社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2008－SLDM－137（7）
2008 / 11 / 17

高効率列処理演算器によるマルチレート対応 高スループットイレギュラーLDPC復号器の実装と評価

長島 諒侑† 今井 優太† 戸川 望† 柳澤 政生† 大附 辰夫†

† 早稲田大学大学院基幹理工学研究科情報理工学専攻
〒 169-8555 東京都新宿区大久保 3-4-1
E-mail: †nagashima@togawa.cs.waseda.ac.jp

あらまし 近年携帯電話や無線 LAN といった無線端末が普及し放送もデジタル化するなど，無線通信の利用が急激に加速している．通信環境の変動しやすい無線端末において，高い通信品質を保つことが大きな課題となっている．LDPC(Low Density Parity Check) 符号は高い誤り訂正能力を持つため次世代の誤り訂正符号として注目され，IEEE802.11n で規格化されている．本稿では，IEEE802.11n の規格に準拠したイレギュラーな検査行列の復号が可能な復号器を提案する．提案する復号器は列処理演算の並列性に着目することで符号化率や符号長が変化しても加算回路を共有し演算器の使用率を向上させる．並列性に行方向と列方向があり，既存研究では行方向の並列性を持たせていないことから，提案復号器では行方向，列方向ともに並列性を持たせることで高スループット化を実現する．さらに高符号化率になるにつれて列処理演算器の演算並列度を向上させることができるとともに，短い符号長でも演算器の使用率の低下を抑えることで従来手法よりも高いスループットを実現できる．提案手法により，既存研究に対して 12.5%の面積減少と 81%のスループット向上を確認した．

キーワード LDPC 復号器，イレギュラー，マルチレート，列処理演算

Multi-Rate Compatible High Throughput Irregular LDPC Decoder Based on High-Efficiency Column Operation Unit

Akiyuki NAGASHIMA†, Yuta IMAI†, Nozomu TOGAWA†, Masao YANAGISAWA†, and Tatsuo OHTSUKI†

† Dept. of Computer Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku, Tokyo 169-8555, Japan
E-mail: †nagashima@togawa.cs.waseda.ac.jp

Abstract Low Density Parity Check (LDPC) code is expected to be an error correcting code for next generation networks since it shows high error correcting performance and is incorporated in IEEE802.11n the next standard of wireless network. In this paper, we propose a multi-rate compatible irregular LDPC decoder enhancing column operation parallelism. Focusing on column-wise parallelism of column operations, uplift usage rate of operational unit and throughput by calculating all inputs simultaneously. The decoder achieves 12% savings in area and 81 % increase in throughput compared to recent architectures.

Key words LDPC decoder, Irregular, Multi-Rate, column operation

1. はじめに

LDPC 符号は，1960 年代に Gallager によって発見され，実用化が期待されている誤り訂正符号である [3]．発見された当時はハードウェア化が難しいことから注目されなかったが，高い誤り訂正能力を持ち線形時間で復号可能であるため，半導体の微細加工技術の進歩により符号化器や復号器のハードウェア化が現実的になった近年になって脚光を浴びている．

最近 5 年の間に LDPC 符号は盛んに研究され，復号性能を生かすため復号器のハードウェア化が研究対象となっている [6] [8] [10]．ハードウェア化の研究が始められた当初は設計の容易性からレギュラー LDPC 符号が研究の中心であり，レ

ギュラー LDPC 符号を対象とした復号器のスケジューリングの工夫[8] や符号長の変化に対応した復号器[6], 復号器の高速化[10] が研究されている．一方複数の符号化率に対応するLDPC 符号の有効性を示す研究もなされている[1].

2006 年，無線 LAN の規格である IEEE802.11n に LDPC 符号が採用されることが決まってからは規格に準拠する形で復号器を設計する研究もなされている [5]．LDPC 復号器は無線 LAN や携帯電話などの移動可能な無線機器に搭載されることが想定されるため，持ち運びによる位置の変化や時間などの要因による大きな通信環境の変動のもとで使用されると考えられ，符号化率を変化させて通信環境に適した復号性能を持たせる必要性がある．そこで (1)IEEE802.11n に定められたイレギュラーパリティチェック行列を用い，(2) 通信環境の変動に対して符号化率を変更することができ，(3) 高いスループットを実現できる復号器の実現が求められている．IEEE802.11n を意識したマルチレートな復号器を実現したものとしては，文献 [2], [4], [7] が知られ，符号化率の変化による演算器の引数の変化に対してシリアルに演算することで対応し (1), (2) の条件を満たしている．しかしながら文献 [2], [4], [7] はどれも各演算機ごとにシリアルな演算を用い処理時間が長くなるため，(3) に対応するために多くの演算器を必要としている．これらの演算器は符号長の変化に対応するため必要となる最大の個数が用意されており，短い符号長では多くの演算器が使用されなくなりスループットの向上に寄与しなくなる．そのため短い符号長の場合に (3) に対応できない．

