4U-1　ＡＩチップ（ＩＰ１７０４）のアーキテクチャ

皆川健二　斎藤光男　相川　健
的場　司　岡村光善　石井忠俊
（株）東芝

１．はじめに

近年，高速な実行速度が得られ，プログラミング環境のよいＡＩプロセッサが開発されている。また、汎用言語を高速に実行するＲＩＳＣアーキテクチャが提案されている。リスプやプロローグ言語をサポートするＲＩＳＣマシンも、いくつか伝えられている。

我々はすでに、新しいアーキテクチャに基づくＡＩプロセッサ（ＩＰ７０４）を開発した[1]。それはＲＩＳＣアーキテクチャに修正と拡張を施し、WAM (Warren Abstract Machine) 命令とリスプ命令をサポートしている。ＡＩプロセッサチップ（ＩＰ１７０４）はＩＰ７０４をＬＳＩ化する目標で開発された。チップ化にあたって、アーキテクチャは若干、修正された。ハードウェアデコードとマイクロプログラムデコードを併用し、デコードステージとレジスタリードステージをオーバーラップさせる方式と、ディレイドライティングによるディレイドキャッシュヒット方式を新たに開発した。本文では、ＩＰ１７０４のアーキテクチャおよび、その性能評価を述べる。

２．設計思想

ＩＰ１７０４ではＲＩＳＣアーキテクチャを出発点とし、ＡＩ言語をサポートするための拡張を行なった。命令セットと高級言語、ＡＩ言語とのセマンティックギャップは、コンパイラ技術により補うことができた。

ＲＩＳＣアーキテクチャは規模が小さいため、ローコストで実現可能でき、ＬＳＩ化も容易であるが、我々はＩＰ１７０４の開発の上で特に、ＡＩ言語をサポートする拡張ハードウェアがサイクルタイムを損なうことなく、少ない実装面積で実現できるよう注意を払った。

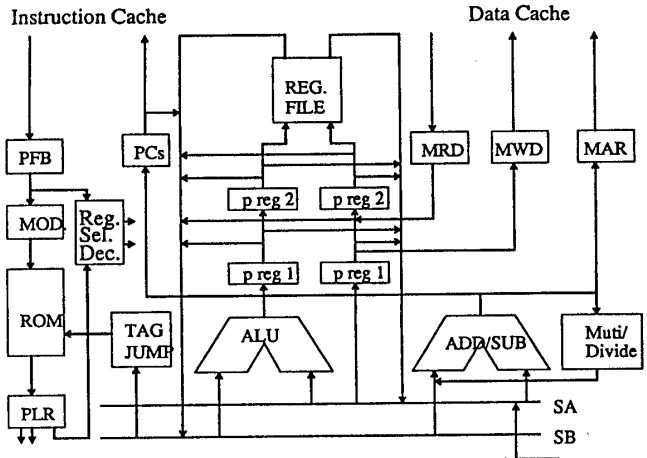

Fig. 1 Architecture of IP1704

図1　ＩＰ１７０４のアーキテクチャ

３．ＩＰ１７０４のアーキテクチャ

ＩＰ１７０４のブロック図を図１に示す。３２ビット固定長を持つ、３２ビットマシンである。

命令フォーマットは８ビットのオペコードフィルドと５ビットの最大３つのレジスタフィールドとイミーディエイト、ディスプレイスメントフィールドを有している。

パイプラインは５段とした（図２）。このパイプラインはフェチ、デコード／レジスタリード、実行、メモリアクセス、ライトステージよりなる。メモリステージはメモリアクセス命令以外は実行されない。　レジスタファイルは４ポートの４８ワード、３２ビットレジスタである。

ＩＰ１７０４はＩＰ７０４で採用した命令すりかえ機能、２ポートバスアーキテクチャ、タグ処理、トラップ処理機能を引き継いでいる。その他のＡＩ言語を高速に実行させるための機能を述べる。

3.1 ハードウェアデコードとマイクロプログラムの併用

マイクロプログラム方式の利点は、(1) ＡＩ言語をサポートするハードウェアを簡素化、高速化できる。(2) 条件ジャンプやマルチウェイジャンプの際のペナルティーが少ない。マイクロプログラムフィールドは長いのでＡＬＵ機能とジャンプフィールドを１ワードにできる。この機能はプロローグのユニフィケーション、フェイルプロセスに効果がある。(3) マクロコードサイズ少なくし、キャッシュミスを低下させる。

上記のようにマイクロプロクラム方式は、ＡＩ言語をサポートするのに適しているが、マイクロプログラムデコードはハードウェアデコードより低速であるため、デコードステージを必要とする。また、ＩＰ１７０４ではレジスタファイルの読出し時間がサイクルタイムに近い値であるので、レジスタリードパイプラインステージが必要となった。しかしブランチ命令の際、実行ステージの前に余分なパイプラインステージを設けることでパイプブレーク段数が増し、性能低下につながる。

これを避けるために、リードレジスタ番号に、ハードウェアデコードとマイクロプログラムデコードを併用した。デコードステージとレジスタリードステージのオーバーラップ化を行なった。すなわち、マイクロ命令の１サイクルめだけはリードレジスタ番号をハードウェアデコードしている。従って、マイクロ命令をリードしている間に、レジスタを読み込むことが可能である。マイクロ命令中のレジスタ番号は次のサイクルで使用される。同様のことをイミーディエイト生成においても行なっている。

