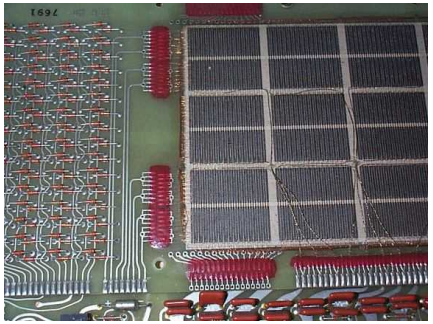
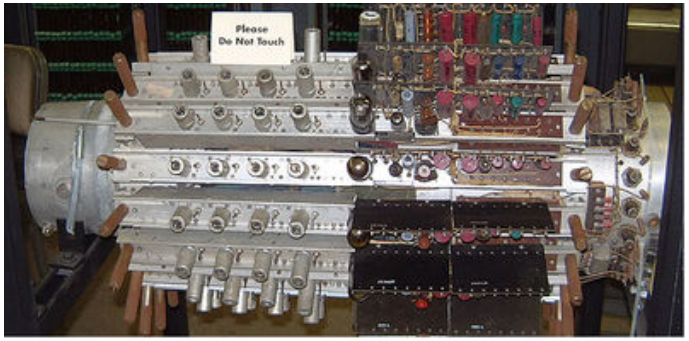
「メモリとI/O」
2012/7/6, 7/13, …

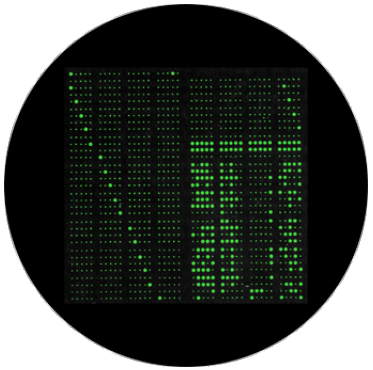
記憶装置(メモリ)に関して

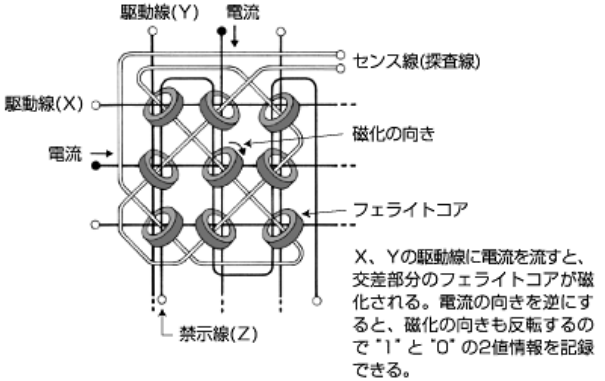
- 前回までプロセッサの内部構造を勉強してきた
- 今度は外側に目を向ける、特にメモリおよびI/O(Input, Output)に関して

- 今まではメモリは単純に巨大なレジスタと同様だとみなしてきた
  - マルチプレクサ・デコーダとD-フリップフロップの組み合わせ
- 実際のメモリは実装が異なる; 特に大容量の確保の必要性
  - レジスタと同様の実装だと高速だが、トランジスタと配線が大量に必要で容量を稼げない
- 昔より種々のデバイスによる実装
  - 太古: 水銀遅延線、撮像管、磁気コア。。。







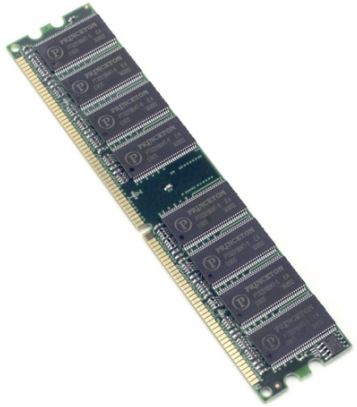


記憶装置の特性(1)

- 実装デバイス: 用途に応じて種々の記憶デバイス
  - 半導体メモリ (Silicon Memory)
  - フラッシュメモリ (Flash Memory)
  - ハードディスク (HDD)
  - 光ディスク (CD, DVD, BD, MO, …)
  - 磁気テープ (Magnetic Tape)

- メモリ(記憶)階層(詳しくは後述):
  - キャッシュメモリ: (後述)
  - 一次記憶: CPUに直結され、プログラム実行中に直接アドレス指定されてアクセスされる記憶装置。通常は小容量の揮発性の半導体メモリ
  - 二次記憶: CPUにはI/O経由で接続され、データを通常はファイルの形で保存し、プログラムの実行や電源on/offをまたいで記憶する装置。通常は大容量のハードディスクやフラッシュメモリ
  - 三次記憶：二次記憶よりさらに長期保存や大容量を目的とした記憶装置。磁気テープ、光ディスク 等(近年のクラウドではハードディスクも)

一次記憶



DRAM

EP-ROM
(紫外線消去可能)

二次記憶

ハードディスク

フラッシュ(SSD)
Solid-State Disk

三次記憶

テープロボット

光ディスク

記憶装置の特性(2)

- 揮発性 / 不揮発性
  - 不揮発性メモリ: 電力を供給しなくとも格納した情報を保持可能。長期間の記憶に適している。例:フラッシュメモリ、ハードディスク
  - 揮発性メモリ: 情報を保持するのに電力供給が必要。一般的に高速。例:半導体メモリ

- ダイナミック / スタティック
  - ダイナミックメモリ: 揮発性メモリのうち、定期的な読み込み(メモリリフレッシュ)をしないと格納情報が消えてしまうもの。Dynamic RAM (DRAM)
  - スタティックメモリ: 揮発性メモリのうち、リフレッシュを必要としないもの。Static RAM (SRAM)

- 書き換え可能 / 読み出し専用
  - 読み書き可能ストレージ(リードライトメモリ, Read-Write Memory):情報をいつでも上書き可能なメモリ。例: RAM, フラッシュ、DVD-RAM, HDD
  - 読み取り専用ストレージ(リードオンリーメモリ、Read-Only Memory, ROM): 媒体製造時に情報を記憶させるものと、製造後に一度だけ書き込むことができるライトワンス型がある。例: ROM, CD-ROM, CD-R

記憶装置の特性(3)

- アクセス方法
  - ランダムアクセス: 任意の位置へのアクセスはほぼ一定で、一次記憶装置や二次記憶装置に適している。例: 半導体メモリ、
  - シーケンシャルアクセス:逐次的に情報にアクセスし、そのアクセス時間は前回アクセスした位置に依存。オフラインストレージ用。例:磁気テープ

- データの構成とインデックス指定
  - 平坦なメモリ空間に対する数値アドレス指定:メモリは巨大な一次元配列状に構成されており、そのアクセスは数値的かつ連続的なメモリアドレスによる。例:メインメモリ
  - ブロック・ファイル名指定:メモリはハードウェアの構成に合わせブロック単位で多次元の構造で配置されており、各次元への数値や名前(ファイル名等)インデクスを指定する。例: ハードディスクおよびファイルシステム
  - コンテンツ指定:ハッシュ値や短い識別子でアクセスする情報を選択。コンテンツ指定可能ストレージはソフトウェアでもハードウェアでも実装できる。ハードウェアの方が高速だが、高価。

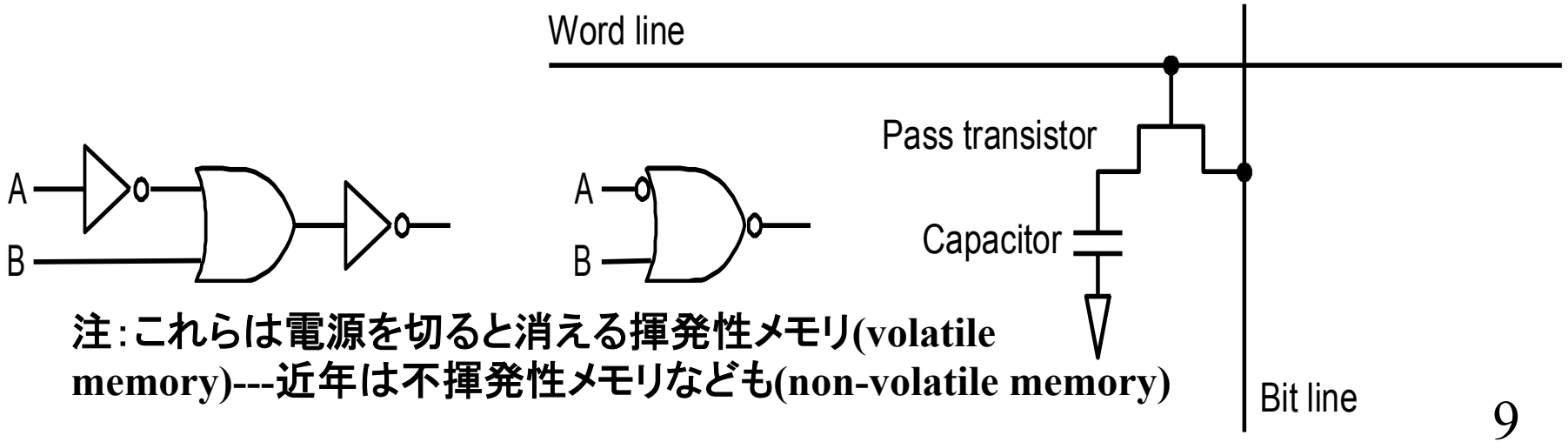
記憶装置の特性(4)

- 容量 (Capacity)
  - 記憶容量:デバイスまたは媒体が記憶できる全容量.
    - 1bit, 1Byte = 8 bits, 1Word = 2/4/8 Bytes (アーキテクチャ依存)
    - 半導体メモリは基数1024 = 2^10を一般的に用いる: K(ilo) = 1024, M(ega) = K^2=2^20, G(iga) = K^3=2^30, T(era) =K^4, P(eta)=K^5,
    - 例: TSUBAME2の半導体メモリ総容量は約100TByte=100xK^4Byte
    - ハードディスクは基数1000を用いる。 K(ilo) = 1000, …
  - 記憶密度:情報をどれだけコンパクトに記憶できるかの尺度。単位長、単位面積、単位堆積あたりの記憶容量。例: bits/inch^2
- 性能 (Performance)
  - レーイテンシ:アドレスを与えてからストレージの特定位置にアクセスするのにかかる時間。一次記憶装置の場合はナノ秒単位、二次記憶装置ではマイクロ~ミリ秒単位、三次記憶装置では秒単位程度。読み取りと書き込みで異なる場合もある。シーケンシャルアクセス式の場合、ばらつきが大きいので、最小・最大・平均で表す。
  - スループット: 単位時間当たりの読み書き速度(Mbyte/秒など)。読み取りと書き込みで異なる場合もある。また、ハードディスク等ではインデクスが逐次的なアクセスの方がランダムなアクセスよりスループットが高くなる。

半導体メモリ: 大まかな分類

- **Static RAM (SRAM):**
  - 状態値はフリップフロップに類似した回路に保存
  - 高速(数ns以下)、面積を食う (4～6 トランジスタ必要)、電力消費大

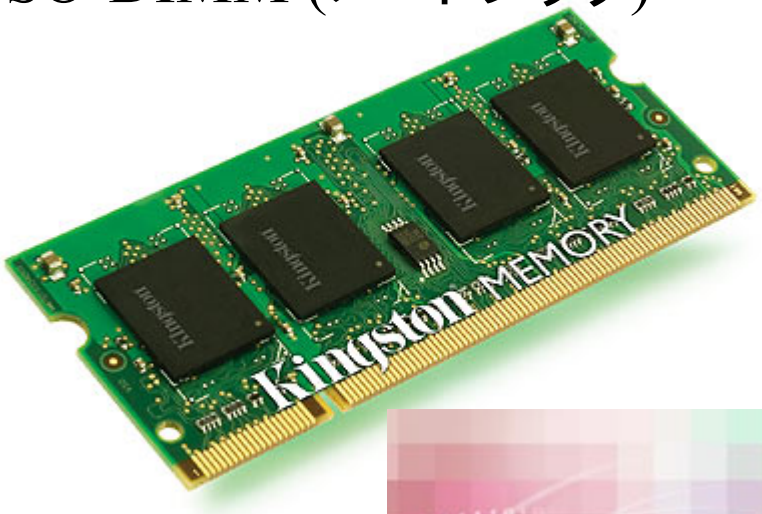
- **Dynamic RAM (DRAM):**
  - 状態値は、コンデンサの充電電荷で表現 (しかも、放電するので、定期的なリフレッシュ操作が必要→約2msごと)
  - 個々のメモリセルは大変小さいので、高容量が可能だが、SRAMと比較するとアクセス時間が遅い (5-10倍程度, 数n~数十ns)



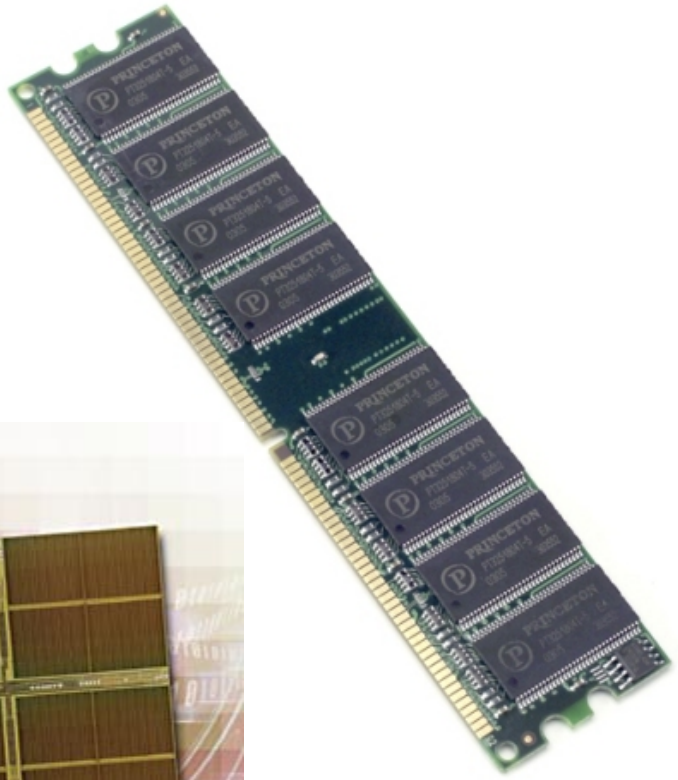
注:これらは電源を切ると消える揮発性メモリ(volatile memory)---近年は不揮発性メモリなども(non-volatile memory)

