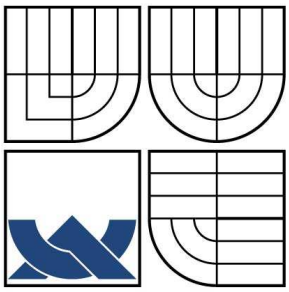
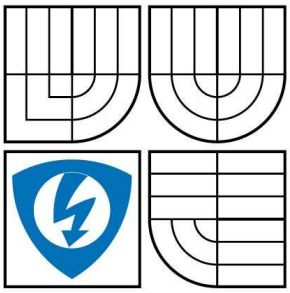


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKACNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

PROTICHYBOVÉ SYSTÉMY VYUŽÍVAJÍCÍ BLOKOVÉ KÓDY

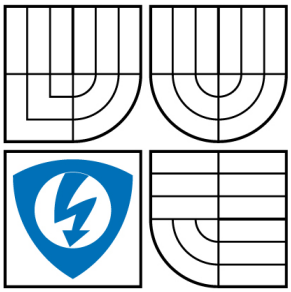
ANTI-ERROR CODING SYSTEMS WITH BLOCK CODES

BAKALÁŘSKÁ PRÁCE
BACHELOR THESIS

AUTOR PRÁCE — JAKUB PACHER
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. KAREL NĚMEC, CSc.
SUPERVISOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Pacher Jakub	***ID:***	78392
Ročník:	3	***Akademický rok:***	2007/2008

NÁZEV TÉMATU:

Protichybové systémy využívající blokové kódy

POKYNY PRO VYPRACOVÁNÍ:

Vypracujte návrh protichybového kodeku, který zabezpečí digitální přenos dat proti shlukujícím se chybám pomocí blokového korekčního kódu. Distribuce chyb v kanálu zadána takto: Bitová chybovost pb = 2,1 .10-6 chyb/bit, délka shluku chyb b je menší nebo rovna 22 bitů, délka bezchybného intervalu A je větší nebo rovna 1800 bitů. V návrhu respektujte skutečnost, že protichybový kodek je součástí protichybového kódového systému. Ověřte funkční schopnosti tohoto protichybového kodeku metodou, kterou považujete pro tento návrh za nejvhodnější.

DOPORUČENÁ LITERATURA:

[1] ADÁMEK,J. Kódování. Nakladatelství technické literatury, Praha 1989.

[2] LEE, L.H.CH. Error-Control Block Codes for communications Engineers. Artech House, Boston, London, ISBN 1-58053-032-X.

Termín zadání:	11.2.2008	***Termín odevzdání:***	4.6.2008

Vedoucí práce: doc. Ing. Karel Němec, CSc.

prof. Ing. Kamil Vrba, CSc.
předseda oborové rady

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Jakub Pacher
Bytem: Renčova 1619/32, 62100, Brno - Řečkovice
Narozen/a (datum a místo): 28.3.1986, Malacky

(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 244/53, 60200 Brno 2
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Ing. Kamil Vrba, CSc.

(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - ☐ disertační práce
   - ☐ diplomová práce
   - ☑ bakalářská práce

   jiná práce, jejíž druh je specifikován jako .....

(dále jen VŠKP nebo dílo)

Název VŠKP: Protichybové systémy využívající blokové kódy
Vedoucí/školitel VŠKP: doc. Ing. Karel Němec, CSc.
Ústav: Ústav telekomunikací
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

- ☑ tištěné formě - počet exemplářů 1
- ☑ elektronické formě - počet exemplářů 1

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - ☑ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

ABSTRAKT
Abstrakt práce v češtině

Práce se zabývá výběrem a následným ověřením funkčnosti samoopravného kódu vhodného pro zabezpečení proti shlukujícím se chybám. Nejprve je podán rozbor protichybového kódového systému a požadavky kladené na něj. Dále je uveden přehled a vlastnosti často používaných kódů. Poté se zaměřuje na podrobný popis Fireova, Hammingova a dvou BCH kódů. Po zdůvodnění výběru právě Hammingova kódu je popisováno odzkoušení funkčnosti protichybového systému v prostředí MathCad.

KLÍČOVÁ SLOVA
Klíčová slova v češtině

samoopravný kód, zabezpečení dat, shluk chyb, protichybový kódový systém, Fireův kód, Hammingův kód, BCH kód

ABSTRACT
Abstract in English

This work deals with selection and proven suitable error-correcting code securing aggregating errors. First we give analysis anti-error coding system and requirements rapid/fire at them. The next text is focusing on survey and characteristics often used codes, mainly Fire, Hamming and two BCHs codes. After an explanation of Hamming code use, the text describes verification of proper function anti-error system in the MathCad enviroment.

KEYWORDS
Keywords in English

error-correcting code, data security, burst error, anti-error coding system, Fire code, Hamming code, BCH code

PACHER J. *Protichybové systémy využívající blokové kódy*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 45 s. Vedoucí bakalářské práce doc. Ing. Karel Němec, CSc.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Protichybové systémy využívající blokové kódy“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ……. ……..

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval svému vedoucímu bakalářské práce doc. Ing. Karlu Němcovi, CSc. za příkladnou metodickou, pedagogickou a odbornou pomoc a za další cenné rady při zpracování mé bakalářské práce.

V Brně dne ……. ……..

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

1 ÚVOD

Přenos informací na vzdálená místa se stal součástí našeho života. Hlavním cílem přenosu informace je její přenesení v co největším množství a pokud možno v co nejkratší době. Řešením se stává komprese informace, tedy zbavení redundance (nadbytečnosti) a irelevance (zbytečnosti).

Na druhé straně však nastává problém s bezchybným přenosem informace. Při přenosu, ať už pomocí kabeláže, radiovými vlnami nebo optickými vlnami, dochází k různým rušivým vlivům, které užitečný signál znehodnocují a zavádí chyby. Pro odstranění tohoto problému se využívají zabezpečovací kódy, které pro změnu k užitečné informaci přidávají jistou nadbytečnost na základě které se na přijímací straně detekují či korigují vzniklé chyby. Tento problém řeší teorie kódování, která vznikla již ve čtyřicátých letech pracemi Shannona [4], věnovanými teorii informace, a Hamminga s Golayem, kteří konstruovali první lineární kódy.

Jistou podskupinou zabezpečovacích kódů jsou kódy blokové a těmito se budeme v této práci dále zabývat.

1.1 Cíl práce

Ze zadání vyplývá, že máme navrhnout protichybový kodek, který bude opravovat shlukující se chyby. Máme přitom využít blokového kódu patřícího mezi samoopravné kódy. Skupina blokových kódů je rozsáhlá a proto v úvodu nejdříve rozebereme možné varianty a vhodnost jejich použití vzhledem k zadání. Při výběru a rozboru vhodného kódu je třeba brát v úvahu schopnosti kódů opravovat samostatné chyby či shluky chyb, ale také jejich náročnost realizace z hlediska kódovacího a dekódovacího procesu a zpoždění během těchto procesů.

Jelikož je třeba při návrhu respektovat, že protichybový kodek bude součástí protichybového kódového systému, rozebereme požadavky kladené právě na něj.

Pro samotnou realizaci a ověření funkčnosti se nabízí více variant. Rozebereme je a zvolíme tu, která z hlediska časové i finanční stránky bude nejméně náročná.

2 ZABEZPEČENÍ SAMOOPRAVNÝM KÓDEM

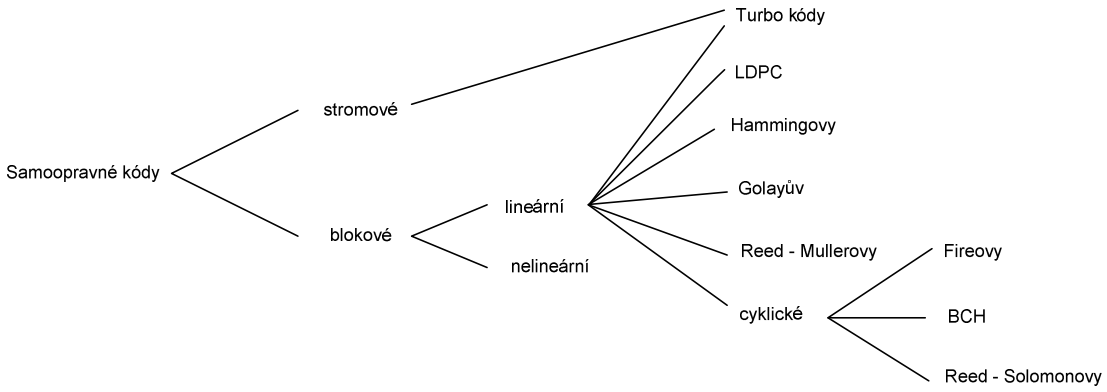
Díky rušivým vlivům při přenosu číslicového signálu vznikají chyby. Pro příjemce má však význam pouze bezchybně přijatá zpráva a tak je třeba přenos nějak zabezpečit. Využít při tom můžeme samoopravných kódů, které vlivem zavedené redundance dokáží na přijímací straně chyby odstranit.

Při opravě chyb (forward error correction, FEC) dochází v dekodéru k přiřazování přijaté posloupnosti symbolů nejpravděpodobnější posloupnosti, která byla vysílačem vyslána. Opravou chyb tedy dosáhneme menší bitové chybovosti (bit error rate, BER) oproti případu, kdy nepoužijeme žádného zabezpečení, či k dosažení stejné bitové chybovosti nám stačí menší odstup signál od šumu.

Pro opravu chyb existuje ještě jeden mechanizmus a to oprava chyb na základě automatické žádosti o opakování (automatic request for retransmission, ARQ). Využívá se při tom zpětného kanálu, kde jsou tyto žádosti v případě zjištění chyby vysílány od přijímače k vysílači. Ze zadání však vyplývá, že tento kanál nemáme k dispozici a tak se touto variantou dále nebudeme zabývat.

2.1 Přehled některých samoopravných kódů

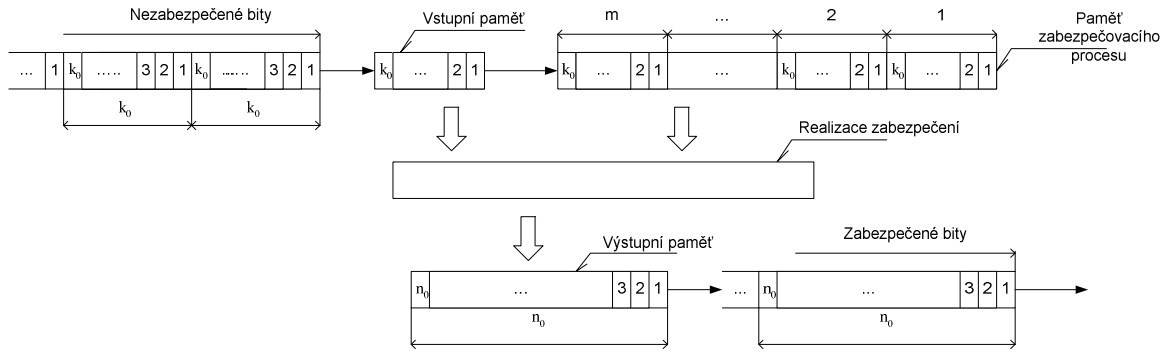
Samoopravné kódy můžeme rozdělit do dvou velkých skupin, a to blokové a konvoluční. Další dělení je naznačeno na obr. 2.1.



Obr. 2.1: Rozdělení některých samoopravných kódů

Pro přesnější orientaci v obr. 2.1 uvádíme stručnější popis těchto kódů.

