DAP/DNAを用いたSysteMorphプロトタイピング

吉田, 真
福岡県産業・科学技術振興財団 福岡知的クラスター研究所

曽我, 武史
福岡県産業・科学技術振興財団 福岡知的クラスター研究所

村上, 和彰
九州大学大学院 システム情報科学研究院

林田, 隆則
九州大学大学院 システム情報科学府

出版情報：電子情報通信学会技術研究報告, 第1回リコンフィギュラブルシステム研究会論文集, pp.221-226, 2003-09. 電子情報通信学会第1回リコンフィギュラブルシステム研究会
バージョン：
権利関係：

DAP/DNA を用いた SysteMorph プロトタイピング

吉田　真[†]　曽我　武史[†]　村上 和彰[††]　林田　隆則[†††]

†福岡県産業・科学技術振興財団　福岡知的クラスター研究所
〒816-8580 福岡県春日市春日公園 6-1
九州大学先端科学技術共同研究センターFS501
†† 九州大学大学院　システム情報科学研究院
〒816-8580 福岡県春日市春日公園 6-1
††† 九州大学大学院　システム情報科学府
〒816-8580 福岡県春日市春日公園 6-1
E-mail:　ngarch@fleets.jp

あらまし　近年，再構成可能なシステムを用いた動的最適化技術に注目が集まっている．我々は，九州大学にて研究されている SysteMoprh という適応型動的最適化技術の概念を用い，その応用システムの研究を行っている．本論文では，SysteMorph 概念を検証する IPFlex 社 DAP/DNA プロトタイピングの実装方法について述べる．
また，我々が提案するオンライン・プロファイリング手法と，DAPDNA オンライン合成実装方法について紹介する．
キーワード　動的最適化，再構成可能，オンライン・プロファイリング，オンライン合成

SysteMorph Prototyping on DAP/DNA

Makoto YOSHIDA[†]　Takeshi SOGA[†]　Kazuaki MURAKAMI[††] and　Takanori HAYASHIDA[††]

† Fukuoka Industry, Science & Technology Foundation
FLEETS (Fukuoka Laboratory for Emerging & Enabling Technology of SOC)
FS501,KASTEC 6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580, Japan
†† Department for Informatics, Graduate School of Information Science and Electrical Engineering
Kyushu University
6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580, Japan

E-mail:　ngarch@fleets.jp

Abstract　Dynamic optimization for software and hardware is gaining the attention of computer system researchers. We are researching application system for SysteMoprh technology.　SysteMoprh is Dynamic Optimization technology in Kyushu University. The Paper discusses SysteMoprh prototyping on DAP/DNA for the evaluation of SysteMoprh concept.
Keyword　dynamic optimization, reconfigurable computing, online profiling, online synthesis

1. はじめに

近年，コンピュータシステムを構成するハードウェアやソフトウェアを動的に最適化する技術が注目されている．動的な最適化により，システム運用中のシステムの実態に応じたシステム特性（性能，消費エネルギー）の最適化が可能になる．さらに，システム開発時の最適化作業を簡略に行い，システムを使用しながら徐々に最適化していくことで，システムの開発期間やコストの低減も可能になる．

近年の半導体技術の進歩による回路規模の増大やシステムの複雑化により，特にシステム LSI などの組み込みコンピュータシステムでは，開発コストや期間の短縮と同時に，最適化による高性能化や低消費エネルギー動作も重要視されるようになってきた．

動的最適化には，システムを運用することで初めて得られる情報を用いてシステムを最適化できる為，入力に応じて最適なシステム形態をとるようなシステムの実現が可能である．例えば，携帯端末などの組み込みシステムでは，製品出荷後にユーザーの利用状況に応じた動作性能や消費電力の動的最適化が考えられる．

