逐次リアルタイム復号処理が可能な2値画像圧縮法*

和崎 克己**　不破 泰***　江口 正義****　中村 八束*****

A Bi-level Image Compression Method for Realizing Real-time Sequential Decoding

Katsumi WASAKI, Yasushi FUWA, Masayoshi EGUCHI and Yatsuka NAKAMURA

In this paper, we propose a new compression method for bi-level images which can be applied to a real-time sequential decoding system. When we consider an application such as image comparison processing in which image data, compressed and stored beforehand, must be decoded and compared with image data from some input device (e.g., raster scan-type camera), the work of decoding data must be able to produce outputs one after another in sequence within a prescribed time frame. However, the existing methods of data compression for bi-level images do not satisfactorily fulfill the above listed requirements for real-time decoding processes. For this reason, we propose a compression code called the "Binary Place Compression (BPC) Code", which possesses the real-time nature required for such bi-level image processing as described above. This BPC code has the following features: (1) sequential real-time processing capability, (2) loss-less coding/decoding, (3) simplicity of coding/decoding procedure resulting in easy hardware implementation, etc. Also in this work, we prototyped a decoder for BPC with a gate-array and produced an image processing unit. We verified the applicability of our proposed code by evaluating the prototype in a practical application for constructing a quality evaluation system for printed circuit boards. Finally, we also verified that the compression rate of our code exceeds that of other binary compression codes used in the past.

キーワード：2値画像, 画像符号化, ロスレス, データ圧縮, リアルタイム処理, ゲートアレイ

1　まえがき

画像データはそのままディジタル化すると情報量が非常に多くなるため, 情報圧縮が必須であり, 現在までに様々な符号化方式が提案・実用化されている[1)~12)]. 従来より, 静止画像を対象とした符号化については圧縮率を優先させ, 情報の欠落を許す非可逆符号化方式の需要が高かった. 例えば, 国際標準符号化方式であるJPEG[3)]では適応型DCT(Discrete Cosine Transform)を用いた非可逆符号化が主に用いられてきた. しかし, 医療診断用のレントゲン画像, 人工衛星画像, 製品検査用の良品画像といった対象は情報の完全性が要求され, 情報の欠落なく通信・保存することを目的とした可逆符号化方式の需要は高い.

コンパクト性を持った可逆符号化方式としてHuffman 符号[4)]がある. Run-Length 符号の拡張として,Run 長を単純に2進符号化するのではなく, Run の大きさにより符号長を何段階かに分割する手法（bit 分割符号化方式）[5)]が提案されている. Huang[6)]はRun-Length 符号の様々な拡張と, 符号伝送時の回線誤り率の影響について検討した. Run-Length とHuffman 符号を組み合わせCCITT テストチャート向けに符号長を最適化したファクシミリ通信用の符号化方式としてMH 符号[7)]がある. Kim ら[8)]は2値画像中に存在する閉包画素群に着目し Run-Length 符号を用いて符号化する方式を提案した. JPEG においては, 可逆符号化方式としてDPCM(Differencial Pulse Code Modulation)[3)]を定義しているが圧縮率の点で問題を抱えている. これに対し柳谷ら[9)]は, 多値画像の可逆符号化の際, 適応予測を用いて圧縮率を改善した符号化方式を提案, JPEG-DPCM 方式より圧縮率に関して有利な結果を得ている. 2値画像の階層型な可逆符号化方式としては国際標準符号化方式であるJBIG[10)11)]がある. 勝野ら[12)]は, JBIG のプログレッシブ表示機能を利用したJBIG-CODEC と画像通信システムを試作し性能評価を行っている.

著者らは, あらかじめ圧縮して蓄積した画像データを展開しながら演算処理 (例えばエッジ検出, 他画像とのマッチング処理, 等) を逐次行い, 結果をメモリ上へ格納し, ディスプレイへ表示させるといった用途に適した画像符号化方式について検討を行ってきてい

---

* 1995年5月信学会情報理論研究会予稿に加筆
** 電子制御工学科 助手
*** 信州大学工学部情報工学科 助教授
**** 東京商船大学商船学部 助教授
***** 信州大学工学部情報工学科 教授
1995年度 文部省特定研究経費による
原稿受付　1997年10月31日

