové:

číslo celé : 2

číslo celé : 8

číslo celé : 16

i	div = číslo celé	mod = a[i]	div = číslo celé	mod = a[i]	div = číslo celé	mod = a[i]
	586 : 2		586 : 8		586 : 16	
0	293	0	73	2	36	A
1	146	1	9	1	2	4
2	73	0	1	1	0	2
3	36	1	0	1		
4	18	0				
5	9	0				
6	4	1				
7	2	0				
8	1	0				
9	0	1				

$$(586)_{10} = (1001001010)_2 = (1112)_8 = (24A)_{16}$$

b) převod desetinné části čísla do soustavy

dvojkové:

osmičkové:

šestnáctkové:

číslo celé * 2

číslo celé * 8

číslo celé * 16

i	trunc =	cisnecel	trunc =	cisnecel	trunc =	cisnecel
	a[i]		a[i]		a[i]	
	0.248 * 2		0.248 * 8		0.248 * 16	
-1	0	496	1	984	3	968
-2	0	992	7	872	F = 15	488
-3	1	984	6	976	7	808
-4	1	968	7	808		
-5	1	936				
-6	1	872				
-7	1	744				
-8	1	488				
-9	0	976				
-10	1	952				
-11	1	904				
-12	1	808				

$$(0.248)_{10} = (0.001111110111)_2 = (0.1767)_8 = (0.3F7)_{16}$$

Tedy:

$$(586.248)_{10} = (1001001010.001111110111)_2 = (1112.1767)_8 = (24A.3F7)_{16}$$

$$z_1 = z_2^j \quad 8 = 2^3 \quad 16 = 2^4$$

Zpětný převod čísla ze soustavy o základu z do desítkové soustavy je jednoduchý:

$$(\text{číslo})_{10} = a[n-1] * z^{n-1} + a[n-2] * z^{n-2} + \dots + a[-m] * z^{-m}$$

$$2^9 + 2^6 + 2^3 + 2^1 + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} + 2^{-10} + 2^{-11} + 2^{-12} =$$

$$= 8^3 + 8^2 + 8^1 + 2 * 8^0 + 8^{-1} + 7 * 8^{-2} + 6 * 8^{-3} + 7 * 8^{-4} =$$

$$= 2 * 16^2 + 4 * 16^1 + 10 * 16^0 + 3 * 16^{-1} + 15 * 16^{-2} + 7 * 16^{-3}$$

$$= 586.24780$$

586.24780 # 586.248 Chyba oseknutí (Truncation Error) !

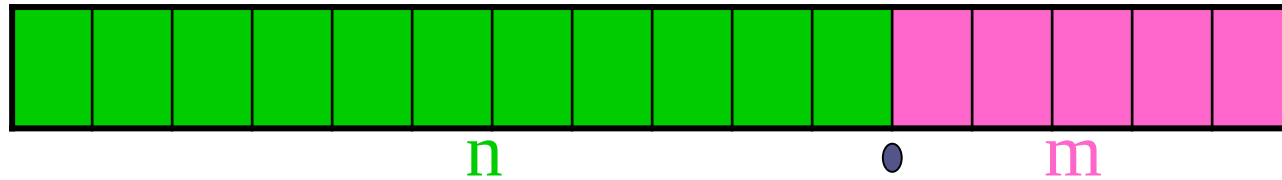
Zobrazení čísel dvojkové soustavy v počítači:

Základní pojmy:

bit	b inary digit – dvojková číslice
rozsah zobrazení	interval ohraničený zleva nejmenším a zprava největším zobrazitelným číslem
rozlišitelnost zobrazení	nejmenší (kladné) zobrazitelné číslo
přesnost zobrazení	počet platných dekadických číslic, které je možné zobrazit v daném paměťovém prostoru (hodnota nezávislá na velikosti zobrazovaného čísla !)

a) Zobrazení čísel v pevné řádové čárce

Uvažujme k -bitový paměťový prostor, které má n míst vlevo a m míst vpravo od řádové tečky ($k = n + m$):



Pozn.: Hodnoty m , resp. n mohou být i záporné, vždy však musí platit $k = m + n$!

aa) Zobrazení čísel bez znaménka (tj. kladných čísel)

Rozsah zobrazení	$<0, (2^n - 2^{-m})>$
------------------	-----------------------

Rozlišitelnost zobrazení	2^{-m}
--------------------------	----------

Přesnost zobrazení	$k * \log_{10}(2)$
--------------------	--------------------

V současné době se v pevné řádové čárce zobrazují prakticky výlučně jen celá čísla ($m = 0, k = n$). Pak rozsah zobrazení je dán intervalem $\langle 0, (2^n - 1) \rangle$ a rozlišitelnost zobrazení hodnotou 1.

