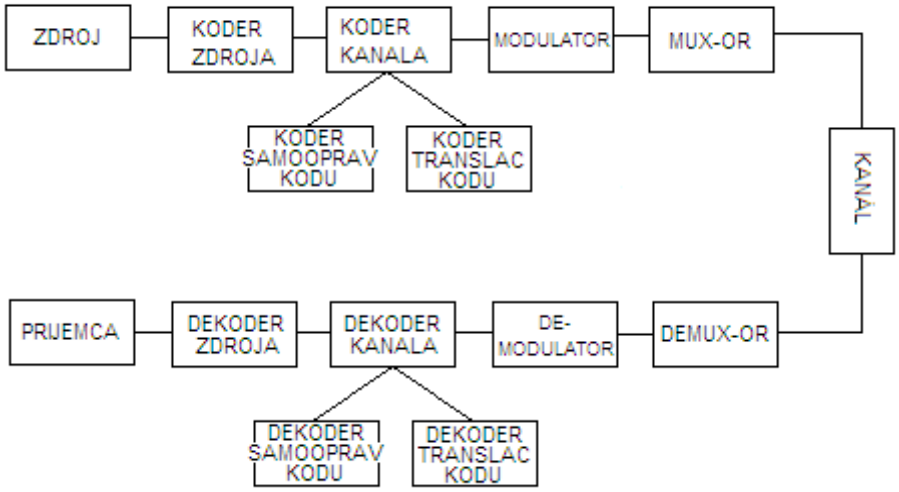
Digitálne komunikácie

1. Model digitálneho systému a funkcie jeho jednotlivých podsystémov a nástroje na opis a analýzu signálov.

1.1 Model digitálneho systému a funkcie jeho jednotlivých podsystémov



Zdroj – generuje správy vo forme slov nad zdrojovou abecedou.
Kóder zdroja – má za úlohu znížiť redundanciu zdroja v tom zmysle, že slovám s najvyššou pravdepodobnosťou sa priraďujú najkratšie kódové slová (nerovnomerné kódy Shannon-Fanov, Huffmanov).*Vyjadrenie informácie čo najúspornejšie*.
Kóder kanála – má za úlohu ochrániť informáciu voči chybám a prispôsobiť ju kanálu.

a) Kóder samoopravného kódu – využíva blokové kódy (RS, lineárne, hammingove...) na zakódovanie informácie, pričom vkladá redundanciu, na základe ktorej možno v dekodéri detekovať, opraviť chyby.

b) Kóder translačného kódu – úlohou je vylúčiť zo zakódovanej postupnosti symbolov také sledy symbolov, ktoré sú pre daný kanál nevhodné.

Modulátor – úlohou digitálneho modulátora (keďže máme digitálny prenosový systém) je preniesť tok bitov cez analógový kanál. Pri digitálnej modulácii je analógový nosný signál modulovaný tokom bitov (z kódera). Ide teda o digitálne-analógovú konverziu. Zmeny v nosnom signáli sú vyberané z konečnej M-prvkovej množiny modulačnej abecedy.
Multiplexor – je prítomný za účelom zdieľania prenosového média. Kombinuje viacero vstupných signálov do jedného výstupného toku.
Prenosový kanál – prenosové médium „vyzbrojené“ rušivými vplyvmi. Tiež sa naň možno na digitálnej úrovni pozerať ako na zdroj správ, ale chybových.
Ostatné bloky sú recipročnou analógiou (okrem príjemcu, ten nič negeneruje, akurát tak môže generovať pocity v závislosti od spokojnosti s prenosom :-)).

1.2 Nástroje na opis a analýzu signálov

Delenie signálov:

- **Deterministické** (vieme určiť v každom t)**:**
  - **1) periodické** $x(t) = x(t+kT),\ T \to \infty$
    - **a) harmonické** (**Fourrierov rad -**spektrum je jedna čiara, konečný počet harmon. zložiek)
    - **b) neharmonické (Fourrierov rad -** spektrum je čiarové, ∞

počet harmon. zložiek)

2) neperiodické $x(t) \neq x(t+kT), T \rightarrow \infty$
(Fourrierova transformácia – spojité spektrum**)**
3) kváziperiodické (FR)

- **Stochastické –** (náhodné) popisujeme štatistickými veličinami, má informačnú hodnotu

2. Linkové signály používané v praxi a ich základné vlastnosti

Sú to binárne kódy, ktoré sa používajú na prenos informácií medzi zariadeniami. V záležitosti od požiadaviek prenosu, možnosti kanálu a ďalších parametrov potrebných na prenos signálu sa používajú viaceré druhy linkových kódov.

Linkové signály: 1) empirický prístup
2) modernejší (teória)

1) požiadavky na linkové kódy
- min. jednosmerná zložka
- vhodné frekvenčné pásmo ->prispôsobenie signálu ku kanálu
- P_B min $f(E_B/N_0) \rightarrow E_B$ max.
- podpora synchronizácie v prijímači
- detekčné a korekčné vlastnosti
- jednoduchosť

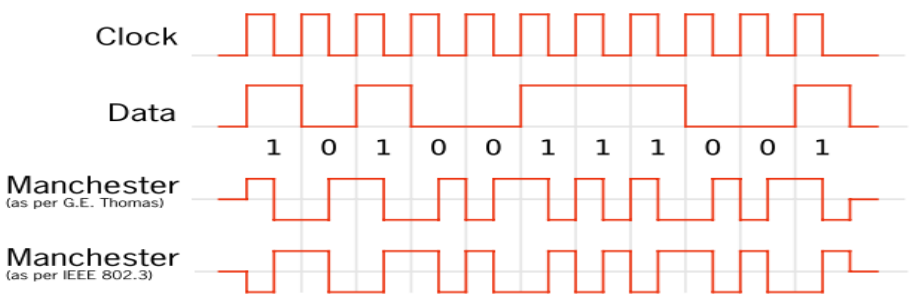
Základné rozdelenie linkových kódov 2):

-	**UP (unipolar)**	**BP (bipolar)**
RZ (return-to-zero)	UPRZ	BPRZ
NRZ (non-RZ)	UPNRZ	BPNRZ

RZ – signál nezaberá celý char. interval
NRZ – zaberá cely char. interval
BP – dá sa dosiahnuť nulová jedn. zložka
UP – nedá sa dosiahnuť nulová jedn. zložka

Ďalšie typy linkových kódov:

AMI (Alternate Mark Inversion) – strieda sa polarita jednotiek; jednotka strieda polaritu, nula ostáva nulov. Prevencia výraznej jednosmernej zložky. Má vždy nulovú jednos. zložku.
Manchester – signál s návratom k nule - RZ



HDBn (High Density Bipolar) – BPNRZ AMI,
n – označuje max dĺžku behu (run-n), sled 1-tiek a 0-úl ktoré nasledujú za sebou,
n→ max počet intervalov kt. sa môžu vyskytnúť v sig. bez prechodu medzi úrovňami,
BnZS (Bipolar with n Zeros Substitution) – substitučný translačný kód, je to špeciálny prípad HDBn kódov, napr. B3ZS

Original data:

- 0 0 0 0 + 0 0 0 - + 0 0 0 0 -

After Substitution:

- 0 0 - 0 + 0 0 + - + - 0 - 0 -

Previous Mark Polarity	Number of marks since the last substitution	
	Odd	Even
-	0 0 -	+ 0 +
+	0 0 +	- 0 -

HDBn – High Density Binary
RDS – Running Digital Sum
LDS – Level Digital Disparita

3. Kanály s obmedzeniami a konštrukcia translačných kódov (TK)

Úlohou translačných kódov je vylúčiť zo zakódovanej postupnosti symbolov určité sledy symbolov, ktoré sú pre daný kanál nevhodné, pričom do kódera translačného kódu môžu vstupovať ľubovoľné sledy symbolov.
Najznámejšie - Linkové kódy, **HDBn, BnZS.**

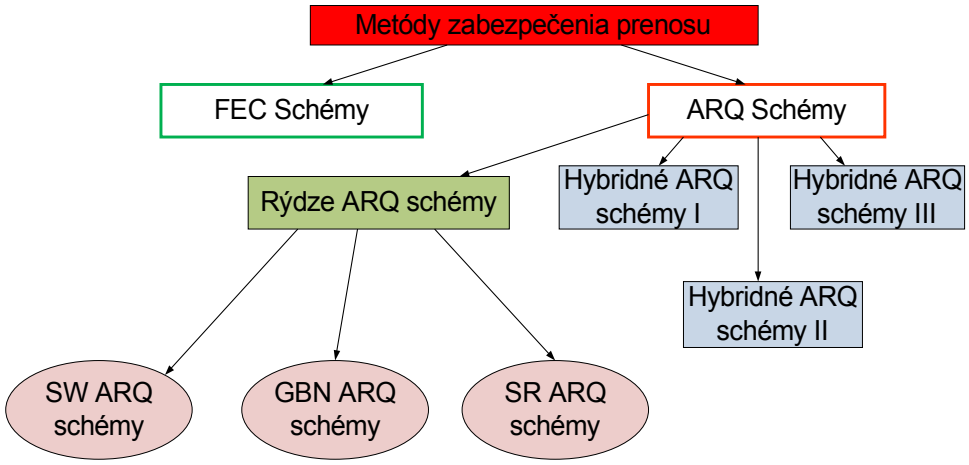
Čo sa týka konštrukcie, viď príklad B3ZS v predchádzajúcej kapitole.
Translačné kódy sa používajú, keď je potrebné predísť n-násobnému opakovaniu rovnakých symbolov, ďalej pre potreby lepšej synchronizácie či odstránenia jednosmernej zložky a na zakódovanie signálu aby sa mohol preniesť cez kanál s obmedzeniami.

Pri TK uvažujeme, že kanál je bez šumu (nie sú chyby, $P_B \rightarrow 0$) a pamäti (jednotlivé symboly nezávisia na predchádzajúcich signálov).
Kanál s obmedzeniami → max dĺžka behu.

Špecifikácia TK:
- slovne
- stavový diagram (opisuje, čo sa mohlo stať v minulosti)
- mriežka
- B matica (matica susednosti)

4. Zvyšovanie spoľahlivosti pomocou metód kontrolujúcich chyby ARQ, FEC

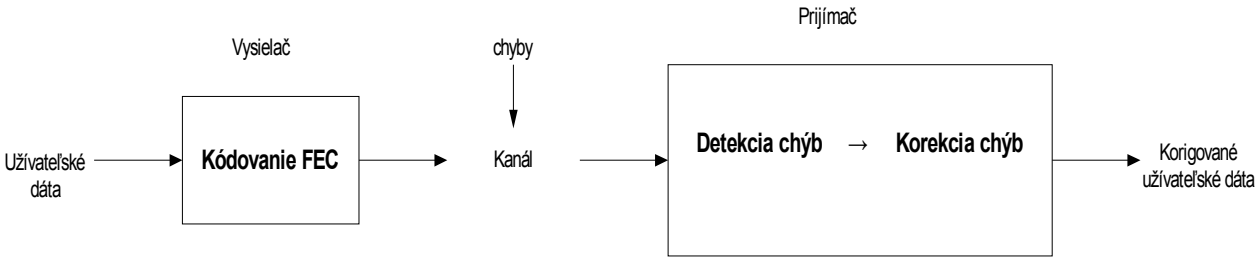
Žiaden reálny prenosový systém nedokáže zabezpečiť bezchybný prenos dát, preto sa snažíme chyby v prenose eliminovať pomocou metód riadenia zabezpečenia prenosu.



Rozdelenie:
FEC (Forward Error Correction - dopredná oprava chýb)
ARQ (Automatic Repeat Request - automatická žiadosť o opakovanie)
Dá sa povedať, že FEC metódy nevyžadujú spätný kanál. Majú konštantnú relatívnu priepustnosť (priepustnosť sa rovná rýchlosti kódu) a je pri nich ťažké dosiahnuť vysokú spoľahlivosť. ARQ metódy naopak vyžadujú spätný kanál, priepustnosť prudko klesá s narastajúcou chybovosťou a sú to vysoko spoľahlivé metódy. Najmä v dátových sieťach sú preto ARQ často uprednostňované pred FEC metódami.

