Primjena LDPC kodova na poboljšanje performansi SLC fleš memorija visokog kapaciteta

Velibor V. Vitomir
Telekomunikacije RS a.d. Banja Luka
Istočno Sarajevo, BiH
velibor.vitomir@mtel.ba

Predrag N. Ivaniš
Elektrotehnički fakultet
Beograd, Srbija
predrag.ivanis@etf.rs

Sažetak—Stalno proširenje kapaciteta fleš memorije u praksi dovodi do gušćeg pakovanja ćelija, što za posljedicu ima povećanje greške očitavanja informacija iz ćelija. Zbog toga postojeće tehnike za obezbeđivanje pouzdanosti memorija kod kojih svaka ćelija ima dva stabilna stanja (*Single Level Cell, SLC)* ne daje više dobre rezultate. U ovom radu je predloženo jedno od riješenja kako dizajnirati memorijsku arhitekturu visoke pouzdanosti uz malu kompleksnost implementacije. Ključnu ulogu u predloženom riješenju ima dizajn kontrolera u kome su primjenjeni kodovi sa proverama parnosti male gustine (*Low-Density Parity-Check*, LDPC). Razmotrena je primjena dvije klase kodova i dvije vrste algoritama za njihovo dekodovanje, sa tvrdim i mekim odlučivanjem.

Ključne riječi - LDPC kodovi, gradijentni bit-fliping algoritam, BP algoritam, SLC fleš memorije.

I. UVOD

Fleš memorije omogućavaju skladištenje velike količine podataka na malom prostoru, uz veoma mali utrošak energije na uređaju bez pokretnih dijelova koji bi bili podložni mehaničkim oštećenjima. Ova tehnologija postaje sve popularnija i komercijalno su dostupni uređaji sa sve većim stepenom integracije, a smanjenje veličine memorijskih ćelija i naboja ćelije direktno dovodi do njihove povećane osetljivosti na šum [1].

Sve ovo dovodi do potrebe za primjenom sve snažnijih zaštitnih kodova, kako bi se i za veliki stepen integracije očuvala visoka pouzdanost zapisa, tako da korisniku i u otežanim uslovima bude dostavljena upisana informacija. Standardno riješenje za primenu u fleš memorijama predstavljaju linearni blok kodovi [2]. Najpoznatiji od njih su BCH (*Bose-Chaudhuri-Hocquenghem*) kodovi, koji su pronađeni nezavisno od strane Hokenhema [3] s jedne, i Bouza i Čodhurija [4] s druge strane. Ovi kodovi se projektuju tako da se njihovim dekodovanjem može garantovano ispraviti određeni broj grešaka, bez obzira na koje pozicije unutar kodne reči su unjeti.

U novije vrijeme, pouzdanost sistema za zapis podataka postiže se primjenom kodova sa proverama parnosti male gustine (*Low Density Parity Check, LDPC*). Iako je težina grešaka koje LDPC kodovi mogu garantovano ispraviti manja u odnosu na BCH kodove, zahvaljujući iterativnom dekodovanju oni mogu ispraviti veći broj sekvenci grešaka velike težine (za istu dužinu kodne reči i isti kodni količnik) [5]. Iz ovog razloga, sistemi za zapis kod kojih se pouzdanost obezbjeđuje pomoću LDPC sa iterativnim dekodovanjem i tvrdim odlukama imaju bolje performanse a srednja verovatnoća da kodna riječ ne bude dekodovana (*Frame Error Rate, FER*) je manja za isto stanje u kanalu [6].

Ako se BCH ili LDPC kodovi dekoduju pomoću iterativnog algoritma sa mekim odlukama, performanse mogu biti još znatno bolje [7], [8]. Ovde se podrazumijeva da su iz dekodera dostupni meki izlazi, što je slučaj kad se dekoder vezuje na izlaz telekomunikacionog sistema u kome djeluje šum. Ipak, originalne NAND flash memorije imaju samo dva nivoa naboja memorijske ćelije, pa se na ulazu dekodera koji je povezan na njihov izlaz pojavljuje po pravilu binarna sekvenca. Ovo u velikoj mjeri otežava primjenu algoritama sa mekim dekodovanjem, iako se ovo djelimično može prevazići primjenom raznih varijanti *belief propagation* (BP) algoritma ili višestrukim očitavanjem podataka [9].

