プロセッサの形式的検証の一手法

研究ノート

石　浦　菜岐佐*

Formal Verification of Processors

Key Words : Processors, verification, logic clesign, model checking

1．は　じ　め　に

本稿の原稿用紙を頂いた折も折，「ペンティアムプロセッサ・バグ」事件が新聞にまで取り上げられ，私の周辺の人間の大きな話題になった．「ペンティアム」はプロセッサの世界シェアNo.1を誇るインテル社の最新鋭プロセッサだが，その倍精度浮動小数点除算の特定パタンで誤りを生じるというバグが見つかったという．

プロセッサの性能競走は熾烈をきわめ，精巧なアーキテクチャや実装技術により，性能を限界まで引き出そうとする試みがなされる[2]．このようなプロセッサの設計では，正常時の動作はいうまでもなく，例外や割り込み時の処理が極めて複雑となり，設計誤りが発生しやすくなる．

このような設計誤りを製造以前に検出するための設計検証は，現在，膨大なテストケースに対する論理シミュレーション（論理レベルのシミュレーション）によって行われている．しかし，これもある意味では，「海から汲み上げたコップ一杯の水を調べている」程度に過ぎず，完全を保証するには程遠い．プロセッサの複雑化につれ，論理シミュレーションをすり抜ける設計誤りの存在が危惧されるようになっているが，今回の騒動は我々にこの問題を再認識させるものといえよう．

論理シミュレーションを補うものとして研究されているのが，ハードウェアの形式的検証法である．これは，何らかの数学的手法により，設計されたハードウェアがあらゆる可能な状態において正しく動作することを示そうとするものである．これまでにも様々な手法が考案されてきたが，本質的に計算量が膨大で，プロセッサ規模のハードウェアへの適用は困難と考えられていた．

以下本稿では，著者が文部省在外研究員としてカルフォルニア大学バークレイ校滞在中に行った，プロセッサの形式検証の研究を紹介させて頂く．

2．ハードウェアの形式的検証法

ハードウェアの形式的検証法は，1）定理証明に基づく方法と 2）モデルベースの方法に大別される．前者は研究も古く実績もあるが，設計の詳細と定理証明系の詳細の両方に関する知識が必要で，証明にも長い期間（中規模のプロセッサでは１年以上）が必要となる．後者は設計対象を抽象機械などの数学的モデルととらえ，そのモデルが所望の仕様を満たすことを証明する方法であり，膨大な計算量を必要とするが，人手の介入を要しない検証が可能である．近年，計算手法の発展によりこのアプローチが急速に注目を集めている，本稿の手法はこのモデルベースの方法に基づくものである．

* Nagisa ISHIURA
1961年２月21日生
昭和61年京都大学大学院工学研究科修士課程情報工学専攻修了
現在，大阪大学工学部，情報システム工学科，助教授，工学博士，VLSIの計算機・援用設計
TEL 06－879－7806(直通)

プロセッサは有限状態機械（FSM）としてモデル化できる．FSMを対象とするモデルベースの検証法の一つに，「時相論理モデル検査法」[4]が知られている．これは，仕様を時相論理（命題論理に時間の概念を追加した論理）で記述し，与えられたFSMがこれを満たすかどうかを判定するというものである．

アルゴリズムの飛躍的な発展[5]により，2^{70}状態程度の状態機械が扱えるようになり，この方法は現在最も注目されている．しかし，プロセッサ全体の検証を行うには，次の点が大きな問題となっていた．

(1) 仕様の記述：

パイプライン，分岐，割り込みなどの複雑な制御を含むプロセッサの全仕様を時相論理で記述することは困難である，

(2) 計算量

レジスタファイル，パイプラインレジスタなどを含めると，プロセッサの状態数は許容限度を遙かに上回る．また，乗算器などの複雑な演算器は計算量を指数的に増大させる．

3．比較と記号モデル検査法による検証

本稿の検証法は，時相論理モデル検査法を利用するものであり，前節の問題点(1)，(2)をそれぞれ次により解決する．

(1) 比較による検証：

仕様は時相論理ではなくリファレンス・プロセッサ（RP）により与え，設計したプロセッサがあらゆる状況においてRPと同じ振舞いをするかどうかを検証する．

(2) 抽象化：

比較における等価性が保存される範囲でメモリー，演算器をより簡単なものに置き換える．

図1に「比較による検証」の概念を示す．IPは検証の対象となるプロセッサである．RPはリファレンスプロセッサで，1命令を1クロックで実行し，パイプラインなどの高速化機構を持たない単純なプロセッサを意味する．与えられた命令セットをそのまま状態機械として実現したものと考えればよい．2つのプロセッサの出力は比較器で比較され，両者の出力が等価なときのみ1が出力される．検証は，比較器の出力okが常に1であるかどうか（時相論理式で表すと$AGok$かどうか）を判定することに帰着される．

