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

2006－ARC－169 (25)
2006／8／2

自己タイミング型パイプラインシステムのオンチップ・マクロシミュレーション手法

三宮秀次[†] 大森洋一[††]
酒居敬一[†] 岩田誠[†]

回路集積技術の進展がもたらす大規模な計算資源を活用すれば，アプリケーションに特化して問題の並列性を最大限に活用するSoC構成が可能になると考えられる．ところが，初期の設計段階からのアーキテクチャの最適化に不可欠な性能評価において，ソフトウェアによるオフラインのシミュレーションは多大な時間を必要とし，また，既存のハードウェア構成を用いたエミュレーションでは柔軟な評価が困難であった．本稿では，新規アーキテクチャをチップ上で直接模擬して迅速なプロトタイプを実現できる，オンチップ・シミュレーション手法を提案し，自己タイミング型パイプラインによる回路構成法を提案する．

An On-Chip Macro-Simulation Mechanism of Self-Timed Pipelined Systems

Shuji SANNOMIYA,[†] Yoichi OMORI,[††] Keiichi SAKAI[†]
and Makoto IWATA[†]

With the future ULSI technology, application specific SoC architecture with dedicated processor cores and hardware modules exploiting the parallelism inherent in the applications will be more promising. The architectural optimization of them requires more precise performance estimation even in the early design phase. However, software-based offline simulation does not achieve fast evaluation while hardware-based emulation is defective in flexibility. In this report, an on-chip simulation mechanism which realizes rapid prototyping by directly simulating target architecture on the LSI chip is proposed, and its circuit module structure is presented by using self-timed pipeline.

1. はじめに

今後の回路集積技術の進展がもたらすチップ上の大規模な計算資源の利用を考えると，プロセッサ等の計算要素をアプリケーション毎に特化させることで，問題の持つ並列性を最大限に引き出すシステム構成が有効である．このような，応用分野に特化したシステムの設計では，システム全体の定量評価に基づいて，多数の組合せから最適なアーキテクチャを選択する必要がある．しかしながら，システム構成やプログラムを具体的に反映する必要からシステム全体の性能見積りが設計のボトルネックとなり得る．本稿では，システム全域をチップ上で直接的に模擬することで，迅速なプロトタイプを実現できる，オンチップ・シミュレーション手法を提案し，自己タイミング型パイプライン（STP）を用いた実現方法を示す．

アプリケーションに特化したシステムの設計では，初期の設計段階から，異なる命令セットを持つプロセッサや特定の処理に特化した専用命令／エンジンを含むマルチコアを搭載したSoC構成を想定する必要がある．この際，新規のSoC構成を容易にシミュレーションできる仕組みをチップ上に搭載しておけば，応用分野の専門家は，ハードウェア実装を待たずに，直接的にSoC構成を最適化できるため，システム開発期間の大幅な短縮が期待できる．

クロック同期型パイプライン構成を基礎としたノイマン型プロセッサでは，複雑なパイプライン構成に起因して，機能追加・変更が困難である．また，マルチコア化した場合には，プロセッサ間同期機構等も考慮に入れる必要があり，各機能要素の正確なクロックサイクルを初期の設計段階から想定することが困難である．

これに対して，STPを基礎としたDDPでは，局所的なタイミング調整のみで機能追加・変更が可能である．また，STPによるプロセッサ間ルータを活用すれば，素直にマルチコア化できる[1]．しかし，従来は，STPのハイドシェイクを逐一模擬しないと正確なデータフローが把握できないため，大規模なSoC構成のシミュレーションが困難であった．

本研究では，STP中の個々のパケットの振舞いをマクロに捉えて，平均的なデータ流として近似することによって，初期の設計段階であっても，オンチップ上で簡易にシミュレーションするための基本機構を提案する．

