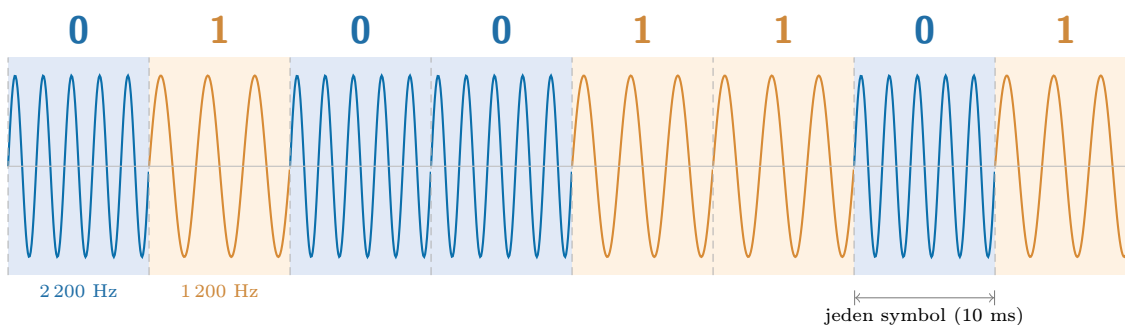


ZADÁNÍ SEMESTRÁLNÍ PRÁCE

Akustický modem

Distribuční trafostanice v odlehlých částech sítě posílají dispečinku telemetrii – okamžité zatížení, napětí na vývodech nebo teplotu oleje transformátoru. Tam, kde není datové připojení, může být jediným dostupným spojem hlasový kanál: služební radiostanice nebo pronajatá telefonní linka. Ta přenese lidskou řeč, tedy zhruba pásmo od 300 do 3 400 Hz, ale žádná data.

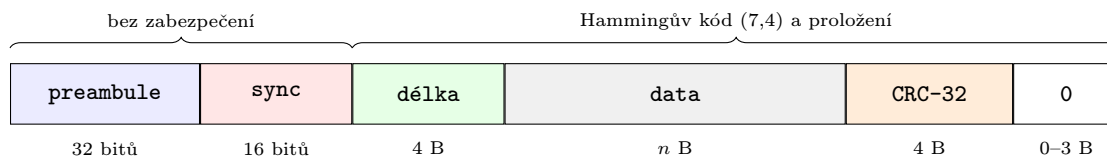
Řešení je prosté: data se **zahrají**. Vysílač převede bity na tóny, kanál je přenese jako jakýkoli jiný zvuk a přijímač z nich bity zase poskládá. Způsob, jakým se tóny střídají, se jmenuje klíčování frekvenčním posuvem (*frequency-shift keying*, FSK): jeden tón znamená nulu, druhý jedničku. Typickým zástupcem je modem Bell 202 z roku 1976, který klíčuje mezi tóny 1 200 a 2 200 Hz modulační rychlostí 1 200 Bd [7]. Modemy Bell 202 pro pronajaté linky se dodnes nabízejí pro přenos dat systémů SCADA z elektráren a rozvodů do řídicích center [8]. Stejně dva tóny se stejným významem – 1 200 Hz pro jedničku, 2 200 Hz pro nulu – budete generovat i vy.



Obrázek 1: Klíčování frekvenčním posuvem: bity zahrané dvěma tóny (ilustrativní příklad).

Dnes tyto tóny můžete slyšet například v průmyslovém protokolu HART, kterým komunikují snímače a polohovače ventilů v procesních provozech. Digitální data se v něm namodulují na analogový signál proudové smyčky 4–20 mA právě klíčováním podle Bell 202 [9]. V Severní Americe se modulací Bell 202 přenáší na pevné lince číslo volajícího, a to v krátkém datovém shluku mezi prvním a druhým zazvoněním. Na pásmu VHF touto modulací vysílají i stanice amatérského systému APRS, jejichž signál umí ze zvukové karty demodulovat například program Dire Wolf [10].