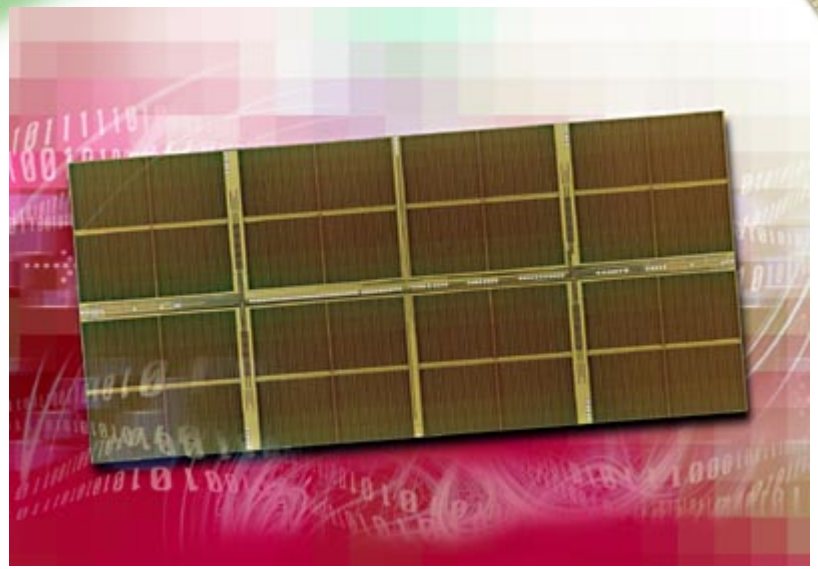
DRAM Modules (DIMMs)

SO-DIMM (ノートブック)

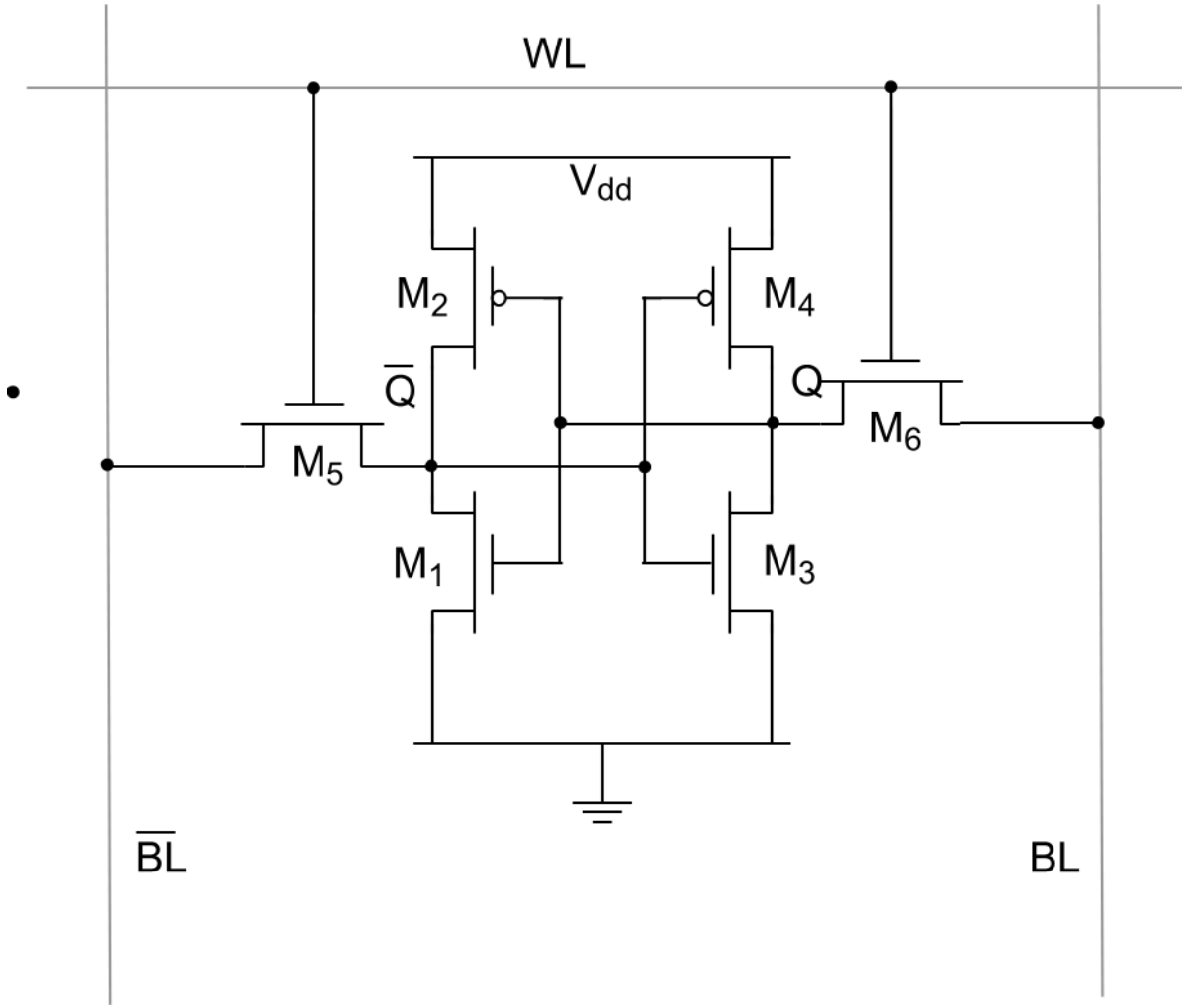


DIMM (デスクトップ)





SRAM Cell (6トランジスタ)



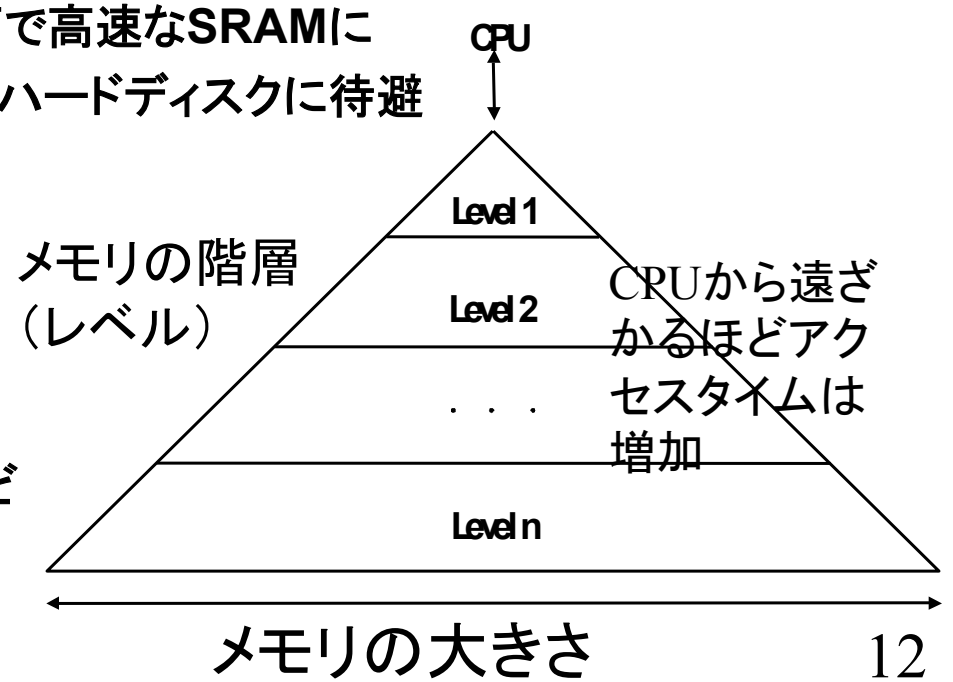
高速化: メモリ階層(memory hierarchy)の活用

- CPUの高速化←クロック速度の上昇→メモリアクセス速度向上の必要性
  「できれば全部のメモリが高速であって欲しい」

2012年

SRAM アクセスタイムは数nsであり、メガバイトあたり 数円以上.
DRAMアクセスタイムは数十nsであり、メガバイトあたり 一円未満.
ハードディスクのアクセスタイムは 数ms であり、メガバイトあたり0.01円以下.

- 観測: メモリのアクセスには局所性がある
  - 頻繁にアクセスされるデータは、高価で高速なSRAMに
  - そうでもないのは、DRAM、あるいはハードディスクに待避
  - そのような特質を持つメモリ階層 (Memory Hierarchy) を構築
    - 一次記憶:半導体メモリ
    - 二次記憶:ハードディスク, SSD
    - 三次記憶:光ディスク、テープなど

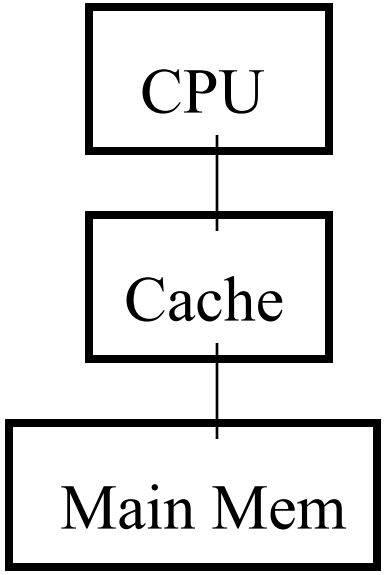


局所性(locality)について

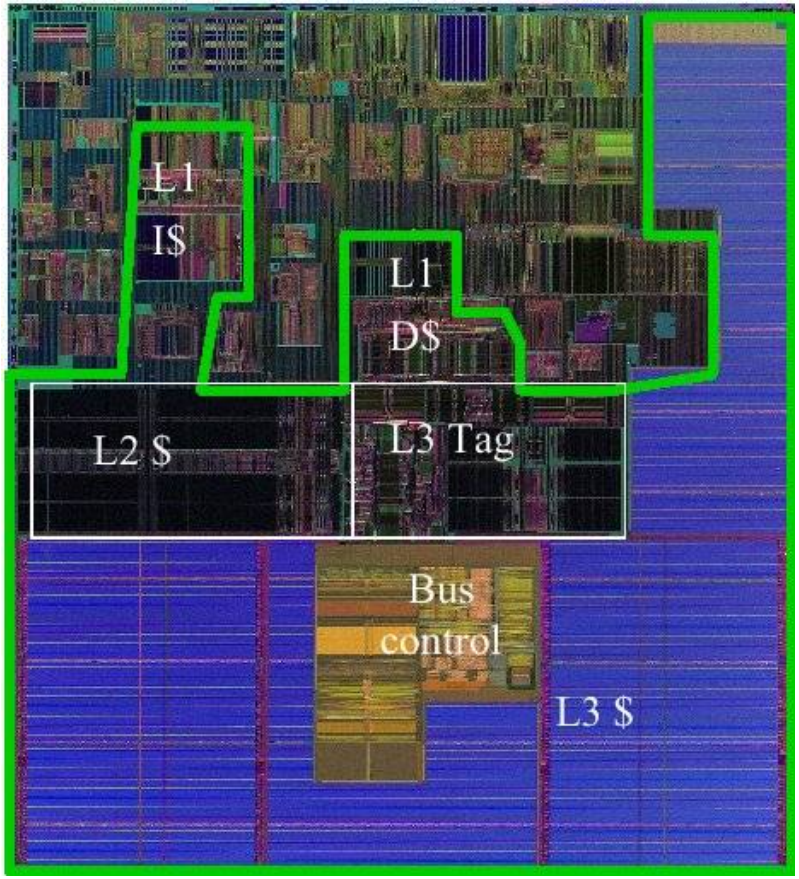
- メモリ階層を有効にする前提
- もしあるメモリが参照されたとすると、
  時間的局所性 (temporal locality): 近い将来同じメモリが参照される可能性が高い
  空間的局所性 (spatial locality): 近いアドレスのメモリが参照される可能性が高い.
  Q: なぜ、プログラムは通常このような局所性を示すのか?
- まず、二階層のメモリ階層を考える (上位、下位)
  - 上位階層に高速な小容量のメモリ、下位階層に低速な大容量のメモリ
  - ブロック: 連続したメモリの固まり、上位と下位の転送の単位
  - ヒット (hit): 上位階層にデータがある場合
  - ミス (miss): 上位階層にデータがない場合

キャッシュメモリ (Cache Memory)

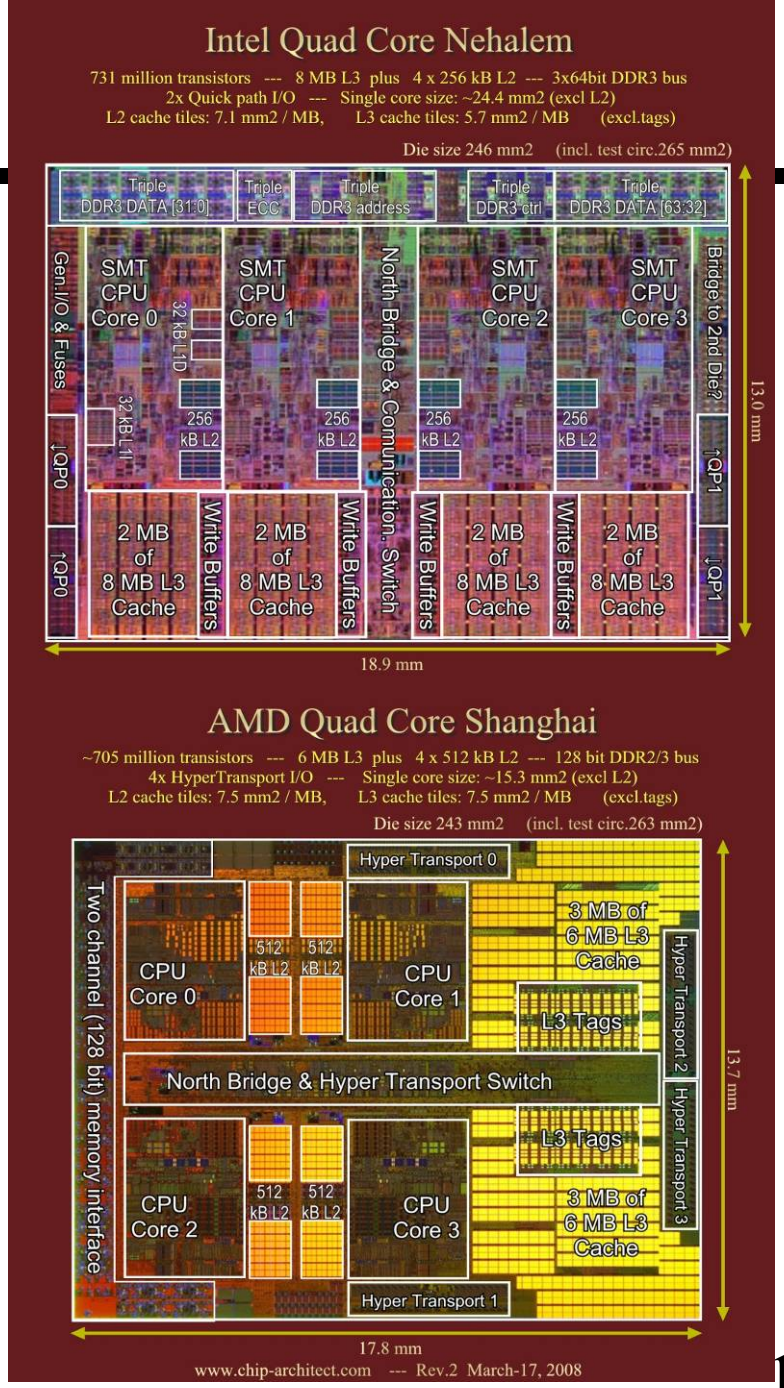
- 上位階層の小容量のメモリを、通常キャッシュメモリあるいはキャッシュと呼ぶ
- 通常、SRAM (4-6 ns アクセスタイム)で構成、容量数kB – 数MB
  - c.f. 下位階層のメインメモリはアクセスタイム60ns程度のDRAM
  - メインメモリの一部をキャッシュの中に自動的にコピーし、高速なメモリに対して読み(書き)ができるようする
  - キャッシュのヒット→データは直接キャッシュから読まれる
  - キャッシュのミス→ハードウェアが自動的にメインメモリをアクセス、コピーを行う
  - 局所性が十分あれば、多くのメモリアクセスはキャッシュに対するものになり、遅いメインメモリへのアクセスが激減
- 問題点:
  - あるデータが、キャッシュに入っているかどうかは、どのように検出する?
  - メインメモリと、キャッシュメモリの対応はどのようにとる?
  - キャッシュが一杯になったら、どの古いデータ(Victim)を捨てる?