しかしながら，動的最適化には克服すべき課題も少なくない．システム運用中に最適化を行うには，出来る限りそのシステムの本来の動作に影響を与えないように最適化を実現しなければならない．また，システムの動作中に最適化が可能なポイントを特定し，最適化方法を決定しなければならない．さらに，最適なシステムへ再構成する機構も予め組み込まれている必要がある．このような動的最適化可能なシステムについて， 我々は SysteMorph[1][2][3]を適用し，その応用システムの実装と評価を行う．

近年，注目されているリコンフィギュラブル・コンピューティングにおいて，FPGA(Field Programmable Gate Array)や PLD(Programmable Logic Device)等の再構成可能ロジックを用いて，アルゴリズムの全体または一部を論理回路として実装することで ASIC の長所である高速性を実現できる．さらに，再構成可能ロジックは回路構成を変更することができるので，ソフトウェア実装の特徴である柔軟性をも実現することが可能となる．我々は，プロセッサおよび再構成可能ロジックを搭載したシステムにおいて SysteMorph を実現する技術研究に取り組んでいる．

本論文では，IPflex 社の DAP/DNA[4]における SysteMorph プロトタイピング実装方法について紹介する．DAP/DNA は DAP(Digital Application Processor)と呼ばれる RISC プロセッサと，DNA(Distributed Network Architecture)と呼ばれる動的に再構成可能な演算ユニットがマトリクス状に配置されたアクセラレータから構成されるダイナミック・リコンフィギュラブル・プロセッサである．DAP で実行されるアプリケーションのオンライン・プロファイル情報から高頻度で実行される命令パスを検出し，オンラインで DNA 構成情報を生成する．高頻度で実行される命令パスの論理を，高速動作可能な専用回路へ動的に再構成できる DNA で実行する．これによりシステム全体の性能を動的に最適化する SysteMorph の検証を行っている．

本論文では，IPflex 社 DAP/DNA ダイナミック・リコンフィギュラブル・プロセッサにおいて SysteMorph のオンライン・プロファイラ技術と動的最適化技術を適用した．4 章でオンライン・プロファイラ実装方法について述べ，5 章でオンライン・ハードウェア構成情報生成（DNA ハードウェア構成情報）について述べる．

また，今回実装したオンライン・プロファイラ手法について NET 法[5]と比較して紹介する．

2. SysteMorph 技術について

SysteMorph とは，主にプロセッサ・ベース・システムにおいて，対象とするアプリケーション・プログラムのプロファイル情報に基づいて，当該アプリケーションを実行中に

・ソフトウェア(当該アプリケーション・プログラムのオブジェクト・コード)

・命令セット・アーキテクチャ（当該アプリケーション・プログラムを実行しているプロセッサの命令セット・アーキテクチャ）

・ハードウェア当該アプリケーション・プログラムを実行しているプロセッサおよび周辺ハードウェア）

のモードならびに構成を変更(morph)して，その性能，消費電力，消費エネルギーを最適化する，すなわち Feedback-Directed Dynamic Software / Instruction Set Architecture / Hardware Co-optimization 技術である．（図 1 参照）

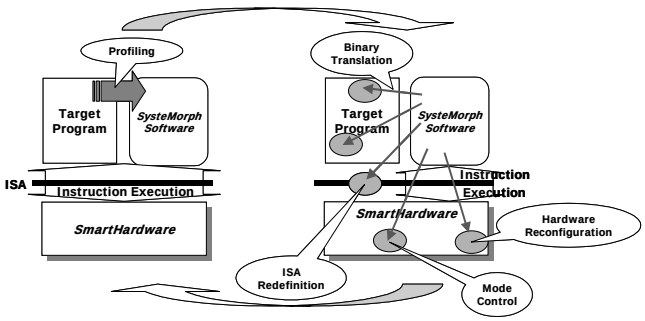

図 1 SysteMorph の概念

本技術により，システム LSI の設計期間の短縮化，システム LSI の寿命の長期化，ならびに，システム LSI

運用時の性能，消費電力，消費エネルギーの最適化を狙うことが可能になる．

SysteMorph はリコンフィギュラブル・コンピューティングとは異なり，ランタイムにハードウェアおよびソフトウェアを動的に最適化する．

3. DAP/DNA の構成について

SysteMorph プロトタイピングのプラットフォームに用いた IPflex 社の再構成可能コプロセッサ・システム DAP/DNA(Digital Application Processor/ Distributed Network Architecture)について述べる．

