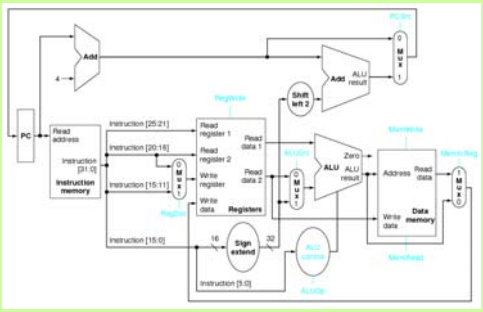
2009年 後学期

計算機アーキテクチャ 第二 (O)

5. パイプライン処理

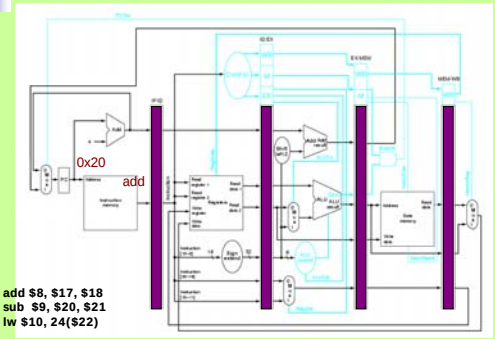
大学院情報理工学研究科 計算工学専攻
吉瀬謙二 kise _at_ cs.titech.ac.jp
S321講義室 月曜日 5, 6時限 13:20－14:50

プロセッサのデータパス(シングル・サイクル)



プロセッサのデータパス(パイプライン処理)

Clock 1:



(1) 0x20: add $8, $17, $18
(2) 0x24: sub $9, $20, $21
(3) 0x28: lw $10, 24($22)

ハザード (hazard)

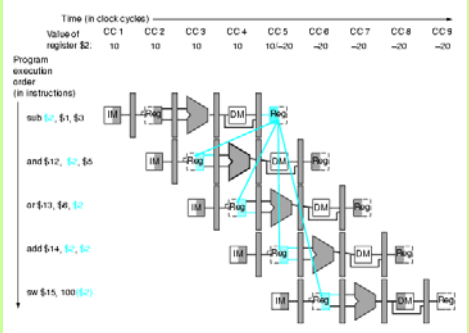
命令を適切なサイクルで実行できないような状況が存在する. これをハザードと呼ぶ.

- 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
- 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

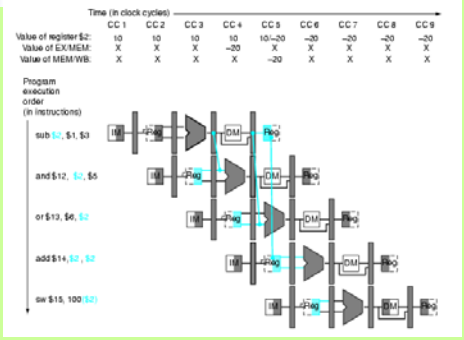
MIPSの基本的な5つのステップ(ステージ)

- **IFステージ**
  メモリから命令をフェッチする.
- **IDステージ**
  命令をデコードしながら, レジスタを読み出す.
- **EXステージ**
  命令操作の実行またはアドレスの生成を行う.
- **MEMステージ**
  データ・メモリ中のオペランドにアクセスする.
- **WBステージ**
  結果をレジスタに書き込む.

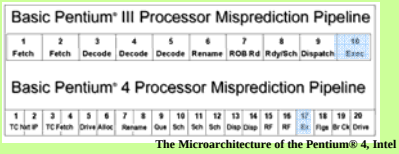
データハザード



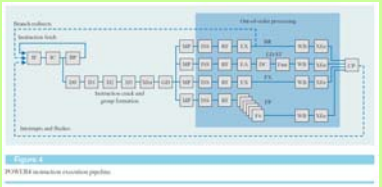
フォワーディングによるデータハザードの回避



プロセッサの命令パイプラインの例



The Microarchitecture of the Pentium® 4, Intel Technical Report



POWER4 System Microarchitecture, IBM Journal

ハザード (hazard)