2. シミュレーションの要件

システム設計の手戻りを局所化かつ最小化するには，初期の設計段階で，応用システムの性能要求を満たせるよう

† 高知工科大学
Kochi University of Technology
†† 九州大学
Kyushu University

に，定量評価に基き，ソフトウェアで実装する機能とハードウェアで実装する機能を切り分ける必要がある．この際，要求性能が満足できなければ，複合命令セットや専用ハードウェアエンジンの導入や，プロセッサコアを増設した新規アーキテクチャの設計も必要になる．そのために，新規アーキテクチャ上で実行される応用プログラムの定量評価を，なるべく高速で精密に行うことが重要である．

ソフトウェアシミュレーションでは，コア数に応じて評価時間が長くなる．これに対して，再構成可能なデバイスを用いて比較的柔軟に高速なエミュレーションを行う手法[2]なども提案されている．しかし，一般的なエミュレーションの場合，ハードウェア実装が完了しないとシステム全体の評価ができないため，初期の設計段階での定量評価には適さない．一方，システムの全域にわたる性能評価では，各コアやエンジン毎の振舞いを並行して模擬する必要があるため，並列性を利用した自然なシミュレーションの高速化が求められる．これには，コア単位で振舞いを模擬できる，簡易な動作モデルも重要となる．

自己タイミング型パイプライン STP によるデータ駆動型マルチプロセッサ DDMP チップを活用した応用システムの開発では，通常，データフローグラフ（DFG）を用いて応用システムの機能仕様を定義する手法が有効である．これは，データ駆動プロセッサでは，割込み制御やスケジューリング等，本来の処理とは無関係の付加的なコードを必要としないため，DFG を階層的に精緻化すれば，機能仕様から実行コードまでシームレスに詳細設計が可能になるためである．

データ駆動原理では，データの到着順序に依らずに，演算結果を保証するため，DDMP システムの機能は，性能とは独立に評価できる．これを利用して，オンチップ・シミュレーションでは，命令の論理的な動作と実行タイミングを独立して評価する．

機能検証

対象の DFG を，機能的に等価な既存の命令セットで置換することで，直接的に実行する．システム機能は，ほぼ実行時間内で評価できるため，オフラインのシミュレーションに比べて，遥かに高速である．このとき，性能を左右する分岐先等の実行時情報（トレース）を取得する．

性能検証

機能検証で取得したトレースを，反映するための擬似命令を用いて，対象の DFG 中の新規命令を置換し，擬似 DFG を得る．性能検証では，トレースを基に命令の実行タイミングを模擬しながら，擬似 DFG を実行することによって，性能を計測し，評価する．

この手法では，トレースの評価時間がボトルネックとならないよう，パイプライン回路を構成する必要がある．また，このように対象命令をバイナリ変換する手法では，サイクルタイムの正確さが保証できないという問題が指摘されている[3]．この要因としては，メモリ階層の構成が性能を支配する点が大きい．DDMP では DFG でプログラムが与えられるため，データのプリフェッチが容易に行える．このため，こうしたデータ配置の問題を容易に回避できる．一方，STP は負荷の変動に対する緩衝能力を備えており，動作モデルは，この特性を正確に反映する必要がある．

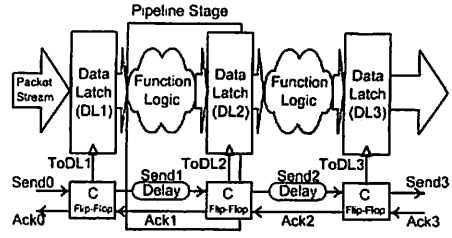

図 1　自己タイミング型パイプライン機構

次章では，負荷に応じた緩衝能力を，プロセッサコア単位で捉える STP の動作モデルを示す．

3. シミュレーションモデル

自己タイミング型パイプライン STP は，隣接するステージ間でのみデータ転送制御信号を局所的に授受（ハンドシェイク）するため，隣接するステージ間のハンドシェイクのみを保証すれば，システム全体の論理動作を保証できる．したがって，追加したパイプライン段のみのタイミング制約を守れば，他の箇所には一切影響を及ぼさないため，容易にチップ上にシミュレーション用の回路を追加できるという特徴を備えている．

