

Číselné soustavy

Desítková - sčítání, odčítání, přenos – u jiných základů principy stejné. (zápis např. $123,123_{10}$)

Dvojková=binární – číslice 0/1 (zápis např. B0101, 0101B, 0101_2)

(Osmičková), Šestnácková=hexadecimální (čísllice: 0-9,A-F, zápis např. 0xa1, 0xA1, a1h, $a1_{16}$)

Převody mezi soustavami, číslo 57(dec) = 0x39 (hexdec,hex) = 111001 (bin)

Dec na bin, odčítání	Dec na bin, zbytek po dělení	Dec na hexdec zbytek po dělení	Bin na dec
(nejvyšší bit) $57 < 128 \rightarrow 0$ $57 < 64 \rightarrow 0$ $57 \geq 32 \rightarrow 1, (57-32 = 25)$ $25 \geq 16 \rightarrow 1 (25-16 = 9)$ $9 \geq 8 \rightarrow 1 (9-8 = 1)$ $1 < 4 \rightarrow 0$ $1 < 2 \rightarrow 0$ $1 \geq 1 \rightarrow 1 (1-1=0)$ 0 $\rightarrow 00111001$	(od nejnižšího bitu) $57/2 = 28 \text{ zb. } 1 \rightarrow 1$ $28/2 = 14 \text{ zb. } 0 \rightarrow 0$ $14/2 = 7 \text{ zb. } 0 \rightarrow 0$ $7/2 = 3 \text{ zb. } 1 \rightarrow 1$ $3/2 = 1 \text{ zb. } 1 \rightarrow 1$ $1/2 = 0 \text{ zb. } 1 \rightarrow 1$ $0/2 = 0 \text{ zb. } 0 \rightarrow 0$ $0/2 = 0 \text{ zb. } 0 \rightarrow 0$ $\rightarrow 00111001$	(od nejnižšího řádu) $57/16 = 3 \text{ zb. } 9 \rightarrow 9$ $3/16 = 0 \text{ zb. } 3 \rightarrow 3$ $\rightarrow 0x39$ hex na dec $0x39 = 3 \cdot 16^1 + 9 \cdot 16^0$ $= 3 \cdot 16 + 9 = 57$ hex na bin a zpět $0x39 \rightarrow 0011 1001$ $0011 1001 \rightarrow 0x39$	$0011 1001$ (bin) = $0 \cdot 2^7$ (tj. $0 \cdot 128$) + $0 \cdot 2^6$ (tj. $0 \cdot 64$) + $1 \cdot 2^5$ (tj. $1 \cdot 32$) + $1 \cdot 2^4$ (tj. $1 \cdot 16$) + $1 \cdot 2^3$ (tj. $1 \cdot 8$) + $0 \cdot 2^2$ (tj. $0 \cdot 4$) + $0 \cdot 2^1$ (tj. $0 \cdot 2$) + $1 \cdot 2^0$ (tj. $1 \cdot 1$) = $32 + 16 + 8 + 1$ $\rightarrow 57$ (dec)

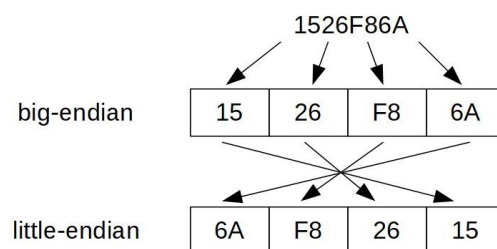
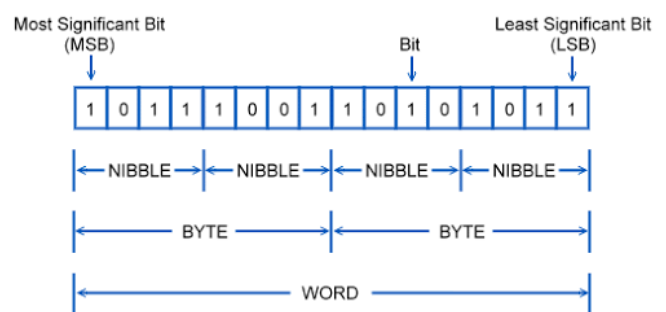
Pozn. Lidé se dělí do 10 skupin: ti co rozumí binárnímu kódu, a ti co ne. (aneb $1+1=10$)

