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

2006－MPS－60 (16)
2006／6／26

マルチスレッドプロセッサにおける再構成可能キャッシュメモリ

小笠原 嘉泰[†] 館 一平[†] 渡邉 聡[†]
佐藤 未来子[†] 笹田 耕一[†,☆] 内倉 要[†,☆☆]
浅野 一成[†] 並木 美太郎[†] 中條 拓伯[†]

近年，リコンフィギャラブルデバイスを用いたシステムが多数登場しており，今後それらに用いられるプロセッサの高性能化が予想される．プロセッサの高性能化にマルチスレッド化があるが，複数のスレッドで１つのキャッシュを共有している場合，キャッシュミスが頻発し，性能が低下する．そこで，本稿では，リコンフィギャラブルデバイスを対象とし，マルチスレッドプロセッサにおいて，プログラムごとにキャッシュ構成を再構成させる方式を提案した．設計したキャッシュ構成の中で性能比較を行い，プログラムに最適な構成を選定し，性能向上率を見積もった．その結果，従来の固定方式に比べ 15.12 %の性能向上を確認した．提案した再構成可能キャッシュメモリは，キャッシュサイズを増加させず，Way とブロックサイズのみを再構成し，性能向上を目指すので，チップ面積の有効活用やキャッシュサイズ増加の抑制につながる．

Towards Reconfigurable Cache Memory for a Multithreaded Processor

Yoshiyasu Ogasawara ,[†] Ippei Tate ,[†] Satoshi Watanabe ,[†] Mikiko Sato ,[†]
Koichi Sasada ,[†,☆] Kaname Uchikura ,[†,☆☆] Kazunari Asano ,[†] Mitaro Namiki [†]
and Hironori Nakajo[†]

Recently reconfigurable devices such as FPGA have improved performance (gate speed and the number of gates) and reconfiguration time. System designers found that they need a high-performance processor for their reconfigurable device based systems. To improve processor performance, a multithreaded architecture has been introduced; however, performance decreases drastically because of cache misses for shared cache among threads. Moreover, each program that a multithread processor executes may have very different cache access pattern, so that cache optimization for a multithread processor becomes much more complex compared to conventional superscalar processors. In this paper, we propose a new cache design which reconfigures cache configuration for each program on reconfigurable device. We found out optimal configuration for each program from designed cache configurations, and estimated improvement rate of reconfigurable cache. The result shows performance gains of 15.12 % higher than fixed cache design.

1. はじめに

FPGA に代表されるリコンフィギャラブルデバイスは，高速化，高集積化，再構成時間短縮の一途をたどっている．さらに近年その傾向は著しく，従来では実現できなかった大規模なプロセッサや複雑な回路を実装することが可能となった．そのため，１チップ中にプロセッサと専用回路を格納し，１つのシステムとする SoC (System on a Chip) や組み込みプロセッサにおいて，ソフトコアを提供する例は数多く見られる．ソフトコアとして Xilinx 社の MicroBlaze[1] や Altera 社の Nios[2] がある．これらのプロセッサは，Intel 社の Pentium[3] などの高性能プロセッサに比べ，非常に小規模であり性能も低い．今後，さらに多くのリコンフィギャラブルデバイスを用いたシステムが登場するとき，それらに用いられるプロセッサの高性能化が予想される．

プロセッサの高性能化として，マルチスレッドアーキテクチャに注目が集まっている．マルチスレッドプロセッサの一つに SMT (Simultaneous MultiThreading) プロセッサがある．SMT は，複数のスレッドでプロセッサ中のリソース (演算器，キャッシュなど) を共有しながら同時に実行するという特徴を持つ．SMT の有効性は，すでに文献[4] において示されており，CMP (Chip Multi Processor) に比べ，少ないハードウェアの追加量でマルチスレッドプロセッサを実現できる．

† 東京農工大学大学院 工学府
Graduate School of Technology, Tokyo University of Agriculture and Technology
☆ 現在，東京大学大学院 情報理工学系研究科
Presently with Graduate School of Information Science and Technology, The University of Tokyo
☆☆ 現在，株式会社 小松製作所
Presently with Komatsu Corporation

