Struktura počítačů

Jan Outrata

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

přednášky

Reprezentace dat

Kódování dat

- typy dat v počítači: celá čísla, (necelá) čísla s řádovou čárkou, znaky a text

→ **binární reprezentace** = **kódování** do binárních hodnot (= posloupnost $\mathbf{0}$ a $\mathbf{I}$)

- **kód** (kódování) = zobrazení čísel a znaků na binární hodnoty – pomocí kódových schémat (algoritmů) a tabulek
- **kód** (kódové slovo) = binární hodnota, obecně posloupnost kódových znaků ($\mathbf{0}$ a $\mathbf{I}$)
- dekódování = (inverzní) zobrazení kódového slova na původní číslo nebo znak
- různé kódy pro uložení dat, zpracování dat (např. přenos), jejich zabezpečení proti chybám, neoprávněnému čtení atd.
- kódující a dekódující log. obvody s pamětí = **kodéry, dekodéry**

Endianita (Endianness)

= pořadí bytů (byte order) v binárních hodnotách delších než 1 byte, např. 2B (16b), 4B (32b), 8B (64b) aj. **slova** v operační paměti (adresované od nižších adres k vyšším)

- **little-endian** = od nejméně významného bytu (LSB) hodnoty k nejvýznamnějšímu
  - např. pro $(30201)_{16}$ ve 4 bytech pořadí $01\ 02\ 03\ 00$
  - rychlejší aritmetika, možno nečíst další nulové byty
  - platformy např. Intel x86, AMD x86-64, DEC Alpha, ARM, Ethernet, USB (pořadí bitů), formát např. GIF
- **big-endian/network order** = od nejvýznamnějšího bytu (MSB) hodnoty k nejméně významému
  - např. pro $(30201)_{16}$ ve 4 bytech pořadí $00\ 03\ 02\ 01$
  - pro člověka čitelnější, znaménko v 1. bytu, řády čísla a znaky řetězce ve stejném pořadí
  - platformy např. Motorola 6800 a 68k, IBM POWER, SPARC, internetové protokoly, formát např. JPEG
- **middle/mixed-endian** = kombinace little a big-endian
  - např. pro $(30201)_{16}$ ve 4 bytech pořadí $03\ 00\ 01\ 02$ nebo $02\ 01\ 00\ 03$
  - platformy např. ARM (pro čísla s plovoucí řádovou čárkou ve formátu double, viz dále)
- řeší překladač/interpret prog. jazyka pro danou platformu, mezi platformami nutné konverze (např. v síťovém API)

Celá čísla

$=$ **interval** $\langle$**min. nekladné, max. nezáporné**$\rangle$ – hranice dané počtem n bitů reprezentace a použitým kódem, výsledek aritmetické operace mimo $=$ **přetečení (overflow) / podtečení (underflow)**

Nezáporná čísla:

Vážený poziční kód

$=$ zápis čísla ve dvojkové poziční číselné soustavě

- např. $123 = (123)_{10} = [\mathbf{0} \ldots \mathbf{IIII0II}]_2$
- $\langle 0, 2^n - 1 \rangle$

Celá čísla

= **interval** $\langle$**min. nekladné, max. nezáporné**$\rangle$ – hranice dané počtem n bitů reprezentace a použitým kódem, výsledek aritmetické operace mimo = **přetečení (overflow) / podtečení (underflow)**

Nezáporná čísla:

Vážený poziční kód

= zápis čísla ve dvojkové poziční číselné soustavě

- např. $123 = (123)_{10} = [\mathbf{0} \ldots \mathbf{IIII0II}]_2$
- $\langle 0, 2^n - 1 \rangle$

Dvojkově desítkový kód (BCD, Binary Coded Decimal)

= zápis každé desítkové číslice čísla (zapsaného v desítkové soustavě) zvlášť ve dvojkové soustavě s pevným počtem 4 dvojkových číslic

- např. $123 = [\mathbf{0} \ldots \ \mathbf{000I\ 00I0\ 00II}]_{BCD}$
- $\langle 0, 10^{n/4} - 1 \rangle$ pro $n = 4k, k \in \mathbb{N}$
- neefektivní, složitější log. obvody, použití pro přesné zobrazení čísel (desítkových cifer)

Celá čísla

Nezáporná i záporná čísla:

Přímý kód (signed magnitude)

= znaménkový bit (sign, $\mathbf{0}$ pro nezáporná, $\mathbf{I}$ pro záporná čísla) + (vážený poziční) kód pro absolutní hodnotu čísla (magnitude)

