社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2003－ＳＬＤＭ－112　(24)
2003／11／28

ＦＰＧＡによる組込み制御を目的としたプロセッサシステムの改良

荒木　英夫†　久津輪　敏郎‡　原嶋　勝美‡

†大阪工業大学工学部　工学研究科　〒535-8585 大阪市旭区大宮5丁目16-1
‡大阪工業大学工学部　電子情報通信工学科 〒535-8585 大阪市旭区大宮5丁目16-1
E-mail: †araki@kesakoy.elc.oit.ac.jp, ‡{kutuwa, harashima}@elc.oit.ac.jp

あらまし　近年のLSI技術の進歩は，多くの機能を一つのLSIに集約するSoCを実用化している．さらに，少量多品種設計において，FPGA等の再構成可能な半導体を用いて，SoCの実現が行われている．LSI設計では，設計資産の再利用をするために様々な機能をIPとして用意されている．IPとしてMPUも提供されているが，カスタム化や修正を行うことが困難である．また，高機能なMPUはFPGAの資源の大半を消費してしまう．これを解決するために，HDLで記述されたMPUをFPGAに最適化・合成を行うことが必要である．著者らは，組込み用途に適した並列実行が可能なMPUを提案した．本論文では，この従来型MPUを改良した目的と手法について説明し，シミュレーション実験を行った結果を示す．

キーワード　FPGA, プロセッサ, 再構成可能, 組込み用途, SoC (System On a Chip)

A Improvement Method of a Processor for a Embedded System on an FPGA

Hideo ARAKI†　Toshiro KUTSUWA‡　and　Katsumi HARASHIMA‡

† Graduate School of Engineering, Osaka Institute of Technology　5-16-1 Omiya, Asahi-ku, Osaka, 535-8585 Japan
‡ Faculty of Engineering, Osaka Institute of Technology　5-16-1 Omiya, Asahi-ku, Osaka, 535-8585 Japan
E-mail: †araki@kesakoy.elc.oit.ac.jp, ‡{kutuwa, harashima}@elc.oit.ac.jp

Abstract　In recent year, LSI technologies are realized economical and high performance integrated circuit, and are realized a design method called "SoC" in same time. Furthermore, a lot of IP is built for designing LSI, and has a MPU. The most of MPUs on IP are distributed as synthesized objects. The MPU is hard to modify functions by user and require many resources on FPGA for high performance. Applying a MPU which is described by HDL is good method for FPGA design without these problems. We already showed MPU for embedded systems. This paper shows methods and simulation results of improvements.

Keyword　FPGA, Processor, Re-configurable, Embedded, SoC (System on a Chip)

1. はじめに

現在，生産される電子機器は，大きく大量生産品と多品種少量生産品に分けることができる．成熟した電子機器は大量生産による低価格化が行われており，発展途上の製品は様々な機能の付加や改良が行われ，新しい製品が絶え間なく発表されている．大量生産品は，製造コストを少なくするために部品点数を減らしている．LSI設計でもこれらの要求から，全ての機能を一つのカスタムLSIに実装するSoC（System on a Chip）設計が実現されている．一方，少量多品種設計においても，製品開発時間の短縮化や研究レベルから開発レベルへの技術移転の容易性などが求められている．これに対応するために，再構成可能なデバイスを用いた柔軟なハードウエアや，高位合成技術の研究・開発が活発に行われている．これらを実現することにより，ユーザサイドにおける，ハードウエアのバージョンアップ等が可能となり，製造メーカのコスト削減や新たなサービスの可能性，ユーザの購入時の安心感などを実現できると考えられる．

このように柔軟なハードウエアを実現するには，FPGA（Field Programmable Gate Array）等の再構成可能なデバイスの利用が一般的である．さらにIP（Intellectual Property）として提供されるMPU（Micro Processing Unit）と組み合わせると，より柔軟なハードウエアの実現が容易となる．しかし，これらのMPUの多くは必要以上に多機能なものが多く，組込み制御などの特定用途に適したMPUはまだ少ない．著者らはこれまでに，FPGA内部に基本処理装置