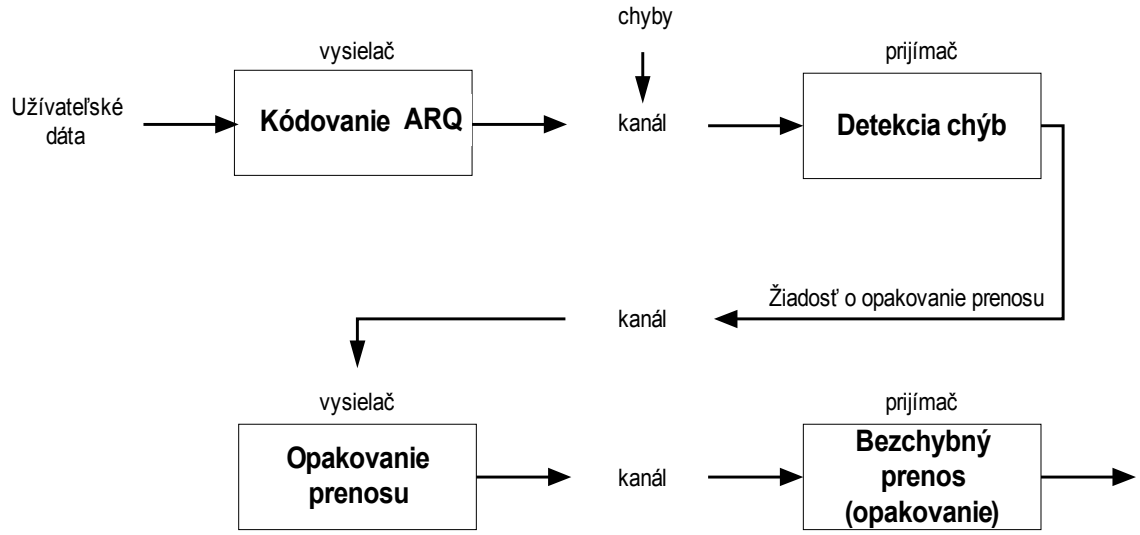
FEC metódy

Metódy, ktorých základný princíp spočíva v pridávaní redundantných bitov k prenášanému signálu. Tieto redundantné bity sa využívajú na detekciu, ale aj na korekciu poškodených bitov.



ARQ metódy

Metódy, ktoré využívajú na riadenie zabezpečenia prenosu dát spätný kanál. Sú založené na princípe, že pri vyskytnutí chyby v prenose je na strane prijímača chyba detekovaná a v prípade chyby je vyžiadané zopakovať dáta. Snaha je dosiahnuť chybovosť 10^{-9} až 10^{-10}.



Hybridné ARQ schémy

V princípe sa líšia od rýdzich ARQ schém v dvoch veciach:

- namiesto detekčných kódov používajú samoopravné kódy,
- majú prepracovanejšiu stratégiu opakovania prenosu chybných blokov.

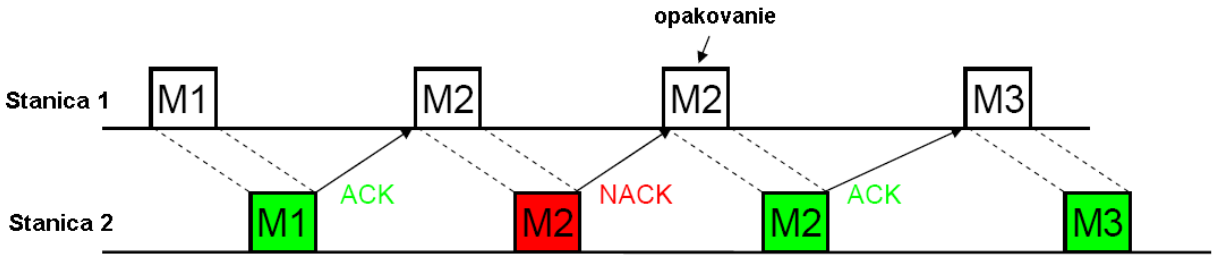
Schémy potvrdenia:
N-schéma – po prijatí NAK sa opakuje vysielanie bloku (NAK – negatívne potvrdenie)
P-schéma – po prijatí ACK sa vysiela ďalší blok
A-schéma - po prijatí ACK sa vysiela ďalší blok, po prijatí NAK sa opakuje vysielanie, po uplynutí intervalu sa opakuje vysielanie
Nie je celkom výhodné, dlho sa čaká, vysielač je neefektívne využitý.

Rýdze ARQ metódy

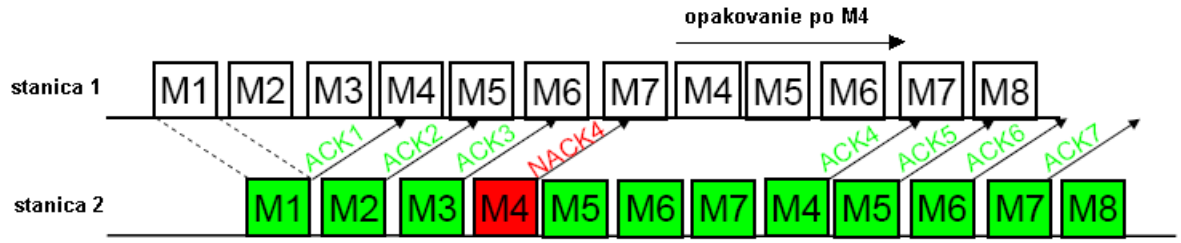
Sú charakteristické tým, že na strane príjemcu sa chyby len detekujú a v prípade zaznamenania chyby prijímač požiada znovu o vyslanie príslušného dátového bloku.
Princíp činnosti rýdzich ARQ metód:

- vysielač vyšle kódovaný blok a čaká na potvrdenie od prijímača,
- ak prijímač zistí chybu v prijatom bloku, vyžiada si od vysielača opakovanie bloku,
- v prípade opakovaného vysielania bloku vysielač vyšle presne rovnaký blok ako bol predchádzajúci,
- používajú sa len detekčné kódy na zistenie správnosti prijatého bloku, preto je realizácia týchto schém jednoduchá.

