組み合わせ回路で実現したReed-Solomon符号・復号器の論理簡単化

1Q－11　森岡 澄夫　片山 泰尚
日本アイ・ビー・エム株式会社 東京基礎研究所

1．まえがき

著者らは，文献[1]で述べたように，Reed-Solomon符号（以下RS符号）の符号器（エンコーダ）および復号器（デコーダ）を組み合わせ回路として実現する方法について検討している．

エンコーダ，デコーダともガロア拡大体上の算術演算を行うが，回路の入出力数が数百本と多かったり，演算が複雑だったりするため，汎用の論理自動合成ツールでは論理が十分に簡単化されない(もとの8割程度)．

本稿では，それらの回路を対象とした簡単化の方針と，実際の簡単化結果について述べる．

2．エンコーダの簡単化方法と適用結果

エンコーダが行う演算について簡単にまとめる．ガロア拡大体 $GF(2^m)$(以下GF)上の，入力シンボル数nで，tシンボル誤り訂正可能な(k,n)RS符号($k=n+2t$)のエンコーダを，組み合わせ回路だけで構成する場合を考える．その場合，回路の入力は$n\cdot m$ビット，出力は$2t\cdot m$ビットである．回路出力$w_0,\ldots,w_{2tm-1}$は，回路入力$v_0,\ldots,v_{nm-1}$を引数とする以下の論理関数で表される．すなわち，各出力は，$n\cdot m$ビットの入力から幾つかを選び出してXOR("⊕")をとったものとなる：

$$\begin{aligned} w_0 &= c_{0,0}\cdot v_0 \oplus \ldots \oplus c_{nm-1,0}\cdot v_{nm-1} \\ &\vdots \\ w_{2tm-1} &= c_{0,2tm-1}\cdot v_0 \oplus \ldots \oplus c_{nm-1,2tm-1}\cdot v_{nm-1} \end{aligned} \qquad \text{式1}$$

ここで各v,wはBooleanの変数，各cはBooleanの定数である．

式1を回路として実装する上で問題となる点は，単体の組み合わせ回路としては大規模なものになり，論理簡単化がやりにくくなることである．実用的なRS符号のパラメータは例えばm=8，t=2～4,n=数十であり，回路入力は数百ビット，回路出力は数十ビット，2項XOR演算の総数は数千～数万個となる(図1)．

論理簡単化においては，通常，そのような規模の組み合わせ回路に対する最簡解を求めることは困難で，ヒューリスティクスを用いて近似解を求めることとなる．しかし，汎用の論理合成ツールではもとの8割程度のゲート数までしか簡単化されない．

そこで著者らは，幾つかの符号パラメータn,tについて式1を調べたうえで，次の(1)～(3)の理由から，回路の部分ごとでなく全体を対象として，出力間の共通項をくくり出す(factoring)のがよいと推測した：

(1) 式1はXORだけからなるが，AND,ORなどXOR以外の演算で表現し直すと式が爆発的に大きくなり，簡単化が難しくなるので，XORのままで簡単化すべきである，

(2) XORが特定の入力や出力に偏らず，ほぼ一様に存在しているので，回路全体に対して簡単化手法を適用すべきである(ただし，部分的にも適用できる手法であれば，なお良い)，

(3) 出力数が数十個と比較的多いので，共通項もまた多く存在し，くくり出しの効果が現れやすいと思われる．

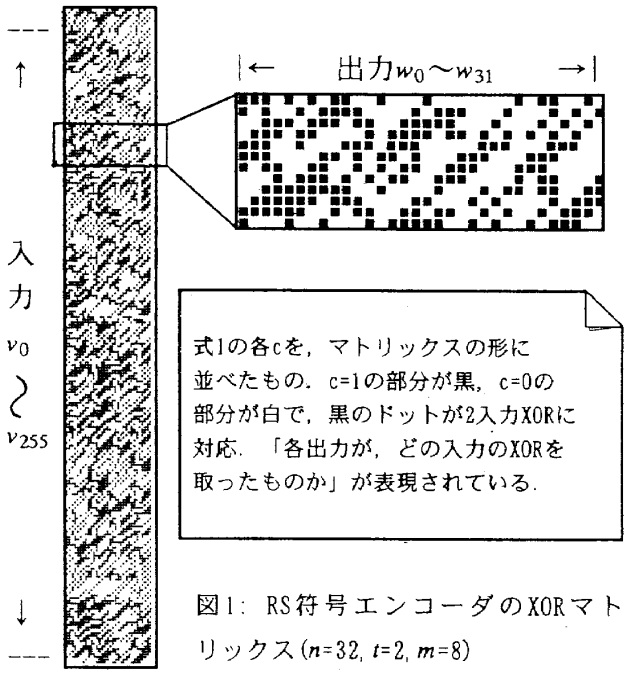


図1：RS符号エンコーダのXORマトリックス(n=32, t=2, m=8)

実際に，最近発表された文献[2]のアルゴリズムを我々独自に実装し，共通項くくり出しを行ったところ，著者らの期待以上にゲートが削減され，もとのゲート数の1/3強になることが分かった．表1に結果を示す．回路速度は，もとの8ゲート・ディレイに対し，3ゲート・ディレイほど遅くなったが，多少圧縮率を減らすことにより，ディレイの増加を抑えることも可能である．

なお，文献[2]のアルゴリズムは，もっとも多くの出力間で共有されている部分項のくくり出しを，共通項がなくなるまで繰り返すものである．文献[2]では，小規模な演算回路(8bit定数乗算)について実験を行い，ゲート数がもとの60%程度になるとしている．ところが，我々のプログラムを大規模回路に適用したところ，ずっとよく回路が簡単化された．

Logic optimization of the Reed-Solomon encoder and decoder represented in combinatorial circuits,
Sumio Morioka and Yasunao Katayama,
IBM Research, Tokyo Research Laboratory, IBM Japan Ltd.,
1623-14, Shimotsuruma, Yamato-shi,
Kanagawa-ken 242-8502, Japan

		簡単化前ゲート数	簡単化後ゲート数
エンコーダ合計		4,084	1,554
デコーダ合計	(注:256x8bit ROMは除く)	8,800	4,258
	デコーダ・シンドローム計算部	3,310	1,379
	デコーダ・係数計算部	1,762	1,192
	デコーダ・誤り計算部	3,728	1,687

表1:RS符号エンコーダおよびデコーダの簡単化結果(n=32, t=2, m=8)

原始多項式は,
$x^8 = x^4 + x^3 + x^2 + 1$
を使用.

