計算機アーキテクチャ特論

安藤秀樹
電子情報システム専攻
http://www.ando.nuee.nagoya-u.ac.jp/

トピック

- 命令の並列処理を行うプロセッサのアーキテクチャとコンパイラ
  - スーパスカラ・プロセッサ
    - 現在のPC、モバイル端末、サーバのプロセッサ
  - VLIW
    - DSP (Digital Signal Processor)
    - Intel Itanium
- 教科書
  - 安藤秀樹、“命令レベル並列処理ープロセッサアーキテクチャとコンパイラ”
  - 講義だけ聴いていても理解できない→本で復習すること

講義関連

- スライドのコピー
  - http://www.ando.nuee.nagoya-u.ac.jp/~ando/aca/
  - 講義の前までに準備
  - ダウンロード、印刷して持参

- 掲示
  - HP

関係のある他のコースと成績評価

- 先にとっておいた方が好ましい講義
  - P&H:「コンピュータの構成と設計（上・下）」を教科書に使う講義
    - 計算機工学
    - 計算機アーキテクチャ
- 評価
  - 期末試験
  - A4用紙1枚（表裏）の「メモ」持ち込み可

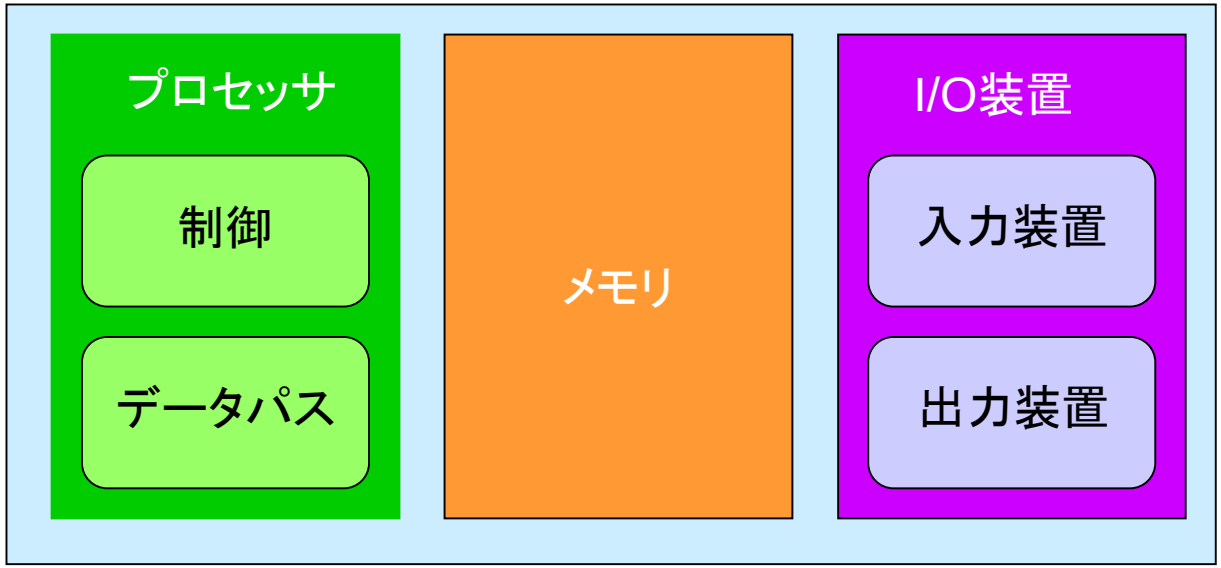
暫定スケジュール

1	復習（基本的動作）
2	スーパスカラの基本構成
3	動的命令スケジューリング
4	リオーダ・バッファ
5	レジスタ・リネーミング
6	分岐予測
7	高バンド幅命令フェッチ、投機的実行支援
8	Intel Pentium 4
9	VLIWと局所命令スケジューリング
10	レジスタ割り当て、広域命令移動の基礎
11	非循環広域命令スケジューリング、ループ・アンローリング
12	期末試験

内容

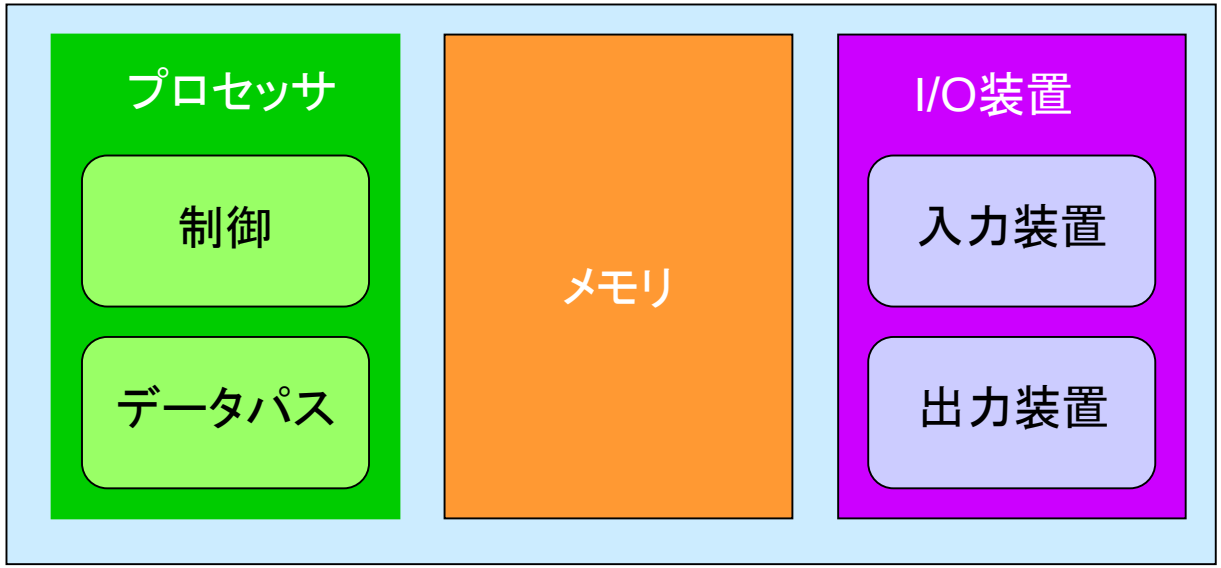
- 復習
  - コンピュータの構成
  - 単一サイクルプロセッサ
  - パイプライン処理

