近細粒度並列処理用シングルチップマルチプロセッサにおけるプロセッサコアの構成

木村啓二[†]　内田貴之[†]
加藤孝幸[††]　笠原博徳[†]

早稲田大学理工学部電気電子情報工学科[†]
（株）本田技研[††]

〒169-8555 東京都新宿区大久保 3-4-1 TEL:03-5286-3371
E-mail: {kimura,uchida,kato,kasahara}@oscar.elec.waseda.ac.jp

あらまし　1チップ上に搭載可能なトランジスタ数が増加するにともない，これらの資源を如何に有効に活用し，どのようにスケーラブルな性能向上を達成するかが大きな課題となっている．とりわけ，チップ内で利用できる命令レベル以外の並列性をより多く抽出する技術がこれまで以上に重要となる．このように命令レベル以外の並列性も有効に活用できるアーキテクチャの一つとして1チップ上に複数のプロセッサコアを搭載したシングルチップマルチプロセッサ（SCM）が大きな注目を集めている．本論文では，マルチグレイン並列処理に適したSCMのプロセッサコアを検討するために，マルチグレイン並列処理の主要技術の一つである近細粒度並列処理を，プロセッサコアの同時命令発行数やプロセッサコア共有グローバルレジスタファイルの本数を変えたSCMアーキテクチャに適用して評価を行ったので，その結果について述べる．

Processor Core Architecture of Single Chip Multiprocessor for Near Fine Grain Parallel Processing

Keiji Kimura[†], Takayuki Uchida[†], Takayuki Kato[††] and Hironori Kasahara[†]

Department of Electrical, Electronics and Computer Engineering, Waseda University[†]
HONDA MOTOR CO., Ltd[††]

3-4-1 Ohkubo Shinjuku-ku, Tokyo 169-8555, Japan Tel: +81-3-5286-3371
E-mail: {kimura,uchida,kato,kasahara}@oscar.elec.waseda.ac.jp

Abstract　With continuously increase of transistors integrated onto a chip, it has been a very important how to achieve scalable performance improvement using these transistors effectively. Especially, exploiting different grains of parallelism in addition to instruction level parallelism and effective use of this parallelism in a single chip is getting more important. To this end, a single chip multiprocessor (SCM) architecture that contains multiple processor cores has been attracted much attention. To decide suitable SCM processor core architecture for multigrain parallel processing, this paper evaluates several SCM architectures which have different instruction issue widths and numbers of global shared register file for near fine grain parallel processing, which is one of the key issues in multigrain parallel processing.

1　はじめに

半導体集積技術の向上にともない，チップ内に投入された大量の資源を有効に活用し，スケーラブルな性能向上を達成するためのアーキテクチャおよびコンパイル技術の研究が一層重要となっている．とりわけ，プログラム中から，従来の命令レベルにとどまらずより多くの並列性を抽出し，チップ内で利用する技術が強く求められている．たとえば，EPIC[1)]ではコンパイラによる静的解析により抽出された条件分岐を越えた命令レベル並列性を，predicated execution等のアーキテクチャサポートにより利用しようとしている．また，Emotion Engine[2)]では，画像・音声等の高速処理を目的とし，RICSコアと二つのベクトル演算ユニットを1チップ上に搭載している．さらに，複数のプロセッサコアを1チップ上に集積するシングルチップマルチプロセッサ（SCM）では，MAJCのようにマルチメディア用途に2基のVLIWコアを集積したもの[3)]，Power4のように高性能サーバ用途に2基のスーパースカラでオンチップL2キャッシュを共有する構成のもの[4)]，Hydra, Multiscalar，Superthreaded，Merlotのような，2

命令発行程度のスーパースカラコアを複数搭載し，スレッド投機実行をサポートするアーキテクチャで命令レベル並列性とスレッドレベル並列性を利用するもの [5)~8)]，RAW や FlexRAM のように簡素なプロセッサとローカルメモリを持つプロセッシングエレメント（PE）をチップ内に多数配置し，その挙動をコンパイラで制御するもの [9),10)] などといった研究がなされている．

