پیاده سازی بهینه دکدر QC-LDPC با توان عملیاتی بالا بر پایه الگوریتم دکدینگ لایه‌بندی‌شده

طاهره حیدری [۱]، ابومسلم جان‌نثاری [۲]

۱. کارشناسی ارشد برق الکترونیک،دانشگاه تربیت مدرس

۲. استادیاردانشکده برق و کامپیوتر،دانشگاه تربیت مدرس، Jannesari@modares.ac.ir

تاریخ دریافت: ۹۲/۱۱/۲۸ تاریخ پذیرش: ۹۳/۱۱/۱۲

چکیده

در این مقاله یک دکدر با توان عملیاتی بالا برای کدهای QC-LDPC ارائه شده است. بر پایه الگوریتم دکدینگ لایه‌بندی شده، یک تکنیک جدید بر اساس همزمانی پردازش سطرها و ستون‌ها ارائه شده است که از تعداد تکرارها کاسته می‌شود و در نتیجه توان عملیاتی افزایش می‌یابد. دکدر پیشنهادی برای طول کدهای متغیر با نرخ ۱/۲ و تعداد ۷ بیت کوانتیزیشن در استاندارد ۸۰۲/۱۶e ارائه شده است. بر پایه طرح پیشنهادی، دکدر QC-LDPC برای طول کد ۲۳۰۴، با استفاده از نرم‌افزار Design Compiler و در تکنولوژی ۱۳۰ نانومتر CMOS سنتز شده است. نتایج به‌دست آمده نشان می‌دهد با میزان توان مصرفی ۱۵۶,۳۹ mW و سطح تراشه ۵,۰۹mm^2 می‌توان به توان عملیاتی ۱۹۸Mb/s در فرکانس کاری ۱۰۰MHz و حداکثر ۱۰ تکرار دست یافت.

کلیدواژه

کدهای بررسی توازن کم چگال، کدهای QC-LDPC، الگوریتم دکدینگ لایه‌بندی شده، ساختار نیمه موازی.

مقدمه

کدهای بررسی توازن کم چگال[۱] دسته‌ای از کدهای اصلاح خطای پیشرو[۲] می‌باشندکه ابتدا توسط گلگر در سال ۱۹۶۰ معرفی شدند[۱]. به علت کمبود امکانات سخت‌افزاری آن زمان،پیاده‌سازی این کدها امکان‌پذیر نبود. به دلیل عملکرد بالای این کدها و نزدیکی عملکرد آنها به حد شانون [۲]، با پیشرفت تکنولوژی و امکان پیاده‌سازی، این کدها دوباره مطرح شدند. توانایی اصلاح خطای این کدها در مقایسه با کدهایی که تاکنون شناخته شده‌اند، تنها کسری از دسی‌بل از حد شانون فاصله‌دارند. به همین منظور، این کدها در بسیاری از استانداردهای ارتباطی رایج از جمله DVB-S2، WiFi IEEE 802.11، WiMAX IEEE 802.16eمورداستفاده قرارگرفته‌اند [۳]و [۴]و [۵].

الگوریتمی که گلگر برای دکدینگ ارائه کرده بود، بر اساس احتمالات بود [۱]. بر اساس نوع کاربرد این کدها، الگوریتم‌های دیگری بعد از کار گلگر از جمله الگوریتم انتشار عقیده[۳]، الگوریتم حداقل مجموع[۴] توسعه یافته است. این الگوریتم‌ها، الگوریتم‌های تکرارشونده هستند و معمولاً الگوریتم‌های عبور پیام نامیده می‌شوند. پیاده‌سازی دکدرهای LDPCبه ۳ دسته کاملاً موازی، کاملاً سریال و سریال موازی تقسیم‌بندی می‌شود. در پیاده‌سازی کاملاً موازی، هر چک‌نود (بیت‌نود)، دارای یک واحد پردازشگر مجزا می‌باشد؛ و واحدهای پردازشگر سطری (ستونی)، به طور همزمان کار می‌کنند. در این نوع پیاده‌سازی با صرف توان مصرفی و حجم تراشه بالا، باعث به‌دست‌آوردن توان عملیاتی بالا می‌شود [۶]. در حالت کاملاً سریال، برای تمام چک‌نودها (بیت‌نودها)، یک واحد پردازشگر در نظر گرفته می‌شود. واحد پردازشگر سطری (ستونی)، پس از انجام پردازش یک سطر (ستون)، پردازش سطر (ستون) بعدی را انجام می‌دهد. به همین خاطر پیاده‌سازی به این روش، دارای سرعت پایینی است و در نتیجه توان عملیاتی پایینی دارد. در پیاده‌سازی به روش سریال موازی، یک هم‌ارزی بین سخت‌افزار و سرعت وجود دارد. در این حالت، برای تعداد معینی چک‌نود (بیت‌نود)، یک واحد پردازشگر در نظر گرفته می‌شود. در هر زیر تکرار، واحدهای پردازشگر به طور همزمان تعداد معینی گره چک (گره متغیر) را پردازش می‌کنند؛ و هنگامی که عملیات پردازش انجام شد، تعداد معینی گره چک (گره متغیر) بعدی را پردازش می‌کنند. با استفاده از این روش، از میزان سخت‌افزار نسبت به حالت کاملاً موازی کاسته می‌شود و به سرعت بالاتری نسبت به حالت کاملاً سریال دست می‌یابد.

