2012年 後学期

計算機アーキテクチャ 第二 (O)

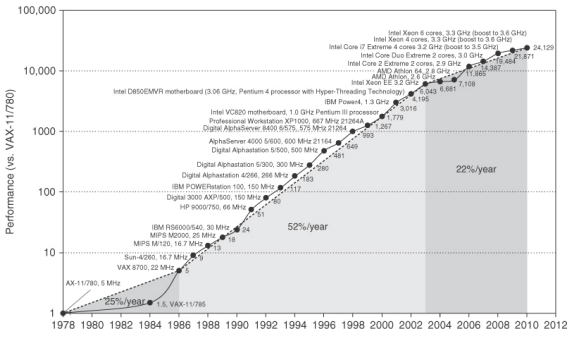
コンピュータの性能

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, エネルギー
- 熱
- 音
- 価格
- 安定性, など

Growth in processor performance



From CAQA 5th edition

Which is faster?


Plane	DC to Paris	Speed	Passengers	Throughput (p x mph)
Boeing 747	6.5 hours	610 mph (1130km/h)	470	286,700 (470 x 610)
BAD/Sud Concorde	3 hours	1350 mph (2500km/h)	132	178,200 (132 x 1350)

- **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

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

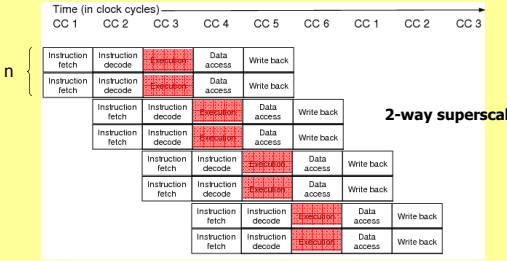
or

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

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

スーパースカラプロセッサと命令レベル並列性

- 複数のパイプラインを利用して IPC (instructions per cycle) を 1以上に引き上げる，複数の命令を並列に実行
  - n-way スーパースカラ
- ハザードの積極的な解消，ストールの隠蔽が重要

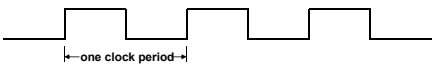


Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005

Remind: Machine Clock Rate

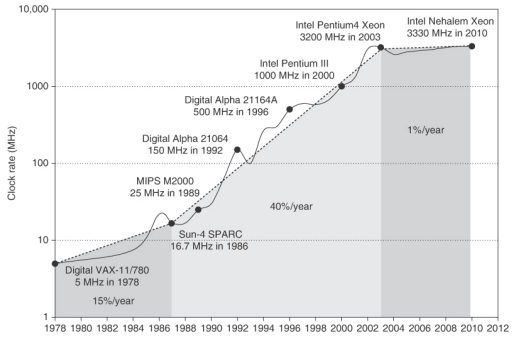
- Clock rate (MHz, GHz) is inverse of clock cycle time (clock period)

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

Growth in clock rate of microprocessors



From CAQA 5th edition

MIPS (Million Instructions Per Second)

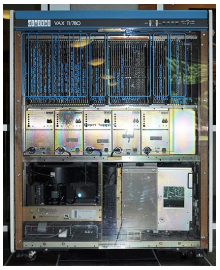
- 1秒当たりに実行された命令の数（単位はMillion）
  - 原始MIPS (native MIPS)
- 注意
  - プロセッサアーキテクチャのMIPSとは関係ない
- MIPSの問題点とは？
  - 命令セットに強く依存する尺度
  - 異なる命令セット，NOP，コンパイラ，性能？

VAX 11/780 (1978) 1 native MIPS

VAX命令 (Complex Instruction Set Computer)

```
MOVL  @40(R4), 30(R2)  ; M[M[40+R4]] <- M[30 + R2]

MOVC3 @36(R9), (R10), 35(R11)
R1 <- 35 + R11,  R3 <- M[36 + R9]
for (R0 <- M[R10]; R0!=0; R0--)
   { M[R3] <- M[R1]; R1++; R3++;  }
R2 = 0; R4 = 0; R5 = 0;
```



1 native MIPS

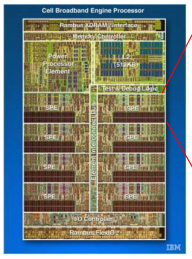
MFLOPS, GFLOPS

- 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



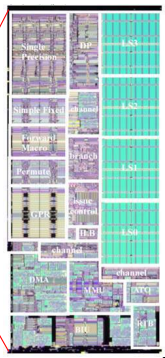


Diagram created by IBM to promote the CBEP, ©2005
WIKIPEDIAより

Cell/B.E. Element Interconnect Bus

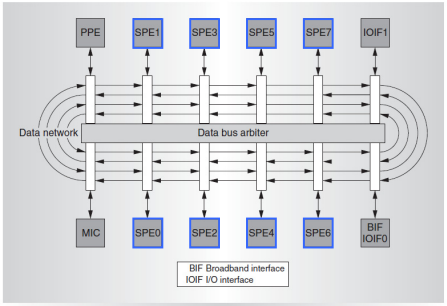


Figure 2. Element interconnect bus (EIB).