命令を適切なサイクルで実行できないような状況が存在する．これをハザードと呼ぶ．

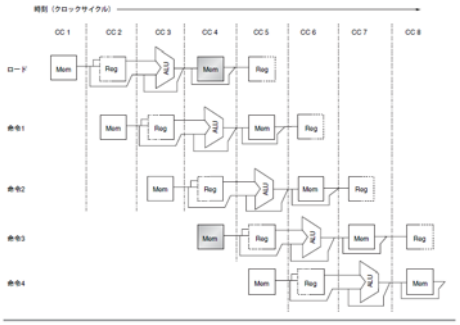
- **構造ハザード (structural hazard)**
  - オーバラップ実行する命令の組み合わせをハードウェアがサポートしていない場合．
  - 資源不足により生じる．
- **データ・ハザード(data hazard)**
  - データの受け渡しの制約によって生じるハザード
- **制御ハザード(control hazard)**
  - 分岐命令，ジャンプ命令によって生じるハザード

単純な5段のRISCのパイプライン

	クロック数								
命令数	1	2	3	4	5	6	7	8	9
命令 i	IF	ID	EX	MEM	WB				
命令 $i+1$		IF	ID	EX	MEM	WB			
命令 $i+2$			IF	ID	EX	MEM	WB		
命令 $i+3$				IF	ID	EX	MEM	WB	
命令 $i+4$					IF	ID	EX	MEM	WB

プロセッサ性能はパイプライン化されていないものと比較して**最大で5倍**になる．

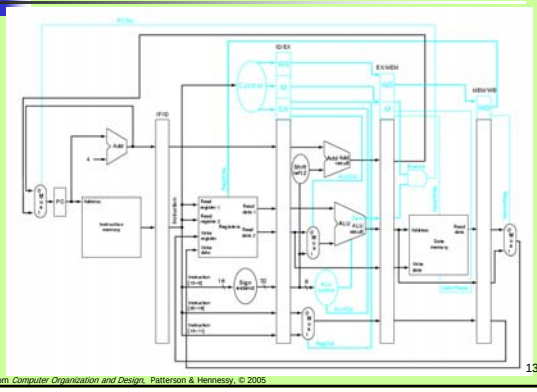
メモリポートを1つしか持たないプロセッサ



構造ハザードによるパイプラインストール

	クロックサイクル									
命令	1	2	3	4	5	6	7	8	9	10
ロード命令	IF	ID	EX	MEM	WB					
命令 $i+1$		IF	ID	EX	MEM	WB				
命令 $i+2$			IF	ID	EX	MEM	WB			
命令 $i+3$				stall	IF	ID	EX	MEM	WB	
命令 $i+4$						IF	ID	EX	MEM	WB
命令 $i+5$							IF	ID	EX	MEM
命令 $i+6$								IF	ID	EX

プロセッサのデータパス（パイプライン処理）



ハザード (hazard)

命令を適切なサイクルで実行できないような状況が存在する．これをハザードと呼ぶ．

- **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
- **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

MIPS **Control Flow** Instructions

- MIPS conditional branch instructions:

```
bne $s0, $s1, Lbl #go to Lbl if $s0≠$s1
beq $s0, $s1, Lbl #go to Lbl if $s0=$s1
```

  - Ex:

```
if (i==j) h = i + j;
        bne $s0, $s1, Lbl1
        add $s3, $s0, $s1
Lbl1:   ...
```

- Instruction Format (I format):

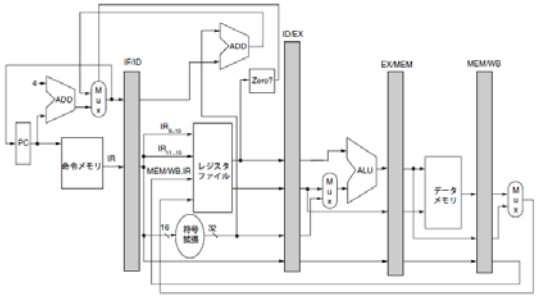
op	rs	rt	16 bit offset

- How is the branch destination address specified?