また，OSCAR マルチグレイン SCM[11)] は，複数命令レベルでの並列処理である（近）細粒度並列処理 [12)] に加え，より大きな並列性を持つループイタレーションレベルの中粒度並列処理 [13)] およびサブルーチンあるいはループ，基本ブロック間の粗粒度並列性 [14),15)] を階層的に組み合わせて使用し，プログラム全体から並列性を引き出すことができる．このマルチグレイン並列処理 [16)] により，SCM アーキテクチャの持つ能力を有効に引き出し，価格性能比の優れたスケーラブルなプロセッサを構築できると考えている．

本論文では，マルチグレイン並列処理をサポートする OSCAR マルチグレイン SCM アーキテクチャのプロセッサコアの構成を検討するため，まずマルチグレイン並列処理の主要構成要素の一つである近細粒度並列処理を，命令発行数の異なるプロセッサコアおよび本数の異なるプロセッサコア間グローバル共有レジスタファイルを持った SCM アーキテクチャに適用して評価を行った．

以下，2 節でマルチグレイン並列処理について，3 節で本論文で評価する SCM について，4 節でこれらのアーキテクチャに近細粒度並列処理を適用して評価した結果について述べる．

2　マルチグレイン並列処理

本節では，OSCAR マルチグレインシングルチップマルチプロセッサで扱う，マルチグレイン並列処理技術について述べる．

マルチグレイン並列処理 [16)] とは，ループやサブルーチン等の粗粒度タスク間の並列処理を利用する粗粒度タスク並列処理（マクロデータフロー処理）[14),17)]，ループイタレーションレベルの並列処理である中粒度並列処理，基本ブロック内部のステートメントレベルの並列性を利用する近細粒度並列処理 [12)] を階層的に組み合わせて，プログラム全域にわたる並列処理を行なう手法である．

2.1　粗粒度タスク並列処理（マクロデータフロー処理）

マクロデータフロー処理では，ソースとなるプログラムを疑似代入文ブロック（BPA），繰り返しブロック（RB），サブルーチンブロック（SB）の三種類の粗粒度タスク（マクロタスク（MT））[14)] に分割する．MT 生成後，コンパイラは BPA，RB，SB 等の MT 間のコントロールフローとデータ依存を解析し，それらを表したマクロフローグラフ（MFG）[17),18)] を生成する．さらに MFG から MT 間の並列性を最早実行可能条件解析 [17),18)] により引きだし，その結果をマクロタスクグラフ（MTG）[17),18)] として表現する．その後，コンパイラは MTG 上の MT を，スタティックスケジューリングの場合にはプロセッサ間データ転送および同期オーバーヘッドを最小化できるようにプロセッサあるいはプロセッサグループ（PG）に割り当て，各プロセッサ用コードを生成する．MTG 中に条件分岐がありダイナミックスケジューリングを用いる場合には，実行時に MT をプロセッサあるいは PG に割り当てる．

2.2　中粒度並列処理（ループ並列処理）

PG に割り当てられた MT が Doall 可能な RB である場合，この RB は PG 内のプロセッシングエレメント（PE）に対して，イタレーションレベルで割り当てられ並列実行される．またこの場合には，各 PE 上のローカルメモリ，あるいは分散共有メモリを有効利用するためのローカライゼーション技術 [19)] を利用可能である．

2.3　近細粒度並列処理

PG に割り当てられた MT が，BPA や中粒度並列処理を適用できない RB である場合，それらはステートメントレベルのタスクに分割され，PG 内の PE により並列処理される．

近細粒度並列処理においては，BPA 内のステートメント，もしくは複数のステートメントから構成される疑似代入文を一つの近細粒度タスクとして定義する．コンパイラは，BPA を近細粒度タスクに分割した後，タスク間のデータ依存を解析してタスクグラフを作成する．次に，このタスクグラフ上のタスクを，データ転送・同期オーバーヘッドを考慮して実行時間を最小化できるように各 PE にスタティックにスケジューリングする．