STP は，一時的な過負荷を緩衝できる特性を備える．これまでは，この特性を正確に反映するために，ハンドシェイクの信号遷移の因果関係を逐一追跡する必要があり，多大なシミュレーション時間と複雑なシミュレーション回路が必要であった．これに対して，STP の負荷を基準に振舞いを模擬することで，シミュレーションに必要なハードウェアを簡素化できる，マクロフローモデルを示す．

3.1 STP のマクロな動作特性

STP の基本構成を，図 1 に示す．STP の各パイプライン段（以下，単にステージ）は，データラッチ，処理回路，及び一致記憶フリップフロップ（Coincidence flip-flop: C 素子）により構成される．STP 内では各パケットは C 素子間のハンドシェイクにより自律的に移動する．

パケットが存在するステージの C 素子は，先行の C 素子に転送を要求する send 信号を出す．先行のステージが空であれば，転送を許可する ack 信号を受けとり，パケットはそのステージを通過する．したがって，send 信号の遅延および ack 信号の遅延を，それぞれ T_f および T_r とすると，パケットの転送間隔は $(T_f + T_r)$ となる．このとき，ある時刻 t におけるパケット間の時間的な距離を $D(t)$ とおくと，$D(t) \geq (T_f + T_r)$ の場合に，パケットは $\frac{1}{T_f}$ の速度で前進できる．一方，先行するステージが空でない場合

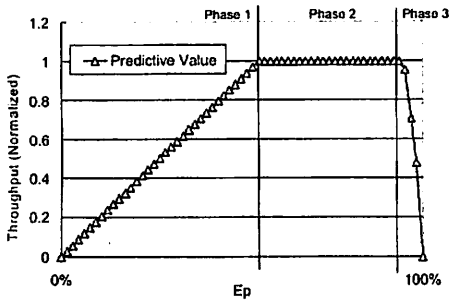

図 2 STP システムのスループット

などで，$D(t) < (T_f + T_r)$ まで詰められると，パケットはラッチされ ack 信号を待つ．これを衝突と呼ぶ．

衝突の影響は，実際の STP チップ（実 STP）では，3 段階の不連続な性能低下として現れることが知られている[4]．実 STP システムのスループットを，図 2 に示す．グラフの横軸は負荷であり，縦軸はスループットの最大値を 1 として正規化したものである．このグラフは，現行の DDMP チップの実測結果である．DDMP チップでは 3 段階に性能特性が変化する原因は，回路実装における $(T_f + T_r)$ の不均質化である．設計ツールや製造環境に委ねられる回路実現工程において，設計レベルで定めた遅延を保つのは困難であるからである．

図中では，性能低下の特性に基づき，各フェーズを，Phase 1，Phase 2，および Phase 3 に分類している．Phase 2 では，プログラムとハードウェア構成の組み合わせが最適である，すなわちステージ数の過不足がなく，最大の性能が達成されていることを示している．一方，Phase 2 の幅や境界は，$(T_f + T_r)$ で決定される．これは，設計の手戻りを局所化かつ極小化するには，初期の設計段階では，$(T_f + T_r)$ のばらつきを反映した，安全側の見積りが不可欠であることを示している．

以上より，$(T_f + T_r)$ のばらつきを見込んで，負荷に応じたパケットの速度を解析しておき，負荷に応じた速度でパケットを転送することで，STP システムの振舞いを高速かつ安全に模擬できる．

3.2 マクロフローモデル

前節に述べたように，リング型 STP では，衝突が生じた場合に，その影響が 3 フェーズの不連続なパケットの速度の変化で現れる．ここから，$D(t)$ と $(T_f + T_r)$ を用いて，各フェーズにおけるパケットの速度（$V(t)$）及び不連続点の境界条件を求める．以降，STP 中のパケット数，STP のステージ数，ステージ i における $(T_f + T_r)$，全ステージの T_f の合計，及び全ステージの T_r の合計を，それぞれ，P_{total}，pl，$(T_{fi} + T_{ri})$，$\sum T_f$，及び $\sum T_r$ とおく．