<i>Počet bitů</i>	<i>Rozsah</i>	<i>Přesnost</i>	<i>Označení</i>
4	$\langle 0, 15 \rangle$	1.2	nibble
8	$\langle 0, 255 \rangle$	2.4	byte
16	$\langle 0, 65535 \rangle$	4.8	word
32	$\langle 0, 4294967295 \rangle$	9.6	doubleword
64	$\langle 0, \approx 1.84 \cdot 10^{19} \rangle$	19.2	quadword
128	$\langle 0, \approx 3.40 \cdot 10^{38} \rangle$	34.8	double quadword

Základní aritmetické operace ve dvojkové soustavě:

sečítání:

$$\begin{array}{rcccccc}
 & & & & & 1 \\
 & & & & & 1 \\
 & 0 & 0 & 1 & 1 & 1 \\
 + & 0 & 1 & 0 & 1 & 1 \\
 \hline
 = & 0 & 1 & 1 & 10 & 11 \\
 & & & \uparrow & \uparrow &
 \end{array}$$

přenosy do vyššího řádu

odečítání:

$$\begin{array}{rcccccc}
 & & & & 10 & 11 \\
 & & & & \uparrow & \uparrow \\
 & 0 & 10 & 1 & 1 & 1 \\
 - & 0 & 1 & 0 & 1 & 1 \\
 \hline
 = & 0 & 1 & 1 & 0 & 1 \\
 & & \downarrow & & \downarrow & \downarrow
 \end{array}$$

výpůjčky z vyššího řádu

násobení:

$$\begin{array}{r}
 0 \quad 0 \quad 1 \quad 1 \\
 * \quad 0 \quad 1 \quad 0 \quad 1 \\
 \hline
 = \quad 0 \quad 0 \quad 0 \quad 1
 \end{array}$$

dělení:

$$\begin{array}{r}
 0 \quad 0 \quad 1 \quad 1 \\
 : \quad 0 \quad 1 \quad 0 \quad 1 \\
 \hline
 = \quad x \quad 0 \quad x \quad 1 \\
 \uparrow \qquad \qquad \uparrow \text{ nepovolené operace}
 \end{array}$$

Aritmetické operace v omezeném paměťovém prostoru (pro jednoduchost uvažujme $k = n = 4$, $m = 0$, $\Rightarrow$ rozsah zobrazení $\langle 0, 15 \rangle$):

Sečítání:

1) Výsledek lze zobrazit v daném prostoru

$$\begin{array}{rcl}
 & 3 & 0011 \\
 + & 7 & 0111 \\
 \hline
 = & 10 & 1010
 \end{array}$$

2) Výsledek nelze zobrazit v daném prostoru

$$\begin{array}{rcl}
 & 9 & 1001 \\
 + & 8 & 1000 \\
 \hline
 \# & 1 & \mathbf{1}0001
 \end{array}$$

Přenos (*carry*) z nejvyššího bitu indikuje chybu - přetečení !!

Odečítání:

1) Výsledek lze zobrazit v daném prostoru

$$\begin{array}{r}
 9 \quad 1001 \\
 - \quad 7 \quad 0111 \\
 \hline
 = \quad 2 \quad 0010
 \end{array}$$

2) Výsledek nelze zobrazit v daném prostoru

$$\begin{array}{r}
 3 \quad \textcolor{red}{1}0011 \\
 - \quad 8 \quad 1000 \\
 \hline
 \# \quad 11 \quad 1011
 \end{array}$$

Výpůjčka (*Borrow*) do nejvyššího bitu
indikuje chybu - přetečení !!

ab) Zobrazení čísel se znaménkem

Necht' x značí zobrazované číslo (kladné nebo záporné) a necht' X značí jeho obraz (kladné číslo ukládané do paměti). K převodu se nejčastěji používají tři transformace:

a) Přímý kód:

$$X = x \quad \text{pro } x \text{ z intervalu } <0, 2^{n-1} - 2^{-m}>$$

$$X = 2^{n-1} - x \quad \text{pro } x \text{ z intervalu } <-(2^{n-1} - 2^{-m}), 0>$$

b) Doplnkový kód:

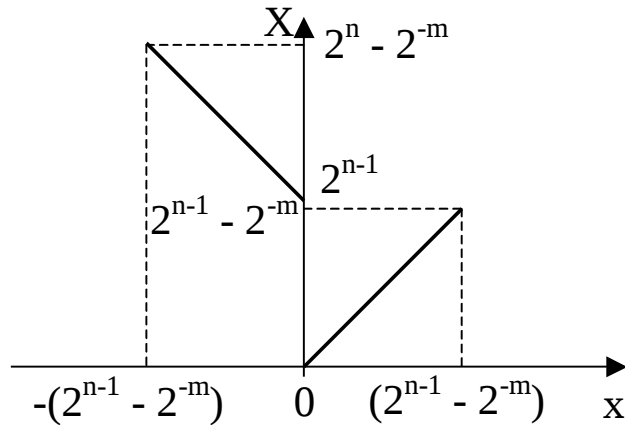
$$X = x \quad \text{pro } x \text{ z intervalu } <0, 2^{n-1} - 2^{-m}>$$

$$X = 2^n + x \quad \text{pro } x \text{ z intervalu } <-2^{n-1}, -2^{-m}>$$

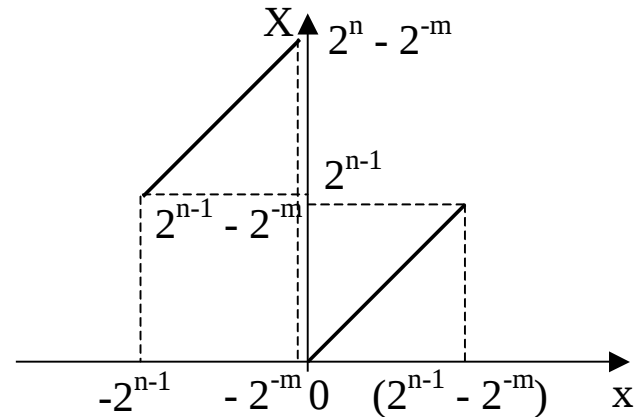
c) Kód transformované nuly (například):

$$X = 2^{n-1} + x \quad \text{pro } x \text{ z intervalu } <-2^{n-1}, 2^{n-1} - 2^{-m}>$$

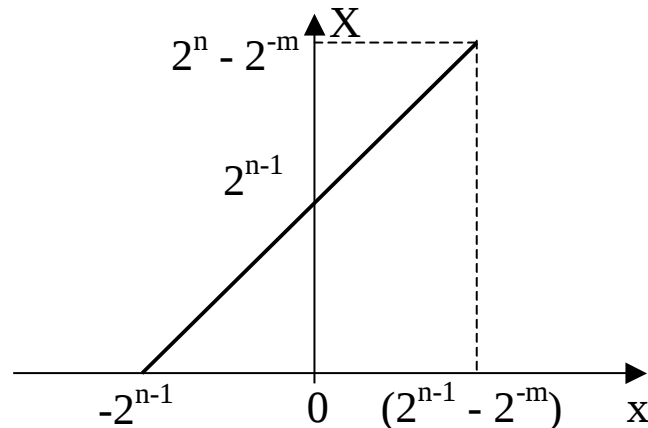
Grafická znázornění uvedených transformací