図2: t増加時における簡単化の効果(n=32)

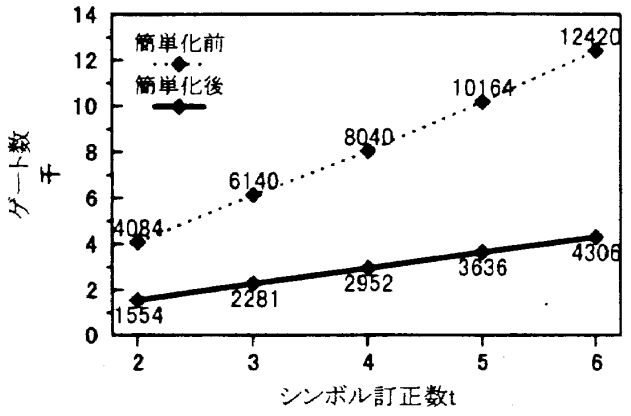
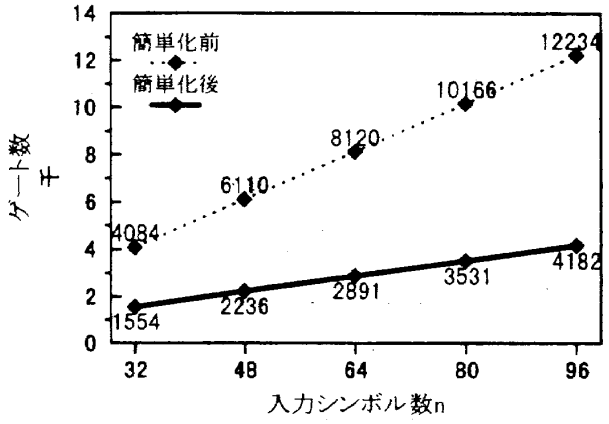


図3: n増加時における簡単化の効果(t=2)



異なる符号パラメータn,tについても，論理が大きく簡単化されるかどうかを調べた結果を，**図2と図3**に示す．すべての例について，先と同様，ゲート数がもとの1/3程度になることが確認された．したがって，今回用いた簡単化手法は，エンコーダを組み合わせ回路で構成する場合には，常に有効な方法と思われる．

3．デコーダの簡単化方法と適用結果

文献[1]で述べたように，デコーダはシンドローム計算部，係数計算部，および，誤り計算部から構成される．

そのうち，シンドローム計算部と誤り計算部は，式2と同じくXORだけからなり，かつ，同程度の回路サイズである．上述の共通項くくり出しを適用したところ，それぞれゲート数がもとの1/2から1/3となった(**表1**)．これらについても符号パラメータを変えて簡単化の度合いを調べたが，1/2～1/3程度であった．

係数計算部ではGF上の2変数間の乗算や除算を主に行うが，乗算器や除算器には，文献[3]等のReed-Muller論理式の簡単化法が有効と思われる．ただし今回は，乗算器は演算の定義通りに，除算器は逆数演算器(256x8bitのROM)と乗算を用いて構成し，係数計算部全体をまとめて汎用論理合成ツールにかけたところ，十分に小さな回路が得られた．

文献[1]で提案した回路構成方式の特徴の一つに，GF上の線形演算(Boolean上ではXOR演算)を回路の両端に集めている点があるが，以上の論理簡単化の結果からも，その構成が良いと思われる．線形演算をなるべくまとめて簡単化した方が，全体としてのゲート削減量が多くなるからである．

4．あとがき

本稿では，RS符号のエンコーダとデコーダを組み合わせ回路のみで構成する場合を対象とし，論理簡単化の方針と簡単化結果について述べた．GF上の大規模な定数乗算・加算については，出力間の共通部分項のくくり出しがかなり有効であり，多くの場合，ゲート数がもとの1/2から1/3程度に減少することが分かった．

これにより，(36,32)RS符号について，エンコーダを約1600ゲート，デコーダを約4300ゲート＋256x8bit ROMと，実用上問題ないサイズに収めることができた．エンコーダとデコーダの両方を一つのFPGA上に実装することも可能である．

注：本稿では，2入力XORゲートを1ゲートとし，ファンアウト調整のドライバは無視してゲート数をカウントした．論理再合成で2入力XORを多入力XORにしたりドライバを入れたりするとゲート数は変わるが，それでも簡単化により回路サイズは半分程度で済む．

参考文献

[1] 片山，森岡："組み合わせ回路による高速Reed-Solomon符号化復号化方式"，第57回情処全大1Q-01(1998-09).
[2] C. Paar: "Optimized Arithmetic for Reed Solomon Encoders", proc. 1997 IEEE International Symposium on Information Theory (ISIT), pp. 250(1997-06).
[3] Y. Ye and K. Roy: "A Graph-Based Synthesis Algorithm for AND/XOR Networks", proc. 34th ACM/IEEE Design Automation Conference (DAC), pp. 107-112(1997).