第1章 コンピュータの抽象化とテクノロジ

大阪大学 大学院 情報科学研究科
今井 正治

E-mail: architecture-2014@vlsilab.ics.es.osaka-u.ac.jp

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

身のまわりでのコンピュータの応用例

- 車載コンピュータ
  - エンジン制御，ブレーキ制御，衝突防止
- 携帯端末
  - 携帯電話，タブレット
- ヒトゲノム研究プロジェクト
  - DNA配列の解析
- World Wide Web
  - 検索エンジン
- ゲーム
  - チェス，将棋，囲碁，オセロ

コンピュータの利用形態の分類

- 汎用コンピュータ
  - ラップトップPC（laptop personal computer）
  - タブレット（tablet）
  - デスクトップ・コンピュータ（desktop computer）
  - サーバー（server）
  - データセンター（data center）
- 組込みコンピュータ（embedded computer）
  - 単一のアプリケーションまたは関連するアプリケーション群を実行

組込みコンピュータの例

- 携帯電話
- ビデオ・ゲーム
- ディジタル・テレビ
- セットトップ・ボックス
- 自動車
- ディジタル・カメラ
- ビデオ・カメラ
- 携帯音楽プレーヤ

本書の内容(1)

- CやJavaのような高水準言語で書かれたプログラムが，ハードウェアの言語にどのように翻訳されるか，また，その結果のプログラムがハードウェアによってどのように実行されるか.
- ハードウェアとソフトウェアのインタフェースとは何を意味し，必要な機能を実行するためにソフトウェアはハードウェアにどのような指示を出すか.

本書の内容(2)

- プログラムの性能は何によって決まり，プログラマはどのようにして性能を改善できるか.
- ハードウェアの設計者は，性能を改善するためにどのような技法を用いているか.
- 逐次処理から並列処理への転換は，どのような理由によるものであり，どのような結果をもたらしたか.（multicore microprocessor）

ハードウェアとソフトウェアが性能に及ぼす影響

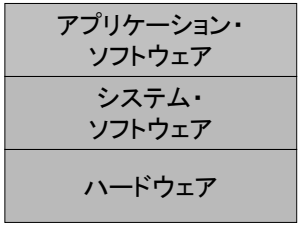
ハードウェアまたはソフトウェアのコンポーネント	そのコンポーネントが性能に及ぼす影響	本書で取り上げている箇所
アルゴリズム	ソース・レベルの文の数と，実行される入出力処理の数の両方を決定	他書にゆずる
プログラミング言語，コンパイラ，アーキテクチャ	ソース・レベルの各文に対応するコンピュータ命令の数を決定	第2章，第3章
プロセッサと記憶システム	命令がいかに高速に実行されるかを決定	第4章，第5章，第7章
入出力システム（ハードウェアおよびオペレーティング・システム）	入出力処理がいかに高速に実行されるかを決定	第6章

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

ソフトウェアの階層

- コンピュータ内のハードウェアは，非常に単純な低水準の命令（機械語命令）を実行できるだけ
- 複雑なアプリケーションを機械語命令に変換する必要がある
- システムソフトウェア
  - オペレーティング・システム
  - コンパイラ
- オペレーティング・システムの役割
  - 基本的な入出力を実行
  - 外部記憶およびメモリの割当
  - コンピュータを同時に使用する複数のアプリケーションの間でコンピュータの資源の共有を図る



コンパイラとアセンブラ

- コンパイラ（compiler）
  - C, C++, Java, Pascal などの高水準言語（high-level language）で記述されたプログラムを，コンピュータのハードウェアが実行可能な命令（instruction）の列に翻訳するプログラム
- アセンブラ（assembler）
  - 記号（symbol）で記述されたプログラムを2進数で表現される機械語（machine language）に変換するプログラム
  - 擬似命令（pseudo instruction）

C言語で記述されたプログラム

```
swap(int v[], int k)
{int temp;
     temp   = v[k];
     v[k]   = v[k+1];
     v[k+1] = temp;
}
```

MIPSのアセンブリ言語に変換されたプログラム

```
swap:
     muli $2, $5, 4
     add  $2, $4, $2
     lw   $15, 0($2)
     lw   $16, 4($2)
     sw   $16, 0($2)
     sw   $15, 4($2)
     jr   $31
```


