ScalableCore システムの挙動検証 ～ソフトウェアシミュレータと比較して～

笹河良介[†] 高前田伸也[†,††]
藤枝直輝[†] 吉瀬謙二[†]

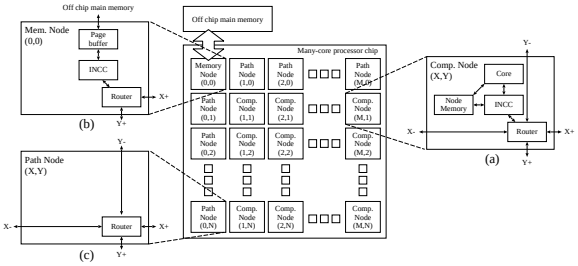
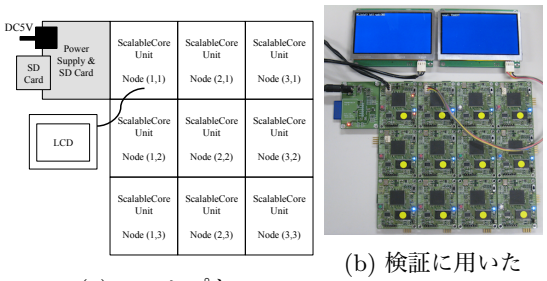


図 1 M-Core アーキテクチャ



(a) コンセプト　(b) 検証に用いた実機

図 2 ScalableCore システム 2.0．FPGA 基板の ScalableCore Unit をタイル状に配置して構成する．

1. はじめに

本稿では，均一なコアを多数搭載するシンプルなメニーコアアーキテクチャである M-Core[1] の挙動をシミュレートする ScalableCore システム[2),3)] において，ソフトウェアによるシミュレータ SimMc[1] と同様のシミュレーション結果を得られるものを実現する．

SimMc と ScalableCore システムの実機との振る舞いには初期化方式の違いがあるが，本稿で述べる同期機構の導入により，両者のシミュレーションの振る舞いを一致させることを可能にする．これを拡張可能性を保ったまま実現することにより，ScalableCore システムの正当性と，優れた機能性を示す．

2. 対象アーキテクチャとそのソフトウェアシミュレータ

議論の対象とする ScalableCore システムは，タイル型のメニーコアアーキテクチャを主なシミュレーション対象とするシステムである．本稿ではシミュレーション対象として M-Core[1] を用いる．**図 1** にその全体構成を示す．M-Core はタイル状に配置されたノードをメッシュネットワークで接続する構成を採用しており，ノード数に対する高いスケーラビリティの実現を目指している．各ノードは DMA 転送を用いて明示的にデータのやりとりを行う．

M-Core アーキテクチャのソフトウェアシミュレータとして SimMc[1] がある．本稿における挙動の検証には SimMc を用いる．また，計算ノードのみを持つ M-Core の構成をその対象とする．SimMc には，コアには MIPS32 ベースのシングルサイクルプロセッサが採用されており，コアに関しては命令レベルのシミュレーションが可能である．INCC (オンチップネットワークへのインターフェース) とルータはサイクルレベルの実装がされているため，ネットワークに関してはサイクルレベルのシミュレーションが可能である．また，既にプログラムが各ノードに配られた状態でシミュレーションを開始する．

3. ScalableCore システム

図 2 に ScalableCore システム 2.0[3] のコンセプトと検証に用いた実機とを示す．ScalableCore システムは，多数の FPGA をタイル状に配置しメニーコアアーキテクチャのシミュレーションシステムを構築する．本稿では ScalableCore システム 2.0 を M-Core アーキテクチャのシミュレーション環境として用いる．

ScalableCore システムの主な構成要素は，シミュレーション対象の一部分に対応するシミュレーションノード **ScalableCore Unit**(以下 Unit と略す) である．隣接する Unit の I/O ポートをそれぞれ接続する．隣接 Unit 間では適宜同期をとりながらシミュレーションを進める．各 Unit には FPGA(Xilinx Spartan-3E XC3S1200E) および 512KB の SRAM などを搭載する．本システムの各 Unit には計算ノード 1 つとシステム制御機構を実装し，シミュレーション対象のノード数に応じて接続する Unit 数を調整する．ScalableCore にはシミュレーション対象アーキテクチャとのインターフェースとして仮想サイクル[2] と呼ばれる仕組みを持っており，シミュレーション対象アーキテクチャの 1 サイクル（1 仮想サイクル）を複数の物理サイクルに分割してシミュレーションを行う．

† 東京工業大学 大学院情報理工学研究科
†† 日本学術振興会 特別研究員

表 1 ハードウェアによる同期機構を追加した場合のそれぞれのシミュレーションサイクル数

Application	SimMc				ScalableCore (Hardware Sync)				Differential Rate			
	10x1	5x2	3x3	2x2	10x1	5x2	3x3	2x2	10x1	5x2	3x3	2x2
do nothing	35	35	35	35	35	35	35	35	0.000%	0.000%	0.000%	0.000%
bottom-right to Node(1,1)	105	101	101	97	105	101	101	97	0.000%	0.000%	0.000%	0.000%
mm_canon: 60	-	-	356531	648233	-	-	356531	648233	-	-	0.000%	0.000%
mm_canon: 120	-	-	2329107	4713157	-	-	2329107	4713157	-	-	0.000%	0.000%
mm_canon: 180	-	-	7357734	15439247	-	-	7357734	15439247	-	-	0.000%	0.000%
random traffic	1191822	879480	741934	688353	1191822	879480	741934	688353	0.000%	0.000%	0.000%	0.000%