Uložení informace v počítači, kódování čísel

Dvojková soustava, bit, byte (8 bitů) (word,long)

pamět: 0xb9 0xab ...bez dohody nevíme - může být číslo, ASCII znak, kód programu, obrázek, atd)

Little / Big Endian (od nejméně/nejvíce významného byte) -



Zobrazení celých čísel

Kladná čísla – 1Byte – bit7..bit0, 0-255 (2^8), 2Byte 0-65535 (2^{16}), ...

(2^N čísel, rozsah 0 až 2^N-1)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	0	0	1	0	0	1	1	
=	=	=	=	=	=	=	=	
128	64	32	16	8	4	2	1	
			16		+		2+1 = 19	

Znaménková čísla (1 Byte)

Přímý kód – nejvyšší bit znaménko, 0=+ (plus), 1= - (minus) , tj. např. 0x84 = - 4.

je třeba testovat znaménko a podle toho sčítat odčítat, dvě nuly

(2^N-1 čísel, rozsah $2^{(n-1)}-1$ až $2^{(n-1)}-1$) (tedy např pro 8-bitů: 255 čísel, -127 až 127)

Inverzní kód – kladná čísla přímo, záporná inverzně, tj. -4 = -00000100 = 11111011.

stále dvě reprezentace nuly, jen myšlenkový přechod k:

Doplňkový kód - kladná čísla přímo, záporná dvokový doplněk (negace a přičtení 1)

tj. -4 = - 00000100 = 11111011 +1 = 11111100

zobrazení -128 az 127 ... 10000000 ... 11111111 00000000 ... 01111111

není dvojitá nula, vhodné při binární operace, přímá návaznost čísel -1 0 1

=snadná realizace aritmetiky (odčítání, sčítání)

Převod: číslo v doplňkovém kódu (např. 0x04=00000100, 0xfc=11111100) na dec

- znaménko dle nejvyššího bitu (0= kladné, 1=záporné)

- je li kladné či 0, pak převedeme nomálně, např.

00000100(bin) → (převod) → 4(dec)

- je li záporné pak: inverze, přičtení 1, např.

11111100(bin) → (neg) → 00000011(bin) → (+1) → 00000100 → (převod) → 4 → -4(dec)

Převod: číslo do doplňkového kódu (např. 4, -4 na 8 bitů)

- zkontrolujeme rozsah (pro 8bitů – je číslo v rozsahu -128 až 127 ?)

- je li kladné či 0, pak převedeme nomálně, např.

4(dec) → (převod) → 00000100(bin)

- je li záporné pak: inverze, přičtení 1, (alternativně odečtení 1, inverze) např.

-4 → 4 → 00000100(bin) → (neg) → 11111011 → (+1) → 11111100(bin)

alternativně: -4 → 4 → (-1) → 3 → 00000011(bin) → (neg) → 11111100(bin)

Kód s posunutou nulou

dohnutá „pozice“ 0, např. 0=127. potom -4 = 123, 4=131

výhoda návaznosti, lepší porovnávání (vyšší číslo je vyšší), snadné sčítání

násobení je třeba ošetřit. Používá se pro reprezentaci exponentu pro reálná čísla

Operace s čísly

Sčítání (doplňkový kód)

$20 + (-13) = 00010100 + 11110011 = 1\ 00000111 = 7$ (po odříznutí přeteklého devátého bitu)

0 0 0 1 0 1 0 0 +

1 1 1 1 0 0 1 1 (červeně jsou místa kde je přenos z nižších řádů. tj. +1)