- např. $-123 = [\mathbf{I0}\ldots\mathbf{IIII0II}]_S$
- $\langle -2^{n-1}+1, 2^{n-1}-1 \rangle$
- neefektivní (nevyužitý kód $\mathbf{I0}\ldots$), nevhodný pro aritmetiku (testování znaménka a velikosti absolutních hodnot čísel, různé postupy sčítání a odečítání)

Celá čísla

Nezáporná i záporná čísla:

Přímý kód (signed magnitude)

= znaménkový bit (sign, $\mathbf{0}$ pro nezáporná, $\mathbf{I}$ pro záporné čísla) + (vážený poziční) kód pro absolutní hodnotu čísla (magnitude)

- např. $-123 = [\mathbf{I0} \ldots \mathbf{IIII0II}]_S$
- $\langle -2^{n-1} + 1, 2^{n-1} - 1 \rangle$
- neefektivní (nevyužitý kód $\mathbf{I0}\ldots$), nevhodný pro aritmetiku (testování znaménka a velikosti absolutních hodnot čísel, různé postupy sčítání a odečítání)

Aditivní kód (excess-M, offset binary)

= vážený poziční kód pro (nezáporné) číslo rovno součtu kódovaného čísla a zvolené **konstanty (bias)** M – obvykle $= \mathbf{2^{n-1} - 1}$

- např. $123 = [\mathbf{0} \ldots \mathbf{IIIII0I0}]_{A(127)}$, $-123 = [\mathbf{0} \ldots \mathbf{I00}]_{A(127)}$
- $\langle -2^{n-1} + 1, 2^{n-1} \rangle$
- jinak reprezentovaná nezáporná čísla, složitější sčítání, použití např. pro exponent u reprezentace čísel s plovoucí řádovou čárkou

Celá čísla

Inverzní kód (jedničkově doplňkový, one's complement)

= pro nezáporná čísla vážený poziční kód, pro záporná log. negace všech bitů váženého pozičního kódu absolutní hodnoty čísla – 1. bit má význam znaménka (sign)

- např. $-123 = [\mathbf{I}\ldots\mathbf{0000I00}]_{1'}$
- $\langle -2^{n-1}+1, 2^{n-1}-1\rangle$
- neefektivní (nevyužitý kód $\mathbf{I}\ldots$), „téměř" vhodný pro aritmetiku (odčítání pomocí sčítání se záporným číslem a přičtení přenosu z nejvyššího řádu)

Celá čísla

Inverzní kód (jedničkově doplňkový, one's complement)

= pro nezáporná čísla vážený poziční kód, pro záporná log. negace všech bitů váženého pozičního kódu absolutní hodnoty čísla – 1. bit má význam znaménka (sign)

- např. $-123 = [\mathbf{I}\ldots\mathbf{0000I00}]_{1'}$
- $\langle -2^{n-1}+1, 2^{n-1}-1\rangle$
- neefektivní (nevyužitý kód $\mathbf{I}\ldots$), „téměř" vhodný pro aritmetiku (odčítání pomocí sčítání se záporným číslem a přičtení přenosu z nejvyššího řádu)

Doplňkový kód (dvojkově, two's complement)

= pro nezáporná čísla vážený poziční kód, pro záporná log. negace všech bitů váženého pozičního kódu absolutní hodnoty čísla **zmenšené o 1** nebo s následným **binárním přičtením I** – 1. bit má význam znaménka

- např. $-123 = [\mathbf{I}\ldots\mathbf{0000I0I}]_{2'}$
- $\langle -2^{n-1}, 2^{n-1}-1\rangle$
- efektivní, vhodný pro aritmetiku (odčítání pomocí sčítání se záporným číslem)

ÚKOL

Vytvořte binární reprezentace několika (kladných i záporných) celých čísel pomocí aditivního, inverzního (jedničkově doplňkového) a (dvojkově) doplňkového kódu.

Čísla s řádovou čárkou

= (konečná) **podmnožina racionálních čísel** – přesnost (precision) na maximální počet platných číslic čísla daný počtem bitů reprezentace

Fixní řádová čárka

= pevně zvolený maximální počet $\mathbf{n}$ **platných číslic pro necelou část čísla** (za čárkou)

- místo čísla $x = \frac{x \cdot B^n}{B^n}$ reprezentována pouze **celočíselná část** $\mathbf{x \cdot B^n}$ $\Rightarrow$ přibližná reprezentace čísla
- např. $0,625 = \frac{\mathbf{62},5}{10^2} = \frac{\mathbf{2},5}{2^2} = \frac{(\mathbf{10},1)_2}{2^2}$
- přesnost B^{-n}, „přesnost na n platných číslic za čárkou“