であるプロセッサエレメント（PE: Processing Element）を多重化した，組込み制御に適した MPU を発表した[1]．この MPU は組込み制御に適した特徴を持つが，PE の増加に伴い処理速度の低下が発生する．そこで，本稿では以前に発表した MPU の拡張に伴う問題点を明らかにし，その解決策について検討と評価を行い，その結果を示す．

2. 既存の研究と従来型 MPU

2.1. 既存の研究

これまでにも FPGA 内に構築する MPU はいくつか発表されている[2]～[5]．FPGA ベンダーである ALTERA 社は，合成可能なソフトウエアコア MPU である NIOS を提供しており[6]，その他にも ARM 社の ARM 7 と互換性のあるハードウエアコア MPU を実装した FPGA デバイスを提供している．他にも Xilinx 社は PicoBlaze といった合成可能な MPU と開発ツールを提供している[7]．これらの MPU は合成済みの IP として提供されることが一般的であり，ユーザ自身が自由に変更を行い合成することはできない．これらに対して，HDL で記述された MPU も公開されている．HDL により記述された MPU はユーザ自ら修正することが可能であるが，多くの MPU は既存の MPU と命令互換であることを特徴とし，この互換性を保ったまま，大きな変更を加えることが難しいといった問題がある．また，ターゲットデバイスを絞っていない記述であるため，FPGA に実装した場合，リソースの消費が大きくなることや，動作速度が目標に達しないなどの問題が発生する可能性がある．さらに，多くの MPU はパイプライン化されており，命令の拡張や修正を行った場合これらの制御系も修正する必要があり，ユーザによる修正が可能な範囲や規模は限られる．

2.2. 著者らが提案した従来型 MPU

これまでに著者らが発表した従来型 MPU[1]は，Verilog-HDL で記述されており，ALTERA 社の FPGA を有効活用することを前提としてデータパスの構造を決定し，また ALTERA 社が提供するライブラリをできるだけ用いて記述されている．このため，ALTERA 社により FPGA が改良された場合でも，多くはその FPGA の特徴を生かして再構成を行うことが可能であると考える．次に，提案する MPU の応用対象については，主にモータ制御などリアルタイム性が必要であるが複雑なアルゴリズムの実装を必要としないものとしている．これにより，MPU の演算機能を縮小し，単一 FPGA 内に複数のプログラムを同時に実行可能な MPU システムを構築した．

従来型 MPU の基本構成を図 1 に示す．PE はプロセッサエレメントであり，このモデルでは 10 個の PE が実装されている．PE-M はマスタ PE と呼ばれ，システム起動直後に動作を開始する．PE-S はスレーブ PE と呼ばれ，PE-M により動作の設定，起動，停止が制御される．PE はハーバード型アーキテクチャの基本構成を持ちプログラムバスとデータバスが分離されている．

MRCU（メモリリソースコントローラユニット）は PE とプログラムメモリ，データメモリの接続を行う．MRCU はメモリへのアクセスの調停をラウンドロビンにより行い，プライオリティは持たない．

PRCU（プロセッサリソースコントローラユニット）は全 PE のプログラムカウンタと割込みコントローラを持ち，PE-M からの制御により PE-S の動作アドレスや起動制御を行う．

PCU（プロセスコントローラユニット）は PE-S が実行するプログラムの実行状態を監視する．ＰＥはプログラムを実行中，プロセスＩＤを持ち，このＩＤによりアクセス可能なリソースが限定される．

