# Model přenosu informace pomocí lineárních bezpečnostních kódů

Jan Sadílek

Bakalářská práce
2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2020/2021

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

| | |
|---|---|
| Jméno a příjmení: | **Jan Sadílek** |
| Osobní číslo: | **A18075** |
| Studijní program: | **B3902 Inženýrská informatika** |
| Studijní obor: | **Softwarové inženýrství** |
| Forma studia: | **Prezenční** |
| Téma práce: | **Model přenosu informace pomocí lineárních bezpečnostních kódů** |
| Téma práce anglicky: | **An Information Transfer Model using Linear Security Codes** |

## Zásady pro vypracování

1. Zpracujte literární rešerši na dané téma.
2. Proveďte analýzu (rešerši) stávajících modelů (nástrojů) ilustrujících přenos informace, kódování informace, zabezpečení informace.
3. Navrhněte a vytvořte interaktivní nástroj (použitelný při výuce) pro simulaci přenosu informace a jejího kódování pomocí vybraných lineárních bezpečnostních kódů.
4. Podrobně popište jednotlivé funkce vytvořeného nástroje, formát vstupů a výstupů.
5. Na vybraném experimentu prezentujte možnosti a využití nástroje.
6. Zhodnoťte přínosy navrženého nástroje.

Forma zpracování bakalářské práce: **Tištěná/elektronická**

Seznam doporučené literatury:


1. JIROUŠEK, Radim. Principy digitální komunikace. Voznice: Leda, 2006. ISBN 80-7335-084-X.
2. VASEGHI, Saeed V. Advanced digital signal processing and noise reduction. 3rd ed. Chichester: John Wiley, c2006, xxvi, 453 s. ISBN 047009494X.
3. FARANA, R. Kapitoly ze základů informatiky. Ostrava, 2003. ISBN 80-248-0265-1.
4. ZELINKA, Ivan a Zdenka PROKOPOVÁ. Základy informatiky. Zlín: Univerzita Tomáše Bati, 2005, 112 s. ISBN 8073182998.
5. BRILLOUIN, L. Science and Information Theory: Second Edition. Dover Publications, 2013, 368 s. ISBN 9780486316413.
6. VLČEK, Karel. Komprese a kódová zabezpečení v multimediálních komunikacích. Praha: BEN – technická literatura, 2000, 225 s. ISBN 8086056686.
7. MACKAY, David J. C. Information theory, inference and learning algorithms. Cambridge: Cambridge University Press, 2003, xii, 628 s. ISBN 0521642981.


Vedoucí bakalářské práce: **doc. Ing. Bc. Bronislav Chramcov, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **15. ledna 2021**
Termín odevzdání bakalářské práce: **17. května 2021**

**doc. Mgr. Milan Adámek, Ph.D.** v.r.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DELAN

**prof. Mgr. Roman Jašek, Ph.D.** v.r.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

**Prohlašuji, že**

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Jan Sadílek, v. r.
podpis studenta

## ABSTRAKT

Tato práce se zabývá lineárními bezpečnostními kódy. V teoretické části je uveden stručný úvod k využití a základní funkcionalitě. Zabývá se jejich kódováním, dekódováním, a především popisu průběhu procesu, který vede k přenosu informace. Dále obsahuje analýzu stávajících nástrojů ilustrujících přenos informace a následně se praktická část zabývá implementací vybraných kódů a modelováním tohoto procesu.



Klíčová slova: bezpečnostní kódy, lineární kódy, kódování, dekódování, Hammingův kód, kontrola parity



## ABSTRACT

This work deals with linear security codes. The theoretical part provides a brief introduction to the use and basic functionality. It deals with their coding, decoding, and especially describing of the process that leads to the transmission of information. It also contains an analysis of existing tools illustrating the transfer of information and then the practical part deals with the implementation of selected codes and modelling the process.



Keywords: security codes, linear codes, encoding, decoding , Hamming’s code, parity check

Děkuji vedoucímu práce panu doc. Ing. Bc. Bronislavu Chramcovovi, Ph.D. za ochotnou pomoc, cenné rady a odborné vedení, které mi poskytl při tvorbě mé práce. Také bych chtěl poděkovat panu Ing. Tomáši Vogeltanzovi za pomoc a rady, nejen při tvorbě této práce, ale i v průběhu studia.

Prohlašuji, že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

# OBSAH

**ÚVOD** ..... **9**
**I TEORETICKÁ ČÁST** ..... **10**
**1 BEZPEČNOSTNÍ KÓDY** ..... **11**
1.1 Úvod ..... 11
1.2 Druhy bezpečnostních kódů ..... 11
1.3 Kód ..... 12
1.4 Kódování ..... 14
1.5 Dekódování ..... 14
1.6 Detekční a korekční vlastnosti ..... 14
**2 LINEÁRNÍ BEZPEČNOSTNÍ KÓDY** ..... **16**
2.1 Kódování ..... 16
2.1.1 Generující matice ..... 16
2.2 Dekódování ..... 17
2.2.1 Kontrolní matice ..... 17
2.2.2 Syndrom ..... 17
2.3 Druhy lineárních bezpečnostních kódů ..... 17
2.3.1 Opakovací kódy ..... 18
2.3.2 Kontrola parity ..... 18
2.3.3 Hammingovy kódy ..... 18
2.3.4 Rozšířený Hammingův kód ..... 21
2.3.5 Cyklické kódy ..... 22
2.3.6 BCH kódy ..... 22
2.3.7 Reed-Solomonovy kódy ..... 22
2.3.8 Golayovy kódy ..... 22
**3 PŘENOS INFORMACE** ..... **23**
3.1 Druhy přenosu ..... 24
**4 LITERÁRNÍ REŠERŠE NÁSTROJŮ ILUSTRUJÍCÍ PŘENOS INFORMACE** ..... **25**
4.1 Online podpora TIK ..... 25
4.2 Bezpečnostní kódy v prostředí Mathematica ..... 25
4.3 Knihovna funkcí pro program Wolfram Mathematica umožňující snadný návrh bezpečnostních kódů ve výuce ..... 26
4.4 Lineární binární bezpečnostní kódy ..... 27
4.5 Cyklické kódy ..... 27
4.6 Kodér a dekodér samoopravného kódu pro programovatelné paměti typu ROM ..... 28
4.7 Program pro demonstraci kanálového kódování ..... 28
4.8 Protichybové zabezpečení v digitálních komunikačních systémech ..... 30
4.9 Shrnutí ..... 32
**5 MATEMATICKÝ ZÁKLAD** ..... **33**

5.1 MATICOVÉ OPERACE ..... 33
5.2 BINÁRNÍ OPERACE ..... 34
5.3 FUNKCE MODULO ..... 35
**II PRAKTICKÁ ČÁST ..... 36**
**6 REALIZACE PRAKTICKÉ ČÁSTI ..... 37**
**7 TECHNOLOGIE POUŽITÉ PRO TVORBU PROGRAMU ..... 38**
7.1 JAZYK C# ..... 38
7.2 JAZYK XAML ..... 38
7.3 WPF ..... 38
7.4 POUŽITÉ BALÍČKY ..... 39
**8 POPIS PROGRAMU ..... 40**
8.1 INTERAKTIVNÍ PROGRAM PRO ILUSTRACI KÓDOVÁNÍ A DEKÓDOVÁNÍ POMOCÍ HAMMINGOVA KÓDU ..... 40
8.1.1 Postup ..... 40
8.1.2 Příklad s žádnou chybou při přenosu ..... 41
8.1.3 Příklad s jednou chybou při přenosu ..... 42
8.1.4 Příklad se třemi chybami ..... 44
8.2 INTERAKTIVNÍ PROGRAM PRO ILUSTRACI KÓDOVÁNÍ A DEKÓDOVÁNÍ POMOCÍ KÓDU KONTROLY PARITY ..... 46
8.2.1 Postup ..... 46
8.2.2 Příklad s žádnou chybou ..... 47
8.2.3 Příklad s jednou chybou ..... 49
8.2.4 Příklad se dvěma chybami ..... 52
**9 ZHODNOCENÍ PŘÍNOSŮ PRÁCE ..... 54**
**ZÁVĚR ..... 55**
**SEZNAM POUŽITÉ LITERATURY ..... 56**
**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK ..... 59**
**SEZNAM OBRÁZKŮ ..... 60**
**SEZNAM TABULEK ..... 61**
**SEZNAM PŘÍLOH ..... 62**

# ÚVOD

V dnešním světě plném moderních technologií hrají prim informace. Každá odeslaná zpráva jako SMS, email, text v chatovací aplikaci či televizní přenos je informace mnohokrát změněná a přetvořená tak, aby bylo možné ji číst, vytvořit, zobrazit, odeslat a přijmout. Tento proces není zdaleka tak jednoduchý, jak se může na první pohled zdát. Člověk potřebuje vidět text, složený z písmen abecedy, jež je specifická pro každý lidský mluvený jazyk a stroj zase potřebuje posloupnost jedniček a nul pro správnou intepretaci dané zprávy. Také je potřeba myslet na samotné přenášení zpráv mezi zařízeními, starající se o příjem a odesílání, pomocí různých fyzikálních dějů. Přenos těchto informací může být kdykoliv v průběhu toho děje narušen a tím dojde ke ztrátě nebo změně hodnot dané přenosové soustavy, který zapříčiní, že je zpráva změněna. Z těchto důvodů potřebujeme tento přenos zabezpečit, abychom co nejvíce omezili dopady dějů narušujících kanál, jímž se zpráva přenáší. Proto do tohoto děje vstupují zabezpečovací techniky, které mají za úkol zajistit, aby odeslaná zpráva odpovídala zprávě přijaté.

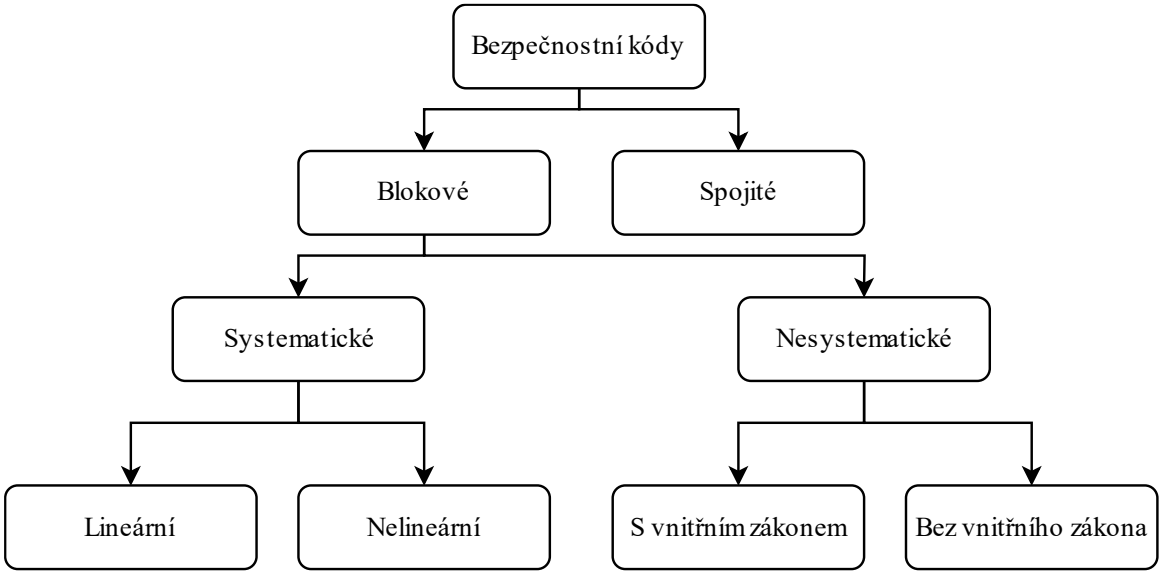
Cílem této práce je vytvoření nástroje, který by našel využití při výuce týkající se teorie přenosu informace. Prakticky jde o poskytnutí lepší podpory vizualizace celého průběhu přenosu informace od kódování přes její přenos, zabezpečení, následného dekódování, odhalení chyb a případně korekce chyb, ke kterým mohlo v průběhu dojít v důsledku působení šumu a jiných vnějších vlivů.

# I. TEORETICKÁ ČÁST

# 1 BEZPEČNOSTNÍ KÓDY

## 1.1 Úvod

Problematika lineárních bezpečnostních kódů se zabývá zabezpečením přenosu informace při průchodu komunikačním kanálem. Zabezpečuje se z důvodů přítomnosti šumu v kanálu, který zapříčiňuje nechtěnou změnu hodnot, například napětí při převádění bitových hodnot na elektrické signály, kterými se přenáší vysílaná informace nebo při samotném průchodu komunikačním kanálem. Na straně příjemce dochází k dekódování dohodnutým způsobem, přičemž se chyba může vyskytnout i zde v důsledku špatné synchronizace či jinak nastavených parametrů kodéru a dekodéru.




Obrázek 1. Rozdělení bezpečnostních kódů

## 1.2 Druhy bezpečnostních kódů

Na obrázku (Obrázek 1.) lze vidět rozdělení bezpečnostních kódů sestávající ze dvou kategorií, jež se dále dají dělit do několika podkategorií vyznačující se specifickými vlastnostmi.

Blokové kódy jsou takové prosté kódy, jež mají pevně danou délku všech kódových slov. Všechny blokové kódy spadají do prefixových kódů. „Prefix slova $b_1b_2b_3...b_k$ je každé ze slov $b_1$, $b_1b_2$, $b_1b_2b_3$, $b_1b_2b_3...$, $b_1b_2b_3...b_k$.“ [1]

Jako prefixový můžeme označit takový kód, jenž je prostý a každé možné zakódované slovo se nedá určit jako prefix jiného zakódovaného slova. Díky této vlastnosti je možné dekódovat přijaté posloupnosti blok po bloku a jejich dekódování se dá jednoznačně určit.

Systematické kódy jsou takové kódy, u nichž známe přesnou polohu informačních a zabezpečujících částí.

Nesystematické kódy jsou takové kódy, u nichž nelze jednoznačně říci, kde se nachází informační a zabezpečující část.

Lineární kódy jsou kódy, kdy například sčítáním dvou kódových slov vznikne jiné kódové slovo.

Nelineární kódy jsou odvozeny z kódovacích tabulek. Nemají vlastnosti lineárních kódů.

U kódů s vnitřním zákonem probíhá kontrola splnění zadaných podmínek pro daný kód.

U kódů bez vnitřního zákona jsou ověřovány postupnou kontrolou všech předepsaných možných hodnot, jež jsou sepsány v předem dohodnuté tabulce. Vyznačují se nesnadnou realizací.

Spojité kódy mají vloženu zabezpečovací část spojitě, což znamená, že informační část nelze na první pohled odlišit od části zabezpečovací.

## 1.3 Kód

Kód je definován jako množina symbolů, kterým přiřazujeme různé stavy systému. Uveďme si oblíbený příklad s hrací kostkou. Na každé straně kostky je daný počet teček, který zastupuje čísla od 1 do 6 dekadického systému čísel.

Čárový kód je také druh kódu, který závisí na různém uspořádání tmavých pruhů a světlých mezer. Důležitá je i šířka pruhů a mezer. Princip spočívá v odrazu červeného světla, jež je vyzařováno a odráženo zpět do snímače. Odražené paprsky jsou změněny na elektrické signály odpovídající přiřazeným číslům a případně doplňujících písmen. Každá sekvence začíná start znakem a končí stop znakem, mezi těmito znaky můžeme mít například 5 různě širokých tmavých pruhů, toto záleží na zvolené specifikaci zvoleného kódu.

