A-03

形式手法 B-Method による簡易 CPU モデルの記述

苫小牧高専　○大西　孝臣、　稲川　清、　阿部　司、　吉村　斎

要　旨

形式手法は、システム構築の首尾一貫性に関する証明可能性や、実現するシステムの機能安全性の観点より注目されている。我々は、本校情報工学科の実験で採用している簡易 CPU モデルを形式手法 B-Method により記述した。

B-Method による記述を通じて、開発ツールにおけるモデルの記述と論理証明、検証ツールにおける認証とアニメーション（シミュレーションに相当）に成功したのと同時に、モデルの記述の証明、認証等に際しての困難性に関する知見を得た。

１．はじめに

形式手法は、システム構築の首尾一貫性に関する証明可能性や、実現するシステムの機能安全性の観点より注目されており、我々は、種々ある形式手法の中より B-Method[1] に着目している。

本稿では、本校情報工学科の実験で採用している簡易 CPU モデルを形式手法 B-Method により記述したので報告する。

２．採用した形式手法の開発ツール・検証ツールについて

本稿における形式手法 B-Method の採用および使用は、本稿謝辞にある研究助成において形式手法の適用を通じてソフトウェア開発に資する事を最終的な目的としており、同時に、将来的には形式手法の本校学生に向けた教育現場への導入を視野に入れる事を目的としている。

そこで、製品開発に向けた実用性の高さと教育現場への導入に向けた低コストを両立できる B-Method のツールとして、仏国 ClearSy 社のソフトウェア開発を目的とした B-Method 用の産業用ツールである Aterier-B[2]、および独国 Heinrich-Heine 大学の B-Method 用の検証ツールに当たるアニメータ･モデルチェッカである ProB[3]を採用した(ここで言う“アニメータ”とは記述モデルを仮想的に実行するシミュレータに相当するツールに対する名称である)。

３．記述対象のモデルについて

我々は本校情報工学科４年学生に対して、ハードウェア記述言語 VHDL による記述およびシミュレーションを行う実習実験を行っており、対象として図１に示すアーキテクチャで構成される簡易 CPU モデル SAP-1(Simple As Possible-1)[4]採用している。我々自身にとっても扱いなれており、学生にも親しみの持てるモデルである SAP-1 を本稿における B-Method による記述の対象にした。

簡易 CPU モデル SAP-1 の主な構成は以下の通りである。

- 8 ビットのデータバス、4 ビットのアドレスバスの時分割多重による W バスと、12 ビットの制御バス
- 命令数は 5 種類
  （アキュムレータ格納・算術加算・算術減算・データ出力（出力レジスタ格納）・実行終了）
- 各命令は 6 クロック固定
  （内、命令フェッチサイクル：3 クロック、命令実行サイクル　　：3 クロック）
- 接続するメモリは 16 ワード×8 ビットの RAM が 1 個
- リセット信号印加により RAM の先頭番地より実行終了命令（HLT）までの命令を順次実行する。従って命令以外のデータは HLT 命令以降の空き RAM 領域に実行直前に格納されているという前提とする。

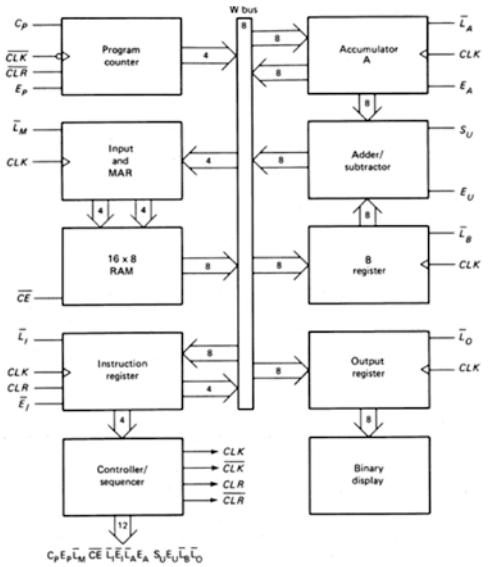

図１　簡易 CPU モデル SAP-1 のアーキテクチャ
（文献[4]より抜粋）

４．モデルの記述について

４－１．記述に際してのモデル記述の条件・モデルの解釈

本稿におけるモデル記述に B-Method を使用するのに際して、２章で述べた B-Method 採用の経緯も踏まえて、記述の条件を以下の通りとした。

- Aterier-B における論理証明を通過する事。
- Aterier-B における BO チェックを通過する事。
  （以上 2 項目について４－２節参照。）
- ProB における認証を通過する事。
- ProB におけるアニメーションが可能である事。
  （以上 2 項目について４－３節参照。）

記述の対象とする簡易 CPU モデル SAP-1 は、形式手法でモデルを記述する視座より解釈すると、シングルプロセスの状態遷移モデルであり、データバス・アドレスバスという“資源”をアキュムレータ・命令レジスタ・プログラムカウンタ・RAM という複数の互いに異質な“利用者”が獲得しようすると競合モデルであると言える。

モデル記述に際しては、W バスによる多重化の解消等の対象モデルの簡単化を行っている一方､VHDL による記述(３章参照）における経緯も踏まえて、本来の文献[4]の仕様には無い、非実行時や想定外時避難用となるアイドル状態やリセット信号印加時のスタンバイ状態といった、状態遷移に関する我々独自の仕様を付加している。結果、次ページ図２に示すように、当 CPU モデルが遷移する全状態数は 20 となった。