$\Rightarrow$ **celočíselná aritmetika (se zachováním přesnosti)**

- n bitů reprezentace $\Rightarrow$ dvojkové číslice a $B = 2$

Čísla s řádovou čárkou

Fixní řádová čárka

Ekvivalentně s využitím zápisu necelé části čísla v (poziční číselné) soustavě o základu B s max. n číslicemi:

- necelá část F čísla jako součet (případně nekonečné) mocninné řady o základu B:

$$F = a_{-1} \cdot B^{-1} + a_{-2} \cdot B^{-2} + \cdots$$

$$\begin{aligned} (0,625)_{10} &= 6 \cdot 10^{-1} + 2 \cdot 10^{-2} + 5 \cdot 10^{-3} = \\ (0,101)_2 &= 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} \end{aligned}$$

$$\begin{aligned} (0,4)_{10} &= 4 \cdot 10^{-1} = \\ (0,0110011\ldots)_2 &= 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} + 0 \cdot 2^{-4} + 0 \cdot 2^{-5} + 1 \cdot 2^{-6} + 1 \cdot 2^{-7} + \cdots \end{aligned}$$

- získání zápisu $(0, S_{-1}S_{-2}\ldots S_{-n})_B$ (hodnoty) necelé části F čísla a naopak: podobné postupy jako pro celá čísla, jen místo dělení je násobení a naopak:

Čísla s řádovou čárkou

Fixní řádová čárka

Získání (případně nekonečného) zápisu $(0, S_{-1}S_{-2}\ldots)_B$ (hodnoty) necelé části F čísla – postupným násobením:

$a_{-1} = 0$
$i = -1$
while $F > 0$ **do**
  $F = F * B$
  $a_i = F \bmod B$
  $F = F - a_i$
  $i = i - 1$

pro $F = 0,625, B = 10$:
$a_{-1} = 0, i = -1$
$0,625 > 0 : F = 6,25, a_{-1} = 6, F = 0,25, i = -2$
$0,25 > 0 : F = 2,5, a_{-2} = 2, F = 0,5, i = -3$
$0,5 > 0 : F = 5, a_{-3} = 5, F = 0, i = -4$
$0 \not> 0$

Čísla s řádovou čárkou

Fixní řádová čárka

Získání (případně přibližné hodnoty) necelé části F čísla z jejího (konečného) zápisu $(0, S_{-1}S_{-2}\dots S_{-n+1}S_{-n})_B$ – postupným dělením:

$F = a_{-n}$
for $i = -n+1$ **to** -1 **do**
  $F = F/B + a_i$
$F = F/B$; dělení s řádovou čárkou

pro $(0,625)_{10}$ $(B = 10, n = 3, a_{-1} = 6, a_{-2} = 2, a_{-3} = 5)$:
$F = 5$
$i = -2 : F = 2,5$
$i = -1 : F = 6,25$
$F = 0,625$

- převod zápisu necelé části čísla v soustavě o základu B^k $(k \in \mathbb{N})$ na zápis v soustavě o základu B (a naopak) stejný jako u celých čísel

Čísla s řádovou čárkou

Fixní řádová čárka

Binární reprezentace:

= **kód celočíselné části čísla vynásobeného $\mathbf{B^n}$** (ekvivaletně kód celé i necelé části čísla s n číslicemi)

- např. doplňkový kód s 2 platnými číslicemi za čárkou $-5,625 = [\mathbf{I}\ldots\mathbf{0I0\ I0}]_{2'}$
- interval čísel (s danou přesností), hranice dané počtem $t = m + n$ bitů reprezentace a použitým kódem, např. pro doplňkový kód: $\langle -2^{m-1}, 2^{m-1} - 2^{-n} \rangle$
- použití při vyžadování konstantní přesnosti výpočtů s čísly nebo kvůli rychlejší celočíselné aritmetice

Čísla s řádovou čárkou

Plovoucí řádová čárka

= **pohyblivá pozice čárky mezi platnými číslicemi celé a necelé části čísla** $\rightarrow$ reprezentace vědecké notace čísla:

- vyjádření čísla x v semilogaritmickém tvaru o základu B: $\mathbf{x = s \cdot B^e}$
  - normalizovaný: (pro $x \neq 0$) $-1 < s \leq -0,1$ nebo $0,1 \leq s < 1$
  - např. $-5,625 = -0,5625 \cdot 10^1 = -0,703125 \cdot 2^3 = (-0,101101)_2 \cdot 2^3 = (-101,101)_2$
- reprezentace: **znaménkový bit**, **exponent** e (včetně znaménka) do m bitů a **normalizovaný significand („mantissa“)** = necelá část absolutní hodnoty „normalizovaného“ s do n bitů
  - exponent v aditivním kódu s konstantou $M = 2^{m-1} - 1$ – udává rozsah reprezentace $\langle -B^{M+1} + B^{-n}, B^{M+1} - B^{-n} \rangle$
  - significand v kódu pro fixní řádovou čárku s n platnými číslicemi – udává přesnost reprezentace B^{-n}

  $\Rightarrow$ přibližná reprezentace čísla

Čísla s řádovou čárkou

Plovoucí řádová čárka

Různé formáty reprezentace s různým rozsahem a přesností – standard **IEEE 754** (1985):

- $B = 2 \Rightarrow$ v „normalizovaném“ s číslice za čárkou vždy $1 \rightarrow s \cdot 2$ ($n \rightsquigarrow n + 1$, tzv. skrytá $\mathbf{I}$), $e - 1$, significand ve váženém pozičním kódu
- **single** (**float**, 32 bitů) – 8 bitů pro exponent, 23 bitů pro significand, rozsah $\sim \langle -10^{38}, 10^{38} \rangle$, asi 7 platných desítkových číslic, např.

$$-5,625 \quad = \quad [\mathbf{I\ IOOOOOOI\ 0IIOIOOOOOOOOOOOOOOOOO}]_{single}$$

- **double** (64 bitů) – 11 bitů pro exponent, 52 bitů pro significand, rozsah $\sim \langle -10^{308}, 10^{308} \rangle$, asi 16 platných desítkových číslic
- další: half (16 bitů, 5 pro exponent), extended (long double, 80 bitů, 15 pro exponent), quad (128 bitů, 15 pro exponent)
- **speciální „čísla“**: $-\infty, +\infty$ (exponent samé $\mathbf{I}$, significand $= \mathbf{0}$), NaN (Not a Number, exponent samé $\mathbf{I}$, significand $\neq \mathbf{0}$), $-0 \neq 0$ (exponent i significand $= \mathbf{0}$), tzv. denormalizovaná (exponent $= \mathbf{0}$, „nenormalizovaný“ significand $\neq \mathbf{0}$)

Čísla s řádovou čárkou

Plovoucí řádová čárka

- **aritmetika s plovoucí řádovou čárkou**
  - zaokrouhlení (significand n platných číslic) a výjimky (pro nedefinované operace)
  - s operacemi propagace chyby (zaokrouhlení) a operace neasociativní a nedistributivní!

  $\Rightarrow$ **POZOR** na porovnání!
  - operace měřítkem výkonnosti počítačů, jednotka **FLOPS** (FLoating point Operations Per Second)
  - implementována ve FPU (Floating Point Unit) – dnes součást CPU
- (mnohem) větší interval čísel než u fixní řádové čárky (hranice dané exponentem), na úkor nižší přesnosti

ÚKOL

Vytvořte binární reprezentace několika (kladných i záporných) racionálních čísel s fixní i plovoucí řádovou čárkou.

Text

= posloupnost tisknutelných znaků = písmen různých abeced a cifer (= alfanumerické znaky) a symbolů (mezera, interpunkce, matematické aj.)

\+ řídící znaky (v textovém terminálu) – některé v **plain text**u, např. pro konec řádku

→ kódování znaků na binární hodnoty pomocí kódových tabulek

Text

= posloupnost tisknutelných znaků = písmen různých abeced a cifer (= alfanumerické znaky) a symbolů (mezera, interpunkce, matematické aj.)

\+ řídící znaky (v textovém terminálu) – některé v **plain text**u, např. pro konec řádku

→ kódování znaků na binární hodnoty pomocí kódových tabulek

ASCII (American Standard Code for Information Interchange, 1967)

- standarní kódová tabulka („kódování“) pro **písmena anglické abecedy a cifry**, symboly (mezery, interpunkce, matematických aj.), a **řídící znaky** (odřádkování, návrat vozíku, backspace, tabulátor aj.)
- znak původně do 7 bitů = 128 znaků
- později 8. bit pro rozšíření o dalších 128 znaků: některé **znaky národních abeced**, další speciální znaky (**grafické**, jednotky aj.)