ASCII kód je kód, kdy je symbolu, například písmenu abecedy, přiřazeno číslo dekadické soustavy čísel, které se také dají vyjádřit v binární, hexadecimální či osmičkové soustavě. V první části tabulky do znaku 31 včetně se nachází řídící znaky. Další část tabulky do znaku 127 je univerzální pro jazyky používající latinku. Od znaku 128 do znaku 255 včetně jsou

definovány znaky specifické pro jednotlivé jazyky. V obrázku níže (Obrázek 2.) je uvedena tabulka ASCII využívající UTF-8 standard. UTF-8 znamená Universal Character Set Transformation Format. Jedná se tedy o univerzální kódovací tabulku znaků převádějící znaky na osmi bitové pole, což je jeden bajt.

| kód | znak | kód | znak | kód | znak | kód | znak | kód | znak | kód | znak | kód | znak | kód | znak | kód | znak |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 32 | | 60 | < | 88 | X | 116 | t | 144 | □ | 172 | ¬ | 200 | Č | 228 | ä |
| | | 33 | ! | 61 | = | 89 | Y | 117 | u | 145 | ‘ | 173 | | 201 | É | 229 | Í |
| | | 34 | " | 62 | > | 90 | Z | 118 | v | 146 | ’ | 174 | ® | 202 | Ę | 230 | ć |
| | | 35 | # | 63 | ? | 91 | [ | 119 | w | 147 | “ | 175 | Ż | 203 | Ë | 231 | ç |
| | | 36 | $ | 64 | @ | 92 | \ | 120 | x | 148 | ” | 176 | ° | 204 | Ě | 232 | č |
| | | 37 | % | 65 | A | 93 | ] | 121 | y | 149 | • | 177 | ± | 205 | Í | 233 | é |
| | | 38 | & | 66 | B | 94 | ^ | 122 | z | 150 | – | 178 | ˛ | 206 | Î | 234 | ę |
| 7 | BEL | 39 | ' | 67 | C | 95 | _ | 123 | { | 151 | — | 179 | ł | 207 | Ď | 235 | ë |
| 8 | BS | 40 | ( | 68 | D | 96 | ` | 124 | \| | 152 | □ | 180 | ´ | 208 | Đ | 236 | ě |
| 9 | TAB | 41 | ) | 69 | E | 97 | a | 125 | } | 153 | ™ | 181 | µ | 209 | Ń | 237 | í |
| 10 | LF | 42 | * | 70 | F | 98 | b | 126 | ~ | 154 | š | 182 | ¶ | 210 | Ň | 238 | î |
| | | 43 | + | 71 | G | 99 | c | 127 | | 155 | › | 183 | · | 211 | Ó | 239 | ď |
| 12 | FF | 44 | , | 72 | H | 100 | d | 128 | € | 156 | ś | 184 | ¸ | 212 | Ô | 240 | đ |
| 13 | CR | 45 | - | 73 | I | 101 | e | 129 | □ | 157 | ť | 185 | ą | 213 | Ő | 241 | ń |
| | | 46 | . | 74 | J | 102 | f | 130 | , | 158 | ž | 186 | ş | 214 | Ö | 242 | ň |
| | | 47 | / | 75 | K | 103 | g | 131 | □ | 159 | ź | 187 | » | 215 | × | 243 | ó |
| | | 48 | 0 | 76 | L | 104 | h | 132 | „ | 160 | | 188 | Ľ | 216 | Ř | 244 | ô |
| | | 49 | 1 | 77 | M | 105 | i | 133 | … | 161 | ˇ | 189 | ˝ | 217 | Ů | 245 | ő |
| | | 50 | 2 | 78 | N | 106 | j | 134 | † | 162 | ˘ | 190 | ľ | 218 | Ú | 246 | ö |
| | | 51 | 3 | 79 | O | 107 | k | 135 | ‡ | 163 | Ł | 191 | ż | 219 | Ű | 247 | ÷ |
| | | 52 | 4 | 80 | P | 108 | l | 136 | □ | 164 | ¤ | 192 | Ŕ | 220 | Ü | 248 | ř |
| | | 53 | 5 | 81 | Q | 109 | m | 137 | ‰ | 165 | Ą | 193 | Á | 221 | Ý | 249 | ů |
| | | 54 | 6 | 82 | R | 110 | n | 138 | Š | 166 | ¦ | 194 | Â | 222 | Ţ | 250 | ú |
| | | 55 | 7 | 83 | S | 111 | o | 139 | ‹ | 167 | § | 195 | Ă | 223 | ß | 251 | ű |
| | | 56 | 8 | 84 | T | 112 | p | 140 | Ś | 168 | ¨ | 196 | Ä | 224 | ŕ | 252 | ü |
| | | 57 | 9 | 85 | U | 113 | q | 141 | Ť | 169 | © | 197 | Ĺ | 225 | á | 253 | ý |
| | | 58 | : | 86 | V | 114 | r | 142 | Ž | 170 | Ş | 198 | Ć | 226 | â | 254 | ţ |
| | | 59 | ; | 87 | W | 115 | s | 143 | Ź | 171 | « | 199 | Ç | 227 | ă | 255 | ˙ |

Obrázek 2. ASCII tabulka [1]

Převádění kódů na jiné kódy se děje s pomocí tabulek nebo algoritmů. Je důležité podotknout, že převedením se nezmění velikost ani množství informace z původního kódu. Nedochází, respektive by nemělo docházet ke ztrátě žádných parametrů, jež určují podobu výstupu.

Příkladem takového kódování může být převod příslušného znaku podle uvedené ASCII tabulky (Obrázek 2.). Pro ukázku si vezmeme znak $L$, jenž můžeme zaměnit za jeho číselnou reprezentaci v dekadické, binární, hexadecimální nebo oktalové (osmičkové) soustavě. Všechny tyto zápisy (1) jsou ekvivalentní vyjádření daného znaku.

$$L = [76]_{10} = [01001100]_2 = [4C]_{16} = [114]_8 \quad (1)$$

## 1.4 Kódování

Abeceda sloužící ke kódování musí být vždy konečná, počet znaků musí být jasně stanoven a každý jednotlivý znak jednoznačně přiřazen. [2]

Kódování je předpis, který každému prvku konečné množiny $A$ přiřazuje právě jedno slovo v konečné množině $B$. Je to tedy zobrazení [1]

$$K: A \rightarrow B^* \tag{2}$$

Množina $A$ obsahuje zdrojové znaky a množina $B$ obsahuje kódové znaky.

Hammingova vzdálenost udává počet znaků, ve kterých se dvě zakódovaná slova liší. Značí se písmenem $d$.

Minimální Hammingova vzdálenost $d_{min}$ je určena výpočtem Hammingovy vzdálenosti mezi všemi dvojicemi různých kódových prvků. S rostoucí minimální Hammingovou vzdáleností se zvyšuje jeho kapacita pro zabezpečení, ovšem úměrně tomu roste i častější opakování prvků tedy redundance, se kterou musíme počítat, pokud chceme mít dostatečně zabezpečený přenos. Jde tedy o nalezení rovnováhy mezi silným zabezpečením a nejmenší možnou redundancí daného kódu.

Hammingova váha se označuje písmenem $w$ a popisuje počet nenulových prvků v zakódovaném slově. Nejvyšší uvažovanou váhu má bit nacházející se nejvíce nalevo.

## 1.5 Dekódování

Dekódování není opačný postup ke kódování. Principy dekódování se u bezpečnostních kódů liší, protože se takto zabezpečená kódová slova skládají z informačních a zabezpečovacích bitů. Pokud se jedná o detekční kód, tak se na konci procesu dozvíme nejen přijatou zprávu nebo znak, ale zároveň informaci o tom, jestli se při přenosu vyskytla chyba. V případě korekčního kódu se dozvíme, jestli se vyskytla jedna nebo i více chyb a opět záleží na druhu kódu, jestli je tyto chyby schopen opravit.

## 1.6 Detekční a korekční vlastnosti

Podle obrázku (Obrázek 3.) můžeme vidět, že pro tříbitový kód existuje v nezabezpečené variantě osm možných kombinací. Pro detekční kód již máme 4 možné kombinace a nakonec korekční kód má pouze 2 možné kombinace. Je zde také patrné, že Hammingova vzdálenost je reprezentována jako hrana krychle, černě vyznačené vrcholy značí jednotlivé možné

kombinace a minimální Hammingova vzdálenost je nejmenší počet hran vedoucí od jedné libovolné bitové kombinaci ke druhé.

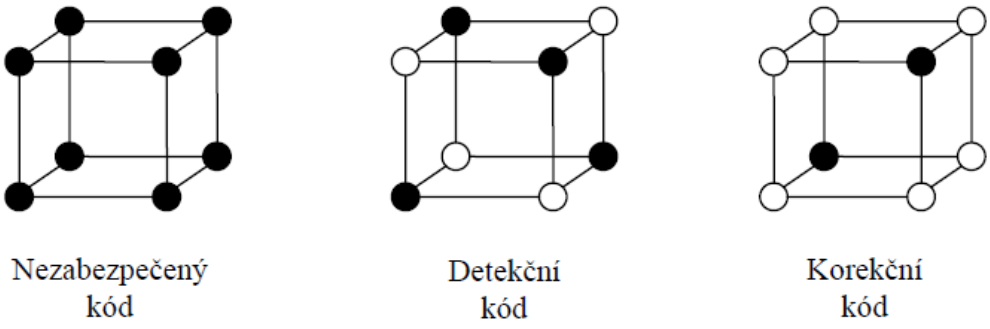
Tyto vlastnosti vyplývají z následující vzorců:

Detekční kód objeví α-násobné chyby

$$d \geq \alpha + 1 \tag{3}$$

Korekční kód opraví β-násobné chyby:

$$d \geq 2 * \beta + 1 \tag{4}$$




Obrázek 3. Ilustrace 3 bitového kódu [3]

# 2 LINEÁRNÍ BEZPEČNOSTNÍ KÓDY

Lineární kódy jsou definovány jako kódy sloužící k detekci a opravě chyb. K tomuto procesu dochází na straně příjemce zakódované zprávy. Kombinací dvou kódových slov vznikne další kódové slovo. Každé kódové slovo sestává z informačních bitů *k*, zabezpečujících bitů *m* a celkový počet bitů je značen písmenem *n*. Každý lineární kód může mít jiný počet celkových bitů a poměr mezi informačními a zabezpečujícími bity se také mění v závislosti na typu lineárního kódu.

## 2.1 Kódování

Při tvorbě lineárních bezpečnostních kódu je potřeba vytvořit sestavu pravidel, podle kterých se bude daný kód řídit. Jedním z těchto způsobů může být vytvoření generující matice kódu od níž se odvíjí, jak se bude kódování i dekódování vyvíjet.

### 2.1.1 Generující matice

Pro tvorbu kódování u lineárních kódů je důležitá generující matice, jež určuje podobu vytvářeného kódu.

„Lineární (*n*, *k*) - kód má celkem $q^k$ kódových slov, protože kódová slova můžeme vyjádřit jejich *k* souřadnicemi zvolené bázi a každá souřadnice má *q* možných hodnot. Kód je *q* znakový, jestliže abeceda *T* (těleso) má počet znaků *q*.“ [1]

Báze kódu *B* je libovolné slovo daného kódu. Sepíšeme-li *k* bází pod sebe, pak dostaneme generující matici

$$G = (b_1 b_2 \dots b_k)^T \tag{5}$$

Nebo může být vyjádřena jako

$$G = [E_k|B] \tag{6}$$

O generující matici tedy hovoříme, pokud splňuje tři základní kritéria: [1]

- Každý její řádek je kódovým slovem.
- Každé kódové slovo je lineární kombinací řádků
- Řádky jsou lineárně nezávislé, takže hodnost matice *G* je rovna *k*.

Zabezpečené kódové slovo vznikne, když vynásobíme nezabezpečené kódové slovo generující maticí *G*. Toto slovo je pak přeneseno kanálem, kde se vlivem šumu mohou objevit chyby. Takové chyby se detekují a případně opravují při dekódování.

## 2.2 Dekódování

Při procesu dekódování je u lineárních kódů využita kontrolní matice, jenž vzniká z báze kódu a jednotkové matice. Využívá se zde syndromu zabezpečeného kódového přijatého slova. Jedná se o vektor hodnot indikující počet a pozici chyb, takže můžeme říci, že jde o dekódování odstraňující chyby.

### 2.2.1 Kontrolní matice

Kontrolní matice je klíčová pro dekódování lineárních kódů. Po vynásobení přijatého zabezpečeného kódového slova dostaneme syndrom, pomocí nějž určujeme počet a pozice chyb.

Matematicky je vyjádřena

$$H = [-B^T | E'_{n-k}] \tag{7}$$

$B^T$ je transponovaná báze kódu $B$ a $E'_{n-k}$ je jednotková matice.

### 2.2.2 Syndrom

Tento vektor vznikne při násobení matice s řádkovým vektorem přijatého zabezpečeného kódového slova transponovaným do vektoru sloupcového. Tento vektor přichází ke slovu u kontroly správnosti přenosu. Pokud po vynásobení zabezpečeného kódového slova dostaneme nenulový syndrom, pak zjistíme, kterému indexu sloupce kontrolní matice tento vektor odpovídá, což udává, na kterém indexu se vyskytuje chyba v přijatém slově.

Například syndrom odpovídá šestému sloupci kontrolní matice, pak se chyba nachází na pozici šestého bitu v kontrolovaném přijatém zakódovaném slově.

## 2.3 Druhy lineárních bezpečnostních kódů

Mezi lineární bezpečnostní kódy patří opakovací kódy, Hammingovy kódy, Reed-Solomonovy kódy, paritní kódy, Reed-Mullerovy kódy, cyklické kódy, polynomiální kódy, Gopa kódy, Golayův kód, torické kódy, expander kódy, turbo kódy kódy, kontroly parity s nízkou hustotou a vícerozměrné kódy kontroly parity. Více informací o vybraných kódech je uvedeno dále v této kapitole.

### 2.3.1 Opakovací kódy

Korekční kódy s daným počtem opakováním jednoho prvku n-krát. Nejméně efektivní kód z hlediska nejvyššího počtu nutných zabezpečovacích bitů.

### 2.3.2 Kontrola parity

Nejjednodušší a nejvyužívanější typ detekčního lineárního kódu. Dělí se na paritu sudou a lichou. Princip spočívá ve spočítání jednotkových bitů, čímž ve své podstatě vypočteme Hammingovu váhu slova, a přidáním jednotkového nebo nulového bitu, aby byla dle typu parity zachována příslušná hodnota funkce modulo ze součtu prvků. Pro lichou paritu tedy chceme mít hodnotu funkce modulo rovnu jedné a pro sudou paritu bude hodnota funkce modulo rovna nule. Doplníme tedy jednotkový nebo nulový bit, aby se zbytek po dělení nezměnil. Minimální Hammingova vzdálenost pro tento kód $d_{min}$ je rovna 2 podle vzorce (3). Z toho vyplývá, že kontrola parity dokáže efektivně detekovat pouze jednu chybu (8).

$$\alpha \leq 2 - 1 = 1 \tag{8}$$

Příklad doplnění na sudou paritu

$$(1 \quad 0 \quad 1)\, mod2 = 0 \rightarrow (1 \quad 0 \quad 1 \quad 0) \tag{9}$$

Chceme zachovat modulo 2 rovno nule, tudíž přidáme nulový paritní bit.

Příklad doplnění na lichou paritu

$$(1 \quad 0 \quad 1)\, mod2 = 0 \rightarrow (1 \quad 0 \quad 1 \quad 1) \tag{10}$$

Zde naopak chceme, aby došlo ke změně modula 2 na hodnotu 1, proto přidáme jednotkový paritní bit.

### 2.3.3 Hammingovy kódy

Hammingův kód je takzvaně perfektní, jelikož se vyznačuje nejmenší redundancí kódových znaků. Dále snadnou detekcí chyb a dokáže opravit jednoduché chyby vyskytující se v zakódovaném slově. V praktické části jsou uvedeny ilustrační příklady demonstrující principy Hammingova kódu, kdy počet celkových bitů je sedm a z toho jsou tři zabezpečující bity. Generující matici nesystematického Hammingova kódu lze odvodit z kontrolní matice (11), která má přesně danou strukturu (v daném sloupci matice je binární vyjádření čísla sloupce).

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{11}$$

Pro odvození je nutné tuto matici vyjádřit ve tvaru (7), tedy přemístit sloupce v původní kontrolní matici, kdy první je zaměněn za sedmý, druhý za šestý a čtvrtý za pátý sloupec. Výsledkem je matice ve tvaru (12)

$$H' = \left(\begin{array}{cccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right) \tag{12}$$

Generující matici ve tvaru (13) pak dostaneme odvozením ze vzorce (6).

$$G' = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right) \tag{13}$$

Pro odvození generující matice, která odpovídá původní kontrolní matici je nutné opět přehodit sloupce, kdy první je zaměněn za sedmý, druhý za šestý a čtvrtý za pátý sloupec. Výsledkem je generující matice Hamingova nesystematického kódu ve tvaru (14).

$$G = \left(\begin{array}{cccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right) \tag{14}$$

Jejich minimální Hammingova vzdálenost je $d_{min} = 3$. Z toho tedy podle vzorce (3) vyplývá, že detekční schopnost se rovná číslu 2.

$$\alpha \leq 3 - 1 = 2 \tag{15}$$

Dále podle vzorce (4) lze odvodit, že korekční schopnost je rovna číslu 1.

$$\beta \leq (3 - 1)/2 = 1 \tag{16}$$

Tento typ kódu tedy detekuje až dvojnásobnou chybu nebo dokáže opravit jednu chybu.

U systematických Hammingových kódů jsou zabezpečovací bity umístěny v řadě za sebou až po informačních bitech. Lze tedy rozeznat informační a zabezpečovací část kódu na první pohled.

Nesystematická verze Hammingových kódů spočívá v přidání zabezpečovacích bitů na místa stupňujících se mocnitelů čísla 2. Začínáme tedy od 1 a postupujeme přes 2 ke 4 a případně dále u delších variant Hammingových kódů.

Pro výpočet počtu jednotlivých druhů bitů v kódovém slově daného kódu jsou udávány vzorce (vycházíme z počtu zabezpečujících bitů)

$$n = 2^m - 1 \tag{17}$$

$$k = n - m \tag{18}$$

Tabulka 1. Kombinace Hammingových kódů

| **Typ Hammingova kódu** | **Zabezpečující bity (m)** | **Počet bitů celkem (n)** | **Informační bity (k)** |
|---|---|---|---|
| K(3, 1) | 2 | 3 | 1 |
| K(7, 4) | 3 | 7 | 4 |
| K(15, 11) | 4 | 15 | 11 |
| K(31, 26) | 5 | 31 | 26 |
| K(63, 57) | 6 | 63 | 57 |

**Ukázka principu kódování a dekódování pomocí nesystematického Hammingova kódu K(7,4)**

Mějme generující matici G ve tvaru (19). Dále uvažujme vektor nezabezpečeného kódového slova b ve tvaru (20) a kontrolní matici H ve tvaru (21).

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{19}$$

$$b = (1 \quad 1 \quad 0 \quad 1) \tag{20}$$

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{21}$$

Nyní chceme příslušné nezabezpečené kódové slovo zabezpečit. Toto provedeme (22) vynásobením vektoru nezabezpečeného kódového slova *b* zprava maticí *G* a dostaneme vektor zabezepečeného kódového slova *c* ve tvaru (23)

$$c = (1 \quad 1 \quad 0 \quad 1) * \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{22}$$

$$c = (1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1) \tag{23}$$

Při přenosu informace se často vyskytují chyby, které lze v některých případech detekovat popřípadě opravit. Pro názornou ukázku detekčních a korekčních vlastností Hammingova kódu nyní zaneseme do přenášeného zabezpečeného kódového slova chybu na šestou pozici, tuto skutečnost demonstruje změna šestého bitu ve vektoru přijatého kódového slova (24).

$$c = (1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1) \tag{24}$$

Toto přijaté slovo převedeme do sloupcového vektoru, pomocí kontrolní matice vynásobíme zleva, výsledkem součinu je syndrom slova ve tvaru (25)

$$s = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \tag{25}$$

Vzniklý syndrom odpovídá šestému sloupci kontrolní matice, proto tedy víme, že je potřeba opravit bit na šesté pozici v přijatém slově

$$c = (1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1) \tag{26}$$

### 2.3.4 Rozšířený Hammingův kód

Spočívá v přidání příslušné hodnoty bitu k doplnění na sudou paritu (sudý počet jedniček) za poslední sloupec generující matice kódu. U kontrolní matice je přidán jednotkový řádek za poslední řádek kontrolní matice kódu a v každém sloupci je doplněn sudý paritní bit. Jedná se tedy o spojení parity a klasického Hammingova kódu. Detekční schopnost tohoto kódu je tedy zvýšena o hodnotu 1. Generující a kontrolní matice můžou tedy vypadat takto

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \tag{27}$$

$$H = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{28}$$

### 2.3.5 Cyklické kódy

Jednoduchá implementace po stránce technické i matematické zapříčinila, že cyklické kódy jsou velmi efektivní a velice využívané. Pokud cyklicky zaměníme prvky v kódovém slově, tak opět vznikne kódové slovo, jež je platné.

