2011年 後学期

計算機アーキテクチャ 第二 (O)

7.パイプライン制御

大学院情報理工学研究科 計算工学専攻
吉瀬謙二 kise _at_ cs.titech.ac.jp
S321講義室 月曜日 5, 6時限 13:20－14:50

VAX 11 (1 native MIPS machine)

VAX命令 (Complex Instruction Set Computer)

MOVL @40(R4), 30(R2) ; M[M[40+R4]] <- M[30 + R2]

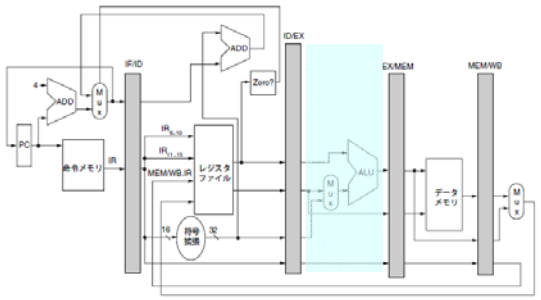
MOVC3 @36(R9), (R10), 35(R11)
R1 <- 35 + R11, R3 <- M[36 + R9]
for (R0 <- M[R10]; R0!=0; R0--)
{ M[R3] <- M[R1]; R1++; R3++; }
R2 = 0; R4 = 0; R5 = 0;

1 native MIPS

パイプラインの実行の困難さ

- 例外への対処
  - I/O デバイスからの要求
  - ユーザプログラムからのOSサービスの呼び出し
  - 命令実行のトレース生成
  - ブレークポイント（プログラマの要求による割り込み）
  - 整数演算命令のオーバーフロー
  - FP 演算命令の不規則さ
  - ページフォールト（メインメモリ内に無い場合）
  - 整列されていないメモリアクセス（整列が必要な場合）
  - メモリ保護違反
  - 未定義あるいは未実装命令の使用
  - ハードウェア異常故障
  - 電源異常
- 命令セットの複雑さ
- 複数サイクル処理の扱い

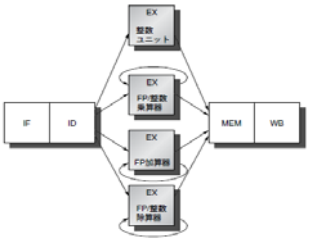
プロセッサのデータパス（パイプライン処理）



パイプラインの実行の困難さ: 複数サイクル命令

- MIPSパイプライに**浮動小数点 (floating point, FP) 演算**命令を追加する.
- 次の4つの機能ユニットを仮定する.
  - ロードとストア, 整数ALU演算, 分岐（1サイクル）
  - FP 加算, 減算, 型変換（複数サイクル）
  - FP/整数の乗算器（複数サイクル）
  - FP/整数の除算器（複数サイクル）

パイプライン化されない3つの機能ユニットの追加



- 実行ステージがパイプライン化されないので, 資源競合が発生する. このため, パイプラインをストールさせる.
- 浮動小数点演算が頻繁に実行される場合, この構成では効率が悪い.

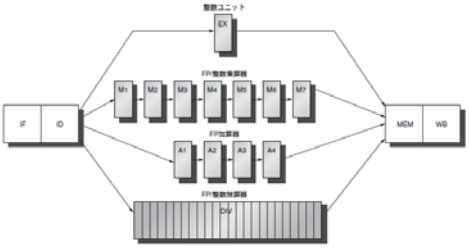
機能ユニットのレイテンシと発行間隔の例

機能ユニット	レイテンシィ	発行間隔
整数ALU	0	1
データメモリ（整数・FPロード）	1	1
FP加算器	3	1
FP乗算器（整数乗算にも使用）	6	1
FP除算器（整数除算にも使用）	24	25

- レイテンシは，結果を作る命令とその結果を利用する命令の間に挿入すべきサイクル数（整数ALUは直後の命令が結果を利用できるので0とする）
- 発行間隔とは，同じ種類の2命令を発行する間に必用とするサイクル数

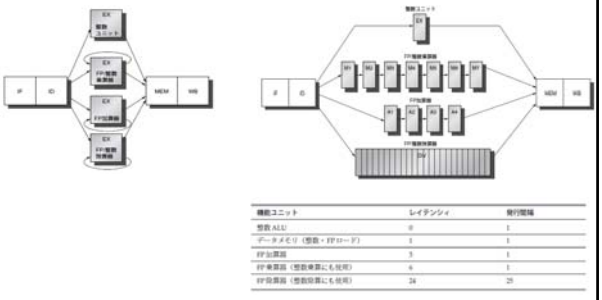
- この構成では，FP加算器は4段にパイプライン化，FP乗算器は7段にパイプライン化されている．