MIPSの機械語に翻訳されたプログラム

```
00000000101000010000000000011000
00000000000011000000110000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```


講義内容

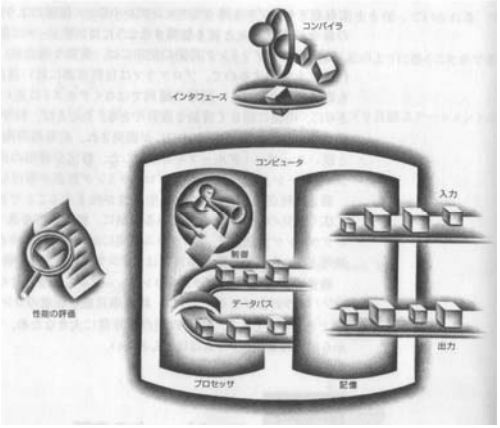
- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

コンピュータの構成要素

- 入力装置(input device)
- 出力装置(output device)
- 記憶装置(memory device)
- データパス(datapath)
- 制御(control)

データパス(datapath)と制御(control)をまとめて「プロセッサ(processor)」

図1.4 コンピュータの古典的な5つの構成要素



入力装置の例

- キーボード
- マウス
- トラックボール
- タブレット
- タッチパッド

出力装置の例

- グラフィック・ディスプレイ（graphic display）
  - CRT（cathode ray tube）（ブラウン管，陰極線管）
  - 液晶ディスプレイ LCD（liquid crystal display）
  - アクティブ・マトリックス・ディスプレイ（active matrix display）
- プリンタ（printer）
- プロッタ（plotter）

画像関連の用語

- 画像 ＝ 画素，ピクセル（pixel）のマトリクス
- ビットマップ（bit map） ＝ ビットで描いた画像
- フレーム・バッファ（frame buffer），ラスター・リフレッシュ・バッファ（raster refresh buffer）

筐体(きょうたい)の内部

- マザーボード(motherboard)
  - 集積回路(integrated circuit)
    - メモリ(memory)
      - DRAM (dynamic random access memory)
      - キャッシュメモリ(cache memory)
      - SRAM (static random access memory)
    - プロセッサ(processor), CPU (central processor unit)
      - データパス(datapath)
      - 制御(control)
  - 記憶装置(storage device)
    - HDD (hard disc drive)

データの格納場所

- 1次記憶(primary memory),
  主記憶(main memory)
  - 揮発性メモリ(volatile memory)
  - DRAM (dynamic random access memory)
- 2次記憶(secondary memory)
  - 不揮発性メモリ(nonvolatile memory)
  - 磁気ディスク(magnetic disc)
  - フラッシュ・メモリ(flash memory)

図1.9 AMD Barcelonaマイクロプロセッサの内部