Cyklický kód vzniká z generujícího polynomu, což je jediné slovo vytvářející celý kód. Z čehož plyne, že kódová slova vzniknou jako násobek tohoto generujícího polynomu. Více informací k cyklickým kódům lze najít zde [4] a [5].

### 2.3.6 BCH kódy

Jedná se o podmnožinu cyklických kódů, nejedná se ovšem o binární kódy, kdy matematický základ pro tyto kódy sahá až k teorii těles. Jejich výhodou je kontrola opravovaných chyb při procesu dekódování, kdy je využito principu dekódování pomocí syndromu. Dokáží opravit dvoj a vícenásobné chyby. Více informací k BCH kódům lze nalézt zde [6].

### 2.3.7 Reed-Solomonovy kódy

Vylepšená verze BCH kódů, kdy je zvýšena jejich Hammingova vzdálenost, čímž je snížena redundance a zvyšuje se efektivita kódu. Více informací k Reed-Solomovým kódům je uvedeno například zde [7].

### 2.3.8 Golayovy kódy

Patří mezi perfektní kódy. Nejvyužívanější varianty jsou $G_{23}$ a $G_{24}$, kdy je ke konstrukci opět využito generující matice $G$. Nejlepším případem pro tento kód je oprava trojnásobných chyb, kdy je tento kód nejúčinnější. Více informací ke Golayovým kódům lze najít zde [8].

# 3 PŘENOS INFORMACE

Samotný přenos kanálem je realizován pomocí signálu (Obrázek 4.). Signál je fyzikální veličina, jež je nositelem vysílané informace.

„Signálem mohou být tlakové vlny vzduchu, hloubka povrchu, tělesná teplota, proud/napětí ve vodiči nebo biologickém systému, světlo, elektromagnetické rádiové vlny nebo objem a hmotnost tělesa. Signál přenáší informaci z hlediska jednoho nebo více atributů zdroje jako je stav, charakteristika, rozložení, trajektorie, vývoj nebo záměr zdroje. Signál je tedy médium, které v různých podobách přenáší informaci s ohledem na minulé, stávající nebo budoucí stavy proměnné[1].“ [9]

Mezi typické zdroje rušivých signálů patří akustické rušení, rušení elektronických zařízení, elektromagnetické rušení, elektrostatické rušení a kanálové zkreslení.

Akustické rušení vzniká z běžných zdrojů zvuku jako je pohyb, vibrace a jiné, jež vzniká při běžných činnostech. Může se jednat i o ozvěnu v uzavřené místnosti nebo u spojení dvou zařízení přenášející zvuk jako je mikrofon.

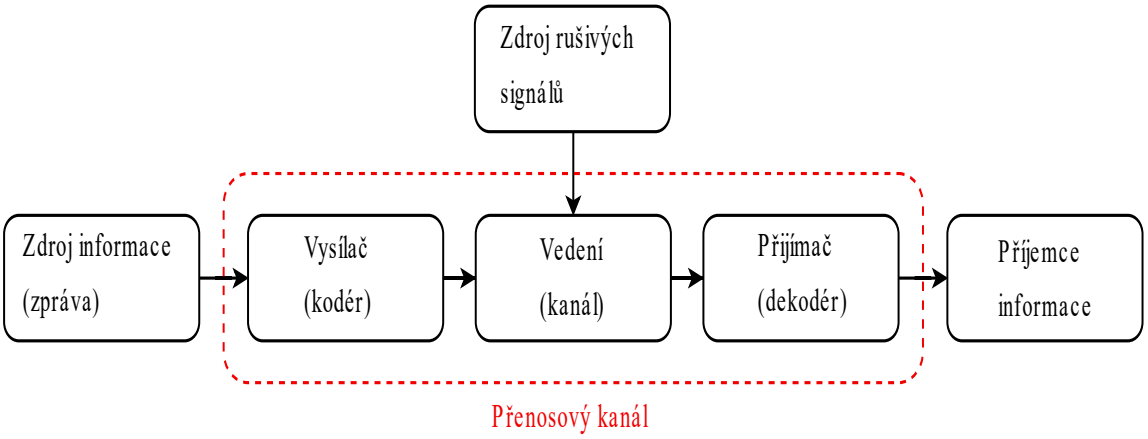
Elektronická zařízení generují celou řadu možných druhů šumu jako je tepelný šum, elektrický šum, statický šum a impulsní šum.

Elektromagnetické rušení je přítomné ve všech používaných frekvenčních pásmech vznikající z umělých a přírodních zdrojů.

Elektrostatický šum se objevuje, když je přítomno napětí.

Kanálové zkreslení je ovlivněno celou řadou parametrů jako je více cest v kanálu, ozvěna a slábnutí signálu.

[1] A signal is the variation of a quantity such as air pressure waves of sounds, colours of an image, depths of a surface, temperature of a body, current/voltage in a conductor or biological system, light, electromagnetic radio waves, commodity prices or volume and mass of an object. A signal conveys information regarding one or more attributes of the source such as the state, the characteristics, the composition, the trajectory, the evolution or the intention of the source. Hence, a signal is a *means of conveying information* regarding the past, the current or the future states of a variable.




Obrázek 4. Obecný komunikační systém

## 3.1 Druhy přenosu

Z hlediska technického řešení přenosu vysílané zprávy jsou k dispozici různé druhy přenosu, které dle aktuální potřeby zajišťují nejlepší řešení pro daný systém.

Sériový přenos se chová tak, že vysílané prvky posloupnosti signálů jsou přenášeny postupně za sebou v jednom přenosovém kanálu.

Při paralelním přenosu je nutný samostatný kanál pro každý jednotlivý signál. Tento typ se hodí pro kratší vzdálenosti z důvodu velké technické náročnosti.

Asynchronní přenos probíhá tak, že do přenášené sekvence signálů jsou přidány příslušné synchronizační oddělovací prvky, které nemění přenášenou zprávu, pouze přidávají prvek mezi přenášené sekvence.

Arytmický přenos je kombinace asynchronního přenosu sekvence signálů nesoucí zprávu, které jsou synchronně doplněny o značku indikující začátek a konec jednotlivých signálů.

U synchronního přenosu jsou signály přenášející zprávu a oddělovací signály jsou přenášeny synchronně, poskládány do bloku a odděleny sledem synchronizačních signálů bez informační hodnoty.

Simplexní přenos znamená, že jde o jednosměrné přenášení zpráv jediným kanálem.

Při poloduplexním přenosu dochází ke střídání mezi vysíláním a přijímáním zpráv jediným kanálem.

Při duplexním přenosu dochází k obousměrnému zasílání zpráv, uskutečňuje se prostřednictvím dvou oddělených kanálů.

# 4 LITERÁRNÍ REŠERŠE NÁSTROJŮ ILUSTRUJÍCÍ PŘENOS INFORMACE

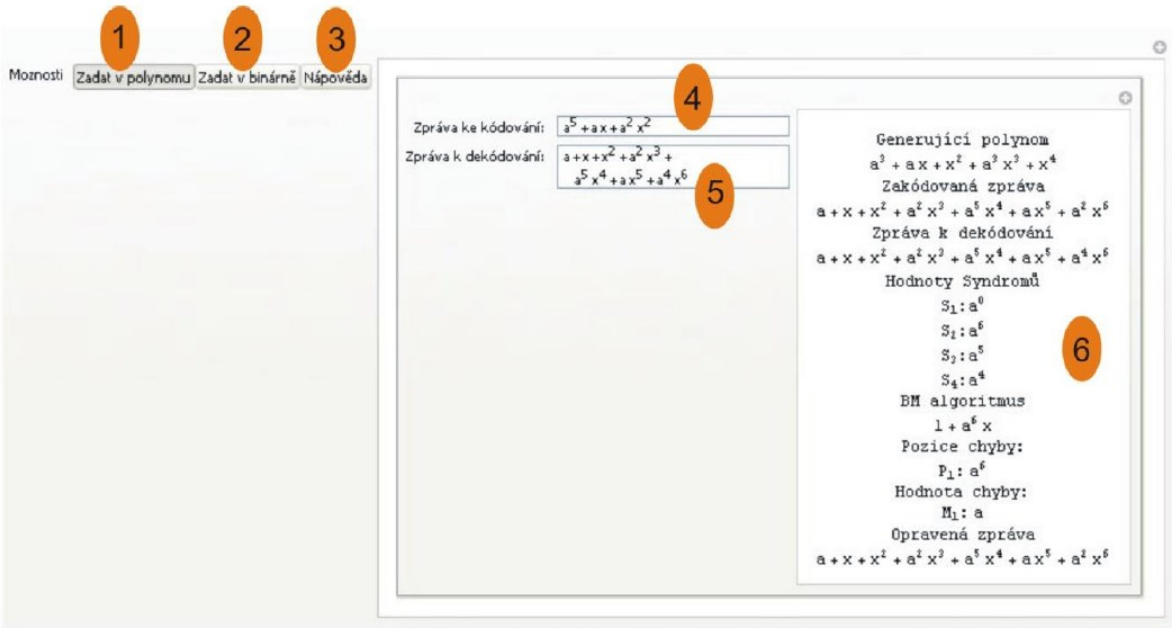
V průběhu let vznikaly na toto téma mnohé práce zabývající se problematikou bezpečnostních kódů pro přenos informace. Pečlivou analýzou těchto prací dostaneme jasný přehled o tom, jaká je situace na poli interaktivní podpory výuky. Rešerše se zaměřuje na přínosy těchto prací, jejich aktuální dostupnost, samotný obsah práce z hlediska typů kódů, jimž se věnují a poskytuje stručný přehled pro získání motivace k vytvoření vlastního interaktivního nástroje s přihlédnutím k této analýze.

## 4.1 Online podpora TIK

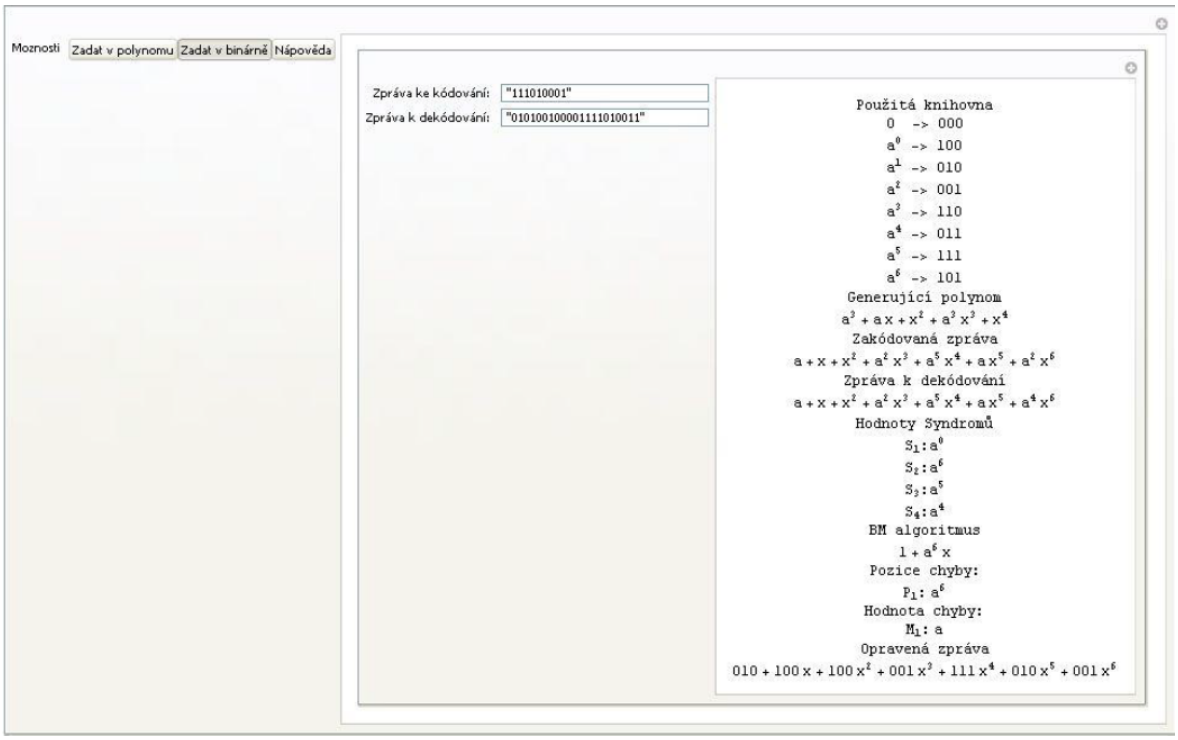
Online podpora výuky [1] od doc. Ing. Radomila Matouška, Ph.D. Mimo nástrojů ilustrující kódování informace, zde najdeme i učební text vztahující se k tématu kódování. Pod záložkou Flash aplikace najdeme složku Detekce/korekce chyb a nalezneme interaktivní příklady několika typů bezpečnostních kódů. Největším problémem je ukončená podpora zásuvného modulu Adobe Flash Player, ve kterém byly tyto příklady naprogramovány. Pro opětovné zpřístupnění je nutno využít zásuvný modul Ruffle pro prohlížeč Mozilla Firefox. Úskalím toho modulu vězí v tom, že nepodporuje standardní české znaky, tudíž je text místy zkomolený. Dále je problém u některých příkladů s posunutím grafických prvků, což taktéž nepřispívá k čitelnosti a přehlednosti nástroje.

## 4.2 Bezpečnostní kódy v prostředí Mathematica

Tato bakalářská práce [10] od Ing. Jiřího Facuny se zabývá Reed-Solomonovy kódy. Program byl vytvořen v prostředí Wolfram Mathematica. Nabízí možnost zadat zprávu ke kódování ve formě polynomu nebo binární interpretaci. Následně jsou zobrazeny údaje o přenášeném polynomu včetně mezivýsledků a přijatá zpráva. Tento nástroj není přístupný na webu, tudíž se nedá použít k výuce.




Obrázek 5. Ukázka z programu – zadání polynomem [10]




Obrázek 6. Ukázka z programu – zadání binárně [10]

## 4.3 Knihovna funkcí pro program Wolfram Mathematica umožňující snadný návrh bezpečnostních kódů ve výuce

Diplomová práce [11]od Ing. Jakuba Doubka. Práce se zabývá návrhem funkcí pro Wolfram Mathematica pro využití do výuky. Nejedná se přímo o nástroj popisující lineární

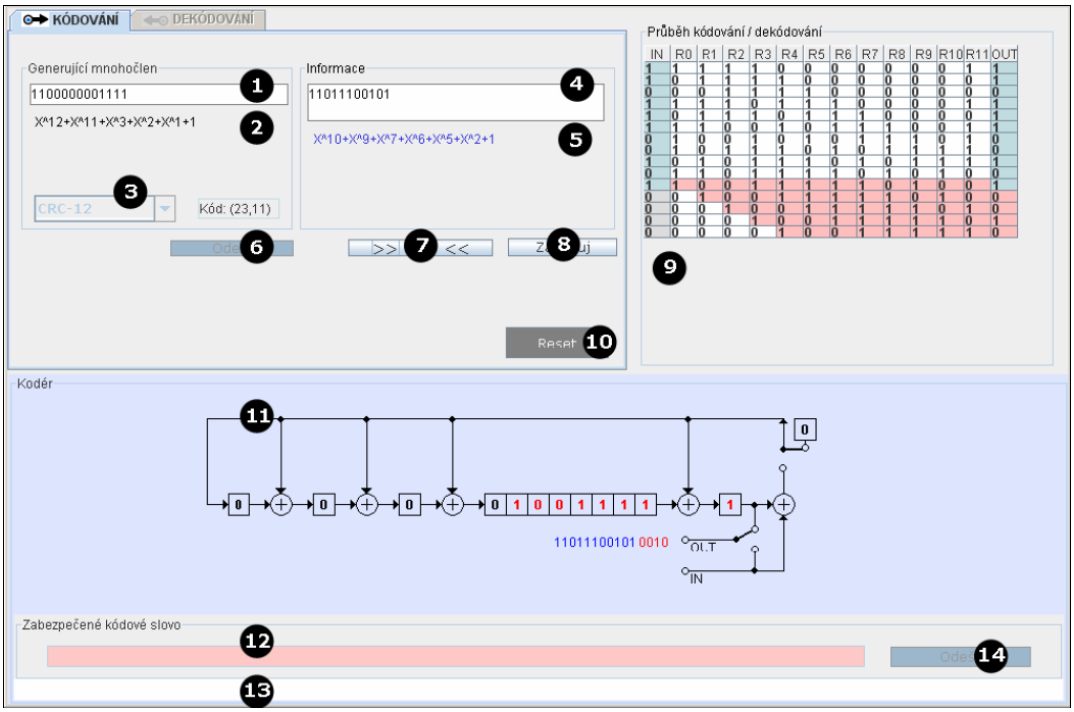
bezpečnostní kódy, ale pokládá základ pro možnost vyzkoušení samostatného naprogramování vybraných lineárních kódů pomocí připravených knihovních funkcí. Tato knihovna ovšem není volně dostupná.

