7P−5

ＯＳ検証用の新コンピュータ・アーキテクチャ・シミュレータ

武内和昭、矢木孝幸、森良哉
（株）東芝　情報処理・機器技術研究所

1.はじめに

従来、新規に計算機を開発する場合、言語処理系、ＯＳなどの検証はターゲットとなる計算機が作成されてから行われた。このため、ターゲットハードウェアがないとＯＳなどの検証が行えないため、計算機の開発効率としてはよいものではなかった。

この問題を解決する方法の１つに、ターゲット計算機をソフトウェアでシミュレートするシミュレータを作成し、有効に使用することである。

今回、我々が開発したＯＳ検証用のシミュレータは、ターゲット計算機提供前に、ＯＳなど従来ターゲット計算機がないと検証できないようなプログラムの検証環境を提供する。このＯＳ検証用シミュレータの内部構造を解説し、このシミュレータを用いて、実際にターゲットハードウェアが必要とされるプログラムの検証の適応効果について論じる。

2.シミュレータによる検証対象

シミュレータでの検証対象は以下のものがある。

(1) コンパイラ、アセンブラなどの言語処理系のチェック
(2) ターゲットとなるアーキテクチャの違いによるプログラムの論理ミスの検出
(3) BOOT、システム初期化プログラム、ＯＳ核など、従来ターゲットハードウェアないと検証できないプログラム

ＯＳ検証用シミュレータでは、ターゲット計算機が提供される前にこれらの問題の解決を目的とする。また、ターゲット計算機での検証がむずかしい場合、シミュレータの使用により、ターゲット計算機での検証を容易にすることも目的とする。

3.ＯＳ検証用シミュレータの構成

今回開発したシミュレータの構成を図1に示す。シミュレータは以下の部分に大別される。

(1) 命令実行、アドレス変換など、ターゲット計算機のアーキテクチャをシミュレートする部分。
(2) 割り込みなどターゲットハードウェアをシミュレートする部分。
(3) 入出力アーキテクチャに依存する部分
(4) ユーザインターフェース部分

これらの機能について、順に示す。

3.1 命令実行シミュレート

ターゲット計算機の命令を解釈、実行する部分である。この部分は、基本的には以下のように処理される。

(1) 命令のフェッチとデコード
(2) 命令の実行
(3) 命令の結果の格納

シミュレータでは、命令デコードの実行に時間がかかる。本シミュレータの効率をあげるため、命令のデコード結果をシミュレータ内部に保存しておき、同じアドレスの命令を実行するときは保存しておいたデコード結果を使用する。

また、ＯＳ検証用のシミュレータでは、未定義命令のいくつかをシミュレータ内の機能を呼び出すために使用している。この機能を使用するかどうかはＯＳ検証用シミュレータ実行中に切り替えることができる。

3.2 命令フェッチチェックシミューレート

この部分は命令実行シミュレートの前に命令取り出しに関する処理を行う部分である。この部分で行う処理は以下のものがある。

(1) 入出力処理チェック
(2) 命令フェッチでのエラーの検出
(3) 割り込みのチェック、割り込み要因があれば割り込みハンドラの起動を行う。
(4) TOD、タイマーなどのカウントとその制御を行う。

3.3 アドレス変換シミュレート

ターゲット計算機では、メモリアクセスごとに仮想アドレスから実アドレスへの変換が必要となる。本シミュレータはターゲットハードウェアと同じ機構をシミュレートする。

今回のターゲット計算機では、仮想アドレスから、実アドレスに変換されるまでにメモリ内のテーブルを２回参照する。変換の効率を向上させるため、実際の計算機と同様にTLBを用いてアドレス変換の効率をあげている。

3.4 入出力シミュレート

ターゲット計算機では、チャネルプログラムでI/Oの実行手順を示し、I/Oプロセッサがチャネルプログラムを処理することで、I/O処理を行っている。

シミュレータでは、チャネルプログラムを解釈し、シミュレーションする。この処理は以下のようになる。

Testing OS behavior on developping computer architecture.
Kazuaki Takeuchi, Takayuki Yagi, Ryoya Mori
TOSHIBA CORPORATION

(1) チャネルプログラムを解釈する。
(2) I/Oに対する入出力の時には、シミュレータのホストマシンの提供する入出力命令に変換して実行する。例えば、ディスクアクセスであれば、ホストマシンのファイルアクセスに変換し、端末I/Oであればホストマシンの端末アクセスに変換する。
(3) チャネルプログラムの終了状態をI/Oの内部情報に設定する。