コンピュータの構成



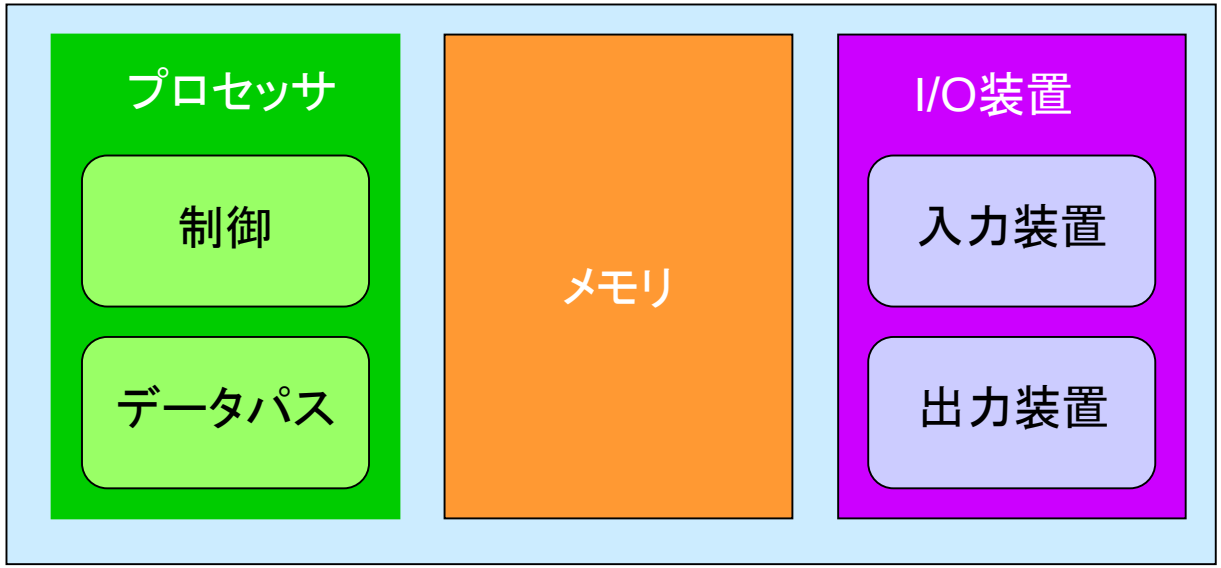
メモリ

- 実行中のプログラム
- 実行中のプログラムが必要とするデータ



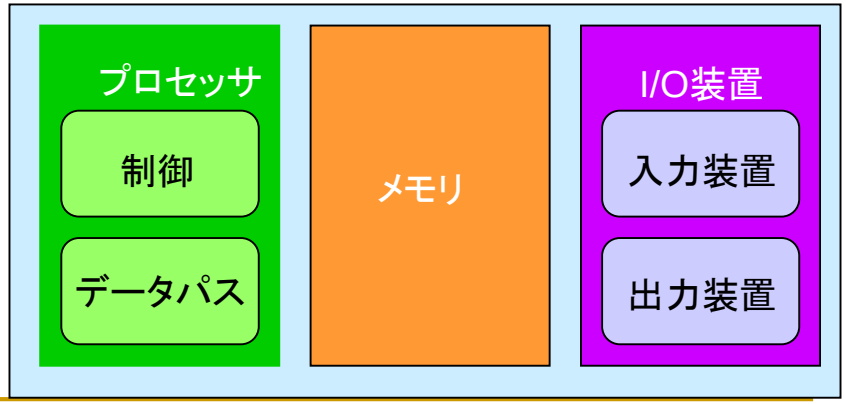
I/O装置

- キーボード、マウス、ディスプレイ、ディスク…

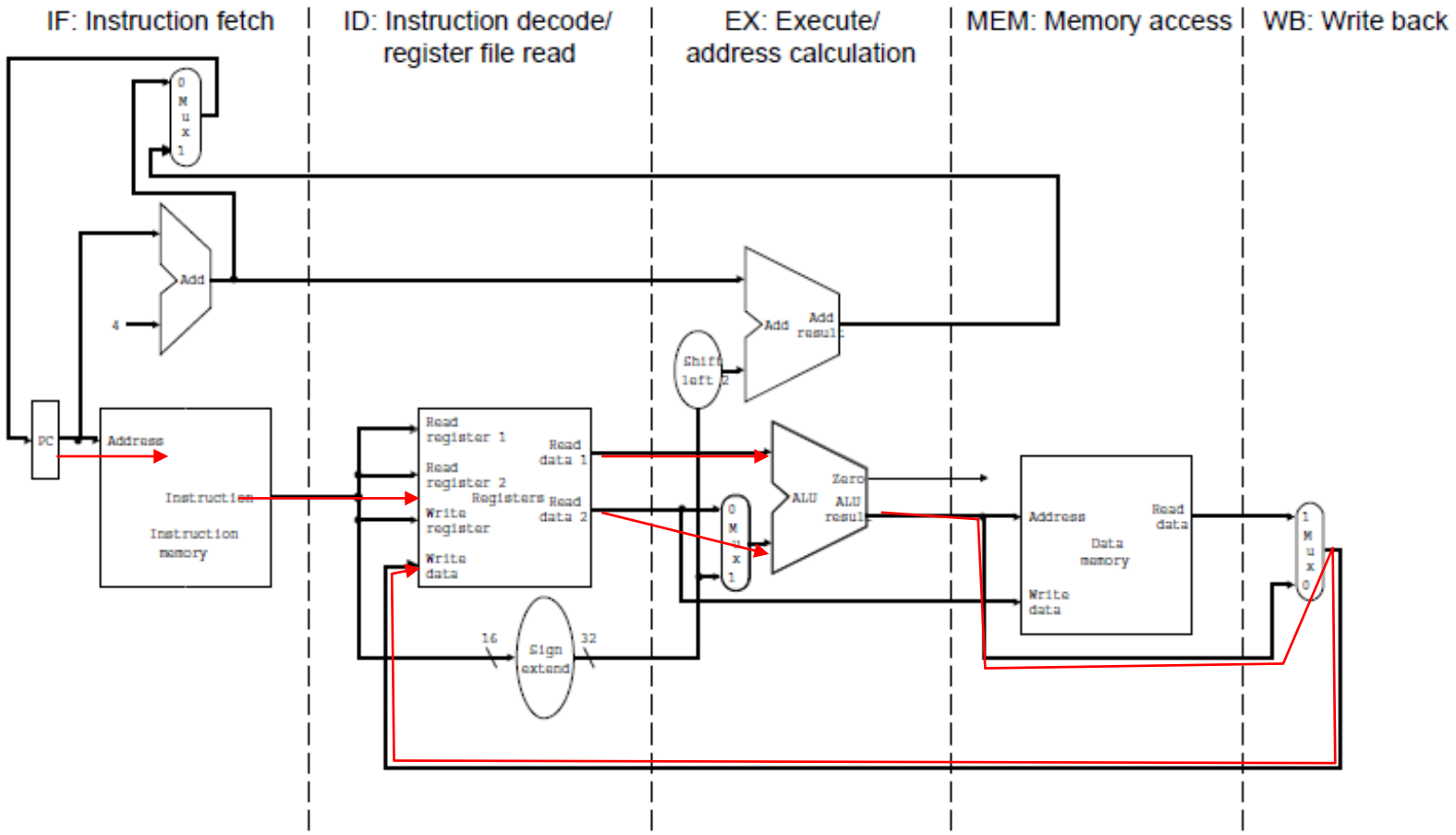


プロセッサ

- プログラムの命令に従って動作する
- 中央演算処理装置 (CPU: central processor unit) とも呼ばれる
- データパス（datapath）: 命令の処理
  - 手足のようなもの
  - 演算、メモリへの（からの）データ転送、I/Oへの信号、…
- 制御（control）
  - 脳・神経系のようなもの
  - データパス、メモリ、I/Oに何をなすべきかを伝える

