Desítková soustava (decimální)

- užívaná většinou lidí na naší planetě
- číslo je složeno z jednoho či více číslic
- číslice mohou nabývat hodnot:
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- pokud by mělo dojít k „přetečení“ čísla přes hodnotu 9, přičteme +1 k číslici o jedna vyšší (carry)
  - 9 + 1 = 10
  - 9 + 2 = 12
  - 999 + 1 = 1000

Dvojková soustava (binární)

- použitá v naprosté většině elektroniky
- číslo je složeno z jednoho či více číslic
- číslice mohou nabývat hodnot:
  - 0, 1
- pokud by mělo dojít k „přetečení“ čísla přes hodnotu 1, přičteme +1 k číslici o jedna vyšší (carry)
  - 1 + 1 = 10
  - 10 + 11 = 101
  - 1111 + 1 = 10000

Hlavní rozdíl mezi desítkovou a dvojkovou soustavou je počet hodnot jednotlivých číslic. Pro počítače je výhodné využít pouze 2 hodnot 0 či 1 (nesvítí / svítí; <1,5V / >1,5V). Rozlišování mezi více hodnotami je pro elektroniku velmi složité (využívá se například u SSD disků, kdy jedna buňka může nabývat několika hodnot, např.: <0,2V; 0,2 – 0,4V; 0,4 – 0,8V; >0,8V => 0, 1, 2, 3, ale takovéto hodnoty jsou vždy rychle převedeny na binární).

U desítkových číslic máme dány řády:

1. → jednotky
2. → desítky
3. → stovky
4. → tisíce
5. → deseti tisíce

číslo 237 se tedy dá rozložit na 200 + 30 + 7.

U dvojkových číslic je vyšší řád vždy dvojnásobkem řádu předchozího:

1. → $2^0 = 1$
2. → $2^1 = 2$
3. → $2^2 = 4$
4. → $2^3 = 8$
5. → $2^4 = 16$

číslo 11101101 se dá rozložit na 1*128 + 1*64 + 1*32 + 0*16 + 1*8 + 1*4 + 0*2 + 1*1 = 237

Kromě dvojkové a desítkové soustavy se můžeme setkat ještě s šestnáctkovou či osmičkovou soustavou. Těmi si ale dnes nebudeme plést hlavu, jen zmíním, že šestnáctková soustava je přímo konvertovatelná do binární a proto je používána při programování (číslice nabývají hodnot 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f)

Porovnání čísel

dec	bin	dec	bin	dec	bin	dec	bin	dec	bin
0	0	10	1010	20	1 0100	30	1 1110	40	10 1000
1	1	11	1011	21	1 0101	31	1 1111	41	10 1001
2	10	12	1100	22	1 0110	32	10 0000	42	10 1010
3	11	13	1101	23	1 0111	33	10 0001	43	10 1011
4	100	14	1110	24	1 1000	34	10 0010	44	10 1100
5	101	15	1111	25	1 1001	35	10 0011	45	10 1101
6	110	16	1 0000	26	1 1010	36	10 0100	46	10 1110
7	111	17	1 0001	27	1 1011	37	10 0101	47	10 1111
8	1000	18	1 0010	28	1 1100	38	10 0110	48	11 0000
9	1001	19	1 0011	29	1 1101	39	10 0111	49	11 0001

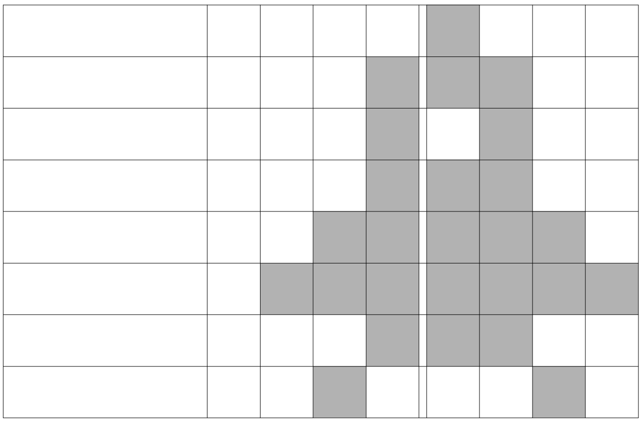
Obrázky

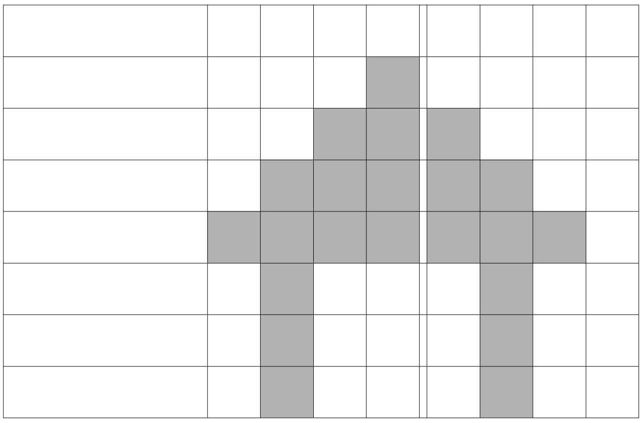
1;8								
1;8								
3;12								
7;14								
15;15								
9;9								
1;8								
3;12								