キャッシュメモリの写真

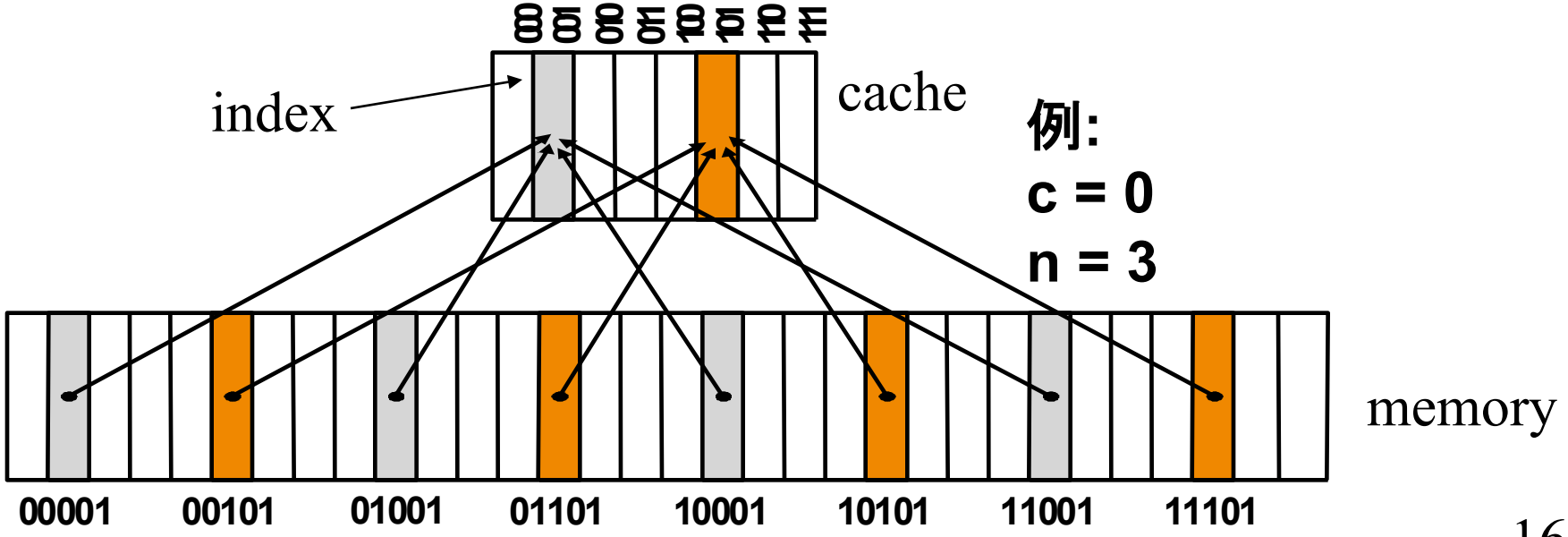


Intel Itanium



最も単純なキャッシュ: Direct Mapped キャッシュ

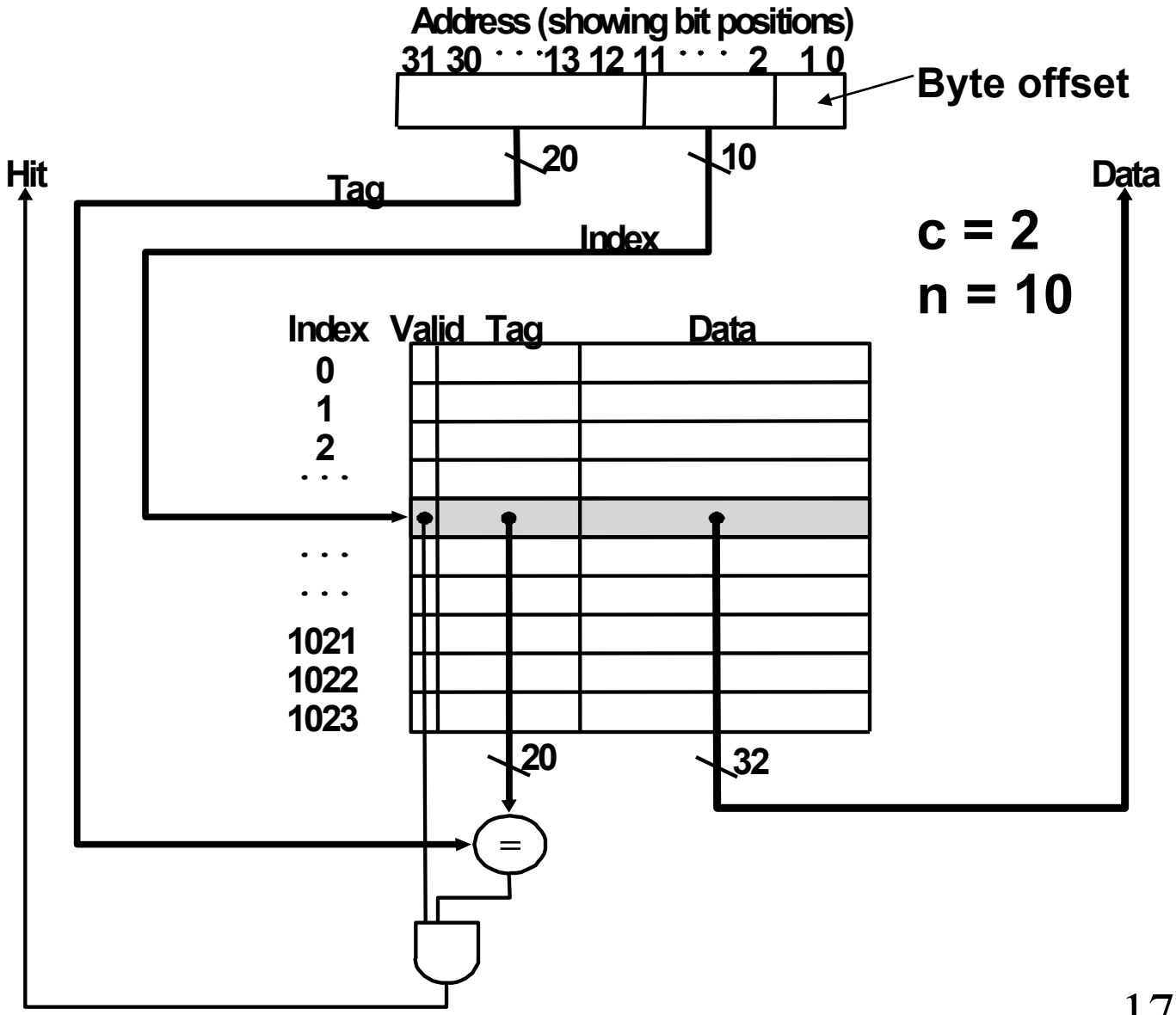
- キャッシュブロックサイズが2^c, ブロック数2^nとする (Pentiumではc = 5 (32バイト), n = 8) →キャッシュは 8kB
- 以下の対応関係: キャッシュメモリの上位nビットのアドレス(インデクス)は、メインメモリのアドレスの上位 (32 - c)ビットのうちの下位nビット→このメモリ「ブロック」をキャッシュ
- 上位(32 - c - n)ビットは、同じキャッシュ「ブロック」に対する異なるメインメモリのアドレスを識別するタグとして使用



Direct Mapped キャッシュの実装(1)

- 実装その1:

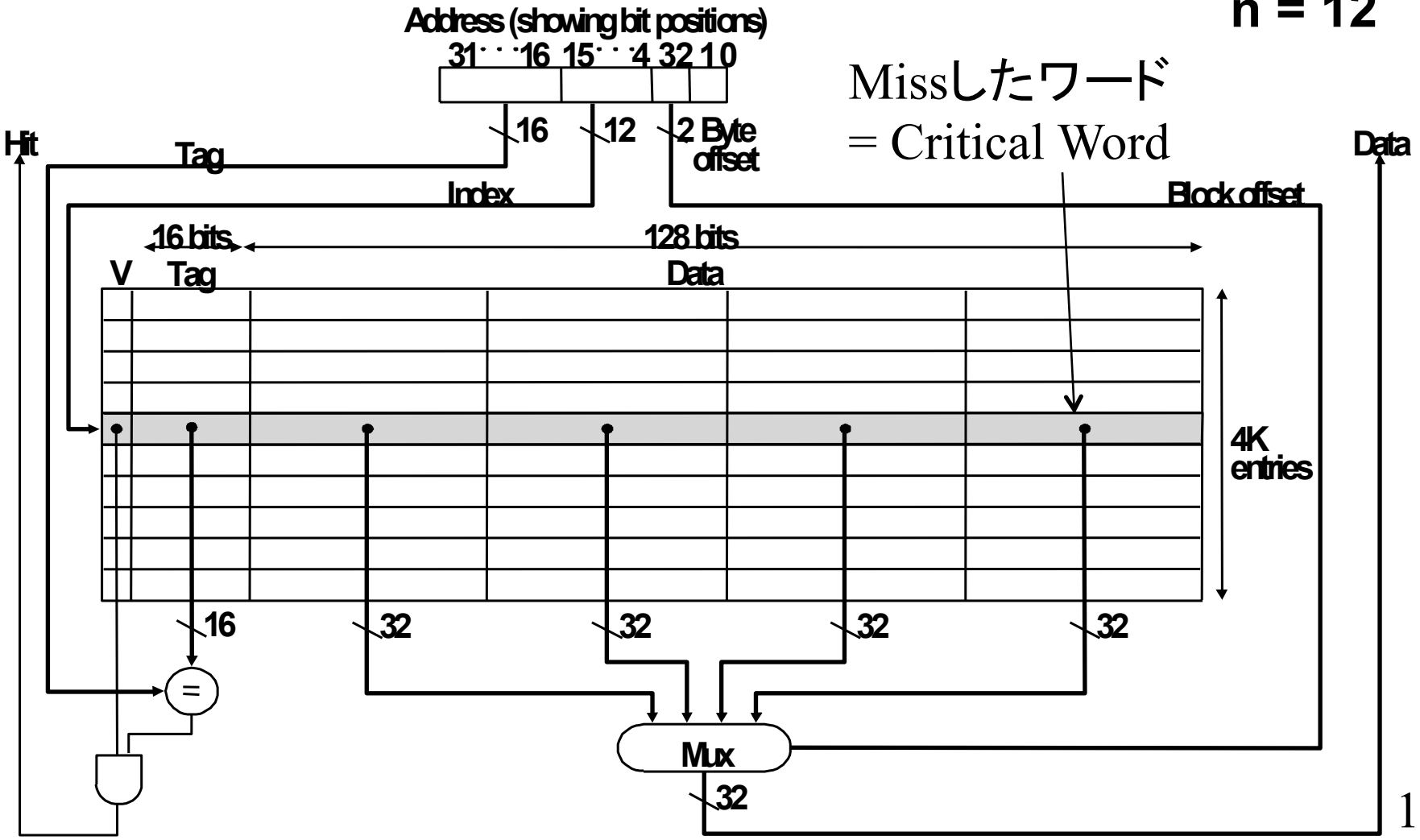
このキャッシュはどのような局所性を利用しているのだろうか?



Direct Mapped キャッシュの実装(2)

- 実装その2: 空間的局所性の活用:

c = 4
n = 12

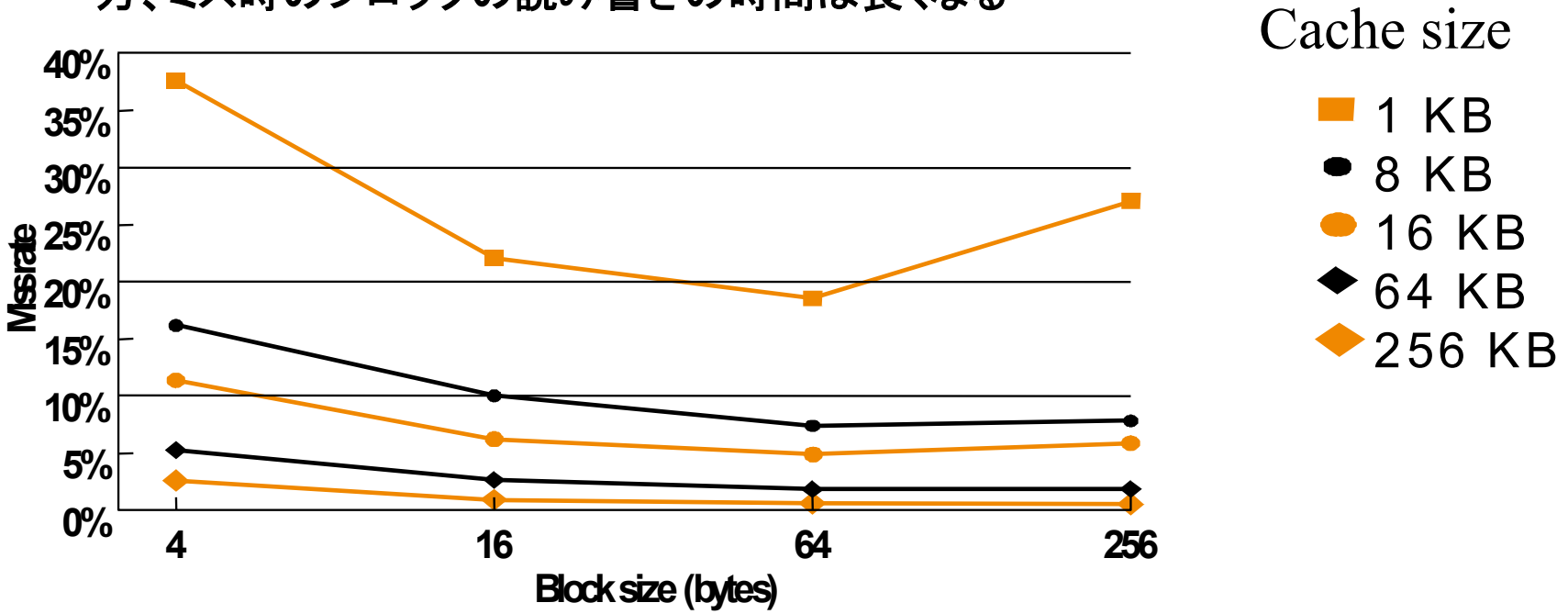


キャッシュのヒットとミス

- キャッシュの読み書きで、ヒットとミスが起きる
- 読みだしのヒット (read hit)
  - これが起きて欲しい!
- 読みだしのミス (read miss)
  - CPUをストールし、メインメモリからブロックを読み出し、キャッシュに書き込んで、実行を再開する
- 書き込み時のヒット (Write hit):
  - キャッシュとメモリの内容を同時に更新 (write-through方式)
  - キャッシュのみに書き込む (write-back方式)
  - write-back方式のほうが高速だが、管理が難しい
    - 前のキャッシュの値をメモリに書き戻さなくてはならない(Victim)
    - 後述のassociative cacheの場合は、それをvictimにするか?
    - マルチプロセッサ時の管理が難しい
- 書き込み時のミス (Write miss):
  - 単純に上書き(write-through方式)
  - Read Missと同様、ブロックをキャッシュに一度読み込み、キャッシュに書き込む(write-back)

キャッシュによる性能向上(1)

- ブロックサイズが大きくなると、ミスの割合(ミスレート)が少なくなる。
  - 一方、ミス時のブロックの読み書きの時間は長くなる



- 多くの場合、命令とデータは別なキャッシュ→命令の方が空間的局所性が高い

Program	Block size in words	Instruction miss rate	Data miss rate	Effective combined miss rate
gcc	1	6.1%	2.1%	5.4%
	4	2.0%	1.7%	1.9%
spice	1	1.2%	1.3%	1.2%
	4	0.3%	0.6%	0.4%

キャッシュによる性能向上(2)

- 性能向上のモデル化:
  実行時間 = (実行サイクル数 + ストールサイクル数) x サイクルタイム
  ストールサイクル数 = 命令数 x ミスの割合 x ミスのペナルティ
  ミスのペナルティ = メインメモリからキャッシュへのブロック転送時間+α

- 性能を向上する二つの手法:
  - ミスの割合を減少させる
  - ミスのペナルティを減少させる

Q:単純に、ブロックサイズを大きくするとどうなる?

