6G－7

i860のデュアルオペレーション命令における データフローグラフのスケジューリング

鈴木　雅之　徐 行俊　土肥 浩　石塚 満

東京大学工学部電子情報工学科

1　はじめに

インテルi860マイクロプロセッサは、単一の素子でスーパーコンピュータレベルの性能を発揮する。i860の特徴の一つであるデュアルオペレーション命令等を用いると、演算速度は向上する。しかし、今のところ、どの程度効率の良いプログラムコードが生成できるかは、使用した言語とコンパイラに依っており、システマティックな手法は明確になっていない。本研究では使用言語やコンパイラに関わらず、i860の高速性を活用するために、一般のデータフローグラフを最適スケジューリングする手法について示す。

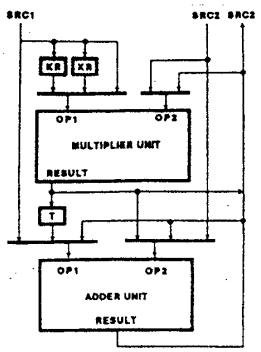
2　i８６０について

i860は、整数、浮動小数点、及びグラフィクスの性能をバランス良く、かつ統合的に取り入れている。特徴としては、浮動小数点の加算と乗算を実行するデュアルオペレーション命令、浮動小数点演算と整数演算を同時に実行するデュアルインストラクションモード、グラフィックス命令などが挙げられる。

浮動小数点演算では、パイプライン処理と加算器、乗算器の並列動作を使えば、１クロックに最高２つの出力結果が得られる。

パイプライン処理においては、加算器、乗算器のまわりのさまざまなデータ経路を選べる。

また、デュアルオペレーション命令とデュアルインストラクションモードを組み合わせれば、３つの命令を同時に実行することも可能である。



i860のデータ経路

3　データフローグラフのスケジューリング

3.1　マトリクス演算

DOP命令を用いて効果が明らかなのは、３次元の座標変換などのマトリクス演算のような問題に対してである。

例として、座標（ｘ，ｙ，ｚ）に対して３×３の行列を掛け合わせる場合を考える。

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

DOP命令を使わないと、浮動小数点加算、乗算がともに３クロックずつかかり、Ｘ、Ｙ、Ｚを求めるまでに１５×３＝４５クロックを要する。

DOP命令を使った場合、加算、乗算どちらについても入力後３クロックを経ないとその出力は得ら

An Instruction Scheduling Method for i860 Dual-operation using Data Flow Graph
Masayuki Suzuki,Xing-jian Xu,
Hiroshi Dohi,Mitsuru Ishizuka
University of Tokyo
7-3-1 Hongo,Bunkyou,Tokyo 113,Japan

れない。しかし、例えば、Xの演算の入力を与えた後、別のYやZの計算をパイプラインに投入することで見かけ上、1クロックで出力を得られる。すると、X、Y、Zを求めるまでに15クロックで済む。

この演算が1回だけなら3倍の速度差が出るが、さらに何回も繰り返して3×3のマトリックス演算を行う場合には、さらに速度差は顕著になる。(n回演算なら、DOP命令を用いない時：45n；用いる時：9n+6)(下図参照)

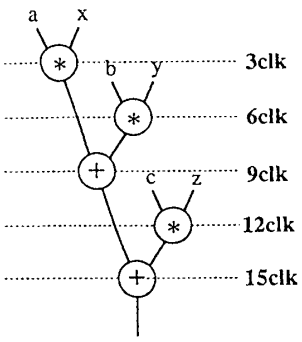


図1 デュアルオペレーションを用いない場合 (これを3回繰り返す)

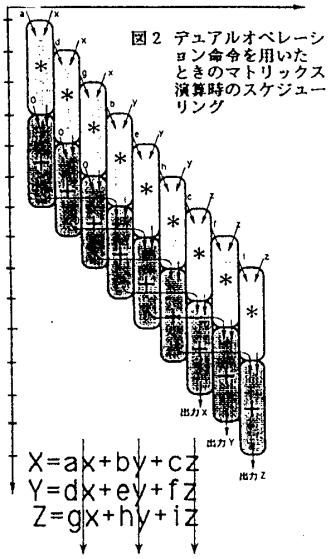


図2 デュアルオペレーション命令を用いたときのマトリックス演算時のスケジューリング

効率的なパイプライン投入順序

さらにこのことは3×3のマトリックス演算だけでなく、3n×3m行列、n×m行列などにも応用できる。

この3×3のマトリックス演算の例について

- 3クロック後に演算結果が現れることを利用。
- 加算、乗算の同時動作を隙間なく行う。

ために、DOP命令を用いたほうが速く処理される。データフローグラフから始める場合にはこの点を考慮して変形を考えなくてはならない。

3.2 3本のデータフローグラフによるスケジューリング

一般に、接続された演算をDOP命令で行うと、3クロックの待ち時間のためにDOP命令を用いない場合より遅くなってしまうことが多い。そこで、3.1を参考に、独立な3本のデータフローグラフを仮定して、スケジューリングを行う。

3.1の例は、X=a×x+b×y+c×zという演算の3回の繰り返しであり、Xのデータフローグラフを3本並列に並べたものである。従って、このことから3本の同じデータフローグラフのスケジューリング時に考慮しなければならないのは、次の3点のみである。

- 1つの演算には3クロックかかること。
- 同時には加算、乗算1つずつしか行えないこと。
- 接続された演算は前の演算の結果が出た後、行うこと。

これにより、このスケジューリングは的確かつ、単純に行える。

また、データフローグラフがn本ある場合については、n=(3の倍数)+(余り)とみなして、3本ずつデータフローグラフをまとめて処理すれば、スケジューリング法は3本の時とほぼ同様になる。

独立な3本のデータフローグラフの場合でも、スケジューリングの際に加わる制限は同じであり、この制限を満たすような最適スケジューリングをすることが可能である。

4 まとめ

現在、独立な3本のデータフローグラフから最適スケジューリングを行うプログラムを試作し、評価中である。

本論文で提案した手法を用いれば、i860だけでなく、複数の演算ユニットがパイプラインで結合されたような構造を持つプロセッサに対し、最適なアセンブラコードの決定支援ができると考えられる。

参考文献

[1] Intel corp. *i860 64bit microprocessor programmer's reference manual*, 1989

[2] J L.Hennessy & D A.Patterson. *Computer Architecture*. Morgan Kaufmann, 1992