

マルチレベルセル NAND フラッシュメモリに適した
誤り訂正符号

2017年1月

千葉大学大学院融合科学研究所

情報科学専攻 知能情報コース

小滝 翔平

(千葉大学審査学位論文)

マルチレベルセル NAND フラッシュメモリに適した
誤り訂正符号

2017年1月

千葉大学大学院融合科学研究所

情報科学専攻 知能情報コース

小滝 翔平

概要

不揮発性メモリの一種であり、データストレージとして広く用いられている NAND フラッシュメモリにおいては、セル間の干渉やセルに格納した電荷の流出など、さまざまな物理的要因による誤りの発生が避けられない。このため、信頼性の確保および長寿命化を目的として、誤り訂正符号の利用が必須となっている。微細化やマルチレベルセル (Multilevel Cell: MLC) の導入はメモリの低コスト化を実現する一方で、ビット誤り率の増大を引き起している。これに伴う誤り訂正のための計算量や検査ビットの増大が、メモリの動作速度やコストの観点から問題となっており、効率の良い誤り訂正符号が求められている。本論文では、MLC の NAND フラッシュメモリを対象とした誤り訂正符号について議論する。誤り訂正符号において、システムに発生する誤りの特徴を反映することは、信頼性の向上を効率よく実現するための典型的な手段である。MLC の NAND フラッシュメモリに発生する誤りは、近傍の閾値電圧レベルへ向かうものほど発生しやすいことを特徴としている。本論文ではこの特徴を考慮し、特に閾値電圧が隣接したレベルを越えて、2 レベル以上の変化をするような場合を考慮した、誤り訂正の手法を提案する。本論文では 3 つの手法を提案する。第 1 の手法は単一 Limited-Magnitude(LM) 誤り訂正符号に分類される。LM 誤り訂正符号は、データを整数値として表したとき、値の変化が一定の範囲内である誤りのみを訂正する。従来の LM 誤り訂正符号と比較し、検査行列の構成法を拡張しており、特定のパラメータで符号長の改善が可能である。また、従来の単一 LM 誤り訂正符号は、通常アルファベットの大きさが 2 のべき乗ではなく、一般的な MLC にそのままでは適用できない。この問題に対処するための符号化法の提案も行う。第 2 の手法は、マルチページプログラミングと併用可能な誤り訂正符号である。マルチページプログラミングは書き込みを高速化する手法であるが、LM 誤り訂正符号を含む従来の多元の符号は、これと併用できない。提案手法は符号構成を変更をせず、復号時にのみ誤りの特徴を反映し、誤り率を改善する。復号に要する時間は増加するが、誤り率が許容不可となる書き込み／消去の回数までは従来手法を利用し、それ以降のみ提案手法に切りかえることが可能である。第 3 の手法は、LM 誤り訂正符号を、ページ毎のビット誤り訂正符号のみを用いて構成するものである。この手法では、一部のページに対し検査ビットが不要であり、空いたビットを活用して追加の誤り訂正を行うことで、誤り率を改善する。提案符号はマルチページプログラミングが可能な従来の符号と比較し、誤り率のパフォーマンスに優れ、加えて構成法が部分的に一致していることを特徴とする。このため、信頼性に問題が出る高い P/E に達した時点での符号の切り替えを行えば、ハードウェアコストを抑えながら、寿命の延長ができる。

目 次

1	はじめに	4
2	NAND フラッシュメモリ	10
2.1	半導体メモリとしての役割	10
2.2	メモリセルの構造と動作	12
2.3	メモリセルの配列	12
2.4	ISPP とマルチページプログラミング	14
2.5	誤りの要因	16
2.5.1	セル間の干渉	16
2.5.2	データ保持	17
2.5.3	データ読み出し	17
2.5.4	ランダムテレグラフノイズ	18
2.6	誤りの特徴	18
2.7	信頼性向上と長寿命化	18
2.7.1	不良ブロック管理	19
2.7.2	ウェアアレベリング	19
2.7.3	ガベージコレクション	19
3	誤り訂正符号	23
3.1	数学に関する表記と諸定義	23
3.2	誤り訂正の基礎	24
3.2.1	情報システムのモデル	24
3.2.2	誤りのモデル	24
3.2.3	ハミング距離と誤り訂正	25
3.2.4	ハミングの限界式	27
3.3	線形符号	28
3.4	巡回符号	30
3.5	BCH 符号	30
3.6	Low-Density Parity-Check 符号	30
3.7	多元非対称誤り訂正符号	31
3.8	Limited-Magnitude 誤り訂正符号	31
3.8.1	諸定義	31
3.8.2	完全・準完全な集合 $B[l](q)$ と $B[\pm l](q)$	32
3.8.3	多重 LM 誤り訂正符号	35
4	NAND フラッシュメモリにおける誤り訂正符号	37

5 提案手法 1：単一対称 LM 誤り訂正符号	40
5.1 概要	40
5.2 定義と定理	40
5.3 提案手法	41
5.3.1 符号構成	41
5.3.2 誤り検出機能の付加	47
5.3.3 復号法	48
5.4 評価	49
5.4.1 情報長と検査長	49
5.4.2 ビット誤り率と符号長	52
5.4.3 復号誤り率	52
5.5 この章のまとめ	52
6 提案手法 2：マルチページプログラミングと近傍値への誤りを考慮した誤り訂正	54
6.1 概要	54
6.2 グレイ符号と誤り訂正	54
6.3 提案手法	55
6.4 ビット誤り率の算出	57
6.5 評価	63
6.5.1 読み出し時のレイテンシとスループット	63
6.5.2 ビット誤り率	66
6.5.3 寿命	67
6.6 この章のまとめ	67
7 提案手法 3：2元の符号を用いた双方向 LM 誤り訂正符号	71
7.1 概要	71
7.2 距離と誤り訂正機能	71
7.3 提案手法	75
7.3.1 写像	76
7.3.2 符号構成法	79
7.3.3 復号法	80
7.4 追加の誤り訂正	81
7.5 実装	82
7.5.1 符号効率	82
7.5.2 復号回路	83
7.5.3 書き込みにおけるレイテンシとスループット	83
7.5.4 読み出しにおけるレイテンシとスループット	84
7.5.5 追加訂正によるページサイズ変更の影響	86

7.6	誤り率のシミュレーション	86
7.6.1	誤りモデル	86
7.6.2	結果	87
7.7	この章のまとめ	92
8	おわりに	93

1 はじめに

NAND フラッシュメモリは、電源を供給しなくてもデータを記憶し続けることができる、不揮発性のメモリの一種である。メモリカードや USB メモリデバイス、PC、スマートフォンなどのコンシューマ向け製品や、企業向けの高性能な計算機、大規模なデータセンターなどにおいて、データストレージとして利用されている [1, 2, 3, 4, 5, 6, 7, 8]。アプリケーションの中でも、とりわけソリッドステートドライブ (Solid State Drive: SSD) は、ハードディスクドライブ (Hard Disk Drive: HDD) の代替としての利用が広がっており、注目を集めている。これは HDD と比較して、サイズが小さく、低消費電力、読み出し／書き込みが高速、耐衝撃性に優れる [2, 4, 8, 9, 10, 11, 12, 13] といった利点に加えて、HDD と比べると高価ではあるが、継続的な低成本化により安価に入手できるようになったためである。NAND フラッシュメモリの急速な低成本化は、製造プロセスの微細化と、ひとつのメモリセルに 2 ビット以上のデータを格納する、マルチレベルセル (Multilevel Cell: MLC) [6, 14, 15] による記憶密度の向上によって実現してきた [3, 8]。15nm のプロセスによるメモリが製品化される現在、リソグラフィーに要するコストの増大や、メモリセル間の干渉、セルに格納する電子数の減少などによる信頼性への影響により、微細化は近い将来に限界を迎えると言われている。しかしながら、近年では 3 次元積層技術によって低成本化が継続されている [16, 17]。

NAND フラッシュメモリは物理的特性から、誤りの発生が避けられない。誤りの主要因としては、セル間の干渉、データ保持の際の電荷の流出や変動、データ読み出しの際の配線ストレス、ランダムテレグラフノイズ (Random Telegraph Noise: RTN) が挙げられる。NAND フラッシュメモリのメモリセルは浮遊ゲートをもつ MOS (Metal-Oxide Semiconductor) トランジスタであり、浮遊ゲートに電荷を格納することで閾値電圧を変化させ、データを記憶する [6]。メモリセルに格納される電荷の数は微細化により減少し、少量の電荷の変動が閾値電圧に大きく影響してしまう。また、メモリセル間の干渉は寄生容量結合によるものであり、セル間の距離が小さいほど、影響が深刻なものとなる。このように、微細化は低成本化を実現する一方で、これら誤りの発生率を増大させてしまう。一方、MLC もまた誤りの発生率を増大させる要因となる。MLC は多段の閾値電圧レベルを用いて多ビットのデータを格納しており、多くのビットを格納するほど、各電圧レベルの間隔が小さくなってしまう。これにより、各レベルにおいて許容される閾値電圧の変動幅が小さくなり、誤りを生じやすくなるためである [1, 4, 5, 9, 10, 18, 19]。

NAND フラッシュメモリの高信頼化技術としては、ウェアアベリング、不良ブロック管理、ガベージコレクション、誤り訂正符号が挙げられる [1, 5]。特に検査ビットを付加することで、誤りを検出、訂正する技術である誤り訂正符号は、他の技術と異なり、例えば電荷流出のような、永続的な影響でない誤りに対しても有効である [5]。微細化により増大を続ける、訂正前の誤り率 (Raw Bit Error Rate: RBER) は

10^{-3} を超える一方で [7], 通常ストレージシステムにおいては, 訂正不可な誤り率 (Uncorrectable Bit Error Rate: UBER) を $10^{-13} \sim 10^{-16}$ 以下に抑えることが求められている [8, 20]. このような厳しい誤り率の要件を満たすために, 多ビットの誤りを訂正する符号—BCH 符号 [10, 13, 14, 19] や LDPC(Low-Density Parity-Check) 符号 [3, 4, 11]—が用いられている. 2010 年代前半には 24 ビット以上の強力な誤り訂正を持つ BCH 符号を用いた製品も出現しており [10, 14], 近年ではより強力な訂正機能が実装されていると思われる. 強力な BCH 符号は多くの検査ビットを必要とし, コスト面で有効とは言えない. LDPC 符号は BCH 符号と異なり, 符号効率の理論限界である, シャノン限界に迫る優れた誤り訂正が可能である. このような優れた特性は通常, 復号アルゴリズムに軟判定の（離散値ではない, または十分に細かく離散化された）演算を用いることで実現される [3, 11]. このような復号法は計算量が大きく, 加えて精密な閾値電圧の情報を得るためにセンシングを多く行う必要があり, 速度の低下を引き起こすといった問題を有する [3]. また, これらの符号は訂正機能が強力であるほど, 復号に要する計算量は増大し, 消費電力の増大や, 許容できないほどの復号時間を要するといった問題が生じる [5, 9, 12]. したがって, 単純な符号の誤り訂正機能の増大に頼らず信頼性を確保する, 効率の良い誤り訂正が求められている. これらの高信頼化技術は信頼性に加えて, メモリの長寿命化を実現する技術でもある. NAND フラッシュメモリはデータを書き換えることができず, データを更新する際には, 一度消去を行う必要がある [21]. しかし, 書き込み／消去サイクル (Program/Erase サイクル: P/E サイクル) の増加はメモリセルの劣化の原因となり, 誤り率が増加してしまう. P/E サイクルがある程度まで増加すると, 誤り訂正符号を用いても十分に低い誤り率を確保できなくなり, メモリの寿命となる. 優れた誤り訂正符号を用いれば, 許容できる P/E サイクルも増加する. これにより, 長寿命化が実現できる.

効率の良い誤り訂正を実現するための手段として, システムに発生する誤りの傾向を考慮した誤り訂正が挙げられる（多くの例が [22] に示されているので, 参照されたい). そこで本論文では, MLC の NAND フラッシュメモリに発生する誤りの特徴を, 符号構成や復号法に反映することで, 検査ビットの削減, 誤り率の改善, 長寿命化を実現する. 言い換えると, 発生率が高い誤りに訂正機能を特化させることで, 効率化を図る. MLC の NAND フラッシュメモリにおける, 誤りの典型的な特徴として, 「誤りによる閾値電圧の変動は, 近傍のレベルへ向かうものであるほど, 発生しやすい」ことが挙げられる. データ保持による誤りを例に説明する. データ保持に際する電荷の流出は, 閾値電圧をランダムに変化させるものではない. P/E サイクルによるセルの劣化と, 電荷保持時間等に依存して, 徐々に閾値電圧を低下させる. このため, 近傍の電圧レベルへ向かう誤りは容易に発生する一方, 遠くのレベルへ向かう誤りの発生は, 極端に劣化したセルにおいて, 長期間のデータ保持を行った場合に限られる（もしこのような誤りが多く発生するならば, 誤りの総数としては, 誤り訂正符号で対応できない程度発生していると考えられる). Cai らに

よる 30~40nm のメモリを用いた実験 [8] では、最も誤り率の高いデータ保持によるセルの誤り全体のうち、およそ 95% が 1 レベルの変化であり、5% が 2 レベルの変化であることが示されている。また、他の誤りの要因についても同様に、大部分は近傍のレベルへ向かう誤りである。文献 [5] においても、2 レベル以上の誤りに相当する、閾値電圧の変化を要因とする MBU(Multiple Bit Upset) の発生が指摘されている。これらを考慮し、本論文では誤りは近傍のレベルに集中し、かつ 2 レベル以上の変化についても発生し得る、という誤りモデルに従い、手法を提案する。

NAND フラッシュメモリの誤り訂正符号として現在利用されている BCH 符号や LDPC 符号は、2 元の誤りモデルに基づく符号であり、上記の誤りの特徴を考慮していない。そこで本論文では、NAND フラッシュメモリの誤りの特徴を反映した、3 つの手法を提案する（以下、これらの 3 つの手法を提案手法 1、提案手法 2、提案手法 3 とする）。提案手法 1 および提案手法 3 は、Limited-Magnitude(LM) 誤り訂正符号の一種である。LM 誤りは、符号アルファベットを整数の集合とした場合に、誤りによる変化が一定の範囲に収まるものを指す。図 1 は、誤りの分類を示している。誤りは、符号アルファベットの大きさによって、2 元の誤りと多元の誤りに分類される。2 元の誤りはさらに対称誤り、非対称誤り、単方向誤りに分類される。対称誤りでは $0 \rightarrow 1$ の変化と $1 \rightarrow 0$ の変化が等確率で発生する。非対称誤りは $0 \rightarrow 1$ の変化と $1 \rightarrow 0$ の変化の発生率が異なるものだが、特に一方の発生率が 0 である誤りを指す場合が多い。単方向誤りでは $0 \rightarrow 1$ の変化と $1 \rightarrow 0$ の変化が双方とも発生するが、多重に発生した場合に、それらが混在しないものを指す。一方、多元の誤りは対称誤りと非対称誤りに分類される。対称誤りは、誤りの発生率が送信シンボルと受信シンボルの組み合わせに依存しないものを指す。非対称誤りは、送信シンボルと受信シンボルの組み合わせによって、誤りの発生率が異なるものを指す。LM 誤りの場合、誤りによる変化がある一定の範囲に収まる場合は発生率が 0 ではなく、収まらない場合は発生率が 0 という異なった値をとるため、多元非対称誤りの一種と考えられる。また、符号アルファベットの大きさを便宜上 2 に設定すると、2 元の誤りモデルと一致するため、その一般化と見なせる。ただし、具体的な誤り訂正符号の構成法については、2 元と多元の場合で全く異なっている。

LM 誤り訂正符号は、单一誤り訂正符号と、多重誤り訂正符号、全誤り訂正符号に分類できる。さらに单一 LM 誤りは、値の増加と減少の、事前に定めたどちらか一方の誤りのみを扱う非対称 LM 誤り、これら両方の誤りを扱い、値の増減の絶対値が等しい対称 LM 誤り、および値の増減の絶対値が異なる不均衡 LM 誤りに分類できる。单一非対称 LM 誤り訂正符号 [23, 24]、および单一対称 LM 誤り訂正符号 [25, 26] は、Kløve らによって提案されている。单一不均衡 LM 誤り訂正符号は Schwertz ら [27] によって提案され、Yari ら [28] によって改良されている。多重 LM 誤りは、値の増加と減少の、事前に定めた一方のみを扱う非対称誤り、値の増加と減少が混在しない場合を扱う单方向誤り、増加と減少が混在し得る双方向誤り（多重の不均衡 LM 誤りに相当）に分類できる。多重非対称 LM 誤り訂正符号は、Kløve ら [24] や

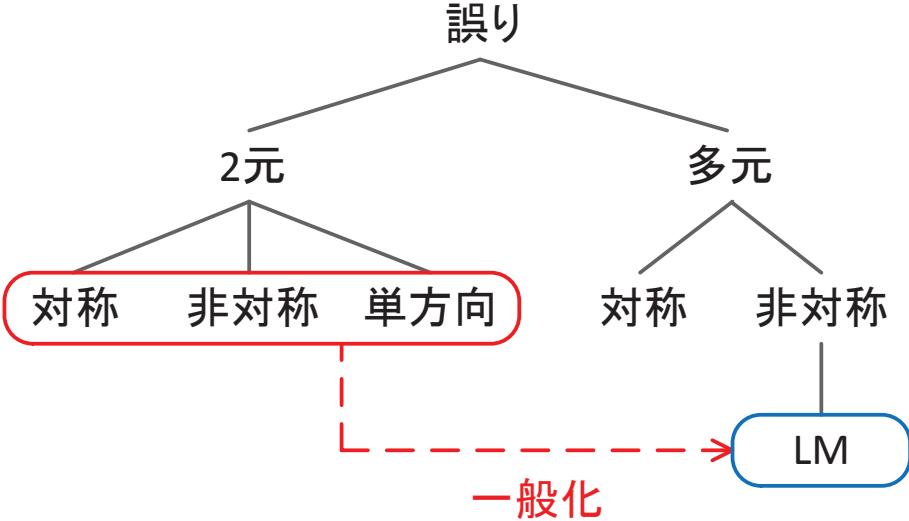


図 1: 誤りの分類.

Cassuto ら [29] によって提案されている. 非対称 LM 誤り訂正符号は单一誤り訂正符号も含め, NAND フラッシュメモリのオーバープログラムによる増加方向の誤りを対象としており, 不正確な書き込みを許容することで, 高速な書き込みを実現することを目的としている. しかし, データ保持の誤りのような減少方向の誤りを, 同時に訂正することができない. 単方向 LM 誤り訂正符号は, 著者らによって提案されている [30]. この符号は, NAND フラッシュメモリのセル干渉は増加の誤り, 電荷保持は減少の誤りのみを発生させ, ある時点では強く作用している一方のみが観測されるため, 増加と減少の混在するような誤りの発生率は低いという考えに基づいている. 多重双方向 LM 誤り訂正符号は Jeon らが提案しており [31], これはパラメータ次第では, 対称 LM 誤り訂正符号や非対称 LM 誤り訂正符号になる(ただし, 検査部において LM 誤り訂正機能が保障されておらず, 厳密には双方向 LM 誤り訂正符号でない). 多重の LM 誤り訂正符号は, パラメータによっては検査部を効率的に利用できない点が問題となる. 一方で, [23, 24, 25, 26, 27, 28] のような整数剰余環上の検査行列を用いた单一誤り訂正符号は, 検査部を効率よく利用できる. 全誤り訂正については, 非対称 LM 誤り訂正符号が, Ahlswede らによって非組織符号 [32], Elarief らによって組織符号 [33] として提案されている. また [33] では対称 LM 誤り訂正組織符号も提案されている. 全誤り訂正符号は必要な検査ビットが多く, 誤り訂正などに利用できる予備のセル数が制限される, NAND フラッシュメモリに向かない. LM 誤り訂正符号の分類を表 1 に示す. 提案手法 1 は単一 LM 誤り訂正符号である [34]. 同じ機能を持つ符号は既に [25, 26] が提案されているが, 提案手法 1 においては新たに,

表 1: LM 誤り訂正符号の分類.

	单一誤り訂正	多重誤り訂正	全誤り訂正
非対称	Kløve[23, 24] (提案手法 1)	Kløve[24] Cassuto[29]	Ahlswede[32] Elarief[33]
単方向	(該当する誤りなし)	Kotaki[30]	Ahlswede[32]
対称	Kløve[25, 26] 提案手法 1	Jeon[31]	Elarief[33]
不均衡 (双方向)	Schwertz[27] Yari[28] (提案手法 1)	Jeon[31] 提案手法 3	(未提案)

1. 符号の構成法が知られていないパラメータについて, 符号を得るためのアルゴリズムを提案
2. 従来手法を拡張した検査行列により, パラメータ次第で符号長の改善
3. 2のべき乗のアルファベットを持つシステムへ適用するための符号化法の提案
4. 大きな変量の誤りに対する検出機能の付加が可能

といった提案を行っている。なお, もともとは対称LM誤り訂正符号として提案されているが, 従来のLM誤り訂正符号の構成法を応用すれば, 1.~4. の提案は非対称, 双方向のLM誤りにも対応できる。

優れた符号長を持ちながらも, これらLM誤り訂正符号を含む, 各メモリセルに記憶される複数ビットを1つのシンボルとする多元の誤り訂正符号は, 書き込み速度を高めるセルアーキテクチャと書き込み方式である, マルチページプログラミング [35] と併用できない点が問題となる。NAND フラッシュメモリの独立した読み書きの単位はページと呼ばれる。マルチページプログラミングでは, 1つのメモリセルに属する各々のビットが, 異なるページに属している。したがって1つのページだけでは, 多元の符号化に必要な, 対象のセルに属する複数ビットの情報を得ることができない。多元の誤り訂正を行いながら, マルチページプログラミングを考慮した手法は, BCH 符号と TCM(Trellis Coded Modulation) の連接符号の文献 [12] において言及されている。ただしこの手法は2ビットセルの場合, 4つの閾値電圧レベルに代わり, TCM の利用における制約から5つの閾値電圧レベルを必要とし, これは2ビットセルのためのプログラミングやセンシングのアーキテクチャでは実現できない。そこで提案手法2として, 従来のマルチページプログラミングと同じ符号構成を用いながら, 復号において誤りの特徴を考慮することによって, 誤り率の改善する手法を提案する。この手法では, 誤りの検出はセルに属する各々のページに用いた2元の符号によって行うが, 誤りの訂正については, 格納された複数ページのデータを整数値として, 多元の操作をすることで実現する。

提案手法2のように、マルチページプログラミングに利用できる符号構成であっても、誤りの特徴を反映することができる。しかしながら、提案手法1のようなLM誤り訂正符号は、より的確に誤りの特徴を捉えることができ、信頼性に優れる。そこで提案手法3として、P/Eサイクルが高くなった時点で、強力な誤り訂正符号に切り替える手法を提案する。単純な切り替えであれば、符号化および復号ハードウェアを別途用意する必要があるが、提案符号はマルチページプログラミングの構成と部分的に一致しており、追加のハードウェアを多く必要としない。具体的な実現法としては、[31]において提案されている双方向LM誤り訂正符号の構成法に基づき、必要な誤り訂正機能を保ちながら検査ビットを削減し、削減したビットを用いて追加の誤り訂正を実行することで、誤り率を改善する。

本論文は全8章で構成されている。第2章はNANDフラッシュメモリの概要を説明する。メモリの構造や動作、誤りの発生要因や、マルチページプログラミングについてまとめている。第3章では誤り訂正符号について説明する。誤り訂正の基礎知識や、LM誤り訂正符号を中心とした、本論文に関連する実用的な符号について扱う。第4章ではNANDフラッシュメモリの誤り訂正に関する先行研究を紹介する。第5章、第6章、第7章はそれぞれ、提案手法1、提案手法2、提案手法3について扱う。最後に、第8章でまとめと今後のNANDフラッシュメモリの誤り訂正符号に関する展望を述べる。

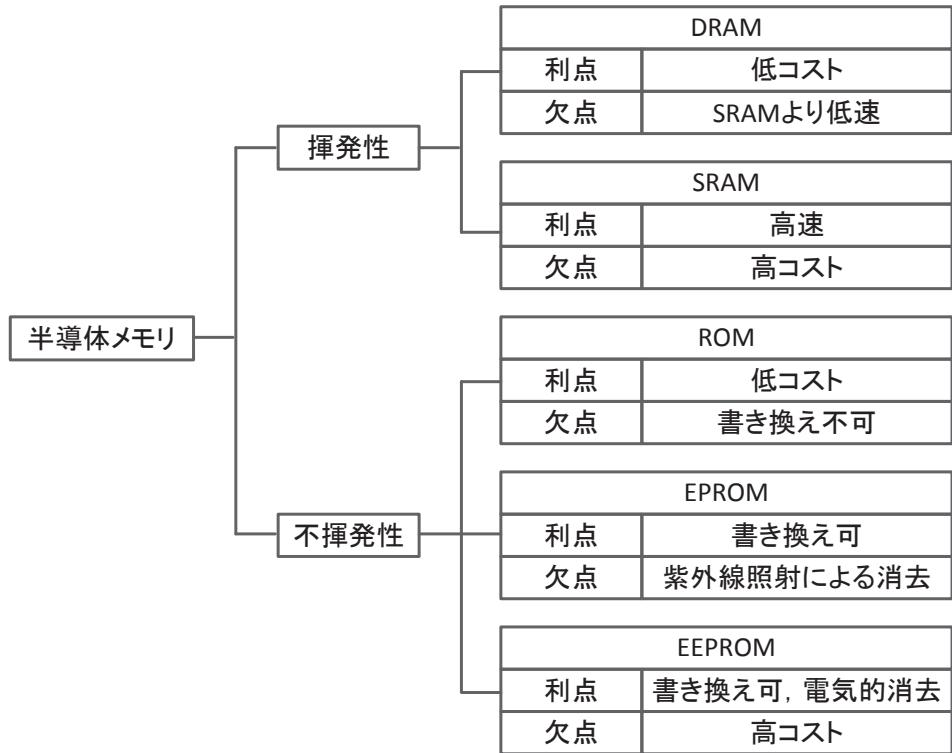


図 2: 半導体メモリの分類.

2 NAND フラッシュメモリ

2.1 半導体メモリとしての役割

現代の情報プロセッサにとって欠かすことのできない要素である半導体メモリは、大きく揮発性メモリと不揮発性メモリに分類される [36]. 挥発性メモリである RAM(Random Access Memory)―DRAM(Dynamic RAM) や SRAM(Static RAM) 一は不揮発性メモリと比較して高速であるため、主記憶やキャッシュメモリなどに用いられているが、電源を供給し続けないと、データを保持することができない。一方、不揮発性メモリである ROM(Read Only Memory), EPROM(Erasable Programmable ROM), EEPROM(Electrically Erasable Programmable ROM) などは、データを保持するために電源を必要としないことが特徴である。半導体メモリの分類を図 2 に示す。フラッシュメモリは EEPROM に属する [36].

実用されているフラッシュメモリには構造の違いにより、NAND 型と NOR 型が存在する。NOR 型は記憶密度は低いが、バイト単位の読み出しが可能である。また、書き込みは 16~64B を単位とし [37]、消去の単位は 64KB のセクタである [36]。NAND 型はストリングと呼ばれる単位でセルが直結されており、NOR 型と比較して記憶密度が高い [36]。書き込みが高速であるが、読み書きは 512B から 16KB のペー

表 2: NAND フラッシュメモリの特徴 [40].

テクノロジ	3-Metal 19nm CMOS
容量	64 Gb (2 bit/cell)
ダイサイズ	112.8 mm ²
ページサイズ	16 KB
プログラム速度	15 MB/s
リードレイテンシ	50 μ s
電源供給	2.7 V ~ 3.6 V

ジが単位となる [5, 12]. また, 消去は複数ページをまとめたブロック単位で行われる [8]. このため組み込みシステムでは, 大容量のデータを蓄積するには NAND 型, 小さいデータサイズで読み出す必要があるプログラムコードの格納には, NOR 型が用いられている [38]. 繙続的な低コスト化を背景に NAND 型の市場は成長を続け, 2014 年には DRAM に匹敵するに至る一方で, NOR 型の市場は 2000 年をピークに縮小している [39]. このような背景から, NAND フラッシュメモリの発展が社会に与える影響は大きく, したがって本論文は NAND フラッシュメモリをテーマとしている. 表 2 に, 2013 年現在の NAND フラッシュメモリの特徴を示す [40]. NAND フラッシュメモリのアプリケーションにおいて注目される SSD は, HDD と比較し高コストであるが, 読み書きが高速であることを特徴としている. このため高性能な計算機で利用したり, コンパクトで耐衝撃性に優れるため, タブレット端末やノート型 PC に利用される. コンシューマ向け PC においては HDD と SDD を併用し, プログラムには SSD, データの保存には HDD と使い分ける利用方法も広がっている. 一方, 書き換え/消去の回数 (P/E サイクル) が限られているのが, NAND フラッシュメモリの欠点であり, ひとつのセルに 1 ビットを格納する SLC(Single Level Cell) では~10,000 回程度, MLC のメモリでは~3,000 回程度で寿命となる. また, データを保持できる期間が 5~10 年程度と限定されていることも, HDD と異なる [8].

NAND フラッシュメモリは微細化の限界から, 今後これまでと同様に大容量化を実現していくのは困難と考えられる一方で, 将来的には DRAM の代替としてメインメモリへの利用や, ストレージより CPU に近い立場のメモリとして, DRAM と併用することが考えられている [39]. これは SLC の NAND フラッシュメモリと比較して, DRAM が約 10^3 倍高速な読み書きのレイテンシと, 書き換え回数に制限がないことを強みとする一方で, NAND フラッシュメモリはコストがおよそ 1/10 と低く, リフレッシュ動作が必要な DRAM とは異なり, データ保持に電力を消費しないためである [39, 41]. 書き換え回数や読み書きの速度等において, NAND フラッシュメモリより優れる次世代の不揮発性メモリとして, PCM(Phase Change Memory), STT-RAM(Spin-Torque Transfer RAM), ReRAM(Resistive RAM) などが研究されている. これらもまた DRAM の代用や併用が期待されているが, 一部は製品化されているものの, NAND フラッシュメモリのように, 安価で大容量のメモリを量産

できる段階ではない。またこれら次世代メモリのうちのどれが、生産可能性、コスト、性能、信頼性などの面において将来性を持つかは定かでないため、既に特徴がよく研究されたNANDフラッシュメモリの役割を置き換えるに至っていないのが現状である[41]。

2.2 メモリセルの構造と動作

フラッシュメモリのメモリセルは浮遊ゲートを持つMOSトランジスタである[6, 42]。データの記憶は浮遊ゲートに一定量の電子を注入することで実現する。図3に、メモリセルの構造を示す。浮遊ゲートは、基盤と制御ゲートから酸化膜により絶縁されており、電源を遮断しても注入した電荷は保持される。これにより、不揮発性の記憶が可能となっている。データの書き込みは制御ゲートに高電圧を加え、Fowler-Nordheim(FN)トンネル現象によって、基盤から浮遊ゲートに電荷を注入することで実現される。FNトンネル現象による電流密度 J_{FN} は、以下のようにモデル化される[36]：

$$J_{FN} = \alpha_{FN} E_{ox}^2 e^{-\beta_{FN}/E_{ox}}. \quad (1)$$

ここで α_{FN}, β_{FN} は定数であり、 E_{ox} は導通方向の電場の強さである。 E_{ox} は制御ゲート電圧に比例する。閾値電圧 V_{th} と浮遊ゲートに格納した電荷量 Q_{FG} には、以下の関係がある[36]：

$$V_{th} = V_{thi} + (-Q_{FG})/C_{pp}.$$

ここで、 V_{thi}, C_{pp} はプロセス依存の定数である。電子は負電荷であるから、格納する電子の増加にしたがって、閾値電圧も増加する。データの消去は制御ゲートに大きい負の電圧を加えることにより、浮遊ゲートから基盤へ電荷を取り除くことで行う[7]。制御ゲートに一定以上の電圧を加えると、ドレインーソース間に電流が流れる。電流が流れるような制御ゲート電圧の最小値を、閾値電圧と呼ぶ。データの読み出しへは、固定値である読み出し電圧を制御ゲート加え、ドレインーソース間の電流を観測することで行う。メモリセルに b ビットを格納するには、 2^b の閾値電圧レベルを用いる[6]。 $b \geq 2$ の場合をMLCと呼び、 $b = 1$ の場合を区別してSLCと呼ぶ[8]。同じレベルに書き込んだ場合でも、閾値電圧はセルごとに異なり、広がりを持つ分布をなす。これを閾値電圧分布と呼ぶ。図4は、4つの閾値電圧レベルを用いて、单一のセルに2ビットを格納する場合の、閾値電圧分布を表している。

2.3 メモリセルの配列

メモリセルの配列は、ページ<ワード<ブロックという階層構造を成している[10, 12]。図5に配列構造を示す。ページは読み書きの単位であり、512B~16KBのユーザーデータと[5, 12]、その3~5%程度のサイズを持つ制御データから成る[4, 13]。誤

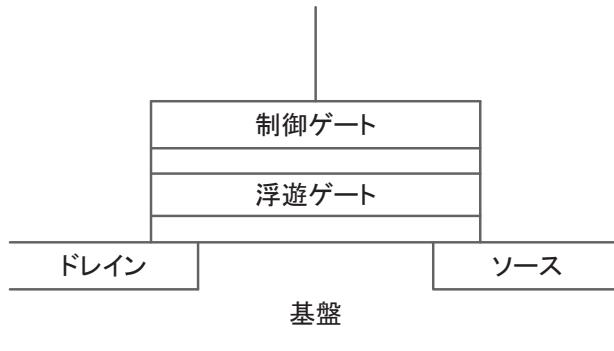


図 3: メモリセルの構造

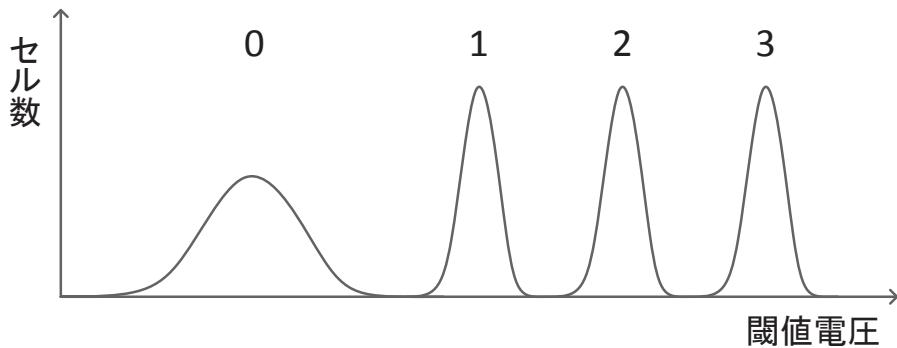


図 4: 2ビットセルの閾値電圧分布

り訂正のための検査ビットは制御データとして格納される。ワードはワード線を共有するメモリセル群であり、MLCの場合複数のページから成る。後述のマルチページプログラミングを用いた b ビットセルの場合、 $2b$ ページがひとつのワード線に含まれる。ブロックはデータ消去の単位であり、ストリングと呼ばれるビット線方向のセル群に対して配線される、16~64のワード線から成る。同一ストリングに属するセルはドレイン—ソースが直列に配置されている。これはそれぞれのセルが独立して配線されている、NOR型との違いの一つである[36]。なおデータは書き換えることができず、データを更新する場合一度消去を行う必要がある[21]。全てのブロックはビット線と、書き込みと読み出しにおけるデータフェッチのための、オンチップのページバッファを共有している[12]。

