論文 / 著書情報
Article / Book Information

題目(和文)	誤り訂正符号とその高信頼化ディジタルシステムへの応用に関する研究
Title(English)	
著者(和文)	藤原英二
Author(English)	EIJI FUJIWARA
出典(和文)	学位:工学博士, 学位授与機関:東京工業大学, 報告番号:乙第1077号, 授与年月日:1981年11月30日, 学位の種別:論文博士, 審査員:当麻喜弘
Citation(English)	Degree:Doctor of Engineering, Conferring organization: Tokyo Institute of Technology, Report number:乙第1077号, Conferred date:1981/11/30, Degree Type:Thesis doctor, Examiner:
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

誤り訂正符号とその高信頼化ディジタルシステムへの応用に関する研究

昭和56年4月

藤原英二

目　　次

第1章　序　　論

1−1　本研究の目的と背景

今日，電子計算機等のディジタルシステムは，現代社会の中で種々の分野に応用され，かつ社会，経済活動の中で中枢の機能を担いつつある。特に，最近の半導体技術の進展は，計算機の経済化，小形化，高信頼化の上で重要なインパクトを与え，システムの高機能化，大規模化，高性能化へ拍車をかけている。

このような高度にコンピュータ化された社会の中で，計算機の故障が社会生活の上に及ぼす影響はますます広範かつ深刻なものとなっている。システムの信頼度を向上させる為，従来より部品信頼度の向上，システムの高信頼度構成等の面で種々の考慮が払われてきた。故障が生じてもそれを正しく検出し直ちに対処する技術，自動的に修復する技術等のフォールトトレラント（Fault－Tolerant）技術は，現在上記の背景の下でますます重要かつ必須の技術として研究が行われている。(10)～(19)

本研究はこのようなフォールトトレラント技術の中で，誤り検出・訂正符号を使用してディジタルシステム，特に計算機システム中で生じた誤りを自動的に検出・訂正する技術に関するものである。誤り検出・訂正符号の計算機システムへの適用は，磁気テープ装置，磁気ディスク装置等のファイル記憶装置を始めとしているが，近年，主記憶装置においても簡明なハミング（Hamming）符号が一般に適用されている。(20)本研究は，計算機システム中，主記憶装置や論理装置等の高速な処理が行われる装置に対して，故障発生傾向に基づく新しい機能を有する誤り検出・訂正符号を考案し，これを効果的に適用して高い信頼度を有するディジタルシステムを構築することを目的とするものである。

1－2　計算機用誤り検出・訂正符号に関する研究の歴史と展望

誤り検出・訂正符号[(1)～(5)]は，１９４８年に発表されたC.E.Shannonの通信理論に始まり，１９５０年R.W.Hamming[(31)]により誤り検出・訂正の基礎が確立された。その後，誤り検出・訂正符号の発展は，情報理論，符号理論のめざましい発展に基づいており，特にディジタル通信システムへの適用を中心に発達した。

誤り検出・訂正符号が，計算機システムの高信頼化を目的に適用されたのはファイル記憶装置であり，その後主記憶装置，演算回路等への適用が考えられるようになった。これらの装置に対する誤り検出・訂正符号の実用化を概観すると次のようになる。

ファイル記憶装置に対しては，ファイア（Ｆｉｒｅ）符号等のバースト誤り検出・訂正の機能を有する符号が主に使用され，復号化は線形フィードバックシフトレジスタ（ＬＦＳＲ）を使用することが多い。最近の高密度磁気テープ装置[(32)]，超大容量記憶装置[(33)]には，バイト誤り訂正符号（ｂ－隣接誤り訂正符号）を使用し，符号理論上で言うErasure訂正，インタリーブ[(1)(4)]等の手法を適用して，訂正能力を増大させて実用化している。

主記憶装置に対しては，IBM7030(1961年)，IBM360/85(1968年)以来，現在の大形計算機システムには，１ビット誤り訂正・２ビット誤り検出（ＳＥＣ－ＤＥＤ）の機能を有する修正ハミング（Hamming）符号が一貫して使用されている。[(15)] 最近では，ミニコンピュータ，マイクロコンピュータの主記憶にもこの符号が採用されている。[(101)(102)]

演算回路（ＡＬＵ）に対しては，主として加算回路を対象にフルサムパリティ検査，ハーフサムパリティ検査等を行う単純パリティ符号が適用されており[(6)]，一部乗算回路に対して剰余符号が誤り検出用に適用されているにすぎない。

一方，主記憶装置，演算回路を対象に，符号としての研究の現状を概観すると次のようになる。

主記憶装置に対する誤り検出・訂正符号の研究は，１９７０年M.Y.Hsiao による奇数重み列ＳＥＣ－ＤＥＤ符号[(34)]の提案に始まり，半導体記憶素子の高集積化と歩調を合わせて研究が進められてきた。研究の方向としては，複数ビットの塊りをバイトと称し，バイトの誤りを検出・訂正する符号の研究と，ＳＥＣ－ＤＥＤの機能を拡張させた２ビット誤り訂正の機能を有する符号の研究とがある。

２ビット誤り訂正符号の研究は，復号化はかなり複雑であるが少ない検査ビット数で構成できるBCH符号を用いた研究[(35)～(37)]と，対称的に検査ビット数は多いが高速で容易な復号化が達成できる多数決復号による研究[(38)～(40)]がある。また最近では，その中間的な符号の研究[(41)]も行われている。

バイトの誤りを検出・訂正する符号の研究は，次の観点に基づいて始められた。すなわち，

バイト出力を有する記憶素子，バイト出力メモリパッケージ等のバイト単位に物理的境界を有するモジュールにより構成される装置においては，バイト内の任意の誤りを検出・訂正する符号が信頼度向上の点から効果的である。このような観点から，１９７０年D.C.Bossenによる単一バイト誤り訂正の機能を有する"b－Adjacent（b－隣接）誤り訂正符号"[42]が提案された。その後，S.J.Hong，A.M.Patelにより,単一バイト誤り訂正符号として任意の検査ビット数を持ち，かつ最大の符号長を有する符号（Maximal Code）[43] の提案がなされた。従来の１ビット誤り訂正（SEC）または，１ビット誤り訂正・２ビット誤り検出（SEC－DED）の機能に加えて単一バイト誤りを検出する符号に関しては，１９７８年D.C.Bossen，C.L.Chang，C.L.Chen[45] およびS.M.Reddy[46] らによる簡明な符号構成が示された。また，単一バイト誤り訂正の機能を有し，さらに二重のバイト誤り検出の機能を有する符号としては，１９６０年にReed－Solomon符号*[47] がすでに発表されている。この符号の応用に関しては，並列復号化回路構成の発表[48]（1979年），バブルメモリの高信頼化を目的にこの符号を適用する発表[49]（１９７３年）がなされている。さらに機能を拡大させた多重バイト誤り検出・訂正の機能を有する符号に関しては，Reed－Solomon符号を使用してその誤り検出能力，復号手法についてW.C.Carter，A.B.Wadia[50] により考察が加えられている（1980年）。麻生，当麻[51] は，ブロック計画を用いた高速復号可能な新しい多重バイト誤り訂正符号を提案している（１９７９年）。

演算回路（ALU）に対する符号の研究は，剰余検査を基本とする符号[7][8]が主体的である。単一誤り訂正の機能を有するAN符号として，１９６０年D.T.Brown，W.W.Petersonによる BP符号[52]が，また単一バイト誤り訂正を有する符号として，１９７３年P.G.Neumann，T.R.N.RaoによるNR符号[53]が著名である。その他，多剰余符号[54][55] としての２剰余符号（Bi－Residue Code）[56]，あるいはgAN符号[57] による単一誤り訂正符号の研究がある。

一方，パリティを基本とする符号による研究としては，１９５８年H.L.Garner[58] による各種算術演算に対する単純パリティ符号による検査手法の研究，１９６８年O.N.Garcia，T.R.N.Rao[59] による論理演算に対するパリティを基本とする符号の適用，また，１９７９年K－Mostashiry[60] によるパリティを基本とする符号による組合せ回路とその入力誤りに対する検査手法の研究がある。１９７７年T.R.N.Rao，H.J.Reinheimerにより剰余符号とパリティ符号を組合わせた組合せ符号（Combination Code）[61] による誤り検出・訂正手法

＊ Reed－Solomon符号は任意の最小符号間距離 d_{min} を有する符号として提案されており，単一バイト誤り訂正・二重バイト誤り検出のためには $d_{min}=4$ として構成すればよい。

の提案がなされている。

以上に示した符号の研究は，いずれも装置個有の特徴，条件等を活かす観点から，各装置に用いる符号は各々個有の符号が考えられている。本研究で扱かう符号は，上記研究のうち，パリティを基本とするバイト誤り検出・訂正符号に関するもので，この符号を主記憶装置，演算回路の双方に共通に適用する点を狙いとするものである。

次に，計算機用誤り検出・訂正符号の今後の展望について私見を述べる。今後，計算機システムの高信頼化を達成する有力な手段として，次の理由から誤り検出・訂正符号の一層の利用が期待できる。

(1) 計算機システムへの高信頼化要求は今後共一層強まる。

(2) 雑音，α線[65][66]，電源変動等の外部条件から誘発される故障は常に存在する可能性がある。

(3) 誤り検出・訂正のための回路は，半導体回路技術の発展から経済性への影響の点で相対的に低下している。

(4) 本手法は，間欠故障，固定故障の双方に対して通常動作中に検出・訂正できる高信頼化手法であり，簡便で，性能へ与える影響が比較的小さい手法である。

(5) 高速処理可能で，検出・訂正能力の高い，計算機に適した符号の研究が進められている。

計算機システムの高信頼化を目的に，今後，誤り検出・訂正符号は単体で使用されるだけでなく，システム構成，実装構成，回路構成に応じて，他の高信頼化手法と複合させてより柔軟な形で多面的に使用されることが考えられる。また，誤り検出・訂正符号は，その応用としてフォールトトレラントゲートの提案[107]，符号間距離の考え方から，試験・診断用の検査パターン[103]として，また連想処理[104][105]，画像処理，パターン認識処理等の分野への応用が考えられている。

1－3　本研究の動機と主要成果

本研究を行う上での主要な動機は以下に示す点にある。

(1)　最近の半導体集積回路技術の進展から，バイトスライスALU，複数ビット出力構成の記憶素子が実用化されており，[64][100]このような複数ビットの塊り（以下バイトと称する）の入出力を有する素子においては，その内の故障は複数ビット内の塊りの誤りとなる比率が高い。このような誤りを完全に検出・訂正するには，従来使用されているSEC－DED符号では不十分である。この種の誤りを高速に検出・訂正する実用的な符号を考案する必要がある。

(2)　論理装置（CPU），主記憶装置（MEM）等の高速な処理が要求される装置に誤り検出・訂正符号を適用する場合，符号化・復号化の遅延は性能に直接影響を与える。そのため，組合せ回路による高速な並列符号化・復号化を必要とする。

CPU，MEMのデータ幅は16ビット～128ビットを考慮すれば十分であり，符号長は短くてすむ。このような短い符号長を有する符号として，符号能力を有効に活用する工夫を払う必要がある。

(3)　符号長が長く*，高い誤り検出・訂正機能を有する符号においては，符号化・復号化回路のゲート量は3,000ゲート程度にもなる。この回路をLSI化できれば，小形化・経済化・高速化を達成できるだけでなく，回路自身の高信頼化を図ることもできる。そこで，種々の機能を有する符号に対し，符号化・復号化回路がLSI化しやすい構成を求める必要がある。

(4)　演算回路（ALU）用符号は，従来，剰余検査を基本とすることから，復号化回路のゲート量は大きくなる欠点を有している。そこで，主記憶装置を対象とした高速で経済的な復号が実現できる符号がALUにも適用できれば，主記憶装置だけでなく，論理装置中のALUを含むデータパス回路に共通に使用することができるはずである。しかも，1個の誤り検出・訂正回路にて，MEMとCPUのデータパス回路の双方を誤り検出・訂正範囲とすることができ，符号としての効果を高めることができる。このような観点から，ALUに対してパリティを基本とする主記憶用符号が適用できるための条件を明らかにする必要がある。

以上の動機の下で本研究を進めた結果，得られた主要な成果は以下の通りである。

---

*　データ幅128ビットの場合

(1) 次に示す新しいバイト誤り検出・訂正符号を求めた。

1ビット誤り訂正・単一バイト誤り検出符号

1ビット誤り訂正・2ビット誤り検出・単一バイト誤り検出符号

単一バイト誤り訂正符号

単一バイト誤り訂正・二重バイト誤り検出符号

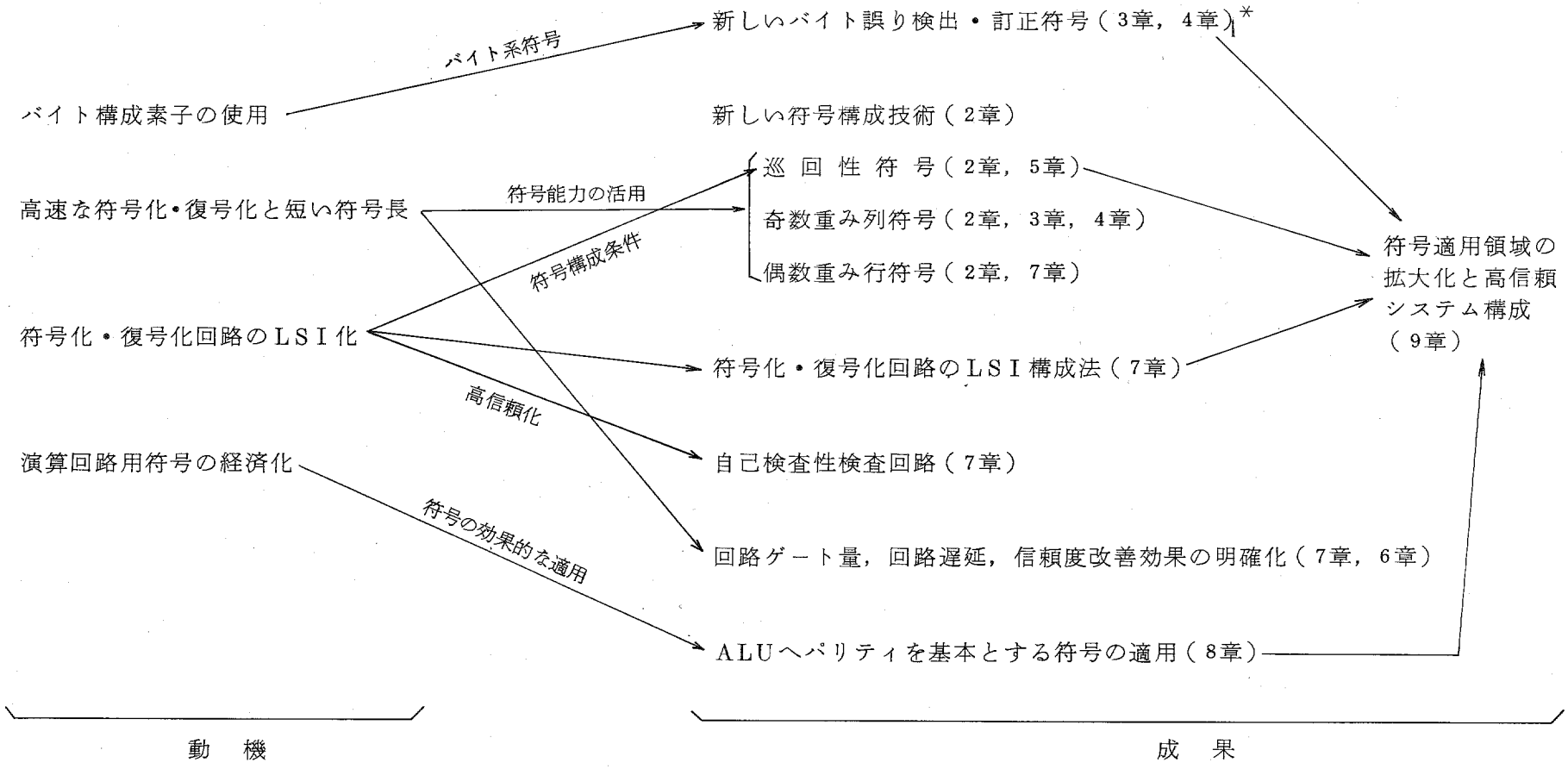
符号構成に当っては，高速な復号を有し，しかも実用的な観点に立って符号能力を出来るだけ活用する立場から，新しい形態を有する符号を求めた。例えば，符号化・復号化回路がモジュラーな構成を有し，LSI化に適する構成を与える巡回性符号，あるいは高い誤り検出能力を与える奇数重み列符号等である。また，これらのバイト誤り検出・訂正の機能を有する符号を用いて，回路ゲート量，回路遅延および信頼度改善効果についても明らかにした。

(2) 高速な符号化・復号化が可能な回路構成を示すとともに，回路のくり返し性を与える巡回性符号を適用し，具体的に汎用性を考慮したLSI化に適する回路構成を求めた。また,回路の高信頼化を目的に，自己検査性（セルフチェッキング）を有する符号化・復号化回路を求めた。

(3) パリティを基本とする経済的で高速復号可能な符号を，主記憶装置だけでなく，論理装置中のALUにも適用するための条件とその具体的手法を新しく求めた。各種の符号を適用して復号化回路の具体的なゲート量，回路遅延等を求め，その結果ALUに対し1.0～1.3倍のゲート量で，回路遅延は主記憶の場合と同等の高速な復号が実現できることを明らかにした。

(4) 巡回性の性質を有するパリティを基本とする符号をシステム全体に統一的に適用し，符号化・復号化回路だけでなく，ALU，MEMを含むシステム全体がモジュラーな構成で実現できることを明らかにした。また，ALUへの入力情報に誤りが存在する場合にも正しく検出・訂正できるための条件と具体的な構成法を求めた。

以上に示した本研究を行う上での動機と，その結果得られた成果の関係を図1－1に示す。



*（　）内は本論文で関係する章を示す。

図１－１　本研究を行う上での動機と主要な成果

1−4 本論文の概要

本論文は全体で１０章からなる。新しく求めた符号について述べた章は，第２章，第３章，第４章，第５章である。第６章は符号を適用することによる信頼度改善効果を，第７章では符号化・復号化回路についてのべる。第８章，第９章では，ＡＬＵへの主記憶用符号の適用に関してのべる。以下に各章の概要を示す。

第２章は，本研究で扱う誤り検出・訂正符号について概説する。特に本研究で扱う符号が，ＣＰＵ，ＭＥＭで使用する並列復号に適する符号でなければならないことを示す。まず，このような高速性が要求される装置に適用する符号として，符号構成上の要求条件，符号設計条件，機能面から分類した符号体系について示す。特に，ビット系符号とバイト系符号に分類し，本研究で主として扱うのはバイト系符号であることを示す。ビット系符号としては，現在研究が進められている２ビット誤り訂正符号について，高速性，経済性の観点からの研究の現状を概説する。次に，符号能力活用の観点から，符号化・復号化回路構成の容易化，符号能力拡大，応用範囲の増大をもたらす巡回性符号，奇数重み列符号，偶数重み行符号の各符号構成手法について新しく提案する。

第３章はバイト系符号のうち，バイト誤り検出符号についてのべる。ここで扱うバイト誤り検出符号とは，従来の１ビット誤り訂正，または１ビット誤り訂正・２ビット誤り検出の機能を有し，かつ単一バイト誤り検出の機能を有する新しい符号構成に関するものである。２種類の符号構成法を示し，それぞれパリティ検査行列（以下Ｈマトリクスと称す）において従来の提案の符号とは異なり，検査ビット位置が明確で任意のバイト長に対して系統的に構成できる符号構成を提案する。符号長の一般式，復号化法等について示すとともに，これらの符号を使用して単一バイト誤り訂正を与える手法についても示す。本章で示す符号は，１ビット誤り訂正－２ビット誤り検出符号と比較して検査ビット数の増加が少なく，しかも符号化・復号化回路はほぼ同一のゲート量となる特長を有する。従って，バイト誤り検出の機能が要求される装置には最適な符号であることを示す。

第４章は，バイト系符号のうちバイト誤り訂正符号について示す。バイト誤り訂正符号として単一バイト誤り訂正の機能を有する符号と，単一バイト誤り訂正・二重バイト誤り検出の機能を有する符号に関してのべる。まず，単一バイト誤り訂正符号として奇数重み列の性質を適用した新しい符号構成を示し，この符号が，Ｈマトリクスのすべての列が奇数重みとなること，２バイト間にまたがる誤りがそれぞれ同一パターンであれば必ず検出できること，バイト長ｂ＝１としたとき，従来最もすぐれていると言われる奇数重み列ＳＥＣ−ＤＥＤ符号に一致する特長を有することを示す。特に，２バイト間誤り検出能力において，バイト長ｂ＝２ビットのとき５〜１５％他の符号と比較して高いことを示す。次に，単一バイト誤り訂正・二重バイト誤

り検出符号として，従来のReed－Solomon符号が，バイト長b＝4ビット以下の範囲では情報ビット数は60ビットまでしかとれず，情報ビット数64，128ビットに対して符号が構成できない問題点を有していることを明らかにする。そこで，既存の本機能を有する符号を組合わせて新しい符号とする構成法を提案し，この符号が任意のバイト長，符号長に対して構成できることを示す。

第5章は，巡回性符号について示す。この符号はHマトリクスにくり返し性を持たせ，これによりその符号化・復号化回路がくり返し構成を有しLSI化に適する構成を与える符号であることを示す。本研究では，この性質を有する符号を巡回性符号と定義する。まず，巡回性符号の概念を簡明な1ビット誤り訂正符号を使用して説明し，具体的にその符号化・復号化回路がくり返し性を有することを示す。次に，第3章，第4章で新しく提案したバイト誤り検出・訂正符号のすべてに対して，巡回性符号としての符号構成，符号長を明らかにする。その結果，巡回性の性質を符号構成に適用することによる符号長の短縮は非常に小さく，この符号構成技術が実用的な手法であることを示す。単一バイト誤り訂正・二重バイト誤り検出符号に対しては，これを巡回性符号として理論的に導出することは困難であることから，バイト長b＝4ビットで情報ビット数64ビット，128ビットの場合を対象に最適な符号を計算機プログラムにより求めている。

第6章は，各種の符号をバイト構成を有する記憶素子からなるメモリアレーに適用して，具体的な信頼度改善を明らかにする。導出に当っては定期保守を考慮している。具体的な計算においては，情報ビット数：64ビット，使用記憶素子：64K語－4ビット構成，定期保守間隔：1週間，全メモリ容量：16Mバイト（バイト＝8ビット）の数値を使用する。符号としては，1ビット誤り訂正・2ビット誤り検出符号，およびバイト長b＝4ビットとする各種のバイト誤り検出・訂正符号を適用した結果を具体的に示す。

第7章は，符号化・復号化回路について示す。第3章，第4章で提案した各種の符号に対して回路ゲート量と回路遅延の関係を示し，その結果，ゲート量は符号機能より符号長，Hマトリクスの重みに主として依存することを明らかにする。回路遅延は組合せ回路による並列復号化手法を採用したことにより，8～14ゲート段数と高速な復号化が実現できることを示す。また，この回路の高信頼化を目的に，自己検査性検査回路の構成法を示す。これは，符号化・復号化回路だけでなく検査回路自身も含めて，その内の単一故障を完全に検出する論理を与えるものであり，その結果25～30％のゲート量増加で実現できることを明らかにする。さらに本章では，第5章で示した巡回性符号を使用して，符号化・復号化回路のLSI構成についても示す。この回路のLSI化に当っては，符号長の拡長性，多機能性，自己検査性検査回路内蔵による高信頼化を目標に，符号構成法，回路構成法等について明らかにする。

第8章は，演算回路（ALU）に対する新しい誤り検出・訂正手法について示す。まず，従来

の手法として，符号を使用しない二重化，三重化による方法と，パリティ符号，剰余符号による符号を使用する手法について概説する。その結果，剰余検査を使用する方法においては，回路ゲート量が非常に大きく，また訂正までの遅延が大きく実用的な面で問題を有していることを明らかにする。本章では，まず加算回路に対する訂正手法として，二重化とパリティ符号を組合わせた簡明な手法による方法と，水平・垂直パリティ符号を使用した方法を提案する。次に，主記憶装置等で使用しているパリティを基本とする符号を使用して，ALUの誤り検出・訂正できるための条件と具体的な構成法を提案する。これは，一般の演算結果を排他的論理和演算結果に線形に変換することを原理としている。また，ALU中に誤りはなく，ALUへの入力に誤りが存在する場合について，このパリティを基本とする符号を使用して誤り検出・訂正できるための条件について示す。

第9章は，第8章で理論的に明らかにしたパリティを基本とする符号によるALUへの適用を具体的な計算機システムに応用し，その構成，評価を示したものである。まず，符号適用に当っては，誤り検出・訂正回路を出来るだけ小さいゲート量で実現すると同時に，誤り検出・訂正範囲にある回路の故障率が大きい程，検出・訂正の効果が大きいことを示す。この観点に立って，ALUと主記憶の双方に対してパリティを基本とする同一符号を適用し，単一の誤り検出・訂正回路を設ける高信頼システム構成を提案する。この構成を簡単な16ビット，32ビット幅を有する計算機システムに適用し，各種の機能を有する符号を使用した結果，もとのALUに対し，1.0～1.3倍のゲート量で誤り検出・訂正回路が構成できることを明らかにする。さらに，巡回性符号を適用することにより誤り検出・訂正回路自身のバイトスライス化が実現でき，バイトスライスALU，バイト出力の記憶素子を使用して全体がモジュラーな構成を有する高信頼システム構成が実現できることを示す。最後に，本提案の手法と三重化（TMR）による方法との比較を示す。

第10章は，以上の各章における結論を総括してまとめたものである。

第2章　誤り検出・訂正符号

2−1 緒　　言

誤り検出・訂正符号[(1)〜(5)]は，大きく分類すると線形符号（Linear Code）と非線形符号（Nonlinear Code）となる。また，符号がブロックを単位として構成されているか，あるいは遂次的に構成されているかに従って，ブロック符号（Block Code）とたたみ込み符号（Convolutional Code）に分けられる。たたみ込み符号は主として通信用に使用されているが，ブロック符号は通信用だけでなく計算機の記憶装置にも使用されている。一方，以上の代数的符号に対して，CPU中の算術論理演算回路（ALU）で生ずる誤り検出・訂正のために，算術演算向きの符号（Code for Arithmetic Operation）[(7)]が研究されている。これは剰余を基本とする符号であり，ハミング距離とは別の算術的距離を導入して構成する符号である。

また，訂正する誤りの種別によってランダム誤り訂正符号とバースト誤り訂正符号に分けられる。また誤り訂正せず，検出することを目的とするとき，誤り検出符号と呼ばれる。

本論文で扱う符号は，上記の符号のうち線形ブロック符号に含まれるものであり，パリティを基本とする代数的符号に関するものである。

本章は，特に高速性が要求される装置にパリティを基本とする誤り検出・訂正符号を適用する場合について，符号構成上の要求条件，符号として必要な機能，また符号構成技術についてのべる。

2－2　符号構成上の要求条件

主記憶装置（MEM），論理装置（CPU）に適用する誤り検出・訂正符号として，符号構成上の要求条件を示す。

(1)　高速な復号

一般に，CPU，MEM内に誤り検出・訂正符号を採用すると，そのための符号化・復号化回路の遅延は性能に直接影響を与える。経済性を損わない範囲で，できるだけ高速な符号化・復号化を図らなければならない。従って，通信系，磁気記録系で採用している線形フィードバックシフトレジスタ（LFSR）を用いた直列復号化手法は採用できない。すなわち，CPU，MEMに対しては，組合せ回路による並列復号化手法を適用する必要がある。また，符号構成に当っては，回路遅延を小さくする考慮を払わなければならない。

(2)　経済性

誤り検出・訂正符号を採用すると，情報ビットに対して検査ビットを付加しなければならない。これは，MEMに対しては検査ビット分の記憶容量の増大，CPUに対してはデータパス回路の増大をもたらすことを意味し，装置の経済性に影響を与える。従って，できるだけ検査ビット数の少ない符号が要求される。これは，符号の"良さ"の点から言えば，一定の検査ビット数に対して符号長を大きくできる符号が良い符号であると言える。また，符号化・復号化回路は，回路自体の信頼性の観点からもできるだけ少ないゲート数で構成しなければならない。

(3)　符号能力の活用

CPUのデータ幅，またはMEMの読み書きデータ幅は16ビット～128ビットの範囲を考慮すれば十分である。これは，通信系，磁気記録系の場合と比較して非常に短い。一般に，符号としてその冗長を完全に使いきった完全符号[(4)]の形で使用しない限り，符号として潜在的に余剰能力を有している。この余剰能力を，少くとも検査ビットの増加を伴わずに符号能力拡大，復号の容易性，くり返し性を有する符号化・復号化回路構成等に活用することが重要である。

(4)　符号適用領域の拡大

符号として誤り検出・訂正できる領域を拡大させることは，符号の効果を増大させる上で重要である。例えば，MEMに適用するとともに，CPUに対しても同一符号能力で適用できれば，単一の符号化・復号化回路でCPUとMEMの双方に対して誤りを検出・訂正できることになる。これは，符号としての有効な使い方を意味するものである。

2－3 符 号 機 能

2－3－1 符号体系

主記憶装置（MEM）に使用する記憶素子は近年の半導体集積技術の著しい進展にともない，高集積化が顕著である。大形汎用計算機の主記憶には，現在，16Kビット／チップ の素子を使用した装置が実用化され商用に供されている。(62)　最近では，64Kビット／チップの素子を使用した装置が一部実用化されている。また，研究段階として，256Kビット／チップ の試作が発表されている。(63)　このような記憶素子は，データとして1ビットの入出力を有する素子が多いが，スタティック形MOS 記憶素子を中心に複数ビット入出力の素子も実用になっている。(100)　今後，さらに記憶素子の高集積化が進展するにつれ，装置のスループット増大を目的に，この種の素子が各種のシステム用主記憶装置に多用されることが予想できる。

一方，ALUにおいても，ECL（Emitler Coupled Logic）による4ビット，8ビット等をバイトとするバイトスライスALUまたはRALUがあり，また，近年4ビット，8ビット，16ビット等のマイクロプロセッサが実用化されている。(64)

1ビット入出力の素子を使用した装置においては，1ビット誤り訂正（SEC*），1ビット誤り訂正・2ビット誤り検出（SEC-DED**），2ビット誤り訂正（DEC***）等の機能を有する符号が効果的である。このような符号を，ここではビット系符号と呼ぶこととする。一方，複数ビット（bビットとする。bは2以上の整数）のデータ入出力を有する記憶素子，またはバイトスライスALUにおいては，素子内の故障はbビットの塊り（バイト）内の誤りに波及する比率が高い。このような素子により構成された装置においては，上述のビット系符号では十分誤りを検出・訂正することはできず，単一バイト誤り訂正（SbEC+），単一バイト誤り訂正・二重バイト誤り検出（SbEC－DbED#）等の機能を有する符号が効果的である。しかし，これらのバイト系符号は，一般にビット系符号と比較して検査ビット数が多く，符号化・復号化回路ゲート量も大きくなる。そこで，ビット系符号の機能を満たしつつ，バイト誤りの検出のみに止めた符号が望まれる。この種の符号は，1ビット誤り訂正・単一バイト誤り検出

---

*　Single Error Correcting

**　Single Error Correcting－Double Error Detecting

***　Double Error Correcting

\+　Single b bit－byte Error Correcting

\#　Single b bit－byte Error Correcting－Double b bit－byte Error Detecting

（SEC−SbED##），1ビット誤り訂正・2ビット誤り検出・単一バイト誤り検出（SEC−DED−SbED###）の機能を有する符号である。後者の符号は，1ビットの誤りがあれば必ず訂正し，バイト間にまたがる2ビット誤りがあれば完全に検出し，またbビットの単一バイト内に限定される任意の複数ビット誤りを完全に検出する符号である。以上のようなバイト単位の誤りの検出・訂正を行う符号を，ここではバイト系符号と呼ぶこととする。

以上から，これらの機能を有する符号を体系づけると図2－1のように示すことができる。図中，破線内が，機能，経済性，回路遅延等を考慮した場合の実用的な符号と言えよう。本研究は，このうちバイト系符号を中心に新しく提案する。

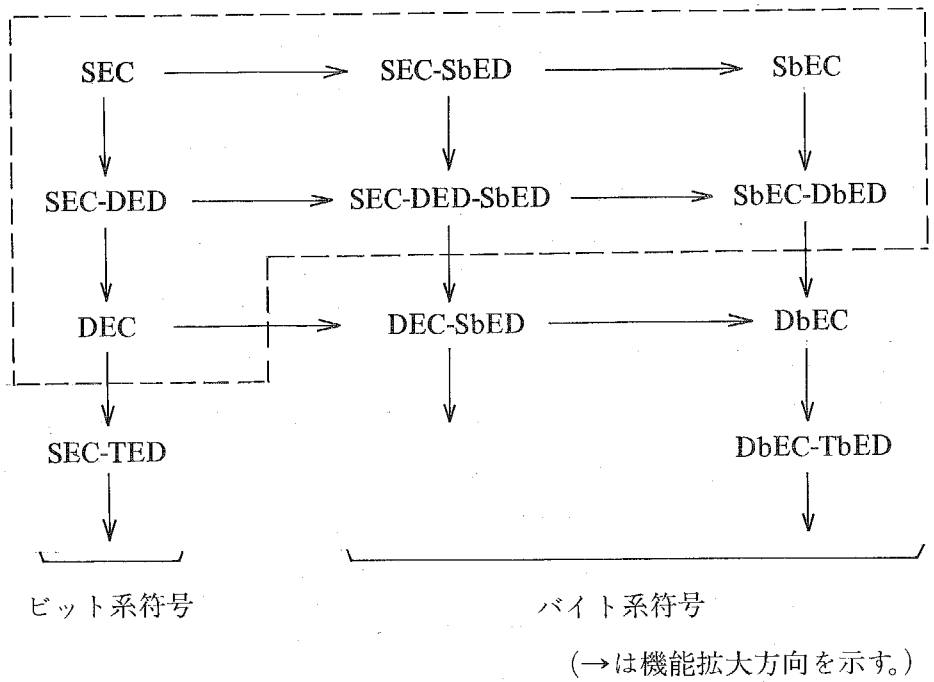


図2－1　符　号　体　系

Single Error Correcting－Single b bit－byte Error Detecting

Single Error Correcting－Double Error Detecting－Single b bit－byte Error Detecting

2－3－2　ビット系符号

ビット系符号中，現在最も実用に供されているのがSEC−DED符号である。しかし，今後の装置に対する高信頼化への要求から，より能力の高い符号が要望されている。この観点から，2ビット誤り訂正（DEC）符号等の多重ランダム誤り訂正符号の研究がある。

DEC符号の研究は，BCH符号等の検査ビット数の比較的少ない符号を用いてその符号化・復号化回路を簡単化，高速化する研究と，検査ビット数はある程度多くても簡単で高速な多数決復号可能な符号を構成しようとする研究の方向がある。

(1)　DEC−BCH符号による方法

符号長$n=2^m-1$を有するDEC−BCH符号を考える。この符号のHマトリクスは次式で表わせる。(1)〜(5)

$$H=\begin{bmatrix}H_1\\H_3\end{bmatrix}=\begin{bmatrix}\alpha^0 & \alpha^1 & \alpha^2 \cdots\cdots & \alpha^{n-1}\\ \alpha^0 & \alpha^3 & \alpha^6 \cdots\cdots & \alpha^{3(n-1)}\end{bmatrix} \qquad (2-1)$$

ここで，αはGF（2^m）上の原始元であり，α^i，α^{3i}（$i=0,1,\cdots,n-1$）はGF(2) 上のm次元列ベクトルとして表わせる。被検査情報Dに対しシンドロームは次式により求められる。

$$\begin{cases}S_1=D\cdot H_1^T\\ S_3=D\cdot H_3^T\end{cases} \qquad (2-2)$$

S_1，S_3はGF（2）上のm次元列ベクトルである。ここで，Tは行列の転置を意味する。この符号の復号は，（2−2）式で得られたシンドロームに基づいて行われる。高速な復号法としてすべての訂正可能な誤りパターンとシンドロームのパターンとを対応づけておき，シンドロームから直接誤りパターンを求める方法があるが，この方法ではゲート量は尨大なものとなる。そこで，以下に示す手法が提案されている。

〔i〕　代数的復号法 (36)

BCH符号の代数的復号法を並列化した方法であり，

$$S_1^2+S_1^{-1}\cdot S_3+\alpha^{2i}=S_1\cdot\alpha^i \qquad (2-3)$$

を満たす i が誤りの位置を与える原理に基づくものである。

〔ii〕 シンドロームトラップ法 (37)

2個の誤りの位置を i, j とすると，この組合せは ${}_nC_2$ 通りある。しかし，$S_1 = \alpha^i + \alpha^j = 1$ となるものに限定すれば，約 n／2 通りしか存在しない。そこで，$S_3 = \alpha^{3i} + \alpha^{3j}$ と誤りの位置 i, j の対応表は簡単にでき，その表を用いて誤り訂正を行うことができる。$S_1 = \alpha^p$ となる一般の場合には，誤りの位置をそれぞれ p だけずらして考えれば同じ対応表を用いて誤り訂正が可能である。このような考え方による復号法をシンドロームトラップ法と呼んでいる。

〔iii〕 遂次訂正法 (35)

1ビットの誤りは並列に訂正し，2ビット誤りはシフトレジスタを用いて遂次的（直列）に訂正する手法である。この方法は，2ビット誤りの発生確率は非常に小さく，この場合には復号化の遅延は大きくともよいとする考え方である。これにより，復号化のゲート量は小さくてすむ特長を有する。この手法では，1ビット誤りか，2ビット誤りかを簡単に区別するため，3ビット誤り検出（TED）の機能を付加したDEC－TED－BCH符号を使用する。

(2) 多数決復号による符号

検査ビット数はある程度多くなる犠牲は払っても，復号化回路の簡単化，高速化を狙う場合には，多数決復号可能な，特に1段直交可能な符号が適している。1段直交可能符号について簡単に示す。

（n, k）線形符号の符号語 $C = (C_0, C_1, \cdots C_{n-1})$ が満足すべきパリティ検査方程式の1個（i番目）を

$$\sum_{j=0}^{l-1}{}^{\oplus} C_{ij} = 0 \qquad \Sigma^{\oplus}\ ;\ \mathrm{mod.\ 2}\ 和 \tag{2－4}$$

とするとき，受信語 $D = (d_0, d_1, \cdots, d_{n-1})$ に対し，

$$S = \sum_{j=0}^{l-1}{}^{\oplus} d_{ij} \tag{2－5}$$

をパリティ検査和という。誤りパターンを $e = (e_0, e_1, \cdots e_{n-1})$ とすれば，$D = C \oplus e$ より，検査和は次のように書ける。

$$S = \sum_{j=0}^{l-1}{}^{\oplus} \; e_{i_j} \qquad (2-6)$$

R個のシンドローム $S_0, S_1, \cdots, S_{R-1}$ を考える。ある正整数 j（$0 \leq j \leq n-1$）に対し，e_j が $S_0, S_1, \cdots, S_{R-1}$ のすべてに含まれ，それ以外の e_i（$i \neq j$）は高々1個のシンドロームにしか含まれないとき，$S_0, S_1, \cdots, S_{R-1}$ は位置 j に関し直交していると言う。j に関し直交しているシンドロームが $2t$ 個あれば，誤りが t 個以下の場合，$S_0, S_1, \cdots, S_{2t-1}$ の多数決をとることにより，e_j を正しく求めることができる。各情報ビット位置に対し，このような $2t$ 個の直交する検査和が存在するとき，この符号を1段直交可能符号と呼び，その復号法を1段多数決復号法と呼ぶ。

1段直交可能符号中，最も簡単なものは自己直交符号である。これは各情報ビット位置に対する検査和をすべて1個の検査行列によって生成されるシンドロームビットの中から選ぶことのできる符号である。直交符号としては，直交ラテン方陣の接続行列に基づいて構成された直交ラテン方陣符号がある。(39)　2ビット誤り訂正の場合，この符号の符号長は m^2+4m，検査ビット数は $4m$ である。自己直交符号の他の構成法は，スタイナー系の接続行列を用いる符号である。(38)　この符号は直交の概念を e_i（$i \neq j$）が μ 個（$\mu \geq 1$）までの検査和に含まれるのを許容するようにして得られる符号である。

このような自己直交符号は検査ビット数が多い。検査ビット数を小さくするには，一般の1段直交可能符号を用いる必要がある。このような観点から，直交ラテン方陣符号の検査行列の一部を簡単化して得られる符号長 m^2+3m+1，検査ビット数 $3m+1$ の符号が提案されている。(40)

(3)　その他の符号

BCH符号と多数決復号可能な符号の中間的符号として，符号長の短いDEC−BCH 符号を組合わせて構成された符号がある。(41)　この符号の復号は，まず誤りのあるバイトを推定し，次にそのバイト内の誤りを訂正するという2段階で行う。この場合，符号長の短いDEC−BCH符号の復号を行えばよいので，DEC−BCH符号をそのまま用いる場合より復号化回路は簡単になる。

以上に述べた各種DEC符号の検査ビット数を表2−1に示す。また，情報ビット数64ビットの場合について，検査ビット数とゲート量の概略の関係を図2−2に示す。(67)

表２－１　DEC符号の検査ビット数

符　号	情　報　ビ　ッ　ト　数		
	１６ビット	３２ビット	６４ビット
BCH符号	10 ビット	12 ビット	14 ビット
中間的符号 (41)	12	14～16	15～19
１段直交可能符号(40)	13	15～19	22～26
MA符号 *(38)	16	23	31
直交ラテン方陣符号 (39)	16	24	32

＊　スタイナー系の接続行列を用いる符号

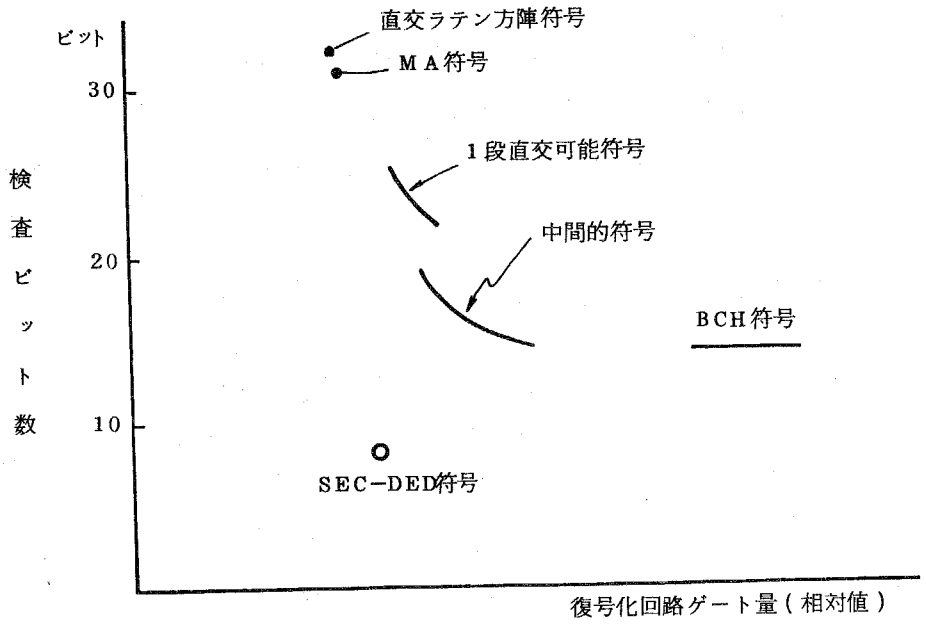


図２－２　ＤＥＣ符号の検査ビット数と復号化回路ゲート量の関係
（情報ビット数64）〔文献(67)による〕

2－3－3　バイト系符号

バイト系符号は，bビットの塊りであるバイト内のすべての誤りを検出または訂正する符号である。

今，64ビットの情報ビット数を有する装置が，4ビットのバイト出力を有する素子により構成される場合，これにSEC−DED符号を適用した場合の誤り検出能力について調べてみよう。符号として，検出能力のすぐれている奇数重み列SEC−DED符号[34]を例として適用する。この符号を，最初のビットから4ビット毎に区切り，1ビット，2ビット，3ビット，4ビット誤りの発生確率は等しいとしたとき，バイト内3ビット誤り，4ビット誤りに対する検出能力を評価すると以下のようになる。

$$\text{バイト内3ビット誤り；}\ \frac{\text{検出できる場合の数}}{\text{誤りパターンの全数}} = \frac{32}{72} = 0.44$$

$$\text{バイト内4ビット誤り；}\ \frac{\text{検出できる場合の数}}{\text{誤りパターンの数}} = \frac{10}{18} = 0.55$$

これから，3ビット誤りに対しては56%が，4ビット誤りに対しては45%が検出できず，誤訂正（ミスコレクト）することになる。これから，ビット系符号では，このようなバイト誤りの検出・訂正には不十分であり，バイト系符号が望まれる。

次に，バイト誤りとバースト誤りの相違についてのべる。いずれもbビットの長さをもつ誤りとしよう。バースト誤りは任意の位置から始まるbビットの塊りの誤りである。バイト誤りはバースト誤りの一種であるが，誤りの始まりと終りがあらかじめ決められている点が大きな相違点である。これから，バイト誤り訂正符号は，バースト誤り訂正符号に対し条件が付加されている分だけ冗長度が小さくなる，すなわち検査ビット数は少なくてすむことになる。符号理論上では，ここで言うバイト誤りは"B2型（Type B2）バースト誤り"と呼ばれ，普通のバースト誤りは"B1型（Type B1）バースト誤り"と呼ばれる。[1]〜[5]

バースト誤り訂正符号は，一般に連続する記録媒体（ディスク，ドラム，テープ等）のバースト誤りを訂正する場合に適する。一方，複数ビット出力を有する記憶素子，ALUのようにバイト単位に物理的に境界が明確な素子を使用する場合には，バイト誤り訂正符号が適することは明らかである。以上の関係を図2－3に示す。

本論文では，このようなバイト誤りを検出・訂正できる符号として，SEC−SbED，SEC−DED−SbED，SbEC，SbEC−DbEDの各機能を有する符号をとりあげ，新しい符号構成法，復号法等について明らかにする。

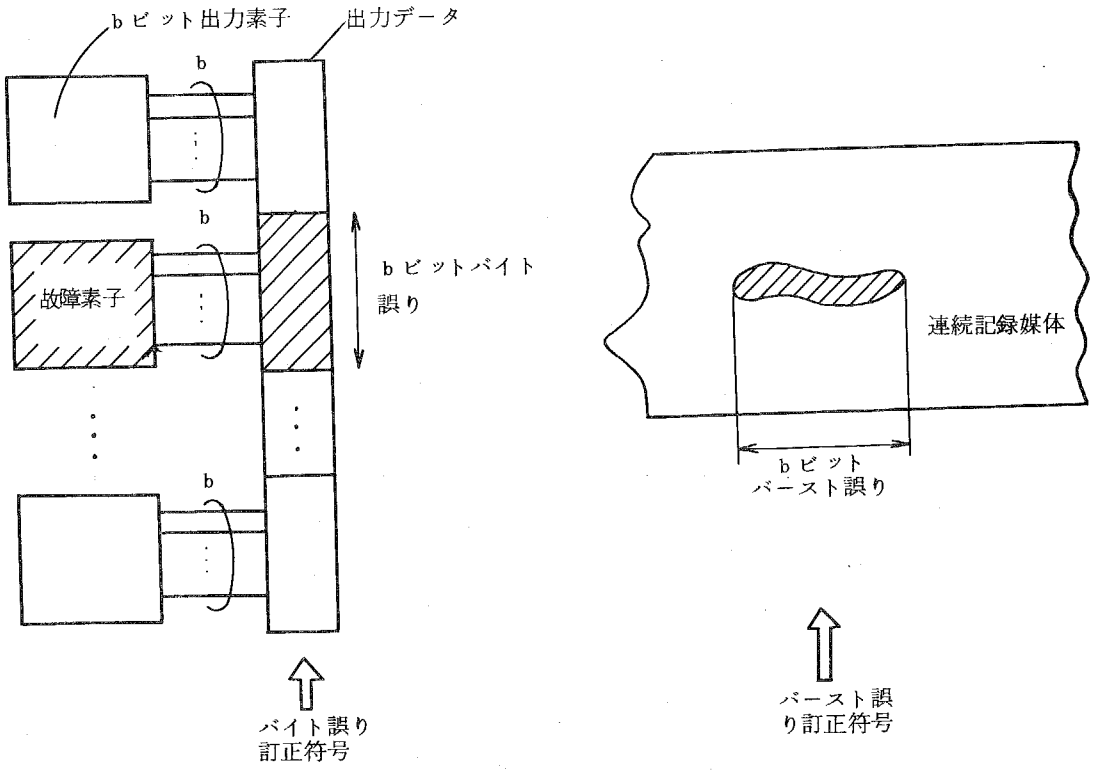


図２－３　バイト誤りとバースト誤り

2－4　符号構成技術

各種機能を有する符号に共通に適用できる新しい符号構成技術を提案する。この技術は，符号機能とは独立であり，任意の機能の符号に対して符号構成上に特徴を与え，使い易く実用的な形にするための技術である。具体的には，巡回性の性質を有する符号，奇数重み列の性質を有する符号，偶数重み行の性質を有する符号である。これらの性質は，組合わせて使用することも可能である。

符号の呼称として，これらの性質を与える場合，その名称を先頭に付して表わすこととする。例えば，巡回性（72，64）*S2EC符号，巡回性奇数重み列（72，64）S2EC符号等である。

2－4－1　巡回性符号

（定義2－1）

r行n列で構成されるHマトリクスが，次に示すd（dはrの約数）個の部分Hマトリクスで表現できるとき，巡回性符号と呼ぶ。(21)～(23)

$$\left.\begin{aligned} H &= [\, H_0 \vdots H_{r/d} \vdots \cdots \vdots H_j \vdots \cdots \vdots H_{(d-1)r/d} \,]_{r\times n} \\ H_j &= R^j \cdot H_0 \qquad\qquad j = 0,\ r/d,\ \cdots,\ (d-1)\,r/d \\ R &= \begin{pmatrix} 0\,0 & & 0\,1 \\ 1\,0 & \cdots & 0\,0 \\ 0\,1 & & 0\,0 \\ \vdots & \ddots & \vdots \\ 0\,0 & \cdots & 1\,0 \end{pmatrix}_{r\times r} \qquad \{0,\ 1\} \in GF(2^b) \end{aligned}\right\} \quad (2\text{–}7)$$

□

本符号構成法の与える特徴は以下に示す点にある。

(1)　各部分Hマトリクスは，隣接する部分マトリクスに対し，列方向にr／d 行巡回シフトした構成を有する。

(2)　本符号によって構成できる符号化・復号化回路は，各部分マトリクス単位にd個の同一回

＊　（n，k）は符号の長さを表わすもので，nビットの符号ビット長，kビットの情報ビット数を有する符号を意味する。従って，検査ビット数は（n－k）ビットである。

路により全体を構成することができる。

(3) 本符号構成法は，符号の機能とは独立であり，原理的にすべての機能の符号に対して付与できるものである。

特に，(2)に示す特徴が本符号構成手法による効果の重要な点である。これは，符号化・復号化回路の経済化，高信頼化を考慮するとき，回路のＬＳＩ化に大きな効果を発揮する。

また，符号長の観点から言えば，本符号構成法を適用しても，符号長が短くなる率は非常に小さい特徴を有する。

２－４－２　奇数重み列符号

（定義２－２）

ｒ行ｎ列のＨマトリクスにおいて，各列ベクトルに次の条件が常に成立するとき，ＧＦ(２)上の要素｛０，１｝で表わしたＨマトリクスは，すべて奇数重みの列から構成される。この符号を奇数重み列符号と呼ぶ。(22)

$$\left.\begin{array}{l} \sum_{i=0}^{r-1} h_{i,j} = 1 \\ \\ h_{ij} \quad ; \text{Hマトリクス} i \text{行} j \text{列目の要素} \in GF(2^b) \\ 1 \quad ; GF(2^b) \text{上の単位元} \\ \sum \quad ; GF(2^b) \text{上の和} \\ i = 0, 1, \cdots, r-1 \\ j = 0, 1, \cdots, n-1 \end{array}\right\} \quad (2-8)$$

□

本符号構成手法の特徴は以下に示す点にある。

(1) ＨマトリクスをＧＦ(2)上の要素で表現したとき，各列ベクトルは常に'１'の数が奇数個存在する。

(2) 本性質を有する符号は，一般に符号の誤り検出能力を増大させる可能性を有する。

本手法の重要な効果は，上記(2)にある。この手法を例えばＳＥＣ－ＤＥＤ符号に適用した場合，情報ビット数６４ビットのとき３ビット誤りを検出する割合は４３.８％* になる例が示されてい

＊　Ｈマトリクス中の'１'の数が最小の条件を付加した場合の数値。この条件をなくした場合の最大検出能力（４５.１７％）を有する符号が示されている。(69)

る。[34] 本手法を適用しないとき悪い場合で，24.12％と低い。[34] 本論文においても，この特徴を生かして，バイト誤り検出・訂正符号にこの手法を大いに適用している。

2－3－3　偶数重み行符号

（定義2－3）

r行n列のHマトリクスにおいて，各行ベクトルに次の条件が常に成立するとき，GF(2)上で表現したHマトリクスはすべて偶数重みの行から構成される。この符号を偶数重み行符号と呼ぶ。

$$\left.\begin{aligned} &\sum_{j=0}^{n-1} h_{i,j} = O \\ &h_{i,j} \ ;\text{Hマトリクス}\ i\ \text{行}\ j\ \text{列目の要素}\in GF(2^b) \\ &O \ ;GF(2^b)\text{上の零元} \\ &\Sigma \ ;GF(2^b)\text{上の和} \\ &i = 0, 1, \cdots, r-1 \\ &j = 0, 1, \cdots, n-1 \end{aligned}\right\} \quad (2-9)$$

□

本符号構成の特徴は以下に示す点にある。

(1) HマトリクスをGF(2)上の要素で表現したとき，各行ベクトルは常に'1'の数が偶数個存在する。

(2) 本符号による符号語をC_Wとするとき，ビット単位に補数をとった語$\overline{C_W}$も符号語となる。

本符号構成の主要な効果は上記(2)にあり，この性質を有する符号はTransparent Codeとも呼ばれている。[70][71] この性質を利用して，次に示す応用が考えられる。

〔1〕 マスク訂正[72]

stuck at－'0'または'1'に固定故障したメモリにおいて，本符号を採用して誤りを検出した場合，その補数をとった情報を再度書込み，読出すことにより，その固定故障による情報はマスクされて出力される。この出力情報は符号語であり，これを反転させることにより元の正しい情報とすることができる。

〔2〕　記憶素子試験時間の短縮

誤り検出・訂正符号を用いた主記憶装置においては，検査ビット部分の情報を蓄える記憶素子が必ず存在する。このような素子を含むメモリアレー全体の正常性を試験する場合，'0'の情報と'1'の情報双方に対して正しく読み書きできることを確認しなければならない。情報Dに対して付加される検査データCに対して，本符号を採用すれば$\bar{D}$に対して$\bar{C}$となる。従って，(D, C)と($\bar{D}$, $\bar{C}$)にて試験は完了する。しかし，本性質を有さない符号を用いれば，$\bar{D}$に対して必ずしも$\bar{C}$は保証されない。そのため，検査ビット部分の記憶素子に対して$\bar{C}$となる情報D'を用いて再度試験しなければならず，2倍程度の試験時間を必要とする。

〔3〕　ADR(Alternate Data Retry)[73]

入力xを回路fに印加したとき，

$$\overline{f(\mathbf{x})} = f(\bar{\mathbf{x}}) \qquad (2-10)$$

を満足する回路を自己双対回路と呼ぶ。ADRはxをこの回路に入力して誤りを検出した場合，xの補数をとった$\bar{x}$を再入力することにより，誤りをマスクして正しい出力を得る方法である。このとき，誤り検出のため偶数重み行符号を用いれば，反転した入力を入れたとき出力は符号語となり，誤りはマスクされ訂正できる。これはfを記憶素子とすれば，〔1〕と等価である。

奇数重み列の性質を有し，さらに偶数重み行の性質をも満足する符号について，以下の定理が成立する。

(定理2－1)

奇数重み列の性質と偶数重み行の性質の双方の性質を有する符号の符号長は偶数である。 □

(証明)

r行n列のHマトリクスにおいて，i列目の重みをW_iとすれば，Hマトリクスの全重みは$\sum_{i=0}^{n-1} W_i$に等しい。一方，j行目の重みをW_jとすればHマトリクスの全重みは$\sum_{j=0}^{r-1} W_j$となる。これらは互いに等しい関係にあるから，

$$\sum_{i=0}^{n-1} W_i = \sum_{j=0}^{r-1} W_j \qquad (2-11)$$

が成立する。ここで，W_i ＝奇数，W_j ＝偶数より，（2－11）式右辺は偶数に等しい。よって，（2－11）式左辺が偶数であるためには，n＝偶数でなければならない。

（証明終り）

2－5 結　　言

本論文で扱う符号は，計算機システム中のCPU，MEMに適用することを目的とするため，特に高速な符号化・復号化が要望される。このような観点に立って，CPU，MEMに適用するための符号への要求条件，符号機能等を示した。また，並列復号を能率よく行う上でHマトリクスの構成に特徴を与える巡回性符号，奇数重み列符号，偶数重み行符号の符号構成技術を示した。主要な結論は以下の通りである。

(1) バイトスライスALU，バイト出力の記憶素子等複数ビットの塊りであるバイト単位の独立な実装構成，回路構成をもつLSIまたはモジュールに対しては，この内の故障は複数ビットの塊り（バイト）の誤りとなる比率が高い。これに対しては，バイト誤り検出・訂正の機能を有する符号が効果的である。

(2) 符号能力を活用する観点から，検査ビット数を増加させない範囲で，符号化・復号化回路のくり返し性，符号能力の向上，マスク訂正等の応用を与える巡回性符号，奇数重み列符号，偶数重み行符号等の符号構成技術を適用することが重要である。

(3) 巡回性符号は，そのHマトリクスを複数個の部分マトリクスに分割し，隣接する部分マトリクスに対し行単位に巡回シフトした構成を有する。この構成を有する符号は，その符号化・復号化回路において，部分マトリクス対応に複数個の同一回路で全体を構成でき，回路のLSI化に適する。

(4) Hマトリクスの列ベクトルが奇数重みを有する奇数重み列符号は，誤り検出能力を増大させる可能性を有する。

(5) Hマトリクスの行ベクトルが常に偶数重みの性質を有する偶数重み行符号は，ビット単位に補数をとっても符号語となる特徴を有する。これから，マスク訂正，記憶素子試験の容易化等の応用を与えることができる。

第3章　バイト誤り検出符号

3−1 緒　　言

本章では，バイト誤り検出符号として，従来の1ビット誤り訂正（SEC）または，1ビット誤り訂正・2ビット誤り検出（SEC−DED）の機能を有しつつ，さらに単一バイト誤り検出（SbED）の機能を有する新しいSEC−SbED符号，SEC−DED−SbED符号を提案する。本符号はバイトの誤りは訂正できないが，次章で示すバイト誤り訂正符号と比較して検査ビット数は少ない。従来のSEC−DED符号と比較しても検査ビット数の増加は少なく，符号化・復号化回路のゲート量は同程度で実現できる特長を有する。

この種の符号は，1978年SEC−SbED符号を中心に，D.C.Bossen，L.C.Chang，C.L.Chen [45] およびS.M.Reddy [46] により簡明な符号構成が提案された。但し，Hマトリクスに行列操作を加えないと検査ビット位置が明確とならない構成を有しており，またbが奇数と偶数で構成法が異なる等の問題点を有している。本提案の符号は，Hマトリクスとして最初から検査ビット位置が明確な構成を有しており，任意のbに対し系統的に構成できる特長を有する。ここでは，2種の符号構成を示し，そのうちの一種は巡回性の符号構成に適する符号である。

以下，SEC−SbED符号，SEC−DED−SbED符号について，それぞれ2種類の符号に対してその符号構成法，符号長，復号法等についてのべる。また，これらの符号を使用して単一バイト誤り訂正を与える手法についても示す。

3－2　SEC-SbED 符号

本符号は，１ビット誤りを訂正し，かつ単一バイト内の誤りは完全に検出する符号である。ここで，次の２種類の誤りパターンの集合$\{E_1\}$，$\{E_2\}$を定義する。

$\{E_1\}$；１ビット誤りによって生ずる誤りパターンの集合

$\{E_2\}$；２ビット以上の単一バイト内の誤りによって生ずる誤りパターンの集合

このとき，$\{E_1\}$，$\{E_2\}$の間には明らかに次の関係が成立する。

$$\{E_1\} \cap \{E_2\} = \emptyset \quad (\emptyset \text{は空集合})$$

１ビット誤りを訂正し，かつ単一バイト誤りを検出する符号の条件を次の定理に示す。

（定理３－１）[46]

$\{E_1\}$に含まれるすべてのパターンの誤りを訂正し，かつ$\{E_2\}$に含まれるすべてのパターンの誤りを検出する２元線形符号（そのＨマトリクスをＨとする）の必要十分条件は次に示す条件に等しい。

（ａ）　すべての$E \in \{E_1\} \cup \{E_2\}$に対して　$H \cdot E^T \neq O^*$

（ｂ）　$\{E_1\}$に含まれる任意のE_i，E_jに対して　$H \cdot E_i^T \neq H \cdot E_j^T$

（ｃ）　$E_S \in \{E_2\}$に対して，$H \cdot E_S^T = H \cdot E_i^T$が成立するような$E_i \in \{E_1\}$は存在しない。　□

ここでは，上記定理を満足するＳＥＣ－ＳｂＥＤ符号として，２種の符号構成を提案する。

３－２－１　符号構成Ａ

定理３－１を満足する符号構成を図３－１に示す。この符号は（ｒ－ｂ）行ｂ列のA_0～$A_{2^{r-b}-1}$のマトリクスと，ｂ行ｂ列のH_f，H_eの組合せにより構成される。H_fはｂ行ｂ

＊　零列ベクトル

列の単位行列であり，すべての列は重み１を有する。H_e はｂ行ｂ列の正方行列であり，すべて零の１列と，重み２を有する（ｂ－１）列より構成される。従って，H_f は重み奇数(1)の列ベクトルから構成され，H_e は重み偶数（零または２）の列ベクトルから構成される。一方，A_i（$i=0,1,\cdots,2^{r-b}-1$）は，整数 i を２進表示した（ｒ－ｂ）個の要素からなる列ベクトルｂ列から構成される。図３－１の符号は下段にH_f またはH_e を，上段にA_i を並べ，H_f またはH_e に対して$A_0 \sim A_{2^{r-b}-1}$ を最大限ならべた構成を有している。特に下段がH_e に対しては$A_1 \sim A_{2^{r-b}-1}$ をならべ，すべて零のＨマトリクス列ベクトルを避けるためA_0 は除いている。このとき，Ｈマトリクスの列ベクトルとして重み１を有する列ベクトルｒ個が検査ビット部を構成する。これはH_f に対してA_0 をならべたｂビットの部分と，H_e 中すべて零の列に対してA_i のうち i の２進表示の重みが１の（ｒ－ｂ）ビットの部分に相当する。

具体的な符号例として，ｂ＝４，ｒ＝８とする（７２，６４）ＳＥＣ－Ｓ４ＥＤ符号を図３－２に示す。これから，本符号構成では，検査ビットと情報ビットは特別なマトリクスの変形なしにその位置が明確にわかる特徴を有する。

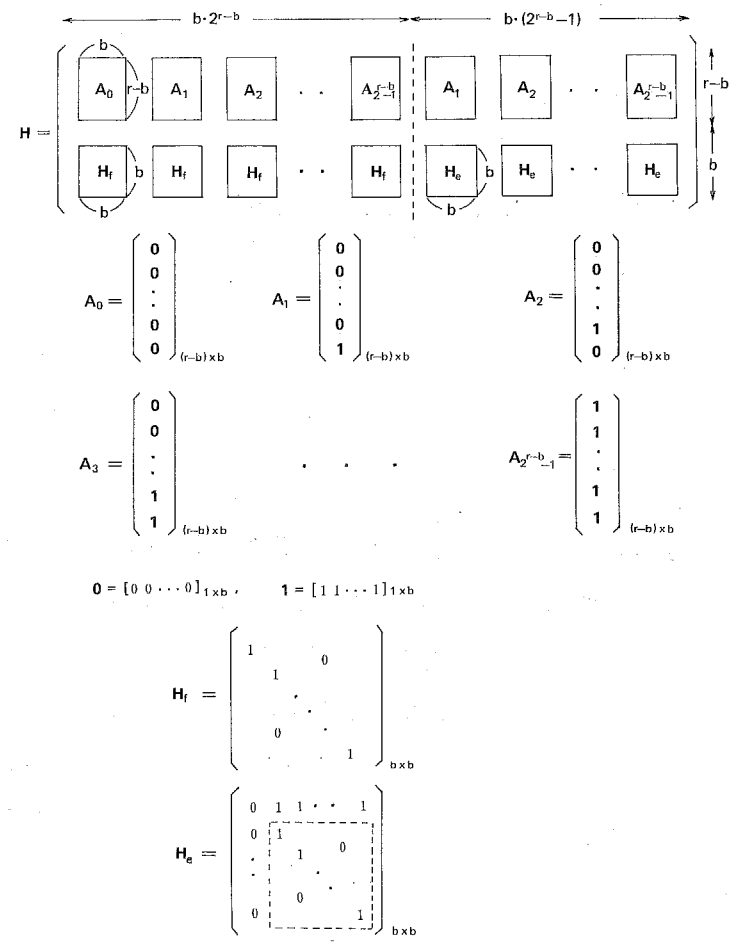
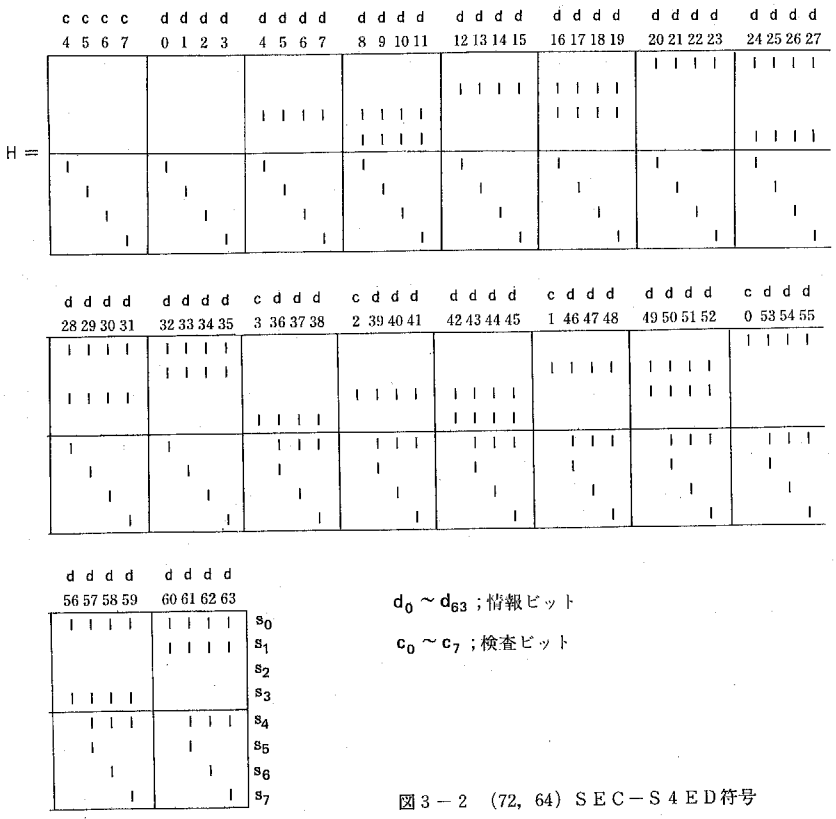


図３－１　ＳＥＣ－ＳｂＥＤ符号のＨマトリクス



$d_0 \sim d_{63}$ ；情報ビット

$c_0 \sim c_7$ ；検査ビット

図３－２ （72, 64）ＳＥＣ－Ｓ４ＥＤ符号

（定理３－２）

図３－１に示すＨマトリクスは，ＳＥＣ－ＳｂＥＤの機能を有する。また，この符号の最大符号ビット長は，次式で表わせる。

$$N = b\,(2^{r-b\pm 1} - 1) \qquad (3-1)$$

□

（証明）

定理３－１の（a）（b）に対しては，図３－１に示す符号のＨマトリクスの列ベクトルが非零であり，すべて相異なることから，１ビット誤り訂正（ＳＥＣ）の機能を有することは自明である。そこで，定理３－１の（c）について示す。すなわち，バイト誤りのシンドロームが非零で，かつ１ビット誤りのシンドロームと一致しないことを示す必要がある。１ビット誤りのシンドロームはＨマトリクスの列ベクトルに等しく，バイト誤りのシンドロームは同一バイト内の２個以上のＨマトリクス列ベクトルのmod．２加算をとることにより得られる。図３－１のＨマトリクスから得られるシンドロームを次に示す構成とする。

$$S=\begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{r-b-1} \\ \cdots\cdots\cdots \\ s_{r-b} \\ \vdots \\ s_{r-1} \end{pmatrix} = \begin{pmatrix} S_A \\ \cdots\cdots \\ S_B \end{pmatrix} \qquad (3-2)$$

S_A はHマトリクスの上段に対応して（r－b）ビットからなり，S_B はHマトリクス下段に対応してbビットから構成される。これらの重みを次のように定義する。

$$\begin{cases} W_A = W(S_A) = S_A \text{ の重み} \\ W_B = W(S_B) = S_B \text{ の重み} \end{cases} \qquad (3-3)$$

このとき，シンドロームの重みと誤りパターンの集合 $\{E_1\}$，$\{E_2\}$ に含まれるそれぞれの誤りパターン $E_i \in \{E_1\}$，$E_S \in \{E_2\}$ との関係を表3－1に示す。誤りパターン E_S の重みを表3－1においては $W(E_S)$ として示している。これは E_S の誤りビット数に等しい。この表から，バイト誤りのシンドロームは非零であることがわかる。また，1ビット誤りのシンドロームは，バイト誤りのそれと異なる。但し，図3－1に示すHマトリクス中下段に H_e を有する部分において $W(E_S)=3$ のとき，E_S に対するシンドロームの重みは2となる場合があり，これは重みの関係において E_i に対するシンドロームと等しくなる。しかし，この場合，E_i に対するシンドロームは常に $s_{r-b}=1$ であるのに対し，E_S に対しては常に $s_{r-b}=0$ となり，シンドロームは両者で異なる。以上から，バイト誤りに相当するシンドロームは非零であり，1ビット誤りのそれと異なる。よって，定理3－1の（c）についても証明できた。これから，バイト誤りは完全に検出することができ，図3－1に示す符号がSEC－SbEDの機能を有することが証明された。

図3－1から明らかなように，この符号の符号長は下段に H_f を有する部分で最大 $b \cdot 2^{r-b}$ ビット，下段に H_e を有する部分で最大 $b \cdot (2^{r-b}-1)$ ビットとなる。これから，最大符号ビット長Nは次式で表わすことができる。

$$\begin{aligned} N &= b \cdot 2^{r-b} + b(2^{r-b}-1) \\ &= b \cdot (2^{r-b+1}-1) \end{aligned}$$

（証明終り）

表３－１　１ビット誤りパターンE_iとバイト誤りパターンE_Sに対するシンドローム重みの関係

誤り ＼ 誤り位置	$E_i \in \{E_1\}$，（１ビット誤り）		$E_S \in \{E_2\}$（バイト誤り）	
	情報ビット部分における誤り	検査ビット部分における誤り		
Hマトリクス下段にH_fを有するデータ部分で誤り発生	$W_A \neq 0$ $W_B = 1$	$W_A = 0$ $W_B = 1$	$W(E_S)$＝２以上の偶数	$W_A = 0$ $W_B = W(E_S)$
			$W(E_S)$＝３以上の奇数	$W_A \neq 0$ $W_B = W(E_S)$
Hマトリクス下段にH_eを有するデータ部分で誤り発生	$W_A \neq 0$ $W_B = \begin{cases} 0 \\ 2 \end{cases}$	$W_A = 1$ $W_B = 0$	$W(E_S)$＝２以上の偶数	$W_A = 0$ $W_B = W(E_S)$
			$W(E_S)$＝３以上の奇数	$W_A \neq 0$ $W_B = \begin{cases} W(E_S)-1 \\ W(E_S)+1 \end{cases}$

（３－１）式で示す符号長は，この機能を有する符号として，現在の所の理論的な最大符号長に一致する。[45][46] 表３－２にｂ，ｒに対するＮの関係を示す。また，図３－３に情報ビット数（Ｎ－ｒ）に対する検査ビット数ｒの関係を示す。これから，ｂ＝２の場合で，従来のＳＥＣ－ＤＥＤ符号の場合と比較して検査ビット数は１ビット少なく，ｂ＝４の場合でほぼ等しい。また，ｂ＝８で３ビットの増加となる。

表３－２　SEC－SbED符号（構成Ａ）の符号長Ｎ

r＼b	1*	2	3	4	5	6	7	8
2	3	2	0	−**	−	−	−	−
3	7	6	3	0	−	−	−	−
4	15	14	9	4	0	−	−	−
5	31	30	21	12	5	0	−	−
6	63	62	45	28	15	6	0	−
7	127	126	93	60	35	18	7	0
8	255	254	189	124	75	42	21	8
9	511	510	381	252	155	90	49	24
10	1023	1022	765	508	315	186	105	56
11	2047	2046	1533	1020	635	378	217	120
12	4095	4094	3069	2044	1275	762	441	248

*　SEC符号に一致

**　－；符号長が存在しないことを示す。

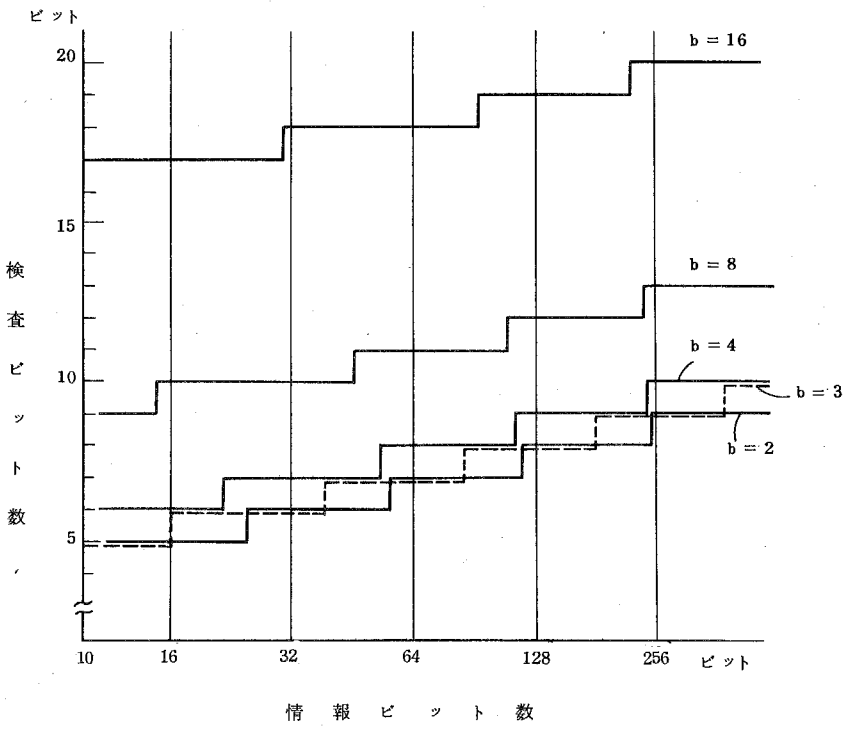


図３－３　ＳＥＣ－ＳｂＥＤ符号（構成Ａ）の情報ビット数に対する検査ビット数の関係

次に，この符号の復号法についてのべよう。シンドロームの作成と１ビット誤り訂正までの過程は，従来のＳＥＣ符号またはＳＥＣ−ＤＥＤ符号の復号化と同一である。唯一異なる点は，単一バイト誤りの検出回路が存在する点である。そこで，この回路についてその構成法を示す。バイト内２ビット以上の誤り検出は，図３−１の符号をもとにして，ｒビットのシンドロームから次のいずれかの条件に一致するとき求めることができる。

（ｉ）　シンドローム中下位ｂビットのパターンが，H_f，H_e の列ベクトルと一致しない。

（ii）　シンドローム中下位ｂビットのパターンが，H_e 中の重み２を有する（ｂ−１）個の列ベクトルの１個と一致しても，上位（ｒ−ｂ）ビットのシンドロームはすべて零である。

図３−２に示す符号例をもとに，上記（ｉ），（ii）の条件を満足する具体的な検出回路（ＳｂＥＤ回路）を図３−４に示す。この回路は１４ゲートからなるが，前段の９ゲートは本来１ビット誤り訂正のためのデコード回路中に含まれるものであることから，ＳｂＥＤ回路自体としては５ゲートである。ｂ＝８に対しても高々１０ゲート程度であり，従ってＳＥＣ−ＳｂＥＤ符号の復号化回路はＳＥＣ符号の復号化回路に対して５〜１０ゲート程度の増加で実現できる。

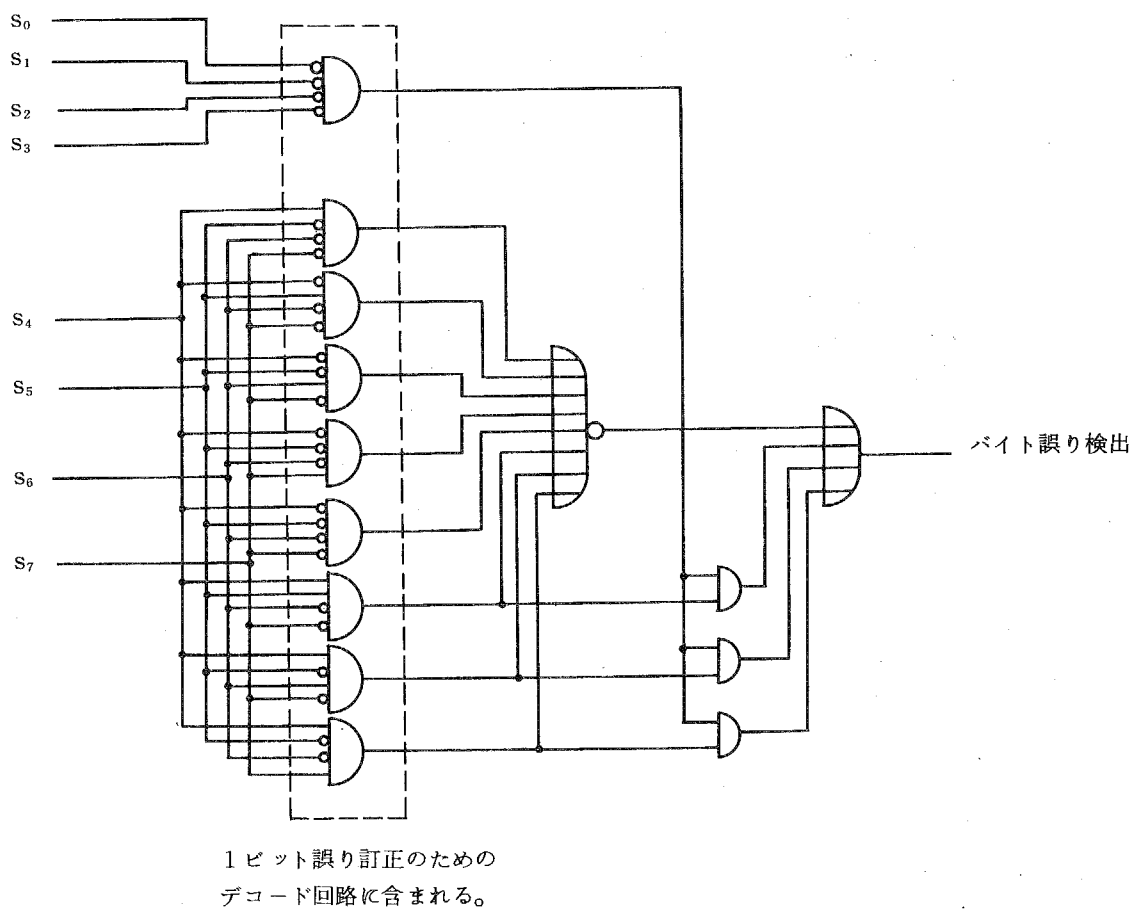


図３−４　バイト誤り検出回路（図３−２に示す符号の場合）

3－2－2　符号構成B[25]

構成Aの符号では，巡回性符号を構成することは困難である。そこで，巡回性符号として容易に構成可能な新しい考え方に基づく符号構成を示す。巡回性符号としての具体的な構成は第5章に示す。

（定義3－1）正則行列P_q

b行b列の2進正則行列をPとし，次のように表わす。

$$P=[\,\alpha_0,\alpha_1\ \cdots,\ \alpha_i,\ \cdots\alpha_{b-1}\,]_{b\times b} \qquad (3-4)$$

α_i ：b個の{0，1}の元を有する列ベクトル

この行列を，行方向にq(mod.b)ビット巡回置換して得られる正則行列をP_qとする。P_qは次式で表わせる。

$$P_q=[\,\alpha_{b-q},\ \alpha_{b-q+1},\ \cdots,\ \alpha_{b-q+i},\ \cdots\alpha_{b-q-1}\,]_{b\times b} \qquad (3-5)$$

$b-q+i$ ；mod.b　□

一般に，正則行列に関して以下の性質が成立する。

（性質）　α_i,α_jを正則行列（b行b列）の列ベクトルとする。

$i,\ j\in\{0,\ 1,\ \cdots,\ b-1\}$

1）　α_i またはα_jについて，すべて零の元からなる列ベクトルは存在しない。

2）　$i\neq j$ mod.bに対して$\alpha_i\neq\alpha_j$となる。

3）　正則行列中の任意の列ベクトルをとり出して，これらのmod.2加算を行っても，すべて零の元からなる列ベクトル結果は生じない。　□

（定義3－2）正方行列M_u

整数u（$0\leq u\leq 2^b-1$）のbビット2進表示を（$a_{b-1},a_{b-2},\cdots,a_1,a_0$）とするとき，その列ベクトルを$U=(a_{b-1},a_{b-2},\cdots,a_1,a_0)^T$とする。この列ベクトルUから，次のb次の正方行列を定義する。但し，UとしてP_qの列ベクトルα_i（$i=0,1,\cdots,b-1$）と一致するものは除外する。

$$M_u = [\,U,\ U,\ \cdots,\ U\,]_{b\times b}$$
$$U \neq \{\,\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\,\} \qquad (3-6)$$
□

以上に定義したP_q およびM_uは，それぞれｂ個，（2^b-b）個存在する。これらの行列P_qおよびM_uを用いて，新しいＳＥＣ−ＳｂＥＤ符号を構成する。Ｈマトリクスを次に示す。

$$H = \begin{Bmatrix} h_{0,0} & h_{0,1} & \cdots & h_{0,j} & \cdots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & \cdots & h_{1,j} & \cdots & h_{1,n-1} \\ \vdots & \vdots & & \vdots & & \vdots \\ h_{i,0} & h_{i,1} & \cdots & h_{i,j} & \cdots & h_{i,n-1} \\ \vdots & \vdots & & \vdots & & \vdots \\ h_{r-1,0} & h_{r-1,1} & \cdots & h_{r-1,j} & \cdots & h_{r-1,n-1} \end{Bmatrix}_{r\times n} \qquad (3-7)$$

$$h_{i,j} \in \{\,P_q,\ M_u\,\}$$
$$q = 0,\ 1,\ \cdots,\ b-1$$
$$u = 0,\ 1,\ \cdots,\ 2^b-1 \quad \left(\begin{array}{l}\text{但し，}\ u\ \text{は}\ \alpha_0,\ \alpha_1,\ \cdots,\ \alpha_{b-1}\ \text{で}\\ \text{表わされる数値に一致しない。}\end{array}\right)$$