## 4.4 Lineární binární bezpečnostní kódy

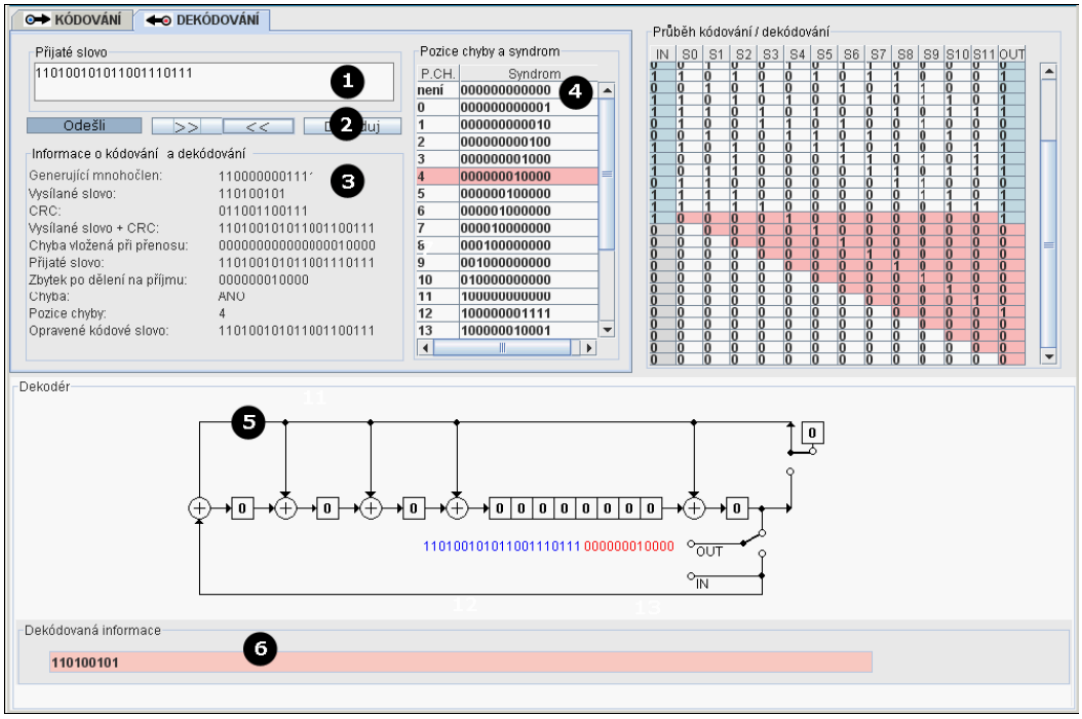
Bakalářská práce [12] od Ing. Jakuba Kupčíka se zabývají doplněním studijních materiálů, pro již dnes neexistující stránky pro podporu předmětu Základy informatiky v prostředí WebMathematica. V práci řeší kódování a dekódování Hammingových a Reed-Mullerových kódů. Jak bylo zmíněno, tato podpora není bohužel pro studenty nadále dostupná, pokládá ovšem základ pro rozšíření vhodných studijních materiálů pro výuku. Obrázky z práce nejsou dostatečně kvalitní k tomu, aby byly reprodukovány zde.

## 4.5 Cyklické kódy

Bakalářská práce [5] Ing. Jakuba Kettnera se zabývá především cyklickým kódem. Program je vytvořen v jazyce Java. Obsahuje krokování algoritmu postupného kódování a dekódování se znázorněním obvodu kodéru a dekodéru. Zobrazuje mezivýsledky a přenesenou zprávu, která je případně opravena. Tento program ovšem není dostupný, neboť odkaz v práci již neodkazuje na stránku s programem.




Obrázek 7. Ukázka z programu – Kódování cyklického kódu [5]



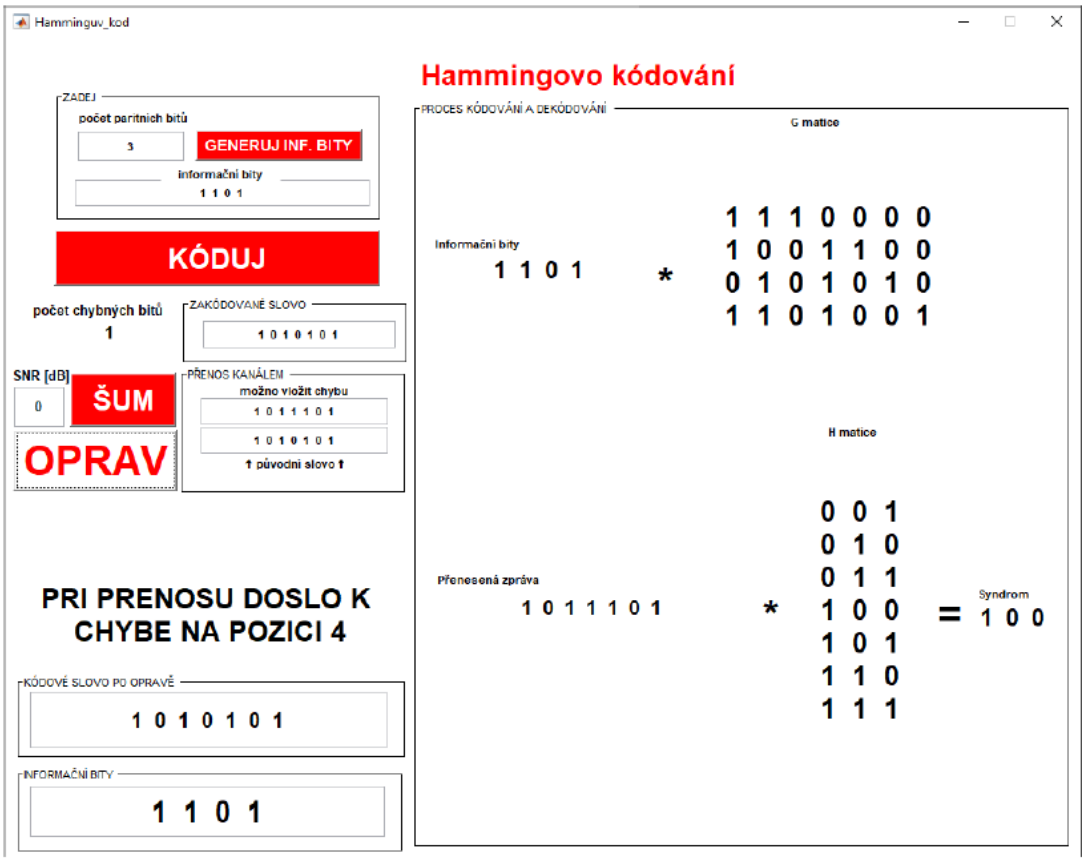

Obrázek 8. Ukázka z programu – Dekódování cyklického kódu [5]

## 4.6 Kodér a dekodér samoopravného kódu pro programovatelné paměti typu ROM

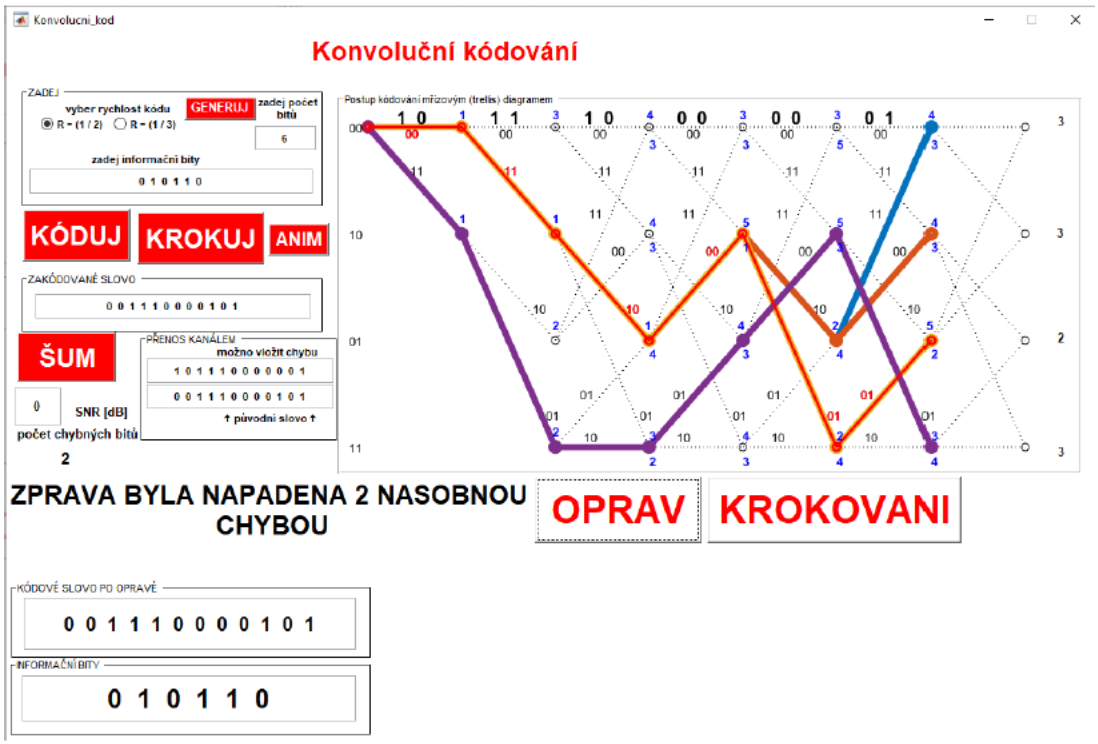
Bakalářská práce [13] Ing. Jana Bareše se zabývá kódováním a dekódováním bezpečnostních kódů z elektrotechnického hlediska pomocí jazyka VHDL. Neobsahuje však program, který by tyto procesy ilustroval. Jedná se o vytvoření aplikace, která je schopná generování kodéru a dekodéru za použití jazyka VHDL.

## 4.7 Program pro demonstraci kanálového kódování

Diplomová práce [14] Ing. Radka Zázvorky se zabývá kódováním přenosu skrze přenosový kanál. Obsahuje program, vytvořený v prostředí Matlab, který ilustruje algoritmus kódování a dekódování Hammingova kódu, cyklického kódu se znázorněním logického obvodu, konvolučního kódu a LDPC kódu. Program je navržen k samostatné interaktivní práci na počítačovém cvičení, které si mají studenti projít sami. Nástroj není volně dostupný pro studenty z jiných vysokých škol. Pro ukázku jsou zde uvedeny i některé obrázky z programu.



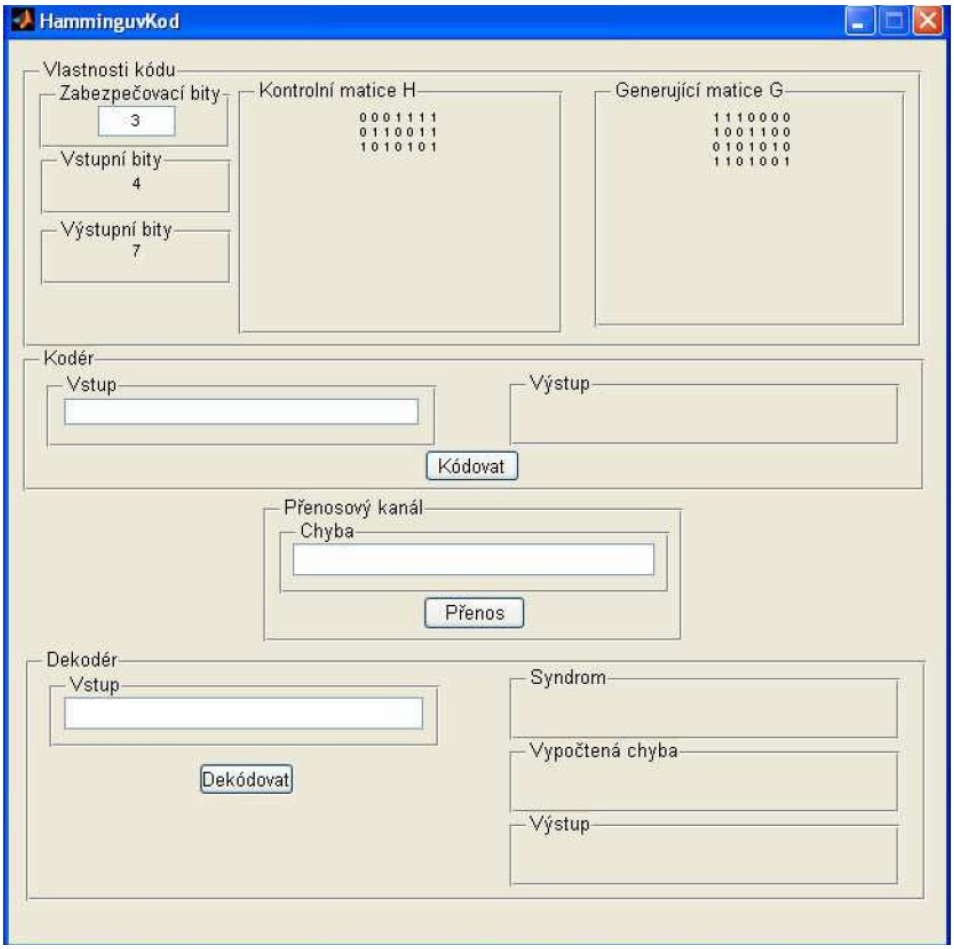

Obrázek 9. Ukázka z programu – Hammingův kód [14]



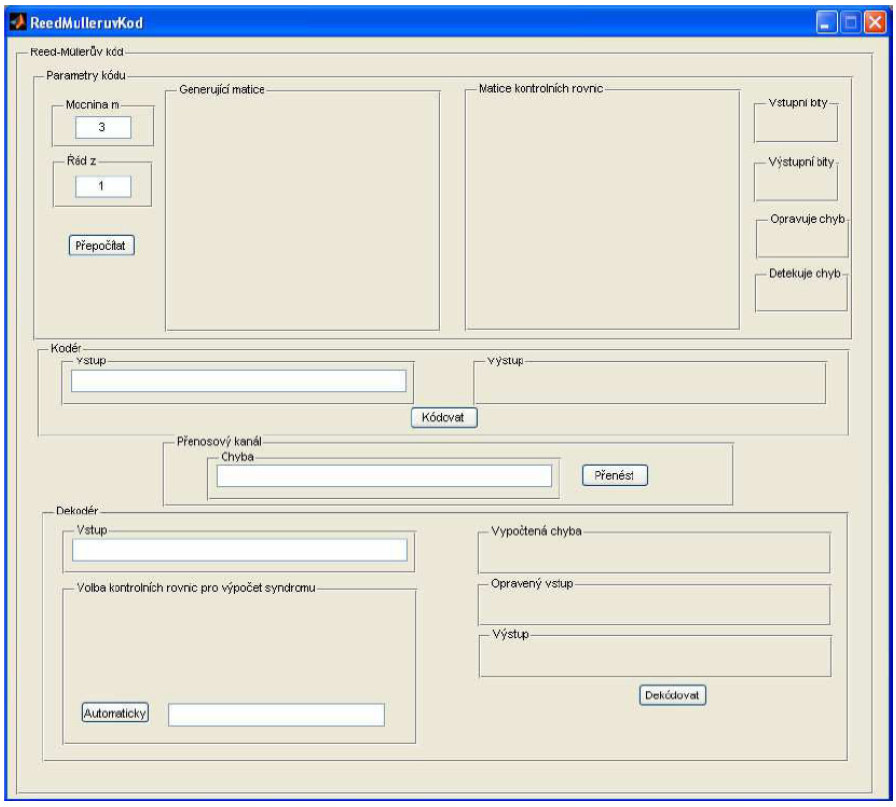

Obrázek 10. Ukázka z programu – Konvoluční kódování [14]

## 4.8 Protichybové zabezpečení v digitálních komunikačních systémech

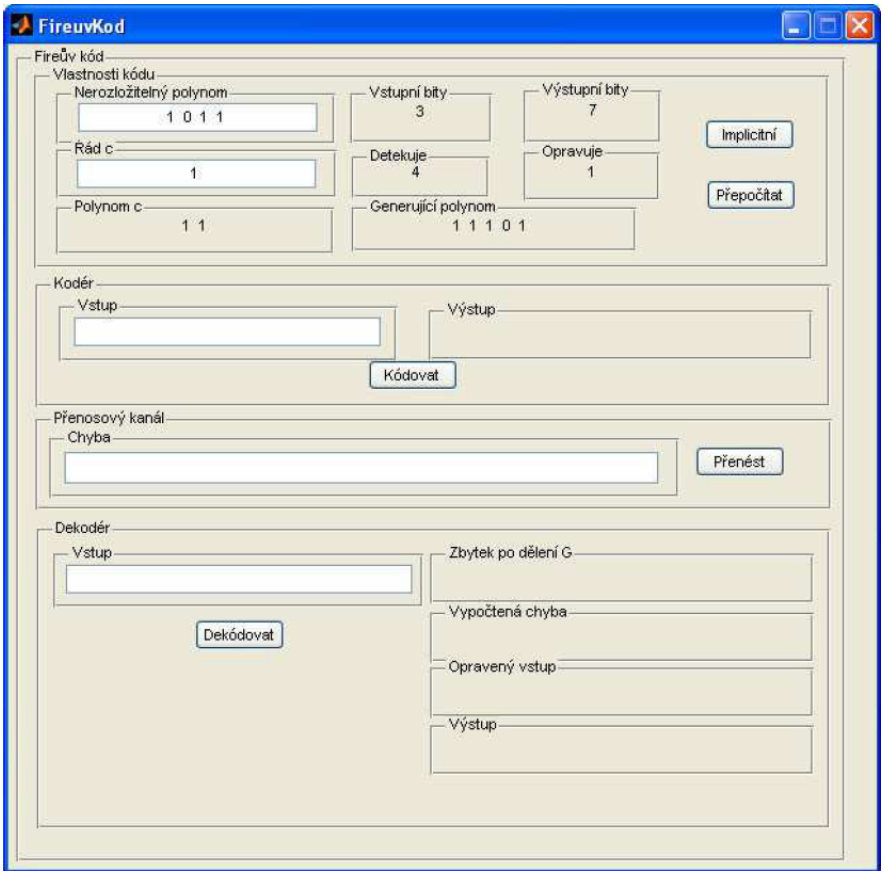
Diplomová práce [15] Ing. Jana Kostrhouna se zabývá dopřednou protichybovou korekcí přenášených zpráv. Nástroj je naprogramován v prostředí Matlab. V práci nalezneme tyto kódy: Hammingův, Reed-Mullerův, Fireův, Reed-Solomonův a Trellis kódová modulace. Nástroj nabízí možnost vložení základních parametrů, podle kterých je určován průběh procesů kódování a dekódování. Program není online dostupný. Pro ukázku jsou přiloženy vybrané obrázky.