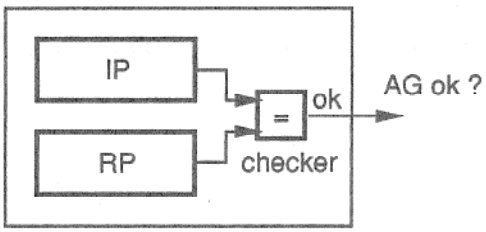

図1　比較による検証の概念

ただし，ここでの出力の等価性は信号値系列の単純な等価性ではない．パイプラインなどのアーキテクチャをとるIPと単純なRPでは動作のタイミングが異なるので，これを考慮した等価性（図2参照）とその判定法を定義する必要がある（詳細は[3]参照）．

図2　等価と考えるべき信号値系列．

図3はここでいう「抽象化」の具体的手法を示したものである．×(IP)，×(RP)はそれぞれIP，RP内の演算器であるが，これを一つの共通のユニットに置き換える．このユニットは，演算を行うのではなく非決定的な系列を出力するが，IPとRPの対応する演算器のオペランドが等しいときには，対応する出力に等しい値を出すというものである．この置き換えにより，実際の値としての情報は失われるが，IPとRP

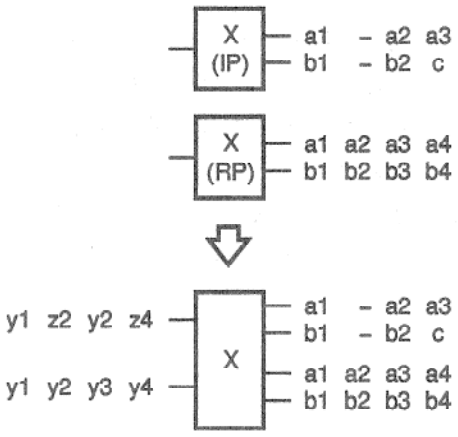

図3　メモリー，演算器の置換（抽象化）．

の等価性の情報は残されることになり，比較に基づく検証の枠組みにおいては，精度を落とすことなくモデルサイズを削減することができる．なお，実際には検証の精度が失われないための必要十分条件に関して詳細な議論が必要となる（詳細は[3]）．

4．実　験　例

上記の検証法をDLXプロセッサ（文献[2]の教科書的プロセッサ）のサブセットに適用する実験を行なった．プロセッサの命令セットを表1に示す，演算器の抽象化を行なったため，2項演算はすべてALU命令で代表されている．プロセッサは標準的な5段のパイプライン構成を持ち，パイプラインのストールを防止するバイパス回路を備えている．

表1　実験に用いたプロセッサの命令セット．

op code		function
ALU	op, r_1, r_2, r_w	pc<-pc+4, R[r_w]<-R[r_1]op[r_2],
ALUI	op, r, imm, r_w	pc<-pc+4, R[r_w]<-R[r_1]op imm,
LOAD	r, adr	pc<-pc+4, R[r]<-M[adr],
STORE	r, adr	pc<-pc+4, M[adr]<-R[r],
BEQZ	$r, dspl$	if (R[r]==0) pc<-pc+4+$dspl$ else pc<-pc+4,
BNEZ	$r, dspl$	if (R[r]!=0) pc<-pc+4+$dspl$ else pc<-pc+4,
J	$adrs$	pc<-$adrs$,
JAL	$adrs$	pc<-$adrs$, R[1]<-pc+4,
JR	$adrs$	pc<-R[r],
JALR	$adrs$	pc<-R[r], R[1]<-pc+4,

時相論理モデル検査のツールとしてはカーネギーメロン大学で開発されたSMV[5]（DEC-station 5000/260上にC言語で実装されている）を用いた．

抽象化の適用と，さらなる簡単化により，IP，RP，比較器からなるモデル全体の状態数は2^{56}まで削減できた．検証の結果，プログラムカウンタの書き込みの際の優先順位の設定ミス，ロード命令によりロードしたデータを直後の命令で使う場合にはバイパス論理が正しく動作しない，等の設計誤りを検出することができた．これらの検証に必要な記憶領域は40MB，計算時間は16時間程度であった．

5．お　わ　り　に

プロセッサの形式的検証の一手法を紹介し，小規模プロセッサに適用可能であることを示した．しかし，実用に供するにはまだまだ計算量が大きすぎるといえる．本研究とほぼ同時期に，MITやスタンフォード大学では，プロセッサの構造や動作原理の情報をアドホックに利用することにより，時相論理モデル検査よりもさらに計算量の小さなアルゴリズムで検証を行なうという方法が提案されている[1]．当面は，このようにとにかく計算量を減らして実用を目指すという方向に研究が進むものと思われる．

6．参　考　文　献

1) J. Burch and D. Dill：“Automatic verification of pipelined microprocessor control”, in *Proc. Int. Conf. Computer-Aided Verification* (June 1994).

2) J. L. Hennessy and D. A. Patterson：*Computer Architecture：A Quantitative Approach*, Morgan Kaufmann Publishers, CA, USA (1990).

3) N. Ishiura and R. K. Brayton：“A Comparative approach to processor verification using syhmbolic model checking”, *Memorandum*, UCB/ERL M94/59, University of California at Berkeley (Aug. 1994).

4) D. E. Long：“Model checking, abstraction and compositional verification”, CMUCS-93-178, School of Computer Science, Carnegie Mellon University (Ph. D. Dissertation), (June 1993).

5) K. McMillan：*Symbolic Model Checking*, Kluwer Academic Publishers (1993).