

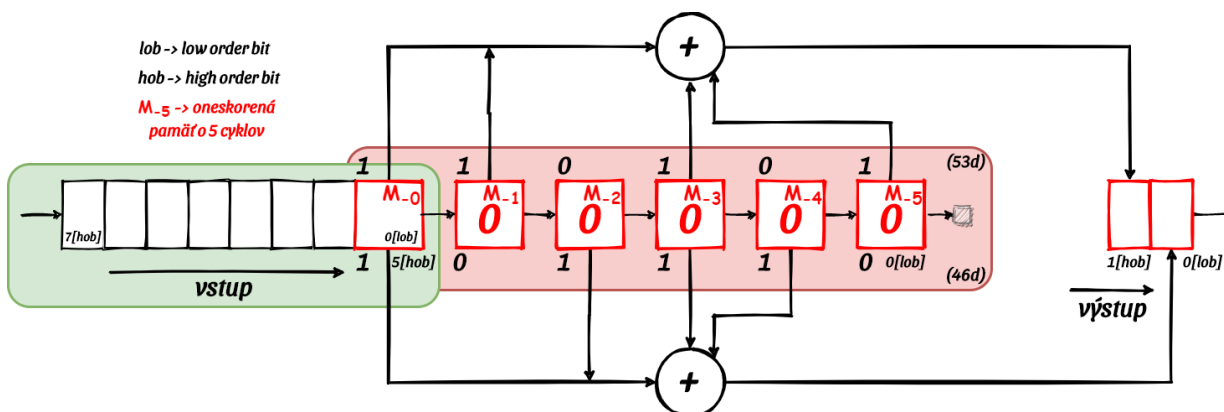


BMS - Bezdrátové a mobilní sítě

Informácie k projektu

Projekt BMS

Naprogramujte v jazyku C/C++/Python jednoduchú konzolovú aplikáciu (*bms*), ktorá bude realizovať kódovanie (encoding) a dekódovanie (decoding) vstupnej správy pomocou konvolučného kodéra vo variante znázornenej na obrázku nižšie. Kódový pomer konvolučného kodéra je $\frac{1}{2}$.



Parametre programu:

```
./bms -e <<< vstup=[ASCII znaky] (služi pre zakódovanie)
```

```
./bms -d <<< vstup=[znaky hodnôt jednotlivých bitov] (služi pre dekódovanie)
```

Funkcie programu:

Program berie vstup zo štandardného vstupu **stdin** a výstup programu je vypísaný na štandardný výstup **stdout**. Vstupom programu bude reťazec znakov (vo variante **-e** napr. "test", podporuje iba ASCII znakovú sadu, na vstupe očakávejte iba znaky [a-z][A-Z][0-9]; vstup pre variantu **-d** je napr. "01101110..", na vstupe očakávejte iba znaky [0,1]).

Kódovanie (-e)

- Program si musí vstupný reťazec skonvertovať do jednotlivých bajtov a tie zakódovať pomocou konvolučného kodéra. Príklad ako vyzerá použitie programu v tomto režime je:

```
$ ./bms -e <<< test
001011100111001101000001010010010001101100000101001000001001001101010110000

$ ./bms -e <<< BMS
0010011101100101111111001101010010001111110100010110110111

$ ./bms -e <<< A
00100111011011100111011011
```

Príklady vychádzajú priamo zo zadanej schémy konvolučného kodéra a môžete si pomocou nich overiť správnú funkčnosť Vášho programu.

Dekódovanie (-d)

