コンピュータアーキテクチャ（8）

坂井　修一
東京大学大学院 情報理工学系研究科 電子情報学専攻
東京大学　工学部　電子情報工学科／電気電子工学科

- はじめに
- 命令レベル並列処理（１）

はじめに

- 本講義の目的
  - コンピュータアーキテクチャの基本を学ぶ
- 時間・場所
  - 火曜日 8:40 – 10:10　工学部2号館241
- ホームページ（ダウンロード可能）
  - url:　http://www.mtl.t.u-tokyo.ac.jp/~sakai/hard/
- 教科書
  - **坂井修一『コンピュータアーキテクチャ』（コロナ社、電子情報レクチャーシリーズC-9）**
  - **坂井修一『実践　コンピュータアーキテクチャ』（コロナ社）**

  教科書通りやります
- 参考書
  - D. Patterson and J. Hennessy, Computer Organization & Design、3rd Ed.（邦訳『コンピュータの構成と設計』(第3版)上下（日系ＢＰ））
  - 馬場敬信『コンピュータアーキテクチャ』（改訂２版）、オーム社
  - 富田眞治『コンピュータアーキテクチャⅠ』、丸善
- 予備知識：　論理回路
  - 坂井修一『論理回路入門』、培風館
- 成績
  - 試験＋レポート＋出席

講義の概要と予定（１／２）

１．コンピュータアーキテクチャ入門

ディジタルな表現、負の数、実数、加算器、ＡＬＵ，フリップフロップ、レジスタ、計算のサイクル

２．　データの流れと制御の流れ

主記憶装置、メモリの構成と分類、レジスタファイル、命令、命令実行の仕組み、実行サイクル、算術論理演算命令、シーケンサ、条件分岐命令

３．命令セットアーキテクチャ

操作とオペランド、命令の表現形式、アセンブリ言語、命令セット、
算術論理演算命令、データ移動命令、分岐命令、アドレシング、
サブルーチン、RISCとCISC

４．パイプライン処理（１）

パイプラインの原理、命令パイプライン、オーバヘッド、構造ハザード、データハザード、制御ハザード

５．パイプライン処理（２）

フォワーディング、遅延分岐、分岐予測、命令スケジューリング

６．キャッシュ

記憶階層と局所性、透過性、キャッシュ、ライトスルーとライトバック、
ダイレクトマップ型、フルアソシアティブ型、セットアソシアティブ型、キャッシュミス

7. 仮想記憶

仮想記憶、ページフォールト、TLB、物理アドレスキャッシュ、仮想アドレスキャッシュ、メモリアクセス機構

講義の概要と予定（２／２）

８． 基本CPUの設計

ディジタル回路の入力、Verilog HDL、シミュレーションによる動作検証、アセンブラ、基本プロセッサの設計、基本プロセッサのシミュレーションによる検証

９．命令レベル並列処理（１）

並列処理、並列処理パイプライン、VLIW、スーパスカラ、並列処理とハザード

１０．命令レベル並列処理（２）

静的最適化、ループアンローリング、ソフトウェアパイプライニング、トレーススケジューリング

１１．アウトオブオーダ処理

インオーダーとアウトオブオーダー、フロー依存、逆依存、出力依存、
命令ウィンドウ、リザベーションステーション、レジスタリネーミング、
マッピングテーブル、リオーダバッファ、プロセッサの性能