Phase 1

全ステージにおいて $D(t) \geq (T_{fi} + T_{ri})$，すなわち衝突がない状態を Phase 1 と呼ぶ．この状態では，STP のエラスティック能力により，$(T_{fi} + T_{ri}) = T_{max}$（最大の $(T_f + T_r)$）であるステージ i（以降，最大ステージ）を通過したパケット群では $D(t)$ は T_{max} 以上に保たれる．この時，各ステージでのパケットの転送時間において T_r は無視でき，パケットが一周する時間は $\sum T_f$ である．したがって，$V(t)$ は定数 $\frac{pl}{\sum T_f}$ となる．

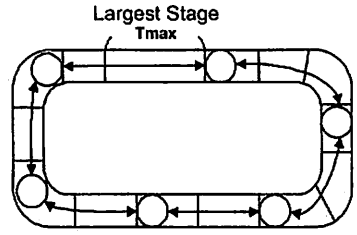

図 3 パケットの模式図（Phase 1）

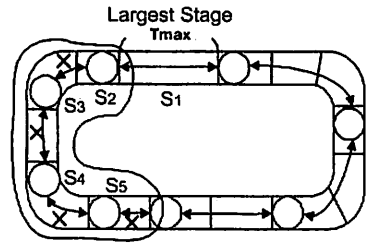

図 4 パケットの模式図（Phase 2）

全ステージのハンドシェイク時間から $D(t) = \frac{\sum T_f + \sum T_r}{P_{total}}$ であるが，Phase 1 では衝突が発生せず，$\sum T_r = 0$ である．このため，図 3 に示すように，P_{total} は，$D(t) \geq T_{max}$ となるよう，

$$\sum T_f \leq T_{max} \times P_{total} \tag{1}$$

を満たす必要がある．すなわち，Phase 1 は，P_{total} が，

$$P_{total} \leq \frac{\sum T_f}{T_{max}} \tag{2}$$

を満たす場合である．

Phase 2

P_{total} の増加により，式 (2) が満たされないとき，最大ステージの通過に際し，最初のパケット衝突が発生する．この時，STP のエラスティック能力により，最大ステージを通過するパケットは，T_{max} 以上に隔てられるため，パケットは最大ステージの直前で，最大 T_{max} の間待つことになる．つまり，パケットが周回する時間は，式 (2) を超過するパケット 1 つにつき，T_{max} 増加する．この状態を Phase 2 と呼ぶ．したがって，Phase 2 における $V(t)$ は，P_{total} の増加に対する速度の変化が一定であるから，P_{total} に関する一次式となる．

Phase 2 では，図 4 に示すように，衝突は，最大ステージ

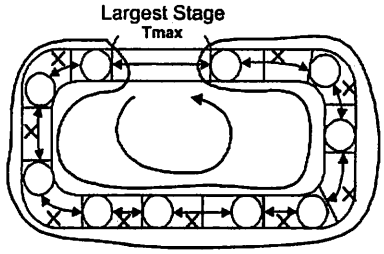

図 5 パケットの模式図 (Phase 3)

を起点に後続ステージに伝播する．この範囲は，各ステージが持つ時間的な余裕で決まる．最大ステージを S_1，後続するステージを順に S_2，S_3，$\cdots$，S_{pl} とおくと，S_i が持つ余裕時間は，$T_{max} - (T_{fi} + T_{ri})$ となる．よって，衝突に影響されるステージ数 n は，

$$P_{over} \times T_{max} \leq \sum_{i=1}^{n} \{T_{max} - (T_{fi} + T_{ri})\} \quad (3)$$

を満たす最小数となる．ここで，P_{over} は Phase 1 を超過したパケット数である．

Phase 2 と Phase 3 の境界条件は，式 (3) より，

$$P_{over} \leq \frac{\sum_{i=1}^{n} \{T_{max} - (T_{fi} + T_{ri})\}}{T_{max}} \quad (4)$$