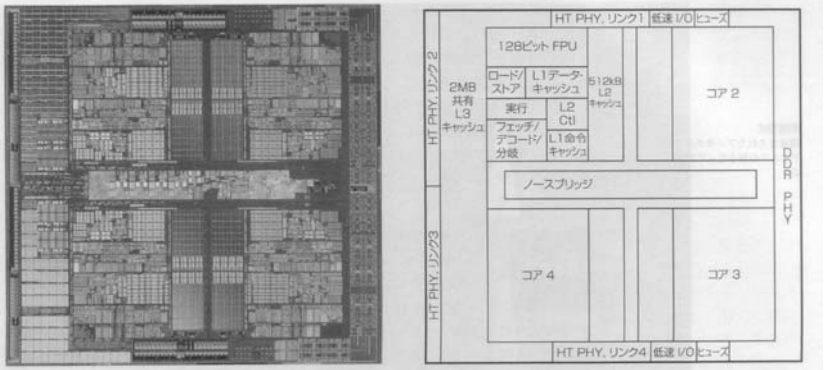


図1.9 AMD Barcelona マイクロプロセッサの内部. 左側は AMD Barcelona プロセッサ・チップの顕微鏡写真である. 右側はこのプロセッサの主要なブロックを図示したものである. このチップには 4 つのプロセッサすなわち「コア」がある. 図1.7 のパソコンのマイクロプロセッサは, Intel Core 2 Duo と言い, チップにコアが2つ載っている.

プロセッサおよびメモリの製造技術

年	コンピュータのテクノロジ	相対コスト性能比
1951	真空管(vacuum tube)	1
1965	トランジスタ(transistor)	35
1975	集積回路(IC: integrated circuit)	900
1995	超大規模集積回路 (VLSI: very large scale integration)	2,400,000
2005	超々大規模集積回路 (ULSI: ultra large scale integration)	6,200,000,000

図1.12 DRAMチップ当たりの容量の推移

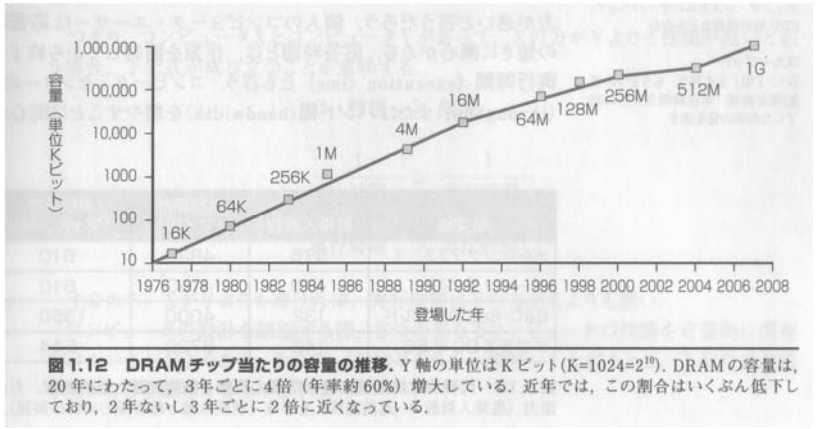


図1.12 DRAMチップ当たりの容量の推移. Y軸の単位はKビット(K=1024=2^{10}). DRAMの容量は，20年にわたって，3年ごとに4倍（年率約60%）増大している．近年では，この割合はいくぶん低下しており，2年ないし3年ごとに2倍に近くなっている．

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

図1.13 代表的な民間航空機の諸性能

航空機	搭乗人員数	航続距離（マイル）	巡航速度（マイル/時）	輸送能力（人・マイル/時）
ボーイング 777	375	4630	610	228,750
ボーイング 747	470	4150	610	286,700
BAC/Sud コンコルド	132	4000	1350	178,200
ダグラス DC-8-50	146	8720	544	79,424

性能の定義

- 目的に応じて複数の定義が存在する
- 個人のコンピュータ・ユーザ（実時間システム）
  - 応答時間（response time）が重要
  - 作業を開始してから終了するまでの時間
- コンピュータ・センターの管理者
  - スループット（throughput）
  - バンド幅（bandwidth）
  - 一定時間内に終了した作業の総量

応答時間を用いた性能の定義(1)

□ 定義

$$\text{性能}_X = \frac{1}{\text{実行時間}_X}$$

□ 性能の性質

$$\text{性能}_X > \text{性能}_Y$$

のとき，次の関係が成立

$$\frac{1}{\text{実行時間}_X} > \frac{1}{\text{実行時間}_Y}$$

$$\text{実行時間}_Y > \text{実行時間}_X$$

応答時間を用いた性能の定義(2)

□ 性能比(XはYよりもn倍速い)

$$\frac{\text{性能}_X}{\text{性能}_Y} = n$$