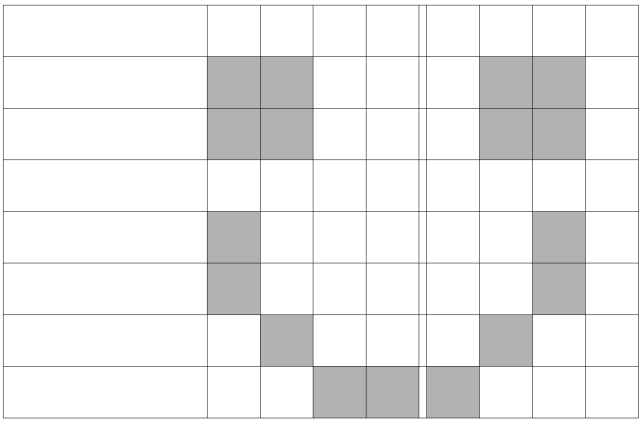
7;14								
8;1								
10;5								
8;1								
10;5								
9;9								
8;1								
7;14								

0;0								
6;6								
9;9								
9;9								
4;2								
2;4								
1;8								
0;0								

1;4								
0;8								
1;4								
0;8								
0;8								
3;14								
1;12								
1;12								







Registry počítače

Registry počítače slouží k uložení čísel a hlavním parametrem je jejich bitová šířka, tj. kolik číslic hodnot 0 či 1 mohou uložit. Typické šířky jsou:

- 4bit – používalo se u prvních čipů pro kalkulačky, umí pracovat s čísly 0 – 15
- 8bit – první herní konzole a první PCčka (XT); pracují s čísly 0 - 255
- 16bit – PC/AT, zpravidla DOS či win 3.11; pracují s čísly 0 – 65535
- 32bit – éra windows 95 – win XP; pracují s čísli 0 – 4294967295
- 64bit – moderní PC; pracují s čísli 0 – 18446744073709551615
- 128/256bit – některé DSP čipy

Reprezentace hodnot

Uložená čísla mohou reprezentovat různé věci a je na nás, abychom se k nim správně chovali. Zkusme například vzít počítadlo, standardně má 10 kuliček na řádku. Jedna plná řádka je tudíž 10, 2 řádky jsou 20, 3 řádky 30.

My si ale můžeme představit, že jednotlivé řádky jsou „řády“ v naší desítkové soustavě. Jednu kuličku si budeme muset vyndat (či nepoužívat), neboť v desítkové soustavě mohou být pouze hodnoty 0 – 9. Jedna plná řádka pak bude 9, 2 plné řádky 99 a 3 plné řádky 999.

Sčítání

Sčítání na klasickém počítadle funguje tak, že postupně přidáváme požadovaný počet kuliček.

Při sčítání na desítkovém počítadle musíme postupovat po cifrách, čili jednotky sečteme s jednotkami, desítky s desítkami a podobně. Při „přetečení“ (dojdou nám kuličky u jedné cifry) vynulujeme aktuální cifru a zvýšíme cifru vyššího řádu (podobně jako u sčítání pod sebou).

Záporná čísla

Pokud potřebujeme počítat se zápornými čísly (menšími než 0), musíme se domluvit, jak je reprezentovat. Jedna z možností je využít nejvyššího řádu k identifikaci záporného čísla:

2 == 0010
-2 == 1010

Nevýhodou je, že máme dvě nuly (0000 a 1111) a počítání nefunguje úplně jednoduše (0010 + 1010 => 1100). Proto se spíš používá druhý doplněk, kdy znegujeme (prohodíme 0 za 1 a naopak) hodnotu a přičteme 1:

2 == 0010
-2 == -0010 → 1101 + 1 → 1110

Nyní máme pouze jednu nulu (0000) a aritmetika funguje (0010 + 1101 => 0000).

Všimněte si, že použitím nejvyššího řádu jako znaménka omezíme rozsah našeho 4 bitového čísla z 0 - 15 na -8 – 7.