Zvuková cesta ovšem není spolehlivá. Do signálu se přimíchá šum, hlasitost kolísá, na lince cvakne relé a kus přenosu prostě zmizí. Vysílač proto musí data zabezpečit tak, aby je přijímač dokázal obnovit i z poškozené nahrávky. V této semestrální práci si postavíte obě strany takového spoje. Data mezi nimi putují v rámci, jehož skladbu ukazuje obrázek 2.



Obrázek 2: Struktura přenášeného rámce.

Zadání

Naprogramujte v jazyce ANSI C přenositelnou¹ **konzolovou aplikaci**, která umí zakódovat obsah *libovolného* souboru do zvukové nahrávky ve formátu WAV a z takové nahrávky jej zase obnovit.

Program bude spouštěn příkazem `modem.exe`² s kombinací následujících argumentů – výrazy v ložených závorkách (`<>`) označují povinné argumenty (příklad spuštění programu je uveden v ukázce konzolového výstupu 1):

- `-e <input-file>` Zapne režim vysílání. Obsah souboru `<input-file>` bude zakódován do nahrávky. V případě, že uživatel zadá neexistující soubor, program vypíše chybové hlášení `"Input file not found!\n"` a vrátí hodnotu 1. Je-li soubor prázdný nebo větší než 4 096 bajtů, program vypíše hlášení `"Invalid input size!\n"` a vrátí hodnotu 4.
- `-d <input-file>` Zapne režim příjmu. Ze souboru `<input-file>` ve formátu WAV budou obnovena přenesená data. Chybějící soubor je ošetřen stejně jako v případě přepínače `-e`.
- `-o <path>` Výstupní soubor. Pokud umístění neexistuje nebo do něj nelze zapisovat (např. jde o adresář), program vypíše hlášení `"Invalid output destination!\n"` a skončí s návratovou hodnotou 2.

Přepínače `-e` a `-d` se vzájemně vylučují a právě jeden z nich musí být zadán, přepínač `-o` musí být zadán vždy. Neplatnou kombinací argumentů je tedy chybějící přepínač `-e` i `-d`, současné zadání obou těchto přepínačů nebo chybějící přepínač `-o`. Na takovou kombinaci program reaguje vypsáním nápovědy, která stručně popíše jeho použití a všechny přepínače, a návratovou hodnotou 3. Podoba nápovědy je na vás. Opakuje-li se některý přepínač, platí jeho poslední hodnota.

Program nejprve zkontroluje argumenty, poté zpracuje vstupní soubor a výstupní soubor otevře až ve chvíli, kdy do něj má co zapsat. Pokud zpracování skončí chybou, program výstupní soubor vůbec neotevře. Proběhne-li přenos v pořádku, program vypíše hlášení `"Encoded <n> B into <t> s of audio.\n"`, resp. `"Decoded <n> B from <t> s of audio.\n"`, kde `<n>` je počet přenesených bajtů a `<t>` délka nahrávky v sekundách zaokrouhlená na dvě desetinná místa, a skončí s návratovou hodnotou `EXIT_SUCCESS`. Hlášení o úspěšném přenosu vypisujte na standardní výstup (`stdout`), všechna ostatní hlášení včetně nápovědy na standardní chybový výstup (`stderr`).

Hotovou práci odevzdejte v jediném archivu typu ZIP prostřednictvím automatického odevzdávacího a validačního systému. Postupujte podle instrukcí uvedených na webu předmětu. Archiv nechtě obsahuje všechny zdrojové soubory potřebné k přeložení programu, **Makefile** pro Windows i Linux (pro překlad v Linuxu připravte soubor pojmenovaný **Makefile** a pro Windows **Makefile.win**) a dokumentaci ve formátu PDF vytvořenou v typografickém systému $\text{T}_{\text{E}}\text{X}$ ($\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$). Bude-li některá z částí chybět, kontrolní skript vaši práci odmítne.