$$\frac{\text{性能}_X}{\text{性能}_Y} = \frac{\text{実行時間}_Y}{\text{実行時間}_X} = n$$

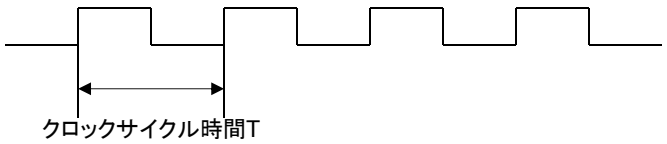
性能の測定

□ 実行時間
- 応答時間(response time)，経過時間(elapsed time)
- タスクの完了に要した合計時間
- ディスク・アクセス，メモリ・アクセス，入出力動作，OSのオーバーヘッドなどを含む

□ CPU実行時間(CPU execution time)，CPU時間(CPU time)
- ユーザCPU時間(user CPU time)
- システムCPU時間(system CPU time)

クロック

□ クロック・サイクル時間(clock cycle time)
- クロック，クロック・サイクル，クロック時間，クロック周期(clock period)，サイクル・タイム
- クロック周波数(クロック・サイクル時間の逆数)



CPU性能とその要因

- 基本的な測定基準
  CPU実行時間
  ＝ クロック・サイクル数 × クロック・サイクル時間
  $= \frac{\text{クロック・サイクル数}}{\text{クロック周波数}}$
- CPUの性能を向上させる方法
  - クロック・サイクル時間を短縮
  - クロック・サイクル数を減らす

命令の性能

- 命令の実行に必要なクロック・サイクル数は命令によって異なる
- プログラムの実行時間
  CPUクロック・サイクル数＝ 実行命令数 ×
  命令あたりの平均クロック・サイクル数
- CPI（cycle per instruction）
- 古典的なCPU性能方程式

$$\text{CPU時間} = \frac{\text{実行命令数} \times \text{CPI}}{\text{クロック周波数}}$$

CPIの計算例（1）

- 命令クラスごとのCPI

	命令クラスごとのCPI		
	A	B	C
CPI	1	2	3

$$\text{CPUクロック・サイクル数} = \sum_{i=1}^{n} (CPI_i \times C_i)$$

C_i: クラスiの命令の実行回数

CPIの計算例（2）

- 命令クラス別の実行命令数

コード系列	命令クラスごとの実行命令数		
	A	B	C
1	2	1	2
2	4	1	1

- CPIの比較

$$CPI_1 = \frac{CPU\text{クロック・サイクル数}_1}{\text{実行命令数}_1} = \frac{10}{5} = 2.0$$

$$CPI_2 = \frac{CPU\text{クロック・サイクル数}_2}{\text{実行命令数}_2} = \frac{9}{6} = 1.5$$

例題: CPU性能の比較

- 同じ命令セットを持つ2種類のコンピュータで同じプログラムを実行する場合のCPIは次のとおり.

コンピュータ	クロック・サイクル時間	CPI
A	250 ps	2.0
B	500 ps	1.2

- このプログラムに関して，どちらのコンピュータがどのくらい速いか?

例題: CPU性能の比較(解答)

- コンピュータAのCPU時間

$$I \times 2.0 \times 250\ ps = I \times 500\ ps$$