a) Přímý kód

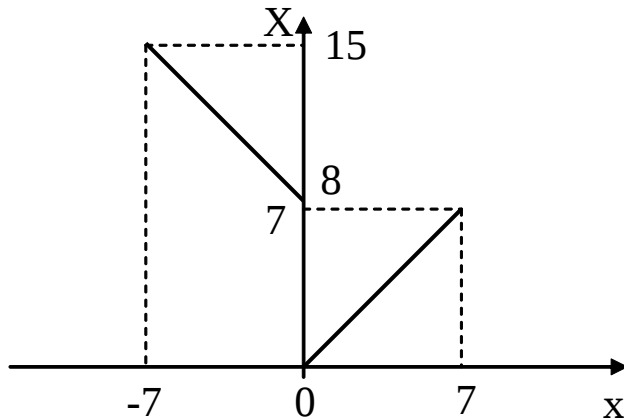


b) Doplňkový kód

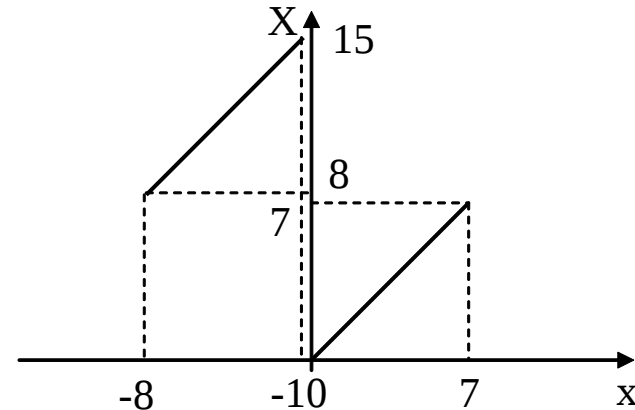


c) Kód transformované nuly

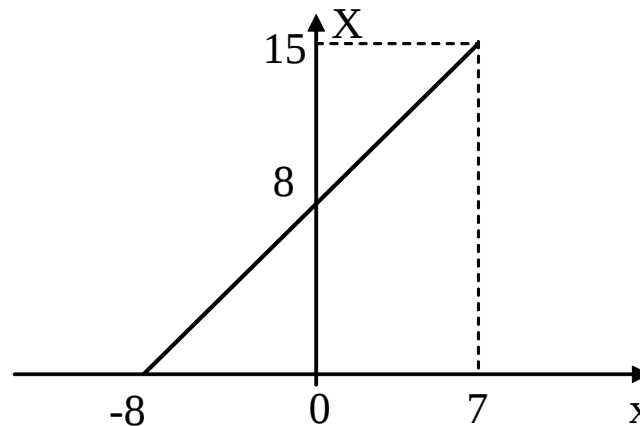
Příklad: $k = n = 4$, $m = 0 \rightarrow X \in \langle 0, 15 \rangle$



a) Přímý kód $x \in \langle -7, 7 \rangle$



b) Doplnkový kód $x \in \langle -8, 7 \rangle$



c) Kód transformované nuly $x \in \langle -8, 7 \rangle$

Přímý kód:

$$X = x \quad \text{pro } x \text{ z intervalu } <0, 2^{n-1} - 2^{-m}>$$

$$X = 2^{n-1} - x \quad \text{pro } x \text{ z intervalu } <-(2^{n-1} - 2^{-m}), 0>$$

Nechť $k = n = 8$ bitů, $m = 0$. Pak:

$$X = x \quad \text{pro } x \text{ z intervalu } <0, 127>$$

$$X = 128 - x \quad \text{pro } x \text{ z intervalu } <-127, 0>$$

Příklad převodu (např. čísel 28, -28 a 0):

$$x = 28 \rightarrow X = x = 28 \quad 00011100$$

$$x = -28 \rightarrow X = 128 - (-28) = 156 \quad 10011100$$

$$x = 0 \rightarrow X = x = 0 \quad 00000000$$

$$\rightarrow X = 128 - 0 = 128 \quad 10000000$$

Číslo kladné se od stejného čísla záporného v přímém kódu liší pouze hodnotou nejvyššího bitu (0 pro kladná čísla, 1 pro záporná čísla)! Nula v tomto kódu má dva rovnocenné obrazy!

Doplňkový kód:

$X = x$ pro x z intervalu $\langle 0, 2^{n-1} - 2^{-m} \rangle$

$X = 2^n + x$ pro x z intervalu $\langle -2^{n-1}, -2^{-m} \rangle$

Nechť $k = n = 8$ bitů, $m = 0$. Pak:

$X = x$ pro x z intervalu $\langle 0, 127 \rangle$

$X = 256 + x$ pro x z intervalu $\langle -128, -1 \rangle$

Příklad převodu (např. čísel 28, -28):

$x = 28 \rightarrow X = x = 28$ 00011100

$x = -28 \rightarrow X = 256 + (-28) = 228$ 11100100

Číslo kladné se od stejného čísla záporného v doplňkovém kódu liší hodnotou nejvyššího bitu (0 pro kladná čísla a nulu, 1 pro záporná čísla) i hodnotou všech ostatních bitů! Prakticky se číslo s opačným znaménkem získá inverzí hodnot všech bitů a aritmetickým přičtením jedničky k nejnižšímu bitu!