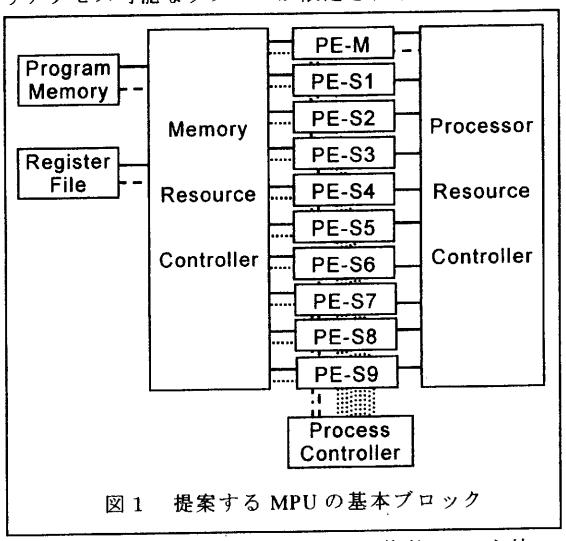

図 1　提案する MPU の基本ブロック

この MPU システムの特徴は内部に複数の PE を持つことである．PE は命令制御部と演算部からなり，それぞれの PE は独立した動作を可能としている．PE はプログラムメモリとレジスタファイルを共有する．動作のシーケンスを図 2 に示す．これらのことから，従来型 MPU は次のような特徴を持つ．

- プログラムメモリとレジスタファイルメモリを共有した複数の PE が，それぞれ独立したプログラムを実行可能．
- PE は独立しており，お互いに影響は及ぼさない．
- マスタ PE は，スレーブ PE の制御が可能．
- PE の数は FPGA のリソースが許す限り，上限はない．
- PE の増加は，1PE あたりのメモリバスの帯域を

狭めるため，PEの処理速度の低下をまねく．

本稿では，最後にあげた特徴（欠点）である，PEの増加に対する処理速度の低下に対する改良について取り上げる．従来型では，10個のPEを実装したMPUを30MHzのシステムクロックで動作させると，PE1つあたりの動作速度はクロック数をPEの数で割った値となり，1PEあたり3MHzとなる．このとき，処理の内容によりリアルタイム性が保証可能なPEと不可能なPEが現れた場合，従来型ではシステムの動作速度を速くする必要があった．しかし，一定の条件下において，システムの動作速度を変化させずに，必要とするPEの処理速度を向上させる方法を検討する．

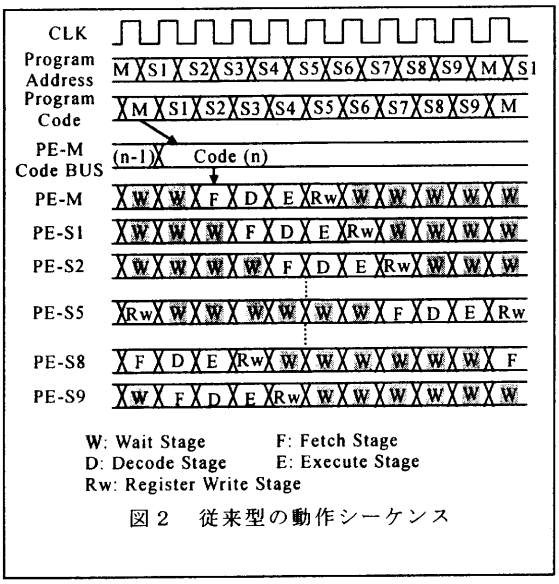

図2　従来型の動作シーケンス

3. 対象とするMPUの改良手法

3.1. 改良方法の検討

2.2.で述べた問題を回避するために，次のように改良を行った．主な改良を行う際に着目したことは，

(1) 各PEに対して均等であったメモリバス（コードメモリ，データメモリ）のアクセス割り当てを，マスタMPUにより不均等に設定可能とし，特定のPEについて，処理速度の向上を可能とする．

(2) フェッチ，デコード，実行，書込み，（ウエイト）と4つの基本ステージをそれぞれ1クロック以内で実行することを要求してきたが，実行ステージを複数クロックとして割り当てることを可能とし，これまではウエイトであったステージを有効に利用する．これにより，繰り返し演算や，処理速度が1クロック以内で実現不可能な演算を実装可能とする．

(3) 停止中のプロセッサへのメモリアクセス権限の割り当てを停止する．これにより，停止しているPEが多いと，動作中のPEに多くのメモリアクセス権が割り当てられる．