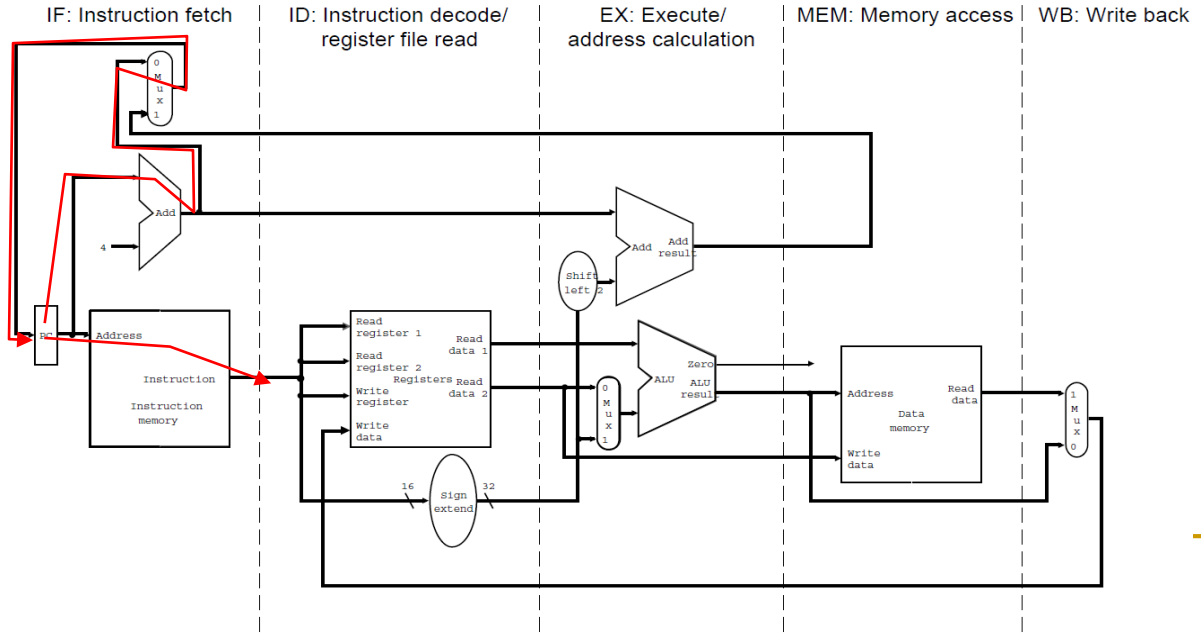


単一サイクル制御方式


IF: Instruction fetch
ID: Instruction decode/ register file read
EX: Execute/ address calculation
MEM: Memory access
WB: Write back
M u x
Add
4
PC
Address
Instruction
Instruction memory
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
Shift left 2
Add
Add result
ALU
Zero
ALU result
Address
Data memory
Read data
Write data
16
Sign extend
32