OSCAR Fortran コンパイラ [20] における近細粒度タスクの PE へのスケジューリングにおいては，スケジューリング手法として，データ転送オーバーヘッドを考慮し実行時間を最小化するヒューリスティックアルゴリズムである CP/DT/MISF 法，CP/ETF/MISF 法，ETF/CP 法，および DT/CP 法 [18] の 4 手法を同時に適用し最良のスケジュールを選んでいる．また，このようにタスクをスタティックにプロセッサに割り当てることにより，BPA 内で用いられるデータのローカルメモリ，分散共有メモリ，レジスタへの配置等，データ配置の最適化やデータ転送・同期オーバーヘッドの最小化といった各種最適化が可能になる．

スケジューリング後，コンパイラは PE に割り当てられたタスクに対応する命令列を順番に並べ，データ転送命令や同期命令を必要な箇所に挿入し，各 PE 毎に異なるマシンコードを生成する．近細粒度タスク間の同期にはバージョンナンバー法を用い，同期フラグの受信は受信側 PE のビジーウェイトによって行なわれる．この時，OSCAR マルチグレイン SCM のように分散共有メモリ（DSM）を持つアーキテクチャでは，データ転送および同期フラグのセットは受信側 DSM への直接ストアの形で行われ（3.1 節），受信側 PE のビジーウェイトも自 PE 上の DSM へのビジーウェイトになるので，プロセッサ間相互接続網のバンド幅低下は起らない．

また，共有グローバルレジスタ（3.3 節）を持つ SCM アーキテクチャでは，可能な限りグローバルレジスタを用いてデータ転送を行なう．データ転送のための共有グローバルレジスタ割り当てには，レジスタカラーリングの拡張を使用する．

3 評価対象 SCM アーキテクチャ

本節では，今回評価を行なったシングルチップマルチプロセッサ（SCM）アーキテクチャおよびそのプロセッサコアについて述べる．評価のため，以下のアーキテクチャを厳密に評価できるクロックレベルシミュレータを開発した．

3.1 ネットワークおよびメモリアーキテクチャ

本論文で評価する SCM のネットワークおよびメモリアーキテクチャは，近細粒度並列処理において従来の評価で共有キャッシュ型より良い性能を示している OSCAR マルチグレイン SCM アーキテクチャ [11],[20] とする．OSCAR マルチグレイン SCM アーキテクチャとは，図 1 のように，CPU，データ転送ユニット（DTU），ローカルプログラムメモリ（LPM），ローカルデータメモリ（LDM），および分散共有メモリ（DSM）を持つプロセッシングエレメント（PE）を相互接続網（図 1 の場合は 3 本のバス）で接続し 1 チップ上に搭載したアーキテクチャである．本論文の評価では，PE は 1 チップに 1 – 4 基搭載されるものとする．

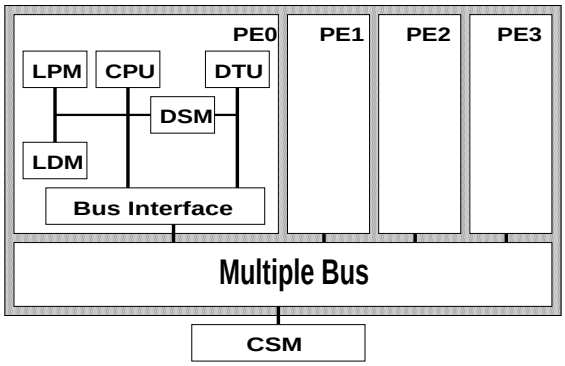

図 1: OSCAR マルチグレイン SCM アーキテクチャ

LPM は各々の CPU で実行するプログラムを格納し，今回の評価では 1 クロックでアクセスできるものとする．同様に，LDM は PE 固有のデータを保持するために使用し，その容量は 1PE あたり 1M バイト，レイテンシは 1 クロックとする．また，DSM は自 PE と他 PE の双方から同時にアクセス可能なマルチポートメモリであり，近細粒度タスク間のデータ転送や，マクロデータフロー処理におけるダイナミックスケジューリング時のスケジューリング情報の通知等に使用する．DSM の容量は 1PE あたり 16K バイトとし，自 PE からのアクセスレイテンシは 1 クロック，他 PE へのリモートアクセス時のレイテンシは 4 クロックとする．さらに，チップ外部には集中共有メモリ（CSM）が接続され，各 PE で共有されるデータを格納する．この CSM のアクセスレイテンシは，今回の評価では 20 クロックとする．

