

宇宙用誤り制御符号と画像圧縮方式の研究

Research on Error Control Coding and Image Compression for Spacecraft Data-Handling Systems

総合技術研究本部 エレクトロニクス技術グループ 金子晴彦
Haruhiko Kaneko, Spacecraft Electrical Engineering Group
Institute of Space Technology and Aeronautics

Abstract

Source coding and channel coding are essential techniques for efficient data processing in spacecraft. Although many of conventional coding techniques can be effectively applied to spacecraft data-handling systems, we need to develop some new source coding and channel coding techniques to design high-performance data-handling systems for spacecraft. Therefore, in our research, we have developed new coding methods suitable for spacecraft, that is, spotty byte error control codes for semiconductor memory systems, such as Solid-State Recorders (SSRs), and low-complexity video encoding method based on low-density parity check code for efficient compression of video data.

1 はじめに

本研究は宇宙機搭載データ処理システムに適した通信路符号化方式及び情報源符号化方式を構築するためのものであり、具体的には以下に示す2種の符号化方式について研究開発を行った。

(a) 宇宙機搭載半導体メモリ装置用誤り制御符号の研究：宇宙機搭載の半導体メモリ装置においては、陽子等の衝突によりSEU(Single Event Upset)が多く発生することから、これによるデータの誤りを効率的に訂正／検出するための誤り制御符号の開発と符号化／復号回路の実装を行った。本研究は、平成10年度より東京工業大学・大学院情報理工学研究科・藤原英二教授との共同研究により実施したものである。

(b) 低密度パリティ検査符号を用いた低計算量動画像圧縮方式の研究：既存のMPEG等の動画像圧縮方式は圧縮処理において動き予測を行うため、圧縮のために多くの計算処理を必要とする。宇宙機における圧縮処理のための計算量を大幅に削減するため、低密度パリティ検査(LDPC)符号を用いた動画像圧縮方式の基礎研究とシミュレーションを行った。本研究は、平成17年度より東京工業大学・大学院情報理工学研究科・藤原英二教授との共同研究により実施しているものである。

2 研究の概要

宇宙機搭載半導体メモリ装置用誤り制御符号の研究として、以下の項目を実施した。

- (a-1) 宇宙機において要求される誤り制御符号の機能を決定するため、MDS-1搭載SSRにおける誤り率及び誤りパターンの解析を行った。
- (a-2) 上記解析結果を基にスポッティバイト誤り制御符号の機能と符号パラメータを決定した。
- (a-3) スポッティバイト誤り制御符号及びReed-Solomon(RS)符号の符号化回路及び並列復号回

路の試作を行い、回路量、遅延、等の評価を行った。

低密度パリティ検査符号を用いた低計算量動画像圧縮方式の研究として、以下の項目を実施した。

- (b-1) 本圧縮法の圧縮アルゴリズム及び伸長アルゴリズムを構築した。
- (b-2) 上記圧縮アルゴリズム及び伸長アルゴリズムをソフトウェアにより実装し、画質(PSNR: Peak Signal-to-Noise Ratio)の評価を行った。

3 成果の概要

3.1 宇宙機搭載半導体メモリ装置用誤り制御符号の研究 [1, 2]

コンピュータシステムに対し効率的に誤り制御符号を適用するためには、一般に以下の手順により符号の開発と実装を行う必要がある [3, 4]。

- (i) 誤り率、誤りパターン、誤りの種類の解析。
- (ii) 符号パラメータ、符号機能の決定。
- (iii) 符号化／復号回路の設計と実装。

本研究では、MDS-1搭載のSSR(MDS-1/SSR)から取得したデータを用いて誤り率、誤りパターン、等の解析を行い、これに基づいて宇宙機搭載半導体メモリ用誤り制御符号のパラメータ、符号機能を決定した。また、FPGAを用いて符号化回路及び並列復号回路の実装を行った。本研究成果の概略を以下に示す。

3.1.1 MDS-1/SSRにおける誤り率／誤りパターン解析

MDS-1/SSRの構成をFig. 1に示す。MDS-1/SSRの記憶部は2枚のメモリスライス(MEMS1及びMEMS2)より構成され、それぞれのメモリスライスは6個のメモリモジュール(A,B,C,D,E,F)を搭載

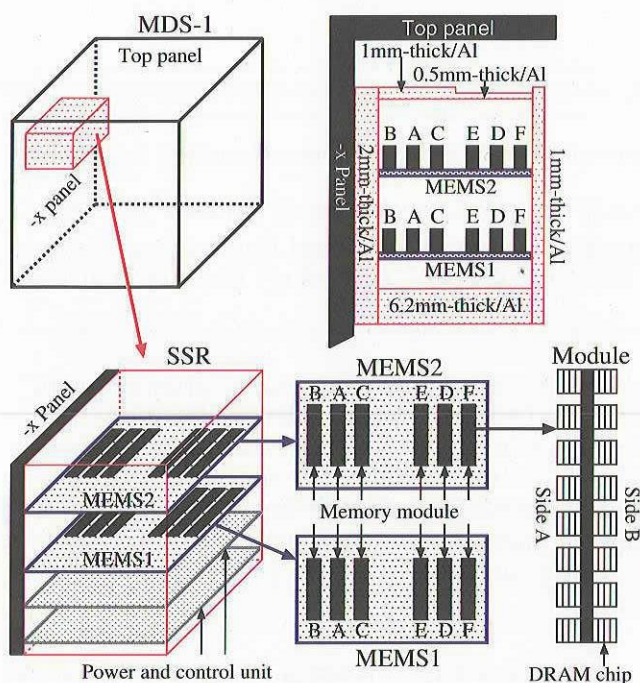


Figure 1: Configuration of the MDS-1/SSR.

する。また、各メモリモジュールは記憶容量 64Mb を有する DRAM 素子 64 個から構成される。

メモリモジュールの誤り率を Fig. 2 に示す。ただし、モジュール C 及び F の誤り率はシステムの設計上取得できていない。MEMS1 の上部は MEMS2 によって遮蔽されていることから、MEMS1 のモジュールの誤り率は MEMS2 と比較して低い。また、MEMS1 においては衛星中心側のモジュール E 及び D が低い誤り率を有するが、MEMS2 においてはすべてのモジュールが同程度の誤り率を有する。

MDS-1/SSR は 1 アドレスに 48 ビットのデータを保持し、従ってデータの入出力は長さ 48 ビットを有する I/O ワード単位で行われる。MDS-1/SSR はデータ入出力幅 4 ビットの DRAM 素子を用いており、I/O ワードと DRAM 素子は Fig. 3 に示すような関係で対応付けられる。I/O ワード中の各ビットの誤り率を Fig. 4 に示す。I/O ワード中の各ビットの誤り率に大きな偏りがある場合、不均一誤り制御 (UEC: Unequal Error Control) 符号を用いた方が効率的に符号化を行える場合がある。しかし、UEC 符号適用のためには誤り率の偏りをより高い精度で解析する必要があり、本解析のみから UEC 符号を適用することは不可能である。

単一の DRAM 素子から入出力される長さ $b = 4$ ビットのデータブロックをバイトと称する。バイト誤りパターンと誤り率を Table 1 に示す。バイト内 1 ビット誤りがほとんどであるが、バイト内 2 ビット誤り及びバイト内 3 ビット誤りも発生していることから、宇宙機搭載半導体メモリ装置に対してはこのような誤りを訂正/検出可能な符号の適用が必要である。以下では、長さ b ビットを有するバイト中の $t (\leq b)$ ビット誤りを t/b 誤りまたはスポッティバイト誤りと称する。MDS-1/SSR から取得したデータによると、単一の I/O ワード中に複数のスポッティバイト誤りが発生する場合がある。I/O ワード中の

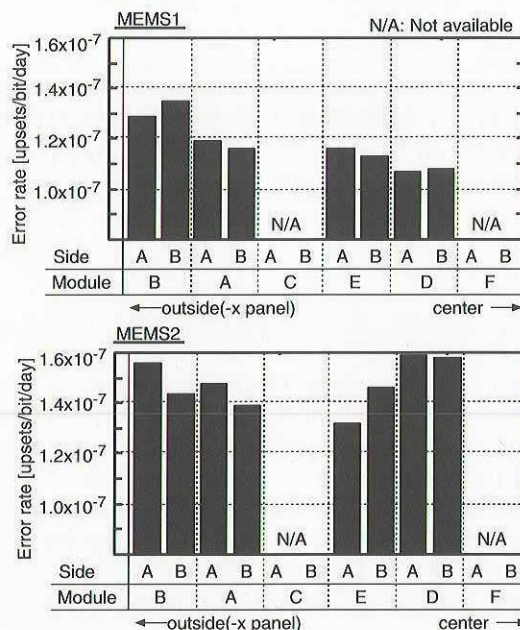


Figure 2: Error rates of memory modules.

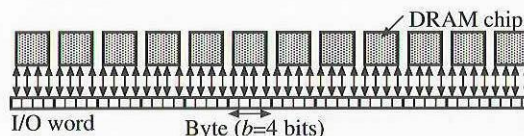


Figure 3: I/O word structure.

誤りパターンの例と誤り率を Table 2 に示す。

3.1.2 誤り制御符号の機能とパラメータの決定

上記のデータ解析より、宇宙機搭載半導体メモリ装置に対してはスポッティバイト誤りを訂正/検出可能な誤り制御符号を適用する必要がある。本研究では、この条件を満足する符号として、Table 3 に示す 3 種のスポッティバイト誤り制御符号、及び 3 種の RS 符号を選択した。Table 3 に示す 6 種の誤り制御符号は線形符号であり、検査行列 H の零空間として定義できる。ガロア体 $GF(2^{16})$ 上の短縮 RS 符号 (ii) の検査行列の例を以下に示す [3]。

$$H = \begin{bmatrix} I & I & I & I & \dots & I \\ I & T & T^2 & T^3 & \dots & T^{10} \\ I & T^2 & T^4 & T^6 & \dots & T^{20} \end{bmatrix}$$

ただし、 T は原始多項式 $p(x) = x^{16} + x^{12} + x^3 + x + 1$ により定義される 16×16 随伴行列である。

スポッティバイト誤り制御符号の構成法は文献 [5],[6] 及び [7] において詳細に述べられている。例として、 $(160,128) S_{4/16}EC-D_{4/16}ED$ 符号の検査行列を以下に示す。

$$H = \begin{bmatrix} I & I & I & I & \dots & I \\ \gamma^0 H' & \gamma^1 H' & \gamma^2 H' & \gamma^3 H' & \dots & \gamma^9 H' \\ \gamma^0 H' & \gamma^2 H' & \gamma^4 H' & \gamma^6 H' & \dots & \gamma^{18} H' \end{bmatrix}$$

ただし、 I は 16×16 単位行列、 $\gamma \in GF(2^8)$ は $GF(2)$ 上の 8 次の原始多項式 $p(x) = x^8 + x^7 + x^2 + 1$ の根

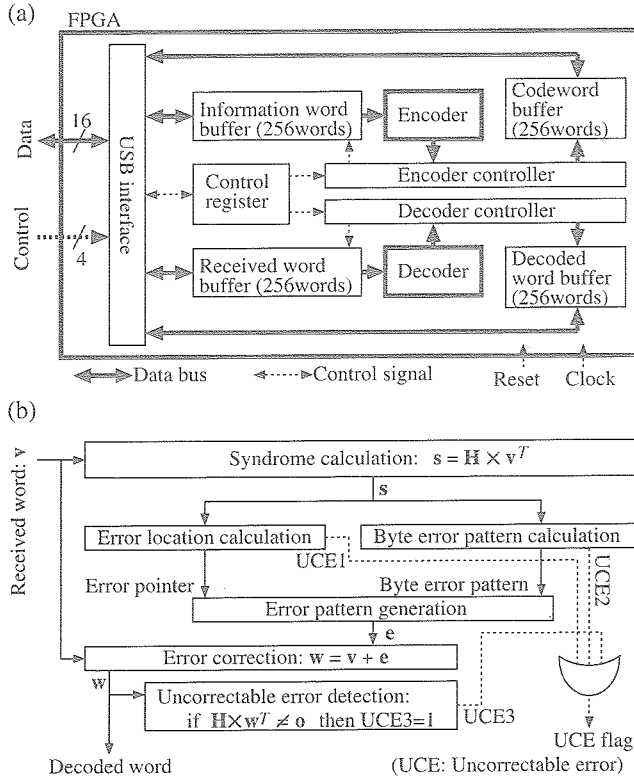


Figure 5: (a) Block diagram of the developed FPGA. (b) Flow of parallel decoding.

Table 5: Clock frequencies and delays.

Code	(i)	(ii)	(iii)	(iv)	(v)	(vi)
Delay [ns]	17.623	16.160	15.096	19.523	19.863	17.459
Frequency [MHz]	56.74	61.88	66.24	51.22	50.34	57.28

3.2.1 圧縮手順

本圧縮法において、動画画はIピクチャ及びLピクチャの2種のピクチャより構成され、IピクチャはLピクチャの間に一定間隔で挿入する。IピクチャはJPEGと同様の手順でフレーム内符号化を行う。Lピクチャの圧縮手順は以下のとおりである。

- (1) Lピクチャの大きさを $H\sigma \times W\sigma$ とし、これを $\sigma \times \sigma$ の大きさを有する $N = HW$ 個のマクロブロック L_0, \dots, L_{N-1} に分割する。マクロブロックベクトルを $\mathbf{L} = (L_0, \dots, L_{N-1})$ とおく。
- (2) 2元 $(N, N-M)$ LDPC 符号の検査行列を

$$\mathbf{H} = \begin{bmatrix} a_{0,0} & a_{0,1} & \dots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \dots & a_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M-1,0} & a_{M-1,1} & \dots & a_{M-1,N-1} \end{bmatrix}$$

とおく。行列 \mathbf{H} を用いて、長さ N を有するマクロブロックベクトル \mathbf{L} を、長さ $M (< N)$ を有するシンδροームブロックベクトル $\mathbf{S} = (S_0, \dots, S_{M-1})$ に変換する。ただし、

$$S_i = \sum_{j \in \{J | a_{i,j}=1\}} L_j$$

は $\sigma \times \sigma$ シンδροームブロックである。

- (3) シンδροームブロック \mathbf{S} に対し JPEG と同様の非可逆圧縮 (DCT, 量子化, ジグザグスキャン, ランレングス符号化, 可変長符号化) を適用し、圧縮データを得る。

上記の圧縮手順では動き予測を用いないことから、従来の動画画圧縮法と比較して非常に少ない計算量で実行できる。また、上に示すようにLピクチャの圧縮時にはフレーム間の相関を用いないが、伸長時間にフレーム間の相関を利用することから、Motion JPEG等の従来のフレーム内符号化方式よりも高い圧縮率を得ることができる。

3.2.2 伸長手順

IピクチャはJPEGと同様の手順で伸長する。Lピクチャの伸長手順は以下のとおりである。

- (1) 圧縮手順 (3) により圧縮したシンδροームブロックベクトル \mathbf{S} を JPEG と同様の手順で伸長し、 $\mathbf{S}' = (S'_0, \dots, S'_{M-1})$ を得る。ここで、JPEG は非可逆圧縮であるから $\mathbf{S} \simeq \mathbf{S}'$ である。
- (2) Lピクチャの前方及び後方に位置する伸長後のIピクチャ I_1 及び I_2 を用いて、動き補償フレーム補間 (MCFI) によりLピクチャの予測画像 L' を構成する。画像 L' のマクロブロックベクトルを $\mathbf{L}' = (L'_1, \dots, L'_{N-1})$ とする。また、動き予測と同時に L' の各ピクセルにおける予測値の「不確かさ」を求め、これよりエラーブロックベクトル $\mathbf{E} = (E_1, \dots, E_{N-1})$ を構成する。ただし、 E_j は L'_j に対応するエラーブロックである。
- (3) 予測画像 L' には予測誤りが存在することから、シンδροームブロックベクトル \mathbf{S}' を用いてLDPC符号の反復復号を行い、 L' に存在する予測誤りを訂正する。すなわち、すべての $i \in \{0, \dots, M-1\}$ に対し $S'_i = \sum_{j \in \{J | a_{i,j}=1\}} L'_j$ を満足するように、エラーブロックベクトル \mathbf{E} を用いて \mathbf{L}' を反復更新し、復号後のLピクチャ $\hat{\mathbf{L}}$ を構成する。以下に \mathbf{L}' の反復復号手順を示す。マクロブロック L'_j 、エラーブロック E_j 及びシンδροームブロック S'_i をそれぞれ以下のような $\sigma \times \sigma$ 行列で表現する。

$$L'_j = \begin{bmatrix} l_j^{0,0} & l_j^{0,1} & \dots & l_j^{0,\sigma-1} \\ \vdots & \vdots & \ddots & \vdots \\ l_j^{\sigma-1,0} & l_j^{\sigma-1,1} & \dots & l_j^{\sigma-1,\sigma-1} \end{bmatrix}$$

$$E_j = \begin{bmatrix} e_j^{0,0} & e_j^{0,1} & \dots & e_j^{0,\sigma-1} \\ \vdots & \vdots & \ddots & \vdots \\ e_j^{\sigma-1,0} & e_j^{\sigma-1,1} & \dots & e_j^{\sigma-1,\sigma-1} \end{bmatrix}$$

$$S'_i = \begin{bmatrix} s_i^{0,0} & s_i^{0,1} & \dots & s_i^{0,\sigma-1} \\ \vdots & \vdots & \ddots & \vdots \\ s_i^{\sigma-1,0} & s_i^{\sigma-1,1} & \dots & s_i^{\sigma-1,\sigma-1} \end{bmatrix}$$

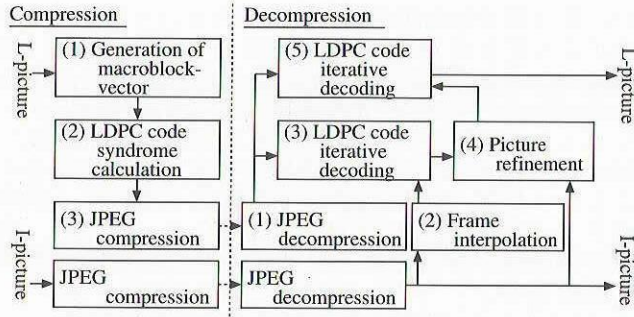


Figure 6: Compression / decompression process.

すべての $(p, q) \in \{(p, q) \mid 0 \leq p, q \leq \sigma - 1\}$ に対して、それぞれ独立に以下の処理を行う。

- (i) すべての $i \in \{0, 1, \dots, M - 1\}$ に対し、シンドロームブロックベクトル \mathbf{S}' とマクロブロックベクトル \mathbf{L}' の不一致度 $x_i^{p,q}$ を求める。

$$x_i^{p,q} = s_i^{p,q} - \left(\sum_{j \in \{j \mid a_{i,j}=1\}} l_j^{p,q} \right)$$

- (ii) エラーブロックベクトル \mathbf{E} を用いて、すべての $(i, j) \in \{(i, j) \mid a_{i,j} = 1\}$ に対し、重み付き不一致度 $y_{i,j}^{p,q}$ を求める。

$$y_{i,j}^{p,q} = x_i^{p,q} \times \frac{e_j^{p,q}}{\sum_{k \in \{k \mid a_{i,k}=1\}} e_k^{p,q}}$$

- (iii) すべての $j \in \{0, 1, \dots, N - 1\}$ に対し、次式により $l_j^{p,q}$ の値を更新する。

$$l_j^{p,q} := l_j^{p,q} + \frac{\sum_{i \in \{i \mid a_{i,j}=1\}} y_{i,j}^{p,q}}{w_j}$$

ただし、 w_j は \mathbf{H} の第 j 列の Hamming 重みである。

- (4) 上記の LDPC 符号の復号においては誤訂正が生じる場合があることから、 I_1, I_2 , 及び \hat{L} を用いて動き補償ノイズ除去をおこない \hat{L} の補正を行う。

- (5) 上記の画像補正において、誤った補正を行う場合があることから、手順 (3) と同様に LDPC 符号の反復復号を用いてこの誤りを訂正する。

手順 (4) 及び (5) を繰り返して行うことにより画質の向上が可能である。圧縮及び伸長手順を Fig. 6 に示す。テスト画像「mobile」[14] を提案手法により圧縮し、これを伸長するときの例を Fig. 7 に示す。

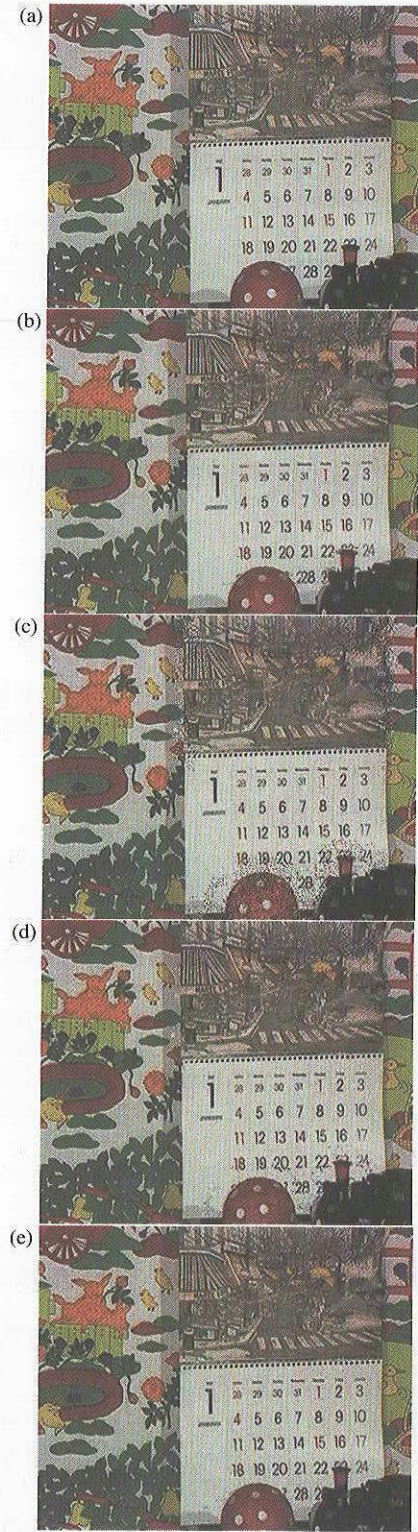


Figure 7: Example of decompression process: (a) Original image. (b) Estimated image by step (2). (c) Decoded image by step (3). (d) Refined image by step (4). (e) Final output of the decompression process with four iterations of steps (4) and (5).