１２．入出力と周辺装置

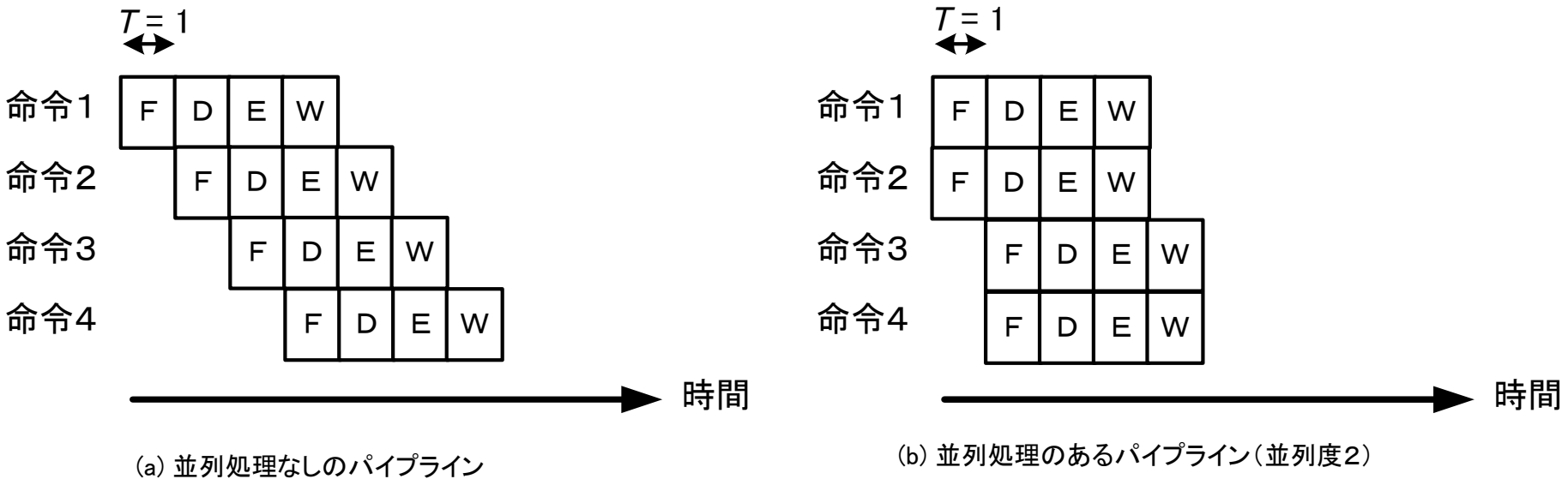
周辺装置、ディスプレイ、二次記憶装置、ハードウェアインタフェース、割り込みとポーリング、アービタ、DMA、例外処理

試験： ７月後半

9. 命令レベル並列処理

- 内容
  - 命令レベル並列処理
    - 命令レベル並列処理とはなにか
    - 並列処理パイプライン
  - VLIW
    - VLIWプロセッサの構成と動作
    - VLIWの特徴
  - スーパスカラ
    - スーパスカラプロセッサの構成と動作
    - 並列処理とハザード
    - VLIWとスーパスカラの比較