Sa druge strane, od početka istraživanja vezanih za LDPC kodove poznat je algoritam tvrdog dekodovanja male kompleksnosti, poznat kao bit-fliping (*bit flipping*, BF) [10]. Pre nekoliko godina objavljene su modifikacije ovog algoritma zasnovane na probabilističkom pristupu [11], maksimizaciji gradijentne kriterijumske funkcije [12], dok su ova dva metoda kombinovana u cilju razvoja algoritma pogodnog za primenu u binarnom kanalu [13].

U ovom radu biće prikazano da se pomoću ovih algoritama dekodovanja može dizajnirati memorijska arhitektura visoke pouzdanosti i male kompleksnosti, sa boljim performansama od riješenja koja se koriste u postojećim riješenjima. Razmotrene su dvije klase kodova koji su dobijeni determinističkim postupkom, pri čemu jedna obezbjeđuje garantovano ispravljanje velikog broja grešaka i malo kašnjenje pri dekodovanju a druga malu kompleksnost dekodera i efikasnu implementaciju dekodera (uz mali utrošak energije). Performanse obe klase kodova date su i za slučaj tvrdog i za slučaj mekog dekodovanja.

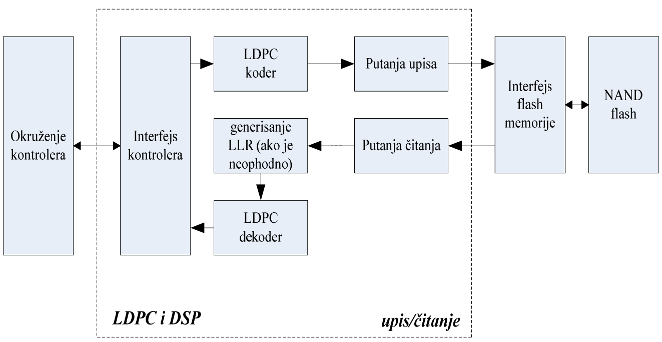
U II odjeljku objašnjeno je kako se LDPC kodovi mogu iskoristiti za povećanje pouzdanosti zapisa u NAND fleš memorijama. Notacija za opis LDPC kodova data je u III odjeljku, a zatim je ukratko opisan način konstrukcije kodova baziranih na konačnim geometrijama. U IV odjeljku opisano je nekoliko algoritama za iterativno dekodovanje LDPC kodova. U V odjeljku dati su numerički rezultati dobijeni Monte Karlo simulacijama a u poslednjem dijelu rada data su neka zaključna razmatranja.

II. PRIMJENA LDPC KODOVA U MEMORIJAMA

Informacija koja je trenutno uskladištena u fleš memoriji može biti pogrešna, a ima više faktora koji utiču na njenu ispravnost. Povećanje kapaciteta memorije dovodi do smanjenja veličine memorijske ćelije, što dovodi do povećanja varijacija odnosa signal-šum od ćelije do ćelije, efekat “istrošenosti” (*wear-out*) i međusobna ćelijska kapacitivnost su izraženiji. Sve ovo dovodi do grešaka nastalih u samoj memoriji (iščezavanja informacije). Pored toga, mogu se pojaviti i greške prilikom upisa u memoriju, greške prilikom isčitavanja kao i greške u samom dekoderu.

Da bi se povećala pouzdanost skladištenja informacija, prilikom čitanja iz memorije informacija se propušta kroz dekoder. Po izvršenom dekodovanju dobijena informacija prosleđuje se na magistralu, kao što je prikazano na slici 1. Konvencionalni SSD kontroleri koriste Hemingove ili BCH kodove i algoritme dekodovanja zasnovane na tvrdom odlučivanju. Originalne NAND fleš memorije su zasnovane na SLC (*Single Level Cell*) tehnologiji, gdje je moguće izvršiti upisivanje jednog bita po ćeliji i gdje je očitavanje bazirano na tvrdom odlučivanju. Stalni zahtjev za proširenjem kapaciteta SSD memorija uskovljen je gušćim pakovanjem ćelija, što se djelimično može ublažiti korišćenjem ćelija sa više nivoa - MLC (*Multi Level Cell*). S obzirom na činjenicu da su performanse LDPC kodova uz primjenu algoritama mekog odlučivanja, veoma bliske Šenonovoj granici za procese koji se modeluju ABGŠ, LDPC kodovi su već našli primjenu kod MLC tehnologije prilikom višestrukog očitavanja koje omogućuje povećanje preciznosti, što je preduslov mekom odlučivanju [9]. Pokazuje se da se primenom LDPC kodova daleko nadmašuju performanse dobijene korišćenjem BCH kodova [6].