このとき，次の定理に示すＨマトリクス構成により，ＳＥＣ−ＳｂＥＤ符号が構成できる。

（定理３－３）

Ｈマトリクスの列ベクトルをｒ個のM_u，P_qの元を使用して構成するとき，この列ベクトル中には少くとも１個のP_qを含むとする。このような条件を有する相異なる列ベクトルを並べてＨマトリクスを構成したとき，このＨマトリクスはＳＥＣ−ＳｂＥＤ符号を表現する。また，この符号の最大符号ビット長は次式で表わすことができる。

$$N = 2^{br} - (2^b - b)^r \qquad (3-8)$$
□

（証明）

正則行列P_qの性質１），２）より，ｂ個の列ベクトルにはすべて零の元から構成されるベクトルはなく，しかも各列ベクトルは相異なることから，ＳＥＣ機能を有することは明らかである。次に，ＳｂＥＤの機能について考える。バイトj（＝０，１，…，$n-1$）に誤りパター

ンE$_j$ が生じたとする。このバイト誤りのシンドロームは次式で与えられる。

$$E_j \cdot \begin{pmatrix} h_{0,j} \\ h_{1,j} \\ \vdots \\ h_{i,j} \\ \vdots \\ h_{r-1,j} \end{pmatrix} = \begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_i \\ \vdots \\ S_{r-1} \end{pmatrix} \qquad (3-9)$$

このとき，符号構成条件から$h_{0,j}$, $h_{1,j}$, ‥ $h_{i,j}$ ‥ $h_{r-1,j}$ 中には，少くとも1個P_qを含む。ここで，$h_{i,j} = P_q$としても一般性を失わない。jバイト目の誤りパターンE$_j$ に対するシンドロームの関係を表3－3に示す。表中，W(E$_j$)は誤りパターンの重み，すなわち誤りビット数を示す。これから，バイト誤りのシンドロームが1ビット誤りのシンドロームと一致することはなく，よってバイト誤りは完全に検出できる。

表3－3　jバイト目の誤りパターンE$_j$ に対するシンドロームの関係

シンドローム ＼ 誤り	W(E$_j$)＝1,（1ビット誤り）	W(E$_j$)≧2,（バイト誤り）
$h_{i,j} = P_q$ である i に対するS$_i$	$S_i \in \{\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\}$ $\neq 0^*$	$S_i \notin \{\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\}$ $\neq 0^*$
$h_{k,j} = M_u$ である k（$\neq$ i）に対するS$_k$	$S_k = u \in \{\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\}$	$S_k = 0^*$ または $= u \in \{\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\}$

$i, k \in \{0, 1, \cdots, r-1\}$,　　＊0；零列ベクトル

次に，この符号の最大符号ビット長を求める。Hマトリクスのすべての列ベクトルは少くとも1個α_iを含む。従って，すべてのα_i，Uを組合わせた$(2^b)^r$個の列ベクトルから，U∈$\{\alpha_0, \alpha_1, \cdots, \alpha_{b-1}\}$のみをr個ならべた列ベクトルを差引けば最大符号ビット長が求められる。Uは(2^b-b)個存在し，これからr個のUを並べた列ベクトルは$(2^b-b)^r$個存在する。従って，最大符号ビット長Nは次式で与えることができる。

$$N=(2^b)^r-(2^b-b)^r$$

（証明終り）

正則行列P_qとしては，種々のものが考えられる。実用的な観点から，次の条件を満足する正則行列が最適である。

1） 行列中の'1'の数が少ない。

2） 行列中の各列ベクトルの重みが一定である。

1）の条件は，その符号化・復号化回路を簡単にする点で重要である。また2）の各列ベクトル重み一定条件は，後のSEC−DED−SbED符号を構成するとき，奇数重み列条件を満足させる場合に便利である。

（補題）

上記条件1），2）を満足する唯一の正則行列はb行b列の単位行列Iである。

□

証明は容易であるので省略する。

Iをqビット行方向に巡回置換した正則行列をI_qとする。

$$I_q=_{b-q)}\begin{bmatrix} & & \overbrace{1}^{q} & \\ & & & \ddots & \\ & & & & 1 \\ 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \end{bmatrix}_{b\times b} \qquad q:\mathrm{mod.b} \qquad (3-10)$$

ここで，b＝3の場合のI_q，M_uの例を示す。

$$I_0=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\quad I_1=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}\quad I_2=\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix}$$

$$M_0=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}\quad M_3=\begin{bmatrix}0&0&0\\1&1&1\\1&1&1\end{bmatrix}\quad M_5=\begin{bmatrix}1&1&1\\0&0&0\\1&1&1\end{bmatrix}\qquad(3-11)$$

$$M_6=\begin{bmatrix}1&1&1\\1&1&1\\0&0&0\end{bmatrix}\quad M_7=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

これらを用いて，r＝2とする（39，33）SEC−S3ED符号の例を図3－5に示す。同様に，b＝2，r＝3の場合の例として（56，50）SEC−S2ED符号を図3－6に示す。b＝2に対するI_q，M_uは次のようになる。

$$I_0=\begin{bmatrix}1&0\\0&1\end{bmatrix}\quad I_1=\begin{bmatrix}0&1\\1&0\end{bmatrix}\quad M_3=\begin{bmatrix}1&1\\1&1\end{bmatrix}\quad M_0=\begin{bmatrix}0&0\\0&0\end{bmatrix}$$

（3−12）

この符号のb，rに対するNの関係を表3－4に示す。また，情報ビット数（N−br）に対する検査ビット数brの関係を図3－7に示す。

$$H = \left[\begin{array}{ccccccccccc|cc} I_0 & I_0 & I_0 & I_0 & M_6 & M_5 & M_3 & M_7 & I_0 & I_0 & I_0 & I_0 & M_0 \\ M_6 & M_5 & M_3 & M_7 & I_0 & I_0 & I_0 & I_0 & I_0 & I_1 & I_2 & M_0 & I_0 \end{array}\right]$$

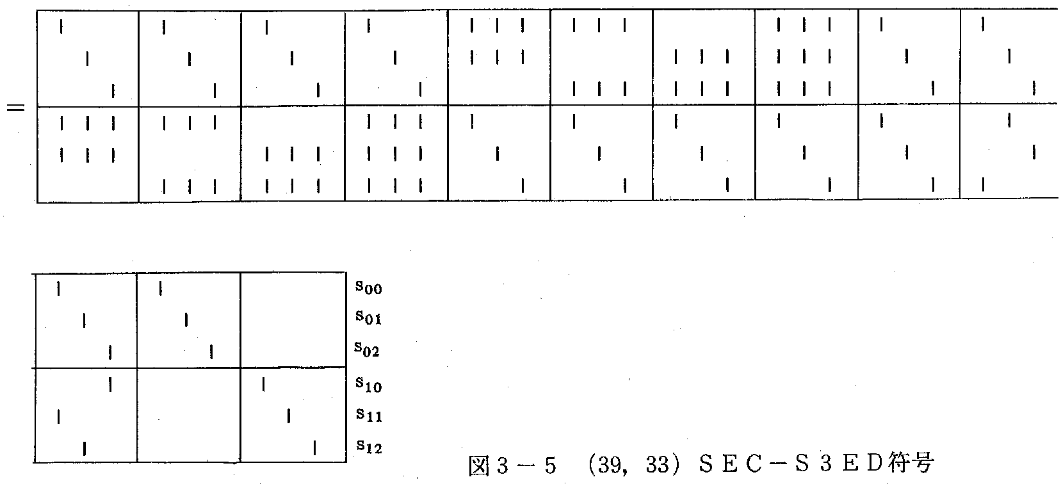


図３－５　(39, 33) ＳＥＣ－Ｓ３ＥＤ符号

$$H = \left[\begin{array}{ccccccccccccccccccccccccc|ccc} I_0 & M_0 & M_3 & I_0 & M_3 & M_0 & I_0 & M_3 & I_0 & I_0 & M_3 & I_1 & I_0 & M_3 & M_3 & I_0 & M_0 & I_0 & I_0 & M_0 & I_1 & I_0 & I_0 & I_1 & I_0 & I_0 & M_0 & M_0 \\ M_3 & I_0 & M_0 & M_0 & I_0 & M_3 & I_0 & I_0 & M_3 & I_1 & I_0 & M_3 & M_3 & I_0 & M_3 & I_0 & I_0 & M_0 & I_1 & I_0 & M_0 & I_0 & I_0 & I_0 & I_1 & M_0 & I_0 & M_0 \\ M_0 & M_3 & I_0 & M_3 & M_0 & I_0 & M_3 & I_0 & I_0 & M_3 & I_1 & I_0 & M_3 & M_3 & I_0 & M_0 & I_0 & I_0 & M_0 & I_1 & I_0 & I_0 & I_1 & I_0 & I_0 & M_0 & M_0 & I_0 \end{array}\right]$$

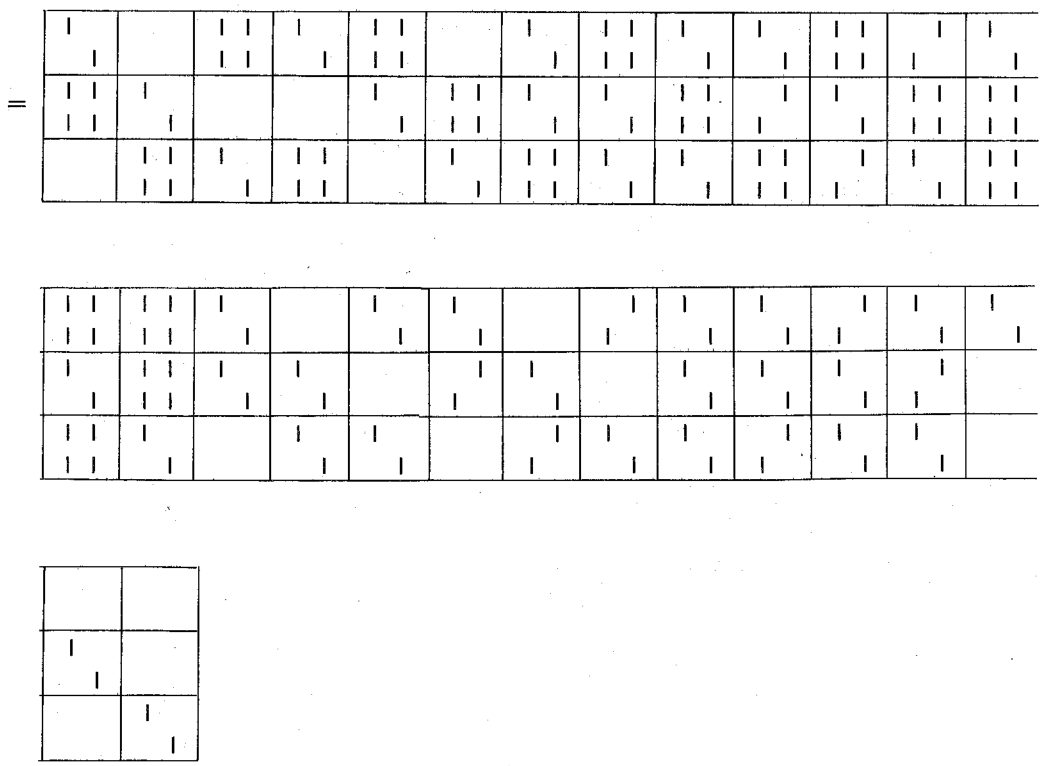


図３－６　(56, 50) ＳＥＣ－Ｓ２ＥＤ符号

表３－４　SEC-SbED符号（構成Ｂ）の符号長Ｎ

r* \ b	1**	2	3	4	5	6	7	8
2	3	12	39	112	295	732	1743	4032
3	7	56	387	2368	13085	67032	325591	1524224
4	15	240	3471	44800	517135	5460720		
5	31	992	29643	799744				
6	63	4032	246519					
7	127	16256						
8	255	65280						
9	511							

＊　検査ビット数＝ｒ・ｂ　　（空欄は未計算部）

＊＊　SEC 符号に一致

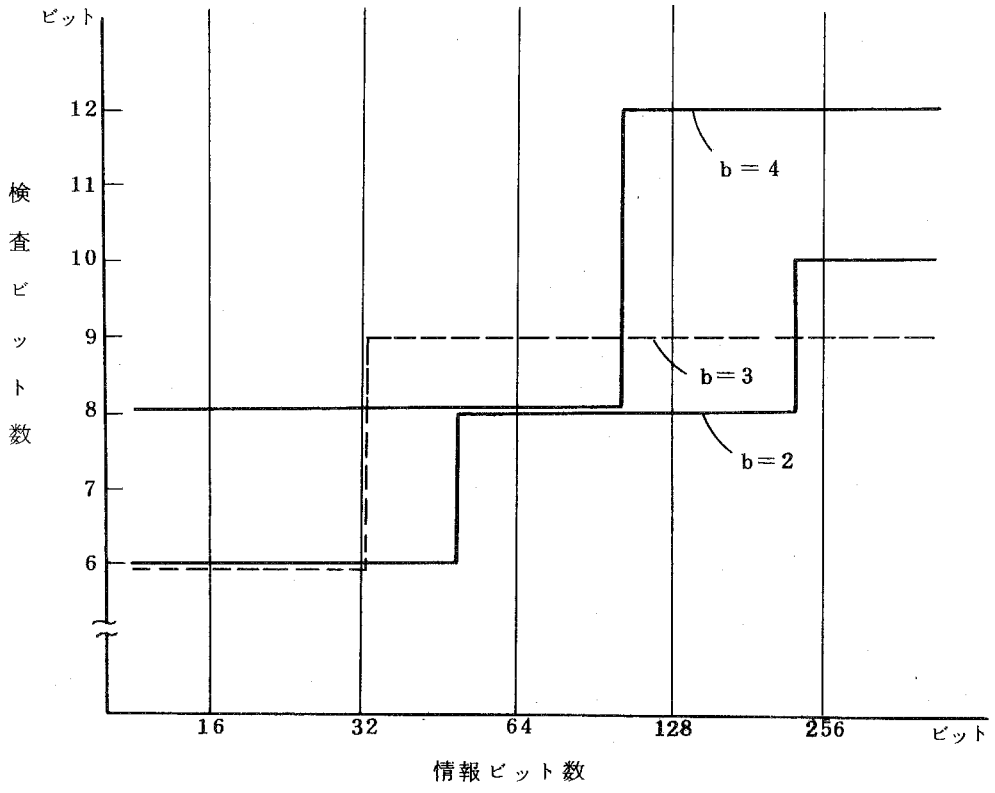


図３－７　SEC－SbED符号（構成Ｂ）における情報ビット数に対する検査ビット数の関係

次に，この符号の復号法についてのべる。１ビット誤り訂正のための並列復号は従来の方法と全く同一である。異なるのは，単一バイト誤り検出（ＳｂＥＤ）の機能である。本符号の単一バイト誤り検出の論理条件は，ｂビット単位ｒ個の非零シンドロームに対して，どのｂビット単位にも重み１のものはない場合に単一バイト誤りであると判定することができる。図３－５の符号例に対するＳ３ＥＤ回路を図３－８に示す。この符号の場合には，１２ゲート程度で構成できる。しかし，前段の８ゲートは本来１ビット誤り訂正に対するデコード回路に含まれるものであることから，実質のゲート数は４ゲートである。この回路はｂが大きい場合，あるいは符号長が大きい場合でも，実質のゲート増加は数ゲート程度である。

構成Ａの符号に対する本構成の符号の特徴を以下に示す。

（ｉ） 構成Ｂの符号は検査ビット数がｂビット単位に増加する。構成Ａの符号と比較して一般に能率は良くない。

（ii） 構成Ｂの符号は，第５章でのべる巡回性符号として容易に構成できる。構成Ａの符号では，巡回性符号として構成できない。

（iii） 構成Ｂの符号では検査ビット部が集中している。構成Ａの符号は分散している。

（iv） 構成Ｂの符号の方が，Ｈマトリクス行方向の重みが平均しており，一般に，符号化・復号化が高速に実行できる。

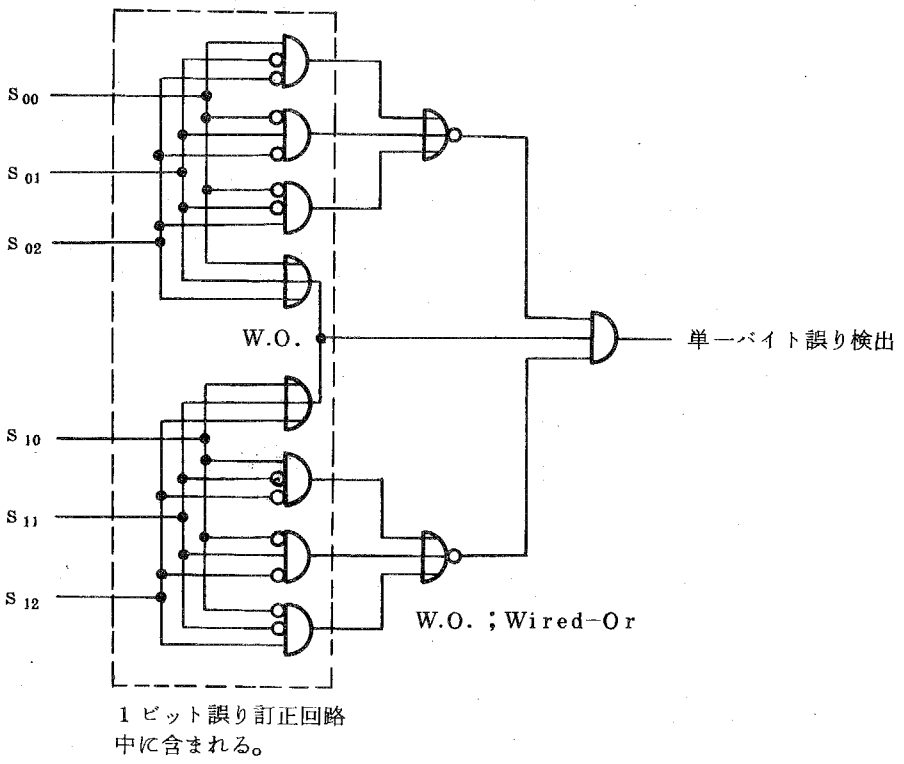


図３－８ 単一バイト誤り検出回路（図３－５の符号の場合）

3－3　SEC-DED-SbED 符号

本符号は，前述のＳＥＣ－ＳｂＥＤ符号に対して，さらに２ビット誤り検出（ＤＥＤ）の機能を追加したものである。これは，ＳＥＣ－ＳｂＥＤ符号からＨマトリクスの列ベクトルが奇数重みとなるものを選択してＨマトリクスを構成すれば，ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号を構成することができる。このような考えの下で，前節で提案した構成Ａ，構成Ｂの符号を基にして，それぞれＳＥＣ－ＤＥＤ－ＳｂＥＤ符号を構成する。

３－３－１　符号構成Ａ [24]

ＳＥＣ－ＳｂＥＤ符号の構成Ａを基にして，奇数重み列の考えを導入したＳＥＣ－ＤＥＤ－ＳｂＥＤ符号を構成する。H_f は単位行列でありその列ベクトルは常に重み１（奇数）をもち，H_e は常に重み０または２（偶数）を有することから，H_f に対しては上段に A_i として i を２進表示したとき偶数重みのものを，H_e に対しては i として奇数重みとなる A_i を並べれば奇数重み列を有するＨマトリクスを構成できる。この考えの下で構成した符号を図３－９に示す。また，このＨマトリクスは明らかにＳＥＣ－ＳｂＥＤ符号のＨマトリクスの一部であり，従ってＳＥＣ－ＳｂＥＤの機能は当然有する。以上から，次の定理が成立する。

$$H=\left[\begin{array}{cccccc|cccccc} \overbrace{A_0}^{b}\ \overbrace{\ }^{r-b} & A_3 & \cdots & A_i & \cdots & A_{N_e} & A_1 & A_2 & \cdots & A_j & \cdots & A_{N_f} \\ \underbrace{H_f}_{b}\ \underbrace{\ }_{b} & H_f & \cdots & H_f & \cdots & H_f & H_e & H_e & \cdots & H_e & \cdots & H_e \end{array}\right]$$

A_0，A_1，A_2，…，H_f，H_e は図３－１で定義したマトリクスである。

i ；i 自身の２進表示が偶数重みとなる整数

N_e ；i として（$2^{r-b}-1$）を越えない最大整数

j ；j 自身の２進表示が奇数重みとなる整数

N_f ；j として（$2^{r-b}-1$）を越えない最大整数

図３－９　ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号（構成Ａ）のＨマトリクス

（定理３－４）

図３－９に示す符号は，１ビットの誤りを訂正し，２ビットの誤りまたは単一バイト誤りを検出するＳＥＣ−ＤＥＤ−ＳｂＥＤ符号である。この符号の最大符号ビット長は次式に示すことができる。

$$N = b \cdot 2^{r-b} \qquad (3-13)$$

□

（証明）

図３－９に示す符号がＳＥＣ−ＤＥＤ−ＳｂＥＤの機能を有することは，上記の説明から明らかである。また，符号長については次のようになる。A_i（$i = 0, 1, \cdots, 2^{r-b}-1$）自身は，その列ベクトルとして奇数重みか偶数重みのいずれかである。しかも，iは0から$2^{r-b}-1$までの最大2^{r-b}個の値をとることができ，A_iとしてH_fまたはH_eのいずれかの上段に並べられる。これから，明らかに最大符号長は（３－１３）式で示すことができる。

（証明終り）

表３－５にｂ，ｒに対する符号長Ｎの関係を示す。また，図３－１０に情報ビット数（Ｎ−ｒ）に対する検査ビット数ｒの関係を示す。この符号は，ｂ＝１または２で従来のＳＥＣ−ＤＥＤ符号に一致する。図３－１０から明らかなように，本符号はｂ＝４で従来のＳＥＣ−ＤＥＤ符号に比較して１ビット，ｂ＝８で４ビットの増加で構成できる。図３－１１に（７３，６４）ＳＥＣ−ＤＥＤ−Ｓ４ＥＤ符号の例を示す。

復号化については，本符号はＳＥＣ−ＳｂＥＤ符号に対してＤＥＤの機能が追加されたものに等しい。これは，この符号が奇数重み列を有する符号であることから，すべては零でないシンドロームに対してこれらの排他的論理和を実行して零となるとき，２ビット誤りと判定すればよい。

表３－５　SEC-DED-SbED符号（構成Ａ）の符号長Ｎ

r \ b	1*	2*	3	4	5	6	7	8
2	2	2	—**	—	—	—	—	—
3	4	4	3	—	—	—	—	—
4	8	8	6	4	—	—	—	—
5	16	16	12	8	5	—	—	—
6	32	32	24	16	10	6	—	—
7	64	64	48	32	20	12	7	—
8	128	128	96	64	40	24	14	8
9	256	256	192	128	80	48	28	16
10	512	512	384	256	160	96	56	32
11	1024	1024	768	512	320	192	112	64
12	2048	2048	536	1024	640	384	224	128

*　SEC-DED 符号に一致

**　符号長が存在しないことを示す。

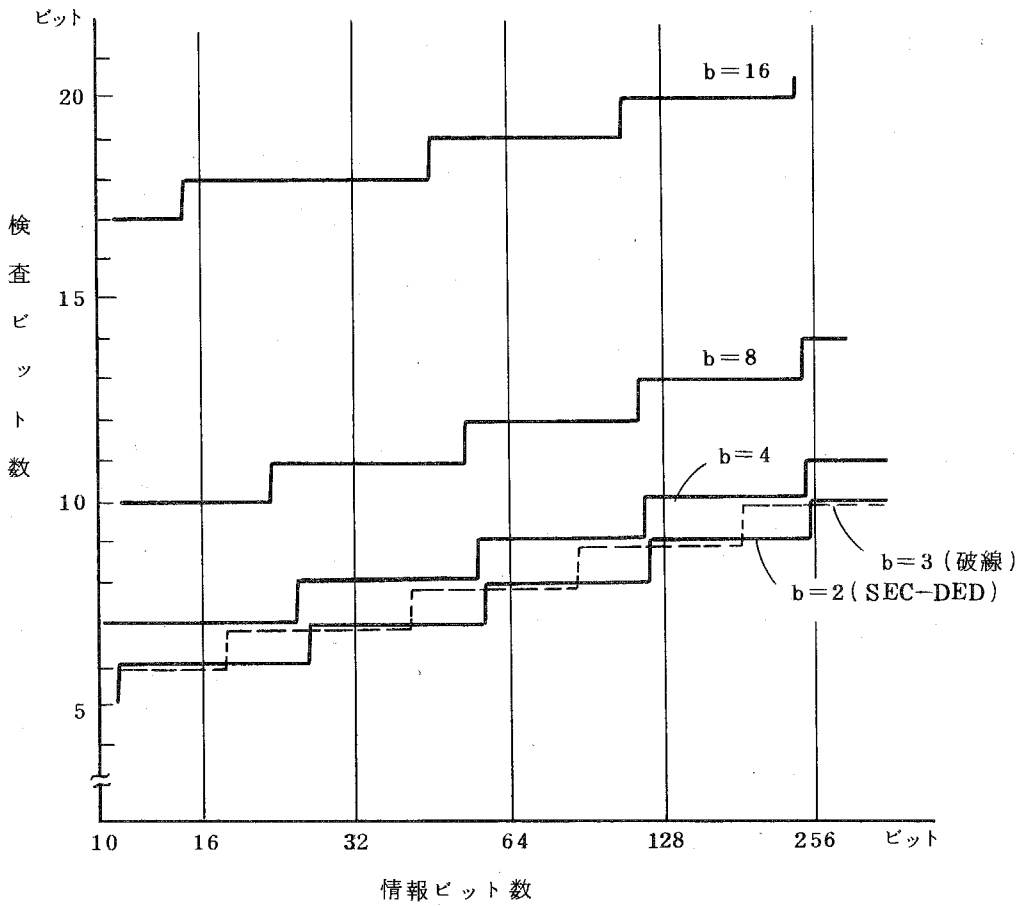


図3－10　SEC-DED-SbED符号（構成Ａ）における情報ビット数に対する検査ビット数の関係

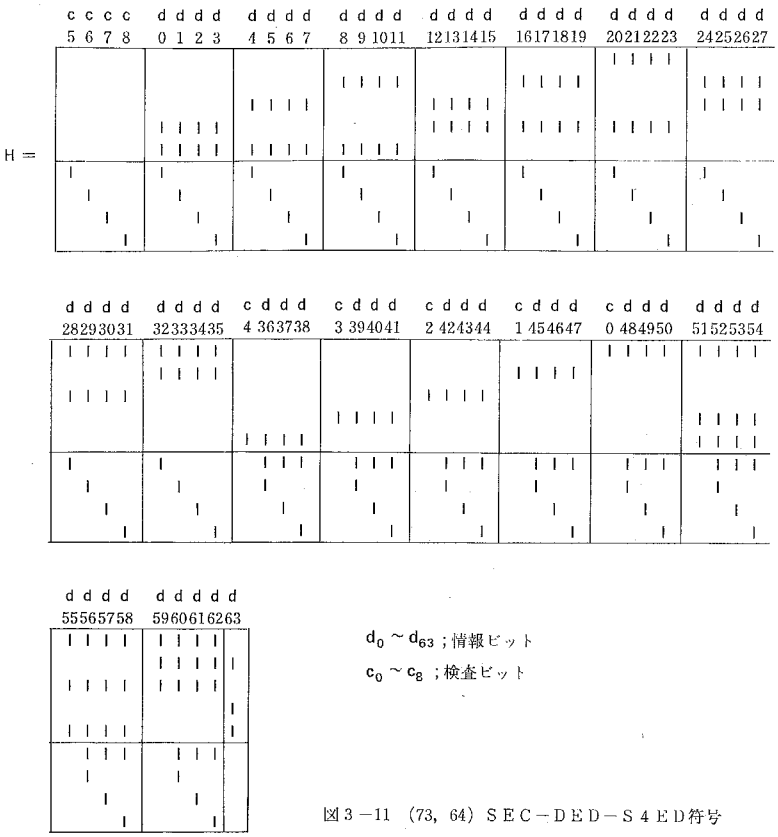


図3－11 (73, 64) SEC－DED－S4ED符号

3－3－2　符号構成B (25)

SEC－SbED符号の構成Bを基にして，奇数重み列の考えを導入したSEC－DED－SbED符号を構成する。符号構成要素として，正則行列は単位行列を採用し，I_q と M_u とから符号を構成することとする。

この符号構成においては，I_q の列ベクトルは常に重み1であり，M_u の列ベクトルの重みは奇数，偶数ともとり得る。これらの要素 I_q，M_u をr個並べ，しかも少くとも1個は I_q を含む条件を加味して全体としてHマトリクスが奇数重みの条件を満足させる。そのためには，r個の中に奇数個，重み奇数の要素が含まれねばならない。このような場合の数は次式で与えられる。

$$\sum_{\substack{e=1 \\ e\,;\,odd}}^{m} {}_rC_e \qquad (3-14)$$

$$m = 2\left\lceil \frac{r+1}{2} \right\rceil - 1$$

$\lceil x \rceil = x$ を越えない最大整数

例えば，r＝3のときには，次の4種類存在する。

$$\begin{bmatrix}奇\\奇\\奇\end{bmatrix}\quad\begin{bmatrix}奇\\偶\\偶\end{bmatrix}\quad\begin{bmatrix}偶\\奇\\偶\end{bmatrix}\quad\begin{bmatrix}偶\\偶\\奇\end{bmatrix}$$

このとき，線形独立性に注意して，「奇」の部分には I_q または M_u 中 u の2進表示が奇数重みのものを入れ，「偶」の部分には M_u 中 u の2進表示が偶数重みのものを入れる。但し，「奇」が1個の場合には必ず I_q を入れ，複数の「奇」が存在する場合でも，少くとも1個は I_q を入れなければならない。

M_u の中で偶数重み列を有するものは 2^{b-1} 個，奇数重み列を有するものは $(2^{b-1}-b)$ 個存在する。列ベクトル中 I_q の個数をpとするとき，「奇」が ℓ 個存在する列ベクトルの数 δ_ℓ は次式で求められる。

$$\begin{aligned}\delta_\ell &= (2^{b-1})^{r-\ell}\sum_{p=1}^{\ell}{}_\ell C_p\,(b)^{p-1}(2^{b-1}-b)^{\ell-p}\\ &=\frac{1}{b}\{(2^{b-1})^r-(2^{b-1})^{r-\ell}(2^{b-1}-b)^\ell\}\end{aligned}\qquad(3-15)$$

(3－14)式を考慮して，全体の列ベクトル数nは次式から求めることができる。

$$\begin{aligned}n &= \sum_{\substack{\ell=1\\ \ell:odd}}^{m}{}_r C_\ell\,\delta_\ell\\ &=\frac{1}{2b}\{2^{br}+b^r-(2^b-b)^r\}\end{aligned}\qquad(3-16)$$

また，ここで考慮した奇数重みの列ベクトルはそれぞれ互いに相異なることは明らかであり，よって(3－16)式から本構成のSEC－DED－SbED符号の符号長が求められた。

(定理3－5)

定理3－3で示される符号のHマトリクスのうち，奇数重みを有する列ベクトルにより構成したSEC－DED－SbED符号の最大符号ビット長Nは次式で与えられる。

$$N = n \cdot b = \frac{1}{2}\{ 2^{br} + b^{r} - (2^{b} - b)^{r} \} \qquad (3-17)$$

□

この符号のb，rに対する最大符号ビット長Nの関係を表３－６に示す。また，情報ビット数（N－br）に対する検査ビット数rbの関係を図３－１２に示す。これから，b＝４ビットで，情報ビット数６４ビットに対して検査ビット数は１２ビットとなり，構成Aの場合（検査ビット数は９ビット）と比較して大きい。

表３－６　SEC-DED-SbED符号（構成B）の符号長N

r* \ b	1**	2**	3	4	5	6	7	8
2	2	8	24	64	160	384	896	2048
3	4	32	207	1216	6605	33624	162967	762368
4	8	128	1776	22528	258880	2731008		
5	16	512	14943	400384				
6	32	2048	123624					
7	64	8192						
8	128							

＊　検査ビット数＝r・b　　（空欄は未計算部）

＊＊　SEC-DED符号に一致

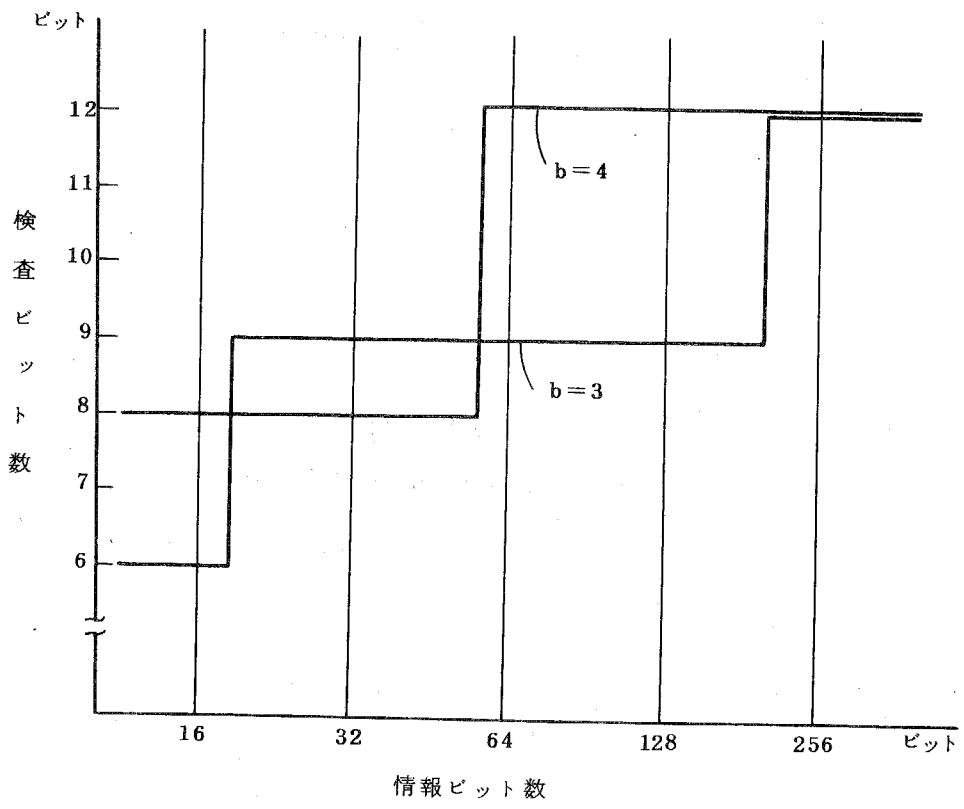


図３－12　SEC-DED-SbED符号（構成B）における情報ビット数に対する検査ビット数の関係

具体的な符号構成例を示そう。図３−１３は図３−５に対応した符号であり，ｂ＝３，ｒ＝２の（２４，１８）ＳＥＣ−ＤＥＤ−Ｓ３ＥＤ符号の例である。また，図３−１４はｂ＝３，ｒ＝３の場合の（２０７，１９８）ＳＥＣ−ＤＥＤ−Ｓ３ＥＤ符号の例である。図３−１５はｂ＝４，ｒ＝３の（７６，６４）ＳＥＣ−ＤＥＤ−Ｓ４ＥＤ符号の例である。

次に，復号化については，１ビット誤り訂正のための回路以外に，図３−８に示したと同様の単一ｂビットのバイト誤り検出回路と２ビット誤り検出回路を設けることにより実現できる。

$$H = \left[\begin{array}{cccccc|cc} I_0 & I_0 & I_0 & M_6 & M_5 & M_3 & I_0 & M_0 \\ M_6 & M_5 & M_3 & I_0 & I_0 & I_0 & M_0 & I_0 \end{array}\right]$$

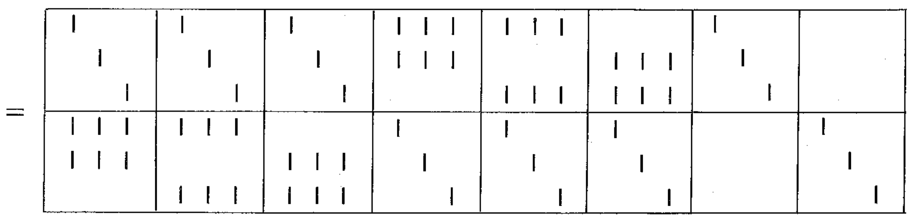


図３−13　(24, 18)　ＳＥＣ−ＤＥＤ−Ｓ３ＥＤ符号

H =

I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	M_0	M_0	M_0	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_0	M_0	M_0	M_3	M_3	M_3	M_5
M_3	M_5	M_6	M_0	M_0	M_0	M_3	M_3	M_3	M_5	M_5	M_5	M_6	M_6	M_6	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	M_0	M_0	M_0	M_3	M_5	M_6	M_3	M_5	M_6	M_3
M_0	M_0	M_0	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_3	M_5	M_6	M_0	M_0	M_0	M_3	M_3	M_3	M_5	M_5	M_5	M_6	M_6	M_6	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0	I_0

																	*							*	*			
M_5	M_5	M_6	M_6	M_6	I_0	M_7	M_7	I_0	I_0	I_0	I_0	I_1	I_2	M_7	M_7	M_7	I_0	I_0	I_1	I_0	I_0	I_2	I_0	I_0	I_0	I_0	M_0	M_0
M_5	M_6	M_3	M_5	M_6	M_7	I_0	M_7	I_0	I_1	I_2	M_7	M_7	M_7	I_0	I_0	I_0	I_0	I_0	I_0	I_1	I_0	I_0	I_2	I_1	I_2	M_0	I_0	M_0
I_0	I_0	I_0	I_0	I_0	M_7	M_7	I_0	M_7	M_7	M_7	I_0	I_0	I_0	I_0	I_1	I_2	I_0	I_1	I_0	I_0	I_2	I_0	I_0	I_2	I_1	M_0	M_0	I_0

図３−14　(207, 198)　ＳＥＣ−ＤＥＤ−Ｓ３ＥＤ符号

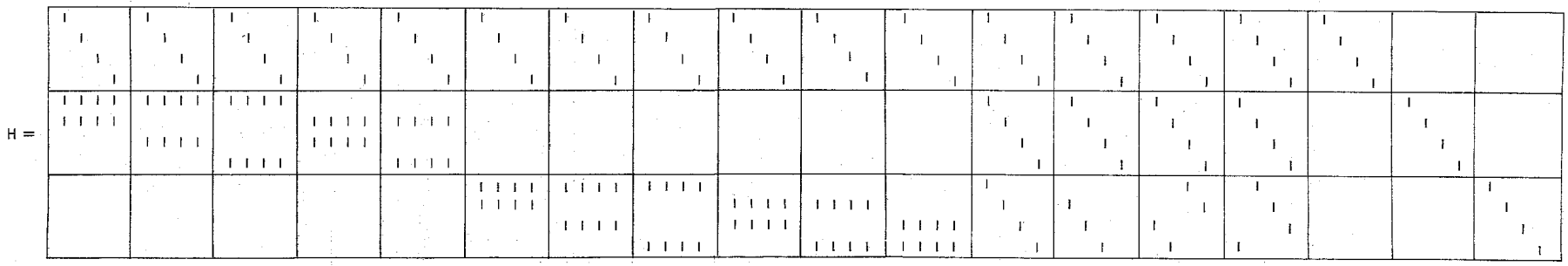


図３－15　(76, 64) ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号

3－4　任意の検査ビット数を有する符号構成B[25]

符号構成Bの符号は，SEC－SbED符号，SEC－DED－SbED 符号共検査ビット数はbの倍数となる特徴を有していた。 そこで，ここでは基本となるrb（rは2以上）ビットの検査ビットを有する符号構成Bの符号を基にして，検査ビットを（b－1）ビット以下の範囲で1ビットずつ増加させて，任意の検査ビット数を有する符号構成法を提案する。

3－4－1　SEC－SbED符号

検査ビット数rbビットを有するSEC－SbED符号に，ℓ（$0 \leq \ell \leq b-1$）ビットの検査ビットを付加する。

（定理3－6）

検査ビット数rbビットを有するSEC－SbED符号のHマトリクスをH_sとするとき，次に示すHマトリクス構成は検査ビット数rb＋ℓビットを有するSEC－SbED符号である。

$$H=\left[\begin{array}{ccccccc|c}
\overbrace{H_s}^{n_0} & \overbrace{H_s}^{n_0} & \overbrace{H_s}^{n_0} & \cdots & \overbrace{H_s}^{n_0} & \cdots & \overbrace{H_s}^{n_0} & \overbrace{\begin{matrix}W_0\\ \vdots \\ W_0\end{matrix}}^{\ell} \\
\hline
Q_0 & Q_1 & Q_2 & \cdots & Q_i & \cdots & Q_{2^{\ell}-1} & I_0^{\ell}
\end{array}\right]\begin{matrix}\\ \\ \\ \\ \\ \\ \\ \end{matrix}\begin{matrix}r\cdot b \\ \\ \\ \\ \\ \\ \ell\end{matrix} \qquad (3-18)$$

$$I_0^{\ell}=\left[\begin{array}{ccccc}1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \end{array}\right]_{\ell\times\ell} \qquad 0\leq\ell\leq b-1$$

$$Q_j=[\,J\ J\cdots J\,]_{\ell\times n_0} \qquad 0\leq j\leq 2^{\ell}-1$$

J ；jをℓビットで2進表示した列ベクトル

n_0；H_sの列ベクトル数（$n_0\leq 2^{br}-(2^b-b)^r$）

W_0；$b\times\ell$の零行列

また，この符号の最大符号ビット長は次式で表わすことができる。

$$N = \{ 2^{br} - (2^b - b)^r \} 2^e + \ell \qquad (3-19)$$

□

（証明）

（3－18）式に示す符号がSECの能力を有することは，各列ベクトルがすべて相異なることから明らかである。また，I_0^e を含む検査部を除くとSbEDの能力を有することも明らかである。I_0^e を含む部分で2ビット以上の誤りが生じたとき，上位rbビットのシンドロームは常に'0'であり，他の部分で生じた1ビット誤りのシンドロームと一致することはない。また，逆に他の部分におけるバイト誤りが I_0^e を含む部分の1ビット誤りと一致することはない。以上から，（3－18）式に示すHマトリクスはSEC－SbEDの能力を有する。また，H_s の最大符号長は（3－8）式に示すように $2^{br} - (2^b - b)^r$ である。ℓ 行からなる Q_e は最大 2^e 個存在し，また検査ビット数が ℓ ビット増加することから，この符号の最大長は（3－19）式となる。

（証明終り）

この符号の検査ビット数はR＝rb＋ℓ である。そこで，情報ビット数に対する検査ビット数の関係を図3－16に示す。図3－17にR＝7ビットの検査数を有する（79，72）SEC－S3ED符号の例を示す。これから，3－2－1の構成Aの符号に近い符号長を有する。

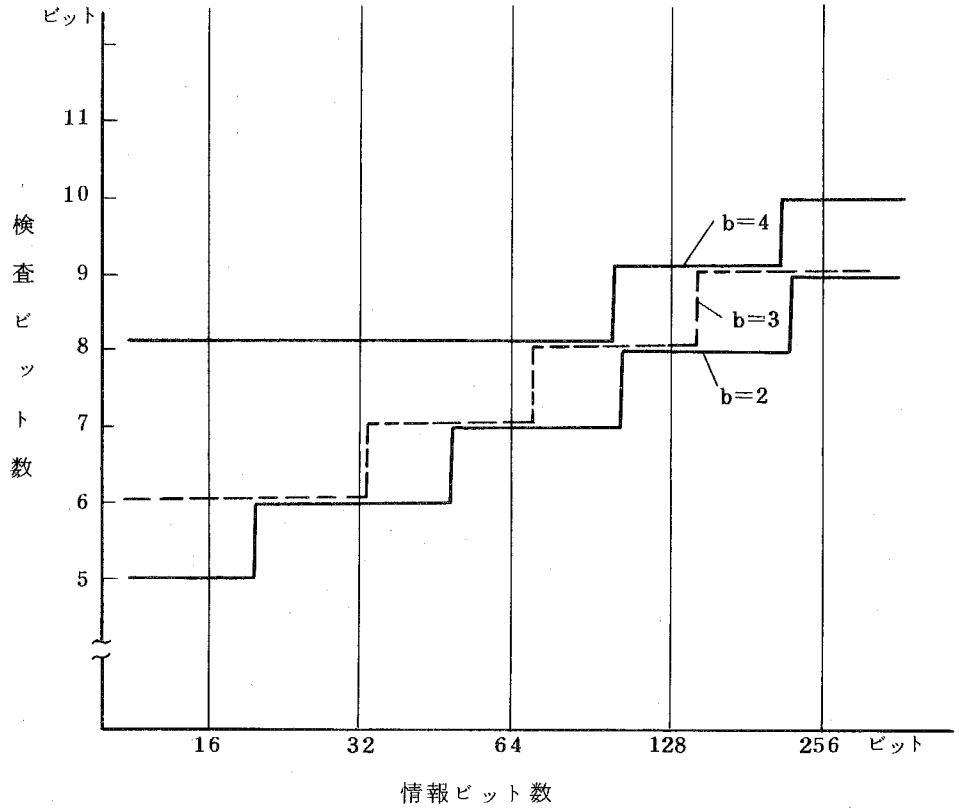


図3－16　SEC－SbED符号（構成Bの修正形）の情報ビット数に対する検査ビット数の関係

3－4－2　SEC－DED－SbED符号

3－4－1でのべたSEC－SbED符号に対して，奇数重み列の条件を考慮すればこの符号を構成できることは容易に理解できる。そこで，まず検査ビット数r・bからなるSEC－SbED符号のHマトリクスを奇数重み列のみを集めて作成したマトリクスH_pと，偶数重み列のみを集めて作成したマトリクスH_qとに分割する。H_pの列ベクトル数をn_p，H_qの列ベクトル数をn_qとすると，$n_p + n_q = 2^{br} - (2^b - b)^r$に等しいはずである。そこで，$H_p$の下段にはそのℓビットの2進表示が偶数重みとなるJから作られるQ_jを，H_qの下段にはそのℓビットの2進表示が奇数重みとなるJ'から作られるQ_j，を設置することにより，全体を奇数重みとすることができる。また，最後に（3－18）式に示したと同じ単位行列I_0^{ℓ}を含む列を追加することにより，全体のHマトリクスを構成できる。この符号の例を前の図3－5で示した（39，33）SEC－S3ED符号を基本にして作成した（80，72）SEC－DED－S3ED符号を図3－18に示す。この符号は，次に示す符号構成となる。

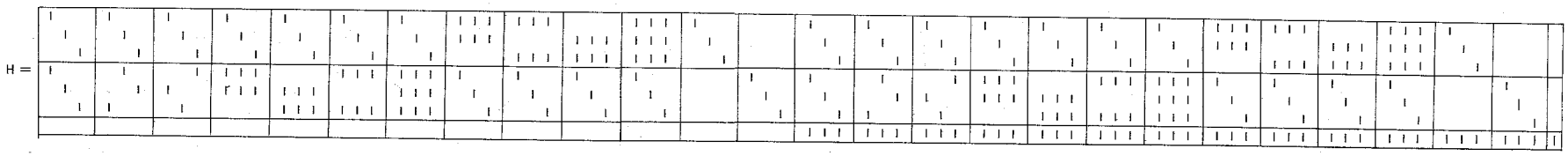


図３－17　(79, 72) ＳＥＣ－Ｓ３ＥＤ符号

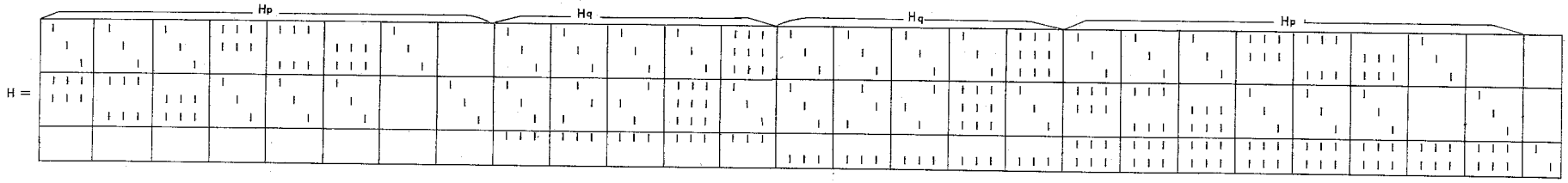


図３－18　(80, 72) ＳＥＣ－ＤＥＤ－Ｓ３ＥＤ符号

$$H = \left(\begin{array}{c|c|c|c|c|c} H_p & H_q & H_q & H_p & \begin{matrix} M_0 \\ M_0 \end{matrix} & 6 \\ \hline Q_0 & Q_1 & Q_2 & Q_3 & I_0^{\ell} & 2 \\ \multicolumn{2}{c}{n_p + n_q = 39} & \multicolumn{2}{c}{n_p + n_q = 39} & \ell = 2 & \end{array}\right) \qquad (3-20)$$

この符号では$\ell = 2$であり，全体の符号長はもとの符号長$n_p + n_q = 39$の2倍に$\ell = 2$ビットを加えただけ符号長が拡大している。以上から，この符号の最大符号ビット長は，一般に次式で表わすことができる。

$$N = \{ 2^{br} - (2^b - b) \} 2^{\ell - 1} + \ell \qquad (3-21)$$
$$(1 \leq \ell \leq b-1)$$

但し，$\ell = 0$またはbのときは，最大符号ビット長は（3－17）式に一致する。

本符号の情報ビット数に対する検査ビット数の関係を図3－19に示す。これから，b＝4ビットで，情報ビット数64ビットのときには検査ビット数は9ビットで構成できる。これは，3－3－1の構成Aで示した符号と同一の検査ビット数である。3－3－2の構成Bの符号では12ビットであったことを考慮すれば，この符号構成の効果は大きい。この（73，64）SEC－DED－S4ED符号の例を図3－20に示す。

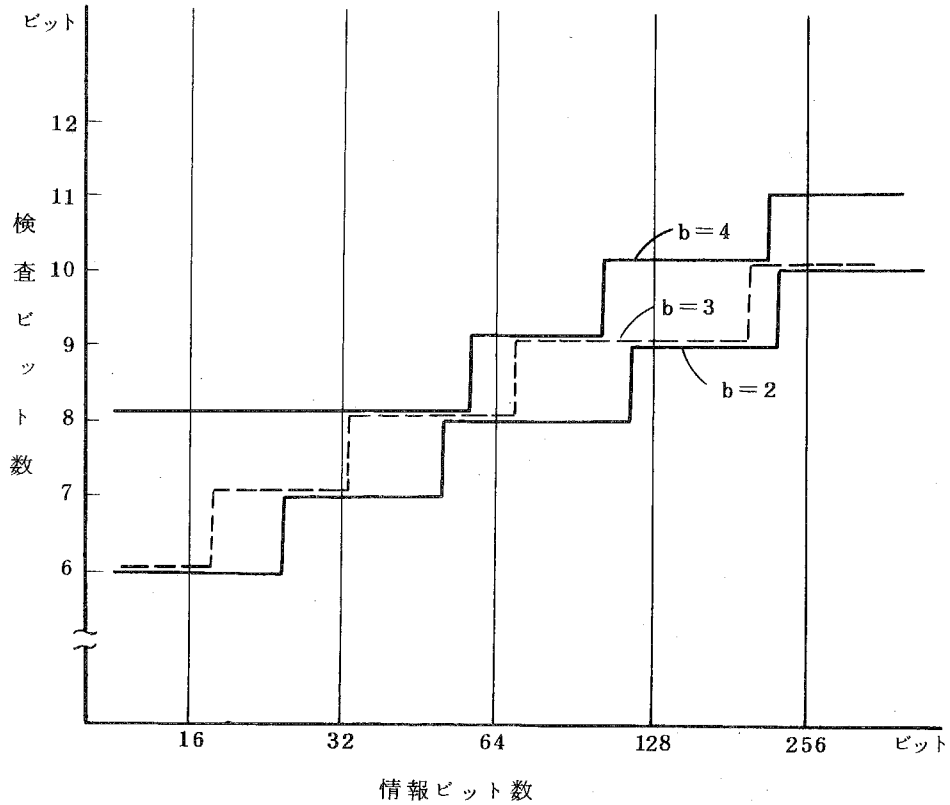


図3－19　SEC－DED－Sb ED符号（構成Bの修正形）の情報ビット数に対する検査ビット数の関係

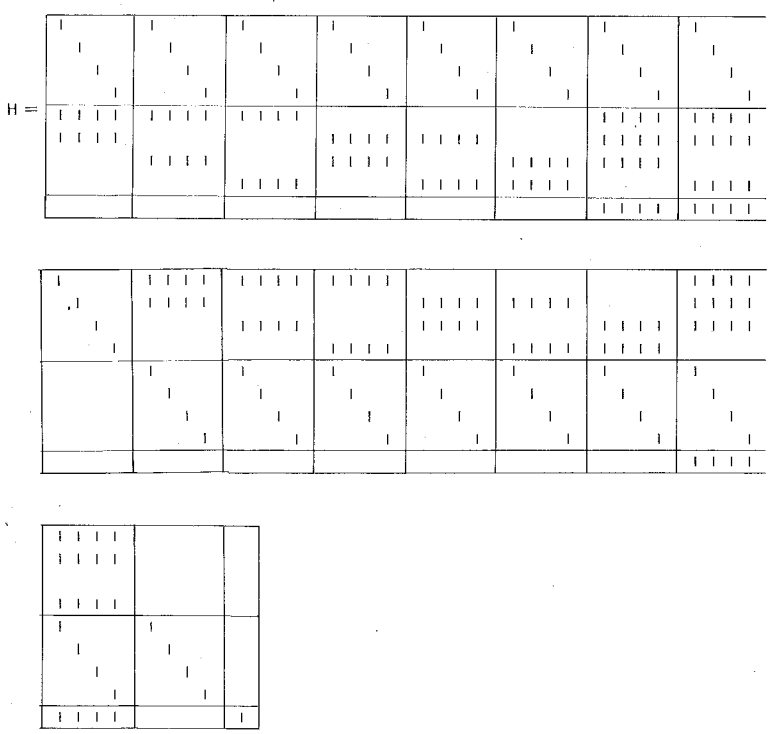


図3－20　(73, 64) SEC－DED－S4ED符号

本符号構成において，３－３－１で示した構成Ａで示した符号と比較して次の定理で示す範囲で符号長が大きくなる。

（定理３－７）

$1 \leq \ell \leq b-1$に対して（３－２１）式，および$\ell = 0$またはbに対して（３－１７）式で表現される本符号の最大符号ビット長は，（３－１３）式に示した構成Ａのそれと比較して，$b \geq 3$，検査ビット数$R \geq 3b$の範囲で大きい。 □

証明は長くなるので，付録１に示す。

3－5　SEC-(DED)-SbED 符号による単一バイト誤り訂正

ＳＥＣ－ＳｂＥＤ符号またはＳＥＣ－ＤＥＤ－ＳｂＥＤ符号（以下ＳＥＣ－（ＤＥＤ）－ＳｂＥＤ 符号と略す。）を用いて単一バイト誤り訂正を行う手法を示す。これは，あらかじめ固定的な障害が判明しており，相当するバイト位置が知らされているとき，そのバイト位置指定ポインタ（誤りバイトポインタ）を用いることにより当該バイト誤りを訂正する手法である。このようにポインタを与えて誤り検出・訂正符号の機能を拡大させる手法を，符号理論上ではＥｒａｓｕｒｅ Ｃｏｒｒｅｃｔｉｏｎと呼んでいる。[(1)～(5)]　このような訂正は，メモリ等であらかじめ欠陥位置が判明しているような場合に有用である。例えば，ＳＥＣ－ＤＥＤ符号の場合には，あらかじめ１ビットに相当する固定故障位置が判明しているとき，２ビット誤り検出時に２ビット誤りを訂正することができる。一般に，ｔビット誤り訂正－（ｔ＋１）ビット誤り検出符号においては，１ビット障害位置をあらかじめ知ることにより（ｔ＋１）ビットの誤りを訂正することができる。

ここでは，このような考えの下でＳＥＣ－ＳｂＥＤ符号，またはＳＥＣ－ＤＥＤ－ＳｂＥＤ 符号を用いて，あらかじめ単一バイト誤り位置を知ることにより，バイト誤り検出時にその誤りを訂正できることを示す。

（定理３－８）

ＳＥＣ－ＳｂＥＤ符号またはＳＥＣ－ＤＥＤ－ＳｂＥＤ符号において，あらかじめ単一バイト誤り位置を知ることにより，任意の単一バイト誤りを訂正することができる。　□

（証明）

ＳＥＣ－（ＤＥＤ）－ＳｂＥＤ符号において，単一バイト内のいかなる誤りもそのシンドロームは非零である。これは，単一バイト内の複数ビット誤りのシンドロームは，同一バイト内の他の異なる複数ビット誤りのシンドロームとは一致しないことを意味する。すなわち，単一バイト内のすべての誤りは，相異なるシンドロームを有することになる。従って，ＳＥＣ－（ＤＥＤ）－ＳｂＥＤ符号において単一バイト誤りを検出した場合，そのバイト位置を知ることができればすべてのバイト誤りは訂正できる。

（証明終り）

本符号を用いて単一バイト誤りを訂正できる手順を図３－２１に示す。　この手法によれば，その復号化回路は単一バイト内のすべての誤りを識別して訂正する回路が必要となり，ほぼＳｂＥＣ符号と同程度のゲート量が必要になる。しかし，本手法はＳｂＥＣ符号と比較して少ない検査ビット数で同一機能を実現できるところに長所が存在する。

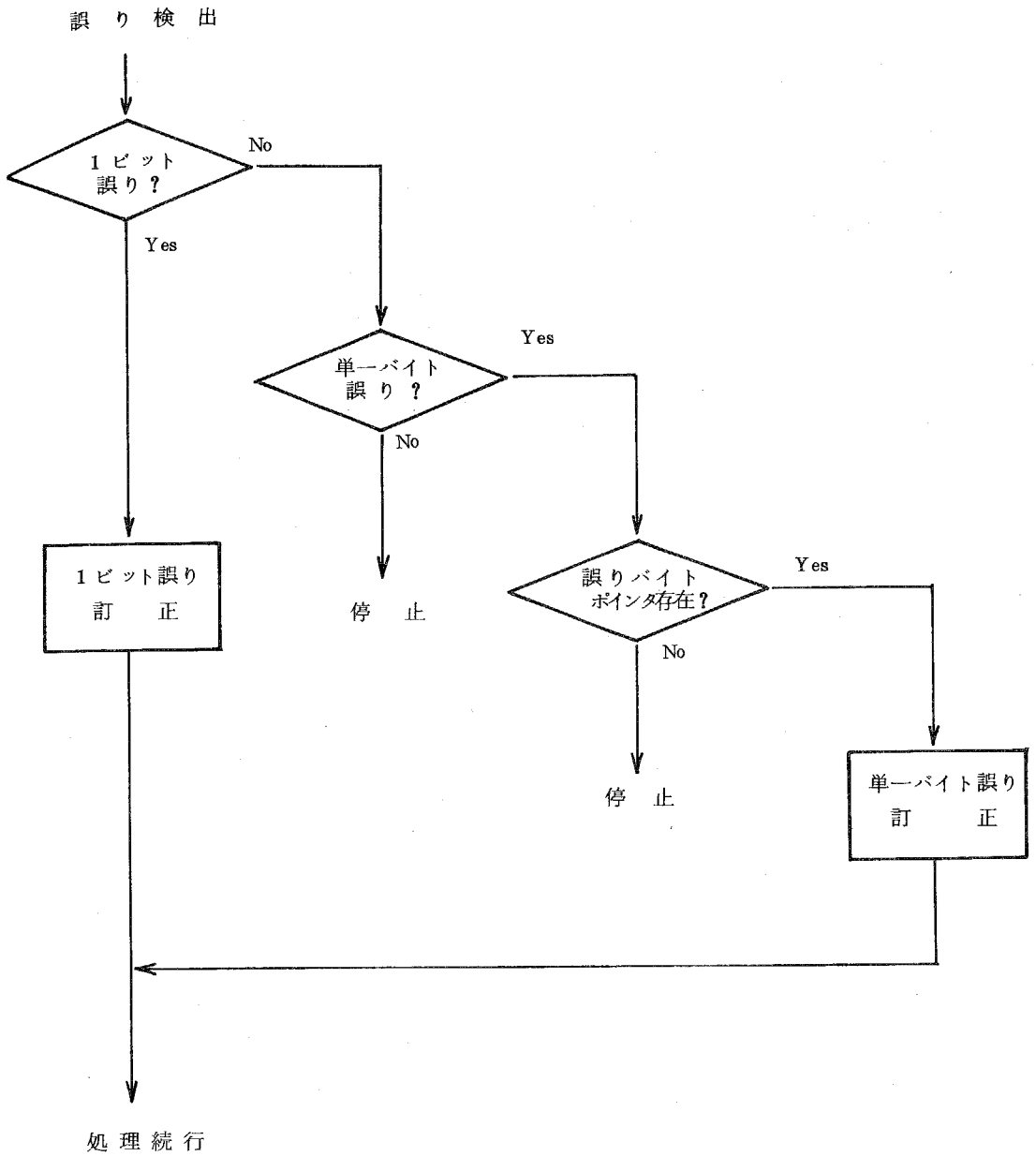


図3－21　ＳＥＣ－(ＤＥＤ)－ＳｂＥＤ符号を使用した単一バイト誤り訂正

3－6 符 号 評 価

SEC－SbED符号，SEC－DED－SbED符号において，保証される検出・訂正能力以上の誤りに対する検出能力について評価する。符号として，情報ビット数６４ビット，バイト長４ビットの場合をとりあげ，それぞれ図３－２のSEC－S４ED符号（構成Ａ），図３－11のSEC－DED－S４ED符号（構成Ａ），図３－１５のSEC－DED－S４ED符号（構成Ｂ），図３－２０のSEC－DED－S４ED符号（構成Ｂに対する修正符号）を対象とする。その結果を表３－７に示す。特に，各符号における２バイト間誤りに対する詳細評価結果をそれぞれ付録２に示す。これから，図３－１５に示すSEC－DED－S４ED符号は，ミスコレクト率は小さいが，これは検査ビット数が１２ビットと他の符号に比較して大きいためである。同じ９ビットの検査ビット数では，図３－２０に示す符号がすぐれている。

表３－７　情報ビット数６４ビットを有するSEC－S4ED符号，
SEC－DED－S4ED符号のミスコレクト率

機能 / 図番 / 検査ビット数 / 誤り種別	SEC－S4ED	SEC－DED－S4ED	SEC－DED－S4ED	SEC－DED－S4ED
	図３－２（構成Ａ）	図３－11（構成Ａ）	図３－１５（構成Ｂ）	図３－２０（構成Ｂの修正形）
	８ビット	９ビット	１２ビット	９ビット
ランダム２ビット誤り	３５.２１％	0	0	0
ランダム３ビット誤り	３５.９７％	３４.６５％	１０.９３％	３０.９５％
ランダム４ビット誤り	２９.９３％	０.５０％	０.１５％	０.４４％
ランダム２バイト誤り*	３６.３７％	２７.８０％	１４.７０％	２３.３３％

* 詳細は付録２に示す。

3-7 結　　言

本章は，容易な符号構成を有するSEC−SbED符号，SEC−DED−SbED符号について，2種類の符号を示した。いずれの符号共，その符号構成として始めから検査ビット位置が明確な構成を有し，しかも任意のバイト長に対して系統的に構成できる。主な結論を以下に示す。

(1) 符号構成Aにもとずく SEC−SbED 符号は，その符号ビット長は $N=b\cdot(2^{r-b}\pm1-1)$ に等しく，現在の所最大の符号長を有する。

(2) 符号構成Bは，原則として正則行列と整数の2進表示列ベクトルを並べた正方行列を用いて構成することを特徴とする。特に，正則行列としては単位行列を選択すると実用的である。符号構成上，必ずHマトリクス列ベクトル中に少くとも正則行列は1個以上存在することを条件とする。Hマトリクス列ベクトル中の要素の位置は規定されないことから，巡回性符号としてこの符号構成は適する。

(3) 符号構成Bは，その検査ビット数はbの倍数となる。そこで，この符号構成に対しても2bビット以上の検査ビット数を有する符号に対して，1ビットずつ検査ビットを増加させて構成する手法を示した。これにより，SEC−DED−SbED 符号に関して，符号構成Aによる符号と比較して，$b\geq3$，検査ビット数$\geq3b$の範囲において符号長は大きくなる特徴を有する。

(4) 符号構成Aにもとずく符号，および符号構成Bを修正して2b以上の任意の検査ビット数で構成できるようにした符号において，SEC−SbEDの機能に対しては，b＝4でSEC−DED符号と比較して検査ビット数はほぼ同一で構成でき，b＝8で3ビットの増加となる。SEC−DED−SbEDの機能に対しては，b＝4で1ビットの増加，b＝8で4ビットの増加である。

(5) 符号化・復号化のゲート量増加は，従来のSEC符号またはSEC−DED符号と比較して，SbEDの機能を満足する回路のみである。この回路は，実質的に5〜10ゲートで構成でき，ゲート量の増加は非常に小さい。

(6) 本章で示したバイト誤り検出符号を用いて，単一バイト誤り訂正を行う手法を示した。これは，固定的な誤りを有するバイト位置指定ポインタ（誤りバイトポインタ）を復号化の過程において提示しておくことにより，当バイト中の誤りを訂正する原理である。この手法は，ゲート量は単一バイト誤り訂正符号とほぼ同一の量となるが，検査ビット数が少なくできる点で有利である。

(7) 情報ビット数64ビット，バイト長b＝4ビットを有し，最少検査ビット数9ビットを有するSEC−DED−S4ED符号における誤り検出能力を評価した。その結果，ランダム3ビ

ット誤りの検出能力は６５～７０％，ランダム４ビット誤りは９９.５％，ランダム２バイト誤りは７３～７７％である。

第4章　バイト誤り訂正符号

4−1 緒　　言

本章は，複数ビットの塊りであるバイト内に生じた任意の誤りを訂正するバイト誤り訂正符号について示す。実用的な観点から，単一バイト内のすべての誤りを訂正する単一バイト誤り訂正（Single b−bit byte Error Correcting，略してSbEC）符号，あるいはさらに二重のバイトにわたる誤りを検出する単一バイト誤り訂正・二重バイト誤り検出（Single b−bit byte Error Correcting−Double b−bit byte Error Detecting，略してSbEC−DbED）符号をとりあげ，新しい符号構成を示す。

SbEC符号としては，新しく奇数重み列の性質を有する誤り検出能力の高いSbEC 符号を提案する。SbEC−DbED符号としては，従来の最小符号間距離4を有するReed−Solomon符号とは異なる，任意の符号長，バイト長に対して構成できる新しい符号構成を示す。

4－2　SbEC　符　号

単一バイト内の任意の誤りを訂正できるＳｂＥＣ符号について，その符号構成法，符号長の一般式，復号法等について示す。まず，この機能を有する符号に関して既に示されている従来の符号を示し，次に新しく奇数重み列の性質を適用した符号について提案する。

4－2－1　従来のＳｂＥＣ符号

単一バイト内のすべての誤りを訂正するＳｂＥＣ符号は，符号理論上では，単一誤り訂正 q（$=2^b$）元線形符号，またはＢ２型（Ｔｙｐｅ　Ｂ２）バースト誤り訂正符号として知られている。[1]～[5]　Ｄ.Ｃ.Ｂｏｓｓｅｎ，Ｓ.Ｊ.Ｈｏｎｇ，Ａ.Ｍ.Ｐａｔｅｌ は，実用的な観点から構成したＳｂＥＣ符号を提案しており，これらは具体的に磁気テープ装置，超大容量記憶装置においてすでに応用されている。[32][33]　すなわち，Ｄ.Ｃ.Ｂｏｓｓｅｎ[42]は，単一誤り訂正 q 元線形符号の観点からＧＦ（2^b）上の元をＧＦ（２）上の元で表現するｂ×ｂの同伴行列を与えて符号を構成する手法を提案し，またＳ.Ｊ.Ｈｏｎｇ，Ａ.Ｍ.Ｐａｔｅｌ[43]は，検査ビット数がｂの倍数によらない最大符号長を与えるＳｂＥＣ符号を提案している。また，ＧＦ（2^b）上の単一誤り訂正Ｒｅｅｄ－Ｓｏｌｏｍｏｎ符号[47]，あるいはバースト誤り訂正符号の観点からＢｕｒｔｏｎ符号[74]が提案されている。また，今井，上柳は符号化・復号化回路のゲート量削減の観点から，逆行列形ＳｂＥＣ符号[44]を提案しており，松澤，当麻は，非常に高速な復号が可能な自己直交符号を拡張させたＳｂＥＣ符号[38]を提案している。

ここでは，Ｄ.Ｃ.Ｂｏｓｓｅｎにより"ｂ－ａｄｊａｃｅｎｔ（ｂ－隣接）符号"として提案された簡明なＳｂＥＣ符号をとりあげ，その構成，復号法等について紹介し，本章への導入とする。

ＧＦ（2^b）上の元を次に定義するｂ×ｂのＧＦ（２）上の元で表現する同伴行列により表わす。

（定義４－１）同伴行列[42][43]

ＧＦ（２）上のｂ次の原始多項式をｇ（ｘ）とするとき，次に示すｂ×ｂの正方行列Ｔを同伴行列と呼ぶ。

$$g(x)=\sum_{i=0}^{b} g_i x^i, \quad g_0=g_b=1$$

$$T = \begin{bmatrix} 0 & 0 & \cdots & 0 & g_0 \\ & & & & g_1 \\ & I_{b-1} & & & \vdots \\ & & & & g_{b-1} \end{bmatrix}_{b \times b} \qquad (4-1)$$

I_{b-1}：（b－1）×（b－1）の単位行列 □

この同伴行列に関して次の性質が成立する。(43)

（性質1） $x^y \equiv 1 \mod g(x)$ を満足する最小の整数yをeとする。

(i) Tは正則である。

(ii) $T^0 = T^e = I_b$ （b×bの単位行列）

(iii) $i \equiv j \mod e$ であれば $T^i = T^j$ □

（性質2） T^j の i 番目の列は（b－1）次の多項式である。$x^{i+j-1} \mod g(x)$ の係数をベクトルとしたものに等しい。 □

（性質3） vを $v(x) = \sum_{i=0}^{b-1} v_i x^i$ の係数（$v_0\ v_1 \cdots v_{b-1}$）を並べた列ベクトルとし，v'を $v'(x) = \sum_{i=0}^{b-1} v_i' x^i$ の係数（$v_0'\ v_1' \cdots v_{b-1}'$）を並べた列ベクトルとする。このとき，$x^i v(x) = v'(x) \mod g(x)$ であれば，$T^i \cdot v = v'$ が成立する。 □

このような性質を有する同伴行列Tを用いて，Tのべき乗の集合 $\{O, I, T, T^2, \cdots T^{2^b-2}\} = GF(2^b)$ を作成することができる。

これらの$GF(2^b)$上の元を用いて，r＝2行構成のHマトリクスにより表現できる2－冗長SbEC符号が構成できる。

$$H = \begin{bmatrix} I & I & I & \cdots & I & \cdots & I & I & O \\ I & T & T^2 & \cdots & T^j & \cdots & T^{2^b-2} & O & I \end{bmatrix} \qquad (4-2)$$

この符号の最大符号長Nは，明らかに次のように示すことができる。

$$N = b \cdot n = b \cdot (2^b + 1) \qquad (4-3)$$

この符号の復号法は次のようになる。すなわち，今j番目のバイトに誤りEが存在するとき，シンドローム$[S_0, S_1]^T$は次式で表わされる。

$$\begin{cases} S_0 = E \\ S_1 = T^j \cdot E \end{cases} \qquad (4-4)$$

これより，誤りEはS_0から直接求められ，誤りバイト位置を示す誤りバイトポインタG_jは，次式により求められる。

$$G_j = [\, S_1 = T^j \cdot S_0 \,] \qquad (4-5)$$

すなわち，〔　〕の中の式が成立するとき，$G_j = 1$とする。これから，j番目のデータD_jは次のように訂正されて，訂正データ$\hat{D}_j$を得る。

$$\begin{cases} G_j = 1 \text{のとき} \quad \cdots \quad \hat{D}_j = D_j \oplus E \\ \qquad\qquad\qquad\qquad\quad = D_j \oplus S_0 \\ G_j = 0 \text{のとき} \quad \cdots \quad \hat{D}_j = D_j \end{cases} \qquad (4-6)$$

（4－2）式に示すHマトリクスにおいては，上位行にGF（2^b）上の単位元Ⅰを並べ，下位行にGF（2^b）上の異なる元を並べた簡明な構成を有しており，このように構成すれば自動的に列ベクトル間の線形独立性が保証される。一般のrに対して線形独立性を満足する列ベクトルを並べれば，自動的にSbEC符号を構成できるはずである。このようにGF（2^b）上の元からr個の元を用いて列ベクトルを構成するとき，線形独立性を満足する最大列ベクトル数をnとすれば，この符号の最大符号長Nは次式で与えられる。(1)～(5)

$$N = b \cdot n = b \cdot \frac{2^{br}-1}{2^b-1} \qquad (4-7)$$

b，rに対するnの関係を表4－1に示す。また，情報ビット数に対する検査ビット数の関係を図4－1に示す。また，r＝4，b＝2の場合の（72，64）S2EC符号の例(42)を図4－2に示す。この符号は，Hマトリクス中の'1'の数が最小に近いように作成されている。

表４－１　ＳｂＥＣ符号の最大符号長 n*

r* \ b	1**	2	3	4	5	6	7	8
2	3	5	9	17	33	65	129	257
3	7	21	73	273	1057	4161	16513	65793
4	15	85	585	4369	33825	266305		
5	31	341	4681	69905	1082401			
6	63	1365	37449	1118481				
7	127	5461	299593					
8	255	21845	2396745					

* 符号ビット長 = n•b,　　検査ビット数 = r•b　　（空欄は未計算部）

** SEC 符号に一致

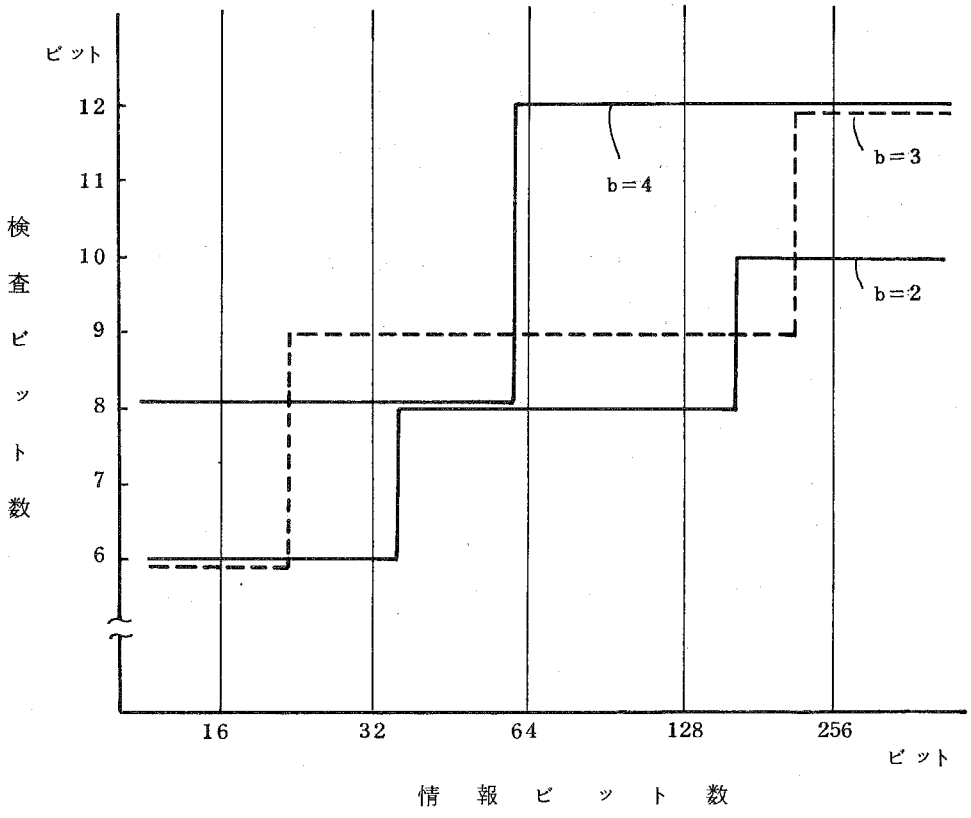


図４－１　ＳｂＥＣ符号における情報ビット数と検査ビット数の関係

$$H = \begin{bmatrix} I & I & I & T^2 & T & I & I & I & T^2 & T & 0 & 0 & I & I & I & I & I & I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & T & I & I & 0 & 0 & 0 \\ T^2 & T & I & 0 & 0 & 0 & I & I & I & I & T^2 & T & 0 & 0 & I & I & I & I & 0 & I & 0 & 0 & 0 & T^2 & T & I & I & I & I & 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I & I & I & T & T^2 & I & I & I & I & T^2 & T & 0 & 0 & 0 & I & I & I & I & I & I & 0 & 0 & 0 & T^2 & T & I & 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & T^2 & T & 0 & 0 & I & I & I & I & T^2 & T & I & 0 & I & I & T^2 & T & I & I & I & I & 0 & 0 & 0 & I & I & I & 0 & 0 & 0 & I \end{bmatrix}$$

$$0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \quad T^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

図４－２　(72, 64) Ｓ２ＥＣ符号の例[42]

４－２－２　奇数重み列ＳｂＥＣ符号[22][23]

従来の符号と同等な高速な並列復号を与え，かつ誤り検出能力の高い奇数重み列ＳｂＥＣ符号を新しく提案する。符号構成法，復号法，符号長の一般式，符号能力評価等について明らかにする。この符号の重要な特長は以下の点にある。

(ｉ)　ＧＦ(２)上で表現したＨマトリクスの列ベクトルは，すべて重みが奇数である。

(ii)　２バイト間にわたる誤りが，同一誤りパターンであればすべて検出できる。これから，２バイト間誤り検出能力が高い。

(iii)　ｂ＝１としたとき，ＳＥＣ－ＤＥＤ符号として従来最も誤り検出能力の点ですぐれている奇数重み列ＳＥＣ－ＤＥＤ符号[34]に一致する。これから，本提案の符号はこの符号を包含する符号と言える。

〔１〕　符号構成

ＧＦ(2^b)上の元を用いて構成したｒ行ｎ列のＨマトリクスを図４－３に示す。

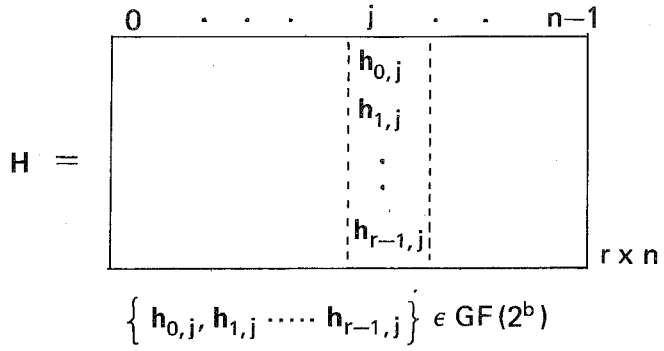


図４－３　Ｈマトリクス構成

（定理４－１）

（２－８）式の関係を有するＨマトリクスにおいて，これをＧＦ（２）上の元で表現したとき，すべての列ベクトルは奇数重みとなる。　□

（証明）

定義４－１に示したようにＧＦ（2^b）上のすべての元は，ｂ×ｂのＧＦ（２）上の元で表わした同伴行列のべき乗で表現することができる。特に，単位元Ｉはｂ×ｂの単位行列に等しく，零元はｂ×ｂの零行列に等しい。ＧＦ（2^b）上の元$h_{i,j}$を次のように表現する。

$$h_{i,j} = \begin{bmatrix} \cdot\cdot & \overset{e}{(a_{0,e})_i} & \cdot \\ \cdot\cdot & (a_{1,e})_i & \cdot \\ & \vdots & \\ \cdot\cdot & (a_{e,e})_i & \cdot \\ \cdot\cdot & \vdots\ (a_{b-1,e})_i & \cdot \end{bmatrix} \begin{matrix} (e \\ \\ \\ \\ \\ \\ \end{matrix} \quad \begin{matrix} e=0,\ 1,\ \cdots,\ b-1, \\ i=0,\ 1,\ \cdots,\ r-1, \end{matrix} \qquad (4-8)$$

$$\{(a_{0,e})_i,\ (a_{1,e})_i \quad \cdot\cdot \quad (a_{e,e})_i \quad \cdot\cdot \quad (a_{b-1,e})_i\ \} \in GF(2)$$

図４－３に示すＨマトリクスをもとに，各列ベクトルが（２－８）式の関係を満足することから，次式が成立する。

$$\sum_{\substack{i=0 \\ 0 \le e \le b-1}}^{r-1}{}^{\oplus} (a_{e,e})_i = 1\ , \qquad \sum^{\oplus} : \text{mod. 2和} \qquad (4-9)$$

これは，r個の各正方行列$h_{i,j}$，$i=0, 1, \cdots, r-1$の相当する対角要素の和（mod.2和）は単位行列の相当する対角要素の1に一致することを意味している。これから，集合$\{(a_{\ell,\ell})_0, (a_{\ell,\ell})_1, \cdots (a_{\ell,\ell})_{r-1}\}$は奇数個の1を持つことになる。

一方，次式が成立する。

$$\sum_{i=0}^{r-1}{}^{\oplus} (a_{m,\ell})_i = 0 \qquad 0 \leq m \neq \ell \leq b-1 \qquad (4-10)$$

これは，r個の各正方行列の相当する非対角要素の和（mod．2和）は，単位行列の相当する非対角要素の0に一致することを意味している。これから，集合$\{(a_{m,\ell})_0, (a_{m,\ell})_1, \cdots (a_{m,\ell})_{b-1}\}_{m \neq \ell}$は，偶数個の1を有することになる。以上，（4－9），（4－10）式から次の関係を得る。

$$\sum_{i=0}^{r-1}{}^{\oplus} \sum_{m=0}^{b-1}{}^{\oplus} (a_{m,\ell})_i = 1 \qquad (4-11)$$

これは，ℓ番目の列ベクトルが奇数個の1を有する奇数重み列ベクトルであることを示している。（4－11）式の関係はすべての整数ℓ，jに対して成立するものであり，これから，2進で表現したHマトリクスのすべての列ベクトルは奇数重みとなる。

（証明終り）

（定理4－2）

（2－8）式の条件を満足する相異なる2列ベクトルは互いに線形独立である。

□

（証明）

相異なる２列ベクトルを次のように表現する。

$$H_i = \begin{bmatrix} h_{0,i} \\ h_{1,i} \\ \vdots \\ h_{r-1,i} \end{bmatrix} \qquad H_j = \begin{bmatrix} h_{0,j} \\ h_{1,j} \\ \vdots \\ h_{r-1,j} \end{bmatrix}$$

$$0 \leq i \neq j \leq n-1$$

$$\{ h_{0,i}, h_{1,i}, \cdots h_{r-1,i}, h_{0,j}, h_{1,j} \cdots h_{r-1,j} \} \in GF(2^b)$$

仮定から，次の関係が成立する。

$$\sum_{k=0}^{r-1} h_{k,i} = 1 \qquad (4-12)$$

$$\sum_{k=0}^{r-1} h_{k,i} = 1 \qquad (4-13)$$

$GF(2^b)$上の任意の元βに対して，次の関係が成立したとする。

$$H_i = \beta \cdot H_j \qquad (4-14)$$

$$\beta \in GF(2^b) - \{ 0, 1 \}$$

（４－１４）式の右辺に対して，次式が成立する。

$$\beta \cdot \sum_{k=0}^{r-1} h_{k,j} = \beta \cdot 1 \neq 1 \qquad (4-15)$$

一方，（４－１４）式の左辺に対しては，（４－１２）式が成立する。これは，（４－１４）式の等号関係の仮定に反する。よって，（４－１４）式は成立せず，次のように表わすことができる。

$$H_i \neq \beta \cdot H_j \qquad (4-16)$$

これから，H_i とH_j は互いに線形独立であることが証明された。

（証明終り）

定理４－２から，（２－８）式を満足する列ベクトルを並べていけば，自動的に奇数重み列の性質を有するＳｂＥＣ符号のＨマトリクスが構成できる。

（２－８）式の性質を有する列ベクトルを数え挙げることにより，この符号の最大符号長を求めることができる。

（定理４－３）

ＧＦ（2^b）上において，ｒ個の元を用いて奇数重み列の条件を満足する列ベクトルは$2^{b(r-1)}$個存在する。 □

（証明）

（２－８）式を満足する（ｒ－１）個の元の和（ＧＦ（2^b）上の和）を今$\alpha \in$ＧＦ（2^b）とする。このとき，残りの１個の元は，$\alpha + 1$（＋はＧＦ（2^b）上の和）として与えられる。すなわち，ｒ個の元の内１個は残りの（ｒ－１）個の元により決められることになる。これから，（ｒ－１）個の元はＧＦ（2^b）上の2^b 個の任意の要素を選ぶことができ，よって奇数重み列の条件を満足する列ベクトル数ｎは，$n=(2^b)^{r-1} = 2^{b(r-1)}$ となる。