1 . Low density parity check codes(LDPC)
2 . Forward error correction
3 . Belief propagation
4 . Min_Sum algorithm

−1	94	73	−1	−1	−1	−1	−1	55	83	−1	−1	07	00	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1
−1	27	−1	−1	−1	22	79	09	−1	−1	−1	12	−1	00	00	−1	−1	−1	−1	−1	−1	−1	−1	−1
−1	−1	−1	24	22	81	−1	33	−1	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1	−1	−1	−1
61	−1	47	−1	−1	−1	−1	−1	65	25	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1	−1	−1
−1	−1	39	−1	−1	−1	84	−1	−1	41	72	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1	−1
−1	−1	−1	−1	46	40	−1	82	−1	−1	−1	79	00	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1
−1	−1	95	53	−1	−1	−1	−1	−1	14	18	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1
−1	11	73	−1	−1	−1	02	−1	−1	47	−1	−1	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1	−1
12	−1	−1	−1	83	24	−1	43	−1	−1	−1	51	−1	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1
−1	−1	−1	−1	−1	94	−1	59	−1	−1	70	72	−1	−1	−1	−1	−1	−1	−1	−1	−1	00	00	−1
−1	−1	07	65	−1	−1	−1	−1	39	49	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	00	00
43	−1	−1	−1	−1	66	−1	41	−1	−1	−1	26	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	00

شکل ۱: ماتریس بررسی توازن استاندارد WiMAX

توان عملیاتی خروجی یک دکدر LDPC توسط رابطه زیر محاسبه می‌گردد [۷]:

$$\text{Throughput} = \frac{\text{Freq}*\text{Block Length}}{\text{Cycles per Iter}*\text{Num of Iter}} \quad (۱)$$

با توجه به رابطه فوق، با افزایش فرکانس کاری، کم کردن تعداد تکرارها و هم‌چنین کاهش تعداد چرخه‌ها (تاخیر)در هر تکرار، می‌توان توان عملیاتی را افزایش داد.

در این مقاله یک ساختار جدید برای پیاده‌سازی دکدر LDPC بر پایه الگوریتم لایه‌بندی‌شده[۵] ارائه شده است؛ که با توجه به آناز تعداد چرخه‌ها در هر تکرار کاسته و باعث افزایش توان عملیاتی خروجی می‌گردد. برای داشتن یک تناسب بین سرعت و سخت‌افزار، پیاده‌سازی به صورت نیمه موازی می‌باشد.

در ادامه، در بخش دوم به ارائه پیش‌زمینه‌ای از کدهای LDPC و الگوریتم SPA و دیگر الگوریتم‌های اصلاح‌شده پرداخته و در بخش سوم، طرح ارائه‌شده برای افزایش توان عملیاتی مورد بررسی قرار گرفته است. در بخش چهارم نتایج پیاده‌سازی و مقایسه عملکرد روش پیشنهادی با طرح‌های موجود بیان می‌گردد.