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	*[NULL]*	32	20	*[SPACE]*	64	40	@	96	60	`
1	1	*[START OF HEADING]*	33	21	!	65	41	**A**	97	61	**a**
2	2	*[START OF TEXT]*	34	22	"	66	42	**B**	98	62	**b**
3	3	*[END OF TEXT]*	35	23	#	67	43	**C**	99	63	**c**
4	4	*[END OF TRANSMISSION]*	36	24	$	68	44	**D**	100	64	**d**
5	5	*[ENQUIRY]*	37	25	%	69	45	**E**	101	65	**e**
6	6	*[ACKNOWLEDGE]*	38	26	&	70	46	**F**	102	66	**f**
7	7	*[BELL]*	39	27	'	71	47	**G**	103	67	**g**
8	8	*[BACKSPACE]*	40	28	(	72	48	**H**	104	68	**h**
9	9	*[HORIZONTAL TAB]*	41	29	)	73	49	**I**	105	69	**i**
10	A	*[LINE FEED]*	42	2A	*	74	4A	**J**	106	6A	**j**
11	B	*[VERTICAL TAB]*	43	2B	+	75	4B	**K**	107	6B	**k**
12	C	*[FORM FEED]*	44	2C	,	76	4C	**L**	108	6C	**l**
13	D	*[CARRIAGE RETURN]*	45	2D	-	77	4D	**M**	109	6D	**m**
14	E	*[SHIFT OUT]*	46	2E	.	78	4E	**N**	110	6E	**n**
15	F	*[SHIFT IN]*	47	2F	/	79	4F	**O**	111	6F	**o**
16	10	*[DATA LINK ESCAPE]*	48	30	**0**	80	50	**P**	112	70	**p**
17	11	*[DEVICE CONTROL 1]*	49	31	**1**	81	51	**Q**	113	71	**q**
18	12	*[DEVICE CONTROL 2]*	50	32	**2**	82	52	**R**	114	72	**r**
19	13	*[DEVICE CONTROL 3]*	51	33	**3**	83	53	**S**	115	73	**s**
20	14	*[DEVICE CONTROL 4]*	52	34	**4**	84	54	**T**	116	74	**t**
21	15	*[NEGATIVE ACKNOWLEDGE]*	53	35	**5**	85	55	**U**	117	75	**u**
22	16	*[SYNCHRONOUS IDLE]*	54	36	**6**	86	56	**V**	118	76	**v**
23	17	*[END OF TRANS. BLOCK]*	55	37	**7**	87	57	**W**	119	77	**w**
24	18	*[CANCEL]*	56	38	**8**	88	58	**X**	120	78	**x**
25	19	*[END OF MEDIUM]*	57	39	**9**	89	59	**Y**	121	79	**y**
26	1A	*[SUBSTITUTE]*	58	3A	:	90	5A	**Z**	122	7A	**z**
27	1B	*[ESCAPE]*	59	3B	;	91	5B	[	123	7B	{
28	1C	*[FILE SEPARATOR]*	60	3C	<	92	5C	\	124	7C	|
29	1D	*[GROUP SEPARATOR]*	61	3D	=	93	5D	]	125	7D	}
30	1E	*[RECORD SEPARATOR]*	62	3E	>	94	5E	^	126	7E	~
31	1F	*[UNIT SEPARATOR]*	63	3F	?	95	5F	_	127	7F	*[DEL]*

ASCII tabulka, zdroj

Rozšířené ASCII

- např. ISO 8859-1, CP1252 (Microsoft), CP437 (IBM) pro západoevropské jazyky
- pro znaky české abecedy (východoevropské/středoevropské jazyky):
  - **ISO 8859-2 (ISO Latin 2)**: dříve používaná v unixových operačních systémech (OS), na webu a v e-mailu
  - **Windows 1250 (CP1250)** (Microsoft): používaná v OS MS Windows,
  - **Mac CE** (Apple): používaná v Apple Mac OS
  - CP852 (PC Latin 2) (IBM): používaná v OS MS DOS, grafické znaky
  - další (česko-slovenské): kód Kamenických (hojně používaný v OS MS DOS), KOI8-ČS (v rámci RVHP) aj.
- programy pro konverzi textů mezi kódováními

Text

Rozšířené ASCII

- např. ISO 8859-1, CP1252 (Microsoft), CP437 (IBM) pro západoevropské jazyky
- pro znaky české abecedy (východoevropské/středoevropské jazyky):
  - **ISO 8859-2 (ISO Latin 2)**: dříve používaná v unixových operačních systémech (OS), na webu a v e-mailu
  - **Windows 1250 (CP1250)** (Microsoft): používaná v OS MS Windows,
  - **Mac CE** (Apple): používaná v Apple Mac OS
  - CP852 (PC Latin 2) (IBM): používaná v OS MS DOS, grafické znaky
  - další (česko-slovenské): kód Kamenických (hojně používaný v OS MS DOS), KOI8-ČS (v rámci RVHP) aj.
- programy pro konverzi textů mezi kódováními

ASCII art

= výtvarné umění kresby obrázků pomocí znaků ASCII (v neproporcionálním fontu)

- např. emotikony („smajlíky“): :-), :-(aj.