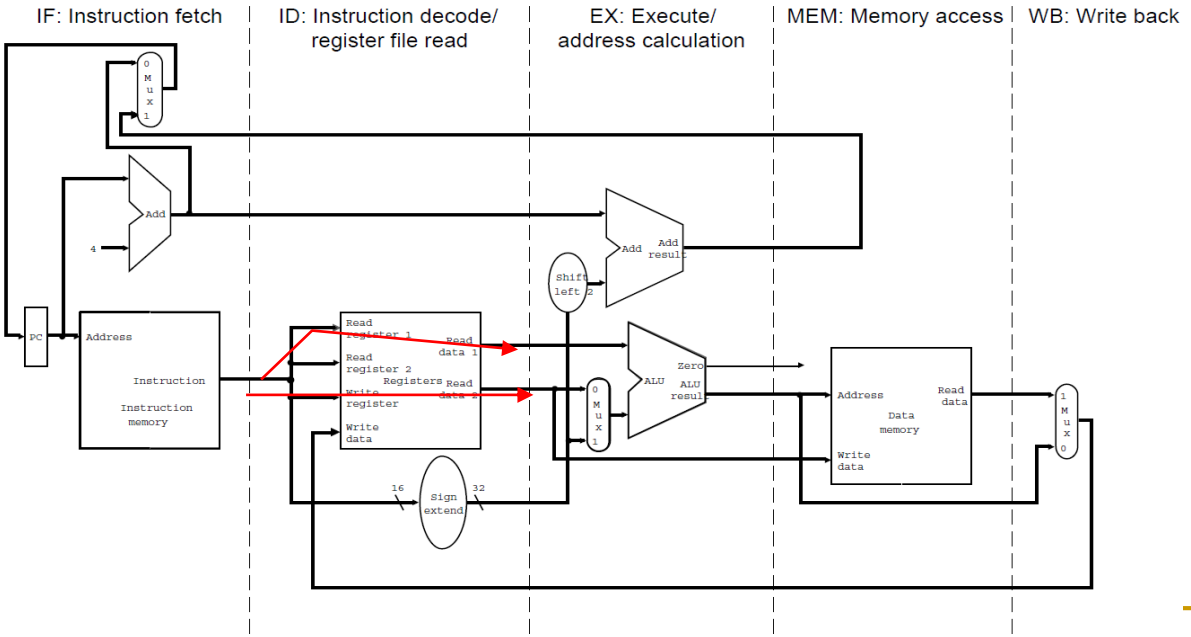
命令フェッチ

- IF: instruction fetch
  - PC（プログラム・カウンタ）
    - 命令アドレス
    - 実行する命令を保持しているメモリ内の位置
  - 命令を命令キャッシュから読み出す
    - キャッシュ：メモリの一部を保持する高速なバッファ
  - PC←PC+4



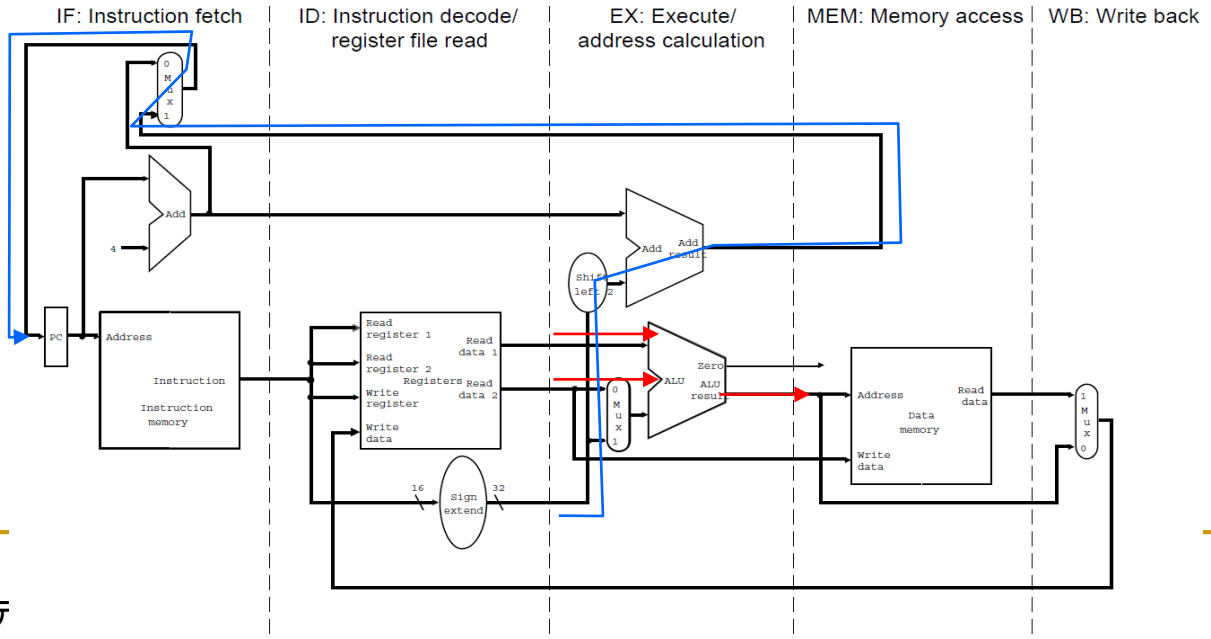
命令デコードとレジスタの読み出し

- ID: instruction decode/register read
  - 制御信号の生成
  - レジスタ･ファイルをレジスタ指定子で参照



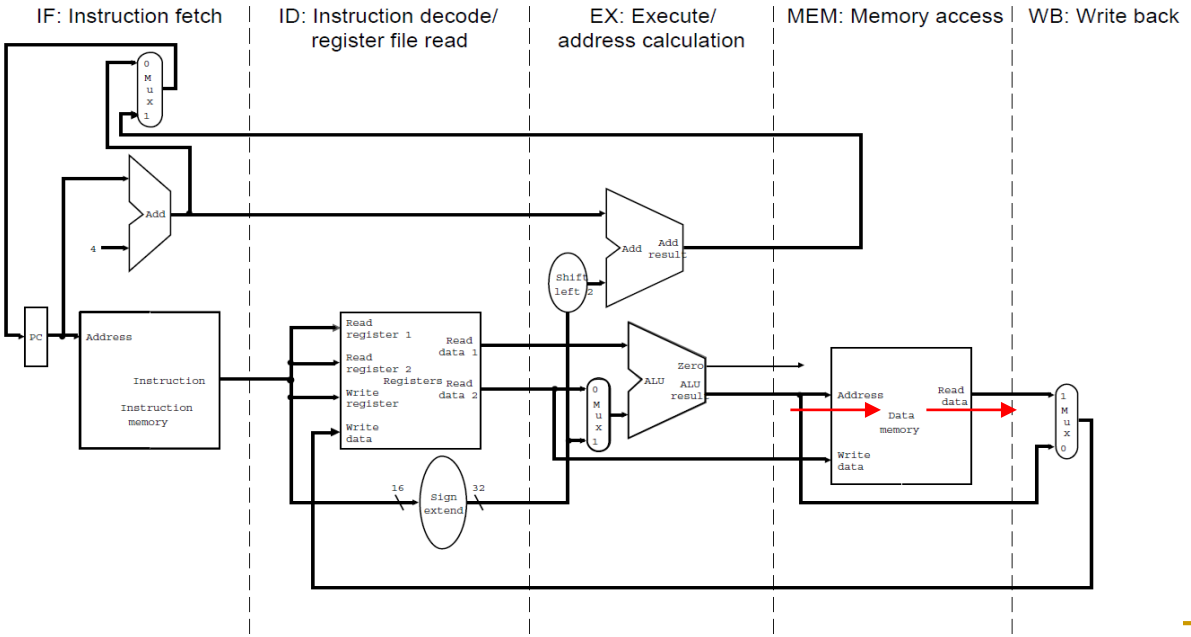
実行またはアドレス計算

- EX: execution / address calculation
  - ALU実行
    - ALU：算術論理演算ユニット(arithmetic logic unit)
  - ロード／ストアのデータ・アドレス計算
  - 分岐先の計算