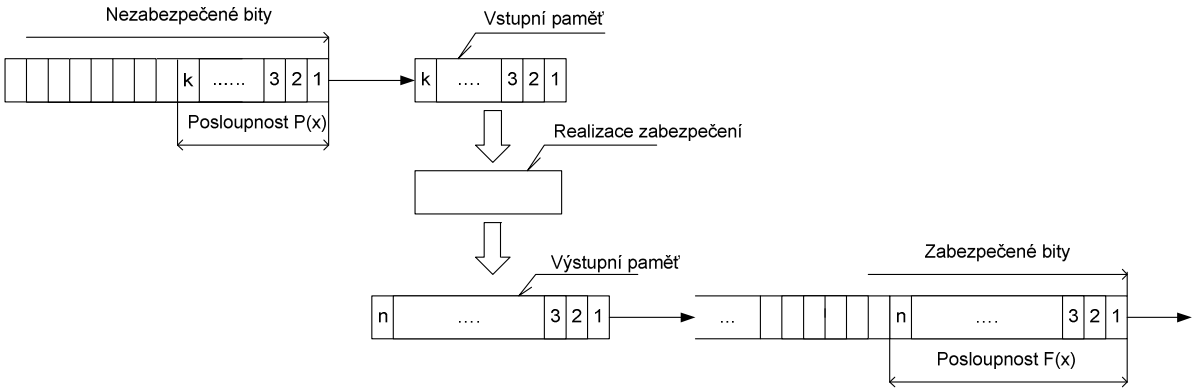
Stromové kódy - Název vychází ze způsobu vyjádření kódovacího procesu, kdy se nejčastěji používá graf typu strom. Hlavní rozdíl oproti blokovým kódům spočívá v realizátoru, který navíc obsahuje paměť zabezpečovacího procesu. Do ní se ukládají úseky nezabezpečené zprávy k_0 z m předcházejících časových okamžiků. Pro samotnou realizaci zabezpečení se poté nepoužívají pouze prvky ze současného k_0, ale právě i prvky, které jsou uloženy v paměti zabezpečovacího procesu. Na výstupu získáváme úseky zabezpečené zprávy n_0.



Obr. 2.2: Realizace zabezpečení stromovým kódem

Blokové kódy - Jak již sám název napovídá, pracují s tzv. bloky. Jsou to úseky vyňaté z celkové posloupnosti signálových prvků. Úseky nazýváme kódové kombinace.

Na vstup realizátoru kódu přichází mnohočlen nezabezpečené zprávy *P(x)* složen z *k* prvků nezabezpečeného toku dat. Ten je přeměněn na mnohočlen zabezpečené zprávy *F(x),* kdy se ke *k* prvkům pomocí jistého algoritmu přidá *r* prvků zabezpečovacích. Na přijímací straně poté do dekódovacího zařízení vstupuje mnohočlen přenesené zprávy *J(x).*



Obr. 2.3: Realizace zabezpečení blokovým kódem

Hammingovy kódy - Jedná se o blokové korekční kódy, dokáží opravit jednu chybu a jejich minimální Hammingova vzdálenost je $d_{min} = 3$. V případě Hammingových kódů pro smíšené zabezpečování, kdy se přidá celková kontrola parity (tj. přidá se jeden kontrolní symbol navíc), dochází ke korekci dvou chyb a $d_{min} = 4$. Jejich hlavní výhodou je, že dokáží opravit jednoduché chyby s co nejmenší myslitelnou redundancí. Takovéto kódy se nazývají kódy perfektní.

Golayův kód G_{23} – Kód binární lineární (23,12) s dvanácti informačními a jedenácti kontrolními znaky. Jedná se o kód perfektní pro trojnásobné chyby. Může být rozšířený na kód (24,12) přidáním celkové kontroly parity. Používá se v aplikacích, kde není zapotřebí velká průchodnost.

Fireovy kódy – Cyklický kód opravující shluky chyb určený vytvářecím mnohočlenem tvaru $G(x) = N(x) \cdot Q(x)$, kde $N(x)$ je nerozložitelný mnohočlen řádu m. Mnohočlen $N(x)$ je řádu m dělí-li mnohočlen $x^m + 1$ a žádný jiný mnohočlen nižšího stupně. Mnohočlen $Q(x)$ je tvaru ve $x^c + 1$.

Bose-Chaudhuriho-Hocquenghemovy kódy – Cyklické blokové binární kódy, které zvládají opravu vícenásobných chyb. Mezi jejich přednosti patří velká volitelnost parametrů, dobrý vztah mezi počtem informačních znaků a počtem opravovaných chyb a detailně vypracované dekódovací metody. Jejich název je zkracován na BCH kódy.

Reed-Solomonovy kódy – Vychází z BCH kódů a jejich předností je, že mají nejlepší opravné schopnosti pro kód dané délky. Ve srovnání s BCH kódy mají však složitější způsob dekódování.

LDPC kódy – Blokové kódy, které mají dlouhé kódové kombinace a jsou charakteristické řídkou kontrolní maticí. Objeveny byly již v roce 1961, ale pak byly na dlouho dobu zapomenuty díky složitému způsobu dekódování. V současné době dekódování probíhá např. pomocí iterativní metody. Představují alternativu k turbo kódům.

Turbo kódy – Tyto kódy vznikají paralelním zřetězením dvou nebo více kódů za použití prokládání. Mohou být při tom využiti jak blokové tak i konvoluční kodéry. Jejich objevení se datuje rokem 1993 a s jejich využitím se lze přiblížit Shannonově mezi.

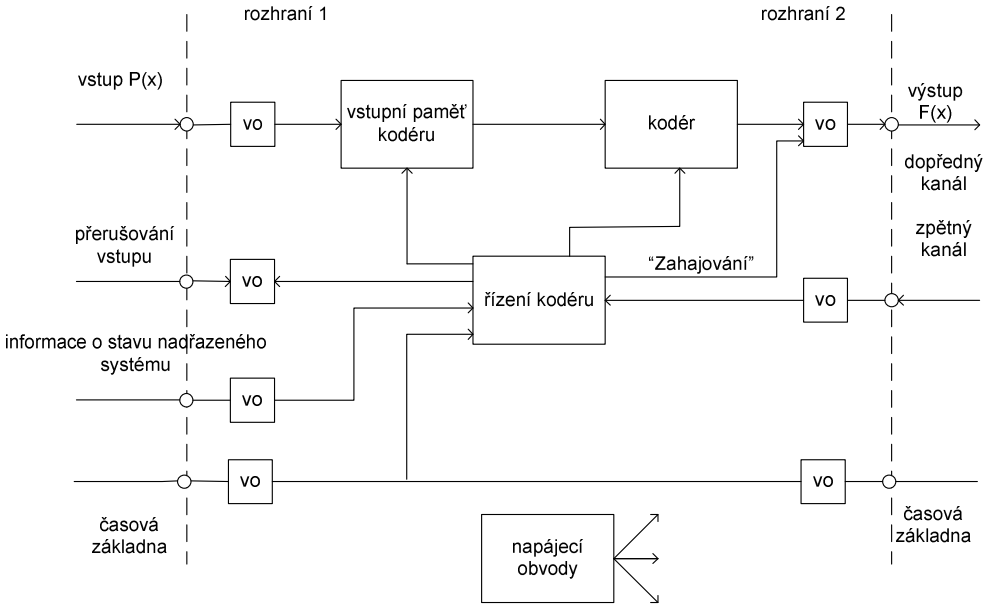
2.2 Diskuse zadání

Z předcházejícího přehledu popisujícího některé blokové kódy vyčleníme jen ty, které by byli vhodné pro naše zadání. Máme za úkol zabezpečit shluk chyb *b* maximální délky *22* bitů, který následuje vždy minimálně po $A = 1800$ bezchybně přenesených bitech. Pro řešení problému se nabízejí dvě varianty a to použití blokového kódu opravujícího přímo shluky chyb anebo kódu opravujícího jednu, dvě či tři chyby a kombinovat jej s tzv. prokládáním.

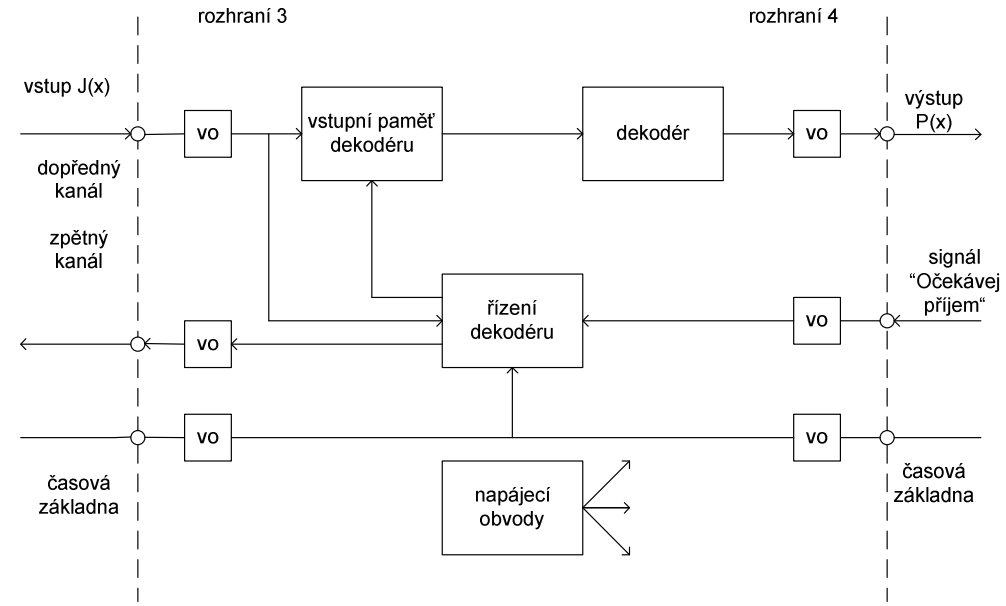
Pro první variantu se nabízí použití Reed-Solomonova kódu, díky jeho složitému způsobu dekódování jej však nebudeme uvažovat a tak se zaměříme pouze na kód Fireův. Pro druhou variantu můžeme pro opravu jednoduchých chyb použít perfektní kód Hammingův, pro opravu dvou či tří chyb lze vybrat Golayův kód, jelikož je však vhodný pouze pro aplikace s nízkou latencí, vybereme BCH kód.

3 PROTICHYBOVÝ KÓDOVÝ SYSTÉM

Při řešení našeho zadání musíme brát v úvahu také to, že optimalizace protichybového kódu probíhá na základě jeho použití v příslušném protichybovém kódovém systému PKS. Při výběru kódu se tedy berou v úvahu i požadavky kladené na PKS o kterých bude pojednáno v kapitole 3.1. Vycházíme přitom z popisu struktury PKS, a to jak ze struktury vysílače tak i přijímače PKS odvozených z lit [3].

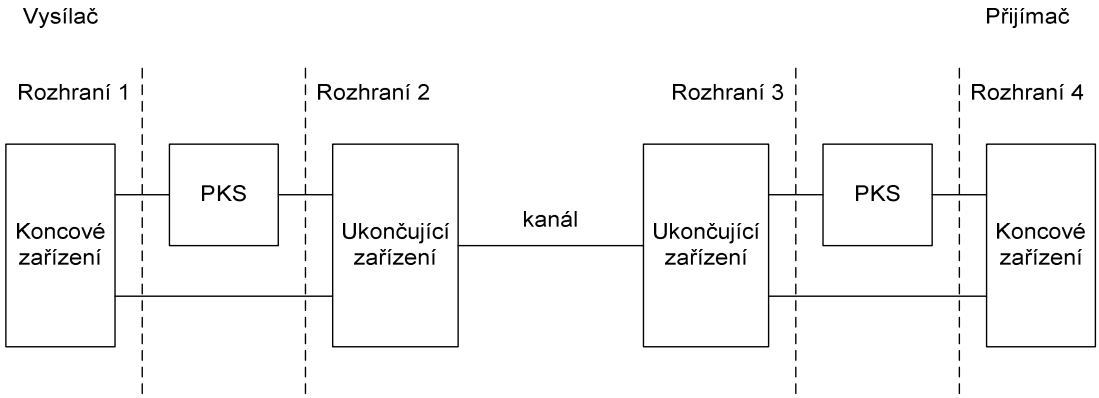


Obr. 3.1: Blokové schéma vysílače PKS



Obr. 3.2: Blokového schéma přijímače PKS

Protichybový kódový systém je zařazován do Systému přenosu dat SPD a stává se tak podřazeným systémem. Mimo něj se zde nachází koncová zařízení(zdroje/spotřebiče dat), ukončující datová zařízení(měniče datového signálu na signál vhodný pro přenos) a přenosový kanál. Hlavní věc, kterou je třeba zdůraznit je, že PKS je v SPD rozdělen na dvě samostatné části uvedených na obr. 3.1 a obr. 3.2, přičemž jedna je umístěna na vysílací a druhá na přijímací straně.