Příklad praktického převodu mezi kladným a záporným číslem:

28	00011100	-28	11100100
	11100011		00011011
	+1		+1
-28	11100100	28	00011100

Doplňkový kód má ve výpočetní technice zásadní důležitost pro jednoduchost svých aritmetických operací!

Doplňkový kód - aritmetické operace v omezeném paměťovém prostoru (pro jednoduchost opět uvažujeme $k = n = 4$, $m = 0$, $\Rightarrow$ rozsah zobrazení $\langle -8, 7 \rangle$ a sledujeme přenosy/výpůjčky z a do nejvyššího bitu):

Sečítání (přenosy z (C) a do (P) nejvyššího bitu):

1) Dvě kladná čísla, výsledek je zobrazitelný

$$\begin{array}{rcl}
 3 & 0011 & \\
 + 2 & 0010 & \\
 \hline
 = 5 & 0101 & C = 0, P = 0
 \end{array}$$

2) Dvě kladná čísla, výsledek není zobrazitelný

$$\begin{array}{rcl}
 5 & 0101 & \\
 + 6 & 0110 & \\
 \hline
 \# -5 & 1011 & C = 0, P = 1
 \end{array}$$

3) Dvě záporná čísla, výsledek je zobrazitelný

$$\begin{array}{rcl}
 & -4 & 1100 \\
 + & -2 & 1110 \\
 \hline
 = & -6 & \textcolor{red}{1}1010
 \end{array}
 \qquad C = 1, P = 1$$

4) Dvě záporná čísla, výsledek není zobrazitelný

$$\begin{array}{rcl}
 & -4 & 1100 \\
 + & -6 & 1010 \\
 \hline
 \# & 6 & \textcolor{red}{1}0110
 \end{array}
 \qquad C = 1, P = 0$$

5) Kladné a záporné číslo, výsledek je kladný (musí být zobrazitelný)

$$\begin{array}{r}
 5 \quad 0101 \\
 + \quad -2 \quad 1110 \\
 \hline
 = \quad 3 \quad \textcolor{red}{1}0011
 \end{array}
 \quad C = 1, P = 1$$

6) Kladné a záporné číslo, výsledek je záporný (musí být zobrazitelný)

$$\begin{array}{r}
 4 \quad 0100 \\
 + \quad -6 \quad 1010 \\
 \hline
 = \quad -2 \quad 1110
 \end{array}
 \quad C = 0, P = 0$$

Odečítání (výpůjčky do (C) a z (P) nejvyššího bitu):

1) Dvě kladná čísla, výsledek je kladný (musí být zobrazitelný)

$$\begin{array}{rcl}
 & 7 & 0111 \\
 - & 2 & 0010 \\
 \hline
 = & 5 & 0101
 \end{array}
 \qquad C = 0, P = 0$$

2) Dvě kladná čísla, výsledek je záporný (musí být zobrazitelný)

$$\begin{array}{rcl}
 & 5 & \textcolor{red}{1}0101 \\
 - & 6 & 0110 \\
 \hline
 = & -1 & 1111
 \end{array}
 \qquad C = 1, P = 1$$

3) Dvě záporná čísla, výsledek je záporný (musí být zobrazitelný)

$$\begin{array}{r}
 -4 \quad \textcolor{red}{1} \ 1100 \\
 - \quad -2 \quad 1110 \\
 \hline
 = \quad -2 \quad 1110
 \end{array}
 \qquad C = 1, P = 1$$

4) Dvě záporná čísla, výsledek je kladný (musí být zobrazitelný)

$$\begin{array}{r}
 -4 \quad 1100 \\
 - \quad -6 \quad 1010 \\
 \hline
 = \quad 2 \quad 0010
 \end{array}
 \qquad C = 0, P = 0$$

5) Kladné a záporné číslo, výsledek je kladný a zobrazitelný

$$\begin{array}{rcl}
 & 5 & \textcolor{red}{1}0101 \\
 - & -2 & \underline{1110} \\
 = & 7 & \underline{0111}
 \end{array}
 \quad C = 1, P = 1$$

6) Kladné a záporné číslo, výsledek je záporný a zobrazitelný

$$\begin{array}{rcl}
 & -6 & \textcolor{red}{1}1010 \\
 - & -4 & \underline{1100} \\
 = & -2 & \underline{1110}
 \end{array}
 \quad C = 1, P = 1$$

7) Kladné a záporné číslo, výsledek je záporný a není zobrazitelný

$$\begin{array}{rcl}
 & -6 & 1010 \\
 - & 4 & 0100 \\
 \hline
 \# & 6 & 0110
 \end{array}
 \quad C = 0, P = 1$$