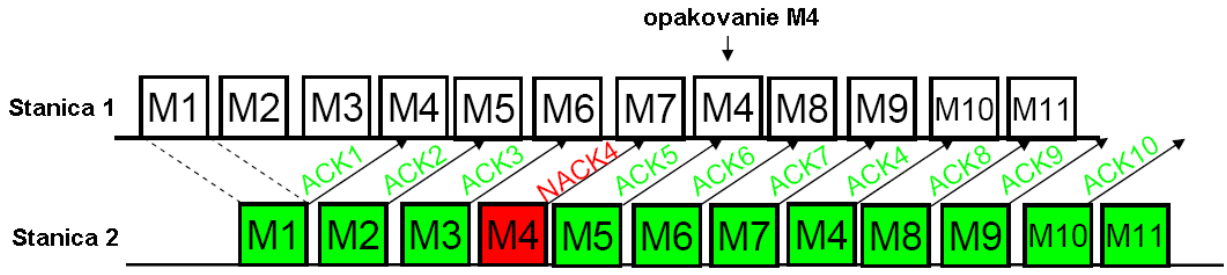
Rýdze ARQ metódy sa rozdeľujú do troch základných skupín, ktoré sa navzájom líšia stratégiou opakovania prenosu chybných blokov :
Send-and-Wait (S&W) schéma – ARQ s prerušovaným prenosom
Systémy využívajúce S&W schému fungujú nasledovne - po vyslaní správy sa ďalšie vysielanie zastaví a čaká sa na príchod pozitívneho potvrdenia správneho prijatia ACK (positive acknowledgement), alebo prijatia negatívneho potvrdenia NAK (negative acknowledgement), ktoré indikuje nesprávny prenos a vysielač posiela blok znova. Opakované vysielanie trvá až do prijatia ACK.



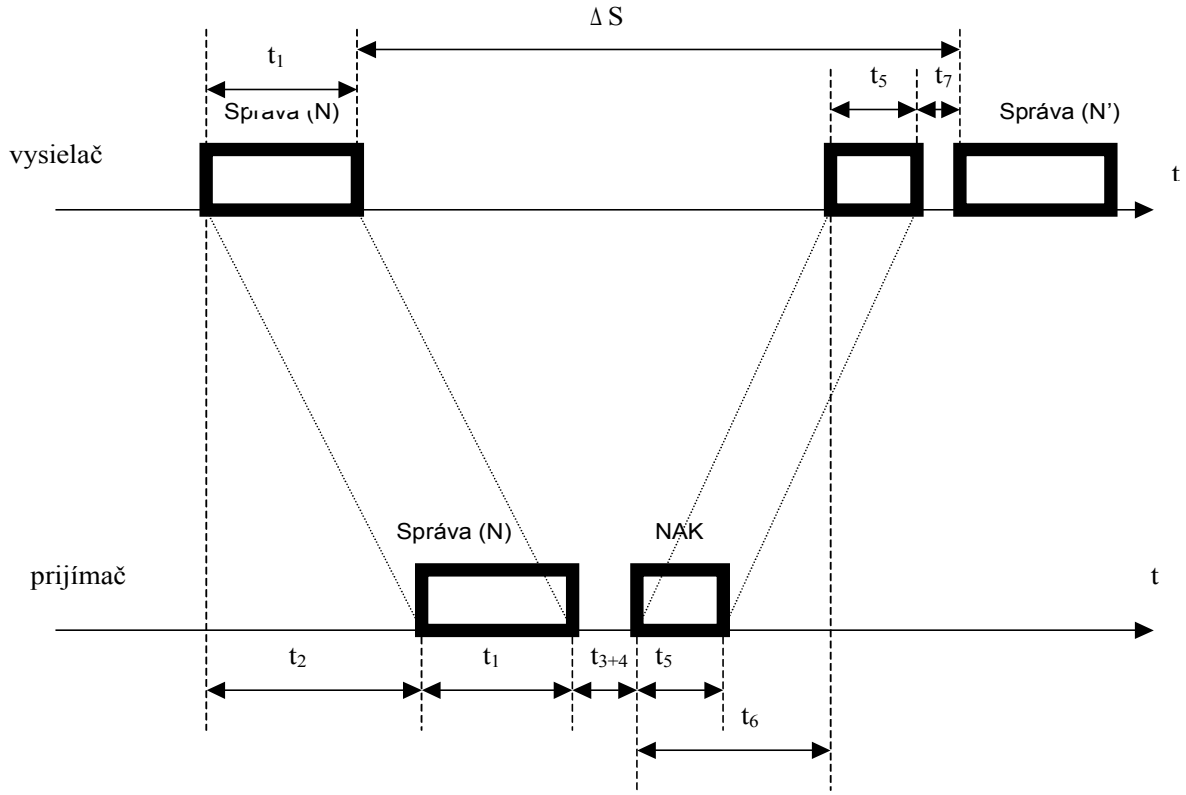
Go-Back-N (GBN) schéma – ARQ s kontinuálnym prenosom
Vznikla na základe idey využiť nevyužitý čas čakania na potvrdenie pri S&W schéme na vysielanie nasledujúcich blokov dát. Princíp je nasledovný: vysielač nepretržite vysiela bloky dát. Ak prijímač prijme niektorý blok chybne, pošle vysielaču NAK. Vysielač preruší vysielanie, vráti sa o N blokov späť k chybne prijatej správe a od tejto správy začne znovu vysielať všetky nasledujúce správy znova bez ohľadu na to, či niektorá z nich bola správne alebo nesprávne prijatá. Koncepcia GBN ARQ nevyžaduje na strane prijímača vyrovnávacie pamäte, pretože prijímač „zahodí“ všetky bloky ktoré mu prišli po chybnom bloku a čaká na ne znova. Táto schéma vzhľadom k svojej jednoduchosti a zároveň účinnosti je najviac používanou v bezdrôtových a satelitných, ale aj v iných systémoch.



Selective Repeat (SR) schéma – ARQ so selektívnym opakovaním
Keďže je opakovanie správne prenesených blokov pri GBN schéme principiálne zbytočné, pri SR schéme sa nerobí. Princíp je nasledovný - vysielač opakuje len chybne prijatý blok. Po jeho opätovnom vyslaní pokračuje vo vysielaní blokov tam, kde prestal. Aby bol prenos možný, je potrebné, aby prijímač mal na svojej strane vyrovnávaciu pamäť. Uchováva si všetky bezchybné bloky, a po prijatí opakovaných blokov ich zaradí na správne miesto. Táto schéma by vyžadovala nekonečne veľkú vyrovnávaciu pamäť – čo nie je možné, preto v praxi bývajú použité SR schémy s konečnou vyrovnávacou pamäťou a teda už nemajú ani prenosové vlastnosti Ideálnej SR schémy.