Text

ASCII art, zdroj

Text

EBCDIC (Extended Binary Coded Decimal Interchange Code, 1964, IBM)

- základní osmibitový, rozšířené 16-bitové – různé pro různé národní abecedy
- **nespojitý pro písmena anglické abecedy**, dnes nepoužívaný

Text

Unicode (1987–1991)

- rozšířené ASCII nestačí a jsou ad-hoc (a navíc problematické pro východoasijské, arabské, hebrejské aj. znaky)
- původně 16-bitová kódová tabulka znaků **UCS-2 (Universal Character Set)**, později pro znaky místo kódů tzv. kódové body $\mathbf{U} + (\text{čísl}\mathbf{o})_{\mathbf{16}}$ kódované do binární reprezentace

= **ISO/IEC 10646** (definice UCS-4, 31-bitová) + kódování, algoritmy pro texty zprava doleva a oboustranné texty, porovnávání textů aj.

- UCS = **otevřená množina pojmenovaných znaků všech abeced**, symbolů a řídících znaků, téměř 155 000 znaků (2024)
- znakové sady (bloky) = podmnožiny znaků, např. původní ASCII (prvních 128) a její rozšíření (ISO 8859-1, prvních 256), BMP (Basic Multilingual Plane) = prvních 65536 znaků UCS (národní abecedy, symboly, CJK, Han aj.)

Text

UTF (UCS Tranformation Format)

= kódování kódových bodů do binární reprezentace

- **UTF-8**: do posloupnosti 1 až 6 bytů, všeobecně používaný (zejména na Internetu/webu dle RFC 3629 a v unixových OS)

Tabulka: Kódování UTF-8

$U+00000000$ – $U+0000007F$	$0xxxxxxx$
$U+00000080$ – $U+000007FF$	$110xxxxx$ $10xxxxxx$
$U+00000800$ – $U+0000FFFF$	$1110xxxx$ $10xxxxxx$ $10xxxxxx$
$U+00010000$ – $U+001FFFFF$	$11110xxx$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$
$U+00200000$ – $U+03FFFFFF$	$111110xx$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$
$U+04000000$ – $U+7FFFFFFF$	$1111110x$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$ $10xxxxxx$

- např. „Příliš“ $= [50\ C599\ C3AD\ 6C\ 69\ C5A1]_{16}$ (ř $= U+(159)_{16} = (101\ 011001)_2$, í $= U+(ED)_{16} = (11\ 101101)_2$, š $= U+(161)_{16} = (101\ 100001)_2$)
- BMP 1 až 3 byty, české znaky 1 nebo 2 byty (diakritické)
- byty FE_{16}, FF_{16} nepoužity
- nezávislý na endianitě systémů

Text

UTF (UCS Tranformation Format)

- **UTF-16**: do posloupnosti 2 nebo 4 bytů, používaný zejména v OS MS Windows a programovacím jazyku Java, dříve UCS-2

Tabulka: Kódování UTF-16

$U+000000 - U+00FFFF$	$xxxxxxxx\ xxxxxxxx$
$U+010000 - U+10FFFF$	$110110xx\ xxxxxxxx\ 110111xx\ xxxxxxxx$

  - např. „Příliš“ $= [0050\ 0159\ 00ED\ 006C\ 0069\ 0161]_{16}$ (ř $= U+(159)_{16}$, í $= U+(ED)_{16}$, š $= U+(161)_{16}$)
  - BMP včetně českých znaků 2 byty
  - **BOM (Byte-Order Mark, UTF signatura)** = znak $\mathbf{U+FEFF}$ („nedělitelná mezera nulové šířky“) na začátku textu (souboru) k rozlišení endianity systému ($FE_{16}FF_{16}$ v big-endian, $FF_{16}FE_{16}$ v little-endian, opačně neplatný kód)

Text

UTF (UCS Tranformation Format)

- **UTF-16**: do posloupnosti 2 nebo 4 bytů, používaný zejména v OS MS Windows a programovacím jazyku Java, dříve UCS-2

Tabulka: Kódování UTF-16

$U+000000 - U+00FFFF$	$xxxxxxxx\ xxxxxxxx$
$U+010000 - U+10FFFF$	$110110xx\ xxxxxxxx\ 110111xx\ xxxxxxxx$