- Validným vstupom programu sú iba symboly [0,1] reprezentujúce jednotlivé bity prenášanej správy (Pozor!, do vstupu sa počítajú aj zakódované bity inicializačných registrov, viď. sekcia "ďalšie poznámky"). Program musí vstup spracovať, dekódovať a následne dekódovanú správu vypísať na výstup vo forme ASCII reťazca. Dekódovanie bude prebiehať pomocou Viterbiho algoritmu [1] s pomocou vytvoreného Trellis prechodového diagramu [2], ktorý prislúcha zadanému konvolučnému dekóderu. Keďže počas prenosu správy cez prenosové médium môže dôjsť ku chybám (prehodenie hodnôt bitov na rôznych pozíciách) a zadaný konvolučný dekóder spadá do kategórie kódov s opravou chyby (ECC - Error Correction Code) [3], tak Vaše dekódovanie by malo minimálne dokázať úspešne dekódovať (opraviť) aj viacnásobné bitové chyby na rôznych pozíciách (pre zjednodušenie úlohy, počítajte maximálne s 2-bitovými chybami na 3 rôznych pozíciách {chyby nie sú v rámci jedného celého zhluku}, viď. príklady použitia). Príklad ako vyzerá použitie programu v tomto režime je:

```
$ ./bms -d <<< 0010011101100101111111001101010010001111110100010110110111
BMS
```

```
$ ./bms -d <<< 1110011101100101111111001101010010001111110100010110110111
BMS
```

```
$ ./bms -d <<< 1110011101101001111111001101010010001111110100010110110111
BMS
```


```
$ ./bms -d <<< 1110011101101001111111001011010010001111110100010110110111
BMS
```

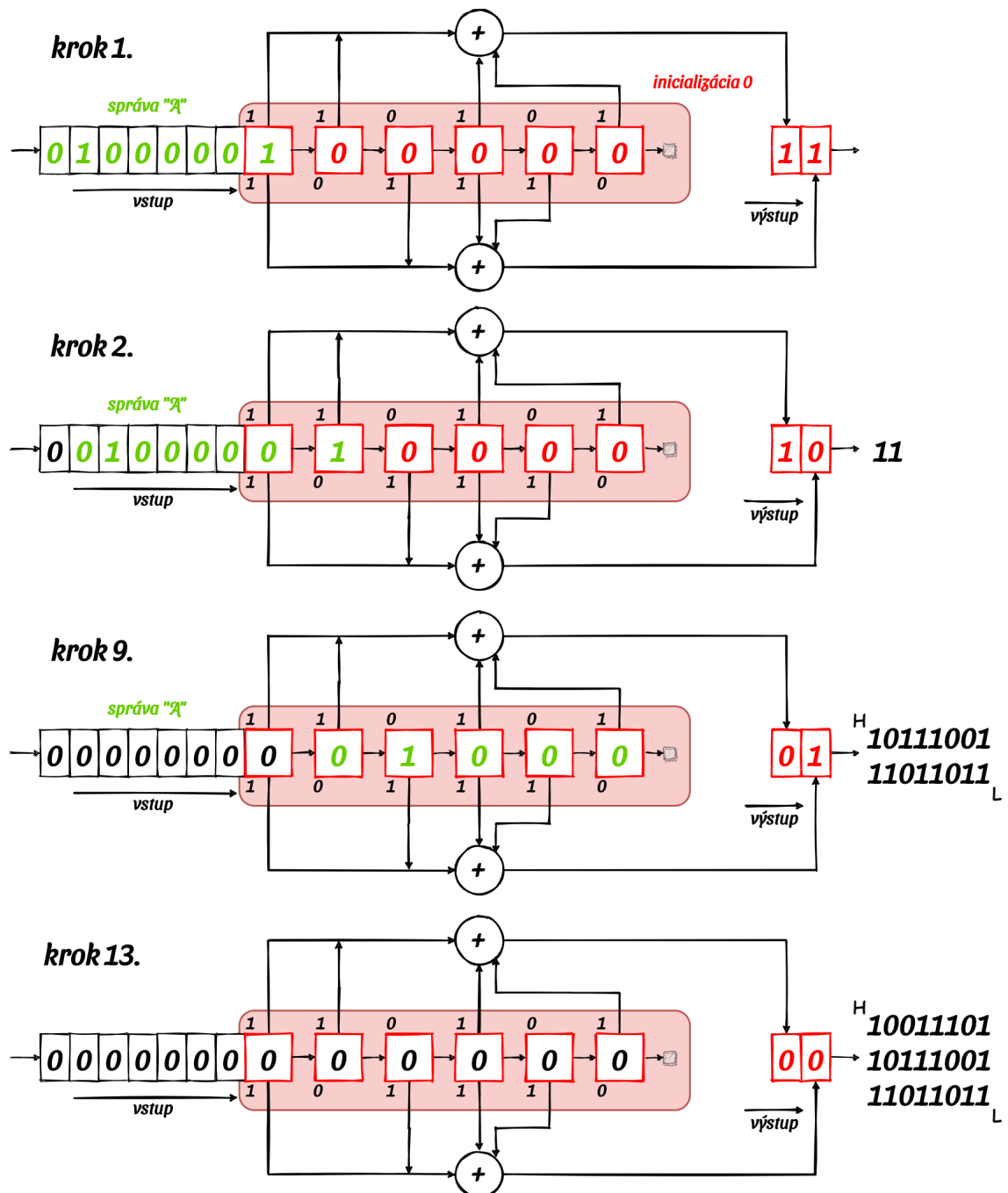
```
$ ./bms -d <<< 001001110110011110001010101001111010011001101101111011101001000001000010100100100001101011
CC0de
```

```
$ ./bms -d <<< 00100111011011100111011011
A
```