以上の3つの機能を実現した．この3つの機能は，これまでに提案してきた従来型MPUのアーキテクチャに大きな影響を及ぼす．従来型MPUは，いかに単純な制御により並列演算が可能なMPUを構築するかという点に重点をおいており，特にデータパスにおけるスループットを低下させないように配慮され設計されている．今回の改良は，データパスの切り替え制御回路が複雑になるため，データパスのセットアップ時間が短くなり，最高動作速度の低下が見込まれる．これらを踏まえ，各機能の実装方法について検討を行った．

3.2. 改良手法1

従来型MPUの特徴は，動作速度よりも全PE間の独立性にあり，これを実現するためにメモリへのアクセスを順序化し一切の例外を認めないこととした．これにより，単純な回路構成で効率のよいMPUの独立動作を可能としている．しかし，9個以上のMPUを実現すると，実行ステージよりもウエイトステージが多くなる．また，ただ一つのPEの処理速度が足りないために，システム全体を高速化させる必要があった．そこで，高速な処理を必要とするMPUのメモリへのアクセス権を2倍にすることにより，そのPEはそれまでに比較して動作速度が2倍になる．しかし，他のPEからは，PEが1つ増えたことと同等であり，処理速度に対して大きな影響を与えない．図3に改良手法1を適用した動作シーケンスを示す．

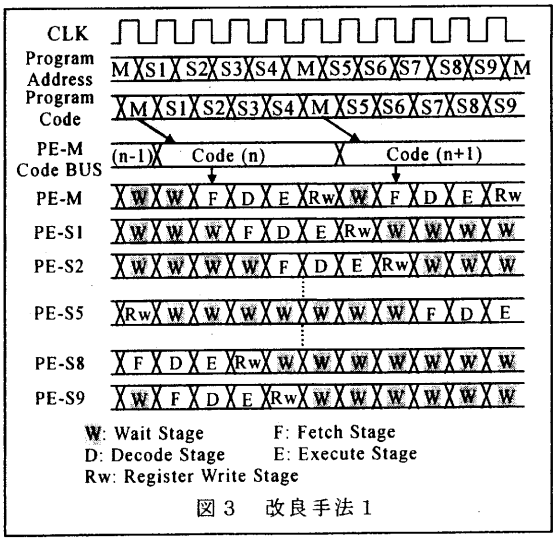

図3　改良手法1

ただし，本手法では PE 単体でのスループットは向上していない．このため，システムのステージ数が，PE 単体が必要とするステージ数の２倍以上となった場合に，本手法を用いた割り当てが可能となる．本手法の実装には，MRC の修正が必要となる．これまではメモリへのアクセスはラウンドロビンによる単純な割り当てであるが，高速化を行う PE は４システムクロック以上経過後に再割り当てを行うことにより実現した．

3.3. 改良手法2

次に示すのは，２クロック以上の動作が必要となる実行ユニットを実装した場合についての検討である．実行ユニットが複数クロックを用いて動作すると，問題になるのはレジスタファイルへの書込みに関する調停を乱すことである．そこで，複数クロックを用いた演算命令については，レジスタファイルへの書き戻しを許可しない．これにより，レジスタファイルの調停が乱されること無く，システムに大きな影響を及ぼさない．しかし，演算結果をレジスタに書き込む必要がある場合には，PE は１クロック以内で処理可能な演算に比較して，さらに１サイクルを必要とする．

提案手法２を適用した MPU の動作シーケンスを図４に示す．この例では，PE-S8 に８クロックまで要する演算ユニットを実装可能であることを示している．結果は W レジスタ（アキュームレータ）に保管されているため，条件分岐や次の演算に用いる場合は次のサイクルで結果を利用可能である．