連想性 (associativity)増加によるミスの割合の減少

連想メモリ(associative memory): 理想

任意のアドレス値に対し、値を格納できるようなメモリ

キャッシュメモリの連想性

「同じindexに写像されるメモリに対し、幾つのブロックを対応させるか」

direct map: 1つ

n-way set associative: n 個

fully associative: 任意個 (キャッシュメモリの容量まで) → 完全な連想メモリ

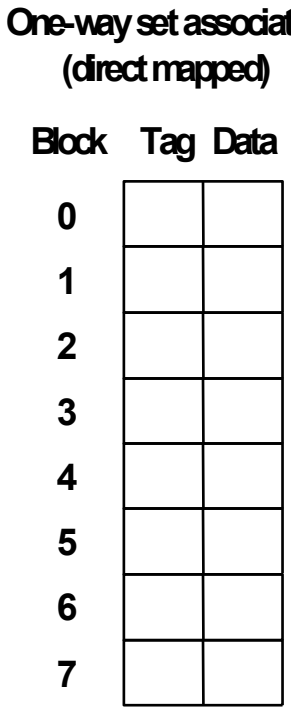
同インデクスに対し、タグで区別

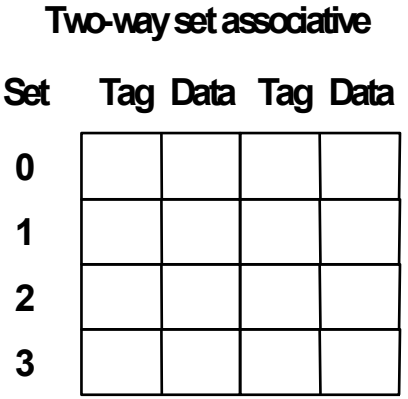
ただし、n-wayにすると、インデクスの範囲は1/nになる。また複雑、遅くなる

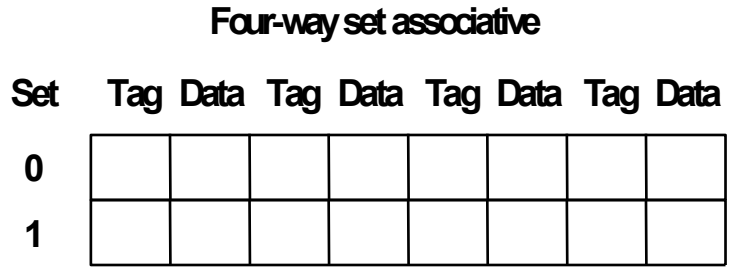
ミスの割合が向上するか否かは、メモリアクセスの性質による

Q: direct map / 2-wayが有利な場合は?

問題: n-wayの場合、どのwayをvictimにするのか（victim selection)

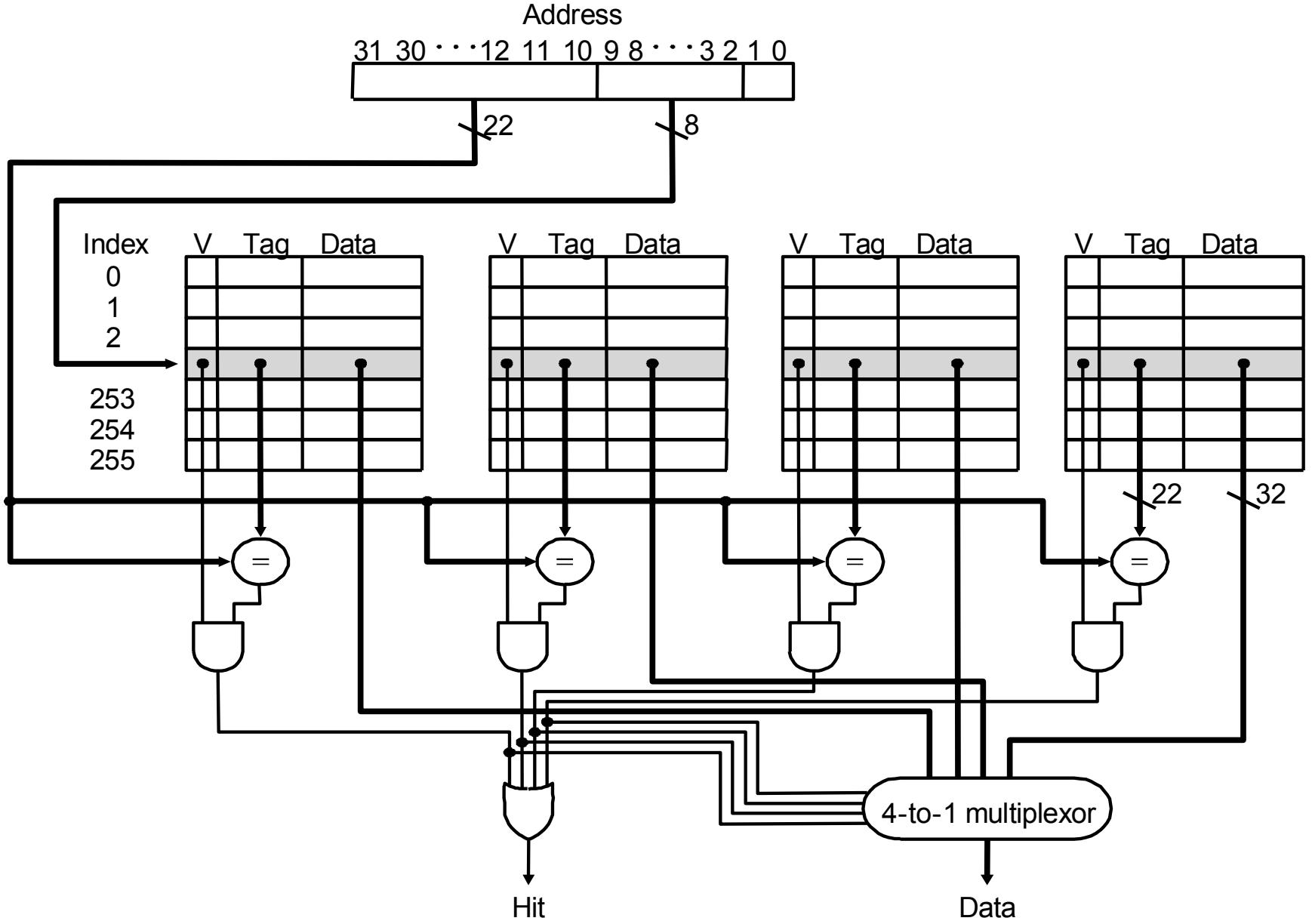




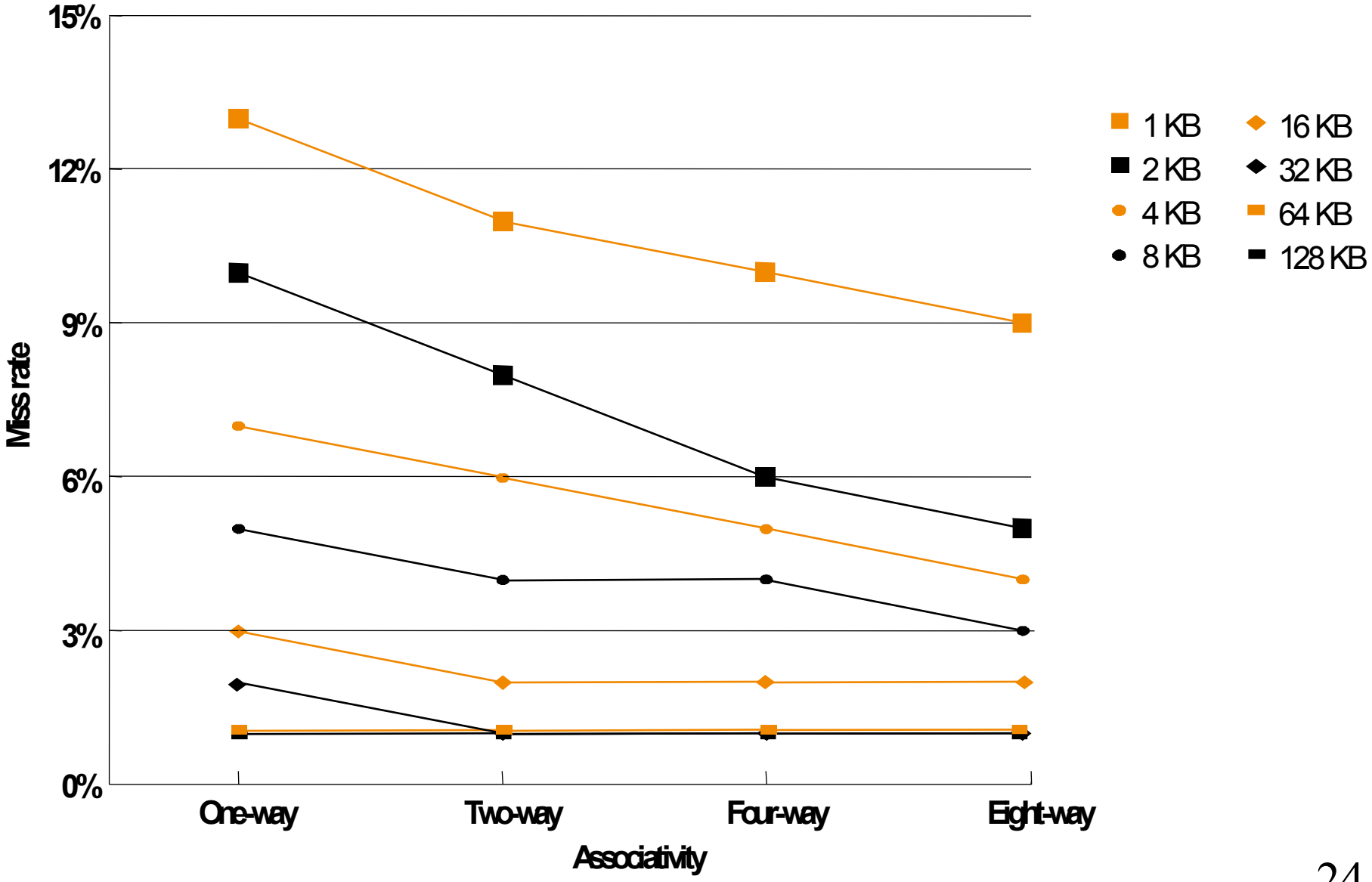


LRU = Least Recently Used

4-way set associative cacheの実装



Associative Cacheによるミスの割合の改善



多階層キャッシュによるミスペナルティの減少

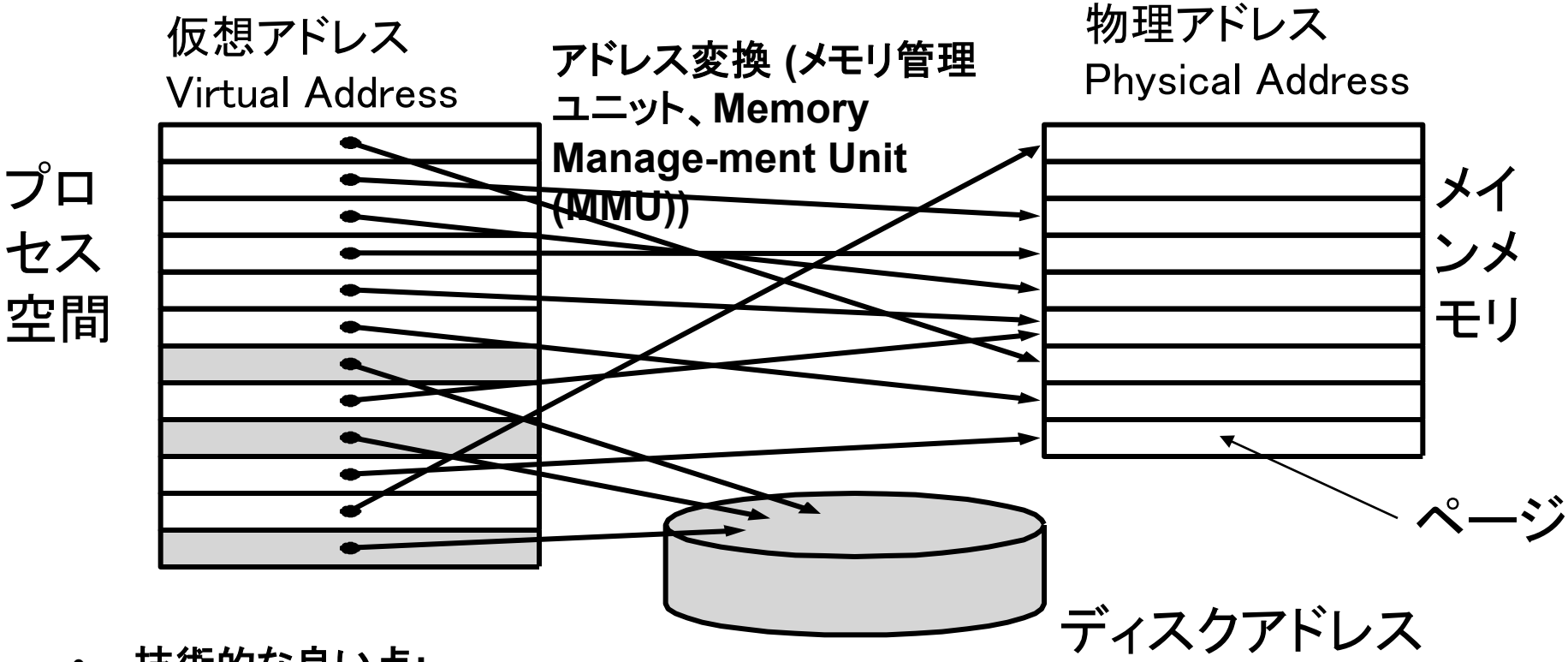
- さらに、第二のキャッシュの階層を追加 (二次キャッシュ) :
  - 一次キャッシュは通常プロセッサと同じチップ上に実装
  - 高速なSRAMを用いて、一次キャッシュとメインメモリの間に一次キャッシュより大容量の二次キャッシュを設置する
  - 二次キャッシュにデータがあれば、ミスのペナルティは減少(アクセスタイムがはるかに高速)