である．左辺は，式 (2) より，

$$P_{over} = P_{total} - \lfloor \frac{\sum T_f}{T_{max}} \rfloor \quad (5)$$

となる．右辺は衝突が一周する $n = pl$ のとき最大で，

$$\begin{aligned} & \frac{\sum_{i=1}^{pl} \{T_{max} - (T_{fi} + T_{ri})\}}{T_{max}} \\ = & \frac{\sum_{i=1}^{pl} T_{max}}{T_{max}} - \frac{\sum_{i=1}^{pl} T_{fi} + \sum_{i=1}^{pl} T_{ri}}{T_{max}} \\ = & pl - \frac{\sum T_f + \sum T_r}{T_{max}} \end{aligned} \quad (6)$$

となる．式 (4) に式 (5) と式 (6) を代入すると，

$$\begin{aligned} P_{total} - \lfloor \frac{\sum T_f}{T_{max}} \rfloor \leq & pl - \frac{\sum T_f + \sum T_r}{T_{max}} \\ \lceil \frac{\sum T_r}{T_{max}} \rceil \leq & pl - P_{total} \end{aligned} \quad (7)$$

となる．すなわち，Phase 2 は，P_{total} が，

$$\frac{\sum T_f}{T_{max}} < P_{total} \leq pl - \lfloor \frac{\sum T_r}{T_{max}} \rfloor \quad (8)$$

を満たす場合である．

Phase 3

さらなる P_{total} の増加により，式 (8) が満たされないとき，図 5 に示すように，各ステージに衝突を緩衝する余裕がないので，衝突の伝播は一周し，最大ステージでの新たな衝突にまで波及する．すなわち，一旦発生した衝突は STP を周回し続け，すべてのパケットがステージを前進するたびに衝突する．この状態を Phase 3 と呼ぶ．

Phase 3 では，全ステージで衝突が発生しており，パケットはバブルとすれ違わない限り前進できない．STP 中のバブルの総数を B_{total} $(= pl - P_{total})$ とすると，パケットは，バブルの一団と 1 回すれ違うことで，B_{total}[ステージ] だけ前進できるから，パケットが STP を 1 周するには，$\frac{pl}{B_{total}}$ 回だけバブルの一団とすれ違うことになる．また，1 回すれ違うために，バブルの一団は，$pl - B_{total}$[ステージ] だけ前進する必要がある．これらのことから，パケットが STP を 1 周する時間は，バブルの一団が STP を $\frac{pl}{B_{total}} \times (pl - B_{total}) \div pl = \frac{pl - B_{total}}{B_{total}}$ 回だけ周回する時間で規定できる．ここで，バブルが STP を 1 周する時間は，$\sum T_r$ である．

以上より，Phase 3 における $V(t)$ は，

$$V(t) = pl \div \frac{(pl - B_{total}) \times \sum T_r}{B_{total}} = \frac{pl}{\sum T_r} \times \frac{pl - P_{total}}{P_{total}} \quad (9)$$

となり，P_{total} で捉えることができる．

以上より，リング型 STP の性能が，P_{total} に関する 3 段階のフェーズにおいて，$V(t)$ を用いて統一的に扱えるマクロフローモデルが定義できる．以降，P_{total} に対応するパケットの速度を $V(P_{total})$ とおく．

実システムのシミュレーション

マクロフローモデルに基づくシミュレーションでは，各フェーズにおける $V(P_{total})$ が必要である．これらは，システムの設計時に見積られた，pl と $(T_{fi} + T_{ri})$ から算出する．すなわち，システムの機能をパイプライン分割する際に見積られる論理ゲートや配線遅延に基づき，T_{fi} と T_{ri} を決定し，パラメータ値を求める．この際，遅延のばらつきを見越して，$\sum T_f$ と T_{max} を設定すれば，安全側でのシミュレーションができる．