る．このような画像処理の用途は，工場ラインにおける製品検査を始めとした様々な需要がある．

2つの画像を比較することにより製品表面の検査を行うアプリケーションを考える．検査は，対象画像と前もって用意しておいた基準画像の各画素同士を比較することで行う．このとき，検査対象物が単一でなく，少量多品種な生産ラインでは，製品の種類毎に基準画像を持たなければならない．基準となる画像は，可逆符号化で事前に圧縮した上で，多くの種類を複数蓄えておく必要がある．次に，検査対象物をカメラ等を使用して取り込むと同時に，蓄積した基準画像を展開し，画素間の比較処理を行う．この時，カメラから取り込む前に，事前に対応する圧縮画像を展開してバッファに蓄えておくのでは，取り込み前に無駄な時間が必要で，一画面分のバッファを用意しなければならないという問題が生じる．カメラからの出力(走査)と同じ速度で逐次展開処理を行い，カメラからの各画素の出力に合わせて，対応する基準画像の画素が展開される手法があれば，この問題は生じない．以上のような用途では，圧縮より展開時に処理の高速性が要求されており，一定時間間隔毎に順番に展開されたデータがとぎれることなく出力される事(以後，このような復号を逐次リアルタイム復号と呼ぶ)が可能で，かつハードウェアで実現可能な程度に簡単である処理方式が必要である．

復号処理を高速に行う試みは，G4 ファクシミリ用高速復号 LSI の試作をはじめ，様々な研究がある[13)14)15)]．復号処理の高速化は，画像向け以外でも必要であり，様々な研究がある．例えばハードディスクの記録符号の復号処理において，高速転送の要求を満たす復号方式とそれを実現する LSI の開発に関する研究等がある[16)]．

本論文では，逐次リアルタイムに復号可能な2値画像の符号化方式として，2進桁圧縮符号(Binary Place Compression code：以後,BPC 符号)と，その復号ハードウェアの構成方法を提案する．本符号は，文献[5)]で提案された bit 分割符号と同様に，Run 長を一旦2進数へ変換し，各桁に対応した可変長符号を対応付ける．本方式は逐次リアルタイム復号が可能な他，圧縮率は MH,bit 分割符号化方式と同程度，復号処理が簡単でハードウェア実装が容易で規模も小さい，等の特徴を有している．

復号処理性能を向上するために，新しいパイプライン形の復号処理方式を提案する．復号処理をパイプライン化し，1クロック毎に1bit の速度で定常的に復号した画素を出力するために，圧縮データを数 bit 分先読みするバッファを設ける等，逐次リアルタイム復号に適したパイプライン構成法を示す．本符号を用いることで，上記の高速復号処理が可能なハードウェアは容易に実現可能であり，有効であることを示すのが本論文の主眼である．この高速復号ハードウェアを，実際にゲートアレイ上へ実装し試作した．復号ハードウェアを，逐次リアルタイムな復号処理が必要な画像処理装置に組み込んで評価試験を行ったところ，提案方式の有効性が確認された．

2 2進桁圧縮 (BPC) 方式

2-1 BPC 方式の基本システム

BPC 方式の基本システムを図1に示す．符号器では，まず Run 長を調べそれを2進展開形式で表現する．次に，展開された Run 長の各項(桁)に対応する可変長符号を符号テーブル(Code Table)より参照する．最後に白/黒のどちらが Run で連続しているかを示す 1bit の識別子 (0/1) と合わせて符号化する．復号器では，入力した圧縮データに対し，まず最小符号長である 3bit 分を先読みレジスタ (Look-ahead Register) に格納する．2進展開形式に展開された Run 長の各桁を復号するため，可変長符号の探索 (BPC-Tree Serching) を行う．当該 Run 長を算出し,Run の白/黒の識別子を調べ，Bit 出力カウンタ (Bit Output Counter) によって元の2値画像を再生する．

2-2 符号器

2-2-1 Run-Length 計算

入力画像の bit 系列を考える．連続している0または1の部分を Run とする．n 番目の Run を $Block_n$，Run の長さを $Length_n$ とする ($n = 1 \sim N$,Nは Run の個数)．$Length_n$は1から $2^K - 1$(Kは量子化のための最大2進桁数で Run 長の最大値を規定する)の範囲とする．この最大値を越える長さの Run は，一旦 $2^K - 1$ 個で打ち切り，改めて Run とする．$Block_n$の連続している bit の値が0または1のとき，$Value_n$=0 または1とする．$Block_n$は，$Value_n$と $Length_n$の組，$Block_n$=($Value_n$, $Length_n$) で表す．