ところが，SMT では 1 つのキャッシュを複数のスレッドで共有するため，キャッシュブロックの競合や枯渇が発生する．また，SMT のメモリアクセスパターンは，複数のスレッドによる同時実行のため，シングルスレッドプロセッサに比べ，煩雑化する．そのため，SMT のキャッシュはプログラムごとに最適な構成を大きく変化させている場合が多い．ASIC のような固定デバイスにキャッシュを実装するとき，キャッシュの構成は，どのプログラムにおいても性能を維持させるべく平均的，一般的な構成を採用する．しかし，プログラムによっては固定構成により性能低下を招く場合があり，特に SMT などのマルチスレッドプロセッサにおけるキャッシュでは，性能低下が顕著になる恐れがある．

そこで本稿ではリコンフィギャラブルデバイスの再構成という機能を用い，SMT などマルチスレッドプロセッサにおけるキャッシュ構成を再構成する方式を提案する．この方式はマルチスレッドプロセッサにおけるキャッシュの短所を解決し，性能向上を目指す．また，リコンフィギャラブルデバイス上では，プロセッサと共に他の専用回路を多く実装するので，キャッシュのチップ面積を抑えなければならない．本方式はキャッシュサイズを増加させず，構成のみを変え，性能向上を目指すので，チップ面積の有効活用やキャッシュサイズ増加の抑制につながる．

2. 再構成可能キャッシュメモリ

2.1 再構成可能キャッシュメモリの背景

メモリアクセスパターンは，実行するプログラムの特性や実行時間により変化する．そのため，キャッシュの最適な構成もプログラムにより変化する．特に，SMT などのマルチスレッドプロセッサでは，メモリアクセスパターンが煩雑化し，キャッシュにおいて，スレッドによるブロック共有という概念が発生する．ブロック共有とは，一定の時間帯において，同じキャッシュブロックを複数のスレッドが共通に使用し，キャッシュを効率的に利用することをいう．そのため，キャッシュの最適な構成は，シングルスレッドプロセッサとマルチスレッドプロセッサで異なる．

また，メモリアクセスパターンは，プログラムをどのようにスレッド分割するかによっても変化する．このようなマルチスレッドプロセッサの特性から，キャッシュ構成をプログラムごとに再構成することは性能向上につながると考えられる．

2.2 再構成可能キャッシュメモリの概要

ここで，我々は，動作させるプログラムによってキャッシュ構成を変化させ，最適な構成とする**再構成可能キャッシュメモリ (Reconfigurable Cache Memory)** について提案する．

再構成するキャッシュの構成要素として，Way とブロックサイズを取り上げる．Way の増加は，キャッシュの競合ミス (Conflict misses) を抑え，性能向上を促す．しかし，同時にキャッシュへのアクセス時間が増加してしまい，性能低下につながってしまう．また，ブロックサイズの増加は，キャッシュの初期ミス (Compulsory misses) を抑え，性能向上を促す．しかし，同時にキャッシュとメモリ間の転送ブロックデータが増加してしまい，ミスペナルティが大きくなってしまう．このような Way とブロックサイズのトレードオフを考慮し，再構成する．Way とブロックサイズを変化させると，タグの変化やマルチプレクサなどの付加が発生するので，ハードウェア量はわずかに変化する．しかし，キャッシュのハードウェア量の大部分はキャッシュサイズが占めるので，Way とブロックサイズの変化による影響は少ない[6]．

再構成するキャッシュは，データキャッシュのみとし，命令キャッシュは再構成しない．命令キャッシュは，データキャッシュとくらべてシーケンシャルアクセスが多く，最適な構成があまり変化しないためである．

再構成するタイミングは，プログラムの実行前，もしくはプログラムとプログラムの切り替え時とし，その再構成する指令は OS などシステムソフトウェアが行うものとする．ここで，問題となるのが再構成する時間である．再構成に多大な時間を要してしまう FPGA などのリコンフィギャラブルデバイスの場合は，プログラム実行前にキャッシュを最適な構成に変え，実装する．一方，再構成時間が数 μs オーダ，さらに高速なものであると数十〜数百サイクルであるダイナミックリコンフィギャラブルデバイス[7] が近年登場しつつある．これらのデバイスを使用することで再構成時間の問題を緩和すると，プログラムの切り替え時にキャッシュを再構成させても大きなコストとならない．そのため，プログラムの切り替え時にキャッシュを再構成でき，プログラムごとに最適なキャッシュ構成を実装できる．デバイス技術の向上により，今後さらに再構成時間は短縮するものと考えられる．また，部分リコンフィギャラブルデバイスや部分再コンフィギュレーション[8] を使用することで，プロセッサの構成を変えず，キャッシュのみを再構成すれば再構成時間はさらに短縮する．