IEEE Micro, Cell Multiprocessor Communication Network: Built for Speed

Cell Broadband Engine (GFLOPSの例)

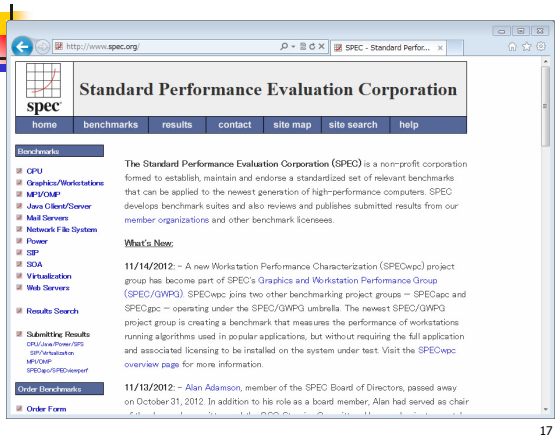
- ピーク性能
  - 1サイクルで積和演算を1回実行できる演算器
    (**2** FLOP/cycle)
  - 1命令で4つの演算を並列処理(SIMD, Single Instruction Multiple Data構成)で，SPEあたりの並列性 **4**
  - チップ内のSPEの数 **8**
  - 動作周波数 **4**GHz

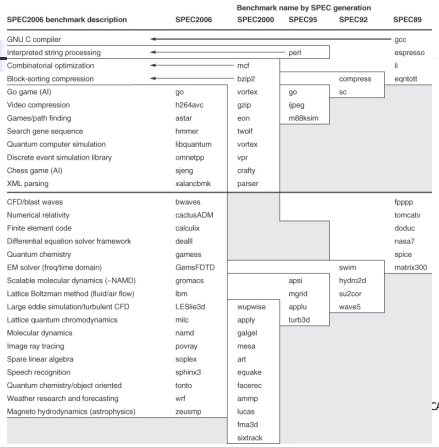
  - **2** × **4** × **8** × **4** = 256 GFLOPS
  - 積和演算 × SIMD化 × マルチコア × 動作周波数
    - **ペンティアムは 8GFLOPS 程度** (256/8 = 32)

  - 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 . . .
    - 実効性能

相対性能，ベンチマーク

- 合成ベンチマーク (Synthetic Benchmark)
  - Whetstone
  - **Dhrystone**
  - Livermore loops
  - SPEC CPUベンチマーク
    - **SPEC89, SPEC92, SPEC95, SPEC2000, SPEC2006**
  - Intelの新しいベンチマーク
    - Recognition, Mining, and Synthesis
  - 行列積，LU分解
  - 画像処理 (FPS, frame per second)
- 合成ベンチマークの問題点は？
  - 実アプリ，チート，誰がどうやって？





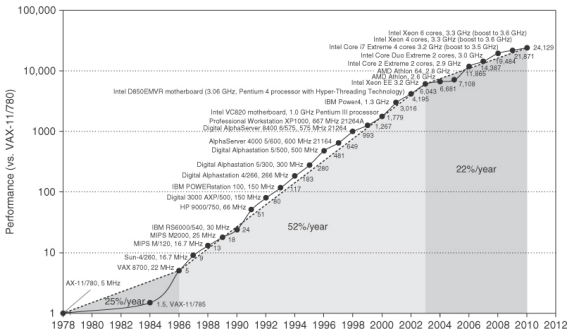
CAQA 5th edition

SPEC CINT2000 Summary example

164.gzip	1400	565	248*
175.vpr	1400	655	214*
176.gcc	1100	376	292*
181.mcf	1800	1193	151*
186.crafty	1000	236	424*
197.parser	1800	1023	176*
252.eon	1300	357	365*
253.perlbmk	1800	685	263*
254.gap	1100	525	210*
255.vortex	1900	758	251*
256.bzip2	1500	534	281*
300.twolf	3000	1285	234*
SPECint_base2000			249
SPECint2000			

19

Growth in processor performance



From CAQA 5th edition

20

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.
- Amdahl's Law states that the performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.

21

落とし穴

- 性能の尺度に性能方程式の一部を使用すること.

CPU execution time for a program = # CPU clock cycles for a program x clock cycle time

22

補足:コンピュータの性能

- 性能を議論するためには尺度が必要
  - 尺度の達成が目的化することは危険.
  - テクノロジが進化するとき，尺度も大きく変化する.
  - 時に，適切な尺度を定義することが重要.
  - これにより，飛躍的に進歩することがある.