（証明終り）

定理４－３から，奇数重み列ＳｂＥＣ符号の最大符号長Ｎは次式で表わすことができる。

$$N = b \cdot n = b \cdot 2^{b(r-1)} \qquad (4-17)$$

表４－２にｂ，ｒに対するｎの関係を示す。特に，ｂ＝１の場合には，Ｈｓｉａｏ の提案による奇数重み列ＳＥＣ－ＤＥＤ符号に一致することに注意する必要がある。図４－４に情報ビット数に対する検査ビット数の関係を示す。奇数重み列ＳｂＥＣ符号の通常のＳｂＥＣ符号に対する符号長比rは次のように表わすことができる。

$$r = 1 - \frac{2^{b(r-1)}-1}{2^{br}-1} \qquad (4-18)$$

$$\approx 1 - \frac{1}{2^b} \qquad (r \cdot b \gg 1)$$

この関係を示したものを表４－３に示す。これから，b≧４では，奇数重み列の条件を付与したことによる符号長の短縮は非常に小さく，符号長への影響は小さい。

図４－５にb＝２，r＝４に対する奇数重み列（７２，６４）S２EC符号，図４－６にb＝４，r＝３に対する奇数重み列（７６，６４）S４EC符号の例を示す。それぞれの符号に使用する同伴行列は，各々既約多項式g(x)＝x^2+x+1，g(x)＝x^4+x+1より作成している。

表４－２　奇数重み列SbEC符号の符号長 n^*

r* \ b	1**	2	3	4	5	6	7	8
2	2	4	8	16	32	64	128	256
3	4	16	64	256	1024	4096	16384	65536
4	8	64	512	4096	32768			
5	16	256	4096	65536				
6	32	1024	32768					
7	64	4096						
8	128	16384						

＊　符号ビット長＝ n・b，検査ビット数＝ r・b　（空欄は未計算部）

＊＊　奇数重み列SEC−DED符号

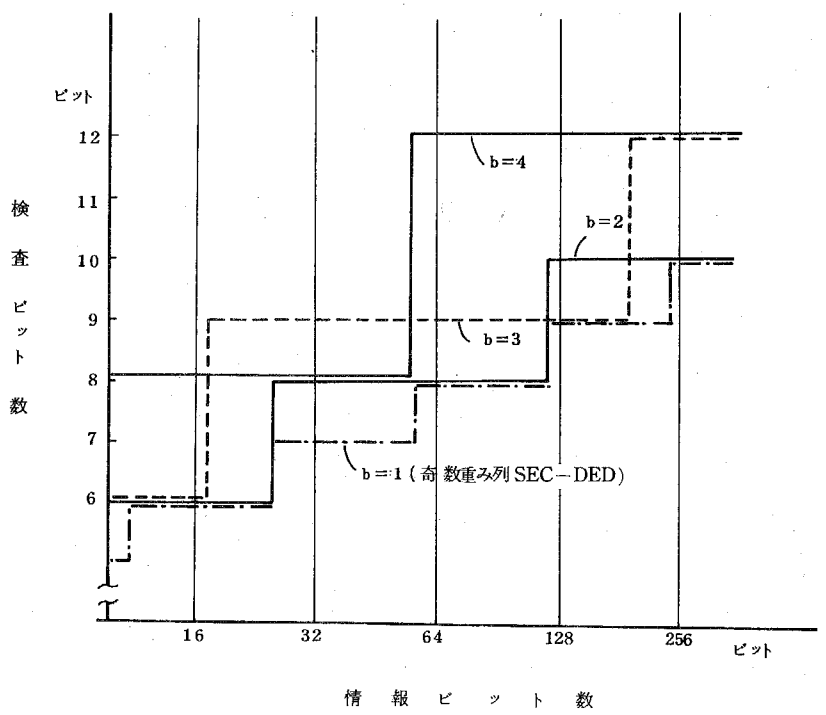


図４－４　奇数重み列ＳｂＥＣ符号の情報ビット数に対する検査ビット数の関係

表４－３　奇数重み列ＳｂＥＣ符号の符号長比*

(%)

r \ b	1**	2	3	4	5	6	7	8
2	66.7	80.0	88.9	94.1	97.0	98.5	99.2	99.6
3	57.1	76.2	87.7	93.8	96.9	98.4	99.2	99.6
4	53.3	75.3	87.5	93.75	96.87	98.4	〃	〃
5	51.6	75.1	87.5	93.75	96.87	〃	〃	〃
6	50.8	75.0	〃	〃	〃	〃	〃	〃
7	50.4	75.0	〃	〃	〃	〃	〃	〃
8	50.2	〃	〃	〃	〃	〃	〃	〃
	↓	↓	↓	↓				
	1-1/2	$1-1/2^2$	$1-1/2^3$	$1-1/2^4$				

* $(2^{b(r-1)}-1)/(2^{br}-1)$

** SEC-DED 符号と SEC 符号の符号長比

$$H = \begin{bmatrix} T & T^2 & T & I & I & T & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & I & T & I & T & T^2 & T^2 & T & 0 & I & T & T & I & I & I & 0 & 0 & 0 \\ T^2 & T & 0 & I & T & T & I & I & T & T^2 & T & I & I & T & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & I & T & I & T & T^2 & 0 & I & 0 & 0 \\ 0 & 0 & T^2 & I & T & I & T & T^2 & T^2 & T & 0 & I & T & T & I & I & T & T^2 & T & I & I & T & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & I & T & I & T & T^2 & T^2 & T & 0 & I & T & T & I & I & T & T^2 & T & I & I & T & T & T^2 & 0 & 0 & 0 & I \end{bmatrix}$$

$$0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \quad T^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

図４－５　奇数重み列（72, 64）Ｓ２ＥＣ符号

$$H = \begin{bmatrix} T^4 & T^{14} & T^{13} & T & T^3 & T^6 & 0 & 0 & 0 & 0 & 0 & 0 & T & T^3 & T^6 & T^4 & I & 0 & 0 \\ T & T^3 & T^6 & T^4 & T^{14} & T^{13} & T^4 & T^{14} & T^{13} & T & T^3 & T^6 & 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & T & T^3 & T^6 & T^4 & T^{14} & T^{13} & T^4 & T^{14} & T^{13} & T & 0 & 0 & I \end{bmatrix}$$

$$T = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 0 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{pmatrix} T^{14} + T^3 = I \\ T^{13} + T^6 = I \\ T^4 + T = I \end{pmatrix}$$

図４－６　奇数重み列（76, 64）Ｓ４ＥＣ符号

〔2〕 復号法

図4－3に示すHマトリクスの構成を有する奇数重み列SbEC符号の復号法を示す。今，j番目のバイトに誤りパターン$E_j=(e_0, e_1, \cdots, e_{b-1})$（$e_i \in \{0, 1\}$，誤りのとき$e_i=1$）が生じているとき，シンドローム$S_0 \sim S_{r-1}$は次式により求められる。

$$S_i = h_{i,j} \cdot E_j \qquad (4-19)$$

$$i=0, 1, \cdots, r-1$$

（2－8）式を考慮して，（4－19）式の両辺を$i=0$からr－1まで加算（GF(2^b)上の加算）すると，次式の様に誤りパターンE_jが得られる。

$$E_j = \sum_{i=0}^{r-1} S_i \qquad (4-20)$$

$$j=0, 1, \cdots, n-1$$

E_jは誤りバイト内の誤りパターンを示すものであり，復号化のためにはさらに誤りバイト位置を示す誤りバイトポインタが必要である。j番目の誤りバイトポインタG_j（$j=0, 1, \cdots, n-1$）は次式により求められる。

$$G_j = \bigcap_{i=1}^{r-1} \left[S_i = h_{i,j} \cdot h_{0,j}^{-1} \cdot S_0 \right] \qquad (4-21)$$

式中，〔　〕内の＝は等号関係を示し，∩は論理積条件を示す。すなわち，（r－1）個の〔　〕の中の関係がすべて成立するとき$G_j=1$となる。

j番目のバイトを示す入力データをD_jとするとき，訂正後のデータ$\hat{D}_j$は次のように表わすことができる。

$$\begin{cases} G_j = 1 \quad \cdots\cdots \quad \hat{D}_j = D_j \oplus E_j \\ \qquad\qquad\qquad\quad = D_j \oplus \sum_{i=0}^{r-1} S_i \\ G_j = 0 \quad \cdots\cdots \quad \hat{D}_j = D_j \end{cases} \qquad (4-22)$$

以上の復号手法を図４－７に示す。この符号の復号化回路は，従来の高速な並列復号を行うＳｂＥＣ符号と比較して，ほぼ同一のゲート量と回路遅延を有する。具体的なゲート量，回路遅延の比較は第７章に示す。

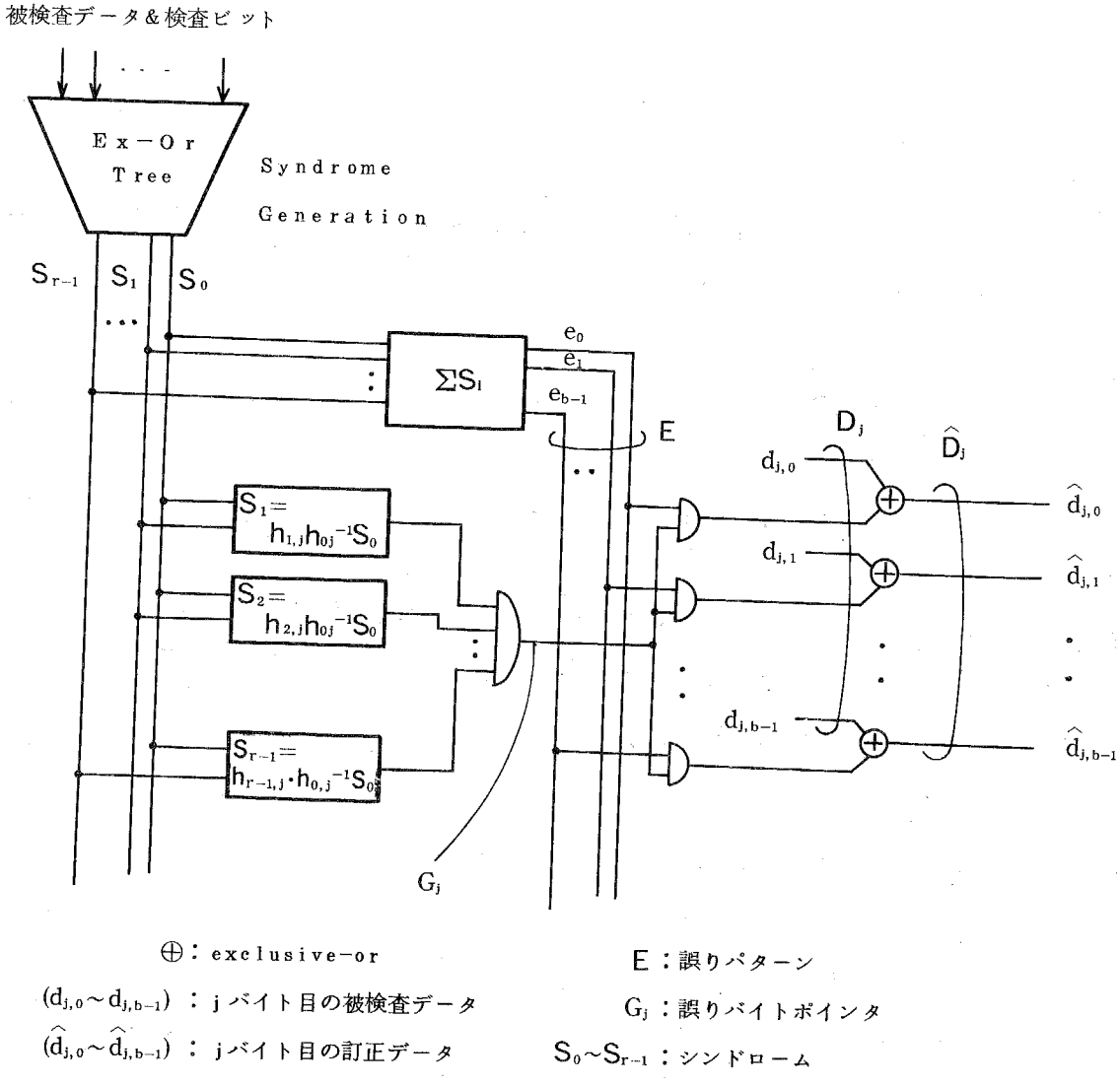


図４－７　奇数重み列ＳｂＥＣ符号の復号

〔3〕 符号評価

2バイト間の誤りに対して次の定理が成立する。

（定理4－4）

i番目のバイトの誤りパターンをE_i，j番目のバイトの誤りパターンをE_jとすると，奇数重み列SbEC符号においては$E_i = E_j$であれば，このような2バイト間にわたる誤りはすべて検出できる。 □

（証明）

2バイト間の誤りを検出できるためには次の条件が必要である。すなわち，i番目，j番目，k番目のバイト誤りをそれぞれE_i，E_j，E_kとし，H_i，H_j，H_kをそれぞれHマトリクスのi番目，j番目，k番目の列ベクトルとする。

$$H_i = \begin{bmatrix} h_{0,i} \\ h_{1,i} \\ \vdots \\ h_{r-1,i} \end{bmatrix}, \quad H_j = \begin{bmatrix} h_{0,j} \\ h_{1,j} \\ \vdots \\ h_{r-1,j} \end{bmatrix}, \quad H_k = \begin{bmatrix} h_{0,k} \\ h_{1,k} \\ \vdots \\ h_{r-1,k} \end{bmatrix}, \tag{4-23}$$

このとき，次の関係が成立しなければならない。

$$\begin{aligned} &E_i \cdot H_i + E_j \cdot H_j + E_k \cdot H_k \neq 0 \\ &E_i \neq 0, \quad E_j \neq 0, \quad E_k \neq 0 \\ &i, j, k \in \{0, 1, 2, \cdots, n-1\} \end{aligned} \tag{4-24}$$

（4－23）式に対しては，奇数重み列符号の条件から次式が成立する。

$$\sum_{t=0}^{r-1} h_{t,i} = 1, \quad \sum_{t=0}^{r-1} h_{t,j} = 1, \quad \sum_{t=0}^{r-1} h_{t,k} = 1 \tag{4-25}$$

今，（4－24）式のr個のすべての関係式を加算すると，（4－25）式を利用して次式を得る。

$$E_i + E_j + E_k \neq 0 \tag{4-26}$$

これから，もし$E_i = E_j$であれば，この式は$E_k \neq 0$であることから常に成立する。これから，任意の2個の誤りパターンが等しければ，（4－26）式は常に成立することになり，このような2バイト間誤りはすべて検出できることになる。

（証明終り）

この定理4－4は，部分的な二重バイト誤り検出（Double b bit−byte Error Detecting，略してDbED）の機能を有する符号と言うことができる。特に，2バイト間でバイト内の全ビット誤り，すなわち，2bビットの誤りに対してはすべて検出できることになる。さらに，奇数重み列の性質から次の関係が成立するはずである。

1） 2バイト間にまたがる偶数ビットの誤りは，単一バイト内の奇数ビット誤りとして誤訂正（ミスコレクト）することはない。

2） 2バイト間にまたがる奇数ビット誤りは，単一バイト内の偶数ビット誤りとして誤訂正（"正しい"と誤って判断する場合も含む）することはない。

以上から，奇数重み列SbEC符号は，2バイト間にまたがる誤りに対して高い検出率を有することが予想できる。

そこで，具体的な符号を設定してこれを確認してみよう。バイト長b＝2ビット，情報ビット数64ビット，符号長72ビットのS2EC符号を例にとり詳細に評価してみる。奇数重み列S2EC符号と奇数重み列の性質を有さないS2EC符号の場合でミスコレクト率（ミスコレクト数／全誤りパターン数）を比較したものを図4－8，図4－9に示す。Hマトリクス中の'1'の数（Hマトリクスの重み）を横軸にとり，2バイト間のミスコレクト率を縦軸に示している。図4－8は，2バイト間の平均のミスコレクト率を示すものであり，図4－9は発生頻度が高いと考えられる2バイト間にわたる2ビット誤りの場合のミスコレクト率を示す。これから，一般に奇数重み列S2EC符号はそうでない符号に対して数%程度2バイト間誤りの検出能力が高く，特に2バイト間にわたる2ビット誤りに対しては15%程度検出能力が高い。

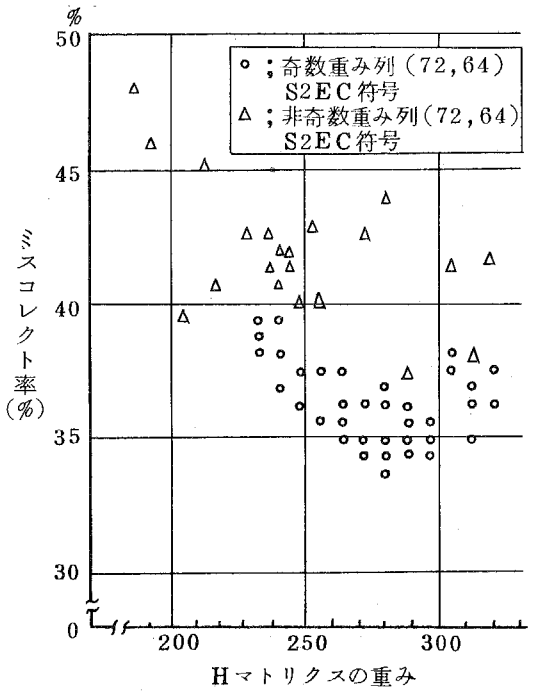


図4－8　2バイト間誤りに対するミスコレクト率の比較

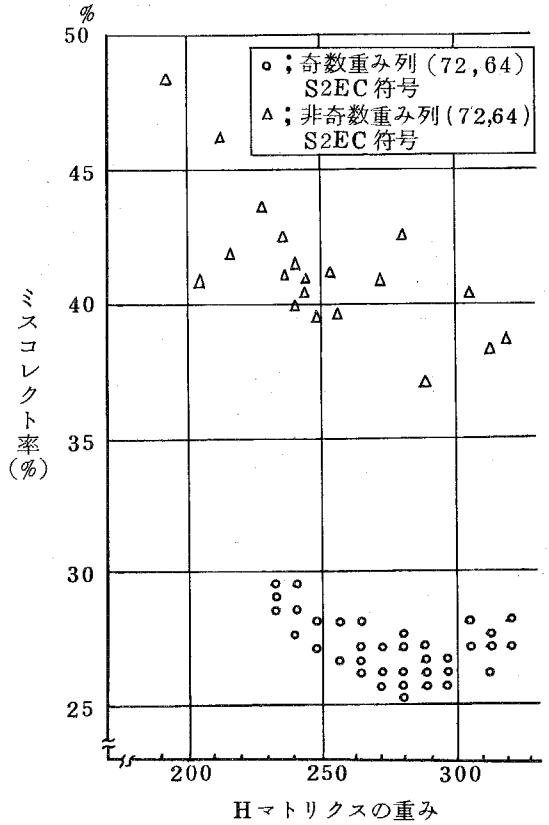


図4－9　2バイト間にわたる2ビット誤りに対するミスコレクト率の比較

現在までに求めた奇数重み列S2EC符号で最も検出能力の高い符号を図4－10に示す。図4－11は図4－10に示した符号の詳細な評価結果を示す。これは，2バイトの誤りE_i，E_jの各誤りパターンに対するミスコレクトの場合の数を具体的に示したものであり，これから$E_i = E_j$のときミスコレクトする場合は全く存在せず，定理4－4に示した性質を示している。また，$E_i = E_j$以外でミスコレクト数が零となるのは，前記1)，2)に相当するためである。図4－2に，奇数重み列の条件を有さない最小に近い重みのHマトリクスを有する符号を示した。この符号に対する同様の詳細な評価を図4－12に示す。表4－4に図4－10に示す符号と図4－2に示す符号のミスコレクト率を整理して示す。

$$H = \left[\begin{array}{cccccccccccccccccccccccccccccccccccc}
I & I & I & T & T & I & T^2 & T^2 & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & T & T & T & I & T^2 & T^2 & I & I & I & T & I & T & T^2 & I & T^2 & I & 0 & 0 & 0 \\
I & I & T & I & T & T^2 & I & T^2 & I & I & I & T & T & I & T^2 & T^2 & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & T & T & T & I & T^2 & T^2 & I & 0 & I & 0 & 0 \\
T^2 & T & T & T & I & T^2 & T^2 & I & I & I & T & I & T & T^2 & I & T^2 & I & I & I & T & T & I & T^2 & T^2 & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I & 0 \\
T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 & T^2 & T & T & T & I & T^2 & T^2 & I & I & I & T & I & T & T^2 & I & T^2 & I & I & I & T & T & I & T^2 & T^2 & 0 & 0 & 0 & I
\end{array}\right]$$

図4－10　奇数重み列（72，64）S2EC符号

E_j ＼ E_i	10	01	11
10	0	0*	324
	0	312**	0
01	0	0	312
	324	0	0
11	312	324	0
	0	0	0

* 上段; 1ビット誤りとしてミスコレクトする数
** 下段; 単一2ビット-バイト誤りとしてミスコレクトする数

図4－11　2バイト間誤り（E_i，E_j）に対する評価
（図4－10に示す符号）

E_j ＼ E_i	10	01	11
10	269*	160	156
	80**	99	103
01	160	269	156
	99	80	103
11	202	202	160
	57	57	189

＊上段；1ビット誤りとしてミスコレクトする数
＊＊下段；単一2ビット・バイト誤りとしてミスコレクトする数

図4－12　2バイト間誤り（E_i，E_j）に対する評価（図4－2に示す符号）

表4－4　（72，64）S2EC符号のミスコレクト率

符　　号	2バイト間誤りに対するミスコレクト率	2バイト間2ビット誤りに対するミスコレクト率
従来の符号		
符号1（図4－2）	45.9％	48.3％
符号2*	41.9％	41.6％
奇数重み列符号		
符号1（図4－10）	33.6％	25.2％

＊　文献（21）の図4に示す符号

4－3　SbEC-DbED 符号

ＳｂＥＣ符号は単一バイト内のすべての誤りを訂正するが，２バイトにまたがる，特に２ビットの誤り等に対しては検出の保証を与えていない。これに対処するため，ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号が効果的であるが，２バイトにまたがる任意の誤りに対する検出能力は持ち合わせていない。ＳｂＥＣ－ＤｂＥＤ符号は，このような観点に立って単一バイトの誤りを訂正し，２バイトにまたがる任意の誤りを検出する機能を有する符号である。

ＳｂＥＣ－ＤｂＥＤ符号としては，従来符号間最小距離４を有するＲｅｅｄ－Ｓｏｌｏｍｏｎ符号[47]が提案されている。また，この符号を使用した並列符号化・復号化回路が示されている。[48]この符号は，ｂが大きい領域には構成が容易であるが，ｂが小さい領域，例えばｂ＝４ビット以下では，情報ビット数６０ビットまでしかとれず，例えば情報ビット数６４ビット，１２８ビットに対して符号が構成できない。そこで，ｂが小さい領域に対しても系統的に構成できる新しいＳｂＥＣ－ＤｂＥＤ符号が要求されていた。

Ｗ.Ｃ.Ｃａｒｔｅｒ，Ａ.Ｂ.Ｗａｄｉａは，これを解決するため，あらかじめ必要なバイト長（ｂ）より大きいｑを選んでＲｅｅｄ－Ｓｏｌｏｍｏｎ符号を構成しておき，その後各バイトをｂビットに削って必要な情報ビット数を確保する簡便な符号構成法を提案した。[50]　本節では，この手法とは異なる新しい符号構成法を提案する。

４－３－１　修正Ｒｅｅｄ－Ｓｏｌｏｍｏｎ符号

〔１〕　修正符号１

Ｒｅｅｄ－Ｓｏｌｏｍｏｎ符号は，バイト系符号を考えるとき，ＴをＧＦ（2^b）上の元とし，ｎをＴの位数とするとき，次式で与えられる符号である。[47]

$$H=\begin{bmatrix} I & T & T^2 & \cdots & T^{n-1} \\ I & T^2 & T^4 & \cdots & T^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ I & T^{d-1} & T^{2(d-1)} & \cdots & T^{(d-1)(n-1)} \end{bmatrix} \qquad (4-27)$$

この符号は，符号長がＧＦ（2^b）上でｎ，検査長が（ｄ－１），最小ハミング距離 $d_{min}=d$ を有する符号である。ＳｂＥＣ－ＤｂＥＤ符号としては，$d_{min}=4$ であるため，（４－２７）式は次の

ように書くことができる。

$$H=\begin{bmatrix} I & T & T^2 & & T^i & & T^{2^b-2} \\ I & T^2 & T^4 & \cdots & T^{2i} & \cdots & T^{2(2^b-2)} \\ I & T^3 & T^6 & & T^{3i} & & T^{3(2^b-2)} \end{bmatrix} \qquad (4-28)$$

ここで，Tは原始多項式g(x)から作成される同伴行列であり，そのべき乗の集合は，{O, I, T, T^2, …, T^{2^b-2}}＝GF(2^b)である。このときの最大符号長はN＝b(2^b-1)である。

符号の能力を変えないで(4－28)式に3列の検査部を追加できることがJ.K.Wolfにより提案されている。(75)

$$H=\left[\begin{array}{ccccccc|ccc} I & T & T^2 & & T^i & & T^{2^b-2} & I & O & O \\ I & T^2 & T^4 & \cdots & T^{2i} & \cdots & T^{2(2^b-2)} & O & I & O \\ I & T^3 & T^6 & & T^{3i} & & T^{3(2^b-2)} & O & O & I \end{array}\right] \qquad (4-29)$$

このように符号長を拡張した符号をここでは，修正Reed－Solomon符号(修正符号1)と呼ぶことにする。この符号の最大符号ビット長Nは次式で表わすことができる。

$$N=b(2^b+2) \qquad (4-30)$$

bに対するNの関係を示すと次のようになる。

b	1	2	3	4	6	8	12	16
N	4	12	30	72	396	2064	49176	1048576

検査ビット数　R＝3b

これから，このように符号を拡長してみても，b＝2，3，4ビットに対して，情報ビット数64，128ビットを有する符号が構成できないことがわかる。

次に，この修正符号1をもとにして，次の簡便な修正符号2を提案する。

〔２〕　修正符号２

（４－２８）式の第１行目の非零元をすべて単位元Ｉとするために，各列ベクトルを規格化する。このような変形を行っても符号の機能は変わらないことは自明である。

$$H=\left[\begin{array}{ccccccc:ccc} I & I & I & & I & & I & I & 0 & 0 \\ I & T & T^2 & \cdots & T^i & \cdots & T^{2^b-2} & 0 & I & 0 \\ I & T^2 & T^4 & & T^{2i} & & T^{2(2^b-2)} & 0 & 0 & I \end{array}\right] \qquad (4-31)$$

そこで，この情報部のすべて"Ｉ"とした第１行目に注目して，この行を奇数個使用することにより符号を拡張する。（４－３１）式は一般的に次のように表現することができる。

$$H=\left[\begin{array}{ccccccc:ccc} I & I & \cdot & \cdot & \cdot & & I & I & 0 & 0 \\ \multicolumn{7}{c:}{M} & 0 & I & 0 \\ & & & & & & & 0 & 0 & I \end{array}\right] \qquad (4-32)$$

ここで，Ｍは２行（2^b-1）列の（４－３１）式で情報部の最上位行を除いた部分に一致する。このＭとすべて"Ｉ"の行を用いて，一般に次のようにして符号を拡張させることができる。

$$H=\left[\begin{array}{ccccccc:ccc} Q_1 & Q_2 & Q_4 & \cdots & Q_i & \cdots & Q_{N_{r_0}} & I & & \\ & & & & & & & I & \ddots & \\ \multicolumn{7}{c:}{\text{---}} & & \ddots & I \\ M & M & M & \cdots & M & \cdots & M & & & I \end{array}\right] \begin{array}{l} r_0 \\ \\ \\ 2 \end{array} \qquad (4-33)$$

（空白は零元）　$r=r_0+2$

ここで，Q_i は i をＩにより２進表示したとき，その中にＩが奇数個含まれる列ベクトルを（2^b-1）個並べた行列を意味する。

$$Q_1=\left[\begin{array}{ccc} 0 & 0\cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0\cdots & 0 \\ I & I\cdots & I \end{array}\right]_{r_0\times(2^b-1)} \quad Q_2=\left[\begin{array}{ccc} 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0\cdots & 0 \\ I & I\cdots & I \\ 0 & 0\cdots & 0 \end{array}\right]_{r_0\times(2^b-1)} \quad Q_4=\left[\begin{array}{ccc} 0 & 0\cdots & 0 \\ \vdots & & \vdots \\ I & I\cdots & I \\ 0 & 0\cdots & 0 \\ 0 & 0\cdots & 0 \end{array}\right]_{r_0\times(2^b-1)}$$

$$Q_7=\begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ I & I & \cdots & I \\ I & I & \cdots & I \\ I & I & \cdots & I \end{bmatrix}_{r_0\times(2^b-1)} \qquad Q_8= \qquad \cdots$$

N_{r_0} は（$2^{r_0}-1$）を越えない整数で，Iにより2進表示したときIが奇数個含まれる最大整数である。

b＝2の場合の構成例を示そう。（4－31）式から，その情報部は次のようになる。

$$\left[\begin{array}{ccc} I & I & I \\ \hline I & T & T^2 \\ I & T^2 & T \end{array}\right] = \left[\begin{array}{ccc} I & I & I \\ \hline & M & \end{array}\right] \qquad M=\begin{bmatrix} I & T & T^2 \\ I & T^2 & T \end{bmatrix}$$

これを単位にr＝6のとき，図4－13に示す（60，48）S2EC－D2ED符号を構成することができる。

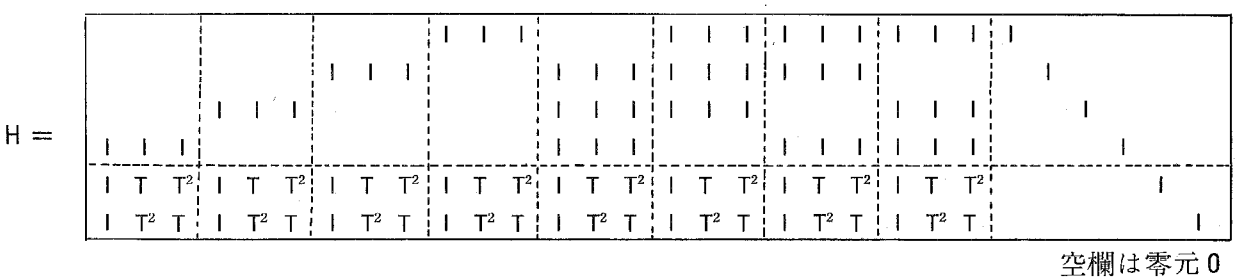


図4－13 （60，48）S2EC－D2ED符号

（定理4－5）

（4－33）式に示す符号はSbEC－DbEDの機能を有する。この符号の最大符号ビット長は次式となる。

$$N=b(2^b-1)2^{r-3}+rb \qquad (4-34)$$

□

（証明）

各列ベクトルはすべて相異なり，線形独立であることから，単一バイト誤りは訂正できる。2バイト間にわたる誤りに対しては次のようになる。今，（2^b-1）の長さのM単位内に2バイト誤りが生じたとき，Mの上段には必ず少くとも1行すべて"1"の行が存在することから，これはReed－Solomon符号に一致し，完全に検出できる。また，異なるM単位間にまたがる2バイト誤りに対しては，必ずシンドローム上位r_0個の元の中には偶数個非零元のシンドロームが存在し，これから他の単一バイト誤りのシンドロームと一致することはなく検出できる。また，検査部中の1バイト誤りと任意のM単位中の1バイト誤りに対しても，同様に1バイトの誤りとしてミスコレクトすることはない。以上から，（4－33）式は単一バイト誤り訂正・二重バイト誤り検出の機能を有する。

次にこの符号の符号長を求める。検査長は（4－33）式から明らかなように$r=r_0+2$である。よって，検査ビット数は$R=b\cdot r=b(r_0+2)$である。Mの長さは$b(2^b-1)$ビットであり，Mの個数はr_0に依存する。すなわち，r_0中すべて"1"の行は奇数行存在できることから，このような場合の数は最大2^{r_0-1}である。これから，情報部の最大ビット長は$K=b(2^b-1)\cdot 2^{r_0-1}$となる。以上から，この符号の最大長は，

$$N=K+R=b(2^b-1)2^{r_0-1}+b(r_0+2)$$

となる。ここで，$r_0=r-2$を代入して

$$N=b(2^b-1)2^{r-3}+br$$

を得る。

（証明終り）

r，bに対するNの関係を表4－5に示す。これから，任意のbに対して，任意の符号長のSbEC－DbED符号を構成できることになる。これから，b＝2で情報ビット数64ビットの符号を構成するためには，検査ビット数は14ビット必要となる。また，情報ビット数128ビットに対しては16ビットとなる。b＝4では情報ビット数64ビットに対して16ビット，128ビットに対しては20ビットの検査ビットを必要とする。

この符号は簡便に構成できる利点は存在するが，必ずしも符号としての能率（符号長に対する検査長の割合）は良くない。そこで，この考え方をさらに拡張して，能率の良い新しい符号構成を次に提案する。

表４－５　ＳｂＥＣ－ＤｂＥＤ符号（修正符号２）の符号長Ｎ

r \ b	1	2	3	4	5	6	7	8
3	3	12	30	72	135	396	910	2064
4	6	20	54	136	260	780	1806	4112
5	9	34	99	260	505	1542		
6	14	60	186	504	990			
7	23	110	357	988				
8	40	208	296					
9	73	402						
10	138							

（空欄は未計算部）

４－３－２　新しいＳｂＥＣ－ＤｂＥＤ符号[26]

〔１〕　符号構成

この符号は，既存の２種類のＳｂＥＣ－ＤｂＥＤ符号を組合わせて，新しいＳｂＥＣ－ＤｂＥＤ符号とするものである。既存の２個の符号のＨマトリクスをH_V，H_Wとする。これらは，$GF(2^b)$上で最小ハミング距離４をもつ符号である。V_i（$i=0, 1, \cdots, n-1$）をマトリクスH_V中の列ベクトルとする。一方，他方の符号のＨマトリクスH_Wについて，すべての要素が１となる１行を有するように変形する。このように常に変形できることは容易に証明できる。このように変形したマトリクスをH_W'とする。次に，このすべて"１"を有する１行をH_W'からとり除き，このマトリクスをH_W''とする。このH_W''中のｊ番目の列ベクトルをW_j（$j=0, 1, \cdots, m-1$）とする。以上の関係を図４－１４に示す。以上の準備の下で，次の定理により新しい符号を構築することができる。

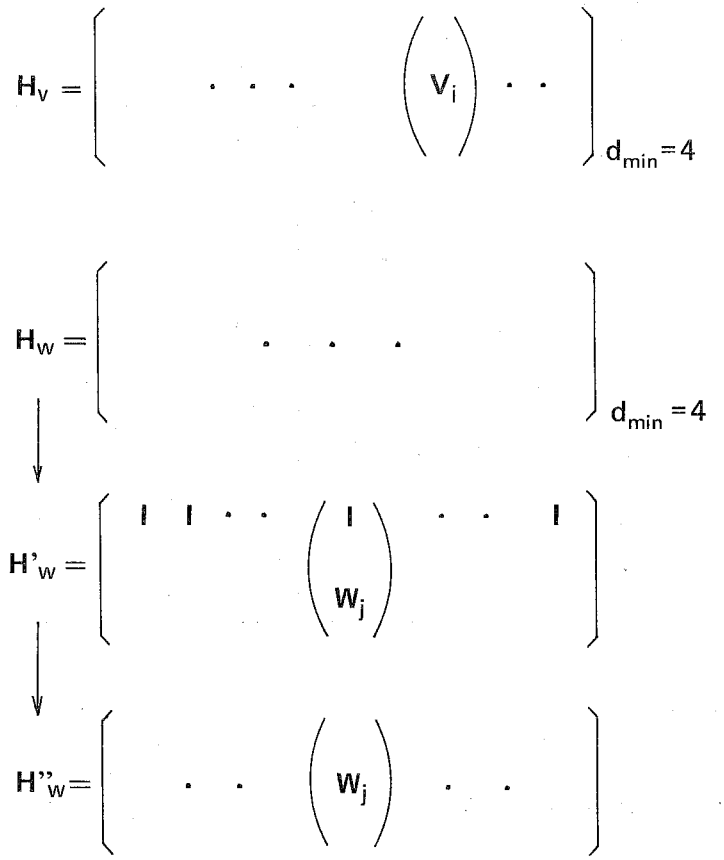


図4－14　Hv ，Hw ，H'w ，H''w　の関係

（定理4－6）

$C_{i,j}$ を次の（4－35）式で定義する列ベクトルとする。このとき，$C_{i,j}$（$i=0, 1, \cdots, n-1$，$j=0, 1, \cdots, m-1$）を列ベクトルとして構成するHマトリクスはGF（2^b）上で最小ハミング距離 $d_{min}=4$ を有する符号のHマトリクスである。

$$C_{i,j} = \begin{bmatrix} V_i \\ W_j \end{bmatrix} \qquad (4-35)$$

また，この符号の最大符号長は，m・nである。 □

証明は長くなるので，付録3に示す。

この定理で示す符号は，従来のReed－Solomon符号とは異なり，これらの既存の符号を使

って新しく構成した符号と言うことができる。具体例を示そう。次の符号はｂ＝２とする修正Reed－Solomon（１２，６）Ｓ２EC－D２ED符号である。これをH_Wとする。

$$H_W = \left[\begin{array}{ccc|ccc} I & I & I & I & 0 & 0 \\ I & T^2 & T & 0 & I & 0 \\ I & T & T^2 & 0 & 0 & I \end{array}\right] \qquad (4-36)$$

このＨマトリクスに対して，第２行目にＴを掛けこれを新しく第２行目とし，これと第３行目を加え，さらに第１行目と加えることにより，第１行目がすべて非零元となる次のマトリクスを得る。

$$\left[\begin{array}{cccccc} T & T & I & I & T & I \\ T & I & T^2 & 0 & T & 0 \\ I & T & T^2 & 0 & 0 & I \end{array}\right] \qquad (4-37)$$

第１行目をすべて"Ｉ"とするために各列を規格化する。結果のマトリクスをH'_Wとする。

$$H'_W = \left[\begin{array}{cccccc} I & I & I & I & I & I \\ I & T^2 & T^2 & 0 & I & 0 \\ T^2 & I & T^2 & 0 & 0 & I \end{array}\right] \qquad (4-38)$$

以上の操作を加えても，マトリクスとしての機能は変化しないことは明らかである。（４－３８）式からH''_Wとして次のようになる。

$$H''_W = \left[\begin{array}{cccccc} I & T^2 & T^2 & 0 & I & 0 \\ T^2 & I & T^2 & 0 & 0 & I \end{array}\right] \qquad (4-39)$$

ここで，H_Vとして（４－３６）式に示すH_Wと同一のマトリクスを採用する。これから，（４－３５）式に定義した$C_{i,j}$を作成することができ，ｒ＝５行を有するＳ２EC－D２ED 符号が構成できる。これを図４－１５に示す。

同様の方法でｒ＝７行を有する符号を構成することができる。すなわち，上記の手法で得られた５行構成の符号をH_Vとし，別の３行構成の符号H_Wとから，同様の手法により７行構成の符号を得る。以上の手法を拡張させて，一般にその行数が奇数（ｒが奇数）となるＳｂEC－

DbED符号を構成することができる。

偶数行の符号に対しては，H'_W として次のマトリクスを適用する。

$$H'_W = \begin{bmatrix} I & I \\ O & I \end{bmatrix} \qquad (4-40)$$

例として，図4－15に示した5行構成の符号と（4－40）式に示すH'_Wとを用いて，6行構成のS2EC−D2ED符号を図4－16（a）に示す。このHマトリクスでは検査バイト位置が不明確である。そこで，このマトリクス中，第4行，5行，6行目をそれぞれ第1行目に加えることにより，列中，唯一"I"要素の現われる検査バイトを得ることができる。これを図4－16（b）に示す。図中，この検査バイト位置を示すためIを丸で囲って示している。

以上の符号構成法から明らかなように，本符号の符号長は次の定理で示すことができる。

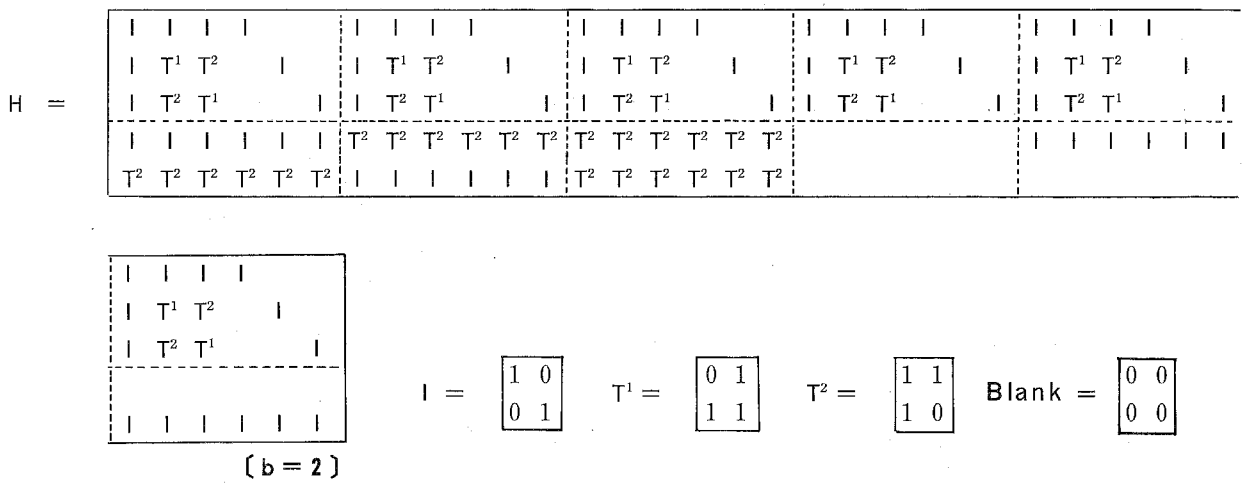


図4－15 (72, 62) S2EC－D2ED符号

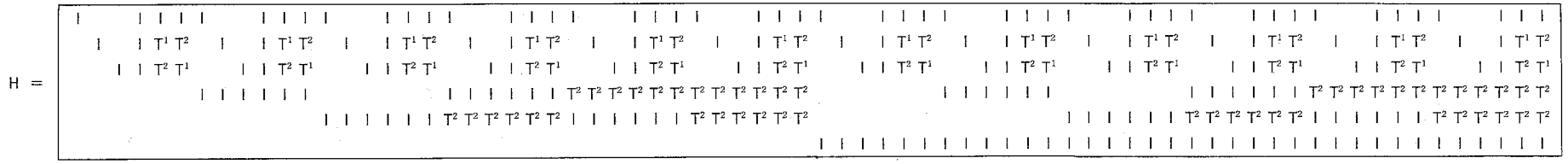


図４－16(ａ)　(144, 132) Ｓ２ＥＣ－Ｄ２ＥＤ符号

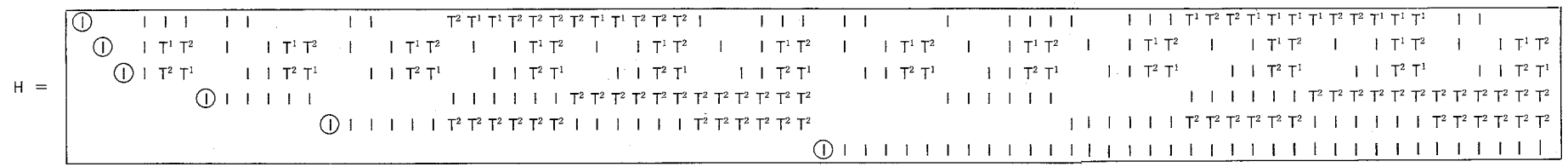


図４－16(ｂ)　(144, 132) Ｓ２ＥＣ－Ｄ２ＥＤ符号

（定理４－７）

H_V，H_W および（４－４０）式に示すマトリクスを使用して，定理４－６に定義する符号を構成したとき，最大符号長は，次式で示すことができる。

$$\begin{aligned} &r = 3 \text{以上の奇数のとき} \quad : \quad N = b\left(2^b + 2\right)^{\frac{r-1}{2}} \\ &r = 4 \text{以上の偶数のとき} \quad : \quad N = 2b\left(2^b + 2\right)^{\frac{r-2}{2}} \end{aligned} \qquad (4-41)$$

□

ここで，ｒは検査バイト数である。

ｂ，ｒに対するＮの関係を表４－６に示す。ｂ＝１の場合は，ＳＥＣ－ＤＥＤ 符号の場合の最大符号長に一致する。図４－１７に情報ビット数（Ｎ－ｂ・ｒ）に対する検査ビット数Ｒ＝ｂ・ｒ の関係を示す。これから，情報ビット数６４ビット，１２８ビットに対しては，いずれもｂ＝２ビットで検査ビット数は１２ビット，ｂ＝４ビットでは検査ビット数は１６ビットとなる。特に，情報ビット数１２８ビットに対してｂ＝４ビットのときは最も能率よく，完全符号の形で適用することができる。また，表４－５に示した修正符号２と比較して，符号長は大きく改善されていることがわかる。

表４－６　ＳｂＥＣ－ＤｂＥＤ符号の符号長Ｎ

ｒ：行数*	ｂ：バイト長（ bit ）						
	1**	2	3	4	5	6	8
3	4	12	30	72	170	396	2064
4	8	24	60	144	340	792	4128
5	16	72	300	1296	5780	26136	532512
6	32	144	600	2592	11560	52272	1065024
7	64	432	3000	23328	196520	1724976	
8	128	864	6000	46656			

* 検査ビット数＝ｒ・ｂ　　（空欄は未計算部）

** ＳＥＣ－ＤＥＤ符号に一致

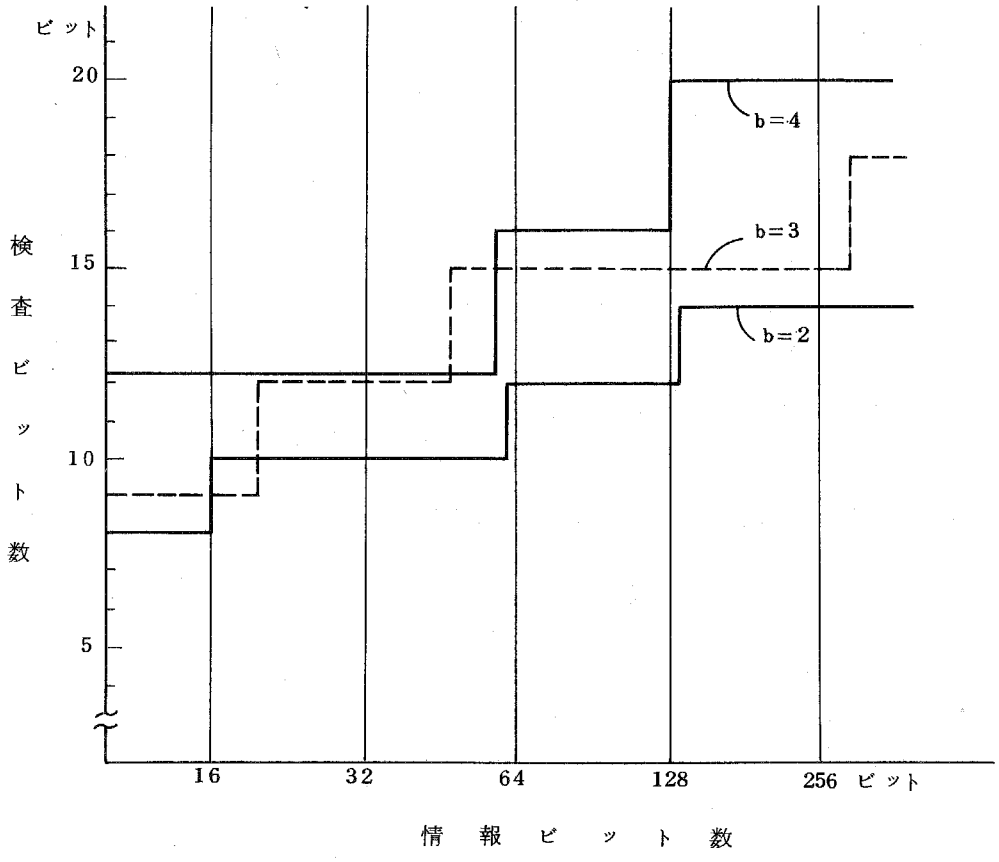


図4－17　ＳｂＥＣ－ＤｂＥＤ符号の情報ビット数に対する検査ビット数の関係

〔2〕　復号法

本提案のＳｂＥＣ－ＤｂＥＤ符号に対する単一バイト誤り訂正の手法は，前節でのべたＳｂＥＣ符号の場合と同一である。すなわち，誤りバイト位置を示す誤りバイトポインタG_jと誤りパターンE_jを求めることにより，具体的な誤り位置を指摘することができる。一般に，ＳｂＥＣ－ＤｂＥＤ符号のＨマトリクスを図４－３に示す形とするとき，誤りバイトポインタは（4－21）式から求められる。また，誤りパターンE_jは次の関係から求めることができる。

$$S_i = h_{i,j} \cdot E_j \quad \rightarrow \quad E_j = h_{i,j}^{-1} \cdot S_i \qquad (4-42)$$

$$i = 0,\ 1,\ \cdots,\ r-1$$

訂正データは，この誤りパターンE_jを用いて（4－22）式と同様にして得ることができる。

次に二重バイト誤り検出（ＤｂＥＤ）の論理は次のようになる。すなわち，非零のシンドロー

ム（誤り検出）に対して，すべての誤りバイトポインタが零であるとき二重バイト誤りと判定する。このときは，二重バイト誤り以外の多重バイト誤りも含まれている。このための論理は次のように表わすことができる。

$$\left[\bigcup_{i=0}^{r-1}(S_i \neq 0)\right]\cap\left[\bigcup_{j=0}^{n-1}(G_j = 0)\right] = 1 \qquad (4-43)$$

$\cup$：論理和，$\cap$：論理積

具体的な回路ゲート量，回路遅延等は第 7 章で示す。

4—4 結　　言

本章では，単一バイト誤り訂正能力を有するＳｂＥＣ符号，およびさらに二重のバイト 誤り検出能力を有するＳｂＥＣ−ＤｂＥＤ符号について新しい符号構成を示した。 主な結論は以下の通りである。

(1) 奇数重み列の性質を適用した新しいＳｂＥＣ符号の構成を示した。この符号は，特にｂ＝1とすると従来最も誤り検出能力の点で優れている奇数重み列ＳＥＣ−ＤＥＤ符号に一致する。また，２バイト間にわたる誤りが同一パターンであるとき，すべて検出できる特長を有し，これら２バイト間誤り検出能力が他の奇数重み列の性質を有さない符号と比較して５〜１５％程度高い。

(2) 奇数重み列の性質を適用することによる符号長の短縮は，概略 $1-1/2^b$ となり，これからｂ≧４では短縮は非常に小さい。

(3) 符号間最小距離４を有するReed−Solomon ＳｂＥＣ−ＤｂＥＤ符号は，ｂ＝2〜4 の小さい値に対して，情報ビット数は高々６０ビットまでしか成立せず，情報ビット数６４ビット，１２８ビットを有する装置に適用できない。

(4) 既存の２個のＳｂＥＣ−ＤｂＥＤ符号を組合わせて，新しいＳｂＥＣ−ＤｂＥＤ符号を構成した。この符号は，任意のバイト長，任意の情報ビット数に対して構成が可能である。ｂ＝２ビットで，情報ビット数６４ビット，１２８ビットに対して，いずれも検査ビット数は１２ビット，ｂ＝４ビットでいずれも１６ビットである。

第5章　巡回性符号

5−1 緒　　言

高速な復号が要求される誤り検出・訂正符号の符号化・復号化回路は，並列な復号化手法を採用しており，そのため符号長の増大，機能の向上により回路ゲート量は3,000ゲート程度にもなる。この回路は，一般にくり返し性の性質を有していない。そのため，この回路にくり返し性をもたせ，ＬＳＩ化が実現できれば，回路の経済化，高速化，高信頼化が達成できる。

このような観点から，本章ではこの回路にくり返し性を与える巡回性符号を新しく提案する。また，前章までに示した各種のバイト系符号に巡回性の手法を適用した巡回性符号について，その符号構成法，符号長等について示す。その結果，巡回性の手法がもとの符号に対して，ほとんど符号長を短縮させることなく構成できる実用的な手法であることを示す。

以下に，まず巡回性符号の定義とその概念を示し，次に各バイト系符号に対して巡回性符号としての構成，符号長等についてのべる。

5－2　巡回性符号の概念

定義２－１で巡回性符号を定義した。この符号構成に対する具体的な概念を与えるため，簡単な機能を有する符号を巡回性符号として構成し，これによりその符号化・復号化回路がくり返し性をもつことを示す。

１ビット誤り訂正（ＳＥＣ）の機能を有する符号に対して具体的に構成してみよう。１ビット誤り訂正・２ビット誤り検出（ＳＥＣ－ＤＥＤ）の機能を有する符号に対しては，文献（７６）にその符号構成とその一部の回路（チェックビット生成／シンドローム生成回路）に対するＬＳＩ構成が示されている。ＳＥＣ符号のＨマトリクスは，｛０，１｝で表わした列ベクトルが互いに線形独立であればよい。従って，これを巡回性符号とするためには，そのＨマトリクスは線形独立の条件を満たしつつ，（２－７）式の構成を満足しなければならない。例えば４行構成からなる最大符号長を有する巡回性ＳＥＣ符号は，次のように示すことができる。

$$H=\left[\begin{array}{ccc:ccc:ccc:ccc} c_0 & d_0 & d_1 & c_1 & d_2 & d_3 & c_2 & d_4 & d_5 & c_3 & d_6 & d_7 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \end{array}\right] \qquad (5-1)$$

$$\underbrace{\quad}_{H_0} \quad \underbrace{\quad}_{H_1} \quad \underbrace{\quad}_{H_2} \quad \underbrace{\quad}_{H_3}$$

この符号は，情報ビット数はd_0～d_7の８ビットであり，検査ビット数はc_0～c_3の4ビットである。H_1はH_0の各行ベクトルを１行下へ巡回的にシフトした関係にあり，H_2はH_1に対して，H_3はH_2に対して同様の関係にある。H_3をさらに１行巡回シフトすればH_0に一致する。

この符号の最大符号ビット長Ｎは，一般に次式により表わすことができる。[4]

$$N=\sum_{d\mid r}\mu(d)\cdot(2^{r/d}-1) \qquad (5-2)$$

ここで，$\sum_{d\mid r}$はｒの約数ｄに対するすべての和を示し，$\mu(d)$はメビウス（Möbius）関数[1][4]である。表５－１に符号ビット長Ｎと検査ビット数ｒの関係を示す。参考として，巡回性の性質を持たない通常のＳＥＣ符号の符号ビット長も示している。

次に，この符号を使用して具体的な符号化・復号化回路を構成してみよう。この回路のブロ

ック図を図５－１に示す。CG／SGは検査ビットの生成（CG）およびシンドロームの生成（SG）を行う回路であり，信号Bにより制御される。すなわち，符号化時にはB＝0，復号化時にはB＝1である。SDは，SGより生成されたシンドロームから具体的に誤りビット位置を指摘するシンドロームデコード回路である。この回路により指摘された誤りビットは，CORで示される反転回路にて反転され訂正される。

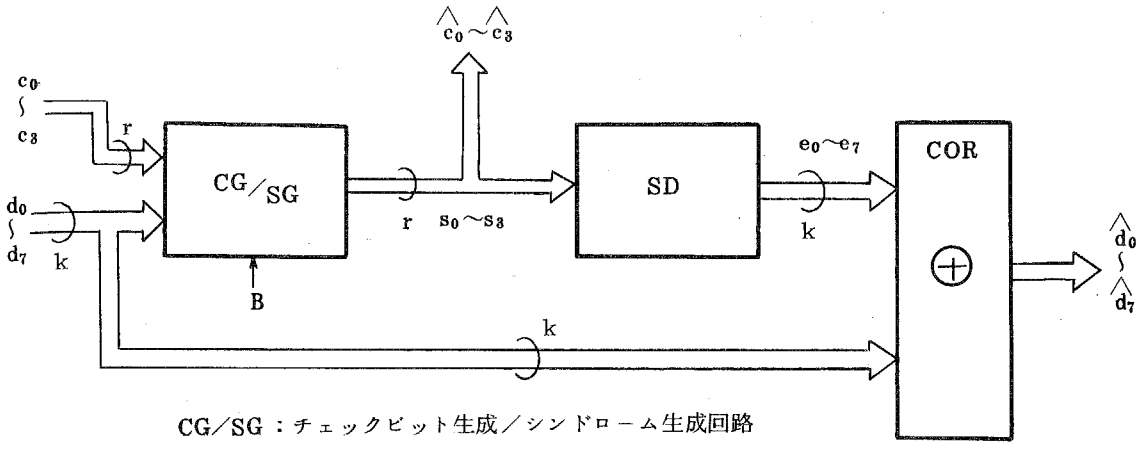
図５－２はCG／SGの回路を示し，それぞれ部分HマトリクスH_0～H_3に対応して，回路GEN_0～GEN_3が構成できる。また，GEN_0～GEN_3はすべて同一回路であり，入出力間を接続することにより，各々から検査ビット（$\hat{c}_0$～$\hat{c}_3$）またはシンドローム（s_0～s_3）が生成される。検査ビットの生成とシンドロームの生成は，前述の通り信号Bにより制御される。

図５－３はSDの回路であり，部分マトリクスH_0～H_3に対応して同一回路DEC_0～DEC_3により構成される。DEC_0～DEC_3への入力s_0～s_3が各々の回路においてシフトした関係で入力されている点に注意する必要がある。

以上の回路とCORを結合させて全体の符号化・復号化回路は，各部分Hマトリクス対応に同一回路４個で構成できる。図５－４は，このうちの１回路を示すものであり，入力９本，出力５本，計１４本の入出力信号線，約１６ゲートの回路となることを示している。

表５－１　巡回性ＳＥＣ符号の符号ビット長

検査ビット数 r	巡回性ＳＥＣ符号の符号ビット長	ＳＥＣ符号の符号ビット長
3	6	7
4	12	15
5	30	31
6	54	63
7	126	127
8	240	255
9	504	511
10	990	1023
11	2046	2047
12	4020	4095



CG／SG ：チェックビット生成／シンドローム生成回路
SD ：シンドロームデコード回路
COR ：情報反転（訂正）回路

図５－１ （ｋ＋ｒ，ｒ）符号に対する並列符号化・復号化回路

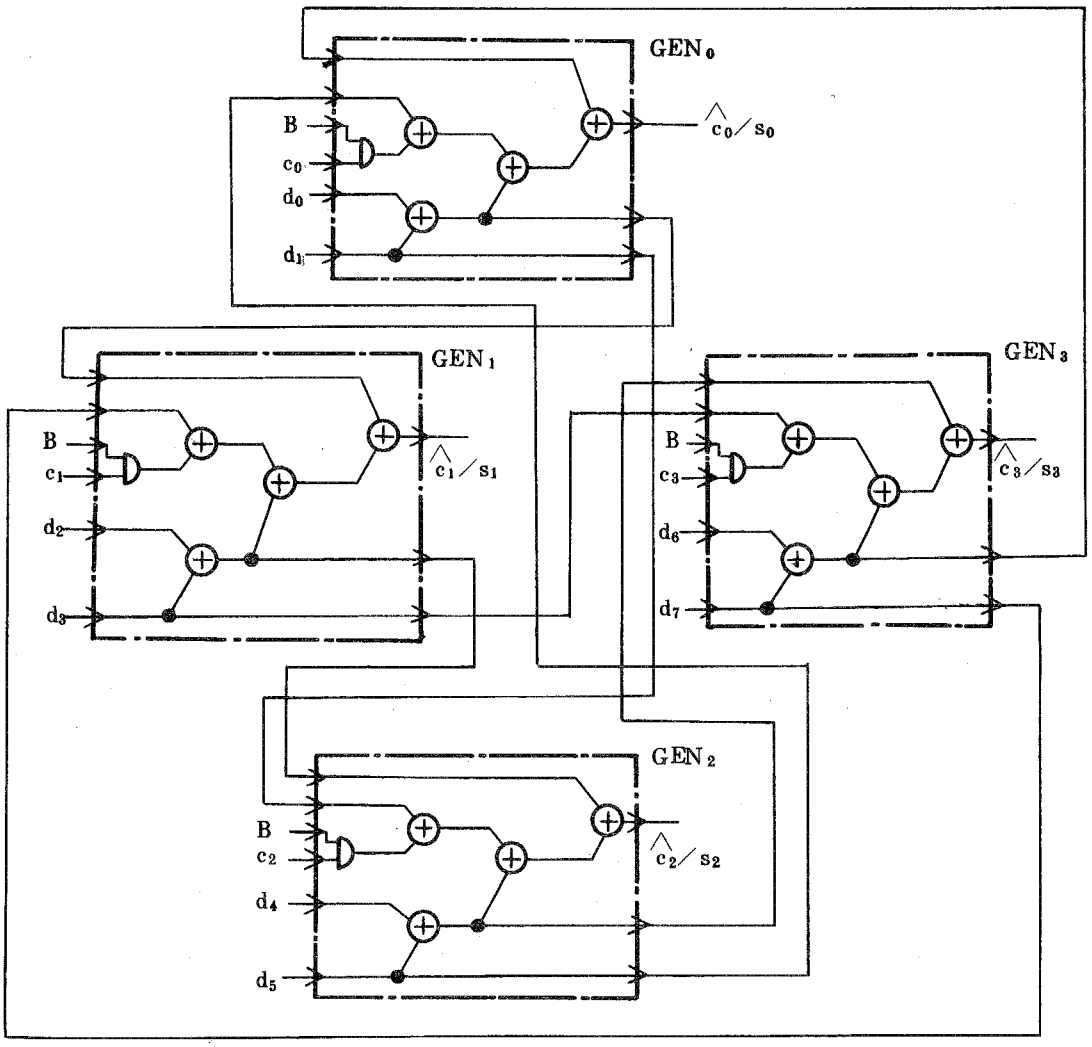


図５－２ 検査ビット生成／シンドローム生成回路（CG／SG）

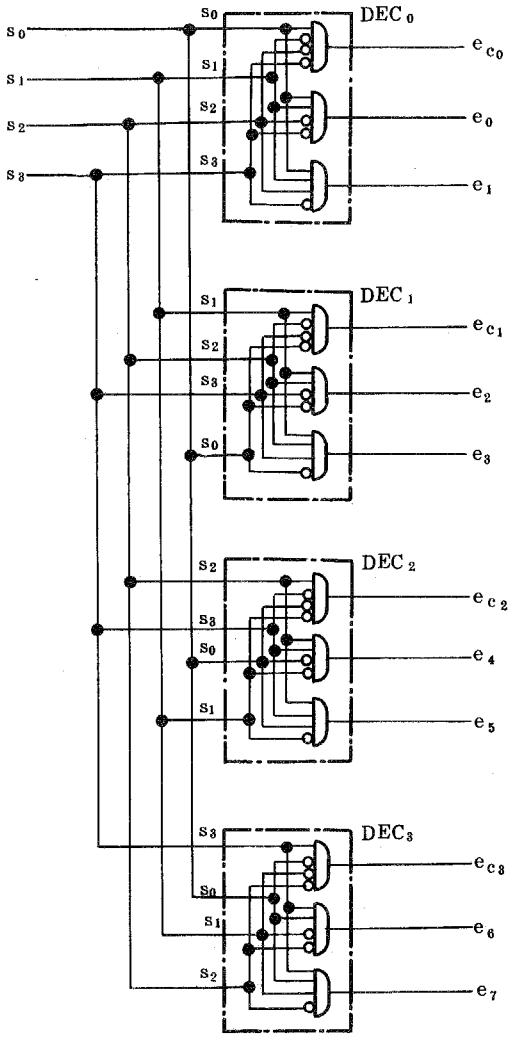
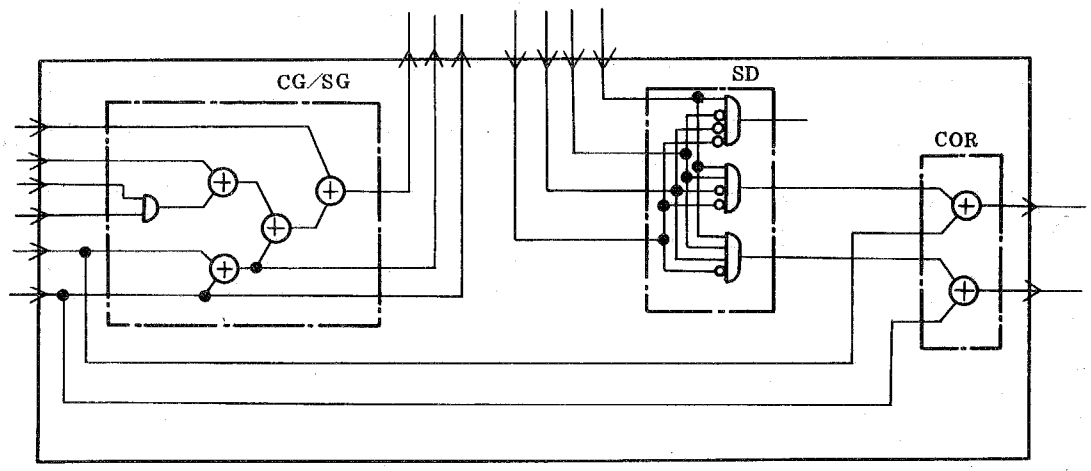


図５－３　シンドロームデコード回路（ SD ）



この回路４個にて全体の符号化・復号化回路が構成できる。

図５－４　１ＬＳＩ 回路構成

5－3　巡回性 SEC-SbED／巡回性 SEC-DED-SbED 符号

ＳＥＣ－ＳｂＥＤまたはＳＥＣ－ＤＥＤ－ＳｂＥＤの機能を有する符号は，D.C.Bossen, L.C. Chang, C.L.Chen[45]，S.M.Reddy[46] および第３章で示した構成Ａ，構成Ｂの符号がある。一方，A.S.Liu, M.Arbab は，（４０，３２）ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号を示した。[77] この符号を図５－５に示す。この符号は次に示す特徴を有する。

(1)　８ビット単位に，すべて'1'の行を４個有する。これにより，８ビット単位のパリティ符号との変換が容易となる。

(2)　各列ベクトルはすべて重み３，検査部の列ベクトルは重み１を有する。

(3)　情報ビット部の4ビットの内３ビット誤りは必ず重み５を有し，１ビット誤りと誤訂正することはない。

(4)　検査ビット部の4ビットの内３ビット誤りは重み３となるが，上位または下位４ビットのシンドロームはすべて'0'となり，Ｈマトリクス各列ベクトルの重み３のパターンとは異なる。

この符号は，定義に示す巡回性符号には一致しないが，近い構成を有している。

ところで，D.C.Bossen らと S.M.Reddy による符号，および構成Ａの符号に対して巡回性の性質を適用することは困難である。構成Ｂの符号はＨマトリクス列ベクトル中の要素の位置を特に規定していないことから，巡回性符号として構成することが可能である。そこで，本節ではこの構成の符号に対して具体的に巡回性符号としての構成法，符号長等を求めることにする。

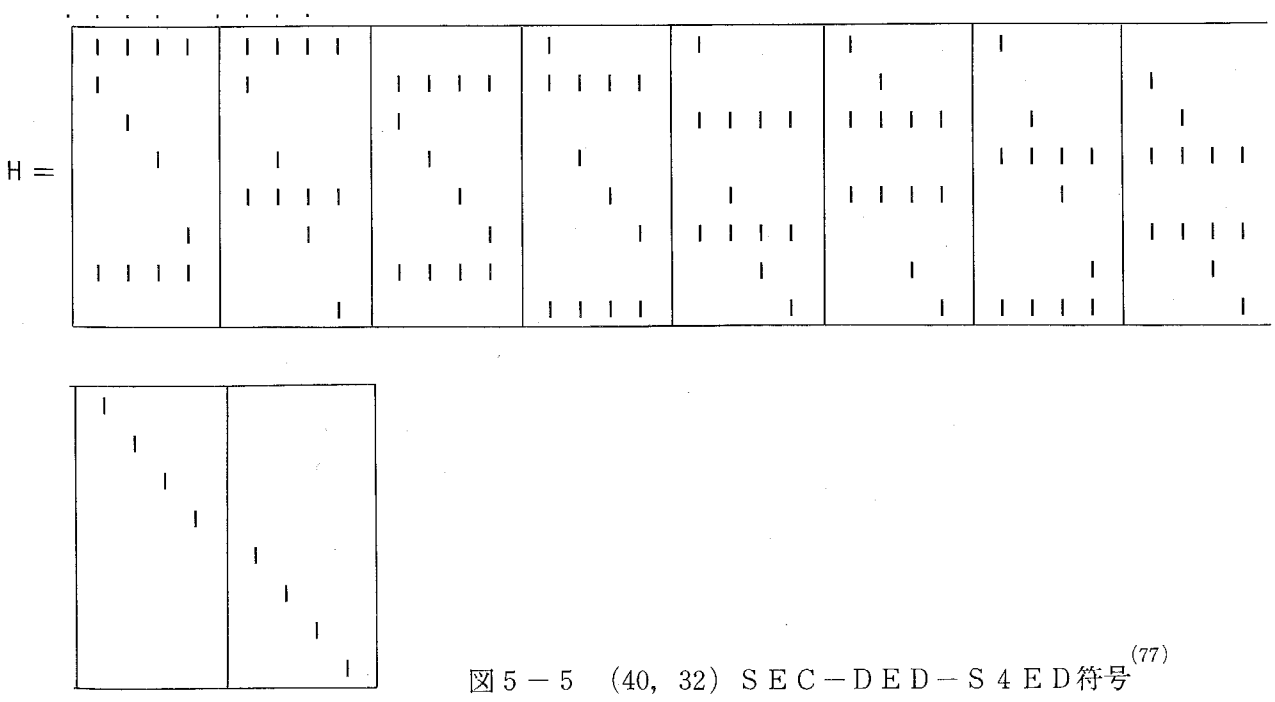


図５－５　(40, 32) ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号[77]

5－3－1　巡回性SEC－SbED符号[25]

構成Bの符号は，I_q，M_u を用い，I_q を少くとも各列中に1個含む構成であった。この考え方は，I_q を各列ベクトル中のどの位置に置いてもよいことを意味する。例えばr＝2の2段構成の場合，上段と下段で入れかえた形で構成でき，特に検査部はこの考え方に従っている。例としてr＝2，b＝3については，すべての列ベクトルの組合せは，図3－5に示すごとく13個存在する。このうち $\begin{bmatrix} I_0 \\ M_6 \end{bmatrix}\begin{bmatrix} I_0 \\ M_5 \end{bmatrix}\begin{bmatrix} I_0 \\ M_3 \end{bmatrix}\begin{bmatrix} I_0 \\ M_7 \end{bmatrix}\begin{bmatrix} I_0 \\ M_0 \end{bmatrix}$ については，上段，下段を置換した $\begin{bmatrix} M_6 \\ I_0 \end{bmatrix}\begin{bmatrix} M_5 \\ I_0 \end{bmatrix}$ $\begin{bmatrix} M_3 \\ I_0 \end{bmatrix}\begin{bmatrix} M_7 \\ I_0 \end{bmatrix}\begin{bmatrix} M_0 \\ I_0 \end{bmatrix}$ が存在し線形独立性を満している。しかし，$\begin{bmatrix} I_0 \\ I_0 \end{bmatrix}$ は上，下置換すると自分自身に一致し，また $\begin{bmatrix} I_0 \\ I_1 \end{bmatrix}$ を置換して $\begin{bmatrix} I_1 \\ I_0 \end{bmatrix}$ としたものは $\begin{bmatrix} I_0 \\ I_2 \end{bmatrix}$ を，また $\begin{bmatrix} I_0 \\ I_2 \end{bmatrix}$ を置換して $\begin{bmatrix} I_2 \\ I_0 \end{bmatrix}$ としたものは $\begin{bmatrix} I_0 \\ I_1 \end{bmatrix}$ を行方向に1ビット巡回させたものに一致する。以上から，上，下段とも I_q の要素からなるものは，必ずしも巡回性符号の列ベクトルとすることはできない。この場合には，$\begin{bmatrix} I_0 \\ I_1 \end{bmatrix}\begin{bmatrix} I_1 \\ I_0 \end{bmatrix}$ または $\begin{bmatrix} I_0 \\ I_2 \end{bmatrix}\begin{bmatrix} I_2 \\ I_0 \end{bmatrix}$ のいずれかを選ぶことになる。これから，図5－6に示すr＝2の巡回性（36，30）SEC－S3ED符号を構成することができる。同様にして，r＝3とする巡回性（54，48）SEC－S2ED符号を図5－7に示す。

$$H = \left[\begin{array}{cccccc|cccccc} I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & M_6 & M_5 & M_3 & M_7 & I_1 & M_0 \\ M_6 & M_5 & M_3 & M_7 & I_1 & M_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 \end{array}\right]$$

$$= \left[\begin{array}{cccccccccc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot \\ \cdot\cdot\cdot & \cdot\cdot\cdot & & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot & & \cdot & \cdot & \cdot & \cdot \end{array}\right]$$

$$\left[\begin{array}{cc} \cdot\cdot\cdot & \cdot \\ \cdot\cdot\cdot & \cdot \\ \cdot\cdot\cdot & \cdot \end{array}\right] \quad \left[\begin{array}{cc} \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \end{array}\right]$$

図５－６　巡回性（36，30）ＳＥＣ－Ｓ３ＥＤ符号

$$H = \left[\begin{array}{ccccccccc|ccccccccc|ccccccccc} I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & M_0 & M_0 & M_0 & M_3 & M_3 & M_3 & M_3 & M_2 & M_0 & M_1 & M_2 & M_3 & M_0 & M_1 & M_2 & M_3 & M_1 & M_0 \\ M_1 & M_2 & M_3 & M_0 & M_1 & M_2 & M_3 & M_1 & M_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & M_0 & M_0 & M_0 & M_3 & M_3 & M_3 & M_3 & M_2 & M_0 \\ M_0 & M_0 & M_0 & M_3 & M_3 & M_3 & M_3 & M_2 & M_0 & M_1 & M_2 & M_3 & M_0 & M_1 & M_2 & M_3 & M_1 & M_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 & I_0 \end{array}\right]$$

H_0　H_1　H_2

$$H_0 = \left[\begin{array}{ccccccccc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & & & \cdot\cdot & \cdot\cdot & & \\ & & & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \end{array}\right]$$

図５－７　（54，48）ＳＥＣ－Ｓ２ＥＤ符号

任意のｒに対して同様の考え方でｒの巡回度を有するＳＥＣ－ＳｂＥＤ符号を構成することができる。この最大符号ビット長Ｎは次の定理で与えることができる。

（定理５－１）

巡回性ＳＥＣ－ＳｂＥＤ符号の最大符号ビット長Ｎは次式で表わすことができる。

$$N = n \cdot b = \sum_{d \mid r} \mu(d) \{ (2^b)^{r/d} - (2^b - b)^{r/d} \} G.C.D.(d, b) \qquad (5-3)$$

ここで，G.C.D.*（ｄ，ｂ）はｄとｂの最大公約数である。 □

この定理の証明は複雑であるので省略する。表５－２にｂ，ｒに対するＮの関係を示す。これから，ｂ＝１の場合には表５－１に示す巡回性ＳＥＣ符号に一致する。図５－８に情報ビット数に対する検査ビット数の関係を示す。表３－４，または図３－７と比較して符号長の短縮度は小さいことがわかる。

表５－２　巡回性 SEC-SbED符号の符号長Ｎ

r* \ b	1**	2	3	4	6	8
2	2	8	36	104	720	4016
3	6	54	378	2364	67014	1524216
4	12	216	3432	44576	5459256	
5	30	990	29640	799740		
6	54	3912	246024			
7	126	16254				
8	240					

＊検査ビット数＝ｒ・ｂ　　（空欄は未計算部）

＊＊巡回性ＳＥＣ符号に一致

＊　G.C.D.；Greatest Common Divisor

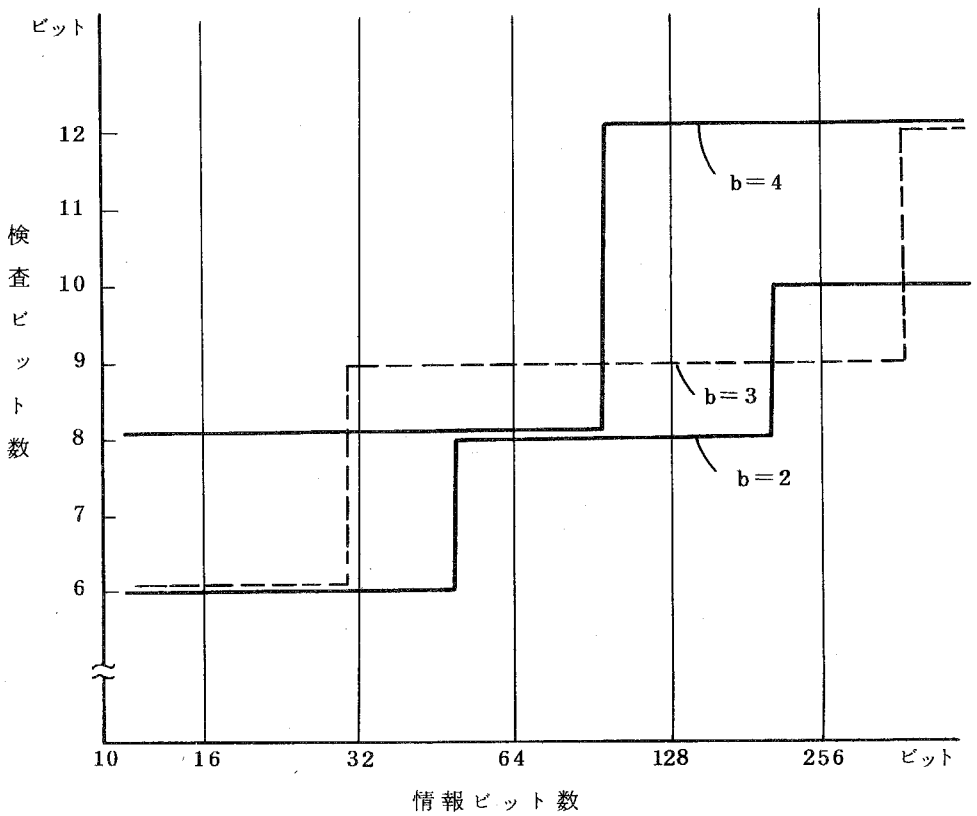
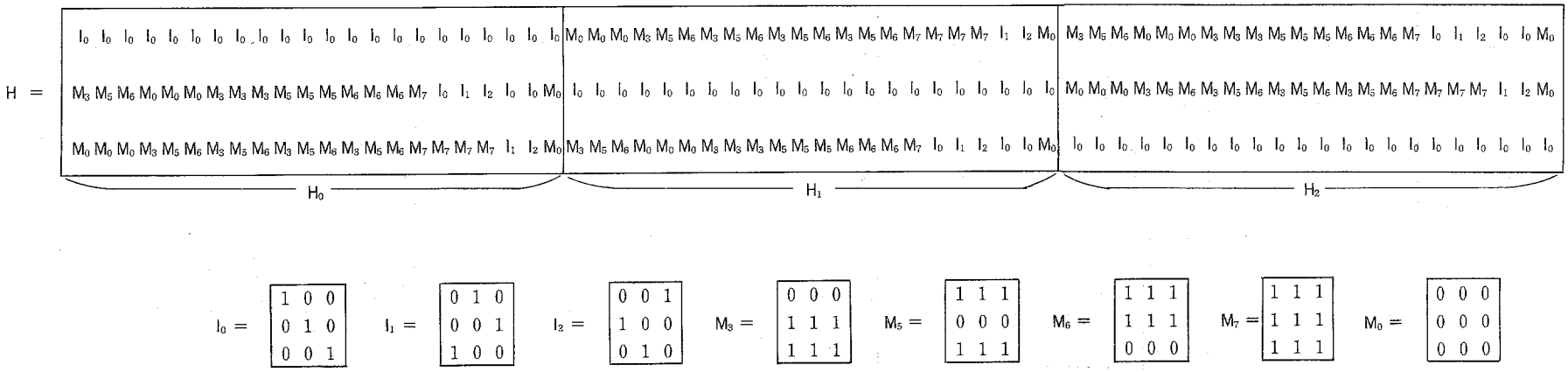


図５－８　巡回性ＳＥＣ－ＳｂＥＤ符号における情報ビット数と検査ビット数の関係

５－３－２　巡回性ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号[25]

巡回性を有するＳＥＣ－ＤＥＤ－ＳｂＥＤ符号についても，３－３－２で示した構成ＢのＳＥＣ－ＤＥＤ－ＳｂＥＤ符号から列を選び出して構成することができる。図３－１４に示したｂ＝３，ｒ＝３の（２０７，１９８）ＳＥＣ－ＤＥＤ－Ｓ３ＥＤ符号をもとにして，巡回度３の巡回性（１９８，１８９）ＳＥＣ－ＤＥＤ－Ｓ３ＥＤ符号を図５－９に示す。この符号は，図３－１４に示したＨマトリクス中，＊印を付した列ベクトルを除いて構成したものである。＊印の列ベクトルは巡回させて他の列ベクトルと一致することから除かれねばならない。このようにして構成した巡回性符号の一般の符号長は，次の定理により与えることができる。



$I_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $I_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$ $I_2 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ $M_3 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ $M_5 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$ $M_6 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $M_7 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ $M_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

図５－９　巡回性（198, 189）ＳＥＣ－ＤＥＤ－Ｓ３ＥＤ符号

（定理５－２）

巡回性ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号の最大符号長Ｎは，次式で表わすことができる。

$$N = n \cdot b = \frac{1}{2} \sum_{\substack{d \mid r \\ d:odd}} \mu(d) \{ (2^b)^{r/d} + b^{r/d} - (2^b - b)^{r/d} \} G.C.D.(d, b) \qquad (5-4)$$

□

この定理の証明は定理５－１と同様複雑であるので省略する。表５－３にｂ，ｒに対するＮの関係を示す。ｂ＝１または２のとき，巡回性ＳＥＣ－ＤＥＤ符号に一致する。図５－１０に情報ビット数に対する検査ビット数の関係を示す。表３－６，図３－１２と比較して，符号長の短縮度は小さいことがわかる。

ここで，巡回性ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号として実用的な（４０，３２）ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号を示そう。これを図5－11に示す。この符号構成は各行ベクトルの重みがすべて一定であり，しかもＨマトリクスの重みとして最小となるように考慮している。また，８ビット単位にすべて'1'を持つ構成を有し，８ビット単位のパリティ符号との符号変換が容易な構成となっている。図５－１１に示す符号は，巡回性符号として部分マトリクスH_0，H_1から構成される。図５－１２は符号化・復号化回路のうちＣＧ／ＳＧの回路を示し，図５－１３はＳＤの回路を示す。全体の回路は図５－１４に示す構成となる。これから，図５－１１に示すH_0，H_1に対応して，全体の回路は図５－１４の破線で示すごとく２個の同一回路で構成でき，ＬＳＩ化に適する構成を与えることができる。この回路は１５０ゲート，５８入出力信号ピンのＬＳＩ２個で構成することができる。

