洗練されたメニーコアアーキテクチャの開発

植原 昂 †　　佐藤 真平 †　　三好 健文 †‡　　吉瀬 謙二 †

東京工業大学大学院情報理工学研究科 †　　独立行政法人 科学技術振興機構 ‡

1　はじめに

プロセッサアーキテクチャは，数十以上のコアを 1 チップに搭載するメニーコアへと着実に向かっている．このため，メニーコアにおけるソフトウェアおよびハードウェアの研究・教育が今後ますます重要になる．

メニーコアに関する研究・教育を支援するため，以下のコンセプトを満たすアーキテクチャが要求される．(1) 理解しやすさ，拡張性を備える．(2) 現実のハードウェアに実装できる．(3)RISC などの既存の資源を有効活用する．(4) 小規模で均一のコアを多数並べる．(5) シンプルで効率的なコア間通信機構を備える．(6) 1 チップに搭載するコア数が数百，数千までスケールする．我々の調べた限り，これらを全て満たすアーキテクチャは存在しない．このため，新規に M-Core アーキテクチャ[1] を開発した．本稿では，M-Core アーキテクチャについて述べ，メニーコアの研究・教育に対する実用性を示す．

2　M-Core アーキテクチャ

2.1　アーキテクチャ

図 1 に M-Core アーキテクチャを示す．図 1 中央に示すように，M-Core アーキテクチャは多数のノードをタイル上に配置する．ノードには，図 1(a) の計算ノード，図 1(b) のメモリノード，図 1(c) のパスノード，の 3 種類がある．計算ノードはアプリケーションプログラムを実行し，データを転送するノードである．内部は，コア（プロセッシングエレメント），ノードメモリ（小規模なローカルメモリ），INCC（Inter Node Communication Comtroller，データ送受信ユニット），ルータ（データ転送ユニット）で構成される．メモリノードはメインメモリと接続するノードである．内部は，ページバッファ（メモリアクセスレイテンシを隠蔽するためのバッファ），

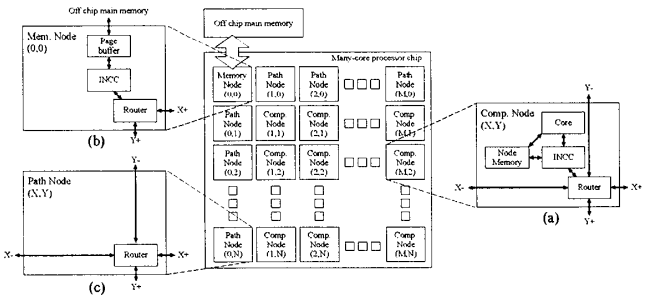

図 1: M-Core アーキテクチャ．

Development of Refined Many-core Architecture
Koh UEHARA†, Shimpei SATO†, Takehumi MIYOSHI†, and Kenji KISE†
†Graduate School of Information Science and Engineering, Tokyo Institute of Technology
‡Japan Science and Technology Agency

INCC，ルータで構成される．パスノードは，主にデータを転送するノードである．内部はルータのみで構成される．コア以外の構成要素は，命令セットに依存しない．このため，様々な命令セットのコアを使用できる．

各ノードは 2 次元メッシュにより接続される．各ノードには X 座標と Y 座標の組み合わせで表現するノード ID を割り当てる．これは，ノード間で通信する際に指定する．ノード間の通信は DMA を利用し，パケット転送方式を採用する．DMA の起動は，コアが INCC に要求を発行することで実現する．計算ノードのコアは自身のノードのノードメモリに対して，ロード/ストアおよび命令フェッチでアクセスする．一方，他のノードのノードメモリおよびメインメモリに対しては，DMA を用いてアクセスする．そして，プログラムの実行に必要な命令やデータは，ノードメモリに転送して処理する．

このように，M-Core アーキテクチャは，シンプルで理解しやすく，拡張性がある．また，DMA を用いることで，効率的なノード間通信を実現する．

2.2　マイクロアーキテクチャ

M-Core アーキテクチャの具体的な設計の 1 つとして，M-Coreα を定義する．まず，ネットワークの仕様について述べる．トポロジは 2 次元メッシュ，ルーティングは XY 次元順ルーティング，パケットの転送方式はワームホール，フロー制御は Xon/Xoff，とする．1 つのパケットは，最大で 10 フリットを格納する．フリットは，6bit の制御情報と，32bit のデータで構成される．通信にはノード ID を用いる．ノード ID は X，Y 座標それぞれを 8bit で表す．従って，搭載可能な最大ノード数は 2^{16} つまり 65,536 となる．

次に，各構成要素の仕様を定義する．コアは，MIPS32 のシングルサイクルプロセッサとする．ノードメモリは，典型的には 512KB の容量を持つ．構成をシンプルにするため，ノードメモリに対するキャッシュは備えない．INCC のマイクロアーキテクチャを図 2 左に示す．図の Core-INCC interface は，メモリマップされたレジスタである．DMA を発行する際に，コアはこのレジスタに対して要求をストアする．要求が発行されると，INCC はノードメモリからデータを読み出し，フリットに加工してルータに転送する．ここで，フリットはパケットを 1