命令レベル並列処理とはなにか

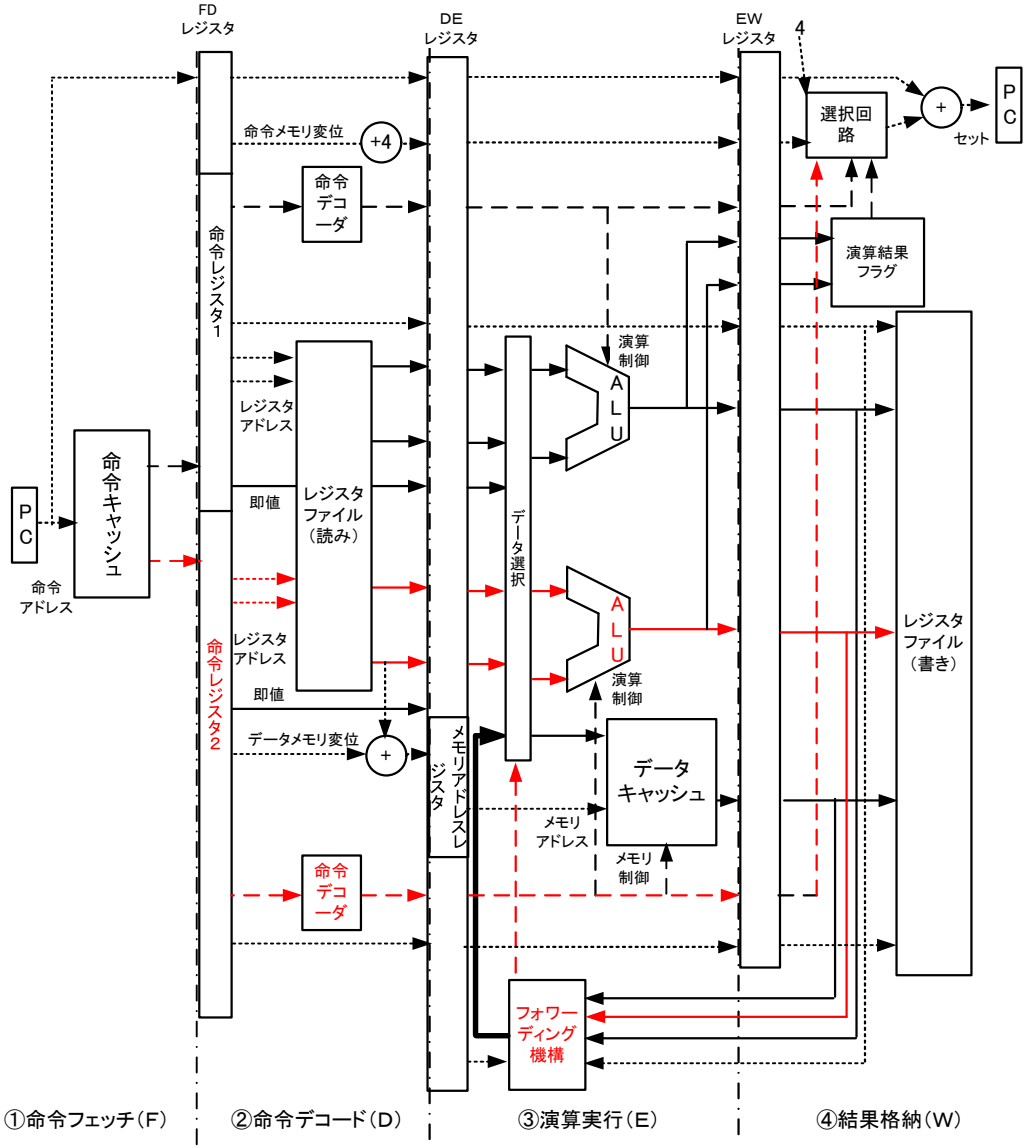


(a) 並列処理なしのパイプライン

(b) 並列処理のあるパイプライン（並列度2）

F: 命令フェッチ、 D:命令デコード、 E:演算実行、 W:結果格納

並列処理パイプライン


FD
レジスタ
DE
レジスタ
EW
レジスタ
4
選択回路
セット
PC
命令メモリ変位
+4
命令
デコーダ
命令レジスタ1
演算結果
フラグ
演算
制御
ALU
レジスタ
アドレス
即値
レジスタ
ファイル
（読み）
データ選択
PC
命令
アドレス
命令キャッシュ
命令レジスタ2
レジスタ
アドレス
即値
ALU
演算
制御
レジスタ
ファイル
（書き）
データメモリ変位
メモリアドレスレジスタ
データ
キャッシュ
メモリ
アドレス
メモリ
制御
命令
デコーダ
フォワーディング機構
①命令フェッチ（F）
②命令デコード（D）
③演算実行（E）
④結果格納（W）
アドレスの流れ
制御の流れ
データの流れ