2-2-2 2進桁対応とテーブルによる符号化

符号化は,$Block_1 \sim Block_N$の順にそれぞれ対応する圧縮 bit 列を出力することで行う．$Block_n$に対応

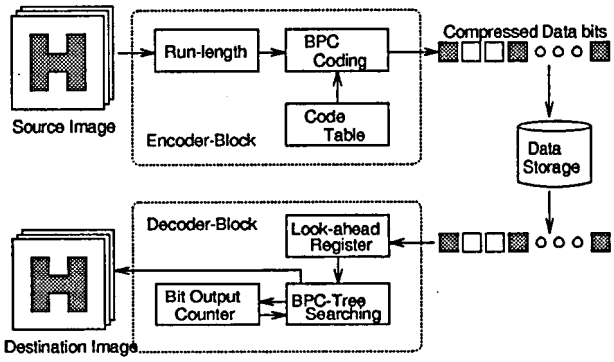

図1 BPC方式の基本システム図

する圧縮bit列の生成方法について述べる．まず，前段のRun-Length計算の結果であるRun長 $Length_n$ を2進数に変換する．$Length_n$は1から 2^K-1 の範囲であるから,2進数に展開しても高々K桁に納まる．つまり，

$$Length_n = a_0 * 2^0 + a_1 * 2^1 + \cdots + a_{K-1} * 2^{K-1}$$

（ここで，各桁 $a_k(k=0 \sim K-1)$ は0または1）

次に，各桁 a_k を下位の桁から探索し,a_k=1の場合はその桁に対応する可変長符号 $Code_k$をテーブル（表1）から参照し出力する．この表は2進数展開形式の各桁に対応する符号と符号長を並べたものであり，$K=15$ で打ち切ってある．また a_k=0の場合はコードの出力は行わない．各桁に対応する可変長符号は，画素の値である $Value_n$ の値（0か1の1bit）を先頭に付加して出力する（表1では，x で示している）．このことにより，復号時には符号語の先頭の1bitを参照するだけで，当該ブロックで出力すべき画素の値(0/1)が判る．

以上の出力を，a_0から a_{K-1}まで順に行い，$Block_n$ に対する処理を終了する． この処理を $Block_1$から $Block_N$まで行った後，画面の終端を識別するための符号である $EndCode$(表1参照) を最後に付加して，圧縮データの終わりとする．

2-2-3 符号化の例

図2に示すような入力bit列 ($N=4$) を考える．図中，黒い網掛けの領域は値が1の黒画素を示す．$Block_n$=($Value_n$, $Length_n$)($n=1 \sim N$) は，

$$Block_1 = (0,1) \qquad Block_2 = (1,2)$$
$$Block_3 = (0,4) \qquad Block_4 = (1,24)$$

表1 符号化テーブル

k	:	number of binary places
x	:	prefix of Run-Length value
$Code_k$	:	BPC code of binary place k
$\|Code_k\|$	:	code length of BPC code $Code_k$

2^k	$Code_k$	$\|Code_k\|$
2^0	$x00$	3
2^1	$x01$	3
2^2	$x10$	3
2^3	$x110$	4
2^4	$x1110$	5
2^5	$x11110$	6
2^6	$x1111100$	8
2^7	$x1111101$	8
2^8	$x1111110$	8
2^9	$x1111111000$	11
2^{10}	$x1111111001$	11
2^{11}	$x1111111010$	11
2^{12}	$x1111111011$	11
2^{13}	$x1111111100$	11
2^{14}	$x1111111101$	11
$EndCode$	$x111111111$	10

ブロック長 $Length_n$を2進数に変換する．変換後の2進数は左側の桁が上位となっている．

$$Length_1 = 000001_2 \qquad Length_2 = 000010_2$$
$$Length_3 = 000100_2 \qquad Length_4 = 011000_2$$

まず，

$$\begin{aligned} Value_1 &= 0 \\ Length_1 &= 1*2^0 + 0*2^1 + 0*2^2 + \cdots \end{aligned}$$

テーブル（表1）から2^0の桁に対応するコードを参照すると，$Code_0 = x00$ である．x の位置に $Value_1$=0を挿入して，$Block_1$は[000]と符号化される．同様に，

$$\begin{aligned} Value_2 &= 1 \\ Length_2 &= 0*2^0 + 1*2^1 + 0*2^2 + \cdots \end{aligned}$$

2^1の桁に対応するコードは $Code_1 = x01$ である． $Value_2$=1 より $Block_2$は [101] と符号化される． $Block_3$,$Block_4$についても同様の操作で各々[010],[111011110]と符号化される．最後に $EndCode$

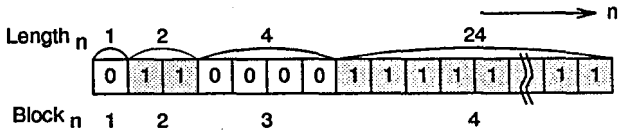

図 2 入力 bit 列の例

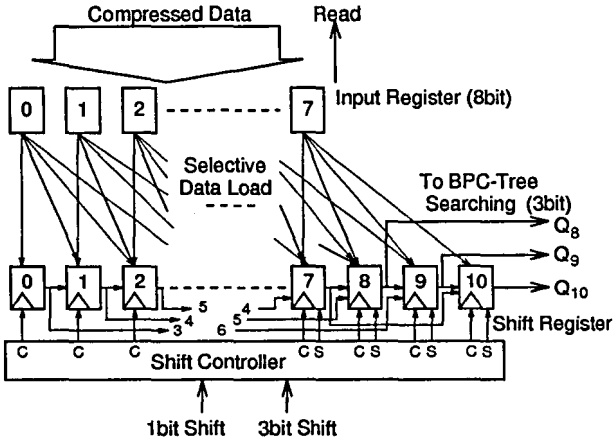

図 3 先読みレジスタの構成

を付加し以下の符号語を得る.

[000][101][010][111011110][0111111111]

2-3 復号器

逐次リアルタイム復号に適した新しいパイプライン復号器の構成方法を以下に説明する.

2-3-1 3bit 先読みレジスタ

復号器が圧縮データを読み込む際には,3bit 分の先読みを行う. 後段の符号木探索のために, 11bit のシフトレジスタ (Look-ahead Register) があって, 次に読みだすべき圧縮データの最初の 3bit が同時に参照できる. 先読みレジスタの構成を図 3 に示す. 先読みレジスタは, word 単位の圧縮データを一旦格納しておくための, 8bit 幅の入力レジスタ (Input Register) と, 1bit 幅の 11 段シフトレジスタ (Shift Register) から構成される. シフトレジスタは,2-3-2で示すステートマシンからの指示で, 1bit 又は 3bit シフトを行う.

圧縮データは, まず入力レジスタに格納される. 次に, 入力レジスタの出力が, 11 段あるシフトレジスタの 8bit に選択的にロードされる. ロードの種類は図 4

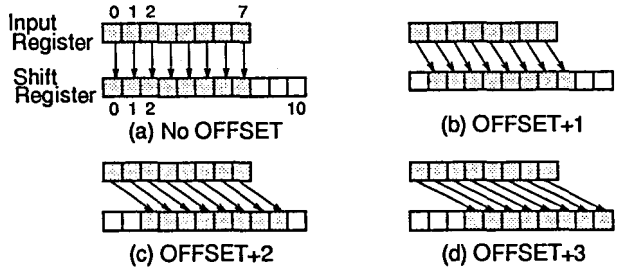

図 4 選択的なシフトレジスタへのデータロード

に示す様に (a)〜(d) の 4 種類あり, それぞれシフトレジスタへロードする場所が異なる. この機能により, 後段のシフトレジスタが 3bit シフトを行った場合でも, 1word の圧縮データを 0〜3bit 分オフセットしながら選択的にロード可能となる. シフトレジスタ内のデータは, 図中の 1bit Shift 又は 3bit Shift の端子に与えられる動作クロックに同期して 1bit 又は 3bit それぞれシフトする. シフトレジスタの「下流側」の 3bit(図中 Q_8,Q_9, および Q_{10}) は, 後段の符号木探索の際に参照される.

シフトした結果, シフトレジスタの 7bit 目に有効データが無くなると, 入力レジスタの値をシフトレジスタへロードする. シフトレジスタ内の有効データの残り方によって, 図 4のロード方法が選ばれる. 例えば, 有効データが 8・9・10bit 目に残っている時は (a), 9・10bit 目ならば (b), といった様な選択が行われる. また, ロード後, 入力レジスタには次の圧縮データが読み込まれる.

2-3-2 符号探索ステートマシンと Bit 出力カウンタによる展開データ出力

復号のためのステートマシンを図 5 に示す. r_0,$State$,d_n,wはステートマシンの各状態, 矢印付近の 3bit の数字 (Q_{10}, Q_9, Q_8) は遷移条件で, 図 3 の Q_{10}, Q_9, Q_8の値を示している. 状態 d_nに対応する $data_n$は, その次の状態へ遷移を行う時, 後段の Bit 出力カウンタへセットする値を示す. $State$ の下に書かれた記号が,$S1$ の時, その次の状態に遷移する際, 先読みレジスタを 1bit シフトし,$S3$ の時は 3bit シフトすることを示す.

ステートマシンは, 逐次リアルタイム性を確保するために, 符号の展開が完全に終る前に展開データの出力を開始する. コードの展開とは, 各符号の $Value$ と $Length$ を明らかにすることである. このうち,$Value$

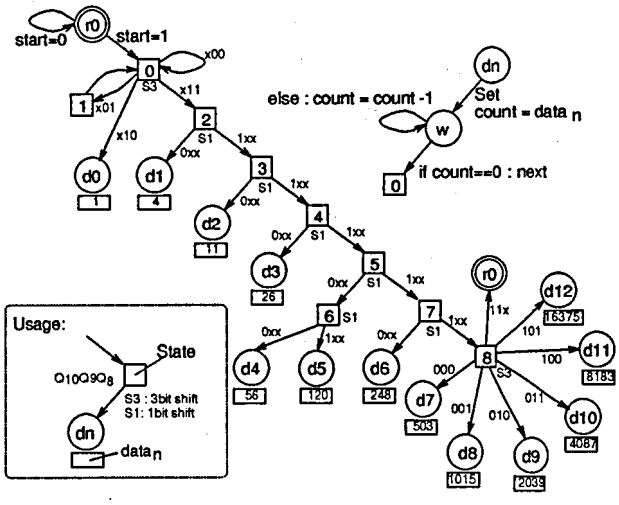

図 5　復号のためのステートマシン

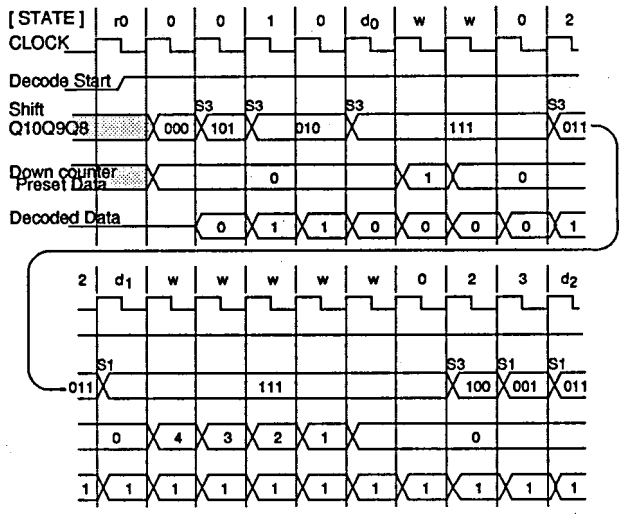

図 6　復号例のタイミングチャート

はコードの先頭 bit に示されているため, 展開前に明らかになる. このため, 先頭 bit で示される *Value* の出力をステートマシンを動作させるクロックに合わせて開始し, それと並行してコードを読み始めて *Length* を決定するという工夫をしている. ただし, この場合, 決定された *Length* の値よりも決定に要したステップ数が等しいか短い必要がある. このため,*Length* が小さい場合がある時の展開処理は,3bit シフトを用いた先読みを行い, 短いステップ数で処理が完了するようにした.

Bit 出力カウンタは, 同一出力値を連続して出力する際に, 出力回数をあらかじめセットし,1 回出力する毎に 1 減らすものである. ステートマシンは, カウンタが 0 になるまで状態 wで待ちながら, 当該ブロックの bit 値をクロックに同期して出力する. セットする値は対応する桁で出力すべき bit 数から, 値を決定するまでに要したステップ数を引いた値である.

2–3–3　復号化の例

復号化の例として, 先の **2–2–3** で得られた圧縮データを使用し, 元データが逐次リアルタイム性を確保して復元出来ることを示す. 復号の様子を示したタイミングチャートを図 6 に示す.

まず, 復号処理開始が指示されるのを待つ (初期状態 $r0$). 復号開始信号線 (Decode Start) が High になって処理の開始が指示され (状態 0), Q_{10},Q_9,Q_8を参照すると $(Q_{10}, Q_9, Q_8) = (0,0,0)$ であるので, 展開データ (Decoded Data) として 0 を出力し, 先読みレジスタに 3bit シフトを指示し, 状態 0 へ戻る. ここで $(Q_{10}, Q_9, Q_8) = (1,0,1)$ なので 1 を出力し,3bit シフトを指示し, 状態 1 へ遷移した後, 状態 0 へ戻る. 結果,2 クロックを要して値 1 の bit を 2 個出力する.

次は $(Q_{10}, Q_9, Q_8) = (0,1,0)$ なので 0 を出力し,3bit シフトを指示し, 状態 d_0へ遷移する. カウンタへ値 1 をセットし状態 wへ遷移した後デクリメントが一回行われ, 状態 0 へ戻る. 結果,4 クロックを要して値 0 の bit を 4 個出力する.

以下, 圧縮データ [111011110] についても同様に行い, 状態 $0 \to 2 \to d_1 \to w \to \ldots \to 0 \to 2 \to 3 \to d_2 \to w \to \ldots$と遷移することで, 24 クロックを要して値 1 の bit を 24 個出力する. この様に, 復号開始信号の入力から展開データが出力され始めるまでに, 状態 $r0 \to 0 \to 0$ と遷移し, 次のクロックから, 復号器の出力を利用する機能ブロック (例えばマッチング処理など) が展開データを参照できる.

3　評　価

3–1　復号ハードウェアのゲートアレイへの実装

本提案方式を実際の画像処理装置に組み込み, その有効性を明らかにするために, BPC 符号の復号器を, ゲートアレイ上に試作した. 試作には, Actel 社の FPGA(Field-Programmable Gate Arrays) [17]を用いた. 試作ゲートアレイの機能ブロックを図 7 に示す. 内部は先読みレジスタ (Look-ahead Register), BPC デコーダ (BPC Decoder), Bit 出力カウンタ (Bit Output Counter) の 3 ブロックで構成され

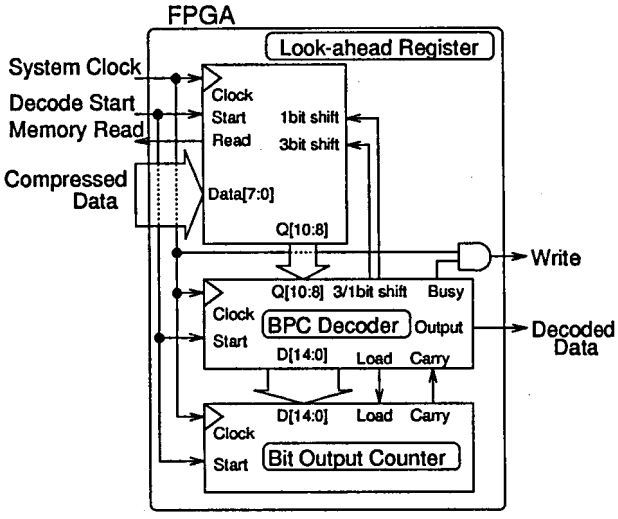

図 7 試作ゲートアレイの機能ブロック図

る. この構成は前述の図 1と対応する. 各ブロックは, ラッチ, 順序回路の全てをシステムクロック (System Clock) に同期して動作する様に設計した. 圧縮データは先読みレジスタへ入力される. 展開されたデータは, BPC デコーダより出力される. 出力のタイミングはシステムクロックと同期している.

先読みレジスタブロックは, **2–3–1**で説明した図 3 に示している内部構成をとる. BPC 符号木探索ブロックは, **2–3–2**で説明した図 5 に示したステートマシンを同期式順序回路で実装したものである. Bit 出力カウンタブロックは,**2–3–2**で説明した,ダウンカウント動作を行うものである. 試作の結果, 最高動作クロックの周期が 85nsec となり, 復号データ出力間隔は 85nsec で動作することを確認した. これは, 通常の NTSC カメラからの, 一行あたり約 600 画素の入力画像に同期して, 展開データを出力できる性能を有している. 復号器の回路規模は 1750NAND ゲート換算に収まり, 小さなハードウェア規模で構成可能であることも確認している.

3–2 画像処理装置への組み込み

上述の FPGA に実装した BPC 復号器を, マッチング処理を行う画像処理装置へ組み込み, プリント基板 (PCB) の配線パターンの品質検査システムを試作した. このシステムは, あらかじめ用意しておいた基準画像と, カメラから取り込んだ検査対象画像を比較することにより, 配線パターンの欠け・短絡といった不良箇所の検出を行う. 検査システムの機器構成を図 8

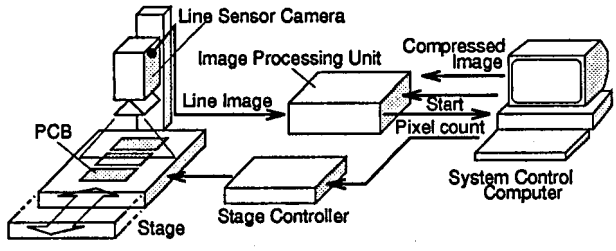

図 8 画像処理装置のシステム構成

に示す. 本システムは, 検査工程の制御用コンピュータ (System Control Computer), BPC 復号とマッチング処理を行う画像処理装置 (Image Processing Unit), PCB 表面の画像を取り込む 2 値のラインセンサカメラ (Line Sensor Camera), PCB を移動するためのステージ (Stage) 及びコントローラ (Stage Cntroller) から構成される.

ラインセンサカメラは 1 ライン当り 4096 画素の解像度であり,1 ライン毎の取り込みを行うので, カメラの走査に合わせて, ステージをカメラのラインと直角方向にステップ移動させ,4096 ステップで一枚の検査画像 (4096×4096 画素) を構成する. 検査を行う前には, 制御用コンピュータから画像処理装置内のストレージメモリへ, あらかじめ BPC 符号で圧縮しておいた基準画像 (Compressed Image) を複数枚, 転送しておく. 配線パターンの基準画像例を図 9に示す.

各機能ブロックは駆動クロックに同期したパイプライン処理を行う. **3–1**で説明した BPC 復号処理を行う試作ゲートアレイは, 図中の Decoder に対応する. ストレージメモリ上の基準画像は, 制御用コンピュータからの復号開始信号 (Decode Start) により, 復号器によって逐次リアルタイムに展開される. 復号する基準画像の選択は制御用コンピュータから指示される. 同時にカメラからの入力も開始し, 各画素を連続的に比較処理する. 比較結果から, パターンの欠けや短絡といった不良箇所の画素数 (Pixel Count) を求める.

本システムでは, 画像の取り込み,BPC 符号の復号処理, およびマッチング処理を, 85nsec/画素の速度で同時に行うことが可能であり, 4096×4096 画素サイズの画像を全て処理するために要する時間は, 約 1.43 秒となった. どのような基準/検査画像であっても, この時間を経過した後には画像取り込み, 復号, マッチング, および不良画素数のカウントの全ての処理が完了しており, 次のプリント基板の処理が可能となる. 我々は, 実際に本画像処理装置を製造ラインに組み込

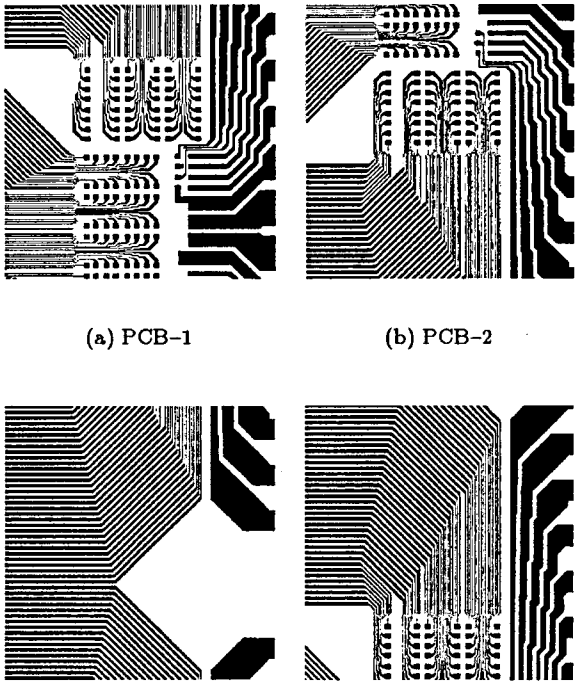

図 9 PCB 配線パターン画像

表 2 テスト画像に関する圧縮率 (%) の比較

	Run-Length	MH	BPC
CCITT-1	6.2	9.8	9.1
CCITT-2	5.7	7.8	9.3
CCITT-3	11.9	13.5	17.6
CCITT-4	21.3	22.4	30.3
CCITT-5	12.7	14.1	18.3
CCITT-6	9.5	11.0	14.5
CCITT-7	22.3	21.5	33.3
CCITT-8	11.6	12.8	18.4
Average	12.7	14.1	18.9
PCB-1	17.5	27.3	16.9
PCB-2	23.9	15.5	21.0
PCB-3	11.8	20.6	11.8
PCB-4	23.3	17.4	22.2
Average	19.1	20.2	18.0

み, 連続して製造されるプリント基板の検査装置として, 本復号処理方式が有功であることを確かめた.

3-3 他方式との圧縮率比較

Run-Length 符号,MH 符号との性能比較のため, 様々な 2 値画像データの符号化実験を行う. テスト画像は,CCITT テストチャート (CCITT-1～8, 1728×2376bit) と, 4 種類の PCB 配線パターン画像 (図 9:PCB-1～4, 4096×4096bit) を用いた. 圧縮率 (%) の比較を表 2に示す.