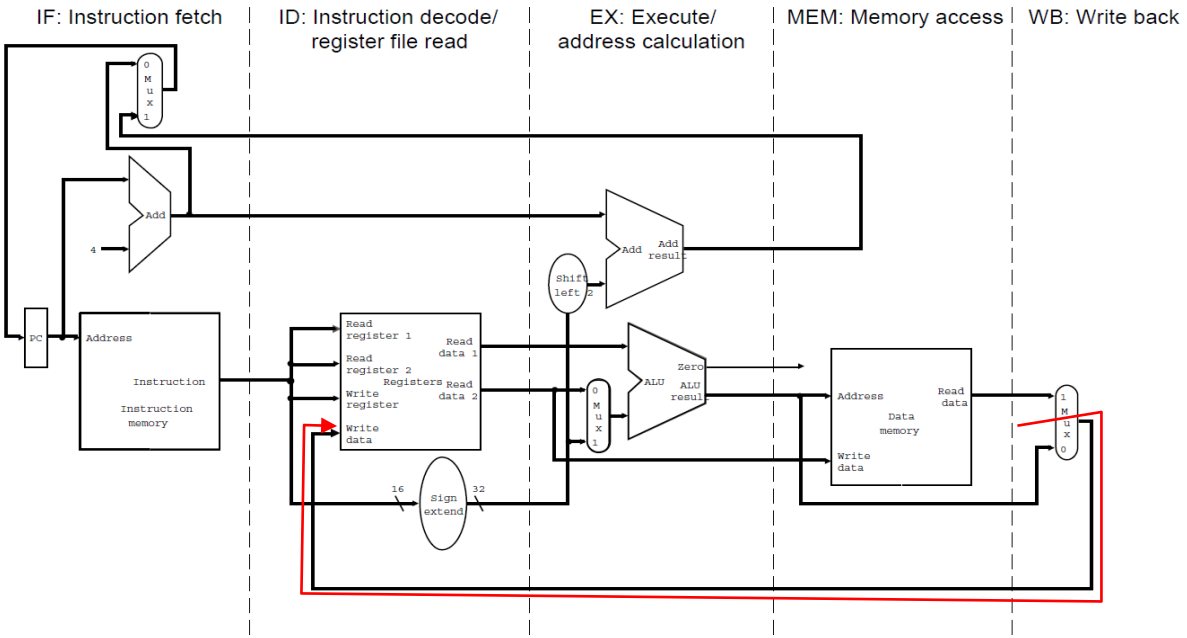
メモリ・アクセス

- MEM: memory access
  - ロード：メモリの読み出し
  - ストア：メモリの書き込み

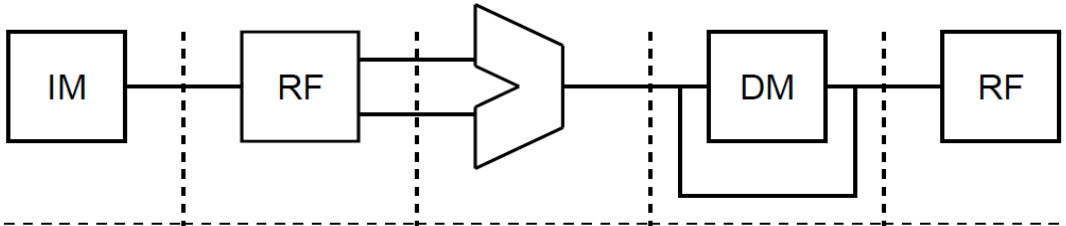


書き戻し

- WB: write back
  - レジスタへの書き込み



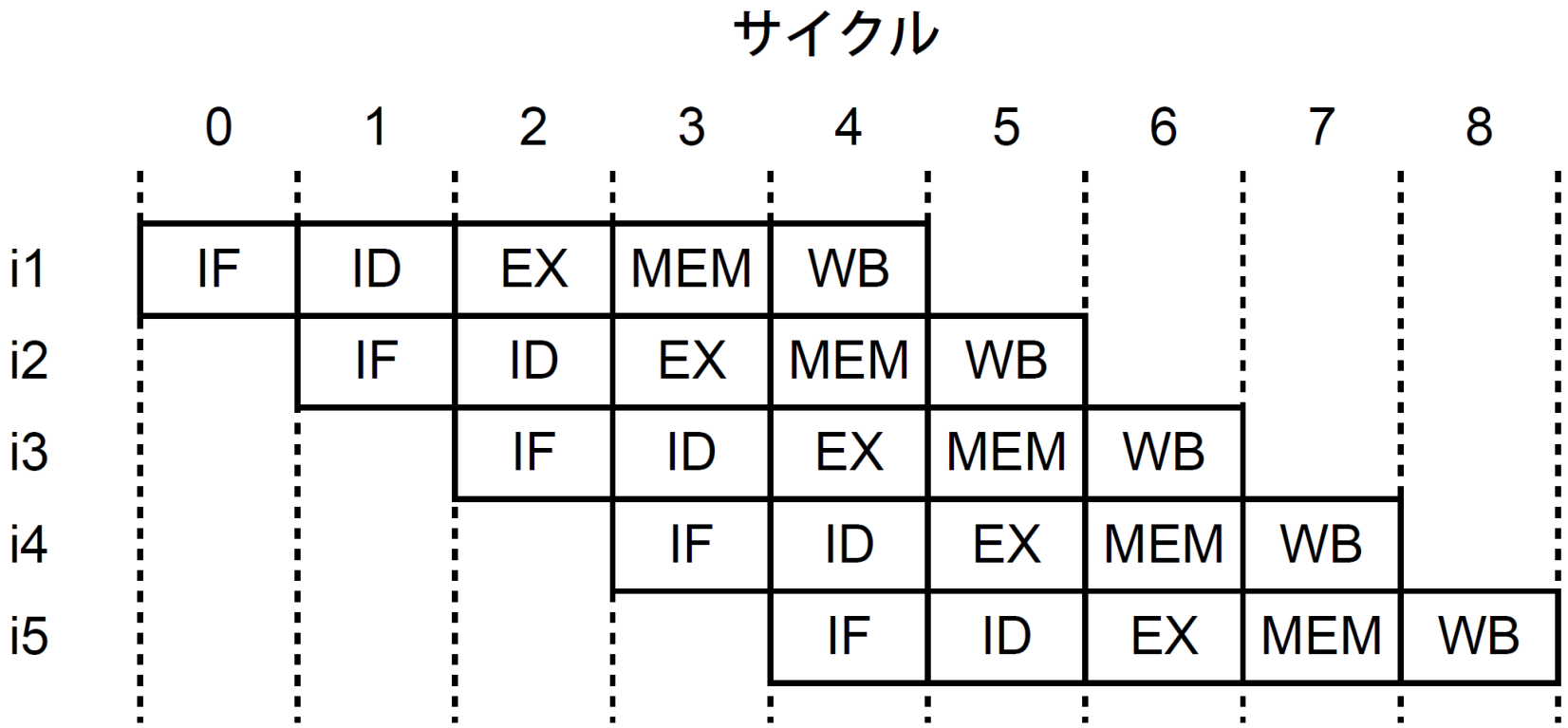
パイプライン処理