OSCAR マルチグレイン SCM アーキテクチャでは，これら 4 種類のメモリに対しコンパイラが適切なデータ配置を行うことにより，効率の良い並列処理を行うことができる．

3.2 プロセッサコア

各PEが持つCPUは，SPARC V9規格に準拠したスーパースカラプロセッサであるSun Microsystems社のUltraSPARC II[21]をモデルにしたプロセッサである．本論文で評価したSCMアーキテクチャでは，このUltraSPARC IIプロセッサにバリア同期機構等用の特殊レジスタや共有グローバルレジスタを操作するための命令を付加した．

このプロセッサは，9段のパイプラインを複数持つスーパースカラプロセッサである．今回の評価では，このパイプライン本数を1, 2, 4と変えることによってシングルイシュー，2イシュー，4イシューのスーパースカラプロセッサとして使用する．動的スケジューリングの対象となる命令群が格納される命令バッファのエントリ数は12である．

3.3 共有グローバルレジスタ

本論文では，共有グローバルレジスタ（GR）を付加したアーキテクチャについても評価を行なう．

GRには，各PE内のCPUが同時にアクセスすることができるものとする．このときのアクセスタイムは1クロックとし，これらは近細粒度タスクのデータ転送に使用する．また，GRの本数は8, 16, 32と変えて評価を行う．

4 性能評価

本節では，シングルイシューあるいはマルチプルイシューのプロセッサコアを持つPEを複数搭載したSCMアーキテクチャに対し，近細粒度並列処理を適用して評価した結果について述べる．評価に使用したプログラムは以下の3例である．

スパースマトリクスの求解[11] このプログラムは算術代入文のみからなるFortranループフリーコードであり，94個の近細粒度タスク（ステートメント）持つ．

NS3D/SUB4[11] このプログラムは，航空宇宙技術研究所のCFDプログラム「NS3D」のサブルーチン「SUB4」の最外側ループに内包される4つのDo-loopのうち，最も実行時間の大きい2番目のDo-loopのループボディを取り出したもので，429個の近細粒度タスクを持つ．SUB4の最外側ループはDoall可能なループであり，今回の評価は，外側ループの並列性はSCMを複数接続しているマルチプロセッサのSCM間で使用し，SCMに割り当てられたイタレーションをさらにSCM内のPE間で近細粒度並列処理を行なうことを想定して評価を行なっている．

FPPPP/FPPPP このプログラムは，SPECfp95ベンチマーク「FPPPP」からサブルーチン「FPPPP」を抜き出したものである．このサブルーチンはプログラム全体の実行時間の約35%を占める部分であり，サブルーチン全体が333個の近細粒度タスクから構成される．

4.1 命令発行数とPE数に関する評価

はじめに，命令発行数がそれぞれ1，2，4（1Issue, 2Issue, 4Issue）であるプロセッサコアを持つSCMアーキテクチャにおいて，PE数を1から4まで変えて評価を行った．さらに，各々のアーキテクチャにおいて32本の共有グローバルレジスタを付加したアーキテクチャ（GR）に関しても評価を行った．それらのアーキテクチャの性能を，単一のシングルイシュープロセッサによる実行時間に対する速度向上率として，図2，3，4に示す．

まず，各アーキテクチャにおいてほぼ同一のトランジスタ数となる総同時命令発行数4 （1Issue × 4PE, 2Issue × 2PE, 4Issue × 1PE）の時の性能に着目すると，1Issue × 4PEは図2のスパースマトリクス求解では2Issue × 2PEの1.47倍，4Issue × 1PEの2.49倍の性能を得ている．同様に，図3のNS3Dでは2Issue × 2PEの1.23倍，4Issue × 1PEの1.89倍，図4のFPPPPでは2Issue × 2PEの1.40倍，4Issue × 1PEの2.41倍の性能がそれぞれ得られ，いずれの場合も1Issue × 4PEの時が最も良い性能を示している．この理由として，スーパースカラプロセッサコア単体では，12エントリの命令バッファという限られた範囲から実行時に抽出される局所並列性しか利用できないが，SCMアーキテクチャではPE数を増やすことにより，基本ブロック全体から静的に抽出したより大きな並列性を全て利用することができるためであると考えられる．

