組み合わせ回路による高速Reed－Solomon符号化復号化方式

1Q－1

片山泰尚 森岡澄夫
日本アイ・ビー・エム株式会社 東京基礎研究所

1 はじめに

一般にＲｅｅｄ－Ｓｏｌｏｍｏｎ（ＲＳ）符号の符号化復号化回路は、符号の長さｎでの訂正可能な誤りのシンボル数ｔが２以上の場合（通常ｎ＞＞ｔである）、順序回路を用いて実現される［１］。これは、比較的データ転送速度の要求が緩やかな外部記憶装置や通信向けの用途で適用可能だが、このような訂正能力の高い符号をキャッシュメモリや主記憶向けにしようとすると性能が追いつかない。更に、順序回路を使った場合、性能だけでなく、クロックに伴う余計なスイッチングにより、電力を消費する。一方、組合せ回路のみを使えば、上記の問題は解決されるが、従来からある順序機械向けの復号化アルゴリズムではそのまま組合せ回路に展開すると回路規模が爆発的に増大する。

本発表では、ＲＳ符号化復号化回路が、格段に高速、さらに低消費電力でありながら、現実的な規模の組合せ回路で実現できる方式をアルゴリズムならびにアーキテクチャの観点から述べる。

2 符号化

符号化は、ＧＦ（2^m）上での多項式演算として定式化される通常のアルゴリズムを行列の形に書き下すことにより、線形演算のみで構成され、組み合わせ回路に展開できる。回路規模（この場合ＸＯＲゲートの数）は、単純には、入出力の数によって決定される行列のサイズ（$2m^2nt$）と、その行列の要素（成分のどれだけが０でないか）で決定される。

組合せ回路による符号化回路は、性能差はクロックマージンを考えるとｎ倍以上だが、回路規模の差はさほどは開かない。なぜなら、順序回路による回路では、２ｍｔ個のラッチ、クロック生成回路、順序制御回路などにより、かなりの回路を消費する一方、組合せ回路による場合は、論理レベルでのかなりの簡単化が期待できるからである［２］。

3 既存アルゴリズムを使った組み合わせ復号化回路の問題点

符号化回路は線形なので、既存のアルゴリズムベースで組合せ回路に展開できるが、ＲＳ符号の復号化回路の組合せ回路への展開では、既存のアルゴリズムをそのまま適用することは難しい。これは、最初に受信語からDFT(Discrete Fourier Transformation) により Frequency domainでの２ｔ個の連続したスペクトル（シンドローム）$S_0,S_1,\ldots,S_{2t-1}$ の計算までは符号化回路と同じく線形演算のみで構成されるが、ここ以降の誤りの情報を推定する部分は、非線型演算が必要になり、組合せ回路に展開した時に回路規模が爆発的に増大するためである。

例えば、連立方程式を解く事により除算を使って誤りを求めるAlgebraic decodingをそのまま適用した組み合わせ回路は、

－最後に誤り位置多項式$\Lambda(x)$と誤り評価多項式$\Omega(x)$のIDFTから誤り値を計算するForney Algorithmで、ｎ個の除算器を用意する必要があり、回路規模が大きくなる。

－除算器の全てが有効に使われるのではなく，ｔ個分の除算器しか実質的に働いていない。

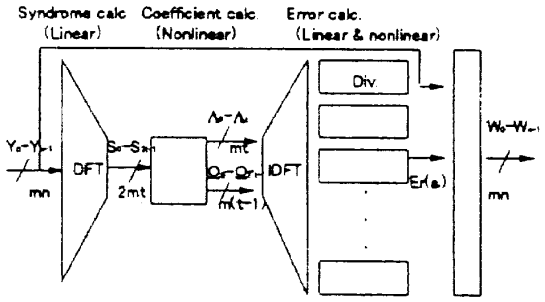

図1 Algebraic decodingを使った組合せ回路

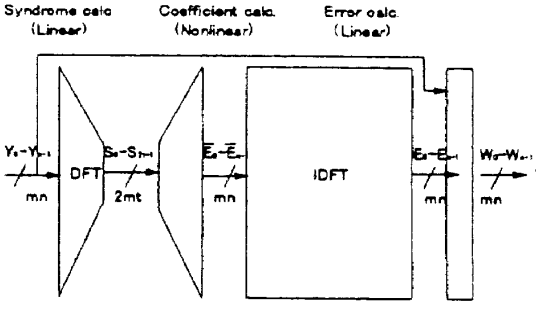

図2 Transform decodingを使った組合せ回路

Combinatorial circuit representation of the Reed-Solomon encoder and decoder
IBM Research, Tokyo Research Laboratory,
1623-14 Shimotsuruma, Yamato, Kanagawa 242-8502 Japan.