8) Kladné a záporné číslo, výsledek je kladný a není zobrazitelný

$$\begin{array}{rcl}
 & 5 & \textcolor{red}{1}0101 \\
 - & -7 & 1001 \\
 \hline
 \# & 4 & 0100
 \end{array}
 \quad C = 1, P = 0$$

Ve všech příkladech byl výsledek správný, pokud byly oba přenosy C i P stejné ($C = P = 0$, nebo $C = P = 1$) a výsledek byl nesprávný (nezobrazitelný), pokud došlo pouze k jednomu z těchto dvou přenosů ($P \neq C$)!

Zjištěná vlastnost doplňkového kódu (lze ji snadno matematicky dokázat) je při operacích sečítání a odečítání využívána k nastavení příznaku přetečení O (*overflow*):

ISU	Aritmetické operace v omezeném paměťovém prostoru
	- Přenos C (<i>Carry</i>) reprezentuje přetečení z nejvyššího bitu (resp. výpůjčku do nejvyššího bitu), což značí chybu při práci s čísly bez znaménka. - Výsledek operace se znaménkem s čísly v doplňkovém kódu - byl správný $\Leftrightarrow$ oba přenosy C a P byly stejné: $C = P = 0$ nebo $C = P = 1$, - nebyl správný (nebyl zobrazitelný) $\Leftrightarrow$ došlo pouze k jednomu z přenosů P nebo C. - Tato vlastnost doplňkového kódu (lze ji matematicky dokázat) je při operacích sčítání a odčítání využívána k nastavení příznaku přetečení při práci se znaménkem O (<i>Overflow</i>):
	$O = P \oplus C = P \neq C = P \Leftrightarrow C = P \text{ xor } C$
	O (<i>Overflow</i>) indikuje chybu při práci s čísly se znaménkem. C (<i>Carry</i>) indikuje chybu při práci s čísly bez znaménka.

Násobení:

Násobení se obvykle provádí s kladnými čísly a pokud byl jeden a právě jeden z činitelů záporný, změní se znaménko výsledku.

Uvažujme $k = n = 8, m = 0 \Rightarrow x \in \langle 0, 255 \rangle$.

1) Výsledek je zobrazitelný:

$$\begin{array}{r} 11 * 14 \\ \hline 11 \\ 44 \\ \hline = 154 \end{array}$$

$$\begin{array}{r} 00001011 * 00001110 \\ \hline 00000000 \\ 00000000 \\ 00000000 \\ 00000000 \\ 00001011 \\ 00001011 \\ 00001011 \\ 00000000 \\ \hline 0000000010011010 = 154 \end{array}$$

2) Výsledek není zobrazitelný (je větší než 255):

$$\begin{array}{r}
 18 * 17 \\
 \hline
 18 \\
 126 \\
 \hline
 = 306
 \end{array}$$

$$\begin{array}{r}
 00010010 * 00010001 \\
 \hline
 00000000 \\
 00000000 \\
 00000000 \\
 00010010 \\
 00000000 \\
 00000000 \\
 00000000 \\
 00010010 \\
 \hline
 000000100110010
 \end{array}$$

= 306
50

Celočíselné dělení:

Pro celočíselné dělení jsou definovány dva výsledky: celočíselný podíl (operátor *div*) a zbytek po celočíselném dělení (operátor *mod*):

a	b	$a \text{ div } b$	$a \text{ mod } b$
114	5	22	4
-114	5	-22	-4
114	-5	-22	4
-114	-5	22	-4

Oba výsledky jsou vždy zobrazitelné (dělení nulou je zakázané!).

Dělení se opět obvykle provádí s kladnými operandy a upraví se pouze znaménka výsledků.