複数のFP演算をサポートするパイプライン



- FP乗算器，FP加算器をパイプライン化
- 実は，FP除算器はパイプライン化されていない．完了までに24サイクル．
- このパイプラインのタイミングを考える．

パイプラインの比較



ハザード (hazard)

- **構造ハザード (structural hazard)**
  - オーバラップ実行する命令の組み合わせをハードウェアがサポートしていない場合．
  - 資源不足により生じる．
- **データ・ハザード(data hazard)**
  - データの受け渡しの制約によって生じるハザード，命令i の後に命令j が実行される場合．
    - RAW (read after write) 命令iが書き込み前に，命令jがそれを読みだそうとする．
    - WAW (write after write) 命令iが書き込む前に，命令jが書き込もうとする．
    - WAR (write after read) 命令iが読む前に，命令jがそこに書こうとする．
- **制御ハザード(control hazard)**
  - 分岐命令，ジャンプ命令によって生じるハザード

データ依存関係のない命令列

レジスタ

```
R3 := R3 + 1        (1)
R4 := R4 + 1        (2)
R5 := R5 + 1        (3)
R6 := R6 + 1        (4)
```

代入

並列に実行可能

データ依存関係のある命令列

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)

If R3=20, R5=3
60 := 20 x 3         (1)
61 := 60 + 1         (2)
4  := 3  + 1         (3)
244:= 4  x 61        (4)
```

真のデータ依存 (true data dependence)

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)

If R3=20, R5=3
60 := 20 x 3         (1)
61 := 60 + 1         (2)
4  := 3  + 1         (3)
244:= 4  x 61        (4)
```

3番目の命令が完了する前に，4番目の命令を実行してはいけない．
RAW (read after write)

出力依存 (output dependence)

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)

If R3=20, R5=3
60 := 20 x 3         (1)
61 := 60 + 1         (2)
4  := 3  + 1         (3)
XXX:= 60 x 61        (4)
```

1番目の命令の代入が3番目の命令の代入より後に完了してはいけない．
WAW (write after write)

逆依存 (antidependence)

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)

If R3=20, R5=3
60 := 20 x 3         (1)
XX := 4  + 1         (2)
4  := 3  + 1         (3)
XXX:= 4  x XX        (4)
```

2番目の命令が実行を始める前に，3番目の命令を完了してはいけない．
WAR (write after read)

データ依存関係の例（演習）

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)
```


Exercise

データ依存関係の例（演習）
(A) RAW, 真のデータ依存 (true data dependence)
(B) WAW, 出力依存 (output dependence)
(C) WAR, 逆依存 (anti-dependence)

氏名，学籍番号，
学籍番号マーク欄(**右詰で**)

裏に記入

データ依存関係の例

```
R3 := R3 x R5        (1)
R4 := R3 + 1         (2)
R3 := R5 + 1         (3)
R7 := R3 x R4        (4)
```

補足：データ依存関係の例（リネーミング）