کدهای LDPC

کدهای LDPC، دسته‌ای از کدهای بلوکی هستند که می‌توان آن‌ها را توسط ماتریس‌مولد با ابعادM*N، که M معرف تعداد معادله‌های بررسی توازن[۶] و N تعداد بیت ارسالی می‌باشد، بیان کرد. اکثر درایه‌های این ماتریس ۰ و تعداد کمی ۱ می‌باشد. این پراکندگی ماتریس بررسی توازن در پیچیدگی تکرار دکدینگ اهمیت بسیار دارد. به گونه‌ای که با افزایش طول کد، این پیچیدگی افزایش می‌یابد [۳]. کدهایLDPC را می‌توان به صورت گرافیکی توسط گراف تنر[۷] بیان کرد. گراف تنراز دو دسته نود، N نود برای بیت‌های کلمه کدکه به آن‌ها بیت‌نود گفته می‌شود و M نود برای معادلات بررسی توازن که به آن‌ها چک‌نود گفته می‌شود، تشکیل شده است. ماتریس H این نوع کدها به دو صورت QC [۸]و تصادفی ساخته می‌شوند [۸]. در نوع تصادفی، ۱ها به صورت تصادفی و به صورت کاملا اسپارس[۹] در تمام ماتریس توزیع می‌شوند. در ساختار کدهای QC-LDPC، ماتریس H ترکیبی از زیر ماتریس‌های چرخشی می‌باشد که اتصالات شبکه‌ای بین واحدهای پردازش سطری[۱۰] و واحدهای پردازش ستونی[۱۱] را تعیین می‌کند [۹]. کدهایQC_LDPC، به خاطر نظمی که در پیاده‌سازی ماتریس H دارند، نسبت به کدهای تصادفی از پیچیدگی کمتری و از سرعت بالاتری برخوردار هستند [۱۰ و ۱۱]. شکل ۱ ماتریس بررسی توازن را برای استاندارد IEEE 802.16e نشان می‌دهد که یک کد QC-LDPC می‌باشد.

الگوریتم حداقل مجموع

همان طور که بیان شد الگوریتم انتشار عقیده[۱۲] که معمولاً با نام SPA هم نامیده می‌شود، برای دکد کردن کدهای LDPC به کار می‌رود. این الگوریتم از دو دسته عملیات پردازش سطری و ستونی تشکیل شده است. در پردازش سطری، چک‌نود، پیام‌های بیت‌نود به چک‌نود (پیام‌های بتا) را از بیت‌نودهای متصل به آن دریافت می‌کند و پس از انجام عملیات، نتایج (پیام‌های آلفا) را به بیت‌نودها ارسال می‌کند. در پردازش ستونی، بیت‌نودها با توجه به پیام دریافت شده (پیام‌های آلفا)، از چک‌نودهای متصل به آن‌ها و هم چنین پیام دریافتی از کانال مقدار بیت دکد شده را تخمین می‌زنند و دوباره نتایج (پیام‌های بتا) را به چک نودها ارسال می‌کنند.

این الگوریتم از نظر عملکرد، بهترین الگوریتم می‌باشد؛ ولی درعین‌حال بسیار پیچیده است. برای بهبود الگوریتم SPA و کم کردن پیچیدگی توابع محاسباتی پیچیده به‌کاررفته در آن، الگوریتم MSA ارائه شده است. این الگوریتم که ساده‌شده

5.Layer decoding Algorithm
6.Parity check
7.Tanner graph

8.Quasi Cyclic
9.Sparse
10.Check node updating units(CNUs)
11.Variable node updating units(VNUs)
12.Belief Propagation Algorithm(BPA)

الگوریتم SPA می‌باشد، نسبت به الگوریتم SPA از پیچیدگی محاسباتی کمتری برخوردار است و هم‌چنین دیگر نیازی به دانستن پارامتر سیگنال به نویز برای کانال AWGN نمی‌باشد. الگوریتم MSA را به طور خلاصه به صورت زیر می‌توان بیان کرد:

در معادلات زیر α_{ij} پیام ارسالی از چک نود i به بیت نود j، β_{ij} پیام ارسالی از بیت نود j به چک نود i و λ_j اطلاعات دریافتی از کانال می‌باشد. V(i)\j به عنوان مجموعه بیت‌نودهای متصل به چک‌نود C_i به جز بیت‌نود j و C(i)\j به عنوان مجموعه چک نودهای متصل به بیت نود V_i به جز چک نود j تعریف می‌شود.