Príklady vychádzajú priamo zo zadanej schémy konvolučného kodéra a môžete si pomocou nich overiť správnu funkčnosť Vášho programu.

Ďalšie poznámky

- Pri implementácii je možné použiť ľubovoľnú knižnicu dostupnú na servery *merlin.fit.vutbr.cz*, na ktorom se budú Vaše programy testovať!
- Pri riešení projektu využite materiály, ktoré máte poskytnuté v rámci prednášok BMS ( **BMS04_GSM_MNG.pdf**, kapitola Convolutional codes)
- Ďalšie zdroje: [\[1\]](#), [\[2\]](#), [\[3\]](#), [\[4\]](#), [\[5\]](#)
- **Bonusové body:** Bonusové body získate, pokiaľ Váš program dokáže robiť operácie kódovania, dekódovania v rámci ľubovoľného jednoduchého konvolučného kodéra (tzn. kódový pomer je $\frac{1}{2}$ a je v ňom horná a dolná spätná väzba), ktorý bude parametrizovaný 3 hodnotami v tvare **(X,Y,Z)**. Parameter X vyjadruje počet oneskorených stupňov pamäťových blokov. Parameter Y je číslo vyjadrujúce prepojenie hornej spätnej väzby a parameter Z je číslo vyjadrujúce prepojenie dolnej spätnej väzby. V projekte máte napevno zadany konvolučný kodér typu (5,53,46) -> 5 členov oneskorenia $\{M_{-1} \text{ až } M_{-5}\}$, 53d=110101b {horná schéma prepojenia}, 46d=101110b {dolná schéma prepojenia}. Pre viac informácií ohľadom bonusového zadania, kontaktujte konzultanta.
- Podrobnejší príklad s krokmi ako vyzerá *zakódovanie správy* konvolučným kodérom z Vášho zadania na reálnom vstupe je vyobrazený na obrázku:



vstup (správa) + inicializované registre, po 13. krokoch (kódovanie) =

**0100 0001 => 00 1001 1101 1011
1001 1101 1011**

- Všimnite si, že síce vstup bol 8 bitový (1 bajt), pri kódovom pomere $\frac{1}{2}$ by výstup mal byť 16 bitov (2 bajty), ale výstup v tomto prípade bude celkovo až 28 bitov. Tento dôvod je prostý a to kvôli tomu, že do kódovania sa musia započítať aj posuvné pamäťové bloky ktoré boli inicializované 0. **Pozor!** vo Vašom projekte je požadované dodržať túto skutočnosť.

- Projekt odovzdajte ako súbor vo formáte **ZIP** (TAR, 7Zip, RAR a iné == 0 bodov)
- Daný archív pomenujte Vaším loginom: **xlogin00.zip**
- Daný archív neobsahuje v sebe žiadne priečinky (zložky), priečinky pomenované Vaším loginom, src, projekt a podobné == 0 bodov
- Makefile vytvorí program **./bms**
- Počas behu programu nevypisujte na stdout zbytočnosti, prípadné chyby vypisujte na **stderr**
- Dokumentácie: dostatočne okomentovaný a pochopiteľný zdrojový kód

Termín odevzdání: 16.12.2020