0 0 0 0 0 1 1 1 → 7

Odčítání (doplňkový kód)

$20 - (-13) = 00010100 - 11110011 = 00100001 = 33$

0 0 0 1 0 1 0 0

-1 1 1 1 0 0 1 1 (červeně jsou místa kde je přenos z nižších řádů – tj, -1)

0 0 1 0 0 0 0 1

$-76 - 27 = 10110100 - 00011011 = 10011001 = -103$

1 0 1 1 0 1 0 0

-0 0 0 1 1 0 1 1

1 0 0 1 1 0 0 1

Násobení

$7 * 5 = 0111 * 0101 = 0111 * 0101 = 100011 = 35$

0 1 1 1

*0 1 0 1

0 1 1 1

0 0 0 0

0 1 1 1

0 0 0 0

0 1 0 0 0 1 1

Příznakové bity, přetečení, přenos

Příznakové bity informují o výsledku operace, např. bdyž výsledek operace se nevejde do rozsahu.

Příznakové bity:

Z = **Zero flag** - pokud je výsledek operace = 0, nastaví se na 1

C = **Carry flag** - Pokud došlo k **přenosu** do vyššího (MSB+1 = neexistujícího) bitu při operaci s unsigned číslem, nastaví se na 1 – tj. Hodnota MSB+1

V (O) = **Overflow** - Pokud došlo k **přetečení** u signed operace, nastaví se na 1. (tj. změna MSB)

N (S) = **Negative/Sign** - Pokud je výsledek záporný, nastaví se na 1 (kopie nejv.bitu ,MSB)

H = **Half Carry** - Pokud došlo k přenosu z nižších 4 bitů do vyšších, nastaví se na 1 (pro BCD)

Příklad:

10000001 + 10000000 = (1)0000001 → Z=0,C=1,V=1,S=0,H=0 (přízn.bity)

129 + 128 = 1 (!) (257 s carry bitem)

-127 + -128 = 1 (!)

Příklad2:

11111111 + 00000001 = (1)00000000 → Z=1,C=1,V=0,S=0,H=1 (přízn.bity)

255 + 1 = 0 (!) (256 s carry bitem)

-1 + 1 = 0

Příklad3:

01000000 + 01000001 = (0)10000001 → Z=0,C=0,V=1,S=1,H=0 (přízn.bity)

64 + 65 = 129

64 + 65 = -127 (!)

Příklad3: Určete kdy dojde k přetečení:

A	B	operace			
		sčítání		odčítání	
		signed	unsigned	signed	unsigned
50	50	ne	ne	ne	ne
100	50	ano	ne	ne	ne
50	100	ano	ne	ne	ano
150	150	-	ano	-	ne
-50	100	ne	-	ano	-
-50	-100	ano	-	ne	-

Zobrazení Reálného čísla

Fixní desetinná tečka

$$0001.1100 = 1.75 = 28/16$$

Exponenciální formát (IEEE 754)

FORMÁT: S Exp Mantisa

S znaménko čísla (0=kladné, 1=záporné), Mantisa v Přímém kódu, Exponent v kód s posunutou nulou

Normalizovaný tvar Mantisy:

1.xxx

První jedničku (která je tam vždy kromě výsledku 0) vynecháváme.

Příklad:

-0.312510:

$$-0.3125 \text{ (dec)} = -0.0101 \text{ (bin)} = -1.01 \text{ (bin)} \times 2^{-2}$$

$$S=1 \text{ (-1)}$$

$$E=-2\dots = (\text{kod s pos.nulou}) -2+127 = 125=01111101$$

Mantisa=.010... (vynechali jsme 1.)

Výsledek:

1 01111101 01000 0

Zvláštní čísla v exp.Formátu

0 - Exponent i mantisa jsou nulové

Nekonečno (inf) - Exponent obsahuje samé jedničky, a mantisa samé nuly. Znaménkový bit označuje, zda-li jde o kladné, nebo záporné nekonečno.