¹Je třeba, aby bylo možné Váš program přeložit a spustit na PC s operačním prostředím Win32/64 (tj. operační systémy Microsoft Windows NT/2000/XP/Vista/7/8/10/11) a s běžnými distribucemi Linuxu (např. Ubuntu, Debian, Red Hat, Arch atp.). Server, na který budete Vaši práci odevzdávat a který ji otestuje, má nainstalovaný operační systém Debian GNU/Linux 11 (bullseye) s jádrem verze 6.1.0-50-amd64 a s překladačem gcc 12.2.0.

²Přípona `.exe` je povinná i při sestavení pro Linux, zejména při automatické kontrole validačním systémem.

Specifikace nahrávky

Nahrávky můžete číst i zapisovat vlastním kódem, nebo libovolnou knihovnou (např. `dr_wav` [1]), jejíž zdrojové soubory pak musí být součástí odevzdaného archivu. Program pracuje výhradně s nahrávkami s parametry z horní části tabulky 1, jinou nahrávku odmítne hlášením `"Unsupported WAV format!\n"` a návratovou hodnotou 10. Není-li vstupní soubor platnou nahrávkou WAV nebo neobsahuje-li žádný vzorek, skončí program s hlášením `"Invalid WAV file!\n"` a návratovou hodnotou 11.

Vzorkovací frekvence	44 100 Hz
Počet kanálů	1 (mono)
Formát vzorku	PCM, 16 bitů se znaménkem
Vzorků na symbol	441, tj. modulační rychlost 100 Bd
Tón nulového bitu	2 200 Hz (22 period na symbol)
Tón jedničkového bitu	1 200 Hz (12 period na symbol)
Amplituda tónů	16 383 (polovina maxima 32 767)
Ticho před rámcem a za ním	4 410 vzorků, tj. 100 ms

Tabulka 1: Parametry nahrávky a modulace.

Vzorky symbolu jsou dány vztahem

$$x_n = A \sin\left(\frac{2\pi f n}{44\,100}\right),$$

kde $n = 0, 1, \dots, 440$ je pořadí vzorku v symbolu, f frekvence tónu a $A = 16\,383$ amplituda.

Struktura rámce

Data se přenášejí v jediném rámci, jehož skladba je naznačena na obrázku 2. Bity se do nahrávky ukládají v pořadí, v jakém jsou zde popsány, vždy od nejvýznamnějšího bitu.

Preamble	32 bitů, ve kterých se pravidelně střídají nuly a jedničky, počínaje nulou. Slouží přijímači k nalezení hranic symbolů.
Sync	Synchronizační slovo <code>0x2DD4</code> o délce 16 bitů, které odděluje preambuli od zabezpečené části rámce.
Délka	Objem přenášených dat v bajtech, uložený jako čtyřbajtové celé číslo v pořadí big-endian.
Data	Vlastní přenášená data o velikosti 1 až 4 096 bajtů (viz přepínač <code>-e</code>).
CRC-32	Kontrolní součet přenášených dat (bez pole délky), uložený jako čtyřbajtové celé číslo v pořadí big-endian. Používá se běžná varianta CRC-32, stejná jako ve formátech ZIP nebo PNG [3]. Pro kontrolu: kontrolní součet řetězce <code>123456789</code> je <code>0xCBf43926</code> .

Preamble a synchronizační slovo se přenášejí tak, jak jsou. Zbytek rámce, tj. délka, data a kontrolní součet, se zabezpečí a proloží postupem popsaným dále. Prokládá se po 4B blocích, proto se tato část rámce na konci doplní nulovými bajty na celý násobek čtyř bajtů. Doplněné bajty se do kontrolního součtu nezapočítávají a přijímač je ignoruje.