チップ内PE数4の時に注目すると，図2では1Issue × 4PEに対し，2Issue × 4PEが13.3%，4Issue × 4PEは13.8%の性能向上を得ている．同様に，図3では2Issue × 4PEで13.0%，4Issue × 4PEで13.5%，図4では2Issue × 4PEで19.6%，4Issue × 4PEで21.4%の性能向上を得ている．このこと

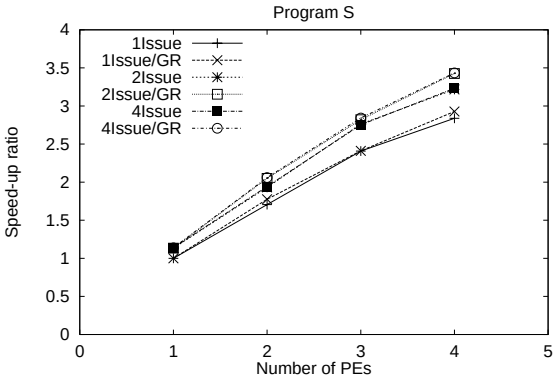

図 2: ランダムスパースマトリクスにおける速度向上率

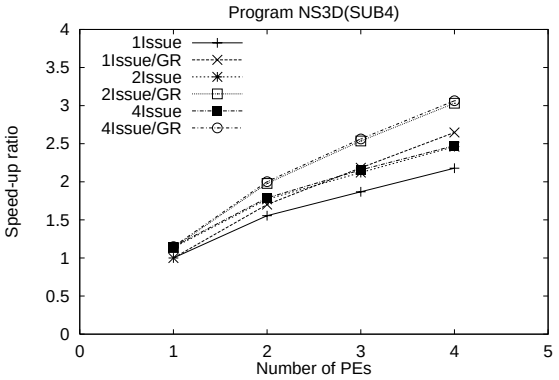

図 3: NS3D における速度向上率

から，プロセッサコア単体の命令発行数を 4 にしても，命令発行数が 2 の時に得られる性能とほとんど変わらないことがわかる．

4.2 共有グローバルレジスタに関する評価

共有グローバルレジスタの効果に着目すると，チップ内に 4PE 集積されているとき，プロセッサ間データ転送の少ない図 2 のランダムスパースマトリクス求解では 1Issue で 3.7%，2Issue で 6.5%，4Issue で 6.3%の性能向上を得ている．データ転送の多い図 3 の NS3D では，1Issue で 21.5%，2Issue で 23.1%，4Issue で 23.8% の性能向上を，同様に図 4 の FPPPP では，1Issue で 14.4%，2Issue で 14.4%，4Issue で 14.7% の性能向上が得られ，共有グローバルレジスタが有効であることがわかる．

次に，NS3D および FPPPP において共有グローバルレジスタファイルのレジスタ本数を変えて評価を行った．レジスタ本数は 0 本（GR0），8 本（GR8），16 本（GR16），32 本（GR32）とした．PE 数 4 の