Obr. 3.3: Blokové schéma SPD se zařazeným PKS

3.1 Požadavky kladené na PKS

Podle [9] musíme při návrhu zabezpečovacího kódu brát ohled nejenom na kód samotný, ale i na celkový protichybový kódový systém. Jelikož tento je zařazován do systému přenosu dat, jsou na něj kladeny jisté požadavky.

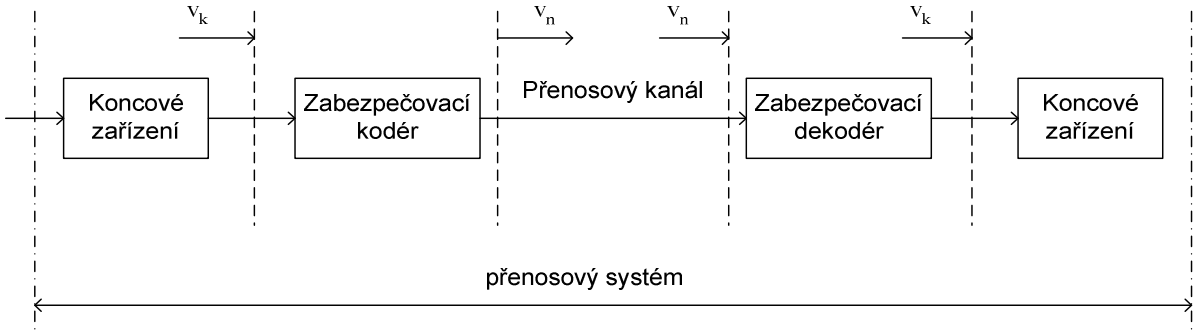
3.1.1 Požadavky plynoucí z různých délek zpracovávaných posloupností

Různé délky posloupností signálových prvků se objevují na vstupu kodéru, kde se nachází nezabezpečená data, a dále na výstupu z kodéru kde k nezabezpečeným datům jsou přidány zabezpečovací prvky.

Nejvíce se řeší problém, jestli můžeme nezabezpečenou zprávu délky K rozdělit na úseky k < K, a zda je kanál schopen přenést právě N signálových prvků.

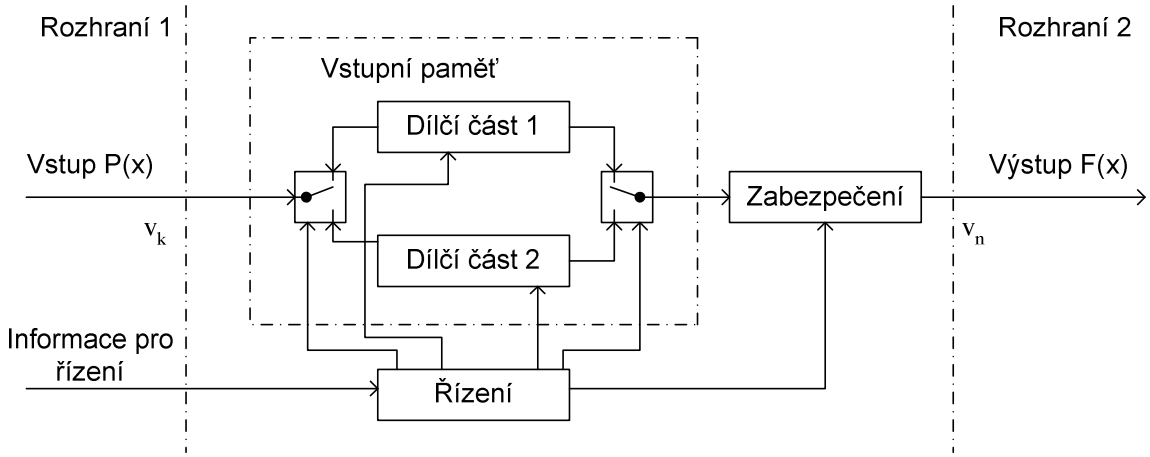
3.1.2 Požadavky plynoucí z užívání různých přenosových rychlostí

V kodéru můžeme předpokládat 2 i více různých přenosových rychlostí, které jsou na rozhraní 1 a 4, rychlost v_k, a na rozhraní 2 a 3, rychlost v_n . Obecně platí, že $v_n > v_k$. To, jestli máme v kodéru více než jednu přenosovou rychlost závisí na tom, zda-li chceme na vstupu PKS míti nepřerušovaný či přerušovaný bitový tok.



Obr. 3.4: Obrázek zachycující různé v_p na různých rozhraní

Pokud je na vstupu nepřerušovaný bitový tok, v systému se vyskytují alespoň 2 různé přenosové rychlosti. Tím pádem je zapotřebí do PKS začlenit nejméně dvě vstupní přepínatelné vyrovnávací paměti.



Obr. 3.5: PKS pracující s nepřerušovaným vstupním tokem

Pokud však budeme vstupní tok přerušovat můžeme lehce docílit rovnosti obou rychlostí $v_n = v_k$. Jediný problém nastává s řešením přerušování zmíněného bitového toku.

3.1.3 Požadavek na propustnost PKS

Propustnost zařízení je definována jako maximální možné přenesené množství informace za jednotku času. Přenášená informace je v daném zařízení zpožďována především procesy, kterými je podrobována. Tyto mohou být realizovány buď pevně zapojenými elektrickými obvody, či programově pomocí mikropočítače.

3.1.4 Požadavky na zahajování přenosu

Jak bylo již dříve zmíněno, systém přenosu dat je rozdělen na vysílací a přijímací část. Každá z těchto částí musí být před samotným přenosem užitečné informace jistým způsobem nastavena. To se děje pomocí tzv. *zahajovací posloupnosti ZP*.

Na vysílací straně je tedy PKS, pomocí zahajovací posloupnosti, vyrozuměn o tom, kdy má začít zabezpečovat. Dále mu tato posloupnost poskytuje informaci o zahájení zabezpečování, tzn. že se na jeho vstupu objevil první prvek, ze kterého budou odvozeny zabezpečovací prvky.

Na přijímací straně, před samotným přenosem, který přijímač již očekává, je vysílána právě zahajovací posloupnost. Ta slouží především k tomu, aby se přijímač spustil ve správný časový okamžik.

Z předešlého vyplývá, že bezchybný přenos zahajovací posloupnosti je velmi důležitý, a její případné znehodnocení má ohrožující vliv na celkový přenos dat. Zahajovací posloupnost je tedy chráněna proti chybám úplně jiným zabezpečovacím kódem, než poté užitečné informace.

3.1.5 Požadavek na spolehlivost PKS

Spolehlivost je schopnost zařízení plnit svou funkci po určitou dobu při zachování technických parametrů.

Ke snížení spolehlivosti PKS může dojít díky:

a) zhoršení provozních podmínek(kolísání teploty, vlhkosti, napájecího napětí…)
b) špatně navržené spolehlivosti PKS vzhledem k celkovému SPD
c) zhoršení chybovosti kanálu v SPD vlivem nedodržení vlastností rozhraní

3.1.6 Požadavky na synchronizaci

Systém je synchronizován právě tehdy, když všechny charakteristické úseky dvoustavového signálu, tedy bity, trvají stejný čas. To zajišťujeme shodou generovaných kmitočtů časových základen na vysílací a přijímací straně SPD.

Dále je třeba brát v úvahu tzv. Fázování, kdy v daném bloku dat musíme rozlišit více druhů prvků a znát jejich přesnou polohu. Hlavně zjišťujeme polohu informačních prvků (nesou informaci) a prvků služebních a zabezpečovacích (slouží ke správné činnosti systému a zabezpečení přenášeného bloku).

4 ROZBOR VHODNÉHO KÓDU

V kapitole 2.2 jsme vybrali 3 kódy, které by mohly být vhodné pro řešení našeho problému z hlediska vhodnosti pro opravu shlukujících se chyb. Nyní se zaměříme na jejich podrobnější popis a kritéria podle kterých vybereme optimální kód pro naše zadání. S výběrem kódu je však třeba brát v úvahu i požadavky kladené na protichybový kódový systém PKS jenž byly rozebrány výše.

Kritéria pro výběr vhodného kódu:

- informační poměr kódu R
- celková složitost kodeku S [PB]
- celkové zpoždění způsobené kodekem Z [krok]

Informační poměr kódu je dán vztahem (4.7). Složitostí kodeku S rozumíme počet paměťových buněk, ze kterých bude kodek realizován a nakonec zpoždění, které v kodéru vzniká vlivem odvození zabezpečovacích prvků, a v dekodéru vzniká vlivem zjišťování a následné opravy chyb.

4.1 Fireův kód

4.1.1 Základní popis

Fireův kód *(n,k)* je dle [1] blokový cyklický kód jehož vytvářecí mnohočlen je následujícího tvaru

$$G(x) = N(x) \,.\, (x^c + 1)\,, \qquad (4.1)$$

kde je *N(x)* nerozložitelný mnohočlen řádu *m*, náležející exponentu *e*, při čemž platí, že *c* není dělitelné *e*.

Exponent *e* se dá vypočítat dle následujícího vztahu

$$e = 2^m - 1. \qquad (4.2)$$

Parametry Fireova kódu jsou:

- délka kódové kombinace $n = e \,.\, c$ (4.3)
- počet zabezpečovacích prvků $r = c + m$ (4.4)
- počet zabezpečovaných prvků $k = n - r$ (4.5)

Fireův kód může pracovat ve 3 třech režimech – detekční mód, korekční mód a v tzv. smíšeném zabezpečení. Pro naše zadání použijeme korekčního módu pro který platí následující.

Řád mnohočlenu *N(x)* určuje délku shluku chyb *b* v kódové kombinaci chyb *n* a zároveň musí platit

$$c \geq 2b - 1 \;,\; m \geq b \,. \qquad (4.6)$$

4.1.2 Aplikace na zadání

Pro stanovení parametrů vycházíme z toho, že naším kódem potřebujeme opravit shluk chyb délky 22. Při zjišťování požadované délky kódové kombinace přitom použijeme vztahu (4.6), kdy za délku shluku *b* dosadíme právě hodnotu 22 získáváme $c \geq 43$ a $m \geq 22$.

Dále podle vztahu (4.2) odvodíme velikost exponentu $e = 2^m - 1 = 2^{22} - 1 = 4\,194\,303$.
Nyní máme k dispozici hodnoty pro výpočet délky kódové kombinace *n*, která se tedy rovná dle vztahu (4.3)

$n = e \,.\, c = 4\,194\,303 \,.\, 43 = 180\,355\,029$,

a následně ze vztahů (4.4) a (4.5) získáváme počet zabezpečovacích prvků $r = 65$ a počet zabezpečovaných prvků $k = 180\,354\,964$.

Nerozložitelný mnohočlen *N(x)* bude tedy řádu 22. K nalezení přesného tvaru mnohočlenu použijeme tabulky nerozložitelných mnohočlenů *N(x)* pro konstrukci Fireových kódů, které můžeme nalézt v [1]. Vybíráme polynom tvaru

$$N(x) = x^{22} + x^{21} + x^{20} + x^{17} + x^{16} + x^{10} + x^{9} + x^{6} + x^{5} + x^{2} + 1 ,$$

druhý polynom bude dle vztahu (4.1) tvaru $x^{43} + 1$.

Roznásobením těchto dvou polynomů získáváme vytvářecí polynom Fireova kódu (180 355 029 ; 180 354 964)

$$G(x) = x^{65} + x^{64} + x^{63} + x^{60} + x^{59} + x^{53} + x^{52} + x^{49} + x^{48} + x^{45} + x^{43} + x^{22} + x^{21} + x^{20} + x^{17} + x^{16} + x^{10} + x^{9} + x^{6} + x^{5} + x^{2} + 1.$$

Došli jsme k závěru, že nás kód bude potřebovat délku kódové kombinace n = 180 355 029, tato délka je však pro naše zadání příliš dlouhá. Náš přenosový systém pracuje s bloky délky n = 1822, což vznikne jako součet shluku chyb 22 bitů a 1800 bezchybných bitů. Pro naše potřeby je třeba tedy Fireův kód zkrátit. Použijeme přitom pravidel uvedených v lit. [1].