また，DDMP コアでは，隣接する先行ステージにパケットを複製する．この実装では，パケットの複製時に，一時的な衝突が発生し，エラスティック能力の時間的な余裕を消費する．この際，巨視的には P_{total} が一定であっても，パケットの複製回数 (C) に応じて，各フェーズの境界点が変動し得る．これに対して，エラスティック能力を最も消費する，複製パケットが連続しており，直後に消去パケットが連続する場合 (以後，最悪ケースと呼ぶ) の $V(P_{total})$ を用いて，安全側の見積りを行う．つまり，最悪ケースでのパケットの周回時間を，ハンドシェイクを逐一模擬する等して算出し，C に応じた $V(P_{total})$ を求めておけばよい．

以上のモデルによって，オンチップ・シミュレーションでは，P_{total} の変動を模擬して，$V(P_{total})$ に基づいて，パケットを転送するのみで，簡易に STP システムの安全な模擬ができる．

4. オンチップ・マクロシミュレータ構成

本章では，命令の解釈・実行を STP に実装した DDMP コアに，負荷に応じてパケットの転送時間を調整する機構

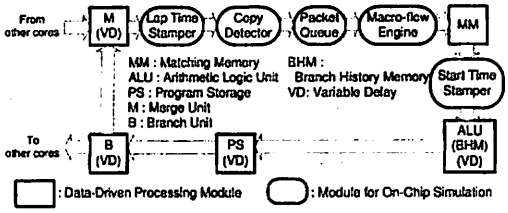

図 6 オンチップ・シミュレータの構成

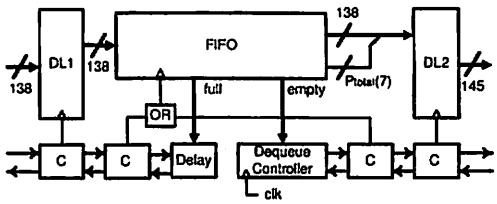

図 7 Packet Queue のパイプライン回路

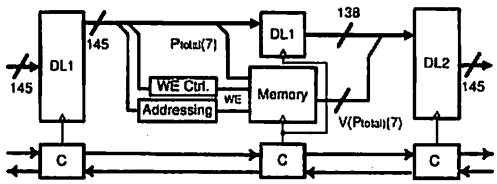

図 8 Macro-flow Engine のパイプライン回路

を追加した，シミュレーション回路を示す．

4.1 パイプライン回路構成

DDMP コアでは，データ駆動原理に基づき命令を解釈・実行するモジュールを STP でパイプライン分割しており，さらにデータを分流／合流するモジュール（B/M）により出入り口を接続して，リング型の STP を形成している．このリング構成により，空間的な並列性を時間的（パイプライン的）に重畳することができる．すなわち，各種粒度の並列性を自然に引き出すことで，高いパイプライン処理効率を達成できる．

この構成では，データがリング型 STP を 1 周する時間（周回時間）が命令の実行時間に相当する．つまり，$V(P_{total})$ に応じて，周回時間を動的に調整することで，対象システムが模擬できる．このとき，パケット数の変化を追跡するためには，トレースに基づく擬似 DFG の実行・解釈が必要となり，シミュレーションのオーバヘッドとなり得る．

これに対して，オンチップ・シミュレーションでは，動的なパケット数を計測するモジュールを DDMP コアに搭載し，擬似 DFG を直接的に実行・解釈すると同時に，周回時間を調整して，性能を計測する．図 6 に，シミュレータの構成を示す．DDMP コアでは，ALU 部にはトレースを評価する分岐履歴メモリ（BHM）を内蔵させ，各データ駆動処理モジュールには可変遅延機構（VD）を内蔵させる．BHM は，トレースとして分岐方向の保持と評価を行うパイプライン回路であり，VD は，パケット中の選択信号で任意の遅延を選択することで，パケットの転送時間を動的に調節する機構である[5]．