そこで本稿では，(1)IEEE802.11n で提唱されている LDPC 符号のパリティチェック行列を用い，(2) 通信環境の変動に対して符号化率を変更することが可能であり，なおかつ，(3) ハードウェアを共有してパラレルな演算を実現可能にすることで，符号長が変化してもハードウェア使用率の低下を抑え，スループットを向上させる復号器を提案する．IEEE802.11n で提唱されているパリティチェック行列では異なる符号化率を実現する際に列ブロック数は変化させずに行ブロック数を変化させている．行ブロック数の変化は，復号処理において列処理演算の引数を変化させる．列処理演算とは与えられた引数をすべて加算する演算である．演算内容が加算であることから，引数の最小公倍数だけ入力を用意すれば引数の個数が少なくなる場合にも演算器を共有することができる．引数が少ない時は入力数と同じになるように並列度を上げて列処理を行うことで用意した演算器をすべて同時に用いることができ，ハードウェアの使用率の向上が可能となる．これにより (2) の符号化率の変化への対応と (3) の高スループット化を同時に実現することができる．

本稿は以下のように構成される．2 章では LDPC 符号の基本的な特徴や復号法について説明し，本稿で用いる IEEE802.11n で提唱されている LDPC 符号を紹介する．3 章では提案する復号器の構成を説明し，その中でも特に重要な列処理演算器の構成を詳しく説明する．4 章では提案復号器を VHDL を用いて記述し論理合成した結果を記載し，面積と遅延を評価する．5 章ではシミュレーションにより提案復号器のスループットを評価する．最後に 6 章で本稿をまとめる．

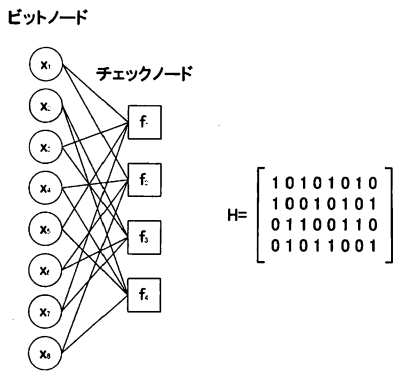

図 1　パリティチェック行列とタナーグラフの関係.

2. LDPC 符号

LDPC 符号は線形符号の一種である．本章では LDPC 符号の特性および復号アルゴリズムについて文献 [9] を元に紹介する．

2.1 LDPC 符号の定義

パリティチェック行列

LDPC 符号はパリティチェック行列と呼ばれる疎な 2 元行列で定義される．Low Density という名のとおり，パリティチェック行列中の要素 '1' の個数は要素 '0' の個数に比べて非常に少ない．このパリティチェック行列を H とし，符号語を C とするとパリティチェック行列と符号語の間には式 (1) が成り立つ．

$$CH^T = 0. \tag{1}$$

パリティチェック行列は式 (1) を満たせばどのような構成を取っても良い．一般にパリティチェック行列は 2 種類に分類され，それぞれレギュラー (正則) 行列，イレギュラー (非正則) 行列と呼ばれる．行列中の各行各列の要素 '1' の個数がすべて等しい行列がレギュラー行列であり，そうでない行列がイレギュラー行列である．本稿で取り扱う IEEE802.11n で提唱されている LDPC 符号はすべてイレギュラー LDPC 符号である．式 (1) のパリティチェック行列と符号との関係は，行列がレギュラーかイレギュラーにかかわらず図 1 に示したようなタナーグラフと呼ばれる 2 部グラフで表現できる．2 部グラフの一方のノードはチェックノードと呼ばれ，パリティチェック行列の各行における行処理演算を行う．他方のノードはビットノードと呼ばれ，各列における列処理演算を行う．