  - např. „Příliš“ $= [0050\ 0159\ 00ED\ 006C\ 0069\ 0161]_{16}$ (ř $= U+(159)_{16}$, í $= U+(ED)_{16}$, š $= U+(161)_{16}$)
  - BMP včetně českých znaků 2 byty
  - **BOM (Byte-Order Mark, UTF signatura)** = znak $\mathbf{U+FEFF}$ („nedělitelná mezera nulové šířky“) na začátku textu (souboru) k rozlišení endianity systému ($FE_{16}FF_{16}$ v big-endian, $FF_{16}FE_{16}$ v little-endian, opačně neplatný kód)
- další: UTF-32/UCS-4 (pevně do 4 bytů, příliš nepoužívané), **UTF-7** (do posloupností 7-bitových ASCII kodů, pro e-mail) aj.
- programy pro konverzi textů mezi kódováními

Text

Kód pro nový řádek

- různý v různých operačních systémech
- **LF (Line Feed, odřádkování, $\mathbf{A_{16}}$)**: v unixových OS (včetně Apple Mac OS X)
- **CR (Carriage Return, návrat vozíku, $\mathbf{D_{16}}$) + LF**: v OS MS DOS a Windows
- **CR**: v Apple Mac OS do verze 9
- programy pro konverzi

Text

Kód pro nový řádek

- různý v různých operačních systémech
- **LF (Line Feed, odřádkování, $\mathbf{A_{16}}$)**: v unixových OS (včetně Apple Mac OS X)
- **CR (Carriage Return, návrat vozíku, $\mathbf{D_{16}}$) + LF**: v OS MS DOS a Windows
- **CR**: v Apple Mac OS do verze 9
- programy pro konverzi

Escape sekvence

= posloupnosti znaku **ESC (Escape, $\mathbf{1B_{16}}$)** následovaného jedním nebo více znaky z ASCII

- speciální významy (interpretace), např. pro specifikaci pozice kurzoru, barvy nebo fontu v textovém terminálu, přepnutí módu zařízení aj.
- používané zejména v unixových OS

ÚKOL

Vytvořte binární reprezentace několika českých slov s diakritickými znaky pomocí kódování UTF-8 a UTF-16. K dispozici máte Unicode tabulku znaků (UCS) s kódovými body.

Detekční a samoopravné kódy

= **zabezpečení** (binární reprezentace) **dat proti chybám** při ukládání a přenosu

- chyba = **změna bitu**
- detekční: při čtení dat (příjemcem) umožňují detekovat v datech určité chyby, při chybě data obvykle znovu vyžádána
- samoopravné (error correction codes, ECC): navíc možnost opravy určitých chyb

= (většinou) **redundantní doplnění dat o detekční/samoopravný kód dat**

- při čtení dat (příjemcem), příp. včetně přidaného kódu, také výpočet kódu a pokud je jiný než přijatý, příp. nenulový, detekuje/opraví chyby

Detekční a samoopravné kódy

Detekční kódy (error detection codes)

Opakování

- data rozdělena do bloků, bloky opakovány = kód
- příjemce porovná původní (první) a opakované bloky, různé = chyba
- jednoduché, neefektivní, nedetekuje stejné chyby ve všech blocích

Detekční a samoopravné kódy

Detekční kódy (error detection codes)

Opakování

- data rozdělena do bloků, bloky opakovány = kód
- příjemce porovná původní (první) a opakované bloky, různé = chyba
- jednoduché, neefektivní, nedetekuje stejné chyby ve všech blocích

Parita

- data rozdělena do bloků, **sudá/lichá** = pro lichý/sudý počet $\mathbf{I}$ v bloku je kód (**paritní bit**) roven $\mathbf{I}$, jinak $\mathbf{0}$
- příjemce provede totéž a porovná paritní bit, různý = chyba
- výpočet paritního bitu pomocí log. operace XOR, příjemce provede XOR i s paritním bitem, nenulový (sudá)/nejedničkový (lichá) = chyba
- např. pro $\mathbf{II0I0}$ je $\mathbf{I}$ (sudá)/$\mathbf{0}$ (lichá)
- detekuje pouze lichý počet chyb
- použití pro detekci chyb při přenosu z/do pamětí a u diskových zařízení

Detekční a samoopravné kódy

Detekční kódy (error detection codes)

Kontrolní součet (checksum)