Obrázek 11. Ukázka programu – Hammingův kód [15]




Obrázek 12. Ukázka z programu – Reed-Mullerův kód [15]




Obrázek 13. Ukázka z programu – Fireův kód [15]

## 4.9 Shrnutí

Aktuálně výše uvedené nástroje jsou špatně dostupné, těžce dohledatelné, uběhla jejich podpora nebo je zapotřebí speciálního softwaru k jejich správné funkci. Z tohoto důvodu se tato práce zaměřuje na poskytnutí lepší podpůrné nadstavby pro výuku předmětů spjatých s touto problematikou pro lepší představu, jak se lineární bezpečnostní kódy chovají z hlediska celého procesu od kódování, přes následný přenos a jeho podobu až k dekódování přijaté zprávy. Z rešerše vyplývá, že vizuální podpora pro odzkoušení vybraných lineárních kódů není dostatečná a na tento problém cílí tato práce.

# 5 MATEMATICKÝ ZÁKLAD

Praktická část práce se zabývá principy kódování a dekódování pomocí Hammingova kódu a kontroly parity. Pro pochopení principů těchto kódů je nezbytné mít základní znalosti z oblasti lineární algebry. Tato kapitola tedy podává přehled o základních pojmech a matematických operací, které doprovázejí kódování a dekódování příslušných kódů. Budou nastíněny principy maticových a vektorových operací, základní logické funkce a další.

## 5.1 Maticové operace

Matice typu $m \times n$ je definována jako

$$A_r = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \tag{29}$$

kde $a_{ij}$, pro $i = 1 \ldots m$ a $j = 1 \ldots n$ jsou reálná čísla.

Matice, jež vznikne záměnou řádků za sloupce se nazývá transponovanou maticí k matici $A$. Platí tedy

$$A^T = (a_{ji}) \tag{30}$$

kde $a_{ij}$ jsou prvky matice $A$.

Jednotková matice $E$ je speciální případ matice, jež má na hlavní diagonále prvky o hodnotě 1 a zbytek jsou 0. V případě násobení libovolné matice o stejných rozměrech je výsledek stejný jako násobení libovolného čísla jedničkou.

$$E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{31}$$

Matici o rozměru $1 \times n$ nebo $m \times 1$ můžeme definovat jako vektor $B$. V prvním případě jde o řádkový vektor a ve druhém o sloupcový. Vynásobením vektoru a matice dostaneme opět matici nebo vektor podle toho, z jaké strany toto násobení probíhá. Aby bylo možno násobit matici a vektor musí mít stejný řád, což znamená, že například matici o čtyřech sloupcích a sedmi řádcích je nutno násobit řádkovým vektorem o počtu čtyř prvků.

$$(1 \quad 0 \quad 1) * \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} =$$

$$(1 * 1 + 0 * 0 + 1 * 1 \quad 1 * 0 + 0 * 1 + 1 * 1 \quad 1 * 1 + 0 * 1 + 1 * 1 \quad 1 * 0 + 0 * 0 + 1 * 1)$$

$$= (0 \quad 1 \quad 0 \quad 1) \tag{32}$$

## 5.2 Binární operace

Ve dvojkové neboli binární soustavě probíhá násobení jinak, než je zvykem u soustavy desítkové. Při sčítání dostaneme pro $1 + 1 = 10$, ale pro kódování není možno využít dvoubitových hodnot, proto se využívá exkluzivního součtu XOR (Tabulka 2.) a logického součinu AND (Tabulka 3.). Dalšími logickými operacemi jsou logický součet OR (Tabulka 4.), negace logického součinu NAND (Tabulka 5.) a negace logického součtu NOR (Tabulka 6.). [16]

Tabulka 2. Exkluzivní součet - XOR

| $\oplus$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

Tabulka 3. Logický součin - AND

| $\otimes$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

Tabulka 4. Negace logického součinu NAND

| NAND | 0 | 1 |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | 0 |

Tabulka 5. Logický součet OR

| OR | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 1 |

Tabulka 6. Negace logického součtu NOR

| NOR | 0 | 1 |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 0 | 0 |

## 5.3 Funkce modulo

Pomocí funkce modulo zjišťujeme zbytek po dělení. V této práci se jedná především o modulo čísla dvě. Tato operace je využita v programu při násobení vektorů a matice, jelikož program pracuje s dekadickými hodnotami nul a jedniček, nikoliv s jejich binární reprezentací, proto při násobení vznikají čísla vyšší než jedna v důsledku sčítání mezivýsledků k dosažení konečných hodnot jednotlivých prvků. Proto využíváme vlastnosti funkce modulo čísla dva, která dává zbytky jedna nebo nula, jež odpovídají výsledkům, které by byly stejné při ekvivalentních binárních operacích.

# II. PRAKTICKÁ ČÁST

# 6 REALIZACE PRAKTICKÉ ČÁSTI

K úspěšné realizaci bylo třeba nastudovat podrobněji problematiku týkající se nejvhodnější vývojové technologie k naprogramování zamýšleného programu v jazyce C#, který je mi jako programovací jazyk nejbližší a hodnotím se ve schopnosti využití jeho potenciálu jako středně pokročilý. Toto je vysvětleno v kapitole 7. Volba padla na technologii WPF jevící se jako nejlepší varianta, kvůli zamýšlené animaci přenosu signálu, pro kterou má WPF nejlepší podporu.

Pro matematické operace uvedené v teoretické části jsem našel specializovaný matematický balíček, jež mi umožnil soustředit se na programování samotného algoritmu kódování a dekódování více než na vyvíjení vlastních metod pro výpočet maticových operací.

Pro matematický zápis výstupu programu jsem využil balíček Wpf-Math, jenž mi pomohl dosáhnout lepší přehlednosti a pěknějšího grafického vzhledu, neboť v sobě tento balíček ukrývá parser umožňující převést vkládaný text do formátu LaTeX.

Jako nejlepší kandidáty pro ukázku algoritmů kódování a dekódování jsem naprogramoval Hammingův kód K(7, 4) v jeho systematické i nesystematické formě a kontrolu parity. Tímto se zabývá kapitola 8.

Výstupem praktické části je tedy výukový program, jenž bude využit jako pomocná pomůcka při studiu problematiky přenosu informace nebo kódování a dekódování.

Na konci je v kapitole 9 uvedeno zhodnocení přínosů práce, kterých bylo dosaženo.

# 7 TECHNOLOGIE POUŽITÉ PRO TVORBU PROGRAMU

Program byl vytvořen ve Visual Studio Community 2019. Bylo využito technologie WPF od společnosti Microsoft. Oproti starší verzi .NET platformy WinForms podporující desktopové aplikace, WPF usnadňuje tvorbu programu z grafické stránky a poskytuje lepší odezvy na změny, jež se odehrávají v kódu na pozadí. K tvorbě grafického rozhraní je používán jazyk XAML. Aplikace je spustitelná pouze na platformě Windows. Testováno na Windows 10 Home.

## 7.1 Jazyk C#

Jazyk C# spatřil světlo světa v roce 2000, kdy jej přivedl k životu Anders Hejlsberg. Tento softwarový inženýr Microsoftu má na svém kontě kromě již zmíněného C# také TypeScript a Delphi. Jedná se o moderní objektově orientovaný programovací jazyk primárně určen pro .NET platformu, ale lze jej nezávisle použít na jiné open source platformě. Nepatří mezi nejlehčí programovací jazyky k naučení jako je Python, ovšem oproti Javě, která je svou komplexností vyhlášená, je programátorsky přívětivější. Řadí se mezi vyšší programovací jazyky, což jej činí vhodnou volbou pro pokročilé a expertní programátory. Lze jej využít pro vývoj mobilních aplikací, desktopových aplikací, cloudové služby, podnikové aplikace a také hry. [17]

## 7.2 Jazyk XAML

XAML je zkratou pro Extensible Markup Language, jež vychází z jazyku XML. Oba se řadí mezi takzvané značkovací jazyky. XML je starší verze pro definici, který je univerzální, tudíž se dá použít například u jiných značkovacích jazyků jako je DOC, PDF, HTML a jiné. XAML se využívá pro samostatnou deklaraci grafických elementů uživatelských rozhraní aplikací. Tento princip umožňuje snadněji vytvářet nezávislé prvky rozhraní na kódu v pozadí. XAML je úzce spojen s WPF ovšem rozšířil se mezi další produkty Microsoftu, například Windows Phone, Silverlight, UWP a další. Rozdělení grafické a programové stránky aplikace přináší řadu výhod od ušetření času při vývoji po snadnější implementaci jednotlivých prvků funkcionality. [18] [19]

## 7.3 WPF

Zkratka znamená Windows Presentation Foundation, což je framework neboli aplikační rámec nezávislý na rozlišení zařízení, na kterém daná aplikace běží. Pro vykreslování

využívá vektorové jádro. Pro návrh aplikace se používá již zmíněný jazyk C# a XAML. WPF nabízí plně modifikovatelné rozložení grafických prvků s možností deklarace svých vlastních elementů dle potřeby. Umožňuje napojení vlastností objektů na kód v pozadí tak, aby se při změně v jedné části promítly i do druhé. Z hlediska animací obsahuje podporu pro celou škálu volitelných animací, z čehož tato práce těží. [20]

## 7.4 Použité balíčky

V aplikaci jsou použity nugety neboli balíčky funkcí, které usnadňují práci při vytváření specifických operací s kódem v pozadí nutných pro správnou funkcionalitu vyvinuté aplikace.

Math.NET Numerics je široce rozšířený balíček pro práci s pestrou paletou matematických funkcí. V aplikaci je implementována část týkající se lineární algebry k práci s maticemi a vektory. [21]

Balíček WPF-Math slouží k zobrazení matematických formulí v přehledném formátování stylu LaTeX určený pro WPF. Je odvozen z projektu JMathTeX, jež měl podobné záměry z hlediska funkcionality. [22]

# 8 POPIS PROGRAMU

Program je určen pro usnadnění pochopení základních principů využití bezpečnostních kódů a procesů, které se odehrávají na pozadí. Obsahuje jednoduché uživatelské rozhraní a umožňuje uživateli projít jednotlivými kroky celého průběhu přenosu zprávy. Nachází se v něm systematický a nesystematický Hammingův kód K(7,4) a kontrola parity. Uživatel vkládá vstupní znaky ze zdrojové abecedy.

## 8.1 Interaktivní program pro ilustraci kódování a dekódování pomocí Hammingova kódu

Uvažujeme zdrojovou abecedu o délce 16 znaků (písmena A-P včetně) pro Hammingův kód K(7, 4). V tomto případě je ilustrováno kódování, přenos a dekódování jednoho znaku. Uživatel si může zvolit, jestli chce algoritmus krokovat nebo jej chce zobrazit celý najednou. Případně má možnost začít znovu pomocí tlačítka pro vymazání obsahu okna. Obrázky z ukázek níže jsou rozděleny do dvou částí ke zlepšení čitelnosti textu.

### 8.1.1 Postup

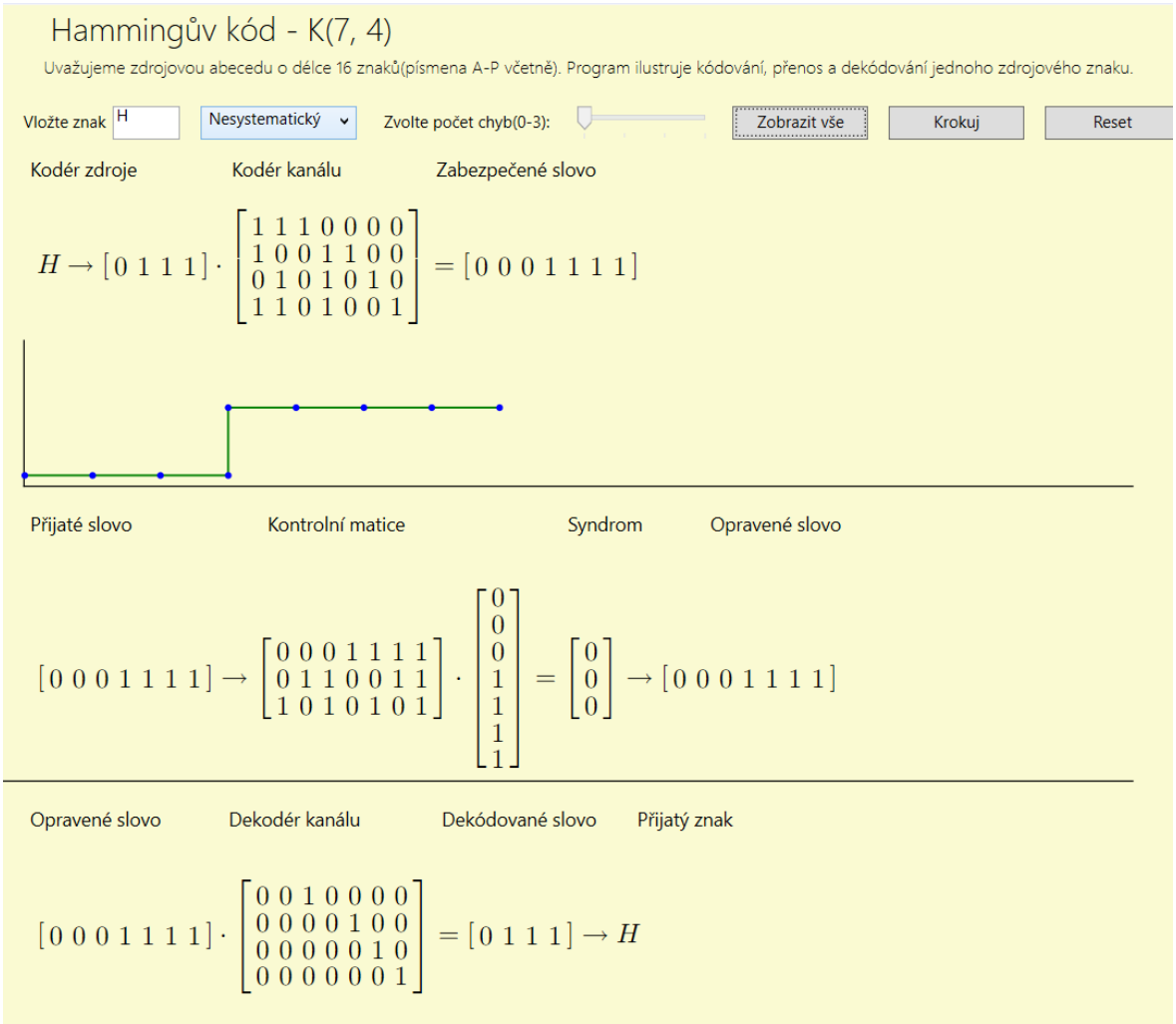
- Uživatel vloží do pole pro textový řetězec znak ze zdrojové abecedy.
- Vybere z rozbalovacího menu systematický nebo nesystematický typ kódu (výchozí hodnota je systematický).
- Zvolí počet chyb v intervalu od nuly do tří včetně (výchozí hodnota nastavena na nula chyb).
- Znak je převeden na předem stanovenou posloupnost čtyř bitů přiřazené z tabulky.
- Vynásobením této posloupnosti bitů generující maticí vznikne zabezpečené kódové slovo o velikosti sedmi bitů.
- Zabezpečené kódové slovo je převedeno na soustavu signálů, kde vzestupná hrana o velikosti 3,3 V reprezentuje bitovou jedničku a sestupná hrana o velikosti 0 V zastupuje bitovou nulu. Správný přenos je indikován zeleně a pokud uživatel zvolil od jedné do tří chyb, pak se tyto chyby projeví červenou barvou vykreslovaného signálu.
- Příjemce obdrží zabezpečené kódové slovo přenesené kanálem.
- Zabezpečené kódové slovo je vynásobeno kontrolní maticí.

- Pokud se ve slově vyskytují chyby, vyjde nenulový syndrom, podle tohoto syndromu je provedena oprava nalezené chyby.
- Opravené kódové slovo je převedeno zpět na čtyřbitové slovo.
- Následně je tato posloupnost převedena na znak ze zdrojové abecedy, které přijaté posloupnosti odpovídá.