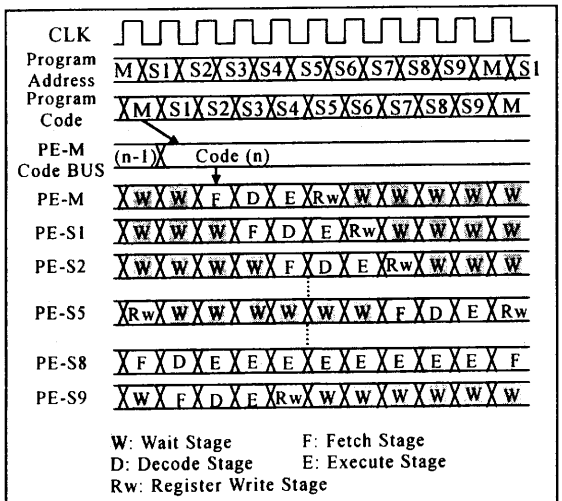

図４　改良手法２

3.4. 改良手法3

手法３は，利用されていない PE を停止させるものである．従来型では，動作の状態にかかわらず，メモリアクセス権限は自動的に与えられてきたが，ここでは動作していない PE に対しては，メモリの割り当てを停止する．この手法は，全ての PE が動作している場合，性能の改善は現れないが，一定期間必ず停止する PE が存在するような条件下ではメモリバスは停止している PE が占有しており，これを開放することにより動作中の PE の処理能力の向上を見込むことができる．ただし，この手法を用いると PE の処理速度が変動するため，時間を要素とする処理を行っている PE 間のハンドシェイクや待ち時間をソフトウエアにより制御している場合には問題が発生する可能性がある．この改良手法を実装するために，MRC の制御部を修正した．ここで，これまでラウンドロビンであったためシーケンサの動作からアドレスの選択・出力が，100MHz の動作で実現可能であったが，アドレスのセットアップタイム（1/2clk:5nSEC）を満たすことが困難となった．現在は，動作周波数を低下させた状態で評価中であるが，今後の拡張を踏まえ，MRC とメモリの間にラッチを挿入し，出力を全体に１クロックシフトすることにより，動作周波数の高速化を行う修正を検討中である．

提案手法３を提供した MPU の動作シーケンスを図５に示す．この例では，PE-S1 と PE-S5 が停止しており，動作中の PE のシーケンスが短くなっていることが確認できる．これにより，PE のスループットの向上を見込むことができる．

本手法は，動作する PE が４個になったときに，スループットが最大に達する．４個以下になった場合には，PE の処理が間に合わないため，MRC が待ち状態となり，PE の処理におけるスループットは向上しない．

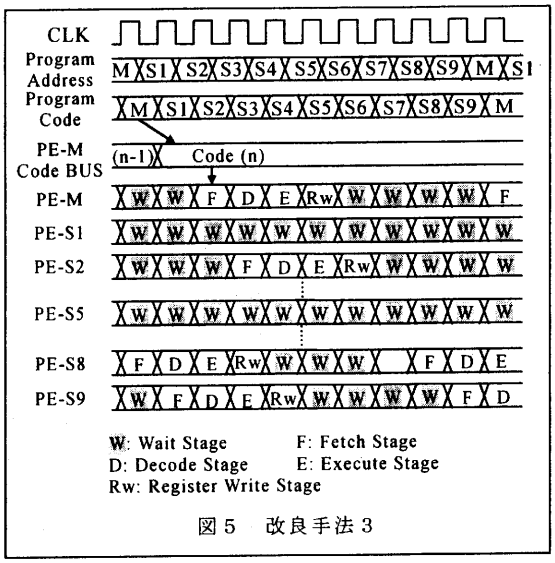

図５　改良手法３

4. 実験と考察

4.1. 実験方法

実験のために，それぞれの改良手法を組み込んだMPU を Verilog-HDL により記述し，ALTERA 社製 Quartus II2.2 を用いて論理合成とシミュレーションを行った．本実験ではターゲットデバイスとしてALTERA 社 APEX シリーズ EP20K300EQC240-1X を選択した．従来型のプロセッサにおいて，PE を 10 個実装した結果を表に示す．これまで提案した MPU はACEX1K シリーズを対象としており，APEX20K シリーズは ACEX シリーズに比較して大規模であるため，PE を 10 個実装しても全体の半分程度の消費であることがわかる．このモデルを基本に，改良手法の実装を行った．それぞれの手法を適用した状態の合成結果を表 1 に示す．

