システムＬＳＩ設計技術　109－9
(2003. 3. 11)

遺伝的アルゴリズムを用いたITC符号化プロセッサの開発

古川　剛史†　宮内　新†　石川　知雄†

† 武蔵工業大学工学部　〒158–8557 東京都世田谷区玉堤 1-28-1
E-mail: †{furukawa,miyauchi,ishikawa}@ic.cs.musashi-tech.ac.jp

あらまし　ＩＴＣ符号化とは画像パターンの冗長性を利用して画像データの圧縮を行なう符号化技術の１つである．ＩＴＣ符号化はＪＰＥＧと同等の圧縮率を持ちながらも復号化が高速に行なえるという特徴を持つ．しかし符号化処理には，自己相似部分の探索を必要とし膨大な演算時間がかかる．そこで自己相似部分の探索に遺伝的アルゴリズムを用い，さらにアルゴリズムに適したプロセッサの開発を行い符号化の高速化を図る．

キーワード　遺伝的アルゴリズム，ITC符号化，プロセッサ

ITC PROCESSOR APPLYING GENETIC ALGORITHM

Takeshi FURUKAWA†, Arata MIYAUCHI†, and Tomo ISHIKAWA†

† Faculty of Engineering, Musashi Institute of Technology　Tamazutsumi 1–28–1, Setagaya-ku, Tokyo, 158–8557 Japan
E-mail: †{furukawa,miyauchi,ishikawa}@ic.cs.musashi-tech.ac.jp

Abstract　This paper proposes a processor for ITC (Iterated Transform Coding), in which GA (genetic algorithm) is applied.　ITC is one of image-coding technology, in which Fractal theory is applied.　ITC has almost same rate of compression as JPEG and it can be decoded faster than JPEG.　But, encoding needs much computation time for searching partial self-similarities.　The purpose of this research is to reduce the time for encoding. We apply GA (genetic algorithm), which is one of the search algorithms for searching partial self-similarities.　We examined the architecture for high-speed execution of this algorithm.

Key words　ITC, GENETIC ALGORITHM, PROCESSOR

1. はじめに

近年，急速なモバイル機器の普及に伴い携帯電話，PDAなどの処理能力の限定されている環境においてディジタル画像・映像が使用されることが増加傾向にある．画像符号化の分野ではJPEGやJPEG2000といった符号化方式が標準規格として確立されているが，携帯電話のコンテンツ提供などのシステムにおいてはJ-PHONEのNancy Codecのような独自の符号化技術を採用している場合がある．このように標準化が進められている符号化方式に限定されることなく，携帯電話などの画像処理処理能力の低い環境であっても高速に復号処理を行うことができる符号化技術が求められている．

そこで本研究では復号処理が高速に行えるという特徴の持つITC符号化 (Iterated Transform Coding) について取り扱う．ITC符号化はA.Jacquinにより提案されたフラクタル理論を用いた符号化方式の１つである．一般的に自然画像には部分的自己相似性が多くみうけられる．この部分的自己相似性という同じ画像内に存在する画像パターンの冗長性を利用して画像データの圧縮を行なう方式がITC符号化である．ITC符号化はJPEGと同等の圧縮率，画質を持ちながらもDCT変換，wavelet変換を用いた画像圧縮手法と比較し復号処理が高速に行えるといった特徴がある．

ITC符号化はこのような優れた特徴を持つ反面，符号化時には同一画像内での自己相似部分を探索するのに膨大な計算量を必要とし，処理時間が多くなるという欠点を持つ．ITC符号化を高速実行するアプローチとして本研究室では自己相似部分の探索に遺伝的アルゴリズム (Genetic Algorithms:以下GA) を用いたアルゴリズムを提案し，従来の手法に比べ符号化処理の実行時間を短縮できることを示した[1]．しかし，依然GAを用いたITC符号化アルゴリズムを汎用逐次処理コンピュータ上で実行するには処理時間が多い．そこで符号化時の膨大な実行時間を短縮するため，GAを用いたITC符号化の高速実行に適したプロセッサの開発を行なう．GAを高速化するアプローチとしては，広島大学の”遺伝的アルゴリズムの高速実行に適した命令セットを持つRISCプロセッサDLX-GA”[2]や，立命館大学の”GAエンジンのPE-LSI”[3]が提案されており実行

クロック数を大幅に減少させる成果をあげている．本研究ではGAの高速実行のアプローチの1つであるハードウェア化を画像符号化という，より実用的なアルゴリズムに特化させ実行に適したアーキテクチャを提案する．

2. ITC符号化の基本原理

ITC符号化は任意の初期画像に対して縮小変換を反復的に施して得られる画像が収束するという原理に基づいている．符号化時には変換係数を求め自己相似部分を探索する．符号化の過程を示す．

まず入力画像を互いに重ならないレンジブロックと呼ばれる小さなブロックの集約に分割する．次に各レンジブロックに対して，入力画像内でドメインブロックと呼ばれるレンジブロックに似ている画像パターンを探索する．この探索の過程で各ドメインブロックからレンジブロックへの変換係数が求められる．ただし，ドメインブロックからレンジブロックへの変換は縮小写像でなければならないため，ドメインブロックはレンジブロックよりも大きくなければならない．またドメインブロックは重なり合ってもよい．

レンジブロックとドメインブロックの相似性を表すものとして，式(1)の誤差値を定義する．cはコントラスト，bは輝度オフセット，r_iはレンジブロック内のピクセル，d_iはドメインブロック内のピクセルとする．コントラストとオフセットとはドメインブロックをアフィン変換する係数である．コントラストはドメインブロックの輝度を定数倍する係数であり，オフセットとはドメインブロックの輝度に加算される定数である．

$$E(c,b)=\sum_{i=1}^{n}(cd_i+b-r_i)^2 \qquad (1)$$

誤差値が最小値となるコントラストcと輝度オフセットbを算出し，式(1)から誤差値を求める．この誤差値が最小となる最適なドメインブロックを探索し，ドメインブロックのアフィン変換係数であるコントラストcと輝度オフセットbとブロックの位置をITCコードとして記録する．このことで画像データの冗長性が取り除かれデータサイズの圧縮が実現される．

相似性の高いドメインブロックを探索する手法として，遺伝的アルゴリズムを適応させた手法が本手法である．

3. 遺伝的アルゴリズム

3.1 遺伝的アルゴリズムの基本原理

遺伝的アルゴリズム（GA）は生物の進化過程における遺伝子の複製・選択・淘汰のメカニズムをモデルとした確率的探索・学習・最適化の一手法である．GAはVLSI設計の自動化，画像処理等の組み合わせの最適化問題を中心として工学的に広く応用されており，ヒューリスティックな手法とランダム探索法を組み合わせた多点探索を繰り返し行う手法である．このことにより従来の手法では解くことが困難であった問題に対しても高い性能を示すことが多い．

GAは遺伝子に対し以下に示す3種類の操作，選択，交叉，突然変異の遺伝子操作を基本として構成される．

・選択　：繁殖のための個体を集団の中から選択するものである．より適合度（個体が与えられた環境に適応している度合いを表すもの）の高い染色体ほど，繁殖の機会が多くなるように選択される．

・交叉　：2つの個体を親とし，それらの遺伝子から子の遺伝子を生成する操作である．個体を選択し，2つの個体間で遺伝子の位置を交換し，子孫を生成する．

・突然変異　：個体の遺伝子をランダムに変更する操作である．突然変異は，局所的な解に陥った時にそこから脱出する働きがある．

4. ITC符号化へのGAの適応

4.1 GA適応範囲の検討

様々な特徴をもつ画像全体を一様なサイズのレンジブロックで分割するのは望ましくない．よって16×16，8×8，4×4の3サイズで分割する．まず16×16のレンジブロックに対し相似性の高いドメインブロックを探索する．閾値以上の相似性があるドメインブロックが存在した場合にはドメインブロックに対するITCコードを記録する．閾値以下の場合はレンジブロックを分割しドメインブロックが存在するか，最小のレンジブロックサイズ(4×4)まで分割した場合ITCコードを記録する．このことで特徴のあまり無い部分には16×16のレンジブロックを適用し圧縮率を上げ，輪郭など細かな部分には4×4のレンジブロックを適用する．このことにより圧縮率を上げながらも画質を劣化させないことが可能となる．CIFサイズの256階調の画像に対して，レンジブロック，ドメインブロックの個数，比較回数を表1に示す．

表1　レンジブロックサイズによる比較回数

レンジブロックサイズ	レンジブロックの個数	ドメインブロックの個数	比較の回数	比較の割合
16×16	369	99	36531	0.3 %
8×8	1620	396	641520	5.7 %
4×4	6480	1620	10497600	94.0 %

16×16，8×8のレンジブロックに遺伝的アルゴリズムを適用し近似解を求めるとブロックサイズが大きいため極端な画質低下が見られる．また，2つのサイズの探索回数を合計しても全体の10 %未満であることや，探索範囲が狭くGAが適応しにくい．そのため、これらのサイズのレンジブロックには全探索を行い，探索回数の多い4×4サイズのレンジブロックのみ遺伝的アルゴリズムを適応し探索の高速化を図った．

4.2 GAのハードウェア化に対する検討

また，GAを用いたITC符号化アルゴリズムに適したアーキテクチャを構成するにあたり，よりハードウェア化に適したGAの戦略を選択する必要がある．ハードウェアとして実装することに対し交叉，突然変異操作では戦略の差異はないが，選択操作(繁殖のために親を集団の中から選択する操作)は戦略によってハードウェア化に大きな差が生じると考える．選択操作には，適応度比例戦略，ランク戦略などの様々な戦略がある．

各戦略についての説明を以下に示す.

＜適応度比較（ルーレット）戦略＞

適応度の比例した確率を各個体に割り当て、この確率にしたがって各個体を淘汰・増殖する.

＜ランク戦略＞

ランク戦略は、適応度によって各個体をランク付けし、あらかじめ各ランクに対して固定された確率で子孫を残せるようにする.

選択操作のハードウェア化を考えるにあたり，各ランクに対して決められた確率で子孫を残せるようにするランク戦略がハードウェア化に適していると考えた．選択戦略の違いが実行時間，画質に与える影響について考察する．選択操作に適応度比例戦略，ランク戦略を用いたアルゴリズムに対しビットレートごとの実行時間，画質の測定を行い結果を図 1～4 に示す．評価画像は salesman を用い，サイズは CIF,QCIF サイズを用いた．実験環境は，CPU:Pentium(R)III Processor 450MHz，メモリ:192MB，OS:Windows2000 の汎用 PC を用いた.

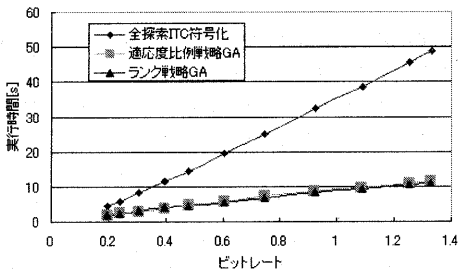

図 1　各アルゴリズムによる実行時間の比較
(評価画像:salesman　画像サイズ:CIF)

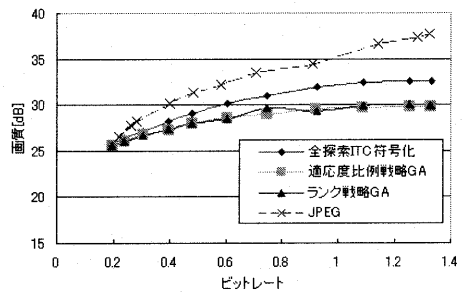

図 2　各アルゴリズムによる SN 比の比較
(評価画像:salesman　画像サイズ:CIF)

GA を適応することで実行時間は全探索に比べ CIF サイズで約 70～80 %削減することができた．QCIF サイズでは画像サイズが小さく探索範囲も小さいため，探索回数は減少したものの遺伝子操作の処理が増加し，実行時間を遺伝子操作が多く占めるようになり実行時間の削減は約 30 %程度となった．SN 比は全探索と GA を用いたものを比較すると GA を用いたアルゴリズムの方が画質の低下が見られる.

また，適応度比例戦略とランク戦略では実行時間，SN 比共に大きな違いは見られなかった．このことから，ITC 符号化に適応させた GA の選択操作において戦略による差異はないと考える．よって，ITC 符号化に適応させる GA にはハードウェア化が容易であると考えられるランク戦略を採用する．後述する選択回路は，ランク戦略に基づく固定された確率で個体が選択される回路となる.

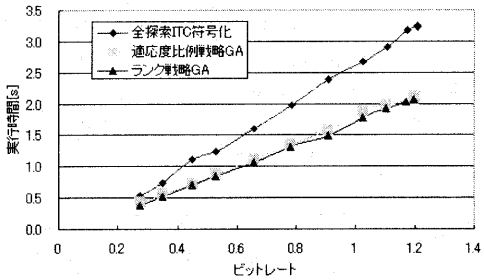

図 3　各アルゴリズムによる実行時間の比較
(評価画像:salesman　画像サイズ:QCIF)

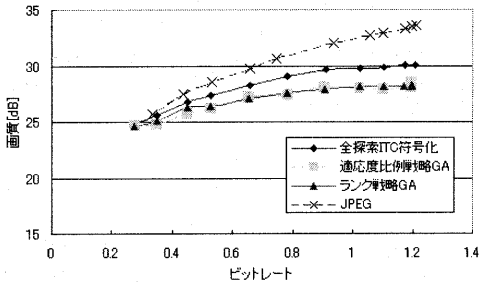

図 4　各アルゴリズムによる SN 比の比較
(評価画像:salesman　画像サイズ:QCIF)

5.　ITC 符号化プロセッサの実装環境

本プロセッサを実装する環境として Nios Embedded Board を用いる．また，ベースプロセッサとして 32bit-RISC プロセッサである Nios プロセッサを用いる．Nios プロセッサとは，ALTERA 社製のプログラマブル・ロジック専用に開発されたコンフィギュラブルな RISC 型ソフト・プロセッサ・コアである．NIOS プロセッサをベースプロセッサとし専用回路の追加実装を行い GA を用いた ITC 符号化プロセッサを作成する．そして実行クロック数や，符号化した画像データの画質を測定しアーキテクチャの有用性を検証する．Nios プロセッサのブロック図を図 6 に示す.

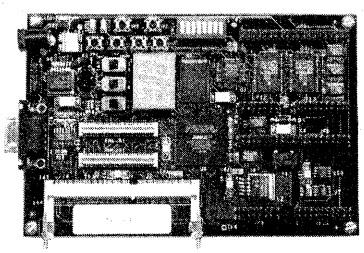

図 5　Nios Embedded Board

後述する積和演算器，選択回路，交叉回路，突然変異回路はすべて図7のようにNiosプロセッサのALU部分に追加実装される．それぞれが積和演算命令，選択命令，交叉命令，突然変異命令としてプロセッサに実装される．表2にベースプロセッサの論理合成結果を示す．ベースプロセッサの動作周波数を低下させないように上記の回路を設計した．

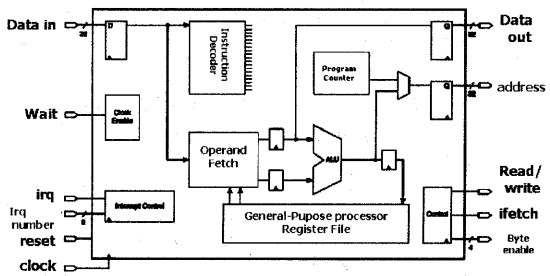

図6　Niosプロセッサ

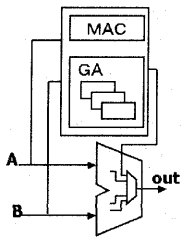

図7　専用回路の追加

表2　論理合成結果の比較
(APEX EP20K200EQC208)

	ベースプロセッサ
動作周波数	50.8[MHz]
ピン数	109/136 (80.15%)
回路面積 (LCs)	3040/8320(36.54%)

6. 専用回路の実装

GAを用いたITC符号化プロセッサはNiosプロセッサをベースプロセッサとしITC符号化の高速実行に適した回路と，GAの高速実行に適した回路を追加実装することによって実行クロック数の削減を図る．ITC符号化の高速実行に適した回路はレンジブロックとドメインブロックの相似性比較に要する実行クロック数を削減する．またGAの高速実行に適した回路は遺伝子操作に要する実行クロック数を削減する．本章ではそれぞれの回路の詳細を示す．

6.1 ITC符号化の高速実行に適したハードウェア

ITC符号化において最も処理時間を要するレンジブロックとドメインブロックの相似性の評価について処理時間の短縮を図る．前述の式(1)はドメインブロックを縮小変換しアフィン変換した後にレンジブロックとの二乗誤差の和である．この誤差値をコントラストcと輝度オフセットbの関数とし最小自乗法を用いてこの誤差値が最小となるコントラストcと輝度オフセットbを求める．式(1)をc, bで偏微分し連立方程式を解くと式(2)と式(3)が求められる．

$$c = \frac{\sum r_i \cdot d_i - \sum r_i \sum d_i}{n \sum {d_i}^2 - (\sum d_i)^2} \tag{2}$$

$$b = \frac{\sum r_i - c \sum d_i}{n} \tag{3}$$

上記の式から，誤差値が最小値となるコントラストcと輝度オフセットbを算出し，さらに式(1)で誤差値を求める．この誤差値が最小となる最適なドメインブロックを探索する．このドメインブロックのコントラストcと輝度オフセットbとブロックの位置をITCコードとして記録する．この誤差値を算出する演算で，最も演算量が多い部分が式(2)の$\sum r_i \cdot d_i$の演算である．$\sum r_i$, $\sum d_i$,$\sum {r_i}^2$,$\sum {d_i}^2$は，レンジブロック，ドメインブロック固有の値であるが，$\sum r_i \cdot d_i$はレンジブロック，ドメインブロックの関係で決定される．そこで$\sum r_i \cdot d_i$の演算を行う積和演算器を作成した．積和演算器のブロック図を図8に示す．

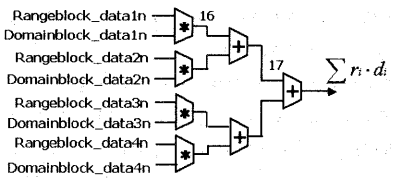

図8　積和演算器

この積和演算器はレンジブロック，ドメインブロックの画素値を入力とし積和演算を行い結果を出力する．入力を8bitと限定した乗算器を用いることで遅延を抑え，積和演算命令を1クロックサイクル内で実行可能とした．

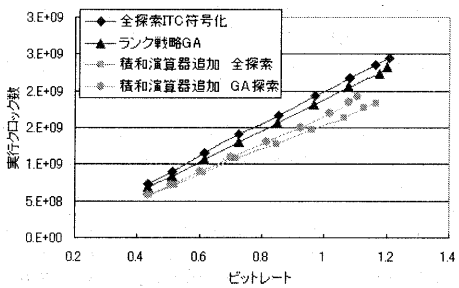

図9　プロセッサごとの実行クロック数比較

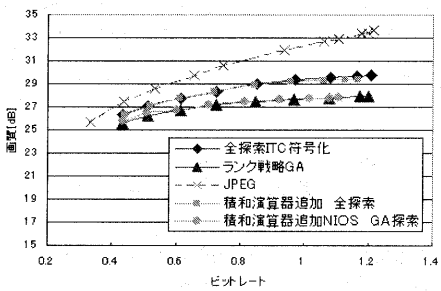

図10　プロセッサごとの画質評価

積和演算器を追加したNIOSプロセッサ上で全探索ITC符号化とGAを用いたITC符号化プログラムを動作させ実行クロック数，SN比を測定し結果を図9，10に示す．評価画像はsalesmanを用いサイズはQCIFサイズを用いた．

積和演算器を追加すると全探索アルゴリズムで実行クロック

数は約 10〜20 %削減されており SN 比は大きな変化は見られない．GA を用いた ITC 符号化アルゴリズムでは全探索アルゴリズムに比べ実行クロック数の削減率は少なくなっている．これは GA によりレンジブロックとドメインブロックの比較回数が減り，積和演算器の使用率が減少しているためと考えられる．この場合においても積和演算器の追加による画質への影響は見られない．このことにより積和演算器は ITC 符号化に有用であると考える．また，積和演算回路の論理合成結果を表 3 に示す．動作周波数は 56.6[MHz] となりベースプロセッサの動作周波数には影響を与えないと考える．

表 3　積和演算回路の論理合成結果 (APEX EP20K200EQC208)

最大遅延	17.64[ns]
動作周波数	56.6[MHz]
回路面積 (LCs)	550/8320

6.2　遺伝的アルゴリズムのハードウェア化

遺伝的アルゴリズムの高速実行を実現させるためにハードウェアとして実装を行う．GA は選択，交叉，突然変異の 3 つの操作で構成されており，各操作に対する専用回路を実装し高速化を図る．GA は各操作において乱数を多用する．乱数は普通 10〜50 以上の命令が必要となり演算コストは高い．そのため各操作で用いる乱数は線形フィードバックシフトレジスタに基づいた M 系列乱数ジェネレータを用いハードウェアにおいて発生させる．

6.2.1　M 系列乱数ジェネレータ

M 系列 (Maximum length sequence) は図 11 に示すような n 段のシフトレジスタの各段に f(0 または 1) なる係数をかけ，フィードバックをかけた回路で発生される 2 値系列である．10〜50 命令必要だった乱数の発生を乱数ジェネレーターを用いることにより 1 クロックサイクルで行なうことができる．この乱数ジェネレーターは後述の選択回路，交叉回路，突然変異回路に用いられている．

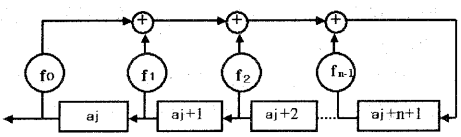

図 11　M 系列乱数ジェネレータ

6.2.2　選択回路

選択は繁殖のための個体を集団の中から親となる個体選択する操作でより適応度の高い染色体ほど繁殖の機会が多くなる．選択操作の戦略は前述の結果を踏まえランク戦略を実装する．乱数ジェネレータによって発生した乱数により固定された選択確率でランクが選択される．そして，ランク内に存在する親遺伝子の番号の中から均等な確率で親の番号が出力される．選択命令が発行されていない場合でも乱数ジェネレーターは乱数を出力し続けるので高いランダム性を持った乱数が得られ，さらにランク戦略選択命令は 1 クロックサイクルで実行可能となる．ランク戦略選択操作をソフトウェアで実装した場合 NIOS プロセッサ上では 203 クロック実行に要したが，ハードウェアで実装した場合 1 クロックサイクルで可能である．

6.2.3　交叉回路

交叉は 2 つの遺伝子を親としそれらの遺伝子から次の世代の遺伝子を生成する操作である．図 13 に交叉回路を示す．乱数ジェネレータとバレルシフタによってマスクを作成し親 1 とマスク，親 2 と反転したマスクの論理積，結果に論理和を行うことによって実現される．交叉命令も選択回路同様 1CLK サイクルで実行可能となる．交叉をソフトウェアにて実装した場合，192CLK サイクル必要であった処理を本回路では 1CLK で実現できる．

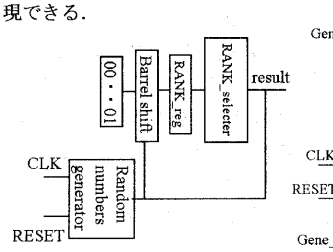

図 12　選択回路

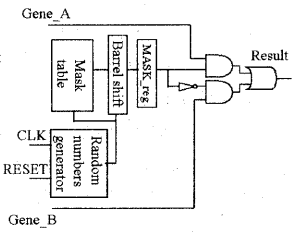

図 13　交叉回路

6.2.4　突然変異回路

突然変異は，個体の遺伝子をランダムに変更する操作であり局所的な解に陥ったときにそこから脱出する働きがある．乱数により突然変異をする箇所を決定しマスクを作成する．マスクと排他的論理和を行うことにより，指定した箇所のみ反転し突然変異の操作が行える．図 14 に突然変異させる回路を示す．突然変異回路では交叉回路同様，命令が発行されていない状態であっても 1CLK サイクルごとにマスクを生成している．突然変異命令も 1CLK サイクルで実行可能である．突然変異をソフトウェアにて実装した場合，119CLK サイクル必要であった処理を本回路では 1CLK サイクルで実現できる．

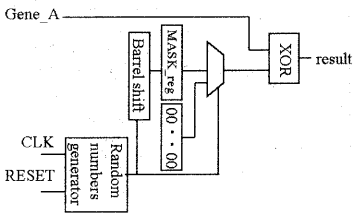

図 14　突然変異回路

各回路の論理合成結果を表 4 に示す．各回路共にほぼベースプロセッサと同じ動作周波数で動作し，ベースプロセッサの動作周波数には影響を与えないと考える．

表 4　各回路の論理合成結果 (APEX EP20K200EQC208)

	選択回路	交叉回路	突然変異回路
最大遅延	20.19[ns]	20.19[ns]	20.19[ns]
動作周波数	49.2[MHz]	49.2[MHz]	49.2[MHz]
回路面積 (LCs)	966/8320	992/8320	976/8320

7. 測定結果と考察

上記の GA 回路を追加実装したプロセッサとさらに積和演算回路を追加実装したプロセッサにおいて，実行クロック数と SN 比を測定した．それぞれの結果を図 15，16 に示す．評価画像は salesman 用いサイズは QCIF サイズを用いた．ベースプロセッサと GA 回路を追加したプロセッサの実行クロック数を比較すると約 50 %程度削減されている．画質については全探索アルゴリズムと GA を用いたものでは差があるものの，GA 回路を持つプロセッサとベースプロセッサで GA を実行したものでは差はなかった．このことから GA をハードウェアで実装することで ITC 符号化に及ぼす影響は無いと考えられる．

さらに積和演算器を実装することでさらに 10〜20 %のクロック数が減少でき，ベースプロセッサと比較すると約 60 %の実行クロック数が削減された．画質についても積和演算器による大きな影響は見られなかった．これらのことから積和演算器と GA 回路は本アルゴリズムに有用であると考える．

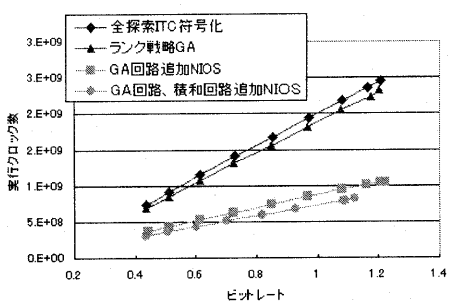

図 15　各プロセッサにおける実行クロック数

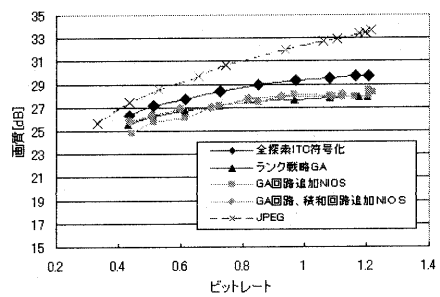

図 16　各プロセッサにおける画質

また，積和演算器，GA 回路を追加実装したプロセッサについて論理合成を行った結果を表 5 に示す．論理合成には Mentor Graphics 社製の LeonardoSpectrum を用い，対象デバイスは ALTERA 社製の APEX EP20K200EQC208 を用いた．

回路面積に関しては，積和演算器，GA 回路を追加実装することで積和演算器がデバイス全体の約 7%，GA 回路がデバイス全体の約 12%を使用し増加しているが動作周波数は専用回路を追加することで大きな変化は見られない．GA 回路においては，各遺伝子操作回路が持つバレルシフタがクリティカルパスとなりプロセッサ全体の動作周波数を低下させてしまっている．動作周波数は積和演算器で 0.3[MHz]，GA 回路でも 1.6[MHz] 減少しているが全体の動作周波数に比べ，低下率が低いため専用演算器を追加実装することで動作周波数に与える影響はほとんど無いと考える．

よって，GA 回路を追加実装することで QCIF サイズの画像において GA を用いた ITC 符号化の処理に対し，ベースプロセッサの動作周波数を下げることなく約 50%の実行クロック数を削減させた．さらに積和演算器を追加実装することで約 10%の実行クロック数が削減され，ベースプロセッサと比較すると，動作周波数を低下させることなく約 60%実行クロック数を削減させた．

表 5　論理合成結果の比較 (APEX EP20K200EQC208)

	ベースプロセッサ	積和演算器追加プロセッサ
動作周波数	50.8[MHz]	50.5[MHz]
ピン数	109/136 (80.15%)	109/136(80.15%)
回路面積 (LCs)	3040/8320(36.54%)	3623/8320(43.44%)

	GA 回路追加プロセッサ	積和演算器&GA 回路追加プロセッサ
動作周波数	49.2[MHz]	49.2[MHz]
ピン数	109/136 (80.15%)	109/136(80.15%)
回路面積 (LCs)	4860/8320(58.41%)	5638/8320(67.76%)

8. ま と め

・GA の高速実行のアプローチの 1 つであるハードウェア化を画像符号化という，より実用的なアルゴリズムに適応させ，アルゴリズムの実行に適してるアーキテクチャを提案した

・GA を適応することで CIF サイズの画像に対し実行時間を約 70〜80 %減少することができた．QCIF サイズでは探索回数は減少したものの遺伝子操作の処理が増加し，実行時間の割合を遺伝子操作が多く占めるようになり実行時間の削減は約 30 %程度となった．

・積和演算器，GA 回路を用い GA を用いた ITC 符号化アルゴリズムの実行に適したアーキテクチャを提案，実装しベースプロセッサの動作周波数を低下させることなく実行クロック数を約 60 %削減させ，本アーキテクチャの有用性を示した．今後，動画像に対する検討も行い動画像の実時間符号化処理をめざす．

文　献

[1] 宮内新，穴田高康，石川知雄；遺伝的アルゴリズムを用いた動画像の反復変換符号化，映像情報メディア学会誌 vol.53・No.11，pp.1633-pp.1635，November 1999

[2] 小泉 慎哉，若林 真一，小出 哲士，井村 紀道，藤原 一成:広島大学，"遺伝的アルゴリズムの高速実行に適した命令セットを持つ RISC プロセッサ DLX-GA"，2001 年度情報処理学会計算機アーキテクチャ　研究報告 NO.141，pp.65-pp.70

[3] 吉川 雅弥，今井 哲也，寺井 秀一，山内寛紀:立命館大学，"GA エンジンの PE-LSI"，第 3 回 LSI IP デザイン・アワード受賞論文