### 8.1.2 Příklad s žádnou chybou při přenosu

Pro tuto ukázku jsou parametry:

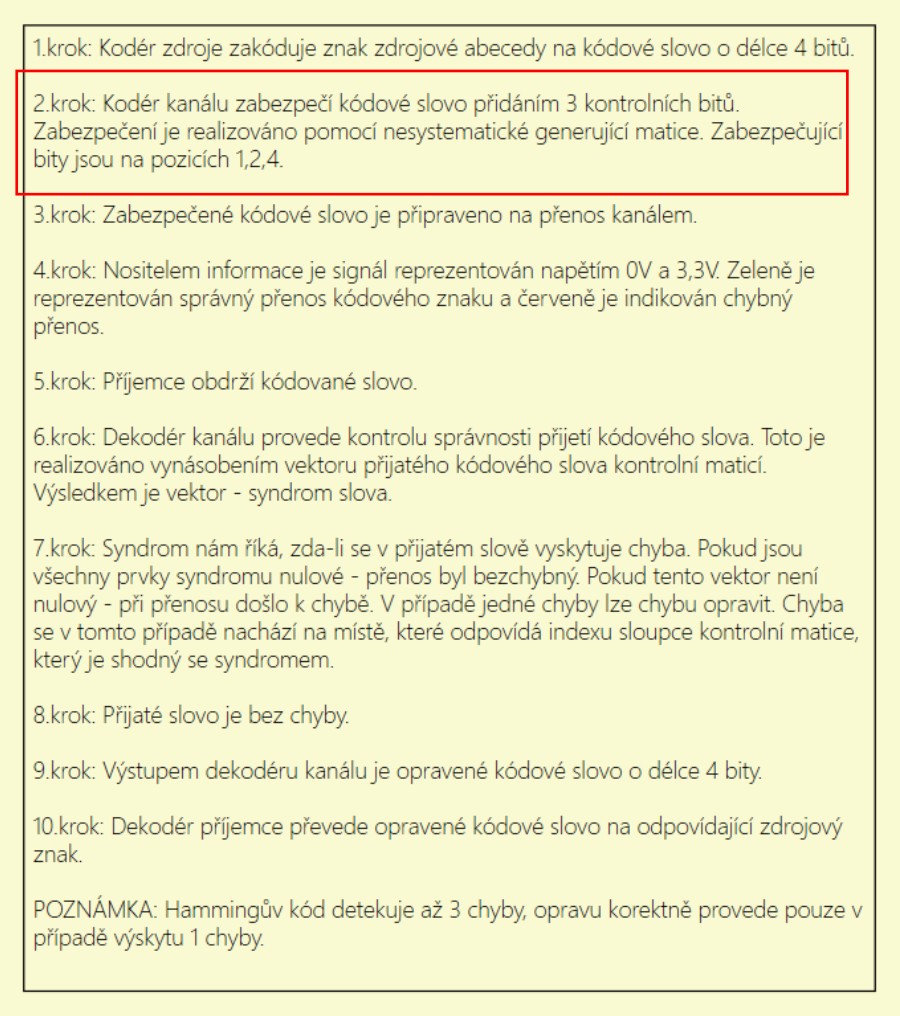
- Vstupní znak: H
- Typ kódu: Nesystematický
- Počet chyb: 0




Obrázek 14. Okno programu – Hammingův kód bez chyby (výpočetní část)

V této ukázce (Obrázek 14.) je ilustrován přenos bez chyby, tudíž nejsou změněny žádné prvky bitové posloupnosti přenášeného slova. Syndrom je nulový a ve výsledku dostaneme vyslaný znak tak, jak by měl být v návaznosti na to. V osmém kroku (Obrázek 15.) je uživatel informován o tom, že přijaté slovo je bez chyby.

Byl vybrán nesystematický kód, což se projeví na popisné části (Obrázek 15.) výstupu tak, že se změní text ve druhém kroku a informuje, že je kódování realizováno pomocí nesystematické generující matice a zabezpečující bity se nachází na pozicích 1, 2, 4.



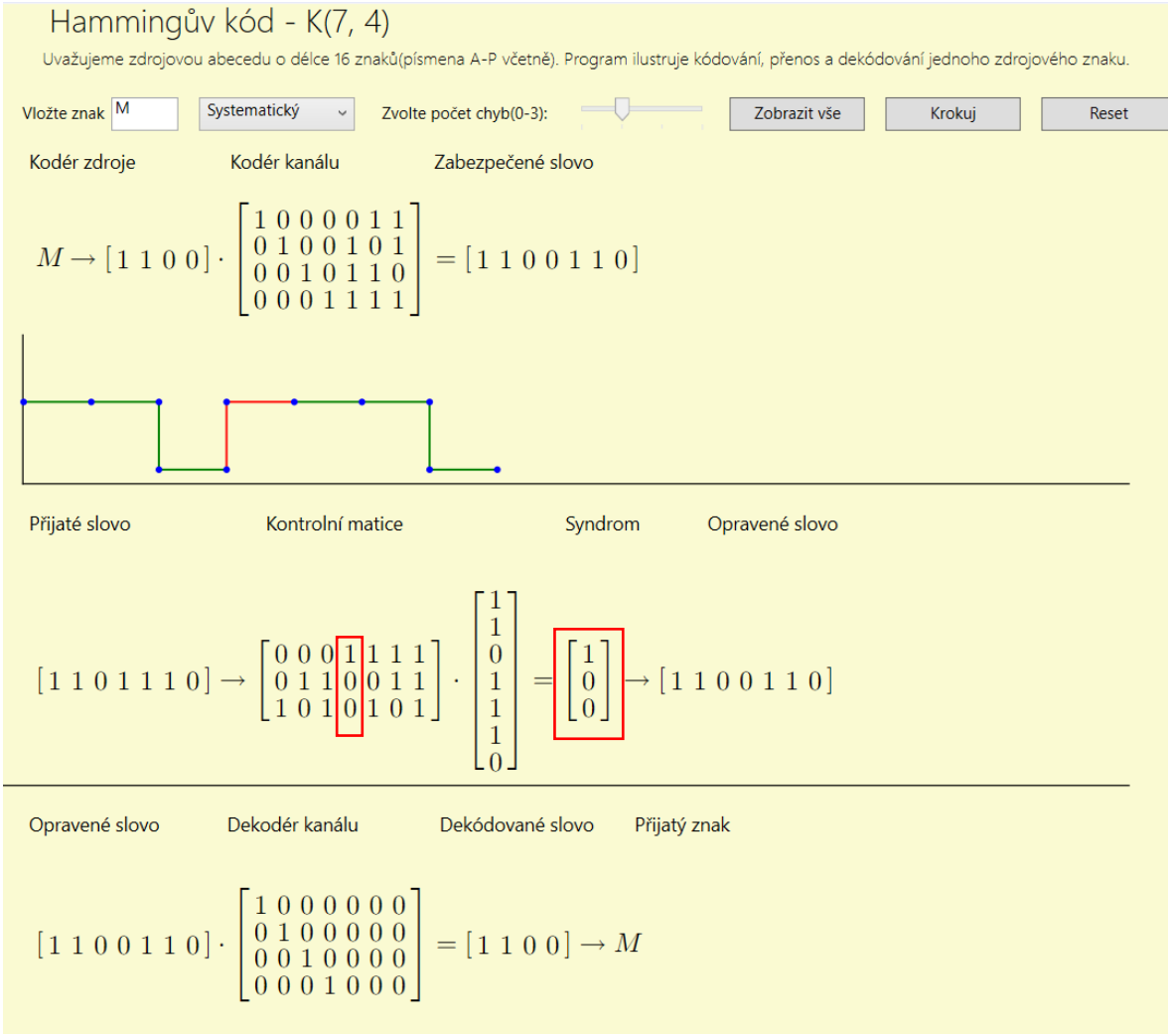

Obrázek 15. Okno programu – Hammingův kód bez chyby (popisná část)

### 8.1.3 Příklad s jednou chybou při přenosu

Pro tuto ukázku jsou parametry:

- Vstupní znak: M
- Typ kódu: Systematický

- Počet chyb: 1




Obrázek 16. Okno programu – Hammingův kód s jednou chybou (výpočetní část)

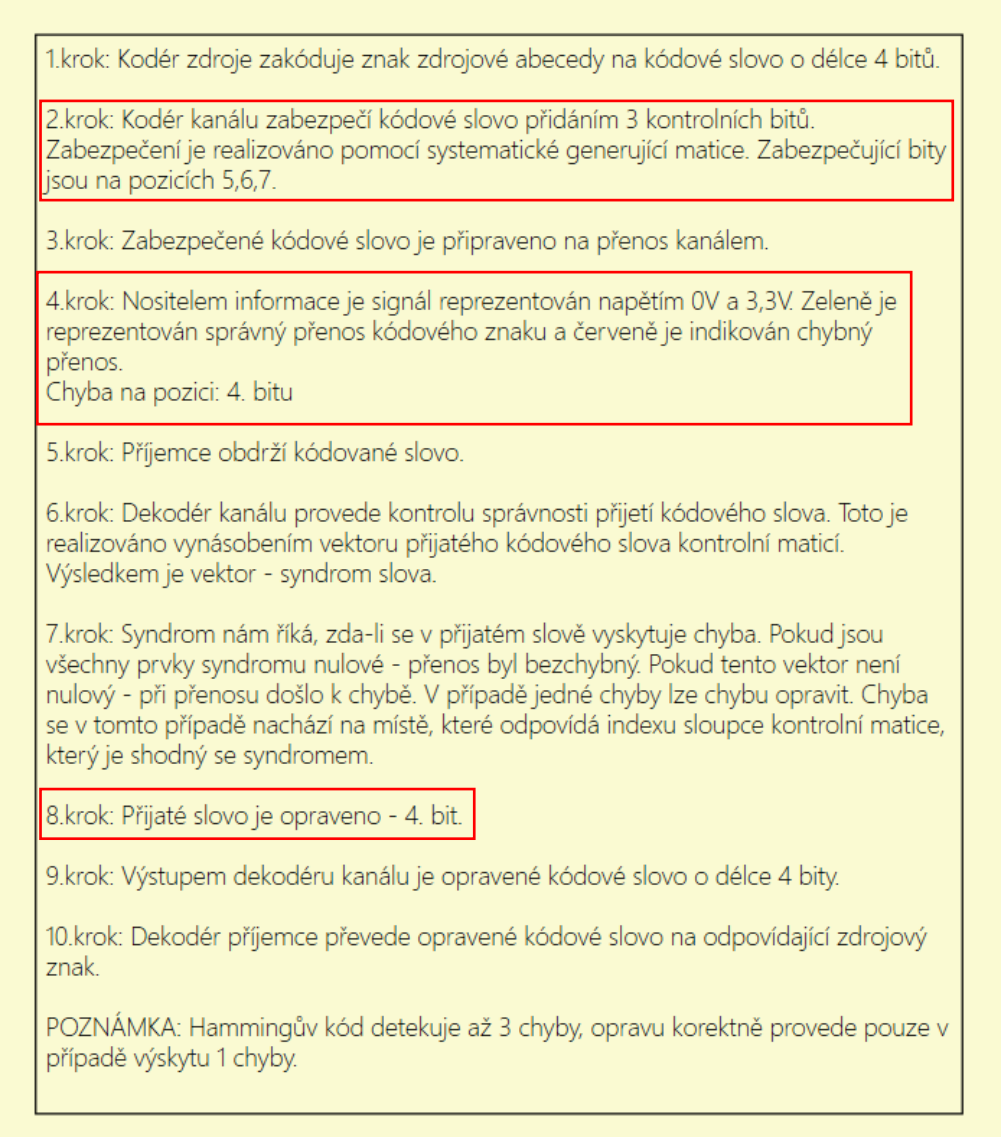
V této ukázce (Obrázek 16.) je realizován přenos s jednou chybou. Tato chyba se projeví při animaci signálu červenou barvou, aby byl uživatel schopen chybu ihned zaregistrovat. V popisné části (Obrázek 17.) je ve čtvrtém kroku přímo uvedeno, kolikátý bit byl postižen šumem.

Vybrán pro přenos byl systematický kód, což se projeví v popisné části tak, že je uživatel informován o využití systematické generující matice a o tom, že zabezpečující bity se nachází na pozicích 5, 6, 7.

Syndrom je nenulový, což indikuje, že nastala chyba. Provede se porovnání sloupců kontrolní matice a je zjištěno, kolikátému sloupci odpovídá vektor syndromu. Tento index sloupce z kontrolní matice je pak index chyby v přijatém slově. V osmém kroku (Obrázek

17.) je uživatel informován, že bylo přijaté zabezpečené kódové slovo opraveno, a že chyba se vyskytuje na pozici čtvrtého bitu.

Výstupem je opět přijatý znak, jež odpovídá znaku vyslanému.

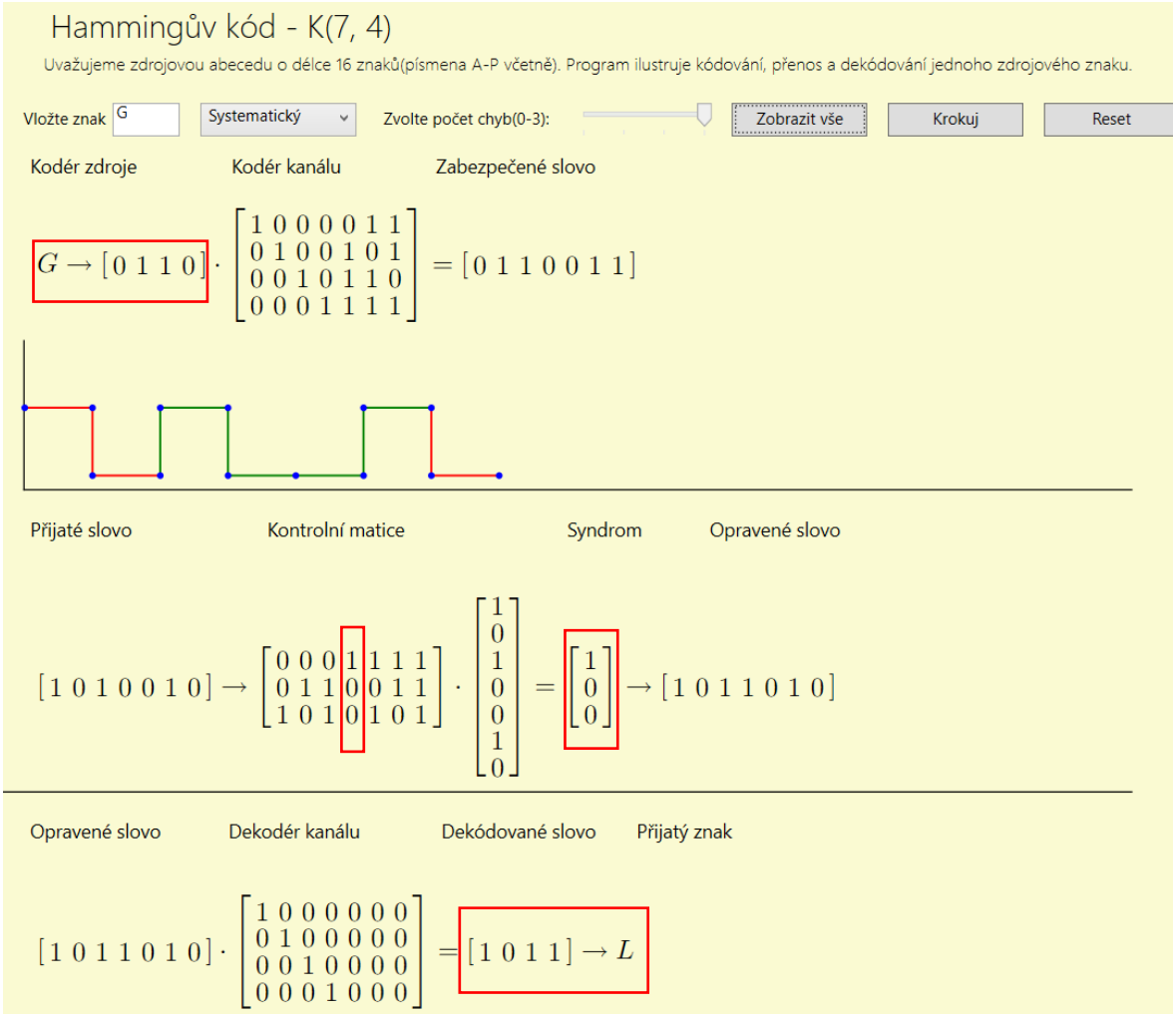



Obrázek 17. Okno programu – Hammingův kód s jednou chybou (popisná část)

### 8.1.4 Příklad se třemi chybami

Pro ukázku jsou zvoleny tyto parametry:

- Vstupní znak: G
- Typ kódu: Systematický
- Počet chyb: 3



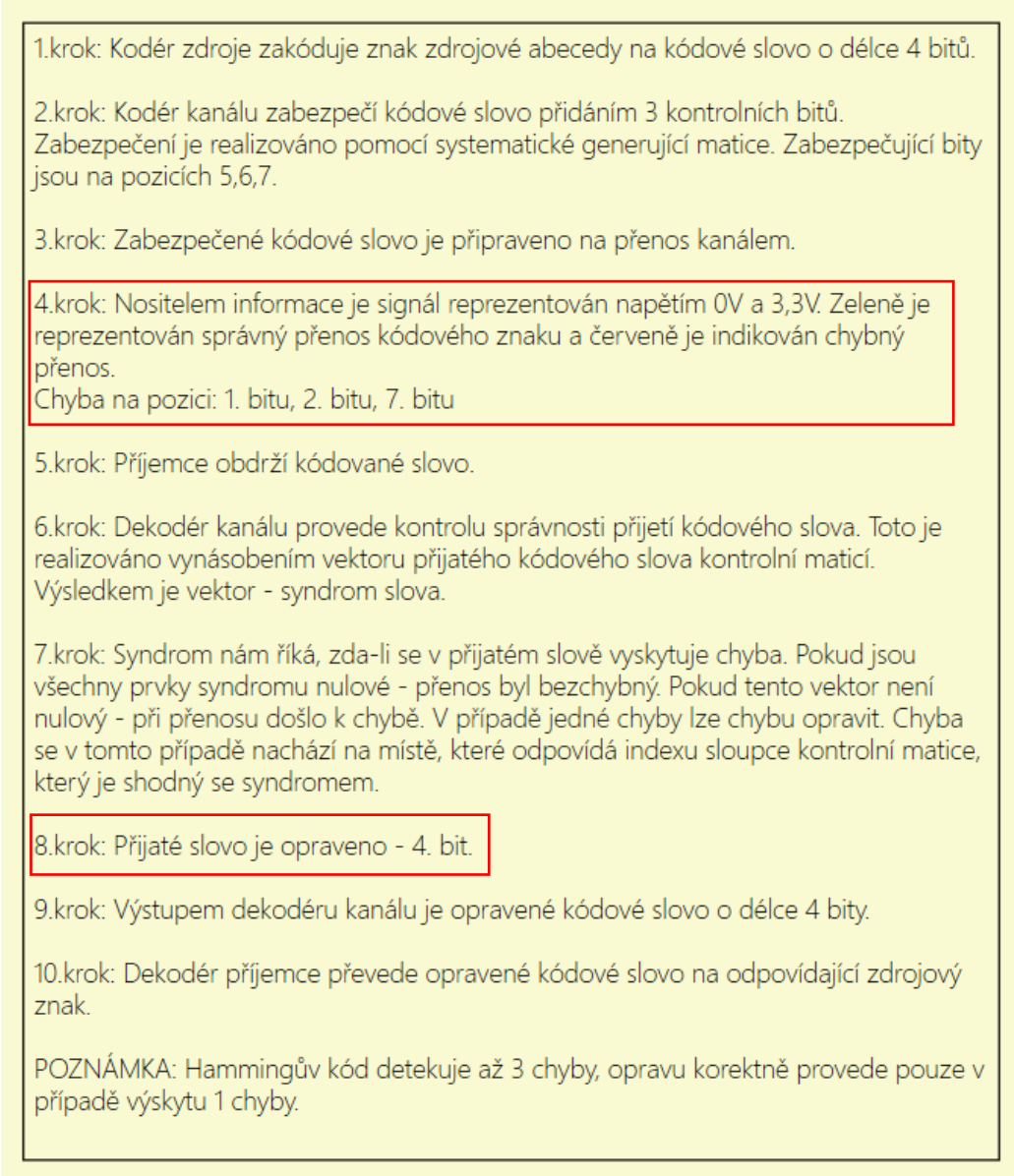

Obrázek 18. Okno programu – Hammingův kód se třemi chybami (výpočetní část)

V této ukázce (Obrázek 18.) jsou do přenosu zaneseny tři chyby, tento jev je indikován ve čtvrtém kroku (Obrázek 19.), kdy uživatel zjistí, že se chyby nachází na pozicích prvního, druhého a sedmého bitu.