表５－３　巡回性ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号の符号長Ｎ

r* \ b	1**	2	3	4	6	8
2	2	8	24	64	384	2048
3	3	30	198	1212	33606	762360
4	8	128	1776	22528	2731008	
5	15	510	14940	400380		
6	30	2040	123552			
7	63	8190				
8	128					

＊ 検査ビット数＝ｒ・ｂ　　（空欄は未計算部）

＊＊ 巡回性ＳＥＣ－ＤＥＤ符号に一致

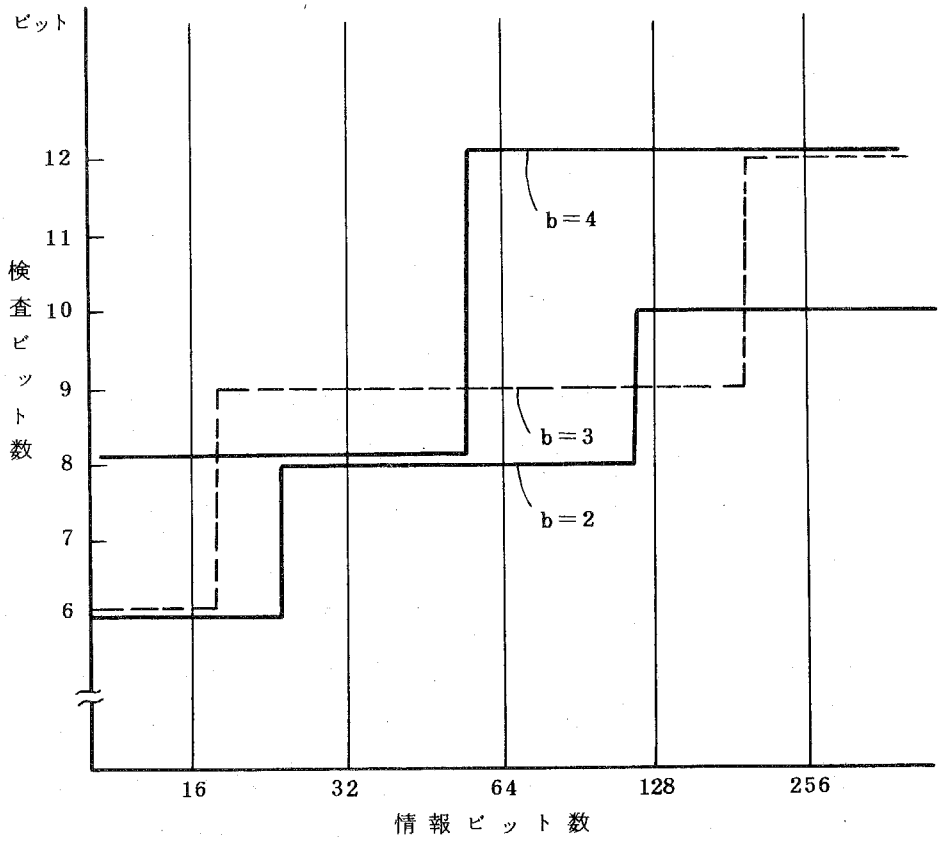


図5－10　巡回性SEC－DED－SbED符号における情報ビット数と検査ビット数の関係

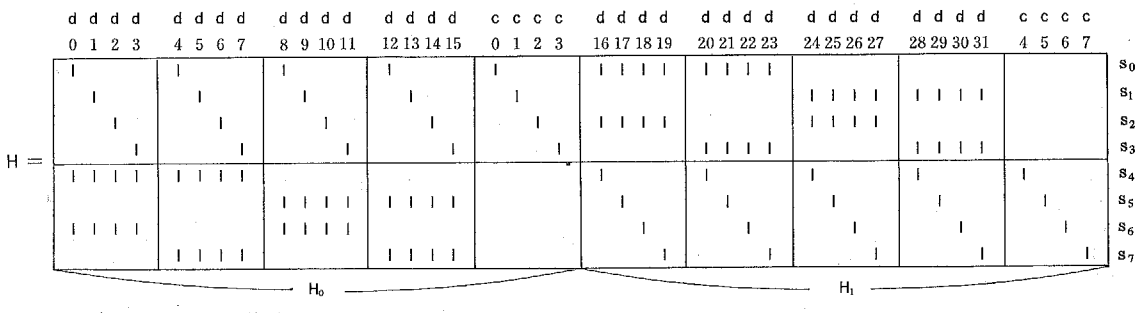


図5－11　巡回性（40，32）SEC－DED－S4ED符号

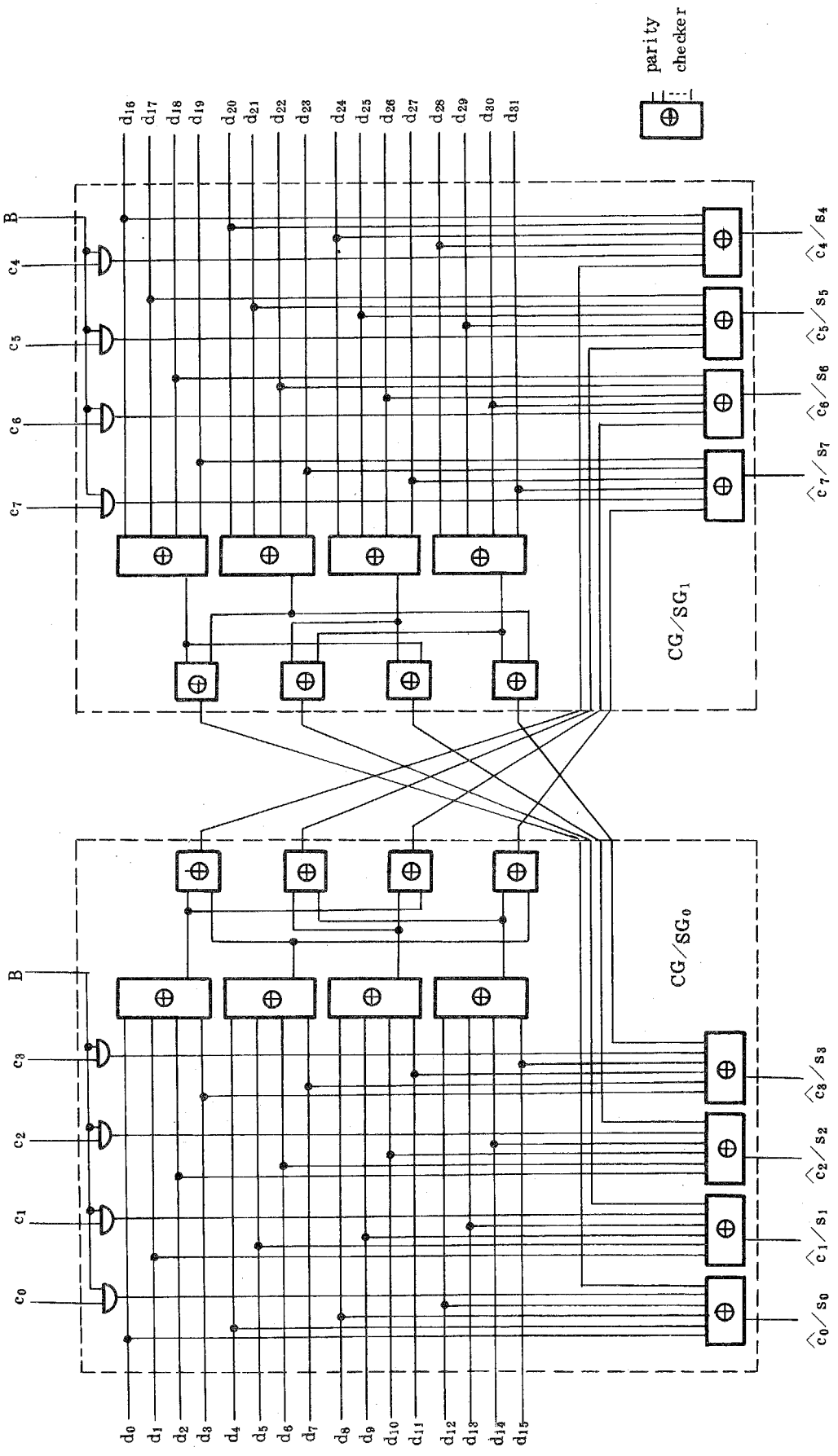


図 5－12　検査ビット生成／シンドローム生成回路（CG／SG）

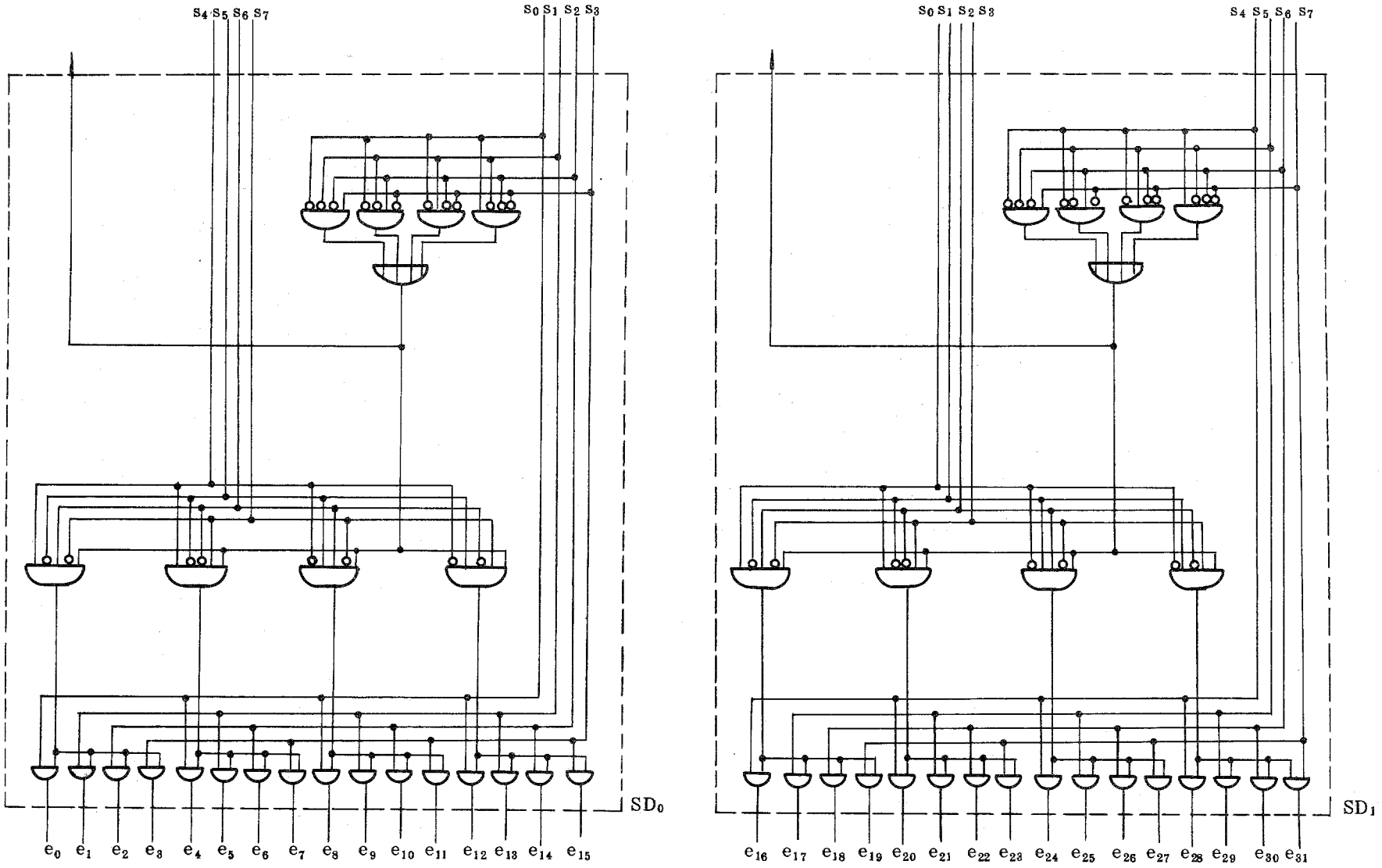


図5－13　シンドローム デコード回路（SD）

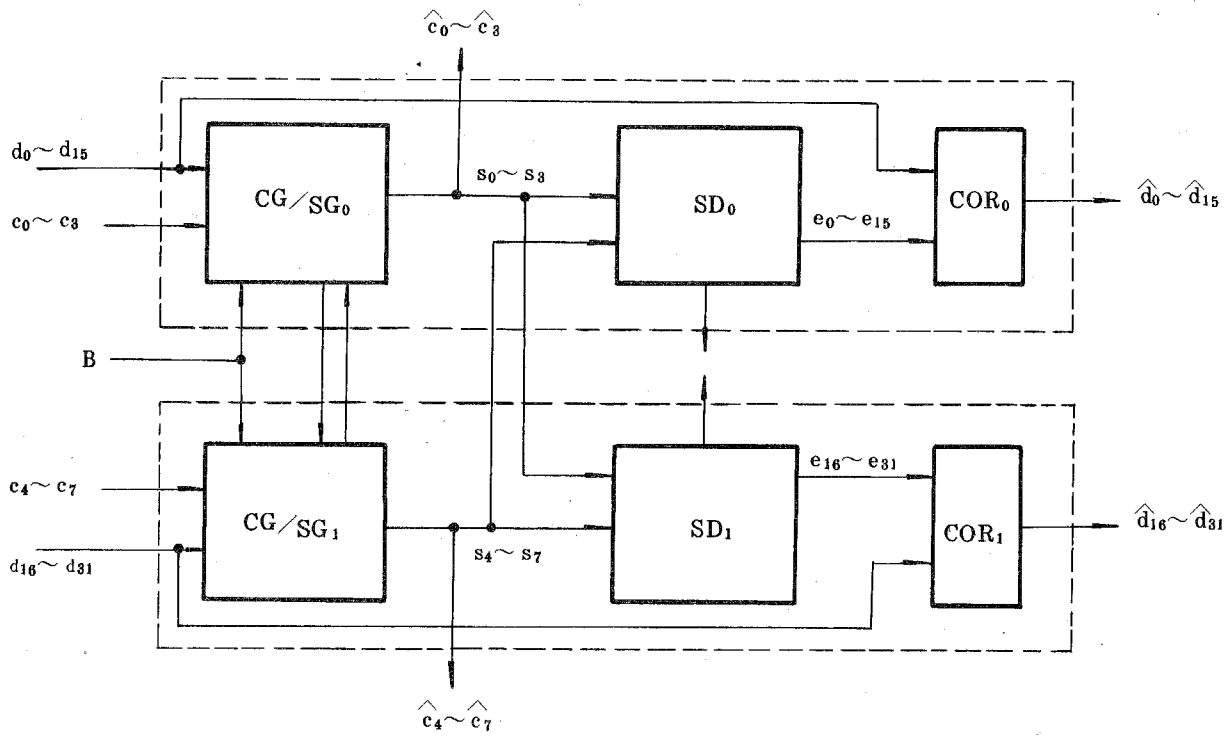


図5－14　ＬＳＩ化に適する全体の符号化・復号化回路
（図5－11の符号の場合）

5－4　巡回性奇数重み列 SbEC 符号[21][22]

5－4－1　巡回性ＳｂＥＣ符号

ＳｂＥＣ符号は，前章で示した通り $q=2^b$ とした単一誤り訂正 q 元線形符号である。この符号に（2－7）式に示す巡回性の構造を与えるためには，ＧＦ（2^b）上の元で構成したＨマトリクス列ベクトルが線形独立の性質を保ちつつ，巡回性の条件を満足する列ベクトルを選択しなければならない。この符号の最大符号ビット長Ｎは，今井，上柳により次の定理により与えられている。[44]

（定理５－３）

巡回性ＳｂＥＣ符号の最大符号ビット長は次式で表わすことができる。

$$N=\frac{b}{2^b-1}\sum_{d\mid r}\mu(d)\{(2^b)^{r/d}-1\}G.C.D.(d,\ 2^b-1) \qquad (5-5)$$

□

ｂ，ｒに対する符号長ｎ＝Ｎ／ｂの関係を表５－４に示す。ここで，特にｂ＝１とした場合には表５－１に示した巡回性ＳＥＣ符号に一致する。

具体的な巡回性ＳｂＥＣ符号の例を図５－１５に示す。この符号はｒ＝３で３回の巡回性を有するｂ＝４の（１４０，１２８）Ｓ４ＥＣ符号である。この符号をｒ＝４で４回の巡回性を有する符号とすると，図５－１６に示す符号が得られる。この符号は，くり返し性として８のくり返し性を有する（１４４，１２８）Ｓ４ＥＣ符号の例である。

表５－４　巡回性ＳｂＥＣ符号の符号長ｎ*

r* \ b	1**	2	3	4	5	6	7	8
2		4	8	16	32	64	128	256
3	6	18	72	270	1,056	4,158	16,512	65,790
4	12	80	576	4,352	33,792	266,240	2,113,536	
5	30	340	4,680	69,900	1,082,400	17,043,520		
6	54	1,332	37,368	1,118,160				
7	126	5,460	299,586					
8	240	21,760	2,396,160					

（空欄は未計算部）

＊　符号ビット長＝ｎ・ｂ，検査ビット数＝ｒ・ｂ

＊＊　巡回性ＳＥＣ符号に一致

$$H_0 = \left[\begin{array}{cccccccccccc|c} I & I & I & I & I & I & I & I & I & I & I & I & I \\ I & T & T^2 & T^3 & T^4 & T^5 & T^6 & T^7 & T^8 & T^9 & T^{10} & & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 \end{array}\right] \qquad T = \left[\begin{array}{cccc} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

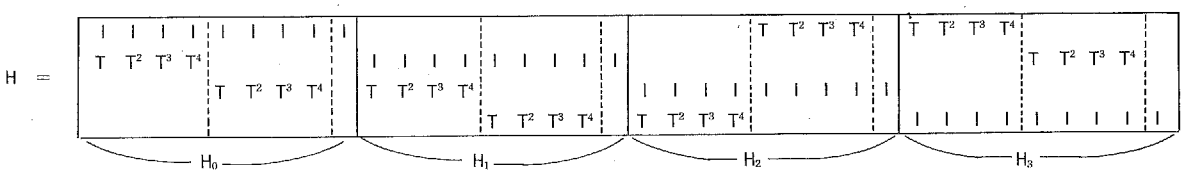
$$H = [H_0 \mid H_1 \mid H_2^{*}]$$

$$H_1 = R \cdot H_0$$
$$H_2 = R^2 \cdot H_0$$

$$R = \left[\begin{array}{ccc} 0 & 0 & I \\ I & 0 & 0 \\ 0 & I & 0 \end{array}\right]$$

＊　H_2 の情報部の最後の１列ベクトルは削除

図５－１５　巡回性（１４０，１２８）Ｓ４ＥＣ符号



空欄は零元０

図５－16　くり返し８を有する巡回性（144，128）Ｓ４ＥＣ符号

5－4－2　巡回性奇数重み列SbEC符号

次に，定義2－2に示した奇数重み列の条件を適用したSbEC符号に対して，巡回性符号を構成してみよう。この符号の符号長は奇数重み列の条件を有し，かつ巡回性の性質を満足する列ベクトルの数に等しい。この時，メビウス（Möbius）の反転公式[4]に修正を加えた次の定理を示す必要がある。

（定理5－4）修正メビウス反転公式[22]

正整数rを変数とする2個の関数$f(r)$と$g(r)$に対し，

$$f(r)=\sum_{(r/d);odd} g(d) \qquad (5-6)$$

の関係が成立するとき，$g(r)$は$f(r)$によって

$$g(r)=\sum_{\substack{d\mid r\\ d;odd}} \mu(d)\cdot f\left(\frac{r}{d}\right) \qquad (5-7)$$

と表わせる。 □

（証明）

（5－6）式はあらゆる正整数に対して成立するため，rの任意の約数dに対し，

$$f\left(\frac{r}{d}\right)=\sum_{(r/d)/c;odd} g(c) \qquad (5-8)$$

が成立する。これを（5－7）式へ代入し，さらに$\underset{d;odd}{d\mid r}$，$(r/d)/c;odd$ を満たすd，c の組合せすべての集合が，$c\mid r$，$\underset{(r/c);odd}{d\mid(r/c)}$ を満たすd，c の組合せすべての集合と一致することから，次式が成立する。

$$\sum_{\substack{d\mid r\\ d;odd}} \mu(d)\cdot f\left(\frac{r}{d}\right)=\sum_{\substack{d\mid r\\ d;odd}} \mu(d)\cdot\sum_{(r/d)/c;odd} g(c) \qquad (5-9)$$

$$= \sum_{c \mid r} g(c) \sum_{\substack{d \mid (r/c) \\ (r/c);odd}} \mu(d)$$

ここで，メビウス関数において，

$$\sum_{d \mid n} \mu(d) = \begin{cases} 1 & ; \quad n = 1 \\ 0 & ; \quad n > 1 \end{cases} \qquad (5-10)$$

の性質が成立することから，（5－9）式の右辺は $g(r)$ に一致する。すなわち，（5－7）式が成立することが導びけた。

（証明終り）

今，GF（2^b）上のd個の元を用いて列ベクトルを構成し，これのGF（2^b）上の和が単位元となる関係を満足し，しかもd回巡回しても線形独立な関係を満足する列ベクトルの数を N(d,b)とする。Hマトリクスの列ベクトルが，上記のd個の元からなる列ベクトルを含み，全体としてr個の元より構成されるとすれば，dはrの約数でなければならない。しかも，Hマトリクスの列ベクトルは奇数重み列の条件を満足しなければならないことから，このようなd個の元からなる列ベクトルは，r個の元よりなる列ベクトル中に奇数個存在しなければならない。すなわち，r／dは奇数でなければならない。以上から次の関係が成立する。

$$2^{b(r-1)} = \sum_{(r/d);odd} d \cdot N(d, b) \qquad (5-11)$$

（5－11）式に先の定理5－4を適用すると，長さrに対するN（r，b）を求めることができる。

$$N(r, b) = \frac{1}{r} \sum_{\substack{d \mid r \\ d;odd}} \mu(d) \cdot 2^{b(\frac{r}{d}-1)} \qquad (5-12)$$

これから，巡回性奇数重み列SbEC符号の最大符号ビット長Nは次の定理により表わすことができる。

（定理5－5）

巡回性奇数重み列SbEC符号の最大符号ビット長は次式により表わすことができる。

$$N = b \cdot n = b \cdot r \cdot N(r, b) = b \sum_{\substack{d \mid r \\ d:odd}} \mu(d) \cdot 2^{b\left(\frac{r}{d}-1\right)} \qquad (5-13)$$

□

ここで，b，rに対する符号長n＝r・N(r，b)の関係を表5－5に示す。また，情報ビット数に対する検査ビット数の関係を(5－5)式に示した巡回性SbEC符号の場合と合わせて図5－17に示す。ここで，b＝1としたとき，巡回性奇数重み列SEC−DED符号に一致することに注意する必要がある。奇数重み列SbEC符号に対して巡回性の条件を付与した符号(表5－5に示す)と，巡回性の条件を付与しない符号(表4－2に示す)を比較するとほとんどその符号長は変わらないことがわかる。これは，表5－4と表4－1の場合と比較しても同様である。これから，巡回性の条件は符号長をほとんど短縮せずに符号に巡回性の条件を与え，その符号化・復号化回路にくり返し性を与える条件と言える。

また，巡回性の符号としても奇数重み列SbEC符号の復号法，および定理4－1，4－2，4－4に示す性質は保たれることは自明である。

巡回性奇数重み列SbEC符号の例を挙げよう。b＝2とし，既約多項式$g(x)=x^2+x+1$より構成される同伴行列Tを使用して，図5－18に巡回性奇数重み列(72，64)S2EC符号を示す。この符号は現在得られている(72，64)S2EC符号中，最も検出能力の高い符号†の一例である。この符号の場合，2バイト間誤りに対するミスコレクト率は34.3％，2バイト間2ビット誤りに対するミスコレクト率は25.7％である。

表5－5　巡回性奇数重み列SbEC符号の符号長n^*

r* \ b	1**	2	3	4	5	6	7	8
2	2	4	8	16	32	64	128	256
3	3	15	63	255	1023	4095	16383	65535
4	8	64	512	4096	32768	262144		
5	15	255	4095	65535	1048575			
6	30	1020	32760	1048560				
7	63	4095	262143					
8	128	16384	2097152					

＊符号ビット長＝n・b，検査ビット数＝r・b　　(空欄は未計算部)

＊＊巡回性SEC−DED符号に一致

† 表4－4参照

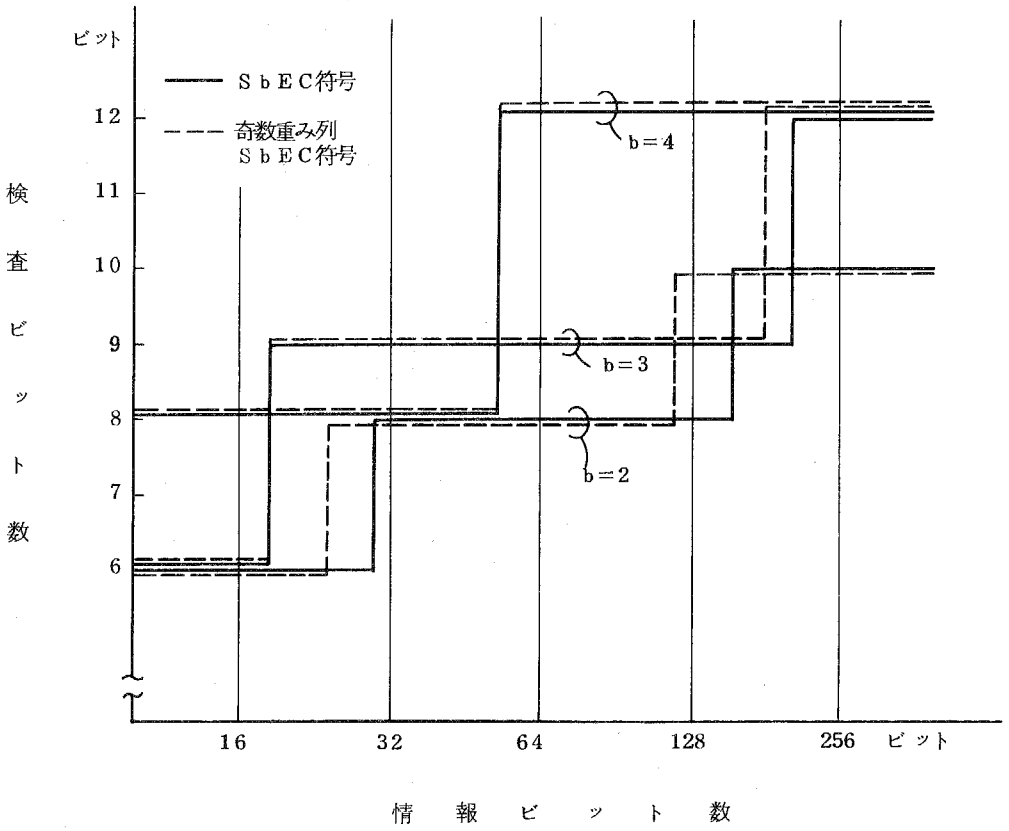


図５－17　巡回性ＳｂＥＣ符号と巡回性奇数重み列ＳｂＥＣ符号の情報ビット数に対する検査ビット数の関係

154

$$H_0 = \begin{bmatrix} I & I & T & T^2 & I & I & T & T^2 & I \\ I & I & T & I & T & T^2 & T^2 & T & 0 \\ T & T^2 & T & T^2 & T & T^2 & 0 & 0 & 0 \\ T^2 & T & T^2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \qquad T^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad 0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$H = [\, H_0 \mid R \cdot H_0 \mid R^2 \cdot H_0 \mid R^3 \cdot H_0 \,]$$

$$R = \begin{bmatrix} 0 & 0 & 0 & I \\ I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \end{bmatrix}$$

$$\{0,\ I,\ T,\ T^2\} \in GF(2^2)$$

$$R^0 = R^4$$

図５－18　巡回性奇数重み列（72, 64）Ｓ２ＥＣ符号

5－5　巡回性 SbEC-DbED 符号

現在，ＳｂＥＣ－ＤｂＥＤ符号を系統的に巡回性符号の形に構成するのは，理論的に求められていない。そこで，最も実用的と考えられるバイト長ｂ＝４ビット，情報ビット数６４，１２８ビットの場合について計算機プログラムにより求めることとする。この場合，特にＧＦ（2^b）上の元を選択する上で，全体のＨマトリクスの重みが最小となるように考慮を加えている。

まず，巡回性の性質を促進する観点から，２－モジュラー化符号（２－Modularized Code）について提案し，次にこれを用いて一般の巡回性ＳｂＥＣ－ＤｂＥＤ符号を示す。

５－５－１　２－モジュラー化ＳｂＥＣ－ＤｂＥＤ符号[26]

（定理５－６）

ＴをＧＦ（2^b）上の原始元，Ｉを単位元とするとき，次に示す符号はＳｂＥＣ－ＤｂＥＤの機能を有する。

$$H=\left[\begin{array}{cccccc:cccccc:c}
I & I & I & & I & & I & I & I & & I & & I & \\
T & T^2 & T^3 & \cdots & T^p & & T^{-1} & T^{-2} & T^{-3} & \cdots & T^{-p} & & & I & \\
T^{-1} & T^{-2} & T^{-3} & & T^{-p} & & T & T^2 & T^3 & & T^p & & & & I \\
\multicolumn{5}{c}{\longleftarrow\ \text{モジュール}\ 0\ \longrightarrow} & & \multicolumn{5}{c}{\longleftarrow\ \text{モジュール}\ 1\ \longrightarrow} & & & &
\end{array}\right] \qquad (5-14)$$

ここで，$p=2^{b-1}-1$ である。 □

（証明）

検査バイトを追加した（４－２９）式に示す修正Reed－Solomon符号を再掲する。

$$H=\left[\begin{array}{ccccccc:ccc}
I & I & I & & I & & I & I & & \\
I & T & T^2 & \cdots & T^i & \cdots & T^{2^b-2} & & I & \\
I & T^2 & T^4 & & T^{2i} & & T^{2(2^b-2)} & & & I
\end{array}\right] \qquad (5-15)$$

この符号において，第２列目をＴで割り，第３列目をT^2で割り，検査部をのぞいて同様に第（$i+1$）列目をT^iで割る。その結果，情報部については次のようになる。

$$\begin{bmatrix} I & T^{-1} & T^{-2} & & T^{-i} & & T^{-(2^b-2)} \\ I & I & I & \cdots & I & \cdots & I \\ I & T & T^2 & & T^i & & T^{2^b-2} \end{bmatrix} \qquad (5-16)$$

このマトリクスから第1列目を除去し，全体を1行上へ巡回シフトする。

$$\begin{bmatrix} I & I & & I & & I \\ T & T^2 & \cdots & T^i & \cdots & T^{2^b-2} \\ T^{-1} & T^{-2} & & T^{-i} & & T^{-(2^b-2)} \end{bmatrix} \qquad (5-17)$$

このとき，$i > 2^{b-1}-1$ なる i に対しては4－2－1で示した同伴行列Tの性質1より

$$i \rightarrow -(2^b-1-i)$$

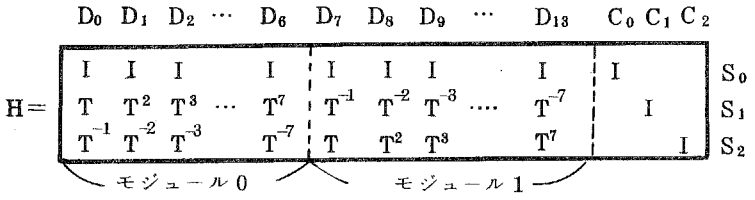
なる置換をほどこす。このようにすれば，第2行，第3行は真中で対称となる次の形を有する。ここで，$p = 2^{b-1}-1$ である。

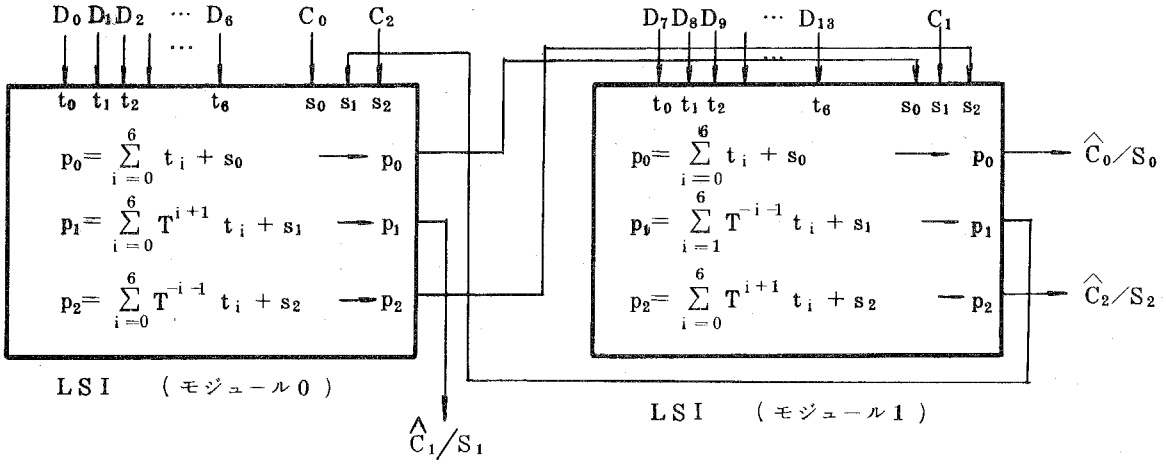
$$\left[\begin{array}{ccc:ccc} I & I & & I & I & & I \\ T & T^2 & \cdots & T^p & T^{-p} & T^{-p+1} \cdots & T^{-1} \\ T^{-1} & T^{-2} & & T^{-p} & T^p & T^{p-1} & T \end{array}\right] \qquad (5-18)$$

これに，検査部を付加すれば，（5－14）式が求まる。

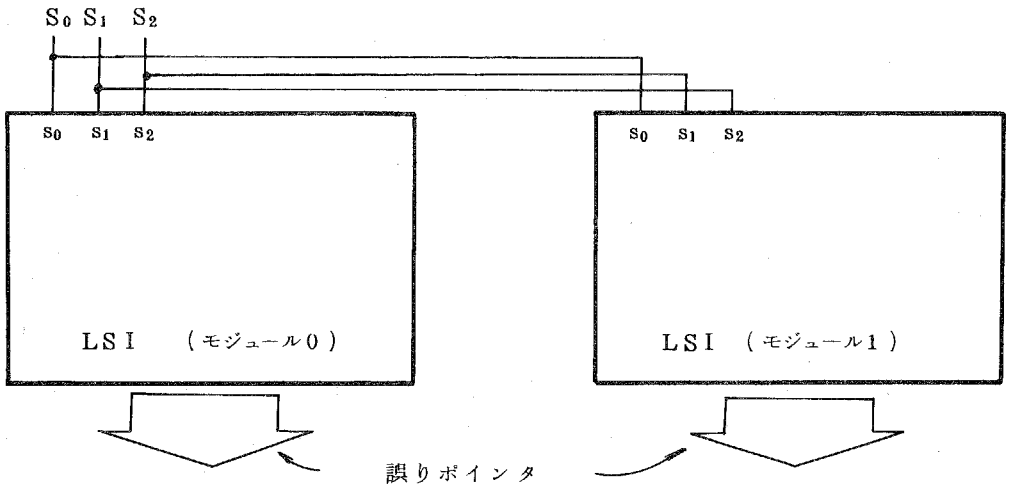
（証明終り）

この符号はもとの修正Reed－Solomon符号から1列分を差引いた符号長を有することから，最大符号ビット長はN＝b（2^b+1）となる。（5－14）式に示す符号は，情報部は単に第2行目と第3行目が置換された対称な2領域（モジュール0，モジュール1）に分割されている。これから，図5－11～図5－14に示したと同様に前半部分（モジュール0）に対する符号化・復号化回路は，後半部分（モジュール1）に対して，単に入出力関係を置換しただけで回路構成は全く同一の回路として使用することが可能である。従って，符号化・復号化回路は，全体として2分化された同一回路で構成できることになる。以上の関係を図5－19に示す。





検査ビット生成／シンドローム生成回路（CG／SG）



シンドローム デコード回路(SD)

図5－19　符号化・復号化回路

また，この構成の符号に対しては，検査部をのぞいて第2行目と第3行目の要素の積はIとなり一定値を有する。この一定値は，マトリクスの行列操作により必ずしも単位元Iである必要はなく，GF(2^b)上の非零元であればよいことは容易に理解できる。但し，第2行目の要素と第3行目の要素が等しい場合には，その列は省かれねばならない。一定値T^3を有する(44，32)S4EC−D4ED符号の例を次に示す。

$$H=\left[\begin{array}{cccc:cccc:cc} I & I & I & I & I & I & I & I & I & \\ T^{13} & T^{14} & I & T & T^5 & T^4 & T^3 & T^2 & & I \\ T^5 & T^4 & T^3 & T^2 & T^{13} & T^{14} & I & T & & & I \end{array}\right] \qquad (5-19)$$

以上の符号構成に示すごとく，符号化・復号化回路に2個のモジュラー化を与える符号を，2−モジュラー化符号(2−Modularized Code)と呼ぶことにする。

(定理5−7)

次に示すHマトリクスは，SbEC−DbED符号である。符号長はGF(2^b)上で2k+rである。

$$H=\left[\begin{array}{cccc:cccc:cc} I & I & \cdots & I & 0 & 0 & \cdots & 0 & I & \\ h_{1,0} & h_{1,1} & \cdots & h_{1,k-1} & h_{r-2,0} & h_{r-2,1} & \cdots & h_{r-2,k-1} & & I \\ h_{2,0} & h_{2,1} & \cdots & h_{2,k-1} & h_{r-3,0} & h_{r-3,1} & \cdots & h_{r-3,k-1} & & & I \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & & & & \ddots \\ h_{r-2,0} & h_{r-2,1} & \cdots & h_{r-2,k-1} & h_{1,0} & h_{1,1} & \cdots & h_{1,k-1} & & & & & I \\ 0 & 0 & \cdots & 0 & I & I & \cdots & I & & & & & & I \end{array}\right] \qquad (5-20)$$

ここで，I，0はそれぞれGF(2^b)上の単位元，零元である。また，$h_{i,j}$，(i=1，2，…，r−2，j=0，1，…，k−1)はGF(2^b)上の元であり，これらを用いた次の(r−1)行k列のHマトリクスはSbEC−DbED符号とする。

$$H=\left[\begin{array}{cccccc:ccc} I & I & \cdots & I & \cdots & I & I & & \\ h_{1,0} & h_{1,1} & \cdots & h_{1,j} & \cdots & h_{1,k-1} & & I & \\ \vdots & \vdots & & \vdots & & \vdots & & & \ddots \\ h_{r-2,0} & h_{r-2,1} & \cdots & h_{r-2,j} & \cdots & h_{r-2,k-1} & & & I \end{array}\right] \qquad (5-21)$$

□

証明は容易であるので省略する。（５－２０）式に示す符号も明らかに２－モジュラー化符号である。図５－２０にこの定理の手法を用いた（７６，６４）Ｓ２ＥＣ－Ｄ２ＥＤ符号を示す。

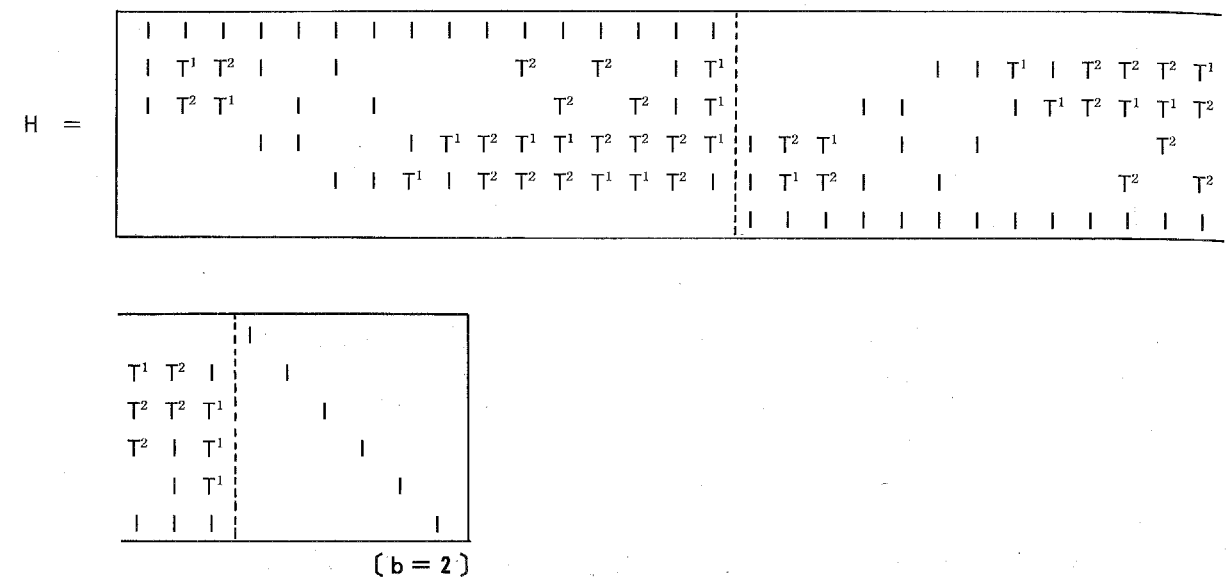


図５－20　２－モジュラー化（76，64）Ｓ２ＥＣ－Ｄ２ＥＤ符号

５－５－２　巡回性ＳｂＥＣ－ＤｂＥＤ符号[26]

バイト長ｂ＝４ビット，情報ビット数６４，１２８ビットに対して，計算機プログラムにより最適化を図った巡回性ＳｂＥＣ－ＤｂＥＤ符号を示す。最適化とは，Ｈマトリクスの重みを最小とすることであり，これにより符号化・復号化回路のゲート数は最小となる。このプログラムはＤ４ＥＤ（二重バイト誤り検出）機能の評価と，重み最小化を行う機能を有するもので，ＰＬ／Ｉで２８０ステップである。

なお，同伴行列Ｔは$g(x)=x^4+x+1$なる既約多項式から作るものとする。

〔１〕　巡回性（１４４，１２８）Ｓ４ＥＣ－Ｄ４ＥＤ符号

巡回性符号としては，（２－７）式の巡回性符号の定義からｒ＝４，ｄ＝４の符号となる。まず，定理５－６に示す２－モジュラー化符号の構成より，次のＳ４ＥＣ－Ｄ４ＥＤ符号を得る。ここで，一定値はT^{14}である。

$$H=\left[\begin{array}{cccc:cccc:c} I & I & I & \cdots & I & I & I & \cdots & I & I & \\ I & T & T^2 & \cdots & T^6 & T^{14} & T^{13} & \cdots & T^8 & & I \\ T^{14} & T^{13} & T^{12} & \cdots & T^8 & I & T & \cdots & T^6 & & & I \end{array}\right] \qquad (5-22)$$

このうち，情報部として８列ベクトルと検査部の１列ベクトルを選択し，さらに，すべて'0'の行を加えることにより，巡回性符号としての基本ＨマトリクスH_0を得る。例えば，２－モジュラー化符号の性質を活かして，次のH_0（一定値＝T^{14}）を求めることができる。

$$H_0=\left[\begin{array}{cccc:cccc:c} I & I & I & I & I & I & I & I & I \\ T^{14} & T^{13} & T^{12} & T^{11} & I & T & T^2 & T^3 & 0 \\ I & T & T^2 & T^3 & T^{14} & T^{13} & T^{12} & T^{11} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \qquad (5-23)$$

但し，このH_0を３回巡回させてH_0を含む４個を連接させて構成するＨマトリクスが，S4EC－D4ED符号である保証は必ずしもない。すなわち，以上の過程による符号構成ではＳ４ＥＣの機能は満足しているが，特にＤ４ＥＤの機能は必ずしも保証されていないからである。（5－22）式から，３５（$={}_7C_4$）種の符号が１５個の一定値の下で構成できる。しかし，この機能の確認はプログラムにより実行する。

表５－６に，各一定値に対する（１４４，１２８）Ｓ４ＥＣ－Ｄ４ＥＤ符号の数を示す。これらの符号のうち，Ｈマトリクスとして最少の'1'の数を有するＨマトリクスを図５－２１に示す。このＨマトリクスの全体の重みは５９２である。参考までに，Ｓ４ＥＣ－Ｄ４ＥＤ符号として最大の重みのものは８２４である。図５－２１に示す符号は，基本部分ＨマトリクスH_0内を２－モジュラー化していることから，全体として８個のモジュラー化が実現でき，符号化・復号化回路も８個の同一回路で構成可能となる。

図５－２２には，H_0内を上述と同様２－モジュラー化するとともに，一定値を与える条件を有さないで最小重みを与えた４個の巡回性（１４４，１２８）Ｓ４ＥＣ－Ｄ４ＥＤ符号の基本部分マトリクスH_0を示す。いずれの符号も全体のＨマトリクスの重みは５６８を有し，図５－２１に示す符号よりすぐれている。

表５－６　巡回性（144，128）S4EC-D4ED符号の数

一定値	符号数	一定値	符号数
I	0	T^8	4
T	4	T^9	11
T^2	4	T^{10}	0
T^3	11	T^{11}	11
T^4	4	T^{12}	11
T^5	0	T^{13}	11
T^6	11	T^{14}	11
T^7	11		

H ＝

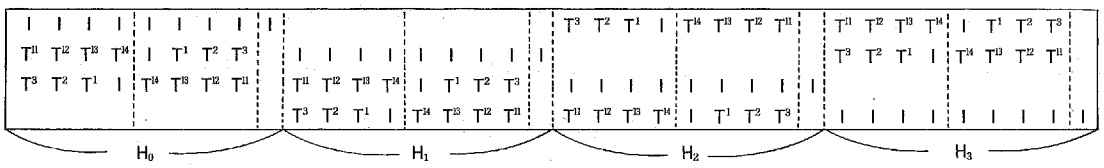


図５－21　一定値(T^{14})を有する最小重みの巡回性（144，128）
S４ＥＣ－Ｄ４ＥＤ符号　〔ｂ＝４〕

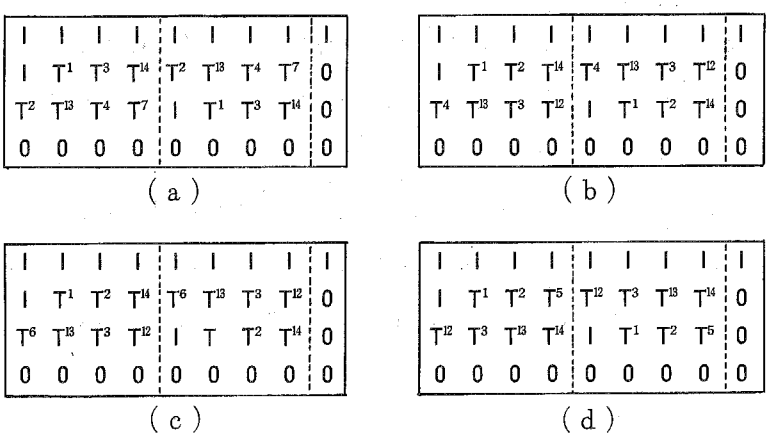


図５－22　最小重みを有する巡回性（144，128）Ｓ４ＥＣ－Ｄ４ＥＤ
符号の基本部分Ｈマトリクス(H_0)

〔2〕　巡回性（80，64）S4EC−D4ED符号

情報ビット数64ビットに対して，最小重みを与える巡回性（80，64）S4EC−D4ED符号を図5−23に示す。Hマトリクスの重みは240に等しい。この符号は，GF（2^4）上の元0，I，T，$T^2 \cdot\cdot T^{14}$ のうちで，最も重みの小さい元0，I，T，T^2，T^{14} を用いて構成したものである。

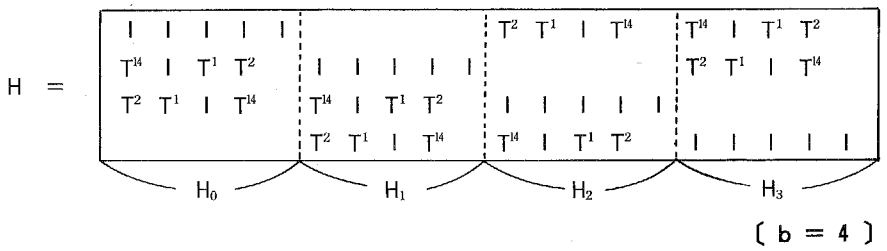


図5−23　最小重みを有する巡回性（80，64）S4EC−D4ED符号

5－6 結　　言

本章では，符号化・復号化回路にくり返し性を与え，ＬＳＩ化に適する構成を与える巡回性符号の構成を，前章までに示した各種のバイト誤り検出・訂正符号に適用した。主な結論を以下に示す。

(1) 奇数重み列ＳｂＥＣ符号，ＳＥＣ－ＳｂＥＤ符号，ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号，ＳｂＥＣ－ＤｂＥＤ符号に対して，巡回性の性質を有する符号構成を具体的に示した。また，巡回性の性質を各符号に与えてその符号長の一般式を求めた。結果をまとめて表５－７に示す。その結果，符号長の短縮は高々７～８％程度と小さいことが判明し，これにより巡回性の符号構成は実用的な方法であることを明らかにした。

(2) 巡回性ＳｂＥＣ－ＤｂＥＤ符号を理論的に導出することは困難であり，これに対しては，バイト長ｂ＝４ビットで情報ビット数６４ビット，１２８ビットを有する符号を計算機プログラムにより求めた。特に導出に当っては，Ｈマトリクス重みが最小となるように考慮し，具体的な符号を求めた。

(3) くり返し性を有するＳｂＥＣ－ＤｂＥＤ符号として，２－モジュラー化符号を示した。これは，くり返し性２を有する符号であり，これを巡回性符号の基本部分マトリクスに適用することにより，全体としてさらにくり返し性を増大させることが可能となる。

表５－７　符号長の一般式

符号機能		最大符号長N（ビット） 非巡回性符号	最大符号長N（ビット） 巡回性符号	
SEC		2^r-1	$\sum_{d\mid r}\mu(d)\cdot(2^{r/d}-1)$	$\mu(d)$；メビウス関数 $\sum_{d\mid r}$；rの約数dに対する和 $\sum_{\substack{d\mid r\\ d:odd}}$；rの約数で奇数のdに対する和 G.C.D.(x,y)；x,yの最大公約数
SEC−DED		2^{r-1}	$\sum_{\substack{d\mid r\\ d:odd}}\mu(d)\cdot 2^{(r/d-1)}$	
SEC−SbED	構成A	$2^{br}-(2^b-b)^r$	—	
	構成B	$b(2^{r-b+1}-1)$	$\sum_{d\mid r}\mu(d)\cdot\{(2^b)^{r/d}-(2^b-b)^{r/d}\}$G.C.D.(d, b)	
SEC−DED−SbED	構成A	$b\cdot 2^{r-b}$	—	
	構成B	$\frac{1}{2}\{2^{br}+b^r-(2^b-b)^r\}$	$\frac{1}{2}\sum_{\substack{d\mid r\\ d:odd}}\mu(d)\{(2^b)^{r/d}+b^{r/d}-(2^b-b)^{r/d}\}$G.C.D.(d, b)	
SbEC	非奇数重み列	$b(2^{br}-1)/(2^b-1)$	$\frac{b}{2^b-1}\sum_{d\mid r}\mu(d)\{(2^b)^{r/d}-1\}$G.C.D.$(d,2^b-1)$	
	奇数重み列	$b\cdot 2^{b(r-1)}$	$b\sum_{\substack{d\mid r\\ d:odd}}\mu(d)\ 2^{b(r/d-1)}$	
SbEC−DbED		$\begin{cases} b(2^b+2)^{\frac{r-1}{2}} & \cdots r=3\text{以上の奇数} \\ 2b(2^b+2)^{\frac{r-1}{2}} & \cdots r=4\text{以上の偶数} \end{cases}$	—*	*本論文では，b＝4，k＝64，128に対して，計算機プログラムにより最適符号を求めている。

第6章　信頼度改善効果

6－1 緒　　言

バイト誤り検出・訂正符号による信頼度改善効果について示す。特に，ここでは主記憶装置を対象とし，定期保守を実行する場合を考える。定期保守は，固定故障が生じて符号により訂正が行われている場合，定期保守時に正常な予備品と取り替えることにより，全く故障なしの状態とする効果を与えることができる。例えば，SEC－DED符号を主記憶装置に適用している場合，故障が生じて1ビット誤りの状態に対しては，自動的に訂正が行われる。さらに，もう1ビットの誤りが生ずると2ビット誤りとなり，符号により検出して装置は使用不能となる。しかし，この追加の1ビット誤りが生ずる以前に定期保守を行うことができれば，結果的に装置を正常な状態として処理の続行を図ることが可能となる。このように，定期保守の効果は誤り検出・訂正符号の効果を増大させる上で重要である。

本章では，バイト構成の記憶素子を有するメモリアレーに対し，各種のバイト誤り検出・訂正符号を適用して，信頼度の改善度を具体的に求める。

6－2　モ　　デ　　ル

メモリの構成を，符号語としてビット方向にＮビット，また語（ワード）方向にＭ個の素子を並べたアレー構成とする。使用する記憶素子は，ｂビットの入出力を有する素子とし，従って，上記メモリ構成はビット方向にＮ／ｂ個の素子を並べた形となる。メモリ構成を図６－１に示す。

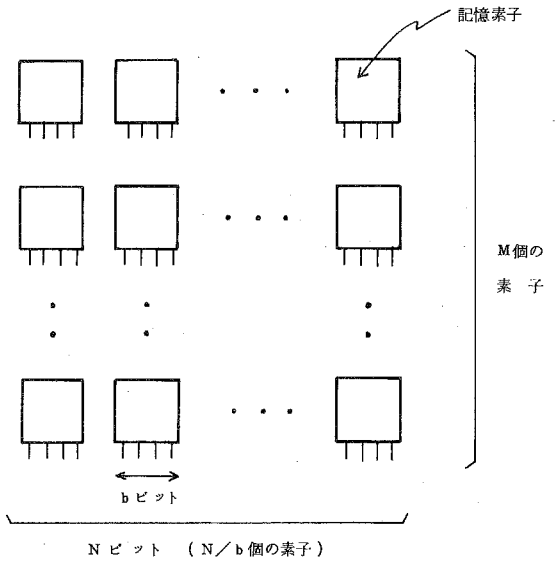


図６－１　メモリアレー

故障に関し，次の仮定を置くことにする。

(1)　素子中の故障は，すべてランダム生起とする。

(2)　素子に故障が生ずると，その素子の全語（ワード）は故障状態に陥るとする。

(3)　誤りが生じたとき，ｂビット中２ビット以上の複数ビット誤りが生起する確率をβとし，１ビット誤りの確率を（１－β）とする。このとき，２ビット以上の各多数ビット誤りの生起確率は等しいとする。

また，ここでは評価の対象をメモリアレーに限定し，記憶装置中のデータパス回路，制御回路，符号化・復号化回路等は含めないこととする。

今，誤り訂正符号を使用しない場合の全故障率F_0は，次式のように表わすことができる。

$$F_0 = \frac{N-R}{b} \cdot M \cdot \lambda_m \qquad (6-1)$$

ここで，Ｒは全検査ビット数であり，（Ｎ－Ｒ）は誤り訂正符号を適用しない場合の情報ビット

数に等しい。また，λ_m は1素子当りの故障率を示す。

また，定期保守間隔を τ とする。このとき，1素子が時間 t＝0 から t＝τ までの間に故障する確率は次式で与えられる。(78)

$$p_m = 1 - \exp(-\lambda_m \cdot \tau) \qquad (6-2)$$

図6－1に示すメモリアレー中の1行（N／b 個の素子からなる）が，全く故障のない確率 p_0 は次式のように表わすことができる。

$$p_0 = (1 - p_m)^{N/b} \qquad (6-3)$$

また，アレー中，ある1行において1個の素子が故障する確率，2個の素子が故障する確率は，それぞれ次のように表わすことができる。

$$p_1 = \frac{N}{b} \cdot p_m \cdot (1 - p_m)^{\frac{N}{b}-1} \qquad (6-4)$$

$$p_2 = {}_{N/b}C_2 \cdot p_m^2 \cdot (1 - p_m)^{\frac{N}{b}-2} \qquad (6-5)$$

C ; combination

そこで，p_0～p_2 と各符号のもつ誤り検出率を用いて，各符号対応に，t時間後に全メモリアレーが正しい状態（符号により訂正できる状態を含む）である確率 $P_c(t)$，誤り検出可能である確率（検出しても訂正できない状態の確率）$P_d(t)$，非検出でミスコレクトする確率 $P_e(t)$ が計算できる。このとき，t時間後にこのメモリアレーが故障している確率 P(t) は次式のように表わすことができる。

$$P(t) = P_d(t) + P_e(t) \qquad (6-6)$$

以上の条件の下で，定期保守を考慮したメモリアレー全体の故障率 F_t，および非検出率 F_n は次のように表わすことができる。(78)

$$F_t = \frac{P(\tau)}{\int_0^{\tau} \{1-P(t)\}dt} \qquad (6-7)$$

$$F_n = \frac{P_e(\tau)}{\int_0^{\tau} \{1-P_e(t)\}dt} \qquad (6-8)$$

具体的な数値計算において，以下の数値を使用することとする。

情報ビット数	64ビット
全メモリ容量	16Mバイト（バイト＝8ビット）
	（検査ビット分の容量は別途加わる）
使用記憶素子	64K語×4ビット構成（b＝4）
定期保守間隔（τ）	1週間（τ＝168時間）
複数ビット誤り生起確率（β）	0.5　または　0.1

また，以下に示す各計算結果の図中，F_t' は τ を無限大とした場合，すなわち，定期保守を行わないで符号のみによる場合のメモリアレー故障率を示す。

6－3　SEC-DED 符号適用の場合

図６－１に示すバイト構成を有する記憶素子を用いたメモリアレーに対し，ＳＥＣ－ＤＥＤ符号を適用した場合について考える。*

この場合の誤り状態とその確率を示したものを図６－２に示す。単一素子故障の場合には，３個の状態に分かれる。すなわち，１ビット誤りとなる場合と２ビット以上の複数ビット誤りとなる場合があり，後者は符号により検出できる場合と非検出（ミスコレクト）の場合となる。次に，１ビット誤りとなる単一素子故障の状態で，さらにもう１素子故障して２素子故障の場合となったときを考える。このとき，全体として２ビット以上の複数ビット誤りとなり，この符号では訂正できず，検出できる状態か，非検出の状態のいずれかとなる。

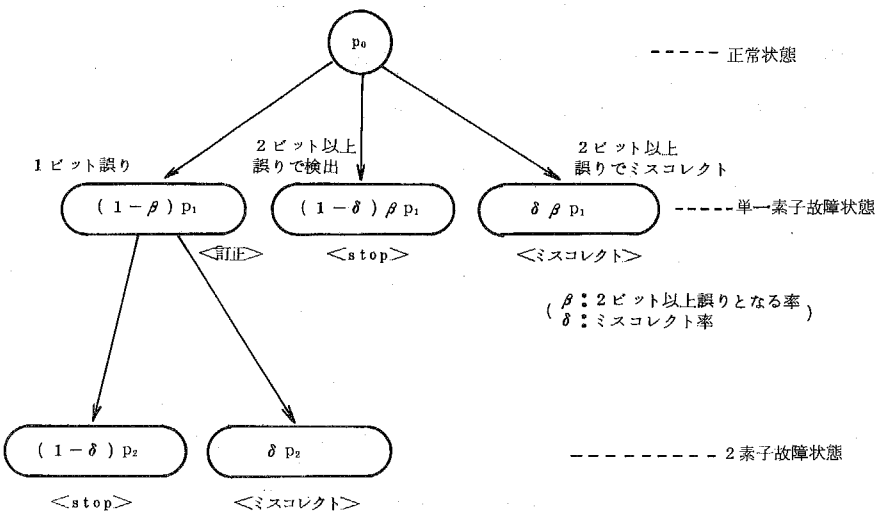


図６－２　ＳＥＣ－ＤＥＤ符号適用の場合の故障状態

１ビット誤りは訂正でき正しい状態とすることができるから，この符号を適用した場合には P_c は次のように表わすことができる。

$$P_c = \{(1-p_m)^{N/b} + (1-\beta)p_1\}^M \qquad (6-9)$$

＊　１ビット出力の記憶素子を用いたメモリアレーに対し，この符号を用いた場合の信頼度改善については，文献（７９）にすでに示されている。

非検出でミスコレクトする場合は，主として1バイト内の3ビット以上奇数ビット誤り，または2バイト間にわたる3ビット以上の奇数ビット誤りである。本符号では，4ビット以上の偶数ビット誤りはほとんど（99%以上）検出することができることが知られており，この場合には省略することができる。また，3以上の奇数であるℓに対して，この符号によりℓビット誤りを検出できる確率をα_ℓとする。

$$P_e = M \cdot p_e \qquad (6-10)$$

$$p_e = (1-\alpha_3)\frac{\beta}{b-1} p_1 + \{(1-\alpha_3)\frac{\beta}{b-1}(1-\beta) + (1-\alpha_5)\frac{\beta}{b-1}(1-\beta)\} p_2$$

p_eの第1項は，単一バイト内3ビット誤りの場合であり，第2，第3項はそれぞれ2バイト間にわたる誤りのうち，1バイトでは1ビット，他バイトでは2ビット，4ビット誤りの場合である。

これから，P_dは次のように求めることができる。

$$P_d = (1-p_e)^M \qquad (6-11)$$

以上から，全メモリアレーのF_t，F_nは（6－6）式～（6－8）式に代入することにより求められる。具体的な数値を代入した結果を図6－3に示す。これから，F_tはβの値に依存しており，ほぼβ倍に故障率は減少する。なお，ここで対象とした符号は文献（34）に示す図5の符号である。

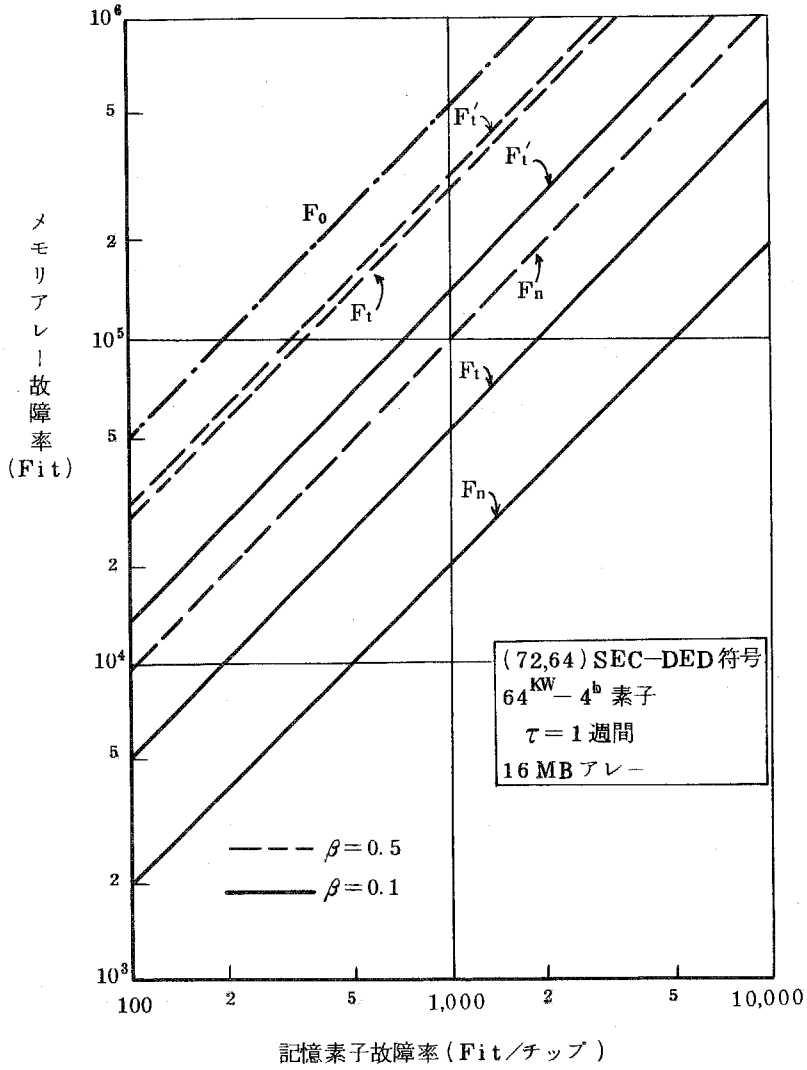


図 6－3　SEC—DED符号による信頼度改善

6－4　SEC-DED-SbED 符号適用の場合

ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号を適用した場合の故障状態を図６－４に示す。この場合，図６－２と異なるのは，この符号のＳ４ＥＤの機能により単一素子故障では非検出の状態がないことである。この場合，P_e は（６－９）式に一致する。

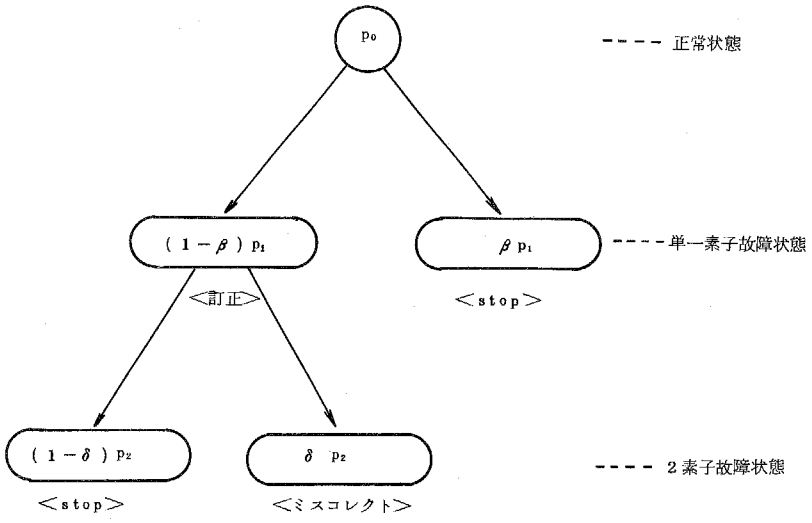


図６－４　ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号適用の場合の故障状態

本符号を適用して非検出となる場合を考える。２素子間にまたがる３ビット以上の場合がこの場合に相当する。この符号の場合，２ビット誤り，４ビット誤りは１００％近い検出率を有することから，P_e は次式により求めることができる。

$$P_e = M \cdot p_e \qquad (6-12)$$

$$p_e = \{ (1-\alpha_3)(1-\beta)\frac{\beta}{b-1} + (1-\alpha_5)(1-\beta)\frac{\beta}{b-1} \} p_2$$

p_e として第１項は２バイト間で１ビット誤りと２ビット誤りの場合，第２項は２バイト間で１ビット誤りと４ビット誤りの場合である。

P_d は次式のように求めることができる。

$$P_d = (1-p_e)^M \qquad (6-13)$$

以上から，本符号適用の場合のメモリアレー故障率F_t，非検出率F_nは（6－6）式～（6－8）式に代入することにより求められる。具体的な数値を代入して求めた結果を図6－5に示す。これから，F_tは前のSEC－DED符号の場合とほぼ同一となるが，F_nは2桁～3桁も改善されることがわかる。なお，ここで対象とした符号は，図3－11に示す（73，64）SEC－DED－S4ED符号である。この場合の符号パラメータα_3，α_5は付録2中の付表（2）から与えられる。

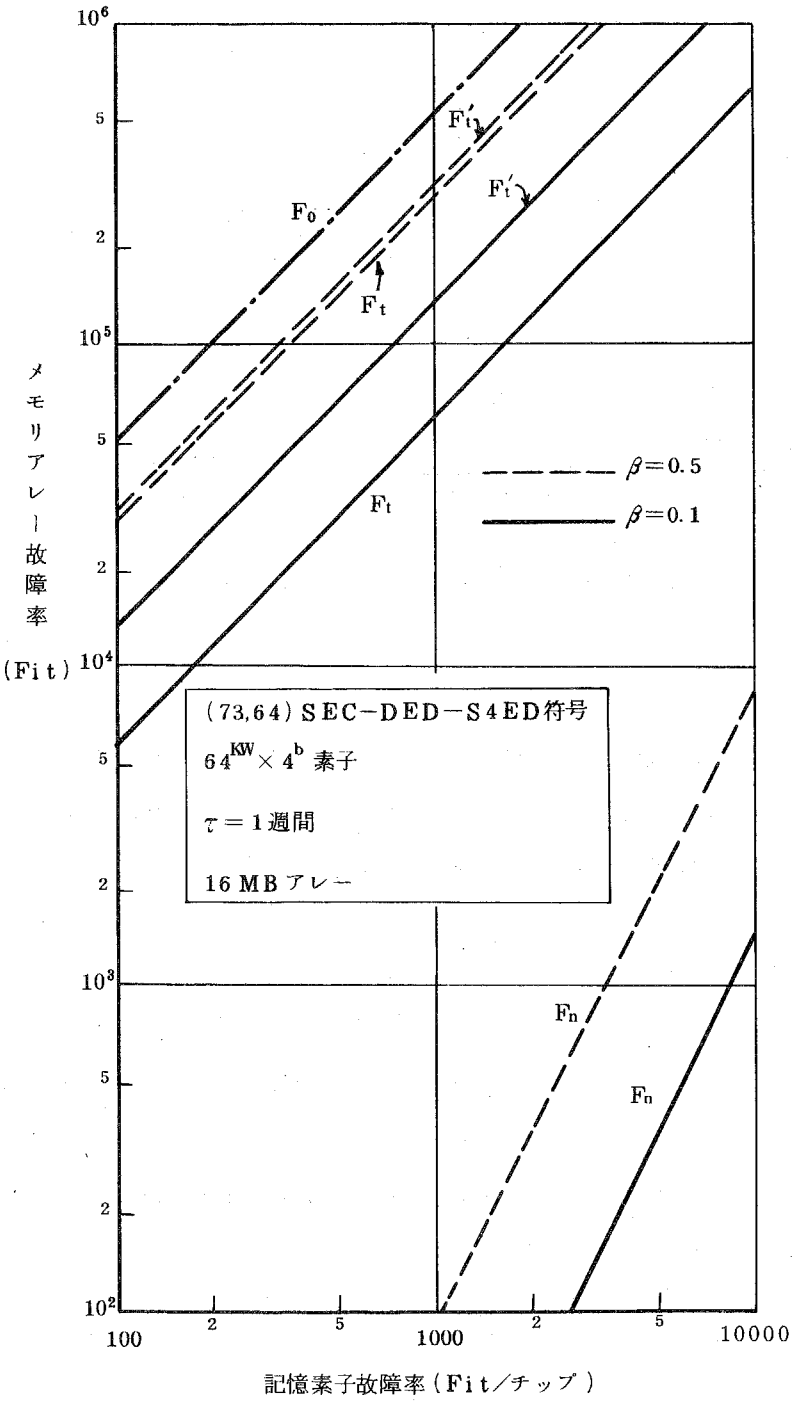


図6－5　SEC－DED－S4ED符号による信頼度改善

6－5　SbEC 符号，SbEC-DbED 符号適用の場合

Ｓ４ＥＣ符号，Ｓ４ＥＣ－Ｄ４ＥＤ符号を適用した場合の故障状態を示すと，それぞれ図６－６，図６－７となる。いずれの符号も単一素子故障による誤りはすべて訂正することができる。また，Ｓ４ＥＣ－Ｄ４ＥＤ符号では，２素子故障でその誤りを検出して停止する。

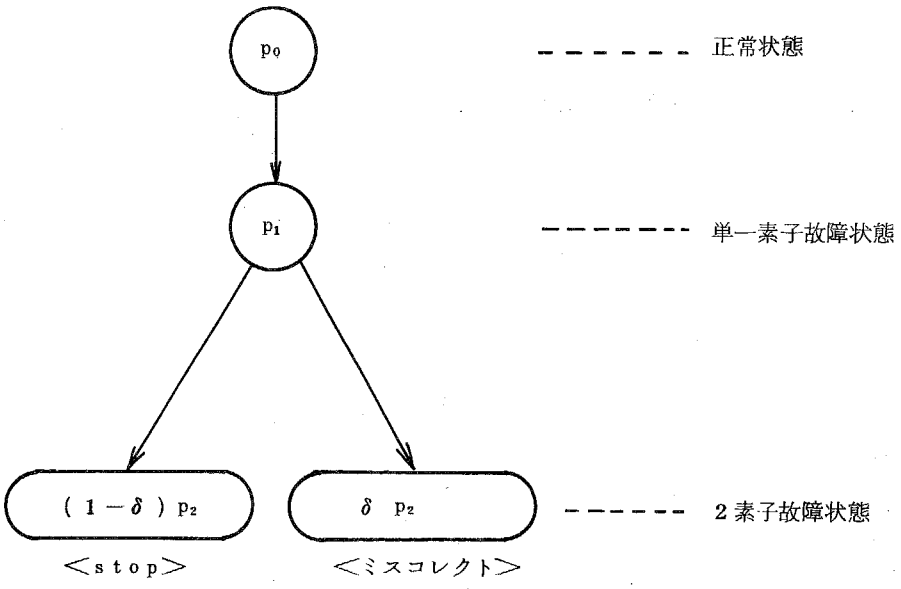


図６－６　Ｓ４ＥＣ符号適用の場合の故障状態

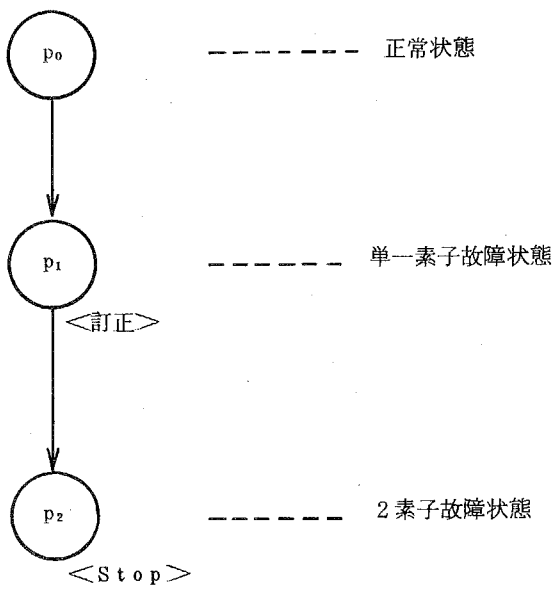


図６－７　Ｓ４ＥＣ－Ｄ４ＥＤ符号適用の場合の故障状態

P_c は次式で与えることができる。

$$P_c = (p_0 + p_1)^M = \{ (1-p_m)^{N/b} + \frac{N}{b} p_m (1-p_m)^{\frac{N}{b}-1} \}^M \qquad (6-14)$$

また，S4EC符号の場合，δ を二重バイト誤りが検出できない確率とすると，P_e は次のようになる。

$$P_e = M \cdot p_e$$
$$p_e = \delta \cdot p_2 \qquad (6-15)$$

一方，S4EC−D4ED符号に対しては，$p_e = 0$ である。

さらに，P_d は次のように求めることができる。

$$P_d = (1-p_e)^M \qquad (6-16)$$

以上から，符号を適用した場合の故障率 F_t，非検出率 F_n は，（6−6）式〜（6−8）式に代入することにより求めることができる。具体的な数値を代入することにより，信頼度改善を求めてみよう。SbEC符号に対しては図6−8に，SbEC−DbED符号に対しては図6−9にその結果を示す。S4EC符号としては図4−6に示す奇数重み列（76，64）S4EC符号を対象としており，S4EC−D4ED符号としては図5−23に示す巡回性（80，64）S4EC−D4ED符号を対象としている。前者の符号の δ は $\delta = 0.168$ である。

これから，SEC−DED符号またはSEC−DED−S4ED符号と比較して，S4EC符号，S4EC−D4ED符号の場合の故障率は1桁〜2桁改善される。また，S4EC−D4ED符号の非検出率は零に等しい。

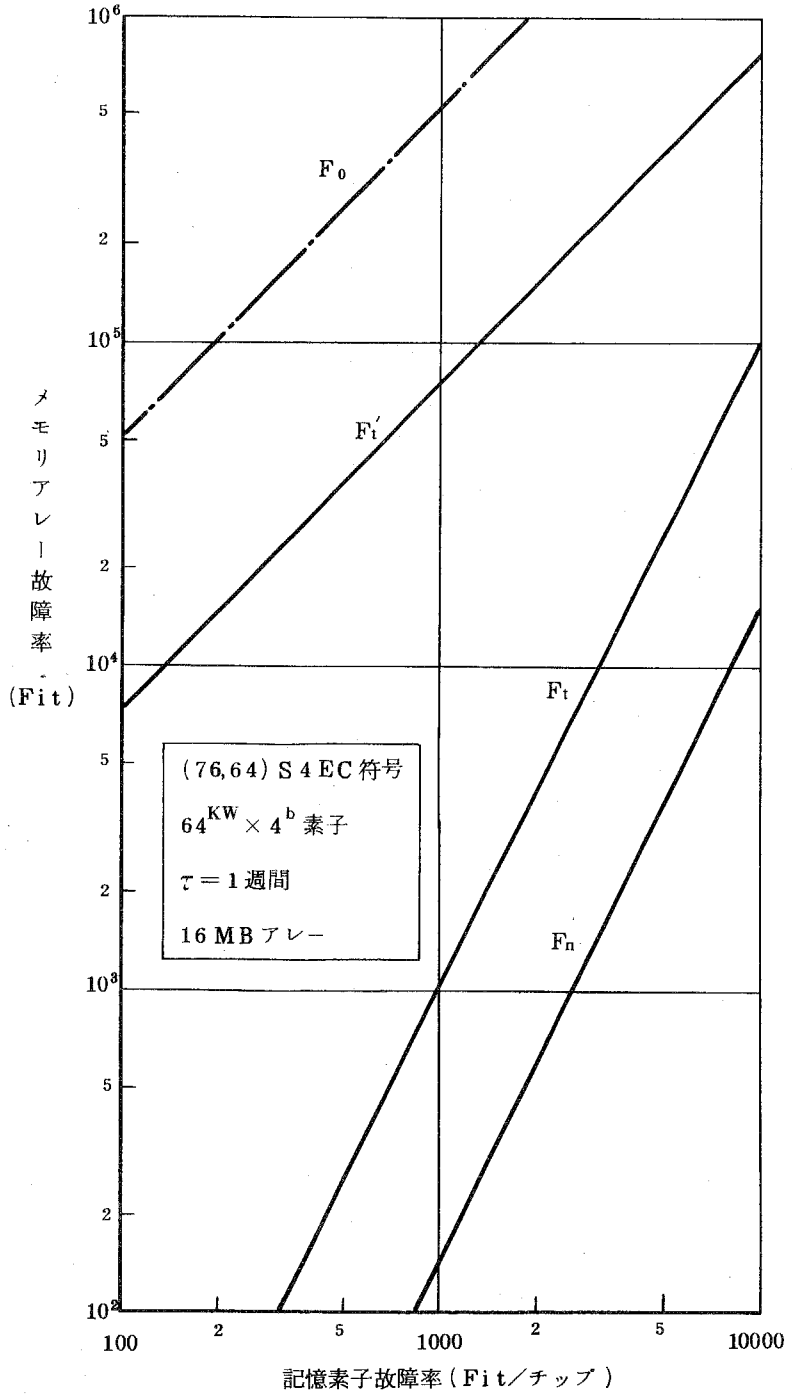


図 6－8　S 4 EC符号による信頼度改善

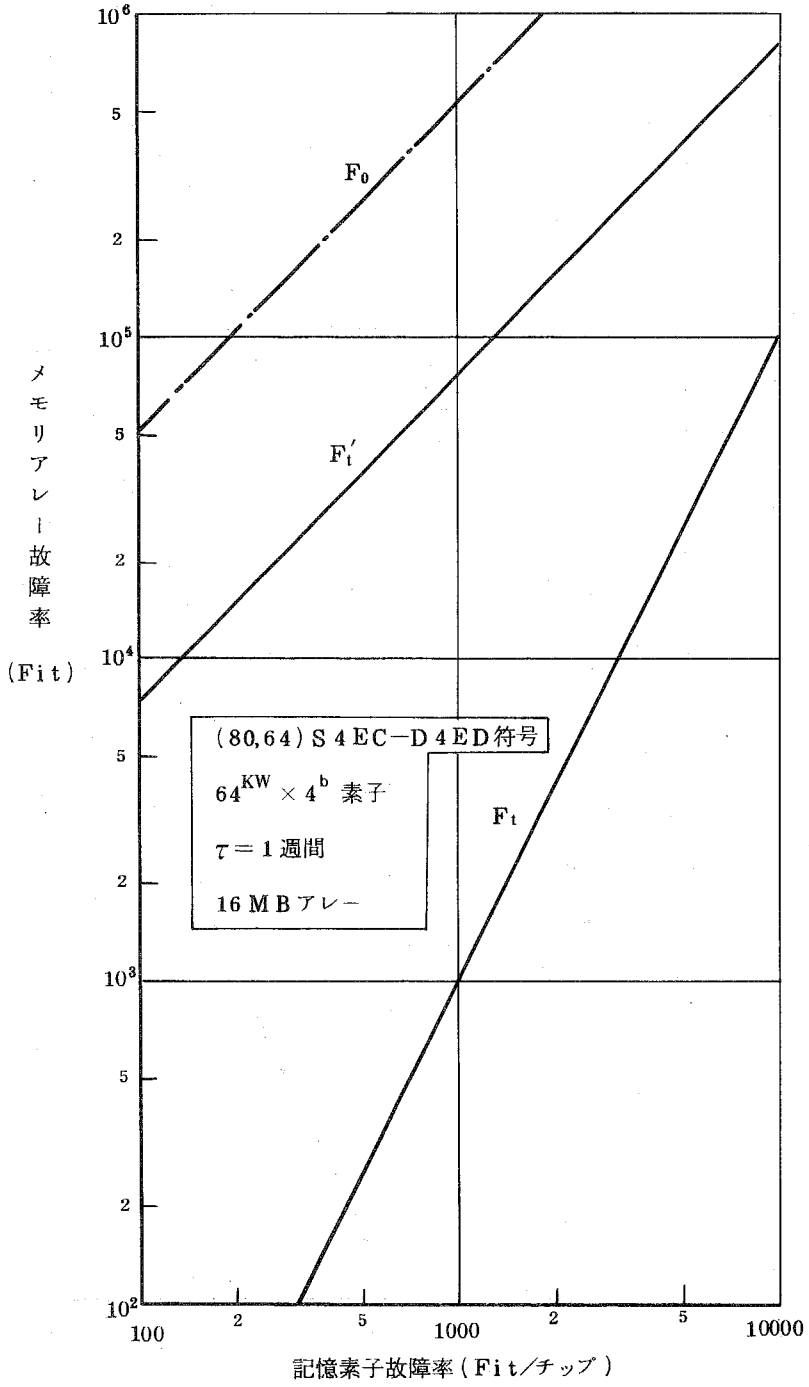


図6－9　S4EC−D4ED符号による信頼度改善

6－6 結　　言

本章では，バイト出力構成の記憶素子を有するメモリアレーに対して，各種のバイト誤り検出・訂正符号を適用して，その信頼度の改善効果を具体的に求めた。導出に当っては，定期保守を考慮した。具体的な計算においては，情報ビット数：６４ビット，使用記憶素子：64K語×４ビット構成，定期保守間隔：１週間，全メモリアレー容量：１６Mバイト（バイト＝８ビット）の数値を使用した。主な結論を以下に示す。

(1) ＳＥＣ－ＤＥＤ符号を適用した場合，信頼度改善の効果は，単一素子中に誤りが生じたとき２ビット以上の誤りとなる確率βに依存する。$\beta = 0.5$であれば，符号を適用しない場合と比較して約２倍信頼度は改善し，$\beta = 0.1$であれば１桁改善する。

(2) ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号を適用した場合，信頼度改善はＳＥＣ－ＤＥＤ符号の場合と同一である。但し，符号を適用しても検出できない率は，ＳＥＣ－ＤＥＤ符号の場合と比較して２～３桁小さくなる。