DAP/DNA は IPflex 社独自プロセッサコアを含んだ DAP と再構成可能ロジックの DNA に分かれている．全体の概略ブロック図を図 2 に示す．

DAP の構成は次の通りである．

- 32 ビット RISC プロセッサ(レジスタ，メモリ，I/O バス幅が 32bit)
  DNA 処理命令やコンフィギュレーション情報を変更するための命令が搭載されている．
- 命令キャッシュ８K バイト
- データキャッシュ８K バイト
- 可変長命令フォーマット(1 ハーフワード 16bit，1～3 ハーフワード構成)
- 1 命令 1 クロック，動作周波数 100MHz
- アドレス空間４G バイト

DNA の構成は次の通りであり，３つのセグメントに分けられている．

- エレメント
  演算や遅延などを行う基本要素 144 個が配置されている(表 1 参照)．
- コンフィギュレーション RAM
  DNA３面分のハードウェア構成情報を保持可能であり， DAP により制御される．
- 入力バッファ， 出力バッファ
  DAP とのデータ交換用バッファを 4 系統もっている．
- 内部バス
- 動作周波数 100MHz

DNA マトリクスは演算器同士を接続してパイプライン構成としたデータパスを形成したり，演算器の間の配線や演算内容を変更して様々な論理回路を実現することが出来る．

表 1 エレメント表

エレメント名	処理	個数
LD	データ入力	4
ST	算術演算 (乗算なし) とデータ出力	4
LDA/STA	DNAバッファとマトリクス間のデータ転送用アドレス生成	4/4
BLA/BSA	外部メモリとDNAバッファ間のデータ転送用アドレス生成	4/4
RAM	データ保持, 参照用メモリ (32bit × 2Kword)	6
SMA	算術演算 (乗算なし)	62
AMAM	算術演算 (16bit × 16bit 乗算あり)	12
DEL	データフローの遅延制御	44

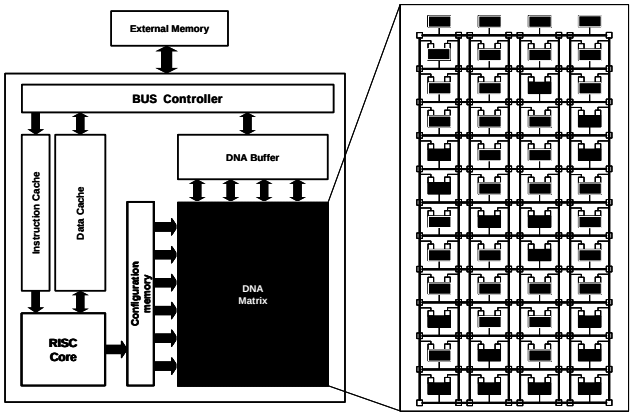

図 2　DAP/DNA 概略ブロック図

4. オンラインプロファイリング

オンライン・プロファイリングは，アプリケーションの振舞いに関する情報をその実行中に収集する技術である．オンライン・プロファイリングに対して，アプリケーション全体の振舞いに関する情報を収集する手法がオフライン・プロファイリングと呼ばれる．

オンライン・プロファイリングはオフライン・プロファイリングとは異なり，プログラムの実行途中のプロファイル結果に基づき，当該プログラムの残りの実行に対する「最適化のヒント」を導き出す必要がある．オンライン・プロファイラはその性質上， 次のような要求を満たさなければならない．