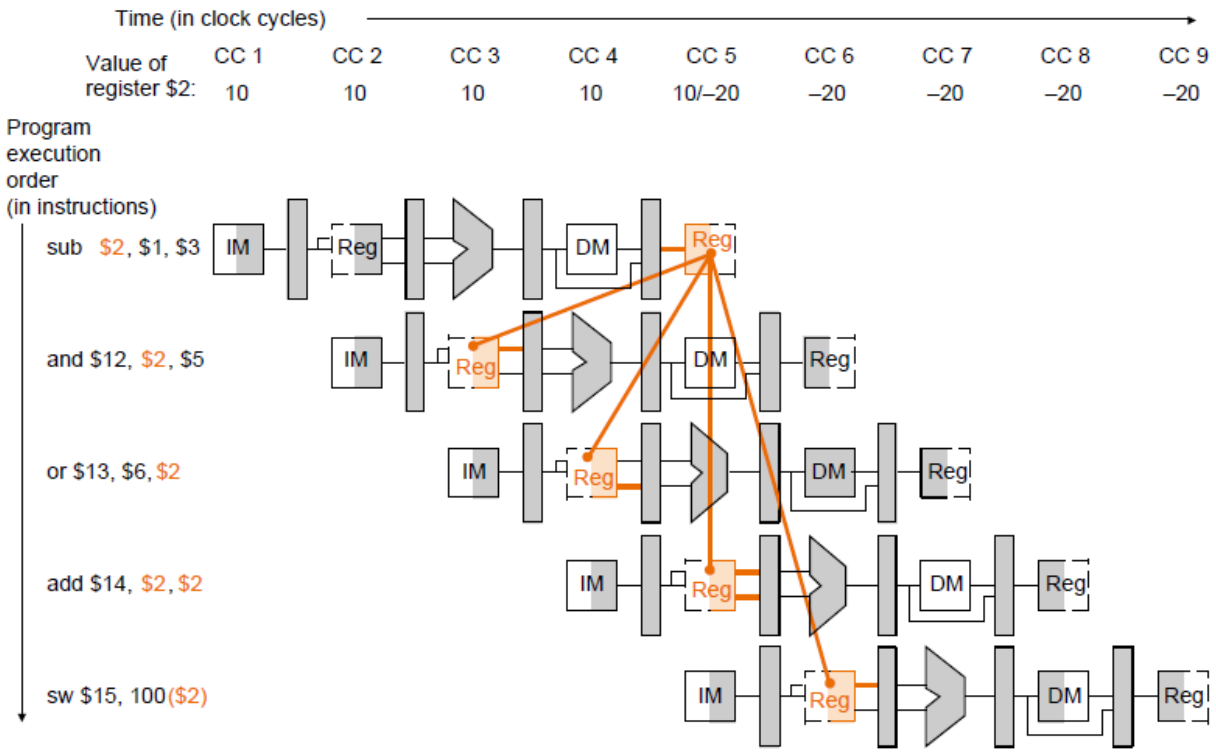
サイクル					
1	i1				
2	i2	i1			
3	i3	i2	i1		
4	i4	i3	i2	i1	
5	i5	i4	i3	i2	i1
6	i6	i5	i4	i3	i2
7	i7	i6	i5	i4	i3
8	i8	i7	i6	i5	i4
9	i9	i8	i7	i6	i5

- 命令の実行が終了する前に、次の命令の実行を開始し、複数の命令をオーバラップして実行する方式
  - 流れ作業
  - 同時に5つの命令がオーバラップ→時間的並列実行

