2E−1

マルチプロセッサシステムのゲートレベルシミュレーション環境の構築と評価

†佐伯賢治　‡佐々木敬泰　‡弘中哲夫　‡藤野清次
E-mail: kenji@csys.ce.hiroshima-cu.ac.jp
†広島市立大学 情報科学部 情報工学科
‡広島市立大学大学院 情報科学研究科 情報工学専攻

1　はじめに

オンチップマルチプロセッサなどに代表される大規模システムLSIでは，LSI内部にバグが潜みやすく，設計の後戻りのために開発期間が伸びてしまうという問題がある．このような問題を解消するためには，以下のような用件を満たしたシミュレーションが望ましい．

- 機能レベルからゲートレベルまでの多レベルのシミュレーションが可能，
- デバッグの為の情報抽出が容易，
- 設計変更などに対する柔軟性が高い．

上記の点を満たすシミュレーション手法として，HDL(Hardware Description Language)設計におけるシミュレーションがある．ただし，HDLでのゲートレベルシミュレーションは，CPU速度・メモリ容量などのコンピュータの能力的な問題により，従来は，一部の高性能なスーパコンピュータや専用計算機など以外では実用的な時間での実行が困難であった．

本研究は，近年高速化してきたワークステーションにおいて大規模システムのHDLシミュレーションの実用性を再評価することを目的する．評価用大規模システムとしてR3000互換プロセッサ16台構成のマルチプロセッサシステムのシミュレーション環境を構築した．そして，構築された環境においてシミュレーションを実際に行い，その実用性について評価を行なった．

2　シミュレーション手法の比較

これまでシステム単位でのシミュレーションにおいて主流であった手法は，(1)C言語等で記述されたソフトウェアでのシミュレーション，(2)論理シミュレーション用の専用ハードウェアや専用計算機を用いた方法，(3)設計データをシミュレーション専用言語に変換した後，シミュレーション専用ソフトウェアでシミュレーションを実行する方法，など3つがあげられる．それらに加え近年では(4)Verilog-HDL，VHDL等のHDLを用いて設計し，機能レベルからゲートレベルまでのシミュレーションを同一のHDLで行なう，といった方法が使われるようになってきた．(4)の手法は表1で示すように先にあげた(1)〜(3)の手法に比べ，第1節で示した要件をもっとも良く満たしており，このことから大規模なシステム設計にもっとも適した手法であると言える．ただし，大規模なシミュレーションに必要な時間のみに注目すると(3)のシミュレーション専用言語と(4)のHDL設計は，もっとも計算時間のかかる手法であり，大規模なシステムのシミュレーションを行なう上で障害になる．そこで本研究では実際に大規模システムにおいて現実的な時間でシミュレーションが可能か調査した．

表1: 各シミュレーション手法の特徴

シミュレーション手法	複数レベルへ対応	情報抽出	柔軟性
(1)C言語等	×	△	△
(2)ハードウェア	○	×	×
(3)専用言語	×	○	○
(4)HDL設計	○	○	○

3　評価対象とするアーキテクチャ

実用性を評価するために，マルチプロセッサシステムのゲートレベルでのシミュレーション環境を構築した．評価対象システムは以下のような特徴を持つ．

- **要素プロセッサ1〜16個構成：**要素プロセッサはR3000互換の命令セットのサブセットを実装した32ビットパイプラインプロセッサである．
- **SSH(Scheduling Support Hardware)を実装：**現在，我々の研究室で研究が進められているもので，プロセッサへのタスクのスケジューリングを支援するハードウェアである．

Implementation of gate level multiprocessor simulation and its evaluation
Kenji SAEKI, Takahiro SASAKI, Tetsuo HIRONAKA, and Seiji FUJINO
Hiroshima City University
3-4-1 Ozuka-Higashi, Asa-Minami-Ku, Hiroshima 731-3149

表 2: プロセッサ 1 台当たりの規模

	合成前		合成後	
	CPU	SSH	CPU	SSH
Verilog の行数 (行)	5,467	342	14,855	1,477
ゲート数 (ゲート)	−		15,082	1,815