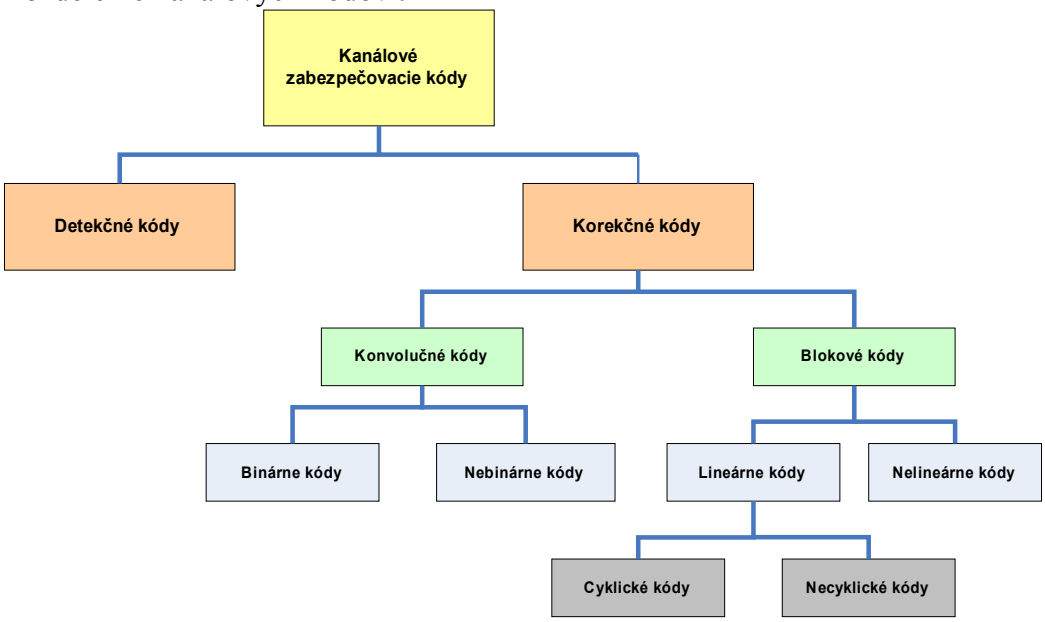
ARQ stratégie



t_1 – čas vysielania bloku
t_2 – oneskorenie prenosu V→P
t_{3+4} – spracovanie a vyhodnotenie bloku
t_5 –čas vysielania potvrdenia (ACK,NACK)
t_6 – oneskorenie prenosu P→V
t_7 – čas spracovania potvrdenia
ΔS – oneskorenie retransmisie

5. Lineárne binárne blokové kódy kontrolujúce chyby

Rozdelenie kanálových kódov :

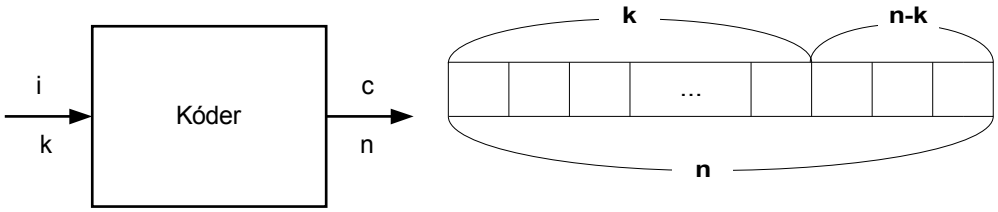


,

Blokové kódovanie

-*blokový* → každé blokové slová majú rovnakú dĺžku
- postupnosť symbolov vstupujúca do kódera sa delí na bloky po k bitoch a tieto sa kódujú na kódové slová po n bitov
- označujú sa BC(n,k), alebo podrobnejším označením (n,k,d_{min}), pričom n>k

k.....počet informačných bitov (dĺžka vstupného informačného slova i)
r = n-k...počet kontrolných bitov
n.....dĺžka výstupného kódového bloku c
d_{min}.....minimálna Hammingova vzdialenosť kódových blokov



Charakteristika BK

- istý aktuálny výstupný kódový blok je odvodený od jediného jemu odpovedajúceho vstupného bloku dát ⟹ blokový kóder nemá pamäť ⟹ minimálne oneskorenie kóderu je rovné dobe trvania bloku dát T_d a ak k tomuto času pridáme ešte dobu elektronických kódovacích obvodov T_p dostaneme celkové oneskorenie $T_{dtot}=T_d+T_p$
- Hammingova vzdialenosť 2 slov d - najmenší počet rozdielnych bitov vo dvoch rovnako dlhých kódových slov, teda počet miest, kde majú slová rozdielne symboly
    - Pr. c_1=0101010
    - c_2=1111000 $d(c_1,c_2)=3$
- minimálna Hammingova vzdialenosť kódu d_{min} – minimálna vzdialenosť zo všetkých dvojíc
- korekčná schopnosť BC – maximálny počet chýb t, ktoré je blokový kód schopný opraviť

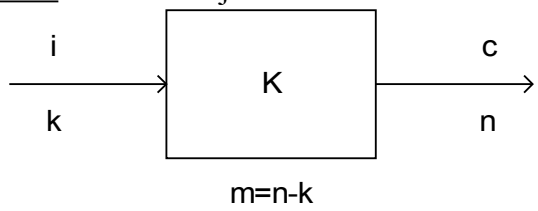
$$t \prec \frac{d_{\min}}{2}$$

    -
- Rýchlosť kódu $r = \frac{k}{n}$
    - Príklad: ak r=0,5 potom každý bit nesie 0,5 bitu informácie
- Pretože n>k prenášaná postupnosť obsahuje redundantné symboly, ktoré nenesú žiadnu informáciu - redundancia (nadbytočnosť) kódu $\frac{n-k}{k}$
- Hammingova váha kódového slova w - je daná jej vzdialenosťou od nulovej značky a u binárneho kódu počtom symbolov “1“

- Pr. $w(c_1)=3$
  - $w(c_2)=4$

Lineárne blokové kódy

Lineárny kód → súčetom dvoch kódových slov je tretie kódové slovo.
LBK → slová majú rovnakú dĺžku.