本シミュレータ構成では，擬似 DFG を直接的に評価する．すなわち，まず，評価に先立って，$V(P_{total})$ を Macro-flow Engine に格納する．また，トレース情報と擬似 DFG を，それぞれ，BHM と PS に格納する．入力されたパケットは，Packet Queue で一時的にキューイングされる．Packet Queue のパイプライン回路を図 7 に示す．Packet Queue は，キューイングされたパケットのタイムスタンプ値から，P_{total} を算出する．Macro-flow Engine は，ルックアップテーブルを持ち，$V(P_{total})$ を読み出す．読み出した $V(P_{total})$ を模擬できる遅延を VD から選出する選択信号をパケットに付与する．これにより，算出された P_{total} に基づき，各 VD が適切に選択され，パケットの周回時間が模擬される．Macro-flow Engine のパイプライン回路を図 8 に示す．また，周回開始時刻を与える Start Time Stamper と，タイムスタンプ値に周回時間を積算する Lap Time Stamper により，パケットの滞在時間から，P_{total} の算出を可能とする．

DDMP コアでは，命令によりパケットを明示的に複製あるいは消去することができ，定常的に高いパイプライン利用率を確保することが容易である．3.2 節に示したとおり，このパケットの複製・消去が連続する場合，巨視的には P_{total} が一定であっても，マクロな性能特性が変化する．これに対して，Copy Detector で，パケットの複製回数 C を検出して，パケットに付帯させる．次いで，Macro-flow Engine で，C に応じた $V(P_{total})$（$V(P_{total}, C)$）をルックアップすることで，複製・消去の影響を反映する．

このような構成により，1 命令の模擬が，ほぼ 1 命令の実行時間内で行えるため，シミュレーション時間は大幅に短縮できる．

5. 評　　価

前章で提案したオンチップ・シミュレータ構成では，見積りの精度は，マクロフローモデルに依存する．本章では，マクロフローモデルの妥当性を示す．また，シミュレータの実現可能性を示す．

見積り精度

マクロフローモデルに基づくシミュレータは，Java を用いて実装した．本シミュレータは，DFG，入力パケット，及び pl 等の DDMP コア単位の情報を入力として，タイムスタンプを付与した処理結果パケットを出力する構成となっている．

また，LSI の製造工程における，$(T_{fi} + T_{ri})$ のばらつきも反映したパケットの速度を得るため，1 チップ実装され

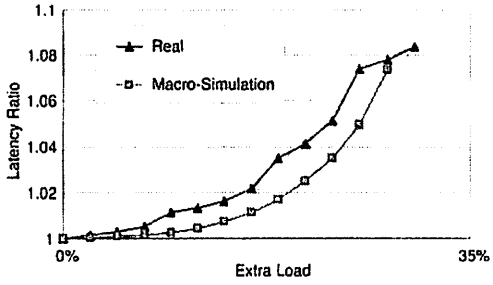

図 9 実システムの性能見積り

たDDMPにおいて，3.2節に述べたパラメータやパケットの周回時間を採取できるDFGを実行した．これらは，回路設計時に見積ったパラメータと同等のものである．さらに，パケットの複製・消去の反映のため，$V(P_{total}, C)$ は，最悪ケースにおける周回時間と複製がない場合の周回時間との平均値とした．これは，DDMPでは，通常，定常的に高いパケット流量を保てるようにプログラムが最適化されるため，結果的に複製・消去は分散されるためである．

評価に先立って，実DDMPチップでパケットの速度を測定したところ，Phase 1であってもパケットの速度は微かに変動し，約3%のゆらぎがあった．これは，回路設計で仮定した $(T_{fi} + T_{ri})$ のばらつきによる，パケット群の偏りが主因であると考えられる．このようなばらつきは，LSIチップの製造ばらつきやチップ温度・電圧変動等に起因しており，現行のLSI製造・実装技術では不可避である．ところが，そのような偏りの性能見積りへの反映は，半導体素子レベルでの模擬を必要とし，設計中には困難である．したがって，性能見積りでは，この揺らぎの分だけ，設計余裕をとる必要がある．このことから，マクロフローモデルによる見積りと実STPの実効性能の誤差は，この設計余裕内に収まっていれば許容できる．