Příklad (114 / 5):

$$\begin{array}{r}
 \underline{01110010} / \underline{00000101} = 00010110 = \text{podíl} = 22 \\
 - 00000101 \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 - 00000101 \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 - 00000101 \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 - \underline{00000101} \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 \quad 00100010 \\
 - 00000101 \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 - \underline{00000101} \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 \quad 00001110 \\
 - \underline{00000101} \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 \quad 00000100 \\
 - 00000101 \quad \text{---} \quad \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow \\
 \quad 00000100 \quad \leftarrow = \text{zbytek} = 4
 \end{array}$$

Kód transformované nuly (například):

$$X = 2^{n-1} + x \quad \text{pro } x \text{ z intervalu } \langle -2^{n-1}, 2^{n-1} - 2^{-m} \rangle$$

Nechť $k = n = 8$ bitů $m = 0$. Pak:

$$X = 128 + x \quad \text{pro } x \text{ z intervalu } \langle -128, 127 \rangle$$

Příklad převodu (např. čísel 28, -28):

$$x = 28 \quad \rightarrow \quad X = 128 + 28 = 156 \quad 10011100$$

$$x = -28 \quad \rightarrow \quad X = 128 + (-28) = 100 \quad 01100100$$

Číslo kladné se od stejného čísla záporného v kódu transformované nuly liší opět hodnotou nejvyššího bitu (tentokrát 1 pro kladná čísla, 0 pro záporná čísla!) i hodnotou ostatních bitů! Ve **výše uvedené variantě** tohoto kódu se číslo prakticky získá z čísla v **doplňkovém kódu** změnou hodnoty nejvyššího bitu!

Příklad ($k = n = 8$ bitů):

$(28)_{\text{DOP}}$ 00011100

$(28)_{\text{TN}}$ 10011100

$(-28)_{\text{DOP}}$ 11100100

$(-28)_{\text{TN}}$ 01100100

Informaci o znaménku čísla ve všech uvedených kódech nese nejvyšší bit, který se proto nazývá **bit znaménkový**.

Všechny ostatní bity nesou informaci o hodnotě čísla a souhrnně se pak označují jako **bity významové**.

Procesory architektury IA-32 používají:

- Doplnkový kód pro zobrazení čísel celých.
- Přímý kód pro zobrazení mantis reálných čísel.
- Kód transformované nuly (poněkud odlišný od právě prezentovaného kódu) pro zobrazení exponentů reálných čísel.

b) Zobrazení čísel v pohyblivé řádové čárce

$\pm$	exponent	mantisa
-------	----------	---------

Pozn.: Zobrazení reálných čísel (standard IEEE) bude podrobně popsáno v jedenácté přednášce.

c) **Zobrazení BCD čísel** (Binary Coded Decimal)

Každá desítková číslice je zobrazena samostatně čtyřmi bity dvojkové soustavy.

$$(586.248)_{10} = (\overline{0101} \ \overline{1000} \ \overline{0110} \ . \ \overline{0010} \ \overline{0100} \ \overline{1000})_{\text{BCD}}$$

d) Zobrazení znaků

Různé kódy reprezentují jednotlivé znaky čísla bez znaménka, tzv. pořadovými/ordinálními čísly znaků.

Požadavky na tyto kódy jsou následující:

- Různé znaky musí být reprezentovány různými čísly.
- Písmeno větší v abecedě musí být reprezentováno větším číslem.
- Číslice musí být reprezentovány jako BCD číslice na nejnižších čtyřech bitech svých ordinálních čísel.

ASCII (American Standard Code for Information Interchange): 47 / 49

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

Pouze pro zajímavost:

Všechny dříve uvedené kódy lze používat pro zobrazení záporných čísel v libovolných soustavách.

Uvažujme například desítkovou soustavu a doplňkový kód:

$$X = x \quad \text{pro } x \text{ z intervalu } <0, 10^{n-1} - 10^{-m}>$$

$$X = 10^n + x \quad \text{pro } x \text{ z intervalu } <-10^{n-1}, -10^{-m}>$$

Příklad převodu čísla ($k = 6$, $n = 4$, $m = 2$):

$$x = -0567.23 \rightarrow X = 10000 + (-567.23) = 9432.77$$

Ke stejnému číslu dospějeme, zaměníme-li číslice původního kladného čísla jejich doplňky do devíti (9-0, 8-1, 7-2, 6-3, 5-4) a k nejnižšímu číslu přičteme jedničku:

$$\begin{array}{rcl} 0567.23 & \rightarrow & 9432.76 \\ & & + \quad 1 \\ & & 9432.77 \end{array}$$

Aritmetika je pak opět podobná:

Příklad: $786.59 - 567.23 = 219.36$

$-567.23 = (9432.77)_d$

$$\begin{array}{r} 0786.59 \\ - 0567.23 \\ \hline 0219.36 \end{array}$$

$$\begin{array}{r} 0786.59 \\ + 9432.77 \\ \hline 10219.36 \end{array}$$