# パイプライン制御を対象としたテストプログラム自動生成

1H－7

岩下 洋哲　中田 恒夫　広瀬 文保
富士通研究所

## 1　はじめに

マイクロプロセッサにおけるパイプラインの複雑化は，制御回路の設計誤りを増加させるとともに，その検証をより困難なものにしている．マイクロプロセッサの論理検証では人手で作成したテストプログラムを用いることが多いのが現状であるが，テストプログラムの開発効率と検証の信頼性を高めるために，我々は効果的なテストプログラムを自動生成する手法に関して研究を行なっている．

我々はこれまでに，簡単なパイプライン制御を対象としたテストプログラム自動生成システムを開発した[1, 2]．ここでは，より現実的なパイプライン構成のプロセッサを取り扱うための，システムの拡張について報告する．

## 2　パイプラインモデル

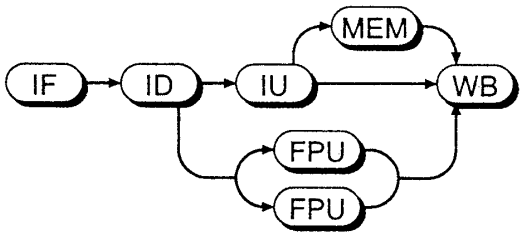
我々の従来のシステムは，基本的なシングルパイプラインのプロセッサを対象としていた．ここでは，現実の多くのRISCプロセッサで採用されているin-order-issue, out-of-order completionのパイプラインが取り扱えるよう，モデルの拡張を行なう．ここで対象とするパイプラインは次のようなものである．

- 各命令が実行ステージに入る順序はプログラム順．
- 使用するハードウェアユニットの種類は命令毎に任意．
- 各ステージのレイテンシは命令毎に任意．
- 命令実行ユニットを複数持ち，同時実行が可能．
- 各命令が実行を完了する順序は任意．

対象とするパイプラインの一例を図1に示す．

## 3　パイプライン動作の仕様

ここでは，プロセッサのすべての命令についての処理実行の手順を記述したものをパイプライン動作の仕様とする．これは，具体的には時刻毎のハードウェア資源の占有およびレジスタやメモリ等のデータ読み書きの動作を記述したものである．図1のパイプラインを持つプロセッサの仕様の例を図2に示す．ここでは簡単のために仕様の一部のみを示している．IF, ID, IU, FPU, MEM, WBはその時刻にそれぞれのパイプラインステージを占有することを表し，GPR/RおよびGPR/Wはそれぞれその時刻に汎用レジスタのデータを読み出すことおよび書き込むことを表す．




IF　命令フェッチ
ID　命令デコード／レジスタフェッチ
IU　整数・論理演算
FPU　浮動小数点演算
MEM　メモリアクセス
WB　ライトバック

図1　対象とするパイプラインの例

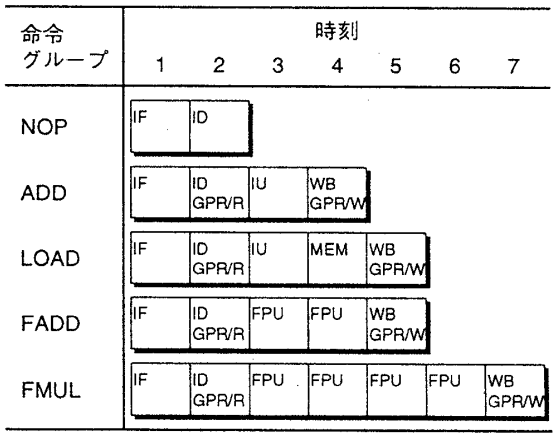



図2　仕様の例

Automatic Test Program Generation for Pipeline Controllers
Hiroaki Iwashita, Tsuneo Nakata, and Fumiyasu Hirose
Fujitsu Laboratories Ltd.
1015 Kamikodanaka, Nakahara-ku, Kawasaki 211, Japan