符 号 化 率

符号化率とは 1 符号語内の情報ビットが占める割合を表す．符号長を N，パリティチェックビット数を M とすると符号化率 R は式 (2) で表される．

$$R = (N - M)/N. \tag{2}$$

他の誤り訂正符号と同様に R の値が小さくなるほど誤り訂正能力が高くなる．

2.2 LDPC 復号アルゴリズム

LDPC 符号の復号アルゴリズムには対数領域 Sum-Product 復号法と呼ばれるものと，Min-Sum 復号法と呼ばれるものの 2

Step1 [初期化]
パリティチェック行列 H において $H_{mn}=1$ を満たすすべての組 (m,n) に対して対数事前比 $\beta_{mn}=0$ とする．また反復回数を l とし，最大繰り返し回数 l_{max} を設定する．
Step2 [行処理]
$m=1,2,...,M$ の順に $H_{mn}=1$ を満たすすべての組 (m,n) に対して式 (3) を利用して対数外部値比 α_{mn} を求める．

$$\alpha_{mn}=\left(\prod_{n'\in A(m)\backslash n}\mathrm{sign}\left(\beta_{mn'}\right)\right)\cdot\min_{n'\in A(m)\backslash n}\left|\beta_{mn'}\right| \qquad (3)$$

$$\mathrm{sign}=\begin{cases}1 & (x>0)\\ -1 & (x<0)\end{cases}$$

Step3 [列処理]
$n=1,2,...,N$ の順に $H_{mn}=1$ を満たすすべての組 (m,n) に対して式 (4) を用いて β_{mn} を求める．

$$\beta_{mn}=\sum_{m'\in B(n)\backslash m}\alpha_{m'n}+\lambda_n \qquad (4)$$

Step4 [一時推定後の計算]
$n\in\{1,2,...,N\}$ について

$$\hat{c_n}=\begin{cases}0 & \left(\mathrm{sign}\sum_{m'\in B(n)}\alpha_{m'n}+\lambda_n=1\right)\\ 1 & \left(\mathrm{sign}\sum_{m'\in B(n)}\alpha_{m'n}+\lambda_n=-1\right)\end{cases}$$

を求める．
Step5 [パリティチェック]
一時推定語が正しく復号されたかどうかをチェックする．一時推定語 $(\hat{c_1},...,\hat{c_N})$ が $(\hat{c_1},...,\hat{c_N})H^T=0$ を満たすとき，復号語として出力しアルゴリズムを終了する．そうでなければ Step6 へ進む．
Step6 [繰り返し復号回数のカウント]
$l<l_{max}$ ならば l をインクリメントして Step2 以降を繰り返す．
$l=l_{max}$ ならば 一時推定語を復号語として出力しアルゴリズムを終了する．

図 2 Min-Sum アルゴリズム [9]

種類が存在する．これら 2 種類のアルゴリズムはともに最尤復号法の近似アルゴリズムである．Min-Sum 復号法は対数領域 Sum-Product 法に含まれる対数演算を更に近似したアルゴリズムであり，ハードウェア化に向いたアルゴリズムである．本稿においても Min-Sum 復号法を採用する．図 2 にアルゴリズムを示す．アルゴリズム中で使用される変数 α は，2.1 章で紹介したタナーグラフのチェックノードで計算されてビットノードへ送られる情報であり，β はビットノードで計算されてチェックノードへ送られる情報である．$A(m)\backslash n$ はパリティチェック行列 H の m 行目で要素が '1' になっている列番号の集合から n 列目を除いてできる集合であり，$B(n)$ も同様に n 列目で要素が '1' になっている行番号の集合から m 行目を除いてできる集合である．λ は通信路化の環境により得られる値であり，パリティチェックビット数と同じ長さのベクトルである．$(\hat{c_1},...,\hat{c_N})$ は復号によって得られた一時推定語であり，LDPC 符号の条件を満たせば復号語とみなすことができる．