LBK je k-rozmerný podpriestor lineárneho n-rozmerného prostoru nad konečných poľom GF(2). $GF(2) => g \in \{0,1\}$
kód: (n, k, d_{min})
i – informačné slovo dĺžky k
c – kódové slovo dĺžky n
d_{min} – v G matici nájdeme riadok s najmenším počtom jednotiek a dostaneme d_{min}
w – váha kódového slova – počet nenulových stavov
$w(c_1) = w(0101010) = 3$
$w(c_2) = w(1111000) = 4$
d – vzdialenosť dvoch kódových slov – počet stavov kde sa líšia
$d(c_1, c_2) = w(c_1 \oplus c_2) = w(1010010) = 3$

$$c_1 \oplus c_2 = \begin{array}{r} 0101010 \\ 1111000 \\ 1010010 \end{array}$$

Ak chceme opraviť t-chýb tak potom musí platiť:

$$d_{min} \leq 2t + 1$$

Ak chceme detekovať chybu, tak potom musí platiť:

$$d_{min} \leq t_D + 1$$

G – generujúca matica kódu a určuje nám kód
VLASTNOSTI:

1) Súčet všetkých kódových slov sa rovná 0

$$\sum_{i=0}^{15} c_i = [0000000]$$

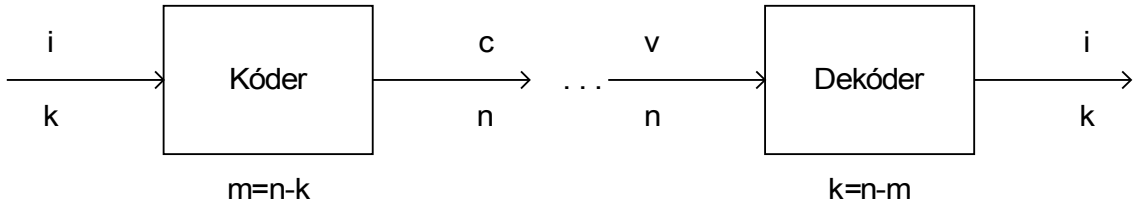
2) **Linearita** – súčet dvoch kódových slov vám dá tretie kódove slovo
3) $d_{min} = 3$

Kódovanie lineárnym blokovým kódom

- informačné slovo dĺžky k vynásobíme generujúcou maticou o rozmeroch $k \times n$ a dostaneme kódové slovo dĺžky n

$$i_k . G_{n \times k} = c_n$$

Dekódovanie lineárnym blokovým kódom



v – prijaté kódové slovo
H – kontrolná matica
H^T – transponovaná kontrolná matica
$I_{k \times k}$ – jednotková matica
$\bar{s}$ – syndróm, ak je rovný nule, tak chyba nenastala, alebo nie je detekovateľná, ak je rôzny od nuly, tak nastala chyba (opravým pomocou syndrómovej tabuľky)

$$v . H^T = \bar{s}$$

$$G_{k \times n} = [I_{k \times k} | P_{k \times m}]$$

$$H_{m \times n} = [P^T_{m \times k} | I_{m \times m}] => H^T_{n \times m}$$

Hamingov kód

- blokový korekčný kód
- opravuje vždy iba jednu chyby => $d_{min}=3$

$$d_{min} \leq 2t + 1$$

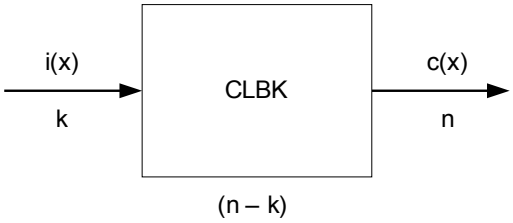
t – počet chýb ktoré dokáže opraviť

6. Cyklické kódy

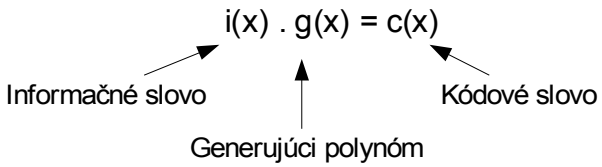
- podmnožina lineárnych blokových kódov
- lineárny kód je cyklickým kódom ak platí, že každé kódové slovo keď posunieme, dostaneme tiež kódové slovo daného kódu:
- generujeme ich pomocou generujúceho polynómu g(x)
- g(x) cyklického kódu (n,k) je polynóm stupňa r = (n - k) a ($x^n + 1$) je bez zvyšku deliteľný g(x)

$$g(x) = g_0 + g_1 . x + g_2 . x^2 + \ldots + g_r . x^r$$

Kódovanie



- nesystematický tvar – nevieme priamo zistiť informačné symboly



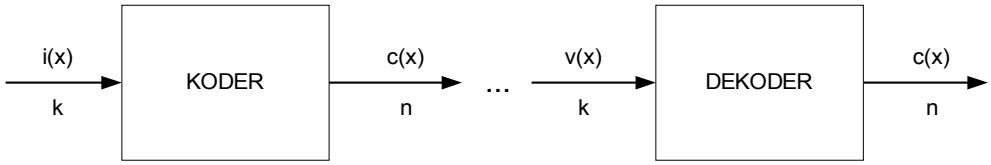
- systematický tvar – efektívnejší spôsob kódovania, kde samotná kódovaná správa je súčasťou zakódovaného kódového slova

1. $i(x).x^{\deg[g(x)]} = i(x).x^{n-k}$ deg|g(x)| - stupeň poly.
2. $i(x).x^{n-k} . \mod g(x) = Q(x)$
3. $c(x) = i(x).x^{n-k} + Q(x)$

kódové slovo: $c = (n-k)$paritných bitov + kbitov správy

Dekódovanie

– využíva sa syndrómová metóda



v(x).....polynóm prijatého kódového slova
e(x).....polynóm opisujúci chybové slovo

$v(x).\mod g(x) = s(x)$

syndróm $s(x) \rightarrow s(x) = 0 \rightarrow$ priate slovo v(x) je správne

$s(x) \rightarrow s(x) \neq 0 \rightarrow$ pozriem syndrómovú tabuľku $\rightarrow$

$s(x) = e(x).\mod g(x)$

$c(x) = v(x) + e(x)$

7. Nebinárne blokové kódy

Reed-Solomonove kódy

RS kódy sú aparátom využívaným pri prenosoch informácií kanálom na doprednú ochranu

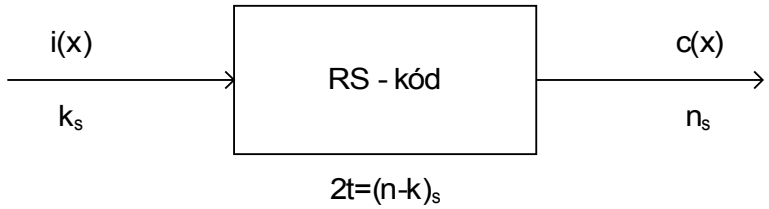
údajov (FEC). Sú to blokové kódy, ktoré pridaním nadbytočnosti – ochranných bitov vypočítaných algoritmom, ochránia prenášanú informáciu. Ich výhodou je, že dokážu opraviť až niekoľko chybných prvkov prenášaných za sebou, bez potreby dodatočného premiešania bitov. Pričom sú to nebinárne kódy, kde prvok je hneď množina bitov, podľa typu použitého RS kódu. [http://skola.durcik.sk/bc/rskody/index.php?section=rsinfo]