2.3 キャッシュメモリの設計

再構成可能キャッシュメモリの性能評価を行うため，表 1 のようなキャッシュメモリを設計した．このキャッシュメモリは実行駆動型シミュレータである．1 次キャッシュは，命令キャッシュとデータキャッシュを区別し，2 次キャッシュは命令，データキャッシュ共通とした．全てのキャッシュにおいて，リプレースメント方式は LRU とし，ライト方式はライトバックとした．1 次命令キャッシュはスレッドの個数分用意する．1 次データキャッシュは全てのスレッドで共通に扱う．

ここで，プログラムに最適なデータキャッシュ構成

表 1 設計したキャッシュメモリのパラメータ
Table 1 Parameters of designed cache memory

項目		構成要素
サイズ	L1命令キャッシュ	8KB
	L1データキャッシュ	4KB, 8KB, 16KB, 32KB
	L2キャッシュ	512KB
Way	L1命令キャッシュ	1
	L1データキャッシュ	1, 2, 4
	L2キャッシュ	16
ブロックサイズ	L1命令キャッシュ	32B
	L1データキャッシュ	16B, 32B, 64B,128B,256B
	L2キャッシュ	256B
レイテンシ	L1命令キャッシュ	1サイクル
	L1データキャッシュ	1サイクル
	L2キャッシュ	20サイクル

表 2 レイテンシの増加幅
Table 2 Increment of lantency

項目	レイテンシ増加幅
Way 1	L1データレイテンシ
Way 2	L1データレイテンシ+1サイクル
Way 4	L1データレイテンシ+1サイクル
ブロックサイズ16B	L2レイテンシ
ブロックサイズ32B	L2レイテンシ+1サイクル
ブロックサイズ64B	L2レイテンシ+3サイクル
ブロックサイズ128B	L2レイテンシ+6サイクル
ブロックサイズ256B	L2レイテンシ+12サイクル

を選定するために，1 次データキャッシュの Way とブロックサイズを表 1 のようにいくつか用意した．また，様々なキャッシュサイズの最適な構成を見つけるため，1 次データキャッシュのみ，キャッシュサイズをいくつか用意した．

Way とブロックサイズを変化させると，それぞれ 1 次キャッシュへのレイテンシと，1 次キャッシュと 2 次キャッシュ間のレイテンシが変化する．Way とブロックサイズが変化した時のレイテンシ増加幅を表 2 のように設定した．本研究において，このキャッシュのレイテンシは重要な要素のひとつとなる．そのため，表 1 のレイテンシと表 2 のレイテンシ増加幅は文献 6) 等 9) を参考にして設定した．

3. 性能評価

本章では，再構成可能キャッシュメモリの有効性について評価する．性能評価は実行駆動型シミュレータ MUTHASI[5] と設計したキャッシュシミュレータを用いた．評価時の MUTHASI のパラメータを表 3 に示す．スレッド数ごとに各パラメータを設定する．

実行したベンチマークプログラムは以下の 3 つである．

- 128 × 128 行列 LU 分解 (LU)
- 64 × 64 行列乗算 (Matrix)
- Radix ソート 基数 : 8 (Radix)

LU 分解，Radix ソートは SPLASH-2 ベンチマークより採用した．全てのプログラムは並列マクロを用いて，明示的にマルチスレッド化している．

3.1 最適なキャッシュ構成

データキャッシュサイズ 4KB のとき，各ベンチマーク

表 3 シミュレータの環境
Table 3 An environment of simulation.