- コンピュータBのCPU時間

$$I \times 1.2 \times 500\ ps = I \times 600\ ps$$

- CPU性能の比

$$\frac{\text{CPU性能}_A}{\text{CPU性能}_B} = \frac{\text{実行時間}_B}{\text{実行時間}_A} = \frac{600\ \times\ I\ ps}{500\ \times\ I\ ps} = 1.2$$

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

図1.15 25年にわたる8世代のIntel x86マイクロプロセッサのクロック周波数と消費電力

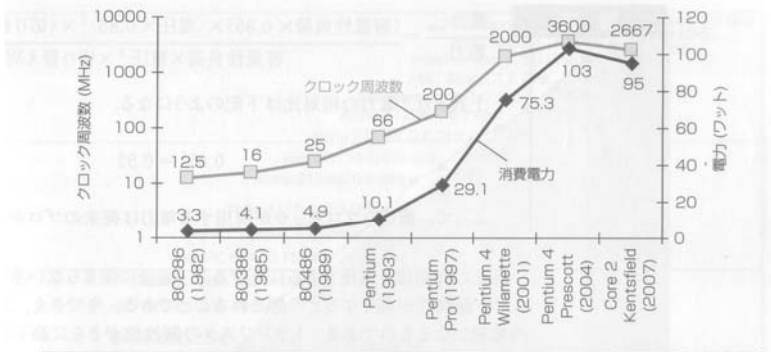


図 1.15 25 年にわたる 8 世代の Intel x86 マイクロプロセッサのクロック周波数と消費電力. Pentium 4 では，クロック周波数と消費電力が飛躍的に上昇したが，性能はそれほどでもなかった. Prescott では，熱の問題にぶつかり，Pentium 4 系列を放棄することにつながった. Core 2 系列では，クロック周波数が低い単純なパイプラインに戻り，1 チップに複数のプロセッサを載せるようになった.

消費電力

- CMOS回路の消費電力

$$P = F \times C \times V^2$$

P: 消費電力
F: 切替え周波数
C: 容量性負荷
V: 電源電圧

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換: 単体プロセッサからマルチプロセッサへ
- 実例: AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

プロセッサの性能向上(1)

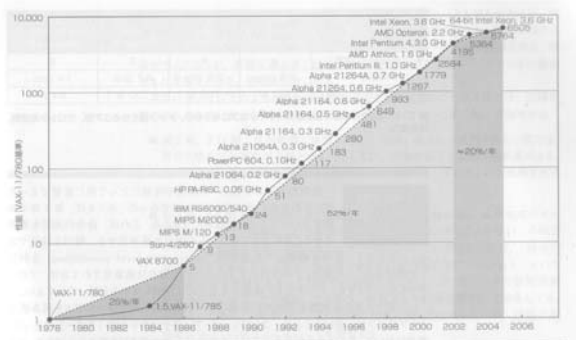


図1.16 1980年代中盤以降のプロセッサの性能の向上. この図は,VAX 11/780を基準にして,SPECintベンチマーク(1.8節を参照)によって測定した相対性能を,グラフに示したものである.1980年代の中盤以前は,プロセッサの性能の向上は主としてテクノロジの進歩に牽引されたものであり,年率で平均約35%であった.それ以降の向上率は約52%に達する.それに寄与したのは,より高度なアーキテクチャおよび構成面での工夫である.2002年までには,このふたつの成長による性能向上の差は約7倍に至った.浮動小数点演算の性能向上はさらに早かった.2002年以降は,電力の限界,利用可能な命令レベル並列性の限打ち,メモリ・レイテンシの長さにより,単体プロセッサの性能の向上は鈍化し,年率約20%になった.

プロセッサの性能向上(2)

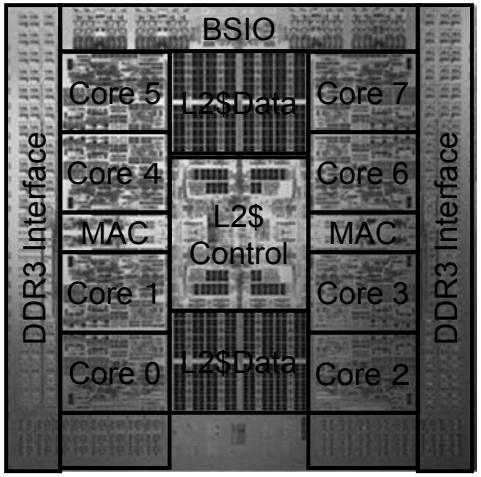
- マイクロ・プロセッサの性能の向上率
  - 1986年～2002年までは年率52%
  - 2002年以降は年率20%に低下
- 応答時間の改善からスループットの改善への方向転換
  - マルチコア・マイクロプロセッサ
    - デュアルコア(dual core) = 2 cores