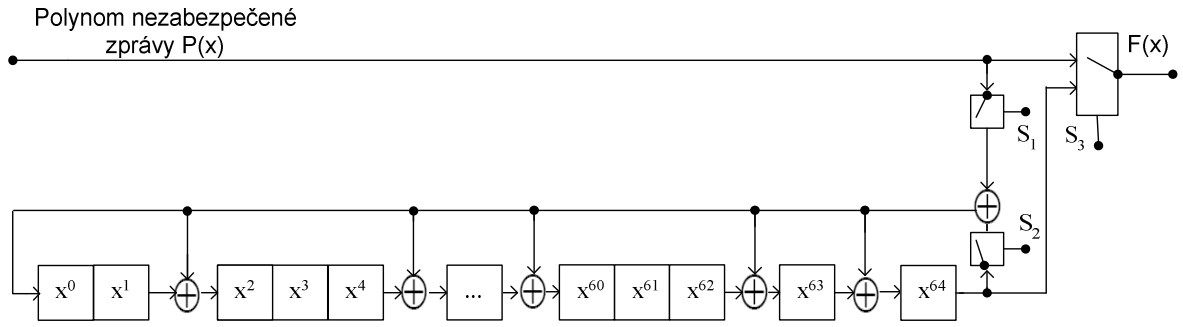
Jelikož potřebujeme Fireův kód, který bude zpracovávat bloky délky 1822, tak stupeň zkrácení i = 180 355 029 – 1822 = 180 353 207. Vzniká zkrácený Fireův kód (1822 ; 1757) s informační rychlostí 0,9643 definovanou dle následujícího vztahu

$$R = \frac{k}{n} . \qquad (4.7)$$

Princip zkracování délky kódové kombinace v podstatě spočívá předpokladu, že na prvních i místech nezkrácené kódové kombinace jsou nuly.

4.1.3 Kodér a dekodér zkráceného Fireova kódu (1822 ; 1757)

Kodér zkráceného Fireova kódu (1822 ; 1757) vychází z vytvářecího mnohočlenu Fireova kódu (180 355 029 ; 180 354 964) odvozeného v předchozí kapitole a bude tedy obsahovat, jako všechny kodéry cyklických kódů, děličku mod *G(x)* s přednásobením x^r, v našem případě x^{65}.



Obr. 4.1: Kodér zkráceného Fireova kódu (1822 ; 1757) zadaného vytvářecím mnohočlenem $G(x) = x^{65} + x^{64} + x^{63} + x^{60} + x^{59} + x^{53} + x^{52} + x^{49} + x^{48} + x^{45} + x^{43} + x^{22} + x^{21} + x^{20} + x^{17} + x^{16} + x^{10} + x^{9} + x^{6} + x^{5} + x^{2} + 1$

Pokud bychom uvažovali, že do kodéru vstupuje nepřerušovaný tok bitů, tak je třeba aby vstupní paměť kodéru uvedena na obr. 3.1 byla rozdělena na dvě části podle obr. 3.5. Bude obsahovat 2x1757 paměťových buněk a zpoždění, které tato vyrovnávací paměť bude vnášet do přenosu signálového prvku bude 1757 kroků.
Jelikož uvažujeme nepřerušovaný tok bitů na vstupu kodéru, tak výstupní rychlost bude větší než vstupní. Tím pádem kodér nezpůsobuje zpoždění průchodu signálových prvků přes PKS.

Pro dekódování se podle [1] používá nejčastěji tzv. Meggitův dekodér, který pracuje ve dvou krocích:

První krok : Kontrola správnosti přenosu. Po *n* krocích je v paměťových buňkách děličky *G(x)* syndrom *S(x)*. Pokud je nulový, přenos proběhl bezchybně, pokud je nenulový nastala chyba.

Druhý krok: Pokud první krok není beze zbytku, dochází k hledání *chybového mnohočlenu* a ke *korekci*. Toto opět trvá *n* kroků.

Při realizaci dekodéru musíme brát v úvahu, že na prvních *i* místech jsou nuly. Toto se řeší např. zařazením *i* paměťových buněk před vstup do děličky mod *G(x)*.
Při dekódování dále musíme přijatý mnohočlen *J(x)* vynásobit x^i, jelikož na vysílací straně jsme se vypuštěním prvních *i* prvků dopustili dělení původního mnohočlenu prvkem x^i. Na přijímací straně však používáme děličku mod *G(x)* s přednásobením x^r a tak přijatý mnohočlen *J(x)* je třeba vynásobit členem x^{i+r}.

Děličku mod *G(x)* s přednásobením x^r musíme doplnit o další vstupy do posuvného registru této děličky tak, aby vše odpovídalo struktuře mnohočlenu *f(x)*, jenž odpovídá zbytku po dělení x^{i+r} vytvářecím mnohočlenem *G(x)*.

Náš Fireův kód má zabezpečovacích prvků $r = 65$ a stupeň zkrácení jest $i = 180\,353\,207$. Mnohočlen *f(x)* tedy bude roven

$$f(x) = \frac{x^i}{G(x)} = \frac{x^{180353272}}{x^{65} + x^{64} + x^{63} + x^{60} + x^{59} + x^{53} + x^{52} + x^{49} + x^{48} + x^{45} + x^{43} + x^{22} + x^{21} + x^{20} + x^{17} + x^{16} + x^{10} + x^9 + x^6 + x^5 + x^2 + 1}$$

Zde však narážíme na omezení použitelnosti Fireova kódu, kdy pro opravu velkých shluků chyb narůstá délka kódové kombinace do extrému. Je tedy nutné ji zkrátit, čímž vzniká zkrácený Fireův kód. Při určování stupně zkrácení se však dostáváme do řádu stovek milionů a tím pádem se výpočet zbytku po dělení stává velmi výpočetně náročný. Pro výpočet jsme použili matematické systémy Maple, Matlab i MathCad, přičemž pro všechny byla tato operace natolik náročná, že se ji bohužel nepodařilo realizovat.

Shrnutí protichybového kódového systému

Variantu použití Fireova kódu pro opravu shluku chyb $b = 22$ nacházející se vždy po $A = 1800$ bezchybně přenesených bitech zamítáme.

4.2 Hammingův kód

4.2.1 Základní popis

Jedná se o blokový korekční kód, jehož minimální Hammingova vzdálenost je $d_{min} = 3$. V případě Hammingova kódů pro smíšené zabezpečování dochází ke korekci dvou chyb, $d_{min} = 4$. Jeho hlavní výhodou je, že pomocí něj dokážeme opravit jednoduché chyby s co nejmenší myslitelnou redundancí [5].

Parametry kódu:

- délka kódové kombinace v bitech $n = 2^m - 1$ (4.8)

- počet zabezpečovacích bitů $r = m$ (4.9)
- počet informačních bitů $k = 2^m - 1 - m$ (4.10)
- pro parametr m platí $m \geq 3$ (4.11)
- minimální vzdálenost $d_{min} = 3.$ (4.12)

4.2.2 Základní popis metody prokládání

Jelikož Hammingův kód slouží obecně pro opravu jednoduchých je třeba jej kombinovat s prokládáním. Princip prokládání je podle [1] následující. Tok nezabezpečených bitů vstupuje do kodéru, kde se k němu přidá (n - k) zabezpečovacích prvků pomocí nichž lze v dekodéru opravit až t chyb. Následuje ukládání kódových kombinací po řádcích do prokládací matice, která má rozměr n x j. Pro parametr j, tedy počet řádků matice, musí platit

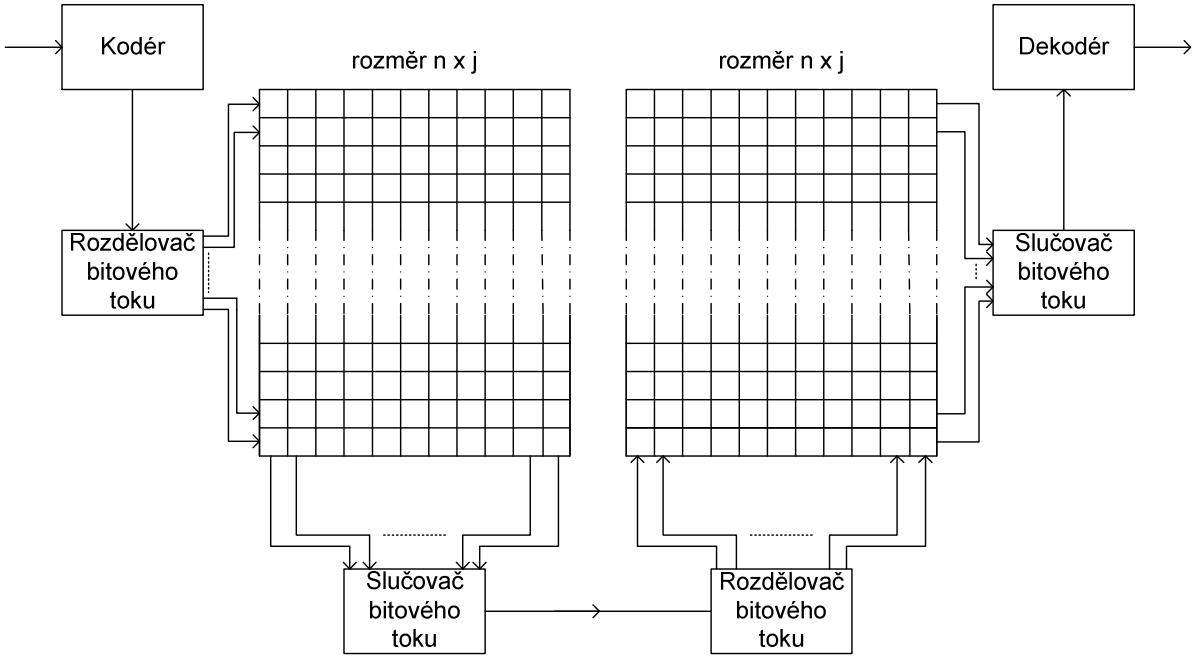
$$j \geq \frac{b}{t}, \tag{4.13}$$

kde b značí velikost shluku chyb, které chceme opravit a t je počet chyb, které je náš kód schopen opravit.
Pro parametr n, tedy počet sloupců matice musí platit

$$n \leq \frac{A+b}{j}, \tag{4.14}$$

kde A značí minimální délku bezchybného intervalu mezi shluky chyb.
Z matice jsou poté prvky čteny po sloupcích a přenášeny k přijímači. Zde se opět nachází prokládací matice, do které je zapisováno po sloupcích a čteno po řádcích. Vlivem prokladu je jakýkoliv shluk chyb délky b na přijímací straně rozložen na samostatné, či více násobné chyby, které už však dekodér dokáže opravit.
Z popisu vyplývá, že zpoždění vznikající prokladem bude rovno dvojnásobku rozměru prokládací matice, tedy

$$D_t = 2 \,.\, j \,.\, n \,. \tag{4.15}$$



Obr. 4.2: Schématické znázornění protichybového systému s prokladem

4.2.3 Aplikace na zadání

Pro naše zadání použijeme základního Hammingova kódu (7 ; 4), který opravuje jednu chybu a budeme jej kombinovat s prokládáním. Bylo by možné například použít Hammingův kód (63 ; 57), který má informační rychlost přesahující 0,9, ale jeho vytvářecí i kontrolní matice by obsahovala velký počet řádku i sloupců a tak i realizace kodéru a dekodéru by byla složitá. Vytvářecí matice Hammingova kódu (7 ; 4) $[G] = [M_k;C_{rk}]$ bude mít 7 sloupců a 4 řádky a podle [2] bude následujícího tvaru

$$[G_{(7;4)}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{matrix} \dots p_1 \\ \dots p_2 \\ \dots p_3 \\ \dots p_4. \end{matrix}$$

Jeho kontrolní matice [H] má podle [2] tvar

$$[H_{(7;4)}] = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} \dots 3 \\ \dots 2 \\ \dots 1. \end{matrix}$$

Ze struktury kontrolní matice [H] vychází zapojení kodéru. Tedy princip odvození zabezpečovacích prvků. Jelikož je tento Hammingův kód kódem systematickým, tak zabezpečovací prvky budou přímo za užitečnými prvky zprávy a budou odvozeny dle pravidla, které říká $[H] \times [F]^T = 0$, kde [F] je kódový vektor zprávy.