Zabezpečení a proložení

Každá čtveřice bitů se zabezpečí Hammingovým kódem (7,4) [4], z každého bajtu nejprve horní a poté dolní čtveřice. Ze čtveřice tak vznikne sedmibitové kódové slovo ve tvaru $p_1 p_2 d_1 p_3 d_2 d_3 d_4$, kde d_1 až d_4 jsou datové bity od nejvýznamnějšího a paritní bity se počítají jako

$$p_1 = d_1 \oplus d_2 \oplus d_4, \quad p_2 = d_1 \oplus d_3 \oplus d_4, \quad p_3 = d_2 \oplus d_3 \oplus d_4.$$

Takový kód dokáže opravit jeden chybný bit v každé sedmici. Sám o sobě je ovšem proti skutečnému rušení skoro k ničemu: cvaknutí na lince nezničí jeden osamocený bit, ale několik bitů za sebou, které padnou do jednoho či dvou sousedních kódových slov.

Proto se kódová slova ještě **proloží** [5]. Vždy osm po sobě jdoucích slov tvoří blok, ze kterého se nejprve vyšlou bity p_1 všech osmi slov v jejich pořadí, pak bity p_2 a tak dále až po bity d_4 . Blok tedy nese 4 bajty dat a zabere 56 přenášených bitů.

Příjem

Nalezení rámce v nahrávce je na celé úloze nejzajímavější část a její řešení je zcela na vás. Uvědomte si, že přijímač předem nezná ani přesnou pozici, na které rámec začíná, ani hlasitost signálu, a že šum v nahrávce může být silnější než samotný signál. Ticho před rámcem a za ním z tabulky 1 musí dodržet váš vysílač, přijímač se na ně ale spoléhat nesmí: v přijímané nahrávce může rámec začínat na libovolném vzorku a nahrávka může končit hned za ním.

Program musí obstát v následujících situacích, které validační skript ověřuje:

1. nahrávka obsahuje rámec bez jakéhokoli rušení,
2. k signálu je přimíchán bílý šum až do odstupů -10 dB, tj. výkon šumu je desetkrát větší než výkon užitečného signálu (ten se počítá jen z části nahrávky obsahující tóny); aby se vzorky vešly do 16bitového rozsahu a neořízly se, validační skript nahrávku po přimíchání šumu celou ztlumí,
3. v zabezpečené části rámce je souvislý výpadek signálu v délce nejvýše 70 ms,
4. hlasitost nahrávky je zeslabena na dvacetinu nebo je k signálu přičtena stejnosměrná složka,
5. nahrávka neobsahuje žádný rámec (např. jen ticho, šum nebo samotnou preambuli).

Pokud program v nahrávce žádný rámec nenajde, vypíše hlášení "No signal detected!\n" a skončí s návratovou hodnotou 20. Najde-li rámec, ale obnovená data neodpovídají kontrolnímu součtu, vypíše hlášení "Corrupted data!\n" a skončí s návratovou hodnotou 21. Stejně program reaguje i na rámec, jehož obnovená délka neleží v rozsahu 1 až 4 096 bajtů nebo jehož konec se do nahrávky nevejde. V obou případech program výstupní soubor vůbec neotevře.

Užitečné techniky a odkazy

- [1] **Knihovna dr_wav** (čtení a zápis souborů WAV)
https://github.com/mackron/dr_libs
- [2] **Goertzelův algoritmus** (výpočet jediné složky Fourierovy transformace)
https://en.wikipedia.org/wiki/Goertzel_algorithm
- [3] **Cyklický redundantní součet CRC-32** (kontrolní součet přenášených dat)
https://en.wikipedia.org/wiki/Cyclic_redundancy_check