MIPSの基本的な5つのステップ（ステージ）

- **IFステージ**
  メモリから命令をフェッチする．
- **IDステージ**
  命令をデコードしながら，レジスタを読み出す．
  分岐命令である可能性を考慮し，読み出されたレジスタの間で一致比較を行う．必要であれば，命令のオフセットフィールドを符号拡張し，インクリメントされたPCに符号拡張されたオフセットを足し合わせて分岐先のアドレスを計算する．条件が成立した場合には分岐先アドレスをPCにセットして，このステージで分岐命令は完了する．

プロセッサのデータパス（パイプライン処理）



静的に採用できる制御ハザードの対処（演習）

- 戦略1
  - 分岐方向が判明するまで分岐命令の後続命令を止める．
  - IDステージで分岐命令が完了することに注意．

分岐命令	IF	ID	EX	MEM	WB
分岐先命令					
分岐先命令＋1					
分岐先命令＋2					

静的に採用できる制御ハザードの対処

- 戦略1
  - 分岐方向が判明するまで分岐命令の後続命令を止める.
  - IDステージで分岐命令が完了することに注意.
  - 分岐命令の出現毎に1サイクルのストールが発生する.

分岐命令	IF	ID	EX	MEM	WB		
分岐先命令			IF	ID	EX	MEM	WB
分岐先命令＋1				IF	ID	EX	MEM
分岐先命令＋2					IF	ID	EX

戦略2：predicted-not-taken方式 (Exercise)

- すべての分岐命令を not taken（不成立）として処理を進める.

Untaken分岐命令	IF	ID	EX	MEM	WB
命令i＋1					
命令i＋2					
命令i＋3					
命令i＋4					

Taken分岐命令	IF	ID	EX	MEM	WB
命令i＋1					
分岐先					
分岐先＋1					
分岐先＋2					

Exercise

氏名，学籍番号，
学籍番号マーク欄(**右詰で**)

戦略2：predicted-not-taken方式

- すべての分岐命令を not taken（不成立）として処理を進める.
  - 分岐結果が不成立であれば，ペナルティは生じない.
  - 分岐結果が成立であれば，1サイクルのペナルティ

Untaken分岐命令	IF	ID	EX	MEM	WB			
命令i＋1		IF	ID	EX	MEM	WB		
命令i＋2			IF	ID	EX	MEM	WB	
命令i＋3				IF	ID	EX	MEM	WB
命令i＋4					IF	ID	EX	MEM WB

Taken分岐命令	IF	ID	EX	MEM	WB			
命令i＋1		IF	idle	idle	idle	idle		
分岐先			IF	ID	EX	MEM	WB	
分岐先＋1				IF	ID	EX	MEM	WB
分岐先＋2					IF	ID	EX	MEM WB

戦略3：predicted-taken方式

- すべての分岐命令を taken（成立）として処理を進める.
- IDステージが終了して，分岐と判定するとすぐに分岐成立として処理を継続.
- 今考えているパイプライン構成では，この方式の利点はない.

戦略4：遅延分岐 (delayed branch)

- 分岐命令の後続の幾つかの命令を実行した後に，分岐する.
  1サイクルの遅延を持つ命令実行順は次の通り.
  - 分岐命令を実行
  - 分岐命令の次アドレスの命令を実行
  - 分岐成立では，飛び先アドレスの命令を実行（不成立では，分岐命令の次の次のアドレスの命令を実行）

戦略4：遅延分岐 (delayed branch)

- 分岐命令の後続の幾つかの命令を実行した後に，分岐する．

Untaken 分岐命令	IF	ID	EX	MEM	WB
分岐遅延命令 (*i* + 1)					
命令 *i* + 2					
命令 *i* + 3					
命令 *i* + 4					