Sa druge strane, u slučaju memorija zasnovanih na SLC tehnologiji dobijanje meke informacije iz kanala je znatno otežano. Zato je posebno izazovno dizajnirati kontroler koji bi obezbijedio pouzdan rad NAND fleš memorije SLC tipa. Kako povećani kapacitet memorije dovodi do veće nepouzdanosti medijuma za zapis, zahtjeva se primjena kodova sa superiornijim performansama od postojećih. Zato bi bilo interesantno provjeriti da li je moguće primjeniti LDPC kodove u odsustvu meke informacije iz kanala, pri čemu je poželjno da kompleksnost realizacije ne bude previše velika.



Sl. 1. Blok dijagram LDPC kontrolera

U ovom radu biće pretpostavljeno da se greške u medijumu mogu modelirati binarnim simetričnim kanalom (*Binary Symmetric Channel, BSC*), pri čemu je verovatnoća greške u kanalu ravna α. Ovo je uobičajena pretpostavka u slučaju kada se smatra da je glavni uzrok iščeznuća informacije u medijumu šum a da se greške pri upisu i čitanju mogu zanemariti (ili se pojavljuju nasumično).

III. KONSTRUKCIJA LDPC KODOVA

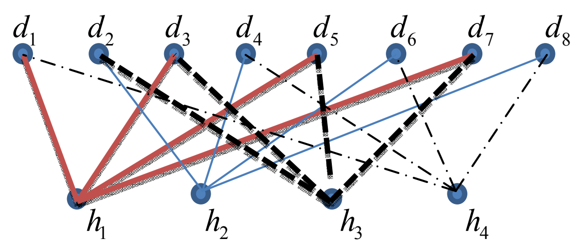
LDPC kodovi pripadaju klasi linearnih blok kodova. Karakteriše ih relativno mali broj jedinica u odnosu na dimenzije H matrice. Pritom je sa γ ($\gamma \geq 3$) označen broj jedinica po koloni, a ρ je broj jedinica po vrsti kontrolne matrice. Regularni LDPC kodovi imaju osobinu da im je isti broj jedinica u svakoj koloni i vrsti γ i ρ respektivno, dok kod iregularnih LDPC kodova to nije slučaj.

Ove kodove je predložio Robert Gallager u svojoj doktorskoj disertaciji iz 1960 godine [10]. Ovi kodovi dugi niz godina nisu bili uopšte razmatrani, zbog složenosti računarske implementacije kodera i dekodera. Ponovnu zainteresovanost u naučnim krugovima potstakli su MacKay i Neal u svom zajedničkom radu iz 90-ih godina [14]. Već je rečeno da H matricu karakteriše mali broj jedinica. Njihova raspodjela u matrici je od velike važnosti, jer od nje zavise performanse koda. Za analizu performansi koda koristi se matrična prezentacija (uobičajeno pomoću kontrolne matrice H) ili predstava pomoću bipartitnog Tanerovog grafa.

Tanerov graf je bipartitni graf koji povezuje simbolske čvorove u oznaci d_j (j=1,2,.....,n) i kontrolne čvorove u oznaci h_j (j=1,2,.....,m). Simbolski čvorovi d_j predstavljaju bite riječi koja se dekoduje, a kontrolni čvorovi h_j predstavljaju jednačine provjere na parnost, koje su zadovoljene za svaku kodnu riječ posmatranog koda. Tanerov graf koji odgovara kontrolnoj matrici

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

ilustrovan je na slici 1. Primjećeno je da prisustvo malih petlji u grafu ima za posledicu značajniju degradaciju dekodera. Na sl. 2. prikazana je jedna takva petlja $h_1 - d_3 - h_3 - d_7 - h_1$.