بیت‌نودها در ابتدا با مقادیری که از کانال دریافت می‌شود، مقداردهی اولیه می‌شوند و آن را به عنوان پیام به چک‌نودها ارسال می‌کنند:

$$\beta_{ij} = \lambda_i , i = 0,1, \dots, N-1 \qquad (۲)$$

در قسمت پردازش سطری، که توسط CNU انجام می‌گیرد، پیام ارسالی چک‌نود به بیت‌نود، با استفاده از پیام‌های دریافتی از بیت-نودها، محاسبه می‌شود. . برای بهبود عملکرد اصلاح خطا این پیام با ضریبی کمتر از یک ضرب می‌گردد و پیام حاصل ارسال می‌گردد [۱۲]. این پیام از رابطه زیر محاسبه می‌شود:

$$\alpha_{ij} = S_{MS} \prod_{j\epsilon V(i)\backslash j} \operatorname{sign}\left(\beta_{ij}\right) * \min_{j\epsilon V(i)\backslash j}\left(|\beta_{ij}|\right) \qquad (۳)$$

که در رابطه فوق S_{MS} ضریب اصلاح می‌باشد.

برای هر سطر i دامنه پیام α به صورت زیر محاسبه می‌گردد:

$$Min1_i = \min_{j\epsilon V(i)}(|\beta_{ij}|) \qquad (۴)$$

$$Min2_i = 2nd\ \min_{j\epsilon V(i)}(|\beta_{ij}|) \qquad (۵)$$

و سپس

$$|\alpha_{ij}| = Min2_i \text{ if } j == Min1_{index} \quad else\ Min1_i \qquad (۶)$$

پیام محاسبه‌شده به بیت‌نودها ارسال می‌گردد. باVNU توجه به پیام دریافت شده از چک‌نودها و هم چنین اطلاعات دریافتی از کانال، پردازش ستونی را انجام می‌دهد و پیام ارسالی از بیت‌نود به چک‌نود را مطابق با رابطه زیر محاسبه می‌کند:

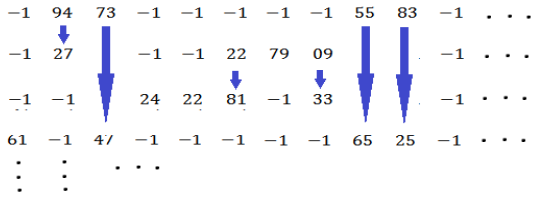
$$\beta_{ij} = \lambda_j + \sum_{i\epsilon C(j)\backslash i} \alpha_{ij} \qquad (۷)$$

و مقداری که برای هر بیت نود در نظر گرفته می‌شود از رابطه زیر محاسبه می‌کند:

$$LQ_j = \lambda_j + \sum_{i\epsilon C(j)} \alpha_{ij} \qquad (۸)$$

الگوریتم دکدینگ لایه‌بندی شده

در الگوریتم SPA پیام‌های ارسالی از چک‌نود به بیت‌نود و بیت‌نود به چک‌نود در واحدهای پردازش جداگانه محاسبه و به‌روز می‌گردد و به دیگری ارسال می‌گردد. با توجه به رابطه (۷)، پیام چک‌نود به بیت‌نود تا زمانی که تمام پیام‌های بیت‌نود به چک‌نود محاسبه نگردد، محاسبه نمی‌شود و بالعکس. با استفاده از پیاده‌سازی سریال‌موازی ، ماتریس H به Lr لایه سطری و Lc لایه ستونی تقسیم می‌گردد، به گونه‌ای که وزن ستونی (سطری) هر لایه حداکثر برابر با یک گردد. سطرهای هر لایه به طور همزمان پردازش می‌شوند، هنگامی که لایه اول پردازش شد لایه دوم پردازش می‌گردد و هر لایه بعد از لایه دیگر پردازش می‌گردد. هنگامی که پردازش سطری پایان یافت، پردازش ستونی آغاز می‌گردد. در این الگوریتم نیز ماتریس H به L لایه سطری تقسیم می‌شود. با این تفاوت که هر لایه به صورت مستقل پردازش می‌شود و از اطلاعات به‌دست آمده در پردازش لایه بعدی استفاده می‌شود(شکل ۲). این روش باعث کاهش تعداد تکرارها و افزایش سرعت همگرایی می‌شود . روابط استفاده شده در این الگوریتم مشابه روابط الگوریتم MSA می‌باشد. این الگوریتم را می‌توان به صورت زیر خلاصه کرد:



شکل ۲. انتقال پیام بین لایه‌ها در الگوریتم لایه‌بندی شده

Layered Decoding Algorithm

for iteration = 1 to iteration_max

if iteration = 1

$$\beta_{ij} = \lambda_i , i = 0,1, \dots, N-1 \qquad (۹)$$

end

for l = 1 to L

if l = 1

$$\beta_{ij}^{iter,l} = LQ_j^{iter,L} - \alpha_{ij}^{(iter-1),l} \qquad (۱۰)$$

else

$$\beta_{ij}^{iter,l} = LQ_j^{iter,l-1} - \alpha_{ij}^{(iter-1),l} \qquad (۱۱)$$

end

$$\alpha_{ij} = S_{MS} \prod_{j\epsilon V(i)\backslash j} \operatorname{sign}\left(\beta_{ij}\right) * \min_{j\epsilon V(i)\backslash j}(|\beta_{ij}|) \qquad (۱۲)$$

$$LQ_j^{iter,l} = \beta_{ij}^{iter,l} + \alpha_{ij}^{iter,l} \qquad (۱۳)$$

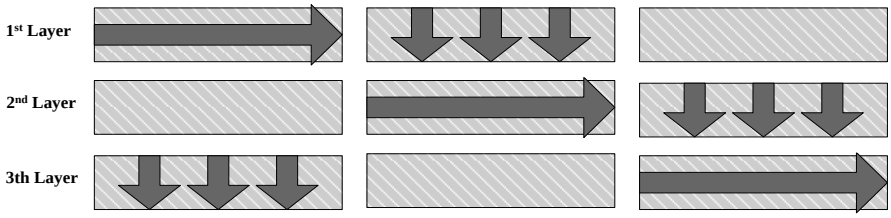
end

$$LQ_j = \lambda_j + LQ_j^{iter,l} \qquad (۱۴)$$

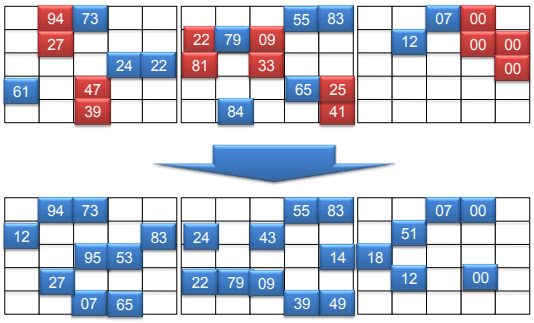
بررسی روش پیشنهادی

در الگوریتم دکدینگ لایه‌بندی‌شده ابتدا پردازش سطری انجام می‌شود و هنگامی‌که پردازش سطری تمام لایه‌ها پایان یافت، پردازش ستونی آغاز می‌شود. در طرح ارائه‌شده نیز ماتریس H به Lr لایه سطری و Lc لایه ستونی تقسیم می‌شود. هنگامی که لایه اول پردازش شد، لایه دوم پردازش می‌گردد و همزمان عملیات پردازش ستونی لایه اول هم انجام می‌گیرد (شکل۳). به این ترتیب با انجام همزمان پردازش سطری و ستونی از تعداد چرخه‌ها در یک تکرار کاسته می‌شود و باعث بهبود توان عملیاتی می‌شود.

در شکل ۴، پنج سطر اول ماتریس بررسی توازن استاندارد WiMAX به همراه شیفت هر زیر ماتریس نشان داده‌شده است. هنگامی که لایه دوم در حال پردازش سطری می‌باشد، لایه اول در حال پردازش ستونی می‌باشد. برای زیر ماتریس‌های ۲ و ۱۴، در حالت معمولی، پردازش سطری با استفاده از اطلاعاتی که زیر ماتریس‌های دوم و چهاردهم محاسبه می‌شود، انجام شود. ولی در این حالت به دلیل این که پردازش ستونی زیر ماتریس دوم در لایه اول و پردازش سطری زیر ماتریس دوم در لایه دوم، به صورت هم-زمان انجام می‌گیرد و هم‌چنین، پردازش ستونی زیر ماتریس چهاردهم در لایه اول و پردازش سطری زیر ماتریس چهاردهم در لایه دوم به صورت هم‌زمان انجام می‌گیرد، پردازش سطری این زیر ماتریس‌ها با استفاده از اطلاعات قبلی انجام می‌شود. برای جلوگیری از افت عملکرد دکدینگ، باید لایه‌ها به گونه‌ای پردازش شوند، که هیچ‌گونه هم‌پوشانی زیر ماتریسی بین دو لایه متوالی وجود نداشته باشد (شکل ۴). با توجه به خاصیت ارتگنال بودن ماتریس H، جابه‌جا کردن سطرها تأثیری در عملکرد دکدینگ ندارد. بنابراین ماتریس H به صورت نشان داده‌شده در شکل ۵ اصلاح می‌گردد.