2.3 IEEE802.11n 提唱 LDPC 符号

次世代の無線 LAN 規格である IEEE802.11n では，誤り訂正符号のオプションとして LDPC 符号が採用されている [5]．LDPC 符号の仕様も [5] に記載され，イレギュラーパリティチェック行列が定義されている．パリティチェック行列はサブブロックに分割した形式を採っており，LDPC 復号器のハードウェア化に適する．サブブロックは $Z\times Z$ の正方行列として定義され，それぞれのサブブロックは単位行列か単位行列の要素を i だけ右シフトしたもの，あるいは 0 行列になっている．

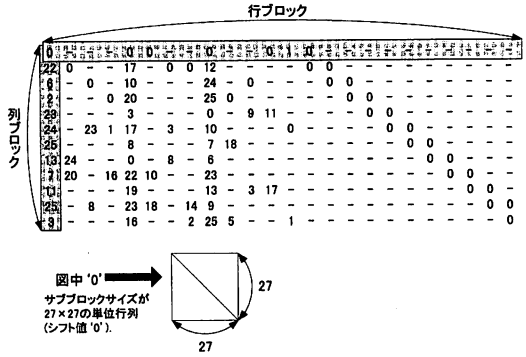

図 3 IEEE802.11n 提唱パリティチェック行列の例 ($Z=27$, $n=648$, $R=1/2$).

同規格では列ブロック数を 24 に固定し，$Z=27,54,81$ とすることで $n=648,1296,1944$ の 3 通りの符号長を採用している．それぞれの符号長に対して行ブロック数を 12, 8, 6, 4 とすることで符号化率 $R=1/2,2/3,3/4,5/6$ を採用し，合計 12 種類の符号が定義されている．図 3 に提唱されているパリティチェック行列の例を示す．図中の数字は各サブブロックの右シフト値 i を表し，"–" はサブブロックが 0 行列であることを表す．

3. 高効率列処理演算器によるマルチレート対応高スループットイレギュラー LDPC 復号器

本章では高効率列処理演算器によりマルチレートに対応し，高スループット化を実現するイレギュラー LDPC 復号器を提案する．LDPC 復号器を高スループット化するためには行処理と列処理の並列度をそれぞれ上げる必要がある．行処理，列処理において行方向の並列度と列方向の並列度がある．行方向の並列度とは検査行列 H の複数行を同時に処理することを表し，列方向の並列度とは複数列を同時に処理することを表す．本稿では列処理演算器の並列度に注目するため，特に断らない場合は列処理の並列度を単に並列度と表す．文献 [2], [4], [7] では列処理の列方向の並列度を上げる手法を取っており，複数の演算器を用意して 1 つの列ブロックの中の複数列を並列に処理する構成になっている．列方向の並列度を上げるのは単純に複数の演算器を用意し，並列に処理を実行することで可能となる．しかし列方向の並列度を上げる手法だけでは列処理における加算が逐次的に処理されるため，各ブロック列に含まれる非 0 行列の個数分だけクロック数が必要となり，高スループット化は望めない．

そこで本稿では列処理の行方向の並列度を向上させた列処理演算器を提案する．行方向の並列度を上げた提案演算器では各列の加算処理を 1 クロックで完了できる．さらにその演算器を複数用意することで列方向の並列性も持たせることができるため，高スループット化が可能である．提案する復号器は，以下のモジュールからなる．

- 入出力制御器
- 行処理演算器
- 列処理演算器

- パリティチェック演算器

まず提案復号器の主要部分である列処理演算器と，行処理演算器の最小値探索回路を提案し，その他のモジュールとメモリの構成を提案する．

3.1 列処理演算器

2 章で紹介したとおり，LDPC 符号は主に行処理演算と列処理演算が復号処理となっている．列処理演算の演算は式 (4) で表わされるとおりである．

IEEE802.11n で提唱されているパリティチェック行列は，符号長にかかわらず符号化率の変化によってパリティチェック行列の行ブロック数が変化する．符号化率 R が 1/2, 2/3, 3/4, 5/6 と変わると，それぞれ行ブロック数は 12, 8, 6, 4 となる．行ブロック数の変化によって列処理演算器の入力数が増減するため，その増減に対応した演算器が必要となる．対応する方法には逐次的な方法とトーナメント方式に演算する方法がある．逐次的な方法では 1 クロックごとに 1 つずつ入力を与え，繰り返し加算の演算を実行する．トーナメント方式では 1 クロックですべての入力を与え，2 つずつ組にした値を加算していくことで和を求める．逐次的な方法では入力数によってはクロック数が増大するためトーナメント方式を採用し，かつパイプライン化することで高スループットを実現する．