Sl. 2. Tanerov graf koda (8,4)

Da bi jedan linearni blok kod bio LDPC kod potrebno je da zadovoljava sledeće uslove:

- Kontrolna matrica **H** ima fiksiran broj jedinica u svakoj vrsti, označen sa ρ.
- Matrica H ima fiksiran broj jedinica γ u svakoj koloni.
- Da bi se izbegli kružni tokovi (ciklusi) u bipartitnom grafu neophodno je da u bilo koje dve kolone (ili vrste) dolazi do preklapanja jedinica na maksimalno jednoj poziciji.
- Parametri γ i ρ treba da budu što je moguće manji u odnosu na dužinu kodne reči *n.*

Teško je realizovati ovakav kod upravo zbog 3. uslova. Jedan od načina konstruisanja kontrolne matrice je predložio *Gallager* ali ovaj metod rezultuje kontrolnom matricom koja nema jasnu strukturu i koja je nezgodna za praktičnu implementaciju na hardveru. Umesto slučajno generisanih kodova, u ovom radu će biti razmotrene dvije klase kodova dobijene determinističkim pristupom.

Prvu klasu čine kodovi bazirani na projektivnoj geometriji, koja je kompletno definisana skupom tačaka, skupom linija i informacijom koja tačka se nalazi na kojoj liniji. Neki od parametara *PG* jesu njena dimenzija, *N* i njen rang, označen sa *q*. Svaka linija sadrži tačno po *q*+1 tačku i $\left(q^N+\ldots+q+1\right)\left(q^{N-1}+\ldots q+1\right)/\left(q+1\right)$ linija. Metod konstrukcije korišćenjem *PG* se zasniva na poljima Galoa, $GF(q)$. Dva vektora, $\vec{\mathbf{u}}$ i $\vec{\mathbf{v}}$, predstavljaju istu tačku, ako postoji $\alpha \in GF\ (q)$, $\alpha \neq 0$, pri čemu važi $\vec{\mathbf{u}}=\alpha\vec{\mathbf{v}}$. Linija koja sadrži tačke *A* i *B* je definisana kao $aA+bB$, za $a,b \in GF(q)$, $(a,b)\neq(0,0)$, gdje su ove tačke predstavljene odgovarajućim vektorima. *PG* kodovi i njihova konstrukcija su obrađeni u više detalja u [15]. Prvi korak u konstrukciji *LDPC* kodova baziranih na *PG* je određivanje kontrolne matrice $\mathbf{H}=\left[h_{cv}\right]_{J\times n}$. Postoji 1-na-1 preslikavanje između skupa linija i skupa kontrolnih suma, $\{h_1\ldots h_m\}$, kao i između skupa tačaka i skupa kodnih bita, $\{d_1\ldots d_n\}$. Svaka vrsta u $\mathbf{H}$ predstavlja kontrolnu sumu i svaka kolona predstavlja kodni bit. Kodni bit učestvuje u kontrolnoj sumi ako odgovarajuća linija sadrži odgovarajuću tačku, tj. $h_{ij}=1$ ako v_j učestvuje u c_i, inače važi $h_{ij}=0$. Na ovaj način generiše se *LDPC* (n,k) kod, gdje je $n-k=\text{rank}(\mathbf{H})$. Više detalja na ovu temu moguće je naći u [7]**.**

Konstrukcija kvazi-cikličnih LDPC kodova polazi od protografa [16]. To je Tanerov graf obično sa malim brojem čvorova i grana. Graf $G=(V_b,V_c,E)$ čini skup simbolskih čvorova V_b, skup kontrolnih čvorova V_c i skup grana *E*. Nadalje se pravi *N* kopija grafa G, tako što se učešljaju sve kopije čvorova jednih pored drugih tj. $\left\{v_x^0,\ldots,v_x^{N-1}\right\}$, $x=\left(1,\ldots,n\right)$ za V_b, kao i $\left\{v_x^0,\ldots,v_x^{N-1}\right\}$, $x=\left(1,\ldots,m\right)$ za V_c. Na skup od *N* grana koje odgovaraju jednoj grani iz protografa *G* primjeni se permutacija grana i tako permutovane grane povezuju prošireni graf.

IV. DEKODOVANJE LDPC KODOVA

Dekodovanje LDPC kodova vrši se pomoću iterativnih algoritama dekodovanja. U osnovi imamo dva koncepta dekodovanja: bit-fliping (BF) dekodovanje sa tvrdim odlukama i SPA algoritam dekodovanja sa mekim odlukama. U praksi imamo dosta varijacija ovih algoritama koji se koriste za dekodovanje LDPC kodova. Neki od njih će biti detaljnije opisani u ovom radu.