Opět je zabezpečení realizováno systematickou generující maticí.

Vektor je nenulový, chyba tedy byla detekována, ovšem v tomto případě je počet chyb příliš vysoký na to, aby mohlo dojít ke správné korekci. Oprava proběhne na čtvrtém bitu, což není správně, protože to neodpovídá ani jedné pozici chyby, které známe z předchozího kroku.

Výstupem je tedy jiný přijatý znak neodpovídající znaku vyslanému. Tady se ke slovu hlásí limitace Hammingova kódu, jež dokáže správně opravit přijaté zabezpečené kódové slovo pouze v případě jedné chyby, jak bylo ilustrováno v ukázce výše.




Obrázek 19. Okno programu – Hammingův kód se třemi chybami (popisná část)

## 8.2 Interaktivní program pro ilustraci kódování a dekódování pomocí kódu kontroly parity

Uvažujeme zdrojovou abecedu o délce 32 znaků (písmena A-Z, čárka, tečka, ?, !, +, -). Program ilustruje kódování, přenos a dekódování zprávy až o třech znacích. Uživatel si může zvolit, jestli chce algoritmus krokovat nebo jej chce zobrazit celý najednou. Případně má možnost začít znovu pomocí tlačítka pro vymazání obsahu okna.

### 8.2.1 Postup

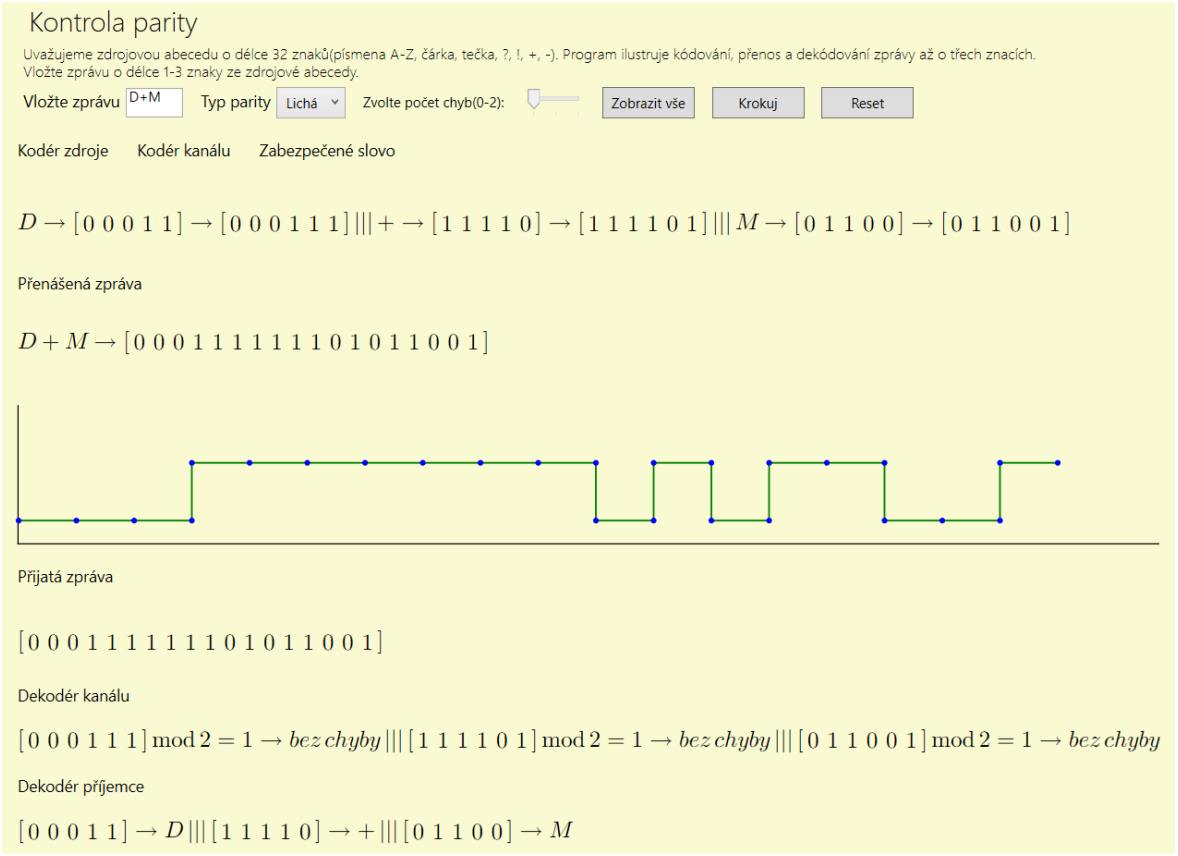
- Uživatel vloží znaky zdrojové abecedy, může vložit až tři znaky, které chce přenést.

- Vybere si z rozbalovacího menu typ parity – sudou nebo lichou (výchozí nastavení je sudá).
- Zvolí počet chyb v intervalu od nuly do dvou, které chce do zprávy zanést. (výchozí nastavení je nula chyb).
- Znak nebo znaky jsou převedeny na posloupnost pěti bitů přiřazené z tabulky.
- Každá tato posloupnost je zabezpečena paritním bitem – vznikne tedy posloupnost o šesti bitech.
- Všechna tato zabezpečená kódová slova jsou spojena do jedné posloupnosti bitů v pořadí, v jakém byly vloženy znaky vysílané zprávy.
- Celá zpráva, reprezentována jednou posloupností bitů, je převedena na soustavu signálů, kde vzestupná hrana o velikosti 3,3 V reprezentuje bitovou jedničku a sestupná hrana o velikosti 0 V zastupuje bitovou nulu. Správný přenos je indikován zeleně. Pokud uživatel zvolil od jedné do dvou chyb, pak se tyto chyby projeví červenou barvou vykreslovaného signálu.
- Příjemce obdrží přenesenou zprávu.
- Zpráva je opět rozdělena na šestici bitů.
- Proběhne kontrola pomocí funkce modulo, která určí, jestli zabezpečená kódové slovo dorazilo bez chyby.
- Zabezpečená kódové slovo je převedeno zpět na pětici bitů.
- Tyto pětice jsou přiřazeny znakům zdrojové abecedy dle tabulky.

### 8.2.2 Příklad s žádnou chybou

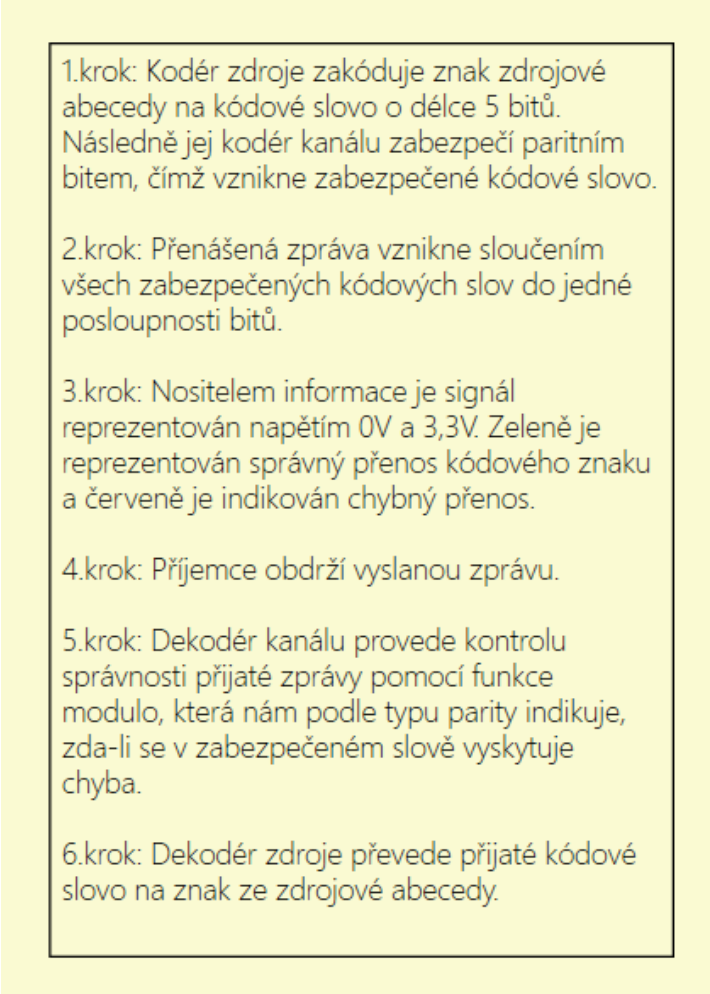
Pro ukázku jsou zvoleny tyto parametry:

- Vstupní znaky: D+M
- Typ parity: Lichá
- Počet chyb: 0




Obrázek 20. Okno programu – Kontrola parity bez chyby (výpočetní část)

V této ukázce (Obrázek 20.) je ilustrován přenos bez chyby pětibitových slov, jenž se po zabezpečení lichým paritním bitem změní na šestibitové zakódované slovo. Při přenosu nedojde k žádné chybě a pro všechny kontroly znaků je indikováno, že daný znak byl přijat bez chyby. V popisné části (Obrázek 21.) je proces popsán krok za krokem.

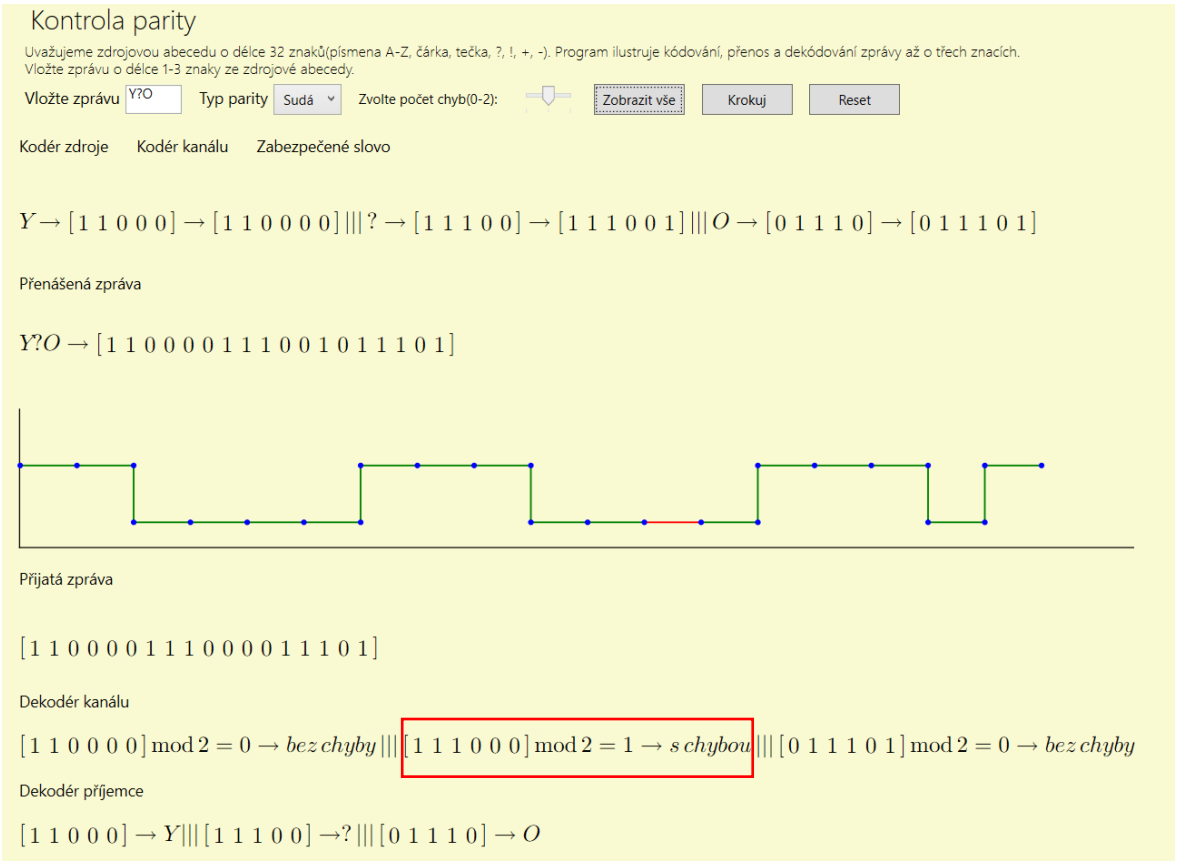



Obrázek 21. Okno programu – Kontrola parity bez chyby (popisná část)

### 8.2.3 Příklad s jednou chybou

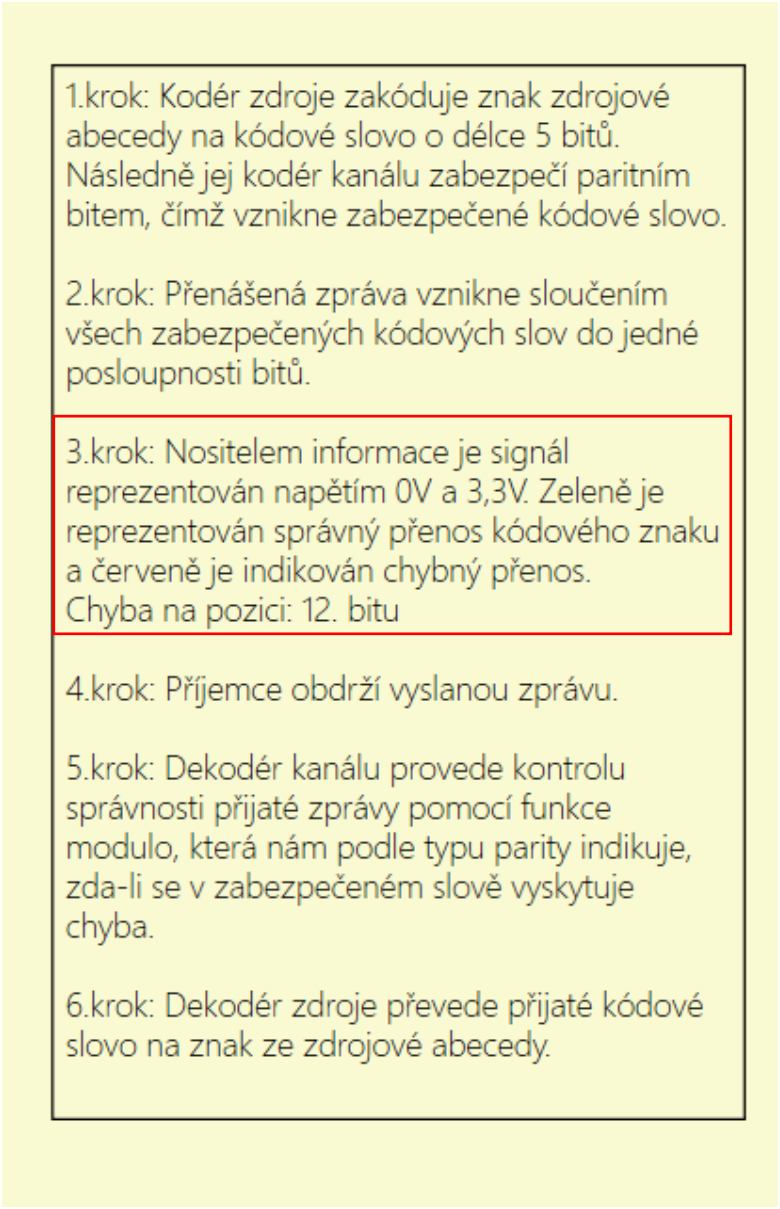
Pro ukázku jsou zvoleny tyto parametry:

- Vstupní znaky: Y?O
- Typ parity: lichá
- Počet chyb: 1



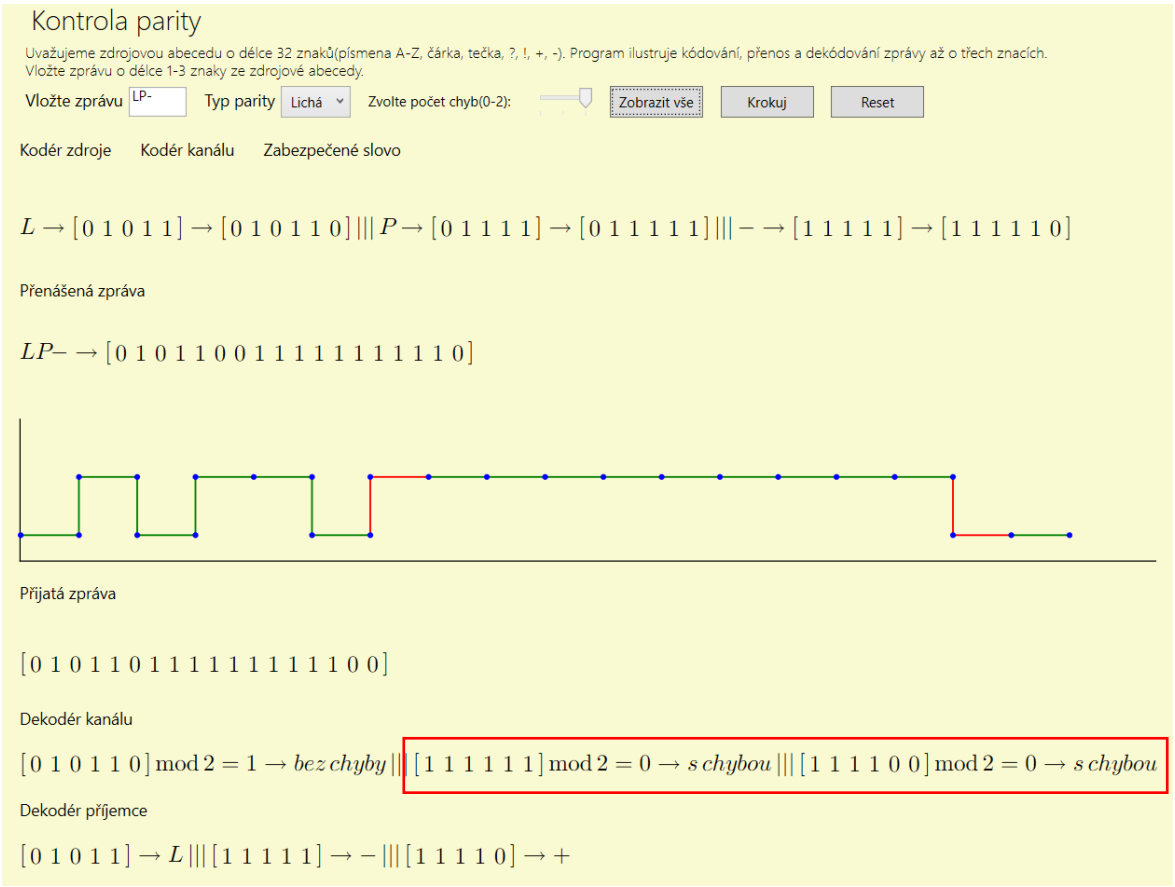

Obrázek 22. Okno programu – Kontrola parity s jednou chybou (výpočetní část)

V této ukázce (Obrázek 22.) je ilustrován přenos s jednou chybou, jež se může náhodně vyskytnout v jakémkoliv bitu přenášené posloupnosti. Chyba se objevila na dvanácté pozici zakódovaného slova. Ve třetím kroku v popisné části (Obrázek 23.) je na tuto skutečnost uživatel upozorněn. Chyba se objevila na paritním bitu, díky čemuž se nezměnil přijatý znak. To v reálné situaci ovšem nelze zjistit. Jakmile je chyba detekována, předpokládá se, že daný přijatý znak nebyl správně přenesen a celý přenos by se tedy měl zopakovat.



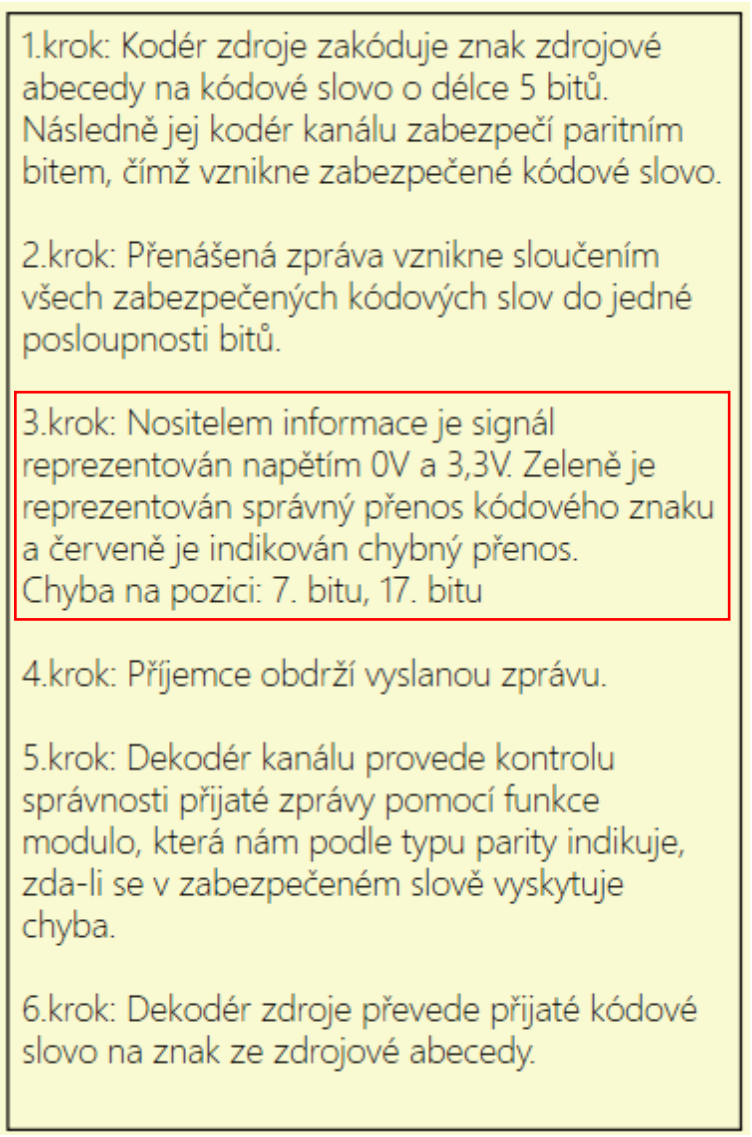

Obrázek 23. Okno programu – Kontrola parity s jednou chybou (popisná část)

### 8.2.4 Příklad se dvěma chybami




Obrázek 24. Okno programu – Kontrola parity se dvěma chybami (výpočetní část)

V této ukázce (Obrázek 24.) jsou do přenosu zaneseny dvě chyby, což se projeví změnou přijatých znaků na znaky jiné. To znamená, že je potřeba přenos zopakovat k přenesení správné sekvence znaků. Chyby se vyskytují na pozicích sedm a sedmnáct, uživatel je informován v kroku tři (Obrázek 25.).




Obrázek 25. Okno programu – Kontrola parity se dvěma chybami (popisná část)

# 9 ZHODNOCENÍ PŘÍNOSŮ PRÁCE

Přínos této práce spočívá v přiblížení fungování algoritmů kódování, přenosu a následného dekódování dvou základních a nejvyužívanějších lineárních zabezpečovacích kódů. Program může být využit při výuce předmětů související s přenosem informace. Usnadňuje pochopení základních principů vedoucích k zabezpečení těmito kódy včetně ilustrace animace přenosu informace. Samotný student by měl při samostudiu s pomocí tohoto programu snadněji pochopit a zapamatovat si důležité informace. Ve výuce tento program také najde využití, neboť může usnadnit výklad vyučujícím. Program může sloužit i pro osoby z řad široké veřejnosti, které přicházejí s danou problematikou do kontaktu. Důraz byl kladen na to, aby byl program správně nejen z hlediska samotných kódů, ale i z matematického pohledu. Program ukazuje klasickou reprezentaci znaků běžné abecedy, jejich binární reprezentaci a celý děj, který se odehrává na straně odesílatele i na straně příjemce zprávy. Cílem bylo rozbít představu o tom, že tato přenášená zakódovaná slova jsou přenášena jako jedničky a nuly, ale jsou určitým způsobem přetvořeny na signály, jež jsou přenášeny kanálem, na jehož konci dojde ke zpětné transformaci signálových prvků na posloupnost bitů a na znaky ze zdrojové abecedy, kterou dokáže přečíst člověk bez problémů.

# ZÁVĚR

V této práci jsem se věnoval lineárním bezpečnostním kódům a jejich využitím. Na základě pečlivého nastudování nejdůležitějších informací k úspěšné implementaci a popsání základních principů vztahující se k dané problematice jsem zpracoval literární rešerši a vytvořil výukový program.

V teoretická část jsem se zaměřil na jednoduché popsání principů, jak vzniká kód, postup kódování a dekódování. Dále jsem popsal, jak můžeme detekovat a opravovat chyby pomocí lineárních bezpečnostních kódů.

Následně jsem zpracoval kapitolu vztahující se k podobným nástrojům, který si předsevzaly stejné cíle, a to je přiblížit problematiku přenosu informace spolu s ilustrací procesu ke snazšímu pochopení daných algoritmů.

V praktické části jsem vytvořil program demonstrující Hammingův kód a kontrolu parity ve vývojovém prostředí Visual Studio s implementací v jazyku C#. Pomocí tohoto programu lze přiblížit fungování a vlastnosti těchto kódů pro využití ve výuce. U obou kódů jsou zobrazeny nejdůležitější mezivýsledky, aby bylo zjevné, jak bylo dosaženo konečného stavu.

Cílem práce bylo vytvořit nástroj pro přiblížení problematiky bezpečnostních kódů.

# SEZNAM POUŽITÉ LITERATURY

[1] MATOUŠEK, Radomil. Metody kódování [online]. 2006. Brno, 2006 [cit. 2021-5-11]. Dostupné z: http://www.uai.fme.vutbr.cz/~matousek/TIK/tik_text4.html

[2] ZELINKA, Ivan a Zdenka PROKOPOVÁ. Základy informatiky. Zlín: Univerzita Tomáše Bati, 2005, 112 s. ISBN 8073182998.

[3] NEVLUD, Pavel a Marek DVORSKÝ. Přenos dat: učební text. Ostrava: Vysoká škola báňská - Technická univerzita, 2012. ISBN 978-80-248-2604-2.

[4] ADAMEC, Milan. Cyklické bezpečnostní kódy. Zlín: Univerzita Tomáše Bati ve Zlíně, 2011, 60 s. (52 640 znaků). Dostupné také z: http://hdl.handle.net/10563/14970. Tomas Bata University in Zlín. Faculty of Applied Informatics, Ústav automatizace a řídicí techniky. Vedoucí práce Chramcov, Bronislav.

[5] KETTNER, Jakub. Kódování - cyklické kódy. Zlín: Univerzita Tomáše Bati ve Zlíně, 2008, 59 s., 3 s. text. příloh. Dostupné také z: http://hdl.handle.net/10563/5509. Tomas Bata University in Zlín. Faculty of Applied Informatics, Ústav aplikované informatiky. Vedoucí práce Krčmář, Miloš.

[6] KAŠPAR, Jaroslav. Zabezpečení přenosu dat BCH kódy [online]. Brno, 2008 [cit. 2021-5-11]. Dostupné z: http://hdl.handle.net/11012/11525. Diplomová práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací. Vedoucí práce Karel Němec.

[7] HORAL, Pavel. Reed-Solomonovy kódy a jejich aplikace. Praha, 2006[cit. 2021-5-11]. Dostupné z: http://hdl.handle.net/20.500.11956/5824. Bakalářská práce. Univerzita Karlova, Matematicko-fyzikální fakulta, Katedra algebry. Vedoucí práce Drápal, Aleš.

[8] LIŠKA, Ondřej. Konstrukce ternárních Golayových kódů. Praha, 2010 [cit. 2021-5-11]. Dostupné z: http://hdl.handle.net/20.500.11956/28486. Bakalářská práce. Univerzita Karlova, Matematicko-fyzikální fakulta, Katedra algebry. Vedoucí práce Drápal, Aleš.

[9] VASEGHI, Saeed V. Advanced digital signal processing and noise reduction. 4th ed. Chichester: John Wiley, 2008, xxvi, 514 s. ISBN 978-0-470-75406-1.

[10] FACUNA, Jiří. Bezpečnostní kódy v prostředí Mathematica [online]. Zlín, 2012 [cit. 2021-05-11]. Dostupné z: https://theses.cz/id/yq0j3f/. Bakalářská práce

Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky. Vedoucí práce Chramcov, Bronislav.

[11] DOUBEK, Jakub. Knihovna funkcí pro program Wolfram Mathematica umožňující snadný návrh bezpečnostních kódů ve výuce. Praha, 2016. Dostupné z: https://users.fit.cvut.cz/~xkubalik/lib/exe/fetch.php?media=project:2016:2016_doubek_jakub.pdf. Diplomová práce. České vysoké učení technické v Praze, Fakulta informačních technologií. Vedoucí práce Kubalík, Pavel.

[12] KUPČÍK, Jakub. Lineární binární bezpečnostní kódy. Zlín: Univerzita Tomáše Bati ve Zlíně, 2007, 54 s. Dostupné také z: http://hdl.handle.net/10563/4230. Tomas Bata University in Zlín. Faculty of Applied Informatics, Ústav aplikované informatiky. Vedoucí práce Chramcov, Bronislav.

[13] BAREŠ, Jan. Kodér a dekodér samoopravného kódu pro programovatelné paměti typu ROM [online]. Brno, 2016 [cit. 2021-5-12]. Dostupné z: http://hdl.handle.net/11012/61552 . Bakalářská práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav mikroelektroniky. Vedoucí práce Martin Šťáva.

[14] ZÁVORKA, Radek. Program pro demonstraci kanálového kódování [online]. Brno, 2020 [cit. 2021-5-12]. Dostupné z: http://hdl.handle.net/11012/189141. Diplomová práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav radioelektroniky. Vedoucí práce Aleš Prokeš.

[15] KOSTRHOUN, Jan. Protichybové zabezpečení v digitálních komunikačních systémech [online]. Brno, 2013 [cit. 2021-5-12]. Dostupné z: http://hdl.handle.net/11012/26559 . Diplomová práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací. Vedoucí práce Radim Číž.

[16] VLČEK, Karel. Komprese a kódová zabezpečení v multimediálních komunikacích. Praha: BEN — technická literatura, 2000, 225 s. ISBN 8086056686.

[17] Everything you need to know about C#. Pluralsight [online]. [cit. 2021-5-12]. Dostupné z: https://www.pluralsight.com/blog/software-development/everything-you-need-to-know-about-c-

[18] BASU, Sam. Telerik. Telerik [online]. 2017 [cit. 2021-5-12]. Dostupné z: https://www.telerik.com/blogs/xaml-standard-demystified

[19] JECHA, Tomáš. Jazyk XAML [online]. 2012 [cit. 2021-5-12]. Dostupné z: https://www.dotnetportal.cz/clanek/198/Jazyk-XAML

[20] Dokumentace k Windows Presentation Foundation. Microsoft [online]. [cit. 2021-5-12]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/desktop/wpf/?view=netdesktop-5.0

[21] Math.NET Numerics Documentation. Math.NET Numerics [online]. [cit. 2021-5-12]. Dostupné z: https://numerics.mathdotnet.com/api/

[22] Wpf-Math. Github [online]. [cit. 2021-5-12]. Dostupné z: https://github.com/ForNeVeR/wpf-math

# SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

DOC Document

HTML Hypertext Markup Language

PDF Protable Document Format

UWP Universal Windows Platform

WPF Windows Presentation Foundation

XAML Extensible Application Markup Language

# SEZNAM OBRÁZKŮ

Obrázek 1. Rozdělení bezpečnostních kódů ..... 11
Obrázek 2. ASCII tabulka [1] ..... 13
Obrázek 3. Ilustrace 3 bitového kódu [3] ..... 15
Obrázek 4. Obecný komunikační systém ..... 24
Obrázek 5. Ukázka z programu – zadání polynomem [10] ..... 26
Obrázek 6. Ukázka z programu – zadání binárně [10] ..... 26
Obrázek 7. Ukázka z programu – Kódování cyklického kódu [5] ..... 27
Obrázek 8. Ukázka z programu – Dekódování cyklického kódu [5] ..... 28
Obrázek 9. Ukázka z programu – Hammingův kód [14] ..... 29
Obrázek 10. Ukázka z programu – Konvoluční kódování [14] ..... 29
Obrázek 11. Ukázka programu – Hammingův kód [15] ..... 30
Obrázek 12. Ukázka z programu – Reed-Mullerův kód [15] ..... 31
Obrázek 13. Ukázka z programu – Fireův kód [15] ..... 31
Obrázek 14. Okno programu – Hammingův kód bez chyby (výpočetní část) ..... 41
Obrázek 15. Okno programu – Hammingův kód bez chyby (popisná část) ..... 42
Obrázek 16. Okno programu – Hammingův kód s jednou chybou (výpočetní část) ..... 43
Obrázek 17. Okno programu – Hammingův kód s jednou chybou (popisná část) ..... 44
Obrázek 18. Okno programu – Hammingův kód se třemi chybami (výpočetní část) 45
Obrázek 19. Okno programu – Hammingův kód se třemi chybami (popisná část) ..... 46
Obrázek 20. Okno programu – Kontrola parity bez chyby (výpočetní část) ..... 48
Obrázek 21. Okno programu – Kontrola parity bez chyby (popisná část) ..... 49
Obrázek 22. Okno programu – Kontrola parity s jednou chybou (výpočetní část) ..... 50
Obrázek 23. Okno programu – Kontrola parity s jednou chybou (popisná část) ..... 51
Obrázek 24. Okno programu – Kontrola parity se dvěma chybami (výpočetní část) 52
Obrázek 25. Okno programu – Kontrola parity se dvěma chybami (popisná část) ..... 53

# SEZNAM TABULEK

Tabulka 1. Kombinace Hammingových kódů .....20
Tabulka 2. Exkluzivní součet - XOR.....34
Tabulka 3. Logický součin - AND.....34
Tabulka 4. Negace logického součinu NAND.....34
Tabulka 5. Logický součet OR .....35
Tabulka 6. Negace logického součtu NOR.....35

# SEZNAM PŘÍLOH

Příloha P I: Seznam příloh na cd

# PŘÍLOHA P I: SEZNAM PŘÍLOH NA CD

fulltext.pdf – plný text bakalářské práce

prilohy.zip – adresář obsahující exe soubor ke spuštění