3.2 キャッシュメモリサポート

データメモリリードとタグメモリリードは同時に行われるので、一般的にディレイドヒットチェックはライト時に２サイクル必要となる。これを図３に示す。しかしパイプライン化により、リードの直後にライトが続かないかぎり、メモリアクセスを１サイクルで行うことができる。

Architecture of AI Processor Chip (IP1704)
Kenji MINAGAWA, Mitsuo SAITO, Takeshi AIKAWA,
Tsukasa MATOBA, Mitsuyoshi OKAMURA, Tadatoshi ISHII
Toshiba Corporation

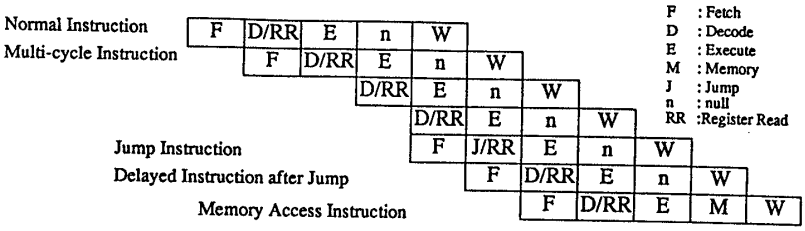

図2　ＩＰ１７０４のパイプラインステージ

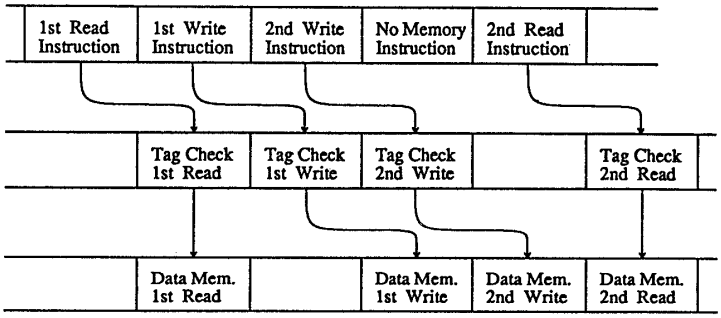

図3　メモリアクセスタイミング

3.3 フローティングポイント（ＦＬＰ）演算

ＩＰ１７０４は、ＦＬＰプロセッサを内蔵していないが、適当なＦＬＰを外付けし、コプロセッサ空間を通して通信をすることでＦＬＰ演算が可能となる。

3.4 乗除算

整数型乗除算器を内蔵している。乗算はブースのアルゴリズムを用い、20ステップ、除算は非回復型アルゴリスムを用い、42ステップで実行可能である。

3.5 メモリインタフェイス

ＩＰ１７０４はいわゆるハーバードアーキテクチャである。メモリアクセスの際にデータと命令の競合が起こらない。しかし、ハーバードアーキテクチャはデータと命令キャッシュの間でコンシステンシーを取りにくい。ＡＩ言語では自己変更コードが許されているので、問題を起こす可能性がある。このために、命令キャッシュとデータキャッシュをフラッシュする命令とアクセスタイプチェック機能をサポートしている。

3.6 命令セット

汎用命令は乗除算以外１サイクル命令とし、ＡＩ命令においても簡単な命令に限定し、コンパイラの性能を十分に発揮させるよう配慮した。リスプではタグチェック命令およびジェネリック算術命令を採用した。ジェネリック演算命令は演算命令とブランチ命令の２つの性格を持っており、タグがfixnumの時、１クロックで演算とブランチを行う。

4．インプリメンテーション

ＣＭＯＳ 1.2μmを用い、チップサイズは14.5mm*14.5mm となった。約168kトランジスタで実現されている。そのうち半数のトランジスタはＲＯＭに使われているが、10パーセントしかチップ面積を占有していない。これはマイクロプログラムがＶＬＳＩ化においても有効であることを示している。

Benchmark	IP1704	IP704
Append (deterministic)	1.5M lips 11 clk/Inf.	667 K lips 15 clk/Inf.
Nrev30 Total 497 Inf.	1.4M lips 5835 clk	534 K lips 9307 clk

(first prototype)

表1　プロローグの性能

Gabriel Benchmark

Benchmark	IP1704 non.	IP1704 opt.	IP704
tak	75	61/0.81	114/1.52
destructive	201	146/0.73	304/1.51
derivative	320	281/0.88	520/1.62
average ratio	1	0.81	1.55(1.91)

times in msec/ratio

表2　リスプの性能

5．ベンチマーク

ベンチマークテストはハードウェアシミュレータを用いて行った。キャッシュヒット率100 パーセント、クロックスピード60ns (16.7MHz) と仮定した。

表１および表２にＡＩプログラムの結果を示す。コンパイル時に最適化をしない場合には、ＭＩＰＳ－Ｘ(50ns)の1.3 倍の性能を示し、最適化を行った場合には1.6 倍の性能を示した。

Ｃ言語(Dhrystone) については、ライブラリ関数(strcopy, strcmp)についてのみ最適化を行った。14700dhrystoneを示し、ＳＵＮ４に匹敵する性能を示している。

6．まとめ

我々はＩＰ１７０４をＶＬＳＩとして開発し、ベンチマークテストにより優れた性能を示すことを確認した。また、このアーキテクチャで採用したＲＯＭデコードとハードウェアデコードの併用、ディレイドキャッシュライトはＡＩ言語をサポートする上で効果があることを確認した。

参考文献

[1] 斎藤，相川，前田，星野，「ＡＩワークステーションの開発思想」，『昭和62年度人工知能学会全国大会（第１回）論文集』論文番号5-1-3　pp.237-248, 1987年7 月