A. Opis probabilističkog BF algoritma

Bit fliping (BF) predstavlja izuzetno jednostavan algoritam dekodovanja LDPC kodova, predložen u [10]. Zasniva se na određivanju inverzne funkcije, koja pokazuje koliko provjera na parnost koje odgovaraju kontrolnim čvorovima grafa pridruženim *j*-tom simbolu primljene sekvence $\boldsymbol{x}$ nije zadovoljeno

$$\Delta_j^{(GDBF)}(x) \triangleq \sum\nolimits_{i\in M(j)} \oplus_{i\in N(i)} x_i \ . \tag{2}$$

Vrijednost ovako definisane inverzne funkcije je maksimalna u slučaju zadovoljenja svih provjera na parnost. Kako je ova inverzna funkcija indikator pouzdanosti *j*-tog simbola, ona se koristi za donošenje odluke o invertovanju simbola. U jednoj iteraciji invertuju se svi simboli koji zadovoljavaju da im je vrijednost funkcije invertovanja minimalna. Isto tako može se postaviti i prag δ koji određuje koji se svi simboli u jednoj iteraciji mogu invertovati, to su simboli za koje je zadovoljeno $\Delta_j^{(BF)}(x) < (\gamma+1)/2$. Tako se dobija modifikovana primjena kodna riječ **r'**. Koristeći modifikovanu primljenu sekvencu **r'**, dekoder ponovo računa inverzne funkcije i nastavlja ovaj iterativni postupak sve dok se ne dobije validna kodna riječ ili se dostigne maksimalan broj iteracija. U modifikaciji algoritma, nazvanoj probabilistički bit fliping (*Probabilistic Bit Flipping, PBF*) [11], od svih simbolskih čvorova koji bi bili invertovani u BF algoritmu u datoj iteraciji bira se podskup čvorova koji se zaista invertuju. Svaki čvor za koji je zadovoljen preduslov ulazi u podskup za invertovanje sa verovatnoćom *p*.

B. Opis probabilističkog GDBF algoritma

Osnovu GDBF algoritma čini standardni BF algoritam, uz modifikaciju inverzne funkcije (3). U radu [11] je pokazano da se znatno bolji rezultati postižu ako se umjesto izraza (2) inverzna funkcija računa kao

$$\Delta_j^{(GDBF)}(x) \triangleq x_j \oplus y_j + \sum\nolimits_{i\in M(j)} \oplus_{i\in N(i)} x_i \ . \tag{3}$$

Prethodni izraz je veoma efikasan u slučaju prenosa kroz kanal sa ABGŠ, kada se najstabilnija konvergencija ka maksimumu kriterijumske funkcije postiže ako se po iteraciji invertuju biti za koju funkcija $\Delta_j^{(GDBF)}(x)$ ima minimalnu vrijednost.

Da bi obezbedili minimalan broj invertovanja bita po iteraciji, a time i povećali stabilnost konvergencije GDBF algoritma, uvodi se i dodatni uslov za invertovanje bita. Naime u radu [12] je predložen sledeći algoritam:

1. Za svaki od simbola x_j računa se vrijednost inverzne funkcije na osnovu izraza (3).
2. Određuju se pozicije svih simbolskih bita k za koje ova vrijednost minimalna k=argmin $\left\{\Delta_j^{(GDBF)}(x)\right\}$ i oni imaju potreban uslov da bi bili invertovani.
3. Generiše se pomoćna slučajna promjenjiva $a_j \sim unif[0,1]$ i samo ako je $a_j < p$, vrši se invertovanje simbola x_j.
4. Postupak se ponavlja dok se ne dekoduje riječ koja zadovoljava $d \otimes H^T = 0$ ili dok se ne dostigne maksimalan broj iteracija.

C. Opis normalizovanog belief-propagation algoritma

Normalizovani *belief-propagation* (NBP) algoritam se zasniva na određivanju *a posteriori* verovatnoća svakog simbola poruke u funkciji od primljenog signala kodne informacije, izražene u jednačinama provjere na parnost, i karakteristika kanala. Za ovaj metod dekodovanja pogodno je koristiti Tanerov graf. BP algoritam zahtjeva da svaki simbolski čvor d_j šalje svakom od svojih kontrolnih čvorova potomaka procijenu verovatnoće Q_{ij}^x da se čvor d_j nalazi u nekom stanju x, pri čemu je ta procena generisana na osnovu informacija dobijenih od drugih njegovih čvorova potomaka izrazom

$$Q_{ij}^x = \beta f_j^x \prod\nolimits_{k \in M(j)\setminus i} R_{ij}^x , \tag{4}$$

pri čemu je sa 0≤β≤1 označen normalizacioni koeficijent.