- 例:
  - (ちょっと古いが) Pentium+EDO RAM, 133Mhz/66Mhz bus clockでは、キャッシュミス時のメモリからのブロックの転送に11サイクル=90nsかかる。
  - 同じアクセスが二次キャッシュにヒットすると、2-3倍アクセスが高速化→ミスペナルティの減少、30% 以上の高速化
  - 近年では、半導体の集積率の増加とともに、二次キャッシュは増加している

- 多階層キャッシュを用いる場合:
  - 一次キャッシュのヒット時の速度を高速化するようにする
  - 二次キャッシュのミスの割合を最小化するようにする

仮想記憶 (Virtual Memory)

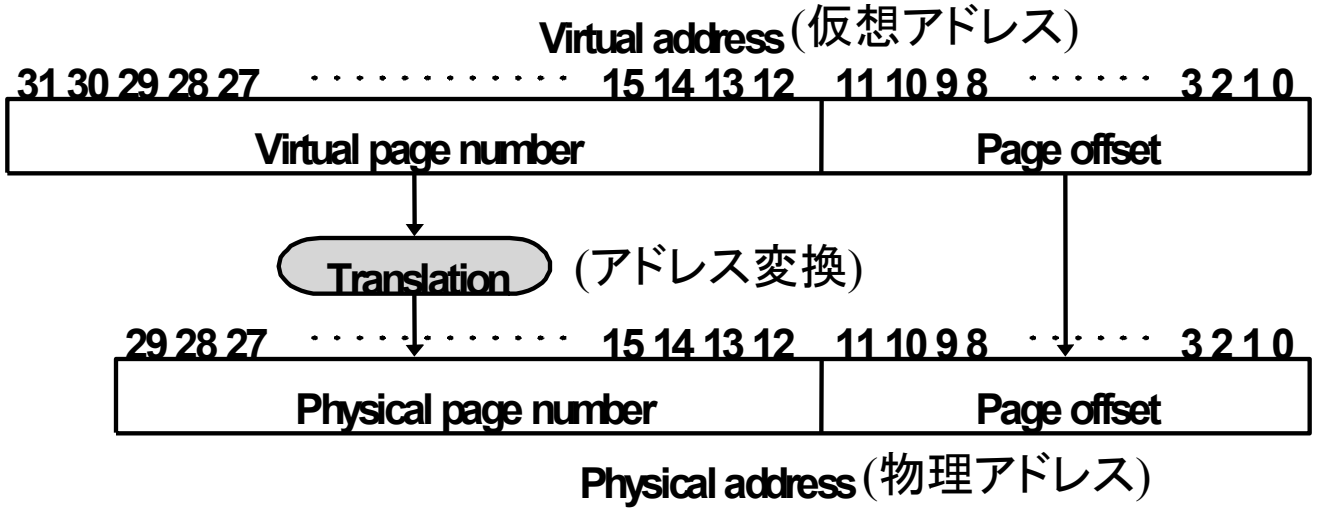
- 記憶階層:メインメモリが二次記憶装置(ディスク)のキャッシュ



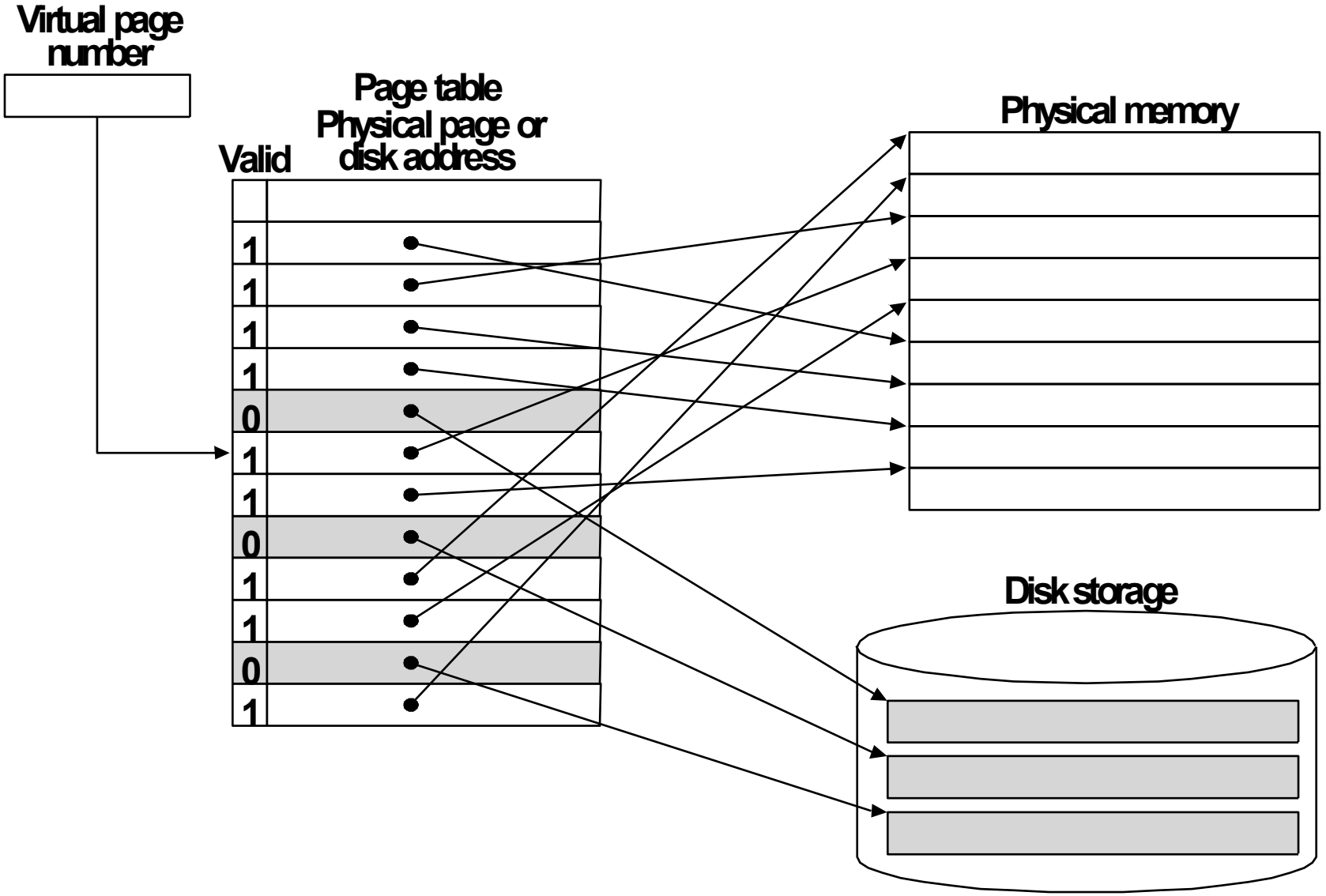
- 技術的な良い点:
  - 物理メモリが増えたように見える
  - プログラムの再配置が容易
  - OSでの保護、その他、新しいOS機能のサポートの基本的メカニズム

ページ: 仮想メモリのブロック

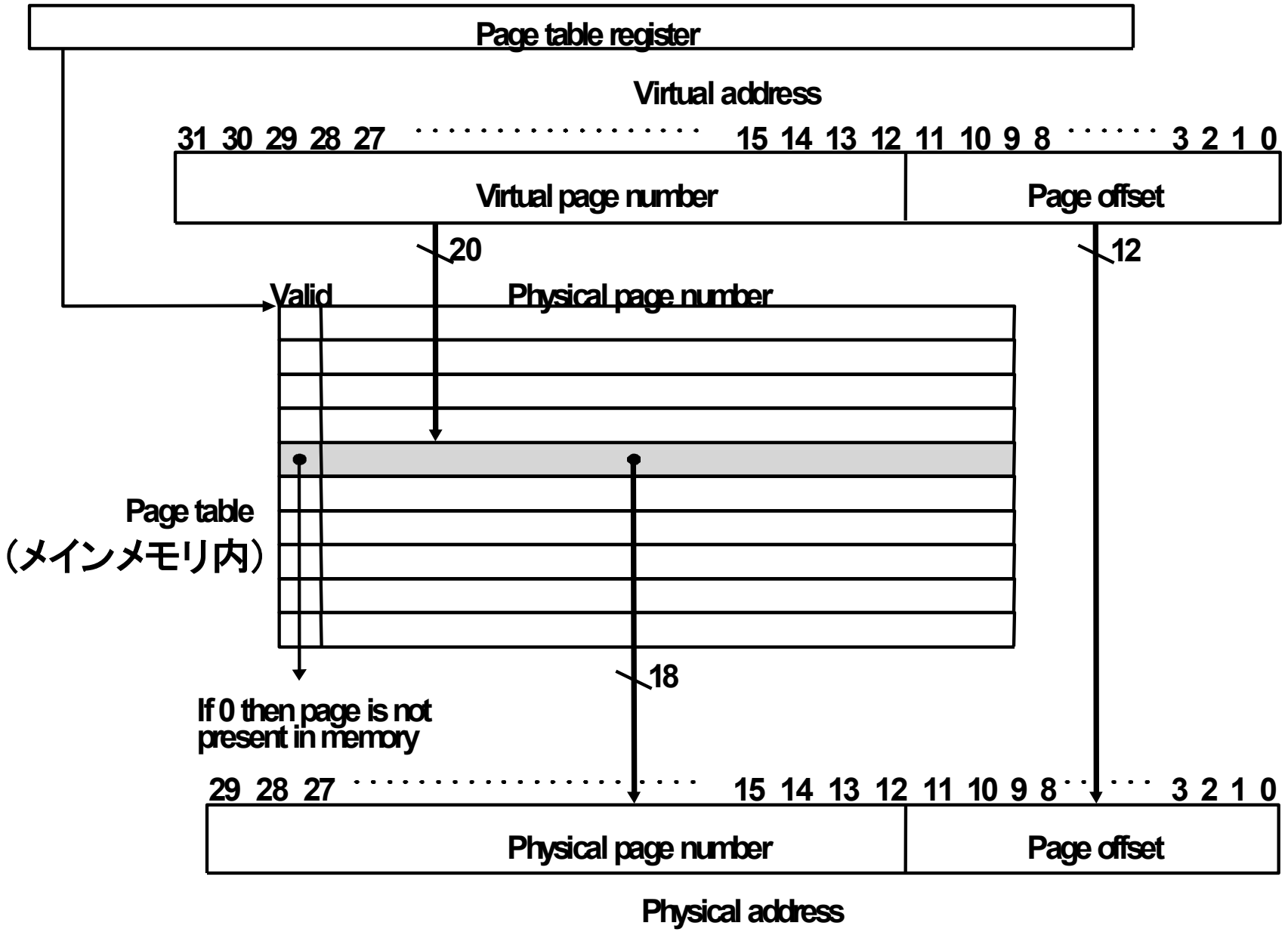
- ページフォルト: ページ内のミス、つまり、データがメインメモリにない
  →ディスクから読み込みが必要
  - ミスペナルティは莫大(ディスクアクセス速度はメモリの1000倍、しかし、スループットは50倍)→ページはなるべく大き目に(e.g., 4KB)
  - ページフォルトの回数を最小化するのが重要
    - どのページを置き換えるか ((Victim) replacement policy)
    - LRU (Least Recently Used)アルゴリズムを通常用いる
  - 速度が遅く、管理が複雑なため、多くの処理をソフトウェアで扱う
  - write throughはコストが高すぎるので、writebackを用いる



