前回:
プログラム例(2), サブルーチン, 機械命令, 合成命令

今回:
コンピュータ開発の歴史
コンピュータ性能評価
CPUの一般論
アドレシングモード

キーワード:
マイクロプロセッサ, RISC, CISC, MIPS値, オペランド指定
ロードストアアーキテクチャ
アドレシングモード, 実効アドレス, インデクスレジスタ,
ベースレジスタ, 直接/間接, 相対, 修飾

第9回 コンピュータの一般論

ここまで, MIPSプロセッサを例として,
具体的なコンピュータについて説明してきた.

ここで, コンピュータの一般論を
まとめておく.

5.1 コンピュータ開発の歴史

最初のコンピュータ(諸説あり)
5.1.1 ENIAC (1946)
  ペンシルベニア大
  Eckert, Mauchly
  真空管による電子式
  プログラム外部供給方式(外部で配線)
  のちプログラム内蔵方式(書き換え不可)

5.1.2 EDSAC (1949)
  ケンブリッジ大
  Wilkes
  プログラム可変内蔵方式(書き換え可能)
  ソフトウェア開発の重要性
  レジスタはひとつだけ
    アキュムレータアーキテクチャ(後述)
  現代につながる技術がいろいろ
    サブルーチン (手続き、関数)
    絶対アドレス、相対アドレス(後述)
    インデックスレジスタ(後述)

5.1.3 マイクロプロセッサ
CPUを単一チップのLSI, VLSIで
  LSI トランジスタ数 1000-10万
  VLSI トランジスタ数 10万-1000万

  intel 4004 (1971) トランジスタ数 2300
  Pentium4 (Prescott) (2004) トランジスタ数 1億2500万
  Core i7 (Six-Core, Gulftown) (2010) トランジスタ数 11億

  ムーアの法則 (Moore's Law)
  集積回路の集積度は1年で倍増(1965)
  集積回路の集積度は2年で倍増(1975)に修正

5.2 RISCとCISC

次第にCPUの構成が複雑になってきた
  複雑化した命令
    ひとつの命令で多種の処理を行う
  データ長の拡大(8->16->32)にともなう互換性問題
  バイト単位の可変長命令形式
    命令読み出しと解読が複雑化
結果として個々の命令の実行が高速化しにくくなった

RISC (reduced instruction set computer)
  801 (1975-)
  MIPS (1986)
  SPARC (1987)
単純な命令を高速で実行しよう
  1命令1クロックサイクル
  ロードストアアーキテクチャ(後述)
  3オペランド形式(後述)
  固定長命令
    解読, 実行が高速に可能
  基本的、汎用的な命令のみ
  パイプライン処理(今後の授業で説明)
  コンパイラ重視
それまでのCPUをCISC (complex instruction set computer)

現代ではRISC/CISCは歩み寄ってきた

5.3 性能比較
比較の指標はいろいろある

プログラムのCPU実行時間 = 命令数×平均CPI×クロックサイクル時間
  CPI (clock cycle per instruction)
  クロックサイクル時間 = クロック周波数の逆数
クロック: 計算機の動作のもとになる周期的パルス信号

MIPS値
  million instructions per second
  問題点
    命令によってCPI値が異なる
      MIPS値も命令により異なる
    命令の機能の高低に差がある
      同じ計算をするにも命令数が異なる
  実際のCPU性能を反映しない場合もあり

ベンチマークテスト
  性能計測用のプログラムのセット
    例 SPEC (Standard Performance Evaluation Corporation)
  利用目的に適したベンチマークを選ぶ必要

5.4 CPUの一般論
分類にいろいろな観点がある

5.4.1 命令のオペランド指定
命令が持つオペランドの個数による分類
オペランド - 命令の演算対象
  レジスタ
  メモリのアドレス
  命令中の定数(即値 immediate)

(1) 1オペランド形式
  レジスタはひとつだけ - 累算器, アキュムレータ
    sub n
  Acc = Acc - メモリのn番地
  アキュムレータアーキテクチャ

(2) 2オペランド形式
    sub x, y
  x = x - y

(3) 3オペランド形式
    sub c, a, b
  c = a - b

5.4.2 オペランド指定による分類
命令にどの形式のオペランド指定ができるかでCPUを分類
(a) レジスタ-レジスタ型アーキテクチャ
  3オペランド形式
  ソースオペランドにはレジスタ

(片方には即値も指定可能)
デスティネーションオペランドにはレジスタ

レジスタとメモリとのやりとりは専用の命令で
ロード命令(読み出し)
ストア命令(格納)