Svaki kontrolni čvor h_i šalje svakom od svojih roditeljskih d_j čvorova procijenu vjerovatnoće R_{ij}^x da je i-ta jednačina provjere na parnost, kojoj odgovara ovaj kontrolni čvor zadovoljena ako se simbolski čvor nalazi u stanju x, uzimajući u obzir informacije dobijene od drugih roditeljskih čvorova sa kojima je h_i povezan izrazom

$$R_{ij}^x = \sum\nolimits_{d:d_j=x} P(h_i / d) \prod\nolimits_{k \in N(i)\setminus j} Q_{ik}^{d_k} . \tag{5}$$

Na početku mora se izvršiti inicijalizacija vrijednosti Q_{ij}^x. One se postavljaju na *a priori* procijene primljenih simbola, označenih sa f_j^x, vjerovatnoća da je j-ti simbol u stanju x i zavisi od modela kanala koji se koristi. U slučaju BSC, ove vjerovatnoće se određuju pomoću izraza

$$Q_{ij}^x = x_i \times \log((1-\alpha)/\alpha). \tag{6}$$

Nakon inicijalizacije počinje razmena informacija između simbolskih i kontrolnih čvorova.

Na kraju svake iteracije, procjena za svaki simbol primljenog vektora, predstavljena u binarnom slučaju je procjena za dvije moguće vrijednosti promenljive x, data sa:

$$d^v = \arg\max \prod\nolimits_{k \in M(j)} R_{kj}^x , \tag{6}$$

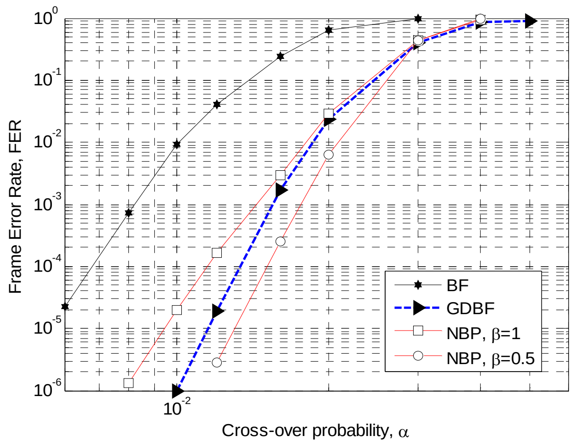
što predstavlja procjenu za simbol na poziciji j. Ako procijenjeni dekodovani vektor d^v zadovoljava uslov da je sindrom jednak nuli, onda se taj kodni vektor smatra ispravnim. U suprotnom nastavlja se sa iteracijom izračunavanja koeficijenata Q_{ij}^x i R_{ij}^x dok dekoder ne dostigne unaprijed definisanu granicu broja iteracija bez pronalaska ispravnog kodnog vektora.

V. Numerički Rezultati

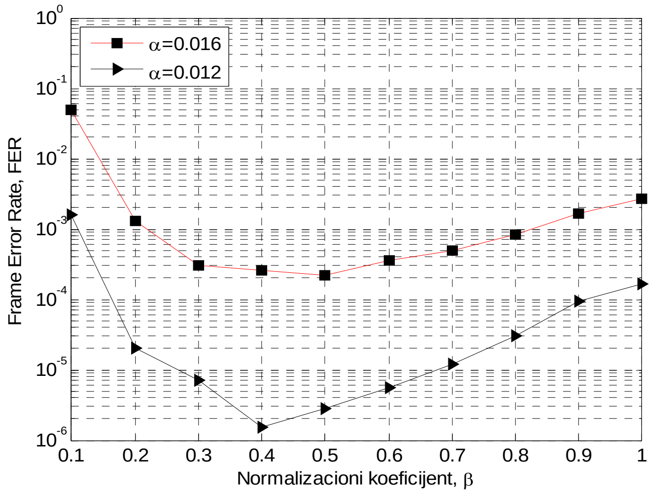
U ovom dijelu rada biće prikazani rezultati u slučaju primjene dvije klase kodova dobijenih determinističkim konstruktivnim metodama, za razne algoritme dekodovanja.