パイプライン処理（別の表現）

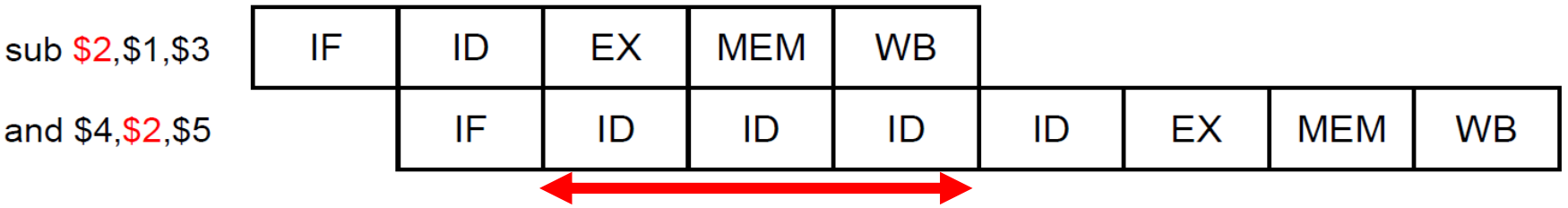


パイプライン処理における問題点



- タイミングによっては正しい結果は得られない
- subの結果を使えるのはいつ？

パイプライン・ストール



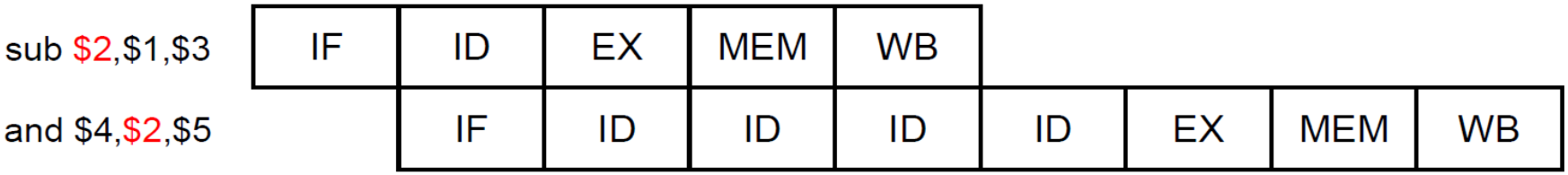
- パイプラインをとめてハザードが解消されるのを待つ

パイプライン・ハザード

- パイプラインは必ずしも理想的には流れない
  - 状況によっては、ストール（停止）させる必要がある
  - 大きな性能低下
- パイプライン・ハザード
  - パイプラインのステージを停止させなければならに状況
  - 3種
    - データ・ハザード： データ依存によるハザード
    - 制御ハザード： 制御依存によるハザード
    - 構造ハザード： 資源競合によるハザード