レジスタは高速なので高速実行が可能
メモリアクセスを演算とは別に行う必要あり

ロードストアアーキテクチャともいう

レジスタの個数は16から64程度
多目的に使う
汎用レジスタともいう
汎用レジスタアーキテクチャともいう

RISCプロセッサ MIPS, SPARCはこれ

(b) レジスタ-メモリ型アーキテクチャ
2オペランド形式が基本
片方のソースオペランドにはレジスタ
もう片方のソースオペランドにはメモリアドレス
(こちらには即値が指定できることが多い)

CISCがこちら
命令が複雑

(c) メモリ-メモリ型アーキテクチャ
3オペランド形式が基本
各オペランドにアドレスを指定可能
即値もOK

レジスタは少なくてすむ
メモリとのやりとりが増大してボトルネック

5.5 アドレス指定の方法 adressing mode
命令のオペランドの指定方法 - おもに3種類ある
メモリ
番地の指定が必要
実効アドレス effective address という
「実際に使われるアドレス」
レジスタ
レジスタ番号の指定が必要
即値 immediate

2018-06-06

機械命令での表現 (図4.1)
MIPSの機械命令の表現の例は授業メモ08にあり

5.5.1 直接アドレス指定 direct addressing
(1-1) レジスタ直接指定 (図4.2)
レジスタ番号を命令中に含む
ex: MIPS の ADD命令

(1-2) 直接アドレス指定 (図4.3)
絶対直接アドレス指定, 絶対アドレス指定ともいう
メモリの番地を命令中に含む

5.5.2 間接アドレス指定 (indirect addressing, deferred addressing)
オペランドはメモリ
(2-1) メモリ間接アドレス指定 (図4.4)
命令には番地が入っている
メモリの指定番地にオペランドの番地が入っている

(2-2) レジスタ間接アドレス指定 (図4.5)
命令にはレジスタ番号が入っている

そのレジスタの内容がオペランドの番地

5.5.3 相対アドレス指定 (relative addressing)
オペランドはメモリ

命令中には変位とレジスタ番号が入っている
レジスタとその変位の和がオペランドの番地

レジスタの値に変位を加算するのを
「修飾」 (modify) という

(3-1) PC相対アドレス指定 (図4.6)
PC - プログラムカウンタ - 次の命令の実行位置を指定
プログラムコードの中にオペランドを含む場合に利用
プログラムの番地を変更(移動)しても修正不要
relocatable 再配置可能
分岐命令の飛び先の指定にも使う
MIPSの分岐命令もこれ

(3-2) ベースレジスタ修飾 (ベースレジスタ相対アドレス指定) 図4.7
ベースレジスタ - なにかの基準位置を保持
プログラム領域の先頭
配列の先頭
スタックフレームの先頭
これも再配置可能
MIPSのメモリアクセスの唯一の方法 (ストア/ロード命令)

(3-3) インデックスレジスタ修飾 (インデックスレジスタ指定)
命令はインデックスレジスタ番号と実効アドレスを含む
インデックスレジスタには変位が入っている
指定した実効アドレスに, 変位を加算

実効アドレスとしてメモリアドレスを指定 (図4.8)

実効アドレスにレジスタを指定 (図4.9)
ベースレジスタ相対インデックスレジスタ修飾
インデックス修飾レジスタ間接

(4) 即値アドレス指定
命令中に定数を含む
小さい(ビット数の少ない)定数はプログラムに頻出
メモリから別に読み出さなくてすむ

adressing mode 整理
+ 即値 (命令中に定数)
+ レジスタ直接
+ メモリ
++ 直接アドレス指定, 絶対アドレス指定
命令中に番地が書いてある
++ 間接アドレス指定
+++ メモリ間接
命令中に番地、その番地の内容がオペランドの番地
+++ レジスタ間接
命令中にレジスタ番号、そのレジスタにオペランドの番地
++++ PC相対アドレス指定
++++ 修飾
レジスタの値＋変位
+++++ 変位: 定数
+++++ 変位: レジスタ (インデックスレジスタ)
変位の量が可変

「番地」の次元をもっているのがベースレジスタ
「変位」の次元をもっているのがインデックスレジスタ

計算機によって使えるアドレス指定はさまざま
CISC
メモリ-レジスタ型アーキテクチャ
メモリ-メモリ型アーキテクチャ

可変長命令
多くのアドレス指定方式をもつ

RISC
ロードストアアーキテクチャ
メモリにアクセスするのはロード命令とストア命令だけ
使えるアドレス指定は少ない
複雑なアドレシングモードは命令を組み合わせて実現