(3) Ｓ４ＥＣ符号の信頼度改善は，素子当りの故障率を1,000～2,000Ｆｉｔとすると，符号を適用しない場合と比較して２桁以上となる。

(4) Ｓ４ＥＣ－Ｄ４ＥＤ符号を適用すると，信頼度改善はＳ４ＥＣ符号の場合と同一であるが，検出できない率は零に等しい。

第7章　符号化・復号化回路

7—1 緒　　言

本章は，高速な符号化・復号化が実現できる回路構成と，具体的な各種の符号に対する回路ゲート量，回路遅延等の関係を示す。また，本回路の高信頼化を目的に，自己検査性を有する検査回路の構成法を示す。これは，符号化・復号化回路だけでなく検査回路自身も含めて，その内の単一故障を完全に検出する論理を与えるものである。さらに，本章では，符号化・復号化回路のＬＳＩ構成について示す。この回路のＬＳＩ化に当っては，第5章で示した巡回性の性質を有する符号を適用して，(1)符号長の拡張性，(2)多機能性，(3)検査回路内蔵による高信頼化を目標に，そのための符号構成法，回路構成法等について明らかにする。

以下に，各種の機能を有する符号に対して，符号化・復号化回路のゲート量，回路遅延の関係を示す。次に，この回路に対する自己検査性検査回路の構成，巡回性符号を用いたＬＳＩ構成について具体的に示す。最後に，これらの結論の一部を採用したＬＳＩ試作例について概説する。

7－2　並列符号化・復号化回路構成

本研究における符号は，CPU，MEM 等の高速性が要求される装置に適用することを考え，①符号化・復号化回路は出来るだけ高速であること，②回路自身の経済化，高信頼化の観点からゲート量は小さいこと，が特に要求される。①に対しては，並列な符号化・復号化を行うため，組合せ回路による構成が必要となる。回路構成を図７－１に示す。この回路は，主としてCG／SG，SD，COR の各回路ブロックからなり，各ブロックの機能概略を表７－１に示す。この回路の特徴は，検査ビットの生成回路とシンドロームの生成回路が兼用化されていることであり，ゲート量の削減が図れる回路構成である。CG／SGは主として排他的論理的ゲートのツリー（Tree）構成を成しており，そのゲート量はHマトリクス中の'1'の数に比例する。また，SDは符号の機能に依存しているが，この回路のゲート量もHマトリクス中の'1'の数に影響を受ける。CORは入力情報ビット数分の排他的論理和ゲートを並列に並べた構成を有する。従って，符号化・復号化回路は主としてHマトリクス中の'1'の数に大きく依存しており，'1'の数が多い（大きい重み）程回路ゲート量は大きくなる傾向を有する。この観点から，同一機能，同一検査ビット数の下でHマトリクスの重みをできるだけ小さくすることは，符号化・復号化回路の最適化につながる。

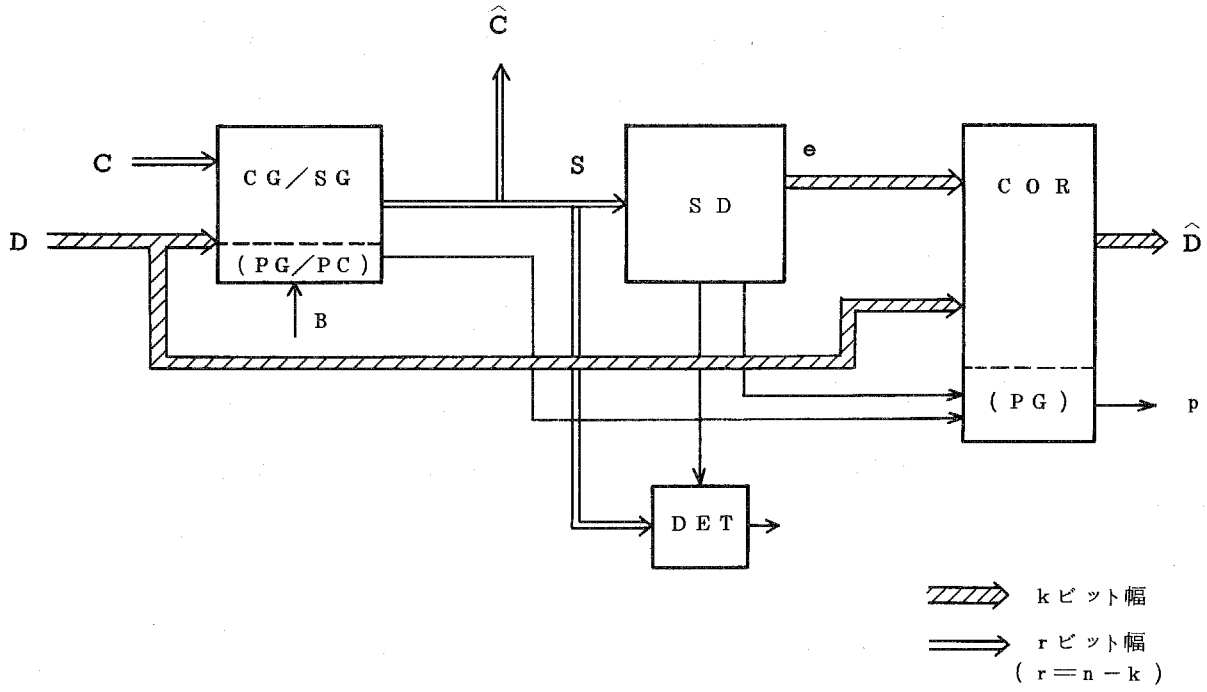


図７－１　（n，k）符号に対する符号化・復号化回路

表７－１　符号化・復号化回路の機能概要

略称	回路名	機能概要
CG／SG	チェックビット生成／シンドローム生成回路	符号化時には，検査ビット$\hat{C}$を生成し，復号時にはシンドロームＳを生成する。これを制御するのが信号Ｂであり，Ｂ＝'1'のとき入力検査データＣと入力データＤからシンドロームの生成を行い，Ｂ＝'0'のとき，入力検査データＣを遮断し，検査ビットの生成を行う。
SD	シンドロームデコード回路	シンドロームＳを入力して，符号に基づき，具体的な誤りビット位置を指示する誤りポインタｅを出力する。
COR	反転回路	シンドロームデコード回路の出力である誤りポインタに基づき，入力データの誤りビットを反転させる。
DET	誤り検出回路	誤りの状態を検出する。
PG	パリティ生成回路	復号後のデータに対して８ビット単位にパリティビットを付加する。
PC	パリティ検査回路	入力データＤに対するパリティ検査を行う。

この回路は符号変換回路（Ｔｒａｎｓｌａｔｏｒ）とみなす場合もある。[80]　すなわち，特定の誤り検出・訂正機能を有する符号とパリティ符号との間の変換である。ＣＰＵとＭＥＭ の間においては，一般に次のような符号の変換が行われる。ＣＰＵ中での処理は，主として８ビットのバイト単位に対するパリティ符号が付加され，ＣＰＵからはパリティ符号つきのデータがＭＥＭに送られる。一方ＭＥＭからのデータはパリティ符号つきのデータをＣＰＵに送り返す必要がある。従って，ＭＥＭに設置される符号化・復号化回路は，入力データに対しては，パリティ検査を行い，メモリ内部の符号に変換すると同時に，読出し後は訂正を行い正しいデータに対してパリティ符号の付加を行う必要がある。入力データに対するパリティ検査は，一般に検査ビットの生成時にＣＧ／ＳＧ回路中で兼用して行えるように工夫がなされる。またパリティ符号の生成は，シンドロームの生成とシンドロームのデコードの過程において，一般のデータに対するデコードと並列して行われ，同様にＣＯＲで反転動作が行われる。

次に，各符号機能，符号長に対する回路ゲート量，回路遅延の関係を示す。回路構成は図７－１の構成を統一的に使用する。ゲート量，ゲート段数の具体的な算定に当っては以下の基準の下で行う。

(1)　ＥＣＬ（Ｅｍｉｔｔｅｒ　Ｃｏｕｐｌｅｄ　Ｌｏｇｉｃ）論理により構成する。

(2) 入力数4までのOR，NORゲートは，1ゲート，1ゲート段数* とする。

(3) 2入力排他的論理和ゲートは，2ゲート，1ゲート段数とする。

(4) ファンアウト，布線長による信号遅延は，ここでは考慮しない。

符号としては，SEC−DED符号を基準にして各種のバイト系符号を対象とする。その結果を表7−2に示す。このうち，ゲート段数は復号に関係するゲート段数を示している。また，復号ゲート段数とゲート量の関係を図7−2に示す。

これから，以下の点が結論できる。

(i) SEC−DED−SbED符号の符号化・復号化回路は，従来使用されているSEC−DED符号とほぼ同一規模のゲート量，回路遅延で実現できる。

(ii) 情報ビット数，符号機能を増大させてもゲート量の増大に比べ，回路遅延の増加は高々2倍であり，高速な復号が可能である。

(iii) ゲート量は，符号機能よりむしろ情報ビット数，Hマトリクスの重みに主として依存する。

以上は組合せ回路による回路構成を示したが，新しい提案としてROMを論理素子として使用することにより，IC数，布線パターンの削減を狙った提案[81]，CPUにあるマイクロプログラム機能，ROM，演算ユニット等を使用して復号の一部を代行させる案[80]，またセル構造のLogic−in−memoryによる構成の提案[82]，等がある。しかし，いずれも高速性の点で組合せ回路による構成と比較して劣る。

---

* 1ゲート段数は1ゲートを信号が伝搬する信号遅延を表わす。

表７－２　符号化・復号化回路規模

情報ビット数（ビット）	符号機能	検査ビット数（ビット）	参照（reference）	CG/SG		SD		COR		DET	全体	
				ゲート量	ゲート段（SG）	ゲート量	ゲート段	ゲート量	ゲート段	ゲート量	ゲート量	ゲート段（復号）
32	SEC-DED	7	文献(34)Fig.4	199	5	87	2	64	1	17	367	8
	SEC-DED-S4ED	8	図5-11	152	5	58	3	64	1	20	294	9
64	SEC-DED	8	文献(34) Fig.5	424	6	154	2	128	1	19	725	9
	SEC-DED-S4ED	9	図3-11	339	6	228	3	128	1	26	721	10
	SEC-DED-S4ED	9	図3-20	313	6	230	3	128	1	22	693	10
	S2EC	8	文献(42) Fig.1	376	6	410	4	128	1	3	917	11
	S2EC	8	図4-5	552	7	553	4	128	1	3	1236	12
	S4EC	12	図4-6	452	6	429	5	128	1	4	1013	12
	S2EC-D2ED	12	図5-20	552	6	584	4	128	1	7	1271	11
	S4EC-D4ED	16	図5-23	464	6	440	5	128	1	6	1038	12
128	SEC-DED	9	文献(76) Fig.3	953	7	286	2	256	1	21	1516	10
	S2EC	10		742	7	732	4	256	1	4	1734	12
	S4EC	12		804	7	721	5	256	1	4	1785	13
	S2EC-D2ED	12	図4-16(b)（2列のぞく）	1270	8	1326	5	256	1	5	2857	14
	S4EC-D4ED	16	図5-21	1120	7	1532	5	256	1	6	2914	13

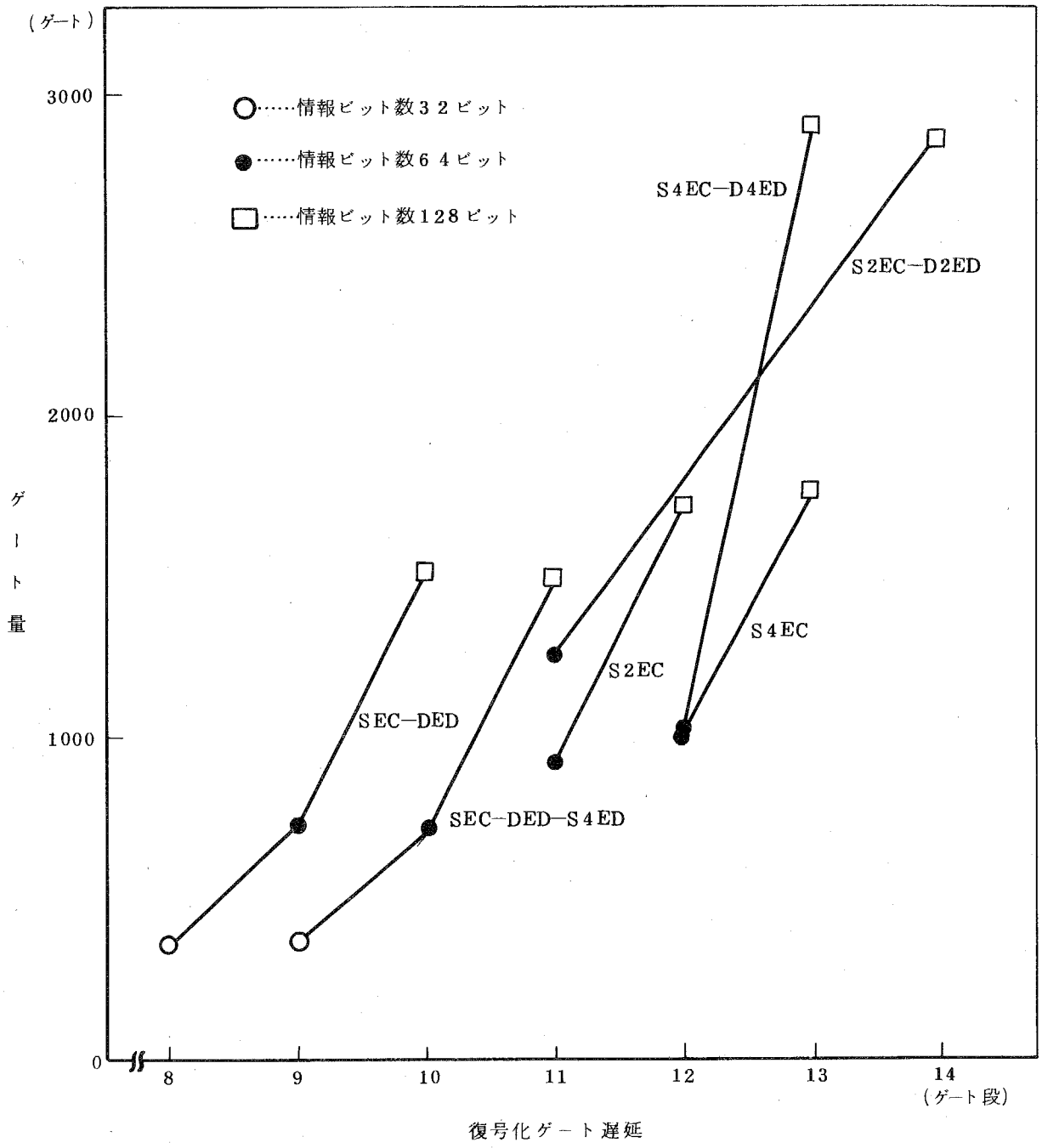


図7－2　符号化・復号化回路のゲート量と復号化ゲート遅延

7－3　自己検査論理による高信頼化

符号化・復号化回路自身の故障は誤ったデータを出力することになり，誤処理をもたらす危険性を有する。この観点から，この回路の高信頼化は重要である。本節では，符号化・復号化回路に対して自己検査論理[8][83]～[90]を適用した自己検査性検査回路を設置して，通常動作実行中に回路故障を迅速に検出する手法について述べる。

自己検査性検査回路とは，対象とする符号化・復号化回路は勿論，検査回路自身の故障も検出する回路であり，このような回路を設けることにより被検査回路（符号化・復号化回路）を自己検査論理で構成したと同等の効果が，小さい金物量増加で得られることが期待できる。この場合，次に示す2点が問題となる。

(1)　故障検出能力が高く，ゲート量増加の小さい最適な検査手法をいかに求めるか。

(2)　その検査手法に対して，いかに自己検査論理で構成するか。

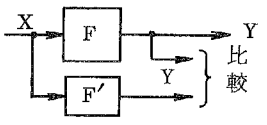
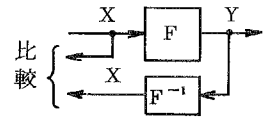
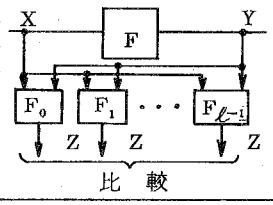
従来この回路に対しては，回路全体に対して自己検査性検査回路を設置する手法[89]が提案されているが，ゲート量増加などに問題を有していた。本節では，この回路を主要機能に分割し，各機能に対して新しい自己検査性検査論理を適用する構成としている。その結果，単一故障を完全に検出できる自己検査性検査回路を実用的なゲート量増加で構成できる結果を得ている。[27]

7－3－1　検査手法と自己検査論理

〔1〕　組合せ回路に対する検査手法[6]

対象とする符号化・復号化回路は，高速性の観点から組合せ回路により構成している。一般に，組合せ論理回路に対する検査手法は種々考案されているが，大別して3種の手法に分類できる。検査対象の論理回路をF，その入力をX，出力をYとしたとき，その検査手法を表7－3に示す。

表７－３　組合せ回路に対する検査手法

手法	構成図	手法の説明	検査速度	物金増加
1		出力を別回路F′によつて求め，その結果と出力Yとを比較する手法	速い	多
2		出力Yを入力としてXを求める逆回路F^{-1}によつて求めた結果とXとを比較する方法	遅い	中～多
3		入力Xと出力Yを用いて，複数個の同一結果Zを出力させ，それらを互いに比較する手法	中	少～中

検査手法１は，出力の線数（ｍ）が入力線数（ｎ）より小さい場合に比較的適した手法である。FとF'が同一の場合には，２重化による検査となる。この手法の特徴は，（ｉ）検査速度が速い，（ii）ゲート量増加が大きい，（iii）いかなる論理回路にも適用できる，点にある。この手法の簡略化としては，出力の一部のみを比較する手法が考えられる。

検査手法２は，入力線数（ｎ）が出力線数（ｍ）より小さい場合（例えば，デコード回路）に適した手法である。この手法の特徴は，（ｉ）出力から逆に入力を求めて比較するため検査速度が遅い，（ii）回路によってはゲート量増加を小さくできる，（iii）F^{-1}が存在せず適用できない場合もある，点にある。検査能力を下げて簡略化した検査としては，入力の一部のみを比較する手法が考えられる。

検査手法３は，共通論理Ｚをうまく選ぶことにより，効果的な検査を行うことができる。本手法の特徴は，（ｉ）Ｚの選び方によっては非常に少ないゲート量増加で，検査速度の速い検査回路実現の可能性がある，（ii）適当なＺが存在しない可能性がある，点にある。

〔２〕　自己検査論理 (8)(85)

ここでは，自己検査論理としてTotally Self－Checking（T.S.C.）の考え方について示す。論理回路をＦとし，Ｆに対する出力のうち検査を行うことによって正常であるとみなされる出力を符号空間内の出力と呼び，これらの集合をＹとする。また，Ｆに対する入力のうち，Ｆが正常であれば出力がＹの要素となる入力を符号空間内の入力と呼び，これらの集合をＸとする。この関係を図７－３（ａ）に示す。次にＦ内に故障が生じたと仮定する。この場合の

入力Xに対する出力は，図７－３（ｂ）に示すように次の３通りが考えられる。

① 符号空間外の出力（検出できる誤り出力）

② 正しい出力（故障がマスクされた状態）

③ 符号空間内の誤った出力（検出できない誤り出力）

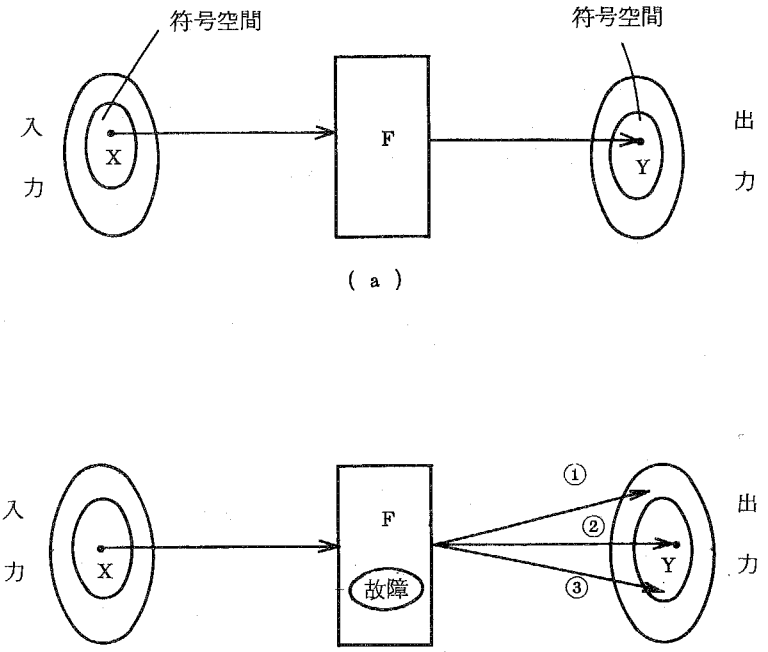


図７－３　自己検査論理の考え方

（定義７－１）[8][85]

Ｓｅｌｆ－Ｔｅｓｔｉｎｇ　：　Ｆ内のいかなる故障に対しても①で示した符号空間外の出力となる入力が，符号空間内入力Ｘの中に少くとも１個存在するなら，ＦはＳｅｌｆ－Ｔｅｓｔｉｎｇである。

Ｆａｕｌｔ－Ｓｅｃｕｒｅ　：　Ｆ内のいかなる故障に対しても③で示した符号空間内の誤った出力をすることが全くなければ，ＦはＦａｕｌｔ－Ｓｅｃｕｒｅである。

Ｔｏｔａｌｌｙ　Ｓｅｌｆ－Ｃｈｅｃｋｉｎｇ＊　：　ＦがＳｅｌｆ－Ｔｅｓｔｉｎｇであり，かつＦａｕｌｔ－Ｓｅｃｕｒｅであれば，ＦはＴｏｔａｌｌｙ　Ｓｅｌｆ－Ｃｈｅｃｋｉｎｇである。

□

＊　符号空間内入力Ｘに対してＳｅｌｆ－Ｔｅｓｔｉｎｇであり，符号空間内入力の部分空間Ｉ∈Ｘに対して，Ｆａｕｌｔ－Ｓｅｃｕｒｅであるとき，Ｐａｒｔｉａｌｌｙ　Ｓｅｌｆ－Ｃｈｅｃｋｉｎｇと定義される。[8]

検査結果を出力するゲートの縮退故障を検出するには，少くとも２本以上の出力によって検査結果を示す必要がある。実際には，２本の出力（C_0，C_1）によって，その出力が（０，１）または（１，０）となる場合と，（０，０）または（１，１）となる場合の２状態で被検査回路に対する検査結果を示す。(90)

Ｆを検査回路とするとき，次に示す性質が要求される。

（定義７－２）(8)

Code Disjoint ： 回路に符号空間内の入力Ｘが入力したとき，符号空間内の出力Ｙが出力し，符号空間外の入力が入力したとき，符号空間外の出力が出力するとき，この回路はCode Disjointであると定義する。

□

７－３－２　符号化・復号化回路に対する検査

本回路に検査機能を付与する場合，回路全体に対して１個の検査機能を与える構成と，回路を分割して各機能に対して検査機能を与える構成が考えられる。前者の構成による手法として，誤り訂正後のデータに対するパリティ検査(89)，あるいは誤り訂正後のデータを用いて再度シンドロームを求め，それらの値がすべて'0'になっていることを検査する手法(49)(50)が提案されている。これらの手法は，どのような符号にも適用できるが，ゲート量増加の点で問題を有していた。そこで，ここでは回路を分割して，各機能に対して検査機能を与える構成を選び，より小さいゲート量増加で構成できる自己検査性検査回路の提案を行う。このような構成であれば，この回路をＬＳＩ化する際に各機能毎に個々のＬＳＩで構成した場合にも有利となる。ここでは，符号化・復号化回路の大半のゲート数を占めるCG／SG，ＳＤを中心にT.S.C.の条件を満足し，単一故障を完全に検出できる検査論理についてのべる。

７－３－３　CG／SGに対する検査論理(27)

本回路に対する検査手法は，表７－３に示した手法３を適用するとよい結果を得ることができる。手法１を適用すると同一回路と更に比較回路を設けることから，ゲート増加が１００％を越え，また手法２は適用不可能である。ここでは，手法３に基づく検査手法とその自己検査論理による構成についてのべる。なお，CG，SGのいずれの回路に対する検査手法も本質的には同一であることから，ここではSGについてのべる。

〔1〕 検査手法

Hマトリクスを H，入力データを D とするとき，SG は $S = D \cdot H^T$（H^T は H の転置行列）なるパリティ結果 S（シンドローム）を生成する回路として表わせる。SG の具体的回路において，その単一故障は単一のパリティ検査結果（1ビットのシンドローム）にのみ影響する。従って，ここでは GF(2) 上で表現した H マトリクスを用いる。

$$H = [\, h_{i,j} \,]_{r \times n} \qquad \begin{matrix} i = 0,\ 1,\ \cdots,\ r-1 \\ j = 0,\ 1,\ \cdots,\ n-1 \end{matrix}$$

$$D = (\, d_0,\ d_1,\ \cdots,\ d_{n-1} \,)_{1 \times n} \qquad (7-1)$$

$$S = (\, s_0,\ s_1,\ \cdots,\ s_{r-1} \,)_{1 \times r}$$

このとき，S，D，H の間に以下の定理が成立する。

（定理 7－1）

r 個の要素からなる S を重複を許さず ℓ 個（$2 \leq \ell \leq r$）のグループに分割し，各々のグループ中の要素の和（mod. 2 和）を S_i（$i = 0,\ 1,\ \cdots,\ \ell-1$）とすれば，任意の n 次の行ベクトル A に対して

$$D \cdot A^T = S_i \oplus D \cdot f_i^T \qquad (7-2)$$

なる n 次の行ベクトル f_i が存在する。また，S の要素の1個に矛盾が生じれば，$D \cdot A^T \neq S_j \oplus D \cdot f_j^T$ なる1個の j（$0 \leq j \leq \ell - 1$）が存在する。 □

（証明）

r 個の要素からなる S を ℓ 個のグループに分割したとき，各グループ内の要素に相当する H マトリクス行ベクトルの和を各々 $g_0,\ g_1,\ \cdots,\ g_{\ell-1}$ とすると，次の関係が成立する。

$$S_i = D \cdot g_i^T \qquad (7-3)$$

そこで，任意の n 次の行ベクトル A に対して $f_i = A \oplus g_i$ なる n 次の行ベクトル f_i を選べば，

$$A = g_i \oplus f_i \qquad (7-4)$$

より，両辺にDをかけて

$$D \cdot A^T = D \cdot g_i^T \oplus D \cdot f_i^T = S_i \oplus D \cdot f_i^T \qquad (7-5)$$

が成立する。よって，（7－2）式を満足するf_iが存在する。

また，矛盾したSの要素は必ずℓ個のグループのいずれか1個に含まれる。そのグループをj番目のグループとすれば，S_jの値が反転する。一方，$D \cdot f_j^T$及び$D \cdot A^T$の値は，Sの要素に影響されない。よって，$D \cdot A^T \neq S_j \oplus D \cdot f_j^T$なるjが1個存在する。

（証明終り）

この定理から，$S_i \oplus D \cdot f_i^T$（$i=0, 1, \cdots, \ell-1$）なるℓ個の値がすべて$D \cdot A^T$の値に一致しているか否かによって，SGの単一故障を検出できることになる。すなわち，入力Dと出力Sとから，（7－2）式に示す共通論理を（$\ell+1$）個作り，これらが等しいか否かを調べればよい。

〔2〕　自己検査論理の適用

定理7－1で求めた検査論理を自己検査論理により構成してみよう。（7－2）式はℓ個の（$S_i \oplus D \cdot f_i^T$）と1個の$D \cdot A^T$の計（$\ell+1$）個の間に等号関係が成立することを示している。一般に，複数個の値が一致していることの必要十分条件は，それらの論理和の値と論理積の値が等しいことである。そこで，（7－2）式に示す論理に対してはこの考えを用いて次のような自己検査論理を求めることができる。

（定理7－2）

（7－2）式に示す論理に対して

$$\begin{cases} C_0 = \{ \bigcup_{i=0}^{\ell-1} (S_i \oplus D \cdot f_i^T) \} \cup D \cdot A^T \\ C_1 = \overline{\{ \bigcap_{i=0}^{\ell-1} (S_i \oplus D \cdot f_i^T) \} \cap D \cdot A^T} \end{cases} \qquad (7-5)$$

∪；論理和　　∩；論理積

なる検査出力（C_0，C_1）を構成すれば，この検査回路はT.S.C.の条件を満足する。ここで，

$$(C_0, C_1) = \left[\begin{array}{l}(1, 0)\text{または}(1, 0)\text{；誤りなし}\\(0, 0)\text{または}(1, 1)\text{；誤り検出}\end{array}\right] \text{とする。} \qquad \square$$

（証明）

（7－5）式による論理回路への入力は，DとSであるが，Dはすべての値，SはS＝D・H^TなるSが符号空間内の入力である。符号空間内出力（C_0，C_1）は（1，0）と（0，1）である。定理7－1から，正しいS，Dに対して（C_0，C_1）＝（1，0）または（0，1）を与える。一方，Sに矛盾が生じていれば（C_0，C_1）＝（1，1）または（0，0）が生ずる。これから，この検査回路はCode Disjointの性質を満足する。

この回路は（$S_i \oplus D \cdot f_i^T$）なるℓ個の値を求める回路，$D \cdot A^T$を求める回路（いずれも排他的論理和回路ですべて構成される。），それらの値の論理和を求めるOR回路，およびNAND回路とから構成される。Dがすべての値をとりうるので，Sの要素も'1'，'0'の双方の値をとることができる。よって，この論理回路を構成するすべてのゲートは符号空間内の入力に対して，'1'と'0'の双方の出力値をとることができる。従って，単一のゲートに故障が生じた場合，それが'1'，'0'のいずれの縮退故障であっても，そのゲート出力を誤った値とするような入力が符号空間内に必ず存在することになる。また，この誤った出力は最終ゲートまで伝搬し，（C_0，C_1）＝（1，1）または（0，0）と符号空間外の出力となってその誤りを検出できる。以上から，Self-Testingの条件を満足している。

一方，故障によってC_0，C_1の双方の値が同時に反転するなら，符号空間内の誤った出力となり検出できない。これは，OR回路とNAND回路の双方に故障が生じた場合か，（$S_i \oplus D \cdot f_i^T$）及び$D \cdot A^T$を出力する回路のすべてに故障が生じて（$\ell+1$）個の出力値がすべて反転した場合にのみ起り得る。しかし，単一故障によってこのような状態になることはあり得ない。よって，Fault－Secureの条件も満足している。以上から，（7－5）式はT.S.C.の条件を満足する。

（証明終り）

ところで，定理7－1から，SGの故障によってSの要素が反転した場合，$S_i \oplus D \cdot f_i^T$（i＝0，1，…，$\ell-1$）のℓ個のうちの1個の値が必ず反転する。よって，これらのℓ個の値がすべて一致しているか否かを検査すれば$D \cdot A^T$の値との一致を検査する必要はなくなる。よって，（7－5）式は以下のように簡略化できる。

$$\begin{cases} C_0 = \bigcup_{i=0}^{\ell-1} (S_i \oplus D \cdot f_i^T) \\ C_1 = \overline{\bigcap_{i=0}^{\ell-1} (S_i \oplus D \cdot f_i^T)} \end{cases} \qquad (7-6)$$

〔3〕 最適な自己検査性検査回路

本検査回路を構成するゲート数は，グループ分割の方法，Aの選び方によって異なる。ゲート数が最少となるとき一般に検査速度も速くなり，最適な論理構成を与えることができる。具体的には（7−6）式の$S_i \oplus D \cdot f_i^T$において，$D \cdot f_i^T$を構成するためのゲート数を少なくすればよい。すなわち，ℓ個の行ベクトルf_0，f_1，f_2 …，$f_{\ell-1}$ の重みの和を最小にすればよい。

① Aの決定法

行ベクトルf_i，A，g_iの間には（7−4）式が成立することから，Aとg_i の各列の要素を比べたとき，一致していない要素の数がf_iの重みに等しいことがわかる。そこで，f_0，f_1，…，$f_{\ell-1}$ の各行の重みの和を最小とするには，グループ分割によって得られたg_0，g_1，…，$g_{\ell-1}$の各列の要素に対して，Aと一致する要素の数を最大にするようにAを選べばよい。つまり，

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{\ell-1} \end{bmatrix}_{\ell \times n} \qquad (7-7)$$

なるℓ行n列の行列Gのi第目の列をG_iとし，その重みを$W(G_i)$で表わせば，Aの要素a_iは次式で決定できる。

$$\begin{cases} a_i = 1 & \cdots \quad W(G_i) > \lceil \ell/2 \rceil \\ a_i = 0 & \cdots \quad W(G_i) \leq \lceil \ell/2 \rceil \end{cases} \qquad (7-8)$$

$$i = 0,\ 1,\ \cdots,\ n-1$$

$\lceil x \rceil$; x を越えない最大整数を表わす。

② グループ分割法

ゲート数を最小とするグループ分割について次の定理 7 － 3 が成立する。

（定理 7 － 3 ）

2 あるいは 3 のグループ分割中に検査回路の構成ゲート数を最小とする分割が存在する。 □

証明は容易であるので省略する。2 グループが有利か，3 グループが有利かはHマトリクスにより異なる。マトリクスの各列の重みが偶数のものが多ければ 2 グループ分割，逆に奇数重みの列が多ければ 3 グループ分割が有利となる。

7 － 3 － 4　SDに対する検査論理 (27)

本回路は，SGによって生成されたシンドロームを入力としてHマトリクスに従ってデコードを行い，誤りデータ位置を指摘する誤りポインタを出力する回路である。

Hマトリクス，シンドローム，誤りポインタをそれぞれ次のように表わす。

$$H = \begin{bmatrix} H_0 \\ H_1 \\ \vdots \\ H_{r-1} \end{bmatrix} = \left[h_{i,j} \right]_{r \times n} \qquad \begin{matrix} i = 0,\ 1,\ \cdots,\ r-1 \\ j = 0,\ 1,\ \cdots,\ n-1 \end{matrix} \qquad (7-9)$$

$$S = [\, s_0\ s_1 \cdots s_{r-1} \,]_{1 \times r}$$

$$E = [\, e_0\ e_1 \cdots e_{n-1} \,]_{1 \times n}$$

s_i , e_j はｂ次の行ベクトル， $h_{i,j} \in GF(2^b)$ とする。このとき，次の関係が成立する。

$$S = E \cdot H^T \qquad (7-10)$$

一方，誤訂正を生じさせない誤りポインタ E' （$\neq E$）に対しては

$$S \neq E' \cdot H^T$$

となる。これから，容易に表７－３に示した手法２による検査手法が適用できる。すなわち，（７－１０）式の右辺を用いて誤りポインタから逆にシンドロームを求め，入力のシンドロームＳと比較すればよい。しかし，この手法は手法１の２重化による手法と同様，大幅なゲート増加をまねく欠点を有する。そこで，この回路に付しても手法３による考え方を導入する。

〔１〕　手法３による検査手法とその自己検査論理

CG／SGに対する検査の考え方と同様に，共通結果を出力させるため，Ｈマトリクスの各行に対して基準となるマトリクスＡを定める。Ｓ，Ｅ，Ｈ及びＡの間には次の定理が成立する。

（定理７－４）

Ａを次に示す $a_k \in GF(2^b)$（ｋ＝0，1，…，ｎ－１）を要素とするマトリクスとする。

$$A = [\, a_0,\ a_1,\ \cdots,\ a_{n-1} \,] \qquad (7-11)$$

Ａを用いて，F_i を

$$\begin{aligned} F_i &= [\, f_{i,0},\ f_{i,1},\ \cdots,\ f_{i,n-1} \,] \\ &= A \oplus [\, h_{i,0},\ h_{i,1},\ \cdots,\ h_{i,n-1} \,] \end{aligned} \qquad (7-12)$$

と定義すれば，$s_i = E \cdot H_i^T = E \cdot [\, h_{i,0},\ h_{i,1},\ \cdots,\ h_{i,n-1} \,]^T$ より

$$E \cdot A^T = s_i \oplus E \cdot F_i^T \qquad (i = 0,\ 1,\ \cdots,\ r-1) \qquad (7-13)$$

が成立する。ここで，$h_{i,j}$，E，s_i はそれぞれ（7－9）式で定義したHマトリクス要素，誤りポインタ，シンドロームである。

符号として誤訂正を与えない範囲で，Eの要素に矛盾が生じれば，

$$E \cdot A^T \neq s_j \oplus E \cdot F_j^T \qquad (7-14)$$

となる少くとも1個の $j \in \{0, 1, \cdots, r-1\}$ が成立する。 □

（証明）

（7－12）式の両辺にEを掛けて

$$E \cdot A^T = E \cdot [h_{i,0}, h_{i,1}, \cdots, h_{i,n-1}]^T \oplus E \cdot F_i^T$$

を得る。（7－10）式より，

$$E \cdot A^T = s_i \oplus E \cdot F_i^T \qquad (7-15)$$

が成立する。今，符号として誤訂正を与えない範囲でEの要素に矛盾が生じてE'（$\neq E$）を出力したとき，

$$E' \cdot A^T = s_i \oplus E' \cdot F_i^T \qquad (7-16)$$

が，$i=0, 1, \cdots, r-1$ なるすべてのiについて成立すると仮定する。（7－10）式，（7－12）式を用いて（7－16）式のA，F_i，s_i を消去すれば，次式を得る。

$$E \cdot [h_{i,0}, \cdots, h_{i,n-1}]^T = E' \cdot [h_{i,0}, \cdots, h_{i,n-1}]^T \qquad (7-17)$$

Hマトリクスの性質より，$[h_{i,0}, \cdots, h_{i,n-1}] \neq O$ なるiは必ず1個以上存在するから，$E = E'$ となる。これは，$E \neq E'$ の仮定に矛盾する。よって，少くとも

$$E' \cdot A^T \neq s_j \oplus E' \cdot F_j^T \qquad (7-18)$$

なるjは1個存在する。

（証明終り）

定理7－4から，$s_i \oplus E \cdot F_i^T$（i＝0，1，…，r－1）なるr個の値がすべて$E \cdot A^T$に一致しているか否かにより，正常性を検査することができる。T.S.C.条件を有する検査回路は，定理7－2と同様の考え方で次のように構成できる。

$$\begin{cases} C_0 = \{ \bigcup_{i=0}^{r-1} (s_i \oplus E \cdot F_i^T) \} \cup E \cdot A^T \\ C_1 = \{ \bigcap_{i=0}^{r-1} (s_i \oplus F \cdot F_i^T) \} \cap E \cdot A^T \end{cases} \qquad (7-19)$$

∪；論理和，　∩；論理積

ここで，$(C_0, C_1) = \left\{ \begin{array}{l} (1, 0) \text{または} (0, 1) \text{；誤りなし} \\ (0, 0) \text{または} (1, 1) \text{；誤り検出} \end{array} \right\}$とする。

(C_0, C_1)を出力するために要するゲート数は，同様にAの選び方によって異なる。ゲート数を最小とするためには，（7－19）式から$E \cdot F_i^T$，$E \cdot A^T$を構成するためのゲート数を最小とすればよい。よって，Hマトリクスの各列に対してa_j，（$f_{0,j}$，$f_{1,j}$，…，$f_{r-1,j}$）の重みの和が最小となるようにa_jを選べばよい。SEC－DED符号を用いた符号化・復号化回路に適用する場合，a_jの選び方は，（7－8）式で示したと同様にG_jをHマトリクスのj番目の列ベクトルとみなして，最適なAを求めることができる。SbEC符号等，GF（2^b）上の要素で構成されたHマトリクスを使用する符号の場合は，Hマトリクスの各列ごとに，その中で最も多く用いられている要素をAの要素として選べばよい。

〔2〕　検査論理の簡素化

〔1〕で示した手法3による検査論理に対して，さらにゲート数削減を図ることを考える。その手法として，故障検出能力を低下させない範囲で，本検査論理に用いる排他的論理和を論理和に置換することを考える。排他的論理和ゲートを2～3ゲート，論理和ゲートを1ゲートと換算すれば，ゲート数削減に大きな効果を与える。ここでは，SEC－DED符号，SbEC符号等の単一誤り訂正符号を対象とする。

（定理7－5）

（7－12）式に定義したF_i（i＝0，1，…，$r-1$）から2個を選び，それらを，F_j，F_k，

j，k∈{0，1，…，$r-1$}とする。また，これらのα番目とβ番目の要素を{$f_{j,\alpha}$，$f_{j,\beta}$}$\in F_j$，{$f_{k,\alpha}$，$f_{k,\beta}$}$\in F_k$とする。このとき，

$$\left.\begin{array}{ll} f_{j,\alpha}=0, & f_{j,\beta}\neq 0 \\ f_{k,\alpha}\neq 0, & f_{k,\beta}=0 \end{array}\right\} \quad \left(\begin{array}{l} 0\leq \alpha,\ \beta \leq n-1,\quad \alpha\neq\beta \\ 0\leq j,\ k\leq r-1,\quad j\neq k \end{array}\right) \qquad (7-20)$$

なる関係が成立するようなα，βの集合Gを作る。このようなGの数をδとしたとき，誤りポインタEの要素に矛盾があれば，

$$\sum_{\delta}^{\oplus}\left(\bigcup_{e\in G} e_e \cdot a_e^T\right) \neq s_m \oplus \sum_{\delta}^{\oplus}\left(\bigcup_{e\in G} e_e \cdot f_{m,e}^T\right) \qquad (7-21)$$

なるm（$0\leq m\leq r-1$）が存在する。ここで，e_e，s_m，a_eは（7－9）式，（7－11）式に定義した要素である。 □

（証明）

（7－21）式は（7－13）式において示した排他的論理和の一部を論理和に置換したものである。Eが正常であれば，2個以上の誤りポインタが同時に出力することはないので，排他的論理和は論理和に置換することができる。よって，このときi∈{0，1，…，r－1}に対して次式が成立する。

$$\sum_{\delta}^{\oplus}\left(\bigcup_{e\in G} e_e \cdot a_e^T\right) = s_i \oplus \sum_{\delta}^{\oplus}\left(\bigcup_{e\in G} e_e \cdot f_{i,e}^T\right) \qquad (7-22)$$

今，α番目のデータが誤り，これによって誤りポインタe_αが正しく出力されたとする。この場合，（7－22）式より，j，kに対して（7－20）式を考慮すれば，

$$e_\alpha \cdot a_\alpha^T = s_j = s_k \oplus e_\alpha \cdot f_{k,\alpha}^T \qquad (7-23)$$

が成立する。この状態でSDに故障が生じ，β番目の誤りポインタe_βが同時に誤って出力した場合を考える。このとき（7－20）式の条件を満足するj，kに対して次のようになる。

$$s_j \oplus (e_\alpha \cdot f_{j,\alpha}^T \cup e_\beta \cdot f_{j,\beta}^T) = s_j \oplus e_\beta \cdot f_{j,\beta}^T \qquad (7-24)$$

$$s_k \oplus (e_\alpha \cdot f_{k,\alpha}^T \cup e_\beta \cdot f_{k,\beta}^T) = s_k \oplus e_\alpha \cdot f_{k,\alpha}^T \qquad (7-25)$$

（7－23）式後半2項から，$s_j = s_k \oplus e_\alpha \cdot f^T_{k,\alpha}$ の関係を代入すれば，（7－24）式と（7－25）式は等しい関係にない。よって，m＝j あるいはm＝k なるmに対して（7－21）式が成立する。

（証明終り）

以上では，j，kの2組の関係についての定理を示したが，拡張して，例えば，F_j，F_k，F_ℓ，$(j,k,\ell) \in \{0, 1, \cdots, r-1\}$ の間で以下の関係が成立したとする。

$$\left.\begin{array}{lll} f_{j,\alpha} = 0, & f_{j,\beta} \neq 0, & f_{j,r} \neq 0 \\ f_{k,\alpha} \neq 0, & f_{k,\beta} = 0, & f_{k,r} \neq 0 \\ f_{\ell,\alpha} \neq 0, & f_{\ell,\beta} \neq 0, & f_{\ell,r} = 0 \\ \left(\begin{array}{l} 0 \leq \alpha,\ \beta,\ r \leq n-1 \\ 0 \leq j,\ k,\ \ell \leq r-1 \end{array}\right) & & \end{array}\right\} \qquad (7-26)$$

このとき，F_j，F_k，F_ℓ のどの2組を選んでも（7－20）式に示したと同様の条件が成立することから，(α, β, r) を集合Gとして拡張することができる。また，ここではF_i（i＝0，1，…，r－1）からの要素を選んだが，AとF_j，$j \in \{0, 1, \cdots, r-1\}$ との間においても，

$$\left.\begin{array}{ll} a_\alpha = 0, & a_\beta \neq 0 \\ f_{j,\alpha} \neq 0, & f_{j,\beta} = 0 \end{array}\right. \qquad \left(\begin{array}{l} 0 \leq \alpha,\ \beta \leq n-1,\ \alpha \neq \beta \\ j \in \{0, 1, \cdots, r-1\} \end{array}\right) \qquad (7-27)$$

が成立する場合には同様のことが言える。

以上，7－3－3におけるSGに対する検査，7－3－4におけるSDに対する検査は，いずれも表7－3に示す手法3による方法を採用した。まとめとして，これらに対する検査論理等の比較を表7－4に示す。

表７－４　手法３によるSG，SDに対する検査の比較

機能	入力	出力	論理式	グループ分割	共通ベクトルA	検査論理条件	単一故障検出条件	T. S. C.条件	検査論理の簡素化
SG	D	S	$S=D\cdot H^T$	ℓ（２or３）個 （重複を許さず） $H=\begin{bmatrix} g_0 \\ \vdots \\ g_{\ell-1} \end{bmatrix}_{\ell\times n}$	$A=f_i\oplus g_i$ $i=0,1,\cdots,\ell-1$	$D\cdot A^T$ $=S_i\oplus D\cdot f_i^T$	（S_j に誤り） $D\cdot A^T$ $\neq S_j\oplus D\cdot f_j^T$ なる１個のjが存在	$C_0=\bigcup_{i=0}^{\ell-1}(S_i\oplus D\cdot f_i^T)$ $C_1=\overline{\bigcap_{i=0}^{\ell-1}(S_i\oplus D\cdot f_i^T)}$	• Aの選び方 $a_i=1\cdots W(G_i)>\lceil \ell/2 \rceil$ $a_i=0\cdots W(G_i)\leq\lceil \ell/2 \rceil$ • グループ分割 $\ell=2$ or 3
SD	S	E	$S=E\cdot H^T$	r個	$A=H_i\oplus F_i$ $i=0,1,\cdots,r-1$	$E\cdot A^T$ $=s_i\oplus E\cdot F^T$	（Eに誤り） $E\cdot A^T$ $\neq s_j\oplus E\cdot F^T$ なる少くとも１個のjが存在	$C_0=\bigcup_{i=0}^{r-1}(s_i\oplus E\cdot F_i^T)\cup E\cdot A^T$ $C_1=\overline{\bigcap_{i=0}^{r-1}(s_i\oplus E\cdot F_i^T)\cap E\cdot A^T}$	• 排他的論理和 →論理和へ置換 • Aの選び方

7－3－5　その他の回路に対する検査論理

符号化・復号化回路として，以上に示したCG／SG，SD以外にCOR（データ反転回路）とPG（パリティ生成回路）がある。

そこで，CORとPGに対して，その内の故障に対する検査論理を求めることができる。すなわち，CORの出力から求めたパリティビットP'とPGの過程から求めたパリティビットPとを比較すればよい。そこで

$$\begin{cases} C_0 = P' \\ C_1 = \overline{P} \end{cases} \qquad (7-28)$$

なる2値（C_0，C_1）を出力する検査回路により，自己検査性検査回路を構成することができる。（7－28）式では，（7－5）式，（7－19）式の場合と同様，（C_0，C_1）＝（1，0）または（0，1）の場合正しいと判定し，（0，0）または（1，1）の場合誤りと判定する。

7－3－6　具体例とその結果[27]

SG，SDを主体に手法3による検査論理を適用した場合に，具体的な検査回路の構成とその評価を示す。具体例として，SEC－DED符号を用いた場合の簡単な例を用いて示す。

〔1〕　具体例

次に示す（22，16）SEC－DED符号を対象とする。

$$H = \left[\begin{array}{cccccccccccccccc:cccccc} 1&0&0&1&0&1&1&0&1&0&1&1&1&1&0&1 & 1&0&0&0&0&0 \\ 0&1&1&1&1&1&0&1&0&1&1&0&1&0&1&1 & 0&1&0&0&0&0 \\ 0&1&1&0&0&1&1&0&1&1&1&0&1&1&0&0 & 0&0&1&0&0&0 \\ 1&1&0&1&1&1&0&1&1&0&0&1&1&0&0&0 & 0&0&0&1&0&0 \\ 0&1&0&1&1&0&1&0&1&1&1&1&1&0&1&1 & 0&0&0&0&1&0 \\ 1&1&1&1&0&1&0&1&1&0&1&0&0&1&1&0 & 0&0&0&0&0&1 \end{array}\right] \begin{matrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{matrix} \qquad (7-29)$$

$$D = \left[\begin{array}{cccccccccccccccc:cccccc} d_0 & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 & d_7 & d_8 & d_9 & d_{10} & d_{11} & d_{12} & d_{13} & d_{14} & d_{15} & c_0 & c_1 & c_2 & c_3 & c_4 & c_5 \end{array}\right]$$

① SGに対する自己検査性検査回路

手法1を適用すると，検査回路は165ゲートとなる。SGのゲート数が135ゲートであるから，ゲート増加比は122％となる。このとき，ゲート増加比は次のように定義される。

$$\text{ゲート増加比} = \frac{\text{検査回路のゲート数}}{\text{被検査対象回路のゲート数}} \times 100(\%) \qquad (7-30)$$

手法3を適用した場合について示す。SGの出力を$s_0 \sim s_5$としたとき，7－3－3〔3〕で示した最適化により，

$$\begin{aligned} S_0 &= s_0 \oplus s_1 \\ S_1 &= s_2 \oplus s_3 \\ S_2 &= s_4 \oplus s_5 \end{aligned} \qquad (7-31)$$

のグループ分割を得る。これから，g_0，g_1，g_2，Aおよびf_0，f_1，f_2は次のように求められる。

②

$$\left.\begin{array}{l} g_0 = [\,1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0 \mid 1\ 1\ 0\ 0\ 0\ 0\,] \\ g_1 = [\,1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0 \mid 0\ 0\ 1\ 1\ 0\ 0\,] \\ g_2 = [\,1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1 \mid 0\ 0\ 0\ 0\ 1\ 1\,] \\ A = [\,1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0 \mid 0\ 0\ 0\ 0\ 0\ 0\,] \\ f_0 = [\,0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0 \mid 1\ 1\ 0\ 0\ 0\ 0\,] \\ f_1 = [\,0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0 \mid 0\ 0\ 1\ 1\ 0\ 0\,] \\ f_2 = [\,0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1 \mid 0\ 0\ 0\ 0\ 1\ 1\,] \end{array}\right\} \qquad (7-32)$$

以上から，（7－2）式は次のようになる。

$$\begin{aligned} D \cdot A^T &= (s_0 \oplus s_1) \oplus (d_1 \oplus d_8 \oplus d_{14} \oplus c_0 \oplus c_1) \\ &= (s_2 \oplus s_3) \oplus (d_3 \oplus d_{10} \oplus c_2 \oplus c_3) \\ &= (s_4 \oplus s_5) \oplus (d_5 \oplus d_{12} \oplus d_{15} \oplus c_4 \oplus c_5) \end{aligned} \qquad (7-33)$$

このための自己検査性回路は（7－6）式から容易に求めることができ，この回路を図7－4に示す。この場合，36ゲートとなりゲート増加比は27％である。

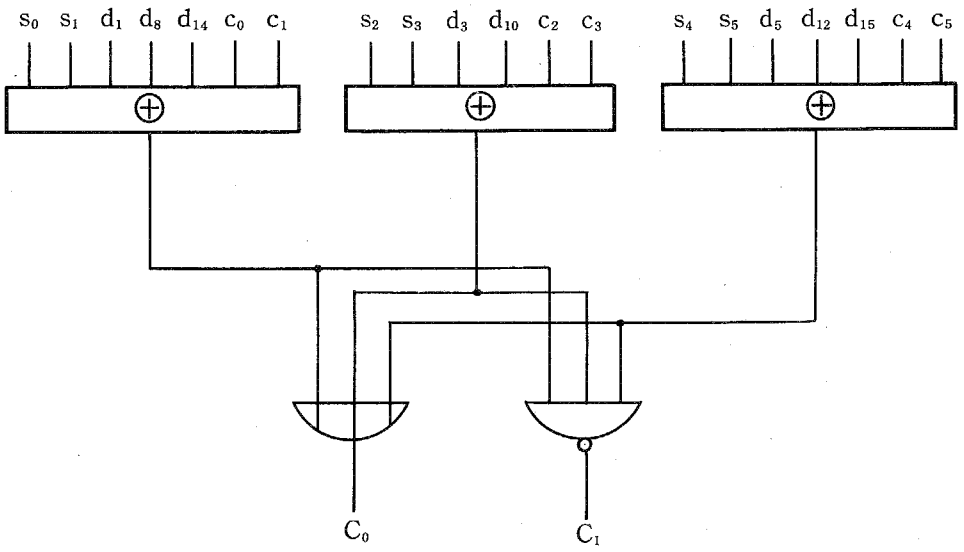


図7－4　SGに対する自己検査性検査回路

② SDに対する自己検査性検査回路

誤りポインタを$E=(e_0, e_1, \cdots, e_{15}, e_{c_0}, e_{c_1}, \cdots, e_{c_5})$，シンドロームを$S=(s_0, s_1, \cdots, s_5)$とする。手法3による検査論理を用いた場合，（7－29）式のHマトリクスに対してゲート数を最小とするAは次のようになり，これから$F_0, F_1, \cdots, F_5$は以下のように定まる。

$$
\left.\begin{array}{l}
A = [\,0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0 \mid 0\ 0\ 0\ 0\ 0\ 0\,] \\
F_0 = [\,1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1 \mid 1\ 0\ 0\ 0\ 0\ 0\,] \\
F_1 = [\,0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1 \mid 0\ 1\ 0\ 0\ 0\ 0\,] \\
F_2 = [\,0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0 \mid 0\ 0\ 1\ 0\ 0\ 0\,] \\
F_3 = [\,1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0 \mid 0\ 0\ 0\ 1\ 0\ 0\,] \\
F_4 = [\,0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1 \mid 0\ 0\ 0\ 0\ 1\ 0\,] \\
F_5 = [\,1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0 \mid 0\ 0\ 0\ 0\ 0\ 1\,]
\end{array}\right\} \quad (7-34)
$$

（7－13）式から，

$$
\left.
\begin{aligned}
E \cdot A^T &= e_1 \oplus e_3 \oplus e_5 \oplus e_8 \oplus e_{10} \oplus e_{12} = t_A \\
s_0 \oplus E \cdot F_0^T &= s_0 \oplus e_0 \oplus e_1 \oplus e_6 \oplus e_{11} \oplus e_{13} \oplus e_{15} \oplus e_{c_0} = t_0 \\
s_1 \oplus E \cdot F_1^T &= s_1 \oplus e_2 \oplus e_4 \oplus e_7 \oplus e_8 \oplus e_9 \oplus e_{14} \oplus e_{15} \oplus e_{c_0} = t_1 \\
s_2 \oplus E \cdot F_2^T &= s_2 \oplus e_2 \oplus e_3 \oplus e_6 \oplus e_9 \oplus e_{13} \oplus e_{c_2} = t_2 \\
s_3 \oplus E \cdot F_3^T &= s_3 \oplus e_0 \oplus e_4 \oplus e_7 \oplus e_{10} \oplus e_{11} \oplus e_{c_3} = t_3 \\
s_4 \oplus E \cdot F_4^T &= s_4 \oplus e_4 \oplus e_5 \oplus e_6 \oplus e_9 \oplus e_{11} \oplus e_{14} \oplus e_{15} \oplus e_{c_4} = t_4 \\
s_5 \oplus E \cdot F_5^T &= s_5 \oplus e_0 \oplus e_2 \oplus e_7 \oplus e_{12} \oplus e_{13} \oplus e_{14} \oplus e_{c_5} = t_5
\end{aligned}
\right\} \quad (7-35)
$$

そこで，t_A，t_0，…，t_5 を用いて，これに対する自己検査性検査回路は以下のようになる。

$$
\begin{cases}
C_0 = \left(\bigcup_{i=0}^{5} t_i \right) \cup t_A \\
C_1 = \overline{\left(\bigcap_{i=0}^{5} t_i \right) \cap t_A}
\end{cases} \quad (7-36)
$$

この場合，構成ゲート数は９８ゲートとなる。SDのゲート数が５０であるから，ゲート増加比は１９６％となる。

次に，定理７－５に基づき簡素化した検査について示す。（７－３４）式から，$F = (F_0, F_1, \cdots, F_5)^T$ としたとき，Fの各列は，その重みが１のものと，３のものとからなる。重みの等しい列の間では，互いに相異なることから，（７－２６）式に示したと同様の拡張を行うことができる。従って，列重みが１の列と３の列で各々集合Gを作ることができる。（７－３４）式からFの列重み１の列に対応したAの要素がすべて'1'のものと，Fの列重み３の列に対応したAの要素がすべて'0'のものについては，（７－２７）式の関係が成立する。よって，この場合，次の列を論理和に置き換えることが可能となる。（７－２２）式の左辺を t_A，右辺を t_i とすれば次式が得られる。

$$
\left.
\begin{aligned}
t_A &= e_1 \cup e_3 \cup e_5 \cup e_8 \cup e_{10} \cup e_{12} \\
t_0 &= s_0 \oplus (e_0 \cup e_1 \cup e_6 \cup e_{11} \cup e_{13} \cup e_{15}) \oplus e_{c_0} \\
t_1 &= s_1 \oplus (e_2 \cup e_4 \cup e_7 \cup e_8 \cup e_9 \cup e_{14} \cup e_{15}) \oplus e_{c_1} \\
t_2 &= s_2 \oplus (e_2 \cup e_3 \cup e_6 \cup e_9 \cup e_{13}) \oplus e_{c_2} \\
t_3 &= s_3 \oplus (e_0 \cup e_4 \cup e_7 \cup e_{10} \cup e_{11}) \oplus e_{c_3} \\
t_4 &= s_4 \oplus (e_4 \cup e_5 \cup e_6 \cup e_9 \cup e_{11} \cup e_{14} \cup e_{15}) \oplus e_{c_4} \\
t_5 &= s_5 \oplus (e_0 \cup e_2 \cup e_7 \cup e_{12} \cup e_{13} \cup e_{14}) \oplus e_{c_5}
\end{aligned}
\right\} \quad (7-37)
$$

よって，（C_0, C_1）は以下のように示すことができる。

$$
\begin{cases}
C_0 = \left(\bigcup_{i=0}^{5} t_i \right) \cup t_A \\
C_1 = \overline{\left(\bigcap_{i=0}^{5} t_i \right) \cap t_A}
\end{cases} \quad (7-38)
$$

上式にもとずいた自己検査性検査回路の構成を図７－５に示す。この場合，３３ゲートで構成でき，ゲート増加比は６６％となる。以上から，論理の簡素化により大幅にゲート数を減らすことができた。

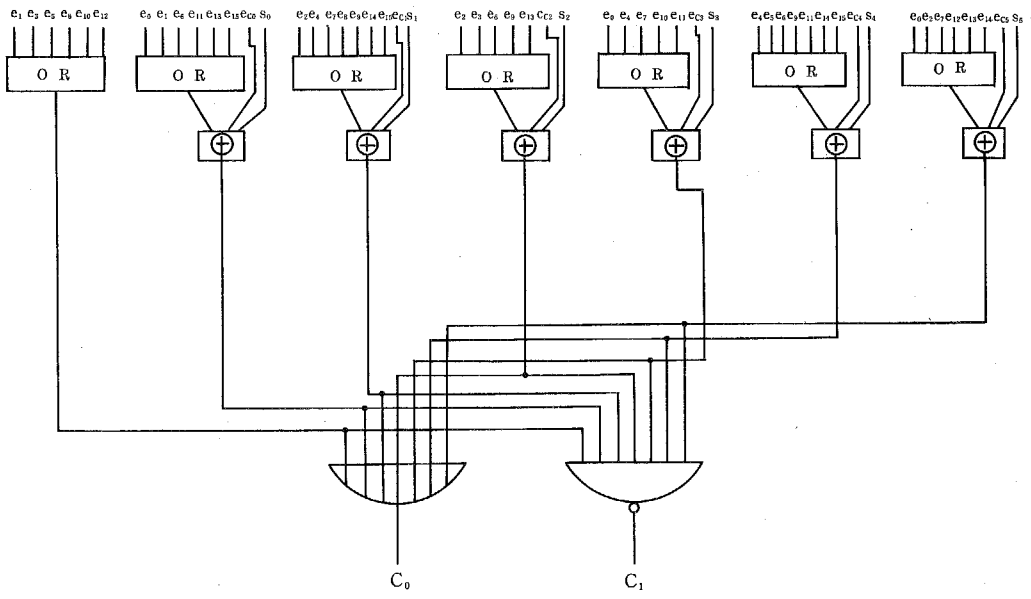


図７－５　ＳＤに対する自己検査性検査回路

〔2〕　評価結果

検査回路に対する評価項目として

ⅰ）　検査能力

ⅱ）　ゲート増加比

ⅲ）　検査速度

の3点がある。先の〔1〕においては，情報ビット数16ビットのSEC−DED符号の場合の例について示したが，ここでは検査ビット数のほぼ等しいS2EC符号の場合も含め，さらに情報ビット数32，64ビットの場合についても評価の対象とする。

まず，検査能力に関しては，前述の通り単一故障検出能力100％のT.S.C.回路である。

ゲート増加比に関してその結果を表7－5に示す。また，SG，及びSDに対する検査として最もゲート増加の小さい手法を組合わせて適用した場合のゲート増加比を図7－6に示す。これから，本検査論理は一般に符号長が大きくなるにつれて，ゲート増加比が小さくなる傾向にある。情報ビット数32ビットのとき，ゲート増加比は30％程度，また大形計算機システム中の主記憶装置で一般に使用している64ビット幅のとき25％程度となり，比較的実用的なゲート数増加で実現できる。

次に，検査速度については手法3によるSGに対する場合，および簡素化した手法3によるSDに対する場合について，表7－5に示す。ここで，相対検査速度は次のように定義する。

$$\text{相対検査速度} = \frac{\text{検査回路の伝搬遅延}}{\text{被検査回路の伝搬遅延}} \qquad (7-39)$$

表７－５　ゲート増加比，検査速度と符号長の関係

符号			SEC−DED			S2EC		
情報ビット数（ビット）			16	32	64	16	32	64
構成ゲート数（ゲート）	SG	被検査対象回路（SG）	135	250	480	同左 *		
		手法3による検査回路	36（26.7）**	50（20.0）	92（19.2）			
	SD	被検査対象回路（SD）	50	96	192	110	210	400
		手法2による検査回路	150（300）	240（250）	538（280）	150（136）	240（114）	538（135）
		手法3による検査回路	98（198）	172（179）	388（202）	106（96.3）	224（107）	498（125）
		簡素化した検査回路	33（66.0）	49（51.0）	72（37.5）	58（52.7）	92（43.8）	148（37.0）
相対検査速度		SG	0.8	1	1.3	0.8	1	1.3
		SD	1.7	1.7	2	1.5	1.5	1.6

＊　奇数重み列Ｓ２ＥＣ符号を使用したため，同一符号でSEC−DED符号との兼用を図っている。

＊＊　（　）内はゲート増加比（%）

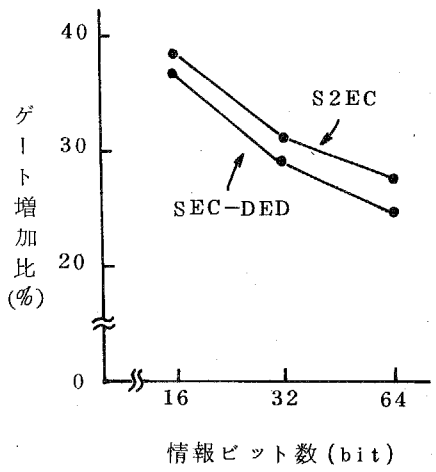


図７－６　ゲート増加比と符号長の関係

7－4　ＬＳＩ構成[28]

第５章において，符号化・復号化回路のＬＳＩ化に適する巡回性符号についてのべた。この符号構成はＨマトリクスに対して部分マトリクス単位に，列方向に巡回置換した構成を有するもので，１個の部分マトリクスをもとにその回路を構成すれば，同一回路複数個を用いて全体の符号化・復号化回路が作成できるものである。

本節では，この性質を有する符号を使用して，符号化・復号化回路のLSI化を考える。一般に，論理回路をＬＳＩ化するに当っては，リピータビリティ向上の観点から，そのＬＳＩ機能の汎用性を考慮する必要がある。この観点から，符号化・復号化回路のＬＳＩ化に対しては，

(1)　符号長の拡張性

(2)　多機能性

(3)　自己検査性検査回路内蔵による高信頼化

を考慮する。(1)の符号長の拡張性とは，１ＬＳＩ内におさめた基本回路で基本符号長に対する符号化・復号化回路が構成できるとともに，これを複数個使用すれば，その分複数倍拡張した符号長に対する符号化・復号化回路が構成できることを意味する。(2)の多機能性とは，符号の機能としてＳＥＣ－ＤＥＤ，ＳｂＥＣ等の多種の機能を同時に持たせ，しかも出来るだけ少ない検査ビット数でこれらの機能を実現することである。(3)は本来，その動作において高信頼化が要求される符号化・復号化回路に対して，前節で示した簡便な自己検査性の性質を有する検査回路を内蔵させて高信頼化を図ることである。

7－4－1　符号構成

〔１〕　基本部分マトリクス

（2－7）式において，巡回性符号の構成を示した。巡回性の性質を有するSEC, SEC−DED等の機能を有するビット系符号，ＳＥＣ－ＳｂＥＤ，ＳＥＣ－ＤＥＤ－ＳｂＥＤ，ＳｂＥＣ，ＳｂＥＣ－ＤｂＥＤ等の機能を有するバイト系符号については第５章において示した。これらの符号を用いて回路のＬＳＩ化を考えるとき，巡回性符号の基本部分マトリクスH_0は次に示す構成であることが望ましい。

$$H_0 = \left[\begin{array}{c|c} H_\alpha & I \\ \hline \multicolumn{2}{c}{O} \end{array}\right]_{r\times n_0} \qquad (7-40)$$

I；（$r-r_0$）×（$r-r_0$）の単位行列（$r_0<r$）

O；$r_0 \times n_0$ 零行列（$n_0=n/r$）

ここで，H_α はHマトリクスの情報部として所定の機能を満足し，しかも列方向に巡回置換して並べた全体のHマトリクスが，同時にその機能を保つようにして設定したマトリクスである。このH_0から$r_0 \times n_0$の零行列Oを除いて，次に示す基本符号長n_0，検査長$r-r_0$を有するHマトリクスH_0'を得る。

$$H_0' = [\, H_\alpha \,|\, I \,]_{(r-r_0)\times n_0} \qquad (7-41)$$

これから，（7－40）式のH_0に示すように，すべて'0'からなるr_0行をあらかじめ設けておくことにより，符号長n_0の符号を構成する場合に，その検査長をr_0だけ短くできる利点が生ずる。

〔2〕 符号の包含関係

（定義7－3）

機能Φ_Aを有する符号C_Aが，機能Φ_Bを有する符号C_Bに対し，$\Phi_A \subset \Phi_B$であれば，符号C_Bは符号C_Aを包含すると言う。 □

図2－1に示す各種の機能を有する符号の包含関係を表7－6に示す。SbEC の機能については，bの約数pに対してSpEC符号を考えるとき，

SbEC ⊃ SpEC

の関係が成立する。奇数重み列SbEC符号はGF(2)上で表現したとき，その奇数重み列の特徴からSEC−DED符号としても使える。しかし，SbECの機能とSEC−DEDの機能は，上記定義に基づく包含関係は成立しないため，その使用は排他的となる。そこで，SbEC, SpEC, SEC−DED の機能を任意に選択できるためには，奇数重み列SbEC符号を採用し，これらの機能から1個を選択して使用すればよい。

表７－６　符号の包含関係

	SEC	SEC-DED	SEC-SbED	SEC-DED-SbED	SpEC*	SbEC*	SbEC-DbED
SEC		∪***	∪	∪	∪	∪	∪
SEC−DED	⊃**			∪			∪
SEC−SbED	⊃			∪		∪	∪
SEC-DED-SbED	⊃	⊃	⊃				∪
SpEC*	⊃					∪	∪
SbEC*	⊃		⊃		⊃		∪
SbEC−DbED	⊃	⊃	⊃	⊃	⊃	⊃	

＊　ｐはｂの約数とする。

** ⊃　横欄の機能が縦欄の機能を包含することを示す。

*** ∪：縦欄の機能が横欄の機能を包含することを示す。

（空欄は包含関係が成立しないことを示す。）

〔３〕　中間拡大体を考慮した符号構成

ＳｂＥＣ符号を構成するに当って，ＳｐＥＣの機能を包含する点から，中間拡大体を考慮した符号構成を考える。ｂ＝ｐ・ｍ（ｍは１より大なる整数）として，基礎体ＧＦ(2)と拡大体ＧＦ（2^b）との間に中間拡大体ＧＦ（2^p）を定義する。ｑ＝2^pとして，ＧＦ(2)上の既約多項式をｇ（ｘ）とすれば，これはｐ次のＧＦ(2)上の元を係数とする多項式である。このとき，同伴行列ｔは定義４－１から，次式のように与えられる。

$$g(x)=\sum_{i=0}^{p} g_i x^i, \quad g_0=g_p=1$$

$$t=\begin{bmatrix} 0 & 0 & \cdots & 0 & g_0 \\ & & & & g_1 \\ & & I_{p-1} & & \vdots \\ & & & & g_{p-1} \end{bmatrix}_{p\times p}$$

$$\{0, 1, g_0, g_1, \cdots, g_{p-1}\} \in GF(2)$$

I_{p-1} ; $(p-1)\times(p-1)$の単位行列

(7−42)

ここで，$\{0, 1, t, t^2, \cdots, t^{2^p-2}\}=GF(q)$である。次に，GF(q) 上の既約多項式を G(X) とすれば，これはm次のGF(q)上の元を係数とする多項式である。上式と同様の考えから，同伴行列Tが求められる。

$$G(X)=\sum_{i=0}^{m} G_i X^i, \quad G_0=G_m=1$$

$$T=\begin{bmatrix} 0 & 0 & \cdots & 0 & G_0 \\ & & & & G_1 \\ & & I_{m-1} & & \vdots \\ & & & & G_{m-1} \end{bmatrix}$$

$$\{0, 1, G_0, G_1, \cdots, G_{m-1}\} \in GF(q)$$

I_{m-1} ; $(m-1)\times(m-1)$単位行列

(7−43)

ここで，$\{0, 1, T, T^2, \cdots, T^{2^m-2}\}=GF(q^m)=GF(2^b)$である。これにより，SbEC符号のHマトリクスはGF(q)上の元を用いて表現することができる。

一般に，包含関係を有する符号間では，その検査ビット数は機能の高い符号のそれに一致してしまうため，低い機能の符号にとって最少の検査ビット数とすることが困難である。しかし，このような中間拡大体を考慮した符号構成は，より低い機能の符号に対しても最少検査ビット数を満足する効果を与えることが可能となる。この点については〔5〕で具体的に示す。

〔４〕　巡回性奇数重み列Ｓ２ＥＣ／ＳＥＣ－ＤＥＤ符号の構成

具体的な符号構成例として，Ｓ２ＥＣの機能とＳＥＣ－ＤＥＤの機能を選択できる巡回性符号を示す。本符号は，情報ビット数として１６ビットを基本単位とし，３２ビット，６４ビットへ拡張できるものとする。すなわち，H_0の情報ビット部は１６ビットである。最大６４ビットの情報ビット数を考慮すると，Ｓ２ＥＣ符号，ＳＥＣ－ＤＥＤ符号共，最大８ビットの検査ビット数を必要とする。今，ＴをＧ（Ｘ）＝X^2＋Ｘ＋１なる既約多項式から作成される同伴行列とすると，ＧＦ（2^2）上で奇数重み列の条件を有し，しかも１個以上$O \in GF(2^2)$の元を有する非縮退巡回同値類の列ベクトルは，次の１１個のみである。

$$\begin{matrix} I & I & T & T & I & T^2 & T^2 & T & T^2 & T & I \\ I & T & I & T & T^2 & I & T^2 & T^2 & T & O & O \\ I & T & T & I & T^2 & T^2 & I & O & O & T^2 & O \\ O & O & O & O & O & O & O & O & O & O & O \end{matrix} \qquad (7-44)$$

そこで，H_0としては，１１列ベクトルのうち最後の１列ベクトルを検査部用，残り１０列ベクトルから８列ベクトルを情報部用に選択してH_0を構成すればよい。（７－４０）式から，例として次に示すH_0が構成できる。

$$H_0 = \left[\begin{array}{cccccccc:ccc} I & T & T & I & I & T^2 & T^2 & T & I & O & O \\ T & I & T & I & T^2 & I & T^2 & O & O & I & O \\ T & T & I & I & T^2 & T^2 & I & T^2 & O & O & I \\ \hdashline O & O & O & O & O & O & O & O & O & O & O \end{array}\right] \qquad (7-45)$$

このH_0をもとにして，１６ビット，３２ビット，６４ビットに対するＨマトリクスが図７－７に示すように構成できる。この符号は，ＧＦ（2^2）上の元をＧＦ（２）上で表現することにより，奇数重み列ＳＥＣ－ＤＥＤ符号としても表わすことができる。特に，１６ビットの基本長に対して（７－４５）式のH_0中にすべて'０'の行を１行設けたことにより，検査ビット数は６ビットと最少検査ビット数に等しくすることができる。この符号をＳＥＣ－ＤＥＤ符号およびＳ２ＥＣ符号として使用した場合のそれぞれの詳細な誤り検出能力を示したものが表７－７である。この符号はＳ２ＥＣ符号およびＳＥＣ－ＤＥＤ符号として誤り検出能力の高い符号の一つである。(22)(69)

次に，この符号に対する復号化は，４－２－２〔２〕に示した奇数重み列Ｓ２ＥＣ符号に対する復号化により実現できる。ここで，奇数重み列Ｓ２ＥＣ符号の復号化回路をＳＥＣ－ＤＥＤ符

号に対する復号に使用する場合には，同一バイト（２ビット）内の２ビット誤りは訂正せず，訂正不可能誤りとして検出しなければならない。

① 情報ビット数　16ビットの場合

$$H = \left[\begin{array}{cccccccc:ccc} 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 & 0 \\ T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 & 0 \\ T^1 & T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 & 0 & 1 \end{array}\right]$$

$$= \left[\begin{array}{cccccccccccc} 1 & & 1 & & 1 & & 1 & 1 & 1 & & 1 & \\ & 1 & 1 & 1 & & 1 & 1 & 1 & 1 & 1 & & \\ & 1 & & 1 & & 1 & 1 & 1 & 1 & & 1 & \\ 1 & & 1 & & 1 & & 1 & 1 & & & & \\ & 1 & & 1 & & 1 & 1 & 1 & & & & 1 \\ 1 & & 1 & & 1 & & 1 & 1 & 1 & & & \\ & 1 & & 1 & & 1 & 1 & & & & & 1 \end{array}\right]$$

② 情報ビット数　32ビットの場合

$$H = \left[\begin{array}{cccccccc:cc:cccccccc:cc} 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 & 0 \\ T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T^1 & T^1 & & 1 & T^2 & T^2 & 1 & T^2 & 0 & 0 & 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 \end{array}\right]$$

③ 情報ビット数　64ビットの場合

$$H = \left[\begin{array}{cccccccc:c:cccccccc:c:cccccccc:cc} 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 & T^1 \\ T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^1 \\ T^1 & T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 & T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^1 & T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 & T^1 & 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 & 1 \end{array}\right]$$

$$\left[\begin{array}{cccccccc:c} 1 & T^1 & 1 & T^2 & 1 & T^2 & 0 & 0 \\ T^1 & 1 & 1 & T^2 & T^2 & 1 & T^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T^1 & T^1 & 1 & 1 & T^2 & T^2 & T^1 & 1 \end{array}\right]$$

図７－７　巡回性奇数重み列Ｓ２ＥＣ/ＳＥＣ－ＤＥＤ符号

表７－７　巡回性奇数重み列S2EC／SEC-DED符号
（図７－７）の誤り検出能力（%）

機能 ＼ 誤り種別 ＼ 情報ビット数		６４ビット	３２ビット	１６ビット
SEC-DED	ランダム　３ビット*	45.00 (45.17)†	67.45	34.81
	ランダム　４ビット	99.20 (99.21)†	99.12	96.69
S2EC	２ブロック**	63.82 (66.35)†	74.74	56.36
	２ビット／２ブロック***	72.86 (74.76)†	81.05	67.27
	ランダム　４ビット	71.81	84.17	65.88

* ランダム３ビット誤りはSEC-DEDとS2ECとで同一の値
** ２ブロック間にわたるすべての誤り
*** ２ブロック間にわたる２ビット誤り
† 最も能力の高い符号の場合の誤り検出能力(22)(69)

〔５〕　巡回性奇数重み列Ｓ４EC／Ｓ２EC／ＳEC－DED符号

多機能性と符号長の拡張性を有する具体的な符号構成例として，Ｓ４EC，Ｓ２EC，ＳEC－DEDの３機能を選択できる巡回性符号を示す。この符号はＳ４ECに対しては，情報ビット数として１６ビット，３２ビット，６４ビット，１２８ビットを，Ｓ２EC，ＳEC－DEDに対しては，１６ビット，３２ビット，６４ビットをとることができるものとする。検査ビット数とくり返し数の関係を表７－８に示す。これから３２ビットの場合を除いて，最少の検査ビット数で実現できる。

表７－８　検査ビット数とくり返し数

機能＼情報長	１６ビット	３２ビット	６４ビット	128ビット
S4EC	8*／1**	8／2	12／3	12／6
S2EC	6／1	8／2	8／4	－
SEC-DED	6／1	8／2	8／4	－

* 検査ビット数

** くり返し数

次に符号の構成に当っては，〔３〕に示した中間拡大体を考慮した手法を採用する。すなわち，S4EC符号としてGF(2^4)上の元{ 0，1，T，T^2，…，T^{14} } はすべて中間拡大体GF(2^2)上の元{ 0，1，t，t^2 }を用いて構成できる。GF(2^2)上の同伴行列t，およびGF(2^4)上の同伴行列Tを，それぞれ$g(x)=x^2+x+1$，$G(X)=X^2+X+t$ より構成する。このとき，GF(2^4) 上の元による奇数重み列を満足する２行の列ベクトルは次の１６個である。

$$\begin{pmatrix}T\\T^4\end{pmatrix}\begin{pmatrix}T^2\\T^8\end{pmatrix}\begin{pmatrix}T^3\\T^{14}\end{pmatrix}\begin{pmatrix}T^5\\T^{10}\end{pmatrix}\begin{pmatrix}T^6\\T^{13}\end{pmatrix}\begin{pmatrix}T^7\\T^9\end{pmatrix}\begin{pmatrix}T^{11}\\T^{12}\end{pmatrix}\begin{pmatrix}T^4\\T\end{pmatrix}$$

$$\begin{pmatrix}T^8\\T^2\end{pmatrix}\begin{pmatrix}T^{14}\\T^3\end{pmatrix}\begin{pmatrix}T^{10}\\T^5\end{pmatrix}\begin{pmatrix}T^{13}\\T^6\end{pmatrix}\begin{pmatrix}T^9\\T^7\end{pmatrix}\begin{pmatrix}T^{12}\\T^{11}\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} \qquad (7-46)$$

これらをGF(2^2)上の元で表現すると次の様になる。

$$\begin{pmatrix} 0 & t^{*} \\ 1 & 1 \\ 1 & t \\ 1 & 0 \end{pmatrix} \begin{pmatrix} t & t \\ 1 & t^2 \\ t^2 & t \\ 1 & t \end{pmatrix} \begin{pmatrix} t & 1^{*} \\ t^2 & 1 \\ t^2 & 1 \\ t^2 & 0 \end{pmatrix} \begin{pmatrix} t^{*} & 0 \\ 0 & t \\ t^2 & 0 \\ 0 & t^2 \end{pmatrix} \begin{pmatrix} 0 & t^2 \\ t^2 & t \\ 1 & t^2 \\ t^2 & t^2 \end{pmatrix} \begin{pmatrix} t^2 & t^{2*} \\ t & 1 \\ t & t^2 \\ t & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ t^2 & t^2 \\ 1 & 1 \\ t^2 & t \end{pmatrix} \begin{pmatrix} 1 & t \\ 1 & 0 \\ 0 & t \\ 1 & 1 \end{pmatrix} \begin{pmatrix} t^2 & t \\ 1 & t \\ t & t \\ 1 & t^2 \end{pmatrix} \begin{pmatrix} t^2 & 1 \\ t^2 & 0 \\ t & 1 \\ t^2 & 1 \end{pmatrix} \begin{pmatrix} t^2 & 0 \\ 0 & t^2 \\ t & 0 \\ 0 & t \end{pmatrix} \begin{pmatrix} 1 & t^2 \\ t & t^2 \\ 0 & t^2 \\ t & t \end{pmatrix} \qquad (7-47)$$

$$\begin{pmatrix} t & t^2 \\ t & 0 \\ t^2 & t^2 \\ t & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ t^2 & t \\ 0 & 1 \\ t^2 & t^2 \end{pmatrix} \begin{pmatrix} 1^{*} & 0^{*} \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0^{*} & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

これらの列ベクトル中，（7－45）式に示す列ベクトルと一致するものは（7－47）式中＊印をつけた列ベクトルである。これ以外に不足する列ベクトルは次のものである。

$$\begin{pmatrix} 1 \\ t \\ t \\ 0 \end{pmatrix} \begin{pmatrix} t \\ t \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ t^2 \\ t^2 \\ 0 \end{pmatrix} \begin{pmatrix} t^2 \\ t^2 \\ 1 \\ 0 \end{pmatrix} \qquad (7-48)$$

そこで，S4EC符号は（7－47）式中で＊印をつけた列を含む（7－46）式中の6列ベクトルと，GF(2)上で表現したとき'1'の数（重み）の小さい2列ベクトルの計8ベクトル（（7－46）式中の＋印）を選択して構成できる。この関係を示したものが図7－8である。これから，S2EC／SEC−DED符号としては図7－7に示す符号に一致し，一方，S4EC符号としては図7－9に示す符号となる。図中，情報ビット数128ビットを有する符号は，H_0，H_1，H_2で構成され，H_0中H_{00}またはH_{01}に検査列ベクトル1列を加えたものを基本部分Hマトリクスとすれば，全体はH_{00}またはH_{01}の各行ベクトルを入れかえた6個の部分Hマトリクスで構成される。これから，くり返し6の符号となる。

以上から，S4EC用とS2EC／SEC−DED用とで中間拡大体を考慮することにより，共通の列ベクトルを設けることができた。これにより，符号化・復号化回路の部分的な共用化が可能となり，個別に構成した場合と比較してゲート量の削減が可能となる。このような共通列