命令レベル並列処理に必要な事項

- ハードウェア資源の投入
  - 命令キャッシュのバス幅、パイプラインレジスタ、デコーダ、演算フラグなどが、並列度Pに比例して大きくなる
- レジスタファイルのポート数
  - 読み出し用のポート数が$2P$、書き込み用のポート数がP、合計$3P$のポートが必要となる。
- フォワーディング機構
  - 前後の命令の間のデータハザードの解消のためには、演算ユニットのそれぞれの出力の データがすべての演算ユニットの入力にフォワードされなければならない。フォワードのデータ 線に加えてマルチプレクサのために$P \times P = P^2$に比例するハードウェアが必要となる。さらに 遅延も大きなものとなる危険がある。
- 並列実行の制御
  - 同時実行する命令間の依存関係がないことを保証する必要がある。依存関係のある命令が同時にフェッチされた場合、どちらかを待たせるなどの工夫が必要である。並列実行される命 令間のデータハザードはフォワーディングによっては解消されないことに注意。

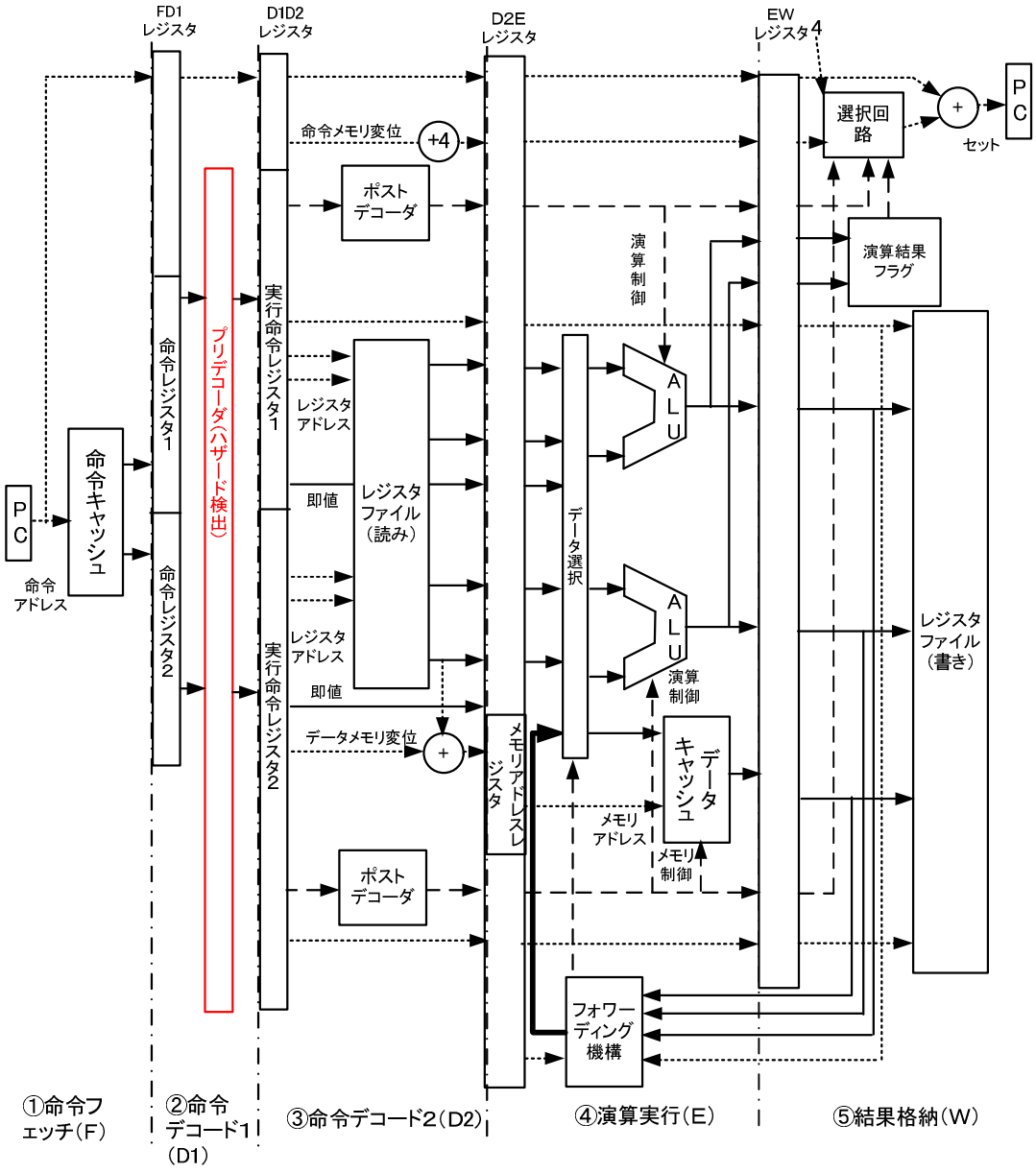
VLIW

- VLIW = Very Large Instruction Word
  - １命令の中に複数の演算を入れたアーキテクチャ
  - 同一命令語中のハザードはすべてコンパイラ(または機械語プログラマ)が静的に解決し、命令語の中の演算はすべて同時に実行する
- VLIWの利点
  - プロセッサの制御ロジックが簡単(高速)
- VLIWの問題点
  - 透過性・互換性がない
  - 静的な並列化の限界
  - 十分に並列化できない場合の命令フィールドの無駄

スーパスカラ

- スーパスカラ(superscalar)
  - 逐次型プログラムを並列実行するアーキテクチャ
    - ハードウェアがみずから並列性を検出し利用する
- 利点
  - 透過性・互換性が維持される
- 問題点
  - ハードウェアの複雑化

スーパスカラプロセッサの構成



スーパスカラプロセッサの動作

①命令フェッチ(instruction fetch, F)

- 命令キャッシュから複数の命令をフェッチする

②命令プリデコード(instruction predecode, D1)