トーナメント方式を採用するにあたって最大の問題は入力数である．トーナメント方式を単純に実現すれば，行ブロック数の最大値である 12 の入力数を持たせることで行ブロック数が 8, 6, 4 の時にも対応できる．しかし最大数の入力を持たせることでそれ以外の行ブロック数のときには入力数が減ってしまい，無駄が生じるという問題がある．それを解決する手段として，列処理演算器の入力数は行ブロック数の最小公倍数である 24 とし，行方向の並列度と同時に列方向の並列度を持たせることとした．それにより符号化率 R が 1/2, 2/3, 3/4, 5/6 のとき，列処理演算器 1 つあたり列方向の並列度を 2 並列, 3 並列, 4 並列, 6 並列とすることができ，行ブロック数が異なっても演算器を共有するためその使用率が下がらない．

文献 [2], [4], [7] では列方向には逐次的な方法を用いているため，高スループットを実現するために演算器の個数を増やし，行方向の並列度を上げている．サブブロックサイズが 27, 54, 81 と変化することから，最大数の 81 に合わせた演算器数になっているため，サブブロックサイズが 27 や 54 のときは演算器の使用率が低下する．複数の列ブロックにまたがって並列処理をしないのは，メモリの構成が原因であるとみられる．提案復号器では列処理演算器を 24 個用意し，各列ブロックに 1 つずつ割り当てる．列処理における提案復号器の並列度は行方向，列方向ともに 24 となり，合計 48 並列となる．それに対して文献 [2], [4], [7] では行方向のみの並列度となり，列方向の並列度を 1 と数えることとして 28, 55, 82 並列となる．既存手法では並列度が 28 まで下がるのに対して，提案手法では常に 48 並列を実現できる．演算器使用率に影響するのは文献 [2], [4], [7] ではサブブロックサイズのみであり，提案手法ではサブブロックサイズと符号化率である．文献 [2], [4], [7] では符号化率にかかわらずサブブロックサイズが最小の場合で演算器使用率が 33%程度に下がるのに対し，提案手法ではサブブロックサイズ

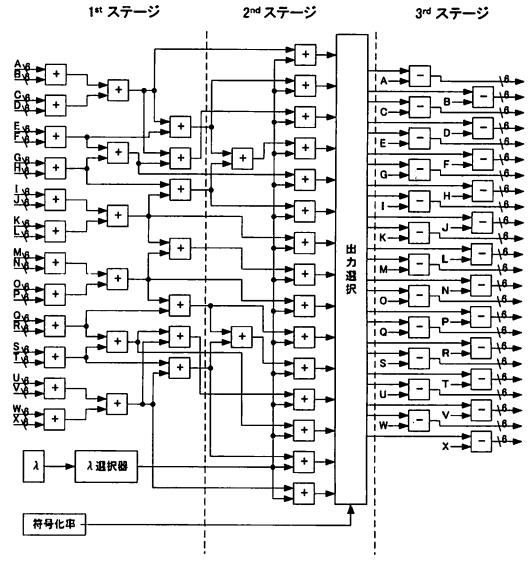

図 4 提案列処理演算器の構成．

表 1 既存手法と提案手法の演算器使用率の比較．

サブブロックサイズ	演算器使用率 [%]				
	文献 [4]	提案手法			
	$R = 1/2, 2/3, 3/4, 5/6$	$R = 1/2$	$R = 2/3$	$R = 3/4$	$R = 5/6$
27	33.3	96.4	100	96.4	90
54	66.7	100	100	96.4	100
81	100	96.4	100	96.4	96.4

や符号化率が変化しても，90%以上の使用率を維持できる．既存手法では最大の並列度は 82 と高いが，提案手法でも演算器使用率が下がることを認めれば演算器を増やして並列度を上げることは可能である．提案列処理演算器と既存手法との演算器使用率の比較を表 1 に示す．

列処理演算は式 (4) 中の $m' \in B(n)\backslash m$ が示すように，パリティチェック行列の i 行 j 列目のメッセージを求める際には加算の引数に α_{ij} は含まれない．列処理演算の結果は行ブロックごとに異なるため，列処理演算器では総和を求めた後で出力ごとにそれぞれの引数にならない値を減算している．これを式で表すと式 (5) となる．

$$\beta_{mn} = \sum_{m' \in B(n)} \alpha_{m'n} + \lambda_n - \alpha_{mn} \tag{5}$$