    - クアッドコア(quad core) = 4 cores
  - 1世代(約2年)ごとにコア数が倍増

図1.17 2008年のマルチコア・プロセッサ

製品	AMD Opteron X4 (Barcelona)	Intel Nehalem	IBM Power 6	Sun Ultra SPARC T2 (Niagara 2)
チップ当たりのコア数	4	4	2	8
クロック周波数	2.5 GHz	~ 2.5 GHz	4.7 GHz	1.4 GHz
消費電力	120 W	~ 100 W?	~ 100 W?	94 W

京コンピュータ

京コンピュータで使われている SPARC64 VIIIfx プロセッサ



並列プログラミングの難しさ

- 並列プログラミングは，性能を向上するためのプログラミング
  - プログラムの正しさの保障
  - 重要な問題（アプリケーション）への解決策の提供
  - 人/他のプログラムに対する有用なインタフェースの提供
  - 高速に実行可能
- アプリケーション分割の最適化
  - 負荷の平準化
  - スケジューリング，通信のオーバーヘッドの隠蔽

講義内容

- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換：単体プロセッサからマルチプロセッサへ
- 実例：AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

図1.18 チップの製造工程

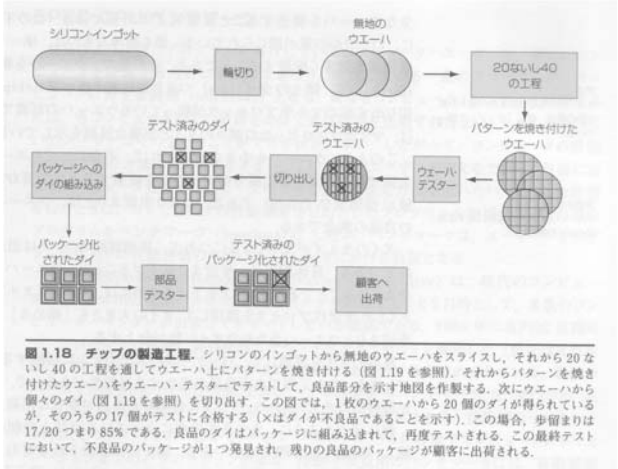


図 1.18 チップの製造工程. シリコンのインゴットから無地のウエーハをスライスし，それから 20 ないし 40 の工程を通してウエーハ上にパターンを焼き付ける（図 1.19 を参照）．それからパターンを焼き付けたウエーハをウエーハ・テスターでテストして，良品部分を示す地図を作製する．次にウエーハから個々のダイ（図 1.19 を参照）を切り出す．この図では，1 枚のウエーハから 20 個のダイが得られているが，そのうちの 17 個がテストに合格する（×はダイが不良品であることを示す）．この場合，歩留まりは 17/20 つまり 85% である．良品のダイはパッケージに組み込まれて，再度テストされる．この最終テストにおいて，不良品のパッケージが 1 つ発見され，残りの良品のパッケージが顧客に出荷される．

図1.19 AMD Optaron X2チップの12インチ（300mm）ウェーハ

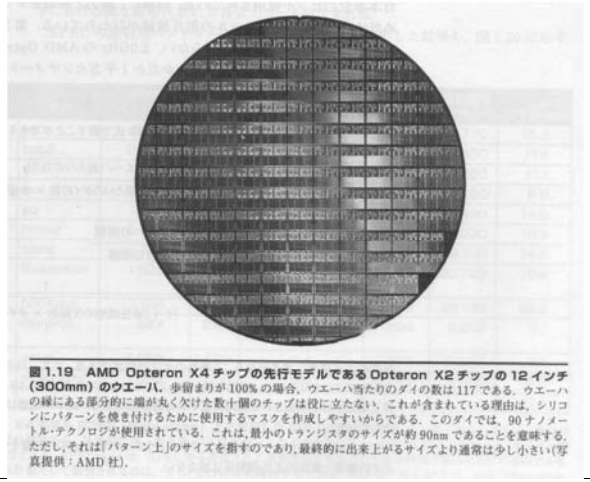


図 1.19 AMD Opteron X4 チップの先行モデルである Opteron X2 チップの 12 インチ（300mm）のウエーハ. 歩留まりが 100% の場合，ウエーハ当たりのダイの数は 117 である．ウエーハの縁にある部分的に端が丸く欠けた数十個のチップは役に立たない．これが含まれている理由は，シリコンにパターンを焼き付けるために使用するマスクを作成しやすいからである．このダイでは，90 ナノメートル・テクノロジが使用されている．これは，最小のトランジスタのサイズが約 90nm であることを意味する．ただし，それは「パターン上」のサイズを指すのであり，最終的に出来上がるサイズより通常は少し小さい（写真提供：AMD 社）．