- フェッチした命令と処理待ちの命令のすべての依存関係を調べ、もし依存関係があれば処理を遅らせる。依存関係のない1つないし複数(図では2つ)の命令を「実行命令レジスタ」に入れる。

③命令ポストデコード(instruction dispatch, D2)

- 実行命令レジスタに入った命令から、演算装置やメモリの制御信号を生成する。同時に、レジスタファイルから演算に必要なレジスタの値を読み出す。

④演算実行(execution, E)

- ポストデコーダで指定された演算群(メモリの読み書きを含む)を同時実行する。結果の格納場所の選択信号をポストデコーダで指示された値にセットする。

⑤結果の格納(write back, W)

- レジスタファイルに実行結果を格納する。プログラムカウンタ(PC)の値を次の命令のためにセットする。

並列処理とハザード

(1)構造ハザード

- 並列処理では、ユニット数やポート数が足りないための構造ハザードが起こる。
- たとえば、データキャッシュが一度にひとつしかアクセス要求を受け付けないとき、ロード・ストア命令を同時に2つ以上実行することはできない。
- 構造ハザードがある場合は、競合する資源を使う命令を、時間をずらして順番に処理することになる。スーパスカラでは、この制御をハードウェアが行う必要がある。

(2)データハザード

- 並列処理する2つの命令の間にはデータ依存関係があってはならない
- プリデコードのステージでデータ依存関係を発見したら、プログラムの中で前に出てくる命令を先に処理し、後の命令は時間をずらして後から処理することになる。

(3)制御ハザード

- フェッチした命令のどちらかが分岐命令の場合、制御ハザードが起こる可能性がある。
- 並列処理をする場合、ストールの影響は大きくなる。遅延分岐はたくさんの共通命令を必要とすることになるし、分岐予測がはずれた場合のペナルティも大きい。

VLIWとスーパスカラの比較

表 6.1　VLIWとスーパスカラの比較

	ＶＬＩＷ	スーパスカラ
透過性・互換性	×	○
ハザード検出・並列化	静的（コンパイラ）	動的（ハードウェア）
ハードウェア	簡単	複雑
制御の遅延	小	大
命令フィールドのむだ	有	無