この手法では一度総和を求めることで一時推定語の計算も同時に実行できるというメリットがある．図 4 に列処理演算器の構成を示す．

列処理演算は符号化率とサブブロックサイズによって必要となるクロック数が変化する．例えば符号化率 R が 1/2 の場合，各列ブロックごとに 2 列を並列して処理できるため，サブブロックサイズが 27 ならば $\lceil 27/2 \rceil = 14$ となり，3 段のパイプライン化のためかかる 2 クロックを足した 16 クロックで列処理演算が終了する．

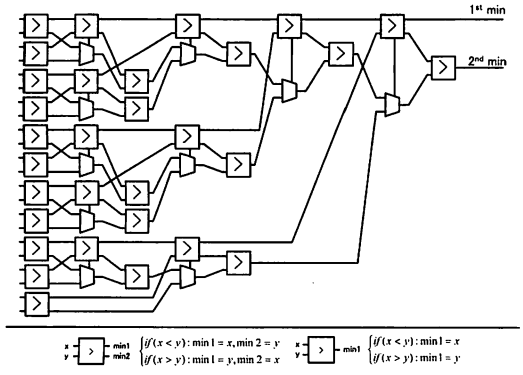

図 5　行処理演算機中の最小値探索回路.

3.2　行処理演算器

行処理演算器は入力値の中から出力先と同じノードから入力される値を除いた残りの値の絶対値の中での最小値と符号の積を掛け合わせる演算である．このことから行処理演算器は正負を表す符号の積を求めるパートと絶対値の最小値・第 2 最小値を求めるパートに分割して設計する．符号の積を求めるパートは符号の排他的論理和を取ればよい．最小値・第 2 最小値を求めるパートでは各行の行処理演算を 1 クロックで完了できるように列方向の並列度を持たせるため，列処理演算と同様にトーナメント方式を採用し，高スループットを実現する．第 2 最小値は最小値以外のすべての値よりも小さいということから，2 回以上他の値よりも大きいと判定された値は第 2 最小値にはなりえないということを用い，比較器の数を減らしたトーナメント方式を提案する．この手法では単純にトーナメントで残った値を比較するのではなく，2 度目以降の比較ではその時点で最小値の候補になっている値よりも大きいと前段で判定された値だけを，マルチプレクサによって選択する．これにより第 2 最小値以外の値は 2 回目に比較される前になくなるため，比較器の個数を抑え小面積化を実現できる．このトーナメント方式の最小値・第 2 最小値探索回路を図 5 に示す．

行処理演算器は 27 個用意し，1 つの行ブロック内で並列に演算する．3 段にパイプライン化しているため，サブブロックサイズが 27, 54, 81 のときに，それぞれ 3, 4, 5 クロックで 1 つの行ブロックの演算が終了する．

3.3　入出力制御器

入出力制御器では復号器への入力を 1 クロック当たり 27 シンボル分 162 ビット受け付け，出力も 27 シンボルずつ出力することとする．復号器への入力値を格納するメモリの書き込み信号を受け，適当なメモリへの read/write 信号とメモリエネーブラ信号を発行する．

3.4　パリティチェック演算器

パリティチェック演算器はパリティチェック行列の各行ごとに，メモリから一時推定語を読み込み，排他的論理和をとる．すべての行において排他的論理和が '0' になった場合は $CH^T = 0$ を満たすため正しく復号されたとみなし，復号が終了したことを示す信号を出力する．排他的論理和が '1' となる行があった場合，次の繰り返しに進むことを伝える信号を出力する．

表 2　提案復号器の論理合成結果.