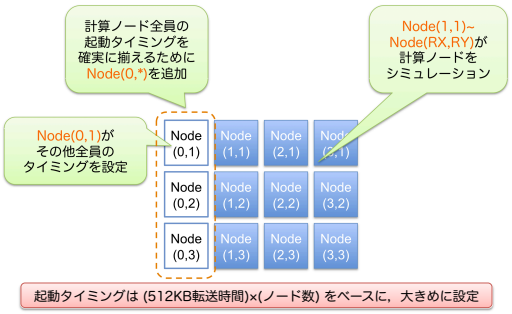


図 3 ハードウェアによるプログラム実行開始の同期

4. ScalableCore システム実機と SimMc とのシミュレーションサイクル数の比較

ScalableCore システムと SimMc は，シミュレーション用プログラムのメモリへの設定方式や各ノードのプログラム実行開始のタイミングの点で異なる．したがって，ScalableCore システムの RTL と SimMc の両者の実装が等価であっても，その実行サイクル数などのシミュレーションの振る舞いは異なるものとなってしまう．本検証では，初期化方式以外で RTL と SimMc との振る舞いがシミュレーションサイクルレベルで等価であることを検証，確認したものを用いて，両環境においてシミュレーションサイクル数を揃えるために，ScalableCore システムにプログラム実行開始タイミングの同期機構を追加する．**図 3** にその様子を示す．

まず，従来のシステムの左端に，図 3 中の Node(0,1)〜Node(0,3) に相当する，1 列の新たなノードを追加する．Node(0,1) はプログラムの配布後，各ノードにプログラムの実行を開始する仮想サイクル数をネットワークを介して通知する．Node(0,1) はすべてのノードに対するプログラムの配布等の初期化が完了する時刻を計算し，適当なマージンを加えることで初期化の完了以降に実行開始時刻が設定されるようにする．初期化に要する仮想サイクル数は

$$\text{マージン} \times \text{ローカルメモリサイズ}\ [bit] \times \text{ノード数}/\text{ネットワークバンド幅}\ [bit]$$

により求められる．すべてのノードは設定された時刻になるとプログラムの実行を自律的に開始する．

上記のハードウェアによる同期機構を追加した ScalableCore システムの実機に対して評価を行う．ScalableCore システムの実機と SimMc とで同一のアプリケーションをそれぞれ実行し，シミュレーションサイクル数を測定する．

シミュレーションサイクル数の測定範囲は，計算ノードの実行開始時刻から，Node(1,1) が全計算ノードからの実行完了を確認するまでとする．両環境共に，10×1，5×2，3×3，2×2 の 4 つのノード構成を用い．以下のアプリケーションで測定を行った．

- do nothing: 何もしないプログラム
- bottom-right to Node(1,1): 右下のノードから左上のノードへの DMA 転送
- mm_cannon: 行列積．行列サイズは 60〜180 の 3 種
- random traffic: ランダム通信

表 1 にハードウェアによる同期機構を追加した場合の ScalableCore システムと SimMc のシミュレーションサイクル数とその誤差率を示す．測定の結果，4 種すべてのアプリケーションでシミュレーションサイクル数が一致した．このようにプログラム実行開始時刻の同期機構の追加により，ScalableCore システムにおいてもソフトウェアシミュレータと同じ開始条件での実行結果が得られるようになった．

5. ま と め

本稿では多数の FPGA を用いたタイル型メニーコアアーキテクチャ向けシミュレーション環境 ScalableCore システムの挙動検証を行った．実機においてのシミュレーションサイクル数の差異を取り除くために，ハードウェアによる同期機構を実装し，評価を行った．Unit と同期機構の追加によりソフトウェアシミュレータとのシミュレーションサイクル数の差異を取り除き，ソフトウェアシミュレータと同じ振る舞い，シミュレーション結果を得ることが可能になった．

また，デモでは 10×10 ノード程度の ScalableCore システムの展示とデモンストレーションを行う予定である．

謝　辞

本研究の一部は，科学技術振興機構・戦略的創造研究推進事業 (CREST) の「アーキテクチャと形式的検証の協調による超ディペンダブル VLSI」の支援による．

参 考 文 献

1) 植原昂, 佐藤真平, 吉瀬謙二: メニーコアプロセッサの研究・教育を支援する実用的な基盤環境, 電子情報通信学会 システム開発論文特集号, Vol. J93-D, No. 10, pp. 2042–2057 (2010).
2) 高前田伸也, 佐藤真平, 藤枝直輝, 三好健文, 吉瀬謙二: メニーコアアーキテクチャの HW 評価環境 ScalableCore システム, 情報処理学会論文誌 コンピューティングシステム (2010).
3) 坂口嘉一, 高前田伸也, 吉瀬謙二: ScalableCore システム 2.0 の実装と評価, 電子情報通信学会研究報告 RECONF 2010-09 (2010).