4.2. 改良手法1

改良手法 1 では，高速に動作させる PE を任意に選択できるレジスタを追加し，指定された PE はあらかじめスケジューリングされたシーケンスに組み込まれ，メモリアクセスを 2 倍にした．システムとしては高速動作の指定が行われた PE の数だけ，PE の数が増えたことと同等である．このため，ソフトウエアの作成やシステム設計時に，他の PE のスループットが低下することに注意が必要である．つまり多くの PE を高速動作指定すると，高速動作を指定したにもかかわらずあまり大きな速度の改善が見られないばかりではなく，高速動作指定されていない PE のスループットは大きく低下することになる．

改良手法 1 の合成結果を表 1 の 2 行目に示す．合成結果の比較から，改良のために Logic Cell と呼ばれる基本部品単位が 146 増加しており，最大動作クロックは 51.85MHz となり，速度が 96.8%に低下していることが判った．動作速度の低下は，シーケンサ部の遷移条件が増えたため，レジスタへの書き込みに遅延が発生したためである．

改良手法 2

改良手法 2 では，評価のために PID 制御専用演算器を作成し，既存の ALU に内蔵した．作成した PID 演算機のシグナルフローを図 6 に示す．今回の設計は評価用であるため，8bit 整数演算用となっている．PID 演算器のみの合成結果を表 2 に示す．次に，PE-S7，PE-S8 に PID 演算器を実装した ALU を載せ，合成を行った．合成結果を表 1 の 3 行目に示す．動作速度は，演算機の内部でクロック分割しているため，最大動作周波数の大きな低下は発生しなかった．実装に要した LogicCell 数について，PID 単体では 287LogicCell 必要としていたが，2 個の ALU に実装した場合 795LogicCell となった．これは，PID 演算器を実装すると命令制御部と ALU 内に専用の制御部が必要となりこれが約 110LogicCell を消費している．また，PID 演算器は内部に 10 個のレジスタと 1 個のカウンタ，1 個のフラグを持っているため，236 のレジスタを消費することを確認した．

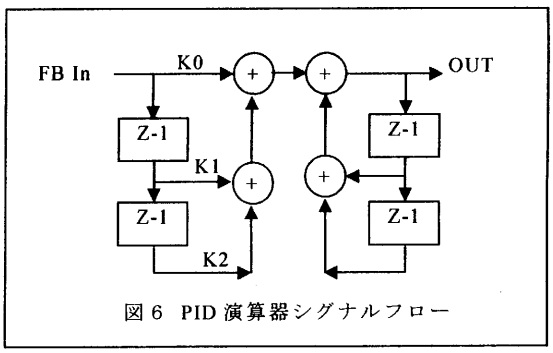

図 6 PID 演算器シグナルフロー

表 2 PID 演算器の合成結果

Logic cells	287 / 11,520 (2 %)
Registers	67 / 12,744 (< 1 %)
Clock	76.01 MHz (period = 13.156 ns)

表 1 合成結果の比較

	Logic cells	Registers	Clock	備考
従来型	5,445 / 11,520 (47 %)	1,571 / 12,744 (12 %)	53.57 MHz (period = 18.668 ns)	
改良手法 1	5,591 / 11,520 (48 %)	1,571 / 12,744 (12 %)	51.85 MHz (period = 19.288 ns)	
改良手法 2	6,240 / 11,520 (54 %)	1,807 / 12,744 (14 %)	53.04 MHz (period = 18.854 ns)	
改良手法 3	5,877 / 11,520 (47 %)	1,731 / 12,744 (12 %)	53.08 MHz (period = 20.752 ns)	
改良手法 4	6986 / 11,520 (53 %)	1,967 / 12,744 (13 %)	50.10 MHz (period = 19.960 ns)	