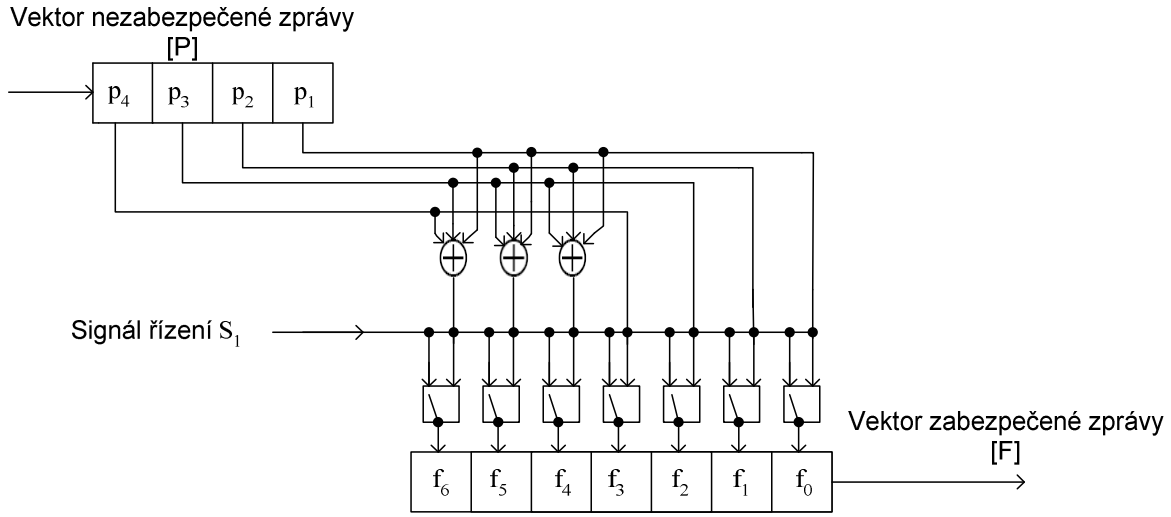
Po rozepsání jednotlivých skalárních součinů dostáváme rovnice:

$$f_0 \oplus f_1 \oplus f_2 \oplus f_4 = 0$$
$$f_0 \oplus f_1 \oplus f_3 \oplus f_5 = 0$$
$$f_0 \oplus f_2 \oplus f_3 \oplus f_6 = 0.$$

Podle předchozích rovnic lze získat předpis pro výpočet kontrolních bitů:

$$f_4 = f_0 \oplus f_1 \oplus f_2$$
$$f_5 = f_0 \oplus f_1 \oplus f_3$$
$$f_6 = f_0 \oplus f_2 \oplus f_3.$$

Poté zapojení kodéru



Obr 4.3: Blokové schéma kodéru Hammingova kódu (7 ; 4)

Zpoždění, které vzniká při odvození zabezpečovacích prvků je nulové. Je třeba však počítat se zpožděním, které je dáno zápisem 4 bitů do vstupní paměti, což se uskuteční během čtyř kroků. Pro další zápis do vstupní paměti však systém musí čekat dalších 7 kroků, než dojde k vyprázdnění kódové kombinace z výstupní paměti. Po jedenácti krocích tedy může dojít k dalšímu odvození zabezpečené zprávy.

Nyní je třeba navrhnout postačující rozměr prokládací matice. Víme, že máme k dispozici kód opravující $t = 1$ chybu. Počet řádku matice tedy musí splňovat podmínku dle vzorce (4.13) $j \geq 22$. Počet sloupců matice musí splňovat podmínku danou vztahem (4.14) $n \leq 82$.
Náš kód má délku kódové kombinace $n = 7$ a tedy podmínku danou vztahem (4.14) splňuje. Za j volíme hodnotu nejnižší možnou, a tedy $j = 22$. Rozměr matice bude 22 x 7, což odpovídá 154 paměťovým buňkám.
Zpoždění vlivem celého procesu prokládání bude rovno podle (4.15) dvojnásobku rozměru matice, čili 308 kroků vlivem prokladu a 308 kroků vlivem zpětného prokladu.

Dekodér Hammingova kódu (7 ; 4) vychází ze struktury kontrolní matice [H], která má dle [2] následující tvar

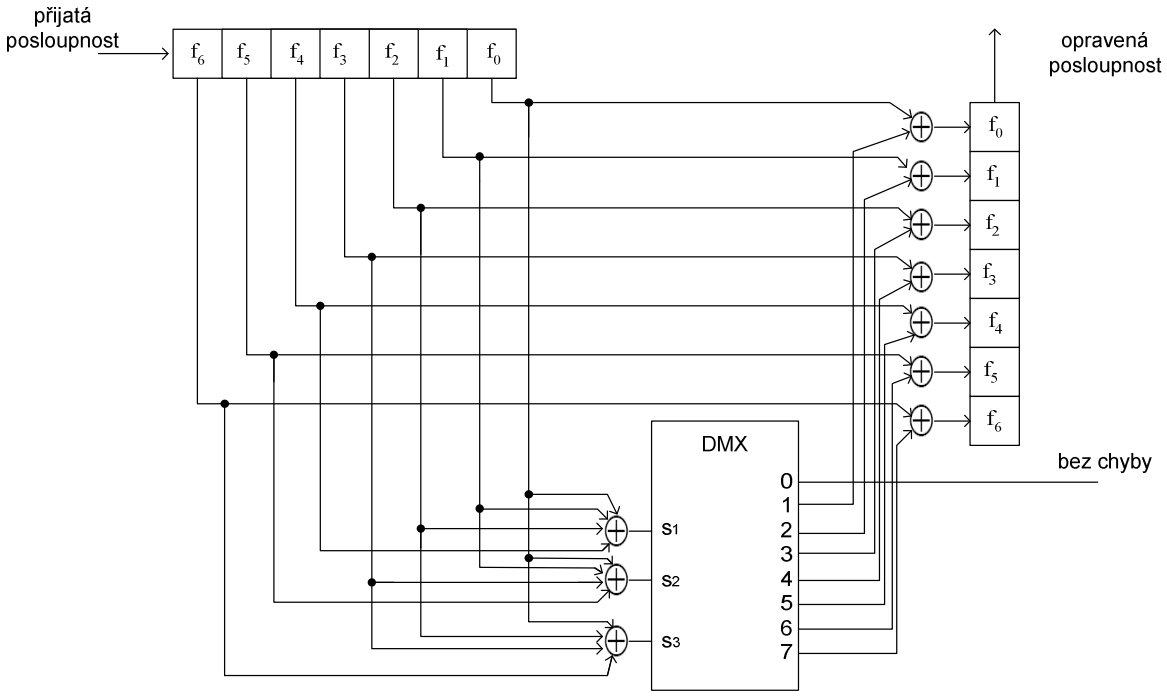
$$[H_{(7;4)}] = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} \rightarrow \; s_1 = f_0 \oplus f_1 \oplus f_2 \oplus f_4 \\ \rightarrow \; s_2 = f_0 \oplus f_1 \oplus f_3 \oplus f_5 \\ \rightarrow \; s_3 = f_0 \oplus f_2 \oplus f_3 \oplus f_6. \end{matrix}$$

Uvedené tři rovnice určují zapojení obvodu pro výpočet syndromu, podle kterých se dle [3] odvodí chybové vektory [E]. Součtem mod 2 chybového vektoru a přijatého vektoru získáváme opravenou zprávu.

Dekodér bude mít dle [2] strukturu uvedenou na obr. 4.4. Zpoždění při výpočtu syndromu, převodu syndromu na chybový mnohočlen a korekci je opět nulové, systém však musí čekat 7 kroku na zaplnění vstupní paměti a dalších 7 kroků na vyprázdnění výstupní paměti s opravenou posloupností.

Shrnutí protichybového kódového systému

- informační poměr kódu R = 0,57
- celková složitost kodeku S = 326 paměťových buněk + 2 x (slučovač + rozdělovač) + demultiplexor
- celkové zpoždění způsobené kodekem Z = 11 kroků (kodér) + 616 kroků (proklad) + 14 kroků (dekodér) = 641 kroků



Obr. 4.4: Blokové schéma dekodéru Hammingova kódu (7 ; 4)

4.3 Bose – Chaudhuriho - Hocquenghemovy kódy

4.3.1 Základní popis

BCH kódy dle [6] patří do skupiny cyklických kódů, jenž pracují s daty v binární podobě. Binární BCH kód (*n ; k*) umožňující opravu *t* chyb existuje pro každé číslo $m \geq 3$ a má následující parametry:

- $n = 2^m - 1$ maximální délka kódového slova (4.15)
- $k \geq n - mt$ počet informačních bitů v kódovém slově (4.16)
- $d_{\min} \geq 2t + 1$ minimální Hammingova vzdálenost (4.17)

Při sestavování generujícího polynomu se vychází z Bosého a Chaudriho teorému (BC teorém) – viz [7]:

Jestliže se mezi kořeny vytvářecího mnohočlenu cyklického (n ; k) kódu nachází určitý počet prvků Galoisova tělesa jdoucích velikostí svých exponentů po sobě, což zapíšeme takto: a^m , a^{m+1} , ... , a^{m+d-2} , pak minimální Hammingova vzdálenost d_{min} tohoto kódu není menší než d .

Pro vytvoření generujícího mnohočlenu BCH kódu délky $n = 2^m - 1$ schopného opravit t chyb se nejdříve vybere *primitivní mnohočlen* řádu m. Primitivní mnohočlen je nerozložitelný mnohočlen stupně m s koeficienty z GF(2), který dělí mnohočlen $x^n - 1$ beze zbytku. Pomocí primitivního mnohočlenu se zkonstruuje těleso GF(2^m). Následně je potřeba nalézt nerozložitelné mnohočleny $g_i(x)$ jejichž kořeny tvoří po sobě jdoucích $2t$ po sobě jdoucích mocnin primitivního prvku α tělesa GF(2^m). Generující mnohočlen $g(x)$ je pak tvořen součinem mnohočlenů $g_i(x)$. Podrobnější popis je uveden v [6].

Pro dekódování je možné použít:

- Výpočetní dekódování. Zde se použije např. Petersonův či Sugyiamův algoritmus. Pro svou složitost je použitelné v podobě programu. Více nalezneme [7].
- Majoritní dekódování. Je snadno realizovatelné pevně zapojenými logickými obvody, ale je použitelné pouze pro několik BCH kódů.

4.3.2 Aplikace na zadání

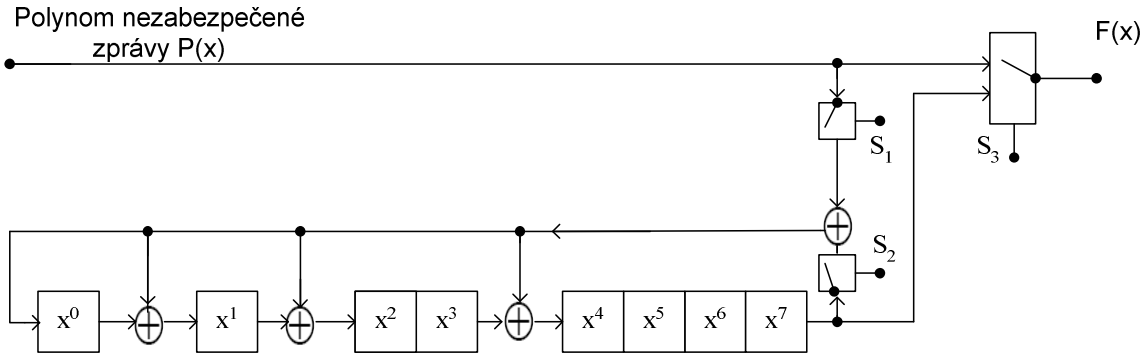
V úvodu jsme zmínili, že pro naše zadání je možné použít dva kódy BCH a to kód opravující $t = 2$ či $t = 3$ chyby. Pro prvně zmíněnou variantu se nabízí BCH kód (*15 ; 7*), který díky svým parametrů právě dvě chyby dokáže opravit. Kód BCH (*15 ; 5*) se poté nabízí pro opravu tří chyb.

4.3.2.1 BCH kód (15 ; 7)

BCH kód, jako každý cyklický kód, je zadán vytvářecím mnohočlen $G(x)$. Za použití Bose Chaudhuriho teorému a po prostudování [6] byl odvozen následující vytvářecí mnohočlen.

$$G(x) = x^8 + x^4 + x^2 + x + 1$$

Z něj je možné podle [3] odvodit schéma kodéru, tedy děličky mod $G(x)$ s přednásobením x^8, které bude následující.