ページテーブル: ページの対応関係の管理

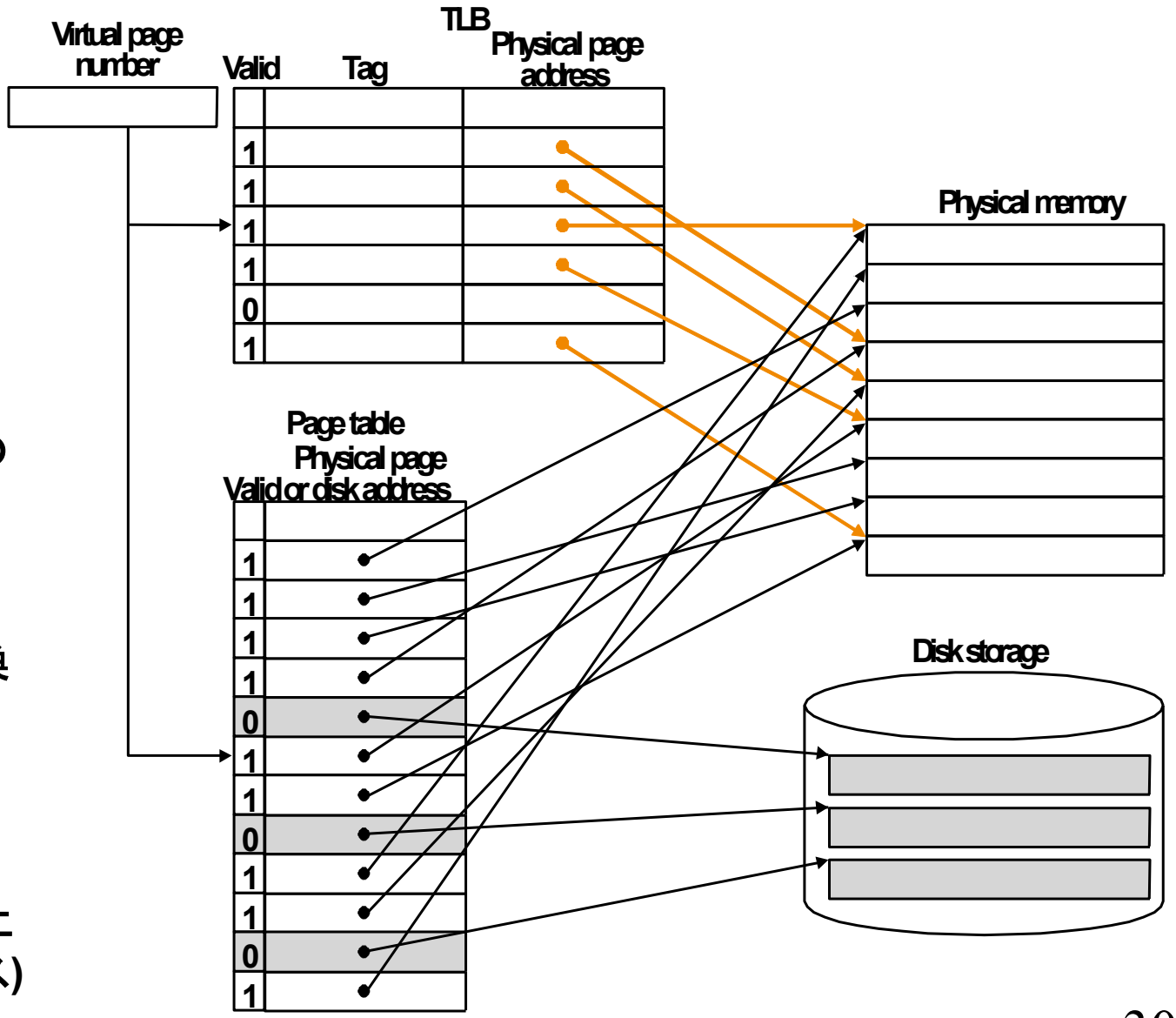


ページテーブルの実装

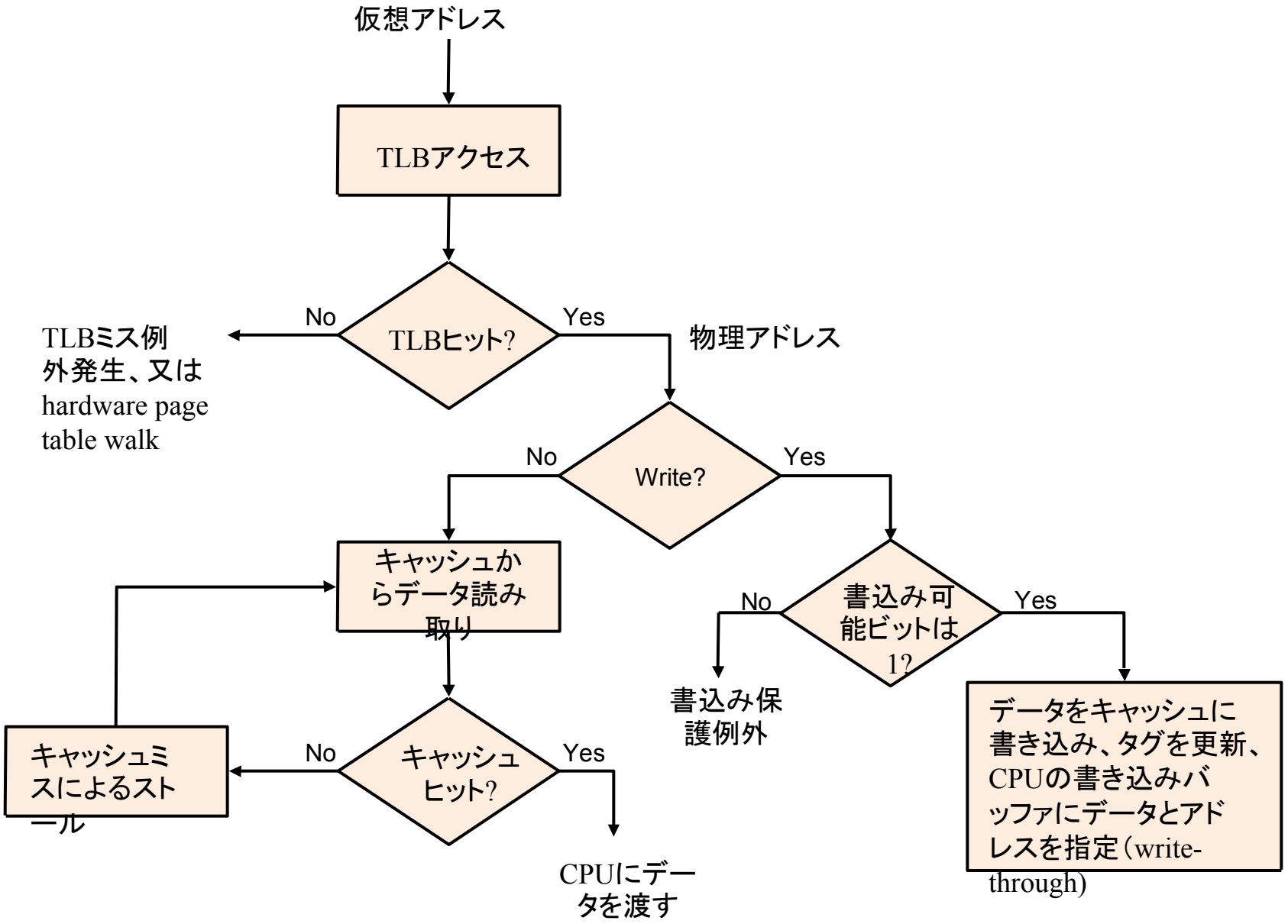

Page table register
Virtual address
31 30 29 28 27 ..... 15 14 13 12 11 10 9 8 ..... 3 2 1 0
Virtual page number
Page offset
20
12
Valid
Physical page number
Page table
（メインメモリ内）
18
If 0 then page is not
present in memory
29 28 27 ..... 15 14 13 12 11 10 9 8 ..... 3 2 1 0
Physical page number
Page offset
Physical address

TLBによるアドレス変換の高速化

- アドレス変換は、実際はメインメモリ内のページテーブルを何回にもわたってアクセス
  - メモリアクセスのたびに行わなくてはならない→著しい速度低下
- そこで、アドレス変換のキャッシュを設ける→ translation lookaside buffer(TLB)
  - 通常は32-128エントリ(インデクス)
  - 高い連想性



TLB とキャッシュの総合連携

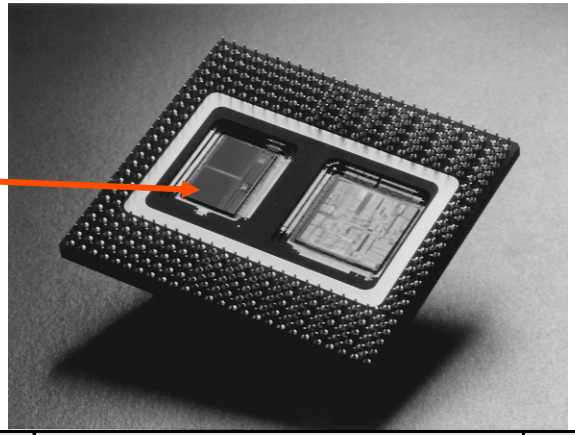


最近のシステムのメモリ階層の実例

- 高速化のために、大変複雑なメモリシステム:

Characteristic	Intel Pentium Pro	PowerPC 604
Virtual address	32 bits	52 bits
Physical address	32 bits	32 bits
Page size	4 KB, 4 MB	4 KB, selectable, and 256 MB
TLB organization	A TLB for instructions and a TLB for data	A TLB for instructions and a TLB for data
	Both four-way set associative	Both two-way set associative
	Pseudo-LRU replacement	LRU replacement
	Instruction TLB: 32 entries	Instruction TLB: 128 entries
	Data TLB: 64 entries	Data TLB: 128 entries
	TLB misses handled in hardware	TLB misses handled in hardware

二次キャッシュ



Pentium Pro

Characteristic	Intel Pentium Pro	PowerPC 604
Cache organization	Split instruction and data caches	Split intruction and data caches
Cache size	8 KB each for instructions/data	16 KB each for instructions/data
Cache associativity	Four-way set associative	Four-way set associative
Replacement	Approximated LRU replacement	LRU replacement
Block size	32 bytes	32 bytes
Write policy	Write-back	Write-back or write-through

現在の技術的トレンド

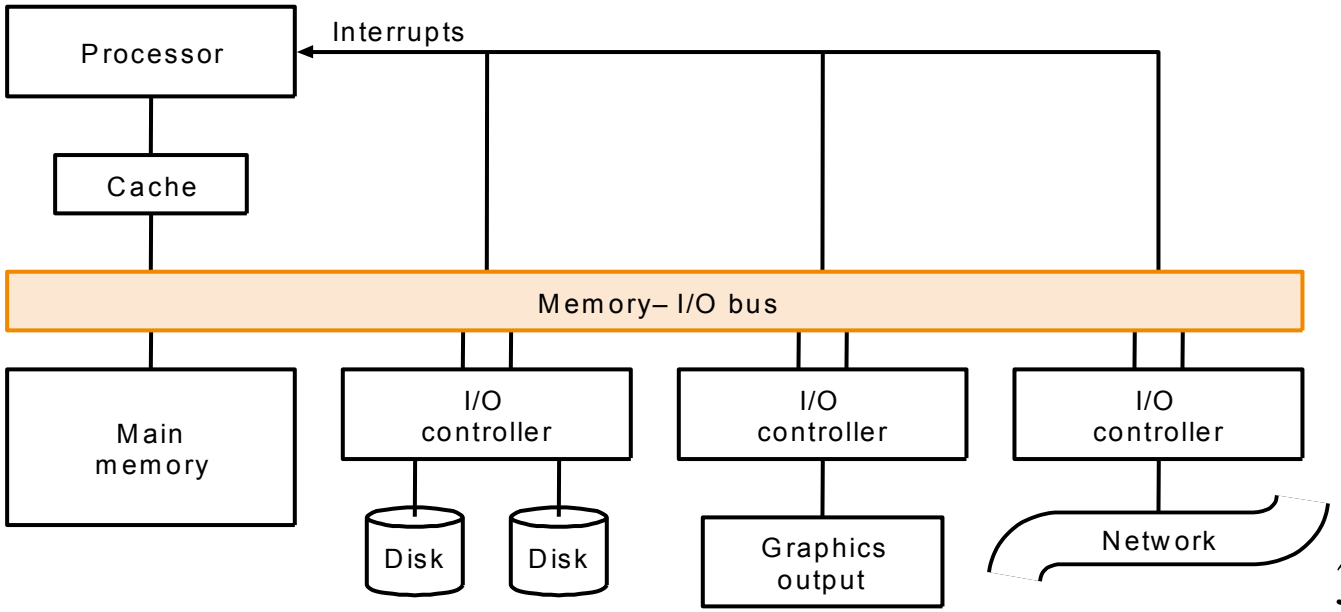
- プロセッサの速度が急激に向上
  — DRAMやディスクの速度向上とは比較にならない
- 技術的課題: この差にどのように対処するか? (メモリーウォール問題: メモリの速度向上がなされないために、プロセッサの速度向上が頭打ちになる)
- 技術的トレンド:
  - ハードウェアによる向上: より深く早いメモリ階層
    - キャッシュの向上: 3次キャッシュ、2次キャッシュの高速化、大容量化
    - メモリスループットの向上: 同期メモリ、(Pipeline Burst SRAM, Synchronous DRAM, DDR/QDR RAM)、インターリービング
    - マルチスレッド(SMT)、マルチコア(CMP)化(レーテンシ隠蔽)
  - ソフトウェアによる向上
    - コンパイラによる局所性の向上
    - プリフェッチング、ポストストア
    - などなど

I/Oに関して

(部分的なカバー)

プロセッサと周辺機器のインターフェース

- I/O: Input/Output (周辺機器との)入出力
- I/O のデザインはさまざまな要因に影響される(拡張性、対故障性)
- 性能に対する要因:
  - アクセスのレーテンシ
  - スループット
  - デバイスとシステムとの結合形態
  - メモリ階層
  - オペレーティングシステム
- さまざまな種類のユーザ (例: 銀行、スパコンによる科学技術計算、技術者、ゲーム、OA)



I/O

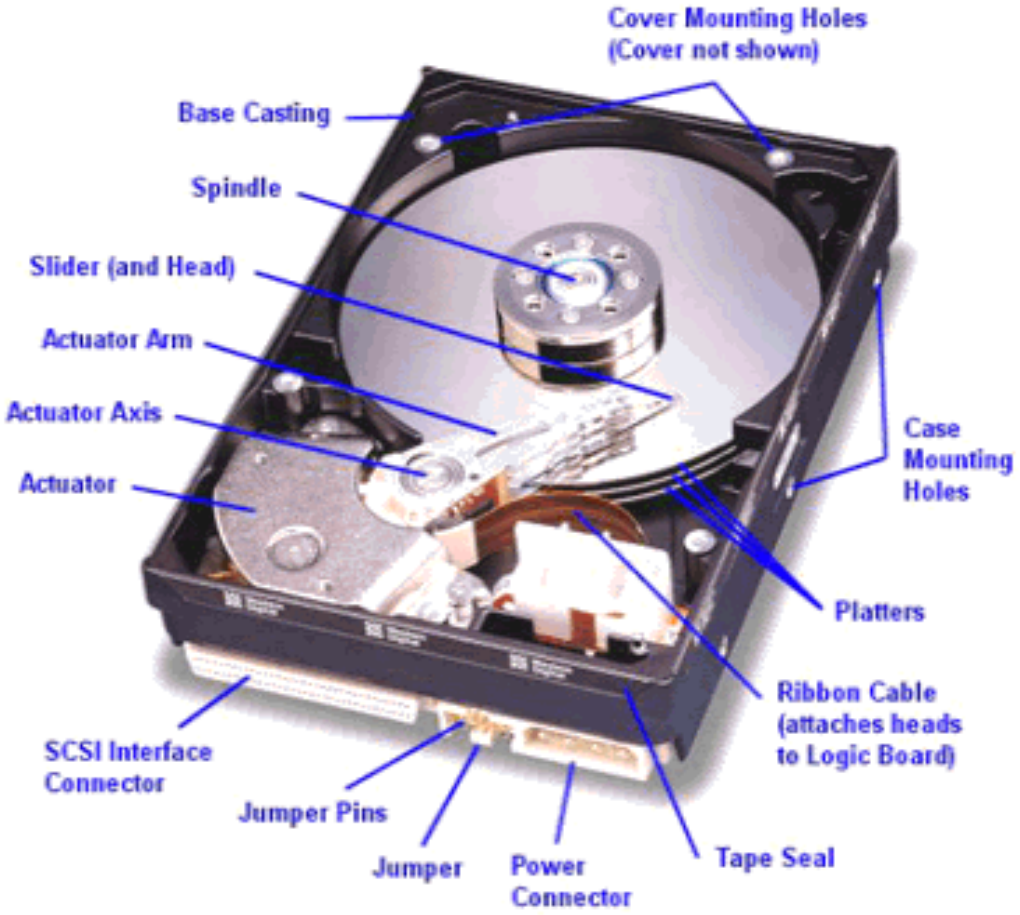
- 重要だが、しばしば設計で無視されることがある
  - 「I/Oシステムの解析や設計に関する困難さはI/Oを下層階級的位置にたらしめてしまった。」
  - 「計算機科学に関するあらゆる授業は、それがプログラミングから計算機アーキテクチャにいたるまで、I/Oをほとんど無視するか、ちゃんとやらない」
  - 「教科書もそれを一番後ろに持ってくるので、時間がないなどの理由で簡単にスキップされる」
  - 我々も同罪である。。。