実際の応用システムの開発においては，要求される入力データ・レートで実行可能かどうかが第一の性能評価指標になる．本評価では，応用プログラムの入力データ・レートを固定して実行し，これと同時並行に，定常的な負荷（Extra Load）を発生するプログラムを多重に実行した．追加する負荷の量を変えることによって，Phase1〜3で応用プログラムが動作する状態を設定でき，3段階のフェーズを扱えるマクロフローモデルの有効性が判断できる．

図9に，負荷変動が大きい，FFT処理を実行したときの評価結果を示す．図中横軸は負荷であり，縦軸は最良値を1とした処理レイテンシである．結果，レイテンシの誤差は，設計余裕内に収まっており，また，オーバフロー状態，すなわちシステムの処理能力を超える負荷を追加した場合の臨界点を安全側に見積っていることが判る．

以上より，マクロフローモデルは，現行のLSI設計技術に不可欠な設計余裕を持って，実効性能を安全側に見積れるがことが判る．

回路規模

オンチップ・シミュレータの実現可能性を実証するために，提案したシミュレータ構成のうち，マクロフローモデルに基づくシミュレーション用の基本モジュール，すなわち，図6中のCopy Detector以外のシミュレーション用モジュールを，0.18μmCMOS 6LMのデザインルールで設計した．設計では，シミュレーション，回路合成，及び配置配線に，それぞれCadence社のNC-sim, Build Gates, およびSilicon Ensembleを使用した．結果，DDMPコアの動作に影響を与えることなく，パイプライン動作できることを確認できた．また，総セル数は3142個であり，回路面積は約 $5mm^2$ に収まることが判った．

6. おわりに

本稿では，初期の設計段階からのアーキテクチャの最適化を迅速かつ安全に行うために，オンチップ・シミュレーションの構想を提案した．その中で，自己タイミング型パイプラインのマクロフローモデルに基づく，シミュレーション回路の構成法を提案した．提案したシミュレータ構成の見積り精度の評価として，マクロフローモデルに基づくシミュレータを実装し，現行のLSI設計技術に不可欠な設計余裕内で実効性能を安全に見積れることを確認した．また，シミュレーション回路の基本モジュールの自己タイミング型回路を提案して，0.18μmCMOS 6LMのデザインルールで設計し，動作確認した．今後は，オンチップ・シミュレータ全体のLSI試作を行い，回路コストや性能評価精度の評価を行う予定である．

謝辞 自己タイミング型パイプライン回路に関して，全般的なご指導を頂いた大阪大学名誉教授寺田浩詔先生，ならびに，技術的なご支援・ご議論を頂いたシャープ株式会社の関係諸氏に深く感謝の意を表します．

参考文献

1) H. Terada, S. Miyata, and M. Iwata, "DDMP's: self-timed super-pipelined data-driven multimedia processors," Proc. IEEE, Vol.87, No.2, pp.282-296, Feb. 1999.
2) 児玉祐悦，片下敏宏，佐谷野健二，"リコンフィギュアラブルシステムREXへの並列計算機EM-Xの実装，" 研究報告 ARC-152, pp.109-114, Mar. 2003.
3) 中田尚，大野和彦，中島浩，"高性能マイクロプロセッサの高速シミュレーション，" 先進的計算基盤システムシンポジウム（SACSIS 2003）論文集, pp.89-96, May. 2003.
4) 岡本俊弥，"データ駆動型メディアプロセッサ，" 情報処理学会誌，Vol.39, No.3, pp.208 214, Mar. 1998.
5) 三宮秀次，小笠原新二，岩田誠，"オンチップ評価機構を搭載した自己タイミング型パイプラインシステムの検討，" 研究報告 ARC-165, pp.93-98, Dec. 2005.