4.3. 改良手法3

改良手法3では，動作していない PE に対して，メモリアクセス権限を与えないようにシーケンサを再構築した．合成結果を表1の4行目に示す．

MRC への修正量は，改良手法1と大差ない.動作速度については，MRC 内のシーケンサは次の状態を決定するために参照が必要なレジスタの数が増加しており，シーケンサの動作速度の低下が現れたが，大きな速度低下は発生しなかった．レジスタの消費が増加しているのが確認できるが，これは停止している PE が増加した場合，PE へのメモリアクセスがシステムサイクルより短くなった場合，ダミーサイクルを挿入するために PE のステートを監視するためのレジスタを挿入したためである．

4.4. 改良手法4(1+2+3)

これまでの改良手法を全て組み込んだ MPU を作成した．これにより，これらの特徴を備えたプロセッサを合成することが可能となった．手法1と手法3は，それぞれ MRC（メモリリソースコントローラ）の拡張を必要とするため，手法1と手法3の実装をそのまま組み込むことは効率の悪化につながる．さらに，手法1と手法3を組み合せることにより，高速動作指定されたプロセッサが動作中の PE の減少によるメモリアクセスの割り当てが増加した場合，プロセッサ1サイクルが終了するまでにメモリアクセス許可が発行され，動作に乱れが発生するといった問題が新たに発生した．この問題を回避するため，このような条件下では，高速動作を無効とする条件を付加した．これら全ての機能を追加したシーケンス制御部を，新たに一体として設計を行った．制御部の複雑さにもかかわらず，ハードウエアリソースの消費は Logic Cell 数では 1.28 倍，レジスタ部では 1.25 倍に留まっていることが確認できた．これは，従来型のアーキテクチャが PE の増減や制御に柔軟に適合できる構造を持っているためこのような拡張を実装した場合に大きな影響が出ないためと考えられる．

4.5. 考察

これまでの実験結果から，提案した改良手法では一部の実装と全ての実装では大きな差が現れないことを確認した．これにより，全ての改良を施した MPU を用いることが適していると考えられる．また，従来型 MPU の「単純な共有シーケンスによる高速なメモリ共有型プロセッサ」という特徴を維持したまま PE を不均等に動作させるシーケンサの実現を行ったが，具体的なアプリケーションの実装と PE の割り当てについての研究を行い，各 PE の動作バランスについて検討を行うことが必要であると考える．

5. おわりに

本稿では，組込み制御に適したプロセッサについて，PE を拡張した場合における問題点について考察し，解決手法の提案を行った．そして 10 個の PE を実装した MPU を設計し，提案手法の適用を行った．最後に，全ての改良手法を行った場合のプロセッサの大きさと動作速度について評価を行った．

今後の研究として，改良型プロセッサを現実的なアプリケーションに適用した条件における処理速度と実装の容易性について検証を行う．

文　献

[1] 荒木英夫，　久津輪敏郎，原嶋勝美，“FPGA による組込み制御を目的としたプロセッサシステムの実装と評価，” 信学論 (C)，vol.J86-C，no.8，pp.799-807，Aug.2003.

[2] 安河内真由美，下尾浩正，山脇彰，岩根雅彦，“1 チップ再構成可能コンピューティングシステムの開発”，情処研報，ARC147-16，pp.91-96，Mar. 2002.

[3] 槻岡秀朗，籠屋健，笹尾和宏，高橋雅哉，中村維男，“SOUND コンピュータの試作”，情処研報，ARC124-8，pp.43-48，June. 1997.

[4] 中垣憲一，井上弘士，久我守弘，末吉敏則，“上級コース向き教育用プロセッサ DLX-FPGA の設計と実装”，信学技報，CPSY94-57，pp.17-24，Sept. 1994.

[5] 身次茂，“ロータリー型コンピュータ”，情処研報，ARC98-19，pp.147-154，Jan. 1993.

[6] ALTERA，“Nios soft core embedded processor data sheet ver.1，” ALTERA，June 2000.

[7] Xilinx，“PicoBlaze 8-bit microcontroller for virtex devices ver.1.2，” Xilinx，April 2002.