PISA based VLIW Processor 向けクロス環境の構築

石井 浩登 † 月川 淳 † 古川 文人 †† 横田 隆史 † 大津 金光 † 馬場 敬信 †
† 宇都宮大学大学院工学研究科情報システム科学専攻 †† 帝京大学ラーニングテクノロジー開発室

1 はじめに

計算機システムの高性能化のための有力な手段として挙げられるマルチ VLIW プロセッサの実験基盤として、本研究室では PISA 命令セット [1] をベースとした VLIW プロセッサ PVP とシミュレーション環境 CHA-MEN[2][3] を開発している。このような新しいプロセッサアーキテクチャにおいては、その有効性を検討するためにベンチマークなどでの評価を行う必要があるが、それらの現実的なプログラムの動作にはランタイムライブラリ (RTL) が利用されており、アーキテクチャに対応した RTL が必要となってくる。そこで、本稿では PVP において現実的なプログラムを動作させるためのクロス環境の構築を行う。

2 PISA based VLIW Processor (PVP)

PVP は PISA 命令セットを用いた VLIW プロセッサ実験環境であり、最大 4 命令の同時実行が可能な構成となっている。

2.1 PISA VLIW 命令セット

PISA とは MIPS 命令セットをベースとした命令セットである。64bit という非常に長い命令長を持ち、先頭 16bit の未使用領域を利用して様々な拡張が可能である。PVP ではこの先頭 2bit を VLIW width field として同時実行命令数の指定に利用している。図 1 に PISA VLIW 命令の概略を示す。

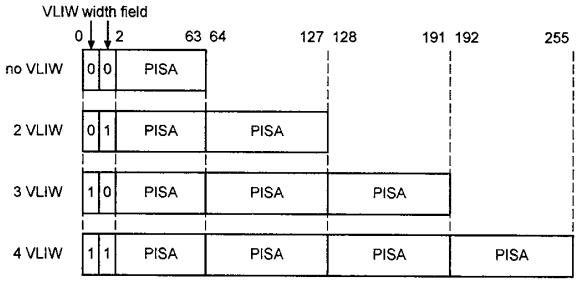

図 1: PISA VLIW 命令

3 CHA-MEN

CHA-MEN は、VLIW アーキテクチャの詳細な定量的評価のために開発されたシミュレーションシステムである。処理系は言語処理系、命令スケジューラ、シミュレータ、プロファイラから構成されている。CHA-MEN の命令セットは PVP と共通して用いられており、CHA-MEN で生成したプログラムの実行イメージを PVP で実行することが可能である。図 2 に CHA-MEN 処理系の構成を示す。図中の斜線部は今回新たに追加した部分である。

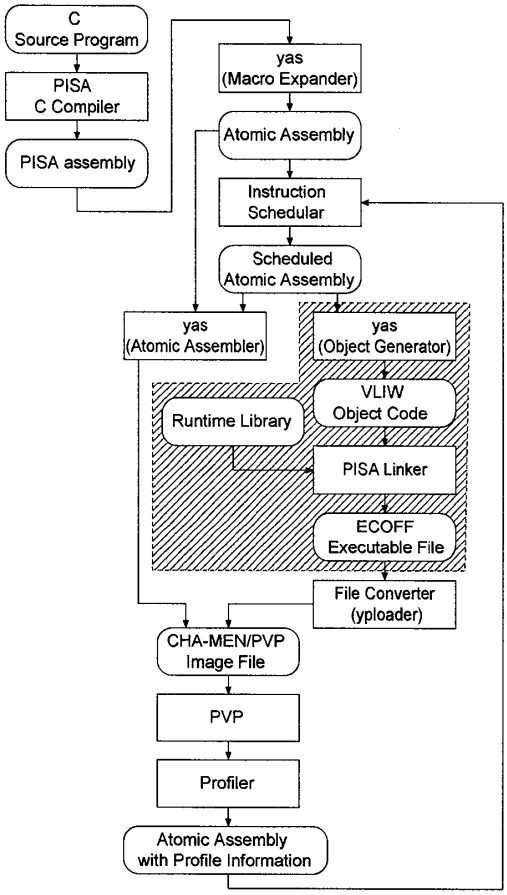

図 2: CHA-MEN 処理系と PVP

3.1 言語処理系

CHA-MEN の言語処理系はアセンブリコードを出力するための PISAgcc、それをスケジューリングが容易なマクロ命令に展開及び実行イメージの生成などを行う YAWARA アセンブラ (yas)、実行バイナリを実行イメージに変換するファイルコンバータ yploader で構成される。