ベクトルを出来るだけ多く選択することが，ゲート量削減上重要である。

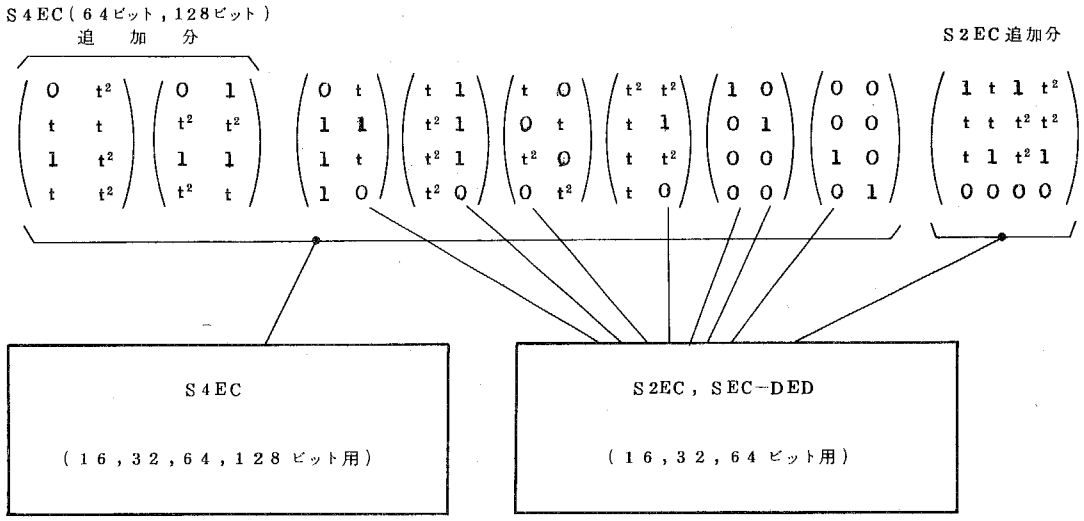


図7－8　3機能とHマトリックス列ベクトルとの対応

① 16ビット構成

$$H = \left[\begin{array}{cccccc|cc} T & T^3 & T^5 & T^7 & 1 & 0 \\ T^4 & T^{14} & T^{10} & T^9 & 0 & 1 \end{array}\right]$$

② 32ビット構成

$$H = \left[\begin{array}{cccccc|cccccc} T & T^3 & T^5 & T^7 & 1 & & T^4 & T^{14} & T^{10} & T^9 & 0 & \\ T^4 & T^{14} & T^{10} & T^9 & 0 & & T & T^3 & T^5 & T^7 & 1 & \end{array}\right]$$

③ 64ビット構成

$$H = \left[\begin{array}{cccccccc|cccccccc} T & T^3 & T^5 & T^7 & T^6 & T^{11} & 1 & & 0 & 0 & 0 & 0 & 0 & 0 & & T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & 0 \\ T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & 0 & & T & T^3 & T^5 & T^7 & T^6 & T^{11} & 1 & & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & & T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & 0 & & T & T^3 & T^5 & T^7 & T^6 & T^{11} & 1 \end{array}\right]$$

④128ビット構成

$$H = \left[\begin{array}{cccccccc|cccccccc|cccccccc} \overbrace{T & T^3 & T^5 & T^7 & T^6 & T^{11}}^{H_{00}} & \overbrace{T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12}}^{H_{01}} & 1 & \overbrace{0 & 0 & 0 & 0 & 0 & 0}^{H_{10}} & \overbrace{0 & 0 & 0 & 0 & 0 & 0}^{H_{11}} & 0 & \overbrace{T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12}}^{H_{20}} \\ T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & T & T^3 & T^5 & T^7 & T^6 & T^{11} & 0 & T & T^3 & T^5 & T^7 & T^6 & T^{11} & T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & T & T^3 & T^5 & T^7 & T^6 & T^{11} & 0 & T & T^3 & T^5 & T^7 & T^6 & T^{11} \end{array}\right]$$

H_0　H_1　H_2

$$\left[\begin{array}{cccccc|c} \overbrace{T & T^3 & T^5 & T^7 & T^6 & T^{11}}^{H_{21}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T^4 & T^{14} & T^{10} & T^9 & T^{13} & T^{12} & 1 \end{array}\right]$$

図7－9　巡回性奇数重み列Ｓ４ＥＣ/Ｓ２ＥＣ/ＳＥＣ－ＤＥＤ符号

7－4－2　回路構成

7－4－1で求めた符号をもとに，LSI化に適した汎用性のある符号化・復号化回路の構成法についてのべる。この回路は，図7－1に示したCG／SG，SD，COR，DET，PG以外に，この回路の正常性を検査する自己検査性検査回路（SC）から構成されるとする。ここで，LSI化の対象とするのは，DETを除くすべての回路である。

〔1〕　巡回性奇数重み列S2EC／SEC－DED符号による回路構成

（7－45）式に示すH_0を採用して具体的に基本回路を構成したものを図7－10に示す。この回路は以下の特徴を有する。

① CG／SG中，検査データ$C_{i,0}$，$C_{i,1}$，$C_{i,2}$（i＝0，1，2，3）はBの制御信号により，符号化動作では遮断され，復号化動作にのみ入力できる構成である。これから，検査ビットの生成（CG）とシンドロームの生成（SG）を，この回路で兼用可能としている。

② 生成された検査ビットまたはシンドロームは完全な形としてはS_0（Hマトリクス第1行目に相当）として出力され，他の出力S_1，S_2（それぞれHマトリクス第2行目，第3行目に相当）は，途中パリティ検査結果として他の基本回路のCG／SGのS''_1，S''_2へ入力する。このとき，同時にS_0は他の基本回路からの途中パリティ検査結果S''_1，S''_2を入力させて作成する。各基本回路からのS_0を集めて，全体の生成検査ビットまたはシンドロームとする。

③ SDへの入力は，S_0以外は他の基本回路で作成したS_0に相当するシンドロームS'_1，S'_2 S'_3である。

基本回路の規模，およびこれを用いて構成した符号化・復号化回路の遅延を表7－9に示す。

次に，図7－10に示す基本回路を複数個使用し，それらに所定の接続を施こすことにより符号長を拡大させた回路構成を具体的に示す。

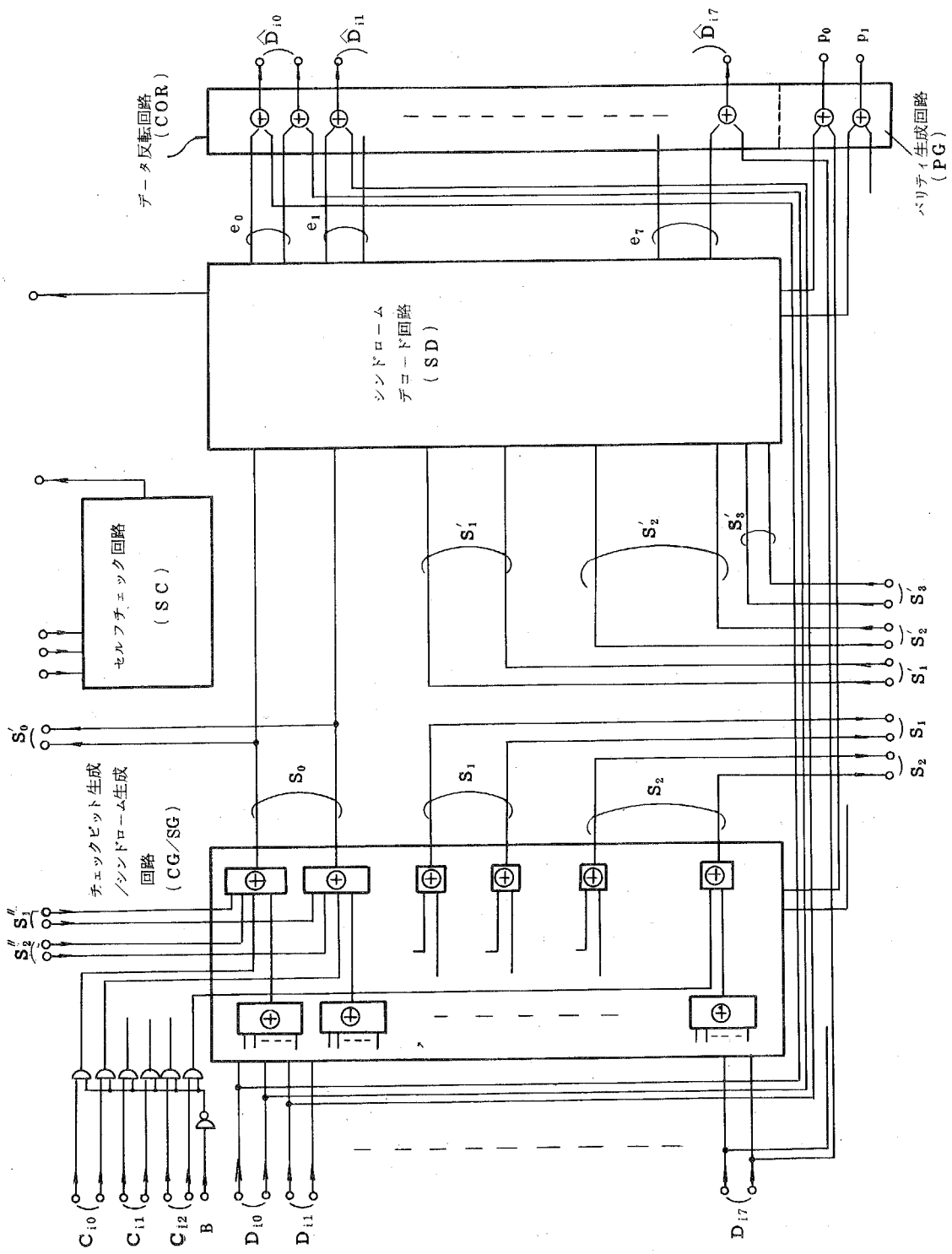


図７－１０　巡回性奇数重み列Ｓ２ＥＣ／ＳＥＣ－ＤＥＤ符号のための基本回路

表7－9　基本回路の規模

金物量	OG/SG	135ゲート	410ゲート	2入力EX-ORゲートは2ゲートで換算
	SD	110 〃		
	COR	30 〃		
	PG	20 〃		
	SC	115 〃		
信号端子数	入力	36 本	62 本	
	出力	26 〃		
回路遅延	符号化	8ゲート段数（16ビット構成では6ゲート段数）		4入力までのAND，OR，NOR，2入力EX-ORゲートは1ゲート段数と換算，ファンアウト，布線長は考慮せず。
	復号化	13ゲート段数（16ビット段数では11ゲート段数）		

（i）　16ビット構成

図7－10に示す回路単体にて構成できる。このとき，出力S_1，S_2はそれぞれ入力S'_1，S'_2と接続し，また使用しない入力S''_1，S''_2，S'_3は論理的に'0'としておかなければならない。この場合，S'_0，S_1，S_2は符号化の場合検査データとなり，復号化の場合シンドロームとなる。この関係を図7－11に示す。

（ii）　32ビット構成

図7－7②のHマトリクスに基づき，図7－10の回路を2個（K_0，K_1）使用し，図7－12に示す接続を行って構成する。生成される検査データは，双方の回路K_0，K_1の出力S_0，S_1となる。復号時には，これらがシンドロームとなる。

（iii）　64ビット構成

図7－7③のHマトリクスに基づき，図7－10に示す基本回路を4個（K_0～K_3）使用し，図7－13に示す接続を行って構成できる。生成される検査データは各回路の出力であるS_0に相当し，復号時においてはこれがシンドロームとなる。

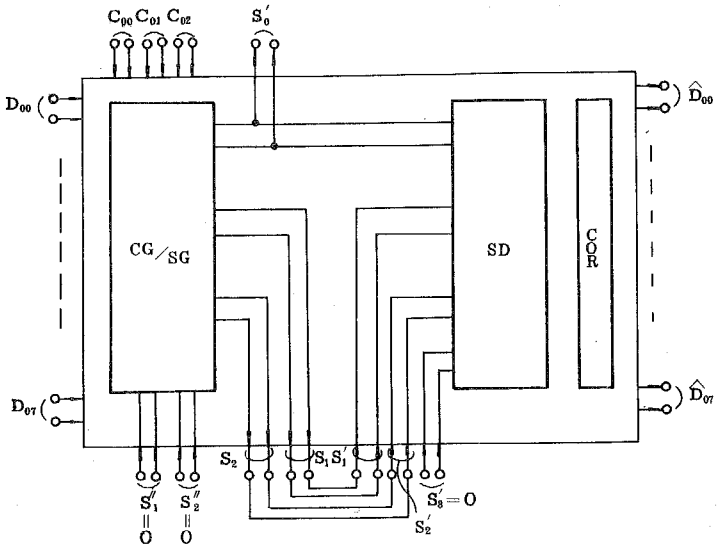


図7－11　16ビット構成

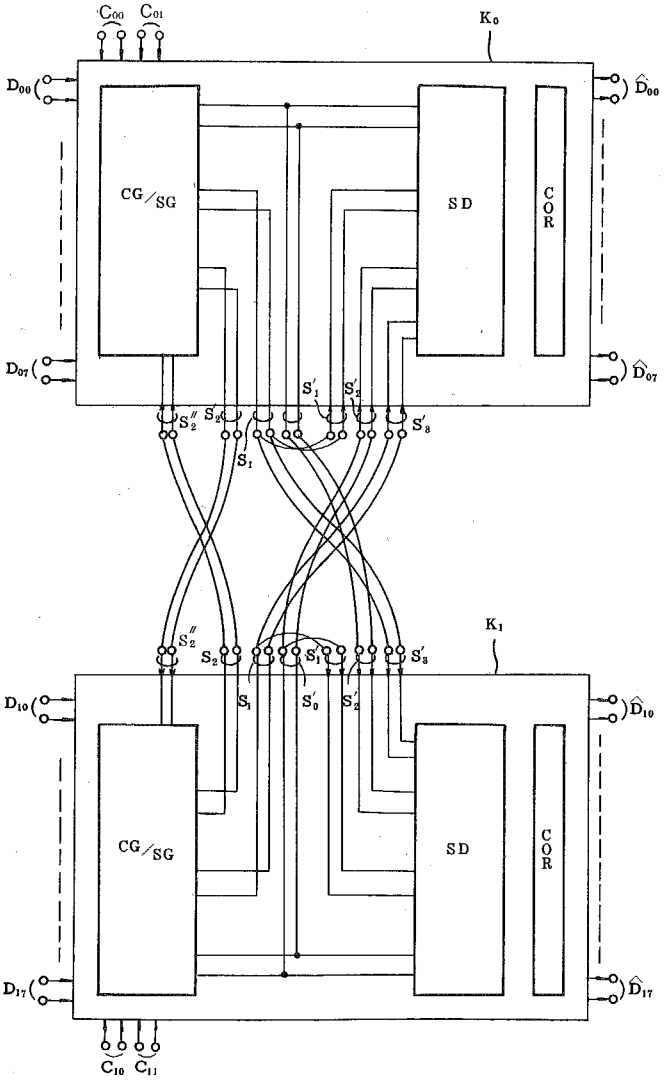


図7－12　32ビット構成

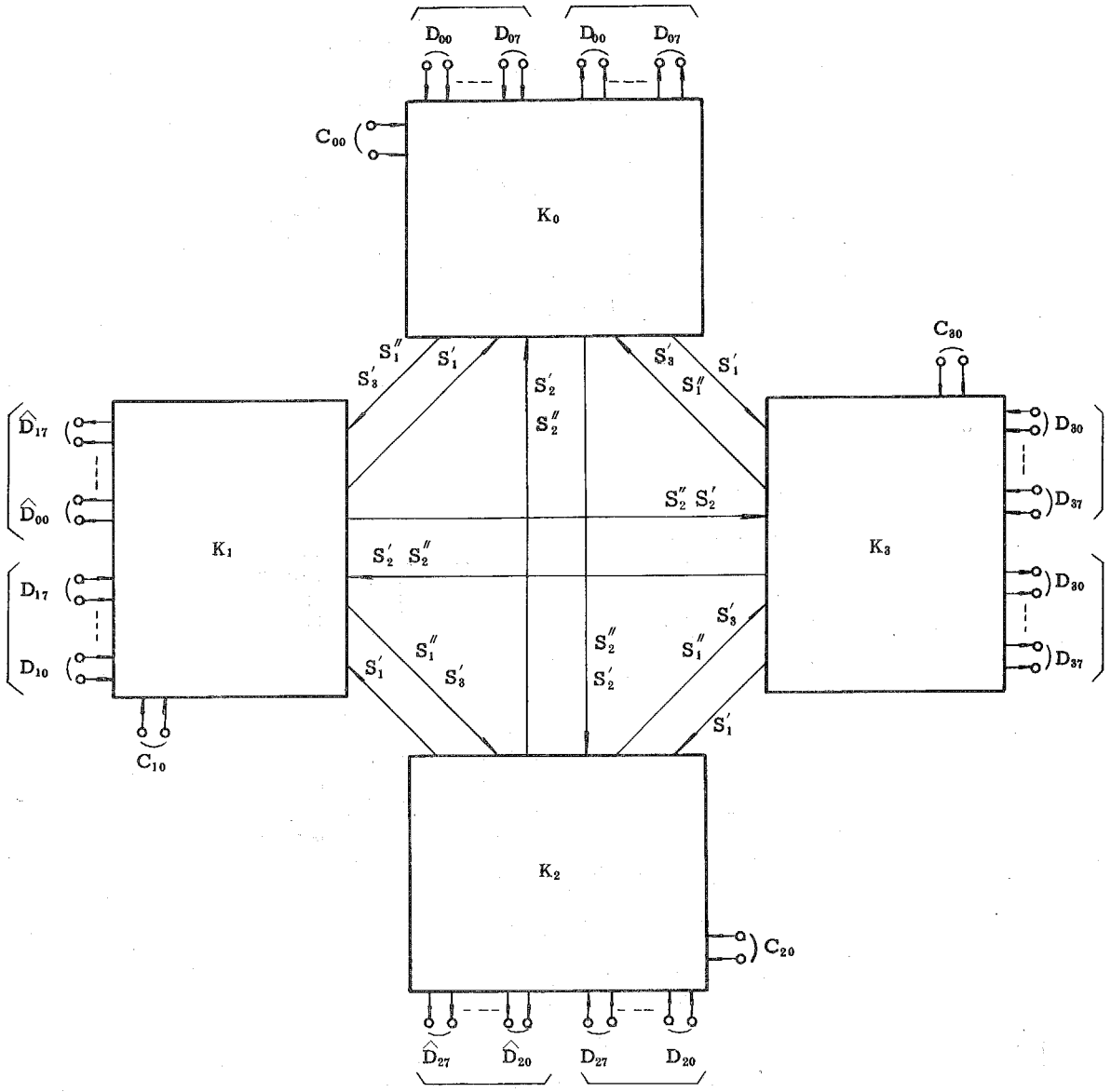
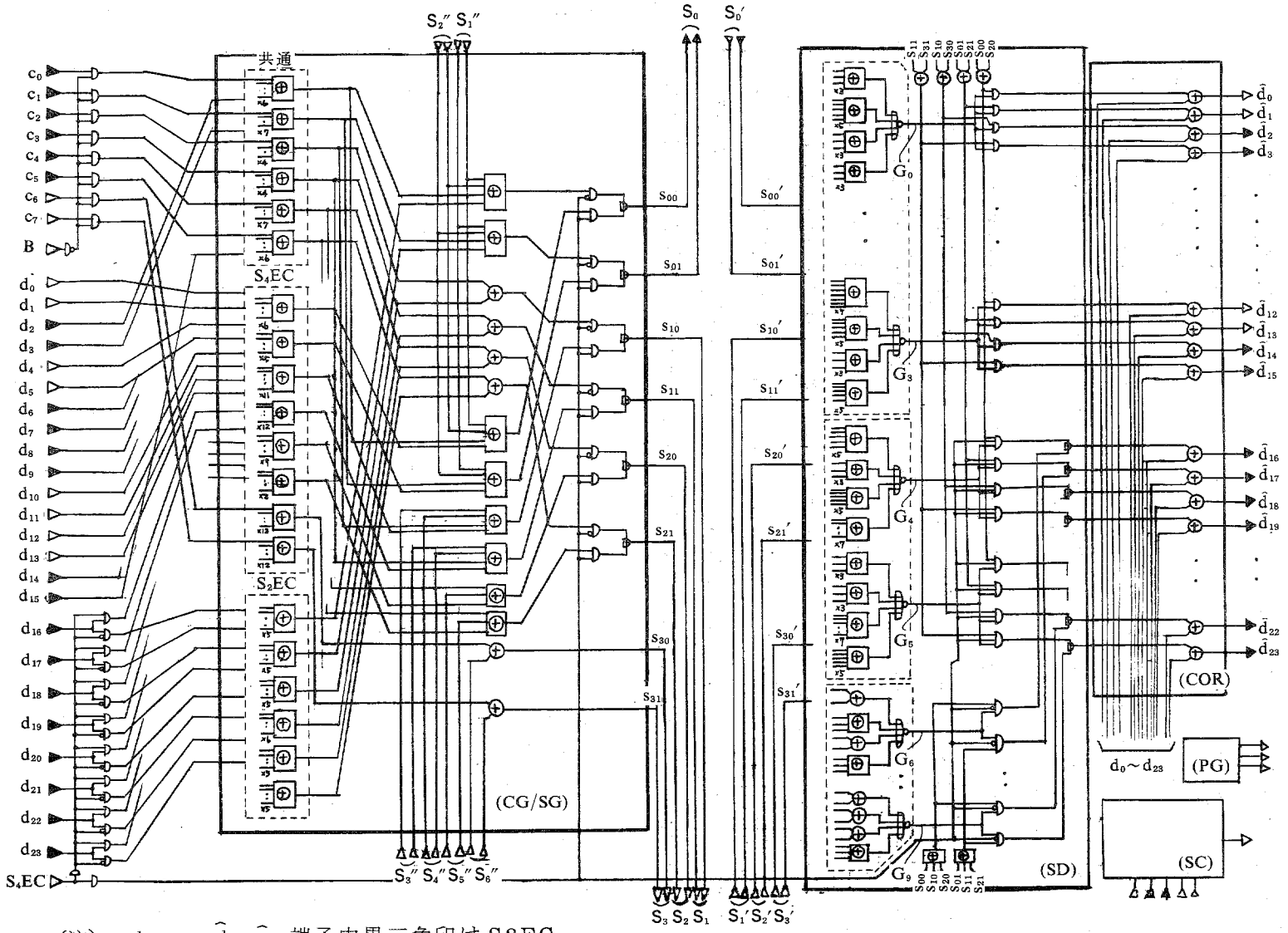


図7－13　64ビット構成

〔２〕　巡回性奇数重み列Ｓ４ＥＣ／Ｓ２ＥＣ／ＳＥＣ－ＤＥＤ符号による回路構成

本符号に対する基本回路構成の考え方は，前の〔１〕の場合と同一である。本回路は図７－８に示した列ベクトルを使用して表わしたＨマトリクスにより構成できる。この場合，Ｓ４ＥＣ用とＳ２ＥＣ／ＳＥＣ－ＤＥＤ用とで，共通に使用する列ベクトルと各々で別個に使用する列ベクトルとがあり，また符号の機能により復号化の過程が異なる部分が存在する。これらは選択ゲートにより切り換えて使用する。この基本回路を図７－１４に示す。また，この回路の規模，回路遅延等を表７－１０に示す。

次に，この基本回路を複数個接続して，符号長を拡張した符号化・復号化回路を構成してみる。この場合，Ｓ２ＥＣまたはＳＥＣ－ＤＥＤの機能を実現する場合と，Ｓ４ＥＣの機能を実現する場合でその構成法は異なる。前者の場合は〔１〕でのべたと全く同一でよい。後者のＳ４ＥＣの場合には，次のようになる。すなわち，１６ビット構成の場合には基本回路１個で，３２ビット構成の場合には２個使用すればよい。６４ビット構成の場合には，図７－９③のＨマトリクスから，基本回路を３個使用する。１２８ビット構成の場合には，図７－９④の構成から，基本回路２個ずつをあらかじめ互いに所定の接続関係を与えて，これを３組用意し，これらを新たな基本回路として３組間の接続を行えばよい。１２８ビットの場合の構成を図７－１５に示す。



(注)　d_i c_i, $\hat{d}_i$, $\hat{c}_i$ 端子中黒三角印はS2EC，
SEC-DED用に用いる端子を表わす。

図7－14　巡回性奇数重み列S4EC/S2EC/SEC－DED
符号のための基本回路

表7－10　基本回路の規模

金物量	CG／SG	340ゲート	890ゲート
	SD	265 〃	
	COR	50 〃	
	PG	20 〃	
	SC	215 〃	
信号端子数	入力	57 本	94 本
	出力	37 〃	
回路遅延	符号化	12ゲート段数（16ビット構成で8ゲート段数）	
	復号化	18ゲート段数（16ビット構成で16ゲート段数）	

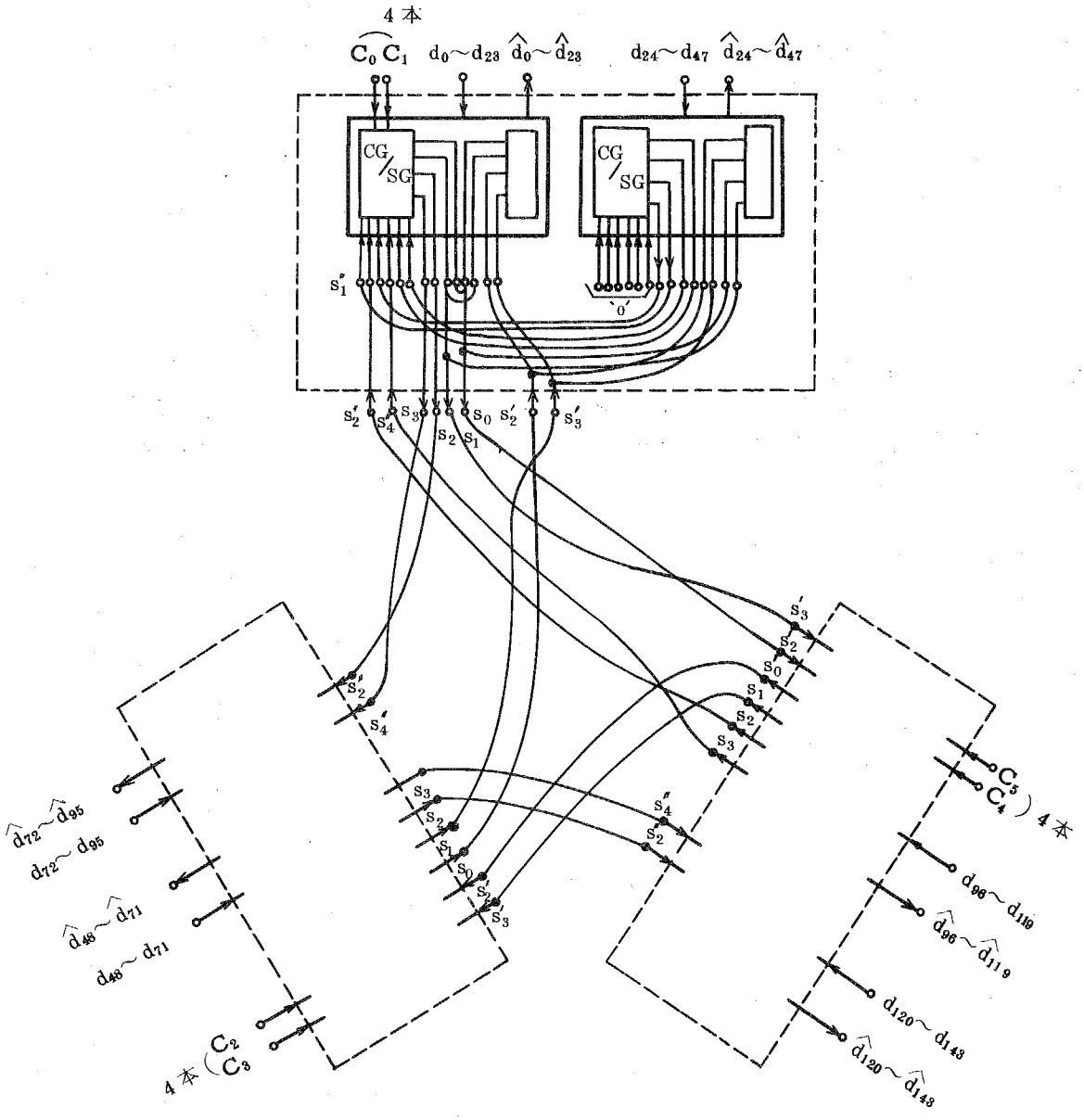


図7－15　情報ビット数128ビットを有するS4EC回路構成

7－4－3　自己検査性検査回路の内蔵

符号化・復号化回路のＬＳＩ化は，それ自体で高信頼化が達成できる。しかし，本節ではさらにＬＳＩ回路中の間欠故障，固定故障を通常動作実行中に検出するために，自己検査性を有する検査回路を内蔵させる。この回路の構成法は**7－3**に示した考え方を採用する。

まず，図7－10に示すＳ2ＥＣ／ＳＥＣ－ＤＥＤ用の回路を対象に検査回路を構成しよう。

〔1〕　CG／SGに対する検査回路

この回路に対するT.S.C.検査回路は，定理7－1～定理7－3をもとに構成する。図7－7①のＨマトリクスを対象に共通論理を形成するための最適なグループ分割を行い，これに対し定理7－2を適用してT.S.C.論理を構成する。入力情報等を以下のように表現する。

$$
\begin{aligned}
&\text{入力情報}\quad D=[\ D_0,\ D_1,\ D_2,\ D_3,\ D_4,\ D_5,\ D_6,\ D_7\ ]\\
&\qquad =[\ d_{00}\ d_{01},\ d_{10}\ d_{11},\ d_{20}\ d_{21},\ d_{30}\ d_{31},\ d_{40}\ d_{41},\ d_{50}\ d_{51},\ d_{60}\ d_{61},\ d_{70}\ d_{71}\ ]\\
&\qquad D_i=(d_{i0},d_{i1})\quad i=0,\ 1,\ \cdots,\ 7\\
&\text{入力検査情報}\quad C=[\ C_0\ ,\ C_1\ ,\ C_2\ ]^{*}\\
&\qquad =[\ c_{00}\ \ c_{01},\ c_{10}\ \ c_{11},\ c_{20}\ \ c_{21}\ ]\\
&\qquad C_j=(c_{j0},c_{j1})\quad j=0,\ 1,\ 2\\
&\text{シンドローム}\quad S'=[\ S'_0\ ,\ S'_1\ ,\ S'_2\ ]^{*}\\
&\qquad =[\ s'_{00}\ \ s'_{01},\ s'_{10}\ \ s'_{11},\ s'_{20}\ \ s'_{21}\ ]\\
&\qquad S'_k=(s'_{k0},s'_{k1})\quad k=0,\ 1,\ 2
\end{aligned}
\qquad (7-49)
$$

最適なグループ分割は，図7－7①のGF(2)上で表現したマトリクスで（0行，1行）（2行，3行）（4行，5行）の3グループ分割となる。この場合のg_0，g_1，g_2，A，f_0，f_1，f_2は次のようになる。

＊　32ビット，64ビットの拡張構成の場合，C_3，S_3が必要となる。

$$
\left.\begin{aligned}
g_0 &= [\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ ] \\
g_1 &= [\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ ] \\
g_2 &= [\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ ] \\
A &= [\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ ] \\
f_0 &= [\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ ] \\
f_1 &= [\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ ] \\
f_2 &= [\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ ]
\end{aligned}\right\} \qquad (7-50)
$$

以上から，（7－2）式に相当する検査論理として

$$
\left.\begin{aligned}
&(s'_{00} \oplus s'_{01}) \oplus (d_{01} \oplus d_{40} \oplus d_{70} \oplus c_{00} \oplus c_{01}) \oplus (s''_{10} \oplus s''_{11} \oplus s''_{20} \oplus s''_{21}) \rightarrow t_0 \\
&= (s_{10} \oplus s_{11}) \oplus (d_{11} \oplus d_{50} \oplus c_{10} \oplus c_{11}) \rightarrow t_1 \\
&= (s_{20} \oplus s_{21}) \oplus (d_{21} \oplus d_{60} \oplus d_{71} \oplus c_{20} \oplus c_{21}) \rightarrow t_2
\end{aligned}\right\} \qquad (7-51)
$$

を得る。* 上式で第1行目を t_0，第2行目を t_1，第3行目を t_2 と表わすと，T.S.C. 論理は次のように表わすことができる。

$$
\begin{cases}
C_0 = t_0 \cup t_1 \cup t_2 \\
C_1 = \overline{t_0 \cap t_1 \cap t_2}
\end{cases} \qquad (7-52)
$$

以上はSGに対する検査論理であったが，CGに対しては（7－51）式，（7－52）式で c_{00}～c_{21} を入力させないこととし，s'_{00}～s'_{21} を生成検査ビット $\hat{c}_{00}$～$\hat{c}_{21}$ とすればよい。（7－52）式の（C_0, C_1）を作成する回路は42ゲートとなる。

＊ 32ビット，64ビットの拡張構成を考慮している。（$s_{10}\ s_{11}$, $s_{20}\ s_{21}$）は途中パリティチェック結果を示す。（$s''_{10}\ s''_{11}$, $s''_{20}\ s''_{21}$）は，他モジュールからの（$s_{10}\ s_{11}$, $s_{20}\ s_{21}$）に相当する途中パリティチェック結果であり，（7－51）式では t_0 中に補正項として使用している。

〔２〕　SD，COR，PGに対する検査回路

SDに対しても定理７－４，定理７－５を適用して，シンドロームと誤りポインタとの間に検査論理を構成することができる。具体的にT.S.C.の条件を満足する検査回路を構成すると５４ゲートとなる。

ここでは検出能力はやや劣るが，一層簡明な検査手法を適用しよう。提案する検査論理は，単一バイト誤り訂正符号においてシンドロームと誤りバイトポインタとの間に，非零のシンドロームに対して単一の誤りバイトポインタ指示が存在しているかどうか，あるいはすべて零のシンドロームに対して誤りバイトポインタ指示がないかどうかの関係を与えるものである。誤りバイトポインタを情報部に対してG_i，i＝0，1，…，n－r－1，検査部に対してG_{c_j}，j＝0，1，…，r－1とし，シンドロームをGF(2)上で表現したとき，次の関係から自己検査論理を得ることができる。

$$\begin{cases} C_0 = \overline{\bigcup_{i=0}^{n-r-1} \bigcup_{j=0}^{b-1} s_{i,j}} \\ C_1 = \left(\sum_{i=0}^{n-r-1 \oplus} G_i \right) \oplus \left(\sum_{j=0}^{r-1 \oplus} G_{c_j} \right) \end{cases} \qquad (7-53)$$

ここで，(C_0, C_1)＝(0，1)または(1，0)で正しいと判定し，(0，0)または(1，1)で誤り検出と判定する。

この論理においては，訂正不可能な多重（偶数）バイト誤り検出の場合に(C_0, C_1)＝(0，0)となって，この回路により誤り検出することができる。但し，訂正不可能な入力に対して，１個の誤りバイトポインタのstuck at '1' 故障は検出できない。このような意味で，(7－53)式はPartially Self－Checking*（P.S.C.）となる。しかし，実用的には訂正不可能な入力は稀にしか生じないことを考えればその使用は問題ない。（７－５３）式に基づく回路は28ゲートとなる。

（７－５３）式はすべてのシンドローム，誤りバイトポインタに対して成立する検査論理であるため，符号を拡張して図７－１０を複数個使用する場合には，自分以外の他の最大３個の回路における誤りバイトポインタに関する情報を入手する必要がある。そのため，図７－１０におけるSCの回路においては，少くとも３本の入力信号を必要とする。

＊　定義７－１の脚註参照

次に，CORとPGに関する検査を考えよう。これは，7－3－5に示した（7－28）式を使用して構成する。8ビット単位にパリティビットを付加すると，この回路は32ゲートとなる。

〔3〕 まとめ

図7－10に示す回路の検査は，以上に示した3種の検査論理によって実現できた。最後に，これら（C_0，C_1）の3対の結果をT.S.C. AND回路[90]によって最終的に一対の（C_0，C_1）として出力させる。これらの関係を図.7－16に示す。その結果，全体の自己検査性検査回路は115ゲートとなる。これは，もとの回路に対して40％のゲート増加に等しい。

一方，図7－14に示した奇数重み列S4EC／S2EC／SEC－DED符号の回路に対しても，上記と全く同様の検査論理を構成することができる。この場合には，S4ECの機能とS2EC／SEC－DEDの機能とで検査論理を共用する回路を除けば切り換える形となる。検査回路は全体で215ゲートとなり，約32％のゲート増加に等しい。

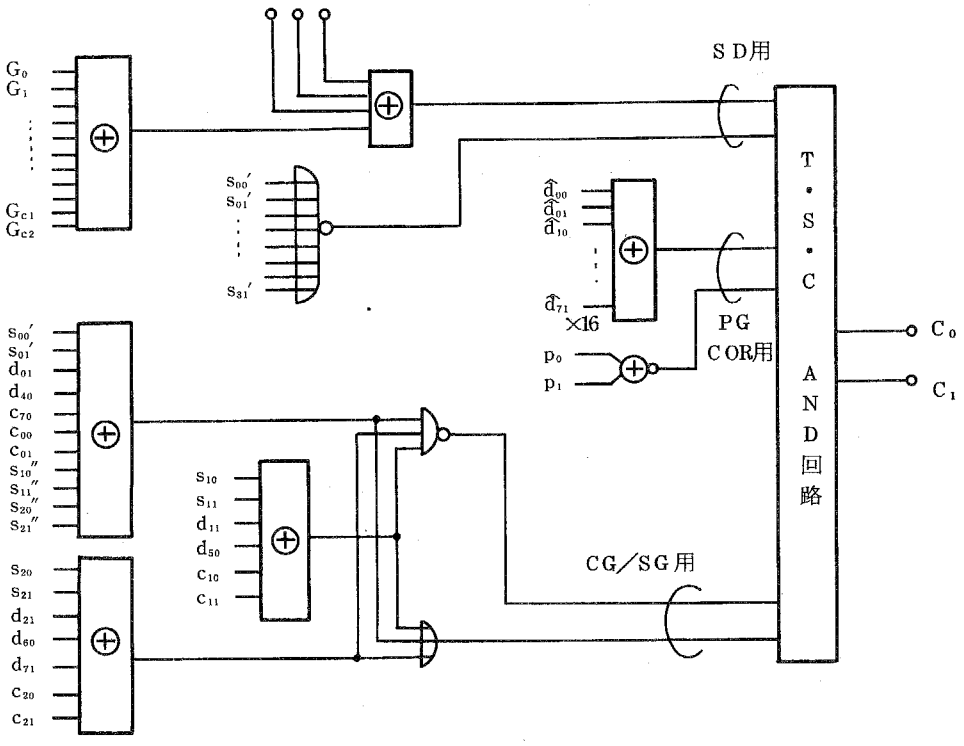


図7－16 S2EC/SEC－DED用符号化・復号化回路(図7－10)に対する自己検査性検査回路

7－4－4　LSI試作

本回路に対するLSI化は，西独ジーメンス社によるLSI[91]，およびモトローラ社によるMSI[92]が発表されている。これらのLSI，MSIは，いずれもチェックビットの生成／シンドロームの生成（CG／SG）の部分に対する回路である。1979年武蔵野通研において，日本電気（株）と協力してこの回路のLSI*を実用化した。[93][94]　このLSIは，CG／SGだけでなく，SD，COR等も含み，単体のLSIで誤り訂正の機能を有するものである。特に高速な復号とLSI端子数の制限から，適用した符号は巡回性奇数重み列SEC－DED符号である。符号長の拡張性の観点から，1LSIで処理する情報ビット数は16ビットであり，これを2個使用して32ビット，4個使用して64ビットを実現できる構成としている。

（7－40）式に相当する基本部分マトリクスH_0を次に示す。

$$H_0 = \left[\begin{array}{cccccccccccccccc|cccccc} 1&1&1&1&1&1&1&1&1&0&0&1&0&0&1&0&1&0&1&0&0&0&0&0 \\ 1&1&1&0&0&0&0&0&1&1&1&1&1&1&1&1&0&1&0&1&0&0&0&0 \\ 0&0&0&1&1&1&0&0&1&1&1&1&0&0&1&1&1&1&0&0&1&0&0&0 \\ 1&0&0&1&0&0&1&1&0&1&0&1&1&1&1&0&1&1&0&0&0&1&0&0 \\ 0&1&0&0&1&0&1&0&0&0&1&0&1&0&1&0&1&1&0&0&0&0&1&0 \\ 0&0&1&0&0&1&0&1&0&0&0&1&0&1&0&1&1&1&0&0&0&0&0&1 \\ \hline 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{array}\right] \qquad (7-54)$$

この部分マトリクスはすべて'0'要素からなる行を2行有する構成であることから，情報ビット数16ビットの場合には6ビットの検出ビット数で実現できる。（7－54）式では，情報部は18ビット分存在するが，32ビット，64ビットのときでも定理2－1からHマトリクスのすべての行が偶数重み行となるよう列ベクトルを選択するため，2列分余分に加えている。

1LSI内の回路構成を図7－17に示す。図7－1に含まれない回路としてSEL，DR，Cが存在する。SELは外部からの入力情報と訂正後の情報とを選択する回路である。訂正後の情報は部分書込み動作時に使用する。[99]　DRはSELで選択された情報を保持する回路である

＊　本LSIは超大容量記憶装置用制御装置または磁気ディスク制御装置における制御メモリの誤り訂正用に開発された。[106]

とともに，故障診断時にスキャンインデータを蓄えたり，診断結果のデータを蓄えてビット単位にスキャンアウトするための回路である。CはSELの選択，DRの動作，情報ビット数の指定，書込み／読出し等の指示を制御する回路である。LSI回路の諸元を表7－11に示す。本LSIはマスタスライス形式で作成されており，この諸元を表7－12に示す。

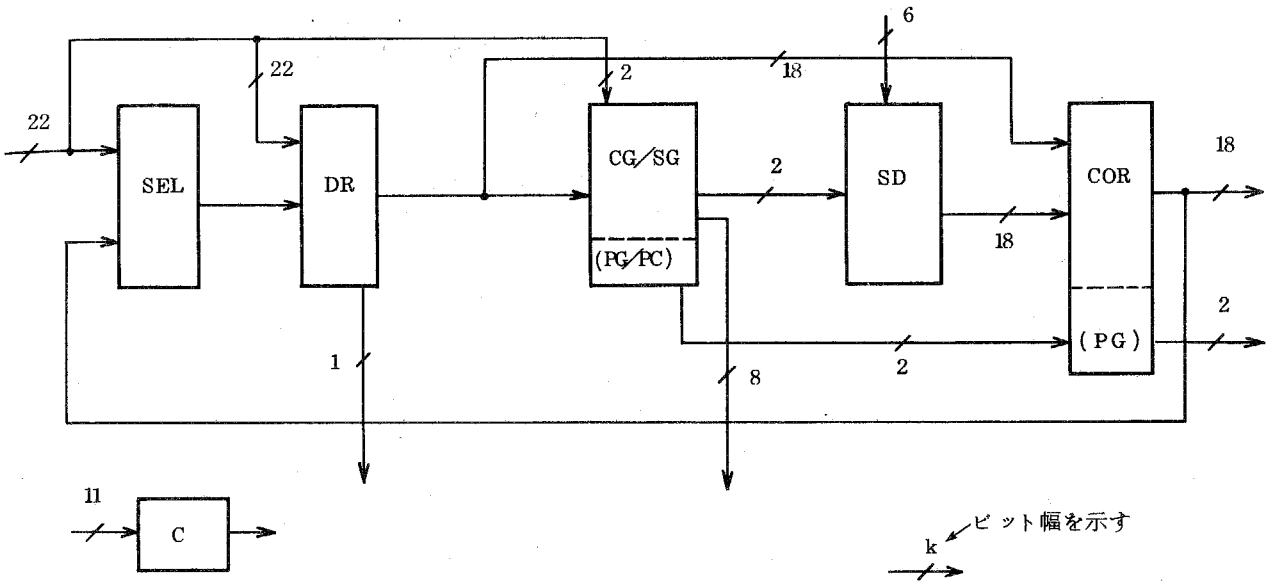


図7－17　回　路　構　成

表7−11　LSI回路諸元

符号機能	SEC−DED
Hマトリクス	巡回性符号 奇数重み列 偶数重み行
情報ビット数	18ビットスライス (16ビット/18ビット，32ビット/36ビット，64ビット/72ビット)*
検査ビット数	6ビット(16ビット長の場合) 8ビット(32/36ビット長の場合，64/72ビット長の場合)
パリティ検査/生成	9ビット単位 奇数パリティ
部分書込み	直列処理**
故障診断	スキャンイン/アウト
使用ゲート数	685ゲート***
入出力信号数	72信号(入力42信号，出力30信号)
復号遅延時間	TYP { 22ns (16ビット長のとき) 24ns (32ビット長のとき) 28ns (64ビット長のとき)
消費電力	約1.5w

* 32ビット/36ビット等の表現は，32ビット用としても36ビット用としても使用できることを意味する。但し，36ビット用等の数値の大きい方に対しては偶数重み行の性質は符号として有さない。

** 読出し訂正後のデータに対して新規の書込みデータを作成し，これから検査ビットを生成する処理過程。一方，高速処理を目的に並列処理する手法もある。(98)

*** EX−ORゲートを4ゲート，F/Fを6ゲートと換算。

表７－１２　ＬＳＩ諸元

ＬＳＩの形式	マスタスライス
回路形式	LCML（２段縦形）*
集積度	max　1200ゲート
トランジスタ数	3840個（240×16）**
チップサイズ	5 mm × 5 mm
セル数	384***
配線	３層
基本ゲートの遅延時間	0.9 ns
電源電圧	－3.3 V
消費電力	TYP　1.5w
入出力端子数	80本（信号72本，電源4本，アース4本）
入出力インタフェース	LCMLレベル（０V，－0.5 V）

* Low Level Current Mode Logic
電源（－3.3 V）とグランドとの間にLCML回路を２段縦積みするシリーズゲート

** 誤り訂正回路を構成するために使用したトランジスタ数　1980個（52%）

*** 〃　〃　セル数　238　（62%）

7−5 結　　言

本章では，高速な符号化・復号化を有する回路構成および具体的に各種のバイト系符号に対する回路ゲート量，回路遅延の関係を求めた。また，この回路の高信頼化を目的に自己検査性検査回路の構成法を示し，さらに巡回性符号を採用した本回路のＬＳＩ 構成を示した。主な結論を以下に示す。

(1) ＳＥＣ−ＤＥＤ−ＳｂＥＤ 符号の符号化・復号化回路は，ゲート量，回路遅延において，ＳＥＣ−ＤＥＤ 符号の場合と比較して同一規模で実現できる。

(2) 組合せ回路を用いた並列符号化・復号化回路は，３２ビット～１２８ ビットの実用的な情報ビット数，符号機能の範囲内ではゲート量は３００～3,０００ ゲートであり，復号のための回路遅延は８～１４ゲート段である。

(3) 符号化・復号化回路のゲート量は，符号機能よりむしろ情報ビット数，Ｈマトリクスの重みに主として依存する。

(4) 符号化・復号化回路の高信頼化を目的に，自己検査性を有する検査回路を構成した。検査の考え方は本回路をいくつかに分割し，それぞれに対しその入出力から共通の論理関係を求め，それらを互いに比較する手法を採用した。この手法は，Ｔ．Ｓ．Ｃ（Totally Self−Checking）論理が容易に構成できる特徴を有する。

(5) ＳＥＣ−ＤＥＤ 符号，Ｓ２ＥＣ 符号に対し，Ｔ．Ｓ．Ｃ．検査回路を具体的に構成した。もとの符号化・復号化回路に対するゲート量増加比は，情報ビット数３２ビットの場合で約３０％，６４ビットの場合で２５％である。

(6) 符号化・復号化回路に対するＬＳＩ 構成を得るため，巡回性符号を採用した。ＬＳＩ 化に当っては，①符号長の拡張性，②多機能性，③Ｔ．Ｓ．Ｃ．検査回路の内蔵による高信頼化，を考慮した。符号機能の包含関係を考慮して中間拡大体を使用した符号構成を求め，これにより複数個の符号機能を包含した多機能性と回路の共通化を実現した。また，巡回性符号の基本部分Ｈマトリクスに，すべて'0'の元を有する行を出来るだけ多く設けることで，符号長が短かい場合にも最少検査ビット数となる構成を求めた。

(7) 具体的にＳＥＣ−ＤＥＤ の機能と Ｓ２ＥＣ の機能を有し，この２機能を選択して使用できるＬＳＩ 回路と，さらに Ｓ４ＥＣ 機能を有し３機能を選択して使用できるＬＳＩ 回路を設計した。前者のＬＳＩ を 1, 2, 4 個使用して所定の接続を行うことにより，それぞれ１６ビット，３２ビット，６４ビットの情報ビット数に対する回路が構成できる。後者のＬＳＩ は，１個～６個使用することにより，１６ビットから１２８ ビットの情報ビット数に対する回路が構成できる。

(8) 1200 ゲートマスタスライスLSI による試作例の概要を示した。SEC−DED 機能を有する巡回性符号を採用し，16ビットを基本情報ビット数としている。本LSI を2個使用して32ビット用，4個使用して64ビット用に構成できる。本LSI はLCML（Low level CML）によるマスタスライス形式を採用しており，$5^{mm} \times 5^{mm}$ のチップ，72信号端子（全80端子）を有し，復号化回路遅延は22ns（16ビットのとき）である。

第8章　算術論理演算回路（ＡＬＵ）に対する誤り検出・訂正

8－1　緒　　言

ＡＬＵに対する誤り検出・訂正手法は，符号を使用しない方法と使用する方法に分類できる。符号を使用しない方法の代表的な例は，二重化により出力を比較して誤りを検出する方法[97]と，三重化により出力を多数決回路に入力して訂正する方法である。符号による方法は，大別してパリティ符号による方法と剰余符号による方法がある。パリティ符号による誤り検出は，その検出回路の簡明さの点から使用されることが多い。しかし，加算回路に対してこの符号を採用する場合には，キャリ誤りによる影響を考慮しなければならない。また，他の論理演算に対してこの符号は特別の考慮を払わない限り一般には適用できない。一方，剰余符号による方法は，ＡＮ符号等による１ビット誤り訂正の機能[52]，または単一バイト誤り訂正の機能[53]を有する符号が提案されている。この符号は，加算，乗算等の算術演算に対する誤り検出・訂正に適するが，検出・訂正回路が複雑となる欠点を有する。そこで，パリティ符号と剰余符号を組合わせて，両符号の特長を生かした組合せ符号（Combination Code）が最近提案されている。[61]

本章では，新しく提案する手法への導入とその位置づけを明確にするため，今まで提案されている主な誤り検出・訂正手法を概説する。次に，加算回路を対象に，パリティ符号と回路の二重化を組合わせた方法による誤り検出・訂正手法と，水平・垂直パリティ符号による誤り検出・訂正手法を提案する。また，主記憶装置等で使用しているパリティを基本とする符号を使用して，ＡＬＵの誤り検出・訂正ができるための条件と具体的な構成法を提案する。ＡＬＵ中に誤りはなく，ＡＬＵへの入力に誤りが存在する場合について，このパリティを基本とする符号を使用して誤り検出・訂正できるための条件についても示す。

以上に示したＡＬＵに対する各種の誤り検出・訂正手法の位置づけを図８－１に示す。このうち，本章で新しく提案する手法を二重の丸で囲んでいる。

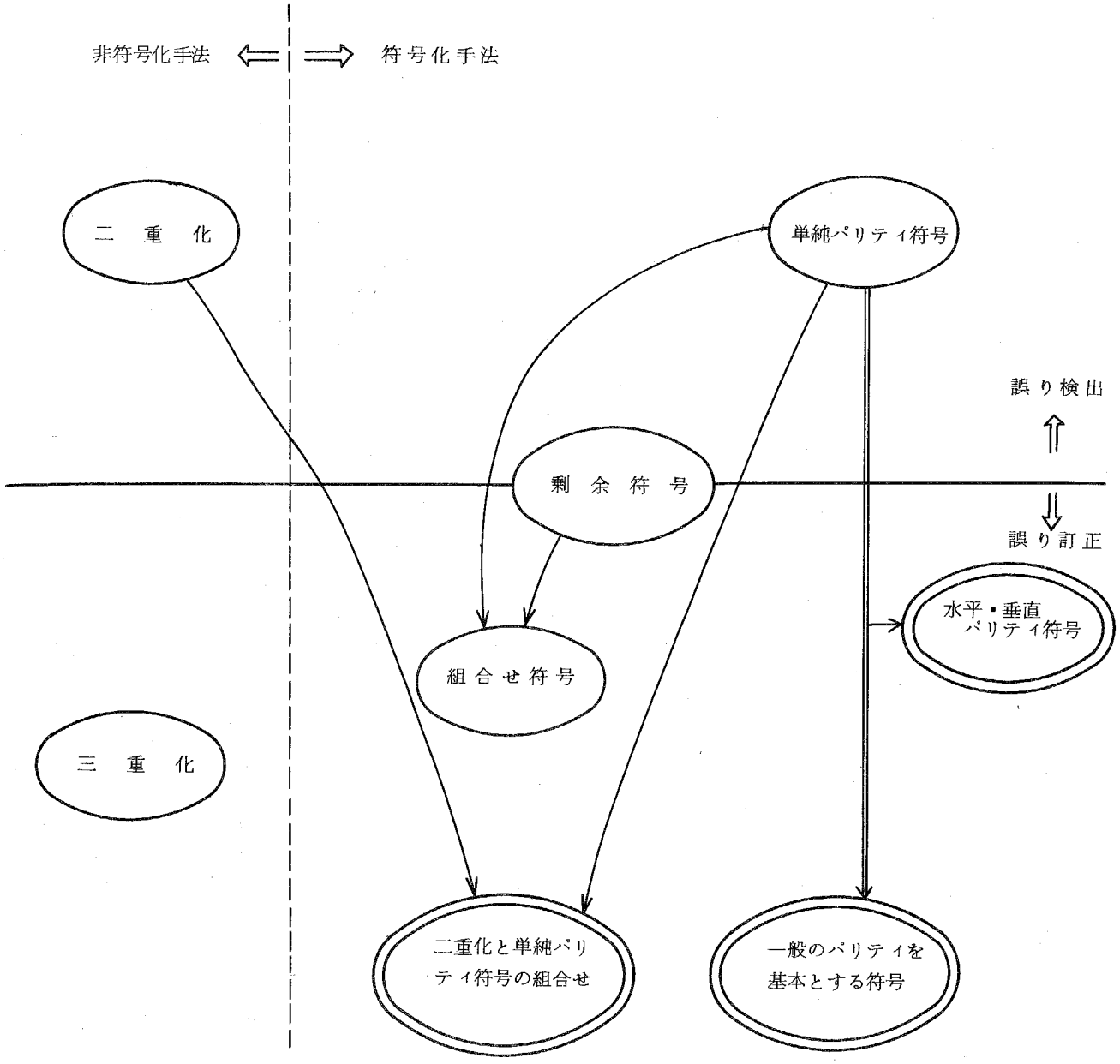


図８－１　ＡＬＵに対する各種の誤り検出・訂正手法の関係

8－2　従来の誤り検出・訂正手法

算術論理演算回路（ＡＬＵ）に対する誤り検出・訂正手法は，大きく分類すると次のようになる。

(1)　回路の二重化・三重化による誤り検出・訂正手法

(2)　パリティ符号による誤り検出手法

(3)　剰余符号による誤り検出・訂正手法

(4)　パリティ符号と剰余符号を組合わせた組合せ符号による誤り検出・訂正手法

このうち(1)による手法は最も構成が容易であり，検査ビットも必要なく，簡明な手法である。(2)はパリティ符号を使用して主として誤りの検出を行う手法である。一般に，パリティ検査はメモリやデータパス系論理回路等の演算が行われない回路の検査には適するが，データが演算によって変化する演算回路に対しては必ずしも適さない。適用に当っては，各種の工夫が必要である。(3)の剰余検査は，一般に加算，乗算等の算術演算の検査に適しており，キャリ等の考慮を払うことなく容易に検査が実行できる。これは，ＡＮ符号等により，誤り訂正も可能である。しかし，剰余検査は論理演算の検査には適さず，これに対しては別途考慮を必要とする。(7)(4)はパリティ符号と剰余符号を組合わせた組合せ符号（Combination Code）による手法であり，T.R.N.Rao, H.J.Reinheimerにより提案された手法(61)である。これは，簡単な検査が行えるパリティ符号と算術演算に適する剰余符号の特長を生かして，比較的実用的な誤り検出・訂正が実行できる手法である。

8－2－1　回路の二重化・三重化による誤り検出・訂正手法

本手法による構成は簡明であり，これを図８－２に示す。

本手法は一般に以下の特徴を有する。

(1)　構成が簡明であり，すべての組合せ回路に対して適用できる。

(2)　検査ビット等の冗長な符号ビットを必要としない。

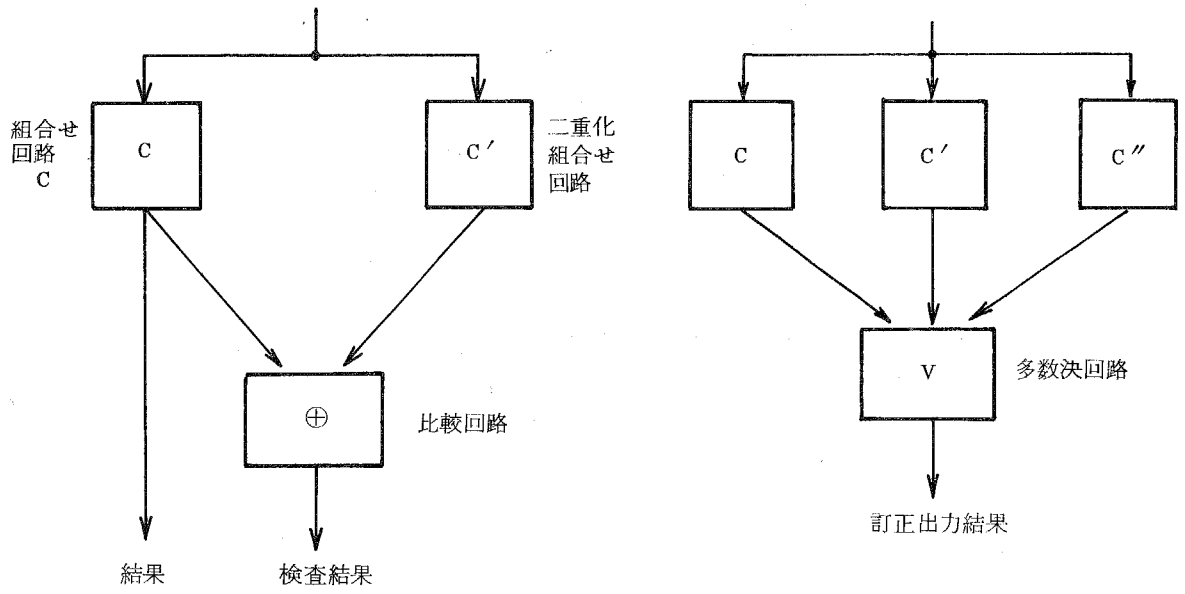
(3)　二重化による検査では，出力結果の比較ですべての誤りを検出でき，検出能力は高い。

(4)　三重化による訂正では，３回路の出力を多数決回路に入力させることにより，すべての誤りを訂正でき，訂正能力は高い。

(5)　高速な誤り検出・訂正が可能である。

(6)　余分に必要とするゲート量は大きい。

(7)　入力誤りは検出・訂正できない。



（ａ）二重化による誤り検出　　（ｂ）三重化による誤り訂正

図８－２　二重化・三重化による誤り検出・訂正

８－２－２　パリティ符号による誤り検出手法

ＡＬＵへ入力する２個のデータＡ，Ｂを次のように定義する。

$$\begin{aligned} A &= \{ a_0, a_1, \cdots, a_{k-1} \} \\ B &= \{ b_0, b_1, \cdots, b_{k-1} \} \end{aligned} \qquad (8-1)$$

このとき，それぞれに対するパリティビット P_A，P_B は次の関係から作成される。

$$P_A = \sum_{i=0}^{k-1}{}^{\oplus} a_i$$

$$, \quad \sum^{\oplus} \ ; \text{mod. 2 和} \qquad (8-2)$$

$$P_B = \sum_{i=0}^{k-1}{}^{\oplus} b_i$$

ＡＬＵと入出力の関係を図８－３に示す。（ｋ＋１）ビットの２入力〔Ａ⋮P_A〕，〔Ｂ⋮P_B〕に対し，演算結果は〔Ｙ⋮P_Y〕である。ここで，出力Ｙおよび P_Y は，次の関係を満足していなければならない。

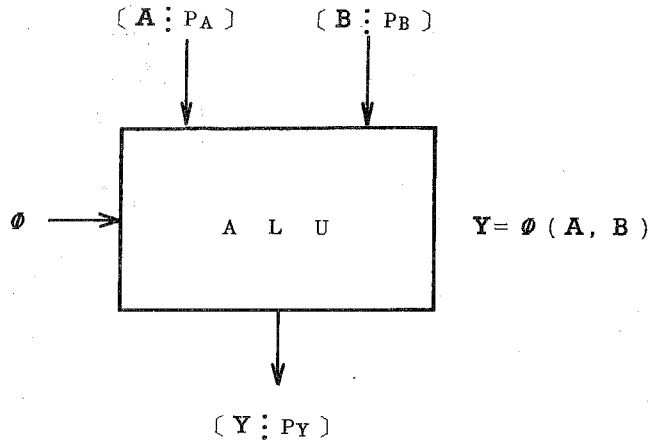


図８－３　ＡＬＵに対する入出力モデル

$$Y = [\, y_0,\ y_1,\ \cdots,\ y_{k-1} \,] \qquad (8-3)$$

$$P_Y = \sum_{i=0}^{k-1}{}^{\oplus}\ y_i \qquad (8-4)$$

ＡＬＵの演算をΦとするとき，ＹとＡ，Ｂの関係を次式で表現する。

$$Y = \Phi(A, B) \qquad (8-5)$$

すなわち，入力Ａ，Ｂに対して，演算Φを実行することによる結果がＹである。

〔１〕　加算演算に対する誤り検出法

加算回路は，キャリ生成の手法から，リップル型，キャリルックアヘッド型が存在する。本研究では高速なキャリ生成が実現できるキャリルックアヘッド（ＣＬＡ）型加算回路を対象とする。特に，情報ビット数が多い場合には，バイトに分割し，これをグループとしてこの間の高速な伝搬キャリ（グループキャリ）を作成して構成するグループルックアヘッド加算回路[95]を採用する。

加算回路に対する検査としては，以下に示す２種類の検査が存在する。[6]

① ハーフサムパリティ検査

ハーフサムビットを

$$h_i = a_i \oplus b_i \qquad i = 0,\ 1,\ \cdots,\ k-1 \tag{8-6}$$

として，加算回路中のハーフサム結果のパリティ和と，P_A，P_B を用いて，次の検査が実行できる。

$$\left.\begin{aligned} S &= \left(\sum_{i=0}^{k-1}{}^{\oplus}\ h_i\right) \oplus P_A \oplus P_B \\ S &= 1 \quad ;\text{誤り検出} \\ S &= 0 \quad ;\text{誤りなし} \end{aligned}\right\} \tag{8-7}$$

この検査は，加算回路の途中結果に対する部分的な検査であり，最終的な加算結果およびキャリに対する十分な検査となっていない。

② フルサムパリティ検査

加算結果Yのパリティ和P_Yとキャリのパリティ和P_Cを定義する。

$$\begin{aligned} P_Y &= \sum_{i=0}^{k-1}{}^{\oplus}\ y_i \\ &= \sum_{i=0}^{k-1}{}^{\oplus}\ (a_i \oplus b_i \oplus c_{i-1}), \qquad c_{-1} = c_{in} \end{aligned} \tag{8-8}$$

$$P_C = \sum_{i=0}^{k-2}{}^{\oplus}\ c_{i-1} \tag{8-9}$$

これらを用いて，次のようなフルサムパリティ検査が実行できる。

$$\left.\begin{aligned} S &= P_Y \oplus P_A \oplus P_B \oplus P_C \\ S &= 1 \quad ;\text{誤り検出} \\ S &= 0 \quad ;\text{誤りなし} \end{aligned}\right\} \tag{8-10}$$

但し，この検査では，加算回路中のキャリ誤りはP_C，P_Y の双方に影響を与え，Sとして誤り検出できない。すなわち，c_iから始まるqビットのバースト誤りは，y_{i+1}から始まるqビットの加算結果の誤りとなり，総計2q（偶数）ビットの誤りとなる。従って，（8－10）式の検査ではこのような誤りは検出できないことになる。そこで，これに対してはキャリディペンデントサム（CDS）加算器[96]を構成して（8－10）式により誤り検出する方法と，キャリ生成回路を二重化して誤り検出する方法[6]がある。

〔2〕 論理演算に対する誤り検出法

ここでは，基本的な論理演算であるEX－OR，AND，ORについて，パリティ符号による誤り検出についてのべる。

まず，EX－OR演算に対しては，

$$y_i = a_i \oplus b_i \tag{8-11}$$

とすれば，$\overset{k-1}{\underset{i=0}{\sum^{\oplus}}} y_i = P_A \oplus P_B$ の関係より，

$$\left.\begin{aligned} S &= \left(\overset{k-1}{\underset{i=0}{\sum^{\oplus}}} y_i\right) \oplus P_A \oplus P_B \\ S &= 1, \quad ;\text{誤り検出} \\ S &= 0, \quad ;\text{誤りなし} \end{aligned}\right\} \tag{8-12}$$

として，容易に検査できる。一方，AND，OR演算に対しては，

$$\left.\begin{aligned} \alpha_i &= a_i \cap b_i \\ \beta_i &= a_i \cup b_i \end{aligned}\right\} \tag{8-13}$$

に対して，次の関係が成立する。

$$\begin{aligned} \alpha_i \oplus \beta_i &= (a_i \cap b_i) \oplus (a_i \cup b_i) \\ &= a_i \oplus b_i \end{aligned} \tag{8-14}$$

これから，EX−OR演算の場合と同様にして，次のように検査できる。

$$\left.\begin{aligned} S &= \{ \sum_{i=0}^{k-1}{}^{\oplus} \ (\alpha_i \oplus \beta_i) \} \oplus P_A \oplus P_B \\ S &= 1 \quad ; \text{誤り検出} \\ S &= 0 \quad ; \text{誤りなし} \end{aligned}\right\} \qquad (8-15)$$

一方，加算回路を構成する生成関数（Generate Function）g_i，伝達関数（Transmit Function）t_i，ハーフサム h_i は，それぞれAND回路，OR回路，EX−OR 回路に等しく，しかも加算結果 y_i に対して次の関係にある。

$$\begin{aligned} y_i &= g_i \oplus t_i \oplus c_{i-1} \\ &= h_i \oplus c_{i-1} \end{aligned} \qquad (8-16)$$

これから，次式が成立する。

$$\frac{dy_i}{dg_i} = 1, \quad \frac{dy_i}{dt_i} = 1, \quad \frac{dy_i}{dh_i} = 1 \qquad (8-17)$$

これより，〔1〕で示した加算回路に対する検査結果を常に調べることにより，これらの基本論理演算の誤りも等価的に検査できることになる。

8－2－3　剰余符号による誤り検出・訂正手法

〔1〕　剰余符号による誤り検出

加算演算を対象に剰余符号による誤り検出・訂正手法についてのべる。整数Xに対してZで割った剰余を $|X|_Z$ と表わす。このとき，次式が成立する。

$$0 \leq |X|_Z \leq Z-1 \qquad (8-18)$$

入力X_1とX_2との加算演算に対して

$$|X_1+X_2|_Z=||X_1|_Z+|X_2|_Z|_Z \qquad (8-19)$$

が成立する。これから，剰余符号〔X_1,$|X_1|_Z$〕，〔X_2,$|X_2|_Z$〕を用いた加算演算に対する検査は図８－４に示す構成となる。（８－１９）式から，この検査はキャリによる影響がないことから簡明な検査手法であるが，図８－４に示す検査回路（破線で囲んでいる）は，パリティ検査による手法に比較して一般にゲート量は大きくなる欠点を有する。

以上の剰余符号は，剰余１個を付加した符号である。これを複数個付加した符号は多剰余符号（Multi－Residue Code）[54][55]と呼ばれる。一般に，ｔ剰余符号はｔ重誤りを検出するか，または⌈ｔ／2⌉重誤りを訂正する能力をもつと言われる。ここで，⌈ｘ⌉はｘを越えない最大整数である。従って，ｔ＝２とする２剰余符号（Bi－Residue Code）[56]は単一誤り訂正の機能を有する符号となる。

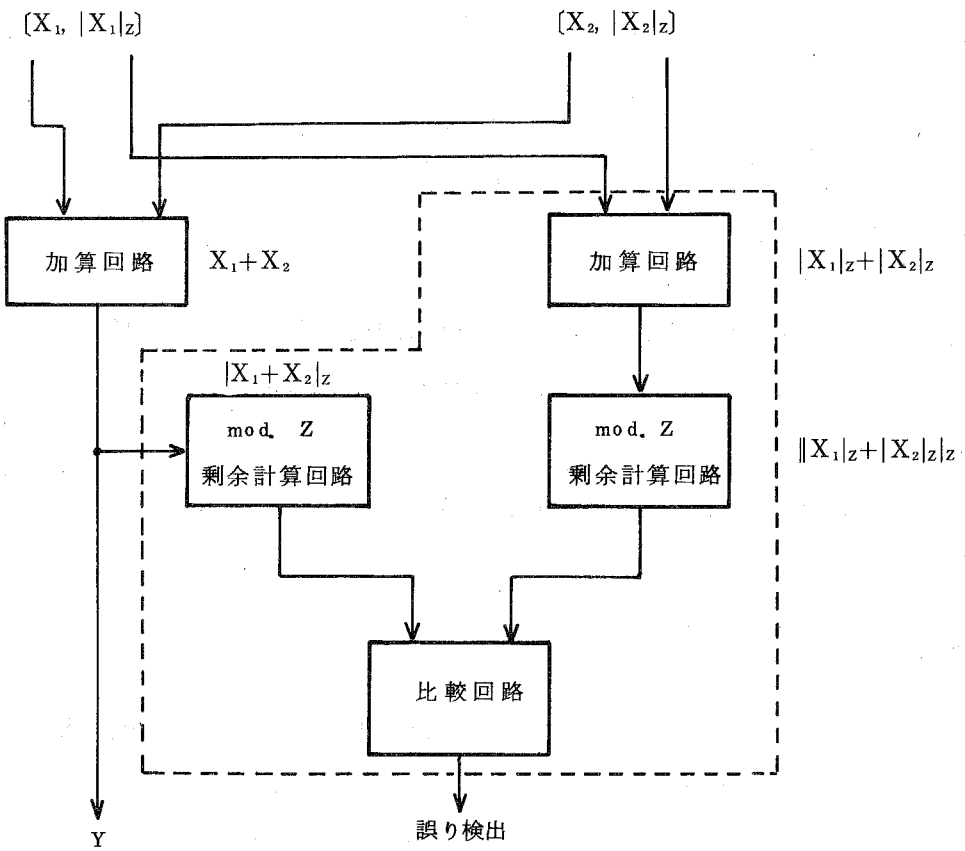


図８－４　剰余符号による加算演算に対する誤り検出手法

〔2〕　AN符号による誤り訂正(7)

剰余符号の形として，もとの数NにAを乗じてANなる数を作成した符号をAN符号と呼ぶ。この符号は検査部が分離されない符号である。Nの範囲が0～m－1のときmを情報域（range of information），Aを生成数（generator）と言う。Nをr進表示したとき，Aはrとは互いに素となるように選ばれる。2個の数N_1（<m），N_2（<m）に対して，加法をmod.M（M=A・m）で定義すると

$$(AN_1+AN_2)_{mod.M}=A\{(N_1+N_2)_{mod.m}\} \qquad (8-20)$$

となり，AN符号は算術演算に適することがわかる。

D.T.Brown．W.W.Petersonにより単一誤り訂正AN符号が提案されている。(52)　この符号は単一誤り訂正BP符号と呼ばれる。本符号を求める理論的な背景は省略して結果のみを次に示す。r＝2の場合，Aに対するm，M＝A・m，情報ビット数k，検査ビット数rの関係を表8－1に示す。

次に，P.G.Neumann，T.R.N.Raoにより単一バイト誤り訂正AN符号が提案されている。(53)　この符号は単一バイト誤り訂正NR符号と呼ばれる。A＝（2^b-1）p，p＝素数としたときのA，pに対するバイト長b，A・M，符号ビット長n，検査ビット数r，情報ビット数kの関係を表8－2に示す。

以上のAN符号は，符号語の形が情報部と検査部が分離されず（非分離符号），しかも演算実行中に情報部と検査部が区別されて処理されない（非組織符号）ことから，復号化において問題がある。この点を解決するため，組織的なAN符号の構成が提案されている。

組織符号として構成したAN符号は，gAN符号と呼ばれ，適当なAとMを選ぶことにより単一誤り訂正の機能を与えることができる。(57)　これを表8－3に示す。

表８－１　単一誤り訂正ＢＰ符号

A	m	M=A・m	情報ビット数	検査ビット数
11	3	2^5+1	1	4
13	5	2^6+1	2	4
19	27	2^9+1	4	5
23	89	$2^{11}-1$	6	5
29	565	$2^{14}+1$	9	5
37	7085	$2^{18}+1$	12	6
47	178481	$2^{23}-1$	17	6
53	1266205	$2^{26}+1$	20	6
59	9099507	$2^{29}+1$	23	6
61	17602325	$2^{30}+1$	24	6
67	128207979	$2^{33}+1$	26	7
71	483939977	$2^{35}-1$	28	7
79	6958934353	$2^{39}-1$	32	7
83	26494256091	$2^{41}+1$	34	7
101	5099523830125	$2^{50}+1$	43	7

表８－２　単一バイト誤り訂正ＮＲ符号

バイト長	p	A=(2^b-1)p	A・M	n	r	k
b=2	23	69	$4^{11}-1$	22	7	15
	47	141	$4^{23}-1$	46	8	38
	71	213	$4^{35}-1$	70	8	62
	79	237	$4^{39}-1$	78	8	70
b=4	31	465	16^5-1	20	9	11
	53	795	$12\cdot16^6+3$	27	10	17
	73	1095	16^9-1	36	11	25
	101	1515	$14\cdot16^{10}+1$	43	11	32
	139	2085	$10\cdot16^{17}+5$	71	12	59
	263	3945	$8\cdot16^{34}+7$	139	12	127
	503	7545	$16^{251}-1$	1004	13	991

表８－３　単一誤り訂正gAN符号

A	M	情報ビット数	検査ビット数
19	$(2^9+1)/19$	4	5
29	$(2^{14}+1)/29$	9	5
47	$(2^{23}-1)/47$	17	6
61	$(2^{30}+1)/61$	24	6
121	$(2^{55}+1)/121$	48	7

８－２－４　組合せ符号（Combination Code）による誤り検出・訂正手法

本符号の符号語を図８－５に示す。バイト長ｂビット単位に１ビットのパリティビットを有し，全体に対し１バイトの剰余符号を有する。

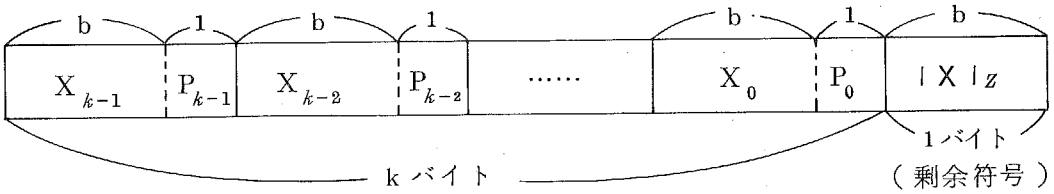
全体の符号長は

$$N = k \cdot b + k + b \quad \text{ビット} \tag{8-21}$$

であり，検査ビット数はｋ＋ｂビットである。

演算回路に入力する情報は〔Ａ：P_A：｜Ａ｜$_Z$〕，〔Ｂ：P_B：｜Ｂ｜$_Z$〕であり，演算Φを実行して結果〔Ｙ：P_Y〕を出力する。ここで，Ｙはｋ・ｂビットの演算結果であり，P_Y はｋビットの予知パリティ（predicted parity）ビットである。演算回路と誤り検出・訂正用回路を図８－６に示す。剰余計算部は演算Φに対応して，剰余｜Ａ｜$_Z$，｜Ｂ｜$_Z$ からＹに対する剰余を予知する。これを｜Ｙ｜$_Z$ とする。そこで，〔Ｙ：P_r：｜Ｙ｜$_Z$〕が符号語であるかデコーダ部で調べ，シンドローム（S_p，S_r）を作成する。ここで，S_p はパリティ検査結果のシンドロームであり，S_r は剰余検査結果のシンドロームである。図８－５に示すパリティ符号の付加方法より，S_p からただちに誤りバイト位置を知る。S_r から，誤りの大きさとバイト内の誤り位置を知り，以上からＹの誤りを訂正する。

〔 X ┆ P_X ┆ | X |$_Z$ 〕



（バイト単位に1ビットのパリティ符号を有する）

図８－５　組合せ符号の符号語

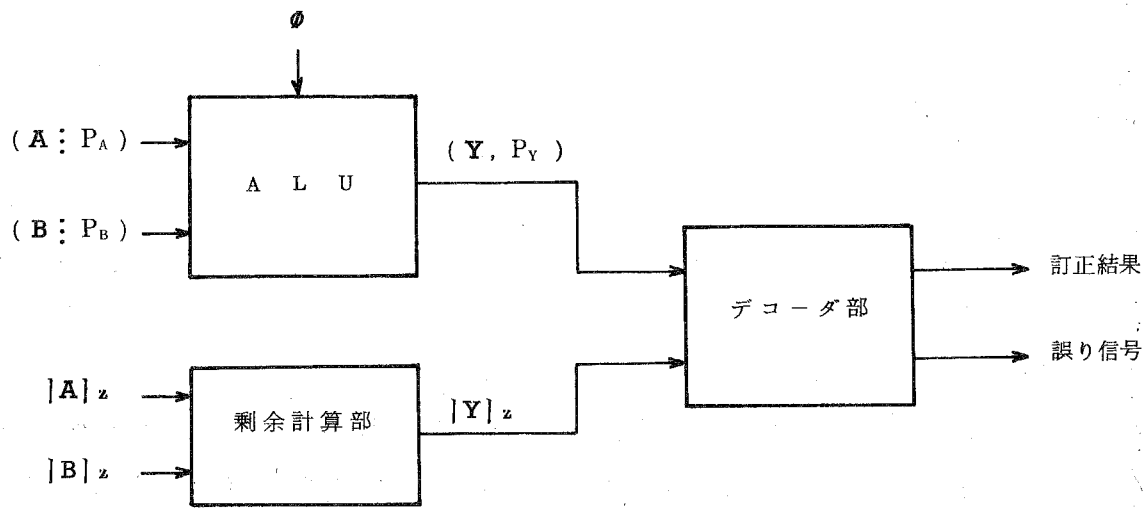


図８－６　組合せ符号による誤り検出・訂正手法

8－3　二重化とパリティ符号による加算回路の誤り訂正手法

加算回路に対する簡明な誤り訂正手法として，加算回路を二重化し，パリティ符号と組合わせた手法を提案する。本手法の訂正原理は次の点にある。

(i) 被検査測の加算回路をホスト加算回路とすれば，この加算回路に対するフルサムパリティ検査の実行

(ii) 両加算回路の出力結果に対するビット単位の比較

すなわち，(i)の検査からホスト加算回路の誤りを可能な限り検出し，さらに(ii)より具体的な誤り位置指摘を行い，これらの論理積から具体的な誤りビット位置を知る。これから，相当するビット位置の結果を反転して訂正する。

(i)のフルサムパリティ検査において検出能力を上げる観点から，使用する加算回路は，キャリディペンデントサム（CDS）加算回路が望ましい。この加算回路を使用すれば，奇数ビットの加算誤り，およびすべてのキャリ誤りはこのフルサムパリティ検査により検出できる。本手法の訂正能力はこのフルサムパリティ検査能力に依存しており，CDS加算回路を使用すれば，キャリ誤りのすべて（それによって生ずる加算出力のすべて）と加算出力の奇数ビット誤りを訂正できる。

本手法による回路構成を図8－7に示す。図中，太線で囲った回路が訂正用の回路であり，次の回路からなる。

(i)	二重化加算回路（キャリルックアヘッド型）	32
(ii)	フルサムパリティ検査回路	18
(iii)	比較回路	8
(iv)	誤り位置指摘回路	4
(v)	反転回路	8
(vi)	誤り判別回路	7

右に示した数値は，4ビット幅の場合のゲート量である。総計77ゲートであり，もとのCDS加算回路は60ゲートであるから，訂正用回路は約1.3倍のゲート量で構成できることになる。

上記の誤り判別回路は次頁に示す誤りを判別する回路である。ここで，フルサムパリティ検査結果をS，比較検査結果を$E_i = y_i \oplus y_i'$（$i = 0, 1, \cdots, k-1$）とする。ここで，y_i'は二重化加算回路の出力ビットである。

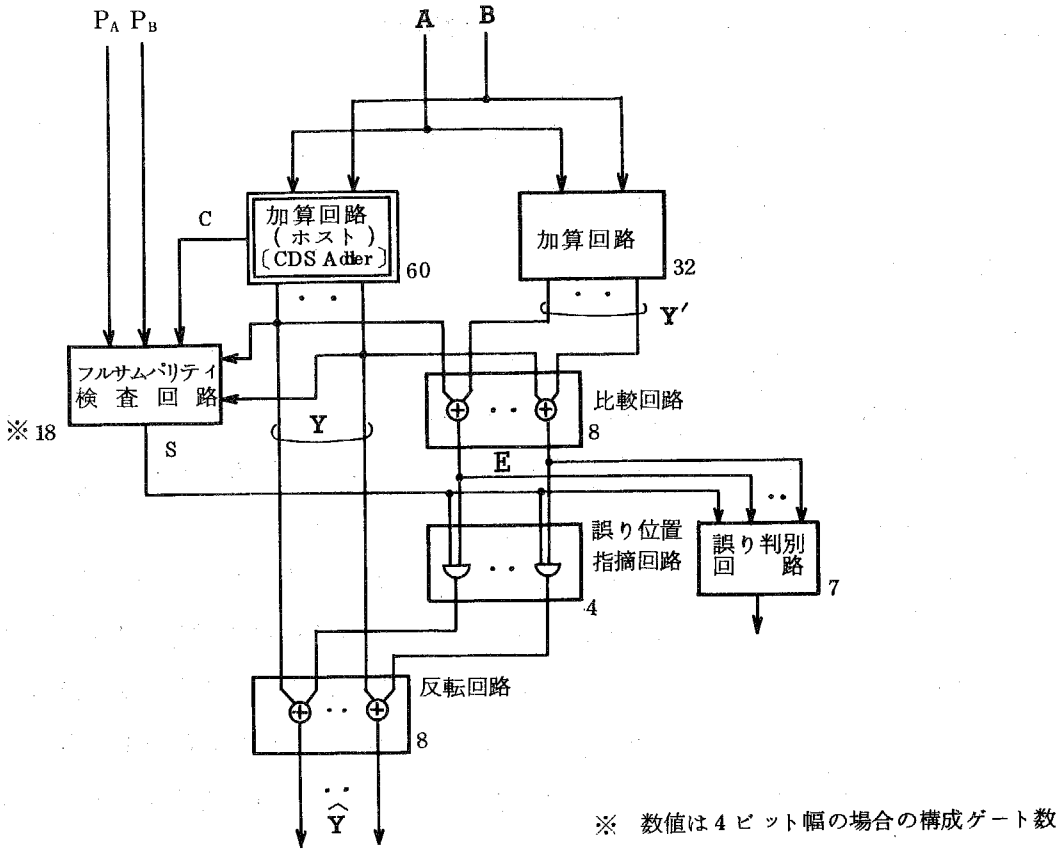


図８－７　二重化とパリティ符号による誤り訂正手法

①　S＝0　，すべての　E_i ＝0　　；誤りなし

②　S＝0　，いずれかのE_i ＝1　　；二重化加算回路または比較回路中の誤り，または加算結果の偶数ビット誤り（訂正不可）

③　S＝1　，すべての　E_i ＝0　　；訂正不可

④　S＝1　，複数個（t個）のE_i ＝1　；t重誤り訂正

特に④に対して具体的に訂正が行われる。

図８－７に示す回路構成から，①｛フルサムパリティ検査回路｝，②｛二重化加算回路および比較回路｝に対して，①②のいずれかに故障が生じても誤訂正を与えることはない。しかし，誤り位置指摘回路，反転回路中の故障は誤訂正をもたらす危険性を有する。これらの回路は，全回路中の２６％に相当する。これから，図８－７に示す回路は，その内の故障に対して誤訂正をもたらす危険性が比較的小さい回路構成と言える。