- [4] **Hammingův kód (7,4)**
[https://en.wikipedia.org/wiki/Hamming\(7,4\)](https://en.wikipedia.org/wiki/Hamming(7,4))
- [5] **Prokládání** (rozprostření shluku chyb)
https://en.wikipedia.org/wiki/Burst_error-correcting_code
- [6] **Klíčování frekvenčním posuvem**
https://en.wikipedia.org/wiki/Frequency-shift_keying
- [7] **Modem Bell 202**
https://en.wikipedia.org/wiki/Bell_202_modem
- [8] **Modem Bell 202T pro pronajaté linky** (příklad nasazení v energetice)
<https://patton.com/raytel/telenetics-miu202t>
- [9] **Protokol HART** (klíčování Bell 202 nad proudovou smyčkou 4–20 mA)
<https://www.fieldcommgroup.org/technologies/hart/hart-technology-explained>
- [10] **Program Dire Wolf** (softwarový modem pro APRS)
<https://github.com/wb2osz/direwolf>
- [11] **Program Audacity** (zobrazení spektrogramu, generování šumu)
<https://www.audacityteam.org>

K zadání jsou přiloženy ukázkové nahrávky: čistá, zašuměná (odstup -10 dB), zašuměná s výpadkem signálu a dvě nahrávky bez rámce (jen šum a samotná preambule se šumem), spolu se souborem s přenášenou zprávou. Můžete na nich ladit přijímač nezávisle na stavu vlastního vysílače. Výsledek vlastního vysílače si můžete prohlédnout ve spektrogramu programu Audacity [11], ve kterém lze k nahrávce také přimíchat šum.

Řešení úlohy je zcela ve vaší kompetenci – uvedené dokumenty je možné využít při řešení úlohy, ale můžete zvolit libovolné algoritmy a techniky, které podle vás nejlépe povedou k cíli.

Až vám program poprvé přenese soubor, zkuste si vygenerovanou nahrávku pustit z reproduktoru jednoho počítače a nahrát ji mikrofonom druhého. Pokud jste demodulátor napsali dobře, data přeletí místností jako pískání a na druhé straně vypadnou nepoškozená. Na kontrolu odevzdané práce to nemá vliv, ale stojí to za to.

Přílohy

Konzolové rozhraní 1: Ukázka sestavení a činnosti programu modem.exe.

```
1 user@machine:~$ cd pc && ls
2 doc src CMakeLists.txt dokumentace.pdf Makefile Makefile.win
3 user@machine:~/pc$ make &>/dev/null && ls
4 build doc src CMakeLists.txt dokumentace.pdf Makefile Makefile.win modem.exe
5 user@machine:~/pc$ ./modem.exe
6 Usage: modem.exe (-e | -d) <input-file> -o <path>
7   -e <input-file>  encode the file into a WAV recording
8   -d <input-file>  decode data from a WAV recording
9   -o <path>        output file
10 user@machine:~/pc$ echo $?
11 3
12 user@machine:~/pc$ wc -c ../message.txt
13 116 ../message.txt
14 user@machine:~/pc$ ./modem.exe -e ../message.txt -o prenos.wav
15 Encoded 116 B into 18.04 s of audio.
16 user@machine:~/pc$ echo $?
17 0
18 user@machine:~/pc$ ./modem.exe -d prenos.wav -o prijato.txt
19 Decoded 116 B from 18.04 s of audio.
20 user@machine:~/pc$ cmp ../message.txt prijato.txt && echo "shoda"
21 shoda
22 user@machine:~/pc$ sox -n -r 44100 -b 16 -c 1 ticho.wav trim 0 5 # 5 s ticha
23 user@machine:~/pc$ ./modem.exe -d ticho.wav -o prijato.txt
24 No signal detected!
25 user@machine:~/pc$ echo $?
26 20
27 user@machine:~/pc$ ./modem.exe -d prenos.wav -o ./out_dir/prijato.txt
28 Invalid output destination!
29 user@machine:~/pc$ echo $?
30 2
31 user@machine:~/pc$ ./modem.exe -d ../message.txt -o prijato.txt
32 Invalid WAV file!
33 user@machine:~/pc$ echo $?
34 11
35 user@machine:~/pc$ touch prazdny.txt && ./modem.exe -e prazdny.txt -o prenos.wav
36 Invalid input size!
37 user@machine:~/pc$ echo $?
38 4
39 user@machine:~/pc$
```