3.1.1 YAWARA アセンブラ (yas)

PISA アセンブリ命令ではアセンブル時に複数の命令に展開されるマクロ命令が存在する。VLIW のスケジューリングは機械語単位で行うため、PISAgcc によるアセンブリの出力のままでは適切にスケジューリングが行えない。YAWARA アセンブラはその問題を解決するためにアセンブリコード中のマクロ命令を展開し、アトミックアセンブリとして出力する。また、アトミックアセンブリからプログラムの実行イメージの出力も行うことが可能である。

3.2 命令スケジューラ

命令スケジューラは、マクロ展開されたアセンブリコードを読み込み、命令レベル並列性の解析と命令の並べ替えを行い、VLIW 化されたアセンブリコードを出力する。

Development of cross environment for PISA based VLIW Processor

†Hiroto Ishii, Atsusi Tsukikawa, Takashi Yokota, Kanemitsu Ootsu and Takanobu Baba
††Fumihito Furukawa
Department of Information Systems Science, Graduate School of Endineering, Utsunomiya University (†)
Learning Technology Laboratry, Teikyo University (††)

4 CHA-MEN の問題点

適切にスケジューリングを行うためにマクロ命令を展開しておく必要性があるなどの理由から従来の CHA-MEN 処理系ではアトミックアセンブリを処理してプログラムコードを生成していた。しかし、ほとんどの現実的なプログラムでは printf 関数などをまとめたランタイムライブラリ (RTL) が利用されている。RTL はリンカによりオブジェクトコードとリンクされて実行プログラムとなる。そのため、ベンチマークなどにより PVP の有効性検討を行うためには RTL とリンク可能な VLIW 化されたオブジェクトコードを生成する機構および PVP で利用できる RTL が必要である。

5 クロス環境の構築

5.1 VLIW 化オブジェクト生成モジュールの実装

オブジェクトファイルは通常、コンパイラによりソースコードから生成されるバイナリコードである。しかし、VLIW 化を行うためには命令コードの解析などが必要であるため CHA-MEN ではコンパイラはアセンブリコードの生成のみを行う。オブジェクトファイルの生成には RTL とリンクを行うためのシンボル・リロケート情報などが必要であり、アセンブリコードからそれらの情報をまとめなければならない。また、ファイルフォーマットに合わせてデータを適切に配置することが必要となる。そこで、yas の実行イメージ生成モジュールによりまとめられたプログラムコードの情報などを利用した上で、オブジェクトファイルのデータを管理するためのライブラリである BFD (Binary File Descriptor) ライブラリを用いて VLIW 化オブジェクト生成モジュールの実装を行った。オブジェクトファイルフォーマットとしては広く利用されており、既存の PISA 用リンカでも用いられる ECOFF (Extended Common Object File Format) を用いることとした。VLIW 化オブジェクト生成モジュールによるアトミックアセンブリからの実行イメージ生成までの処理は図 2 の斜線部内に示すように行う。まず、アトミックアセンブリを yas の VLIW 化オブジェクト生成モジュールによりオブジェクトコードに変換し、それを PISA 用リンカによって RTL とリンクして実行ファイルを生成する。そして、ファイルコンバータ yploader によって PVP で利用可能な実行イメージへ変換する。

5.2 ECOFF (Extended Common Object File Format)

ECOFF のデータ構造は大きく分けて以下の 4 つで構成される。yas ではこれらのデータをアトミックアセンブリより抽出・生成し BFD によって管理することで VLIW 化オブジェクトファイルを生成する。