Použitie:

- pre veľké abecedy
- satelitné a mobilné komunikácie
- DVD
- ochrana uložených údajov
- vysokorýchlostné modemy

- sú podmnožinou cyklických lineárnych blokových kódov
- Reed Solomonové kódy zostrojené nad konečným poľom $GF(q) = GF(2^s)$

$p(x) = x^s + \cdots + 1 \quad \alpha^i = x^i mod\, p(x)$



Špecifikácia – RS (n,k,d)
1symbol = s bitov
t – počet chýb, ktoré vie daný kód opraviť $2t = (n - k)$
dĺžka kódového slova $n = q - 1 = 2^s - 1$
$2t = n - k$
$t = \frac{n-k}{2} => d_{min} = 2t + 1 = n - k + 1$
$g(x) = (x + \alpha^j).(x + \alpha^{j+1}) \ldots (x + \alpha^{j+(2t-1)})$
keď $j = 0$ potom:
$g(x) = \underbrace{(x + \alpha^0).(x + \alpha^1).(x + \alpha^2) \ldots (x + \alpha^{2t-1})}_{2t\ zátvoriek}$

RS kód je taký kód, ktorého každé kódové slovo má 2t za sebou idúcich prvkov konečného poľa a to svoj koreň.
Keď donútime v RS kóde mať 2t za sebou idúcich koreňov pre všetky kódové slová, zaručene dokážeme opraviť 2t chýb.

Kódovanie RS kódom

a) Nesystematické

$c(x) = i(x).g(x)$

b) Systematické

$c(x) = i(x).x^{n-k} + i(x).x^{n-k} mod\, g(x)$

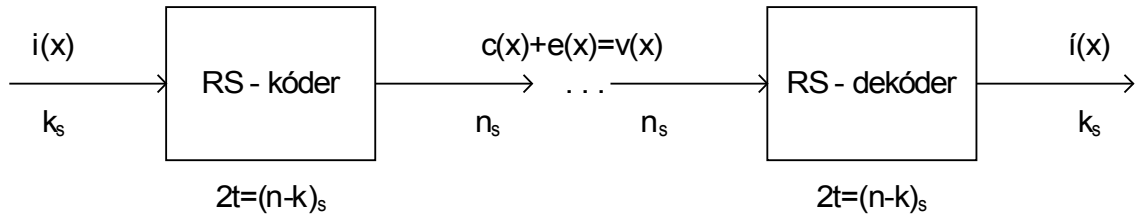
c) Systematické pomocou rovníc

$c(x) = c_{n-1}.x^{n-1} + c_{n-2}.x^{n-2} + \cdots + c_1.x^1 + c_0 \qquad c = \alpha^j$
$c(x \to \alpha^0) = \cdots = 0$

$c(x \to \alpha^1) = \cdots = 0$

riešieme sústavu.

Dekódovanie RS kódom



-dekódovanie vychádza zo syndrómových rovníc:

$$s_k = \sum_{i=1}^{t} Y_i . X_i^k; \quad k = 0,1, \dots, 2t-1$$

X_i – lokátor i-tej chyby (hľadá **polohu** chyby), jeho exponent mu určuje polohu chyby vo v(x)
Y_i – **hodnota**, magnitúda i-tej chyby
(RS dokáže zistiť polohu chyby a jej hodnotu.)

1) VÝPOČET SYNDRÓMOV

-do prijatého kódového slova budeme dosadzovať α^k

$s_k = v(x \to \alpha^k)$

-ak všetky s_k=0 tak potom chyba nenastala

2) NÁJDENIE POLYNÓMU LOKÁTOROV

$\sigma(x) = x^t + \sigma_1 x^{t-1} + \sigma_2 x^{t-2} + \cdots + \sigma_{t-1} x^1 + \sigma_t$

neznáme σ vypočítame zo sústavy syndromových rovníc:

$$s_{t+i} + \sum_{j=1}^{t} s_{t+i-j} . \sigma_j = 0; \qquad i = 0, 1, \dots, t-1$$

3) HĽADANIE KOREŇOV POLYNÓMU LOKÁTOROV

-spočíva v Chienovom algoritme, ktorý poslúži na hľadanie prvkov poľa
-do polynómu lokátorov sa dosadí α, keď pri nej vyjde rovnica rovná nule daná alfa je koreňom X

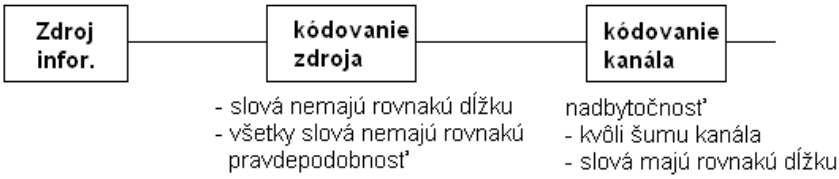
4) VÝPOČET HODNÔT CHÝB

$$s_k = \sum_{i=1}^{t} Y_i . X_i^k; \quad k = 0,1, \dots, 2t-1$$

$c(x) = v(x) + e(x)$

Záver:
RS môžeme n max. toľko, koľko je nenulových prvkov poľa. Pretože polohu chyby musíme určiť pomocou mocniny α. Ak výdu oba syndrómy nulové → buď nenastala chyba alebo je taká, že vzniklo ďalšie slovo a to sa nedá zistiť.