ユニット名	面積 μm^2	遅延 ns
列処理演算器	6346	1.80
行処理演算器	4702	1.85
パリティチェック演算器	38	1.00

パリティチェック演算器は 27 個用意し，サブブロックサイズが 27, 54, 81 のときに，それぞれ 1, 2, 3 クロックで 1 つの行ブロックの演算を終了する．

3.5　メモリ構成

LDPC 符号の復号では入力値を格納するメモリと行処理・列処理の結果を格納するメモリブロックが必要となる．入力値を格納するメモリは 1 ワード 162 ビットの単一アドレスメモリとし，6 ビットに量子化したデータをサブブロックの最小サイズの 27 個格納する．それを 72 個用意し，サブブロックサイズが 27 のときは 24 個，54 のときは 48 個，81 のときは 72 個すべてのメモリを使用する．行処理・列処理の結果を格納するメモリは 162 ビットの単一アドレスメモリとし，各サブブロックに 3 個ずつ割り当てる．入力値を格納するメモリと同様にサブブロックサイズによって使用する個数が決まる．

4.　論 理 合 成

本章では 3 章で提案した行処理演算器・列処理演算器を VHDL で記述し論理合成した結果を示す．論理合成ツールは Synopsys 社の Design Compiler を用い，STARC90nm CMOS ライブラリ(注1)を用いた．表 2 に論理合成結果を示す．

3 章で説明したとおり列処理演算器は 24 個，行処理演算器は 27 個，パリティチェック演算器は 27 個用意するため，これらの 3 つの演算器の総面積は，$6346 \times 24 + 4702 \times 27 + 38 \times 27 = 280284 \mu m^2$ となる．これは文献 [4] の行処理・列処理演算器の総面積が $520000 \mu m^2$ であるのに対し，プロセスルールが異なることを考慮して補正をかけた場合で約 12.5%の面積増加となっている．列処理演算器のみ比較した場合，提案復号器は行方向，列方向の合計 48 並列で $152304 \mu m^2$ であり，文献 [4] では行方向のみ 81 並列で $70000 \mu m^2$ である．列処理演算器においては列方向の並列度を上げるために面積が約 118%増加している．一方，行処理演算器のみ比較した場合，提案復号器では列方向のみ 27 並列で $126954 \mu m^2$ であり，文献 [4] では列方向のみ 81 並列で $450000 \mu m^2$ である．行処理演算器においては 3.2 節で提案した小面積化手法も効果が見られ，演算器の個数も少ないため約 72%の面積の減少を確認した．

5.　提案復号器の性能

本章では提案復号器を C 言語上で再現したシミュレータを用いて得られた，提案復号器の誤り訂正能力と復号スループットについて説明する．

5.1　復号シミュレータ仕様

シミュレータは前章で提案した復号器の仕様に沿って復号処

(注1)：STARC90nm ライブラリは東京大学大規模集積システム設計教育研究センターを通し，株式会社半導体理工学研究センター (STARC) と株式会社先端 SoC 基盤技術開発 (ASPLA) の協力で開発されたものである．

表 3　提案復号器の仕様

項目名	具体名・数
復号アルゴリズム	Min-Sum 法
最大復号繰り返し回数 l_{max}	10 回
1 シンボル当たりの量子化ビット数	6 ビット
対応符号化率 R	1/2, 2/3, 3/4, 5/6
通信環境	AWGN 通信路
変調方式	BPSK 変調

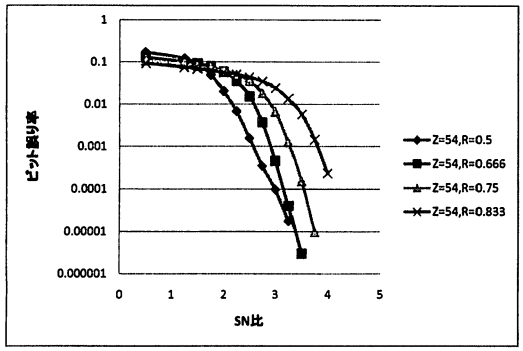

図 6　提案復号器のビット誤り率 (Z=54).

理をシミュレーションする．シミュレータの仕様を表 3 に示す．パリティチェック行列は IEEE802.11n で提唱されている 4 符号化率，3 符号長を組み合わせた 12 種類の行列を用いる．表 3 の条件で Min-Sum アルゴリズムを用いて復号し，誤りブロック数が 15000 に達するまで復号を繰り返す．シミュレーションが停止した時点でビット誤り率，平均繰り返し復号回数 l_{ave} を求める．ビット誤り率は復号器の誤り訂正能力を表し，平均復号繰り返し回数は復号スループットを求めるのに必要となるパラメータである．ビット誤り率のグラフを図 6 に示す．