3.5 割り込みシミュレート

ターゲット計算機の割り込みを処理する部分である。割り込みの要因チェックは、命令フェッチチェックシミュレート部分、命令シミュレート部分、アドレス変換部分で行っている。これに対して、割り込みハンドラへの制御の移行はこの部分で行なう。

ターゲット計算機の持つ割り込み制御機構に加えて、ＯＳ検証用シミュレータでは割り込みごとに割り込みハンドラの起動を制御できる。これにより、割り込みハンドラが作成されていない場合、従来では、割り込みハンドラ不在の割り込みにより、別の割り込みが発生し検証を困難にしていたのに対し、割り込みハンドラ不在の時点で処理を中断できるため、プログラムの検証を容易にしている。

3.6 レジューム機能

シミュレートの対象となるプログラムが大きい場合、ＯＳ検証用シミュレータで実行すると時間がかかる。また、計算機のたち上げなど複数のプログラムが実行される場合、１つのプログラムを差し替える度に最初から実行すると時間がかかる場合がある。

ＯＳ検証用のシミュレータではプログラムの任意の時点でそのときのシミュレータの持つ内部情報、メモリの内容、I/Oの状態を保存することにより、保存した時点の実行を再開できる。

3.7 ユーザインターフェース

ＯＳ検証用シミュレータではユーザインターフェース部分とシミュレータ部分を分離し、２つの部分をシミュレータで定めたインターフェースで制御している。マンマシンインターフェースを独立させることにより、さまざまなマンマシンインターフェースを使用ができる。ＯＳ検証用としては、ターゲットハードウェアの保守用プロセッサのインターフェース、デバッカのインターフェースを提供している。

ＯＳ検証用のシミュレータでは複数のインターフェースを用いることができる。

4.適応評価

ＯＳ検証用シミュレータを適応してターゲット計算機のたち上げプログラムを検証した結果、言語処理系、プログラムのアルゴリズム等の修正点を検出ができた。また、ターゲット計算機では割り込みの検証は大変であるが、シミュレータの割り込みの制御機構により、予想されない割り込みの発生を事前に知ることができるため、割り込みハンドラの検証を容易にしている。

5.おわりに

ＯＳ検証用シミュレータの作成により、ターゲット計算機無しでも言語処理系、ターゲット計算機に依存するプログラムの検証を行えることを確認できた。同時に、ターゲット計算機では困難な検証を容易にすることも確認できた。今後は、ターゲット計算機の検証に使用できるように、シミュレータの付加機能を充実、ユーザインターフェースの改善を検討していきたい。

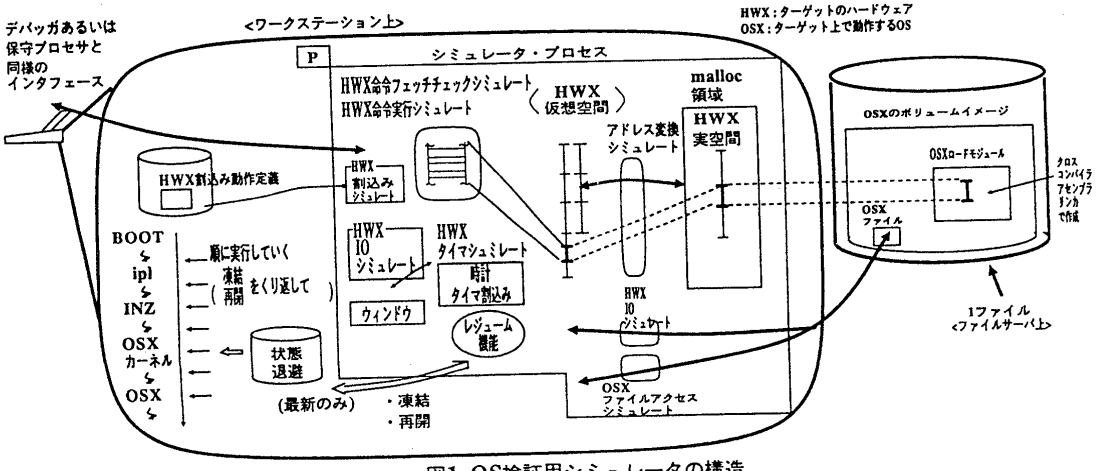

図1 OS検証用シミュレータの構造