Nenormalizované číslo - Při nutnosti zobrazit menší číslo v absolutní hodnotě než je $1.0 \times 2^{-(2n-1)}$

O číslu musí být známo, že je nenormalizované, poznáme to tak, že exponent má samé nuly a mantisa je nenulová.

„**Not a number**“ **NaN** - Exponent obsahuje samé jedničky a mantisa je nenulová.

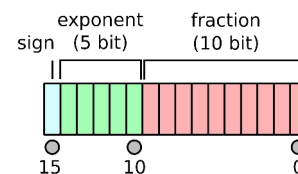
Příklad: float16 (IEEE 754, half-precision floating-point format)

Sign bit: 1 bit

Exponent: 5 bits (kod s pos.nulou. Offset = 15 = 01111)

$E_{min} = -14 = 00001$, $E_{max} = 15 = 11110$, 00000/11111 reserved

Mantisa: 11 bits (10 is explicitly stored)



Exponent	význam
0 (00000)	0 or subnormal numbers
1-30 (00001-11110)	Normalised values
31 (11111)	Nan (infinity)

Binary	Hex	Value	Notes
0 00000 0000000000	0000	0	
0 00000 0000000001	0001	$2^{-14} \times (0 + 1/1024) \approx 0.000000059604645$	smallest positive subnormal number
0 00000 1111111111	03ff	$2^{-14} \times (0 + 1023/1024) \approx 0.000060975552$	largest subnormal number
0 00001 0000000000	0400	$2^{-14} \times (1 + 0/1024) \approx 0.00006103515625$	smallest positive normal number
0 01101 0101010101	3555	$2^{-2} \times (1 + 341/1024) \approx 0.33325195$	nearest value to 1/3
0 01110 1111111111	3bff	$2^{-1} \times (1 + 1023/1024) \approx 0.99951172$	largest number < 1
0 01111 0000000000	3c00	$2^0 \times (1 + 0/1024) = 1$	one
0 01111 0000000001	3c01	$2^0 \times (1 + 1/1024) \approx 1.00097656$	smallest number > 1
0 11110 1111111111	7bff	$2^{15} \times (1 + 1023/1024) = 65504$	largest normal number
0 11111 0000000000	7c00	∞	infinity
1 00000 0000000000	8000	-0	
1 10000 0000000000	c000	-2	
1 11111 0000000000	fc00	$-\infty$	negative infinity

Zobrazení znaků a textu

ASCII

Dohodnutá kódovací tabulka znaků

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

např. 0x30 = '0', 0x31 = '1', 0x41 = 'A', 0x61 = 'a'

text – řetězec znaků, buď známé délky nebo ukončen dohodnutým znakem (0x00)
odřádkování: <CR><LF> (win) vs <LF> (linux) vs <CR> (mac) (+existují i jiné)

Kódy 128-255 využity pro národní abecedy (codepage, ISO 8859-x)
nevýhoda: nutno znát kódování, více možností kódování, obtížné míchaní různojazyčných textů

Unicode

UTF-8 - navrženo jako komatibilní s ASCII, 1 znak je 1-4(6)Byte

UTF-16 – základem je 16bitů na znak

tab. UTF-8

Bits of code point	First code point	Last code point	Bytes in sequence	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
7	U+0000	U+007F	1	0xxxxxxx					
11	U+0080	U+07FF	2	110xxxxx	10xxxxxx				
16	U+0800	U+FFFF	3	1110xxxx	10xxxxxx	10xxxxxx			
21	U+10000	U+1FFFFF	4	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx		
The following sequences are not part of the UTF-8 standard, only part of the original proposal									
26	U+200000	U+3FFFFFFF	5	111110xx	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx	
31	U+4000000	U+7FFFFFFF	6	1111110x	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx