計算機アーキテクチャ 125－6
（1997．8．20）

FPGAによるSIMD型GAマシンの設計

佐野雅彦，井上富夫，高橋義造
徳島大学工学部知能情報工学科

概要：遺伝的アルゴリズム（GA: Genetic Algorithms）のためのSIMD型専用並列マシンを設計中である．GAでは各アプリケーションに応じた多様な処理方法が提案されているが，それらは専用ハードウエア上では適用範囲が制限される．そこで本研究では，書き換え可能なFPGAによる適用性の拡大と，複数FPGAによる並列処理よって速度面での改善を目的とする．本論文では提案するSIMD型並列計算機のプロトタイプのアーキテクチャおよびプロセッサエレメントの概要について述べる．

A Design of the SIMD GA-machine with FPGA

Masahiko Sano, Tomio Inoue, Yoshizo Takahashi
The department of Information Science and Intellignet Systems
Faculty of Engineering, Tokushima University

Abstract. We design a SIMD Parallel computer for Genetic Algorithms. Various processing method for GA has been proposed but its adaptability on the application specific hardware is low. Our goals are to provide more adaptivity for GA's applications with the re-configurable FPGA and to increase computing speed by parallel processing of the FPGAs. In this paper, we describe the prototype architecture of SIMD parallel computer and its processor element.

1．はじめに

近年注目されている組み合わせ最適化問題の探索手法に遺伝的アルゴリズム（GA）[1,2]がある。GAは生物の進化の仕組みを取り入れたもので，遺伝子の交叉と突然変異により，優良遺伝子を次の世代に残す方法である．この手法の特徴は，複雑な問題において他の手法よりも短時間での解の発見が期待できることにあるが，大規模な問題では計算量が多く，並列計算機による高速化が望まれている．このような背景から並列計算機による高速化および並列GA（Parallel GA）が研究されており，グローバルモデル[3]，アイランドモデル[3]，超並列モデル[4]等の幾つかの処理モデルが提案されている．GAでは遺伝子毎に異なる操作や遺伝子（または個体）の評価によるランキング戦略やエリート戦略による大域的通信が行われることも多く，その手法も問題毎に異なり，また他の手法と組み合わせて用いられることも多い．このため単純なGA以外ではMIMD型並列計算機を使用するのが一般的である．一方，SIMD型並列計算機によるPGAでは，プログラムによる問題への適用性はMIMD型並列計算機と比較して劣るものの比較的高く，同数のロセッサエレメント（PE）の場合，SIMD型はMIMD型並列計算機よりも低コストで提供できる利点があり，アプリケーションによってはSIMD型並列計算機の特徴である多数のプロセッサを有効活用できる利点もある．しかし個体（遺伝子）毎に複雑な操作を伴うアプリケーションには，SIMD型並列計算機の性質上対応が難しい点が指摘される．一方専用ハードウエアによる試みも行われている．代表的な専用ハードウエアによるGAは，評価，選択，交叉，突然変異の一連の処理をパイプライン化された専用演算器により処理する方法[5-7]である．これはランキング戦略やエリート戦略等の一般的なGAにおける手法の適用が容易であること，低コストで高速であることが利点であるが，特定の問題を対象としており，VLSI化した場合

に応用範囲が極度に限定される問題がある．このためFPGA等の可変構造デバイスに実装することで応用範囲の制限を緩和する例[5,6]が報告されている．しかしながら，パイプライン方式によるGA専用マシンでは，性能向上はパイプラインの速度に依存するが，現在のFPGAの動作速度はそれ程速くない．このため既存の並列計算機に匹敵する性能を得るには演算器を並列動作させる必要がある．しかしFPGAのプログラム生成（例えばVHDL言語を用いた場合の論理合成からデバイスへの書き込みまでの）の所要時間は一般的な計算機におけるプログラムのコンパイル時間よりも遥かに長時間となるため，ハードウエアとソフトウエアのトレードオフも重要である．

このような背景を踏まえた上で，我々はGA専用SIMD型並列計算機[8,9]を研究中である．本研究のでは多様なGAの並列処理をFPGA上に構築する専用ハードウエアで高速処理することを目的としており，その方針を以下に示す．

・書き換え可能FPGA　各種の処理法方を実装するため書き換え可能なFPGAを採用する．遺伝操作や評価を専用回路化することで高速化を図る．また，PEやCPをFPGA上に実装することにより専用化に伴う柔軟性の低下を防ぐ．

・ハードとソフトのトレードオフ　ハードとソフトの割合を変更することにより，目的に応じた構成とすることが可能となる．

・制御性の向上　制御プロセッサ（CP）からの実行命令アドレス放送とこれを利用した同期機構により，従来ホストから制御されてきた複数の条件ネスト処理やSPMD的動作をCPとプロセッサ要素（PE）間で実現することにより，各PEの制御性の向上を図る．また，制御プロセッサ（CP）の高機能化により，CP単位で異なる処理を可能とする．これらにより複雑な遺伝子構造や個体（または遺伝子）毎に依存する操作に対して柔軟性が向上すると考えられる[9]．

以上のことから，我々はアーキテクチャの検討のため，FPGAによるSIMD並列計算機のプロトタイプを設計中であり，SIMD型専用マシンの有効性および問題点を検討する予定である．現在PEの設計をほぼ終えている．このPEはRSIC的な４段パイプラインを持つアーキテクチャに遺伝操作専用演算器を備えたものであり，ソフトウエア重視の設計としている．本論文では提案するSIMD型並列計算機アーキテクチャのうち，設計したPEの概要およびハードウエアの見積と評価のためにFPGAに実装したPEについて述べる．

２．SIMD型並列計算機におけるSPMD的動作

プログラムの制御構造にはループ処理，IF～THEN～ELSE処理等があり，これらは条件分岐命令を基に実現される．CM-2等のSIMD型並列計算機では全PEに共通な制御部分はホスト又はCPで処理されるが，PE依存の条件処理がネストする場合，実行するPEに対するマスクが必要となり，条件ネストに対して複数ビット割り当てる方法が採られている[10,11]．これに対して提案方式では，CPの命令アドレス放送とこれを利用した同期機構によりSPMD的動作を実現する[9]．

以下ではPEで実行される簡単なプログラムを例に説明する．ただし，P1～P7は制御構造を含まないコード列とする．このプログラム構造を図１に示す．SIMD型並列計算機では実行可能状態にあるPEは同じ命令を実行するためコード列p2,p4は両方ともPEへ送られる必要がある．よってCPはPEに p1 p2 p3 p4 p5 [p6 .. p6] p7の順でコード列を送り出す．ここで２台のPE1,PE2を仮定する．両PEはp1を実行後cond1の条件によりPE1では真，PE2では偽となるとする．CPはコード列p1に続いてP2を送出するためPE1は実行を継続する．PE2はp4に前方分岐するが，コード列p4が送られるまで待機しなければならない．待機する方式としてはPEの状態に応じて条件実行[10]を行う方式や実行をマスクするフラグ[11]を用いて実行制御する方法が提案されているが，本提案方式ではこの前方分岐の実行と同時にp4の先頭で自動的に待機状態となる．待機状態中のプロセッサは目的のコード列を識別して実行を再開する．よって，CPからの明示的な実行制御の必要性が無いのでPE毎の状況に応じた柔軟な処理が可能となる．待機状態にあるPEの実行を再開させるには，各コード列の先頭にどのコード列が送られる

かを示す特別な命令（ラベル命令など）を付加する方法が報告されているが[11]，本研究では単純にCPが命令と共に放送する命令アドレスを監視する方法を用いた．これにより命令アドレスをキーとしたバリヤ同期が簡単に実現され，特別な命令が不用となる利点がある．

```
Program
{
        p1 ;
        if(cond1) {
                p2 ;
                if(cond2) p3 ;
        } else p4
        p5 ;
        while(cond3)  p6 ;
        p7 ;
}
```

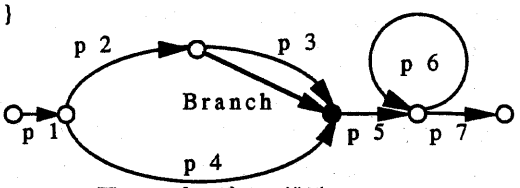


図１　プログラム構造

２.１実行効率の改善

前述の図１の例で，仮にcond1で全PEが条件を満たさないとすると，p2 p3のコード列の送出は無駄である．そこでコード列p2を実行するPEが存在しない場合，CPはコード列p2の送出を中断し，コード列p4を送出することで無駄を最小限に抑えることができる．制御の移行先はコンパイラが予めテーブル作成するものとする．制御の移行は無条件および条件付，前方および後方分岐の組み合わせに分けられる．無条件分岐は条件付分岐の一種と見なせるので，以下では条件付分岐の場合について述べる．

・条件付前方分岐　分岐条件を満たすPEは分岐先のアドレスで待機状態になる．CPは条件付前方分岐を常に不成立とみなしてコード列を継続して送り出す．もし前方分岐するPEが存在しない場合はコード列の送出を中断し，制御テーブルを参照することにより新たなコード列を送り出す．前方分岐が当てはまる制御はIF～THEN～ELSEやCASE等が該当する．

・条件付後方分岐　CPは常に分岐するものとしてコード列を送出する．分岐しないPEは分岐命令の次アドレスに於いて待機状態になる．前方分岐の場合と同様に分岐しないPEが存在しない場合にはコード列の送出を中断する．後方分岐は主にループ処理に該当する．

・サブルーチンコール　サブルーチンコールの場合は次の制約がある．実行の再開は命令アドレスをキーとしているため，同じアドレスをキーとした同期において，PEが異なるネストで待ち状態の場合に誤動作する．このため，再帰呼び出しを行う場合は変数の併用により正しく動作させることができる．なお，この方式ではPEに分岐命令が必要であるため，命令単位の実行制御には従来のフラグを用いた条件制御方式が有効であるが，それ以外の広範囲で複雑な条件がネストした制御には本方式が有効である．一方，本方式と同様な目的で実行状態を任意のレジスタに保存しておくことにより制御の柔軟性を向上させた例[9]が報告されているが，本研究では次に述べる目標アドレスレジスタのみにより，更にネストした条件の処理を可能としている点で異なる．

本方式では，分岐命令と命令アドレスをキーとした同期機構による実行制御を行っているため，通常のプロセッサと同様にネスト条件の深さに制約がなく，かつPE単位で制御できる点で従来のSIMD型並列計算機の制御方式と異なっている．

２.２ 支援機構

前述の機能を実現する支援機構について述べる．支援機構は３つの機能から構成されており，図２にその様子を示す．

・命令アドレスバス（IAB）　CPは現在の命令の次命令アドレスをこのバスに出力する．全PEは現在の命令フェッチと共にこのアドレスを取り込む．次命令アドレスを用いることにより，PE側では現命令アドレスの場合と比較して１命令速く実行判断できる．

・アクティブフラグ（AF）　PEの動作状態を示すフラグで全PEのアクティブフラグの論理和がCPに入力される．CPではこれを監視しており，このフラグが偽となった時点で，命令を実行するPEが無いことを検出し，制御テーブルから次の

命令コード列を求めて新たな命令を送出する．

・目標アドレスレジスタ（TAR）　各PEが持つ実行再開アドレス格納されているレジスタである．PEは待ち状態中は常に命令アドレスを監視しており，TARとIABが一致したとき，次の命令から実行を再開する．

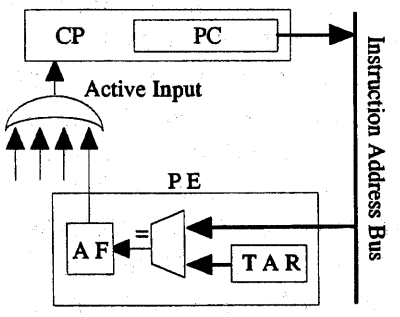


図２　支援機構の概要

表１にCPとPEがパイプライン化されていない場合の前方分岐と後方分岐における動作を示す．

表１　CP/PEの動作

	Condition	Forwad Jump	Backward Jump
CP	AF=1	PC:=PC+1	PC:=operand
	AF=0	PC:=operand	PC:=PC+1
PE	jump condition satisfied	TAR:=operand AF:=0	AF:=1
	jump condition unsatisfied	AF:=1	TAR:=PC+1 AF:=0

3.プロトタイプ

3.1 方針

プロトタイプでは次の方針で設計中である．

・モデル　プロトタイプではPE構造の簡潔性および拡張性の点から、PE間通信の簡素化と局所的な通信による並列性維持が可能な超並列モデル[4]によるPGAを取り扱う．このモデルでは各世代毎に隣接するPE間で情報交換する・優秀な遺伝子は数世代を経て伝搬する．原理的には個体数と同数のPEによる並列処理が可能である．

・個体数の上限　一般的に，１集団当たりの個体数を増加させても，交叉演算などの方法によっては優良個体の破壊につながり，遺伝アルゴリズムの性能が低下することが言われており，本研究では１集団当たりの個体数は数百程度，最大千個体程度とする．

・複数集団の並列処理　複数の集団を処理する並列GAを考慮し，複数の集団を並列可能とする．

・入出力　GAでは画像処理のような大量のデータを処理する用途は少ないと思われる．このため，性能のボトルネックになり易い入出力は余り重視しなくて良い．

・パイプライン処理による高速化　要素プロセッサ（PE）および制御プロセッサ（CP）をパイプライン処理により高速化する．また，CP-PE間を一つのパイプラインとして構成する．パイプライン化によりPEの実行状態の検出が遅れるため分岐距離が数命令と短い場合には２節で述べた制御方式よりも従来のフラグによる実行制御が有効である．ソフトウエア指向のアーキテクチャとする場合，両方式を併用する．

・ワードスライス指向のSIMD型並列計算機　遺伝操作は主としてビットパラレルに処理される頻度が高く，１集団当たりの個体数は数百個体程度のものが多い．このためプロトタイプでは，小数個体の場合にも高速に処理できるようにワードスライス指向とする．ただし，遺伝子長が長くなる場合は，複数PEへ問題分割を検討する．

・遺伝操作の専用化およびFPGAによる構成　遺伝操作の高速化のため，遺伝操作に関わる回路を専用回路化する．対象となる遺伝操作には，交叉，突然変異，評価等があり可能な限り専用回路化が望ましい．この部分はアプリケーション依存でありまたPE自身の汎用性向上のため，FPGAを使用する．

・命令の階層化　本アプリケーション依存であるが，ホストの負担軽減のため予め命令列をCPに送る方式およびホストから命令を逐次発効する方式とし，CPでは命令をマイクロ命令列にデコードし，PEを制御するものとする．

3.2　システム構成

設計中のプロトタイプシステム構成を図３に示す．システムは複数PEと１CPおよびネットワークから構成される計算ユニット（CU）を最小構成とする．１CUは８～３２PE（アプリケーショ

ン依存）で構成される．なお，CPおよびネットワークは現在設計中である．

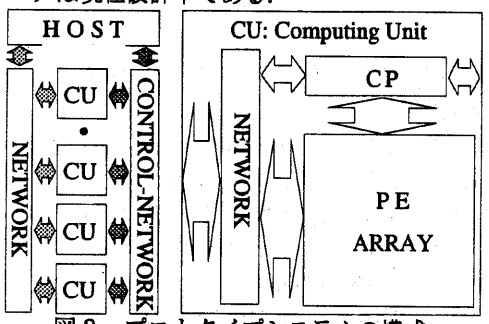


図３　プロトタイプシステムの構成

3.4　PEのアーキテクチャ

・ローカルメモリ　簡単な遺伝子表現または数値表現形式を用いる場合にはローカルメモリは不用であると考えられるが，複雑な表現形式の取り扱いや多数の遺伝子を処理する際のプロセッサの仮想化を行うために高速SRAMを備える．１FPGA当たり２個程度のSRAMを用意する．

・メモリアクセスのパイプライン化　PEのメモリアクセスをパイプライン化し，高速SRAMを用いることで連続したメモリアクセスを行う．FPGAへの実装が前提であるため，PEのクロック速度の上限は１０～２０MHz程度となり，高速SRAMで十分である．このためデータキャッシュが不用となりキャッシュのコヒーレンシに係わるアクセス遅延の変化の問題が発生しない[12].

4．実装

今回設計したPEでは基本的な構成による，ソフトウエア指向のPEとし，回路規模の見積もりのために以下の仕様とした．図４にPEのアーキテクチャを示す．

・３２ビット固定命令長
・１６ビット幅・６４KWのローカルメモリ
・４段パイプライン（F，D，E，W）
・GA専用演算器（交叉，突然変異，）
・特殊演算（ビット逆順序化等）
・通信回路（N，E，W，Sの４方向）

また，レジスタ構成は以下の通りである．

・８汎用レジスタ（内１つは０レジスタ）
・２交叉専用レジスタ
・乱数発生レジスタ，通信レジスタ
・フラグ，命令アドレスレジスタ等

・命令セット　PEの命令としてALU命令，GA専用命令（交叉，突然変異，評価等），分岐命令等を含む約３０種類を用意した．今回は，乗算除算回路およびGA専用命令である評価回路は含まない．また殆どの演算命令は３オペランド形式である．図５に命令形式を示す．先頭の４ビットは条件実行フラグであり，この組み合わせとPEの内部状態により命令実行が判定される．

・GA専用回路　今回は２点交叉回路と突然変異回路を実装した．これらの回路は１クロックで処理可能である．評価回路は今回実装せず，ALUを用いて処理する．

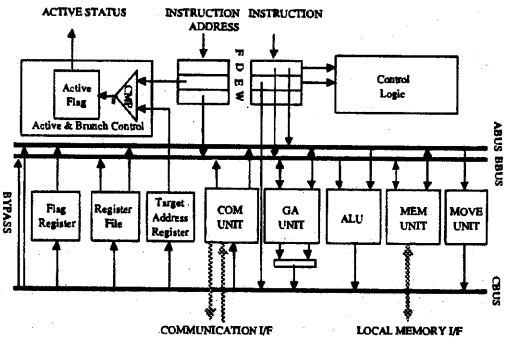
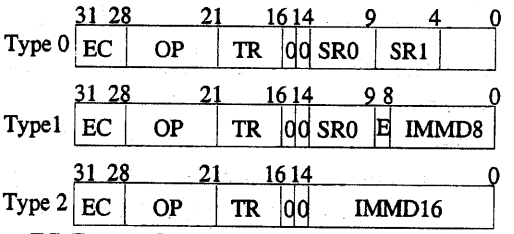


図４　PEのアーキテクチャ



EC: Execute Condition　　OP: Opecode
TR: Target Register　　SR: Source Register
IMMD8: 8bit Immidiate data
IMMD16: 16bit Immidiate data

図５　PEの命令形式

5．実装結果

今回，PEの回路規模の見積と動作評価のためにXILINX FPGA XC4013PQ240（13,000ゲート相当）に実装した．開発はXILINX社製VHDLによる開発システム（Windowsで動作）を使用した．PEは約2400行程度のVHDLで記述されている．論理合成，最適化の手順を経た後，バスやレジスタを手動配置し，自動配置配線処理を行った．表２

に示す内部リソースの使用率からわかるようにバスリソースをほとんど使用しているため，１６ビット幅のPEが限界であるが，機能ブロックには余裕が残されていることから，更に複雑なGA専用回路の実行が可能である．XC4020またはXC4025のデバイスを使用すれば，バスリソースを除いて半分程の回路規模となるため，メモリベースのレジスタを使用するなどの変更により１つのFPGAに２個の実装または３２ビットPEの実装が可能である．また，専用化による不用リソースの削除により少なくとも６４ビットのPEも実現可能である．シミュレーションによる上限速度は約８～１０MHz程度となった．これは論理合成や配置結果により左右されており，更に検討が必要である．

表１　内部リソースの利用率

リソース名	使用率
I/Oピン	174/192 91%
F & G Func. Generators	591/1152 51%
H Func. Generators	138/576 24%
CLB FF	422/1152 37%
3-State Bufers	804/1248 64%
3-State Half Longlines	80/96 83%

６．おわりに

本論文では，並列遺伝的アルゴリズムをSIMD型並列計算機で処理するため基本的なアーキテクチャとSPMD動作を実現する支援機構を提案した．この支援機構はPEのフラグによる条件実行制御と合わせて使用することにより従来のSIMD型並列計算機の制御をより高度にできるものである．また，遺伝処理従来はホストレベルで処理されていた逐次処理部分を各CPで処理することから，CP単位で異なる処理が可能である．

各種GAアプリケーションに対応するため，FPGAによりPEを構成した．今回提案したアーキテクチャはRISC的な４段パイプラインに遺伝操作専用演算器を付加したものであり，ソフトウエア指向のPEである．FPGAへの試験実装を行った結果，３２ビット程度のPEであれば実装可能であり，より特化したアーキテクチャでは６４ビット程度のPEも可能である．

今後の課題はCP，ネットワークおよびその他の詳細設計と支援ソフトウエアの開発がある．

参考文献

[1] J. H. Holland, Adaptation in Natural and Artificial Systems, Univ. of Michigan, 1975.

[2] D. E. Goldberg, GENETIC ALGORITHMS in Search, Optimization and Machine Learning, Addison-Wesley,1989.

[3] V. S. Gordon et. al., Dataflow parallelism in genetic algorithms, R. Manner and B. Manderic (Ed.), Elsevier Science Publishers B.V.,pp.533-542,1992.

[4] Ranjit A. Heny et. al., A Massively Parallel SIMD Algorithm for Combinatorial Optimization, Proc. of the International Conf. of Parallel Processing, 1996.

[5] 大島，松本，平木，再構成可能な遺伝的算法エンジンの研究，The Third Japanese FPGA/PLD Design Conf. & Exhibt, pp.541-548, 1995.

[6] S. D. Scott, A. Samal and S. Seth, HGA: A Hardware-Based Genetic Algorithm, Proc. 1995 ACM/SIGDA Third Int'l Symp. on FPGA, pp. 53-59,1995.

[7]吉田，森本，安岡：SFLによる遺伝的アルゴリズムVLSIの設計,第10回パルテノン研究会資料集, pp.63-70, 1997.

[8]T. Inoue, Y. Takahashi et. al., Design of a Processing Element of a SIMD Computer for Genetic Algorithms, Proc. of HPC Asia '97, Seoul, pp. 688-691, 1997.

[9]Y. Takahashi et. al.,A SIMD Machine for Massively Parallel Genetic Algorithms, Proc. of the 1997 International Conf. on Parallel and Distributed Processing Techniques and Applications, pp.915-923,1997.

[10] W. D. Hillis, The Connection Machine, The MIT Press, 1985.

[11]松田，湯浅：SIMD型超並列計算機SM-1（仮称）の概要，情報処理学会計算機アーキテクチャ研究会資料，92-ARC-95,pp. 127-134,1992.

[12] J. D. Allen, D. E. Schimmel, Issues in the Design of High Performance SIMD Architectures, IEEE Trans. on Parallel and Distribution Systtems, Vol. 7. NO. 8,pp. 818-829, 1996.