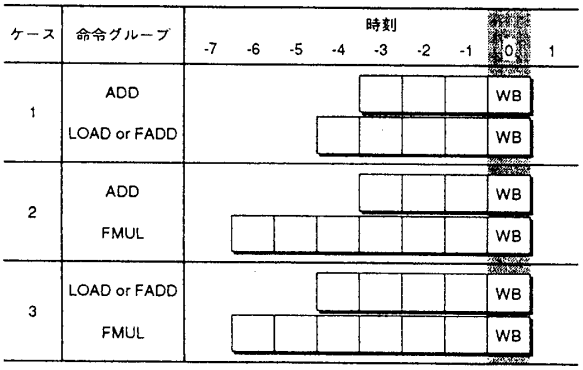



図3　WB の構造ハザードが発生するケース

## 4　テストプログラム生成

本手法では，パイプラインハザードを起こす命令列をプロセッサに与えることにより，パイプライン制御機構の動作をテストする．テストプログラムは次の手順で自動生成する．

1. ハザードの発生ケースを列挙する．
2. それぞれのハザード発生ケースに至るための命令列の条件を求める．
3. 条件を満たす命令列を選んでテストプログラムを生成する．

列挙するハザードには，構造ハザード，データハザード，および制御ハザードがある．このうち制御ハザードは，プログラムカウンタにおけるデータハザードとして取り扱う．図2の仕様から WB の構造ハザードが発生するケースを列挙した例を図3に示す．

ハザード発生ケースに至るための命令列を得るには，ハザード発生時刻を基準（時刻0）としたときの各命令の実行開始時刻を求め，それにしたがって命令を並べるとよい．図3のケース3では，LOAD または FADD を時刻 −4 に，FMUL を時刻 −6 に開始すればよいので，命令を FMUL-*-LOAD または FMUL-*-FADD の順で並べる．ここで，* の位置には干渉しない命令（NOP 等）を選ぶものとする．

もし，FPU が1つしかない場合には，ケース3を実現する命令列として FMUL-*-FADD を選ぶことはできない．なぜなら，WB の構造ハザードが発生する以前に FPU の構造ハザードが発生し，どちらかの命令実行が遅らされてしまう結果，WB のハザードは実際には起こらないからである．このように，ハザード発生ケースに至るための命令列の条件を求める際には，他のハザード発生ケースの影響を考慮する必要がある．

## 5　実験結果

ハザード発生ケースを網羅する命令列を自動生成するシステムを，約 1,000 行の Perl プログラムで実現した．SPARC アーキテクチャのモデルを図1のパイプラインで構成し，このシステムを適用した結果を表1に示す．

表1　実験結果

| | |
|---|---|
| 命令グループ数 | 52 |
| ハードウェア資源数 | 13 |
| ハザード発生ケース | 338 |
| 命令グループの組合せ | 4,446 |
| CPU 時間（SS2） | 7.3 秒 |

命令グループは，パイプライン制御の観点から実際の命令セットを分類したものである．ハードウェア資源については，パイプラインハザードを起こす可能性のあるものについて，仕様に記述した．

命令グループの組合せは，ハザード発生ケースに至るための命令列を命令グループの列として表現したときの場合の数を示したものであり，実際の命令の組合せではさらに多くなる．

## 6　おわりに

本稿では，パイプライン制御に対するテストプログラム自動生成システムの，in-order issue, out-of-order completion のパイプラインへの対応について述べた．

我々は，対象とするパイプラインのモデル化とパイプライン仕様として必要な情報の選定を行なった上で，テストプログラム自動生成システムを実現した．

パイプラインの複雑化にともなって増大したテストケースを効率よく表現する必要があった．また，目的のテストを確実に行なうために，無効な命令列を候補から削除する手続きを加えた．

## 参考文献

[1] H. Iwashita, T. Nakata, and F. Hirose, "Behavioral Design and Test Assistance for Pipelined Processors," *First Asian Test Symposium*, pp. 8–13, 1992.

[2] H. Iwashita, T. Nakata, and F. Hirose, "Integrated Design and Test Assistance for Pipeline Controllers," *IEICE Transactions on Information and Systems*, pp. 747–754, vol. 76, no. 7, 1993.