集積回路のコスト

- ダイ当たりのコスト＝

$$\frac{\text{ウェーハ当たりのコスト}}{\text{ウェーハ当たりのダイの個数} \times \text{歩留まり}}$$

- ウェーハ当たりのダイの個数 ＝

$$\frac{\text{ウェーハの面積}}{\text{ダイの面積}}$$

- 歩留まり（yield） ＝

$$\frac{1}{(1 + (\text{単位面積当たりの欠陥数} \times \text{ダイの面積}/2))^2}$$

SPEC CPUベンチマーク(1)

- ユーザが加えるであろう負荷(workload)に対するコンピュータ性能を正しく測定したい
- ベンチマーク(benchmark):
  ユーザプログラムの代わりに性能評価のためにコンピュータにかける負荷
- SPEC(System Performance Evaluation Cooperative)
  現代のコンピュータ用の標準ベンチマーク・スイートを開発することを目的とする組織

SPEC CPUベンチマーク(2)

- SPEC 89: 1989年に作成された最初のベンチマーク
- SPEC CPU 2006
  - CINT 2006: 整数用ベンチマーク(12本)
    - C言語コンパイラ, チェス・プログラム, 量子コンピュータのシミュレータなど
  - CFP 2006: 浮動小数点用ベンチマーク(17本)
    - 有限要素モデリングのための構造格子のコード, 分子動力学における粒子法のコード, 流体力学における疎な線形代数の方程式を解くコードなど

図1.20 AMD Opteron X4モデル2356 (Barcelona)上で実行したSPECINT 2006

説明	名前	命令数 ($\times 10^9$)	CPI	クロック・サイクル時間(ns)	実行時間(秒)	基準時間(秒)	SPEC ratio
有意の文字列処理	perl	2,118	0.75	0.4	637	9,770	15.3
ブロック・ソート圧縮	bzip2	2,389	0.85	0.4	817	9,650	11.8
GNU Cコンパイラ	gcc	1,050	1.72	0.4	724	8,050	11.1
組合せ最適化	mcf	336	10.00	0.4	1,345	9,120	6.8
囲碁(AI)	go	1,658	1.09	0.4	721	10,490	14.6
遺伝子系列の検索	hmmer	2,783	0.80	0.4	890	9,330	10.5
チェス(AI)	sjeng	2,176	0.96	0.4	837	12,100	14.5
量子コンピュータ・シミュレータ	libquantum	1,623	1.61	0.4	1,047	20,720	19.8
ビデオ圧縮	h264avc	3,102	0.80	0.4	993	22,130	22.3
離散事象シミュレーションのライブラリ	omnetpp	587	2.94	0.4	690	6,250	9.1
ゲーム/経路発見	astar	1,082	1.79	0.4	773	7,020	9.1
XML	xalancbmk	1,058	2.70	0.4	1,143	6,900	6.0
幾何平均							11.7