・オーバヘッドを削減する
・コストを削減する
・振舞いを正確に予測する

オンライン・プロファイラは，アプリケーションの実行時にアプリケーション本来の動作と並行に動作する．しかし，オンライン・プロファイラはアプリケーションの振舞いを監視するものであり，アプリケーションの実行そのものに直接影響を与えるものではない．したがって，オンライン・プロファイラを導入したことによるアプリケーション本来の動作の性能低下をできるだけ少なくする必要がある．また，オンライン・プロファイラはシステムに組み込まれるため，オンライン・プロファイラにかかるソフトウェア的なコスト（プログラムサイズの増大など）， ハードウェア的な

コスト（プロファイラのための回路部分の面積など），消費電力面のコスト（ソフトウェアやハードウェアの追加による消費電力の増大）などをできるだけ少なくすることが要求される．

4.1 オンラインプロファイラ構成例

プロファイラ実行のオーバーヘッドやコストを下げるため，図３のような構成を考える．

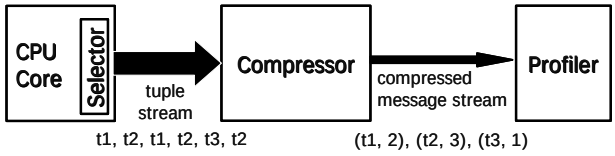

図３　オンライン・プロファイラの概念図

図３のオンライン・プロファイラは，セレクタ，コンプレッサ，プロファイラから構成される．CPU コアで実行される機械命令のうち分岐命令を選択するセレクタから，分岐アドレス(Branch Instruction Address : BIA)および分岐先アドレス(Branch Target Address : BTA)を一組とする tuple stream を出力させる．次に，コンプレッサにより tupple stream を圧縮し，BIA, BTA およびその分岐命令の実行回数を一組とする圧縮された message stream をプロファイラへ送る．最後に，プロファイラに蓄積した message から高頻度に実行される命令パス(Hot Path)を推定する．

4.2　DAP へのオンライン・プロファイラ実装

DAP へのオンライン・プロファイラ実装方法について述べる．図３のオンライン・プロファイラ構成のうちプロファイラ実行のオーバヘッドを考慮すると，全てまたは一部をハードウェア化することが考えられる．しかし，DAP は回路を追加変更できない為，全てソフトウェア処理にて実行する．

DAP にはトレース割込み実行モードが搭載されており，分岐命令を実行する度に分岐トレース割込みハンドラを実行することが出来る．また，分岐トレース割込みハンドラ実行終了後は通常処理に戻り，アプリケーション実行を続行可能である．分岐トレース割込みハンドラにて BIA と BTA を収集し，セレクタとコンプレッサの機能を実現する．また DAP/DNA 付属のμITRON4.0 ベース TalOS 機能を利用し，実行アプリケーションと別タスクでプロファイラを実行する．

4.3　オンライン・プロファイラのアルゴリズム

オンライン・プロファイリングに関して，近年さまざまな研究がされている．Evelyn ら[5]は，実行中のアプリケーション中のパス，すなわち連続して実行される基本ブロックの集合について，頻繁に実行されるパスをアプリケーションの実行時に推定する手法として NET（Next Execution Tail）法を提案している．

我々の提案するオンライン・プロファイリング手法は NET 法に対して，Hot Path 推定手法が異なる．図 4 のような基本命令ブロックおよび基本命令ブロック最終の分岐命令のパス構造であるとき，高い頻度で実行されるループを構成するパスを Hot Path として検出する．プロセッサコアで実行される分岐命令に着目し，BIA と BTA の組の実行回数（図４のある基本ブロックから他の基本ブロックへの矢印の通過回数）の履歴を収集し，解析することにより Hot Path を検出する．

4.4. 処理の流れ

次に処理の流れについて示す．

(1) CPU コアで実行された BIA, BTA を収集する．
(2) (1)で得られた分岐情報を基に図 5 の分岐履歴テーブルを更新する．
(3) (1)で BIA ＜ BTA である，すなわち戻り分岐であるエントリを選択し，戻り分岐履歴テーブルを更新する．
(4) 戻り分岐履歴テーブルにおいて，分岐実行回数が設定閾値より大きいエントリの BTA を Hot Path Head アドレスとする．