項目	構成要素		
PC (AT #)	1	2	4
Fetch Buffer Size	32	16	8
Dispatch Queue Size	64	32	16
Reorder Buffer Size	128	64	32
Normal Reservation Station Size	Simple ALU:8 Complex ALU:4		
LD/ST Reservation Station Size	8		
Branch History Table Size	1024(gshare)		
Integer ALU	Simple ALU:4, Complex ALU:2		
FPU	Simple ALU:2(delay 4cycle) Complex ALU:2(Mult 17 delay, Div. 30 delay)		
Branch Unit	1		

表 4 キャッシュサイズ 4KB の最適なキャッシュ構成，実行サイクル，性能向上率，ヒット率
Table 4 Optimal configuration, cycles, speed up rate and hit rate of 4KB.

(a) LU

	Configuration	Cycles	Speed up rate	Hit rate
1AT	Best (2way, 64B)	32005283	0.00%	98.01%
	Worst (1way, 256B)	38152700	-16.11%	90.55%
2AT	Best (4way, 128B)	21120283	51.54%	97.24%
4AT	Best (4way, 128B)	17834711	79.46%	96.58%

(b) Matrix

	Configuration	Cycles	Speed up rate	Hit rate
1AT	Best (1way, 16B)	7097294	0.00%	53.38%
	Worst (4way, 256B)	11240356	-36.86%	55.16%
2AT	Best (1way, 128B)	6351373	11.74%	68.39%
4AT	Best (4way, 64B)	6838883	3.78%	65.49%

(c) Radix

	Configuration	Cycles	Speed up rate	Hit rate
1AT	Best (1way, 16B)	10619505	0.00%	99.08%
	Worst (2way, 256B)	11524025	-7.85%	99.61%
2AT	Best (1way, 16B)	6351373	67.20%	98.69%
4AT	Best (4way, 64B)	5993029	77.20%	99.21%

クの，最適なキャッシュ構成，実行サイクル数，性能向上率，ヒット率を表 4 に示す．ただし，性能向上率は 1AT (シングルスレッドプロセッサ) を基準とする．

マルチスレッド化による性能向上率が，LU 分解と Radix ソートにおいて高い．1AT と比較すると，LU 分解の 2AT で 51.54 %，4AT で 79.46 %，Radix ソートの 2AT で 67.2 %，4AT で 77.2 %の性能向上率となっている．また，マルチスレッド化することでキャッシュの最適な構成も変化し，特に LU 分解と行列乗算で著しい．例えば，LU 分解は，1AT から 2AT の変化によって，最適な構成が「Way : 2，ブロックサイズ : 64B」から「Way : 4，ブロックサイズ : 128B」に変化している．一方，Radix ソートでは，1AT から 2AT の変化で最適なキャッシュ構成は変わらないが，2AT から 4AT に変化させると，最適な構成が「Way : 1，ブロックサイズ : 16B」から「Way : 4，ブロックサイズ : 64B」と大きく変化する．

表 4 をもとに，2AT の各ベンチマークの最適なキャッシュ構成を調べると表 5 のようになる．マルチスレッド化により，すべてのベンチマークにおいて最適な構成が異なっている．このような場合，再構成可能キャッシュメモリの有効性は高く，性能向上が期待できる．

表 5 キャッシュサイズ 4KB，2AT の最適な構成
Table 5 Optimal configuration of 4KB on 2AT.

プログラム	最適な構成
LU分解	Way：4，ブロックサイズ：128B
行列乗算	Way：1，ブロックサイズ：128B
Radixソート	Way：1，ブロックサイズ：16B

表 6 再構成可能キャッシュメモリの性能向上率
Table 6 Improvement of reconfigurable cache memory

比較対象	(RECONF) の性能向上率
(A)	8.84%
(B)	15.12%
(C)	14.85%

また，最適な構成は，LU 分解，行列乗算でブロックサイズ 128B となるが，不連続アクセスの場合は性能低下を招くような構成であり，一般の固定方式のキャッシュでは採用しない．しかし，このような特殊なキャッシュ構成はプログラムによっては有効であり，それを実現できる再構成可能キャッシュメモリの有効性はさらに高くなる．

3.2 再構成可能キャッシュメモリの性能向上率

再構成可能キャッシュメモリの性能向上率が具体的にどのくらいになるか比較する．一例として，キャッシュサイズ 4KB，2AT のとき，LU 分解，行列乗算，Radix ソートを実行する場合を考える．ここで，以下のような 4 つのキャッシュ構成を (A)，(B)，(C)，(RECONF) と呼ぶ．