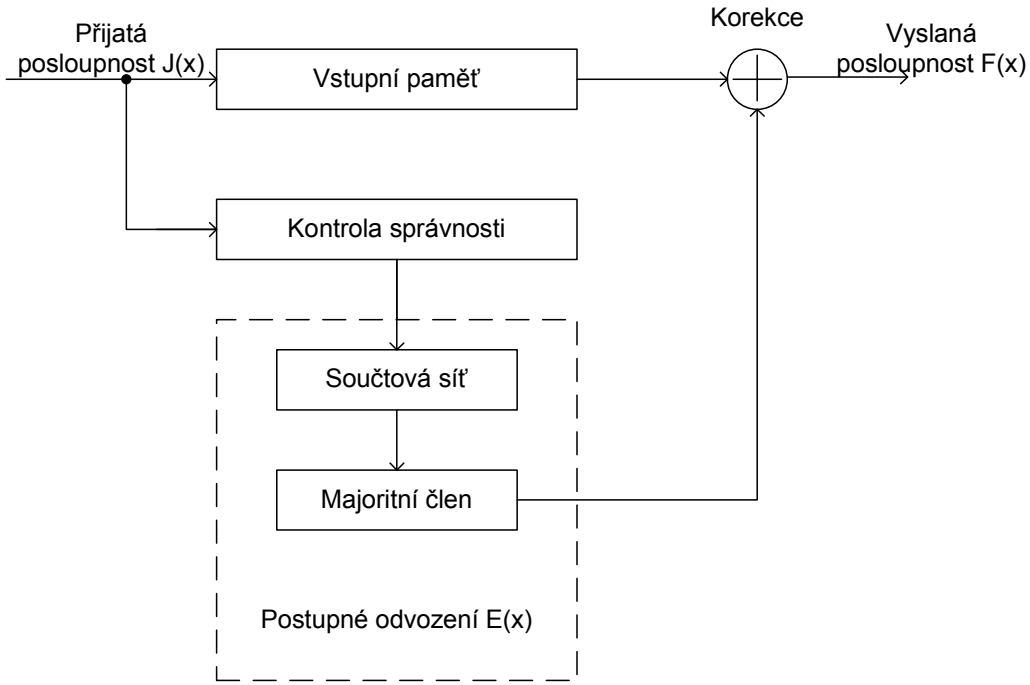
Obr. 4.5: Kodér BCH kódu (15 ; 7)

Zpoždění způsobené odvozením zabezpečovací posloupnosti bude 15 kroků, kdy v prvních $k = 7$ krocích dochází k plnění děličky a poté v $(n - k) = 8$ krocích dochází k jejímu vyprazdňování. Počet paměťových buněk pro realizaci jest 8. Dále je třeba do PKS zahrnout vyrovnávací paměť, která bude obsahovat 7 paměťových buněk pro uložení vstupní nezabezpečené posloupnosti.

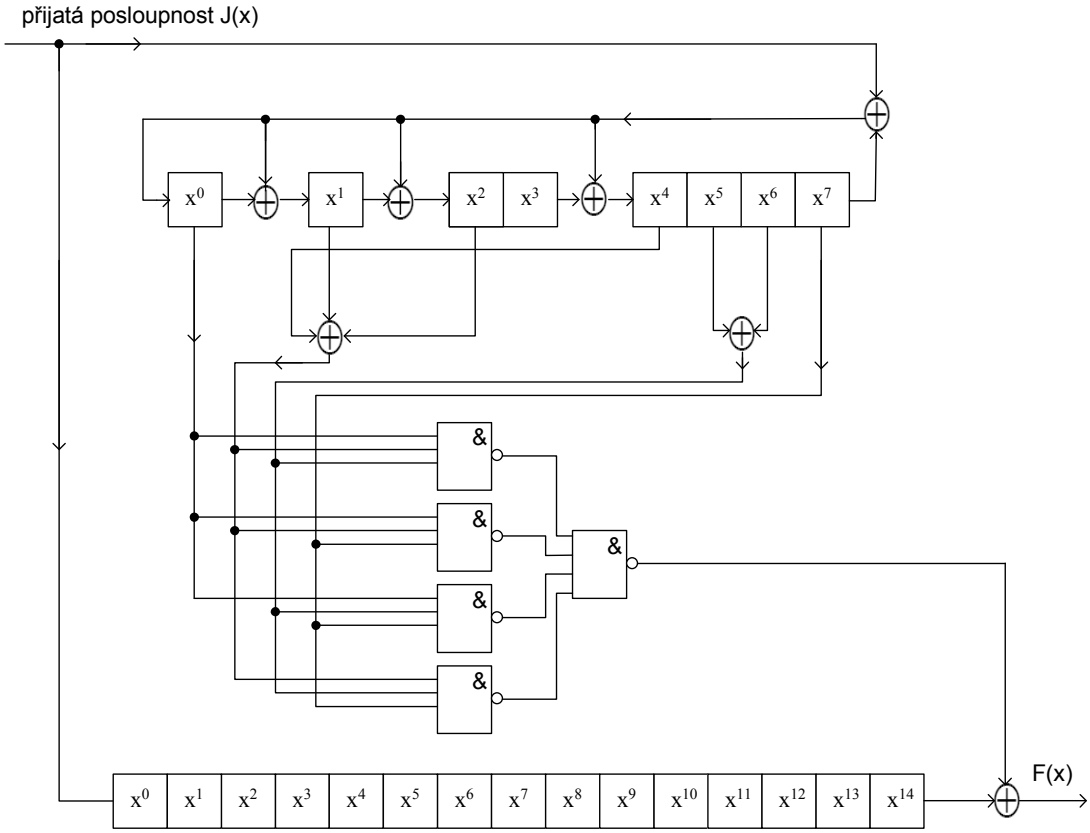
Jelikož tento kód použijeme s prokladem je třeba odvodit podle (4.13) a (4.14) rozměry prokládací. Pro počet řádků tedy platí $j \geq 11$ a pro počet sloupců musí platit $n \leq 152$. Kód BCH (15 ; 7) těmto podmínkám vyhovuje a tak rozměr matice bude 11 x 15, tj. 165 paměťových buněk.
Celkové zpoždění prokladem bude rovno opět dvojnásobku tedy 330 x 2 kroků.

Pro dekódování využijeme majoritního způsobu. Podle [6] je základní schéma majoritního dekodéru uvedeno na obr. 4.6.

V bloku kontrola správnosti bude obsažena dělička mod G(x) s přednásobením x^8, která po $n = 15$ krocích má v paměťových buňkách mnohočlen syndromu S(x). V kroku $n + 1$ dochází k přesunu obsahu paměťových buněk a dochází tak k dělení S(x) „na sebe“, čímž dochází k jeho změně. Součtová siť bude mít strukturu podle [6] přičemž se vychází z kontrolní matice [H] ve které nalezneme $d^* = d_{min} - 1 = 4$ kontrolní součty, které musí být ortogonální vůči jednomu z nich. Podle [8] je majoritní člen tvořen systémem logických obvodů, které mají tolik vstupů kolik je kontrolních součtů a jeden výstup. Na něm se objeví taková hodnota bitu, jaká je většina bitových hodnot na vstupech. Dochází zde tedy k převodu bitů syndromu S(x) na bity chybového mnohočlenu E(x), který se vytváří postupně.



Obr. 4.6: Blokové schéma obvodu pro majoritní dekódování



Obr. 4.7: Majoritní dekodér BCH kódu (15 ; 7)

Majoritní dekodér BCH kódu (15 ; 7) je uveden na obr. 4.7. Zpoždění vznikající při detekci a korekci chyb je dáno odvozením syndromu S(x) v děličce mod G(x) s přednásobením x^8, které spolu s plněním paměti trvá 15 kroků. Po 15 krocích máme v děličce mnohočlen syndromu S(x). V kroku (n + 1) vystoupí z paměti, v níž je uložena přijatá zpráva J(x), první bit a je sečten mod 2 s bitem, který vznikl z kontrolních součtů a průchodem skrz majoritní člen. Jedná o první bit chybového mnohočlenu. Takto po dalších 15 krocích získáváme na výstupu vektor F(x) vyslané zprávy. Celkové zpoždění při odvození vektoru F(x) činí 30 kroků.

Shrnutí protichybového kódového systému

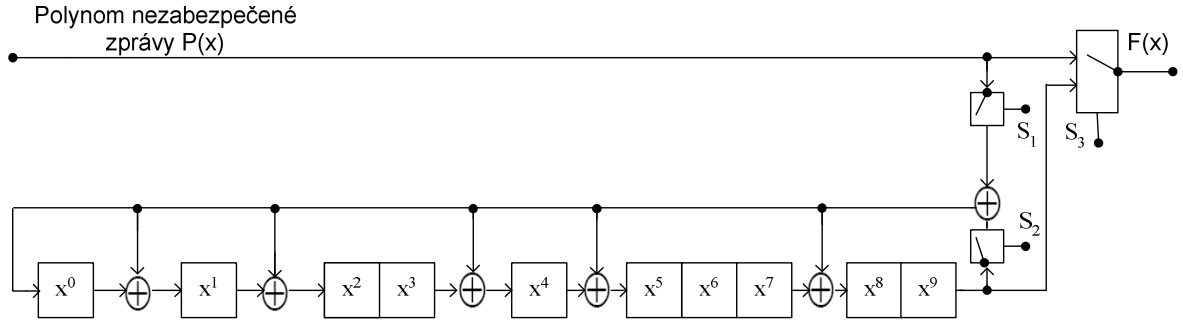
- informační poměr kódu R = 0,4667
- celková složitost kodeku S = 7 (vyrovnávací paměť) + 8 (kodér) + 330 (proklad) + 23 (dekodér) [PB] + majoritní člen
- celkové zpoždění způsobené kodekem Z = 7(vyrovnávací paměť) + 15 (kodér) + 330x2 (proklad) + 30 (dekodér) = 742 kroků

4.3.2.2 BCH kód (15 ; 5)

Kód sloužící pro opravu t = 3 chyb, tedy s minimální Hammingovou vzdáleností d_{min} = 7. Vytvářecí mnohočlen G(x) bude dle BC teorému sestrojen ze součinu těch nerozložitelných mnohočlenů $g_i(x)$, jejichž 2t = 6 kořenů Galoisova tělesa jde podle exponentů po sobě. Pro GF(2^4) lze podle [6] nalézt

$$G(x) = (x^4 + x^2 + 1)\,(x^4 + x^3 + x^2 + x + 1)\,(x^2 + x + 1) = x^{10} + x^8 + x^5 + x^4 + x^3 + x^2 + 1$$

Kodér bude opět tvořen děličkou mod G(x) s přednásobením x^{10}.