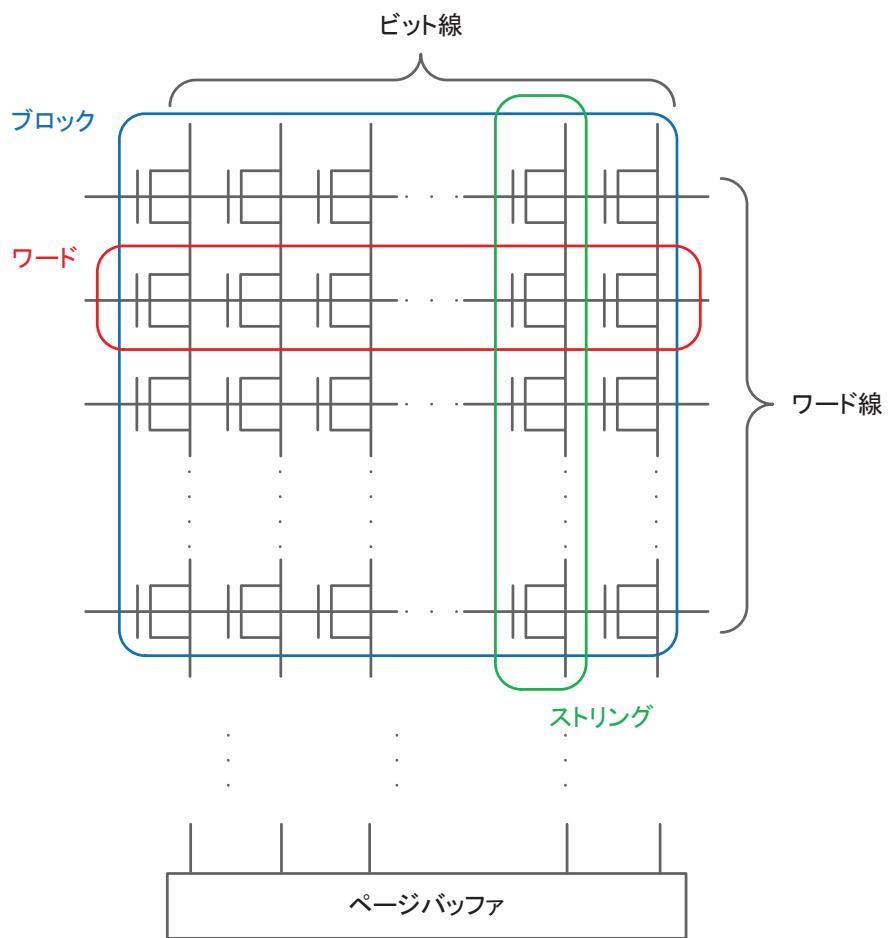


図 5: メモリの配列構造 [12]

2.4 ISPP とマルチページプログラミング

NAND フラッシュメモリの書き込みの単位はページと呼ばれ、同一のページに属するセルは、同一の操作中に書き込まれる。しかしながら、製造時の誤差や、熱などの環境要因、劣化の度合によって、セル毎に書き込み速度のばらつきがある。高いプログラム電圧を用いると、高速な書き込みが可能であるが、オーバープログラムによる誤りが発生しやすく、低い書き込み電圧を用いると、完了までに時間を要する。このため、正確な閾値電圧の操作と、高速な書き込みを両立する方法として、インクリメンタルステッププラスプログラミング (Incremental Step Pulse Programming: ISPP)[43] が利用されている。ISPP では、始めは早く書き込みが完了するセル向けに、比較的小さい書き込み電圧 $V_{pp} = V_{pp0}$ を用いる。セル毎に閾値電圧が十分かどうか確認し、閾値電圧が不足しているセルのみ、書き込み電圧を $V_{pp} + \Delta V_{pp}$ に変更し、再度プログラムを実行する。この操作を、ページ内のすべてのセルの書き込みが完了するまで繰り返す。図 6 に、ISPP による書き込み電圧の時間変化を示す。ISPP による閾値電圧分布は一様分布にモデル化される [44, 45]。また、消去状態の閾値

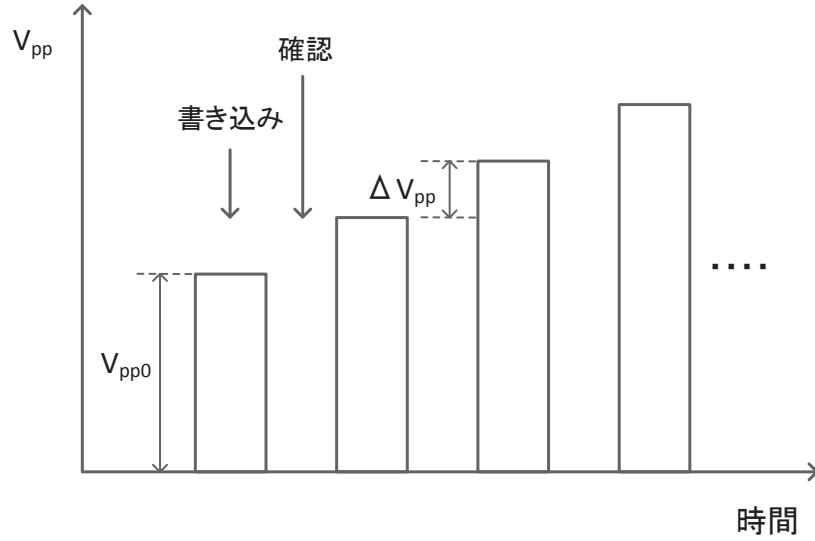


図 6: ISPP と書き込み電圧 [35]

電圧分布はガウス分布となる [44]. すなわち, 第 k レベルの閾値電圧分布を $p_p^{(k)}(x)$, 消去状態の閾値電圧分布を $p_e(x)$ とすると, 以下のように表せる:

$$p_e(x) = \frac{1}{\sigma_e \sqrt{2\pi}} e^{-(x-\mu_e)^2/2\sigma_e^2},$$

$$p_p^{(k)}(x) = \begin{cases} \frac{1}{\Delta V_{pp}} & (V_{th}^{(k)} \leq x \leq V_{th}^{(k)} + \Delta V_{pp}), \\ 0 & (\text{otherwise}). \end{cases}$$

ここで, μ_e, σ_e はそれぞれ $p_e(x)$ の平均, 標準偏差であり, $V_{th}^{(k)}$ は, 第 k レベルに対応する閾値電圧のなす一様分布の左端である. ただし, この理想的な分布はさまざまな誤りの要因によって変動する. ΔV_{pp} が小さいほど幅の小さい閾値電圧分布となるが, プログラム電圧を印加する回数は増大する [35].

マルチページプログラミング [35] は, ひとつのセルに属する複数ビットを, それぞれ別のページに所属させ, 高速な書き込みを実現するアーキテクチャである. 2ビットセル場合の, 従来のプログラミングとマルチページプログラミングの手順, および閾値電圧の変化を, 図 7 および図 8 にそれぞれ示す. 従来の書き込みでは, ISPP における閾値電圧の確認動作を, 電圧レベル 1, 2, 3 それぞれ対し行う必要があった. マルチページプログラミングの場合, 各々のページはセルにおける 1 ビットの情報しか持っていない. このため, 下位のビットが属するページの書き込みでは, 対応する電圧レベルは 0 か 1 のいずれかなので (図 7(b)), 電圧レベル 1 に達したかのみを確認する. 上位のビットが属するページでは, すでに 0 か 1 をとり得る下位のページのプログラムは完了しているため (図 7(c)), 電圧レベル 2 および 3 に達したかを

確認する。これにより、閾値電圧の確認回数が、下位のページでは3回から1回に、上位のページでは2回に削減される。この確認回数の削減により、おなじページサイズで比較すると、従来の2.3倍の速度でプログラムが可能となる。

マルチページプログラミングを併用する誤り訂正符号を設計する際には、以下のことについて注意する必要がある：

- あるページの符号化には、それより下位のページのデータしか用いることができない。例えば、最下位のページの符号化には、そのページの情報部のみしか用いることができない。したがって、複数ページにまたがる情報を利用するような多元の誤り訂正符号は利用できない。
- 復号の際には、複数ページのデータをバッファに蓄えることで、 b ページすべてを訂正の対象にできる。加えて、あるページを書き込む際には、それより下位のページ全ての情報が必要となるため、 b ビットセルのメモリは、もともと b ページのデータを格納するバッファを備えている。読み出しの際には、一度これら b ページ全ての情報を、バッファに取り出すことが可能である[35]。このため、軟判定の復号[3]のようにオンチップで復号を行う場合にも、追加のバッファなしで多元のデータを得ることができる（ただし、復号アルゴリズムが追加のバッファを必要とするとはあり得る）。

2.5 誤りの要因

NAND フラッシュメモリにおいて、閾値電圧が何らかの要因で変化し、他の電圧レベルとの境界を越えて変化した場合、読み出しデータが書き込み時に意図したものと異なってしまう。このとき、データに誤りが発生したことになる。誤りの要因はさまざまだが、誤り率を考えたときに無視できない主要なものとして、セル間の干渉、データ保持、データ読み出し、ランダムテレグラフノイズが挙げられる[7, 44]。ここではそれらの概要を紹介する。

2.5.1 セル間の干渉

あるメモリセルの閾値電圧は、対応する制御ゲート電圧のみでなく、周辺セルの浮遊ゲートと制御ゲートの電位に依存して決まる。これは寄生容量によるものであり、微細化したメモリで特に顕著である[46]。カップリング率 $\gamma^{(k)}$ を、対象のセルと周辺セルの電気容量の総和 C_{tot} 、ある周辺セルと対象のセル間の電気容量 $C^{(k)}$ を用いて、

$$\gamma^{(k)} = \frac{C^{(k)}}{C_{tot}}$$

と定義する。このとき、セル間の干渉による閾値電圧の変量は、周辺セルの閾値電圧の変化 $\Delta V_{th}^{(k)}$ とカップリング率 $\gamma^{(k)}$ を用いて、次のように表せる [3, 46] :

$$F = \sum_k (\Delta V_{th}^{(k)} \cdot \gamma^{(k)}).$$

セルが受ける干渉の程度は、偶数ビット線のセルが、奇数ビット線のセルより先にプログラムされる偶奇ビット線構造、偶奇ビット線のセルが同時に書き込まれる全ビット線構造の、どちらを利用するかによって異なる。偶奇ビット線構造では、偶数ビット線のセルは同一ワード線内の 2 つのセルと、次にプログラムされるワード線内の 3 つのセルの干渉を受ける。奇数ビット線のセルは、次のワード線内の 3 つのセルの干渉のみを受ける。また全ビット線構造では、偶奇によらず次のワード線内の 3 つのセルの干渉のみを受ける [42]。

2.5.2 データ保持

外部から電圧を印加しない場合にも、浮遊ゲートと基盤の間には、格納した電荷によって内部電場が生じる。この電場は以下のようにモデル化される [36] :

$$E_{ox} = \frac{C_{ono}}{C_{ono} + C_{ox}} \cdot \frac{V_{th} - V_{thi}}{T_{ox}}.$$

ここで C_{ono}, C_{ox} は電気容量によって決まる定数であり、 V_{thi}, T_{ox} はプロセス依存の定数である。内部電場 E_{ox} によっても FN トンネル現象が生じ、式 (1) に従い電荷が流出する。この現象を SILC(Stress Induced Leakage Current) と呼ぶ。P/E サイクルが増加すると、酸化膜に電荷が取り込まれる現象により、FN トンネル現象は生じやすくなる [47]。SILC の他に、酸化膜に取り込まれた電荷の再放出も、電荷保持時間に従う誤りである。動作電圧が小さい近年のメモリについて SILC 影響は比較的小さくなるため、内部電場に依存しないこの再放出の影響が顕著になると言われている [48]。30~40nm のメモリのデータ保持においては、SILC の影響を無視できない程度に受けており、閾値電圧の減少方向への誤りを生じている [8]。

2.5.3 データ読み出し

データの読み出しがは、制御ゲートに高電圧を加え、ドレイン—ソース間の電流を観測することで行われる。同一ストリング内のセルのドレイン—ソースは直列に接続されているため、あるセルにおいて電流を観測するためには、接続されたその他のセルにおいて、ドレイン—ソース間が導通していなければならない。このため、読み出し対象でないセルに対して、閾値電圧によらず電流が流れるような導通電圧を印加する必要があり、これは最大の閾値電圧レベルより高い電圧となる。しかし、このような制御ゲートへの電圧の印加は、書き込みと同様な操作であり、式 (1) に従う FN トンネル現象を引き起こし、閾値電圧が増加してしまう [49]。プログラム

電圧と比較すると導通電圧は小さいため、1回の読み出しによる閾値電圧の変動は小さい。しかし、 10^5 回程度の読み出しを繰り返すと、メモリ本来のビット誤り率であるRBERは2倍以上となる。近年のNANDフラッシュメモリの動作速度を考えると、1分程度連續して読みだすことで、この回数に達してしまう[49]。

2.5.4 ランダムテレグラフノイズ

P/Eサイクルによって基盤との境界面近くに取り込まれた電荷が、取り込みと離脱を起こす現象を、ランダムテレグラフノイズ(Random Telegraph Noise: RTN)と呼ぶ。これによって、閾値電圧分布は正負両方向に広がる[50]。閾値電圧分布への影響は以下のようにモデル化される[44, 50]：

$$p_r(x) = \frac{1}{2\lambda_r} e^{-|x|/\lambda_r}.$$

ここで、 λ_r はP/Eサイクルの回数 N について、 N^α に比例して小さくなる。すなわち、RTNによる閾値電圧分布の広がりは、P/Eサイクルに従い大きくなる。ただし、このモデルはRTNの影響のうち、テイルパートと呼ばれるセルを考慮していない。テイルパートとは、RTNの影響を他のセルより強く受ける少数のセルを指す。テイルパート自体は誤り率に大きく影響するものではないが、データ保持などの閾値電圧分布の移動と重なることで、多レベル誤りの要因となり得る。さらに、この影響は微細化に伴い、より深刻になると考えられている[50]。テイルパートのモデル化の試みはなされているものの[51]、その影響の予測は困難である。

2.6 誤りの特徴

前節において議論した要因により発生する誤りは、閾値電圧分布を一定の強さをもって変化させる。このため、すべての電圧レベルにランダムに誤るのではなく、近接レベルへの変化は、遠くのレベルへの変化より発生率が高い(誤りの影響が極めて強い場合この限りではないが、このとき誤り訂正符号での対処自体が困難であるため、メモリの寿命と見なされる)。20~30nmのプロセスで製造されたメモリを用いた実験[8]では、誤りは2レベル以内のものが中心であり、データ保持誤りの5%を2レベルの変化が占めている。この割合はデバイス毎の差異や動作時の条件によって変わると思われるが、正確に見積もる手段は知られていない。

2.7 信頼性向上と長寿命化

NANDフラッシュメモリの誤り訂正是、通常はオフチップのコントローラーで実行される。(ただし、軟判定の復号法においては、入出力の時間を削減するためオンチップで行われる場合がある[52]。)コントローラーで実行される高信頼化と長寿命化に関わる技術には、誤り訂正符号の他に、不良ブロック管理、ウェアアベーリング、

ガベージコレクションが存在する。これらの手法の役割は、劣化など永続的な影響への対処であること、ウェアレベリングやガベージコレクションがメモリ全体の利用傾向を考慮するといった点で、一時的な(P/Eサイクル後には残らない)ビット誤りを、ページ内で機械的に訂正する誤り訂正符号とは異なる。このため、誤り訂正とこれらの技術は併せて用いられている。ここでそれぞれの技術の概略を紹介する。

2.7.1 不良ブロック管理

メモリには製造時の不良によって、正常に動作しないブロックが含まれる。またP/Eサイクルを繰り返すと、酸化膜への電荷の取り込みに伴い、セルが永続的なダメージを受ける。正常にデータを格納することができないブロックを除外し、歩留まりの向上(すなわち、製造コストの削減)や、信頼性の確保を行う技術が、不良ブロック管理である[53]。不良ブロック管理では不良ブロックを示したテーブルを保有している。これにより製造時の不良ブロックを管理し、また動作時に書き込みや消去の失敗をした場合に、テーブルの更新を行う。不良ブロックのデータは空きブロックに移動し、該当ブロックへの新たな書き込みもそのブロックに送られる[54]。

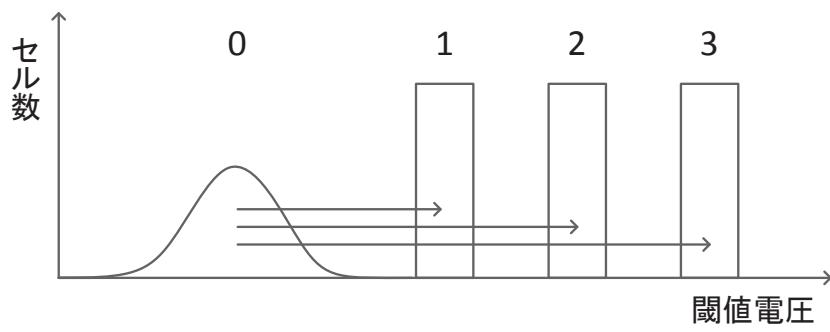
2.7.2 ウェアレベリング

通常、メモリのあらゆるブロックが均等に更新されることはなく、頻繁に更新されるブロックもあれば、滅多に更新されないブロックもある。したがって、ブロック間でP/Eサイクルのばらつきが生じるが、特定のブロックにP/Eサイクルが集中すると、先に記したような不良ブロックが多く発生してしまう。ウェアレベリングは、それぞれのブロックのP/Eサイクルが、できるだけ均等になるようにする技術である[53]。実現方法はさまざまであるが、例えば頻繁に更新されるデータと、更新されにくいデータを分類し、更新されにくいデータを、P/Eサイクルを多く行ったブロックへ移動する方法がある[55]。

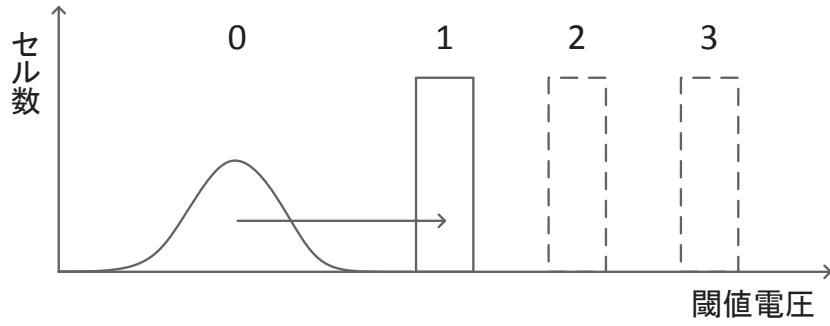
2.7.3 ガベージコレクション

NANDフラッシュメモリの特性として、データは上書きすることができず、既にデータが書き込まれたページを更新したい場合、一度対応するブロックを消去する必要がある。このとき、ブロック内には消去すべきではないデータを保有したページが他に存在するため、データを更新する場合は新たな空きページを利用して書き込みを行い、更新前のデータが存在するページを無効化する[21]。このためデータの更新を繰り返すことにより、不要なデータが格納されているために利用できないページが増えていく。ガベージコレクションはこのような領域を解放し、利用可能な領域として取り戻す技術である[55]。ガベージコレクションの実行に際して、考慮すべき事項がいくつかある。例えば、頻繁な実行は速度低下を招くため、どのタ

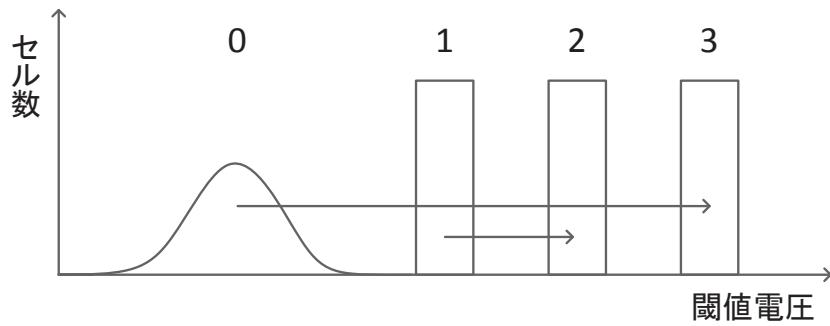
イミングで実行するか、ガベージコレクション自体がP/Eサイクルを必要としているため、いかに特定ブロックへのP/Eサイクルの集中を防ぎながら実行するかが挙げられる[55].



(a)

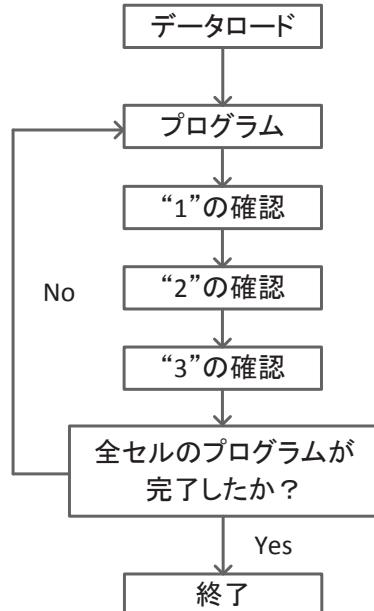


(b)

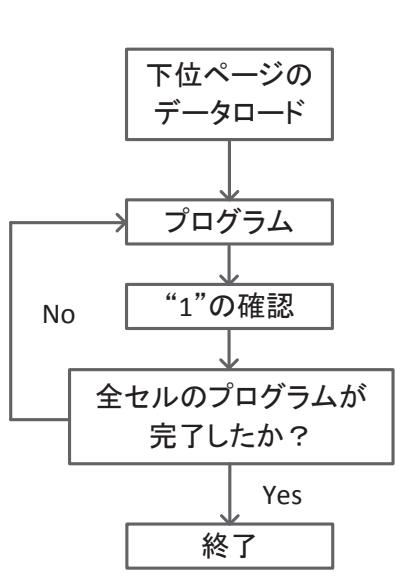


(c)

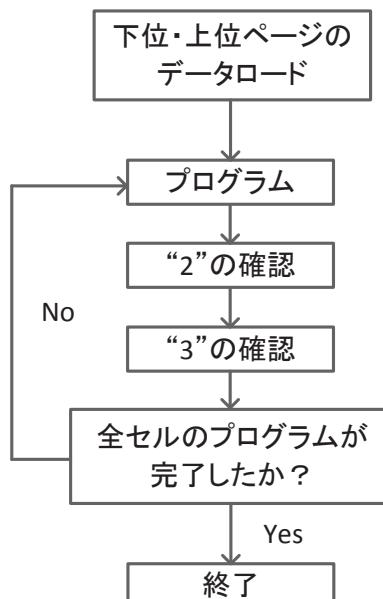
図 7: マルチページプログラミングにおける閾値電圧の変化. (a) 従来プログラミング, (b) マルチページプログラミング下位ページ, (c) マルチページプログラミング上位ページ [35].



(a)



下位ページのプログラム



上位ページのプログラム

(b)

図 8: (a) 従来のプログラミング手順, (b) マルチページプログラミングの手順 [35].

3 誤り訂正符号

情報を扱う通信システムやメモリシステムには、故障やノイズなどの要因によつて、扱うデータに変化が起きる現象、誤りが発生する。誤り訂正符号は、通信システムにおいて送信したり、メモリシステムにおいて格納する情報に、検査用のデータを付加することにより、受信したり格納したデータを利用する際などに、検査を用いた計算を行うことで、誤りの検出や訂正を行う技術である。本章では誤り訂正符号に関する基本事項と、実用される誤り訂正符号について述べる。特に、本論文において提案する LM 誤り訂正符号については詳細に記述するが、その他の手法については概要の紹介に留める。なお、特に断りがない場合、この章の内容は [22, 56, 57] を参考としている。

3.1 数学に関する表記と諸定義

最初に本論文で扱う誤り訂正符号の議論に必要な、整数剰余環と集合に関する表記と定義についてまとめる。

Z を整数環とする。 m を 1 より大きい整数、 a を整数とするとき、

$$C_a = \{x \in Z : x \equiv a \pmod{m}\}$$

によって定義される C_a を、 m を法とする a の剰余類と呼ぶ。 m を法とする剰余類全体の集合、

$$Z_m = \{C_0, C_1, \dots, C_{m-1}\}$$

を考える。簡単のため C_a を \bar{a} と表記する。 Z_m の元 \bar{a}, \bar{b} に、和を写像

$$\begin{array}{ccc} Z_m \times Z_m & \rightarrow & Z_m \\ (\bar{a}, \bar{b}) & \mapsto & \bar{a} + \bar{b} = \overline{a+b} \end{array}$$

によって定義し、積を写像

$$\begin{array}{ccc} Z_m \times Z_m & \rightarrow & Z_m \\ (\bar{a}, \bar{b}) & \mapsto & \bar{a} \cdot \bar{b} = \overline{a \cdot b} \end{array}$$

によって定義すると、 Z_m は環となる [58].

定義 1 上記のように定義された環 Z_m を、整数剰余環と呼ぶ [58].

定義 2 Z_m の元 \bar{a} について、零元でない Z_m の元 \bar{b} が存在し、 $\bar{a}\bar{b} = \bar{0}$ となるとき、 \bar{a} を Z_m の零因子と呼ぶ [58].

定義 3 Z_m の元 \bar{a} について、 Z_m の元 \bar{b} が存在し、 $\bar{a}\bar{b} = \bar{1}$ となるとき、 \bar{a} を Z_m の単元と呼ぶ [58].

すなわち，乗法逆元が存在する Z_m の元が単元である。以降は混乱がない場合，整数剰余環の元 \bar{a} を，整数と区別なく a と表記する。

集合に関する表記について， ϕ を空集合とし，任意の集合 A とその部分集合 B について， $A \setminus B$ を A から B の元を全て除いたもの， $|A|$ を A に含まれる元の数とする。集合 $S \subseteq Z_m$ および $x \in Z_m$ について， $xS = \{xs : s \in S\}$ とする。ベクトル $\mathbf{v} = (v_1, v_2, \dots, v_n) \in Z^n$ と整数 $q > 1$ について，

$$\mathbf{v} \bmod q = (v_1 \bmod q, v_2 \bmod q, \dots, v_n \bmod q)$$

とする。ベクトル $\mathbf{v}' = (\bar{v}_1, \bar{v}_2, \dots, \bar{v}_n) \in Z_m^n$ についても同様に，

$$\mathbf{v}' \bmod q = (\overline{v_1 \bmod q}, \overline{v_2 \bmod q}, \dots, \overline{v_n \bmod q})$$

とする。

3.2 誤り訂正の基礎

3.2.1 情報システムのモデル

通信システムやメモリシステムにおける誤り訂正符号の役割を考える。図9は，情報システムを符号の観点からモデル化したものである。情報源からは情報語が発生する。情報語における各記号は情報アルファベットと呼ばれる集合の元であり，この集合が q 元から成るならば， q 元の情報源と呼ぶ。符号器では，情報語を符号語に変換する。この操作を符号化と呼ぶ。符号語は，符号アルファベットと呼ばれる集合の記号から成り，この集合が q 元から成れば， q 元の符号化と呼ぶ。生じ得るあらゆる情報語を考えた際に，出力される符号語の集合を，符号と呼ぶ。通信路では符号語を入力し，受信語が出力される。受信語の各記号は，通信路アルファベットの元であるが，硬判定の復号器を想定すると，これは符号アルファベットと一致する。符号語と受信語が一致しない場合，誤りが発生したこととなる。復号器では，受信語から推定した情報語である復号語を，あて先へ渡す。この操作を復号と呼ぶ[57]。

3.2.2 誤りのモデル

通信路は，符号語を受信語に確率的に変換する。受信語が符号語と異なるならば誤りが発生したこととなり，この確率的変換をモデル化したものが，通信路モデルである。符号アルファベットに加算と減算が定義されており，通信路において，受信語は符号語に誤り語と呼ばれるベクトルが加算されたものとする。このとき，この誤り語を発生する確率的モデルを誤りモデルと呼ぶ。

通信路モデルとして最も基本的なものは，2元対称誤り通信路である。これはアルファベットが $\{0, 1\}$ からなり， $0 \rightarrow 1$ の遷移と， $1 \rightarrow 0$ の遷移が，等確率で発生するモデルである。ハミング符号やBCH符号など，このモデルに基づく符号は数多

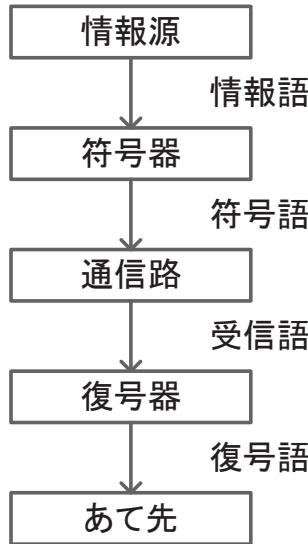


図 9: 情報システムのモデル [57]

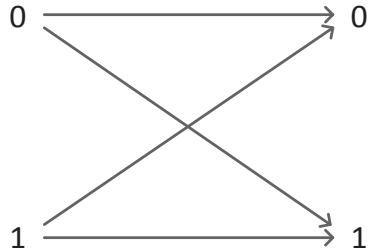


図 10: 2 元対称誤り通信路.

く存在する。通信路モデルを表すには、通信路線図を用いることができる。2元対称誤りの通信路線図を図 10 に示す。

NAND フラッシュメモリの誤りは、LM 誤りモデルと見なすことができる。これは、符号アルファベットを整数値としたとき、変量が一定以上となる値の遷移を起こすような誤りの発生率が、無視できるモデルである。4元の対称シンボル誤りと、LM 誤りの比較を図 11 に示す。この図における LM 誤りは、0 から 2 といったような 1 より大きな値の変化が無視できる場合を表している。

3.2.3 ハミング距離と誤り訂正

受信語が成すべきトル空間を、受信空間と呼ぶ。受信空間に符号語として利用しない領域を作り、そのような領域に属する受信語から適切な符号語を推定し、出力す

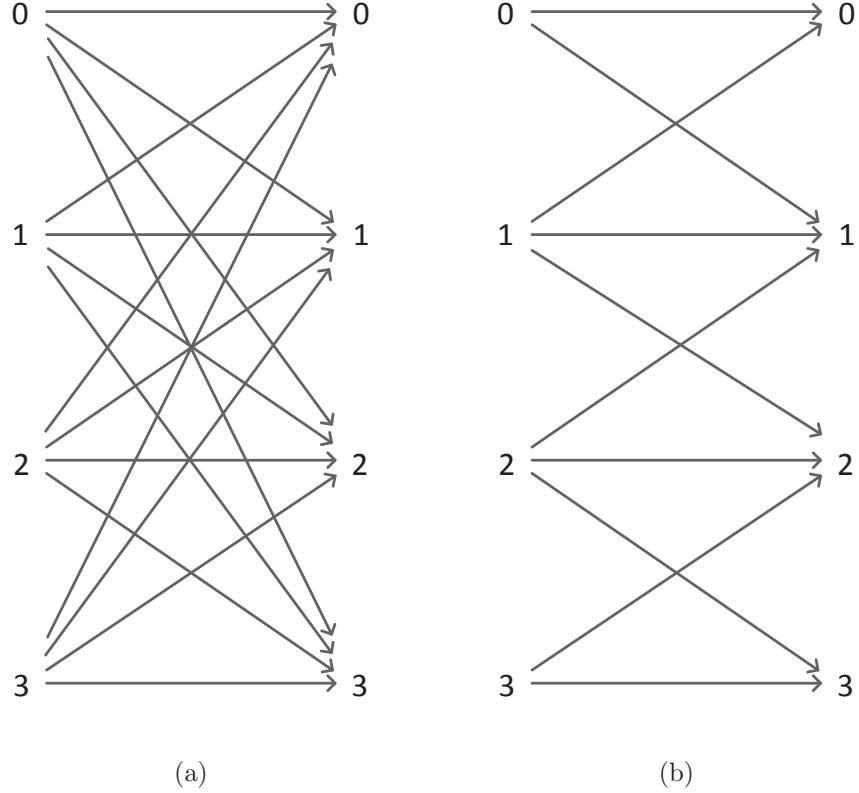


図 11: 4 元の通信路. (a) 対称シンボル誤り, (b) 対称 LM 誤り.

ることが誤り訂正の原理である. 受信空間に距離とよばれる尺度を定義すると, 誤り訂正機能について議論を行う際に便利である. 特に対称誤り訂正機能の必要十分条件を与える概念として, ハミング距離がある.

定義 4 ふたつの記号 x, y について, ハミング距離を

$$d_H(x, y) = \begin{cases} 0 & (u = v), \\ 1 & (u \neq v), \end{cases}$$

によって定義する. また 2 つのベクトル $\mathbf{x} = (x_1, \dots, x_n), \mathbf{y} = (y_1, \dots, y_n)$ について, ハミング距離を

$$d_H(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n d_H(x_i, y_i)$$

によって定義する.

すなわち, ハミング距離は 2 つのベクトル \mathbf{x} と \mathbf{y} の, 異なる要素の数である. 符号 C の対称誤り検出・訂正機能は, 任意の 2 つの異なる符号語間のハミング距離の最小値, 最小ハミング距離 d_{\min} によって定まる. 最小ハミング距離を単にハミング距離や, 距離と呼ぶ場合がある.

定理 1 符号 C が t 重誤りを訂正可能であるための必要十分条件は, $d_{\min} \geq 2t + 1$ を満たすことである.

定理 2 符号 C が d 重誤りを検出可能であるための必要十分条件は, $d_{\min} \geq d + 1$ を満たすことである.

定理 3 符号 C が t 重誤りを訂正可能であり, かつ $d(d > t)$ 重誤りを検出可能であるための必要十分条件は, $d_{\min} \geq t + d + 1$ を満たすことである.