(A) ：構成を Way：4，ブロックサイズ：128B で固定
(B) ：構成を Way：1，ブロックサイズ：128B で固定
(C) ：構成を Way：1，ブロックサイズ：16B で固定
(RECONF) ：再構成可能キャッシュメモリ

このときの性能の性能向上率を表 6 に示す．表 6 から，キャッシュ構成を固定とする場合よりも，(RECONF) の方が性能向上することが分かる．性能向上率は，(B) と比べた場合で 15.12 %になる．

ここでは 3 つのベンチマークを 4 つのキャッシュ構成で実行させた場合の性能比較を行ったが，他の状況においても再構成可能キャッシュメモリは有効である．特に，特殊な構成を最適とするプログラムや，実行時間が長いプログラムにおいてさらに高い性能向上率が得られると考える．

4. おわりに

本稿では，マルチスレッド化によるメモリアクセスパターンの煩雑化やプログラムごとに最適なキャッシュ構成が変化することに着目し，リコンフィギャラブルデバイスを対象として，プログラムごとにキャッシュ構成を再構成させる方式を提案した．設計したキャッシュ構成の中で性能比較を行い，プログラムに最適な構成を選定し，再構成可能キャッシュメモリの性能向上率を見積もった．その結果，固定方式に比べ 15.12 %の性能向上を確認した．提案した再構成可能キャッシュメモリは，キャッシュサイズを増加させず，Way とブロックサイズのみを再構成し，性能向上を目指すため，チップ面積の有効活用やキャッシュサイズ増加の抑制につながる．また，どのような構成もプログラムごとに再構成するので，一般のプログラムでは性能低下を招いてしまうが，特定のプログラムでは性能向上する特殊なキャッシュ構成も，再構成可能キャッシュメモリでは実現できる．

プログラムに最適な構成を，本稿では性能評価により選定した．実際に運用する場合は，キャッシュの性能情報を採取し，統計により選定する方法やアドミッションテストによるキャッシュ構成のスケジューリングを行う選定方法が考えられる．また，性能評価において，1AT (シングルスレッドプロセッサ) でもプログラムよっては最適な構成が変化していた．そのため，マルチスレッド以外のプロセッサにおいても，再構成可能キャッシュメモリは効果的であると考える．

今後は，再構成する構成要素として，リプレースメント方式やライト方式も対象とし，評価する．そして，効果的な再構成可能キャッシュメモリについて再度検討，考察し，実装していく．

参考文献

1) MicroBlaze : *http://www.xilinx.com/ipcenter/catalog/logicore/docs/microblaze_risc_32bit_proc_final.pdf*
2) Nios 3.0 CPU : *http://www.altera.co.jp/literature/ds/ds_nios_cpu.pdf*
3) D. Marr, F. Binns, D. Hill, G. Hinton, D. Koufaty, J. Miller, M. Upton : "Hyper-Threading Technology Architecture and Microarchitecture" , *Intel Technology Journal*, Vol.6, pp.4-15, 2002.
4) D. Tullsen, S. Eggers, H. Levy : "Simultaneous Multithreading:Maximizing On-Chip Parallelism", *Proceedings of 22rd Annual International Symposium on Computer Architecture*, pp.392-403, 1995.
5) 河原章二，佐藤未来，並木美太郎，中條拓伯：“システムソフトウェアとの協調を目指すシングルチップマルチスレッドアーキテクチャの構想”，CSS2002 Vol.2002 No.18 pp.1-8，2002.
6) J L.heneesy, D A.Patterson : "Computer Architecture A Quantitative Approach Third Edition", *Morgan Kaufmann Publishers*, 2002.
7) 天野英晴："リコンフィギャラブルシステム最新情報", *SACSIS 2004*, チュートリアル資料, pp1-11, 2004.
8) Partial Reconfigurability : *http://www.xilinx.co.jp/products/design_resources/design_tool/grouping/adv_design_tech.htm*
9) IA-32 Intel Architecture Software Developer's Manuals : *http://developer.intel.com/design/pentium4/manuals/index_new.htm*