CCITT テストチャートに対する圧縮率について検討する. Run-Length 符号は MH 符号,BPC 符号に比較して良好な結果を示した. これは, テストチャート自身が長大 Run を多く含み, Run 長をそのまま符号化する Run-Length 符号に有利なためである. BPC符号は, 白画素の長大ブロックの比率が高い画像 (CCITT-1,2) で約 10%の圧縮率を得た. これは,BPC 符号は 2^{14}個も連続する長大ブロックに対しても, その桁に対応する符号は 10bit 程度であることによる (表 1) .

PCB 配線パターン画像に対する圧縮率について検討する. BPC 符号は, Run-Length 符号,MH 符号に比較して良好な結果を示した. BPC 符号は, パターンが無い部分が比較的広く, かつ太い配線パターンの比率が高い画像 (PCB-3) で約 12%, 横方向の細い配線パターンの比率が高い画像 (PCB-1) で約 17%の圧縮率が得られた. 一方, 縦方向の細い配線パターンが多い画像 (PCB-2,4) については, 短い Run の比率が多いため,MH 符号の方が有利な結果であった. ただし圧縮率の差は僅かであり, 提案方式は, Run-Length 符号, MH 符号と比較して圧縮率は同程度といえる.

4 まとめ

逐次リアルタイム性を確保しながら高速に復号可能な, BPC 符号化方式と, その復号ハードウェアを提案した. 提案方式は, 可逆符号化方式である, 復号時に逐次リアルタイム性が確保できる, Run-Length 符号,MH 符号と同程度の圧縮率が得られる, 復号処理が簡単でハードウェアへの実装が容易, といった特徴を有している. 提案方式の評価のため, 復号器をハードウェア化しゲートアレイ上に実装して, 実際に逐次リアルタイム復号が可能なことを確認した. 復号用ゲートアレイを組み込んだ画像処理装置を試作し, パターンマッチング処理によるプリント基板の表面検査システムを構築・試験し, 提案方式の有意性を確認した.

今後は, 提案方式の符号器もハードウェア化し, 符号/復号化の両方でリアルタイム処理を実現して, ネットワークを利用した 2 値画像通信への応用を行う予定である.

謝 辞 この研究の一部は, 平成 7 年度文部省特定研究経費により行われた. 関係各位に深謝する.

参 考 文 献

1) 安田: "マルチメディア符号化の国際標準", 丸善 (1991).

2) 原島: "画像情報圧縮", オーム社 (1991).

3) ITU-R Rec. T.81 : "Digital Compression and Coding of Continuous-tone Still Image" (JPEG), (1992).

4) D.A.Huffman : "A Method for the Construction of Minimum Redundancy Codes" , Proc. IRE, 40, 10, pp.1098–1101 (1952).

5) 中山, 末広, 他: " Run-Length 符号化による帯域圧縮の一方式" , テレビジョン学会 第 15 回録画磁気研究会資料 15-1 (1970).

6) T.S.Huang: "Coding of Two-Tone Images" , IEEE Trans. Commun.,COM-25,11,pp.1406–1424 (1977).

7) R.Hunter et al.: "International Digital Facsimile Coding Standards", Proc. IEEE, 68, 7, pp.854 – 867 (1980).

8) S.Kim, J.Lee et al.: "A New Chain-Coding Algorithm for Binary Images Using Run-Length Codes", Comp. Vision Graphics Image Proc., 41,1,pp.114–128 (1988).

9) 柳谷, 上野, 他: "適応予測を用いた多値画像のロスレス符号化", 画電学研究会予稿, 95-04-01,pp.1–4 (1995).

10) 山崎, 小野, 他: "2 値画像階層型符号化方式–JBIG アルゴリズム", 画電学誌,20,1,pp.41–49(1991).

11) ITU-R Rec. T.82 : "Progressive Bi-level Image Compression" (JBIG), (1992).

12) 勝野, 小池, 他: "JBIG コーデックと画像通信システムの開発" 画電学 第 21 回年次大会・Visual Computing '93 予稿集, pp.21–24 (1993).

13) K.Nakamura, Y.Kojima et al: "High Speed Encoding and Decoding Processor for Group 4 Facsimile Aparatus" IEEE ICC'84, pp.219–222 (1984).

14) 佐藤, 村山, 他: " 高速 2 値イメージ圧縮伸長 LSI の開発" , 信学技報, Vol.88, No.255, pp.51–58 (1988).

15) 安部井, 黒須, 他: "画像圧縮伸長の高速化方式", 情処学第 48 回大会, No.2, pp.359 (1994).

16) 長谷, 宮沢, 他: "ウィンドウ調整回路内蔵高速符号・復号 LSI の開発", 信学会秋大, C-551, pp.219 (1990).

17) "The FPGA Data Book and Design Guide : ACT-1 family" , Actel Corporation (1996).