ヘッダ　ヘッダはデータサイズなどオブジェクトファイルに関する全体的な情報を保持している。yas の実行イメージ生成モジュールによりまとめた情報を利用して BFD で管理する。

セクション　セクションは命令コードやプログラムで利用されるデータなどを種類によりまとめてそれぞれ保持する。そのため、セクションは複数存在する。yas ではスケジューリングされたアトミックアセンブリに VLIW width field の付加を行い、それをセクションデータとして BFD で管理する。これより生成されるオブジェクトファイルは VLIW 化されたものとなる。

リロケート　リロケートは RTL とリンクしたときに処理されるルーチンの外部参照の解決、セクション内のアドレス調整を行うための情報を保持する。従来の実行イメージの生成ではプログラム全体を見てアドレスを相対的に決めていたため、シンボル情報を基に適切な再配置情報 (再配置のアドレス、再配置型など) を BFD で管理するためのモジュールを yas に実装した。

シンボル　シンボルはプログラム内で定義されている大域シンボルやローカルシンボルの情報を保持する。これらの情報はリンカで用いられるが、従来の yas ではリンカを利用することを考慮していなかったため、シンボル情報を付加するためのモジュールを yas に実装した。

5.3 実装動作検証

実装したモジュール (リロケート情報・シンボル情報付加モジュール及び ECOFF オブジェクト生成モジュール) により生成した VLIW 化 ECOFF オブジェクトファイルを PISA 用リンカによってリンクした際にリロケートが問題なく行われるか、VLIW width field が保持されるか CHA-MEN シミュレータを用いて検証を行った。その結果、VLIW width field が保持され VLIW 実行が確認できた。

5.4 PVP 用ライブラリの実装

PVP 用ライブラリの実装については、実装を容易に行えるよう既存のライブラリをベースとして開発する。ベースとなるライブラリの候補としては、CHA-MEN とシステムコールの互換を持つ SimpleScalar で利用される glibc、組み込みシステム向けで小規模であり、実装の容易性が高い newlib が挙げられる。システムコールの互換を考慮すると glibc の方が移植性に優れているが、現在の PVP には利用可能な OS が存在しておらず OS に処理を依存して動作するシステムコールを実行することが出来ないという問題があるため、OS のサポートが必要となる glibc を PVP で利用することは難しい。そのため、OS のサポート無しで動作が可能な newlib を PVP 用ライブラリのベースとして PISA VLIW への移植を行う。

6 おわりに

マルチ VLIW プロセッサの実験基盤であるソフトウェア開発のために必要なクロス環境の構築を行った。実装は既存の CHA-MEN 処理系をベースとし、ベンチマークなどの現実のプログラムコードを PVP で動作できるように処理するための RTL の開発を行った。その結果、CHA-MEN にて RTL を利用したプログラムの実行が可能となった。今後は、PVP 実機上でプログラムを動作可能にするための PVP 用ライブラリの実装を行っていく予定である。

謝辞　本研究は，一部日本学術振興会科学研究費補助金（基盤研究 (B)18300014，同 (C)19500037，同 (C)20500047）および宇都宮大学重点推進研究プロジェクトの援助による。

参考文献

[1] Doug Burger, Todd M. Austin: “The SimpleScalar Tool Set, Version 2.0“, University of Wisconsin-Madison Computer Sciences Department Technical #1342, June, 1997.

[2] 月川淳，古川文人，青木隆行，岡大輔，大津金光，横田隆史，馬場敬信: “CHA-MEN: スケジューラ協調開発を支援する VLIW シミュレーション環境 “, コンピュータシステム研究会 (CPSY2005), pp.2-5, 2005.5.

[3] 三村貴志，中島伸吾，横田隆史，大津金光，馬場敬信: “PISA ベース VLIW プロセッサの FPGA による試作 “, 情報処理学会第 69 回全国大会, 2007.3.