以上から，ホストの加算回路としてCDS加算回路を採用することにすれば，本手法の特徴は次に示すようになる。

(1) 訂正能力　　；加算結果の奇数ビット誤り，およびすべてのキャリ誤りにともなう加算結果の誤りを訂正可

(2) 訂正回路のゲート量　；もとのCDS加算回路の1.3倍のゲート量

(3) 検査ビット数　　；パリティビット　1ビット

(4) 訂正速度　　；7ゲート段数（16ビット幅の場合）

(5) 訂正回路中の故障　；全回路の74％中の単一故障は誤訂正を与えない。

8－4　水平・垂直パリティ符号による加算回路の誤り検出・訂正手法

簡単なパリティ符号を直交させた水平・垂直パリティ符号を用いて，加算回路の誤りを検出・訂正する手法を提案する。本手法の特徴を以下に示す。

1）　水平・垂直パリティ符号の特徴から，誤り検出・訂正回路にくり返し性を持たせることができる。

2）　誤り検出・訂正回路の構成から，回路自身の故障による誤訂正への影響は小さい。

3）　検査ビット数は，バイト長をｂビットとするとき，情報ビット数ｋ・ｂビットに対し，（ｋ＋ｂ）ビットとなる。

4）　バイト内奇数ビット誤り，およびバイト内のすべてのキャリ誤りは訂正できる（CDS加算回路使用の場合）。

8－4－1　訂正原理

〔1〕　水平・垂直パリティ符号

ｂビット幅を有するｋ個のバイトを情報部とする水平・垂直パリティ符号の構成を図８－８に示す。この符号の検査ビット数Ｒは，次式となる。

$$R = k + b \quad （ビット）$$

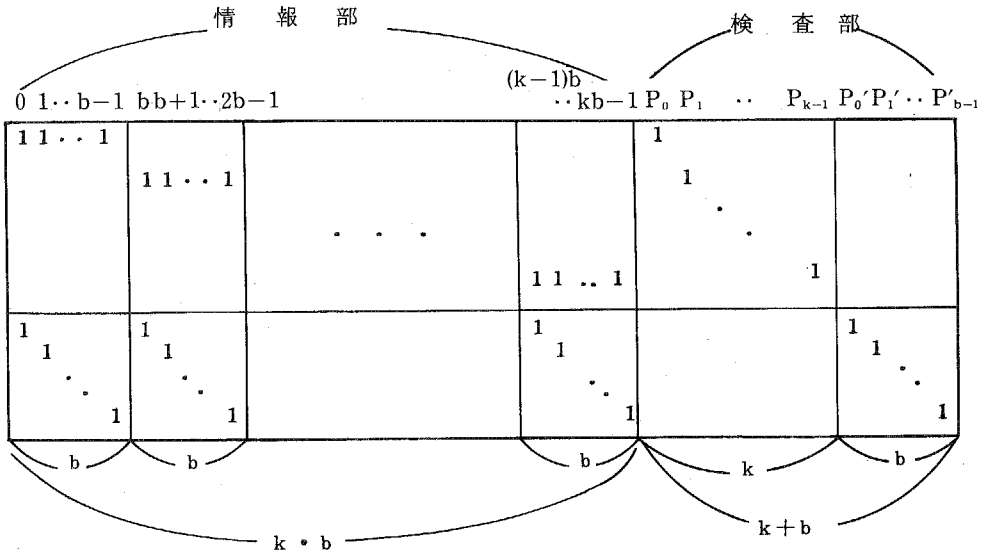


図８－８　水平・垂直パリティ符号

図８－８から，パリティ検査を二種類に分類してパリティＰとパリティP'とする。パリティＰはｋビットからなり，パリティP'はｂビットからなる。このとき，データＸに対する符号語は，ｋ・ｂビットのデータＸにｋビットのパリティ符号P_Xとｂビットのパリティ符号P'_Xを付加して，

$$〔\ X \vdots P_X \vdots P'_X\ 〕$$

の形で表現することができる。入力データＡ，Ｂを次式で表わすとする。

$$\begin{aligned} A &= \{\ (a_0\ a_1 \cdots a_{b-1})(a_b\ a_{b+1} \cdots a_{2b-1}) \cdots (a_{(k-1)b},\ a_{(k-1)b+1},\ \cdots a_{kb-1})\ \} \\ B &= \{\ (b_0\ b_1 \cdots a_{b-1})(b_b\ b_{b+1} \cdots b_{2b-1}) \cdots (b_{(k+1)b},\ b_{(k-1)b+1},\ \cdots b_{kb-1})\ \} \end{aligned} \qquad (8-26)$$

最初のバイトに対するパリティＰは次式で表わすことができる。

$$\begin{aligned} P_{A0} &= a_0 \oplus a_1 \oplus \cdots \oplus a_{b-1} \\ P_{B0} &= b_0 \oplus b_1 \oplus \cdots \oplus b_{b-1} \end{aligned} \qquad (8-27)$$

このようなバイトがｋ個存在することから，Ａ，Ｂ両データに対し，それぞれP_{A_0}～$P_{A_{k-1}}$，P_{B_0}～$P_{B_{k-1}}$で表わされるｋビットの検査ビットが存在する。

一方，パリティP'はパリティＰと直交する関係にある。例えば，P'_0は各バイトの最初のビットのパリティ和で求められる。（８－２６）式で表わされるＡ，Ｂのデータに対して，検査ビットは次式で表わすことができる。

$$\begin{aligned} P'_{A0} &= a_0 \oplus a_b \oplus \cdots \oplus a_{(k-1)b} \\ P'_{B0} &= b_0 \oplus b_b \oplus \cdots \oplus b_{(k-1)b} \end{aligned} \qquad (8-28)$$

同様にして，Ａ，Ｂ両データに対し，それぞれP'_{A0}～$P'_{A_{b-1}}$，P'_{B0}～$P'_{B_{b-1}}$のｂビットの検査ビットが存在する。

〔２〕　パリティ検査（シンドロームの作成）

〔１〕で定義した検査ビットを有する入力〔Ａ┆P_A┆P'_A〕〔Ｂ┆P_B┆P'_B〕を加算回路に

入力したとき，キャリ出力をC，加算出力をYとする。このとき，パリティ検査はパリティPの部分とパリティP'の部分でそれぞれ以下のようになる。

まず，パリティPに対しては，（8－10）式で定義したフルサムパリティ検査による方法にて，各bビットバイト内の誤り検査を行う。例えば，iバイト目の加算出力をy_{ib}，y_{ib+1}，…，$y_{(i+1)b-1}$，キャリ出力をc_{ib-1}，c_{ib}，…，$c_{(i+1)b-2}$，検査ビットをP_{Ai}，P_{Bi}とすると，このバイトに対するフルサムパリティ検査は次式で表わされる。

$$\left.\begin{aligned} S_i &= \left(\sum_{l=0}^{b-1}{}^{\oplus}\ y_{ib+l}\right) \oplus \left(\sum_{l=-1}^{b-2}{}^{\oplus}\ c_{ib+l}\right) \oplus P_{Ai} \oplus P_{Bi} \\ & \quad i = 0,\ 1,\ \cdots,\ k-1 \\ S_i &= 1 \quad ;\text{誤り検出} \\ S_i &= 0 \quad ;\text{誤りなし} \end{aligned}\right\} \qquad (8-29)$$

次に，パリティP'に対する検査は次のような考え方を採用する。加算結果y_jに対して，1ビット前のキャリビットc_{j-1}からハーフサムh_jを作成する。

$$\left.\begin{aligned} h_j &= y_j \oplus c_{j-1} \quad , \quad j = 0,\ 1,\ \cdots,\ kb-1 \\ c_{-1} &= c_{in} \end{aligned}\right\} \qquad (8-30)$$

ハーフサムに対しては，（8－7）式で示した関係を用いて検査ができる。

$$\left.\begin{aligned} S_j &= \left(\sum_{l=0}^{k-1}{}^{\oplus}\ h_{lb+j}\right) \oplus P'_{A_j} \oplus P'_{B_j} \\ &= \left\{\sum_{l=0}^{k-1}{}^{\oplus}\ \left(y_{lb+j} \oplus c_{lb+j-1}\right)\right\} \oplus P'_{A_j} \oplus P'_{B_j} \\ & \quad j = 0,\ 1,\ \cdots,\ b-1 \\ S_j &= 1 \quad ;\text{誤り検出} \\ S_j &= 0 \quad ;\text{誤りなし} \end{aligned}\right\} \qquad (8-31)$$

ここで，$S_j = 1$であれば，あるバイト中のjビット目に誤りが存在することを示す。（8－31）式は，本質的にはフルサムパリティ検査と変わらないが，加算結果とキャリからビット単位に

ハーフサムを作成する点に特徴を有する。

（8－29）式，（8－31）式に示す検査において，その誤り検出能力を高めるために次の手段を採用する。

1）　加算回路はCDS加算回路を採用

2）　キャリ生成回路を別途用意

1）はフルサムパリティ検査において，キャリ誤り検出能力を高めるためである。2）は別に設けたキャリ生成回路の出力を（8－31）式のハーフサムパリティ検査において使用する。すなわち，被検査用の加算回路中のキャリを（8－31）式に使用すると，このキャリ誤りは次の加算結果へ伝搬し，正しく誤りを指摘できない。そのため，別に設けたキャリ生成回路の出力を（8－31）式中のキャリとして使用することにより，加算結果の誤りを正しく検出できる。

〔3〕　シンドロームデコード

本手法に基づく誤り検出・訂正のための回路を図8－9に示す。（8－29）式のS_i，（8－31）式のS_jを用いて，以下に示す誤りの判定を行う。

1）　すべての　$S_i = 0$，すべての$S_j = 0$　；誤りなし

2）　1個の　$S_i = 1$，複数個の$S_j = 1$　；iバイト目のjビット（複数ビット）の誤り（訂正可）

3）　すべての　$S_i = 0$，いずれか$S_j = 1$　；キャリ生成回路，または，ハーフサム生成回路，またはハーフサムパリティ検査回路中の誤り（訂正不可）

4）　いずれか　$S_i = 1$，すべての$S_j = 0$　；フルサムパリティ検査回路中の誤り（訂正不可）

5）　2個以上の$S_i = 1$，複数個の$S_j = 1$　；2バイト以上にまたがる誤り（訂正不可）

訂正可能な場合は，上記2）の場合であり，このとき訂正は反転回路にてデータを反転することにより行う。

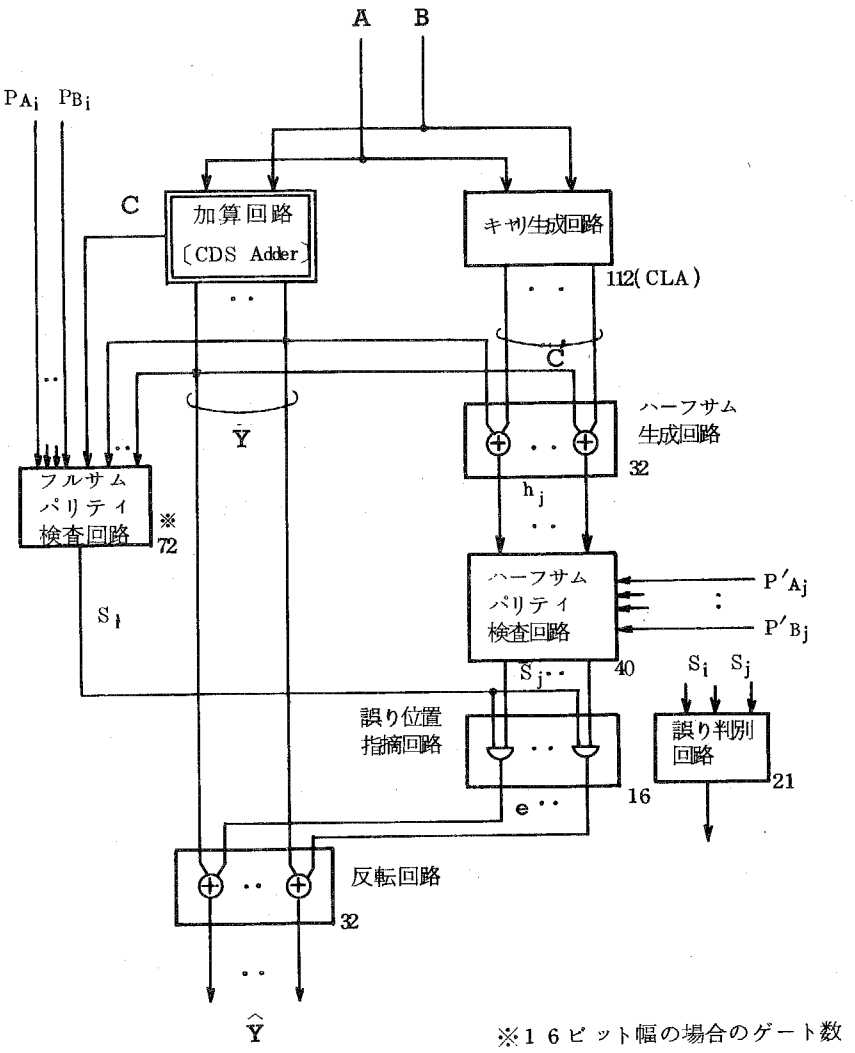


図８－９　水平・垂直パリティ符号による誤り検出・訂正回路

〔４〕　訂正能力

CDS加算回路を使用したときの本手法の誤り訂正能力は次のようになる。

（ｉ）　加算結果のバイト内奇数ビット誤り訂正

（ii）　キャリ誤りによるバイト内すべての誤り訂正

(ii)に関連してキャリ誤りがバイト間にまたがる場合には，２バイト間誤りとなって訂正は不可能である。特に，バイト内の最上位キャリビットの誤りは，加算回路の構成によっては誤訂正を与えることがあり，この誤りは訂正不可能誤りとして検出する必要がある。グループル

ックアヘッド型の加算回路では，グループキャリはバイト内の最上位キャリビットを使用せず，独立に並行して作成されたグループキャリを使用する。そのため，この加算回路では，バイト内で発生したキャリ誤りは次バイトへ誤り波及することはない。但し，このグループキャリを作成する回路の故障によるグループキャリの誤りは，多バイト間に誤りを伝搬させる可能性を有することから，この誤りは訂正不可能誤りとして検出する必要がある。すなわち，作成したグループキャリ C_{G_i} に対して，バイト内最上位のキャリビット C_{B_i} との比較検査により検出できる。(61)

$$E_{C_i} = C_{G_i} \oplus C_{B_i}, \quad i = 0, 1, \cdots, k-1 \qquad (8-32)$$

図８－９に示す回路構成においては，誤り位置指摘回路と反転回路中の故障が誤訂正をもたらす。すなわち，｛フルサムパリティ検査回路｝，｛キャリ生成回路，ハーフサム生成回路，ハーフサムパリティ検査回路｝の２グループに分けると，いずれかのグループ中の単一故障は，前述〔３〕の３）または４）に相当する訂正不可能誤りと判別でき，誤訂正は生じない。誤り位置指摘回路，反転回路のようなハードコアとなる回路ゲート量の全体に対する割合は１５％である。従って，本手法にもとずく誤り訂正回路において，８５％の回路中の単一故障は誤訂正を与えず検出できることになる。

８－４－２　具体的な誤り検出・訂正回路

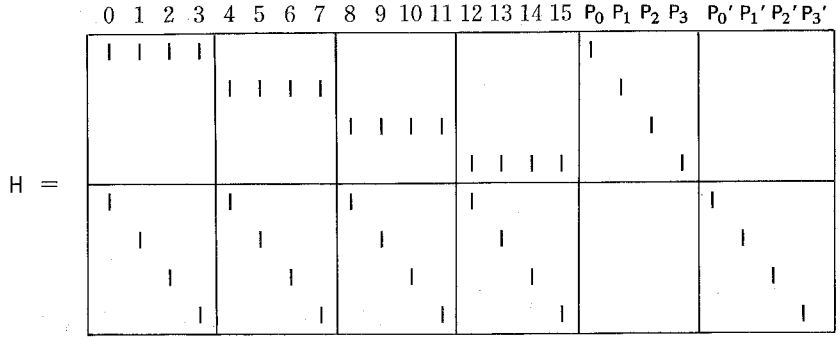
本節は具体的な回路例を示し，ゲート量，訂正のための速度を明らかにする。また，回路がバイト単位にくり返し性を有することを示す。

〔１〕　１６ビットデータに対する誤り検出・訂正回路例

具体例として，バイト長 $b = 4$ ビット，$k = 4$ とする $k \cdot b = 16$ ビットの情報に対する構成を示す。Hマトリクスを図８－１０（ａ）に示す。入力データA，Bは次のように示される。

$$\begin{aligned} A &= \{ (a_0\ a_1\ a_2\ a_3)\ (a_4\ a_5\ a_6\ a_7)\ (a_8\ a_9\ a_{10}\ a_{11})\ (a_{12}\ a_{13}\ a_{14}\ a_{15}) \} \\ B &= \{ (b_0\ b_1\ b_2\ b_3)\ (b_4\ b_5\ b_6\ b_7)\ (b_8\ b_9\ b_{10}\ b_{11})\ (b_{12}\ b_{13}\ b_{14}\ b_{15}) \} \end{aligned} \qquad (8-33)$$

このとき，検査ビットはパリティPに相当するP_0～P_3の4ビットとパリティP'に相当するP_0'～P_3'の4ビットの計8ビットである。P_0～P_3とP_0'～P_3'が互いに直交する関係にあることを図8－10（b）に示す。



(a)

0～15；データ情報

0	1	2	3	P_0
4	5	6	7	P_1
8	9	10	11	P_2
12	13	14	15	P_3
P_0'	P_1'	P_2'	P_3'	

P_0～P_3, P_0'～P_3' } 検査情報

(b)

図8－10　b＝4，k＝4に対する水平・垂直パリティ符号

$$
\left.
\begin{aligned}
P_{A_0} &= a_0 \oplus a_1 \oplus a_2 \oplus a_3 & P_{B_0} &= b_0 \oplus b_1 \oplus b_2 \oplus b_3 \\
P_{A_1} &= a_4 \oplus a_5 \oplus a_6 \oplus a_7 & P_{B_1} &= b_4 \oplus b_5 \oplus b_6 \oplus b_7 \\
P_{A_2} &= a_8 \oplus a_9 \oplus a_{10} \oplus a_{11} & P_{B_2} &= b_8 \oplus b_9 \oplus b_{10} \oplus b_{11} \\
P_{A_3} &= a_{12} \oplus a_{13} \oplus a_{14} \oplus a_{15} & P_{B_3} &= b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15} \\
P'_{A_0} &= a_0 \oplus a_4 \oplus a_8 \oplus a_{12} & P'_{B_0} &= b_0 \oplus b_4 \oplus b_8 \oplus b_{12} \\
P'_{A_1} &= a_1 \oplus a_5 \oplus a_9 \oplus a_{13} & P'_{B_1} &= b_1 \oplus b_5 \oplus b_9 \oplus b_{13} \\
P'_{A_2} &= a_2 \oplus a_6 \oplus a_{10} \oplus a_{14} & P'_{B_2} &= b_2 \oplus b_6 \oplus b_{10} \oplus b_{14} \\
P'_{A_3} &= a_3 \oplus a_7 \oplus a_{11} \oplus a_{15} & P'_{B_3} &= b_3 \oplus b_7 \oplus b_{11} \oplus b_{15}
\end{aligned}
\right\} \qquad (8-34)
$$

まず，フルサムパリティ検査回路は次式により構成できる。但し，出力は，(8－33)式と同様に次のように表わす。

$$
Y = \{ (\ y_0\ y_1\ y_2\ y_3\ )\ (\ y_4\ y_5\ y_6\ y_7\ )\ (\ y_8\ y_9\ y_{10} y_{11}\ )\ (\ y_{12} y_{13} y_{14} y_{15}\ ) \}
$$

$$
\left.
\begin{aligned}
S_{i_0} &= (\ y_0 \oplus y_1 \oplus y_2 \oplus y_3\ ) \oplus (\ c_{in} \oplus c_0 \oplus c_1 \oplus c_2\ ) \oplus P_{A_0} \oplus P_{B_0} \\
S_{i_1} &= (\ y_4 \oplus y_5 \oplus y_6 \oplus y_7\ ) \oplus (\ c_3 \oplus c_4 \oplus c_5 \oplus c_6\ ) \oplus P_{A_1} \oplus P_{B_1} \\
S_{i_2} &= (\ y_8 \oplus y_9 \oplus y_{10} \oplus y_{11}\ ) \oplus (\ c_7 \oplus c_8 \oplus c_9 \oplus c_{10}\ ) \oplus P_{A_2} \oplus P_{B_2} \\
S_{i_3} &= (\ y_{12} \oplus y_{13} \oplus y_{14} \oplus y_{15}\ ) \oplus (\ c_{11} \oplus c_{12} \oplus c_{13} \oplus c_{14}\ ) \oplus P_{A_3} \oplus P_{B_3}
\end{aligned}
\right\}
$$

(8－35)

ハーフサム生成回路は，加算結果とキャリ生成回路とのビット対応の排他的論理和で構成できる。

ハーフサム検査回路は，パリティP'にもとずくパリティ符号から次の関係式を満足する回路にて構成できる。別に設けたキャリ生成回路からのキャリを(c'_{in} c'_0 c'_1 … c'_{14})とする。

$$
\left.
\begin{aligned}
S_{j_0} &= (\ y_0 \oplus c'_{in}\ ) \oplus (\ y_4 \oplus c'_3\ ) \oplus (\ y_8 \oplus c'_7\ ) \oplus (\ y_{12} \oplus c'_{11}\ ) \oplus P'_{A_0} \oplus P'_{B_0} \\
S_{j_1} &= (\ y_1 \oplus c'_0\ ) \oplus (\ y_5 \oplus c'_4\ ) \oplus (\ y_9 \oplus c'_8\ ) \oplus (\ y_{13} \oplus c'_{12}\ ) \oplus P'_{A_1} \oplus P'_{B_1} \\
S_{j_2} &= (\ y_2 \oplus c'_1\ ) \oplus (\ y_6 \oplus c'_5\ ) \oplus (\ y_{10} \oplus c'_9\ ) \oplus (\ y_{14} \oplus c'_{13}\ ) \oplus P'_{A_2} \oplus P'_{B_2} \\
S_{j_3} &= (\ y_3 \oplus c'_2\ ) \oplus (\ y_7 \oplus c'_6\ ) \oplus (\ y_{11} \oplus c'_{10}\ ) \oplus (\ y_{15} \oplus c'_{14}\ ) \oplus P'_{A_3} \oplus P'_{B_3}
\end{aligned}
\right\}
$$

(8－36)

誤り位置指摘回路はS_{i_0}～S_{i_3}とS_{j_0}～S_{j_3}の論理積から作成することができ，2入力ANDゲートにより構成される。

反転回路は誤り位置指摘回路からのポインタで加算結果を反転させる回路であり，16個の2入力排他的論理和回路から構成される。

誤り判別回路は，誤りなし，誤り検出，訂正不可能誤り検出，等を判別する回路である。

以上の各回路のゲート量を表8－4に示す。これから，325ゲートで全体の訂正回路を構成することができ，もとのCDS加算回路が250ゲートであることから，1.3倍のゲート量となる。また，訂正のための速度遅延は8ゲート段数であり，主記憶用の誤り訂正の場合（8～14ゲート段数）と同程度に高速である。

表8－4　訂正回路のゲート量（16ビット幅）

回　　路	ゲート量	記　　事
フルサムパリティ検査回路	72ゲート	
キャリ生成回路	112	キャリルックアヘッド型
ハーフサム生成回路	32	
ハーフサムパリティ検査回路	40	
誤り位置指摘回路	16	
反　転　回　路	32	
誤り判別回路	21	
計	325ゲート	
訂正のための回路遅延	8ゲート段	加算回路の出力から 訂正出力が得られるまで

〔2〕　くり返し回路構成

水平・垂直パリティ符号の形から，この訂正回路がバイト単位のくり返し構成となることを示す。〔1〕に示した16ビット幅の場合では，b＝4ビット単位の4個の同一回路で構成できる。〔1〕での説明から，フルサムパリティ検査回路，ハーフサム生成回路，誤り位置指摘回路，反転回路，ハーフサムパリティ検査回路がbビット単位に区切れることは容易にわかる。

グループルックアヘッド型のキャリ生成回路がバイトスライスできることは，特にグループキャリを生成する回路がバイト単位に同一回路で構成できることを示せばよい。バイト単位の

生成関数G_i，伝達関数T_iは，バイト内のビット単位の生成関数g_j，伝達関数t_jより作成し，これからグループキャリC_a，C_b，C_cが構成される。図8－11にグループキャリ，生成関数，伝達関数の関係を示す。

$$\left.\begin{aligned} C_a &= G_0 + T_0 \cdot c_{in} \\ C_b &= G_1 + T_1 \cdot C_a = G_1 + T_1 G_0 + T_1 T_0 c_{in} \\ C_c &= G_2 + T_2 \cdot C_b = G_2 + T_2 G_1 + T_2 T_1 G_0 + T_2 T_1 T_0 c_{in} \end{aligned}\right\} \quad (8-37)$$

このグループキャリを各バイト間で同一回路で構成するため，（8－37）式中の最大構成を与える回路（この場合，C_cを生成する回路）で共通化する。この回路とC_a，C_b，C_cを生成するための入力条件を図8－12に示す。4ビット単位にバイトスライスしたグループルックアヘッドキャリ生成回路の構成を図8－13に示す。この回路では，グループキャリを生成する回路以外にG_i，T_iを生成する回路，最上位キャリビットc'_{i_3}とグループキャリC_{out}との比較（比較結果をE_{c_i}）を設けている。これは，前述の（8－32）式に相当する検査である。

以上から，誤り検出・訂正回路の1バイト分の回路を図8－14に示す。b＝4ビットの場合で78ゲート，入出力数は48本となる。k＝4として，bを変化させた場合のバイト当りの回路規模を次に示す。

バイト長（ビット）	全情報ビット数	ゲート数	入出力数	ゲート遅延（ゲート段）
3	12	61	39	8
4	16	78	48	8
5	20	156	84	10

図8－14に示す回路を4個用いた16ビット幅の誤り検出・訂正回路の構成を図8－15に示す。

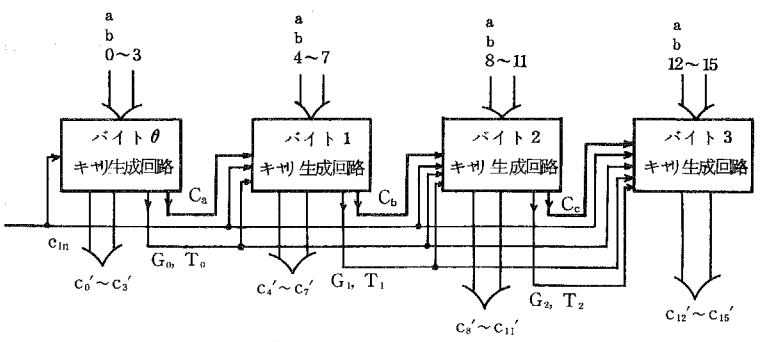


図8－11　グループキャリCa，Cb，Cc

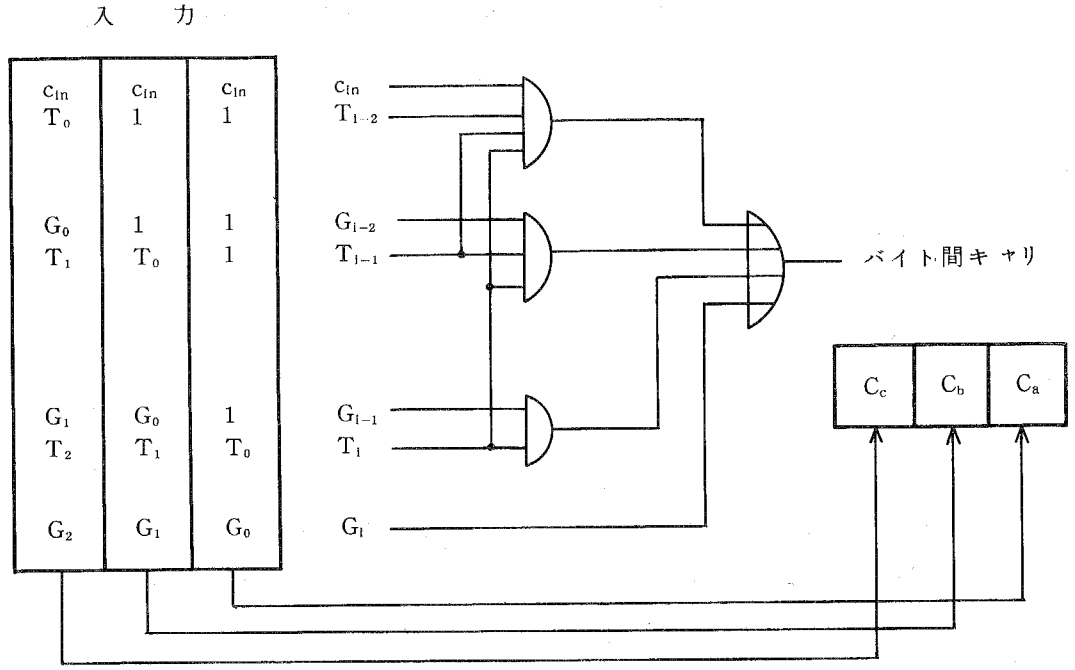


図８－12　バイト間キャリ生成回路

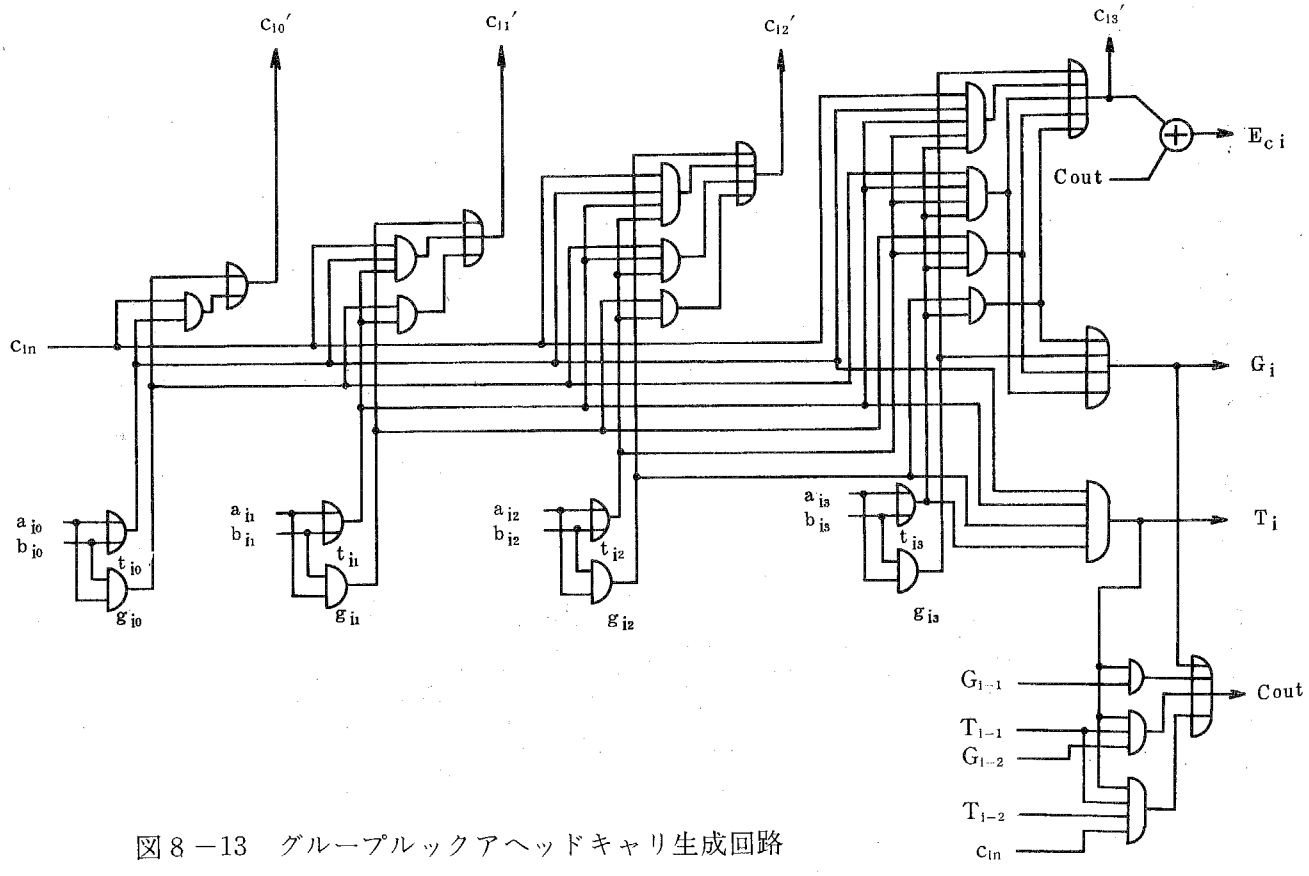


図８－13　グループルックアヘッドキャリ生成回路

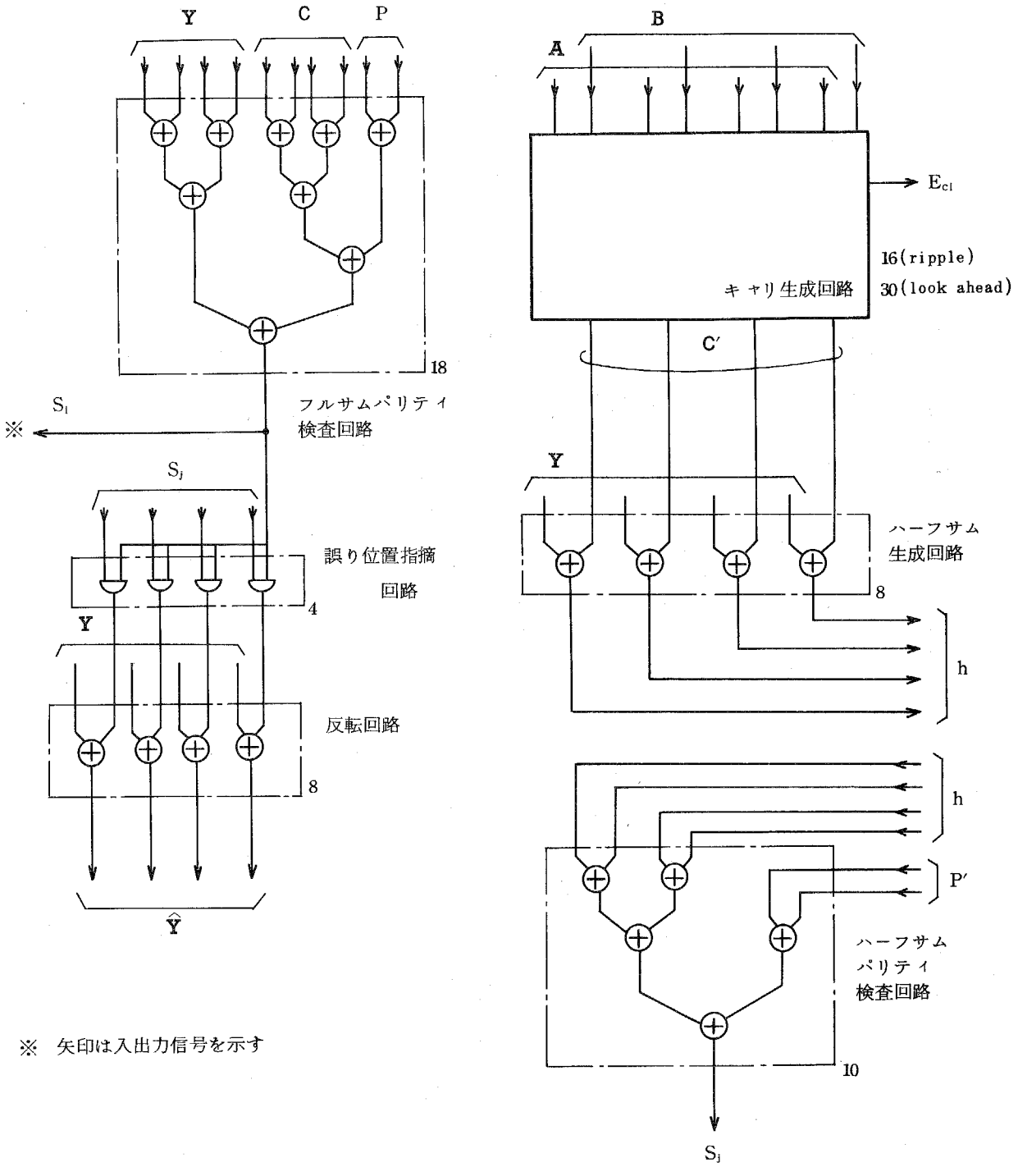


図8－14　1バイト（4ビット）分の誤り検出・訂正回路

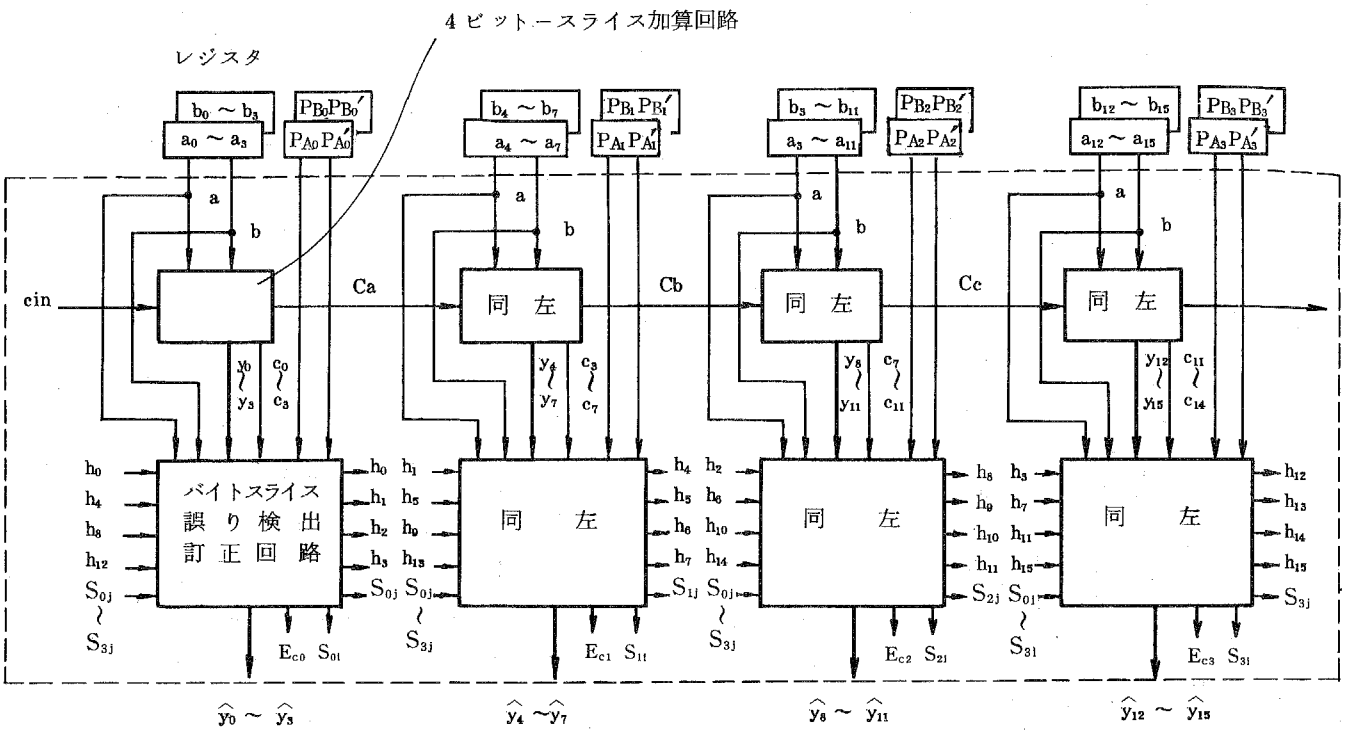


図8－15　加算回路，レジスタを含む誤り検出・訂正回路の
バイトスライス化

8－4－3　他の手法との比較

加算演算を対象に，少くとも1ビットの誤り訂正が可能な各種の手法について，水平・垂直パリティ符号を使用した場合と比較する。結果を表8－5に示す。これから，本論文で提案の水平・垂直パリティ符号による方式，および二重化とパリティ符号を使用した方式では，経済性の点ですぐれている。3重化（TMR）による方法は，訂正速度，検査ビット数，訂正能力の点で他の方式に比較してすぐれている。一方，剰余検査に基づくBP符号による方式は，経済性，訂正速度の点で実用的に問題点があることがわかる。この方式では，回路の簡明さを目的にROMを使用した構成としている。

表8－5　16ビット加算回路に対する誤り訂正手法の比較

項目＼方式	水平垂直パリティ符号方式	二重化＋パリティ符号方式	3重化（TMR）方式	組合せ符号による方式	AN符号（BP符号）による方式
ゲート増加率*（加算回路に対する訂正回路の比率(%)）	130	130	230	170**	480***
訂正速度（ゲート段数）	8	7	3	35†	52††
検査ビット数	8	1	0	8	6
訂正能力	1バイト中のすべてのキャリ誤りとバイト内加算結果の奇数ビット誤り訂正	すべてのキャリ誤りと加算結果の奇数ビット誤り訂正	すべての誤り訂正	1ビット誤り訂正	1ビット誤り訂正
訂正回路中の故障に対する防御	約85%の回路中の故障は誤訂正を与えない	75～80%の回路中の故障は誤訂正を与えない			
訂正回路の構成の容易度	比較的容易	容易	非常に容易	比較的困難	困難
その他の特徴	・訂正回路のバイトスライス化が可能 ・CDS加算回路使用	CDS加算回路使用		訂正回路にROM使用	訂正回路にROM使用

*　16ビット加算回路を250ゲートとする。

**　1k×8ビット ROM 1個＋360ゲート ≈ 420ゲート　（1k×8ビットROM≈60ゲート）（価格換算）

***　4k×8ビット ROM 5個＋200ゲート ≈ 1200ゲート　（4k×8ビットROM≈200ゲート）（価格換算）

†　25ゲート段＋1ROM読出し速度≈35ゲート段　（1ROM読出し速度≈10ゲート段）

††　12ゲート段＋4ROM読出し速度≈52ゲート段　（〃）

8−5　パリティを基本とする符号による誤り検出・訂正

剰余を基本とする符号は，演算回路に対する誤り検出・訂正に適するが，回路が複雑となりゲート量が大きくなる欠点を有する。一方，パリティを基本とする符号は，第3章から第5章に示した通り，各種の機能を有する符号が存在し，その回路は高速で経済的に構成できる特徴を有する。

そこで，ALUに対してもこのパリティを基本とする符号が一般に適用できれば，主記憶装置とALUで同一の符号が使用できることになる。本節では，このような観点に立って，パリティを基本とする符号が，一般の算術論理演算を行う回路にも適用できるための条件を明確にし，具体的な構成方法を示す。

8−5−1　排他的論理和演算に対するパリティ検査

（定義8−1）

2入力データA，Bに対して演算Φを実行して出力Yが得られるとき，すなわち$Y=\Phi(A, B)$が成立するとき，それぞれの入出力データに付加されている検査データC_A，C_B，C_Y の間においても

$$C_Y = \Phi(C_A, C_B)$$

の関係が成立するとき，入出力符号語（A⋮C_A），（B⋮C_B），（Y⋮C_Y）は演算Φに対して"保存される"（preserved）[8] と言う。 □

（定理8−1）

Φ＝EX−ORのとき，パリティ符号は保存される。 □

（証明）

kビットからなる2個のデータA，Bに対するパリティビットP_A，P_B において，（8−1）式，（8−2）式の関係が成立しているとする。このとき，出力Yは次のようになる。

$$y_i = a_i \oplus b_i, \quad i=0, 1, \cdots, k-1 \qquad (8-39)$$

（８－３９）式をｉ＝０からｉ＝ｋ－１まで加算（mod．２）すると，

$$\sum_{i=0}^{k-1}{}^{\oplus} y_i = \sum_{i=0}^{k-1}{}^{\oplus} (a_i \oplus b_i)$$

$$= \left(\sum_{i=0}^{k-1}{}^{\oplus} a_i\right) \oplus \left(\sum_{i=0}^{k-1}{}^{\oplus} b_i\right) \qquad (8-40)$$

が成立する。（８－２）式，（８－４）式から（８－４０）式は次式に等しい。

$$P_Y = P_A \oplus P_B \qquad (8-41)$$

これから，ϕ＝EX－OR演算においては，パリティビット間においてもϕ＝EX－OR 演算の関係が保たれる。

（証明終り）

以上の関係を図８－１６に示す。各種の演算中，パリティ符号を使用してこのような保存の関係が成立するのは，ϕ＝EX－ORまたはEX－NOR演算のみである。

この演算結果Y中に１ビット誤りが存在する場合の検出論理は（８－１２）式により表わされる。この関係を図８－１７に示す。図中，破線で囲んだ回路が検査回路である。

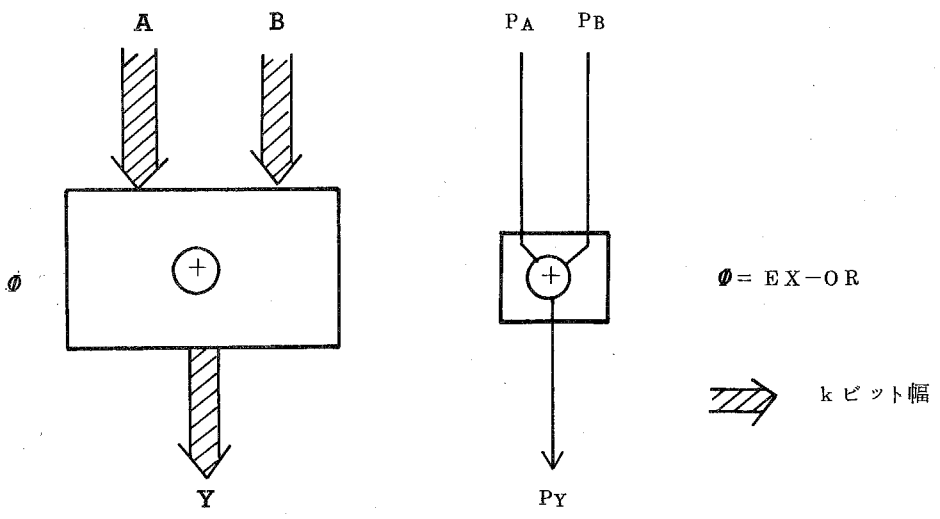


図８－16　排他的論理和演算における出力パリティビットP_Yの生成

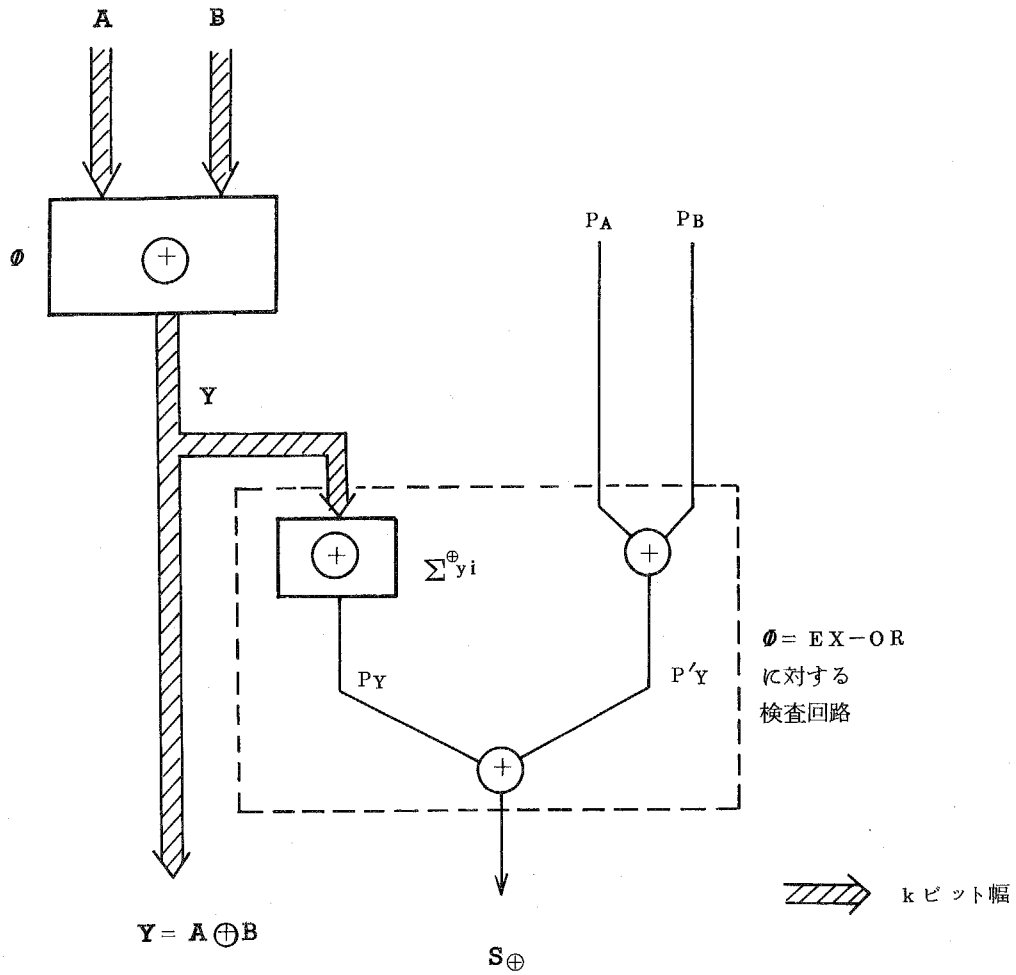


図８－17　掛他的論理和演算におけるパリティ検査

８－５－２　任意の演算に対するパリティ検査[30]

排他的論理和演算に対する検査論理から，任意の演算Φに対するパリティ検査として，次の考え方が推察できる。すなわち，演算Φによる結果を線形に排他的論理和演算結果に変換できれば，その後は排他的論理和演算に対する検査がそのまま適用できるはずである。そこで，演算Φから排他的論理和演算結果への変換関数をF_Φとする。

$$F_\Phi = (f_{\Phi_0}, f_{\Phi_1}, \cdots, f_{\Phi_{k-1}}) \qquad (8-42)$$

すなわち，f_{Φ_i}は$y_i = \Phi (a_i, b_i)$を$a_i \oplus b_i$へ変換する関数である。

$$f_{\phi_i}(y_i) = a_i \oplus b_i, \quad i=0, 1, \cdots, k-1 \qquad (8-43)$$

このとき，次の定理が成立する。

（定理８－２）

y_i から $a_i \oplus b_i$ への変換関数 $f_{\phi}(y_i)$，$i=0, 1, \cdots, k-1$，について，

$$\frac{df_{\phi}(y_i)}{dy_i} = 1 \qquad (8-44)$$

が成立すれば，演算結果Ｙに対するパリティビット P_Y を予測（predict）することができる。しかも，演算結果中の単一誤りは常に変換後の結果へ伝搬する。 □

この定理を証明する前に，次の補題を証明する必要がある。

（補題）

$$\frac{dF}{dY} = 1 \qquad (8-45)$$

が成立するのは，$F=Y \oplus R$ または $\overline{Y \oplus R}$ のときのみである。ここで，Ｒは'0'，'1'の双方の値をとることのできる任意の関数とする。 □

（補題の証明）

（８－４５）式が成立するＦとして，関数＊が存在したとする。

$$F = Y * R \qquad (8-46)$$

このとき，次式が成立しなければならない。

$$\frac{dF}{dY} = (1 * R) \oplus (0 * R) = 1 \qquad (8-47)$$

（8－47）式が成立するのは，（1＊R）および（0＊R）について，次の場合のみである。

	1 ＊ R	0 ＊ R
ケース1	1	0
ケース2	0	1
ケース3	R	$\overline{R}$
ケース4	$\overline{R}$	$\overline{R}$

各ケースが成立するのは次の場合である。

ケース1 ； $\begin{cases} R = 1 \text{ でかつ} & * = \text{AND} \\ R = 0 \text{ でかつ} & * = \text{OR} \end{cases}$

ケース2 ； $\begin{cases} R = 1 \text{ でかつ} & * = \text{NAND} \\ R = 0 \text{ でかつ} & * = \text{NOR} \end{cases}$

ケース3 ； $* = \text{EX-NOR}$

ケース4 ； $* = \text{EX-OR}$

Rは'0'，'1'の双方の値をとりうる関数であるから，ケース1，ケース2は命題に反する。これから（8－45）式を満足するのはケース3，ケース4の場合のみである。これから，Fは

$$F = Y \oplus R \quad \text{または} \quad \overline{Y \oplus R}$$

でなければならない。

（証明終り）

（定理8－2の証明）

補題から

$$f_{\phi}(y_i) = y_i \oplus r_i \qquad (8-48)$$

とする。（8－43）式とから，r_i は，

$$r_i = y_i \oplus a_i \oplus b_i \tag{8-49}$$

から，常に存在する。（8－49）式の両辺をi＝0からi＝k－1まで加算（mod．2）すると次式が成立する。

$$\begin{aligned} \sum_{i=0}^{k-1}{}^{\oplus} y_i &= \sum_{i=0}^{k-1}{}^{\oplus} (r_i \oplus a_i \oplus b_i) \\ &= (\sum_{i=0}^{k-1}{}^{\oplus} r_i) \oplus (\sum_{i=0}^{k-1}{}^{\oplus} a_i) \oplus (\sum_{i=0}^{k-1}{}^{\oplus} b_i) \end{aligned} \tag{8-50}$$

（8－50）式の左辺はP_Yに等しく，$\sum_{i=0}^{k-1}{}^{\oplus} a_i = P_A$，$\sum_{i=0}^{k-1}{}^{\oplus} b_i = P_B$より，

$$P_Y = (\sum_{i=0}^{k-1}{}^{\oplus} r_i) \oplus P_A \oplus P_B \tag{8-51}$$

が得られる。よって，Yに対するパリティビットP_Yは（8－51）式から予測（predict）することができる。

一方，$f_{\phi}(y_i) = \overline{y_i \oplus r_i}$の場合については，全く同様にして次のように$P_Y$を予測できる。

$$P_Y = \begin{cases} (\sum_{i=0}^{k-1}{}^{\oplus} r_i) \oplus P_A \oplus P_B & \cdots\cdots k ; \text{even} \\ (\sum_{i=0}^{k-1}{}^{\oplus} r_i) \oplus \overline{P_A \oplus P_B} & \cdots\cdots k ; \text{odd} \end{cases} \tag{8-52}$$

また，（8－44）式に示すブール微分式は，その定義からy_iの誤りは$f_{\phi}(y_i)$へ伝搬することを意味する。

（証明終り）

r_iの集合を次のように定義する。

$$R = (r_0, \ r_1, \ \cdots, \ r_{k-1}) \tag{8-53}$$

各種演算に対する r_i を表８－６に示す。定理８－２においては，以後簡単の為，$f_{\Phi}(y_i)=y_i \oplus r_i$ を採用する。以上から，任意の演算Φに対するパリティ検査は次の定理に示す論理により実行することができる。

表８－６　演算Φに対する r_i

演　算　Φ	演　算　結　果 $y_i=\Phi(a_i, b_i)$	r_i	
		＊＝Exclusive-Or	＊＝Exclusive-Nor
AND	$a_i \cap b_i$	$a_i \cup b_i$	$\overline{a_i} \cap \overline{b_i}$
OR	$a_i \cup b_i$	$a_i \cap b_i$	$\overline{a_i} \cup \overline{b_i}$
EX-OR	$a_i \oplus b_i$	0	1
EX-NOR	$\overline{a_i \oplus b_i}$	1	0
ADD	$a_i + b_i$	c_{i-1}*	$\overline{c_{i-1}}$

＊ $(i-1)$th bit carry

（定理８－３）

任意の演算Φに対するパリティ検査は次式で表わすことができる。

$$
\left.
\begin{aligned}
S_{\Phi} &= \left(\sum_{i=0}^{k-1}{}^{\oplus} y_i\right) \oplus \left\{\left(\sum_{i=0}^{k-1}{}^{\oplus} r_i\right) \oplus P_A \oplus P_B\right\} \\
&= \left\{\sum_{i=0}^{k-1}{}^{\oplus} (y_i \oplus r_i)\right\} \oplus P_A \oplus P_B \\
&= \left\{\sum_{i=0}^{k-1}{}^{\oplus} f_{\Phi}(y_i)\right\} \oplus P_A \oplus P_B \\
&\quad S_{\Phi} = 1 \quad ; \text{誤り検出} \\
&\quad S_{\Phi} = 0 \quad ; \text{誤りなし}
\end{aligned}
\right\} \qquad (8-54)
$$

□

証明は容易な為，省略する。

図８－１８に（８－５４）式に基づく演算Φに対するパリティ検査回路を示す。これから，一点鎖線内がこの検査回路に相当する。破線は図８－17に示すEX－OR演算に対するパリティ検査回路を示すものであり，F_{Φ}の回路からの出力データは等価的にEX－OR演算結果に変換さ

れた値となっていることを示す。また，定理８－２から，演算ϕの結果Yに単一誤りがあれば，それはF_ϕの回路の出力へ必ず伝搬することから，S_ϕにおいて検出することができる。

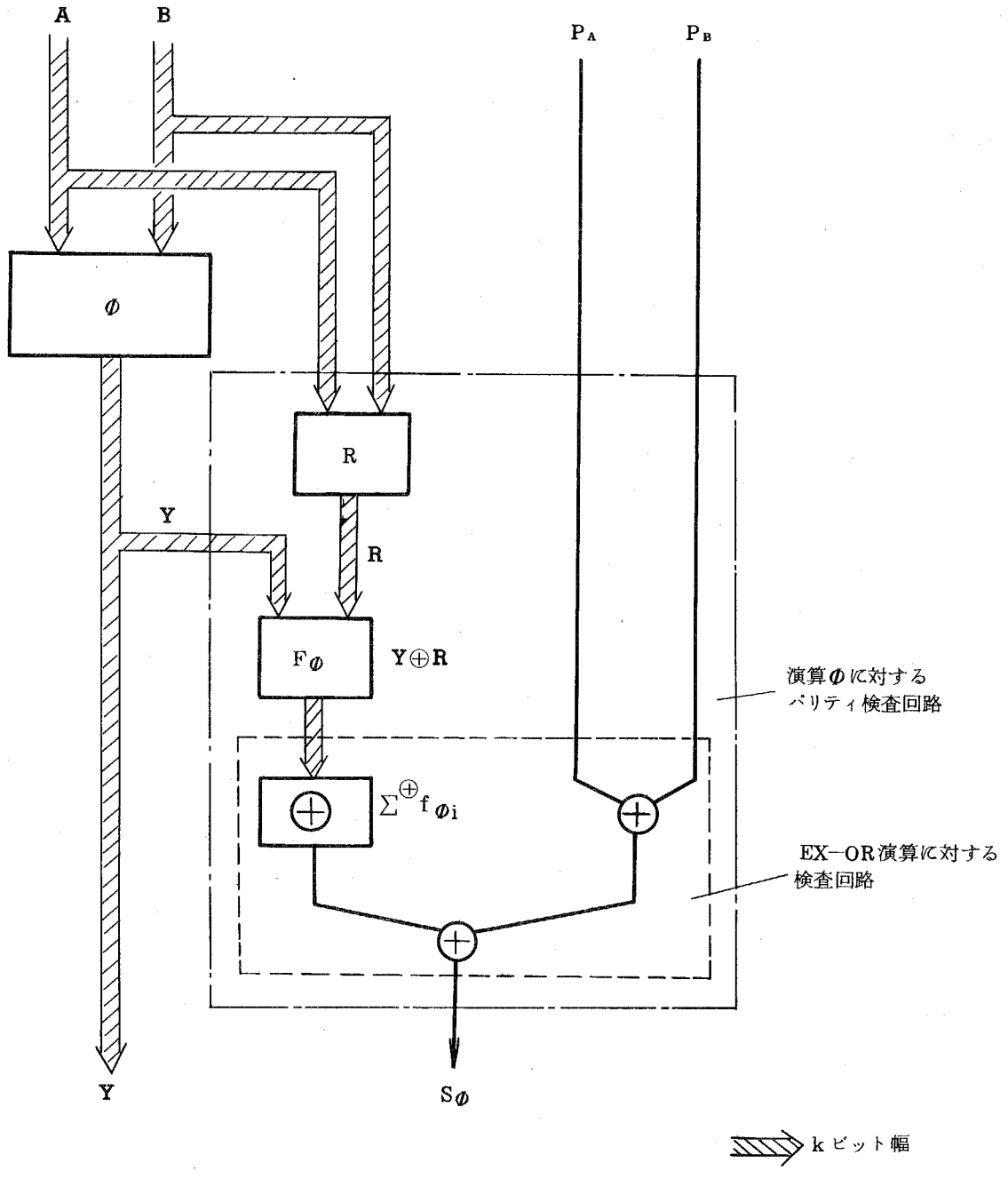


図８－18　任意の演算ϕに対するパリティ検査

8－5－3　パリティを基本とする符号による誤り訂正(30)

パリティを基本とする符号の符号語を〔A ¦ C_A 〕〔B ¦ C_B 〕とする。ここで，A，Bは（8－1）式に示したkビットの情報ビットであり，C_A，C_B は次に示すrビットの検査ビットである。

$$\begin{aligned} C_A &= (c_{A_0}, c_{A_1}, \cdots, c_{A_{r-1}}) \\ C_B &= (c_{B_0}, c_{B_1}, \cdots, c_{B_{r-1}}) \end{aligned} \qquad (8-55)$$

この符号のHマトリクス（r行n列）を次式に示す。

$$H = 〔 H_e \,¦\, I_r 〕_{r \times n} \qquad (8-56)$$

ここで，H_e は符号化Hマトリクス（Encoding H Matrix）であり，I_r はr×rの単位行列である。

また，ALU演算後の出力を〔Y ¦ C_Y 〕とすると，C_Y は出力Yに対するrビットの検査ビットである。

$$C_Y = (c_{Y_0}, c_{Y_1}, \cdots, c_{Y_{r-1}}) \qquad (8-57)$$

前節において，任意の演算に対しパリティ検査が実行できることを示した。（8－56）式で表現される符号は，Hマトリクスの各行で指定されるビットの組r個に対してパリティ検査を実行することを意味する。これから，Hマトリクスを使用して，マトリクスで規定される誤りの検出・訂正が実行できるはずである。

以上から，出力Yに対して誤りパターンE_Y が生じたとき（その出力をY'とする）の訂正手法を以下に示す。

$$\begin{aligned} Y' &= Y \oplus E_Y \\ E_Y &= (e_{y_0}, e_{y_1}, \cdots, e_{y_{k-1}}) \end{aligned} \qquad (8-58)$$

〔1〕　シンドロームの生成

Y'に対する検査ビットC_Y'の生成を行う。

$$C_Y' = Y' \cdot H_e^T = (Y \oplus E_Y) \cdot H_e^T \qquad (8-59)$$

一方，出力Y'を使用しないで検査ビットの予測を行う。これは，（8－51）式から，次のように検査ビットC_pが作成できる。

$$C_p = R \cdot H_e^T \oplus C_A \oplus C_B \qquad (8-60)$$

（8－59）式，（8－60）式を使用して，シンドロームSを作成することができる。

$$\begin{aligned} S &= C_Y' \oplus C_p \\ &= (Y \cdot H_e^T \oplus E_Y \cdot H_e^T) \oplus (R \cdot H_e^T \oplus C_A \oplus C_B) \\ &= E_Y \cdot H_e^T \oplus \{(Y \oplus R) \cdot H_e^T \oplus C_A \oplus C_B\} \end{aligned} \qquad (8-61)$$

ここで，誤りのない出力Yに対しては次式が成立する。

$$(Y \oplus R) \cdot H_e^T \oplus C_A \oplus C_B = O \qquad (8-62)$$

これから，（8－61）式は次式に一致する。

$$S = E_Y \cdot H_e^T \qquad (8-63)$$

一方，入力検査情報C_AまたはC_Bに誤りE_C

$$E_C = (e_{c_0},\ e_{c_1},\ \cdots,\ e_{c_{r-1}}) \qquad (8-64)$$

が存在する場合には，シンドロームは次式で表わせる。

$$S = E_C \cdot I_r \qquad (8-65)$$

〔2〕 シンドロームデコード

（8－63）式，または（8－65）式で求められたシンドロームSから，概略次のようにして誤りを判定する。

$$\left.\begin{array}{ll} W(S)=0 & ;\text{誤りなし} \\ W(S)=1 & ;\text{検査部}C_A\ \text{または}\ C_B\ \text{に誤りが存在} \\ W(S)\geq 2 & ;\text{情報部Yに誤りが存在} \end{array}\right\} \quad (8-66)$$

ここで，W(S)はシンドロームSの重みを表わす。特に，W(S)≧2の場合のE_Yに対しては，マトリクスH_eに基づいてデコードが行われる。これは，符号の機能によって異なり，第3章，第4章で示した各種の符号毎に個有の復号が行われる。すなわち，シンドロームデコードとして，第3章，第4章で示した符号の場合と全く同一の復号手法が適用できる。

〔3〕 誤りの訂正

シンドロームデコードにより求めた誤りパターン（E_Y：E_C）から，訂正出力情報（$\hat{Y}$：$\hat{C}_Y$）を求める。これは，次式から求めることができる。

$$\left\{\begin{array}{l} \hat{Y} = Y' \oplus E_Y \\ \hat{C}_Y = C_p \oplus E_c \end{array}\right. \quad (8-67)$$

ここで注意しなければならないのは，検査部の訂正においてC_Yを訂正するのではなく，C_pに対して訂正を実行する点にある。その理由は次の点にある。

（i） 検査部の誤りは，入力検査情報C_AまたはC_Bに存在する。このとき，C_Yに対して訂正動作を行うと，元来C_Yは正しいことから，誤訂正（ミスコレクト）することになる。

（ii） C_Yに対して訂正を実行すると，出力Yの誤りに対して相当する検査ビットも反転させなければならない。

以上から，情報ビット数kビット，検査ビット数rビットを有するパリティを基本とする符号を用いて，ALUの出力Yまたは検査部に含まれる誤り（E_Y：E_C）を訂正する手法を図8－19に示す。図中，破線で囲む回路は主記憶用の復号化回路である検出・訂正回路(ECC_1)に一致する。図中，ECC_0がALUに対する誤り検出・訂正回路である。これから，ECC_0からECC_1を除いた回路である，Rを生成する回路，C_pを生成する回路を追加するだけで，主記憶用の

回路ECC_1を使用して，ALUに対する誤り検出・訂正が実現できることになる。

具体的な符号に対する回路構成は第9章に示す。

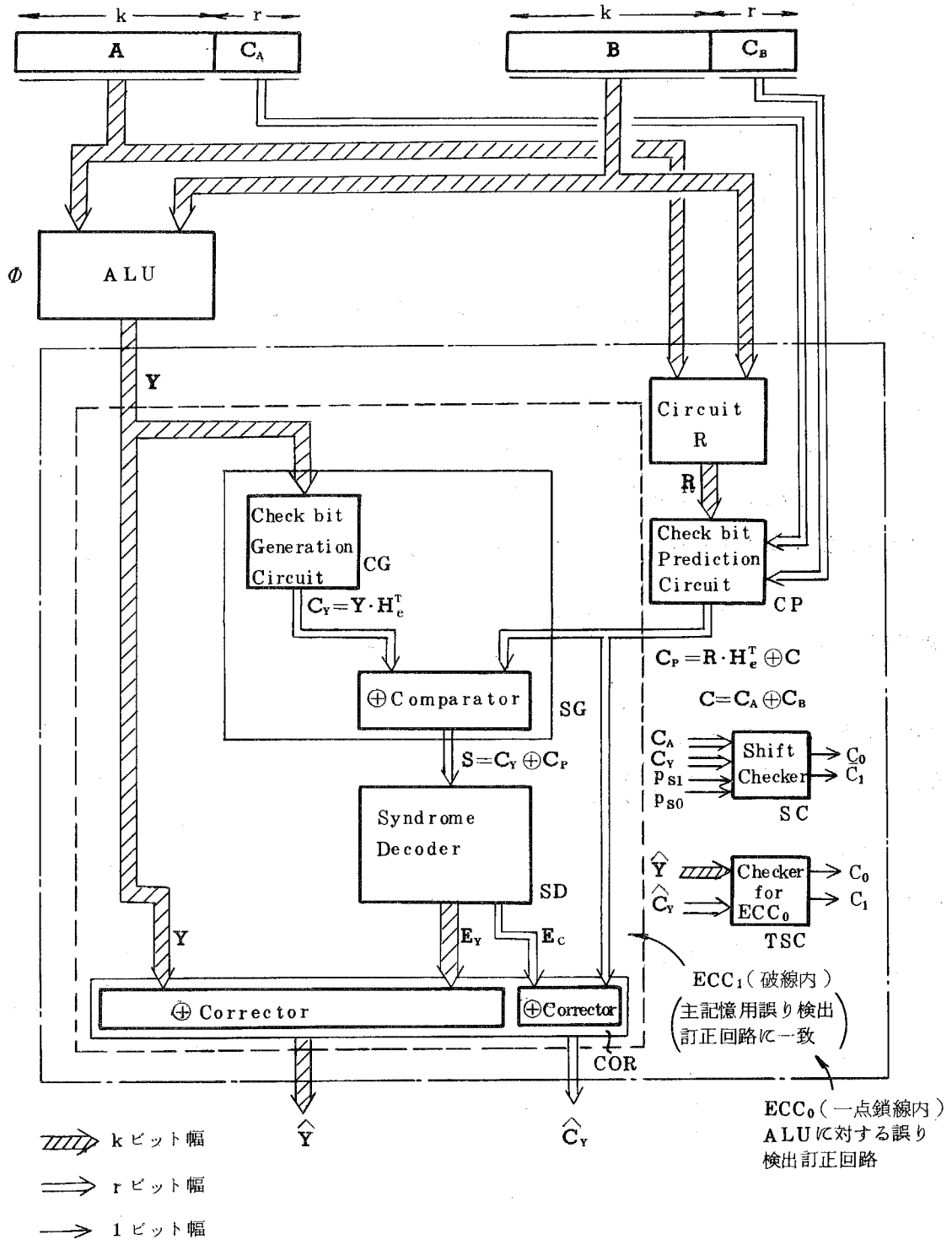


図8－19　ALUに対する誤り検出・訂正回路

8－6　入力誤りに対する誤り検出・訂正

前節までは，誤りは入力情報AおよびBになく，ALU中に存在すると仮定した。ここでは，逆にALU中に誤りは存在せず，入力情報AまたはB中に誤りが存在する場合を考える。本節では，8－5で示した手法を用いて，このような誤りを検出・訂正するための条件と方法を示す。

8－6－1　誤り検出[30][60]

（定理8－4）

図8－18に示すパリティ検査は，入力情報AまたはB中の1ビット誤りを必ず検出する。

□

（証明）

（8－54）式に示すシンドロームS_{ϕ}の作成において，（8－43）式を代入すると次式を得る。

$$S_{\phi} = \left(\sum_{i=0}^{k-1}{}^{\oplus} \; f_{\phi}(y_i) \right) \oplus P_A \oplus P_B$$

$$= \left(\sum_{i=0}^{k-1}{}^{\oplus} \; (a_i \oplus b_i) \right) \oplus P_A \oplus P_B \qquad (8-68)$$

これから，次式が成立する。

$$\begin{cases} \dfrac{dS_{\phi}}{da_i} = 1 \\ \\ \dfrac{dS_{\phi}}{db_i} = 1 \end{cases} \qquad i = 0,\ 1,\ \cdots,\ k-1 \qquad (8-69)$$

よって，AまたはB中の1ビット誤りは常に検出できる。

（証明終り）

この定理は，図8－18に示すパリティ検査が自動的に1ビットの入力誤りを検出できることを意味する。これから，ALUの入力情報に対してALUへ入力する前にパリティ検査を実行する必要はなく，ALUの出力において本検査を実行すれば，入力情報中の誤り，またはALU中の誤りは検出できることになる。

回路を二重化し，その結果を比較して誤りの検出を行う二重化の手法では，このような入力の誤りは全く検出できない。これは，二重化に対して，本検査手法がすぐれていることを意味する。

8－6－2　誤り訂正(30)

ALU出力中の誤り，または入力AまたはB中の誤りが自動的に訂正できれば，訂正回路の受持つ被訂正範囲がALUだけでなく，その入力側の回路も含むことになる。これは，被訂正範囲を拡大することを意味し，等価的に訂正回路の能力を高め，経済的な使用を可能とする。

ALUを三重化してその出力を多数決回路にて誤り訂正する手法においては，入力中の誤りは訂正不可能である。訂正可能とするためには，入力側の回路も三重化しなければならない。

そこで，図8－19に示す訂正手法が，入力情報の誤りを訂正できるための条件を考える。

（定理8－5）

図8－19に示す誤り訂正回路は，

$$\frac{dr_i}{da_i}=\frac{dr_i}{db_i}=0 \quad \text{または} \quad \frac{dy_i}{da_i}=\frac{dy_i}{db_i}=1 \qquad (8-70)$$

$$i=0,\ 1,\ \cdots,\ k-1$$

が成立する場合に，入力誤りを訂正することができる。 □

（証明）

$$f_{\phi}(y_i)=y_i \oplus r_i$$
$$=a_i \oplus b_i$$

より次式が成立する。

$$\frac{df_{\Phi}(y_i)}{da_i} = \frac{df_{\Phi}(y_i)}{db_i} = 1 \qquad (8-71)$$

入力誤りは（8－61）式のシンドローム中Y⊕Rに影響を与える。

$$S = (Y \oplus R) \cdot H_e^T \oplus C_A \oplus C_B$$
$$= F_{\Phi} \cdot H_e^T \oplus C_A \oplus C_B$$

これから，Hマトリクス要素$h_{p,i} = 1$（$p = 0, 1, \cdots, r-1$　$i = 0, 1, \cdots, k-1$）に対して，

$$\frac{dS_p}{df_{\Phi}(y_i)} = 1 \qquad (8-72)$$

となる。ここでS_pはS中のp番目のシンドローム要素である。よって，（8－71）式と（8－72）式から次式が成立する。

$$\frac{dS_p}{da_i} = \frac{dS_p}{db_i} = 1 \qquad (h_{p,i} = 1 \text{に対して}) \qquad (8-73)$$

これから，入力誤りは必ずシンドローム結果に伝搬する。（8－71）式から，入力誤りはYまたはRのいずれかに伝搬し，双方に伝搬することはない。図8－19から入力誤りがYに伝搬し，Rに伝搬しないならば，その誤りは訂正できる。一方，入力誤りがRに伝搬し，Yに伝搬しない場合には，出力Yは正しいはずであるから，CORにて誤訂正（ミスコレクト）することになる。これから，

$$\frac{dr_i}{da_i} = \frac{dr_i}{db_i} = 0 \qquad \text{または}$$

$$\frac{dy_i}{da_i} = \frac{dy_i}{db_i} = 1$$

のとき，入力誤りは訂正できることになる。すなわち，（8－70）式が成立するような場合に入力誤りは訂正できる。

（証明終り）

基本的な演算に対する dy_i / da_i ，dy_i / db_i の関係を表8－7に示す。これから，EX－OR，EX－NOR演算については，入力情報AまたはB中の1ビット誤りは，SEC符号またはSEC－DED符号により訂正できる。

表8－7　演算ϕに対する dy_i/da_i ，dy_i/db_i

ϕ	y_i	$\frac{dy_i}{da_i}$	$\frac{dy_i}{db_i}$
ADD	$a_i \oplus b_i \oplus c_{i-1}$	1	1
EX－OR	$a_i \oplus b_i$	1	1
EX－NOR	$\overline{a_i \oplus b_i}$	1	1
AND	$a_i \cap b_i$	b_i	a_i
OR	$a_i \cup b_i$	$\bar{b}_i$	$\bar{a}_i$

加算演算についても，入力情報中の1ビット誤りはこれらの符号を使用して訂正できる。しかし，これは入力情報中の誤りがキャリビットに伝搬しない場合である。入力情報 a_i または b_i に誤りが存在するとき，

$$\frac{dc_i}{da_i} = b_i \oplus c_{i-1}$$

$$\frac{dc_i}{db_i} = a_i \oplus c_{i-1} \qquad (8-74)$$

から，$dc_i / da_i = 1$（すなわち，$b_i \oplus c_{i-1} = 1$）または，$dc_i / db_i = 1$（すなわち，$a_i \oplus c_{i-1} = 1$）のとき，

$$\frac{dy_{i+1}}{dc_i} = \frac{d}{dc_i}(a_{i+1} \oplus b_{i+1} \oplus c_i) = 1 \qquad (8-75)$$

より，（i＋1）ビット目の加算結果に必ず誤りが伝搬する。また，これは，（i＋2）ビット目以降のキャリビット，および加算結果にも誤り伝搬する可能性を有している。従って，入力情報中の誤りは，キャリビットのバースト誤りとなり，それにともない加算結果のバースト誤りとなる可能性を有する。これから，加算演算に対しては，ビット系の符号よりむしろバイト系符号またはバースト誤り訂正符号を使用する方がよいと言える。

一方，AND，OR演算に対しては，誤り訂正の可能性は a_i または b_i の値自身に依存する。これらの演算に対しては，誤り訂正の為に第9章で示す補正回路等を設置する必要がある。

8－7 結　　言

ALUに対する誤り検出・訂正手法として，従来提案されている手法について整理すると共に，これらに対する問題点を明らかにした。また，加算回路に対する簡明な誤り検出・訂正手法として，二重化と単純パリティ符号を組合わせた手法と，水平・垂直パリティ符号による方法を提案した。さらに，一般にパリティを基本とする符号を使用してALUの誤りを検出・訂正できるための条件と具体的な構成法を示した。主要な結論を以下に示す。

(1) ALUに対する誤り検出・訂正手法として従来提案されている各種の手法を比較した結果，剰余符号による手法は回路ゲート量が非常に大きく（１ビット誤り訂正ＡＮ符号の場合で，もとのALUに対し４８０％のゲート増加率），しかも訂正できるまでの遅延が大きい等の欠点を有していることを具体的に示した。

(2) 二重化と単純パリティ符号を組合わせて，加算回路に対する新しい誤り検出・訂正手法を提案した。本手法による誤り検出・訂正回路は加算回路に対し約１３０％のゲート量増加であり，三重化による方法，ＡＮ符号による方法と比較するとかなり小さい。もとの加算回路をキャリディペンデントサム（ＣＤＳ）加算回路を使用すれば，すべてのキャリ誤りによる誤りと加算結果の奇数ビット誤りを訂正できる。

(3) 水平・垂直パリティ符号による加算回路に対する誤り検出・訂正手法を示した。本手法による誤り検出・訂正回路のゲート量増加は約１３０％で(2)の手法と同程度である。１６ビットの情報ビット数に対して，８ビットの検査ビットを必要とする。ＣＤＳ加算回路を使用し符号長をバイトに区切れば，単一バイト内のすべてのキャリ誤りとバイト内加算結果の奇数ビット誤りを訂正することができる。また，誤り検出・訂正回路としてその内の約８５％に対して，その内の単一故障は誤訂正を与えない構成となる。また，本手法は，誤り検出・訂正回路のバイトスライス化が容易に実現できる特徴を有する。

(4) 一般の主記憶等で用いられているパリティを基本とする符号をALUに適用して，誤り検出・訂正できるための手法を示した。原理は，演算結果を排他的論理和演算結果に線形に変換する点にある。この原理を各種の演算に適用して実現可能であることを示した。

(5) ALU中に故障はなく，入力情報中に誤りが存在する場合，上記(4)に示す原理に基づく手法により誤り検出・訂正できるための条件を求めた。その結果，誤り検出に対しては，入力情報中の１ビット誤りは必ず検出できる。これから，ALUの入力側に位置する回路も誤り検出の範囲とすることができる。一方，誤り訂正できるためには，ｉビット目の演算結果y_iに対する入力情報a_i，b_iについて，$dy_i / da_i = dy_i / db_i = 1$が成立すればよいことを示した。

第9章　高信頼計算機システム

9−1 緒　言

第8章では，パリティを基本とする符号をALUに適用する手法を理論的に明らかにした。本章では，この手法を具体的に計算機システムに適用し，その構成，評価を行う。特に適用に当っては，誤り検出・訂正回路を出来るだけ少ないゲート量で実現すると共に，誤り検出・訂正範囲にある回路の故障率が大きい程，検出・訂正の効果が大きいことを考慮する。この観点に立って，ALUと主記憶間で同一の符号を使用し，単一の誤り検出・訂正回路を設ける高信頼システム構成を提案する。

本章では，各種の符号を適用し，上記観点を満足するシステム構成を明らかにする。また，簡明な訂正手法である三重化（TMR）による手法との比較を行い，本提案に対する評価を行う。

9－2　誤り検出・訂正の効果―システムレベルにおける考察―

ALUに対する高信頼化の手法は，実用的にはパリティ符号が主体であり，一部二重化による方法[97]が採用されている。剰余符号としては，大形機中の乗算回路にその誤り検出として一部採用されているにすぎない。第8章で示したAN符号等の高度な符号が具体的に実用化されない原因は，対象とするALUに対する誤り検出・訂正の効果の点に問題があると考える。ここで，この効果ηを次のように定義する。

$$\eta = \frac{\text{誤り検出・訂正範囲にあるハードウェアの故障率}}{\text{誤り検出・訂正回路のゲート数}} \quad (\text{Fit／ゲート}) \qquad (9-1)$$

すなわち，ηは誤り検出・訂正回路1ゲート当りの誤り検出・訂正できる故障率を表わすものであり，大きい程その効果が大きいと言える。

そこで，主記憶に対して単一誤り訂正の機能を有する符号，およびALUに対して単一誤り訂正の機能を有する符号を適用して，その効果ηを試算し比較してみよう。

主記憶の場合，次に示す仮定の下で試算すると$\eta \approx 94$となる。

- 記憶素子の故障率　；0.01 Fit／bit
- データ幅　；72ビット（8ビットの検査ビットを含む）
- 語の大きさ　；128K語（$K = 2^{10}$）

（1MB容量）

- 1ビット誤り訂正回路のゲート量　；約1,000ゲート

これはメモリアレー　部のみを対象としているが，実際は直接周辺部，一部のデータパス回路も誤り検出・訂正の範囲にある。また，メモリアレーの語の大きさが大きい程，ηは大きくなる。従って，ηは少くとも100以上であろう。

ALUの場合には，次に示す仮定の下で，$\eta \approx 1.7$となる。

- 16ビット幅ALUのゲート数　；400ゲート
- 1ゲート当りの故障率　；5Fit／ゲート
- 1ビット誤り訂正回路のゲート数　；1,200ゲート

これから，ALUに対する誤り検出・訂正符号の効果は，主記憶の場合と比較して約2桁程度も小さいことが判明した。これが，ALUにおいて比較的高度な符号が実用にならない原因と考えられる。

そこで，ALUに対する誤り検出・訂正符号をその効果の点でより実用的な形とするためには，次の2通りの考え方が存在する。

(1) 誤り検出・訂正回路の構成を簡明にし，構成ゲート数，訂正のための回路遅延をできるだけ小さくする。

(2) 誤り検出・訂正回路が担当する検出・訂正のための範囲を出来るだけ大きくする。

AN符号等の剰余検査を基本とする符号は，その回路構成は複雑であり訂正のための速度も遅い。一方，主記憶に対しては，パリティを基本とする符号が実用化され，その回路構成も容易であり，訂正のための速度も速い。そこで，ALUに対してもこのパリティを基本とする符号を採用すれば，上記(1)の観点からその効果を高めることができるだけでなく，主記憶とALUの双方を検出・訂正のための範囲とすることができ，(2)の点からもさらに効果を高めることができるはずである。すなわち，ALUのみでなく主記憶およびその間のレジスタ，選択回路等のデータパス回路もその範囲とすることができる。

以上のような構成が実現できれば，次の仮定の下で$\eta \approx 55$とすることができる。

データ幅	；16ビット
1ビット誤り訂正回路のゲート数	；650ゲート
ALU+レジスタファイル（RALU）のゲート数	；800ゲート
主記憶の構成	；128K語×24ビット（8ビットの検査ビットを含む）
記憶素子故障率	；0,01Fit／ビット
ゲート当りの故障率	；5Fit／ゲート