8. Opis zdroja informácie



Základné princípy kódovanie zdroja

Kóder zdroja realizuje proces zdrojového kódovania. Na jeho vstup prichádzajú digitalizované dáta (ak sú analógové tak ich treba digitalizovať) z diskrétneho zdroja bez pamäti v podobe sekvencie binárnych symbolov (kódových slov) s_k,
Zdrojový kóder musí spĺňať niekoľko podmienok. Jeho vstupné kódové slová by mali mať binárnu podobu. Dekódovanie sa musí uskutočňovať jednoznačne (zo zakódovanej sekvencie b_k sa musí dať dekódovať pôvodná sekvencia s_k). Jednou zo základných funkcií kódera zdroja je potláčanie redundancie a nepodstatnosti obsiahnutých v prenášanom signály, čo potom vedie k redukcií bitovej rýchlosti prenášaného signálu.
Dĺžku kódového slova s_k v bitoch označujeme ako l_k. Potom môžeme definovať *strednú dĺžku kódového slova zdrojového kódu* ako $\bar{L}$ vzťahom:

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k \qquad k = 0,1,\dots,K-1$$

Takto definovaný parameter $\bar{L}$ predstavuje priemerný počet bitov pripadajúcich na jeden zdrojový symbol l_k. Keď L_{min} bude značiť minimálnu možnú hodnotu parametra $\bar{L}$, potom účinnosť kódovania zdrojového kódera je:

$$\eta = \frac{L_{min}}{\bar{L}}$$

Keďže $L_{min} \leq \bar{L}$ a účinnosť je $\eta \leq 1$, potom je zdrojový kóder tým efektívnejší, čím je jeho účinnosť bližšie k jednotke.

Diskrétne zdroje

Diskrétne zdroje generujú postupnosť diskrétnych symbolov **x(kT)**, vybraných z abecedy zdroja diskrétnych v čase i hodnote. Ak abeceda zdroja obsahuje konečný počet symbolov **N**, zdroj sa nazýva konečný diskrétny zdroj.
Ak poznáme pravdepodobnosť každého symbolu $\mathbf{X_j}$ z abecedy, $\mathbf{P(X_j) = p_j}$, potom vieme určiť množstvo informácie $\mathbf{I(X_j)}$ pre každý symbol v abecede.

$$\mathbf{I}(X_j) = -\log_2 p_j \text{ [bit]}$$

Priemerné množstvo informácie pripadajúce na jeden symbol v abecede sa označuje **H(X)** a nazýva sa entropia zdroja. Z hľadiska teórie informácií je entropia jednou zo základných charakteristík zdroja.

$$\mathbf{H(X)} = -\sum_{j=1}^{N} p_j \log_2 p_j \text{ [bit/symbol]}$$

Entropia zdroja je definovaná ako priemerné množstvo informácie pre výstup zdroja alebo je považovaná za priemerné množstvo neurčitosti. Vo všeobecnosti platí, že množstvo informácie v bite na symbol je ohraničené zospodu nulou, $H(X) \geq 0$. Entropia dosahuje maximum ak sú symboly postupnosti nezávislé a vyskytujú sa s rovnakou pravdepodobnosťou $p_j = 1/N$.

$$0 \leq H(X) \leq \log_2 N$$

Najjednoduchším príkladom zdroja je binárny zdroj bez pamäti, ktorý produkuje štatisticky

nezávislé symboly 0 a 1. Symbol 1 produkuje s pravdepodobnosťou P a symbol 0 s pravdepodobnosťou 1-P. Pre j = 2 a N = 1, potom entropia zdroja závisí len od P:

$$H(X) = \sum_{j=1}^{2} P_i \log(P_i)$$
$$_i$$

9. Bezstratové a stratové metódy kompresie ***toto uz nemame zadane***

Kódovanie:

- kompresia → stratové kódovanie (so stratami), znižuje nadbytočnosť a nepodstatnosť
- kompakcia → bezstratové kódovanie, znižuje nadbytočnosť

Kompakčné kódy sú určené na zhustené vyjadrenie informácie z diskrétnych zdrojov.

Huffmanov kód – komakčné kódovanie

H je prefixný kód → žiadne kódové slovo nezačína ako predošlé a nie je predponou iného kódového slova.
Huffmanov kód patrí medzi nerovnomerné kódy. Základná myšlienka Huffmanovho kódu je nasledovná: kódovať znaky s vysokou pravdepodbnosťou výskytu pomocou kratších reťazcov a znaky s nízkou pravdepodobnosťou výskytu pomocou dlhších reťazcov znakov kódovej abecedy (kódových slov).

1) správy zoradíme podľa klesajúcej pravdepodobnosti
2) dve správy (lebo L=2) s najnižľou pravdepodobnosťou zlúčime, pričom im priradíme symboly 0 a 1 (0 pre správu s nižšou pravdepodobnosťou), nova zlúčená správa bude mať pravdepodobnosť rovnú súčtu pôvodných správ
3) takto zredukované správy znova usporiadame a opakujeme krok 2, až kým nezlúčime všetky správy.

Shannon-Fannov kód

- S-F je prefixný kód predponou iného kódového slova
- v praxi je Huffman účinnejší ako S-F
- zohľadnuje periodicitu výskytu – kompakčný kód

1) správy zoradíme podľa klesajúcej pravdepodobnosti
2) rozdelíme ich do dvoch skupín (lebo L=2), tak aby v každej skupine bol súčet pravdepodobností najrovnomernejšie rozdelený (pól na pól)
3) k prvej skupine priradíme každej správe symbol 0 a druhej skupine symbol 1
4) postup opakujeme v skupinách až pokiaľ nedostaneme skupiny o jednej správe

Na rozdiel od Huffmanovho kódu má menšiu účinnosť a jeho výhodou je väčšia rýchlosť a ľahšia implementovateľnosť. (STRANKA WWW:PROJECTS.HUDECOF.NET)

Ziv-Lempelové kódovanie

- kompakčný kóder (bez straty)

- myšlienka algoritmu je v hľadaní podreťazcov v posledných w prečítaných znakoch, ktoré sú zároveň prefixom nasledujúcich p znakoch.

a) Slovníková metóda

- princípom kódovania je vytváranie slovníka priebežne s prijímanými dátami
- štatistika výskytu správ sa nevykonáva

$$kódové\ slovo = adresa\ v\ slovníku + inovačný\ symbol$$

b) Baffrova metóda

- pre svoju činnosť potrebuje dočasnú pamäť (n), ktorá je rozdelená na dve rovnaké časti. Prvú časť naplníme nulami, do druhej časti sa načítajú informačné symboly. Kódovanie musí začať na začiatku druhej časti. Princíp kódovania je hľadanie čo najdlhšej postupnosti našich informačných symbolov v prvej časti pamäti, pričom kódové slovo bude obsahovať tri položky:

  1) adresu tejto postupnosti v prvej časti pamäti
  2) jej dĺžku (ktorá sa rovná)
  3) informačný (nasledujúci) symbol

$$kódové\ slovo = adresa + dĺžka + inovačný\ symbol$$

- najviac môžeme porovnávať

$$\left(\frac{n}{2} - 1\right)$$