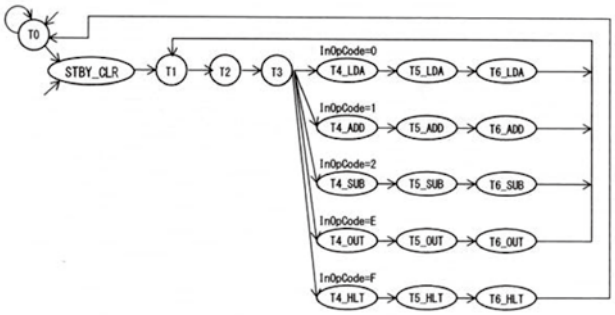

図２　簡易 CPU モデル SAP-1 の状態遷移の概要

４－２．Aterier-B におけるモデル記述・論理証明

B-Method による開発の手順は、システム仕様に対して、まず“抽象機械”コンポーネントによる記述を行い、幾つかの段階を経て“実装”コンポーネントへと“リファインメント”するという形態をとる。抽象機械から実装までの過程において、記述の首尾一貫性・無矛盾性を論理的に証明すべき命題である“証明責務”が開発ツールにより提示されて、開発ツールにおいて自動あるいは手動による証明責務の論理証明を行う事でシステム開発が遂行される。

更に、開発ツールである Aterier-B においては、記述されたモデルと等価の C 言語ソースの出力を可能にさせるために BO チェックを通過する事が求められる。

我々は簡易 CPU モデル SAP-1 に対して Aterier-B を用いた記述を試み、本ページ下部の図 3 に示すコンポーネント（実装機械・実装）の構成を組み上げ、次の図４に示すように、提示された証明責務および BO チェックを通過させる事に成功した。

同時に、各構成要素において並列動作が行われる CPU モデルを Aterier-B を用いて記述する事に際して、1 システムクロック当たりの状態遷移の細分化を要する事が特に実装における証明責務の数量増大を招く事例や、数式の記述に関しての B0 チェックにおける警告を受ける事例等の、モデル記述に際しての困難性に関する知見を得た。

コンポーネント	型チェック	証明責務生成	証明責務	証明済	証明未	B0チェック
ACC	OK	OK	0	0	0	OK
ACC_i	OK	OK	10	10	0	OK
ADDRESSBUS	OK	OK	1	1	0	OK
ADDRESSBUS_i	OK	OK	7	7	0	OK
ALU	OK	OK	0	0	0	OK
ALU_i	OK	OK	47	47	0	OK
BREG	OK	OK	0	0	0	OK
BREG_i	OK	OK	7	7	0	OK
DATABUS	OK	OK	1	1	0	OK

図４　Aterier-B による論理証明・BO チェック（一部）

４－３．ProB における認証・アニメーション

B-Method による抽象機械や実装等のモデル記述に対して、記述したシステムの完備性を担保するために、認証ツール ProB へのロードが成功する事等を通じて、認証を通過する事が求められる。

更に、記述したシステムの実行可能性を担保するために ProB におけるアニメーションが実行可能である事が求められる。

我々は、記述した簡易 CPU モデルにおける総括的な実装コンポーネントである USER_i の ProB へのロードの成功を通じたモデルの認証を確認し、Random Animation 機能を用いて、当モデルにおける簡単な機械語プログラムの実行に関する正しいアニメーション結果を得る事に成功した。

同時に、当モデルに対して ProB におけるより簡便なアニメーション実行の実現を目的として、記述した抽象機械あるいは実装に対するメタなコンポーネント間連携を試みた所、特に実装における証明責務の数量増大を招く事例等の、モデル記述に際しての困難性に関する知見を得た。

５．おわりに

我々は、形式手法 B-Method に着目し、簡易 CPU モデル SAP-1 を B-Method を用いて記述した。

記述を通じて、開発ツールにおけるモデルの記述と論理証明、検証ツールにおける認証とアニメーションに成功したのと同時に、モデルの記述の証明、認証等に際しての困難性に関する知見を得た。

謝辞

本研究を進めるに当たり、平成 26 年度戦略的基盤技術高度化支援事業「農業機械のさらなる高度化と海外進出に資する次世代電子制御ソフトウェア基盤の開発」よりのご支援を頂戴しております。

参考文献および Web ページ

[1] Schneider, “The B-Method: An Introduction”, Palgrave Macmillan, 2001
[2] http://www.atelierb.eu/ja
[3] http://stups.hhu.de/ProB/w/Main_Page
[4] Malvino & Brown, “Digital Computer Electronics”, McGraw-Hill, 1992(3rd. Ed)

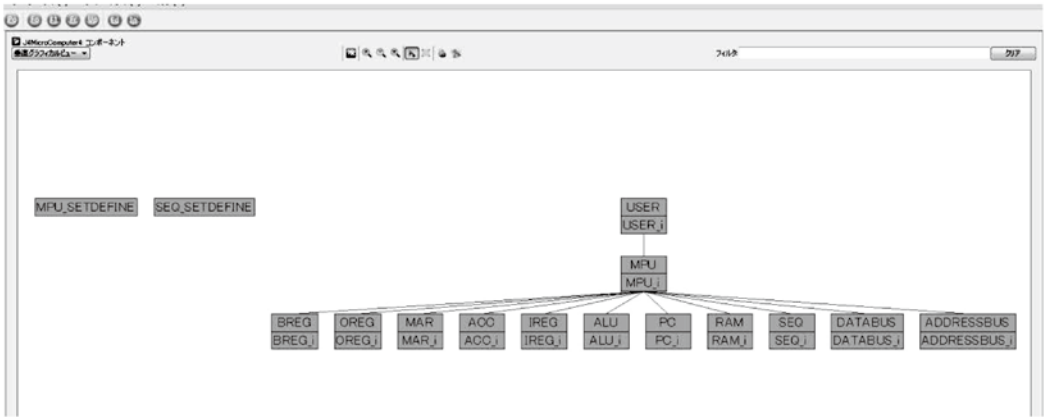

図３　コンポーネント（抽象機械・実装）構成