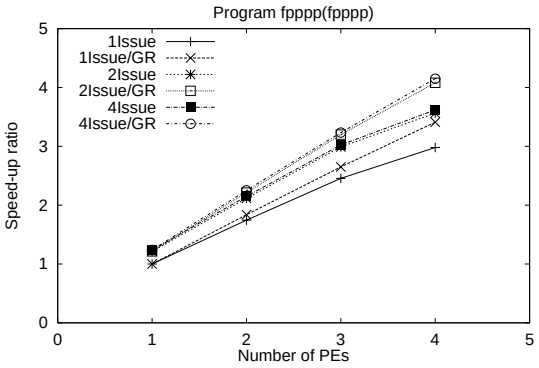

図 4: FPPPP における速度向上率

場合にレジスタ本数を変化させたときの，単一のシングルイシュープロセッサの実行時間を 1 としたときの速度向上率を，図 5，6 に示す．

図 5 の NS3D において，1Issue 時のグローバルレジスタが無い時（GR0）に対してグローバルレジスタが 8 本（GR8）の時は 19.1%の性能向上を得ているが，グローバルレジスタを 32 本（GR32）と増やしても 21.5% の性能向上と，レジスタ本数を増やしても大きな性能向上は見られない．命令発行数を 2 あるいは 4 と増やした時も同様で，2Issue 時に GR8 で 20.5%，GR32 で 23.1%の性能向上，4Issue 時に GR8 で 21.0%，GR32 で 23.8%の性能向上となっており，8 本以上のレジスタを用いても NS3D では大きな性能向上は見られない．

一方，図 6 の FPPPP では，1Issue の時に GR0 に対して GR8 で 8.7%の性能向上，GR32 では 14.4%の性能向上と，レジスタ本数の増加にともない得られる性能向上率も増加している．同様に，2Issue 時に GR8 で 8.9%，GR32 で 14.4%の性能向上，4Issue 時に GR8 で 9.3%，GR32 で 14.7%の性能向上となり，グローバルレジスタ本数の増加とともに性能が向上する．

NS3D では，GR8 の時点で全データ転送・同期の 33.5%が共有グローバルレジスタに割り当てられているが，FPPPP では 18.9%にとどまっている．この違いは，データ転送・同期の出現頻度によると考えられる．

5 まとめ

本論では，マルチグレイン並列処理をサポートするシングルチップマルチプロセッサ（SCM）アーキテクチャのプロセッサ構成を検討するため，マルチグレイン並列処理の一要素である近細粒度並列処理

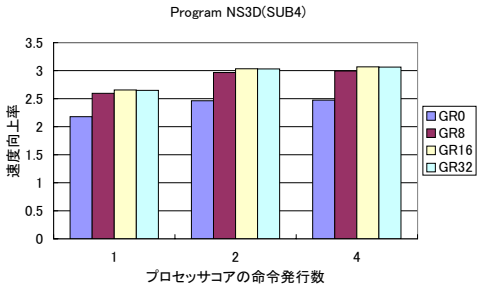

図 5: 共有グローバルレジスタの本数による速度向上率の違い（NS3D）

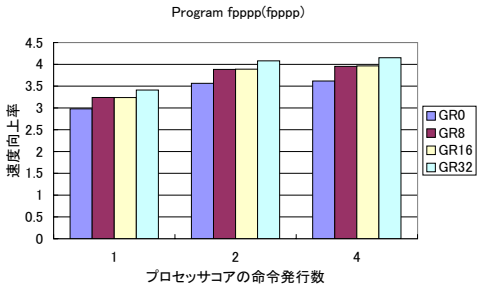

図 6: 共有グローバルレジスタの本数による速度向上率の違い（FPPPP）

を命令発行数の異なるプロセッサコアを持つ SCM アーキテクチャに適用して評価を行った．その結果，数値流体解析プログラムにおいて，1 命令発行 × 4PE の SCM はほぼ同一トランジスタの数の 4 命令発行 × 1PE の 1.89 倍，SPECfp95 の FPPPP においても，1 命令発行 × 4PE の SCM は 4 命令発行 × 1PE の 2.41 倍と，プロセッサコアの命令発行数の増加より，チップ内 PE 数の増加の方が性能向上に貢献することが確かめられた．

共有グローバルレジスタの有無に関しては，数値流体解析プログラムにおいて 1 命令発行 × 4PE の SCM に 32 本の共有グローバルレジスタを付加したときに 21.5%の性能向上を得ることができ，その有効性を確認できた．レジスタ割り当てアルゴリズムの改良により，更なる性能向上を見込めると考えられる．

今後の課題として，大規模アプリケーションに対しマルチグレイン並列化を適用した際の SCM 上での評価，メモリアーキテクチャの更なる検討等が挙げられる．

本研究の一部は，STARC「自動並列化コンパイラ協調型シングルチップ・マルチプロセッサの研究」により行われた．

参考文献

[1] M.Schlansker and B.Rau. EPIC:Explicitly Parallel Instruction Computing. *Computer*, Vol. 33, No. 2, pp. 37–45, 2000.