証明は省略するが, 概略は次の通りである. 符号語からハミング距離が t 以下となる領域(復号領域)を考える. $d_{\min} \geq 2t + 1$ であればこれら領域に重複が無く, また t 重対称誤りが発生した場合の受信語は, もとの符号語の復号領域内に含まれる. よって領域内の符号語を出力することで, t 重対称誤りの訂正が可能となる. 同様に, ある符号語のハミング距離が d となる復号領域が他の符号語を含まなければ, 検出が可能である. 訂正と検出の両方を行う場合は, 訂正のための符号語からハミング距離 t の領域と, 検出のための符号語からハミング距離 d の領域に, 重複が無ければよい. ハミング距離の与える訂正機能は対称シンボル誤りに関するものであり, 非対称誤りなどの訂正機能については, 必要十分条件を与えないことに注意する. ハミング距離について, 次のようにハミング重みを定義する.

定義 5 ベクトル \mathbf{x} のハミング重みを $w_H(\mathbf{x}) = d_H(\mathbf{0}, \mathbf{x})$ によって定義する(ただし, $\mathbf{0} = (0, \dots, 0)$ とする).

ハミング距離とハミング重みには, $d_H(\mathbf{x}, \mathbf{y}) = w_H(\mathbf{x} - \mathbf{y})$ という関係がある. ハミング重みは, 線形符号の誤り訂正機能に関係する.

3.2.4 ハミングの限界式

q 元の t 誤り訂正符号の復号領域は, 符号語を中心とする半径 t の球である. この球に含まれる受信空間上の点の数は,

$$\sum_{i=0}^t \binom{n}{i} (q-1)^i$$

である. 受信空間の点の総数は q^n であり, これらの球に重複はないので, 符号語の数 M は,

$$M \leq \frac{q^n}{\sum_{i=0}^t \binom{n}{i} (q-1)^i} \tag{2}$$

を満たす. この符号語数の限界式をハミングの限界式と呼び, 等号を満たす符号を完全符号と呼ぶ. 完全符号は最大の符号語数を持つという意味で, 最良の符号であ

る。式(2)はハミング距離に基づいたものであるが、より一般の距離においても、同様な議論が可能である。すなわち、受信空間上のすべての点に対して、 A 個の点を含む復号領域が重複なく割り当てられるとき、

$$M \leq \frac{q^n}{A}$$

なる限界式が成り立ち、等号を満たす符号を完全符号と呼ぶ。

3.3 線形符号

ハミング距離の条件を満たしていれば、原理的に誤りの検出や訂正は可能である。しかし、具体的にどのように一定のハミング距離をもつ符号を構成するか、大きい受信空間を持つ符号に対して、どのように現実的な計算量で復号するかが問題となる。このため、符号に何らかの数学的構造を持たせると都合がよく、誤り訂正符号の多くは、理論的に扱いやすい線形符号である。

定義 6 符号長 n の符号語 C がベクトル空間 V^n の部分空間をなすとき、 C を線形符号と呼ぶ。すなわち、任意の 2 つの符号語 $\mathbf{x}_1, \mathbf{x}_2 \in C$ 、および定数 $c_1, c_2 \in V$ について、

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 \in C$$

のとき、 C は線形符号である。

任意の 2 つの符号語の和は符号語となるため、線形符号 C においては、ハミング距離の代わりに、すべての符号語のハミング重みの最小値を調べることで、最小ハミング距離を得ることができる。

線形符号は、生成行列または検査行列によって定義することができる。符号語 C の基底となる一次独立な k の符号語を $\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_k$ とすると、

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \vdots \\ \mathbf{g}_k \end{bmatrix}$$

によって表される $k \times n$ の行列 \mathbf{G} を生成行列と呼ぶ。 $\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_k$ の線形結合、

$$d_1\mathbf{g}_1 + d_2\mathbf{g}_2 + \cdots + d_k\mathbf{g}_k$$

は定義 6 により符号語であり、 $\mathbf{d} = (d_1, d_2, \dots, d_k)$ により一意に定まる。これは符号語 \mathbf{x} が、 $\mathbf{x} = \mathbf{d} \cdot \mathbf{G}$ なる計算により得られることを意味する。

検査行列は、 \mathbf{G} の行空間の直行空間として定義される。すなわち、 $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$ となる $r \times n$ の行列 \mathbf{H} が検査行列である。 $\mathbf{x} = \mathbf{d} \cdot \mathbf{G}$ を用いると、

$$\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0} \tag{3}$$

を得る。生成行列の行数 k は情報長，列数 n は符号長に対応し，検査行列の行数 r は検査長に対応する。符号長が n ，情報長が k の符号を， (n, k) 符号と表記する。

代表的な線形符号として，ハミング符号がある。この符号は零でないすべてのベクトルを検査行列の列ベクトルとしてもつ， $n = 2^r - 1, k = n - r$ の单一ビット誤り訂正符号である。

例 1 $(7, 4)$ ハミング符号は，検査行列

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

によって定義される。

ハミング符号は多元の単一シンボル誤り訂正に拡張することが可能である。 q 元の体を用いて構成されたハミング符号の，符号シンボル長は $n = (q^r - 1)/(q - 1)$ ，情報シンボル長は $k = n - r$ となる。

例 2 ガロア体 GF(4) 上に構成されたハミング $(5, 3)$ 符号の検査行列は，

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & \alpha & \alpha^2 \end{bmatrix}.$$

によって定義される。

線形符号の検査行列を \mathbf{H} ，受信語を \mathbf{x}' とする。このとき，

$$\mathbf{s} = \mathbf{x}' \cdot \mathbf{H}^T$$

をシンドロームと呼ぶ。誤りを \mathbf{e} とおくと， $\mathbf{x}' = \mathbf{x} + \mathbf{e}$ であるから，式(3)を用いて，

$$\begin{aligned} \mathbf{x}' \cdot \mathbf{H}^T &= (\mathbf{x} + \mathbf{e}) \cdot \mathbf{H}^T \\ &= \mathbf{x} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T \\ &= \mathbf{e} \cdot \mathbf{H}^T \end{aligned}$$

を得る。すなわちシンドロームは符号語に依存せず，誤りと符号によって定まる。特に誤りの検出は式(3)により，シンドロームが零であるかを調べることで容易に実行できる。

例 3 2元のハミング符号では，单一誤りに対するシンドロームが，誤りが異なれば必ず異なるように設計されている。すなわち，单一誤りが発生した場合，シンドロームは検査行列の列ベクトルに一致するため，検査行列は相違な列ベクトル全てから成る。誤りの訂正是シンドロームと一致する列ベクトルを特定し，符号語中の対応するビットを反転することで実現する。

3.4 巡回符号

巡回符号は線形符号の一種であり、符号語の巡回シフトもまた符号語となるものである。すなわち、符号語を $\mathbf{x} = (x_1, x_2, \dots, x_n)$ としたとき、 $\mathbf{x}' = (x_n, x_1, x_2, \dots, x_{n-1})$ もまた符号語となる。巡回符号は符号語ベクトルの各記号を、多項式の係数と対応させた符号多項式を用いて記述される。生成行列 \mathbf{G} に対応する、生成多項式と呼ばれる多項式 $G(x)$ の倍多項式は、符号多項式を定義する。巡回符号の特徴として、符号化や復号装置が簡略化できることが挙げられる。例えば、符号化は生成多項式と呼ばれる多項式の除算を用いて実行でき、これは除算回路である線形フィードバックシフトレジスタで実装できる。また、バースト誤り検出機能に優れることや、次節の BCH 符号のように、訂正機能の強い符号の優れた構成法が存在するといった特徴を持つ。

3.5 BCH 符号

BCH 符号は、多重誤り訂正が可能な巡回符号の一種であり、MLC の NAND フラッシュメモリにおいても広く用いられている [12]。2 元の BCH 符号は、任意の $m (m \leq 3), t < 2^m - 1$ について、符号長 $n = 2^m - 1$ 、検査長 $r \leq mt$ 、最小距離 $d_{\min} \geq 2t + 1$ をパラメータとする。符号の構成は、次の定理による最小距離の限界 (BCH 限界) に基づく。

定理 4 (n, k) 巡回符号の C の生成多項式が、 α を 1 の原始 n 乗根としたとき、 $\alpha^l, \alpha^{l+1}, \dots, \alpha^{l+h-1}$ を根とするならば、 C の最小距離は $h + 1$ 以上となる。

$m_i(x)$ を、 α^i を根とする次数最小の多項式とする。2 元の符号では $m_i(x) = m_{2i}(x)$ が成り立つため、BCH 符号の生成多項式は、

$$G(x) = \text{LCM}[m_1(x), m_3(x), \dots, m_{2t-1}(x)]$$

によって表される。ただし、LCM は最小公倍多項式を表す。

3.6 Low-Density Parity-Check 符号

Low-Density Parity-Check(LDPC) 符号もまた、代表的な多重誤り訂正符号である。疎な検査行列により定義される符号であり、sum-product 復号法に代表される軟判定の復号アルゴリズムを用いることで、シャノン限界に迫る符号化率を実現できる [56]。この特徴から、近年の NAND フラッシュメモリで問題となる、高い誤り率に対処可能な符号として利用されている [3, 4, 11]。ただし、通常 BCH 符号と比較して復号は複雑になる。

3.7 多元非対称誤り訂正符号

3.3節で紹介した多元のハミング符号は、単一シンボル誤りであれば、どの値からどの値へ変化したかによらず、訂正できる。しかし、値の遷移によって発生確率が異なる誤りモデルも考えられる。具体例として、キーボード入力や文字認識などの、データ入力システムが挙げられる。シンボル‘ a ’からシンボル‘ b ’に誤入力する確率と、シンボル‘ a ’からシンボル‘ c ’に誤入力する確率は、‘ b ’と‘ c ’が異なるシンボルであれば、一般に等しくない。このような誤りを多元非対称誤りと呼ぶ[59]。ここではこの誤りの定義を示すが、具体的な誤り訂正符号の構成法は、[59]を参照されたい。

定義 7 [59] $A = \{a_1, \dots, a_M\}$ を M 元のシンボルから成る集合とする。非対称誤りの集合 ε を

$$\varepsilon = \{(a_i \rightarrow a_j) : a_i, a_j \in A, \Pr(a_i|a_j) > T, 1 \leq i \leq M, 1 \leq j \leq M, i \neq j\} \quad (4)$$

によって定義する。ここで、 $(a_i \rightarrow a_j)$ は a_i から a_j へと変化するシンボル誤り、 $\Pr(a_i|a_j)$ は $(a_i \rightarrow a_j)$ の発生する確率、 T は確率の閾値とする。

定義 8 [59] $(a_i \rightarrow a_j) \in \varepsilon$ とする。このとき、 a_i から a_j に変化する誤りを、 ε -非対称シンボル誤りと呼ぶ。

すなわち、このモデルに基づく多元非対称誤り訂正符号は、発生率が T より大きいシンボル誤りに限定して訂正を行う。

3.8 Limited-Magnitude 誤り訂正符号

3.8.1 諸定義

LM 誤りは MLC のフラッシュメモリに見られるような、近傍の値への誤りのみを発生するモデルである。以下に[29]における非対称 LM 誤りの定義を示す。ただし、値が増加する誤りと、減少する誤り両方について扱えるよう、定義を拡張している。 $Q_q = \{0, 1, \dots, q-1\}$ とし、符号語を $\mathbf{v} \in Q_q^n$ 、誤り語を $\mathbf{e} \in Q_q^n$ 、受信語を $\mathbf{r} \in Q_q^n$ とする。

定義 9 [29] 式 $\mathbf{v} + \mathbf{e} = \mathbf{r}$ を満たす $\mathbf{e} = (e_1, e_2, \dots, e_n)$ と整数 $l > 0$ について、 $|\{i : e_i \neq 0\}| \leq t$ かつ任意の i について $0 \leq e_i \leq l$ が成り立つとき、 \mathbf{e} を t 非対称 l -LM 誤りと呼ぶ。また、 $|\{i : e_i \neq 0\}| \leq t$ かつ任意の i について $-l \leq e_i \leq 0$ が成り立つとき、 \mathbf{e} を t 非対称 $(-l)$ -LM 誤りと呼ぶ。

同様に、单方向 LM 誤り、不均衡 LM 誤りを以下のように定義する。

定義 10 すべての t 非対称 l -LM 誤りから成る集合を E_1 とし、すべての t 非対称 $(-l)$ -LM 誤りから成る集合を E_2 とする。 $\mathbf{e} \in E_1 \cup E_2$ となる \mathbf{e} を、 t 单方向 l -LM 誤りと呼ぶ。

定義 11 式 $\mathbf{v} + \mathbf{e} = \mathbf{r}$ を満たす $\mathbf{e} = (e_1, e_2, \dots, e_n)$ と整数 $l_u \geq 1, l_d \geq 1$ について, $|\{i : e_i \neq 0\}| \leq t$ を満たし, かつ任意の i について $-l_d \leq e_i \leq l_u$ が成り立つとき, \mathbf{e} を t 不均衡 (l_u, l_d) -LM 誤りと呼ぶ.

不均衡 LM 誤りにおいて, $l = l_d = l_u$ の場合を t 対象 l -LM 誤りと呼び, $t > 2$ である場合を t 双方向 (l_u, l_d) -LM 誤りと呼ぶことがある. また, 便宜上 $l_u = 0$ または $l_d = 0$ とした場合, 不均衡 LM 誤りは非対称 LM 誤りと一致し, $l_u + l_d + 1 = q$ のとき, 対称シンボル誤りと一致する. $t = 1$, つまり單一シンボル誤りとしたとき, 定義 8 の ε -非対称シンボル誤りは, LM 誤りとなるように定義することができる. すなわち, LM 誤りは多元非対称誤りの一種である.

ここで本論文で記述される誤りについての表記を整理する. ビット誤りといった場合, 2元の対称誤りを指すものとする. 多元(特殊な例として2元を含む)の対称誤りは, 対称 LM 誤りと区別して対称シンボル誤りと記述する.

LM 誤り訂正符号の構成には, $B_t(S)(q)$ 集合が重要な役割を果たす. 2つの整数 $a, b (a \leq b)$ について, $[a, b] = \{a, a+1, \dots, b\}$ とする.

定義 12 [23] S を整数から成る集合とし, B を異なる m 個の整数 $b_i \in [1, q-1]$ から成る集合とする. $a_i \in S$ および $b_{i_j} \in B (0 \leq i_1 < i_2 < \dots < i_t \leq m-1)$ について,

$$\left(\sum_{i=1}^t a_i b_{i_j} \right) \mod q$$

によって得られる値が全て異なるとき, $B = \{b_i | 0 \leq i \leq m-1\}$ を(q を法とする) $B_t(S)(q)$ 集合と呼ぶ.

3.8.2 完全・準完全な集合 $B[l](q)$ と $B[\pm l](q)$

$B_1([0, l])(q)$ を, 簡単のため $B[l](q)$ と表記する. 同様に, $B_1([-l, l])(q)$ を $B[\pm l](q)$ と表記する. $B[l](q)$ は單一非対称 l -LM 誤り, $B[\pm l](q)$ は單一対称 l -LM 誤りを訂正する符号の構成に用いる. なお, より一般の單一不均衡 (l_u, l_d) -LM 誤りについても, $B_1([-l_d, l_u])(q)$ 集合を用いることで同様な議論が可能であるが, 詳細は[27, 28]を参照されたい.

\mathbf{H} を, 先頭の非零の要素が B に属するような, Z_q^r の元を列ベクトルとする $r \times n$ 行列とし, $C_r(B)$ を \mathbf{H}^T の直交空間とする. \gcd を最大公約数とする.

定理 5 [23] $B = B[l](q)$ かつ $\gcd(q, l!) = 1$ のとき, $C_r(B)$ は單一非対称 l -LM 誤りを訂正可能である.

(証明) 誤りを $\mathbf{e} = (0, \dots, 0, e_i, 0, \dots, 0) (1 \leq e_i \leq l)$ としたとき, シンドロームは検査行列の第 i 列 \mathbf{h}_i を用いて,

$$\mathbf{s} = e_i \mathbf{h}_i^T$$

と表される。ここで \mathbf{h}_i^T は第 i 列の転置である。シンドロームの最初の非零の要素は、 $z = e_i b$ を満たす。ただし、 $b \in B$ である。集合 B の定義より z から e_i, b の組を一意に特定可能である。 $\gcd(q, l!) = 1$ のとき、 $1, 2, \dots, l$ は単元である。したがって、乗法逆元 e_i^{-1} が存在し、 $e_i^{-1} \mathbf{s} = \mathbf{h}_i^T$ となり列ベクトルを特定可能である。以上により値 e_i の誤りが、受信語の \mathbf{h}_i に対応する要素に発生したことが分かる。（証明終了）

定理 6 [23] $B = B[\pm l](q)$ かつ $\gcd(q, l!) = 1$ のとき、 $C_r(B)$ は單一対称 l -LM 誤りを訂正可能である。

(証明) $\gcd(q, l!) = 1$ のとき、 $-1, -2, \dots, -l$ がすべて単元であることを用いれば、定理 5 と同様に証明可能である。（証明終了）

第 i 要素が B に含まれる \mathbf{H} の列ベクトル数は、 B に含まれる元の数を m とすると mq^{r-i} であるから、符号長 n は

$$n = \sum_{i=1}^r (mq^{r-i}) = m \frac{q^r - 1}{q - 1} \quad (5)$$

で与えられ、したがって m が大きいほど優れた符号長が得られる。このため單一 LM 誤り訂正符号の研究においては、与えられたパラメータに対して、集合 B の元の数をいかに最大化するかが主題となっている。 $B[l](q)$ の定義より、 $1 + lm \leq q$ は必ず満たされる。等号が成り立つとき、 $B[l](q)$ は完全であると言う。また、構成できる $B[l](q)$ の最大の元の数を $M_l(q)$ で表すと、

$$M_l(q) \leq \left\lfloor \frac{q-1}{l} \right\rfloor \quad (6)$$

が満たされる。最大の元の数をもつ $B[l](q)$ について、式 (6) の等号が成り立ち、かつ完全でないとき、 $B[l](q)$ は準完全であると言う [23]。 B が完全であるとき、以下の定理が成り立つ。

定理 7 [23] B が完全な $B[l](q)$ 集合であり、かつ $\gcd(q, l!) = 1$ のとき、 $C_r(B)$ は完全符号である。

(証明) 定義より $|B| = m = (q-1)/l$ であり、 $C_r(B)$ の符号長は式 (5) で与えられる。したがって $n = (q^r - 1)/l$ であり、

$$|C_r(B)|(1 + nl) = q^{n-r} q^r = q^n$$

となる。これは受信空間における点の総数であり、ゆえに $C_r(B)$ は完全符号である。（証明終了）

例 4 p が $p \equiv 3(\text{mod}8)$ または $p \equiv 5(\text{mod}8)$ を満たす素数とする. このとき, 集合

$$Q_p = \{x^2 \bmod p : 1 \leq x \leq (p-1)/2\}$$

は完全な $B[2](p)$ 集合となることが知られている [23]. $p = 13$ とすると,

$$Q_{13} = \{1, 4, 9, 3, 12, 10\}$$

を得る. この集合は $|Q_{13}| = M_2(p) = (p-1)/2$ を満たし,

$$2Q_{13} = \{2, 8, 5, 6, 11, 7\}$$

は Q_{13} と重複がないため, $B[2](p)$ となっていることが分かる.

非対称 LM 誤り訂正符号は, 増加と減少の誤りが両方発生し得る場合には適さず, この場合, 対称 LM 誤り [25, 26] が有効である.

$B[\pm l](q)$ において, 定義より $1 + 2ml \leq q$ は必ず満たされる. 特に等号が成り立つとき, $B = B[\pm l](q)$ とおけば, $C_r(B)$ は完全な単一対象 l -LM 誤り訂正符号となる. このような $B[\pm l](q)$ を完全であると言う. また $B[\pm l](q)$ の取り得る最大の元の数を $N_l(q)$ とすると,

$$N_l(q) \leq \left\lfloor \frac{q-1}{2l} \right\rfloor \quad (7)$$

が成り立つ. 最大の元の数をもつ $B[\pm l](q)$ について, 式 (7) の等号が成り立ち, かつ完全でないとき, $B[\pm l](q)$ を準完全であると言う [23].

完全な $B[l](q)$ が, $q \equiv 1(\text{mod}l)$ のときのみ存在し得るのは, 明らかである. またこれを満たさない場合も, 準完全な $B[l](q)$ は存在する場合がある. 元の数において優れた $B[l](q)$ は, [23, 24] において構成法が研究されているが, 完全, 準完全な $B[l](q)$ は限られた l, q においてのみ得られており, どちらも存在し得ない場合があることに注意されたい. 表 3 に [23] において得られている, $B[2](q)$ の最大の元の数 $M_2(q)$ を例として示す. $q \equiv 1(\text{mod}l)$ を満たしているものの, 完全な集合は太字で示した q のみでしか得られない.

元の数において優れた $B[\pm l](q)$ についても, 限られた l, q についてのみ構成法が知られてられている [25, 26]. この場合も $B[l](q)$ と同様に, 完全または準完全な集合の, どちらも存在し得ない場合がある. 現在知られている優れた $B[\pm l](q)$ の構成法は, 次のように分類できる.

- 完全な $B[\pm l](p)$ 集合 (p は素数)
一部の l, p について, 構成法が知られている (表 4).
- 準完全な $B[\pm l](lp)$ 集合 (p は素数)
一部の l, p について, 構成法が知られている (表 5).

表 3: q が奇数の場合の $M_2(q)$ [23].

q	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33
$M_2(q)$	1	2	2	4	5	6	7	8	9	9	10	12	13	14	12	16
q	37	39	41	43	45	47	49	51	53	55	57	59	61	63	65	67
$M_2(q)$	18	19	20	21	22	22	22	25	26	27	28	29	30	30	32	33

太字は完全な $B[l](q)$ が得られているもの.

表 4: 完全な $B[\pm l](p)$ を与える $l, p (l \leq 5, p \leq 128)$ のもの)[26].

l	2										3	4	5			
p	5	13	17	29	37	41	53	61	97	101	109	113	7	37	97	11

- 準完全な $B[\pm 4](2p)$ 集合 (p は素数)

一部の p について構成法が知られている. p を小さいものから列挙すると, $p = 5, 29, 53, 101, \dots$ となる.

- $B[\pm 2](q)$ 集合

元の数最大のものが任意の q について得られている.

- $B[\pm 3](q)$ 集合

元の数最大のものが一部の q について得られている. q を小さいものから列挙すると $q = 17, 21, 35, 37, 49, 69, \dots$ となる.

また, 2つの集合 $B_1 = B[\pm l](q_1), B_2 = B[\pm l](q_2)$ から, 新たな集合 $B[\pm l](q_1 q_2)$ を得る方法が知られている. 特に B_1 が完全のとき,

- B_2 が完全なら $B[\pm l](q_1 q_2)$ は完全
- B_2 が準完全なら $B[\pm l](q_1 q_2)$ は準完全

となる. 以上により多くの l, q について, 優れた $B[\pm l](q)$ が得られるが, 上記以外のパラメータにも, 実用上重要なものが存在する. 例えば, 第5章の提案手法1にて紹介するように, 集合 $B[\pm 4](31)$ は有用であるが, 元の数を最大化するような構成法は知られていない.

3.8.3 多重LM誤り訂正符号

t 非対称 l -LM 誤り訂正符号の構成法として, $B_t([0, l])(q)$ を利用する方法[24]が知られている. この場合は単一誤り同様, 集合を最大化することで符号長の大きい優れた符号が得られる. また, 従来の t 重対称シンボル誤り訂正符号を利用して符号を構成する方法が知られている[29]. この手法については, 符号化や復号に構成に用いた対称シンボル誤り訂正符号のアルゴリズムを利用できる利点がある. NAND

表 5: 準完全な $B[\pm l](lp)$ を与える l, p ($l \leq 5, lp \leq 128$ のもの)[26].

l	2								3			4		5		
p	3	7	11	19	23	31	43	47	59	7	23	31	11	23	19	23

フラッシュメモリのような t が大きいアプリケーションにおいては、後者のように現実的な計算量で実行できる必要があり、こちらについて記述する。なお定理の証明など本項の詳細については、[29] を参照されたい。

まず、 t 非対称 l -LM 誤り訂正符号であるための必要十分条件を与える距離 d_l を定義する。 $Q_q = \{0, 1, \dots, q-1\}$ とする。 $\mathbf{x} = (x_1, \dots, x_n) \in Q_q^n, \mathbf{y} = (y_1, \dots, y_n) \in Q_q^n$ について、

$$N(\mathbf{x}, \mathbf{y}) = |\{i : x_i > y_i\}|$$

とする。

定義 13 [29] 2つのベクトル \mathbf{x}, \mathbf{y} の距離 $d_l(\mathbf{x}, \mathbf{y})$ を、

$$d_l(\mathbf{x}, \mathbf{y}) = \begin{cases} n + 1 & (\max_i\{|x_i - y_i|\} > l), \\ \max(N(\mathbf{x}, \mathbf{y}), N(\mathbf{y}, \mathbf{x})) & (\text{otherwise}). \end{cases}$$

によって定義する。

定理 8 [29] 符号 C が t 非対称 l -LM 誤りを訂正可能であるための必要十分条件は、任意の異なる 2つの符号語 $\mathbf{x}, \mathbf{y} \in C$ が、 $d_l(\mathbf{x}, \mathbf{y}) \geq t + 1$ を満たすことである。

構成法： Ω を $q' = l + 1$ 元の t 重対称シンボル誤り訂正符号とする。 $q(q > q')$ 元の符号 C を、

$$C = \{\mathbf{x} \in Q_q^n : \mathbf{x} \bmod q' \in \Omega\} \quad (8)$$

によって定義する。

定理 9 定義 8 によって構成される符号 C は、 t 非対称 l -LM 誤り訂正符号である。また $q > 2l$ の場合、逆もまた成り立つ。

この定理は、情報語に対して式(8)を満たすような C への適切な写像を行うことにより、非対称 LM 誤り訂正符号の符号化ができるることを意味している。単純な方法としては、情報語に対して q' で剰余をとり、 q' 元の t 重対称シンボル誤り訂正符号で検査シンボルを求めればよい。この方法では得られた検査シンボルを、定理が満たされるような適切な方法で q 元シンボルに写像する必要がある。例えば $l = 1$ の場合は、検査部をグレイ符号を用いて写像することにより、検査部においても情報部と同じ訂正機能を持つが、一般の q, l について、検査長を犠牲とせずに検査部を q

元に写像する方法は知られていない。このため、单一誤り訂正符号についてもこの構成法は有効ではあるが、整数剩余環上の検査行列により定義される組織符号のほうが優れた符号効率となり得る。

続いて、この構成法の最適性について述べる。次の定理は、 Ω の符号語数と C の符号語数の関係を与える。

定理 10 [29] C の符号語数は、

$$\left\lfloor \frac{q}{q'} \right\rfloor^n \cdot |\Omega| \leq |C| \leq \left\lceil \frac{q}{q'} \right\rceil^n \cdot |\Omega| \quad (9)$$

を満たす。

式(8)による符号構成は、完全符号を得られる場合があるという意味で、最良であると言える。まず式(2)のハミングの限界式を t 非対称 l -LM 誤りについて書き換えると、

$$|C| \cdot \sum_{i=0}^t \binom{n}{i} l^i \leq q^n \quad (10)$$

を得る。定理 10 を用いると、

$$|C| \cdot \sum_{i=0}^t \binom{n}{i} l^i = \left(\frac{q}{q'} \right) \cdot |\Omega| \cdot \sum_{i=0}^t \binom{n}{i} l^i$$

となる。もし Ω が完全符号ならば、

$$|\Omega| \cdot \sum_{i=0}^t \binom{n}{i} l^i = q'^n$$

であるため、

$$|C| \cdot \sum_{i=0}^t \binom{n}{i} l^i = \left(\frac{q}{q'} \right) \cdot q'^n = q^n$$

となる。したがって式(10)の等号が成り立ち、 C は完全符号となる。

4 NAND フラッシュメモリにおける誤り訂正符号

最初に、NAND フラッシュメモリにおける、近年の誤り訂正の研究に至るまでの歴史について簡単に述べる。微細化に伴う誤り率の増大によって、近年特に重要な役割を果たしていると言える誤り訂正符号であるが、その利用自体は古くから行われてきた。誤り率は現在のメモリと比較して低かったが、P/E サイクルに伴って増大する特徴があり、誤り訂正符号を用いることで許容できる P/E サイクル数が改善されるためである。1997 年当時の SLC のメモリにおいては、ページを適切な大き

さのセクタに区切れば、セクタへ SEC(Single Error Correcting) 符号を適用することで、必要な信頼性と寿命を確保することができた [60]. 1998 年に MLC[61] が提案されると、SEC 符号で信頼性を確保することが難しくなり、NAND フラッシュメモリへ適用する符号としても、多ビット誤り訂正符号である BCH 符号が用いられるようになった. 2006 年の時点では、符号長 4148 に対し 4 ビット誤り訂正程度の符号が利用されている [19]. これ以降、急激な誤り率の増大に対処するために、BCH 符号の訂正機能も大きくなる. 2011 年には MLC の SSD において、訂正機能 t が 40 以上にも達している [14]. ところが、BCH 符号復号の複雑度は概ね t^2 に比例し、検査ビット長は概ね t に比例するため、 t の過剰な増大は、許容できないオーバーヘッドとなってしまう [56].

こういった背景から、必要な誤り率の要件を、いかに少ない計算量や検査ビットで実現するかをトピックとした研究が多く行われた. 代表的なアプローチとして、連接符号(もしくは積符号)を利用したものがある. これは複数の符号を組み合わせることによって、要素となる符号の訂正機能を抑えながら、強力な誤り訂正を実現するものである. BCH 符号や RS 符号をハミング符号と併用したもの [5], BCH 符号を複数用いたもの [10], BCH 符号と TCM を用いたもの [12] が提案されている. また、P/E サイクルに加え、データ保持時間も誤りに大きく影響するようになり、優れたパフォーマンスを長く維持するため、誤り率に従い誤り訂正符号を強力なものに切り替える手法が提案されている [62, 63, 64]. NAND フラッシュメモリに適した BCH 符号の復号ハードウェアの実現方法についても、議論がなされている [13, 14, 19].

BCH 符号はまた、訂正機能の増大に伴い符号化率が悪化する問題を有している. 別の言い方をすれば、ある符号化率で達成できる訂正後の誤り率が、訂正機能の増大とともに理論限界から遠ざかってしまう. この観点での理論限界を与えるものがシャノン限界であるが、これに近い優れた誤り訂正が可能な、LDPC 符号 [65] が注目されている. LDPC 符号を NAND フラッシュメモリに適用する場合、軟判定の復号アルゴリズムが特に課題となる. このアルゴリズムでは多段に量子化された閾値電圧の情報を利用するため、センシングのための時間や必要なバッファが増大してしまう. また、このような多量なデータのコントローラへの送信は、速度低下の要因となってしまう. この問題は LDPC 符号をオンチップで実現することで解決されるが、オンチップの復号回路が増大すると、メモリそのもののコストを増大させてしまう [66]. 閾値電圧分布を考慮し、読み出し電圧を不均一に配置させ、従来と同様の誤り訂正のパフォーマンスを実現しながら、センシングの精度を下げる手法が提案されている [3]. また、電荷保持やセル干渉、P/E サイクルによる閾値電圧への影響を考慮することで、多段のセンシングを行わず必要な訂正を行う手法が提案されている [11].

MLC において発生する誤りが多元非対称誤りモデルと見なせることから、メモリに発生する誤りの特徴を考慮することで、効率の良い誤り訂正を実現する手法も提案されている. LM 誤り訂正符号は、NAND フラッシュメモリにおける多くの誤りが近

傍の値へ向かうことを考慮し、制御できる誤りの変量を限定することによって、計算量や検査ビットを削減する手法である。単一 LM 誤り訂正符号 [23, 24, 25, 26, 27, 28] と多重 LM 誤り訂正 [29, 31, 30] が、それぞれ異なったアプローチで研究されている。本論文の以降の章で提案するのも、このような誤りの特徴を考慮する符号(もしくは復号法)である。

5 提案手法 1：单一対称 LM 誤り訂正符号

5.1 概要

2.5 節で紹介したように、NAND フラッシュメモリの誤りの要因には、閾値電圧レベルの増加方向への誤り、減少方向への誤りを生じるものがある。これら両方を訂正することを考え、本章では单一対称 s -LM 誤り訂正符号を提案する [34]。提案手法は整数剩余環上の検査行列によって定義される。整数剩余環上の検査行列により定義され、近傍の値へ向かう誤りを訂正する符号として、单一 Lee 誤り訂正符号が、2 つの異なる構成法で提案されている [67, 68]。しかし、この符号は单一対称 1-LM 誤り訂正符号と一致しており、変量が 2 以上の誤りに対処できない。また、LM 誤りより広義の多元非対称誤りモデルのための符号 [59] は、対称 s -LM 誤りを訂正できるが、非効率である。提案手法はこれら 2 つの符号の中間に位置するものである。この符号の構成法は、[68] の符号の構成法に変更を加え、シンボル内の訂正可能な変量について一般化したものである。

5.2 定義と定理

検査行列 \mathbf{H} が次の条件を満たすとき、 \mathbf{H} によって定義される符号は单一誤り訂正符号となる。

定理 11 検査行列 \mathbf{H} により定義される符号 C が、单一対称 s -LM 誤り訂正符号であるための必要十分条件は、任意の異なる单一対称 s -LM 誤り $\mathbf{e}_a, \mathbf{e}_b$ のシンドローム、 $\mathbf{e}_a \cdot \mathbf{H}^T, \mathbf{e}_b \cdot \mathbf{H}^T$ について、

$$\mathbf{0} \neq \mathbf{e}_a \cdot \mathbf{H}^T \neq \mathbf{e}_b \cdot \mathbf{H}^T \neq \mathbf{0} \quad (11)$$

が成り立つことである。ここで、 \mathbf{H}^T は検査行列の転置である。

(証明) 非零のシンドロームにより誤りの検出が可能となり、シンドロームが異なることにより誤りの訂正が可能となる。 (証明終了)

また、整数剩余環における単元と零因子に関して、以下の補助定理を符号構成に用いる。

補助定理 1 Z_m の零因子でない元は、単元である。

(証明) \bar{a}, \bar{b} を、 Z_m の 2 つの異なる元とし、 \bar{k} を Z_m の零因子でない元とする。 $\bar{k}\bar{a} = \bar{k}\bar{b}$ と仮定すると、 $\bar{k}(\bar{a} - \bar{b}) = \bar{0}$ となる。 $\bar{a} - \bar{b} \neq \bar{0}$ なので、 \bar{k} が零因子となり、矛盾する。したがって、 $\bar{k}\bar{a} \neq \bar{k}\bar{b}$ である。これは $\bar{k}\bar{a}, \bar{k}\bar{b}$ が \bar{a}, \bar{b} の選び方によって、 Z_m の任意の元をとり得ることを意味する。 $\bar{k}\bar{a} = \bar{k}\bar{b} + \bar{1}$ となるように \bar{a}, \bar{b} を選ぶと、 $\bar{k}(\bar{a} - \bar{b}) = \bar{1}$ となり、 \bar{k} は単元となる。 (証明終了)

補助定理 2 Z_m の位数 m が s より大きい素数の積であるとき, $\pm\bar{1}, \pm\bar{2}, \dots, \pm\bar{s} \in Z_m$ は, すべて単元である.

(証明) p を m の最小の約数とする. $\bar{k} \in Z_m (1 \leq k \leq s < p)$ が零因子ならば, $ka = 0 \pmod{m}$ となる $\bar{a} \in Z_m (1 \leq a < m)$ が存在する. ところが k は m の約数でないので, これを満たす k は m の倍数となる. これは $1 \leq k \leq s < p$ と矛盾する. したがって \bar{k} は零因子でなく, 補助定理 1 により単元である. $-\bar{k}$ についても同様に示せる. (証明終了)

5.3 提案手法

5.3.1 符号構成

符号の構成は, 手順 1: 整数環 Z_m の位数 m の決定, 手順 2: Z_m の要素を用いて 3 つの集合を作成, 手順 3: 手順 2 の集合を用いた検査行列の定義, の 3 つの手順による.

手順 1 訂正すべき誤りの変量 s を用いて, 位数 m を定める. 符号長を大きくとるには, 訂正すべき対称 s -LM 誤りが単元となることが望ましい(シンドロームの重複を避ける)ため, 補助定理 2 より, m を s より大きい素数の積とする. すなわち,

$$m = p_1^{s_1} \cdot p_2^{s_2} \cdots \cdot p_u^{s_u}. \quad (12)$$

ただし, $1 \leq i \leq u$ について, p_i は $s < p_1 < p_2 < \cdots < p_u$ を満たす素数とし, $s_i \geq 1$ とする.

M 元の情報に対して誤り訂正を行う場合, $M = m$ とする m が存在する場合, 符号化は単純に実行できる. しかしながら, 例えば $M = 2^b (b \geq 2)$ の情報に対して, $l = 2$ の誤り訂正を行う場合は, $p_1 \geq 3$ である必要があり, $m = 2^b$ とすることができない. この場合の符号化は, 1) m を $m < M$ かつ式 (12) を満たす最大の整数とし, 2) M 元の情報を $\mathbf{d} = (d_0, d_1, \dots, d_{k-1})$ としたとき, $\mathbf{d} \pmod{m}$ を情報と見なし検査シンボル \mathbf{c} を求め, 3) $\bar{a} \in Z_m$ は $\bar{a} \in Z_M$ へと写像し, 元の M 元の情報に付加したもの (\mathbf{d}, \mathbf{c}) を符号語とする. この場合も, LM 誤り訂正機能は保持される(ただし, 値が最小値から最大値へ巡回するような誤りを考慮する, 狹義の LM 誤り [29] は訂正不可となる). $m < M$ の場合の符号化の手順を図 12 に示す.

手順 2 Z_m の部分集合となる 3 つの集合, L , O , U を求める.

集合 L : 誤りの値を区別する役割を果たす集合である. すなわち, 集合 $A = Z_m \setminus \{0\}$, $E = \{\pm 1, \pm 2, \dots, \pm s\} \subset Z_m$ としたとき, 任意の異なる $x, y \in A$, お

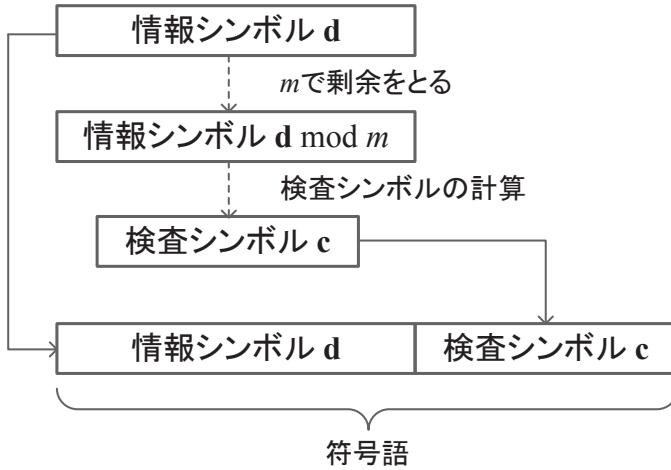


図 12: $m < M$ の場合の符号化手順.

より任意の異なる $a, b \in E$ について,

$$\begin{cases} ax \neq bx, \\ ax \neq ay, \\ ax \neq by, \end{cases} \quad (13)$$

を満たす x, y から成る集合を L と定義する. この L は $B[\pm s](m)$ 集合 (E の定義を変更すれば, より一般に $B_1(E)(m)$ 集合) であり, 従来の構成法が利用できる. しかしながら, 元の数が最大となる $B[\pm s](m)$ の構成法は限られた s, m についてのみ知られており, 手順 1 の方法で得られた m について, 構成法が知られていない場合がある. この場合, L 得るために次のアルゴリズムを利用する.

アルゴリズム 1 (L の構成アルゴリズム) 以下の手順による.

1. $L = \emptyset, A = Z_m \setminus \{0\}, B = \emptyset, E = \{\pm 1, \pm 2, \dots, \pm s\} \subset Z_m$ とする.
2. $B = Z_m \setminus \{0\}$ のとき, アルゴリズムを終了する.
3. x として $x \in A$ かつ $x \notin B$ となる任意の元を選ぶ.
4. 任意の異なる $a, b \in E$ について, $xa = xb$ となるような a, b が存在する場合, $B \cup \{x\}$ を新たに B とおき, 2. に戻る.
5. $xE \not\subseteq A$ のとき, $B \cup \{x\}$ を新たに B とおき, 2. に戻る.
6. $xE \subseteq A$ のとき, $A \setminus (xE)$ を A , $B \cup (xE)$ を B と新たにとおく.
7. $L \cup \{x\}$ を新たに L とおき, 手順 2. に戻る.

このアルゴリズムは任意の s, m について利用可能である。 $|L|$ を最大化することを保障しないが、 $B[\pm s](m)$ の構成法が知られていない s, m についても、 L を得ることができる。得られる集合はアルゴリズムにおける x の選び方に依存する。例えば $s = 4, m = 31$ のとき、 $L = \{1, 5, 6\}$ および $L = \{3, 8\}$ は、ともにこのアルゴリズムにより得られる集合である。 x の選び方を変更してアルゴリズムを繰り返せば、最終的に全探索となり、元の数最大の集合が得られる。完全または準完全の L が得られれば、その時点で元の数が最大であることが保障される。例えば $s = 4$ のとき、手順 1 に従えば $m = 31$ である。このとき完全な L は存在せず、準完全な L の構成法は知られていない。上記アルゴリズムにおいて A の元のうち、つねに最小のものを x として得られる $L = \{1, 5, 6\}$ は、 $\lfloor(m-1)/2s\rfloor = 3 = |L|$ となり、準完全な $B[\pm 4](31)$ を与えている。理論的な構成は一般の s, m について結果を得られるため重要であるが、現行の NAND フラッシュメモリは $m \leq 16$ 程度であり、実用上は全探索が可能である。

例 5 $m = 15, s = 2$ の場合のアルゴリズムの利用例を示す。

1. $L = \phi, A = \{1, 2, \dots, 14\}, B = \phi, E = \{\pm 1, \pm 2\}$ とする。
 2. $x = 1$ とする。このとき、 $1E = \{1, 2, 13, 14\}$ である。
 3. $A \setminus (1E) = \{3, 4, \dots, 12\}$ を新たに A とおく、 $B \cup (1E) = \{1, 2, 13, 14\}$ を新たに B とおく。
 4. $L = \phi \cup \{1\} = \{1\}$ を新たに L とおく。
 5. $x = 3$ とする。このとき、 $3E = \{3, 6, 9, 12\}$ である。
 6. $A \setminus (3E) = \{4, 5, 7, 8, 10, 11\}$ を新たに A とおく、 $B \cup (3E) = \{1, 2, 3, 6, 9, 12, 13, 14\}$ を新たに B とおく。
 7. $L = \{1\} \cup \{3\} = \{1, 3\}$ を新たに L とおく。
 8. $x = 4$ とする。このとき、 $4E = \{4, 7, 8, 11\}$ である。
 9. $A \setminus (4E) = \{5, 10\}$ を新たに A とおく、 $B \cup (4E) = \{1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14\}$ を新たに B とおく。
 10. $L = \{1, 3\} \cup \{4\} = \{1, 3, 4\}$ を新たに L とおく。
 11. $x = 5$ とする。このとき、 $5 \cdot 2 = 5 \cdot (-1) = 10$ であるため、 $B \cup \{5\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14\}$ を新たに B とおく。
 12. $x = 10$ とする。このとき、 $10 \cdot 1 = 10 \cdot (-2) = 10$ であるため、 $B \cup \{10\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$ を新たに B とおく。
 13. $B = Z_m \setminus \{0\}$ なので、アルゴリズムを終了する。
- $\lfloor(15-1)/4\rfloor = 3$ であるので、 L は準完全な $B[\pm 2](15)$ である。

集合 O : この集合は, m の素因数およびその倍数から成る集合で, かつ L の元と誤りとの積が含まれないものである. m の全ての素因数から成る Z_m の部分集合を P とし, $E = \{\pm 1, \pm 2, \dots, \pm s\} \subset Z_m$ とする.

$$O_0 = \{xp : p \in P, x \in Z_m\},$$

$$O_1 = \{ey : y \in L \cap O_0, e \in E\}$$

としたとき, O を次のように定義する :

$$O = O_0 \setminus O_1.$$

ただし, m が素数の場合 $O = \{0\}$ とする. O は, 以下の補助定理で述べられるような性質をもつ.

補助定理 3 $l \in L, e \in E$ のとき, $el \notin O$ となる.

(証明) l が単元のとき, el が零因子だと仮定すると, $(el)a = 0$ を満たす非零の $a \in Z_m$ が存在する. $e(la) = 0$ とすれば, $la \neq 0$ だから, e が零因子となり矛盾する. したがって, el は単元である. O は O_0 の定義より零因子から成る集合であるため, $el \notin O$.
(証明終了)

補助定理 4 $o \in O, e \in E$ のとき, $eo \in O$ となる.

(証明) 任意の $x \in Z_m$ について, $q \in Z_m$ を単元とすると,

$$\{kx : k \in Z_m\} = \{kqx : k \in Z_m\} \quad (14)$$

となる. これは左辺において, $k = k'q^{-1}, k' \in Z_m$ とおけば, 明らかである. m の素因数を, p_1, p_2, \dots, p_n とする. $l \in L$ が m の素因数と $x \in Z_m$ との積であるならば, 単元 $q \in Z_m$ を用いて, $l = qp_{i_1}p_{i_2} \cdots p_{i_n}$ と表せ, O_1 は l とその倍数を含む. 式(14)より, $q' \in Z_m$ について, $o_1 = q'p_{i_1}p_{i_2} \cdots p_{i_n} \in O_1$ が成り立つ. $k \in Z_m$ について, $o \in O$ は $o = kp_i$ かつ $o \neq o_1$ を満たす. $eo = e(kp_i) = o_1$ となるには, e は $k' \in Z_m$ について, $e = k'p_j (j \in \{i_1, i_2, \dots, i_n\})$ となる必要がある. $1 \leq e \leq s$ のとき, $s < p_j$ であるため矛盾する. したがって, $eo \notin O_1$ かつ $eo \in O_0$ であるから, $eo \in O$ である. $-s \leq e \leq -1$ についても同様である $l \in L$ が単元ならば, 明らかに $l \notin L \cap O_0$ であるため, O はこのような l の存在に依存しない.
(証明終了)

集合 U : Z_m の元すべてから成る集合 $U = Z_m$ とする.

手順3 検査行列 \mathbf{H} を定義する. 検査長を r , $N_i \leq |O|^i |L| |U|^{r-i}$ とし

$$\begin{aligned}\mathbf{h}_{i,j} &\in \{(o_1, o_2, \dots, o_{i-1}, l, u_1, u_2, \dots, u_{r-i})^T : o_x \in O, l \in L, u_y \in U\}, \\ \mathbf{H}_i &= [h_{i,1}, h_{i,2}, \dots, h_{i,j}, \dots, h_{i,N_i}]\end{aligned}$$

とする. ただし, $j \neq j'$ のとき, $\mathbf{h}_{i,j} \neq \mathbf{h}_{i,j'}$ とする. 検査行列を

$$\mathbf{H} = [\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_r] \quad (15)$$

によって定義する.

定理 12 式 (15) の検査行列 \mathbf{H} の直行空間によって定義される符号 C は, 単一対称 s -LM 誤り訂正符号である.

(証明) $e, e' \in Z_m$ を単一対称 s -LM 誤りの非零の要素とする. e が符号語における, 検査行列の列ベクトル $\mathbf{h}_{i,j}$ に対応する要素であった場合のシンドロームを $\mathbf{s} = e\mathbf{h}_{i,j}$ とし, 同様に e' が $\mathbf{h}_{i',j'}$ に対応する要素であった場合のシンドロームを $\mathbf{s}' = e'\mathbf{h}_{i',j'}$ とする.

1. $i \neq i'$ の場合

$i < i'$ と仮定する. \mathbf{s} の i 番目の要素は L の元 l と e の積となる. よって補助定理 3 より,

$$s_i = el \notin O. \quad (16)$$

となる. 一方, \mathbf{s}' の第 i 要素 s'_i は, e' と O の元 o との積である. よって補助定理 4 より,

$$s'_i = e'o \in O \quad (17)$$

となる. したがって, $s_i \neq s'_i$ となる.

2. $i = i'$ の場合

(a) $e \neq e'$ の場合

s_i は $l \in L$ と e の積 el となり, s'_i は $l' \in L$ と e' の積 $e'l'$ となる. L の元は (13) を満たすため,

$$s_i = el \neq e'l' = s'_i. \quad (18)$$

(b) $e = e'$ の場合 e は単元であるから, 乗法逆元が存在する. $\mathbf{s} = \mathbf{s}'$ を仮定すると,

$$\mathbf{h}_{i,j} = e^{-1}\mathbf{s} = e^{-1}\mathbf{s}' = \mathbf{h}_{i,j'}. \quad (19)$$

列ベクトルの一貫性は構成法と矛盾する.

以上により, $(i, j, e) \neq (i', j', e')$ のとき, $s \neq s'$ が満たされる. したがって, 定理 11 より, \mathbf{H} によって定義される符号は, 単一対称 l -LM 誤り訂正符号である.

(証明終了)

各 \mathbf{H}_i に含まれる列ベクトルの数は N_i であるため, 符号シンボル長 N は,

$$N = \sum_{i=0}^r N_i \leq |L| \frac{|U|^r - |O|^r}{|U| - |O|}$$

によって求められる.

L の構成アルゴリズムは, $m \geq 2s + 1$ のとき, 1 を要素として選ぶことが可能であるため, 必ず $L \neq \phi$ となる. $|L| \geq 1, |O| \geq 1$ であるため, 少なくとも $N \geq (|U|^r - 1)(|U| - 1)$ を満たす符号が構成できる. $m \leq 2s$ のとき, 式(13)を満たす O は存在しない. $1 \in L$ のとき検査行列は既約標準形となり, 検査シンボルが自明となる. これは L の構成アルゴリズムにおいて, 最初の x として 1 を選ぶことにより, 実現できる. また $s = 1$ とし, L の構成アルゴリズムにおいて常に最小の $x \in A$ を選ぶことにより, 従来の単一対象 1-LM 誤り訂正符号 [68] と一致する符号を得ることができる. $O = \{0\}$ のとき, 検査行列 \mathbf{H} は従来の単一 LM 誤り訂正符号 [25, 26] の検査行列の構成法と一致するため, これを拡張したものとなっている. L が完全な $B[\pm s](m)$ 集合でないとき, $|O| \geq 2$ となる場合があり, このとき従来より符号長が大きくなる.

例 6 $M = 16$ の場合に, 検査長 $r = 2$ の単一対象 2-LM 誤り訂正符号を構成する.

手順 1: m は 2 より大きい素数の積であり, かつ $m \leq M = 16$ を満たす最大の整数である. この場合, $16 = 2^4$ は m として用いることができないので, $m = 3 \cdot 5 = 15$ とする.

手順 2: この場合の L は, 例 5 のものを用いることができ, $L = \{1, 3, 4\}$ とする. 続いて O を求める. O_0 は m 以下の 3 の倍数, および 5 の倍数から成る. また, O_1 は $L \cap O_0 = \{3\}$ なので, $O_1 = \{3 \cdot (\pm 1), 3 \cdot (\pm 2)\} = \{3, 6, 9, 12\}$ となる. したがって, $O = O_0 \setminus O_1 = \{0, 5, 10\}$ となる.

手順 3: 検査行列 \mathbf{H} の列ベクトルは, $l \in L, o \in O, u \in U$ としたとき,

$$\begin{pmatrix} l \\ u \end{pmatrix} \text{ または } \begin{pmatrix} o \\ l \end{pmatrix}$$

となるものである. 最終的に, 符号長 $N = 54$ の符号

$$\mathbf{H} = \left[\begin{array}{cccccc|cccccc} 1 & \cdots & 1 & 3 & \cdots & 3 & 4 & \cdots & 4 & 0 & 0 & 0 & 5 & 5 & 5 & 10 & 10 & 10 \\ 0 & \cdots & 14 & 0 & \cdots & 14 & 0 & \cdots & 14 & 1 & 3 & 4 & 1 & 3 & 4 & 1 & 3 & 4 \end{array} \right]$$

を得る.

例 7 $M = 8$ の場合に, 検査長 $r = 2$ の单一対象 1-LM 誤り訂正符号を構成する.

手順 1 : m は 1 より大きい素数の積 (つまり任意の整数) である. $m = M = 8$ とする.

手順 2 : L の構成アルゴリズムを用いるが, ここでは常に最小の $x \in A$ を選ぶこととする. このとき, $L = \{1, 2, 3\}$ を得る. $O_0 = \{0, 2, 4, 6\}$ であり, $L \cap O_0 = \{2\}$ である. $O_1 = \{2 \cdot (\pm 1)\} = \{2, 6\}$ なので, $O = O_0 \setminus O_1 = \{0, 4\}$ となる.

手順 3 : 符号長 $N = 30$ の符号

$$\mathbf{H} = \left[\begin{array}{cccccc|cccccc} 1 & \cdots & 1 & 2 & \cdots & 2 & 3 & \cdots & 3 & 0 & 0 & 0 & 4 & 4 & 4 \\ 0 & \cdots & 7 & 0 & \cdots & 7 & 0 & \cdots & 7 & 1 & 2 & 3 & 1 & 2 & 3 \end{array} \right]$$

を得るが, これは [68] の符号と一致している.

5.3.2 誤り検出機能の付加

$p_1 = s + 1$ の時, 単一対称 $(s + 1)$ -LM 誤りの検出機能を付加することができる. これは集合 L, O に代わり, 以下に定義する L', O' を用いることで実現される. この場合, O' は L' より先に求める必要があることに注意する.

集合 $O' : Z_m$ の全ての零因子および 0 から成る集合とする.

集合 $L' : L'$ は $B[\pm s](m)$ 集合であり, かつ次の条件を満たすものである.

$$\{el' : l' \in L', -s \leq e \leq s\} \cap \{e'l' : l' \in L', e' = \pm(s + 1)\} = \emptyset.$$

このような集合の構成法は知られていないが, L の構成アルゴリズムに変更を加えた, 次のアルゴリズムによって求めることができる.

アルゴリズム 2 (L' の構成アルゴリズム) 以下の手順による.

1. $L' = \emptyset, A = Z_m \setminus \{0\}, B = \emptyset, E = \{\pm 1, \pm 2, \dots, \pm s\} \subset Z_m$ とする.
2. $B = Z_m \setminus \{0\}$ のとき, アルゴリズムを終了する.
3. x として $x \in A$ かつ $x \notin B$ となる任意の元を選ぶ.
4. (a) $x \in O'$ の場合, $B \cup \{x\}$ を新たに B とおき, 2. に戻る.
(b) 任意の異なる $a, b \in E$ について, $xa = xb$ となるような a, b が存在する場合, $B \cup \{x\}$ を新たに B とおき, 2. に戻る.
5. $xE \not\subseteq A$ のとき, $B \cup \{x\}$ を新たに B とおき, 2. に戻る.
6. $xE \subseteq A$ のとき, $A \setminus (xE)$ を A , $B \cup (xE)$ を B と新たにとおく.
7. $L' \cup \{x\}$ を新たに L' とおき, 手順 2. に戻る.

このアルゴリズムは、 L の構成アルゴリズムと同様に、 $|L'|$ を最大化するものではないため、元 x の選びかたを変更しながら、アルゴリズムを繰り返すことにより、より優れた L' を得られる場合がある。 L' は完全な $B[\pm s](m)$ になり得ず、また準完全な $B[\pm s](m)$ であれば、元の数が最大であることが保障される。

定理 13 L, O を L', O' にそれぞれ置き換え、同様に手順 1～手順 3 によって構成した符号は、單一対称 s -LM 誤り訂正一対称 $(s+1)$ -LM 誤り検出符号である。

(証明) 誤り訂正については、定理 12 と同様に証明できる。訂正する誤りの非零の要素を e とし、検出する誤りの非零の要素を $e' = \pm(s+1)$ とする。任意の $l' \in L'$ と e は単元であるため、訂正する誤りのシンドローム s の i 番目の要素 $s_i = el' \notin O$ であり、検出する誤りのシンドローム s' の i 番目の要素 $s'_i = e'l' \in O$ 、かつ $s'_i \neq 0$ である。よって訂正と検出は $s \neq s'$ によって区別され、かつ $s' \neq 0$ であるため、訂正と検出の両方が可能となる。
(証明終了)

符号シンボル長 N は検出機能を付加しない場合同様に、

$$N \leq |L'| \frac{|U|^r - |O'|^r}{|U| - |O'|}$$

によって求められる。

例 8 $M = 8$ の場合に、検査長 $r = 2$ の單一対象 1-LM 誤り訂正一対象 2-LM 誤り検出符号を構成する。

手順 1 : m は 1 より大きい素数の積 (つまり任意の整数) である。 $m = M = 8$ とする。

手順 2 : $O' = \{0, 2, 4, 6\}$ を求めてから、 L' の構成アルゴリズムを用いる。一例として $L' = \{1, 3\}$ を得たとする。

手順 3 : 符号長 $N = 24$ の符号

$$\mathbf{H} = \left[\begin{array}{cccccc|cccccc} 1 & \cdots & 1 & 3 & \cdots & 3 & 0 & 0 & 2 & 2 & 4 & 4 & 6 & 6 \\ 0 & \cdots & 7 & 0 & \cdots & 7 & 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 \end{array} \right] \quad (20)$$

を得る。

5.3.3 復号法

$E = \{\pm 1, \pm 2, \dots, \pm l\} \subset Z_m$ とし、 \leftarrow を代入とする。

アルゴリズム 3 (復号アルゴリズム) 以下の手順による。

1. 受信語 \mathbf{y} を用いて、シンドローム $\mathbf{s} = \mathbf{y} \cdot \mathbf{H}$ を求める。ただし $m < M$ の場合は、 $\mathbf{y}' = \mathbf{y} \bmod m$ を用いて、 $\mathbf{s} = \mathbf{y}' \cdot \mathbf{H}$ により求める。

2. $\mathbf{s} = \mathbf{0}$ ならば, 誤りなしとみなしアルゴリズムを終了する.
3. $i \leftarrow 1$ とする.
4. $s_i \in O$ ならば, $i \leftarrow i + 1$ とする.
5. 4. を $s_i \notin O$ が満たされるまで繰り返す. すべての $i(i \leq r)$ について $s_i \in O$ ならば, 対称 $(s+1)$ -LM 誤りを検出し, アルゴリズムを終了する.
6. $el = s_i$ となる $l \in L$ および $e \in E$ を探す.
7. $\mathbf{s}^* = e^{-1}\mathbf{s}$ を求める.
8. 値 e の誤りが, $\mathbf{s}^* = \mathbf{h}_{i,j}$ となる \mathbf{H} の列ベクトル, $\mathbf{h}_{i,j}$ に対応する位置に発生していると見なす. 訂正語 \mathbf{w}' を $\mathbf{w}' = \mathbf{y} - \mathbf{e}$ によって求め, アルゴリズムを終了する.

式 (13) によって, l, e の組は唯一に定まることが保障されている. また, $|L||E| < |Z_m| = m$ が満たされるため, l, e の組は m より少ない. MLC の NAND フラッシュメモリにおいて, $m \leq M = 2^b$ としたとき, 手順 6. において総当たりで探索を行う場合, 2^b に比例する程度の計算量を要する. b に関する指數関数的な増加であるものの, 現行のメモリでは $b \leq 4$ と小さく, 実現は困難ではない. この復号アルゴリズムは, L, O を L', O' に置き換えれば, 検出機能を付加した場合にも利用できる.

例 9 式 (20) によって定義される, 単一 1-LM 誤り検出—2-LM 誤り訂正符号を用いた, 復号の例を示す. $L' = \{1, 3\}, O' = \{0, 2, 4, 6\}$ である. シンドロームとして, $\mathbf{s} = (6, 5)^T$ が得られたとする. $6 \in O'$ かつ $5 \notin O'$ なので, $el = 5$ となる e, l の組を探すると, $e = 7, l = 3$ であることが分かる. $\mathbf{s}^* = e^{-1}\mathbf{s} = 7(6, 5)^T = (2, 3)^T$ が得られるので, 列ベクトル $(2, 3)^T$ に対応する位置に, $e = 7 = -1$ の誤りが発生している. 受信語の対応する要素から, (-1) を減算することにより, 復号を終了する. 次に, シンドロームとして $\mathbf{s} = (2, 4)^T$ が得られたとする. 全要素が O' の元かつ零ベクトルでないので, 誤りを検出し復号を終了する.

5.4 評価

5.4.1 情報長と検査長

提案手法と従来の单一誤り訂正符号について, 検査シンボル長に対する最大の情報シンボル長を比較した. 表 6, 表 7, 表 8 は, メモリセルのレベル数 M がそれぞれ 8, 16, 32 の場合を対象に, 符号を構成した場合のものである. 提案手法の構成において, L は完全または準完全な $B[\pm s](m)$ 集合を用い, L' は構成アルゴリズムを用いた全探索を想定した.

表 6: 情報シンボル長の比較 ($M = 8$).

	検査シンボル長			
	2	3	4	5
従来 1	26	247	2036	16369
従来 2	28	249	2036	16363
従来 3	7	70	581	4676
従来 4($m = 7$)	6	54	400	2801
提案 1	28	249	2036	16363
提案 2	22	221	1916	15867
提案 3($m = 7$)	6	54	400	2801
提案 4	構成不可			

表 7: 情報シンボル長の比較 ($M = 16$).

	検査シンボル長			
	2	3	4	5
従来 1	120	2036	32752	524268
従来 2	124	2041	32756	524267
従来 3	15	270	4365	69900
従来 4($m = 15$)	46	720	10844	162718
提案 1	124	2041	32756	524267
提案 2	94	1789	30716	507899
提案 3($m = 15$)	52	834	12632	189778
提案 4($m = 15$)	42	755	12052	185637

従来 1 : グレイ符号とハミング符号を用いた單一対称 1-LM 誤り訂正符号 [69]

従来 2 : 整数剩余環を用いた單一対称 1-LM 誤り訂正符号 [68]

従来 3 : 多元の單一対称シンボル誤り訂正ハミング符号 [22]

従来 4 : 単一対称 2-LM 誤り訂正符号 [25, 26]

提案 1 : 単一対称 1-LM 誤り訂正符号

提案 2 : 単一対称 1-LM 誤り訂正一対称 2-LM 誤り検出符号

提案 3 : 単一対称 2-LM 誤り訂正符号

提案 4 : 単一対称 2-LM 誤り訂正一対称 3-LM 誤り検出符号

$M = 8$ の場合を除いて、提案手法は多元のハミング符号(従来 3)より大きい情報シンボル長を持つ。提案 3 と従来 4 は、完全な $B[\pm s](m)$ が構成可能なとき、符号そのものが一致する。完全な $B[\pm s](m)$ が構成不可な場合は、集合 O の要素数を 2 以

表 8: 情報シンボル長の比較 ($M = 32$).

	検査シンボル長			
	2	3	4	5
従来 1	502	16369	524268	16777191
従来 2	508	16377	524276	16777195
従来 3	31	1054	33821	1082396
従来 4($m = 31$)	190	5955	184700	5725825
提案 1	508	16377	524276	16777195
提案 2	382	14333	491516	16252923
提案 3($m = 31$)	190	5955	184700	5725825
提案 4($m = 27$)	142	4209	116636	3175519

上とできる場合があり、これにより情報シンボル長の改善が可能である。今回評価したパラメータでは、 $M = 16$ としたときに情報シンボル長が改善されている。

$M = 16$ の場合、対称 2-LM 誤りが発生し得るならば、対称 1-LM 誤り訂正符号である従来 1 および従来 2 は、この誤りを訂正できない。この場合、シンボル誤り訂正符号である従来 3、または対称 2-LM 誤り訂正符号である提案 2 を用いればよい。検査シンボル長が 2 の場合で比較すると、従来 3 は最大で 15 の情報シンボルまでの訂正が可能であるのに対し、提案 2 は最大で情報シンボルを 52 まで訂正が可能である。したがって、提案手法は従来より優れた誤り訂正符号と言える。検出機能の有無で比較すると、 $M = 16$ 、検査シンボル長が 2 の場合、提案 3 の情報シンボル長は 52 であるのに対し、対称 3-LM 誤りの検出が可能な提案 4 では 42 となり、検出を行うと情報シンボル長がいくらか小さくなる。同様なことが $M = 32$ の場合や、2 以上の検査シンボル長に対しても言える。また、従来 2 と提案 1 はパラメータによらず同じ情報シンボル長となっているが、これは提案手法が従来の符号の一般化となつておらず、符号そのものが一致しているためである。

提案手法である单一対称 2-LM 誤り訂正符号と、LM 誤りより広義の非対称誤りを訂正可能な多元非対称誤り訂正符号 [59] を、单一対称 2-LM 誤り訂正を訂正できるよう構成した場合の、情報シンボル長の比較を行った。表 9 に $M = 16$ 、表 10 に $M = 32$ として符号を構成した場合を示す。また、单一対称 2-LM 誤り訂正符号が取り得る情報シンボル長の理論限界 [59] も並べて示す。例えば表 9において、検査シンボル長が 2 の場合に、多元非対称誤り訂正符号では情報長は 23 となるが、提案手法では情報長は 52 と 2 倍以上の改善がなされており、限界値である 63 に近づく。多元非対称誤り訂正符号は LM 誤りを含め、より一般化されたモデルに基づいた誤り訂正が可能であるが、対称 LM 誤り訂正については、提案手法がより限界値に近い情報シンボル長を実現でき、優れていると言える。

表 9: 単一対称 2-LM 誤り訂正符号の情報長 ($M = 16$).

検査シンボル長	多元非対称誤り訂正 [59]	提案手法	限界値 [59]
2	23	52	63
3	438	834	1023
4	7221	12632	16383
5	116276	189778	262143

表 10: 単一対称 2-LM 誤り訂正符号の情報長 ($M = 32$).

検査シンボル長	多元非対称誤り訂正 [59]	提案手法	限界値 [59]
2	43	190	255
3	1530	5955	8191
4	49721	184700	262143
5	1596216	5725825	8388607

5.4.2 ビット誤り率と符号長

提案手法で構成した単一対称 2-LM 誤り訂正符号 ($M = 16$) を用いた場合の、訂正不可能な誤り率 UBER を表 11 に示す。ただし [8] の結果を考慮し、2 レベルの変化を起こす誤りが、誤り全体の 5% を占め、その他の誤りは 1 レベルの変化を起こすものとし、訂正前のシンボル誤り率をビット誤り率 RBER に換算している。

ストレージシステムに要求される UBER は $10^{-13} \sim 10^{-16}$ であるが [20]、提案手法は $10^{-6} \sim 10^{-4}$ の RBER (10^{-4} は 30~40nm の 2 ビットセルのメモリにおける、3000P/E サイクルでの実験値に相当 [8]) に対し、要求される UBER を達成できない。これは単一誤り訂正是機能として不十分であることを意味している。実用的な UBER を確保するには、連接符号のような手法を用いる必要がある。

5.4.3 復号誤り率

提案手法の様々なパラメータにおける最大の符号長をとった符号について、訂正(または検出)可能な誤りの変量を超える、すべての誤りを復号した場合の、検出、誤訂正、未検出の確率を表 12 に示す。誤りが単元の場合は、誤訂正が必ず発生する。また、 Z_{16} 上の単一 1-LM 誤り訂正 (126, 124) 符号では変量が $s + 1 = 2$ の誤りの検出率が 4.8%， Z_{15} 上の単一 1-LM 誤り訂正 (54, 52) 符号では変量が $s + 1 = 3$ の誤りの検出率が 0% であるように、検出機能を付加しない場合、対象の誤りの検出はほとんどできない。

5.5 この章のまとめ

単一対象 s -LM 誤り訂正符号を提案した。 s はパラメータとして設定でき、対象 $(s + 1)$ -LM 誤り検出機能を必要に応じて追加可能である。 $s = 1$ とすれば従来の整

表 11: $M = 16$ の单一対称 2-LM 誤り訂正符号を用いた場合の UBER.

		RBER		
符号長	情報長	1.00×10^{-6}	1.00×10^{-5}	1.00×10^{-4}
54	52	2.50×10^{-11}	2.49×10^{-9}	2.47×10^{-7}
34	32	1.59×10^{-11}	1.59×10^{-9}	1.58×10^{-7}
18	16	8.67×10^{-12}	8.67×10^{-10}	8.65×10^{-8}

表 12: 訂正可能な変量を超える誤りの検出, 誤訂正, 未検出の確率.

Z_{16} 上の单一 1-LM 誤り訂正 (126, 124) 符号				Z_{16} 上の单一 1-LM 誤り訂正 —2-LM 誤り検出 (96, 94) 符号			
誤りの値	検出	誤訂正	未検出	誤りの値	検出	誤訂正	未検出
± 2	4.8%	95.2%	0%	(± 2)	100%	0%	0%
± 3	0%	100%	0%	± 3	0%	100%	0%
± 4	19.0%	76.1%	4.8%	± 4	100%	0%	0%
± 5	0%	100%	0%	± 5	0%	100%	0%
± 6	4.8%	95.2%	0%	± 6	100%	0%	0%
± 7	0%	100%	0%	± 7	0%	100%	0%
± 8	76.2%	0%	23.8%	± 8	100%	0%	0%
Z_{15} 上の单一 1-LM 誤り訂正 (54, 52) 符号				Z_{15} 上の单一 1-LM 誤り訂正 —3-LM 誤り検出 (44, 42) 符号			
誤りの値	検出	誤訂正	未検出	誤りの値	検出	誤訂正	未検出
± 3	0%	100%	0%	(± 3)	100%	0%	0%
± 4	0%	100%	0%	± 4	4.3%	95.7%	0%
± 5	89.9%	0%	11.1%	± 5	100%	0%	0%
± 6	0%	100%	0%	± 6	100%	0%	0%
± 7	0%	100%	0%	± 7	4.3%	95.7%	0%

数剰余環上の符号 [68] に一致し, この符号を一般化したものに相当する. 多元のハミング符号, 従来の対称 LM 誤り訂正符号, 多元非対称誤り訂正符号より情報長を大きくとれることが示され, したがって従来より効率が良い誤り訂正が可能である. 従来の対称 LM 誤り訂正符号からの改善点は, アルゴリズムを用いた $B[\pm s](m)$ 集合の構成についての議論, 完全な $B[\pm s](m)$ 集合となる L が得られなかった場合に符号長を改善する, 検査行列の構成法の変更, 大きな変量を持つ誤りへの検出機能の付加が挙げられる. 今後の課題として, L の構成アルゴリズムによって得られる元の数の, 上限や下限についての議論や, より多くの s, m に対する $B[\pm s](m)$ 集合の明示的な構成が挙げられる. また, 単一誤り訂正是 MLC の NAND フラッシュメモリの UBER の要件を満たすには不十分であり, 実用的な誤り率を達成可能とする符号化法の考案が挙げられる.

6 提案手法 2：マルチページプログラミングと近傍値への誤りを考慮した誤り訂正

6.1 概要

近傍の値へ集中する誤りを効率的に訂正する手法として、前章においても提案したLM誤り訂正符号の利用が考えられる。しかし、非対称1-LM誤り訂正符号を除き、LM誤り訂正符号は下位のビットが属する複数ページの情報を符号化に必要とする。2.4節で示したように、この場合はマルチページプログラミングと併用できず、従来の書き込み手法を利用するならば、その速度は1/2.3と大きく低下してしまう[35]。

近傍値の誤りを訂正する他の手法として、グレイ符号[70]と2元の誤り訂正符号の併用が挙げられる[9, 18]。グレイ符号は整数値における隣接1レベルの変化に対応するビット反転が、すべて1ビットとなる数値の表現方法であり、このため隣接1レベルへの誤りが発生したときの誤りビット数を最小化できる。このため、2元の誤り訂正符号と併用すれば、必要な訂正機能を小さく抑えることができる。しかしながら、NANDフラッシュメモリでは5%程度のセル誤りが2レベルの変化を生じる例が報告されている[8]。次節で示す通り、このような多レベルの誤りが発生する場合、グレイ符号と2元の誤り訂正符号の単純な併用は、最適な誤り訂正とは言えない。

本章での提案手法は、マルチページプログラムを採用する従来手法と同様に、全てのページに対し、それぞれビット誤り訂正符号を適用する。復号アルゴリズムを変更し、誤りの検出は従来同様2元の符号で行うが、訂正には多元の整数値を用いる。この際に、誤りモデルから推測される、元の値である可能性が最も高いものに訂正を行う。

6.2 グレイ符号と誤り訂正

データがメモリセルから読みだされた際に、閾値電圧レベルに対応する整数値が、2元の系列に写像される。このとき、通常の2進数への写像を用いると、隣接する整数値に対応する2元の系列間において、複数ビット反転を生じる場合がある。一方、グレイ符号を利用した写像を用いれば、隣接する整数値間のビット反転は、必ず1ビットになる。表13に8元の場合の2進数、グレイ符号を用いた写像を示す。2進数を用いた場合、整数値3と4の間の反転が3ビットとなる一方、グレイ符号では隣接する整数であれば、その選び方によらず1ビットの反転となる。この特徴は2元の誤り訂正符号と併用したときに、隣接値への誤りの訂正に必要な符号の機能を最小化する。また併用する誤り訂正符号が同じであれば、誤り率のパフォーマンスが最適化される。

表 13: 8 元の整数値から 2 元のビット列への写像.

整数値	2 進数	グレイ符号
0(消去)	111	111
1	110	110
2	101	100
3	100	101
4	011	001
5	010	000
6	001	010
7	000	011

しかし、多レベル誤りが起こり得る場合、従来のビット反転による復号は最適ではなくなる。具体例を用いて説明する。8 元の 1 つのシンボルに、整数値 3 から 5 へ向かう誤りが発生し、これをシンボルを構成する 3 ビットそれぞれに適用した 2 元の符号で訂正する。上位のビット（表 13 における左側のビット）から訂正することを考える。グレイ符号において 5 を表す系列 000 は、上位ビットに誤りを含むため、訂正され 100 となる。これは整数値 2 に相当する。次に中位のビットの訂正を行うが、このビットに誤りは含まれていない。最後に下位のビットの訂正であるが、100 は誤りを含んでいるため反転して 101 となり、元の値 3 に訂正される。訂正を完了するために、符号による誤りの検出を上下ビットで実行した。ここで、誤りは近い値へ向う可能性が最も高いことを考慮した復号を考える。上位のビットの訂正の段階でビット誤りが生じていることを検出したら、整数値 5 の系列と比較し上位のビットが反転している整数値 0, 1, 2, 3 の中で、最も 5 に近い整数値 3 に対応する 101 に、直接訂正する。そしてこの場合、下位のビットの誤り訂正是必要なくなり、下位のビットの誤り訂正符号への負荷が軽減される。

6.3 提案手法

提案手法では、マルチページプログラミングと併用する従来の符号構成を想定する。すなわち、セルに属する各々のビットは異なるページに属し、各ページは t ビット誤り訂正符号（例えば BCH 符号）で符号化されていることを想定する。NAND フラッシュメモリの符号化において、ページをいくつかのセクタに分割し、それぞれに誤り訂正符号を適用する手法がある [13]。簡単のためここではそのような符号構成については議論しないが、提案手法は同様に適用可能である。

次に具体的な復号アルゴリズムを示す。書き込みと符号化はページ単位で行う一方で、2.4 節で示した通り、誤り訂正はオンチップで行う場合にも、セルに属するビット全ての情報を利用可能である。提案手法は、多元のセルの誤りを 2 元の符号を用いて検出し、最も元の値である確率の高い近傍値へ訂正するというアイデアに基づ

く。このようにすることで、ある下位のページの誤りを、それより上位のページの誤り訂正の際に、訂正できる場合がある。

定義 14 b をセルの格納するビット数, $\text{Page-}n (1 \leq n \leq b)$ を、上位ビットから数えて n ビット目が属するページ, k を誤りが検出されたセルの 2^b 元の整数値とする。また、 \leftarrow は代入を表すものとする。

たとえば、 $\text{Page-}1$ は最上位のビットが属するページであり、 $\text{Page-}b$ は最下位のビットが属するページである。多元と 2 元との間の写像には、2 進数もしくは [70] の反射 (reflection) の操作を用いて構成したグレイ符号を用いることができる。

[誤り訂正アルゴリズム (整数)]

誤り訂正は次の 5 つのステップで実行される。

1. $n \leftarrow 1$ とする。
2. $\text{Page-}n$ の誤り訂正符号を使って、誤りセルを特定する。
3. c を $0 \leq c < 2^{n-1}$ を満たす整数としたとき、 k が

$$2c \cdot 2^{b-n} \leq k < (2c + 1) \cdot 2^{b-n}$$

を満たすならば $k \leftarrow (2c + 1) \cdot 2^{b-n}$ とし、

$$(2c + 1) \cdot 2^{b-n} \leq k < (2c + 2) \cdot 2^{b-n}$$

を満たすならば $k \leftarrow (2c + 1) \cdot 2^{b-n} - 1$ とする。

4. $n < b$ ならば、 $n \leftarrow n + 1$ として、2. に戻る。
5. アルゴリズムを終了する。

例 10 $b = 3$ の場合の誤り訂正を考える。 $\text{Page-}1$ の訂正では、

- $0 \leq k < 4$ ならば $k \leftarrow 4$
- $4 \leq k < 8$ ならば $k \leftarrow 3$

とする。 $\text{Page-}2$ の訂正では、

- $k = 0, 1$ ならば $k \leftarrow 2$
- $k = 2, 3$ ならば $k \leftarrow 1$
- $k = 4, 5$ ならば $k \leftarrow 6$
- $k = 6, 7$ ならば $k \leftarrow 5$

とする。Page-3の訂正は、通常のビット反転と一致する。

復号は次のように実行される。値2が4へシフトした誤りを考える。2進数の場合2は101、4は011に対応しており、Page-1に属するビットに反転を生じ、符号により検出される。 $4 \leq k = 4 < 8$ が満たされているため、 $k \leftarrow 3$ とする。3は100と対応しており、Page-2においてビット反転は検出されないが、Page-3に属するビットに反転を生じており、符号により検出される。これを反転し、誤り訂正を終了する。

このアルゴリズムは多元の操作を行っているが、2元の各ビットの操作によるアルゴリズムに簡単に書き直すことができる。アルゴリズムは写像に2進数を用いるか、グレイ符号を用いるかで異なるが、両方とも上記整数値を用いたアルゴリズムと同等であり、したがってBERのパフォーマンスに差異はない。

[2元の誤り訂正アルゴリズム(2進数)]

整数を用いた訂正アルゴリズムの3.を、次のものに置き換える。

3'. Page- n のビットが0の場合、そのビットを1とし、Page-1,...,Page-($n-1$)の対応するビットを0...0とする、Page- n のビットが1の場合、そのビットを0とし、Page-1,...,Page-($n-1$)の対応するビットを1...1とする、

[2元の誤り訂正アルゴリズム(グレイ符号)]ここでグレイ符号として、表13のように消去状態が1...1に対応したもの想定する。整数を用いた訂正アルゴリズムの3.を、次のものに置き換える。

3''. Page- n のビットが0の場合そのビットを1、Page-($n+1$)のビットを0、Page-($n+2$),...,Page- b の対応するビットを1...1とする。Page- n のビットが1の場合そのビットとPage-($n+1$)のビットを0、Page-($n+2$),...,Page- b の対応するビットを1...1とする。

6.5節で示すように、提案手法はデータ読み出しに要する時間が増大するため、誤り率が高いときのみ提案手法に切り替える利用方法が考えられる。このため、切り替えまでの符号構成として、グレイ符号と2進数のどちらを利用するかの選択が必要である。グレイ符号の利用は誤り率の改善が見込まれる反面、2進数に基づくデータ読み出しのためのセンスアンプを用いる場合、少なくとも1つの2進数—グレイ符号間の変換ブロックを要する[9]。したがって、2進数とグレイ符号の選択は、従来の速度で読み出し可能となる最大のセル誤り率と、回路量のトレードオフによって判断する。

6.4 ビット誤り率の算出

提案手法は多レベル誤りを効率的に訂正する手法であり、ビット誤り率の改善は各レベルの誤りが発生する割合に依存する。この章では提案手法、および従来手法における各ページのBERを、セルの誤り率と各誤りレベルの誤りが発生する割合から、理論的に求める。

定義 15 E をセルの誤り率, b を单一セルに格納するビット数, R_j をある誤りセルについて, その変量が j である確率とする (つまり, $\sum_{j=1}^{2^b-1} R_j = 1$). BER_1 を, Page-1 のビット誤り率とする. $\text{BER}_n (2 \leq n \leq b)$ を, $\text{Page-1}, \dots, \text{Page-}(n-1)$ を提案手法によって訂正し終えた後の, $\text{Page-}n$ の誤り率とする.

$p_n(x)$ は $x (0 \leq x < 2^{b-1})$ を 2 進数で表したときの, 下位から第 $b-n$ 柄目の値, すなわち

$$x = c_2 2^{b-2} + \dots + c_n 2^{b-n} + \dots c_{b-1} 2^1 + c_b 2^0 (0 \leq c_i \leq 1) \quad (21)$$

としたとき, $p_n(x) = c_n$ とする. $D = 2^{b-n}$ とし, $v(i, n, l)$ を,

$$v(i, n, l) = \begin{cases} kD & (0 \leq 2kD < l < (2k+1)D), \\ kD + l \bmod D & ((2k+1)D \leq l < (2k+2)D \leq 2^{b-i}), \\ 2^{b-i-1} - kD & (2^{b-i} \leq 2kD + 2^{b-i} \leq l < (2k+1)D + 2^{b-i}), \\ 2^{b-i-1} - kD - l \bmod D & ((2k+1)D + 2^{b-i} \leq l < (2k+2)D + 2^{b-i} \leq 2^{b-i+1}), \\ 0 & (l \geq 2^{b-i+1}), \end{cases} \quad (22)$$

とする. ここで k は $0 \leq k < 2^{b-i-1}$ を満たす整数である. また, $g_1(i, l), g_2(i, l), f_1(n, l), f_2(n, l)$ をそれぞれ,

$$\begin{aligned} g_1(i, l) &= \begin{cases} 0 & (0 \leq l < 2^{b-i}, 2^{b-i+1} \leq l), \\ l - 2^{b-i} & (2^{b-i} \leq l < 2^{b-i+1}), \end{cases} \\ g_2(i, l) &= \begin{cases} l - 1 & (0 \leq l < 2^{b-i}), \\ 2^{b-i} - 1 & (2^{b-i} \leq l < 2^{b-i+1}), \\ 0 & (2^{b-i+1} \leq l), \end{cases} \\ f_1(n, l) &= \begin{cases} l & (0 \leq l < D), \\ D - l & (D \leq l < 2D), \\ 0 & (2D \leq l < 2^b), \end{cases} \\ f_2(n, l) &= \begin{cases} 0 & (n = 1), \\ \sum_{i=1}^{n-1} (2^{i-1} v(i, n, l)) & (n \geq 2) \end{cases} \end{aligned}$$

とする.

補助定理 5 以下の式が成り立つ:

$$\sum_{x=g_1(i, l)}^{g_2(i, l)} p_n(x) = v(i, n, l).$$

(証明) $0 \leq l < 2^{b-i+1}$ となる整数 l を D 個の整数を持つ 2^{n-i} 個のブロックに分割する. 例えば, 最初のブロック(ブロック 0)は $0, 1, \dots, D-1$ を含み, 2つめのブロック(ブロック 1)は $D, D+1, \dots, 2D-1$ を含む. 以降のブロックも同様である. 与式左辺の和は, 次の 5 つの条件に分類される.

1. $0 \leq j < 2^{n-i-1}$ となる整数 j について, ブロック $2j$ に属するどの l についても, Dj 個の非零の $p_n(x)$ の和となる.
2. $0 \leq j < 2^{n-i-1}$ となる整数 j について, ブロック $2j+1$ に属するどの l についても, $Dj + l \bmod D$ 個の非零の $p_n(x)$ の和となる.
3. $2-n-i-1 \leq j < 2^{n-i}$ となる整数 j について, ブロック $2j$ に属するどの l についても, $2^{b-i-1} - D(j - 2^{n-i-1})$ 個の非零の $p_n(x)$ の和となる.
4. $2-n-i-1 \leq j < 2^{n-i}$ となる整数 j について, ブロック $2j+1$ に属するどの l についても, $2^{b-i-1} - D(j - 2^{n-i-1}) - l \bmod D$ 個の非零の $p_n(x)$ の和となる.
5. $l \geq 2^{b-i+1}$ のとき, $g_1(i, l) = g_2(i, l) = 0$ であるため, 和は 0 となる.

ここで $k = j - 2^{n-i-1}$ とおくと,

$$\begin{aligned} 2^{b-i-1} - D(j - 2^{n-i-1}) &= 2^{b-i-1} - Dk, \\ 2^{b-i-1} - D(j - 2^{n-i-1}) - l \bmod D &= 2^{b-i-1} - Dk - l \bmod D \end{aligned}$$

が成り立つ. したがって, 式 (22) を得る. (証明終了)

定理 14 $\text{BER}_n (1 \leq n \leq b)$ は以下のように計算される :

$$\text{BER}_n = E \sum_{l=1}^{2^b-1} A_l(n) R_l.$$

ここで,

$$A_l(n) = \frac{2^{n-1} f_1(n, l) + f_2(n, l)}{2^b - l}$$

とする.

(証明) $\text{Page-}n$ におけるある誤りセルに着目し, ビットが反転を生じる整数値の遷移を数える. 簡単のため整数値が増加し, ビット反転が $0 \rightarrow 1$ となる遷移のみを考えるが, 一般性は失われない. u を $1 \leq u < n$ を満たす整数としたとき, u によって 2 つの場合に分けて考える. 1. 任意の u について, $\text{Page-}u$ に誤りが存在しない ($\text{Page-}n$ より上位のページに誤りが存在しない), 2. $\text{Page-}u$ に誤りが存在するような u が, 少なくとも 1 つ存在する ($\text{Page-}n$ より上位のページに誤りが存在する).

1. 2^b 個のセルの取り得る整数値を, $2D$ 個の値を有する 2^{n-1} 個のブロックに分割する. 例えば, 最初のブロックは $0, 1, \dots, 2D - 1$ から成り, 2つめのブロックは $2D, 2D + 1, \dots, 4D - 1$ から成る. 以降のブロックも同様である. それぞれのブロック内において, Page- n にビット反転を生じ得る遷移は $f_1(n, l)$ 個である. 値の遷移が 2つのブロック間にまたがり遷移するとき, ビット誤りは必ず Page- u に含まれるため, ここで考慮すべきではない. このため, $2D \leq l < 2^b$ について $f_1(n, l) = 0$ とする. 同様の計算が 2^{n-1} 個のブロックすべてについて可能なため, 求める遷移の数は $2^{n-1} f_1(n, l)$ である.
2. 整数値の増加方向への誤りは, いずれのページの訂正でも減少方向に向かって行われる. また Page- n における訂正量は一定して 2^{b-n} である. よって, Page- i ($1 \leq i < n$) の訂正を終えた時点で, それ以降に訂正すべき変量を x とすれば, $p_n(x) = c_n = 1$ のときのみ Page- n で訂正が実行され, $p_n(x) = c_n = 0$ ならば実行されない. したがって, $p_n(x)$ により各ページの訂正が実行されるかどうかが分かる. l を固定した場合の x が取り得る値は, 下限を $g_1(n, l)$, 上限を $g_2(n, l)$ である. また, ひとつの x にはひとつの値の遷移が対応する. よって, この場合の反転を生じる遷移の数は,

$$\sum_{x=g_1(i,l)}^{g_2(i,l)} p_n(x)$$

となり, 補助定理 5 よりこれは $v(i, n, l,)$ と一致する. ここで, Page-1, …, Page- $(i-1)$ に誤りはないため, これらのページに誤りを生じる $l \geq 2^{b-i+1}$ については, $g_1(i, l) = g_2(i, l) = 0$ としている. 誤りが存在する最小の i として $1 \leq i \leq n-1$ について考え, 2^{i-1} 個の同様な計算が成り立つブロックについて和を取れば, 求める遷移の数となる. したがって, $f_2(n, l)$ を得る. ただし, $n=1$ については上位のページが存在しないため, $f_2(1, l) = 0$ としている.

ビット反転を生じる割合は, 上記 2つの場合分けの和に対して, 整数値が増加する遷移の総数 $2^b - l$ で除算すればよい. これにより変量 l の誤りにより n ページの訂正でビット反転が生じる割合, $A_l(n)$ を得る. (証明終了)

例 11 定理 14 を用いると, $b = 2$ のときの BER_n は以下のようになる.

$$\begin{aligned} \text{BER}_1 &= E\left(\frac{1}{3}R_1 + R_2 + R_3\right), \\ \text{BER}_2 &= E\left(\frac{2}{3}R_1 + \frac{1}{2}R_2 + R_3\right). \end{aligned}$$

また, $b = 3$ のときは以下のようになる.

$$\begin{aligned}\text{BER}_1 &= E\left(\frac{1}{7}R_1 + \frac{1}{3}R_2 + \frac{3}{5}R_3 + R_4 + R_5 + R_6 + R_7\right), \\ \text{BER}_2 &= E\left(\frac{2}{7}R_1 + \frac{2}{3}R_2 + \frac{3}{5}R_3 + \frac{1}{2}R_4 + \frac{2}{3}R_5 + R_6 + R_7\right), \\ \text{BER}_3 &= E\left(\frac{4}{7}R_1 + \frac{1}{2}R_2 + \frac{3}{5}R_3 + \frac{1}{2}R_4 + \frac{2}{3}R_5 + \frac{1}{2}R_6 + R_7\right).\end{aligned}$$

次にグレイ符号を用いて, それぞれのページに通常のビット誤り訂正をする, 従来手法のビット誤り率を同様な形で求める.

定義 16 $\text{BER}'_n (1 \leq n \leq b)$ を, グレイ符号を用いた場合の $\text{Page-}n$ の $R\text{BER}$ とする.

$D = 2^{b-n}$ とおく. このとき $h_1(n, l), h_2(n, l)$ をそれぞれ,

$$\begin{aligned}h_1(n, l) &= \begin{cases} 2 \cdot (l \bmod 4D) & (0 \leq l < D), \\ 2D & (2D \leq l < 2D), \\ 6D - 2 \cdot (l \bmod 4D) & (2D \leq l < 3D), \\ 0 & (3D \leq l < 4D), \end{cases}, \\ h_2(n, l) &= \begin{cases} 2(2^{n-2} - 1 - u)(l \bmod 4D) & (4Du \leq l < 4Du + 2D), \\ 2(2^{n-2} - 1 - u)(4D - l \bmod 4D) & (4Du + 2D \leq l < 4Du + 4D), \end{cases}\end{aligned}$$

とする. ただし u は $0 \leq u \leq 2^{n-2} - 1$ を満たす整数である.

定理 15 $\text{BER}'_n (1 \leq n \leq b)$ は以下のように計算される :

$$\text{BER}_n = E \sum_{l=1}^{2^b-1} B_l(n) R_l.$$

ここで,

$$B_l(n) = \frac{h_1(n, l) + h_2(n, l)}{2^b - l}$$

とする.

(証明) $\text{Page-}n$ において, ビットが反転する値の遷移を数える. 簡単のため整数値が増加する遷移のみを考えるが, 一般性は失われない. セルの取り得る 2^b 個の整数値を, 値の数が $2D$ 個となる 2^{n-1} 個のブロックに分割する. 例えば, 最初のブロックは $0, 1, \dots, 2D - 1$ から成り, 2つめのブロックは $2D, 2D + 1, \dots, 4D - 1$ から成る.

以降のブロックも同様である。 i を $1 \leq i \leq 2^{n-1}$ となる整数とする。 $s(l)$ と $t(l)$ を

$$s(l) = \begin{cases} l \bmod 4D & (0 \leq \bmod 4D < D), \\ D & (D \leq \bmod 4D < 2D), \\ 3D - l \bmod 4D & (2D \leq \bmod 4D < 3D), \\ 0 \bmod 4D & (3D \leq \bmod 4D < 4D), \end{cases}$$

$$t(l) = \begin{cases} l \bmod 4D & (0 \leq \bmod 4D < D), \\ 2D - l \bmod 4D & (D \leq \bmod 4D < 2D), \\ 0 & (2D \leq \bmod 4D < 4D), \end{cases}$$

と定義する。 i を $0 \leq i < 2^{n-1}$ を満たす整数とし、 j を $0 \leq j < 2^{n-2}$ を満たす整数とする。 i 番目の領域を始点とする $1 \rightarrow 0$ の遷移の個数を $Q_i^{(1 \rightarrow 0)}(l)$ とし、 i 番目の領域を始点とする $0 \rightarrow 1$ の遷移の個数を $Q_i^{(0 \rightarrow 1)}(l)$ とすると、

$$Q_{2j}^{(1 \rightarrow 0)}(l) = \begin{cases} s(l) & (0 \leq l < 2^b - 4Dj), \\ 0 & (2^b - 4Dj \leq l < 2^b), \end{cases}$$

$$Q_{2j}^{(0 \rightarrow 1)}(l) = \begin{cases} s(l - D) & (0 \leq l < 2^b - 4D(j+1)), \\ t(l - D) & (2^b - 4D(j+1) \leq l < 2^b - 4Dj), \\ 0 & (2^b - 4D \leq l < 2^b), \end{cases}$$

$$Q_{2j+1}^{(1 \rightarrow 0)}(l) = \begin{cases} s(l - D) & (0 \leq l < 2^b - 4D(j+1)), \\ 0 & (2^b - 4D(j+1) \leq l < 2^b), \end{cases}$$

$$Q_{2j+1}^{(0 \rightarrow 1)}(l) = \begin{cases} s(l) & (0 \leq l < 2^b - 4D(j+1)), \\ t(l) & (2^b - 4D(j+1) \leq l < 2^b - 4Dj), \\ 0 & (2^b - 4D \leq l < 2^b), \end{cases}$$

と表せる。ビット反転を生じる遷移の数は

$$\sum_{i=1}^{2^b} (Q_i^{(0 \rightarrow 1)}(l) + Q_i^{(1 \rightarrow 0)}(l))$$

によって求められるが、これは $h_1(n, l) + h_2(n, l)$ に等しい。なぜなら、

$$2^b - 4D(j+1) \leq l < 2^b - 4Dj$$

についての和は $h_1(n, l)$ と一致し、

$$0 \leq l < 2^b - 4D(j+1)$$

についての和は $h_2(n, l)$ と一致するためである。これを整数値が増加する遷移の総数 $2^b - l$ で割ると、 $B_l(n)$ を得る。 (証明終了)

特に Page-1 は他のページの訂正の影響を受けないため、 $\text{BER}_1 = \text{BER}'_1$ が常に成り立つ。

例 12 定理 15 を用いると, $b = 2$ のときの BER'_n は以下のようになる.

$$\begin{aligned}\text{BER}'_1 &= \text{BER}_1 = E\left(\frac{1}{3}R_1 + R_2 + R_3\right), \\ \text{BER}'_2 &= E\left(\frac{2}{3}R_1 + R_2\right).\end{aligned}$$

また, $b = 3$ のときは以下のようになる.

$$\begin{aligned}\text{BER}'_1 &= \text{BER}_1 = E\left(\frac{1}{7}R_1 + \frac{1}{3}R_2 + \frac{3}{5}R_3 + R_4 + R_5 + R_6 + R_7\right), \\ \text{BER}'_2 &= E\left(\frac{2}{7}R_1 + \frac{2}{3}R_2 + \frac{4}{5}R_3 + R_4 + \frac{2}{3}R_5\right), \\ \text{BER}'_3 &= E\left(\frac{4}{7}R_1 + R_2 + \frac{2}{5}R_3 + \frac{2}{3}R_5 + R_6\right).\end{aligned}$$

6.5 評価

6.5.1 読み出し時のレイテンシとスループット

ここでは具体例として, 符号構成に BCH 符号を用いて提案手法を実行した場合の, 読み出し時のレイテンシとスループットを, 従来の復号法と比較する. ただし, ページ毎に誤り訂正を実行するならば, 提案手法は(軟判定の復号を含み)任意の対称ビット誤り訂正符号に対して有効であり, それらに対してもこの章の議論はほぼ同様に行えることに注意する. 提案手法は单一のページ読み出しであっても, 複数ページアクセスを前提としている. よって, メモリとコントローラ間の過剰な通信を避けるため, ここではオンチップで誤り訂正を行うこととする.

BCH 符号の復号に要する時間 :

BCH 符号の復号は, [13] に示されるハードウェアでの実装を想定する. 復号は以下の 3 つの手順から成る

1. シンドロームの計算
2. 誤り位置多項式の導出
3. 誤り位置の特定

符号長を n , 情報長を k とする. 1) は線形帰還シフトレジスタ (Linear Feedback Shift Register: LFSR) を利用する. p 重に並列化することで, n/p サイクルで実行できる. 2) は Simplified Inversionless Berlekamp Massey 法で実現し, t サイクルを要する. 3) は Chien 探索で実行し, p 重に並列化することで, n/p サイクルとなる. これら 3 つの手順は独立しており, パイプラインで実行できる. したがって, 復号は合計 $n/p + t$ クロックで実現できる. ただしこれは最大値であり, 誤りが存在しない場合は検出のための k/p サイクルしか要さない.

表 14: BCH 符号の復号に要する時間

	平均			最大
BER	10^{-5}	10^{-4}	10^{-3}	-
$t = 8$	$5.1\mu s$	$5.2\mu s$	$5.5\mu s$	$5.5\mu s$
$t = 16$	$5.1\mu s$	$5.3\mu s$	$5.8\mu s$	$5.9\mu s$

例 13 各 BER において, BCH 符号の復号に要する時間の平均値と最大値を, 表 14 に示す. パラメータは $n = 2144, k = 2048, t = 8$ または $n = 2240, k = 2048, t = 16$ に設定し, クロックは $50MHz$, $p = 8$ として計算している.

レイテンシ :

図 13 は $b = 3$ に設定した場合の, 読み出しにおける各操作の実行時間とタイミングの概略図である. 従来手法のレイテンシは図 13 の (a) を用いて求められる. Page- i の読み出しが, $(2^i - 1)$ 通りの参照電圧を印加することで実現される. したがって, 1つの参照電圧の印加時間を t_{read} , 誤り訂正に要する時間を t_{ecc} , 入出力に要する時間を t_{io} とすると, Page- i を読み出す際のレイテンシは,

$$L_{conv}^{(i)} = (2^i - 1) \cdot t_{read} + t_{ecc} + t_{io}$$

によって求められる. ただし, ページ間での誤り訂正時間の差異は無視できるものとする. 1つのページを読み出す際の平均のレイテンシは, 誤り訂正に要する時間の平均を t_{ecc_ave} とすると,

$$L_{conv} = \frac{1}{b} \sum_{i=1}^b L_{conv}^{(i)} = \left(\frac{2^{b+1} - 2}{b} - 1 \right) \cdot t_{read} + t_{ecc_ave} + t_{io},$$

となる. 提案手法のレイテンシは図中の (b) を用いて求められる. 従来手法と異なり, Page- i を読み出すためには, Page-1 から Page- i までの全てのページを読み出し, Page-1 から順に誤り訂正を行う必要がある. したがって, Page- i を読み出す際のレイテンシは,

$$L_{prop}^{(i)} = (2^i - 1) \cdot t_{read} + i \cdot t_{ecc} + t_{io}.$$

となる. また, 1つのページを読み出す際の平均のレイテンシは,

$$L_{prop} = \frac{1}{b} \sum_{i=1}^b L_{prop}^{(i)} = \left(\frac{2^{b+1} - 2}{b} - 1 \right) \cdot t_{read} + \frac{b+1}{2} \cdot t_{ecc_ave} + t_{io}.$$

となる.

例 14 パラメータを $b = 3$, $t_{read} = 40\mu s$, ページサイズ 4KB, 入出力バンド幅を $1Gb/s$ に設定する. また, BCH 符号のパラメータは例 13 と同様とする. こ

表 15: 読み出しレイテンシ

		平均			最大
BER		10^{-5}	10^{-4}	10^{-3}	-
従来	$t = 8$	182.2 μ s	182.3 μ s	182.6 μ s	316.0 μ s
	$t = 16$	182.2 μ s	182.4 μ s	182.9 μ s	316.4 μ s
提案	$t = 8$	187.3 μ s	187.5 μ s	188.1 μ s	327.0 μ s
	$t = 16$	187.3 μ s	187.7 μ s	188.7 μ s	328.2 μ s

の場合、表 14 の値を誤り訂正符号の復号時間として利用できる。結果を表 15 に示す。 $t = 8$ の場合で比較すると、提案手法のレイテンシは従来より平均で約 5 μ s(約 2.7%) 増大し、レイテンシの最大値は、11 μ s(約 3.4%) 増大する。

スループット：スループットについては、各ページがシーケンシャルにアクセスされるか、ランダムにアクセスされるかによって、状況が異なる。もし完全にランダムにアクセスされる場合、ひとつのページ読み出し毎に前出のレイテンシが発生する。一方、シーケンシャルにアクセスされる場合は、 b ページをまとめて読み出し、訂正の後に output すればよく、要する時間は小さくなる。典型的な NAND フラッシュメモリのページサイズは～16KB 程度であり、たとえばコンシューマ用 PC 向けのストレージなど、多くのアプリケーションにとってそれほど大きくはない。このため、ここではページが完全にシーケンシャルにアクセスされる場合について議論を行う。通常入出力は内部での読み出し操作より高速であるため、 $t_{read} \geq t_{io}$ と仮定する。図 13 の (c) に示されるように、従来手法では、誤り訂正と入出力は、参照電圧の印加と並行して実行できる。したがって、スループットは

$$T_{conv} = \frac{nb}{(2^b - 1) \cdot t_{read}}$$

となる。一方、図 13 の (d) に示されるように、提案手法では Page- i の誤り訂正が Page- $(i + 1)$ ～Page- b に影響するため、 b ページすべての読み出しを行い、その後に誤り訂正を順に実行する。したがって、スループットは

$$T_{prop} = \frac{b \cdot n}{(2^b - 1) \cdot t_{read} + t_{ecc_ave} + t_{io}}.$$

によって求められる。

例 15 パラメータを $b = 3$, $t_{read} = 40\mu$ s, ページサイズ 4KB, 入出力バンド幅を 1Gb/s に設定する。また、BCH 符号のパラメータは例 13 と同様とする。この場合、表 14 の値を誤り訂正符号の復号時間として利用できる。結果を表

15に示す。従来手法のスループットは約351Mb/sであり、提案手法の場合は約310Mb/sとなる(BERに依存するBCH符号の復号時間の差異は、比較的小さい)。これはおよそ15%の低下となっている。

6.5.2 ビット誤り率

変量が2より大きい誤りの発生は無視できるものとする。すなわち、

$$R_3 = R_4 = \dots = R_{2^b-1} = 0$$

とし、便宜上 $R_1 = 1 - R, R_2 = R$ とおく。 $b = 2$ の場合、提案手法を用いた場合のビット誤り率 BER_n 、従来手法を用いた場合のビット誤り率 BER'_n は、

$$\begin{aligned}\text{BER}_1 &= \text{BER}'_1 = E\left(\frac{1}{3} + \frac{2}{3}R\right), \\ \text{BER}_2 &= E\left(\frac{2}{3} - \frac{1}{6}R\right), \\ \text{BER}'_2 &= E\left(\frac{2}{3} + \frac{1}{3}R\right),\end{aligned}$$

となる。また $b = 3$ の場合は、

$$\begin{aligned}\text{BER}_1 &= \text{BER}'_1 = E\left(\frac{1}{7} + \frac{4}{21}R\right), \\ \text{BER}_2 &= \text{BER}'_2 = E\left(\frac{2}{7} + \frac{9}{21}R\right), \\ \text{BER}_3 &= E\left(\frac{4}{7} - \frac{1}{14}R\right), \\ \text{BER}'_3 &= E\left(\frac{4}{7} + \frac{3}{7}R\right),\end{aligned}$$

となる。各式の括弧内の定数項のため、これらの式は R が小さいとき、Page- b の誤り率が他のページと比較して高いことを示している。誤り訂正後の誤り率の差異はさらに顕著となるため、メモリ全体としての信頼性を Page- b の BER を用いて評価する。 $b = 2$ の場合の改善率は

$$F = \frac{\text{BER}'_2}{\text{BER}_2} = \frac{4+2R}{4-R}$$

であり、 $b = 3$ の場合は、

$$F = \frac{\text{BER}'_2}{\text{BER}_2} = \frac{8+6R}{8-R}$$

となる。 F 倍大きな訂正前の誤り率 R_{BER} に対して、同じ訂正後の BER を確保できるため、 F の値はそのまま許容可能な BER の改善率となる。図14に F と R の関係を示す。許容可能な BER は R の増加とともに改善される。とくに $b = 3$ の場合に改善率は大きい。 R の値はメモリ毎に異なると考えられるが、実験値である $b = 2, R = 0.05$ [8] に設定した場合、許容できる誤り率は4%増加し、 $b = 2, R = 0.20$ に設定した場合、改善率は16%に達する。

6.5.3 寿命

提案手法は従来より多くのクロックサイクルを要するが、符号構成そのものは従来のままであり、BCH 符号の復号回路の変更もほぼ必要ない。したがって従来手法を、符号により要求された信頼性を確保できなくなる P/E サイクルまで利用し、提案手法をその後に用いることができる。この場合、メモリの寿命となる P/E サイクルを延ばすことができ、従来手法を用いてる間は追加のクロックサイクルは不要である。この寿命の増加量を評価する。

BER が P/E サイクルに対し指数関数的に増加するモデルを仮定する。すなわち、

$$\text{BER} = A \cdot e^{BN}$$

とする。ただし、 A, B はメモリ依存の定数、 N は P/E サイクルを表す。文献 [7] の実験値である $b = 2$ のセルにおいて、 $N = 15500$ のとき $\text{BER} = 10^{-3}$ 、 $N = 26000$ のとき $\text{BER} = 3 \times 10^{-3}$ を用いることとする。このとき、 $A = 1.96 \times 10^{-4}$ 、 $B = 1.05 \times 10^{-4}$ となる。 N_1 を従来手法の寿命、 N_2 を提案手法の寿命とすると、

$$\begin{aligned}\text{BER} &= A \cdot e^{BN_1}, \\ F \cdot \text{BER} &= A \cdot e^{BN}\end{aligned}$$

であるから、

$$N_2 - N_1 = \frac{\ln F}{B}$$

を得る。寿命となる P/E サイクルを N_l するとき、寿命の改善率の増分は

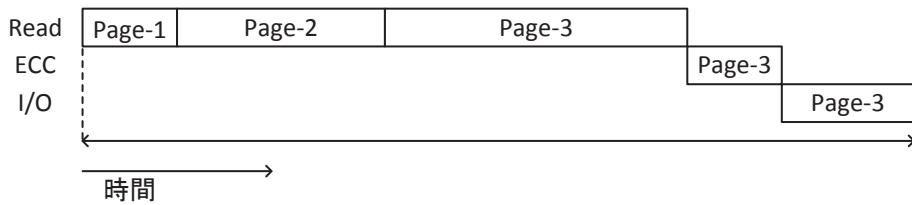
$$F_e = \frac{N_2 - N_1}{N_l} = \frac{\ln F}{BN_l}$$

となる。各レベルの誤りの割合として、6.5.2 項と同じものを用いる。寿命として [7] で示されている $N_l = 15500$ を用いたときの、2 レベル分の誤りの割合 R と F_e の関係を、図 15 に示す。 $R = 0.05$ のとき、寿命の改善率の増加量は $F_e = 2.4\%$ であり、 $R = 0.20$ のとき、 $F_e = 9.1\%$ に達する。

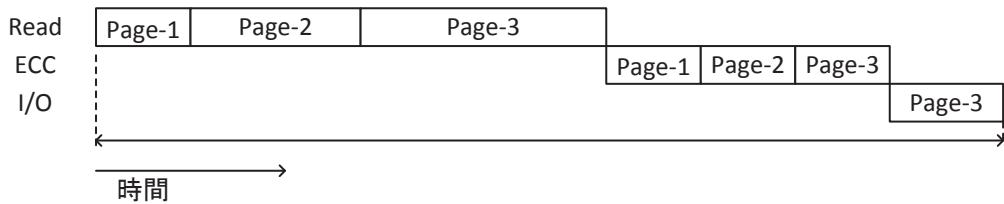
6.6 この章のまとめ

近傍に向かう値への誤りほど発生率が高く、多レベルの誤りも発生するという、NAND フラッシュメモリの誤りの特徴を考慮した誤り訂正の手法を提案した。提案手法は、マルチページプログラミングと併用される従来手法と同様に、各ページへビット誤り訂正符号をそれぞれ適用することで構成する。このため、従来の LM 誤り訂正符号をはじめとする多元の符号と比較して、書き込み速度に強みをもつ。提案手法は復号の際に、誤りの検出はビット誤り訂正符号により行うが、セルの多元

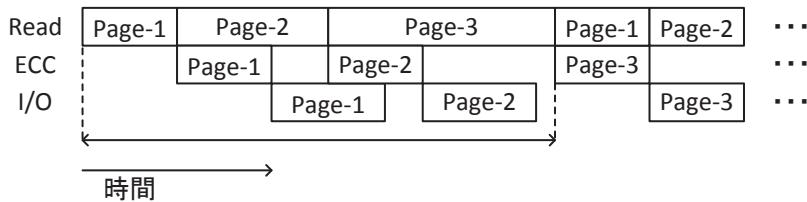
の整数値を用いて、元の値である確率の最も高い近傍値へと訂正を行う。これにより BER を改善でき、その度合は各レベルの誤りの占める割合と、セルのビット数によって決まる。これらパラメータから提案手法と従来手法の両方について、各ページの BER の理論値を求める方法を示した。 $b = 2$ ビットセルで、3 レベルの以上の誤りの発生は無視でき、2 レベルの誤りが 5% を占めるとき、許容可能な BER は 4% 改善する。これは許容できる P/E サイクル数に換算すると、2.4% の増加となる、符号構成に BCH 符号を仮定すると、提案手法によって平均レイテンシは 2.7% 増加、最大レイテンシは 3.4% 増加、スループットは 15% 低下する。ただし、従来の復号法で誤り率が許容できないような P/E サイクルでのみ提案手法に切り替えることで、切り替えまでレイテンシやスループットの悪化を防ぐことができる。



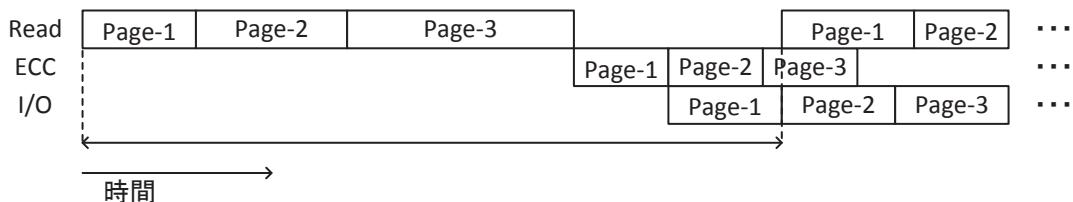
(a) ページ 3 読み出し (従来手法).



(b) ページ 3 読み出し (提案手法).



(c) シーケンシャル 3 ページ読み出し (従来手法).



(d) シーケンシャル 3 ページ読み出し (提案手法).

図 13: 読み出しにおける各操作の実行時間とタイミング ($b = 3$).

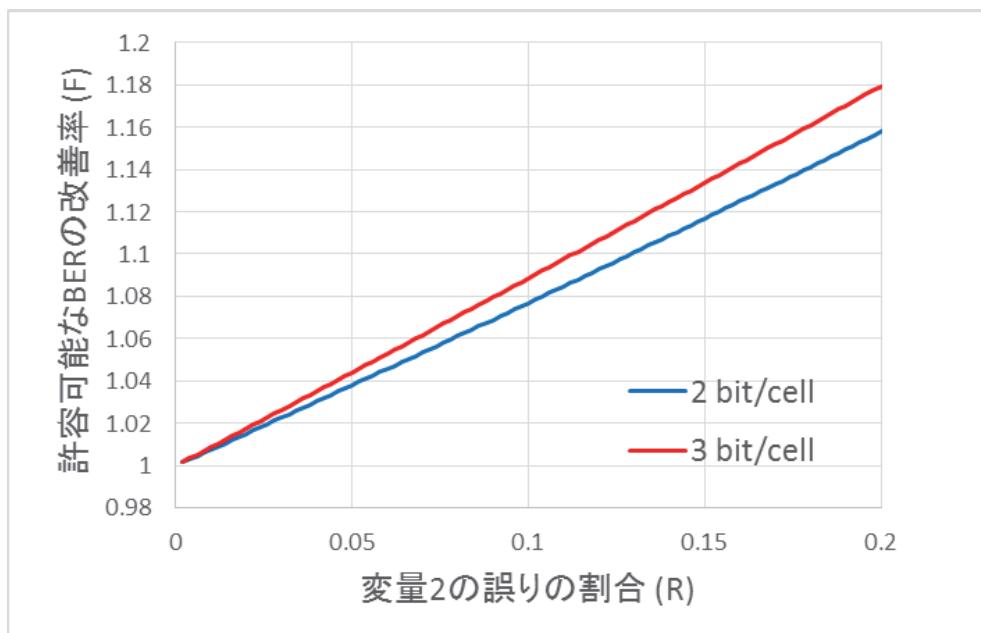


図 14: 許容可能な BER の改善率.

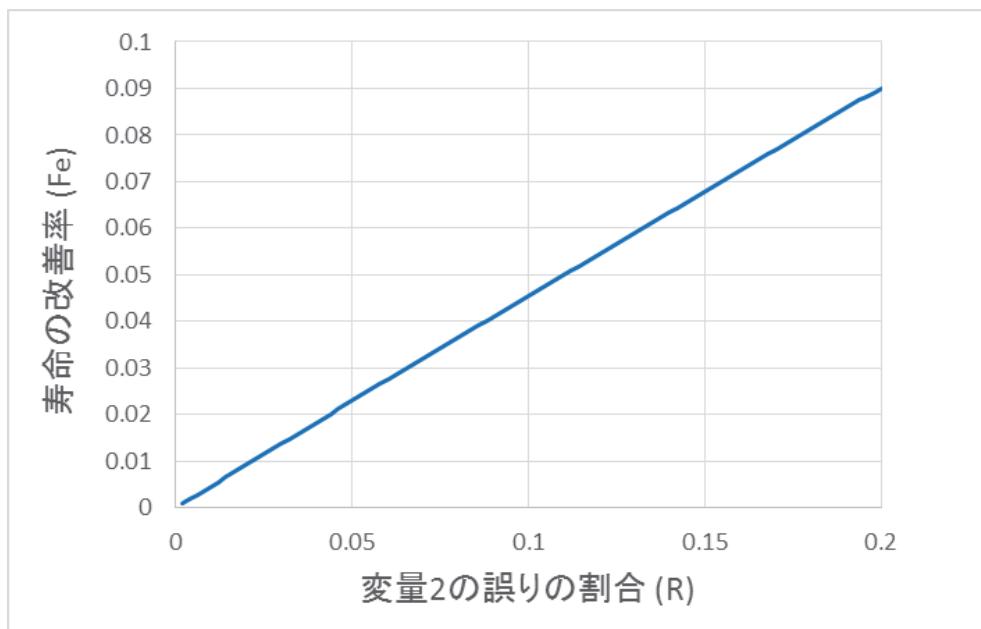


図 15: 寿命の改善率の増加量.

7 提案手法3：2元の符号を用いた双方向LM誤り訂正符号

7.1 概要

これまでに議論したように、マルチページプログラミングを利用することで、書き込みを高速に実現できるが、誤り訂正符号の構成には制限があった。一方、マルチページプログラミングを利用しなければ、符号はより柔軟に誤りの特徴を反映することができ、これについては第5章の符号をはじめとするLM誤り訂正符号が、ひとつつの答えと言えるものであった。本章での提案はこれらの折衷案とも言えるものである。つまり、始めは従来のページ毎の誤り訂正によって、マルチページプログラミングを利用可能とし、信頼性の確保ができないP/Eサイクル数となって時点で、強力なLM誤り訂正符号に切り替え、寿命を延長する。単純な切り替えは、符号化および復号回路を複数用意することとなり、コストがかかるが、提案手法は切り替え以前の符号構成を一部利用することで、この問題を解決する。

本章で提案する符号は、次の2つのアイデアにより構成されている。

1. t 双方向 (l_u, l_d) -LM 誤り訂正符号を、ページ毎の2元の符号のみを用いて構成する。訂正機能をLM誤りに制限することで、上位数ページの符号化が不要となる。
2. 符号化が不要となったことで得られた検査部を利用し、他の誤り訂正符号を追加で適用することで、誤り率を改善する。

1. のLM誤り訂正符号は、メモリの総レベル数を q 、訂正可能な増加量を l_u 、減少量を l_d としたとき、 $q' = l_u + l_d + 1 (q > q')$ 元の対称シンボル誤り訂正符号を用いた構成法が既に知られている[31]。提案手法では $q = 2^b, q' = 2^m (b, m$ は $b > m$ を満たす正整数)にパラメータを制限し、下位 m ページのみに、従来と同様それぞれ t ビット誤り訂正符号を用いて構成する。特に、 $m = 1$ に設定すれば、 t 非対称1-LM誤り訂正符号[29]のある種の構成法となっている。したがって、従来の非対称LM誤り訂正符号を、(元の数を増やすのではなく)用いる2元の符号の数を増やす形で拡張したものに相当する。

7.2 距離と誤り訂正機能

t 双方向 (l_u, l_d) -LM 誤り訂正符号の構成法を以下に示す[31]。 $Q_q = \{0, 1, \dots, q-1\}$ とする。

構成法： Ω を $q' = l_d + l_u + 1$ 元の t 重対称シンボル誤り訂正符号とする。 $q (q > q')$ 元の符号 C を、

$$C = \{\mathbf{x} \in Q_q^n : \mathbf{x} \bmod q' \in \Omega\} \quad (23)$$

によって定義する。このとき、 C は t 双方向 (l_u, l_d) -LM 誤り訂正符号となる。

提案する LM 誤り訂正符号を, この式を満たす符号を定義することで構成する. 文献 [31] ではこの式を満たす, 具体的な t 双方向 (l_u, l_d) -LM 誤り訂正符号を与え, 訂正可能なことが復号アルゴリズムを用いて示されている. しかし, この式を満たすいかなる符号構成も, t 双方向 (l_u, l_d) -LM 誤り訂正が可能である, という十分条件は証明されていない. まずはこれを証明する.

最初に, 双方向 LM 誤り訂正に関する距離を定義する. 以下に示す距離は, 数学的な意味での距離ではないが(三角不等式が満たされない), t 双方向 (l_u, l_d) -LM 誤り訂正符号であるための, 必要十分条件を与える. $\mathbf{x} = (x_1, \dots, x_n) \in Q_q^n$ および $\mathbf{y} = (y_1, \dots, y_n) \in Q_q^n$ について, $N[l_1, l_2](\mathbf{x}, \mathbf{y}) = |\{i : l_1 \leq x_i - y_i \leq l_2\}|$ とする. このとき, $M_1(\mathbf{x}, \mathbf{y}), M_2(\mathbf{x}, \mathbf{y})$ をそれぞれ,

$$\begin{aligned} M_1(\mathbf{x}, \mathbf{y}) &= 2N[\min(l_u, l_d) + 1, l_u + l_d](\mathbf{x}, \mathbf{y}) \\ &\quad + 2N[-(l_u + l_d), -(\max(l_u, l_d) + 1)](\mathbf{x}, \mathbf{y}), \\ M_2(\mathbf{y}, \mathbf{x}) &= N[1, \max(l_u, l_d)](\mathbf{x}, \mathbf{y}) \\ &\quad + N[-\max(l_u, l_d), -1](\mathbf{x}, \mathbf{y}) \\ &\quad + 2N[\max(l_u, l_d) + 1, l_u + l_d](\mathbf{x}, \mathbf{y}) \\ &\quad + 2N[-(l_u + l_d), -(\max(l_u, l_d) + 1)](\mathbf{x}, \mathbf{y}) \end{aligned}$$

とする.

定義 17 2つのベクトル \mathbf{x}, \mathbf{y} の LM 距離 $d_{LM}(\mathbf{x}, \mathbf{y})$ を

$$d_{LM}(\mathbf{x}, \mathbf{y}) = \begin{cases} 2n + 1 & (\max_i\{|x_i - y_i|\} > l_u + l_d), \\ \max(M_1(\mathbf{x}, \mathbf{y}), M_1(\mathbf{y}, \mathbf{x}), M_2(\mathbf{x}, \mathbf{y})) & (\text{otherwise}). \end{cases}$$

と定義する.

定理 16 符号 C が t 双方向 (l_u, l_d) -LM 誤りを訂正可能であるための必要十分条件は, 任意の異なる 2つの符号語 $\mathbf{x}, \mathbf{y} \in C$ が, $d_{LM}(\mathbf{x}, \mathbf{y}) \geq 2t + 1$ を満たすことである.

(証明) 誤り訂正が可能であることは, $\mathbf{x} + \mathbf{e} = \mathbf{y} + \mathbf{f}$ を満たす LM 誤り \mathbf{e}, \mathbf{f} が存在しないことと同値である. よって,

$$\mathbf{x} - \mathbf{y} \neq \mathbf{f} - \mathbf{e} \tag{24}$$

が常に満たされることを証明する. 以下では簡単のため $l_u > l_d$ とするが, 同様な議論は $l_u \leq l_d$ の場合も可能である.

(十分条件) $d_{LM}(\mathbf{x}, \mathbf{y}) \geq 2t + 1$ であるとき, 次の 1.~4. の少なくとも 1 つが満たされる.

1. $\max_i\{|x_i - y_i|\} > l_u + l_d$ のとき :

すべての i について, $|f_i - e_i| \leq l_u + l_d$ であるため, 式 (24) が満たされる.

2. $2M_1(\mathbf{x}, \mathbf{y}) \geq 2t + 1$ のとき :

式を変形して, $M_1(\mathbf{x}, \mathbf{y}) \geq t + 1$ が満たされる. このとき,

$$l_d + 1 \leq x_i - y_i \leq l_u + l_d$$

または,

$$-(l_u + l_d) \leq x_i - y_i \leq -(l_u + 1)$$

を満たす i が, 少なくとも $t+1$ 存在する. 一方, LM 誤りの定義より $-l_d \leq e_i \leq l_u$ が任意の i について成り立ち, また, $|\{i : f_i \neq 0\}| \leq t$ が成り立つ. したがって,

$$l_d + 1 \leq f_i - e_i \leq l_u + l_d$$

または,

$$-(l_u + l_d) \leq f_i - e_i \leq -(l_u + 1)$$

を満たす i は, 高々 t しか存在しない. ゆえに式 (24) が満たされる.

3. $2M_1(\mathbf{y}, \mathbf{x}) \geq 2t + 1$ のとき :

2. と同様である.

4. $M_2(\mathbf{x}, \mathbf{y}) \geq 2t + 1$ のとき :

$\mathbf{x} - \mathbf{y} = \mathbf{f} - \mathbf{e}$ が成り立つと仮定し, 矛盾を導く. 任意の i について, $f_i \leq l_u$ かつ $-e_i \leq l_d < l_u$ であるため,

$$l_u + 1 \leq |x_i - y_i| \leq l_u + l_d$$

を満たす i について, $f_i \neq 0$ かつ $e_i \neq 0$ でなければならない.

$$1 \leq |x_i - y_i| \leq l_u$$

を満たす i について, $f_i \neq 0$ または $e_i \neq 0$ でなければならない. これは $M_2(\mathbf{x}, \mathbf{y}) \leq 2t + 1$ ならば, \mathbf{f} と \mathbf{e} の非零の要素が少なくとも $2t + 1$ 存在することを意味する. このとき, \mathbf{f} と \mathbf{e} の一方は非零の要素数が $t + 1$ 以上となり, t 重誤りであることと矛盾する.

(必要条件) $d_{LM}(\mathbf{x}, \mathbf{y}) \leq 2t$ を満たす任意の \mathbf{x}, \mathbf{y} について, $\mathbf{x} - \mathbf{y} = \mathbf{f} - \mathbf{e}$ を満たす LM 誤り \mathbf{e}, \mathbf{f} が存在することを示せばよい. $d_{LM}(\mathbf{x}, \mathbf{y}) \leq 2t$ であることは,

$$\max_i \{|x_i - y_i|\} \leq l_u + l_d,$$

$$M_1(\mathbf{x}, \mathbf{y}) \leq 2t,$$

$$M_1(\mathbf{y}, \mathbf{x}) \leq 2t,$$

$$M_2(\mathbf{x}, \mathbf{y}) \leq 2t$$

を, すべて満たすことと同値である. このとき, $x_i - y_i$ の値に従って次の 1.~5. のように場合分けをし (これらの条件に重複はない), e_i, f_i を割り当てる.

1. $l_d + 1 \leq x_i - y_i \leq l_u$ のとき : $f_i = x_i - y_i, e_i = 0$.
2. $-l_u \leq x_i - y_i \leq -(l_d + 1)$ のとき : $f_i = 0, e_i = -(x_i - y_i)$.
3. $l_u + 1 \leq x_i - y_i \leq l_u + l_d$ のとき : $f_i = l_u, e_i = l_u - (x_i - y_i)$.
4. $-(l_u + l_d) \leq x_i - y_i < -(l_u + 1)$ のとき : $f_i = l_u + x_i - y_i$ かつ $e_i = l_u$.
5. $0 < |x_i - y_i| \leq l_d$ のとき : $f_i = x_i - y_i$ かつ $e_i = 0$, または $f_i = 0$ かつ $e_i = -(x_i - y_i)$ と割り当てる. ただし,

$$||\{i : e_i \neq 0\}| - |\{i : f_i \neq 0\}||$$

を最小化するような割り当てを選ぶこととする (\mathbf{f} と \mathbf{e} の非零の要素数をできるだけ等しくする).

いずれの割り当ても $\mathbf{x} - \mathbf{y} = \mathbf{f} - \mathbf{e}$, $-l_d \leq e_i \leq l_u$, $-l_d \leq f_i \leq l_u$ を満たしている. よって, $|\{i : e_i \neq 0\}| \leq t$, $|\{i : f_i \neq 0\}| \leq t$ を示せばよい. $M_1(\mathbf{x}, \mathbf{y}) \leq 2t$ は 1., 3., 4. のいずれかを満たす i が高々 t でることを示し, $M_1(\mathbf{y}, \mathbf{x}) \leq 2t$ は 2., 3., 4. のいずれかを満たす i が高々 t でることを示している. また, $M_2(\mathbf{y}, \mathbf{x})$ の各項は, 条件 1. ~ 5. と以下のように関係している.

- $N[1, \max(l_u, l_d)](\mathbf{x}, \mathbf{y}) + N[-\max(l_u, l_d), -1](\mathbf{x}, \mathbf{y})$ は条件 1., 2., 5. のいずれかを満たす i の数と一致する. これらの条件では e_i と f_i の一方のみが非零である.
- $N[\max(l_u, l_d) + 1, l_u + l_d](\mathbf{x}, \mathbf{y})$ は条件 3. を満たす i の数と一致する. このとき $e_i \neq 0$ かつ $f_i \neq 0$ である.
- $N[-(l_u + l_d), -(\max(l_u, l_d) + 1)](\mathbf{x}, \mathbf{y})$ は条件 4. を満たす i の数と一致する. このとき $e_i \neq 0$ かつ $f_i \neq 0$ である.

したがって, $M_2(\mathbf{y}, \mathbf{x}) \leq 2t$ は \mathbf{e} と \mathbf{f} の非零の要素の総和が, 高々 $2t$ であることを意味する. \mathbf{f} の非零の要素数は 1., 3., 4., 5. のいずれかを満たす i の数であり, \mathbf{f} の非零の要素数は 2., 3., 4., 5. のいずれかを満たす i の数である. よって, 5. の割り当てにより, $|\{i : e_i \neq 0\}| \leq t$ および $|\{i : f_i \neq 0\}| \leq t$ が満たされる. (証明終了)

補助定理 6 任意の 2 つのベクトル $\mathbf{x}, \mathbf{y} \in Q^n$ について, ハミング距離 d_H と LM 距離 d_{LM} は, $d_{LM}(\mathbf{x}, \mathbf{y}) \geq d_H(\mathbf{x}, \mathbf{y})$ を満たす.

(証明) $d_H(\mathbf{x}, \mathbf{y}) \leq n$ なので, $\max_i\{|x_i - y_i|\} > l_u + l_d$ のとき $2n + 1 = d_{LM}(\mathbf{x}, \mathbf{y}) \geq d_H(\mathbf{x}, \mathbf{y})$ となる. すべての i について $\{|x_i - y_i|\} \leq l_u + l_d$ のとき,

$$d_H(\mathbf{x}, \mathbf{y}) = N[1, l_u + l_d](\mathbf{x}, \mathbf{y}) + N[-1, -(l_u + l_d)](\mathbf{x}, \mathbf{y})$$

と表せる。したがって、

$$\begin{aligned}
d_{LM}(\mathbf{x}, \mathbf{y}) &= N[1, \max(l_u, l_d)](\mathbf{x}, \mathbf{y}) \\
&\quad + N[-\max(l_u, l_d), -1](\mathbf{x}, \mathbf{y}) \\
&\quad + 2N[\max(l_u, l_d) + 1, l_u + l_d](\mathbf{x}, \mathbf{y}) \\
&\quad + 2N[-(l_u + l_d), -(\max(l_u, l_d) + 1)](\mathbf{x}, \mathbf{y}) \\
&= N[1, l_u + l_d](\mathbf{x}, \mathbf{y}) \\
&\quad + N[-(l_u + l_d), -1](\mathbf{x}, \mathbf{y}) \\
&\quad + N[\max(l_u, l_d) + 1, l_u + l_d](\mathbf{x}, \mathbf{y}) \\
&\quad + N[-(l_u + l_d), -(\max(l_u, l_d) + 1)](\mathbf{x}, \mathbf{y}) \\
&\geq d_H(\mathbf{x}, \mathbf{y})
\end{aligned}$$

が成り立つ。(証明終了)

定理 17 式 (23) によって定義される符号 C は、 t 双方向 (l_u, l_d) -LM 誤り訂正符号である。

(証明) 任意の異なる 2 つの符号語 $\mathbf{x}, \mathbf{y} \in C$ について、 $d_{LM}(\mathbf{x}, \mathbf{y}) \geq 2t+1$ であることを示す。 $|x_i - y_i| \geq l_u + l_d + 1 = q'$ となる i が存在するとき、 $d_{LM}(\mathbf{x}, \mathbf{y}) \geq 2n+1 \geq 2t+1$ となる。続いて、全ての i について $|x_i - y_i| < q'$ となる場合を考える。 $0 \leq x_i \bmod q' < q'$, $0 \leq y_i \bmod q' < q'$ であるため、

$$-q' < x_i \bmod q' - y_i \bmod q' < q'$$

が成り立つ。したがって、

$$\begin{aligned}
x_i \bmod q' - y_i \bmod q' &= 0 \\
\iff (x_i \bmod q' - y_i \bmod q') \bmod q' &= 0 \\
\iff (x_i - y_i) \bmod q' &= 0
\end{aligned}$$

となる。これは $|x_i - y_i| < q'$ のとき、 $\mathbf{x} - \mathbf{y}$ の非零の要素の数、すなわちハミング距離 $d_H(\mathbf{x}, \mathbf{y})$ は、 $\mathbf{x} \bmod q' - \mathbf{y} \bmod q'$ の非零の要素の数と等しい。これは式 (23) により Ω のハミング距離 $2t+1$ 以上であるため、補助定理 6 を用いれば、 $d_{LM}(\mathbf{x}, \mathbf{y}) \geq d_H(\mathbf{x}, \mathbf{y}) \geq 2t+1$ となる。(証明終了)

7.3 提案手法

t 双方向 (l_u, l_d) -LM 誤り訂正符号の構成法を提案する。今回構成する符号は、パラメータを $q = 2^b, q' = 2^m$ (b, m は $b > m$ を満たす正整数) に限定するが、MLCにおいて $q = 2^b$ は典型的なパラメータであり、 $q' = 2^m$ はページ毎に適用された誤り訂正

符号を利用するための制約である。提案符号による誤り訂正は以下のような手順で行う。

1. 誤りの値 $e(-l_d \leq e \leq l_u)$ を、ビット誤り訂正符号を用いて推定する。
2. q 元の誤りシンボルから e を Q_q 上のシンボルとして減算することで、誤りを訂正する。

これは従来の LM 誤り訂正符号の手順と同様であるが、手順 1. を実行するために q' 元の符号を用いず、ページ毎の 2 元の誤り訂正符号を用いて実現する点が異なる。このようにすることで、マルチページプログラミングの符号構成を変更せずに、LM 誤り訂正を実現できる。手順 2. では Q_q 上で演算を行うため、2 元と q 元間の写像が必要となる。通常の 2 進数を写像として用いる場合、式(23)における “mod q' ” は、下位の m ビットのみ取り出し、符号化すれば実現できる。加えて、上位 $b - m$ ビットが符号化されないので、次章に示すように検査部を活用できる。ところが、誤り率のパフォーマンスは、どのような写像を用いるかによって異なってくる。これは、手順 1. が 2 元の符号を用いて実行されるため、2 元に写像したときに、多ビット誤りが生じにくい写像が好ましいためである。したがって、下位 m ビットのみの符号化で LM 誤り訂正を実現可能で、かつ誤り率のパフォーマンスに優れた写像を提案する。

7.3.1 写像

具体的な写像を示す前に、特定の m ビットを符号化することが、 q' を法とした剰余の符号化と対応するために、写像が満たすべき条件について議論する。写像 f を以下のように定義する。

$$\begin{aligned} f : Q_q &\rightarrow Q_2^m \times Q_2^{b-m} \\ x &\mapsto (\mathbf{a}, \mathbf{b}) \end{aligned} .$$

つまり、 f は q 元のシンボル x から、下位 m ビット \mathbf{a} および上位 $b - m$ ビット \mathbf{b} から成る、 b ビットへの写像である。 f が満たすべき条件が 2 つある。まず、写像を行う前後で同等な情報を扱うため、 f は全単射である必要がある。次に、 $x \bmod q'$ を符号化することと、 \mathbf{a} を符号化することが同等であるために、これらの間に 1 対 1 の対応関係がなければならない。これは、 $g(x \bmod q') = \mathbf{a}$ なる全単射 $g : Q_{q'} \rightarrow Q_2^m$ が存在することと同義である。 g が全単射でない場合、前述の e の減算という手順をとることができない。これは以下に示す例からも分かる。

例 16 $b = 3, m = 2$ とし、 $-1 \leq e \leq 2$ なる e を求める手順を考える。写像 f は、 $f(2) = (110), f(3) = (010), f(5) = (111)$ を満たすものとする。 f は全単射となり得るが、 $g(2) = (11), g(3) = (01), g(1) = (11)$ となるため、 g は全単射ではない。下位 2 ビットが (01) から (11) への誤りであることが、適用した誤り訂正符号の機能によ

表 16: 16 元の整数値から 2 元のビット列への写像.

整数値	2 進写像	グレイ写像	提案写像 ($m = 2$)	提案写像 ($m = 3$)
0(消去)	0000	0000	0000	0000
1	1000	1000	1000	1000
2	0100	1100	1100	1100
3	1100	0100	0100	0100
4	0010	0110	0010	0110
5	1010	1110	1010	1110
6	0110	1010	1110	1010
7	1110	0010	0110	0010
8	0001	0011	0001	0001
0	1001	1011	1001	1001
10	0101	1111	1101	1101
11	1101	0111	0101	0101
12	0011	0101	0011	0111
13	1011	1101	1011	1111
14	0111	1001	1111	1011
15	1111	0001	0111	0011

り推定できたとしよう. このとき, $e = -1$ (3 から 2 への変化) なのか, $e = 2$ (3 から 5 への変化) なのかを, 区別することができない. 一方, もし g が全単射であれば, 同様な手順で求まる e は明らかに一意である.

本章で扱う各写像の 16 元(4 ビット)の場合を, 表 16 に例として示す. (本章での数式と統一するため, 左端を下位ビットとしている. また, 消去状態である整数値 0 に系列 0...0 を対応したものであるが, 提案手法 2 の表 13 のように 1...1 を対応させた場合でも, 訂正の手順やパフォーマンスに変化はない.)

定理 18 2進数を用いた写像は, 上記 f に関する 2 つの条件を満たす.

(証明) f は明らかに全単射であるので, g の存在を示す. 任意の $x \in Q_q$ は $a_i \in Q_2 (1 \leq i \leq b)$ を用いて次のように表すことができる.

$$x = a_1 + 2a_2 + 2^2a_3 + \cdots + 2^{b-1}a_b.$$

このとき, 写像 f は $f(x) = (a_1, \dots, a_b)$ によって定義される. ここで,

$$x \bmod q' = a_1 + 2a_2 + 2^2a_3 + \cdots + 2^{m-1}a_m.$$

と表せるため, $g(x \bmod q') = (a_1, \dots, a_m)$ なる全単射を定義できる. (証明終了)

以降, この写像を 2 進写像と呼ぶこととする. 近傍の値へ向かう誤りが多発する場合に, グレイ符号を利用した写像(以降, グレイ写像と呼ぶ)が有効な対策として知られている [9, 18]. これは隣接した 2 つの整数值間で, それらを写像したビット

列間のハミング距離が常に 1 となるためである。例えば、表 16 の 2 進写像において、隣接した値 3 から 4 への誤りは 3 ビット誤りとなるが、グレイ写像では单一ビット誤りとなる。これにより誤り率のパフォーマンスが改善される。ところが、このような「ハミング距離が常に 1」という特徴を持つグレイ写像はいかなる場合も、今回想定する写像 f の条件を満たさない。

定理 19 グレイ写像は、 f に関する条件を満たさない。

(証明) グレイ写像が f の条件を満たすと仮定し、矛盾を導く。 $\mathbf{a} \in Q_2^m$ とし、 $\mathbf{b} \in Q_2^{b-m}$ とし、 $f(0) = (\mathbf{a}, \mathbf{b})$ が成り立つとする。 $g(q' \bmod q') = g(0 \bmod q') = \mathbf{a}$ が成り立ち、 f は全単射だから、 $f(q') = (g(q' \bmod q'), \mathbf{b}') = (\mathbf{a}, \mathbf{b}')$ (ただし、 $\mathbf{b}' \in Q_2^{b-m}$ は $\mathbf{b} \neq \mathbf{b}'$ を満たす) が成り立つ。また、 g は全単射だから $i \neq j$ なる $i, j \in Q_q'$ について、 $g(i) \neq g(j)$ が成り立つ。したがって、 $f(i)$ と $f(i+1)$ の下位 m ビットは、 $0 \leq i < q'-1$ のとき常に異なっている。グレイ符号のビット変化は 1 ビットであることを考えると、 $0 \leq i \leq q'-1$ について $f(i)$ の上位 $b-m$ ビットは \mathbf{b} である。さらに、 $f(0)$ と $f(q'-1)$ の下位 m ビットもまた異なるため、ある $\mathbf{a}' \in Q_2^m$ ($\mathbf{a}' \neq \mathbf{a}$) を用いて、 $f(q'-1) = (\mathbf{a}', \mathbf{b})$ と表せる。以上により $f(q'-1)$ と $f(q')$ のハミング距離が、少なくとも 2 以上となることが示されたが、これは矛盾である。(証明終了)

ここで、上記 f の条件を満たし、かつ計算の実現性、および誤り訂正機能において優れた写像を提案する。この写像は、上位 $b-m$ ビットは 2 進写像と一致し、下位 m ビットはグレイ写像の特徴をもつ。この写像は、グレイ写像のもつ重大な 2 つの特徴を備えている。まず、2 進数から容易に変換することができ、加えて下位 m ビットにおいては、隣接値間の誤りに対するビット反転を 1 ビットに制限できる。以降、この写像を提案写像と呼ぶこととする。以下のアルゴリズムによって構成される。

アルゴリズム 4 (提案写像) 写像にもちいるビット列を、以下の手順により構成する。

1. $i \leftarrow 1$ とし、1 ビットの系列から始める。
2. $i < m$ ならば、既存の i ビットの系列の順序を反転したものを新たな系列として加える。 $i \geq m$ ならば、既存の i ビットの系列を複製したものを新たな系列として加える。
3. もとから存在した系列には 0 を、新たに付加した系列には 1 を、 $i+1$ ビット目として付加する。
4. $i \leftarrow i+1$ とする。
5. $i = b$ ならば、アルゴリズムを終了する。
6. 手順 2. に戻る。

定理 20 アルゴリズム 4 によって構成された写像は、 f の条件を満たす。

0	0	00	00	000
1	1	10	10	100
	1	11	11	110
	0	01	01	010
			00	001
			10	101
			11	111
			01	011
(a)	(b)	(c)	(d)	(e)

- (a) 1 ビットの系列. (b) 1 ビットの系列の順序を反転したものを加える. (c) 0 と 1 を上位に付加.
(d) 2 ビットの系列を複製したものを加える. (e) 0 と 1 を上位に付加.

図 16: $b = 3, m = 2$ の場合の提案写像の構成法

(証明) 下位 m ビットに着目する (アルゴリズム中での, 1 から m ビット目に相当する). これは m ビットグレイ写像の構成法 [70] に基づいており, ゆえに整数値 0 から $q' - 1$ に対応するビット系列はグレイ符号である. 続く系列は上位 $b - m$ ビットを除いて, グレイ符号の複製であるため, x と $x + q'$ を f により写像したもの下位 m ビットは一致する. したがって, 全単射 g を m ビットのグレイ写像として定義できる. さらに, 上位 $b - m$ ビットは 2 進写像の構成法と同様であり, 複製された各々の m ビットグレイ符号の間で, 同じ系列が付加されることはない. よって f は全単射となる. (証明終了)

$b = 3, m = 2$ とした場合の提案写像の構成手順を, 図 16 に示す.

7.3.2 符号構成法

提案写像を用いれば下位 m ビットが $x \bmod q'$ と対応するので, あとはそれらのビットを用いて, 式 (23) における Ω を構成すればよい. 以下に示すように, マルチページプログラミングと併用されるページ毎のビット誤り訂正符号を用いて, Ω を構成することができる. $\mathbf{v} = (v_1, \dots, v_n)$ を q 元の符号語とし, $f(v_i) = (u_{i,1}, \dots, u_{i,b})$ とする. ただし, f は前節の条件を満たす任意の写像とする. これにより, セルの第 i ビットから構成されるページは $\mathbf{u}_j = (u_{1,j}, \dots, u_{n,j})(1 \leq j \leq b)$ と表すことができる. $\mathbf{u}_1, \dots, \mathbf{u}_m$ は, それぞれ t ビット誤り訂正符号の符号語とする. 言い換えると, $\mathbf{u}_j(1 \leq j \leq m)$ の情報部を t ビット誤り訂正符号によってそれを符号化することで, 符号 C を構成する.

定理 21 上記手順で構成された q 元の符号 C は, t 双方向 (l_u, l_d) -LM 誤り訂正符号である.

(証明) $w_i = g^{-1}(u_{i,1}, \dots, u_{i,m})$, $w'_i = g^{-1}(u'_{i,1}, \dots, u'_{i,m})$ とし, $\mathbf{w} = (w_1, \dots, w_n)$, $\mathbf{w}' = (w'_1, \dots, w'_n)$, とする. 式(23)により, $\mathbf{w} \neq \mathbf{w}'$ について, ハミング距離 $d_H(\mathbf{w}, \mathbf{w}') \geq 2t + 1$ となることを示せばよい. C_2 を t ビット誤り訂正符号都市, $\mathbf{u}_j, \mathbf{u}'_j \in C_2$ とする. $\mathbf{u}_j \neq \mathbf{u}'_j$ ならば, 明らかに $d_H(\mathbf{u}_j, \mathbf{u}'_j) \geq 2t + 1$ である. $\mathbf{w} = \mathbf{w}'$ であることと $\mathbf{u}_j = \mathbf{u}'_j$ が $1 \leq j \leq m$ なるすべての j について成り立つことは同値なため, 少なくとも 1 つの j について, $d_H(\mathbf{u}_j, \mathbf{u}'_j) \geq 2t + 1$ が成り立つ. g は全単射であるから, $d_H(\mathbf{w}, \mathbf{w}') \geq 2t + 1$ が成り立つ. (証明終了)

t は最低限保証される LM 誤り訂正機能であり, ビット誤り訂正が可能な限り, より多くの LM 誤りが含まれる場合でも訂正可能である. これは重要な特徴であり, 隣接値へ誤った場合のビット反転を抑える提案写像は, これを利用して 2 進写像より優れた誤り訂正率を実現する.

7.3.3 復号法

提案手法は下位 m ビットのみが符号化されるため, 通常のビット反転では, 上位 $b - m$ ビットの誤りが訂正できない. 以下に示す LM 誤り訂正を実現するアルゴリズムを利用すれば, 上位 $b - m$ ビットに誤りが含まれる場合も訂正できる. $\mathbf{u}_j^{(r)}$ を受信した \mathbf{u}_j とし, それを訂正したものを $\mathbf{u}_j^{(c)}$ とする. 受信語を $\mathbf{v}^{(r)}$, それを訂正したものを $\mathbf{v}^{(c)}$, 推定した誤りを $\mathbf{e} = (e_1, \dots, e_n)$ とする.

アルゴリズム 5 (提案手法 3 の復号) 以下の手順により実行する.

1. $1 \leq j \leq m$ について, $\mathbf{u}_j^{(r)}$ に対してビット誤り訂正符号をそれぞれ用いることで, $\mathbf{u}_j^{(c)}$ を求める.

2. $1 \leq i \leq n$ について, $e_i^{(0)} = g^{-1}(u_{i,1}^{(r)}, \dots, u_{i,m}^{(r)}) - g^{-1}(u_{i,1}^{(c)}, \dots, u_{i,m}^{(c)})$ を求める.

3. $1 \leq i \leq n$ について,

$$e_i = \begin{cases} e_i^{(0)} & (0 \leq e_i^{(0)} \leq l_u) \\ e_i^{(0)} - q' & (l_u < e_i^{(0)} < q') \end{cases}$$

を求める.

4. 訂正語を $\mathbf{v}^{(c)} = \mathbf{v}^{(r)} - \mathbf{e}$ により求める.

5. アルゴリズムを終了する.

この手法では, 符号化および復号のために写像 g (または g^{-1}) を用いる必要がある. 特に 2 進写像を用いる場合, 復号アルゴリズムの手順 2. における減算をハードウェアで実装する場合, 整数の減算器を用いることができる. 続いて提案写像について考える. ある整数を 2 進数で表示したものを (c_1, \dots, c_m) , グレイ符号 ([70] に示さ

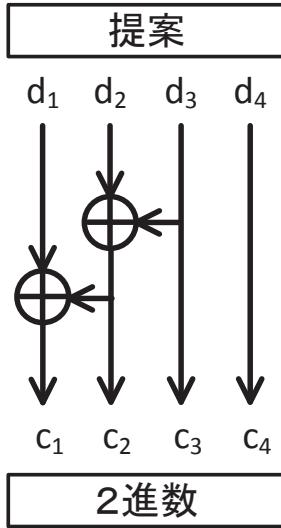


図 17: 提案写像の系列から 2 進数への変換回路.

れる反射 (reflection) の手順により構成されたもの) で表示したものを, (d_1, \dots, d_m) とする. このとき,

$$\begin{aligned} c_m &= d_m, \\ c_k &= c_{k+1} + d_k (1 \leq k \leq m-1), \end{aligned}$$

が満たされる. これにより提案写像から 2 進数まで容易に変換することができ, 同様に減算器の利用が可能となる. 図 17 に $b = 4, m = 3$ の場合の変換回路を示す. 図中の (d_1, d_2, d_3, d_4) は提案写像の系列であり, (c_1, c_2, c_3, c_4) は変換後の 2 進数である.

セルを共有する b ページをシーケンシャルに読み出すのであれば, 提案符号は容易に復号することができる. しかし, ランダムに読み出す場合, 操作は複雑になる. 従来のページ毎の誤り訂正であれば, 単一のページのみで復号が完結したが, LM 誤り訂正を実行するためには, 少なくとも下位 m ページすべてを揃える必要がある. 次節で扱う追加の誤り訂正を考慮すると, b ページすべてが単一ページの復号に際して必要である.

7.4 追加の誤り訂正

NAND フラッシュメモリは, あらかじめ誤り訂正の検査部等に利用する, 予備のメモリセルを備えている. 提案符号においては上位 $b - m$ ページが符号化されないため, 従来と比較して新たに未使用となる部分が現れる. 通常, メモリシステムにおいては情報ビット長は 2 の幂乗に限定されるため, 未使用な部分を新たに情報部

として利用することは、通常できない。そこで、この部分を検査部として利用することで、誤り訂正を追加で行う手法を提案する。

候補となる手法が2つある。1つ目は、混合LM誤り訂正符号[71]の応用である。これは、LM誤り訂正機能に加えて、(LM誤りではない)対称シンボル誤りの訂正機能を持たせる手法である。式(23)を用いて構成されたLM誤り訂正符号に、定義された範囲外の誤りが発生した場合、下位 m ページの誤りは正しく訂正されるが、上位 $b-m$ ページには、それぞれ最大で t 個の誤りが残る。そこで、これら上位ページに対して別途誤り訂正符号を適用し、LM誤り訂正後に追加で訂正を実行することで、これを訂正することが可能となる。この手法は、誤りが理想的なLM誤りモデルではなく、定義された範囲外の誤りも、無視できない確率で発生するような場合に有効となる。

2つ目は、部分的連接符号と呼ばれる手法[71]を、応用したものである。写像と誤りの特徴から、NANDフラッシュメモリのセルに属する各ビット(ページ)間で、誤り率に差がある。この符号は各ページに適用する誤り訂正符号のうち、上位ページに適用する誤り訂正符号の機能を弱めることで検査を節約し、それをを利用して下位ページの誤り訂正を、追加で行う手法である。追加の誤り訂正については、連接符号に類した方法を用いる。提案手法においては、もともと上位ページの検査が未使用であるため、上位ページの訂正機能を配慮せずに、下位ページの訂正機能を向上できる。しかし、この手法の適用によって各ページ間の符号化に依存関係が生じるため、マルチページプログラミングとの併用は不可となる。このため、この追加の誤り訂正是P/Eサイクルが上昇し、信頼性の確保が困難となった段階(すなわち寿命)以降に用いることとする。

7.5 実装

7.5.1 符号効率

情報長を k 、符号長を n とすると、従来手法の符号化率は $R_c = k/n$ である。同様に、提案手法の構成に用いた2元の符号の情報長を k 、符号長を n とする。上位 $b-m$ ビットが符号化されていないため、下位のページの検査に相当するビットに、追加の情報を格納できる。よって符号化率は、

$$\begin{aligned} R_p &= \frac{kb + (n - k)m}{nb} \\ &= R_c + (1 - R_c) \cdot \frac{m}{b} \end{aligned}$$

となり、改善される。しかし、情報長の2の幂乗への制限を考慮すると、各上位ページにおける検査に相当する、 $n - k$ ビット程度のわずかな増加では、追加の情報は格納できない。この場合の符号化率は R_c に等しい。したがって、前節に示したような追加の誤り訂正への利用が有効である。

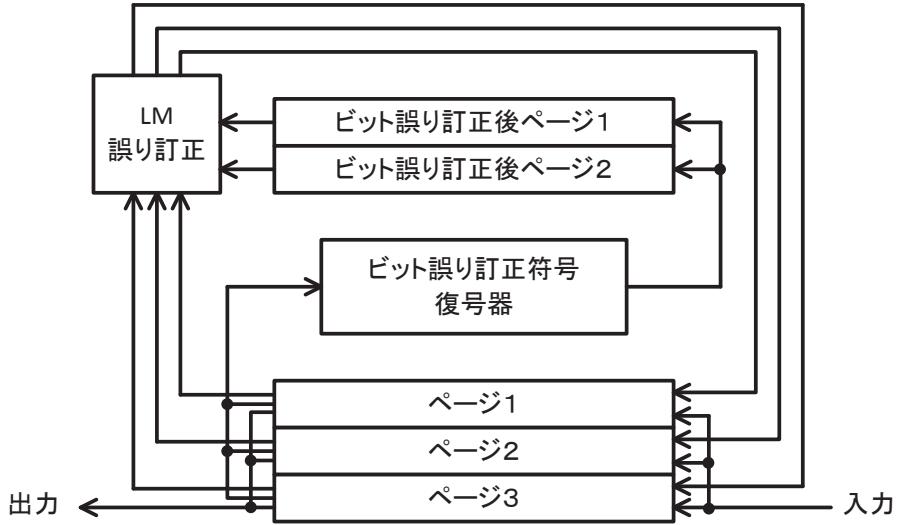


図 18: 復号回路.

7.5.2 復号回路

提案符号のハードウェアによる実装について考える。 $b = 3, m = 2$ の場合の復号回路のブロック図を、図 18 に示す。復号回路は、1 つのビット誤り訂正符号の復号回路、 b ページ分(サイズにして nb ビット)のページバッファ、 m ページ分のビット誤り訂正後のページを格納するバッファ、および復号アルゴリズムの手順 2. における、整数による減算を実行するための LM 誤り訂正ブロックから成る。従来のページ毎のビット誤り訂正符号の場合、ページバッファのサイズは n ビットであるのに対し、提案符号ではページ間にまたがる計算を必要とするため、ページバッファのサイズは $n(b + m)$ 必要となる。一方、提案符号の下位 m ページの誤り訂正符号は従来手法と同じため、各ページを逐次的に訂正することにより、ビット誤り訂正のための回路は従来と同様 1 つでよい。追加の誤り訂正是、訂正機能が下位 m ページと必ずしも一致しないため、通常別途でハードウェアが必要である。しかし構成に BCH 符号を用いるならば、訂正機能の変更が容易であるため [62]、インターリーヴされ実行すれば、同じ回路を流用できる。LM 誤り訂正ブロックは、写像 g^{-1} を行う回路と減算器から成る。第 7.3.3 項で述べたように、2 進写像や提案写像の場合、 g^{-1} のための回路は極めて簡単である。

7.5.3 書き込みにおけるレイテンシとスループット

書き込み時に発生するレイテンシを考える。 t_{IO} を入出力に要する時間、 $t_{pg,i}$ を下位から i 番目のページの書き込みに要する時間から t_{IO} を除いたもの、 t_{enc} を誤り訂正符号の符号化に要する時間とする。従来のページ毎のビット誤り訂正の場合、 i 番

目のページのレイテンシ $L_{pg_i}^c$ は,

$$L_{pg_i}^c = t_{enc} + t_{IO} + t_{pg_i}$$

となる. 提案符号としては追加の誤り訂正を行うものを想定する. この場合, どのページを書き込む場合も, b ページすべてに書き込みを行う必要があり, レイテンシに差異はない. 追加の誤り訂正のための符号化に要する時間を t_{aenc} とする. 通常, t_{IO} , t_{enc} , t_{aenc} は t_{pg_i} より十分小さく, 各ページの書き込みをパイプラインで実行すれば, 最下位ページを除き t_{IO} , t_{enc} , t_{aenc} によるレイテンシはない. したがって, 書き込み全体のレイテンシ L_{pg}^p は,

$$L_{pg}^p = t_{enc} + t_{IO} + \sum_{i=1}^b t_{pg_i}$$

となる.

続いて, 書き込みにおけるスループットを考える. ここでは, 1 ページ目から b ページ目までが, シーケンシャルに書き込まれるものとする. この場合, 最下位のページの入出力と各ページの t_{pg_i} を要する書き込みを除き, パイプラインにより並列に実行される. この操作に従来符号と提案符号による差異はなく, 従来手法のスループット T_{pg}^c , 提案符号のスループット T_{pg}^p は,

$$T_{pg}^c = T_{pg}^p = \frac{nb}{t_{IO} + \sum_{i=1}^b t_{pg_i}}$$

となる.

7.5.4 読み出しにおけるレイテンシとスループット

読み出し時に発生するレイテンシを考える. t_{rd_i} を i 番目のページの読み出しに要する時間から t_{IO} を除いたものとする. ただし, 読み出しには必要最小限の閾値電圧レベルのセンシングを実行するものとする. t_{dec_i} を i 番目のページのビット誤り訂正に必要な時間とする. 従来のページ毎のビット誤り訂正の場合, i 番目のページのレイテンシ $L_{rd_i}^c$ は,

$$L_{rd_i}^c = t_{rd_i} + t_{IO} + t_{dec_i}$$

となる. 提案符号としては追加の誤り訂正を行うものを想定する. b ページすべての読み出しを実行しなければ必要なデータが得られず, 訂正ができない. したがって, どのページの読み出し時もレイテンシに差異はない. ここで, すべての i について, $t_{IO} > t_{dec_i}$ を仮定する(これを満たさない大きな復号時間は, メモリに許容されない復号時間と見なす). このとき, 入出力時間は各ページの訂正とパイプラインで実行でき, 最初のページを除いて考慮する必要はない. また, LM 誤り訂正の

表 17: レイテンシとスループットの比較.

従来符号	$L_{pg_i}^c$	T_{pg}^c	$L_{rd_i}^c$	T_{rd}^c
	319μs	115Mb/s	144μs	318Mb/s
提案符号 A	L_{pg}^p	T_{pg}^p	L_{rd}^p	T_{rd}^p
	853μs	115Mb/s	366μs	298Mb/s
提案符号 B	L_{pg}^p	T_{pg}^p	L_{rd}^p	T_{rd}^p
	853μs	115Mb/s	387μs	298Mb/s

ための整数の減算に必要な時間を t_{LM} とする. これは第 7.5.2 項で述べたように簡単な演算であるが, どの程度並列化するかに依存する. 以上により, 提案符号のレイテンシ L_{rd}^p は

$$L_{rd}^p = \sum_{i=1}^b t_{rd_i} + t_{IO} + t_{dec_b} + t_{LM}$$

となる.

続いて, 読み出しのスループットについて考える. セルに属する b ページをシーケンシャルに読み出すことを想定する. 従来符号の場合, まず b ページすべてをメモリのもつページバッファに読み出し, 各ページの入出力を行う. t_{dec_i} はパイプライン化によって考慮する必要はなく, t_{IO} は最初のページのもののみを考えればよい. したがって, スループット T_{rd}^c は

$$T_{rd}^c = \frac{nb}{t_{IO} + \sum_{i=1}^b t_{rd_i}}$$

となる. 提案符号においても, 違いはパイプライン化のできない t_{LM} のみであるため, スループット T_{rd}^p は

$$T_{rd}^p = \frac{nb}{t_{IO} + t_{LM} + \sum_{i=1}^b t_{rd_i}}$$

となる.

例 17 3ビット/セルにおける, レイテンシとスループットの比較を表 17 に示す. 構成に用いるビット誤り訂正符号は BCH 符号とし, 6.5 節と同様に, [13] の方法で実装することを想定する. クロックは $50MHz$ とし, $p = 8$ に設定する. 符号長 n , 情報長 k , t 訂正の短縮 BCH 符号を (n, k, t) 符号と表記することとする. 従来符号, 提案符号 A, 提案符号 B を以下のような構成とし, 比較を行った.

従来符号 各ページに $(8552, 8192, 24)$ 符号を用いる.

提案符号 A 下位 2 ページには $(8552, 8192, 24)$ 符号を用いる. 追加の訂正是部分的連接符号のみとし, $(8732, 8552, 12)$ 符号を用いる.

提案符号 B 下位 2 ページには $(8552, 8192, 24)$ 符号を用いる。追加の訂正は部分的連接符号として $(8702, 8552, 10)$ 符号、混合 LM 誤り訂正符号として上位ページに $(8552, 8492, 4)$ 符号を用いる。

なお、レイテンシは構成に用いた各符号による誤り訂正が実行されるか否かで変化し、これは誤りモデルに依存して変化する。ここでは最大値を示すこととする。 t_{rd_i} は i によらず $90\mu s$ 、書き込み速度はページによらず $15MB/s$ 、ページサイズは $4KB$ 、入出力バンド幅を $1Gb/s$ に設定している。また、LM 誤り訂正の整数演算は 32 重に並列化し、この場合 $t_{LM} = 20\mu s$ となる。また、提案符号の誤り訂正是 $(LM \text{ 誤り訂正}) \rightarrow (\text{部分的連接符号}) \rightarrow (\text{混合 LM 誤り訂正符号})$ の順に、一度だけ実行する。提案符号では書き込みレイテンシが $534\mu s(167\%)$ 大きくなり、読み出しレイテンシは提案符号 A の場合は $222\mu s(154\%)$ 、提案符号 B の場合は $243\mu s(169\%)$ 大きくなる。また、読み出しがスループットは 6.3% 低下する。

7.5.5 追加訂正によるページサイズ変更の影響

追加の誤り訂正を行う提案符号では、符号化におけるページ間の依存関係により、 b ページをまとめて書き込みを行う必要がある。これは、前項で議論したレイテンシとスループットの低下のみでなく、実質的な容量の低下を招く。 u ページをシーケンシャルに書き込む場合、実際には $b[u/b]$ ページが必要となるためである。したがって、利用できるページの割合は、ページ全体の $(u/b)/([u/b])$ へと減少する。ただし、書き込まれるデータのサイズが大きければ、容量の低下は軽減される。NAND フラッシュメモリのページサイズは、典型的には $16KB$ であり、このような容量の低下が問題とならないアプリケーションも多い。例えば、コンシューマ PC 向けのストレージは、扱うデータサイズはそれほど小さくはなく、キャッシングを用いてある程度のサイズになった時点で書き込むことも可能である。

7.6 誤り率のシミュレーション

7.6.1 誤りモデル

3 ビット/セルの MLC における誤り訂正のシミュレーションにより、 t 双方向 $(-2, 1)$ -LM 誤り訂正が可能な各種法の誤り率の比較を行う。今回想定する誤りは、「近傍の値へむかう誤りほど発生率が高い」という考え方に基づく。しかし、実際にある値の変化を起こす誤りが、どの程度の確率で発生するかを決定するのには、困難が伴う。例えば、[8] では 2 ビット/セルにおける電荷保持誤りの実験で、2 レベルの変化を起こす誤りが全体の 5% であり、その他の誤りは 1% であることが示されているが、3 レベル以上の誤りについてのデータは示されていない。一方、[72] では 3 ビット/セルにおける実験において、2 レベル以上の誤りは全体の 0.02% 以下であったと示されている。このような状況を考え、ここでは以下の 2 つの誤りモデルを用いて実

験を行うこととする. $P(e)$ を, あるシンボル誤りによる値の変化が e である確率とする.

MODEL-L LM 誤りモデルであり, 誤りの値 e は $-2 \leq e \leq 0$ を満たす. すなわち, 2 レベルの誤りと 1 レベルの誤りのみが発生する.

MODEL-E 各レベルの誤りの発生率が, 誤りの値 e の減少に伴い指数関数的に減少するモデルである. あるパラメータ α に対して,

$$P(e) = \begin{cases} \alpha^{|e|-1} P(-1) & (e < 0), \\ 0 & (e \geq 0), \end{cases}$$

および

$$\sum_{e=-(2^b-1)}^{-1} P(e) = 1$$

を満たすものと定義する.

これらは減少する誤りのみを対象としているが, これは以下の 2 つの理由による. NAND フラッシュメモリにおいては増加と減少, 両方の誤りが発生し得るが, それぞれの要因となる物理現象は異なり, 独立して作用する. このため, ある時点で強く作用している一方が高確率で発生し, 値の変化も大きいためである. 加えて, 今回シミュレーションする LM 誤り訂正符号は, 訂正可能な範囲にある誤りについて, それが増加でも減少でも, 同じ訂正機能を持つ. したがって, 例えば MODEL-L は, 同じ 2 レベル誤りの発生率で $-2 \leq e \leq 1$ とした結果と同等である.

7.6.2 結果

評価はセルに何らかの誤りが発生する確率であるセル誤り率に対する, 同一ワード線に属するページ群に誤りが含まれる確率であるワード線誤り率によって行う. 図 19~22 は, 各誤りモデルにおけるシミュレーション結果である. 評価に用いた符号を以下に示す. 構成要素となる符号には短縮 BCH 符号 (n, k, t) 符号 (n は符号長, k は情報長, t は訂正可能な誤りの数) を用いることとし, 以降は単に (n, k, t) と表記する.

提案 1 提案符号の構成による LM 誤り訂正符号. 写像 f として 2 進写像を用いたもの. ひとつのページを 8 つのセクタに分割し, それぞれにビット誤り訂正符号として $(160, 128, 4)$ 符号を用いた.

提案 2 提案符号の構成による LM 誤り訂正符号. 写像 f として提案写像を用いたもの. ひとつのページを 8 つのセクタに分割し, それぞれにビット誤り訂正符号として $(160, 128, 4)$ 符号を用いた.

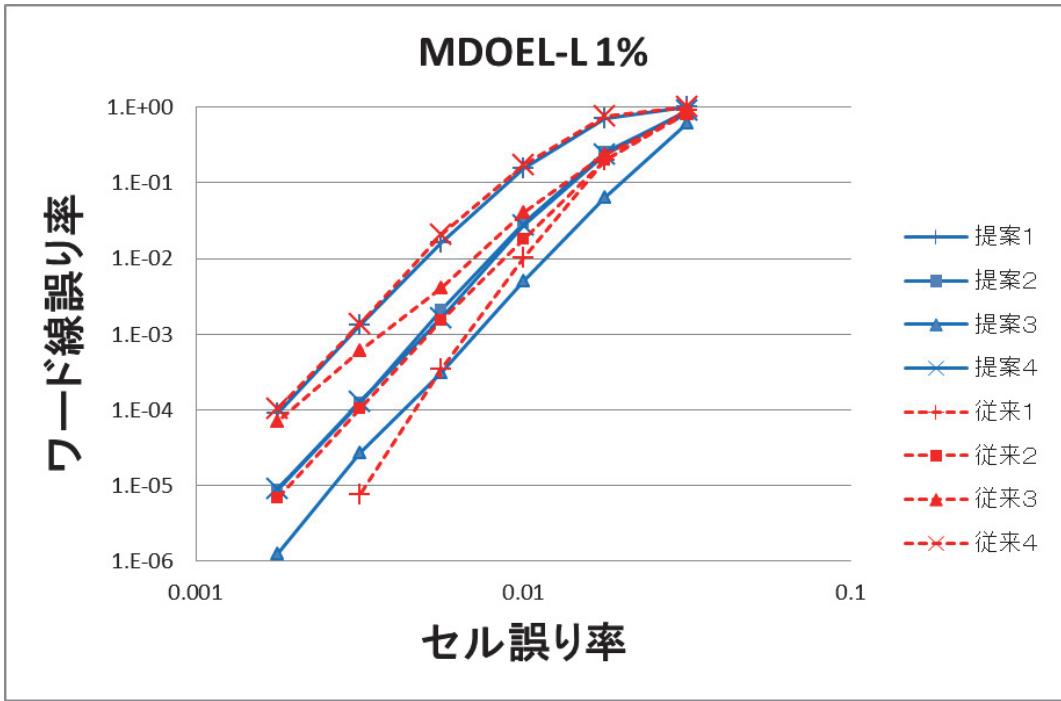


図 19: ワード線誤り率の比較 (MODEL-L, 2 レベル誤り率 1%).

提案 3 提案 2 の構成に加え、部分的連接符号により追加の誤り訂正をしたもの。追加の符号として、下位 2 ページの各セクタを 16 に分割して、それぞれの部分に対して (176, 160, 2) 符号を用いた。復号は (通常のビット誤り訂正) → (追加のビット誤り訂正) の順に一度だけ実行し、最後に LM 誤り訂正を行った。

提案 4 提案 2 の構成に加え、混合 LM 誤り訂正符号と部分的連接符号により、追加の誤り訂正をしたもの。部分的連接符号として、下位 2 ページの各セクタを 16 に分割して、それぞれの部分に対して (168, 160, 1) 符号を用いた。混合 LM 誤り訂正符号として、(160, 144, 2) 符号を上位のページに用いた。復号は (通常のビット誤り訂正) → (部分的連接符号のビット誤り訂正) の順に一度だけ実行し、続いて LM 誤り訂正を実行し、最後に上位のページのビット誤り訂正を行った。

従来 1 多元の符号を用いた LM 誤り訂正符号 [31]。構成には 4 元の (160, 128, 6) 符号を用いた。

従来 2 ページ毎のビット誤り訂正符号。ひとつのページを 8 つのセクタに分割し、それぞれに (160, 128, 4) 符号を用いた。

従来 3 不均一誤り訂正符号 [73]。下位 2 のページを 8 つのセクタに分割し、それぞれに (160, 128, 4) 符号を用いた。部分的連接符号としては (168, 160, 1) 符号を用いて、上位ページの符号としては (160, 144, 2) 符号を用いた。復号は (下位 2 ページのビット誤り訂正) → (部分的連接符号のビット誤り訂正) の順に一度

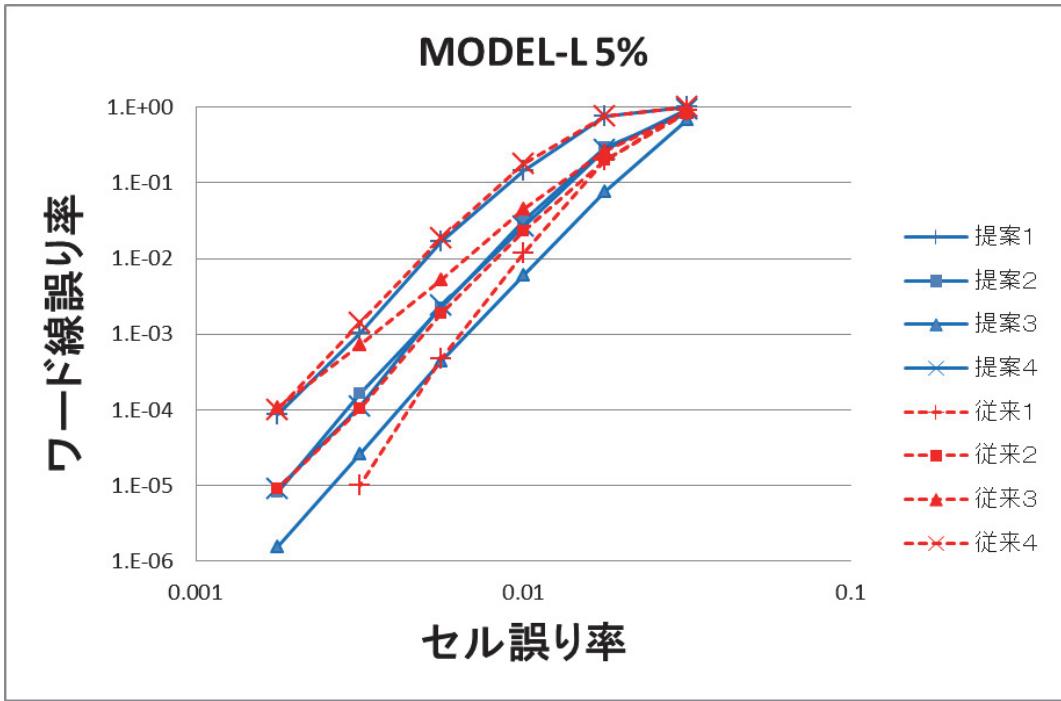


図 20: ワード線誤り率の比較 (MODEL-L, 2 レベル誤り率 5%).