Prvo je razmotren kod PG(1057, 813) koji pripada klasi kodova koji se odlikuju velikom težinom vrsta i kolona ali obezbeđuju veliku brzinu konvergencije i malo kašnjenje pri dekodovanju. Ovaj kod ima kodnim količnik R=0.769 i parametre γ=ρ=33, što kao posledicu ima veliki broj grana koje napuštaju svaki čvor pa zato čak i u slučaju BF dekodera moraju da se koriste XOR i MAJ logička kola sa po 33 ulaza. Performanse ovog koda pri dekodovanju BF, GDBF i NBP algoritmom prikazane su na sl. 3 kada je maksimalan broj iteracija pri dekodovanju L=25.

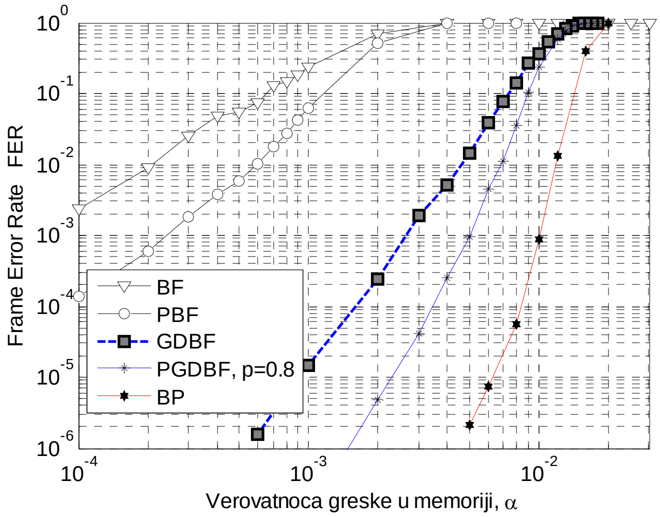
Najbolje performanse se postižu ako se primjeni NBP algoritam, ali se vrijednost normalizacionog koeficijenta mora izabrati veoma pažljivo (zavisnost FER od β data je na sl. 4). Jasno je da su performanse u slučaju primjene BF algoritma značajno lošije nego ako se primjene ostali razmatrani algoritmi. Zanimljivo da se primjenom GDBF dekodera mogu postići rezultati bliski NBP dekoderu uz značajno nižu kompleksnost implementacije. Naime, struktura dekodera se razlikuje of BF samo po tome što u svakom varijabilnom čvoru postoji jedno dodatno XOR kolo i što je potrebno raditi adaptaciju praga u svakoj iteraciji ali se kroz graf i dalje šalju tvrde odluke za razliku od NBP (za više detalja vidjeti [13]).



Sl. 3. Performanse raznih dekodera za kod PG(1057, 813), u zavisnosti od nivoa nepouzdanosti memorije.



Sl. 4. Performanse NBP dekodera za kod PG(1057, 813), razne vrijednosti normalizacionog koeficijenta.



Sl. 5. Performanse raznih dekodera za kod QC(5219, 4300), u zavisnosti od nivoa nepouzdanosti memorije.

Kvaziciklični kod QC(5219, 4300) sa kodnim količnikom R=0.824 i parametrima γ=3, ρ=17 za maksimalno L=100 iteracija pri dekodovanju ima performanse prikazane na sl. 5. U ovom slučaju BP algoritam može da se primjeni i bez normalizacije (β=1) i rezultati su tada najbolji, ali kroz graf se moraju prenositi meke poruke, iako su na izlazu kanala dostupne samo tvrde procjene. Pošto je u slučaju ovog koda γ=3, dekoderi bazirani na BF se realizuju pomoću troulaznih MAJ logičkih kola i kompleksnost dekodera je onda drastično manja nego kod kodova konstruisanih na konačnim geometrijama (PG kao u prethodnom primeru ili EG kao u [2]). Kao i u prethodnom slučaju, malo povećanje kompleksnosti rezultuje velikim poboljšanjem performansi pri primeni GDBF algoritma. Probabilističke modifikacije dekodera sa tvrdim odlučivanjem dovode do znatnog poboljšanja performansi, a posebno se izdvaja PGDBF dekoder zbog dobrih performansi i umjerene kompleksnosti.