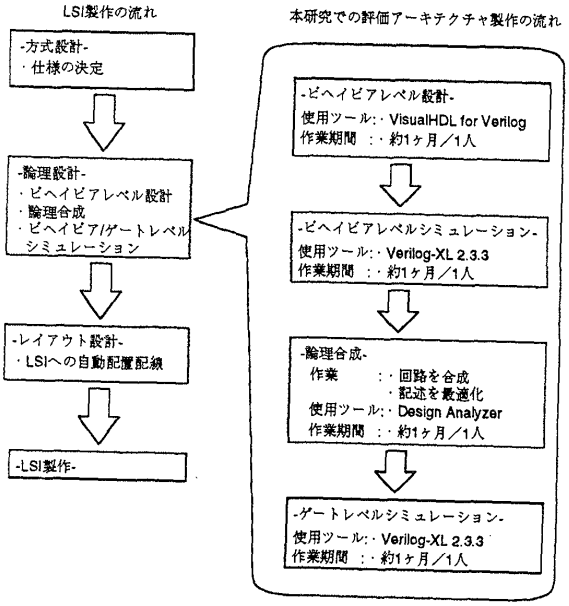

図 1: 設計の流れ

- **GMB 同期機構 [1] を実装：**現在，我々の研究室で研究で開発した，バリア同期方式である．特に同期グループ管理能力にすぐれた同期手法である．

上記の評価対象システムは，Verilog-HDL を用いて機能レベルで設計し，論理合成ツール (design_anaylzer) と VDEC で作成した日立 0.35um LSI 用セルライブラリを用いて論理合成を行なった．製作した評価アーキテクチャのプロセッサ 1 台当たりの規模を表 2に示す．評価対象システムの製作工程および，それぞれの過程においてかかった期間，使用したツールに加えて，評価対象システムの製作工程が LSI 設計の流れのどの部分に当たるかを図 1に示す．

4 評価

シミュレーションは Cadence 社の Verilog-XL を用いて SUN Ultra Enterprise 450 上で行なった．表 3に SUN Ultra Enterprise 450 の仕様を示す．

マルチプロセッサについて論じる前に，プロセッサが 1 台の場合のシミュレーションを行ない，ゲートレベルシミュレーション自体が現実的な時間で実行可能なレベルにあるかを評価する．

シミュレーションにはテストプログラムとして LU 分解を用い，テストプログラムの規模 (LU 分解する行列の元の数) に対するシミュレーション時間の変化

表 3: Ultra Enterprise 450 の仕様

CPU	UltraSPARC-II(300MHz) × 2	
キャッシュ	オンチップ キャッシュ	データ: 16 KB 命令: 16 KB
	外部	2 MB
メモリ		512 MB

表 4: プロセッサが 1 台の時の行列の元と実行時間

元の数	各レベルでの実行時間 (sec)	
	ビヘイビア	ゲート
2	2	24
4	12	142
8	79	938
16	615	7,166
32	4,799	57,329

を (1) 機能レベルシミュレーション，(2) ゲートレベルシミュレーションのそれぞれの場合において測定した．結果を表 4に示す．

詳細については紙面の都合で割愛するがマルチプロセッサ構成でのシミュレーションも行ない小規模のプログラムにおいては現実的な時間で実行されたことが確認された．

以上のシミュレーションの結果より，適切な規模のプログラムを選べば現実的な時間で実行できることが分かった．

5 まとめ

本研究では，マルチプロセッサのゲートレベルシミュレーション環境を構築した．また，それを用いて実際にシミュレーションを行なった．その結果より，適切な規模のプログラムを選べば，現実的な時間で実行可能であることが分かった．今後の課題として，シミュレーションのさらなる高速化を実現する手法を検討する予定である．

謝辞

本研究に際し，協力いただいた (株) セイコーインスツルメンツに深い感謝を表します．なお本研究は，東京大学大規模集積システム設計教育研究センターを通し，(株) 日立製作所の協力で行われたものである．

参考文献

[1] 土江竜雄，弘中哲夫：半順序関係にある同期グループ間の疑似依存を解消したバリア同期機構 LSI，信学技法，1998

[2] Gerry Kane 著，前川 守監訳：MIPS RISC アーキテクチャ −R2000/R3000−，共立出版，1992