だけ実行し、最後に上位のページのビット誤り訂正を行った。

従来 4 混合 LM 誤り訂正符号 [71]. 下位 2 ページには 4 元の (160, 128, 4) 符号を用い、上位ページには (160, 128, 4) 符号を用いた。

これらの符号はすべて、検査部の格納に必要なセル数が同じである。また、実際の NAND フラッシュメモリのページサイズは~16KB 程度とより大きいが、シミュレーションを簡単にするためこのように設定した。

図 19 は、MODEL-L において 2 レベル誤りの割合を 1% に設定した場合の結果である。実際の NAND フラッシュメモリでは $10^{-13} \sim 10^{-16}$ 程度のビット誤り率が要求されるため、セル誤り率の低い領域に注目する。まず、提案 1 と提案 2 を比較すると、提案 2 のほうが 10 倍程度優れたワード線誤り率を達成している。これは提案写像が下位 2 ページのビット誤りを低減できる特徴によるもので、写像の導入は適切であることが確認できる。提案符号の中では、追加の誤り訂正を行う提案 3 が最も優れた誤り率を達成できる。特に、MODEL-L のような理想的な LM 誤りモデルにおいては、上位ページの誤り訂正が不要であり、提案 4 と比較しても優れた結果となっている。従来符号も含めると、従来 1 が最も優れた誤り率となっている。しかし、マルチページプログラミングが許容されない手法であり、構成も全く異なることから、切り替える場合は別途に符号化および復号のハードウェアが必要となる。一方、従来 2 は誤り率において、提案 4 に劣るため、従来 2 から提案 4 への切り替えにより、マルチページプログラミングと寿命の延長という目標を達成できる。な

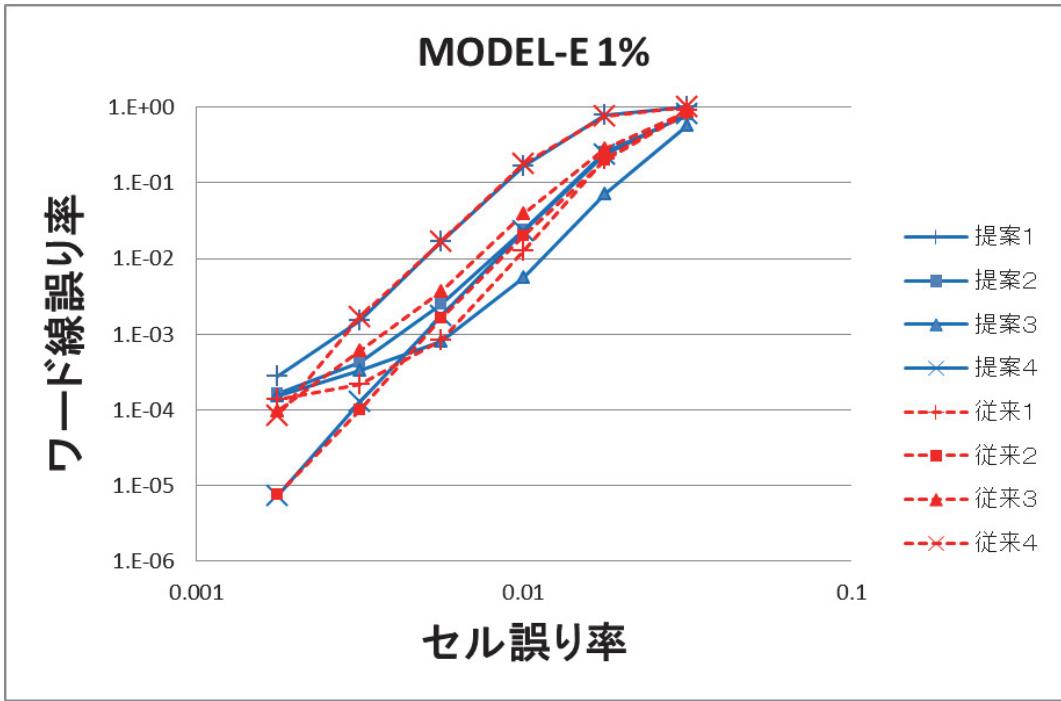


図 21: ワード線誤り率の比較 (MODEL-E, $\alpha = 0.01$).

お, [73] の部分的連接符号で誤り率の改善を行うためには, 訂正機能を適切に設定する必要があり, 訂正数が小さい今回のパラメータでは, 従来 2 より優れた設定ができない. このため, 従来 2 が従来 3 より優れた結果となっている. 図 20 は 2 レベルの誤り率を 5%としたものであるが, 同様な結果となっている.

図 21 は, MODEL-E において $\alpha = 0.01$ に設定した場合の結果である. 提案 1, 提案 2, 提案 3, 従来 1 のワード線誤り率が悪化している. これは 2 レベルより大きい誤りが発生し得るモデルであるが, これらの手法はそのような誤りに対応できないためである. 一方, 混合 LM 誤り訂正符号による対策がされている提案 4 と従来 4, および LM 誤り訂正符号でない従来 2 は, MODEL-L の場合とほぼ変わらない誤り率を達成している. 特に提案 4 と従来 2 は, ともにこのモデルにおいて最良の誤り率を達成している. 図 22 は, $\alpha = 0.05$ に設定した場合の結果である. 同様な結果であるが, LM 誤り訂正符号の誤り率の悪化がより著しいものとなっている.

先に述べたように, 部分的連接符号は訂正機能の設定が重要となるが, 上記実験は訂正機能を 4 と小さく設定しているため, 柔軟な設定ができない. 実際の NAND フラッシュメモリにおいては, 要求される厳しい誤り率を達成するために, 訂正機能は例えれば 24 以上とかなり大きく設定される [10, 14]. このような場合の比較を行ったものが, 図 23 である. 各符号は以下のように, 先のシミュレーションより訂正機能を大きく設定している.

提案 5 提案符号の構成による LM 誤り訂正符号. 写像 f として提案写像を用いたも

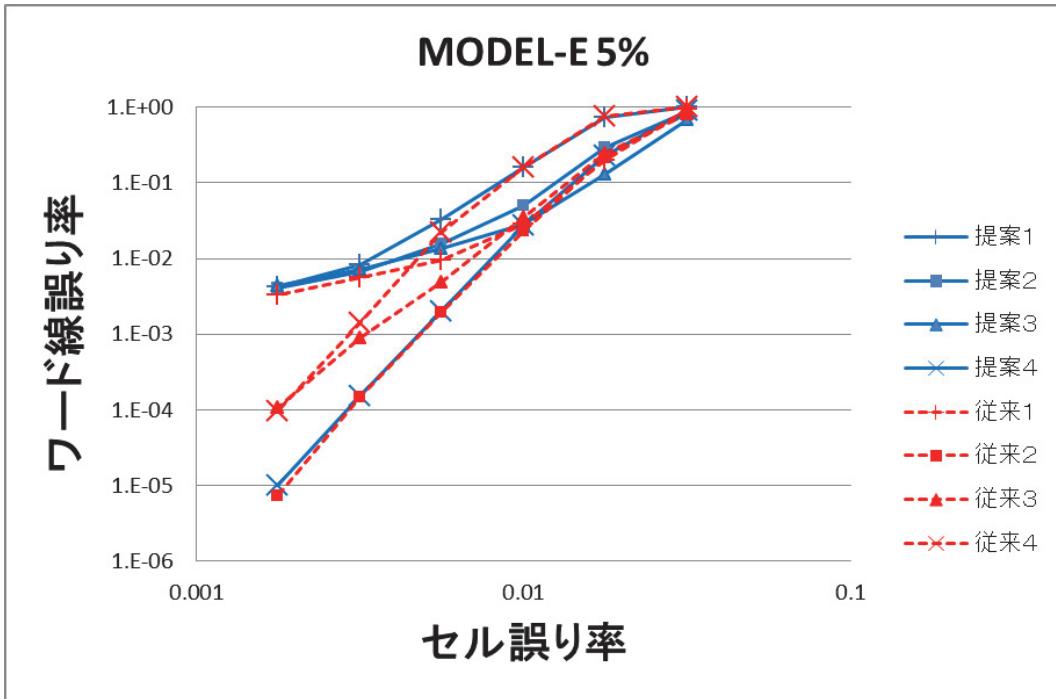


図 22: ワード線誤り率の比較 (MODEL-E, $\alpha = 0.05$).

の。ひとつのページを 8 つのセクタに分割し、それぞれにビット誤り訂正符号として (592, 512, 8) 符号を用いた。部分的連接符号として、下位 2 ページの各セクタを 16 に分割して、それぞれの部分に対して (622, 582, 3) 符号を用いた。混合 LM 誤り訂正符号として、(592, 572, 2) 符号を上位のページに用いた。

従来 5 ページ毎のビット誤り訂正符号。ひとつのページを 8 つのセクタに分割し、それぞれに (592, 512, 8) 符号を用いた。

従来 6 不均一誤り訂正符号 [73]。下位 2 のページを 8 つのセクタに分割し、それぞれに (592, 512, 8) 符号を用いた。部分的連接符号としては (602, 592, 1) 符号を用いて、上位ページの符号としては (592, 532, 6) 符号を用いた。

なお、訂正の手順は提案 4、従来 3 と同様である。提案符号では LM 誤り訂正によって、上位ページの誤りはほとんど訂正されてしまうため、上位ページへの誤り訂正是、かなり小さいもので十分である。これにより、検査ビットの多くを下位 2 ページの訂正に利用できる。一方、従来手法では上位ページの訂正のために、ある程度大きな機能が必要なため、下位 2 ページの訂正に利用できる検査ビットが少ない。このため、提案 5 が最も優れた誤り率となっている。以上により、MODEL-E のように完全には LM 誤りでない場合も、提案符号への切り替えにより、寿命の延長ができることが示された。

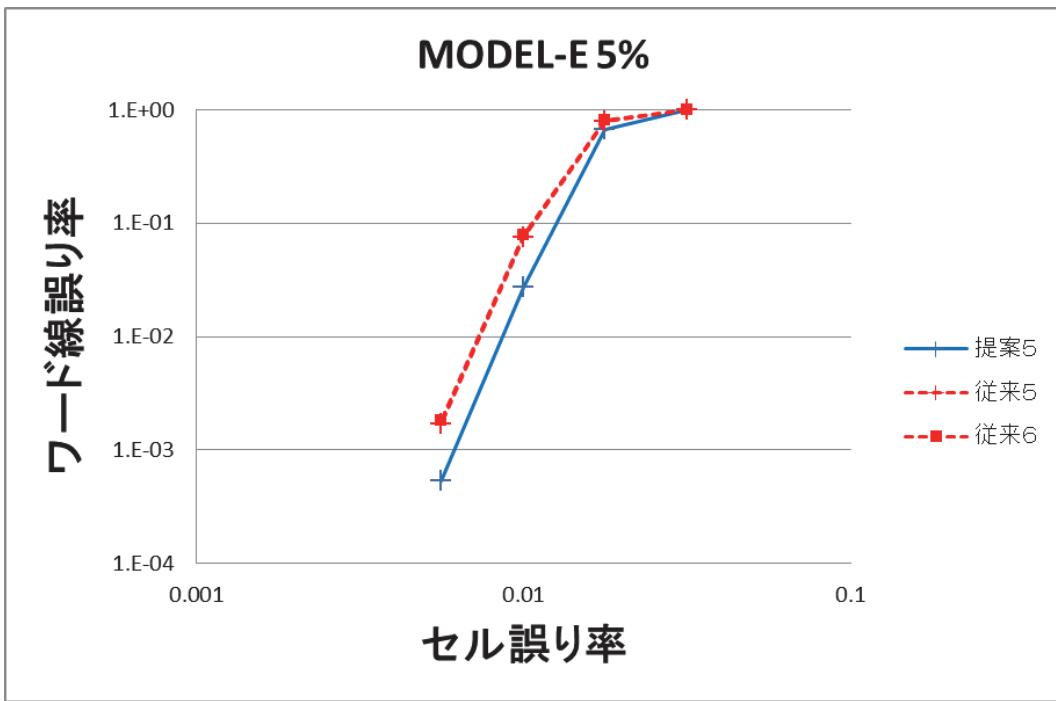


図 23: ワード線誤り率の比較 (MODEL-E, $\alpha = 0.05$).

7.7 この章のまとめ

従来の双方向 LM 誤り訂正符号の構成に関する定理に証明を与え, それに基づき t 双方向 (l_u, l_d) -LM 誤り訂正符号を, ページ毎に適用した 2 元の符号によって構成する方法を提案した. また, 一部ページの符号化が不要であることから, 検査ビットを削減でき, それを利用して追加の誤り訂正を行い, 誤り率の改善を行う手法を提案した. 追加の訂正を行えば, 従来のページ毎の誤り訂正符号より誤り率のパフォーマンスに優れることを示し, 信頼性が問題となる高い P/E サイクル数において, 符号を切り替えて利用する方法を提案した. これにより, 切り替える前にはマルチページプログラミングが可能となり, 切り替えにより寿命を延長できる. また, 構成法が従来のページ毎の誤り訂正符号と部分的に一致していることから, 特に構成に BCH 符号を用いた場合に, 追加のハードウェアを多く必要としない. ただし, 書き込みレイテンシは 167% 増大し, 読み出しレイテンシは 154~169% 増加する. また, 読み出しスループットは 6.3% 低下する.

8 おわりに

NAND フラッシュメモリの物理的な要因から発生する誤りへの対策として、誤り訂正符号が欠かせない。特に MLC のメモリにおいて、近年の製造プロセスでは強力な誤り訂正が必須となるが、検査ビットの量や、復号のための計算量が大きくなってしまうことが問題となっていた。本論文ではこの課題に対処すべく、NAND フラッシュメモリの誤りが近傍の閾値電圧レベルへ向かうものほど発生しやすく、さらに多レベルの誤りの発生し得るというモデルに基づいた、3つの手法を提案した。単一 LM 誤り訂正符号は従来より多くのパラメータについて、優れた符号長で構成できるようになり、多重 LM 誤り訂正符号では、従来不可能であったマルチページプログラミングとの併用を考慮した。各手法の特徴を表 18 にまとめる。検査シンボル長の削減を行いたい場合には提案手法 1 が有効である。ただし、マルチページプログラミングと併用できないため、書き込み速度は低下する。多元の单一対称シンボル誤り訂正符号の代替として、他の誤り訂正符号との連接符号により訂正機能を高める利用方法が現実的である。書き込み速度を落とさず、誤り率を改善したい場合には提案手法 2 を利用することとなる。多重誤りの訂正が可能であり、連接符号を使わなくても必要な誤り率を達成できる。符号構成は従来と変わらないため、いかなるレベルの誤りでも訂正できることを特徴とする。提案手法 3 は、マルチページプログラミングと併用できないが、大きく誤り率を改善できる。この符号も多重誤り訂正が可能であり、追加の誤り訂正符号の構成次第では、いかなるレベルの誤りでも訂正できる。提案手法 2 から提案手法 3 への切り替えも可能と思われるが、詳細な議論は今後の課題とする。

本論文では誤りの特徴を反映するというアプローチで符号の改善を行ったが、NAND フラッシュメモリにおける誤り訂正には、さまざまな課題が残されている。まず、本論文の議論の中心であった LM 誤り訂正符号に関する課題として、各レベルの誤りが、どの程度の誤り率をもって発生するかの見積もりが挙げられる。符号を実装するには必須であるにも関わらず、2 レベル以上の誤りの発生率は、現状ではわずかな実験的根拠を頼らざるを得ない。このような誤りの発生を説明するためのさらなる実験や、物理的モデルを用いたより詳細な議論が必要である。また、整数剰余環

表 18: 提案手法の特徴

	提案手法 1	提案手法 2	提案手法 3
手法の分類	符号構成	復号	符号構成
マルチページプログラミング	不可	可	不可
訂正可能な誤り（個数）	単一誤り	多重誤り	多重誤り
訂正可能な誤り（変量）	LM 誤り	全変量の誤り	全変量の誤り
検査長の削減	可	不可	不可
誤り率	従来と同様	改善率小	改善率大

を用いた構成法の多重誤り訂正への拡張も課題のひとつである。従来用いられているガロア体上の符号を利用した構成は、整数剰余環を用いた場合と比較して、パラメータによっては検査部が有効に利用できなくなってしまうためである。また、「誤りの特徴」という観点で言えば、近年話題となっているメモリの3次元積層技術の発展に期待をしている。例えば、3次元積層をしたメモリは最先端のプロセスルールで必ずしも製造されておらず、これにより信頼性の観点からも2次元のメモリとは異なるためである。

その他の符号では、LDPC符号の計算量の削減が重要な課題である。LDPC符号はシャノン限界に近い優れた誤り訂正が可能であり、BCH符号に代わる選択肢となっている。しかしながら、高いパフォーマンスは軟判定の復号法によって実現されており、詳細な閾値電圧値を得るために多段のセンシングは、読み出し時間を低下させる。加えて、LDPC符号は通常2元の符号であり、本論文で議論したような誤りの特徴は、軟判定を用いても完全には利用できない。したがって、本論文での提案とLDPC符号のセンシング回数削減手法を併用できれば、誤り率の改善や、計算量の削減が可能と思われる。ただし、軟判定の復号はページ間の情報を利用するため、マルチページプログラミングとの併用がそのまま可能については、議論が必要である。

最後に、用途に合わせた誤り訂正符号も課題のひとつである。NANDフラッシュメモリは、例えば書き込みが頻繁に行われる、読み出しが頻繁に行われるといったように、さまざまな利用形態があり、それによって誤りの傾向も変化する。加えて、将来的には現在のようなストレージとしての利用が続くとは限らない。例えば、PCMやReRAMといった、より高速なメモリとの併用も近年提案されている。今後のコンピュータシステムにおけるNANDフラッシュメモリの利用形態を考慮し、その要求を反映した誤り訂正符号の提案が必要と言える。

謝辞

私が符号理論に関心を持ち、このように博士論文として研究成果をまとめることができたのも、大学や大学院にて研究を進めるに当たって、直接ご指導して下さった北神正人准教授のおかげに他なりません。深く感謝致します。並びに、研究に携わる上で心構えなど、さまざまご指導やご助言を頂きました難波一輝准教授に、深く感謝いたします。研究生活においては困難に遭遇することも多くありましたが、挫折することなく取り組むことができたのは、研究室の皆様の、研究に臨む姿勢に刺激されたためだと感じております。深く御礼申し上げます。

参考文献

- [1] C. Yang, D. Muckatira, A. Kulkarni, and C. Chakrabarti, “Data storage time sensitive ECC schemes for MLC NAND Flash memories,” Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pp.2513–2517, May 2013.
- [2] Y. Maeda and H. Kaneko, “Error control coding for multilevel cell flash memories using nonbinary low-density parity-check codes,” Defect and Fault Tolerance in VLSI Systems, 2009. DFT ’09. 24th IEEE International Symposium on, pp.367–375, Oct 2009.
- [3] G. Dong, N. Xie, and T. Zhang, “On the use of soft-decision error-correction codes in NAND Flash memory,” Circuits and Systems I: Regular Papers, IEEE Transactions on, vol.58, no.2, pp.429–439, Feb 2011.
- [4] J. Kim, D. hwan Lee, and W. Sung, “Performance of rate 0.96 (68254, 65536) EG-LDPC code for NAND Flash memory error correction,” Communications (ICC), 2012 IEEE International Conference on, pp.7029–7033, June 2012.
- [5] C. Yang, Y. Emre, and C. Chakrabarti, “Product code schemes for error correction in MLC NAND flash memories,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.20, no.12, pp.2302–2314, Dec 2012.
- [6] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, “Introduction to flash memory,” Proceedings of the IEEE, vol.91, no.4, pp.489–502, 2003.
- [7] Y. Cai, Y. Luo, E. Haratsch, K. Mai, and O. Mutlu, “Data retention in MLC NAND flash memory: Characterization, optimization, and recovery,” High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on, pp.551–563, Feb 2015.
- [8] Y. Cai, E. Haratsch, O. Mutlu, and K. Mai, “Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis,” Design, Automation Test in Europe Conference Exhibition (DATE), pp.521–526, 2012.
- [9] D.C. Huang and H.M. Lin, “A method of correcting the shift error of multilevel flash memory by the skill of gray code,” Instrumentation and Measurement Technology Conference (I2MTC), 2011 IEEE, pp.1–6, May 2011.
- [10] S. gun Cho and J. Ha, “Concatenated BCH codes for NAND flash memories,” Communications (ICC), 2012 IEEE International Conference on, pp.2611–2616, June 2012.

- [11] S. Tanakamaru, Y. Yanagihara, and K. Takeuchi, “Error-prediction LDPC and error-recovery schemes for highly reliable solid-state drives (ssds),” Solid-State Circuits, IEEE Journal of, vol.48, no.11, pp.2920–2933, Nov 2013.
- [12] S. Li and T. Zhang, “Improving multi-level NAND Flash memory storage reliability using concatenated BCH-TCM coding,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.18, no.10, pp.1412–1420, Oct 2010.
- [13] H. Choi, W. Liu, and W. Sung, “VLSI implementation of BCH error correction for multilevel cell NAND Flash memory,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.18, no.5, pp.843–847, May 2010.
- [14] H. Yoo, Y. Lee, and I.C. Park, “Area-efficient syndrome calculation for strong BCH decoding,” Electronics Letters, vol.47, no.2, pp.107–108, January 2011.
- [15] C. Yang, Y. Emre, and C. Chakrabarti, “Product code schemes for error correction in mlc nand flash memories,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.20, no.12, pp.2302–2314, Dec 2012.
- [16] C. Sun, A. Soga, T. Onagi, K. Johguchi, and K. Takeuchi, “A workload-aware-design of 3d-nand flash memory for enterprise ssds,” Quality Electronic Design (ISQED), 2014 15th International Symposium on, pp.554–561, March 2014.
- [17] P.Y. Du, H.T. Lue, Y.H. Shih, K.Y. Hsieh, and C.Y. Lu, “Overview of 3d nand flash and progress of split-page 3d vertical gate (3dvg) nand architecture,” Solid-State and Integrated Circuit Technology (ICSICT), 2014 12th IEEE International Conference on, pp.1–4, Oct 2014.
- [18] B. Chen, X. Zhang, and Z. Wang, “Error correction for multi-level NAND flash memory using reed-solomon codes,” Signal Processing Systems, 2008. SiPS 2008. IEEE Workshop on, pp.94–99, Oct 2008.
- [19] W. Liu, J. Rho, and W. Sung, “Low-power high-throughput BCH error correction VLSI design for multi-level cell NAND flash memories,” Signal Processing Systems Design and Implementation, 2006. SIPS ’06. IEEE Workshop on, pp.303–308, Oct 2006.
- [20] J. Gray and C.V. Ingen, “Empirical measurements of disk failure rates and error rates,” Tech. Rep. MSR-TR-2005-166, Microsoft Research, December 2005.
- [21] S. Ji and D. Shin, “An efficient garbage collection for flash memory-based virtual memory systems,” Consumer Electronics, IEEE Transactions on, vol.56, no.4, pp.2355–2363, November 2010.

- [22] E. Fujiwara, *Code Design for Dependable Systems: Theory and Practical Application*, Wiley-Interscience, 2006.
- [23] T. Kløve, B. Bose, and N. Elarief, “Systematic, single limited magnitude error correcting codes for flash memories,” *Information Theory, IEEE Transactions on*, vol.57, no.7, pp.4477–4487, July 2011.
- [24] T. Kløve, J. Luo, I. Naydenova, and S. Yari, “Some codes correcting asymmetric errors of limited magnitude,” *Information Theory, IEEE Transactions on*, vol.57, no.11, pp.7459–7472, Nov 2011.
- [25] T. Kløve, J. Luo, and S. Yari, “Some codes correcting single symmetric errors of limited magnitude,” *WCC 2011 - Workshop on coding and cryptography*, Paris, France, pp.331–340, April 2011.
- [26] T. Kløve, J. Luo, and S. Yari, “Codes correcting single errors of limited magnitude,” *Information Theory, IEEE Transactions on*, vol.58, no.4, pp.2206–2219, April 2012.
- [27] M. Schwartz, “Quasi-cross lattice tilings with applications to flash memory,” *Information Theory, IEEE Transactions on*, vol.58, no.4, pp.2397–2405, April 2012.
- [28] S. Yari, T. Kløve, and B. Bose, “Some codes correcting unbalanced errors of limited magnitude for flash memories,” *Information Theory, IEEE Transactions on*, vol.59, no.11, pp.7278–7287, Nov 2013.
- [29] Y. Cassuto, M. Schwartz, V. Bohossian, and J. Bruck, “Codes for asymmetric limited-magnitude errors with application to multilevel flash memories,” *Information Theory, IEEE Transactions on*, vol.56, no.4, pp.1582–1595, April 2010.
- [30] S. Kotaki and M. Kitakami, “A class of q-ary unidirectional error correcting codes for mlc flash memories,” *Dependable Computing (PRDC), 2013 IEEE 19th Pacific Rim International Symposium on*, pp.132–133, Dec 2013.
- [31] M. Jeon and J. Lee, “On codes correcting bidirectional limited-magnitude errors for flash memories,” *Information Theory and its Applications (ISITA), 2012 International Symposium on*, pp.96–100, Oct 2012.
- [32] R. Ahlswede, H.K. Aydinian, L.H. Khachatrian, and L.M.G.M. Tolhuizen, “On q-ary codes correcting all unidirectional errors of a limited magnitude,” *CoRR*, vol.abs/cs/0607132, 2006.

- [33] N. Elarief and B. Bose, "Optimal, systematic, q -ary codes correcting all asymmetric and symmetric errors of limited magnitude," *Information Theory, IEEE Transactions on*, vol.56, no.3, pp.979–983, March 2010.
- [34] S. Kotaki and M. Kitakami, "Neighborhood level error control codes for multi-level cell flash memories," *IEICE Transactions on Information and Systems*, vol.96, no.9, pp.1926–1932, 2013.
- [35] K. Takeuchi, T. Tanaka, and T. Tanzawa, "A multipage cell architecture for high-speed programming multilevel NAND flash memories," *Solid-State Circuits, IEEE Journal of*, vol.33, no.8, pp.1228–1238, 1998.
- [36] J. Brewer and M. Gill, *Nonvolatile Memory Technologies with Emphasis on Flash:A Comprehensive Guide to Understanding and Using Flash Memory Devices*, Wiley-IEEE Press, 2008.
- [37] F. Sun, S. Devarajan, K. Rose, and T. Zhang, "Design of on-chip error correction systems for multilevel NOR and NAND flash memories," *Circuits, Devices Systems, IET*, vol.1, no.3, pp.241–249, June 2007.
- [38] C. Lee, S.H. Baek, and K.H. Park, "A hybrid flash file system based on nor and nand flash memories for embedded devices," *Computers, IEEE Transactions on*, vol.57, no.7, pp.1002–1008, July 2008.
- [39] E. Harari, "Flash memory - the great disruptor!," *Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2012 IEEE International, pp.10–15, Feb 2012.
- [40] K. Kanda, N. Shibata, T. Hisada, K. Isobe, M. Sato, Y. Shimizu, T. Shimizu, T. Sugimoto, T. Kobayashi, N. Kanagawa, Y. Kajitani, T. Ogawa, K. Iwasa, M. Kojima, T. Suzuki, Y. Suzuki, S. Sakai, T. Fujimura, Y. Utsunomiya, T. Hashimoto, N. Kobayashi, Y. Matsumoto, S. Inoue, Y. Suzuki, Y. Honda, Y. Kato, S. Zaitsu, H. Chibvongodze, M. Watanabe, H. Ding, N. Ookuma, and R. Yamashita, "A 19 nm 112.8 mm² 64 gb multi-level flash memory with 400 mbit/sec/pin 1.8 v toggle mode interface," *Solid-State Circuits, IEEE Journal of*, vol.48, no.1, pp.159–167, Jan 2013.
- [41] J. Vetter and S. Mittal, "Opportunities for nonvolatile memory systems in extreme-scale high-performance computing," *Computing in Science Engineering*, vol.17, no.2, pp.73–82, Mar 2015.
- [42] G. Dong, S. Li, and T. Zhang, "Using data postcompensation and predistortion to tolerate cell-to-cell interference in mlc nand flash memory," *Circuits and*

Systems I: Regular Papers, IEEE Transactions on, vol.57, no.10, pp.2718–2728, Oct 2010.

- [43] K.D. Suh, B.H. Suh, Y.H. Lim, J.K. Kim, Y.J. Choi, Y.N. Koh, S.S. Lee, S.C. Kwon, B.S. Choi, J.S. Yum, J.H. Choi, J.R. Kim, and H.K. Lim, “A 3.3 v 32 mb nand flash memory with incremental step pulse programming scheme,” Solid-State Circuits, IEEE Journal of, vol.30, no.11, pp.1149–1156, Nov 1995.
- [44] G. Dong, Y. Pan, N. Xie, C. Varanasi, and T. Zhang, “Estimating information-theoretical NAND flash memory storage capacity and its implication to memory system design space exploration,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.20, no.9, pp.1705–1714, 2012.
- [45] C. Calligaro, A. Manstretta, A. Modelli, and G. Torelli, “Technological and design constraints for multilevel flash memories,” Electronics, Circuits, and Systems, 1996. ICECS ’96., Proceedings of the Third IEEE International Conference on, pp.1005–1008 vol.2, Oct 1996.
- [46] J.D. Lee, S.H. Hur, and J.D. Choi, “Effects of floating-gate interference on nand flash memory cell operation,” Electron Device Letters, IEEE, vol.23, no.5, pp.264–266, May 2002.
- [47] R. Degraeve, F. Schuler, B. Kaczer, M. Lorenzini, D. Wellekens, P. Hendrickx, M. van Duuren, G. Dormans, J. Van Houdt, L. Haspeslagh, G. Groeseneken, and G. Tempel, “Analytical percolation model for predicting anomalous charge loss in flash memories,” Electron Devices, IEEE Transactions on, vol.51, no.9, pp.1392–1400, Sept 2004.
- [48] N. Mielke, H. Belgöl, I. Kalastirsky, P. Kalavade, A. Kurtz, Q. Meng, N. Righos, and J. Wu, “Flash eeprom threshold instabilities due to charge trapping during program/erase cycling,” Device and Materials Reliability, IEEE Transactions on, vol.4, no.3, pp.335–344, Sept 2004.
- [49] Y. Cai, Y. Luo, S. Ghose, and O. Mutlu, “Read disturb errors in mlc nand flash memory: Characterization, mitigation, and recovery,” Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on, pp.438–449, June 2015.
- [50] C. Monzio Compagnoni, M. Ghidotti, A. Lacaita, A. Spinelli, and A. Visconti, “Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories,” Electron Device Letters, IEEE, vol.30, no.9, pp.984–986, Sept 2009.

- [51] M.J. Chen, K.C. Tu, H.H. Wang, C.L. Chen, S.Y. Lai, and Y.S. Liu, “A statistical model for the headed and tail distributions of random telegraph signal magnitudes in nanoscale mosfets,” IEEE Transactions on Electron Devices, vol.61, no.7, pp.2495–2502, July 2014.
- [52] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, “On-chip error correcting techniques for new-generation flash memories,” Proceedings of the IEEE, vol.91, no.4, pp.602–616, April 2003.
- [53] R. Micheloni, M. Picca, S. Amato, H. Schwalm, M. Scheppler, and S. Commodo, “Non-volatile memories for removable media,” Proceedings of the IEEE, vol.97, no.1, pp.148–160, Jan 2009.
- [54] C. Wang and W.F. Wong, “Extending the lifetime of nand flash memory by salvaging bad blocks,” Design, Automation Test in Europe Conference Exhibition (DATE), 2012, pp.260–263, March 2012.
- [55] M.C. Yang, Y.M. Chang, C.W. Tsao, P.C. Huang, Y.H. Chang, and T.W. Kuo, “Garbage collection and wear leveling for flash memory: Past and future,” Smart Computing (SMARTCOMP), 2014 International Conference on, pp.66–73, Nov 2014.
- [56] S. Lin and D.J. Costello, Error Control Coding, Second Edition, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2004.
- [57] 今井 秀樹, 符号理論, 電子情報通信学会, 1990.
- [58] 新妻 弘, 木村 哲三, 群・環・体入門, 共立出版, 1999.
- [59] H. Kaneko and E. Fujiwara, “A class of m -ary asymmetric symbol error correcting codes for data entry devices,” Computers, IEEE Transactions on, vol.53, no.2, pp.159–167, Feb 2004.
- [60] T. Tanzawa, T. Tanaka, K. Takeuchi, R. Shirota, S. Aritome, H. Watanabe, G. Hemink, K. Shimizu, S. Sato, Y. Takeuchi, and K. Ohuchi, “A compact on-chip ecc for low cost flash memories,” IEEE Journal of Solid-State Circuits, vol.32, no.5, pp.662–669, May 1997.
- [61] B. Ricco, G. Torelli, M. Lanzoni, A. Manstretta, H.E. Maes, D. Montanari, and A. Modelli, “Nonvolatile multilevel memories for digital applications,” Proceedings of the IEEE, vol.86, no.12, pp.2399–2423, Dec 1998.
- [62] T.H. Chen, Y.Y. Hsiao, Y.T. Hsing, and C.W. Wu, “An adaptive-rate error correction scheme for nand flash memory,” VLSI Test Symposium, 2009. VTS ’09. 27th IEEE, pp.53–58, May 2009.

- [63] L. Yuan, H. Liu, P. Jia, and Y. Yang, “An adaptive ecc scheme for dynamic protection of nand flash memories,” Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on, pp.1052–1055, April 2015.
- [64] L. Yuan, H. Liu, P. Jia, and Y. Yang, “Reliability-based ecc system for adaptive protection of nand flash memories,” Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on, pp.897–902, April 2015.
- [65] R. Gallager, “Low-density parity-check codes,” Information Theory, IRE Transactions on, vol.8, no.1, pp.21–28, January 1962.
- [66] Y. Pan, G. Dong, N. Xie, and T. Zhang, “Using quasi-ez-nand flash memory to build large-capacity solid-state drives in computing systems,” Computers, IEEE Transactions on, vol.62, no.5, pp.1051–1057, May 2013.
- [67] 中村 勝洋, 整数剰余環上の誤り訂正符号とそのデジタル通信系への応用に関する研究, 東京大学工学系研究科計数工学専攻博士論文, 2000.
- [68] H.M.N. Kostadinov, H. Morita, “On (± 1) error correctable integer codes,” Fundamentals of Electronics, Communications and Computer Science, IEICE Transactions on, vol.E93-A, no.12, pp.2758–2461, December 2010.
- [69] 佐藤 和也, 北神 正人, “多レベルセルフラッシュメモリのための1レベル誤り訂正符号(安全性及び一般),” 電子情報通信学会技術研究報告. DC, ディペンダブルコンピューティング, vol.110, no.333, pp.9–14, dec 2010.
- [70] F. Gray, “Pulse code communication,” March 17 1953. US Patent 2,632,058.
- [71] Y. Manzor and O. Keren, “Amalgamated q-ary codes for multi-level flash memories,” 2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp.98–103, Oct 2012.
- [72] T. Tokutomi, S. Tanakamaru, T.O. Iwasaki, and K. Takeuchi, “Advanced error-prediction LDPC with temperature compensation for highly reliable SSDs,” Solid-State Electronics, vol.111, pp.129 – 140, 2015.
- [73] J. Li, K. Zhao, J. Ma, and T. Zhang, “Realizing unequal error correction for nand flash memory at minimal read latency overhead,” Circuits and Systems II: Express Briefs, IEEE Transactions on, vol.61, no.5, pp.354–358, May 2014.