－除算器の共有化は、必要な除算機の数をｔ個に減らすが、誤まりの位置に応じて適切な除算器を割り振る回路は２ｔ個のｎ入力マルチプレクサ（前後にｔ個づつ）と、割り振りを決定するt個のプライオリティエンコーダが必要になる為、復号化の速度が遅くなる。

一方、除算を使わずにsyndoromeから誤りのDFT(Inverse-DFT)成分$\overline{E}_0 - \overline{E}_{n-1}$を計算して、最後にIDFTにより誤りを求めるTrasform decodingでは、以下の点で問題がある、

－$\overline{E}_0 - \overline{E}_{n-1}$の計算にＯ（ｎ）個の２入力乗算回路が必要である。
－IDFT自身は線形であるが膨大な量の回路（$m^2 n^2$）が必要である。

4 新たな復号化方式の提案

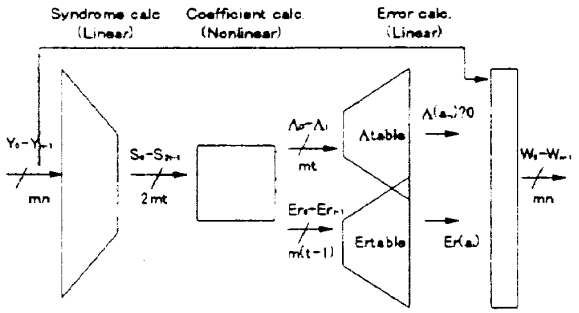

図3 我々の提案する手法

我々が提案する方式を、図１に示す。復号器の前段のシンドローム計算回路と後段の誤り計算回路に線型演算を集めて、中央の係数計算回路に非線型計算を集中させたこと、及び、シンドローム計算回路以降の非線型計算の出力も後半に線形回路(誤り計算回路）を付加してＯ（ｍｔ）にすることにより、係数計算回路の入出力数がＯ(ｍｔ)に抑えられ，回路サイズのうえでボトルネックとなる非線形演算器の個数がｎに依存しなくなり、又、後半の線形回路の大きさが小さくできる事を特徴とする。

しかしながら、このようなアーキテクチャをとるためには、最終段で除算を使わずに誤り値を計算できるアルゴリズムが必要である。この目的の為の新たな誤まり多項式Ｅｒ（ｘ）の導出をForney algorithmからの初等的な式変形でも計算可能なｔ＝２の場合に示す。誤り位置における誤り値は

$$e_i = \frac{S_0+(S_0\Lambda_1+S_1)a^{-i}}{\Lambda_1 a^{-i}} = \frac{S_0}{\Lambda_1}a^i + \frac{S_1^3+S_0^2S_2}{\Lambda_1}$$

であるので、この場合、

$$Er(x) = \frac{S_0}{\Lambda_1}x + \frac{S_1^3+S_0^2S_2}{\Lambda_1}$$

とおく事により、求めるべき多項式が得られる。Forney algorithmを用いた場合のように、多項式に値を代入してから割り算を行うのではなく、多項式の係数計算回路の部分で割り算を行うため、係数だけ求めておけば、あとは、割り算を使わずに誤り値が計算できる。

誤り計算回路では、誤り値は、Algebraic decodingでの誤り位置多項式$\Lambda(x)$と今回新たに求めた$Er(x)$を使って計算する。すなわち、

$\Lambda(a^{-i})$, ｉ＝０．．．ｎ－１

$Er(a^i)$, ｉ＝０．．．ｎ－１

の二つを各々線形なΛtableとErtableを使って同時に計算し、最後に

$$e_i = Er(a^i)(\Lambda(a^{-i})?0)$$

によって誤まりの値を求める。ただし、（Ａ？Ｂ）はＡ＝Ｂが成り立つ時に１、そうでない時に０を返す。必要なテーブルの大きさは

Λtable：　ｍｔ×ｍｎ

Ertable：　ｍ(ｔ－１)×ｍｎ

である。従って、テーブルに必要な回路規模は、符号化回路もしくはシンドローム計算回路と同等である。

我々の方式は、既存のアルゴリズムによる組合せ回路に比べて、回路規模の点だけではなく、復号化の速度の点でも優れている。これは、本方式によりΛtableによる誤り位置の計算とErtableによる誤り値の計算が完全に並列化される事が大きい。

5 これからの展開

今回の手法は、ｔ＝２の場合のＲＳ符合化復号化回路だけではなく一般化が可能である。さらに今後は、今回の組み合わせ回路による展開手法を他の符号化アルゴリズムについても適用していきたい。

参考文献

[1] R. E. Blahut, "Theory and practice of error control codes," Addison-Wesley 1984.
[2] 森岡、片山，"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，"第５７回情処全大１Ｑ－１１(１９９８－０９)。