Taken 分岐命令	IF	ID	EX	MEM	WB
分岐遅延命令 (*i* + 1)					
分岐先					
分岐先 + 1					
分岐先 + 2					

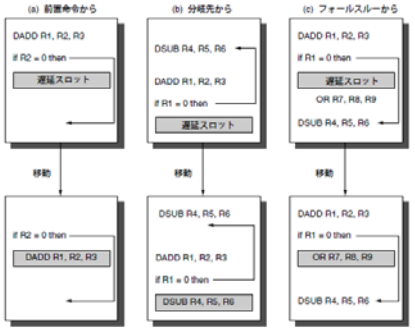
戦略4：遅延分岐 (delayed branch)

- 分岐命令の後続の幾つかの命令を実行した後に，分岐する．分岐命令によるストールは生じない．
- 初期のRISCプロセッサにて利用された．

Untaken 分岐命令	IF	ID	EX	MEM	WB			
分岐遅延命令 (*i* + 1)		IF	ID	EX	MEM	WB		
命令 *i* + 2			IF	ID	EX	MEM	WB	
命令 *i* + 3				IF	ID	EX	MEM	WB
命令 *i* + 4					IF	ID	EX	MEM WB

Taken 分岐命令	IF	ID	EX	MEM	WB			
分岐遅延命令 (*i* + 1)		IF	ID	EX	MEM	WB		
分岐先			IF	ID	EX	MEM	WB	
分岐先 + 1				IF	ID	EX	MEM	WB
分岐先 + 2					IF	ID	EX	MEM WB

遅延分岐スロットのスケジューリング



Nop命令

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

パイプラインの実行の困難さ：例外への対処

パイプラインステージ	起こり得る問題となる例外
IF	命令フェッチ時ページフォールト、不整列メモリアクセス、メモリ保護違反
ID	未定義・不法オペコード
EX	演算例外
MEM	データフェッチ時ページフォールト、不整列メモリアクセス、メモリ保護違反
WB	無し

LD	IF	ID	EX	MEM	WB
DADD		IF	ID	EX	MEM WB

パイプラインの実行の困難さ：例外への対処

1. 次の命令フェッチ時に，トラップ命令をパイプラインに挿入
2. トラップが実行されるまで，フォールトした命令とパイプライン中でそれに後続している命令による書き込みをすべて取りやめる．例外を生じた命令から始まるすべてのパイプライン中の命令に対して，パイプラインラッチにゼロを書き込むことで実現する．その命令より前の命令には施してはならない．この操作により，例外が対処されるまでの未完了の命令の状態を適切に設定する．
3. OSの例外ハンドラのルーチンが制御を獲得したあとで，そのルーチンはフォールトした命令のPCを直ちに保存する．この値は，後ほど例外から戻る時に使用．

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
- 講義用の計算機
  - 131.112.16.56 (情報工学科の演習室からは入れません)
  - ssh archo@131.112.16.56
    - mkdir myname
    - cd myname

計算機アーキテクチャ 第二 (O)

コンピュータの性能

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

計算機アーキテクチャへの要求

- **速度(実行時間)，スループット**
- 消費電力
- 発熱
- 音
- 価格
- 安定性，など

Which is faster?

Plane	DC to Paris	Speed	Passengers	Throughput (pmph)
Boeing 747	6.5 hours	610 mph	470	286,700
BAD/Sud Concorde	3 hours	1350 mph	132	178,200

- **Time to run the task (ExTime)**
  - Execution time, response time, latency
- **Tasks per day, hour, week, sec, ns … (Performance)**
  - Throughput, bandwidth

MPH (Mile Per Hour)

From the lecture slide of David E Culler

Defining (Speed) Performance

- Normally interested in reducing
  - Response time (execution time) – the time between the start and the completion of a task
    - Important to individual users
  - Thus, to maximize performance, need to minimize execution time

$$\text{performance}_X = 1 / \text{execution\_time}_X$$

If X is n times faster than Y, then

$$\frac{\text{performance}_X}{\text{performance}_Y} = \frac{\text{execution\_time}_Y}{\text{execution\_time}_X} = n$$