5.2　復号スループット

復号スループットは式 (4) を用いて求める．

$$S_{thr} = \frac{n_{blk}}{T \times (l_{ave} \times (clk_{row} + clk_{col} + clk_{parity}) + clk_{io})} \quad (6)$$

ここで n_{blk} は符号長，T はクロック周期，clk_{row}，clk_{col}，clk_{parity} は 1 繰り返し復号回に要する行処理，列処理，パリティチェックのクロックサイクル数，clk_{io} は入出力に要するクロックサイクル数である．

論理合成結果からクロック周期は 1.85ns とし，シミュレーションで得られた平均復号繰り返し回数を用いて復号スループットを求めると，符号長 648，符号化率 5/6，SN 比 4.0dB のとき，平均有効スループットは 2.9Gbps に達した．これは文献 [4] の最高スループット 1.6Gbps に対して約 81%の向上である．同じ符号化率，SN 比のとき符号長 1296 で 2.8Gbps，符号長 1944 で 2.7Gbps と符号長が変わっても高いスループットが実現できることが確認された．符号長，符号化率を変えた時の提案手法のスループットと文献[4] の最高スループットの比較を表 4 に示す．符号化率が低いほど提案手法のスループットが下がっているが，これは符号語に含まれる情報ビットの割合が低いためであり，通信路符号化においては一般的な結果である．

表 4　文献 [4] と提案手法のスループットの比較.

サブブロックサイズ	スループット [MHz]				
	文献 [4]	提案手法			
	$R = 5/6$	$R = 1/2$	$R = 2/3$	$R = 3/4$	$R = 5/6$
27	541	932	1880	2453	2931
54	1082	856	1443	2084	2817
81	1618	812	1266	1992	2741

6. むすび

本稿では IEEE802.11n で提唱されている LDPC 符号のパリティチェック行列の符号化率による行ブロック数の変化に合わせた列処理演算器を提案し，それを用いて演算器使用率を向上させ高スループットを実現した LDPC 復号器を設計した．その結果，既存研究と比較して面積が増加したもののスループットで大きく優位性を示した．

文　献

[1] A. I. V. Casado, Wen-Yen Weng and R. D. Wesel "Multiple rate Low-Density Parity-Check codes with constant block-length," in *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004*, vol. 2, pp.2010-2014, Nov. 2004.

[2] L. Fanucci, M. Rovini, Nicola E. L'Insalate and F. Rossi, "High-throughput multi-rate decoding of structured low-density parity-check codes," *IEICE Trans. Fundamentals*, vol. E88-A no. 12, pp.3539-3547, Dec. 2005.

[3] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.

[4] K. Gunnam, G. Choi and W Wang, "Multi-Rate layered decoder architecture for block LDPC codes of the IEEE 802.11n wireless standard," in *IEEE International Symposium on Circuits and Systems, 2007. ISCAS 2007*, pp. 1645-1648, May 2007.

[5] IEEE 802.11 Wireless LANsWWiSE Proposal: High Throughput extension to the 802.11 Standard. IEEE 11-04-0886-00-000n.

[6] M. Karkooti and Joseph R. Cavallaro, "Semi-parallel reconfigurable architectures for real-time LDPC decoding," in *Proceedings of the International Conference on Information Technology: Coding and Computing, ITCC'04*, pp. 579-585, vol 1, Apr. 2004.

[7] M. Karkooti, P. Radosavljevic and J. R. Cavallaro, "Configurable, high throughput, irregular LDPC decoder architecture: Tradeoff analysis and implementation," in *Proceedings of the IEEE 17th International Conference on Application-specific Systems, Architecture and Processors (ASAP'06)*, Vol. 00, pp. 360-367, Sept. 2006.

[8] K. Shimizu, T. Ishikawa, N. Togawa, T. Ikenaga and S. Goto, "Partially-parallel LDPC decoder achieving high-efficiency message-passing schedule," *IEICE Trans. Fundamentals*, vol. E89-A, no. 4, pp. 969-978, Apr. 2006.

[9] 和田山 正, 低密度パリティ検査符号とその復号法, トリケップス, 2002.

[10] Zhongfeng Wang and Qing-wei Jia, "Low Complexity, High Speed Decoder Architecture for Quasi-Cyclic LDPC Codes," in *IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005*, vol. 6, pp.5786-5789, May 2005.