Obr. 4.8: Kodér BCH kódu (15 ; 5)

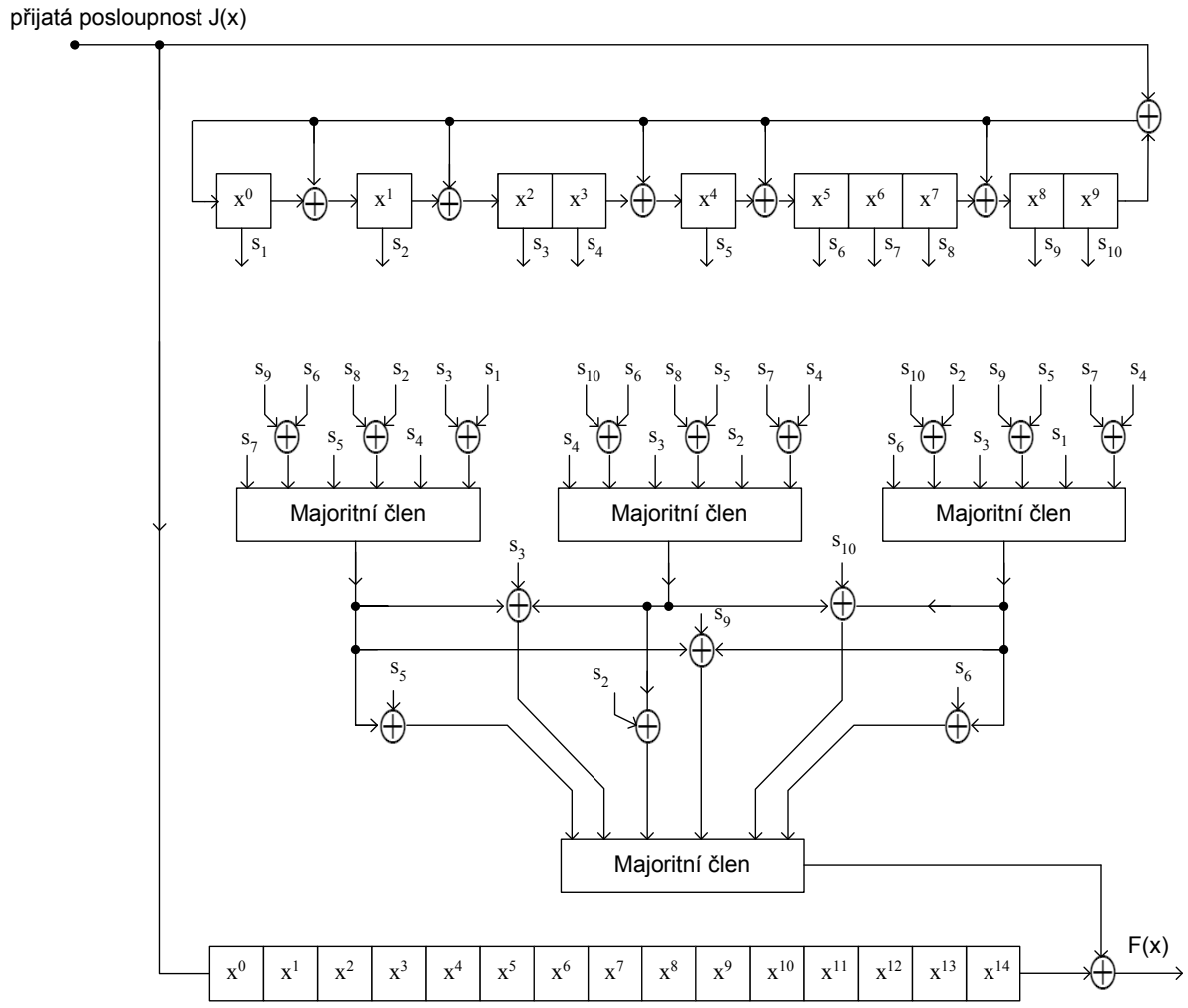
Zpoždění způsobené odvozením zabezpečovací posloupnosti bude 15 kroků. Dále je třeba na vstup umístit vyrovnávací paměť, která bude obsahovat 5 paměťových buněk. Celkový počet paměťových buněk pro realizaci jest 15.

Jelikož tento kód použijeme opět s prokladem je třeba odvodit podle (4.13), (4.14) rozměry prokládací matice. Pro počet řádků tedy platí $j \geq 8$ a pro počet sloupců musí platit $n \leq 228$. Kód BCH (15 ; 5) těmto podmínkám vyhovuje a tak rozměr matice bude 8 x 15, tj. 120 paměťových buněk.
Celkové zpoždění prokladem bude rovno dvojnásobku tedy 240 x 2 kroků.

Struktura dekodéru vychází z poznatků získaných v lit. [6]. Jelikož kód BCH (15 ; 5) má minimální Hammingovu vzdálenost $d_{min} = 7$, je třeba v kontrolní matici [H] tohoto kódu nalézt $d^* = d_{min} - 1 = 6$ kontrolních součtů. Podle [6] však tyto součty nelze nalézt v jednom kroku a tak je třeba použít majoritní dekodér se dvěma kroky majorizace.
Schéma je odvozeno z [6]. Zpoždění je stejné jako u předchozího dekodéru, tedy 30 kroků.

Shrnutí protichybového kódového systému

- informační poměr kódu R = 0,333
- celková složitost kodeku S = 5 (vyrovnávací paměť) + 10 (kodér) + 240 (proklad) + 25 (dekodér) [PB] + majoritní člen
- celkové zpoždění způsobené kodekem Z = 7 (vyrovnávací paměť) + 15 (kodér) + 240x2 (proklad) + 30 (dekodér) = 532 kroků



Obr. 4.9: Majoritní dekodér BCH kódu (15 ; 5)

4.4 Výběr optimálního kódu

V předchozích kapitolách jsme podrobně rozebrali čtyři možné varianty řešení našeho zadání. Prvně byl rozebrán Fireův kód, při jehož návrhu nám vyšla informační rychlost přesahující hodnotu 0,9 a zároveň předpoklad, že není třeba použít prokladu, jelikož kód opravuje přímo shluky chyb, poukazovali zpočátku na vhodnost použití pro naše zadání. S postupujícím návrhem jsme se však setkali s problémem, který obecně Fireův kód pro použití opravy dlouhých shluků chyb omezuje, a to výpočet mnohočlenu *f(x)*. Tento byl natolik výpočetně náročný, že jsme od použití Fireova kódu odstoupili.

Další tři varianty už jsou kombinovány s prokladem. Pro přehlednost jsou v tab. I uvedeny jednotlivé PKS s kódy a jejich výběrová kritéria.

Tab. I: Protichybové kódové systémy a jejich výběrová kritéria

kód	informační poměr R	celková složitost S	zpoždění Z		
			[kroky]		
-	-	[PB]	zpoždění kódu	zpoždění prokladem	celkem
Hammingův (7;4)	0,570	326	25	616	641
BCH (15;7)	0,466	398	52	720	772
BCH (15;5)	0,333	280	52	480	532

Kritérium celková složitost nebudeme dále pro výběr kódu uvažovat, jelikož je úzce spjato s kritériem celkové zpoždění. Co se týče zpoždění vlivem prokladu, tak nejmenšího zpoždění a tedy i nejmenšího rozměru prokládací matice lze dosáhnout použitím kódu BCH (15 ; 5). Díky tomu, že kód dokáže opravit tři chyby, je rozměr matice 8 x 15 paměťových buněk. Co se týče složitosti kodéru a dekodéru, tak nejmenší zpoždění při odvození zabezpečovacích prvků je u Hammingova kodéru. Zde v podstatě v jednom kroku získáváme celou kódovou kombinaci. Je třeba ale vždy počkat 4 kroky než dojde k zaplnění vstupní paměti pro nezabezpečenou posloupnost a dalších 7 kroků než dojde k vyprázdnění paměti se zabezpečenou posloupností. Dekodér, který vnáší nejmenší zpoždění, je opět Hammingův. Systém s BCH kódem (15 ; 7) má ze třech možných systému největší zpoždění a i díky informační rychlosti pod 0,5 jej dále nebudeme brát v úvahu.

Ze dvou zbývající možností má systém s BCH kódem (15 ; 5) sice menší celkové zpoždění, přibližně o 100 kroků, ale téměř poloviční informační rychlost oproti systému s Hammingovým kódem (7 ; 4).

Proto jako nejvhodnější systém volíme systém s Hammingovým kódem (7 ; 4).

5 OVĚŘENÍ FUNKČNOSTI KODEKU

Pro odzkoušení správné funkce námi navrženého kodeku se nabízí více variant. První možností se stává použití technologie TTL. Hlavní nevýhoda této varianty je bezesporu cena a spolu se značnou komplikovaností desky plošných spojů bylo od této varianty opuštěno.
Další možností jest použití programovatelného logického pole. Uvádíme zde přehled hlavních typů programovatelných číslicových obvodů z hlediska velikosti a vlastností připravených funkčních bloků.

- **PLD** - jednoduché obvody s programovatelným polem AND/OR. Umožňují realizovat jednoduché logické funkce

- **CPLD** - složitost je mezi obvody PLD a FPGA. Ve struktuře lze najít prvky z obvodů PLD i obvodů FPGA. Rekonfigurovatelné buňky se mohou propojit pomocí jednoduché propojovací struktury. Obvykle obsahují permanentní paměť. Jsou svým rozsahem integrace vyhovující pro méně náročné návrhy.

- **FPGA** - mají nejobecnější strukturu. Obvody FPGA jsou tvořeny pravidelně uspořádanými logickými bloky a propojovacím systémem. Zpravidla obsahují speciální bloky s pamětí RAM, vestavěné bloky se speciálními funkcemi. Hustota integrace je vyšší než u obvodů CPLD. Jsou vhodné pro náročné návrhy.

Tato varianta je však opět příliš komplikovaná a tak pro prosté odzkoušení funkčnosti našeho navrženého systému jsme se rozhodli využít softwarového řešení s využitím osobního počítače. Jako simulační program použijeme program MathCad 2001 Professional.

5.1 Odzkoušení funkčnosti v programu MathCad 2001 Professional

V matematickém programu MathCad jsme vytvořili námi navržený systém. Při tvorbě systému jsme vycházeli ze schématu uvedeného na obr. 5.1.

Popis zdrojového kódu:

V programu se jako vstupní proměnná používá pocet dat, kterou volíme počet vstupních dat. Je třeba vždy zadat číslo větší nebo rovno a zároveň dělitelné 88, aby došlo k zaplnění celé prokládací matice.

Pro generování náhodné posloupnosti s rovnoměrným rozložením pravděpodobnosti výskytu 1 i 0 se použila funkce runif. Tato se ukládá do proměnné data.

pocet dat $\mathbf{N} := 88$

$\mathbf{data} := \mathbf{floor}(\mathbf{runif}(\mathbf{N}, 0, 2))$

$\mathbf{data}^{\mathbf{T}} =$

	0	1	2	3	4	5	6	7	8	9
0										

vygenerovana posloupnos

$\mathbf{length}(\mathbf{data}) = 88 \quad \mathbf{data}_0 = 0$

Do funkce reprezentující Hammingův kodér (7 ; 4) HAMME vstupuje proměnná data. Tato je rozdělena na čtveřice, z nichž se vždy odvodí tři zabezpečující bity pomocí pravidel uvedených v kap. 4.2.3. Tento cyklus se opakuje nejméně 22krát. Na výstupu získáváme výstupní vektor označený vyst_vec.

$\mathbf{HAMME}(\mathbf{in}) :=$
- $\mathbf{k} \leftarrow \frac{\mathbf{length}(\mathbf{in})}{4}$
- **for** $\mathbf{i} \in 0..\mathbf{k} - 1$
  - $\mathbf{vec} \leftarrow \mathbf{submatrix}(\mathbf{in}, \mathbf{i}\cdot 4, \mathbf{i}\cdot 4 + 3, 0, 0)$
  - $\mathbf{f} \leftarrow \mathbf{vec}$
  - $\mathbf{f}_4 \leftarrow \mathbf{vec}_0 \oplus \mathbf{vec}_1 \oplus \mathbf{vec}_2$
  - $\mathbf{f}_5 \leftarrow \mathbf{vec}_0 \oplus \mathbf{vec}_1 \oplus \mathbf{vec}_3$
  - $\mathbf{f}_6 \leftarrow \mathbf{vec}_0 \oplus \mathbf{vec}_2 \oplus \mathbf{vec}_3$
  - **for** $\mathbf{j} \in 0..6$
    - $\mathbf{v}_{\mathbf{i}\cdot 7+\mathbf{j}} \leftarrow \mathbf{f}_{\mathbf{j}}$
  - $\mathbf{v}$

$\mathbf{vystup\_vec} := \mathbf{HAMME}(\mathbf{data}) \qquad \mathbf{length}(\mathbf{vystup\_vec}) =$ ∎

$\mathbf{vystup\_vec}^{\mathbf{T}} =$ ∎

Dále je třeba realizovat prokládání, toto realizujeme funkcí interleaving. Proměnná vystup_vec je členěna na bloky 154 bitů a pomocí funkce mod(j.7,153) dochází k přeměně pořadí vstupní posloupnosti. Například pro hodnotu *j = 2* získáváme 14. Tedy druhý bit výstupní posloupnosti má být roven čtrnáctému bitu vstupní posloupnosti. Toto odpovídá zápisu do matice 7x22 po sloupcích a čtení po řádcích.

$$\textbf{interleaving}(\textbf{in}) := \left| \begin{array}{l} \textbf{k} \leftarrow \dfrac{\textbf{length}(\textbf{in})}{154} \\ \textbf{for}\ \ \textbf{i} \in 0..\textbf{k}-1 \\ \quad \left| \begin{array}{l} \textbf{for}\ \ \textbf{j} \in 0..152 \\ \quad \textbf{v}_{\textbf{i}\cdot 154+\textbf{j}} \leftarrow \textbf{in}_{\textbf{i}\cdot 154+\textbf{mod}(\textbf{j}\cdot 7,153)} \\ \textbf{v}_{\textbf{i}\cdot 154+153} \leftarrow \textbf{in}_{\textbf{i}\cdot 154+153} \\ \textbf{v} \end{array} \right. \end{array} \right.$$