次に，Hot Path Head アドレスから Hot Path を検出するが，その手法は次の 2 通りある．

① 分岐履歴テーブルを参照しながら，命令パスのツリーを移動してオブジェクト・コードの分岐命令を直接調べる方法
② 拡張した分岐履歴テーブルをのみを用い，オブジェクト・コードを調べずに Hot Path を検出する方法

まず，①の方法について示す．

(5) Hot Path Head アドレスから順に分岐命令が見つかるまでオブジェクト・コードを調べる．
(6) 見つかった BIA のエントリが分岐履歴テーブルにないか調べて，分岐回数の大きい BTA のパスを Hot Path アドレスとする．
(7) Hot Path アドレスから順に分岐命令が見つかるまでオブジェクト・コードを調べる．
(8) みつかった分岐命令アドレスが Hot Path Head アドレスの場合は終了する．そうでない場合は戻り分岐検出まで(6),(7)の処理を行う．

次に②の方法について示す.

処理(2)の分岐履歴テーブルについて,

基本ブロック開始アドレス(Basic block Start Address : BSA)の情報を各エントリに追加した拡張分岐履歴テーブルを更新する. BSA とは, 各基本ブロックの開始アドレスである. BSA は前回の分岐命令の BTA であることから, BTA を 1 個保存するバッファを追加するのみで実現できる.

(3)から(4)の処理は同様である.

②の方法における処理(5)以降について示す.

(5)' 拡張分岐履歴テーブルにおいて Hot Path Head アドレスと BSA が同じエントリを検索する.

(6)' (5)'で検索されたエントリのうち, 実行回数の大きい BTA を Hot Path アドレスとする.

(7)' (6)'の Hot Path アドレスと BSA が同じエントリを拡張分岐履歴テーブルから検索する.

(8)' Hot Path アドレスが Hot Path Head アドレスの場合は終了する. そうでない場合は戻り分岐検出まで(6)',(7)'の処理を行う.

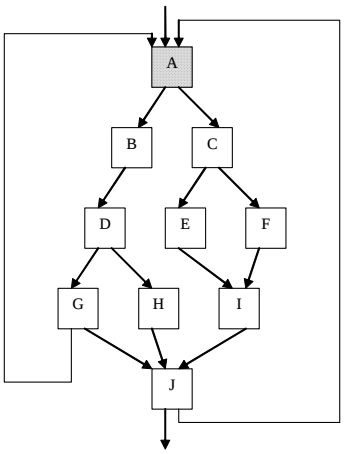

図４　対象プロファイリングパス構造

table memory address

	BIA	BTA	Count
0			
・・・	・・・	・・・	・・・

図５　分岐履歴テーブル

table memory address

	BTA	Count
0		
・・・	・・・	・・・

図６　戻り分岐履歴テーブル

table memory address

	BSA	BIA	BTA	Count
0				
・・・	・・・	・・・	・・・	・・・

図７　拡張分岐履歴テーブル

以上のようにして Hot Path を検出するが, ここで NET 法との違いについて述べる.

NET 法においても, 分岐命令履歴から Hot Path Head を検出するが, NET 法では Hot Path Head が決定されて最初に実行されたパスを Hot Path と決定する. しかし, Hot Path Head が決定された時に必ずしも正しい Hot Path を実行するとは限らない. これに対し, 提案手法は Path 上の全ての分岐命令の履歴から実行頻度の高いパスを Hot Path とすることから, Hot Path Head が決定されるタイミングに依存せず, より正確な Hot Path 検出を行うことができる.

5. オンライン合成

オンライン・プロファイラにより得られた Hot Path 命令列から, DAP/DNA の DNA 構成情報を生成する. DNA のハードウェア構成手法としてヒューリスティック・アルゴリズム[6]が提案されているが, 本論文ではオンラインでの実装方法を述べる.