V. Zaključak

U radu je dat pregled dvije klase LDPC kodova i njihove performanse u slučaju kada se primjene algoritmi dekodovanja pogodni za primjenu u fleš SLC memorijama.

Pokazano je da PG kodovi obezbjeđuju veoma brzo dekodovanje i velike protoke podataka ali uz relativno veliku kompleksnost dekodera. Za ove kodove NBP dekoder obezbjeđuje najbolje performanse ali samo ako se normalizacioni koeficijent izabere optimalno. Kompleksnost dekodera se može djelimično smanjiti ako se koristi GDBF algoritam, bez znatne degradacije performansi.

Kada je kritična kompleksnost dekodera i utrošak energije u njemu, optimalna je primena QC kodova. Tada je poželjno koristiti PGDBF dekoder jer unijeti probabilizam dovodi do razbijanja traping setova prisutnih kod ove klase kodova, posebno kada je vrednost parametra γ mala.

Literatura

[1] Y. Li, S. Lee, and et al. "A 16 Gb 3b/cell NAND Flash Memory in 56nm With 8MB/s Write Rate", *In Proc. of ISSCC*, pp 506–632, Feb. 2008.

[2] J. Kim, W. Sung, "Low-energy error correction of NAND Flash memory through soft-decision decoding", *EURASIP Journal on Advances in Signal Processing*, 012:195, http://asp.eurasipjournals.com/content/2012/1/195.

[3] A. Hocquenghem, "Codes correcteurs d'erreurs", *Chiffres*, Vol. 2, pp. 147-156, 1959.

[4] R. C. Bose, D. K. Ray-Chaudhuri, "On a Class of Error Correcting Binary Group Codes", *Inform. Control*, Vol. 3, pp. 68-79, 1960.

[5] D. Drajić, P. Ivaniš, *Uvod u teoriju informacija i kodovanje*, Akademsaka misao, Beograd, 2009.

[6] E. F. Haratsch, "LDPC Code Concepts and Performance on High-Density Flash Memory," *Flash Memory Summit 2014*
Santa Clara, CA, Aug. 5 - 7, 2014.

[7] S. Brkić, D. Lazarević, P. Ivaniš, "FPGA implementacija sum-product algoritma za dekodovanje LDPC kodova", *INFOTEH JAHORINA 2013*, Vol 12, Ref. KST-3-2, Istočno Sarajevo, 20-22. Mart 2013, str. 428-433.

[8] T. Ž. Jovanović, P. Ivaniš, "Procena performansi dekodovanja BCH kodova pomoću trelisa i BCJR algoritma", *INFOTEH JAHORINA 2010*, Vol. 9, Ref. B-I-6, Istočno Sarajevo, 17-19. Mart 2010, str. 149-152.

[9] J. Wang, K. Vakilinia, T.-Y. Chen, T. Courtade, G. Dong, T. Zhang, H. Shankar, and R. Wesel, **"**Enhanced Precision Through Multiple Reads for LDPC Decoding in Flash Memories**",** *IEEE Journal Sel. Areas Commun.*, Vol. 32, No. 5, May 2014

[10] R. G. Gallager, *Low Density Parity Check Codes*, MIT Press, Cambridge, Mass., 1963.

[11] N. Miladinovic and M. P. C. Fossorier, "Improved bit-flipping decoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, Vol. 51, No. 4, pp. 1594-1606, Apr. 2005.

[12] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit flipping algorithms for decoding LDPC codes," *IEEE Trans. Commun.*, Vol. 58, No. 6, pp. 1610–1614, Jun. 2010.

[13] O.-Al Rasheed, P. Ivanis, B. Vasic, "Fault-Tolerant Probabilistic Gradient-Descent Bit Flipping Decoder," *IEEE Commun. Letters*, Vol. 18, No. 9, pp. 1487–1490, Sept. 2014.

[14] D.McKey, R. Neal, "Good codes based on very sparse matrices," *Cryptography and Coding, 5th IMA Conf, CBoyd,* Ed., Lecture Notes in Computer Science, Oct. 1995

[15] Y. Kou, S. Lin, M. P. C. Fossorier, "Low-Density Parity-Check Codes Based on Finite Geometries: A Rediscovery and New Results", *IEEE Trans. Inform. Theory*, Vol. 47, No. 7, 2001.

[16] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello, "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. Inform. Theory*, Vol. 50, No. 12, pp. 2966–2984, Dec. 2004.