Pro simulaci shluku chyb jsme vytvořili vektor e délky 154 bitů, jenž obsahuje 22 chybových jedniček. Tento přičítáme mod2 k posloupnosti vycházející z prokládací matice vystup_inter, získáváme tak proměnnou prenos, která obsahuje i chybná data.

$$\textbf{prenos} := \overrightarrow{(\textbf{vystup\_inter} \oplus \textbf{e})}$$

Zpětný proklad je realizován obdobně jako dopředný proklad, s rozdílem že používáme funkci mod(j.22,153).

$$\textbf{deinterleaving}(\textbf{in}) := \left| \begin{array}{l} \textbf{k} \leftarrow \dfrac{\textbf{length}(\textbf{in})}{154} \\ \textbf{for}\ \ \textbf{i} \in 0..\textbf{k}-1 \\ \quad \left| \begin{array}{l} \textbf{for}\ \ \textbf{j} \in 0..152 \\ \quad \textbf{v}_{\textbf{i}\cdot 154+\textbf{j}} \leftarrow \textbf{in}_{\textbf{i}\cdot 154+\textbf{mod}(\textbf{j}\cdot 22,153)} \\ \textbf{v}_{\textbf{i}\cdot 154+153} \leftarrow \textbf{in}_{\textbf{i}\cdot 154+153} \\ \textbf{v} \end{array} \right. \end{array} \right.$$

Po zpětném prokladu získáváme proměnnou vystup_deinter, která následně vstupuje do dekodéru.

$$\textbf{vystup\_deinter} := \textbf{deinterleaving}(\textbf{prenos})$$

Dekodér Hammingova kódu (7 ; 4) je opět realizován z poznatků uvedených v kap. 4.2.3.

```
HAMMD(in) := | k ← length(in) / 7
             | for i ∈ 0..k − 1                                          vybirani sedmic z cele posloupnosti
             |   vec ← submatrix(in, i·7, i·7 + 6, 0, 0)
             |   for j ∈ 0..7
             |     s0 ← vec0 ⊕ vec1 ⊕ vec2 ⊕ vec4
             |     s1 ← vec0 ⊕ vec1 ⊕ vec3 ⊕ vec5                        vypocet syndromu
             |     s2 ← vec0 ⊕ vec2 ⊕ vec3 ⊕ vec6
             |   e ← (1 0 0 0 0 0 0)^T  if (s = (1 1 1)^T)
             |   e ← (0 1 0 0 0 0 0)^T  if (s = (1 1 0)^T)
             |   e ← (0 0 1 0 0 0 0)^T  if (s = (1 0 1)^T)               prirazeni syndromu chybovym
             |   e ← (0 0 0 1 0 0 0)^T  if (s = (0 1 1)^T)               mnohoclenum
             |   e ← (0 0 0 0 1 0 0)^T  if (s = (1 0 0)^T)
             |   e ← (0 0 0 0 0 1 0)^T  if (s = (0 1 0)^T)
             |   e ← (0 0 0 0 0 0 1)^T  if (s = (0 0 1)^T)
             |   e ← (0 0 0 0 0 0 0)^T  if (s = (0 0 0)^T)
             |   for n ∈ 0..3                                            oprava chyby, pricteni mod2
             |     v_{i·4+n} ← e_n ⊕ in_{i·7+n}                          chyboveho vektoru
             |   v
```

Výstupní posloupnost vystup poté srovnáváme s proměnnou data, která obsahuje užitečná data vstupující do kodéru. Jsou-li tyto posloupnosti shodné, proměnná kontrola nabývá hodnoty „OK“, tedy posloupnost napadená shlukem chyb je opravena.

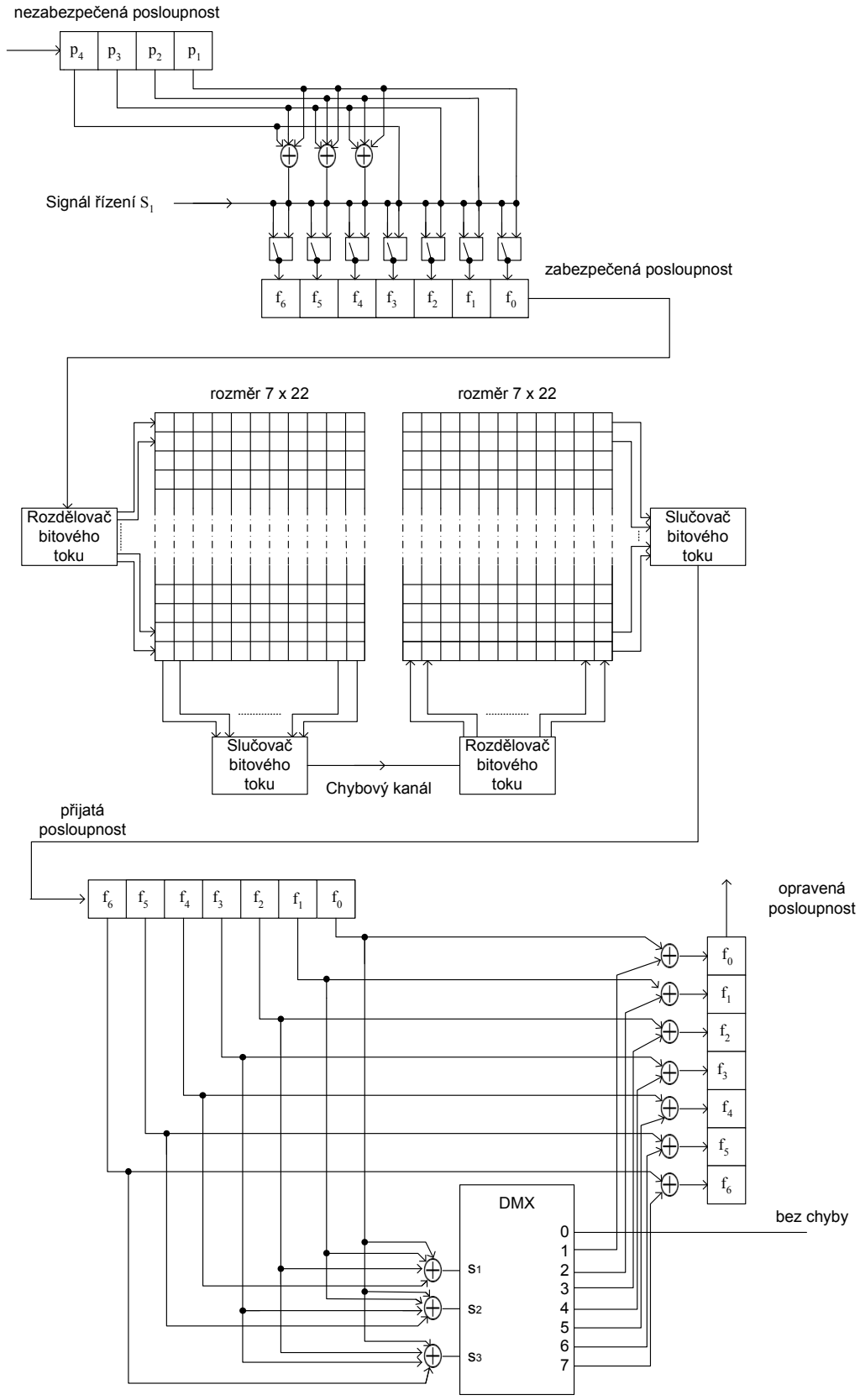
```
vystup := HAMMD(vystup_deinter)

vystup^T = ■                                   vystup z dekoderu, jen uzitecne bity

                                               overeni zda opravdu doslo k oprave

data^T = ■                                     vstup do koderu

kontrola := if( Σ (vystup ⊕ data)→ = 0, "OK", "chyba" )      kontrola = ■      length(vystup) = ■
```



Obr. 5.1: Protichybový systém pro opravu shlukujících se chyb s využitím Hammingova kódu (7 ; 4)

6 ZÁVĚR

Naším úkolem bylo vypracovat návrh protichybového kódového kodeku, který zabezpečí digitální přenos dat proti shlukujícím se chybám. Měli jsme přitom využít blokového korekčního kódu. Jelikož je náš protichybový kodek součástí protichybového kódového systému tak jsme nejdříve rozebrali v kap. 3 požadavky, které jsou kladeny na PKS. Obvyklá struktura PKS, z které se při optimalizaci kódu vycházelo je znázorněna na obr. 3.1 a obr. 3.2.

Dále jsme v kap. 4 přistoupili k rozboru jednotlivých kódů. Fireův kód rozebíraný v kap. 4.1 se zpočátku jevil jako vhodný pro naše zadání. Hlubším rozborem a aplikací na zadání jsme však od jeho použití odstoupili z důvodů uvedených v kap. 4.1.3.

Zbylé tři možné protichybové kodeky již k zabezpečení používají prokládací matice, které jsou vhodné pro rozložení shlukujících se chyb objevujících se během přenosu. Podrobně rozebrány byly všechny tyto tři varianty v kap. 4.2 a 4.3. Jako optimální vyšla v kap. 4.4 možnost s použitím Hammingova kódu (7 ; 4).

Pro odzkoušení funkčnosti navrženého kodeku se nabízelo více variant, jenž jsou uvedeny v kap. 5. Samotné odzkoušení funkčnosti je uvedeno v následující kapitole.
Námi navržený protichybový systém splňuje zadání.

SEZNAM LITERATURY

[1] NĚMEC, K. *Protichybové kódové systémy zabezpečující proti dlouhým shlukům chyb.* [online] 2005. poslední revize Dostupné z: <http://www.elektrorevue.cz/clanky/06034/>

[2] HLAVIČKA, J.; RACEK, S. *Číslicové systémy odolné proti poruchám*. Praha: Vydavatelství ČVUT, 1992. ISBN 80-01-00852-5

[3] NĚMEC, K. *Datová komunikace*. Skriptum. Ústav telekomunikací VUT v Brně, 1998, ISBN 80-7204-088-X.

[4] SHANNON, C.E. *A Mathematical Theory of Communication* .Bell Syst. tech. J., Vol. 27, No. 3, July 1984 , pp. 379-423, and Vol. 27, No. 4, October 1948, pp. 623-656.

[5] ADÁMEK, J. *Kódování*. Nakladatelství technické literatury, Praha 1989.

[6] NĚMEC, K. *Protichybové kódové zabezpečení s BCH kódy.* [online] 2005. poslední revize Dostupné z: <http://www.elektrorevue.cz/clanky/05015/>

[7] PETERSON,W.W. *Error-Correcting Codes*. The MIT Press, Cambridge, 1963.

[8] NĚMEC, K. *Majoritní dekódování blokových kódů*. . [online] 2005. poslední revize. Dostupné z: <www.elektrorevue.cz/clanky/05023/>

[9] NĚMEC,K. *Řešení požadavků kladených na protichybový kódový systém.* [online] 2005. poslední revize. Dostupné z: <www.elektrorevue.cz/clanky/06017/>

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

A	úsek bezchybně přenesených bitů
b	délka shluku chyb
D_t	zpoždění vznikající prokladem
d_{min}	minimální Hammingova vzdálenost
F(x)	mnohočlen zabezpečené zprávy
G	vytvářecí matice
G(x)	vytvářecí mnohočlen
H	kontrolní matice
j	počet řádků prokládací matice
J(x)	mnohočlen přijaté zprávy
k	délka nezabezpečeného bloku
n	délka zabezpečeného bloku
P(x)	mnohočlen zabezpečené zprávy
R	informační rychlost kódu
S	celková složitost kódu
Z	celkové zpoždění způsobené kódem

ARQ	-	Automatic Request for Retransmission
BCH	-	Bose-Chaudhuri-Hocquenghem
BER	-	Bit Error Rate
CPLD	-	Complex Programmable Logic Device
FEC	-	Forward Error Correction
FPGA	-	Field Programmable Gate Array
GF	-	Galois Field
LDPC	-	Low Density Parity Check
PKS	-	Protichybový kódový systém
PLD	-	Programmable Logic Device
SPD	-	Systém přenosu dat
ZP	-	Zahajovací posloupnost