第8章においては，このような観点からパリティを基本とする符号を用いてALUの誤りを検出・訂正する手法を示した。次節以降においては，パリティを基本とする種々の符号を適用した場合のシステム構成等を明らかにする。

9－3　システムモデル

本節は，小形の計算機システムのモデルを対象に，8－4で明らかにしたパリティを基本とする符号を適用する。適用に当っては，次の点に留意する。

(1)　ALUと主記憶で同一の符号が使用でき，これにより双方に対して同一の誤り検出・訂正機能を持つこと

(2)　ALUへの入力誤りを検出・訂正できること

(3)　上記(1)(2)から，ALUと主記憶間で単一の誤り検出・訂正回路を設置して，双方の誤りを検出・訂正できること

すなわち，ALUへの入力誤りを検出・訂正できる工夫をとり入れることによって，ALUと主記憶でその間のデータパス回路も含めて，単一の誤り検出・訂正回路（ECC_0）にて誤り検出・訂正を可能とするものである。

システムモデルを図9－1に示す。システムはALU，レジスタファイル（REG），主記憶ユニット（MEM），制御ユニット（C）および周辺ユニット（P）からなる。REGはALUへの入力データ等を一時的に蓄える記憶部であり，MEMは命令，データを蓄える主記憶である。CはALUの命令のシーケンス制御，アドレス制御等を行う制御部であり，Pはファイルメモリあるいは入出力端末等の周辺部である。これらのユニットはデータバスを通して互いに接続される。制御情報やアドレス情報は，他のバスで接続される（図中には示していない）。

従来，ALU部，主記憶部，周辺部，制御部では，それぞれ異なる高信頼化手法が採用されてきた。ここでは，周辺部，制御部を除いて，ハードウェア量の70％近くを有するALU部と主記憶部のデータパス回路に対して，共通にパリティを基本とする符号を適用する。図9－1においては，ALU，REG，MEMおよびその間のデータ系回路に対して適用する。これらの回路に対する高信頼化は，システム全体の高信頼化に大きな役割を果たす。

一般に，誤り検出・訂正符号を使用した装置またはシステムにおいては，その出力に誤り検出・訂正用の回路（復号化回路）を有し，その入力には符号生成回路（符号化回路）が用意されねばならない。本モデルでは，ALUとMEM間で符号を使用することから，この符号の出口である制御部（C）および周辺部（P）の入口に誤り検出・訂正回路（ECC_1）が設けられる。また，この符号の入口である周辺部（P）においては，符号化回路（G）が設けられる。ALUの出力における誤り検出・訂正回路（ECC_0）とECC_1は，その構成が異なり，図8－19に示すように前者はALU用の回路であり，後者は主記憶用の回路である。ECC_1の方がRを作成する回路や検査ビットの予測回路C_p等を含まない，より簡単な回路構造を有する。

制御部，周辺部においては，別の符号または高信頼化手法が適用される。制御部に対しては，シーケンス制御回路を二重化[8]したり，マイクロプログラムを蓄えるコントロールメモリに

は単純パリティ符号，SEC−DED符号，チェックサム符号[98]等が適用される。また，周辺部としては各種の装置が考えられ，ファイル記憶装置である磁気ディスク，磁気テープ装置等に対してはバースト誤り訂正符号が，また端末装置ではパリティ符号等が適用される。

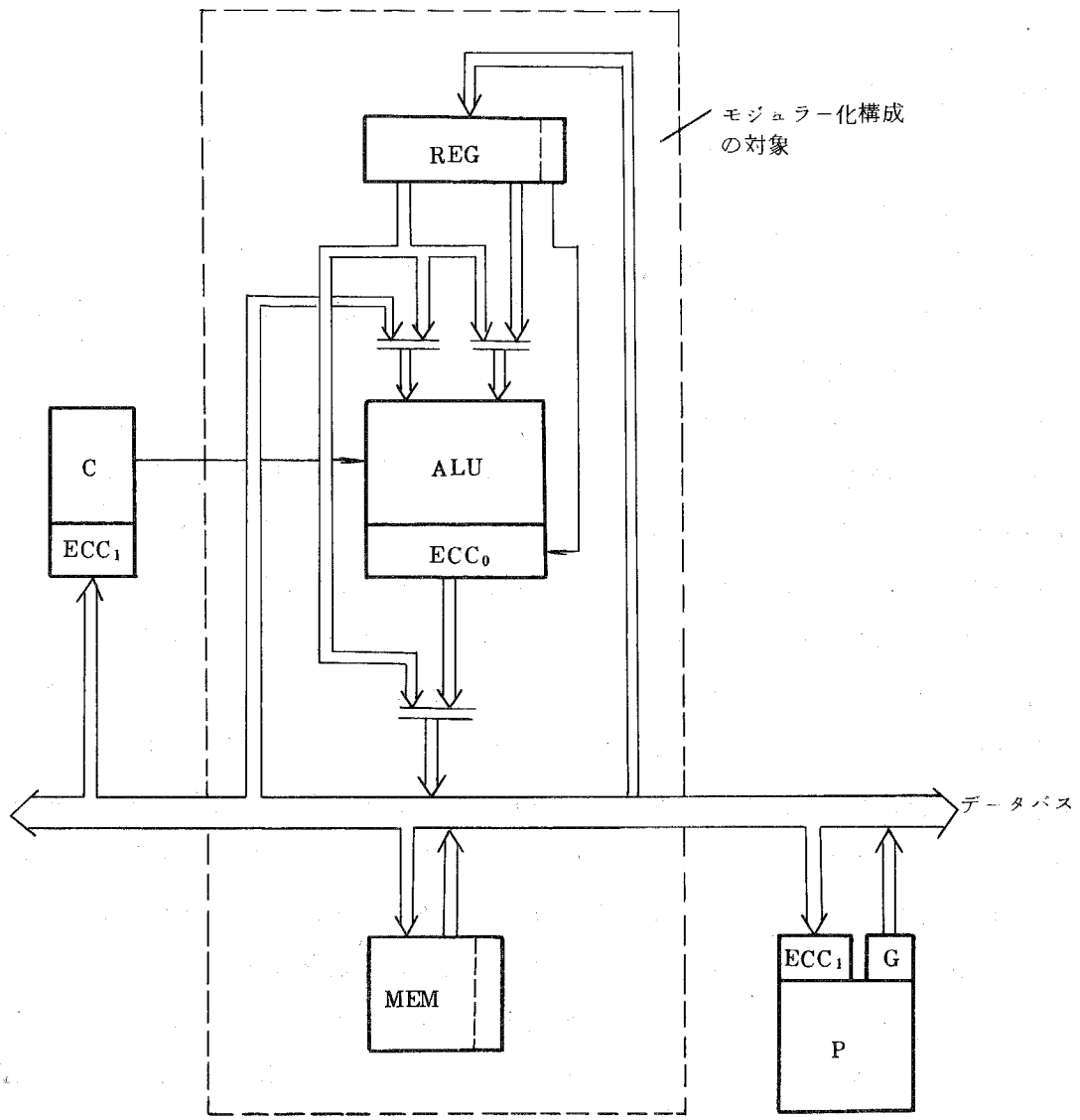


図9−1 システムモデル

9－4　誤り検出・訂正回路（ECC_0）の構成

第3章，第4章においては，パリティを基本とする各種の機能を有する符号を示した。本節では，これらの符号を具体的にALUの誤り検出・訂正に適用した場合の回路ゲート量，訂正速度等を示す。また，ALUへの入力誤りに対する補正回路，および自己検査性を有する誤り検出・訂正回路の構成について示す。

9－4－1　パリティを基本とする符号の適用[30]

ALUに対する誤り検出・訂正のために，具体的にパリティを基本とする符号を適用する。情報ビット数は16ビット，32ビットを対象とし，採用する符号は次の通りである。

① SEC－DED符号

② SEC－S4ED符号

③ SEC－DED－S4ED符号

④ S4EC符号

特に②～④のバイト系符号は，4ビットのバイトスライスALUと4ビットの入出力を有する主記憶用記憶素子を使用するシステムには，非常に効果的である。

また，ALUは加算（A＋B），論理積（A∩B），論理和（A∪B），排他的論理和（A⊕B），シフト，コンプリメント（$\overline{A}$，$\overline{B}$）等の演算が行われるものとする。シフト演算は任意のビット数の右，左シフトが可能であるとし，これに対しては誤りの検出のみに止めることとする。

上記①～④の符号に対して，図8－19に示す誤り検出・訂正回路（ECC_0）の規模を表9－1に示す。ゲート量は図8－19に示すECC_0中，R，C_p，SG，SD，COR，SCの各回路の合計を示す。算出上の条件は，7－2で示したと同一である。キャリ生成回路は4ビットをグループとするグループキャリルックアヘッド型を採用している。

図9－2に情報ビット数16ビットを有する各符号のHマトリクスを示す。表9－1から16ビットALUを400ゲート，32ビットALUを800ゲートとするとき，ECC_0のゲート量は1.0～1.3倍となることがわかる。これは表8－5に示すAN符号，組合せ符号による場合に比較してかなり小さい。また，訂正のための速度は8～12ゲート段数となり，第7章で示したように主記憶用の復号速度とほぼ同一である。

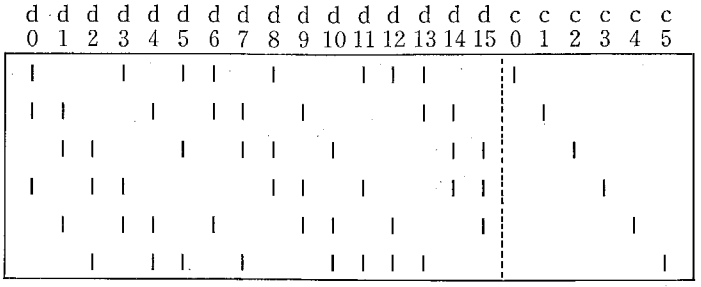
表９－１　誤り検出・訂正回路の規模

符　　号	k＝16 ビット				k＝32 ビット			
	検　査 ビット数 （ビット）	訂正速度* （ゲート 段数）	ゲート量 （ゲート）	** ゲート比率	検査ビット数 （ビット）	訂正速度* （ゲート 段数）	ゲート量 （ゲート）	** ゲート比率
SEC-DED	6	8	428	1.07	7	9	852	1.07
SEC-S4ED	6	9	408	1.02	7	10	782	0.98
SEC-DED-S4ED	7	9	452	1.13	8	10	838	1.05
S4EC	8	11	542	1.35	8	12	1070	1.34

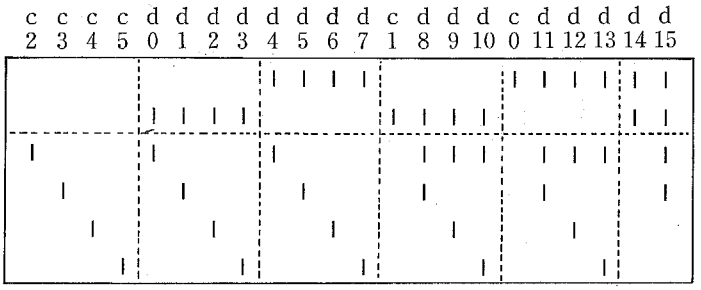
* ALU出力から訂正出力までのゲート段数

** ALUのゲート量に対する比率（１６ビット幅ALU＝400ゲート，32ビット幅ALU＝800ゲートとする）

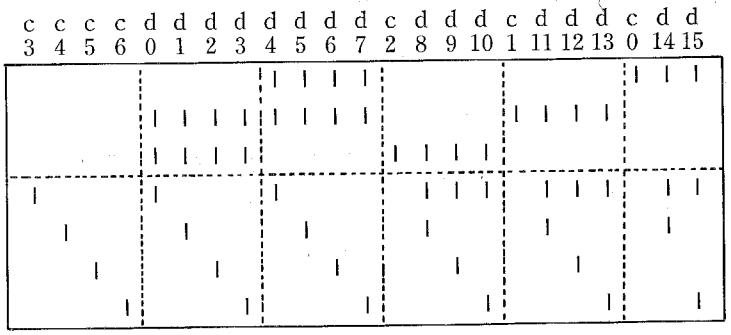
(i) (22,16) SEC-DED



(ii) (22,16) SEC-S_4ED



(iii) (23,16) SEC-DED-S_4ED



(iv) (24,16) S_4EC

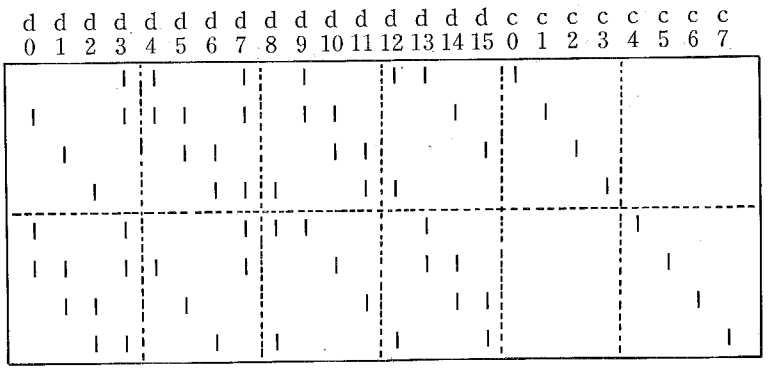


図９－２　誤り検出・訂正符号（16ビット情報幅の場合）

9－4－2　シフト回路とその自己検査性検査回路

シフト演算に対しては，パリティを基本とする符号を用いた誤り訂正は，ゲート量の点から実用的でない。ここでは，誤りの検出のみに止める。さらに，パリティを基本とする符号として奇数重み列符号を採用する。奇数重み列符号に対しては，次の定理に示すパリティ関係が存在する。

（定理９－１）

Ｘをｋビットの情報ビット，$X=(x_0\ x_1 \cdots x_{k-1})$，$C_x$ を奇数重み列符号のｒビットの検査ビット，$C_x=(c_{x_0}\ c_{x_1} \cdots c_{x_{r-1}})$ とするとき，符号語は〔Ｘ：C_x〕で表わされる。このとき，次のパリティ関係が成立する。

$$P_x = \sum_{j=0}^{k-1}{}^{\oplus}\ x_j = \sum_{i=0}^{r-1}{}^{\oplus}\ c_{x_i} \qquad (9-2)$$

ここで，P_x はＸに対する単純パリティである。　□

（証明）

ｉビット目の検査ビット c_{x_i} は，$h_{i,j} \in GF(2)$ をＨマトリクスのｉ行ｊ列の要素とするとき，次の関係が成立する。

$$c_{x_i} = \sum_{j=0}^{k-1}{}^{\oplus}\ x_j \cdot h_{i,j}, \qquad i=0,\ 1,\ \cdots,\ r-1 \qquad (9-3)$$

奇数重み列符号の性質を示す（２－８）式から，Ｈマトリクス要素 $h_{i,j}$ について次式が成立する。

$$\sum_{i=0}^{r-1}{}^{\oplus}\ h_{i,j} = 1 \qquad (9-4)$$

（９－３）式の両辺をｉ＝０からｒ－１までmod．２加算すると，次式が成立する。

$$\sum_{i=0}^{r-1}{}^{\oplus} c_{x_i} = \sum_{i=0}^{r-1}{}^{\oplus} \left(\sum_{i=0}^{k-1}{}^{\oplus} x_j \cdot h_{i,j} \right)$$

$$= \sum_{j=0}^{k-1}{}^{\oplus} \left(\sum_{i=0}^{r-1}{}^{\oplus} x_j \cdot h_{i,j} \right) \qquad (9-5)$$

$$= \left(\sum_{j=0}^{k-1}{}^{\oplus} x_j \right) \left(\sum_{i=0}^{r-1}{}^{\oplus} h_{i,j} \right)$$

$$= \sum_{j=0}^{k-1}{}^{\oplus} x_j = P_x \qquad \left(\because \sum_{i=0}^{r-1}{}^{\oplus} h_{i,j} = 1 \right)$$

よって，（9－2）式が成立する。

（証明終り）

定理９－１に示す性質を使用して，シフト演算に対する誤り検出として次の定理が成立する。

（定理９－２）

奇数重み列符号を使用して，情報〔A：C_A〕のシフト演算（演算結果を〔Y：C_Y〕とする）に対する検査は次式により行うことができる。

$$S = \left(\sum_{j=0}^{r-1}{}^{\oplus} c_{Y_j} \right) \oplus \left(\sum_{j=0}^{r-1}{}^{\oplus} c_{A_j} \right) \oplus P_{SI} \oplus P_{SO}$$

ここで，

$\sum_{j=0}^{r-1}{}^{\oplus} c_{Y_j}$ ；シフト演算結果に対する単純パリティ

$\sum_{j=0}^{r-1}{}^{\oplus} c_{A_j}$ ；入力情報Aに対する単純パリティ （9－6）

P_{SI} ；シフトインデータに対するパリティ

P_{SO} ；シフトアウトデータに対するパリティ

$S = 1$ ；誤り検出

$S = 0$ ；誤りなし

□

この定理は，通常のシフト演算に対するパリティ検査[6]と定理9－1を用いて容易に証明できるので省略する。この検査に対するTotally Self-Checking（T.S.C.）論理は次の定理により示すことができる。

（定理9－3）

奇数重み列符号のシフト演算に対するT.S.C.検査回路は次式で示される。

$$\left.\begin{aligned} C_0 &= \left(\sum_{j=0}^{r-1}{}^{\oplus}\ c_{Y_j}\right) \oplus P_{SI} \\ C_1 &= \overline{\left(\sum_{j=0}^{r-1}{}^{\oplus}\ c_{A_j}\right) \oplus P_{SO}} \end{aligned}\right\} \qquad (9-7)$$

$(C_0, C_1) = (1, 0)$または$(0, 1)$　；誤りなし

$(0, 0)$または$(1, 1)$　；誤り検出

□

（証明）

定理9－2から，$S=0$を与える符号語の入力は$(C_0, C_1) = (1, 0)$または$(0, 1)$を与えることは明らかである。一方，$S=1$を与える符号語でない入力に対しては$(C_0, C_1) = (0, 0)$または$(1, 1)$を与える。これから，この検査回路は"Code Disjoint"＊の性質を満たしている。

T.S.C.回路はSelf－Testingの性質とFault－Secureの性質の双方を有している。（9－7）式で示される検査回路は，EX－ORゲートのツリー（Tree）状で構成され，またC_0の回路はC_1の回路と独立である。また，すべてのEX－ORゲートの出力は，通常入力に対して'0'と'1'の双方の値をとりうる。それ故，stuck at－'1'またはstuck at－'0'に故障した1個のEX－ORゲートは，出力C_0またはC_1のいずれかに誤りをもたらす。これは，$(C_0, C_1) = (0, 0)$または$(1, 1)$となり，誤りであることを表示する。よって，単一のEX－ORゲート故障に対して，$(C_0, C_1) = (0, 0)$または$(1, 1)$となる少くとも1個の入力が存在することから，Self－Testingの性質を満足している。

一方，単一のEX－ORゲートの故障は，同時にC_0，C_1の双方を反転させることはない。これから，単一故障によって$(C_0, C_1) = (1, 0)$または$(0, 1)$となるようなことはなく，従ってFault－Secureの性質を満足している。以上から（9－7）式に示す検査回路はT.S.

＊　定義7－2参照

C．検査回路である。

（証明終り）

ここで，シフト演算は任意の長さのシフトが可能であるとする。従って，シフトのための回路は，元のｋビットのシフトレジスタの他に図９－３に示すように前後にｋ／２ビットの２個のシフトレジスタを設ける。P_{SI} または $P_{S\bar{0}}$ は最大ｋ／２ビットの長さに対するmod．２加算により得られる。例えば，右シフトの演算に対しては，シフトアウトデータは右のｋ／２ビットのシフトレジスタに蓄えられる。一方，シフトインデータ（例えば符号ビットのシーケンス）は，元のｋビットのシフトレジスタの最上位の位置から入力される。一方，左シフトの演算に対しては，シフトインデータは，一般に全ビット'0'である。このとき，シフトイン，シフトアウトに関係しないｋ／２ビットのシフトレジスタの内容は，すべて'0'でなければならない。図９－３には，（９－７）式を満足するT．S．C．検査回路も示している。

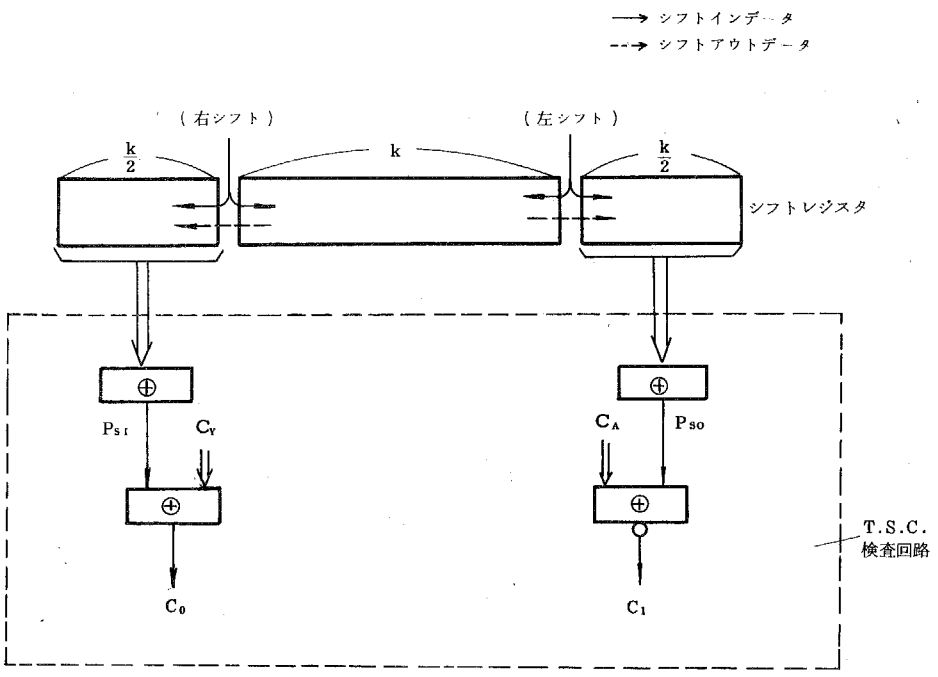


図９－３　シフト演算に対するT．S．C．検査回路

９－４－３　ECC_0 に対する自己検査性検査回路

図８－１９において，回路Ｒを除いたECC_0の回路中の単一故障を考える。ECC_0は主としてEX－ORゲートで作成されているため，この回路の単一故障は１ビット誤りとなる。

奇数重み列符号を採用する場合，次に示すT．S．C．検査回路は常にECC_0と検査回路自身の単一故障を検出する。

（定理９－４）

奇数重み列符号を用いたECC$_0$において，単一故障は次に示すT.S.C.検査回路によって検出できる。訂正後のデータを〔$\hat{Y}:\hat{C}_Y$〕とする。

$$\left.\begin{aligned} C_0 &= \sum_{i=0}^{k-1}{}^{\oplus}\ \hat{y}_i \\ C_1 &= \overline{\sum_{j=0}^{r-1}{}^{\oplus}\ \hat{c}_{Y_j}} \\ (C_0, C_1) &= (1, 0)\text{または}(0, 1) \quad ;\text{誤りなし} \\ &\quad (0, 0)\text{または}(1, 1) \quad ;\text{誤り検出} \end{aligned}\right\} \qquad (9-8)$$

□

この定理は，定理９－１と，定理９－３に示す証明とから，容易に証明できるので省略する。

回路R中の故障について考える。この故障は誤訂正を与える危険性を有する。この回路は，AND，OR，EX−ORゲートを有し，AND，OR演算結果，およびキャリビットを出力する。これから，次に示すEX−OR検査を各ビット毎に実行する。

$$\left.\begin{aligned} S_i &= (a_i \cap b_i) \oplus (a_i \cup b_i) \oplus (a_i \oplus b_i) \\ & \qquad i = 0, 1, \cdots, k-1 \\ S_i &= 1 \quad ;\text{誤り検出} \\ S_i &= 0 \quad ;\text{誤りなし} \end{aligned}\right\} \qquad (9-9)$$

（９－９）式に示す検査により誤りを検出すると，誤りビットを示す誤りポインタは無効にしなければならない。

また，グループキャリルックアヘッド型の加算回路を使用の場合は，グループキャリの誤りを検出するため（８－３２）式で示したようなバイト内最上位キャリビットとグループキャリとの比較検査を行う。(61)

９－４－４　入力誤りに対する訂正手法(30)

ALUおよびECC$_0$に誤りがないとき，入力誤りに対する具体的な誤り訂正手法についてのべる。入力AまたはBのいずれかに誤りが存在すると仮定する。

手法１：補正回路による方法

8－6－2で示した通り，AND，OR演算における入力誤りは，図8－19に示す構成では必ずしも訂正できない。この演算に対しては，以下に示す補正回路を設ける必要がある。

表8－7から，AND演算に対して，入力a_iの誤りは$b_i=0$のときRに伝搬し，Yに伝搬しない。この場合に誤訂正をもたらす。そのため，このときには誤りポインタを'1'から'0'へ強制的に変化させなければならない。同様に，$a_i=0$のときb_iに誤りが存在するとき，誤りポインタを変化させる必要がある。OR演算に対しては，$a_i=1$または$b_i=1$のとき，同様な誤り制御が必要である。奇数重み列符号を使用するとき，定理９－１から次に示す簡明な入力誤り検査が実行できる。

$$\begin{cases} E_A = \left(\sum_{i=0}^{k-1}{}^{\oplus}\ a_i\right) \oplus \left(\sum_{j=0}^{r-1}{}^{\oplus}\ c_{A_j}\right) \\ E_B = \left(\sum_{i=0}^{k-1}{}^{\oplus}\ b_i\right) \oplus \left(\sum_{j=0}^{r-1}{}^{\oplus}\ c_{B_j}\right) \end{cases} \qquad (9-10)$$

iビット目の誤りポインタe_iに対する補正回路を図９－４に示す。

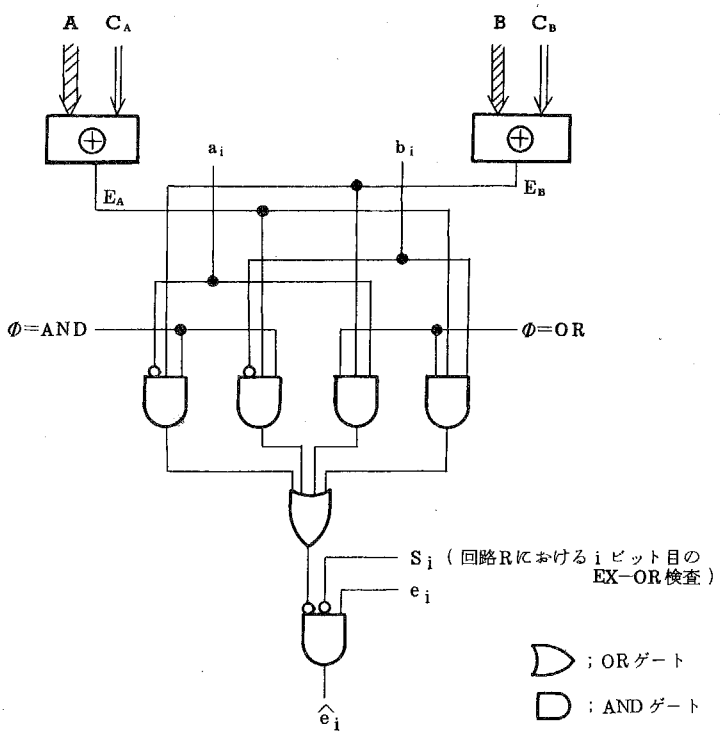


図9－4　誤りポインタe_iに対する補正回路

手法２：リトライによる方法

本手法は，リトライにより入力誤りを訂正する方法であり，これを図９－５に示す。リトライとしてΦ＝EX−OR演算を実行する点が特徴である。すなわち，EX−OR演算は必ず入力１ビット誤りは検出できることを利用している。しかも，この演算に対しては，回路Ｒは関係しない（表８－６より＊＝EX−ORのとき$r_i=0$）。EX−OR演算により誤りを検出するとこれは入力誤りと判定する。次に，入力ＡおよびＢに対して，Φ＝ＡまたはＢの演算（入力ＡまたはＢのデータをそのまま通過させる演算）を実行することにより，入力の誤りを訂正する。訂正されたデータＡまたはＢをＡＬＵへ再入力して，もとの演算Φ＝ＡＮＤまたはＯＲの演算を再実行する。

本手法はマイクロプログラムを使用して実現すると容易である。すなわち，Φ＝ＡＮＤまたはＯＲ演算において，シンドロームが非零になったとき，図９－５に示すフローを実行するマイクロプログラムルーチンが始動する。但し，このとき，回路Ｒの誤り，グループキャリの誤りはないものとする。

９－４－３で示したECC_0に対する検査回路および入力誤りに対する訂正回路は，表９－１に示すゲート量には含まれていない。但し，９－４－２でのべたシフト演算に対する検査回路（ＳＣ）は含まれている。情報ビット数１６ビットの場合，（９－８）式に示すECC_0検査回路，（９－９）式に示すEX−OR検査回路，および入力誤り訂正のための補正回路は，それぞれ４０，７０，１８０ゲートとなる。この場合，補正回路は手法１に基づくもので，手法２を採用することにすれば，ほぼマイクロプログラムの増加のみですむ。手法２を採用することにすると，これらの検査回路を含むECC_0は，もとのＡＬＵに対して1.3～1.6倍のゲート量となる。

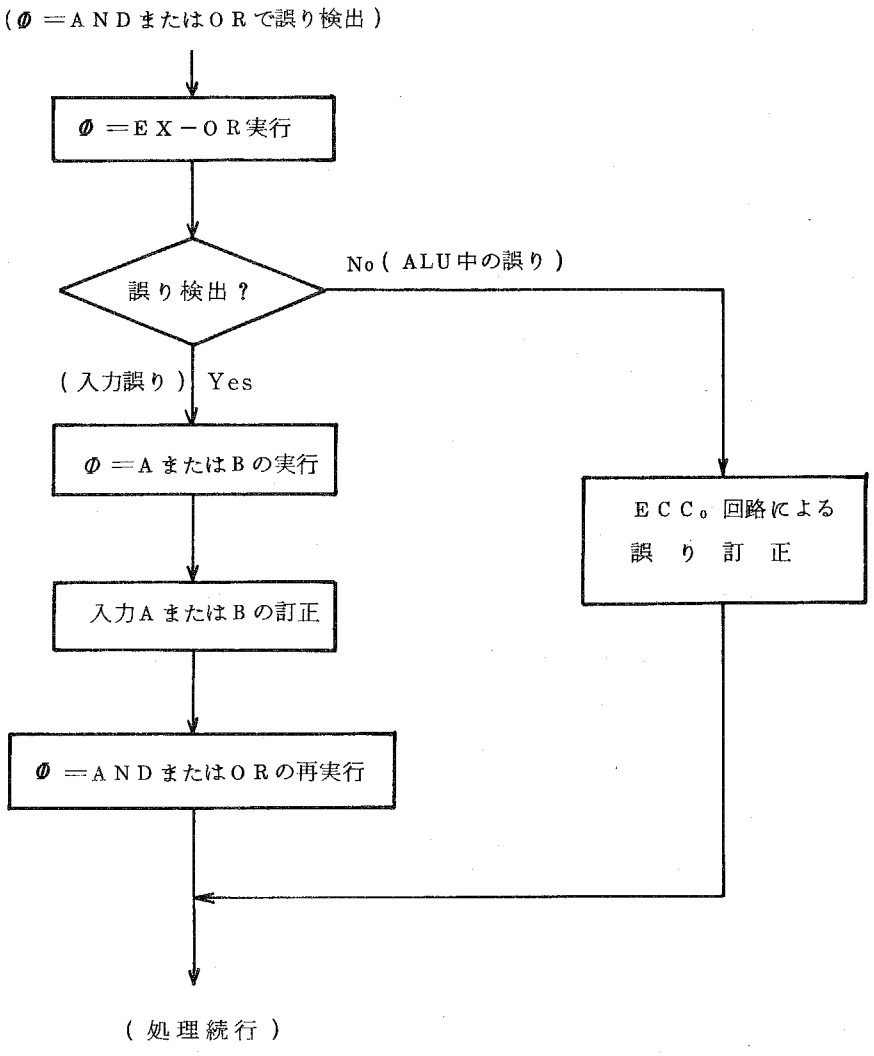


図9－5　リトライによる入力誤りの訂正

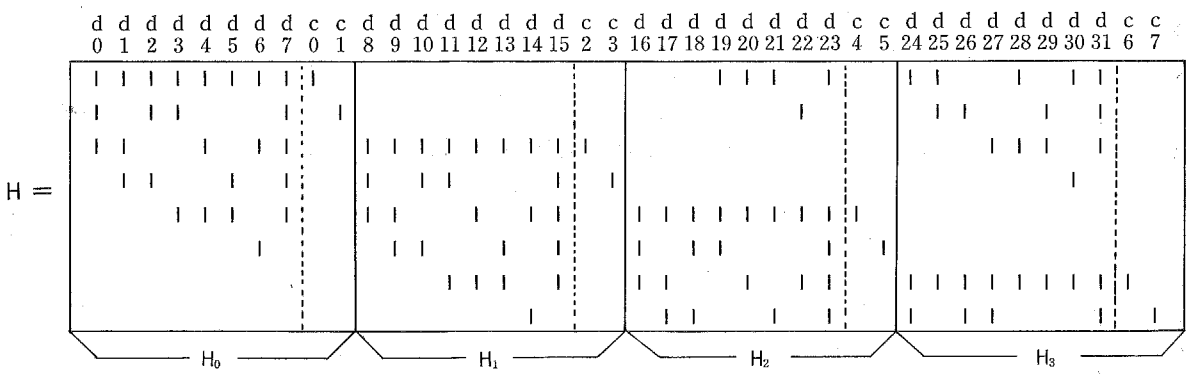
9－5　モジュラーなシステム構成

本節では，パリティを基本とする符号のうち巡回性の性質を有する符号を採用して，バイトスライスALU，バイト出力の記憶素子に合わせて，ECC_0もバイトスライス化することを考える。このようなモジュラー化を図る回路は，図９－１に示すシステム中，破線で囲んだREG, ALU, MEM, ECC_0の部分である。

簡単な例として，図９－６に示す巡回性奇数重み列（４０，３２）SEC−DED符号を採用してみよう。すなわち，ALU，MEMに対する情報ビット数は３２ビットであり，全体のシステムは図９－６中の部分マトリクスH_i（ｉ＝0，１，２，３）に対応して，バイト長ｂ＝１０ビット（検査ビット２ビット分を含む）を有する４個の同一モジュール（REG，ALU，MEM，ECC_0から構成される）で構成される。

バイトスライス化したREG，MEM，ALUは容易に構成できる。この場合，REGは８ビット出力を１塊りとするレジスタファイルであり，MEMは８ビット出力の記憶素子で構成される。また，ALUは８ビットのバイトスライスALUのＬＳＩを使用する。一方，バイトスライスECC_0は第７章で示した考えに従い，次のように構成できる。図９－６をもとに，８ビットの情報ビットと２ビットの検査ビットを入力させ，途中のパリティ検査結果およびシンドロームをバイトスライスECC_0間で交信する。このバイトスライスECC_0の回路規模は次のようになる。

ゲ　ー　ト　量　：　２３８ゲート
入出力信号数　：　５７本（入力３６本，出力２１本）



$d_0 \sim d_{31}$ ；情報ビット
$c_0 \sim c_7$ ；検査ビット

図９－６　巡回性奇数重み列（40，32）ＳＥＣ－ＤＥＤ符号

以上のバイトスライスしたREG, ALU, MEM, ECC_0から，1モジュールが構成できる。各モジールは8ビットの情報ビット，2ビットの検査ビットの10ビットのデータ幅を有し，モジュール当りの入出力信号は約90本となる。その内訳を以下に示す。

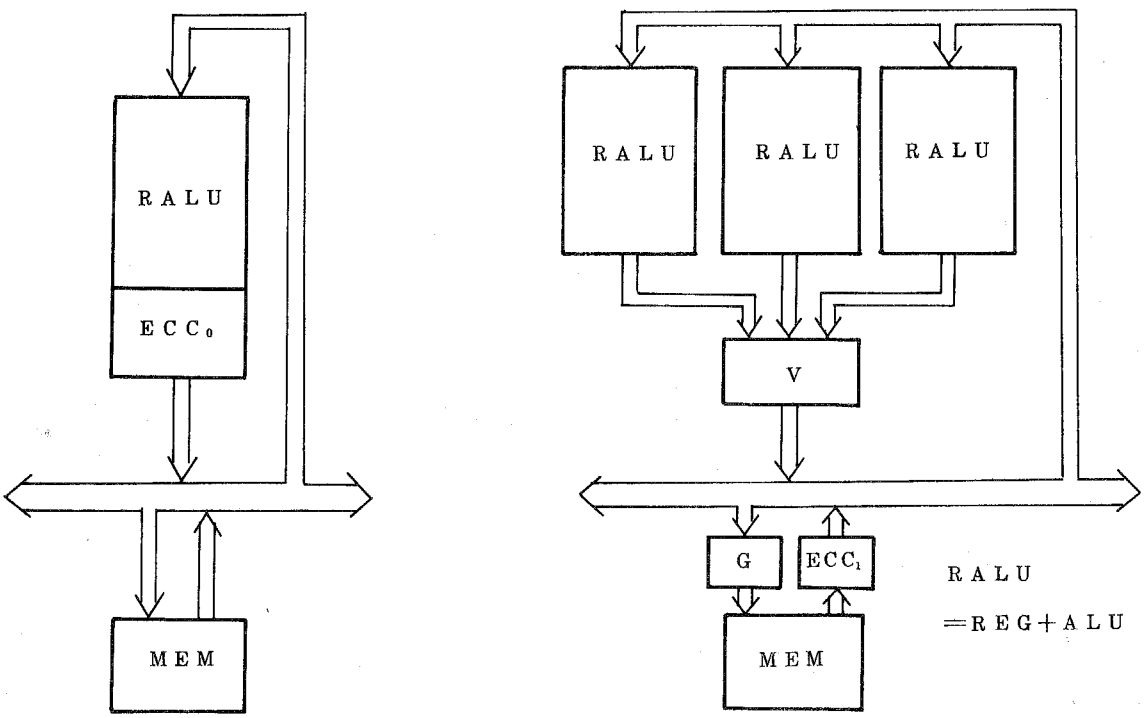
ALUに対する制御信号	約15本	モジュール間共通信号
MEMに対するアドレス，制御信号	約20本	
REG 指定信号	約 6本	
双方向バス入出力信号	20本	
キャリ生成用信号	11本	
ECC_0 の交信信号（シンドローム，途中パリティ検査結果）	約20本	

図9－1中の破線で囲んだ回路中，一部モジュラー化ができない回路が存在する。ECC_0に対するT.S.C.検査回路，シフト演算に対するT.S.C.検査回路がそれであり，これらについてはシステム中共通回路として設けられる。

9－6　TMRとの比較

本提案のシステムの中枢部は，図9－7(a)に示す形で表わすことができる。このシステムは，ALUの入力であるREG，あるいはMEMの誤りも1個のECC_0で検出・訂正可能である。これに対置する三重化構成（TMR）を，図9－7（b）に示す。すなわち，REGとALUを合体させたRALUを三重化し，それぞれの出力を多数決回路Vへ入力させる。一方，MEMの出力に対しては主記憶用の誤り検出・訂正回路（復号化回路）ECC_1，またMEMの入力に対しては符号生成回路Gを設けなければならない。このTMRの構成は，RALU中のすべての故障を訂正できる点ですぐれているが，RALU回路2個，多数決回路V，符号生成回路G，訂正回路ECC_1が余分に必要となる。

表9－2に，情報ビット数16ビットに対する本提案の手法とTMRの手法との比較を示す。この場合，両手法ともRALUは4ビットのバイトスライスRALUを，MEMは4ビット出力の記憶素子を使用する構成とする。表9－2から，TMRによる手法では，本提案の手法と比較して約1.5倍のゲート量を必要とする。訂正のための速度および訂正能力は，TMRの方がすぐれている。しかし，主記憶からの読出し～RALU動作～主記憶への書込みまでの動作時間は，本提案の方法の方が小さい。



(a)　本提案の構成　　(b)　TMR

図9－7　TMRとの構成上の比較

表9－2　TMRとの比較

項目＼手法	提案の手法	TMR	前提条件
構成	図9－7(a)	図9－7(b)	•16ビット情報幅 •4ビットバイトスライスALU使用 •4ビット出力記憶素子使用
全ゲート量（MEMを除く）	約1,500ゲート （RALU；150×4ゲート ECC_0；650　〃 R*　；200　〃）	約2,200ゲート （RALU；150×4×3 V；100 ECC_1+G；300）	4ビットスライスRALU；150ゲート ECC_0；S4EC機能** ECC_1；S4EC機能** *検査ビット分必要となるレジスタ
速度　RALU結果に対する訂正速度	11 ゲート段	3 ゲート段	
速度　MEM読出し後→RALU動作→MEM書込み直前までの速度	T＋11 ゲート段	T＋15 ゲート段	T：RALU動作時間 MEM動作時間は含まず
訂正能力	単一4ビットバイト幅の誤り訂正（RALUとMEMに対して）	•単一RALU中のすべての誤り訂正 •単一4ビットバイト幅の誤り訂正（MEMに対して）	

** （24，16）S4EC符号使用

9－7 結　　言

本章は，パリティを基本とする誤り検出・訂正符号を具体的に計算機システムに適用した結果についてのべた。主な結論を以下に示す。

(1) 誤り検出・訂正符号を装置またはシステムに適用する場合，符号の効果として誤り検出・訂正回路ゲート量に対する検出・訂正範囲にあるハードウエアの故障率の比ηを定義した。実用的観点からηを大きくすることが肝要である。

(2) AN符号等の剰余検査を基本とする符号はηが小さく，実用的観点から問題を有している。一方，パリティを基本とする符号は，検出・訂正用ゲート量が比較的小さいこと，また第8章で示したように入力誤りをも検出・訂正できる手段をとり入れることにより，ALUだけでなく，主記憶，レジスタファイル等のデータパス回路も誤り検出・訂正範囲とすることができることから，ηを大きくすることが可能となる。

(3) 簡単な計算機システムを想定し，これに共通的にパリティを基本とする符号を適用した。扱うデータビット幅を16ビット，32ビットとし，4ビットスライスALU，4ビット出力の記憶素子を使用し，SEC－DED，SEC－S4ED，SEC－DED－S4ED，S4EC の各機能を有する符号を適用した。その結果，もとのALU に対し1.0～1.3倍のゲート量で誤り検出・訂正回路が構成できることを明らかにした。

(4) パリティを基本とする奇数重み列符号を使用して，誤り検出・訂正回路に対するT．S．C．検査回路の構成を示した。また，入力誤りに対する具体的な訂正手法を示し，特にAND，OR演算時に誤りポインタの補正回路を設ける方法と，同演算に対し誤り検出時にEX－OR演算等の演算のリトライを行う方法を提案した。リトライによる方法は，マイクロプログラムにより実現すると効果的である。これらの回路を加えても，全体の誤り検出・訂正回路はもとのALU に対し，1.3～1.6倍のゲート量となることを明らかにした。

(5) 巡回性符号を適用することにより，誤り検出・訂正回路のバイトスライス化が実現でき，バイトスライスALU，バイト出力の記憶素子を使用して，全体がモジュラーな構成を有するシステム構成が実現できることを示した。

(6) 本提案の手法と三重化（TMR）の手法を具体的に比較した。その結果，16ビットのデータ幅を有するシステムにおいて，TMR による手法は本提案の手法と比較して約1.5倍のゲート量となるが，ALU 出力に対する訂正速度，訂正能力の点ですぐれていることを示した。

第10章　結　　論

本論文は，バイト構成を基本とするディジタルシステムにおいて，新しいバイト誤り検出・訂正符号を提案するとともに，これらの符号を用いて高速でＬＳＩ 化に適する符号化・復号化回路，信頼度改善効果，等を明らかにしたものである。その結果，具体的にバイトスライスＡＬＵ，バイト出力構成の記憶素子を用いた計算機システムに対し，これらのパリティを基本とするバイト誤り検出・訂正符号を共通に適用することにより，新しい高信頼ディジタルシステムを構築した。

得られた結論は各章毎にまとめたが，さらに総括的に要約すると以下のようになる。

(1)　論理装置（ＣＰＵ），主記憶装置（ＭＥＭ）の高信頼化を目的に，誤り検出・訂正符号を適用する上での具体的な要求条件を明らかにした。符号適用に当っては，装置内の故障発生傾向に基づいた最適な符号機能を選択することは勿論，高速な符号化・復号化，および検査ビット数，符号化・復号化回路のゲート量等の経済性に考慮を払わなければならない。また，冗長性を増加させない範囲で，検出能力の増大，符号化・復号化回路のＬＳＩ 化を容易にする奇数重み列符号，巡回性符号，等の新しい符号構成技術を採用することが実用性の観点から重要である。

(2)　半導体集積技術の著しい進展から，バイトスライスＡＬＵ，バイト構成を有する記憶素子の出現にともない，経済的なバイト誤り検出・訂正符号の構成が望まれる。この観点から，本論文では以下に示す各種の新しいバイト誤り検出・訂正符号を求めた。

i）従来から使用されているＳＥＣ 符号またはＳＥＣ－ＤＥＤ符号の符号機能を有し，さらに単一バイト内のすべての誤りを検出できるＳＥＣ－ＳｂＥＤ符号，およびＳＥＣ－ＤＥＤ－ＳｂＥＤ符号を新しく求めた。２種類の符号構成を明らかにし，いずれも検査ビット部分が明確で，任意のバイト長に対して系統的に構成できる符号である。ＳＥＣ－ＳｂＥＤ 符号はｂ＝４ビットのバイト長の場合，検査ビット数はＳＥＣ－ＤＥＤ符号とほぼ同一であり，８ビットのバイト長で３ビットの増加である。ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号の場合には，さらに１ビット増加する。これらの符号は比較的少ない検査ビット数で実現できること，符号化・復号化回路がほぼＳＥＣ－ＤＥＤ 符号の場合と同一で実現できることから，バイト誤り検出の機能を要求するシステムには最適な符号である。

ii）単一ｂビットのバイト内のすべての誤りを訂正する符号として，新しく奇数重み列の性質を有するＳｂＥＣ 符号を求めた。この符号は，Ｈマトリクスの列ベクトルがすべて奇数重みとなる特徴を有するとともに，２バイト間にわたる誤りが同一誤りパターンであればすべて検出できる検出能力の高い符号である。具体的に（７２，６４）Ｓ２ＥＣ符号に対して評価した

所，５～１５％他の奇数重み列を有さない符号と比較して検出能力がすぐれていることを明らかにした。また，この符号の場合，ｂ＝１としたときには，従来最も検出能力の点ですぐれている奇数重み列ＳＥＣ－ＤＥＤ符号に一致する。

ⅲ）単一ｂビットのバイト内のすべての誤りを訂正し，さらに二重のバイト間にわたるすべての誤りを検出する符号として，符号間最小距離４を有するReed-Solomon ＳｂＥＣ－ＤｂＥＤ符号が存在する。この符号では，ｂ＝２～４ビットに対して情報ビット数は高々６０ビットまでしか成立せず，情報ビット数６４ビット，１２８ビットを有する装置には適用できない。そこで，本論文では既存の２個のＳｂＥＣ－ＤｂＥＤ符号を組合わせて，Reed-Solomon符号とは異なる新しい符号構成を得た。この符号では，任意のバイト長，任意の情報ビット数に対して構成が可能である。ｂ＝２ビットで情報ビット数６４ビット，１２８ビットに対し，いずれも検査ビット数は１２ビット，ｂ＝４ビットでいずれも１６ビットで構成できる。

ⅳ）以上の各種の新しいバイト誤り検出・訂正符号に対して，符号のくり返し性を与える巡回性符号の構成を求めた。この符号構成はＨマトリクスにくり返し性を与え，基本の部分Ｈマトリクスを求めることにより，これを列方向に巡回シフトした各部分Ｈマトリクスを求め，これらを連接させて符号を構成するものである。このような構成とすることにより，その符号化・復号化回路を部分マトリクス対応に同一回路で構成でき，回路のＬＳＩ化に最適な構成を得ることができる。ＳＥＣ－ＳｂＥＤ符号，ＳＥＣ－ＤＥＤ－ＳｂＥＤ符号，奇数重み列ＳｂＥＣ符号に対し，一般的な巡回性符号としての構成を求めた。ＳｂＥＣ－ＤｂＥＤ符号に対しては，巡回性符号を理論的に求めることが困難であるため，計算機によりバイト長４ビット，情報ビット数６４ビット，１２８ビットの場合に対し，Ｈマトリクスの重みを最小とする巡回性符号を求めた。また，一般に，巡回性の性質を与えることによる符号長短縮の割合は非常に小さいことを明らかにした。

(3) バイト構成を有する記憶素子を用いたメモリアレーに対し，各種の符号を適用した場合の信頼度改善効果を明らかにした。計算に当っては，符号による効果の外に定期保守の効果も加味した。具体的な数値として，情報ビット数６４ビット，使用記憶素子６４Ｋ語×４ビット構成，定期保守間隔１週間，全メモリアレー容量１６Ｍバイト（バイト＝８ビット）を使用した。

ⅰ）ＳＥＣ－ＤＥＤ符号の場合，単一素子中に誤りが生じたとき２ビット以上の複数ビット誤りとなる確率をβとすれば，信頼度改善はβに依存する。すなわち，$\beta=0.1$であれば，信頼度は１桁改善する。

ⅱ）ＳＥＣ－ＤＥＤ－Ｓ４ＥＤ符号を適用すると，信頼度改善はＳＥＣ－ＤＥＤ符号の場合と同一であるが，符号を適用しても検出できずミスコレクトする率は，ＳＥＣ－ＤＥＤ符号と比較して２～３桁小さくなる。

ⅲ) S4EC 符号を適用すると，素子当りの故障率を1,000～2,000Fit とすると，符号を適用しない場合と比較して2桁以上信頼度改善となる。

ⅳ) S4EC−D4ED 符号の場合，信頼度改善はS4EC 符号と同一であるが，検出できない率は零とすることができる。

これから，バイト誤りとなる率の比較的小さい素子を用いた装置に対しては，SEC−DED−SbED 符号が，またバイト誤り率の比較的高い素子を用いた装置にはSbEC 符号，またはSbEC−DbED 符号が望ましい。

(4) 各種の符号を用いて，高速な並列符号化・復号化を実現する回路構成を求めた。

ⅰ) 回路ゲート量は，符号の機能よりむしろ情報ビット数，Hマトリクスの重みに主として依存する。また，回路遅延は，実用的な情報ビット数，符号機能に対して8ゲート段～14ゲート段と高速な復号が可能であることを明らかにした。

ⅱ) 符号化・復号化回路の高信頼化を目的に，自己検査性を有する検査回路の構成を求めた。これは，本回路をいくつかに分割し，それぞれに対しその入出力から共通の論理関係を求め，それらを比較する検査手法を適用した。この検査回路の増加は，もとの回路に対し25～30%となることを明らかにした。

ⅲ) 符号化・復号化回路のLSI 化を実現するため，巡回性符号を採用した。LSI 化に当っては，符号長の拡張性，多機能性，自己検査性検査回路の内蔵を考慮した。その結果，SEC−DEDとS2ECの機能を有し，これらの機能を選択して使用できるLSI 回路と，さらにS2EC の機能を包含するS4ECの機能を有し，3機能から選択して使用できるLSI 回路を求めた。これらの回路は，複数個使用することにより，複数倍符号長の増大した符号化・復号化回路を構成することができる。

(5) ALU に対する既存の誤り検出・訂正手法を整理するとともに，それらに対する問題点を明らかにした。その結果，AN 符号等の剰余符号による誤り検出・訂正手法は，回路ゲート量が大きく復号化時間も大きいことから，実用的に問題があることを明らかにした。そこで，加算回路に対する新しい誤り訂正手法として，二重化と単純パリティ符号を組合わせた手法と，水平・垂直パリティ符号を使用した手法を求めた。これらの手法による回路ゲート量の増加は，もとの加算回路に対し，1.3倍となり，また約85%の回路に対してその内の単一故障は誤訂正を与えない特徴を有する回路構成を得た。

(6) パリティを基本とする符号をALU に適用して，誤り検出・訂正できるための条件を求めた。これは，一般の演算結果を排他的論理和演算結果に線形に変換することを原理とするものであり，各種の基本的な算術・論理演算に対して適用可能であることを示した。また，ALU

中に誤りがなく，入力情報中に誤りが存在する場合について，上記原理にもとずき各種のパリティを基本とする符号を適用して，その誤りを検出・訂正できる条件を求めた。その結果，誤り検出に対しては，入力情報中の誤りは検出可能であるが，訂正に対しては特定の条件を満足する場合に可能となることを明らかにした。

(7) 誤り検出・訂正符号をシステムに適用する場合，符号化・復号化回路ゲート量に対する検出・訂正範囲にあるハードウェアの故障率の比を考え，この比を大きくすることにより符号の効果を大きくすることが実用上重要であることを明らかにした。この点から，パリティを基本とする符号を共通に使用して，ALU と主記憶中の誤りを検出・訂正することにより，単一の符号で検出・訂正範囲のハードウェア量を増大させることができ，符号の効果を増大できる。

(8) 簡単な計算機システムを想定し，パリティを基本とする誤り検出・訂正符号を具体的に適用した。その結果，4ビットスライスALU，4ビット出力の記憶素子を使用し，データビット幅16ビット，32ビットに対しSEC−DED，SEC−S4ED，SEC−DED−S4ED，S4ECの機能を有する符号を適用し，もとのALU に対し1.0～1.3倍のゲート量で誤り検出・訂正回路が構成できることを明らかにした。また，巡回性符号を適用することにより，誤り検出・訂正回路のバイトスライス化が実現でき，バイトスライスALU，バイト出力の記憶素子を使用して，全体がモジュラーな構成を有する高信頼システム構成が実現できることを示した。

(9) ALU，レジスタファイルを三重化し，さらに主記憶に誤り検出・訂正符号を適用する高信頼化手法と，パリティを基本とする符号をALU ～主記憶間に共通に使用する本提案の高信頼化手法との比較を行った。その結果，前者の手法は，後者に対し約1.5倍のハードウェア量となるが，ALU に対する誤り訂正は3倍以上高速となり，しかもALU，レジスタファイル中のすべての誤りを訂正できる特長を有する。しかし，主記憶読出し～演算実行・訂正～主記憶書込みのループを考慮すると，誤り検出・訂正を含む回路遅延は後者の方がすぐれていることを明らかにした。

(10) 本研究で得られた成果の一部は，1979 年武蔵野電気通信研究所で実用化した誤り訂正回路LSI に採用した。このLSI は，SEC−DED の機能を有し，巡回性の性質を有する符号を適用し，これによって16ビットのバイトスライス誤り訂正回路を構成している。1個のLSI で16ビットの情報ビット数を有するSEC−DED 回路が，2個で32ビットの情報ビット数を有する回路が，4個で64ビットの情報ビット数を有する回路が構成できる。この回路は，LCML の回路形式を有する1200 ゲートマスタスライスLSI で作られ，入出力端子数80端子を有するセラミックパッケージにおさめられている。

謝　　辞

本研究を進める上で，種々の暖かい御指導を賜わりました東京工業大学　工学部　当麻喜弘　教授に深く感謝申し上げます。また，本論文をまとめるに当り，御指導賜りました東京工業大学　工学部　榎本肇　教授，深尾毅　教授，柳沢健　教授，辻井重男　教授，志村正道　教授に感謝申し上げます。

本研究を進めるに当り，種々の御指導，御援助賜わりました日本電信電話公社　茨城電気通信研究所　川又晃　元所長，武蔵野電気通信研究所　電子装置研究部　寺島諒　部長，山田正計　統括調査役，飯田麒一郎　電子装置研究室長，檜山泰宏　元記憶装置研究室長（現在，ディーゼル機器㈱），集積記憶研究部　鈴木敏正　統括調査役　に深謝いたします。

本研究を進める上で有益な御助言，御教示頂きました横浜国立大学　工学部　今井秀樹　助教授，また大学卒業研究以来，御激励を賜わりました東京工業大学　工学部　末武国弘　教授，内藤喜之　教授，清水康敬　助教授に感謝申し上げます。

武蔵野電気通信研究所　電子装置研究室　金田重郎　研究主任には，バイト誤り検出・訂正符号および巡回性符号の研究に関して，また同研究室　晴田和夫　研究主任には，自己検査論理，ALU に対する誤り検出・訂正の研究に関して，終始有益な御討論，御援助を頂き，心から感謝申し上げます。

付録 1. 定理 3 － 7 の証明

（3－21）式に示す符号長をN_B，構成Aの符号の符号長（3－13）式をN_Aとする。

$$\begin{cases} N_B = \{ 2^{br} - (2^b - b)^r \} \, 2^{\ell-1} + \ell & \cdots\cdots 1 \leq \ell \leq b-1 \quad (A-1) \\ N_B' = \dfrac{1}{2} \{ 2^{br} + b^r - (2^b - b)^r \} & \cdots\cdots \ell = 0 \text{ または } b \quad (A-2) \end{cases}$$

$$N_A = b \cdot 2^{R-b} \qquad (A-3)$$

数学的帰納法にて証明する。

〔1〕 $1 \leq \ell \leq b-1$ に対して

（i） $R = 3b + \ell$ のとき

$$N_B - N_A$$

$$= [\{ 2^{3b} - (2^b - b)^3 \} \, 2^{\ell-1} + \ell] - b \cdot 2^{2b+\ell}$$

$$> \frac{1}{2^{\ell-1}} \{ 2^{3b} - (2^b - b)^3 - 2b \; 2^{2b} \} = \frac{P_3}{2^{\ell-1}} \qquad (A-4)$$

$$P_3 = 2^{3b} - (2^b - b)^3 - 2b \cdot 2^{2b}$$

$$= b \cdot 2^{2b} - 3 \cdot 2^b \cdot b^2 + b^3 \qquad (A-5)$$

$$= b \{ (2^b - b)^2 - b \cdot 2^b \}$$

$b \geq 3$ で $P_3 > 0$

$\therefore b \geq 3$ で $N_B > N_A$

（ii） ある r（3以上の整数）に対して，

（A－5）式のP_3に相当するP_rは次式で表わせる。

$$P_r = \{ 2^{br} - (2^b - b)^r \} - 2b \cdot 2^{rb-b} \qquad (A-6)$$

$P_r > 0$ と仮定する。

(iii) $r+1$ に対して

$$P_{r+1} = 2^{b(r+1)} - (2^b - b)^{r+1} - 2b \cdot 2^{rb}$$

$$> 2^b \{ 2^{br} - (2^b - b)^r - 2b \cdot 2^{rb-b} \} = 2^b \cdot P_r > 0 \qquad (A-7)$$

以上から，$1 \leq \ell \leq b-1$ に対して，$b \geq 3$ で $N_B > N_A$ となる。

〔2〕 $\ell = 0$ または b に対して

(i) $r = 3$ ($R = 3 \cdot b$) のとき

$$N_B' - N_A$$

$$= \frac{1}{2} [\{ 2^{3b} - (2^b - b)^3 + b^3 \} - 2b \cdot 2^{2b}]$$

$$= \frac{b}{2} (2^b - b) (2^b - 2b) \qquad (A-8)$$

$b \geq 3$ で $N_B - N_A' > 0$

(ii) 任意の r に対して

$$N_B' - N_A$$

$$= \frac{1}{2} [\{ 2^{br} - (2^b - b)^r + b^r \} - 2b \cdot 2^{rb-b}] = \frac{1}{2} (P_r + b^r) \qquad (A-9)$$

$P_r > 0$ と仮定して，$r \rightarrow r+1$ とすれば，上記 (A−7) と同様のことが言える。

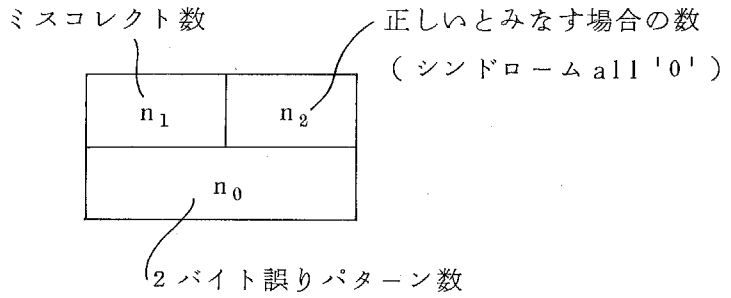
以上から，$\ell = 0$ またはbに対しても， $b \geq 3$ で $N_B' > N_A$ となる。

以上から $b \geq 3$， $r \geq 3$（$R \geq 3b$）に対して，（A－1）式，（A－2）式で表わされる符号は構成Aの符号に比較して符号長は大きい。

（証明終り）

付録 2. バイト誤り検出符号の評価結果

以下の符号評価の見方は次のようである。誤りは2バイトにわたるものとし，それぞれ e_i，e_j とする。



$$p = \text{非検出率} = \frac{\text{全ミスコレクト数（正しいとみなす場合も含む）}}{\text{全2バイト誤りパターン数}} \quad (\%)$$

(1) （72，64）SEC−S4ED符号（図3−2）に対する2バイト誤り評価

e_j ＼ e_i	1 ビット		2		3		4	
ビット 4	0	0	0	0	1224	0	0	153
	1224		1836		1224		153	
3	1080	0	3672	0	900	0		
	4896		7344		2448			
2	3672	0	0	918				
	7344		5508					
1	900	0						
	2448							

$p = 36.37\ \%$

(2) （７３，６４）SEC−DED−S４ED符号（図３−１１）に対する２バイト誤り評価

e_j \ e_i	1 ビット		2		3		4	
ビット4	0	0	0	0	1224	0	0	153
	1242		1836		1224		153	
3	0	0	3672	0	0	0		
	4968		7344		2448			
2	3672	0	0	918				
	7452		5508					
1	0	0						
	2520							

p＝２７.８％

(3) （７６，６４）SEC−DED−S４ED（図３−１５）に対する２バイト誤り評価

e_j \ e_i	1 ビット		2		3		4	
ビット4	48	0	0	0	672	0	0	84
	1368		2052		1368		171	
3	0	0	2016	0	0	0		
	5472		8208		2736			
2	2352	0	0	472				
	8208		6156					
1	0	0						
	2736							

p＝１４.７％

(4)　(73, 64)SEC−DED−S4ED符号(図3−20)に対する2バイト誤り評価

e_j \ e_i	1 ビット		2		3		4	
ビット4	432	0	0	0	648	0	0	72
	1242		1836		1224		153	
3	0	0	1944	0	0	4		
	4968		7344		2448			
2	4464	0	0	432				
	7452		5508					
1	0	0						
	2520							

p＝23.33％

付録 3. 定理 4 － 6 の証明

符号が最小ハミング距離 d_{min} を有する必要十分条件は，そのHマトリクスの（$d_{min}-1$）個以下の列ベクトルが線形独立であることである。この場合には，$d_{min}=4$ であるから，3個以下の列ベクトルが線形独立であることを証明すればよい。

そこで，$C_{i,j}$ 中から，任意の相異なる3列ベクトルを$C_{i_0 j_0}$，$C_{i_1 j_1}$，$C_{i_2 j_2}$ とする。今，それぞれに対応するバイト誤りをE_0，E_1，E_2とし，次の関係式において，$E_0=E_1=E_2=0$が唯一の解であるとき，この符号は最小ハミング距離4を有する符号と言える。

$$E_0 \cdot \begin{bmatrix} V_{i_0} \\ W_{j_0} \end{bmatrix} + E_1 \cdot \begin{bmatrix} V_{i_1} \\ W_{j_1} \end{bmatrix} + E_2 \cdot \begin{bmatrix} V_{i_2} \\ W_{j_2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (A-10)$$

H_V は最小ハミング距離4の符号のHマトリクスであることから，次式を満足するのは，$E_0=E_1=E_2=0$の場合のみである。

$$E_0 \cdot [V_{i'}] + E_1 \cdot [V_{i''}] + E_2 \cdot [V_{i'''}] = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (A-11)$$

$$i' \neq i'' \neq i''' \neq i', \qquad i',\ i'',\ i''' \in \{0, 1, \cdots, n-1\}$$

一方，H_W も同様の関係が成立する。

$$E_0 \cdot \begin{bmatrix} I \\ W_{j'} \end{bmatrix} + E_1 \cdot \begin{bmatrix} I \\ W_{j''} \end{bmatrix} + E_2 \cdot \begin{bmatrix} I \\ W_{j'''} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (A-12)$$

$$j' \neq j'' \neq j''' \neq j' \qquad j',\ j'',\ j''' \in \{0, 1, \cdots, m-1\}$$

今，（A－10）式について，次の3通りの場合を考える。

(i)　$i_0 = i_1 = i_2$ のとき，

W_{j_0} ，W_{j_1} ，W_{j_2} は相異なることから，（A－12）式を考慮すると，$E_0 = E_1 = E_2 = 0$ が（A－10）式を満足する唯一の解である。

(ii)　i_0 ，i_1 ，i_2 中任意の2個が等しく，残りの1個が異なるとき，

$i_0 = i_1$ ，$i_0 \neq i_2$ としても一般性を失わないであろう。これから，V_{i_0} と V_{i_2} は相異なる。これから，次式が成立する。

$$(E_0 + E_1)\cdot\begin{bmatrix} V_{i_0} \end{bmatrix} + E_2\cdot\begin{bmatrix} V_{i_2} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (A-13)$$

これから，E_0 ，E_1 ，E_2 の間には次の関係が成立する。

$$\begin{cases} E_0 + E_1 = 0 \\ E_2 = 0 \end{cases} \qquad (A-14)$$

このとき，（A－10）式は次のように書ける。

$$E_0\cdot\begin{bmatrix} V_{i_0} \\ W_{j_0} \end{bmatrix} + E_1\cdot\begin{bmatrix} V_{i_1} \\ W_{j_1} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (A-15)$$

W_{j_0} と W_{j_1} は相異なることから，（A－15）式が成立するのは，$E_0 = E_1 = 0$ のときのみである。以上から，$E_0 = E_1 = E_2 = 0$ が唯一の解となる。

(iii)　i_0 ，i_1 ，i_2 がすべて相異なるとき，

（A－11）式，（A－12）式から，$E_0 = E_1 = E_2 = 0$ が唯一の解であることは明らかである。

以上の3通りの場合について，$E_0 = E_1 = E_2 = 0$ が唯一の解であり，これから，この符号は最小ハミング距離4を有する符号と言える。

V_i はn個，W_j はm個それぞれ相異なるベクトルとして存在するから，これらをすべて組合わせたベクトル $C_{i,j}$ はm・n個存在する。よって最大符号長はm・nに等しい。

（証明終り）

参 考 文 献

(1) W.W.Peterson, E.J.Weldon, Jr. "Error Correcting Codes," Second Edition, MIT Press, 1971

(2) S.Lin "An Introduction to Error Correcting Codes" Prentice-Hall, 1970

(3) E.Berlekamp "Algebraic Coding Theory" N.Y. McGraw-Hill, 1968

(4) 宮川，岩垂，今井 "符号理論" 昭晃堂，1973年

(5) 嵩，岩垂，都倉，稲垣 "符号理論" コロナ社，1975年

(6) F.F.Sellers, Jr., M.Y.Hsiao, L.W.Bearnson, "Error Detecting Logic for Digital Computers" McGraw-Hill, 1968

(7) T.R.N.Rao "Error Coding for Arithmetic Processors" Academic Press, 1974

(8) W.F.Wakerly "Error Detecting Codes, Self-Checking Circuits and Applications" North-Holland, 1978

(9) 福村，後藤 "算術符号理論" コロナ社，1978年

(10) 北村，富田 "RAS 技術の現状" 情報処理，p497-p504，1971年8月

(11) 喜田 "計算機システムの高信頼化手法" 電子通信学会誌，p58-p67. 1973年1月

(12) 渡辺 "許容故障機械と安全故障機械" 電子通信学会誌，p1375-p1382，1973年10月

(13) 当麻 "フォールトトレラントとその関連技術" 電子通信学会誌，p359-p368, 1976年4月

(14) 当麻 "フォールトトレラントコンピューティングに関する最近の話題 -総論-" 昭和53年電気四学会連合大会 200, 1978年

(15) M.Y.Hsiao, J.T.Tou "Application of Error Correcting Codes in Computer Reliability Studies" IEEE Trans. on Reliability, vol R-18, №3, p108-p118, August, 1969

(16) A.Avizienis "Approaches to Computer Reliability-Then and Now" National Comput. Conf. p401-p411, 1976

(17) A.Avizienis "Fault-Tolerant Systems" IEEE Trans. on Comput. vol C-25, №12, p1304-p1312, Dec. 1976

(18) S.Y.H.Su, R.J.Spillman "An Overview of Fault-Tolerant Digital

System Architecture" National Comput. Conf. p19−p26, 1977

(19) 藤原，青木 "メインメモリの信頼性" 情報処理，16，4，p295−p304，1975年

(20) 檜山，川上，青木，藤原，江川 "DIPS−11 主記憶装置" 通研実報，26，3，p903−p916，1977年

(21) E.Fujiwara "A Modularized b−Adjacent Error Correction Memory Unit" Trans. IECE of Japan, E60, 2, p69−p76, 1977

(22) E.Fujiwara "Odd−Weight−Column b−Adjacent Error Correcting Codes" Trans. IECE of Japan, E61, 10, p781−p787, 1978

(23) E.Fujiwara, T.Kawakami "Modularized b−Adjacent Error Correction" 7'th Int.Symp.on Fault−Tolerant Comput. (FTCS−7), p199, 1977

(24) E.Fujiwara "Error Control for Byte−per−Package Organized Memory Systems" Trans. IECE of Japan, E63, 2, p98−103, 1980

(25) E.Fujiwara, S.Kaneda "Rotational Byte Error Detecting Codes for Memory Systems" Trans. IECE of Japan ,E64 ,5 ,1981

(26) S.Kaneda, E.Fujiwara "Single Byte Error Correcting−Double Byte Error Detecting Codes for Memory Systems" 10'th Int. Symp.on Fault−Tolerant Comput. (FTCS−10), p41−p46, 1980

(27) 藤原，晴田 "自己検査性を有する主記憶用誤り検査訂正回路の構成" 電子通信学会，論文誌(D)，J62−D，6，p419−426，1979年

(28) 藤原，晴田 "汎用性を考慮した主記憶用誤り検査訂正回路の構成" 電子通信学会，論文誌(D)，J63−D，2，p129−p136，1980年

(29) 藤原，晴田 "加算回路に対する誤り検査訂正手法" 昭和54年度，電子通信学会，総合全国大会 №1539，1979年

(30) E.Fujiwara, K.Haruta "Fault−Tolerant Computer Systems Using Parity−Based Codes" Trans. IECE of Japan, 投稿中

(31) R.W.Hamming "Error Detecting and Error Correcting Codes" Bell Sys.Tech. J. vol.26, №2, p147−p160, April, 1950

(32) A.M.Patel, S.J.Hong "Optimal Rectangular Code for High Density Magnetic Tapes" IBM J.Res. Develop. p579−p588, Nov. 1974

(33) A.M.Patel "Error Recovery Scheme for the IBM 3850 Mass Storage System" IBM J.Res. Develop. 24, 1, p32−p42, 1980

(34) M.Y.Hsiao, "A Class of Optimal Minimum Odd−Weight−Column SEC−

DED Codes" IBM J.Res. Develop. p395−p401, July, 1970

(35) 藤原，川上 "主記憶装置における2重誤り訂正の一手法 −BCH符号による遂次訂正法−" 電子通信学会 電子計算機研究会 EC76−20，1976年

(36) 今井，上柳 "2重誤り訂正BCH符号の並列復号器について" 電子通信学会，論文誌(D) J60−D，9，p761−p762，1977年

(37) S.J.Hong, A.M.Patel "Double Error Correcting Method and System" U.S.Patent №3714629, 1973

(38) 松澤，当麻 "主メモリに適した多重誤り訂正の一方式" 電子通信学会，論文誌(D)，J60−D，10，p869−p876，1977年

(39) M.Y.Hsiao, D.C.Bossen, R.T.Chien "Orthogonal Latin Square Codes" IBM J.Res. Develop. 14, 4, p390−p394, 1970

(40) 堀口，森田 "並列メモリの2重誤り訂正の一方式 −2重誤り訂正1段直交可能多数決符号の一構成法−" 電子通信学会，電子計算機研究会，EC75−42，1975年

(41) 今井，上柳 "主記憶のための2重誤り訂正符号の一構成法" 電子通信学会，論文誌(D)，J60−D，10，p861−p868，1977年

(42) D.C.Bossen "b−Adjacent Error Correction" IBM J.Res. Develop. p402−p408, July, 1970

(43) S.J.Hong, A.M.Patel "A General Class of Maximal Codes for Computer Applications" IEEE Trans. on Comput. vol c−21, №12, p1322−p1331, Dec.1972

(44) 今井，上柳 "主記憶のためのバースト誤り訂正符号について" 電子通信学会，論文誌(D)，J62−D，10，p633−p640，1979

(45) D.C.Bossen, L.C.Chang, C.L.Chen "Measurement and Generation of Error Correcting Codes for Package Failures" IEEE Trans.on Comput. vol.C−27, №3, p201−p204, March, 1978

(46) S.M.Reddy "A Class of Linear Codes for Error Control in Byte−per−Card Organized Digital Systems" IEEE Trans, on Comput. vol. C−27, №5, p455−p459, May, 1978

(47) I.S.Reed, G.Solomon "Polynomial Codes over Certain Finite Fields" J.Soc. Ind. Appl. Math. vol.8, p300−p304, June, 1960

(48) A.K.Bhatt, L.L.Kinney "A High Speed Parallel Encoder/Decoder for b−Adjacent Error Checking Codes" 3rd. USA−JAPAN Comput. Conf. p203−p207, 1978

(49) W.G.Bouricius, W.C.Carter, E.P.Hsieh, D.C.Jessep, Jr, A.B.Wadia "Modeling of a Bubble Memory Organization with Self-Checking Translators to Achieve High Reliability" IEEE Trans. on Comput. vol.C-22, №3, p269-p275, March. 1973

(50) W.C.Carter, A.B.Wadia "Design and Analysis of Codes and Their Self-Checking Circuit Implementations for Correction and Detection of Multiple b-Adjacent Errors" 10'th Int. Symp.on Fault-Tolerant Comput. p35-p40, Oct. 1980

(51) 麻生，当麻 "多重隣接誤り訂正符号の構成について" 電子通信学会，電子計算機研究会，EC79-45，1979年12月

(52) D.T.Brown "Error Detecting and Correcting Binary Codes for Arithmetic Operations" IEEE Trans. EC, p333-p337, Sept. 1960

(53) P.G.Neumann, T.R.N. Rao "Error Correcting Codes for Byte-Organized Arithmetic Processors" IEEE Trans. Comput. vol.C-24, №3, p226-p232, March, 1975

(54) T.R.N.Rao, O.N.Garcia "Cyclic and Multiresidue Codes for Arithmetic Operations" IEEE Trans. Inf. theory, vol. IT-17, №1, Jan. 1971

(55) D.Mandelbaum "Error Correction in Residue Arithmetic" IEEE Trans. Comput. vol C-21, №6, p538-p545, June,1972

(56) T.R.N.Rao "Biresidue Error-Correcting Codes for Computer Arithmetic" IEEE Trans. Comput. vol.C-19, №5, p398-p402, May, 1970

(57) T.R.N.Rao "Error Correction in Adders Using Systematic Subcodes" IEEE Trans. Comput. vol.C-21, №3, p254-p259, Mar. 1972

(58) H.L.Garner "Generalized Parity Checking" IRE Trans. EC, p207-p213, Sept. 1958

(59) O.N.Garcia, T.R.N.Rao "On the Method of Checking Logical Operations" Proc. 2nd, Annual Princeton Conf. Inform. Sci. Sys. p89-p95, 1968

(60) B.Khodadad-Mostershiry "Parity Prediction in Combinational Circuit" 9'th Int. Symp. on Fault-Tolerant Comput. p185-p188, June, 1979

(61) T.R.N.Rao, H.J.Reinheimer "Fault-Tolerant Modularized Arithmetic Logic Units" National Computer Conference. p703-p710, 1977

(62) 青木，金田，市毛，小林，川上 "DIPS-11改良主記憶装置" 通研実報，29，3，p375-p384，1980年

(63) T.Mano, K.Takeya, T.Watanabc, et.al" A 256K RAM Fabricated with Molybdnum-Polysilicon Technology" Int.Solid-State Circuit Conf.（ISSCC）FAM 17.8, p234-p235, 1980

(64) ボール，チュー "高集積化によりシステムを高速化するECL のバイト・スライス・マイクロプロセッサ" 日経エレクトロニクス，1980年1月21日号

(65) 川野辺，青木 "主記憶におけるソフトエラー対策について" 昭和54年度電子通信学会，半導体・材料部門全国大会，S1-6，1979年

(66) D.C.Bossen, M.Y.Hsiao "A System Solution to the Memory Soft Error Problem" IBM J.Res. Develop. 24, 3, p390-p397, 1980

(67) 今井 "主メモリにおける誤り訂正符号の応用" 昭和53年電気四学会 連合大会，202，1978年

(68) 今井 "情報の符号化と処理－最近の動向 情報伝送の符号化と処理B" 昭和55年電気四学会連合大会 32-6，1980年

(69) 安積，嵩 "最適な修正ハミング符号について" 電子通信学会，論文誌（A）vol.58-A, №6, p325-p330, 1975年6月

(70) C.W.Sundberg "Erasure and Decoding for Semiconductor Memories" IEEE Trans. Comput. vol.C-27, №8, p696-p705, Aug. 1978

(71) C.L.Sundberg "Properties of Transparent Shortened Codes for Memories with Stuck-at-Faults" IEEE Trans. Comput. vol.C-28, №9, p686-p690, Sept. 1979

(72) W.C.Carter, C.E.McCarthy "Implementation of an Experimental Fault-Tolerant Memory System" IEEE Trans. Comput. vol.C-25, №6, p557-p568, June, 1976

(73) J.J.Shedletsky "Error Correction by Alternate-Data Retry" IEEE Trans. Comput. vol.C-27, №2, p106-p112, Feb. 1978

(74) H.O.Burton "Some Asymptotically Optimal Burst-Correcting Codes and Their Relation to Single-Error-Correcting Reed-Solomon Codes" IEEE Trans. Information Theory, vol.IT-17, №1, p92-p95, Jan。1971

(75) J.K.Wolf "Adding Two Information Symbols to Certain Nonbinary

BCH Codes and Some Applications" Bell Sys. Tech. J. p2405−p2424, Sept. 1969

(76) D.C.Bossen, S.J.Hong, M.Y.Hsiao, A.M.Patel "Modular Distributed Error Detection and Correction Apparatus and Method" U.S.Patent. №3825893, July, 1974

(77) A.S.Lui, M.Arbab "Error Checking and Correcting Device" U.S. Patent, №4077028, Feb. 1978

(78) I.Bazovsky "Reliability Theory and Practice" Prentice−Hall 1961

(79) 青木，藤原，江川 "半導体記憶装置における定期保守間隔と信頼性の関係" 昭和49年度電子通信学会 全国大会 №1815, 1974年

(80) W.C.Carter, K.A.Duke, D.C.Jessep, Jr. "Lookaside Techniques for Minimum Circuit Memory Translators" IEEE Trans. Comput. vol. C−22, №3, p283−p289, March, 1973

(81) H.Mitarai "Design of a Parallel Encoder/Decoder for the Hamming Code Using ROM" 1'st USA−JAPAN Comput. Conf. p531−p537 1972

(82) K.N.Levitt, W.H.Kautz" Cellular Arrays for the Parallel Implementation of Binary Error−Correcting Codes" IEEE Trans. Information Theory, vol.IT−15, №5, p597−p607, Sept. 1969

(83) W.C.Carter, A.B.Wadia, D.C.Jessep. "Implementation of Checkable Acyclic Automata by Morphic Boolean Functions" Polytechnic Inst. Brooklyn. Microwave Inst. Symp. XXI Comput. Automata, p465−p482, April, 1971

(84) W.C.Carter , P.R.Schneider "Design of Dynamically Checked Computers" Proc. IFIPS Conf. Edinburgh, Scotland, p878−p883, Aug. 1968

(85) D.A.Anderson, G.Metze "Design of Totally Self−Checking Check Circuit for m−out−of−n Codes" IEEE Trans. Comput. vol.C−22, p263−p269, March. 1973

(86) W.C.Carter, K.A.Duke, D.C.Jessep "A Simple Self−Testing Decoder Checking Circuit" IEEE Trans. Comput. p1413−p1414, Nov. 1971

(87) J.F.Wakerly "Partially Self−Checking Circuit and Their Use in Performing Logical Operations" IEEE Trans. Comput. vol.C−23,

№7, p658－p666, July, 1974

(88) W.C.Carter, A.B.Wadia, W.G.Bouricius, D.C.Jessep, E.P.Hsieh, C.J.Tan "Cost Effectiveness of Self Checking Computer Design" 7'th Int. Symp. on Fault-Tolerant Comput.(FTCS－7), p117－p123, June, 1977

(89) W.C.Carter, D.C.Jessep, A.B.Wadia "Error Free Decoding for Failure-Tolerant Memories" IEEE Int. Comput. Group Conf. p229－p239, June, 1970

(90) W.C.Carter, K.A.Duke, P.R.Schneider "Self-Checking Error Checker for Two-Rail Coded Data" U.S.Patent, №3559167, Jan.1971

(91) R.Schürba "Chip Cuts Parts Count in Error Correction Networks" Electronics, p130－p133, Nov. 9, 1978

(92) MOTROLA SEMICONDUCTOR PRODUCTS INC. "Volume 4,MECL Integrated Circuits Series A, Semiconductor Data Library, " (MC10163, MC10193)

(93) 高橋，萩原，伊東，松広，中島 "1200ゲート サブナノ秒のマスタスライス形LSI" 電子通信学会 半導体トランジスタ研究会 SSD79－49，1979年10月

(94) 高橋，伊東，小林 "2バイトスライス誤り訂正回路LSI" 昭和55年度 電子通信学会 総合全国大会 №1536，1980年

(95) G.G.Langdon, Jr., C.K.Tang, "Concurrent Error Detection for Group Look-ahead Binary Adders" IBM J.Res. Develop. p563－p573, Sept. 1970

(96) M.Y.Hsiao, F.F.Sellers, Jr. "The Carry-Dependent Sum Adder" IEEE Trans. EC, June 1963

(97) 田中，三好，石井，幸野 "装置ないしモジュールのレベルで2重化を図ったコンピュータ" 日経エレクトロニクス，№156，1977年3月

(98) J.F.Wakerly "Checked Binary Addition with Checksum Codes" J. of Design Automation and Fault-Tolerant Conputing, vol.1, №1, p18－p27, Oct. 1976

(99) E.Kolankowsky, A.K.Pattin "Memory with Error Correction for Partial Store Operation" U.S. Patent, №3573728, Apr. 1971

(100) 岡部 "使いやすくなる小容量システム用RAM(バイト構成RAM，疑似スタチックRAMが登場)" 日経エレクトロニクス，p90－p106，1979.5.14号

(101) E.L.Wall "Applying the Hamming Code to Microprocessor-based Systems" Electronics, Nov.22. 1979

(102) A.Heimlich, J.Korelitz "Memory Finds and Fixes Errors to Raise Reliability of Microcomputer" Electronics, Jan.3, 1980

(103) 石川，松澤 "アドレス間ハミング距離に着目したメモリの試験法" 電子通信学会 電子計算機研究会，EC79-44，1979年4月

(104) 小倉，二階堂，宮原 "大規模連想メモリLSI" 電子通信学会，半導体トランジスタ研究会，SSD80-56，1980年10月

(105) 市川，坂村，諸隅，相磯 "連想プロセッサARES" 電子通信学会 論文誌(D)，vol.J61-D，10，p743-p750，1978年10月

(106) 伊藤，田中，松本，立花 "超大容量記憶装置用LSI 化制御装置の実用化" 通研実報 29，11，p1797-p1805， 1980年

(107) 武藤，池田 "フォールトトレラントゲートの提案" 情報処理 vol.21，№5，p445-p454 1980年5月