شکل۳. پردازش سطری، ستونی



شکل ۴.هم‌پوشانی لایه‌های سطری

−1	94	73	−1	−1	−1	−1	−1	55	83	−1	−1	07	00	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1
12	−1	−1	−1	83	24	−1	43	−1	−1	−1	51	−1	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1
−1	−1	95	53	−1	−1	−1	−1	−1	14	18	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1
−1	27	−1	−1	−1	22	79	09	−1	−1	−1	12	−1	00	00	−1	−1	−1	−1	−1	−1	−1	−1	−1
−1	−1	07	65	−1	−1	−1	−1	39	49	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	00	00
−1	−1	−1	−1	46	40	−1	82	−1	−1	−1	79	00	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1
61	−1	47	−1	−1	−1	−1	−1	65	25	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1	−1	−1
−1	−1	−1	−1	−1	94	−1	59	−1	−1	70	72	−1	−1	−1	−1	−1	−1	−1	−1	−1	00	00	1
−1	11	73	−1	−1	−1	02	−1	−1	47	−1	−1	−1	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1
43	−1	−1	−1	−1	66	−1	41	−1	−1	−1	26	07	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	00
−1	−1	39	−1	−1	−1	84	−1	−1	41	72	−1	−1	−1	−1	−1	00	00	−1	−1	−1	−1	−1	−1
−1	−1	−1	24	22	81	−1	33	−1	−1	−1	00	−1	−1	00	00	−1	−1	−1	−1	−1	−1	−1	−1

شکل ۵. ماتریس بررسی توازن اصلاح‌شده در استاندارد WiMAX

طرح کلی دکدر پیشنهادی در شکل ۶.الف نشان داده شده است. دکدر از یک واحد CNPVNP تشکیل شده است. واحد حافظه LLR در ابتدا برای ذخیره اطلاعات دریافتی از کانال و در طول پروسه برای ذخیره مقادیر محاسبه‌شده برای بیت‌نودها به کار می‌رود. ۴ واحد حافظه برای ذخیره کمترین مقادیر، علامت هر پیام و علامت هر سطر می‌باشد. قسمت CNPVNP(شکل ۶.ب) از ۹۶ واحد پردازش سطری، ستونی تشکیل شده است که به طور همزمان کار می‌کنند. در هر چرخه پیام بتا با استفاده از LQ_i و کمترین مقادیر محاسبه‌شده (روابط۹و۱۰)، و سپس کمترین مقادیر جدید محاسبه می‌شود. این مقادیر برای استفاده در تکرار بعدی در حافظه خارجی ذخیره می‌شوند.

می‌توان به جای ذخیره کردن Min_1^{st} , Min_2^{nd}، مقدار حاصل از ضرب آن‌ها در ضریب اصلاح را ذخیره کرد. با جایگذاری رابطه (۱۲) در روابط (۱۰) و (۱۳) داریم:

$$sign = \prod_{j \in V(i)\setminus j} sign\left(\beta_{ij}\right) \quad (۱۵)$$

$$\beta_{ij} = LQ_i - S_{MS} * Min_{old} * sign_{old} \quad (۱۶)$$

$$LQ_j^{iter,l} = B_{ij}^{iter,l} + S_{MS} * Min_{new} * sign_{new} \quad (۱۷)$$

حال با ضرب کردن ضریب اصلاح در کمترین مقدار و

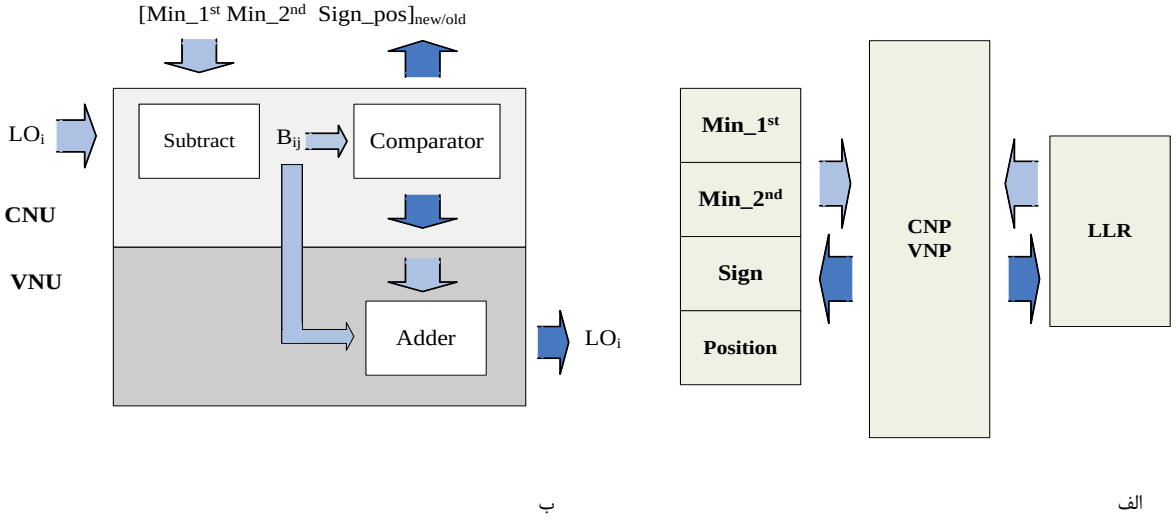
$$Min = S_{MS} * Min \quad (۱۸)$$

جایگذاری در روابط (۱۶) و (۱۷) داریم:

$$\beta_{ij} = LQ_j - Min_{old} * sign_{old} \quad (۱۹)$$

$$LQ_j^{iter,l} = B_{ij}^{iter,l} + Min_{new} * sign_{new} \quad (۲۰)$$

با مقایسه روابط (۱۶) و (۱۷) با روابط (۱۸)، (۱۹) و (۲۰) مشاهده می‌شود که از تعداد ضرب کننده‌های مورد نیاز کاسته می‌شود. علاوه بر این، با توجه به این‌که مقادیر در ضریبی کمتر از یک ضرب می‌شوند، از میزان حافظه مورد نیاز نیز کاسته می‌شود.



شکل ۶. الف)طرح کلی دکدر ب) واحد CNPVNP

نتایج پیاده‌سازی

دکدر پیشنهادی برای نرخ ۱/۲ و طول کد ۲۳۰۴ و تعداد ۷ بیت رزولوشن طراحی شده است. برای ارزیابی عملکرد ساختار پیشنهادی، طرح پیشنهادی در تکنولوژی CMOS 130 nm و ولتاژ ۱,۲ ولت، فرکانس کاری ۱۰۰MHz با استفاده از Design Compiler سنتز شده است. دکدر پیشنهادی قابل‌استفاده برای طول کدهای ۵۷۶:۹۶:۲۳۰۴ می‌باشد. در جدول ۱، نتایج حاصل از شبیه‌سازی با طرح‌های قبلی مقایسه شده است. میزان توان عملیاتی به دست آمده در فرکانس کاری ۱۰۰MHz و حداکثر تعداد تکرار ۱۰، برابر با ۱۹۸ Mb/s می‌باشد. با توجه به رابطه (۱)، توان عملیاتی با تعداد تکرارها رابطه عکس و با فرکانس کاری رابطه مستقیمی دارد، برای داشتن یک مقایسه صحیح برای مقایسه توان عملیاتی ، توان عملیاتی به مقیاس فرکانس کاری ۱۰۰MHz و تعداد تکرار ۱۰ برده می‌شوند. با مقایسه نتایج توان عملیاتی مقیاس بندی شده، دو طرح پیشنهادی، دارای توان عملیاتی به مراتب بهتری نسبت به طرح‌های ارائه‌شده در [۱۳،۱۴و۱۵] می‌باشد. با توجه به این‌که سطح تراشه به نوع استاندارد به‌کاررفته و در نتیجه ولتاژ مرجع در استاندارد بستگی دارد، دکدر پیشنهادی دارای سطح تراشه کمتری نسبت به طرح‌های ۱۳ و ۱۴ است.