I/O デバイス

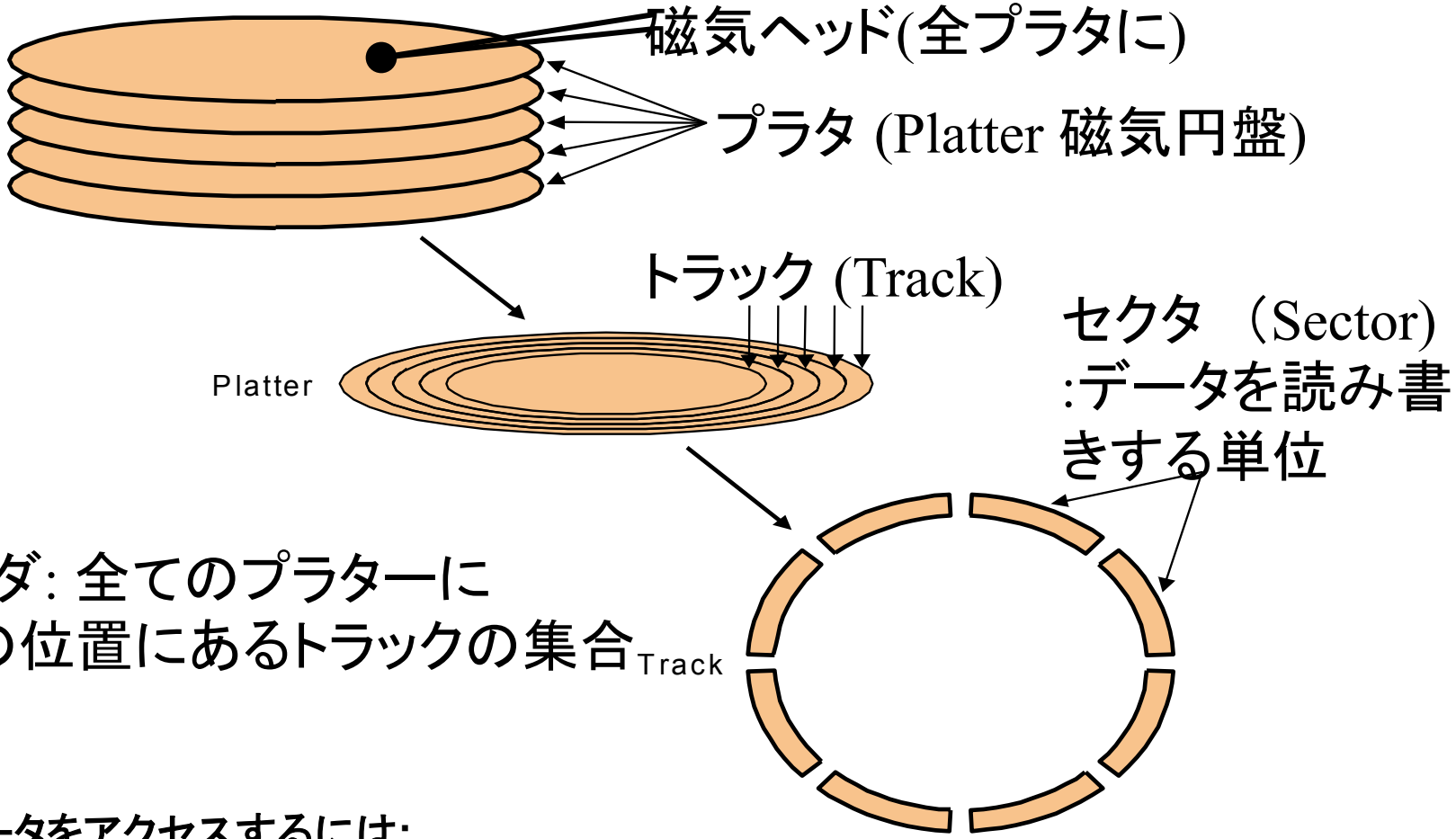
- さまざまな種類のデバイスが存在する
  - 入出力 (input vs. output)
  - 対象とする相手 (who is at the other end?)
  - データ転送の速度

デバイス	Input/Output	対象	データ転送 (KB/秒)
Keyboard	input	human	0.01
Mouse	input	human	0.02
Voice input	input	human	0.02
Scanner	input	human	400.00
Voice output	output	human	0.60
Line printer	output	human	1.00
Laser printer	output	human	200.00
Graphics display	output	human	60,000.00
Modem	input or output	machine	2.00-8.00
Network/LAN	input or output	machine	500.00-6000.00
Floppy disk	storage	machine	100.00
Optical disk	storage	machine	1000.00
Magnetic tape	storage	machine	2000.00
Magnetic disk	storage	machine	2000.00-10,000.00

Hard Disk Pictures



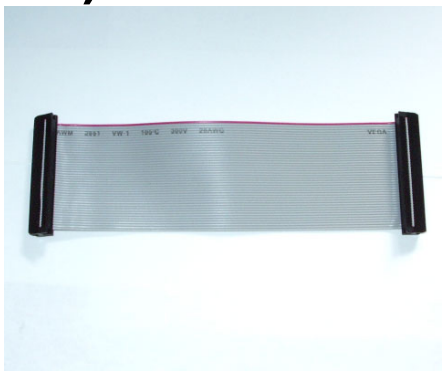
I/O の例: ディスクドライブ



- データをアクセスするには:
  - シーク: プラタ上で適切なトラック上に磁気ヘッドを移動 (平均 数ミリ秒)
  - 回転レーテンシ: 必要なセクタが回転してくるまで待つ (平均.5 / RPM)
  - データ転送: 磁気面への読み書き (複数の連続するセクタ) 50~100 MB /秒

HDDをインターフェースする様々なインターフェース

- HDDの密度の向上 (5 inch 10MB=> 3.5inch 3TB)
  (O(sqrt(d))で向上)
  ＋回転速度の向上 (4200rpm=>15,000rpm)
  - 100MB~200MB/s
- 過去: パラレルインターフェース
  - SCSI – SCSI, UltraSCSI, UltraSCSI2, …
  - IDE – ATA, UltraATA, UltraATA66, …
- 現在: シリアルインターフェース
  - SATA – 1.5Gbps, SATA2 – 3Gbps
    SATA3 – 6Gbps
  - SAS (serial attached SCSI)

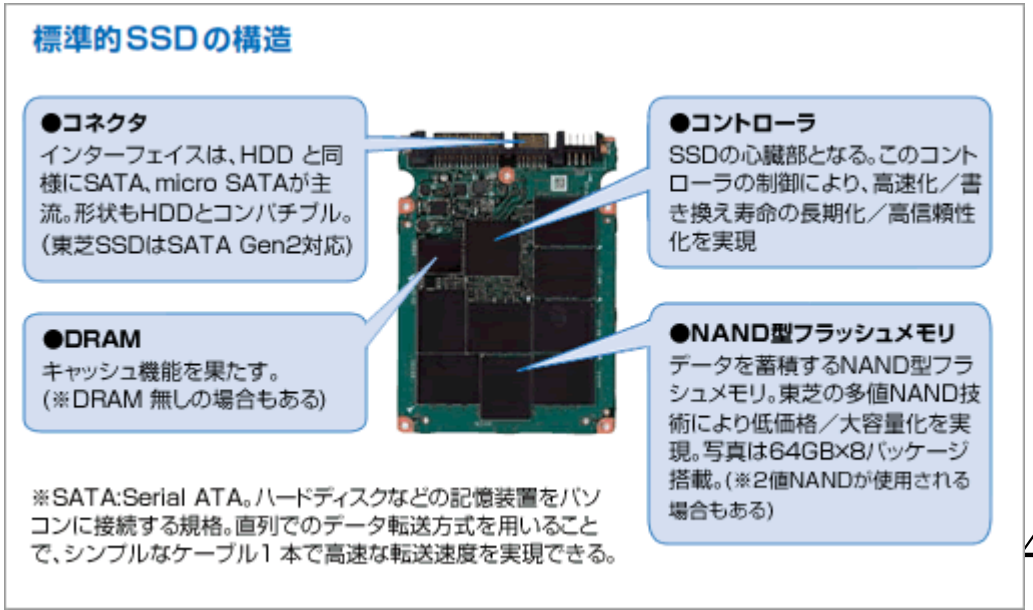


↓ 磁気密度向上の限界 (垂直磁化など)、過度な大容量化

↑ フラッシュメモリの台頭により、更なる高速化

フラッシュメモリ/SSDの台頭

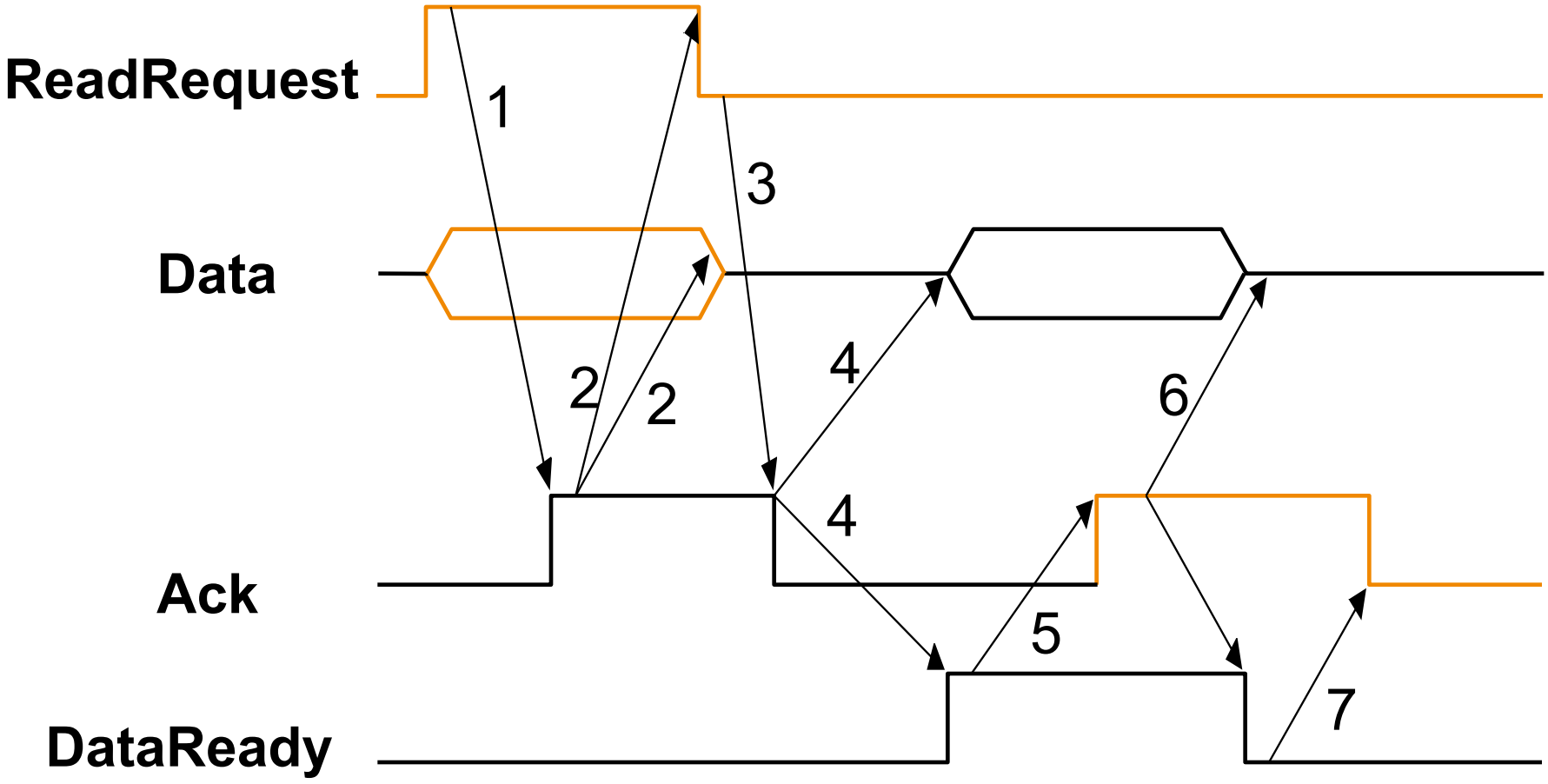
- フラッシュメモリ: 新しい不揮発性メモリの技術。(半導体メモリより) 大容量・低価格・低消費電力
- SDカード/メモリスティック、USBメモリなどリムーバブルメモリに多用
- SSD (Solid State Disk): フラッシュメモリをハードディスク置き換えに応用
  - レーテンシ・スループットはDRAMより遅いがHDDよりかなり高速
  - レーテンシ:　数十倍高速 (数十マイクロ秒　vs. 数ミリ秒)
  - スループット: 数倍高速 (250-300MB/s以上 vs. 100MB/s)
  - より高い耐故障性(振動に強い)
- ビット単価は約10倍程度と多少高額だが、今後ノートやタブレットPCなどで急速に置き換わる可能性大



I/O の例: バス

- 共有される通信リンク (複数のワイヤによる配線)　例: ISAバス、PCIバス
- デザイン上の問題:
  - 速度: ボトルネックになる可能性
  - バスの配線長 (AGPバス: 100Mhzで動作)
  - デバイスの数 (PCIでは通常4-5スロット)
  - トレードオフ (バンド幅を増やすためバッファを入れると、レーテンシが増大)
  - さまざまな種類のデバイスのサポート (ビデオ、オーディオ、ディスク、プリンタ、などなど)
  - コスト (PC マザーボード 1-2万円)
- バスの種類:
  - プロセッサ-メモリ (短配線長、高速 (数百M~数GB/秒)、専用のデザイン
  - バックプレーン (高速, 標準化、例: ISA, PCI, PCI-e)
  - I/O (長配線長, 異なるデバイス, 標準化, 例: SCSI, USB, IEEE1394)
- 同期バス 対 非同期バス
  - 同期バス: クロックを用い、同期プロトコルを用いる、高速で小さいが、全てのデバイスは同じ速度で動作の必要、クロックのスキュー(遅延によるクロックのずれ)が要因でバスは短配線長
  - 非同期バス: クロックを用いず、ハンドシェーキングを行う