```
R3 := R3 x R5     (1)
R4 := R3 + 1      (2)
R8 := R5 + 1      (3)
R7 := R8 x R4     (4)
```


ハザード (hazard)

- **構造ハザード (structural hazard)**
  - オーバラップ実行する命令の組み合わせをハードウェアがサポートしていない場合．
  - 資源不足により生じる．
- **データ・ハザード(data hazard)**
  - データの受け渡しの制約によって生じるハザード．命令i の後に命令j が実行される場合．
    - RAW (read after write) 命令iが書き込み前に，命令jがそれを読みだそうとする．
    - WAW (write after write) 命令iが書き込む前に，命令jが書き込もうとする．
    - WAR (write after read) 命令iが読む前に，命令jがそこに書こうとする．
- **制御ハザード(control hazard)**
  - 分岐命令，ジャンプ命令によって生じるハザード

独立した（データ依存のない）4つの命令のパイプライン

MUL.D	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
ADD.D											
L.D											
S.D											

.D は倍精度(64ビット)浮動小数点命令

- 浮動小数点の乗算ユニットのレイテンシは6（結果を得るまでM1～M7の7サイクル）．
- 命令は異なる実行時間を持つので，1サイクルに必用なレジスタ書き込みが1より大きくなり得る．
- 発行順でWBステージに到着しないので，WAWハザードの可能性がある．
- 発行順ではなく命令が完了する．例外処理が問題となる．
- 実行のレイテンシが長いので，RAWハザードが頻繁に発生する．

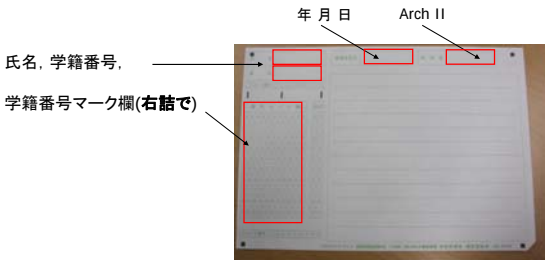
独立した4つの命令のパイプライン

MUL.D	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
ADD.D		IF	ID	A1	A2	A3	A4	MEM	WB		
L.D			IF	ID	EX	MEM	WB				
S.D				IF	ID	EX	MEM	WB			

.D は倍精度(64ビット)浮動小数点命令

- 浮動小数点の乗算ユニットのレイテンシは6（結果を得るまでM1～M7の7サイクル）．
- 命令は異なる実行時間を持つので，1サイクルに必用なレジスタ書き込みが1より大きくなり得る．
- 発行順でWBステージに到着しないので，WAWハザードの可能性がある．
- 発行順ではなく命令が完了する．例外処理が問題となる．
- 実行のレイテンシが長いので，RAWハザードが頻繁に発生する．

Exercise



RAWハザードにより生じるストールの例（演習）

		クロックサイクル															
命令		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L.D	F4,0(R2)	IF	ID	EX	MEM	WB												
MUL.D	F0,F4,F6																	
ADD.D	F2,F0,F8																	
S.D	F2,0(R2)																	

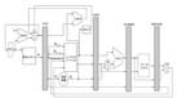
- パイプラインは完全にバイパス及びフォワーディングされていると仮定
- IF, ID, MEM, WBステージには1命令しか格納できないとする．
- 長い実行パイプラインはストールの頻度が高い．

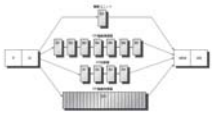
機能ユニット	レイテンシ	発行間隔
整数 ALU	0	1
データメモリ（整数・FPロード）	1	1
FP加算器	3	1
FP乗算器（整数乗算にも使用）	6	1
FP除算器（整数除算にも使用）	24	25

RAWハザードにより生じるストールの例（演習）

命令	クロックサイクル																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L.D F4,0(R2)	IF	ID	EX	MEM	WB												
MUL.D F0,F4,F6		IF	ID	stall	M1	M2	M3	M4	M5	M6	M7	MEM	WB				
ADD.D F2,F0,F8			IF	stall	ID	stall	stall	stall	stall	stall	stall	A1	A2	A3	A4	MEM	WB
S.D F2,0(R2)					IF	stall	stall	stall	stall	stall	stall	ID	EX	stall	stall	stall	MEM

- パイプラインは完全にバイパス及びフォワーディングされていると仮定
- IF, ID, MEM, WBステージには1命令しか格納できないとする.
- 長い実行パイプラインはストールの頻度が高い.





WBステージの競合

命令	クロックサイクル										
	1	2	3	4	5	6	7	8	9	10	11
MUL.D F0,F4,F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		
...						IF	ID	EX	MEM	WB	
L.D F2,0(R2)							IF	ID	EX	MEM	WB

- 実行ステージ以降は複数命令の処理ができると仮定
  - クロック11において，3つの命令がWBで競合する.
  - WBのライトポートを増やすことで解決可能だが，このケースは希なので，構造ハザードとして処理することが一般的.
  - IDステージでストールを検出するか，MEMステージに入るときにストールを検出する.
  - MEMステージに入るときには，どの命令をストールさせるかという選択

WAWハザードの可能性

命令	クロックサイクル										
	1	2	3	4	5	6	7	8	9	10	11
MUL.D F0,F4,F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		
...						IF	ID	EX	MEM	WB	
L.D F2,0(R2)							IF	ID	EX	MEM	WB

- もし，L.D命令が1サイクル早く発行されると
- ADD.D より早く F2 に書き込むので WAW ハザードが生じる.
  - ADD.D の結果(F2)が利用されることなく上書きされる場合に発生する.
  - ADD.D と L.D の間でF2が利用されると，RAWハザードのためストールする.
- (1) L.D の発行を遅らせる.
- (2) ADD.D の結果を書き込まない.

アナウンス

- 講義スライド，講義スケジュール
  - www.arch.cs.titech.ac.jp