23

Session 1 – Processor Pipelines

Increasing Processor Performance by Implementing Deeper Pipelines

Eric Sprangle , Doug Carmean
Pentium Processor Architecture Group,
Intel Corporation

ISCA-2002 pp.25-34

24

プロセッサの命令パイプラインの例

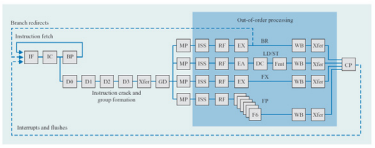
Basic Pentium® III Processor Misprediction Pipeline

1	2	3	4	5	6	7	8	9	10
Fetch	Fetch	Decode	Decode	Decode	Rename	ROB Rd	Rdy/Sch	Dispatch	Exec

Basic Pentium® 4 Processor Misprediction Pipeline

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
TC Nxt IP	TC Fetch	Drive	Alloc	Rename	Que	Sch	Sch	Sch	Disp	Disp	RF	RF	Ex	Flgs	Br Ck	Drive			

The Microarchitecture of the Pentium® 4, Intel Technical Report



POWER4 System Microarchitecture, IBM Journal

背景と目的

- プロセッサの動作周波数の決定はプロセッサ設計者の直面する本質的な課題となっている。
- パイプラインが深くなると、設計の複雑さと工程は劇的に増加する。
- パイプラインの深さとキャッシュサイズの関数として、プロセッサ性能を予測するモデルを構築し、シミュレーションにより性能を評価する。
- Pentium 4プロセッサをベースラインとして、深いパイプラインが性能向上につながることを示す。

Simulated 2GHz Pentium 4 like processor config.

Core
- 3-wide fetch/retire
- 2 ALUs (running at 2x frequency)
- 1 load and store / cycle
- In-order allocation/de-allocation of buffers
- 512 rob entries, load buffers and store buffers

Memory System
- 64 kB/8-way I-cache
- 8 kB/4-way L1 D-cache, 2 cycle latency
- 256 kB/8-way unified L2 cache, 12 cycle latency
- 3.2 GB/sec memory system, 165ns average latency
- Perfect memory disambiguation
- 16 kB Gshare branch predictor
- Streaming based hardware prefetcher

Skeleton という実行駆動のシミュレータを用いて評価する。

Simulated Benchmark Suites

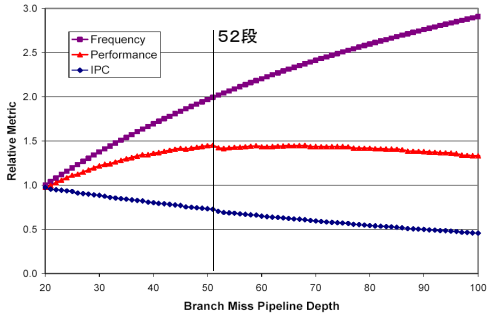
Suite	Number of Benchmarks	Description
SPECint95	8	spec.org
Multimedia	22	speech recognition, mpeg, photoshop, ray tracing, rsa
Productivity	13	sysmark2k internet/business/ productivity, Premiere
SPECfp2k	10	spec.org
SPECint2k	12	spec.org
Workstation	14	CAD, rendering
Internet	12	webmark2k, specjbb

パイプラインのオーバヘッド

- Conservative ASIC design
  - Clock skew + jitter = 51ps
  - Standard 0.18um process, flop overhead is 3 FO4 = 75ps
  - Pipeline overhead = 51ps + 75ps = 125ps
- Custom design
  - Most of clock skew and jitter overhead can be hidden.
  - Pipeline overhead = 75ps
- Extreme custom design
  - Sub-50ps at the cost of a much larger design cost
- Pentium 4 overhead
  - Pipeline overhead = 90ps
  - Use 90ps as a baseline overhead time

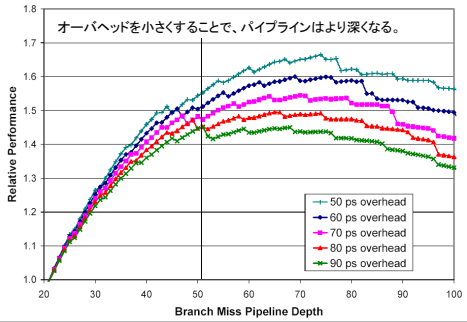
パイプライン段数を変化させた時の動作周波数、IPC、性能の評価結果

- パイプラインが52段で、動作周波数が2倍になるまで性能が向上



パイプライン段数とオーバヘッドを変化させた時の性能の評価結果

- 評価には、パイプライン段数に影響を受けず一定のオーバヘッドを想定
- 2GHz のパイプラインピッチ 500ps、Pentium 4のオーバヘッドは 90ps

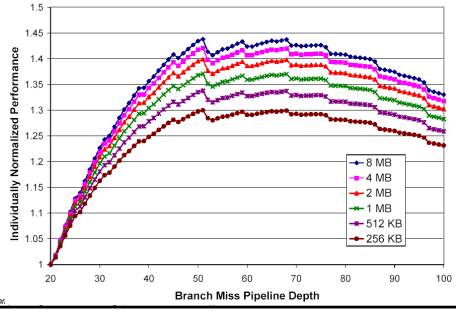


Adapted fr

31

パイプライン段数とキャッシュサイズを変化させた時の性能の評価結果

- キャッシュサイズを変化させることで相対性能は変化するが、最適なパイプライン段数はほとんど変化しない。



Adapted from *Cor*

32

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

33

Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005