جدول ۱. مقایسه طرح پیشنهادی با طرح‌های موجود

	دکدر ارائه‌شده	[۱۳]	[۱۴]	[۱۵]
نرخ کد	۱/۲	۱/۲و۲/۳و۳/۴	۱/۲و۲/۳و۳/۴	۱/۲و۲/۳و۳/۴
تعداد تکرارها	۱۰	۱۵	۲۰	۱۰
توان عملیاتی(Mb/s)	۱۹۸	۲۰۵	۱۰۵	۶۴۰
توان عملیاتی مقیاس شده (Mb/s)	۱۹۸	۱۱۸	۱۴۰	۵۱۲
سطح تراشه(mm^2)	۵،۰۶	۶/۳	۶،۲۶	۱۴،۳
توان مصرفی(mW) **در فرکانس کاری**	۱۵۶،۳۹@۱۰۰	۲۷۰@۱۲۵	۲۶۴@۱۵۰	۷۸۷@۱۲۵
فرکانس کاری(MHz)	۱۰۰	۲۶۰	۱۵۰	۱۲۵
تکنولوژی CMOS	۱۳۰	۱۳۰	۹۰	۱۸۰

نتیجه‌گیری

در این مقاله یک ساختار جدید برای پیاده‌سازی دیکدر QC-LDPC برای افزایش توان عملیاتی برای استاندارد ۸۰۲/۱۶e ارائه شد. با همزمان کردن پردازش سطری و ستونی و هم‌چنین استفاده از الگوریتم لایه‌بندی، از تعداد تکرارها و هم‌چنین تعداد چرخه‌ها در هر تکرار کاسته شد و توان عملیاتی حدود Mb/s ۱۹۸ به‌دست آمد. روش پیشنهادی همچنین مصرف توان را در دیکدر به میزان قابل‌توجهی کاهش داده است.

مرجع‌ها

[1] R. Gallager, "Low-density parity-check codes," *IRE Trans. Information Theory*, pp. 21-28, January1962.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 33, pp. 457–458, Mar. 1997.

[3] F. Kienle, T. Brack, and N.Wehn, "A synthesizable IP core for DVB-S2LDPC code decoding," *inProc. DATE*, pp. 100–105, 2005.

[4] *Wireless LAN medium access control (MAC) and physicallayer(PHY)specifications: enhancementsfor higherthroughput*, IEEE Std. P802.11 n/D7 .0, 2008.

[5] *IEEE Standard for Local and Metropolitan Area Networks, Part 16:Air Interface for Fixed and Mobile Broadband Wireless Access Systems, Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1*,IEEEStd802.16e-2005 and IEEE Std 802.16-2004/Cor1-2005, Feb. 2006.

[6] K. He, J. Sha, L. Li, Z. Wang, "Low Power Decoder Design for QC-LDPC Codes," *ISCAS*, pp .3937 – 3940, June. 2010.

[7] X.Huang, Z. Wang, "High-throughput layered decoder implementation for quasi-cyclic LDPC codes," *IEEE Journal on Selected Areas in Communications*, VOL. 27, NO. 6, pp. 985-994, August 2009.

[8] Y. Chen, K. K. Parhi, "Overlapped message passing for quasi-cyclic low-density parity check codes," *IEEE Trans. on Circuits and Systems I, Reg. Papers*, VOL. 51, NO. 6, pp. 1106-1113, June. 2004.

[9] F. Yi , P. Wang , "Low Complexity Decoding Algorithm of QC-LDPC Code," *APSCC*, pp. 531 – 534, Dec. 2010.

[10] Z. Li, L. Chen, L. Zeng, S. Lin, and W. Fong, "Efficient encoding of quasi-cyclic low-density parity check codes," *IEEE Transaction. Communications*, vol. 54, pp. 71–81, Jan. 2006.

[11] K. Zhang, X. Huang, and Z. Wang, "A dual-rate LDPC decoder for china multimedia mobile broadcasting systems," *IEEE Transactions on Consumer Electronics*, Vol. 56, pp. 399-407, 2010.

[12] J. Chen, M. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," *IEEE Transaction on Communications*, Vol. 50, pp. 406–414, 2002.

[13] S. Huang, D. Bao, B. Xiang, Y. Chen and X. ZengA, "Flexible LDPC Decoder Architecture Supporting Two Decoding Algorithms," *ISCAS*, pp 3929 – 3932, June. 2010.

[14] K. C.-H. Liu, S.-W. Yen, C.-L. Chen, H.-C. Chang, C.-Y. Lee, Y.-S.Hsu, and S.-J. Jou, "An LDPC Decoder Chip Based on Self-Routing Network for IEEE 802.16e Applications," *IEEE J. Solid-State Circuits*, vol. 43, no. 3, pp. 684-694, Mar. 2008.

[15] Mansour, M.M., Shanbhag, N.R., "A 640-Mb/s 2048-Bit Programmable LDPC Decoder Chip," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, VOL. 41, NO.3, MARCH 2006.