まず, DNA 処理のエレメントや配線等の制約条件を満たす前段階として, DAP 機械命令を解析する次の処理は次の 2 点である.

・命令コード間の依存関係を調べる.
（依存性がない同一レジスタを異なるレジスタに分離する. また, loop に入った 1 回目には依存関係は無いが, loop した後は依存するレジスタをチェックする.）

・コードの最適化を行う.
（同一機能を一つに纏め, 並列性を検出して DNA の並列処理を可能にする.）

次に, DNA エレメントの配置を次の処理の流れで行う.

(1) 定数だけの論理やソースが 1 個だけの SEL 論理を削除し, エレメントに割り当てが必要な回路を抜き出す.

(2) (1)で抜き出した論理からコントロール・フロー・グラフを作成する.

(3) ランクの低いノードからエレメントを配置する. 配置が出来なかったノード以降は, 別のコンフィグレーション・メモリ・バンクに設定する.

ここで, ランクとはデータ・フロー・グラフの何段目かの段数を示す. データ・フロー・グラフの k 段目をランク k（k=1,2,・・・,N）のように定義する. ただし, データ・フロー・グラフの段数を N とする.

(4) エレメント間を配線する.

配線できない場合は

① 空いていてかつ接続可能なエレメントを検索する.

② 空きが無ければ, よりランクが高いノードを配置したエレメントで, 接続可能なエレメントと配置場所を交換する.

配置配線ができなかったノードがある場合は, そのランク以降のノードは別のコンフィギュレーション・メモリ・バンクに設定する.

最後に, DNA 配置後の後処理を行う. 次のように, DAP で実行されていた Hot Path の命令列を DNA で実行する為の DAP 命令を生成する.

(1) DAP レジスタ退避を行うための命令列を生成する.

(2) Hot Path を配置配線した情報を DNA コンフィギュレーション・メモリ・フォーマットに変換し, コンフィギュレーション･メモリへのライト命令列を生成する.

(3) DAP から DNA を起動する命令列を生成する. 複数のバンクを使用する場合, バンク切り替えの命令列も生成する.

(4) DNA の処理終了割込みを確認する命令列を生成する.

(5) (3),(4)で生成された命令列を Hot Path とバイナリ書き換えを行う.

(6) Loop Out 時のレジスタ状態を DAP へ取り戻すための処理を行う.

6. おわりに

本論文では, IPflex 社の DAP/DNA を用いた SysteMorph プロトタイピング実装方法について紹介した. また, 我々のオンライン・プロファイラの手法についても提案し, 評価を行っている.

今後の課題として, 様々なアプリケーション・プログラムの実装を行い, 動的最適化した結果をフィードバックし, 動的最適化を繰り返す SysteMorph の更なる検証を考えている.

文　献

[1] 林田隆則, 数勇介, 村上和彰, “SysteMorph：Functionality Morphing”, The 4th Summer Workshop on Embedded System Technologies (SWEST4), 2002.

[2] 林田隆則, 村上和彰, “「プログラム実行の局所性」の活用法に関する検討”, pp.65-68, Technical Report of IEICE, August 2000.

[3] Takanori Hayashida, Kazuaki Murakami, “Evaluating Online Hot Instruction Sequence Profilers for Dynamically Reconfigurable Functional Units”, pp.901-908, IEICE TRANS. INF. & SYST., VOL. E86-D, NO.5, MAY 2003.

[4] IPflex 社 DAP ユーザーズマニュアル, DAP 命令セットマニュアル, DNA Specification, ホームページ http://www.ipflex.com/

[5] E.Duestenwald and V.Bala, “Software Profiling for Hot Path Prediction: Less is More”, Proc. of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), pp.202-211, Nov.,2000.

[6] 数勇介, “動的システム最適化を対象としたハードウェア構成情報のオンライン生成について”, 九州大学大学院システム情報科学府修士学位論文, Feburary 2003.