例: PCI等非同期バスのハンドシェーキング

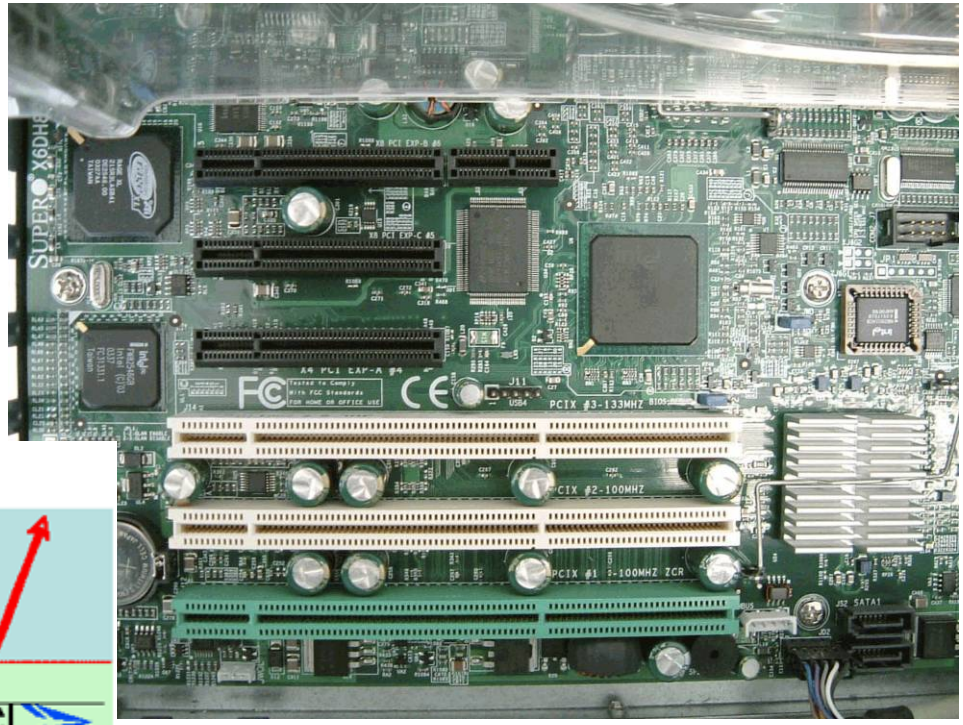


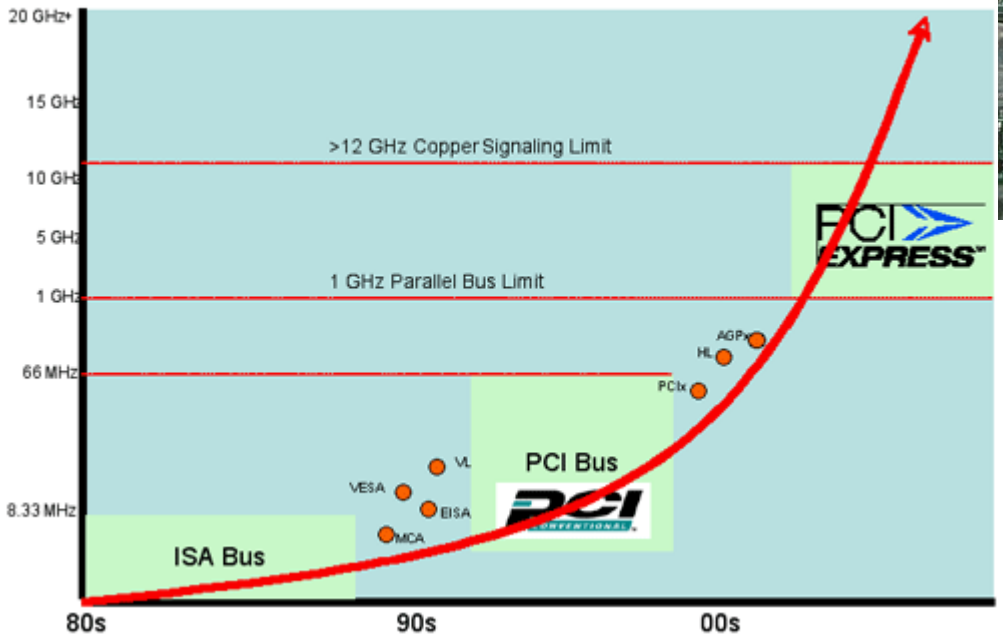
I/Oの他の重要な項目

- バスの調停 (Bus Arbitration):競合が起こったらどうするか
  - daisy chain arbitration (公平でない)
  - centralized arbitration (調停回路arbiterが必要), e.g., PCI
  - self selection, e.g., NuBus (古いMacintoshで使用)
  - collision detection, e.g., Ethernet
- オペレーティングシステム:
  - ポーリング (polling)
  - 割り込み (interrupts)
  - DMA
- 性能評価の手法:
  - 待ち行列理論 (queuing theory)
  - シミュレーション simulation
  - 解析
- たくさんの新たな発展
  - 高速化、大規模大容量、遠距離、さまざまな種類のデバイス、などなど

ISAからPCI から PCI Expressへ

- ISA – 8MB/s
- PCI – 133MB/s
- PCI-X – 512MB/s-1GB/s
- PCI-Express x4 – 1GB/s
- PCI-Express 2.0 x16 – 8GB/s
- PCI-Express 3.0 x16 – 16GB/s





ネットワーク等

- イーサネット Ethernet
  - アロハネットワークを元に、1970年代にゼロックス社パロアルト研究所で開発
  - 40年間で10000倍に増速 (~3Mbps => 40Gbps)
  - CSMA/CDプロトコル（1000Base-xまで)
  - 媒体はCategory Xの銅線ケーブル、および光ファイバ
- 無線LAN
  - 802.11規格など
  - 現在最高600Mbps
- スパコンではより高速なネットワークも用いられる
  - Infiniband QDR (40Gbps)
  - Infiniband FDR (56Gbps)

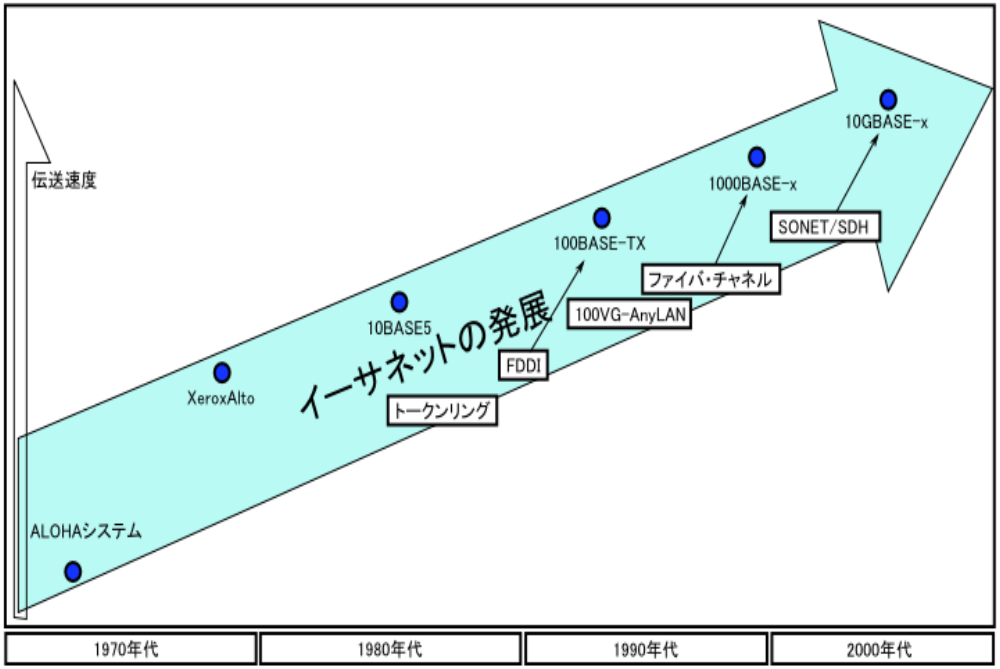
イーサネットの基本的なフレーム形式

DIX(イーサネットII)規格

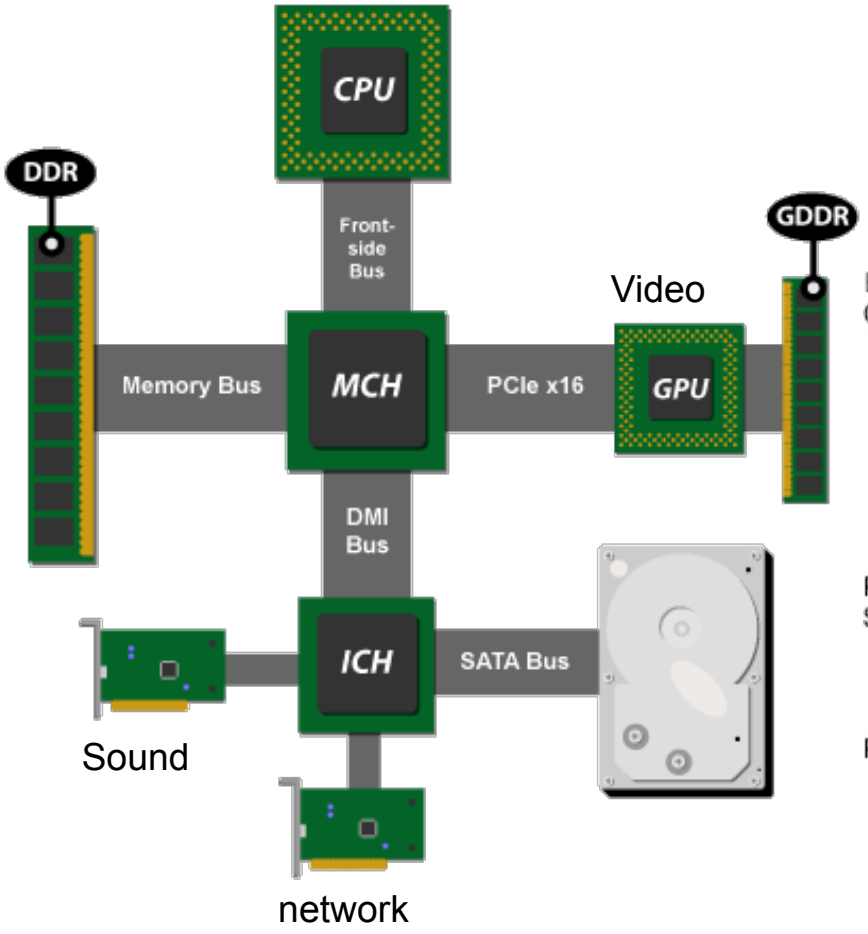
あて先アドレス	送信元アドレス	タイプ	データ	エラー検出
6バイト	6バイト	2バイト	46～1500バイト	4バイト

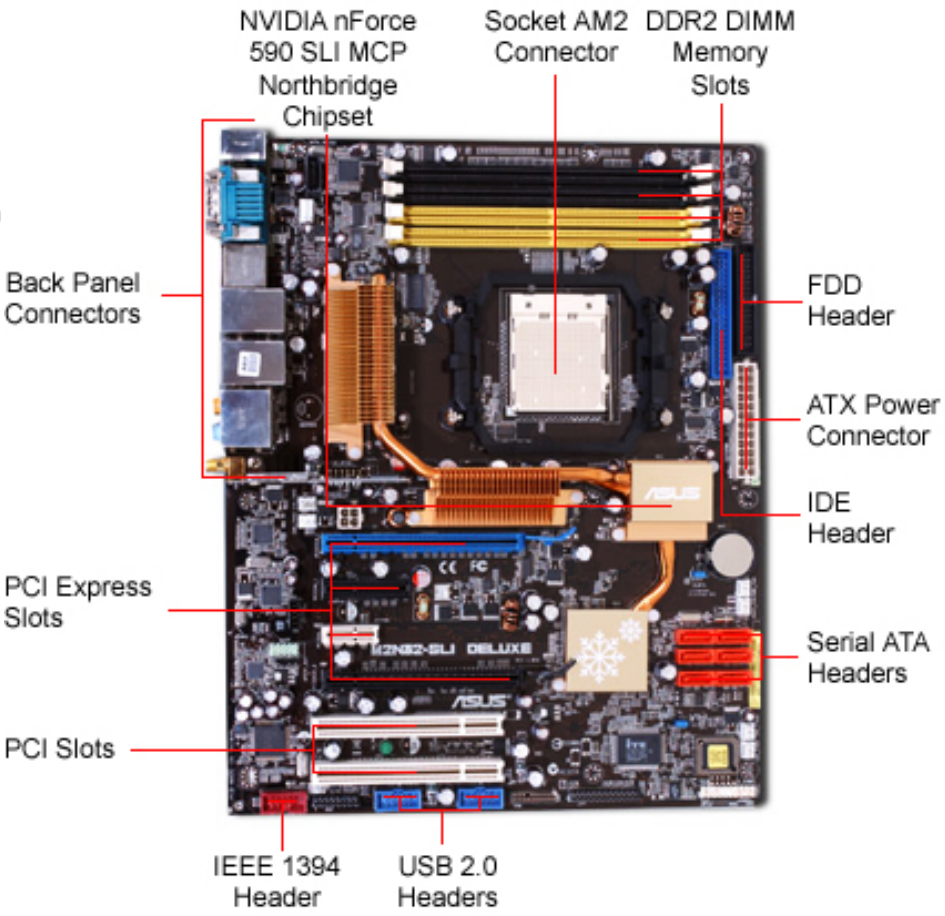
IEEE 802.3規格

あて先アドレス	送信元アドレス	長さ/タイプ	データ	エラー検出
6バイト	6バイト	2バイト	46～1500バイト	4バイト



Modern Day PC Architecture and Motherboard

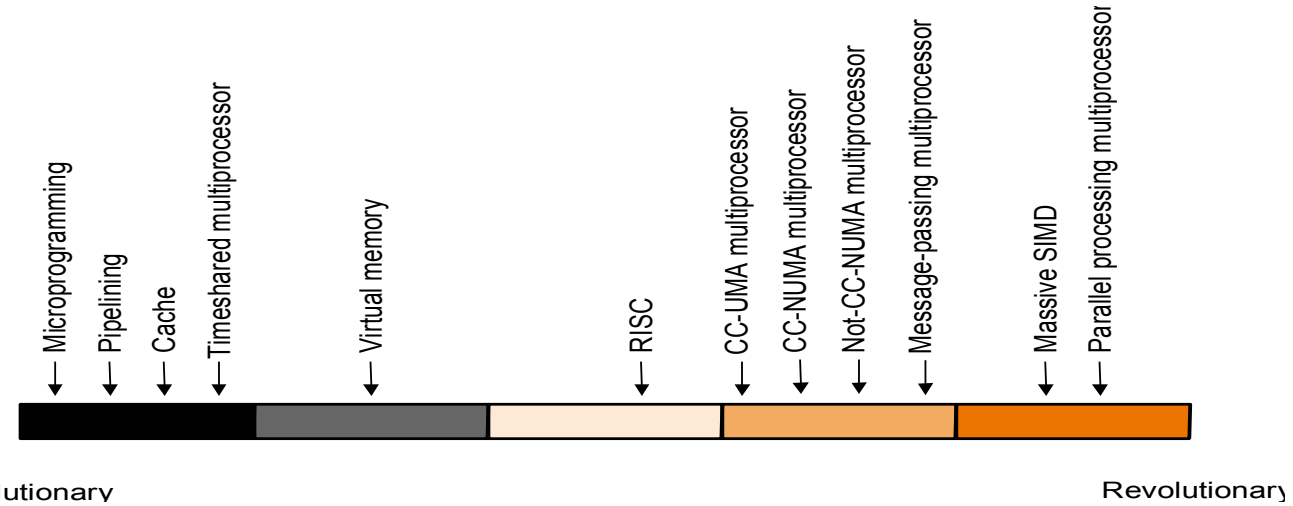




最後に

- **Evolution vs. Revolution (進化 対 革命)**

“More often the expense of innovation comes from being too disruptive to computer users”



“Acceptance of hardware ideas requires acceptance by software people; therefore hardware people should learn about software. And if software people want good machines, they must learn more about hardware to be able to communicate with and thereby influence hardware engineers.”

- **「訳: ハード屋はソフトを学べ、ソフト屋はハードを学べ」**