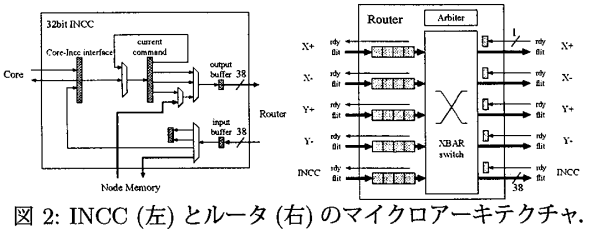

図 2: INCC (左) とルータ (右) のマイクロアーキテクチャ．

サイクルの転送単位に分割したものである．一方，INCCはルータからフリットを受信すると，データをノードメモリに書き込む．ルータのマイクロアーキテクチャを図2右に示す．受信したフリットは入力バッファに格納される．入力バッファに格納されたフリットは，Arbiterに調停され，XBAR switchを通り，適切な方向に出力される．フロー制御のため，1bitのシグナルを隣接するルータ間で互いに通信する．メインメモリは，64bitのアドレス空間があり，アクセスレイテンシは40サイクルとする．ページバッファは，4KBの容量があり，アクセスレイテンシは1サイクルとする．

このように，M-Coreαはスケーラビリティがあり，MIPSを採用することにより，RISCの資源を有効活用している．さらに，現実的なハードウェアに実装することを考慮に入れて定義されている．

2.3 API (Application Programing Interface)

メニーコアにおける並列プログラムの教育・研究を支援するため，APIとしてMClib，MPIMCの2種類を提供する．MClibはM-Coreアーキテクチャで動作する並列プログラムを開発する上で基本的な関数群から成る．これを用いることで，プログラムをC言語で記述することができ，初心者が容易に並列プログラムを開発できる．

MPIMCは，MPIを用いて実装した並列プログラムが動作するように実装したMPIのサブセットである．内部でMClibを利用している．現在はNas Parallel Benchmarks (NPB) で使用する関数を実装している．

3 ソフトウェア評価環境

メニーコアにおけるアーキテクチャおよびソフトウェアの評価を支援するため，シミュレータSimMcを開発している．SimMcは可読性と拡張性を第一の指針として設計している．また，サイクルレベルの評価精度を備えており，実行サイクル数を評価できる．主な機能として，計算ノードのコアによる出力機能，DMAの詳細な履歴の表示機能，ネットワークトラフィックの可視化機能，マルチバイナリ機能，を提供する．

図3に，SimMcによる評価の例として，M-CoreαにおけるNPBの速度向上率を示す．図より，1,000ノードのシミュレーションも可能であることがわかる．また，計算結果が正しいこと，および一般的な計算機環境における評価結果と同様の傾向が見られることから，シミュレーションは正確である．

4 M-Coreアーキテクチャの実用性

研究に対するM-Coreアーキテクチャの実用性を示すため，これを利用している実研究を述べる．[2]では，M-Coreアーキテクチャのルータを拡張することで，複数のコアの多重実行を支援し，ネットワークのバンド幅向上およびディペンダビリティ向上を目指す．[3]では，データ供給支援コアを提案している．これは，他のコアに対してメインメモリの代わりにデータを供給する専用のコアである．データ供給支援コアを利用することで，メモリアクセスレイテンシの削減，およびメインメモリ

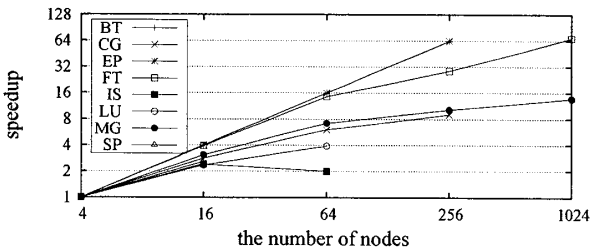

図3: M-Coreαにおける NPB の性能向上率．

のアクセス回数の削減を達成する．ここでは，M-Coreアーキテクチャを評価対象のアーキテクチャとして利用している．[4]では，FPGAを用いた，メニーコアの高速なシミュレーション環境を提案している．ここでは，シミュレーション対象のメニーコアアーキテクチャとしてM-Coreアーキテクチャを利用している．

次に，教育に対するM-Coreアーキテクチャの実用性を示すため，以下の実験をおこなった．学部および修士の学生に対し，M-Coreアーキテクチャ，APIを用いたプログラム開発，SimMcによるシミュレーション方法，を解説した．その後，行列積を求める逐次プログラムを並列化し高速化せよ，という課題を与えた結果，42人からレポートの提出があった．集計結果によると，解答に要した時間は約9時間で，16ノード実行時に平均で4,5倍の性能向上を達成していることがわかった．他との比較はおこなっていないが，アーキテクチャの理解およびプログラムの並列化による高速化を，十分に短い期間の中で達成したと考えられる．

これらの実績から，メニーコアの研究・教育において，M-Coreアーキテクチャは実用的であることがわかる．

5 おわりに

本稿では，メニーコアの研究・教育を支援するため，シンプルで実用的なM-Coreアーキテクチャを開発した．さらに，これをベースとするマイクロアーキテクチャM-Coreαを定義し，APIを開発した．

そして，M-Coreアーキテクチャで，実用的なベンチマークが1,000コアまで動作することを示した．また，メニーコアの研究・教育に適用した例を示し，M-Coreアーキテクチャの実用性を明らかにした．

参考文献

[1] Koh UEHARA, et al. A Study of an Infrastructure for Research and Development of Many-Core Processors. UPDAS, 2009.

[2] 佐藤真平，他．メニーコアプロセッサのオンチップネットワーク性能を向上させるSmartCoreシステム．SACSIS2009論文集．

[3] Yosuke Mori, et al. The Cache-Core Architecture to Enhance the Memory Performance on Multi-Core Processors. UPDAS, 2009.

[4] 高前田伸也，他．メニーコアアーキテクチャ研究のためのスケーラブルなHW評価環境ScalableCoreシステム．情処研報 2009-ARC-185.