[2] M.Oka and M.Suzuoki. Designing and programming the emotion engine. *IEEE MICRO Magazine*, Vol. 19, No. 6, pp. 20–28, 1999.

[3] http://www.sun.com/microelectronics/MAJC/. *MAJC Home Page*. Sun Microsystems, Inc., 2000.

[4] K.Diefendorff. Power4 focuses on memory bandwidth. *Microprocessor Report*, Vol. 13, No. 13, 1999.

[5] L.Hammond, B.Hubbert, M.Siu, M.Chen, and K.Olukotun. The Stanford HYDRA CMP. *IEEE MICRO Magazine*, Vol. 20, No. 2, pp. 71–84, 2000.

[6] T.N.Vijaykumar and G.S.Sohi. Task Selection for a Multiscalar Processor. In *31th Int'l Conf. on Microarchitecture (MICRO-31)*, Nov-Dec 1998.

[7] J.-Y. Tsai, Z. Jiang, E. Ness, and P.-C. Yew. Performance Study of a Concurrent Multithreaded Processor. In *Proc.4th Int'l Conf. on HPCA-4*, Fec 1998.

[8] http://www.labs.nec.co.jp/MP98/. *MP98 Project*. NEC Corporation, 2000.

[9] R.Barua, W.Lee, S.Amarasinghe, and A.Agarwal. Maps:A Compiler-Managed Memory System for Raw Machines. In *Proc. of ISCA-26*, June 1999.

[10] Y.Kang, M.Huang, S.Yoo, Z.Ge, A.Keen, V.Lam, P.Pattnaik, and J.Torrellas. FlexRAM:An Advanced Intelligent Memory system. In *Proc. Int'l Conf. on Computer Design*, Oct 1999.

[11] 木村, 尾形, 岡本, 笠原. シングルチップマルチプロセッサ上での近細粒度並列処理. 情報処理学会論文誌, Vol. 40, No. 5, pp. 1924–1934, May 1999.

[12] 笠原. マルチプロセッサシステム上での近細粒度並列処理. 情報処理, Vol. 37, No. 7, pp. 651–661, Jul 1996.

[13] Padua and Wolfe. Advanced compiler optimization for super computers. *C.ACM*, Vol. 29, No. 12, pp. 1184–1201, 1996.

[14] 笠原, 合田, 吉田, 岡本, 本多. Fortran マクロデータフロー処理のマクロタスク生成手法. 信学論, Vol. J75-D-I, No. 8, pp. 511–525, 1992.

[15] 本多, 合田, 岡本, 笠原. Fortran プログラム粗粒度タスクの oscar における並列実 行方式. 信学論（D-I）, Vol. J75-D-I, No. 8, pp. 526–535, 1992.

[16] Kasahara, Honda, and Narita. A multigrain parallelizing compilation scheme for oscar. In *Proc.4th Workshop on Lang. and Compilers for Parallel Computing*, Aug 1991.

[17] 本多, 岩田, 笠原. Fortran プログラム粗粒度タスク間の並列性検出法. 信学論 (D-I), Vol. J73-D-I, No. 12, pp. 951–960, 1990.

[18] 笠原. 並列処理技術. コロナ社, 1991.

[19] 吉田, 越塚, 岡本, 笠原. 階層型粗粒度並列処理における同一階層内ループ間データローカライゼーション手法. 情報処理学会論文誌, Vol. 40, No. 5, pp. 2054–2063, May 1999.

[20] 笠原, 尾形等. マルチグレイン並列化コンパイラとそのアーキ テクチャ支援. 信学技報, IDC98-10, CPSY98-10, FTS98-10, pp. 71–76, 1998.

[21] Sun Microelectronics. *UltraSPARC^{TM} User's Manual*. Jul. 1997.