Čísla s plovoucí desetinou čárkou

Pro počítání s opravdu velikými čísly buď potřebujeme širší registry, nebo můžeme využít méně přesná čísla, která následně vynásobíme určitou hodnotou. Takovým číslům se říká čísla s plovoucí desetinou čárkou. Matematika s nimi je o mnoho složitější, pomalejší, ale umožní nám kombinovat různě velká čísla. Podobně jako u záporných čísel i zde je na nás, abychom si rozmysleli, jak rozdělíme naše číslo. Pokud například máme 8bitové číslo, můžeme si říci, že první 4 číslice využijeme pro „exponent“ a druhé 4 číslice pro vlastní hodnotu.:

0000 0000 → 0 * 10^{0} == 0
0000 0010 → 2 * 10^{0} == 2
0010 0010 → 2 * 10^{2} == 200
0010 0000 → 2 * 10^{0} == 0

Oproti obyčejným 8 bitovým číslům omezíme rozsah z 0 – 255 na 0 – 15, ale můžeme si číslo „posunout“ o 0 – 15 řádů, čili můžeme mít čísla 0 – 15000000000000000, ale vždy jen složené z jednoho čísla 0 – 15 doplněným o určitý počet nul. Pokud tedy budeme vždy přičítat jedničku, vyjde nám řada: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 15, 13, 14, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 200, 300, 400, …

Opět bychom mohli využít dvojkový doplněk a uvažovat záporná čísla (-8 – 7), případně pro reprezentaci malých hodnot i záporný exponent a vyjádřit čísla -0,00000008 – 70000000. Zde by došlo k ještě většímu zaokrouhlení hodnot, neboť přičítáním jedničky dostaneme řadu 1, 2, 3, 4, 5, 6, 7, 10, 20, 30, 40, 50, 60, 70, 100, 110, 120, 130, 140, …

Čísla s plovoucí desetinou čárkou se využívají zpravidla v 16 a více bitových počítačích, kde již udávaná přesnost stačí a nedochází k tak markantním skokům.

Písmena

Registry mohou obsahovat nejen čísla, ale i písmena. Nejrozšířenějším formátem je kódování ASCII, které se rozšířilo na 8 bitových počítačích. Písmena A-Z jsou definovány jako čísla 65 – 90 bez diakritiky. Celou tabulku si můžeme prohlédnout níže:

0 NUL	NULL character	32 SP	64 @	96 `
1 SOH	Start of Header	33 !	65 A	97 a
2 STX	Start of Text	34 "	66 B	98 b
3 ETX	End of Text	35 #	67 C	99 c
4 EOT	End of Transmission	36 $	68 D	100 d
5 ENQ	Enquiry	37 %	69 E	101 e
6 ACK	Acknowledge	38 &	70 F	102 f
7 BEL	Bell	39 '	71 G	103 g
8 BS	Backspace	40 (	72 H	104 h
9 HT	Horizontal Tab	41)	73 I	105 i
10 LF	Line feed	42 *	74 J	106 j
11 VT	Vertical Tab	43 +	75 K	107 k
12 FF	Form Feed	44 ,	76 L	108 l
13 CR	Carriage return	45 -	77 M	109 m
14 SO	Shift Out	46 .	78 N	110 n
15 SI	Shift In	47 /	79 O	111 o
16 DLE	Data Link Escape	48 0	80 P	112 p
17 DC1	Device Control (XOn)	49 1	81 Q	113 q
18 DC2	Device Control	50 2	82 R	114 r
19 DC3	Device Control (XOff)	51 3	83 S	115 s
20 DC4	Device Control	52 4	84 T	116 t
21 NAK	Negative Acknowledge	53 5	85 U	117 u
22 SYN	Synchronous Idle	54 6	86 V	118 v
23 ETB	End of Transmission Block	55 7	87 W	119 w
24 CAN	Cancel	56 8	88 X	120 x
25 EM	End of Medium	57 9	89 Y	121 y
26 SUB	Substitute	58 :	90 Z	122 z
27 ESC	Escape	59 ;	91 [	123 {
28 FS	File Separator	60 <	92 \	124 |
29 GS	Group Separator	61 =	93]	125 }
30 RS	Record Separator	62 >	94 ^	126 ~
31 US	Unit Separator	63 ?	95 _	127 DEL

LF = Line feed = Odřádkování = Enter (někdy nutno LF+CR)
SP = Space = Mezera

Pokud bychom tedy chtěli přečíst hodnotu 4 registrů o kterých víme, že obsahují znaky v ASCII kódu, nahradíme jejich hodnotu odpovídajícím znakem:

1000001, 1001000, 1001111, 1001010, == 65, 72, 79, 74 == AHOJ