  - Throughput – the total amount of work done in a given time
    - Important to data center managers
  - Decreasing response time almost always improves throughput

Performance Factors

- Want to distinguish elapsed time and the time spent on our task
- CPU execution time (CPU time) – time the CPU spends working on a task
  - Does not include time waiting for I/O or running other programs

$$\text{CPU execution time for a program} = \text{\# CPU clock cycles for a program} \times \text{clock cycle time}$$

or

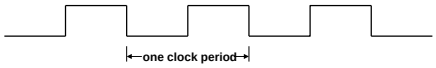
$$\text{CPU execution time for a program} = \frac{\text{\# CPU clock cycles for a program}}{\text{clock rate}}$$

- Can improve performance by reducing either the length of the clock cycle or the number of clock cycles required for a program

Review: Machine Clock Rate

- Clock rate (MHz, GHz) is inverse of clock cycle time (clock period)

Clock rate = 1 / Clock period



10 nsec clock cycle	=>	100 MHz clock rate
5 nsec clock cycle	=>	200 MHz clock rate
2 nsec clock cycle	=>	500 MHz clock rate
1 nsec clock cycle	=>	1 GHz clock rate
500 psec clock cycle	=>	2 GHz clock rate
250 psec clock cycle	=>	4 GHz clock rate
200 psec clock cycle	=>	5 GHz clock rate

MIPS (Million Instructions Per Second)

- 1秒当たりに実行された命令の数（単位はMillion）
  - 原始MIPS (native MIPS)
- 注意
  - プロセッサアーキテクチャのMIPSとは関係ない
- MIPSの問題点とは？
  - 命令セットに強く依存する尺度
  - 異なる命令セット，NOP，コンパイラ，性能？

MFLOPS, GFLOPS

- MFLOPS (Million Floating-point Operations Per Second)
- GFLOPS (Giga Floating-point Operations Per Second)
- MIPSとGFLOPSとの相違は？
  - 命令セット，浮動小数点演算

先端マイクロプロセッサ Cell Broadband Engine

- ヘテロジニアス チップマルチプロセッサ
  - PowerPC Processor Element (PPE) 1個
  - Synergistic Processor Element (SPE) 8個

PlayStation3 の写真は PlaySation.com (Japan) から

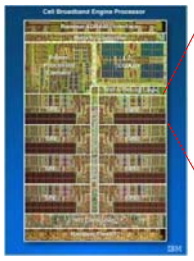


Diagram created by IBM to promote the CBEP, ©2005 WIKIPEDIAより

I0

Cell/B.E. Element Interconnect Bus

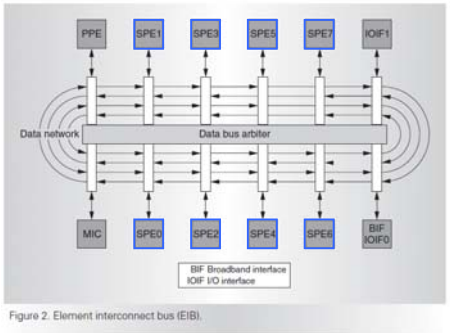


Figure 2. Element interconnect bus (EIB).

IEEE Micro, Cell Multiprocessor Communication Network: Built for Speed

Cell Broadband Engine

- ピーク性能
  - 1サイクルで積和演算を1回実行できる演算器 (2 FLOP/cycle)
  - SIMD構成で，SPEあたりの並列性 4
  - チップ内のSPEの数 8
  - 動作周波数 4GHz
  - 2 × 4 × 8 × 4 = 256 GFLOPS
  - 積和演算 × SIMD化 × マルチコア × 動作周波数
    - **ペンティアムは 8GFLOPS 程度**
  - 性能を引き出す鍵は
    - DMA転送とローカルストアの使い方，SIMD化，並列化 . . .