幾何平均

- 定義式

$$\sqrt[n]{\prod_{i=1}^{n} \text{実行時間比}_i}$$

$$\prod_{i=1}^{n} x_i = x_1 \times x_2 \times \cdots \times x_n$$

SPECpower

- 消費電力評価用ベンチマーク
- Javaビジネス・アプリケーション用ベンチマーク
  - Java仮想マシン，コンパイラ，ガーベージ・コレクタ，オペレーティング・システムの種々の機能
    プロセッサ，キャッシュ，主記憶の動作
  - 性能の評価方法：スループット
    毎秒当たりのビジネス処理数
  - ワット当たりの総合ssj_ops

$$(\sum_{i=0}^{10} ssj_{ops_i})/(\sum_{i=0}^{10} \text{電力}_i)$$

図1.21 AMD Opteron X4モデル2356（Barcelona）を用いて実行したSPECpower_ssj2006

負荷レベル	性能（ssj_ops）	平均電力（ワット）
100%	231,867	295
90%	211,282	286
80%	185,803	275
70%	163,427	265
60%	140,160	256
50%	118,324	246
40%	92,035	233
30%	70,500	222
20%	47,126	206
10%	23,066	180
0%	0	141
合計	1,283,590	2,605
Σ_{ssj_ops}/Σ電力		493

講義内容

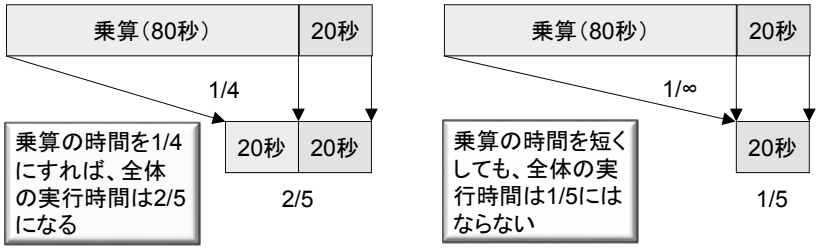
- はじめに
- プログラムの裏側
- コンピュータの内部
- 性能
- 電力の壁
- 方向転換：単体プロセッサからマルチプロセッサへ
- 実例：AMD Opteron X4の製造技術とベンチマーク・テスト
- 誤信と落とし穴

誤信と落とし穴

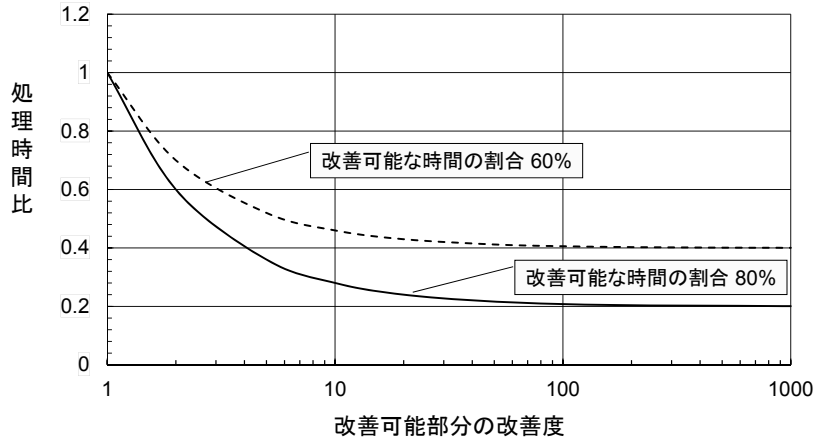
- 落とし穴：
  コンピュータのある面を改善することによって，その改善度に等しい性能向上を期待すること.
- 誤信：
  消費電力はコンピュータの利用率に比例する.
- 落とし穴：
  性能の尺度に性能方程式の一部分だけを使用すること.

Amdahlの法則

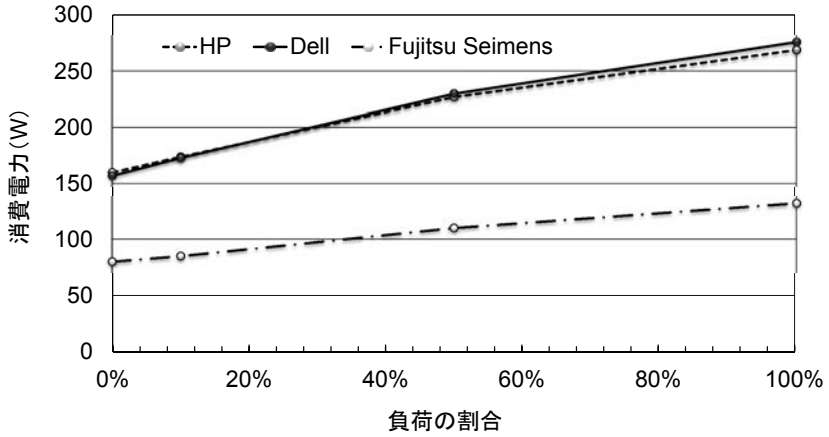
□ 改善後の実行時間 =
$$\frac{\text{改善の影響を受ける実行時間}}{\text{改善度}} + \text{改善の影響を受けない実行時間}$$



改善可能な時間の割合と全体の処理時間の改善



3種類のサーバーの消費電力の比較



性能の評価尺度 MIPS

□ MIPS（million instructions per second）

$$MIPS = \frac{\text{実行命令数}}{\text{実行時間} \times 10^6}$$

$$MIPS = \frac{\text{実行命令数}}{\frac{\text{実行命令数} \times CPI}{\text{クロック周波数}}} \times 10^6 = \frac{\text{クロック周波数}}{CPI \times 10^6}$$