- sudá parita = log. operace XOR bloků dat
- modulární součet = blok (dvojkového) **doplňkového kódu aritmetického součtu** čísel reprezentovaných bloky dat ve váženém pozičním kódu
- a jiné
- příjemce provede XOR/součet i s kódem, nenulový = chyba
- např. pro **II00 0I0I I0I0** je **00II** (při XOR)/**0I0I** (při aritm. součtu)
- detekuje lichý počet chyb na stejných pozicích v blocích
- nedetekuje změnu pořadí bloků nebo přidání/odebrání nulových bloků
- použití u diskových zařízení a komunikačních protokolů

Detekční a samoopravné kódy

Detekční kódy (error detection codes)

Cyklický redundantní součet (Cyclic Redunadacy Check, CRC)

- založen na **binárních cyklických kódech** (vychází z algebraické teorie binárních konečných polí/okruhů a binárních polynomů nad nimi)
- teoreticky: bity dat reprezentují koeficienty polynomu, který je vydělen tzv. generujícím polynomem řádu n (pro kód řádu n), kód tvoří koeficienty zbytku
- prakticky: za data se přidá blok nul velikosti n (pro kód řádu n), bin. reprezentace generujícího polynomu (divisor) má $n+1$ bitů, od 1. nenulového bitu dat se opakovaně provádí XOR s divisorem dokud nejsou všechny bity dat rovny $\mathbf{0}$, kód $=$ přidaný blok
- příjemce provede totéž s kódem místo bloku nul, nenulový $=$ chyba
- blok např. byte (CRC-8), 2 byte (CRC-16), 4 byte (**CRC-32**) – použití u počítačových sítí a úložných zařízení
- např. pro $\mathbf{II0I00II}$ a divisor $\mathbf{I00II}$ (gen. polynom x^4+x+1, CRC-4) je $\mathbf{I00I}$
- sudá parita je speciální případ (CRC-1, gen. polynom $x+1$)

Další: založené na **Hammingově vzdálenosti**, lib. **hashovací funkce** aj.

Detekční a samoopravné kódy

Cyklický redundantní součet (Cyclic Redunadacy Check, CRC)

	data								kód					
$D3_{16}$	**I**	**I**	**0**	**I**	**0**	**0**	**I**	**I**	**0**	**0**	**0**	**0**	0_{16}	
13_{16}	**I**	**0**	**0**	**I**	**I**									gen. polynom $x^4 + x + 1$
$4B_{16}$	**0**	**I**	**0**	**0**	**I**	**0**	**I**	**I**	**0**	**0**	**0**	**0**	0_{16}	XOR
		I	**0**	**0**	**I**	**I**								
7_{16}	**0**	**0**	**0**	**0**	**0**	**I**	**I**	**I**	**0**	**0**	**0**	**0**	0_{16}	XOR
						I	**0**	**0**	**I**	**I**				posun na 1. nenulový bit dat
3_{16}	**0**	**0**	**0**	**0**	**0**	**0**	**I**	**I**	**I**	**I**	**0**	**0**	12_{16}	XOR
							I	**0**	**0**	**I**	**I**			
1_{16}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**I**	**I**	**0**	**I**	**0**	10_{16}	XOR
								I	**0**	**0**	**I**	**I**		
0_{16}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**I**	**0**	**0**	**I**	9_{16}	XOR

Obrázek: CRC-4: postup výpočtu

Detekční a samoopravné kódy

Samoopravné kódy (Error Correction Codes, ECC, Forward Error Correction, FEC)

- použití u úložných zařízení a bezdrátové komunikace

Opakování

- většinově se vyskytující blok je správný

Detekční a samoopravné kódy

Samoopravné kódy (Error Correction Codes, ECC, Forward Error Correction, FEC)

- použití u úložných zařízení a bezdrátové komunikace

Opakování

- většinově se vyskytující blok je správný

Multidimenzionální parita

- data organizována po blocích do mřížky a spočítány parity pro řádky i sloupce
- pro chybný bit jsou chybné řádková i sloupcová parita

0	1	1	1	0
1	1	0	0	1
0	1	1	0	0
0	0	1	1	1
0	1	0	1	1

Obrázek: 2-dimenzionální lichá parita

- n-dimenzionální parita umožňuje opravit $n/2$ chyb

Detekční a samoopravné kódy

Samoopravné kódy (Error Correction Codes, ECC, Forward Error Correction, FEC)

Hammingův kód

- založen na Hammingově vzdálenosti a paritě
- umožňuje detekovat až 2 současné chyby a opravit 1 chybu (Hammingova vzdálenost ≤ 1)
- složitější konstrukce
- použití u operačních pamětí

Další (výkonnější): **Reed-Solomonovy kódy** (CD/DVD/BD), BCH kódy, konvoluční kódy aj.