データ・ハザード

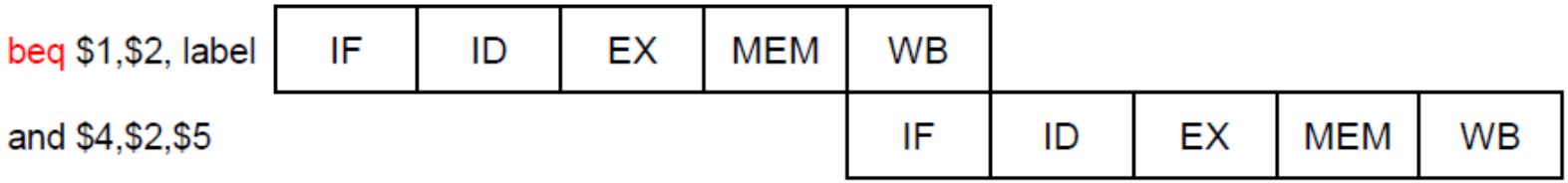
- データ依存によるハザード



  - 定義と参照の関係
  - 定義されなければ参照できない
- 関係のあるステージ
  - レジスタ： WB→ID
  - メモリ： MEM→MEM

制御ハザード

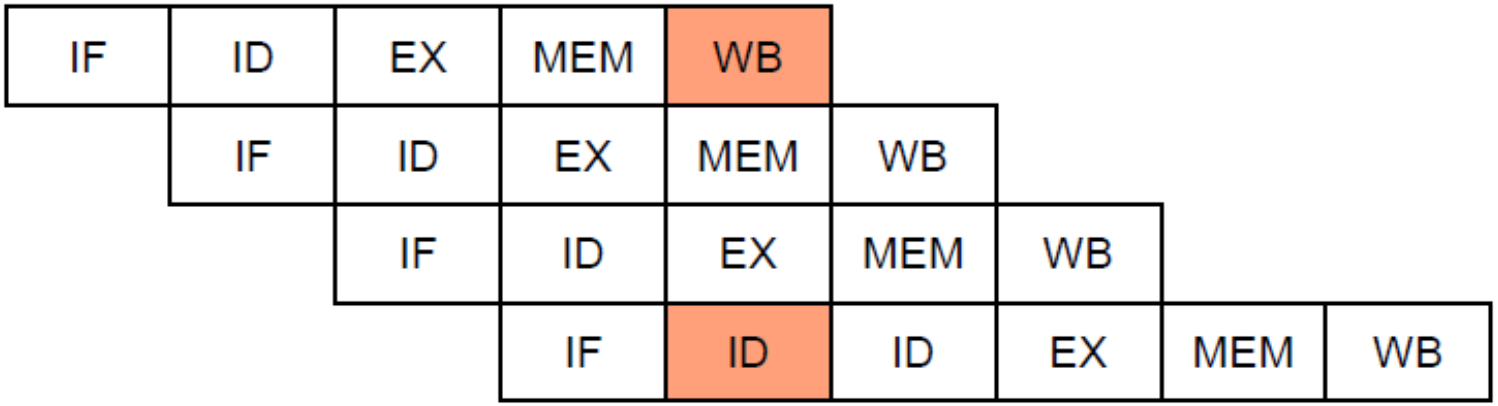
- 制御依存によるハザード



  - 分岐があると次に実行すべき命令がわからない
- 関係のあるステージ
  - MEM→IF

構造ハザード

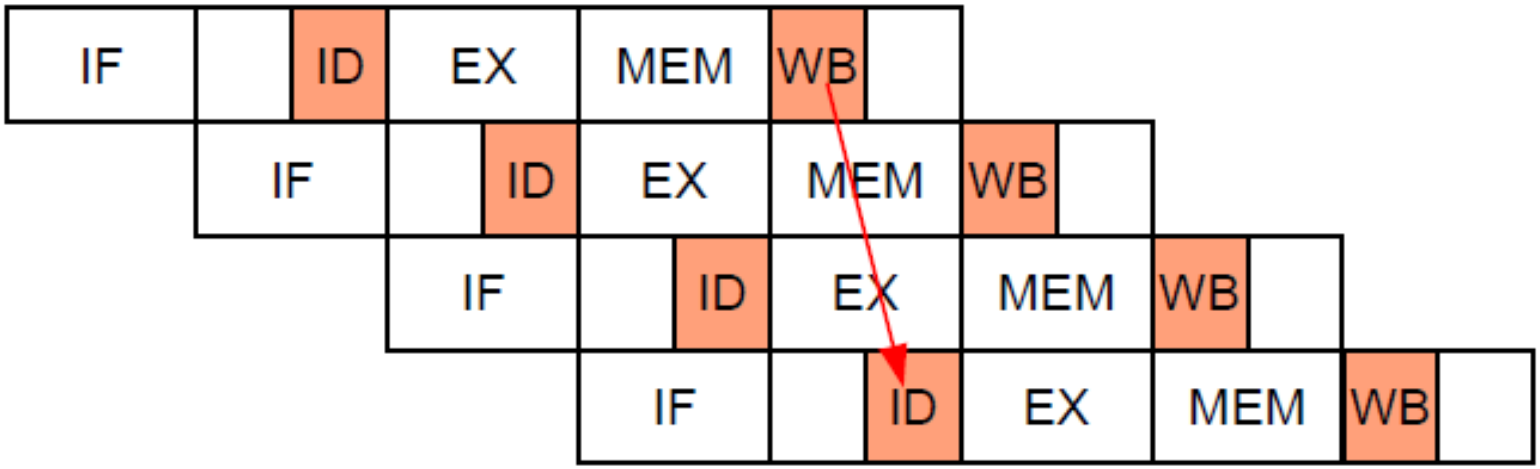
- 資源競合によるハザード



  - 資源(部品)を複数の命令が要求する
  - 要求＞応答能力
    - 先の命令が優先、後の命令は待たされる
- 関係あるステージ
  - WB→ID

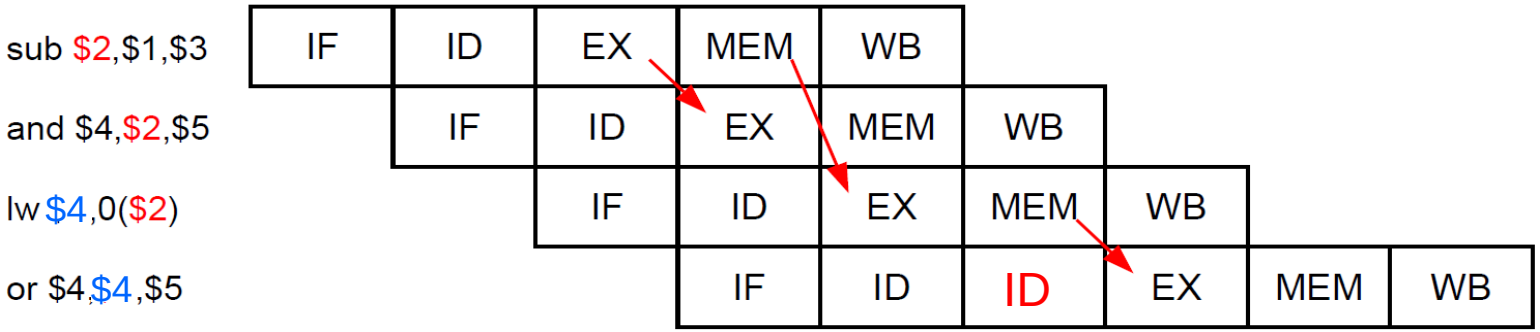
レジスタ・アクセス構造ハザード解消

- 時分割
  - 前半で書く
  - 後半で読む
  - レジスタファイルアクセス時間≦0.5×クリティカル・パス遅延



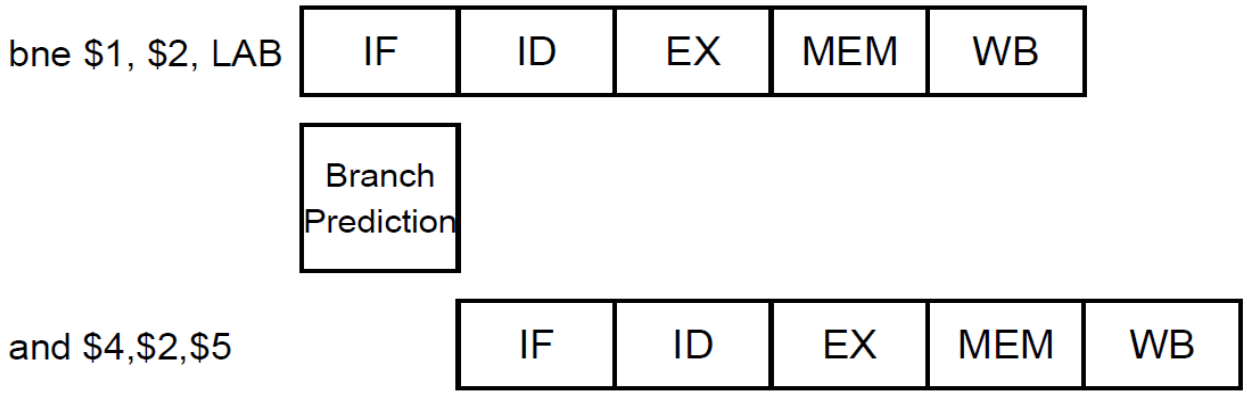
データ・ハザード解消

- オペランド・フォワーディング
  - 実行結果を書き戻す前に、後続の命令にデータを渡す
  - 次の命令が待ち合わせる時間＝実行の時間だけ
    - R形式命令のレイテンシ＝1サイクル
    - レイテンシ：要求から応答までの時間（遅延）
  - load→useはストール
    - レイテンシ＝2サイクル
    - 遅延ロードと命令スケジューリング



制御ハザードの解消

- 分岐予測



- 過去の分岐の振る舞いから将来の分岐結果を予測
- 分岐方向予測
  - 1ビット予測、2ビット・カウンタ予測
- 分岐先予測
  - 分岐先バッファ (BTB: branch target buffer)
- 予測を誤れば、
  - パイプラインを無効化
  - 正しい方向の命令をフェッチ

まとめ

- コンピュータの構成
- 単一サイクルプロセッサ
  - 単純
  - 最も長い処理時間に合わせたサイクル時間
- パイプライン処理
  - 命令のオーバラップ実行
  - 3種のハザード