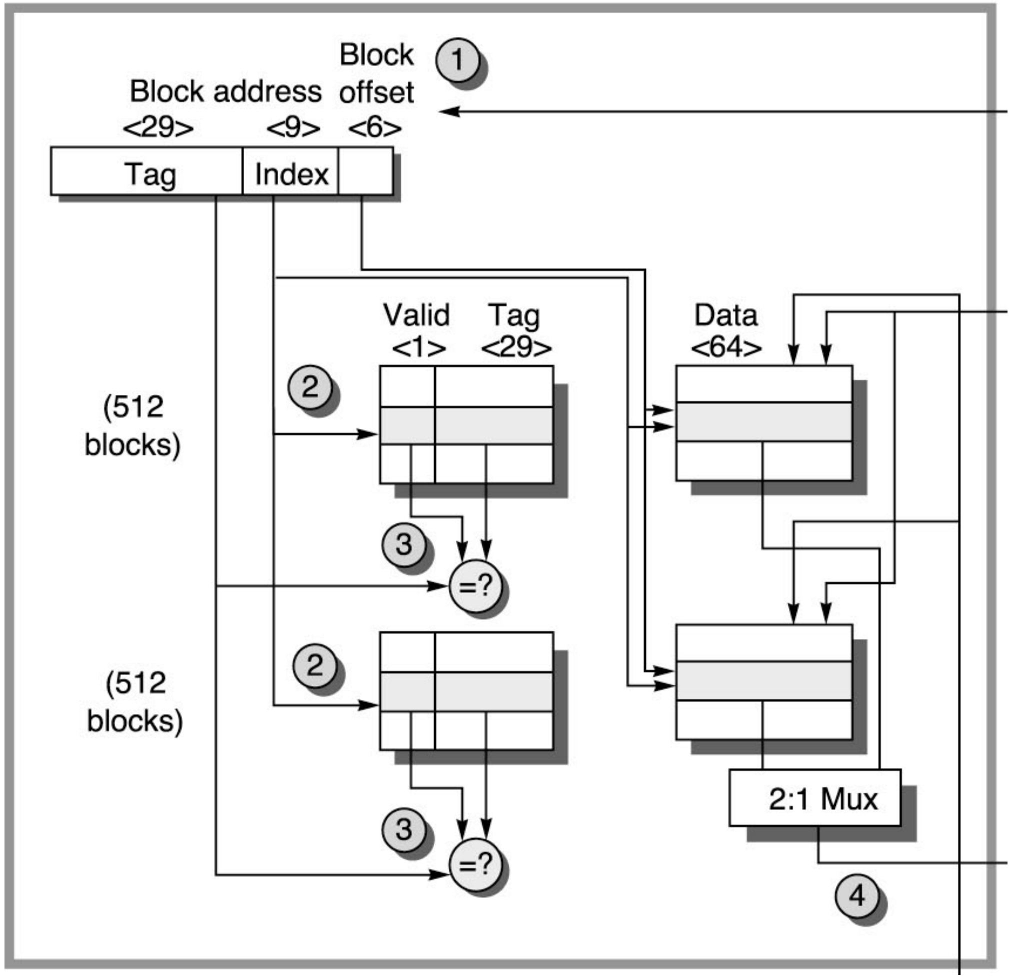
ΕΠΛ605:ΠροχωρήμενηΑρχιτεκτονική Υπολογιστών

Γιάννος Σαζεϊδης

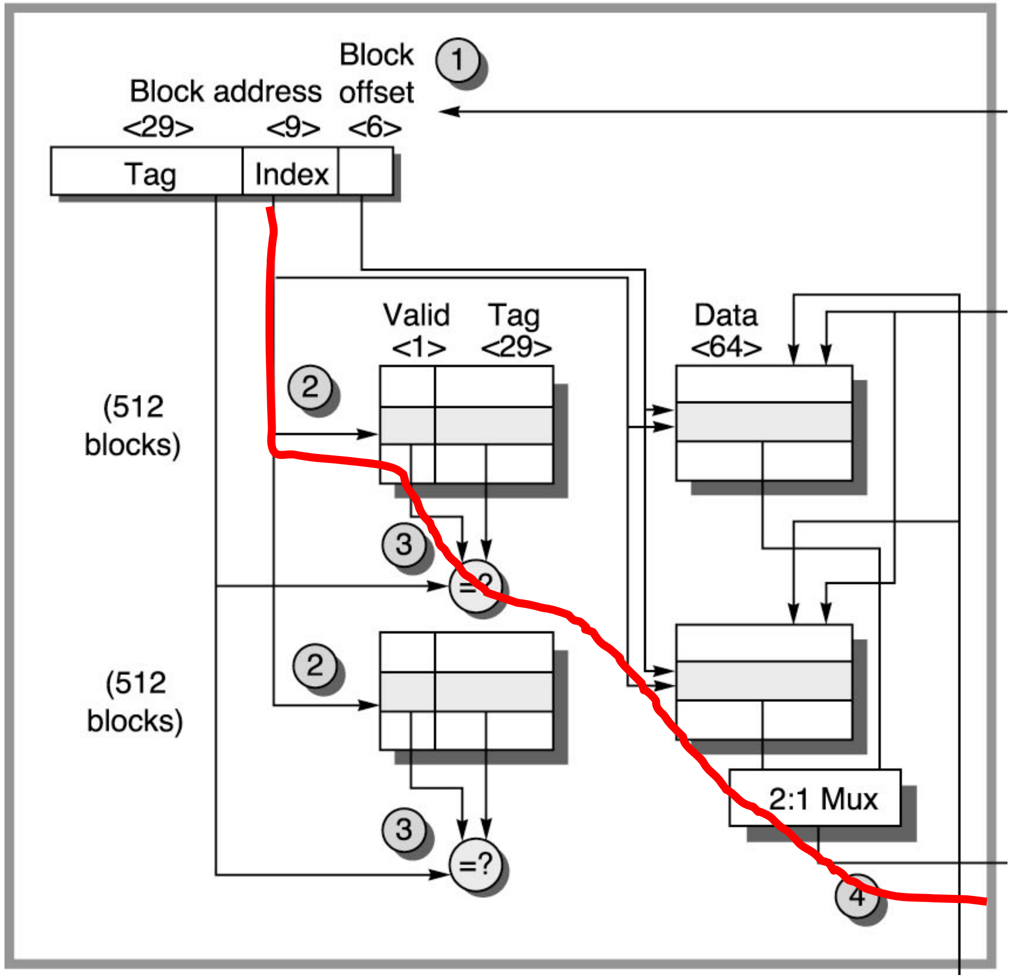
Ιεραρχίας Μνήμης (Memory Hierarchy)
Memory Optimizations

Εαρινό Εξάμηνο 2017

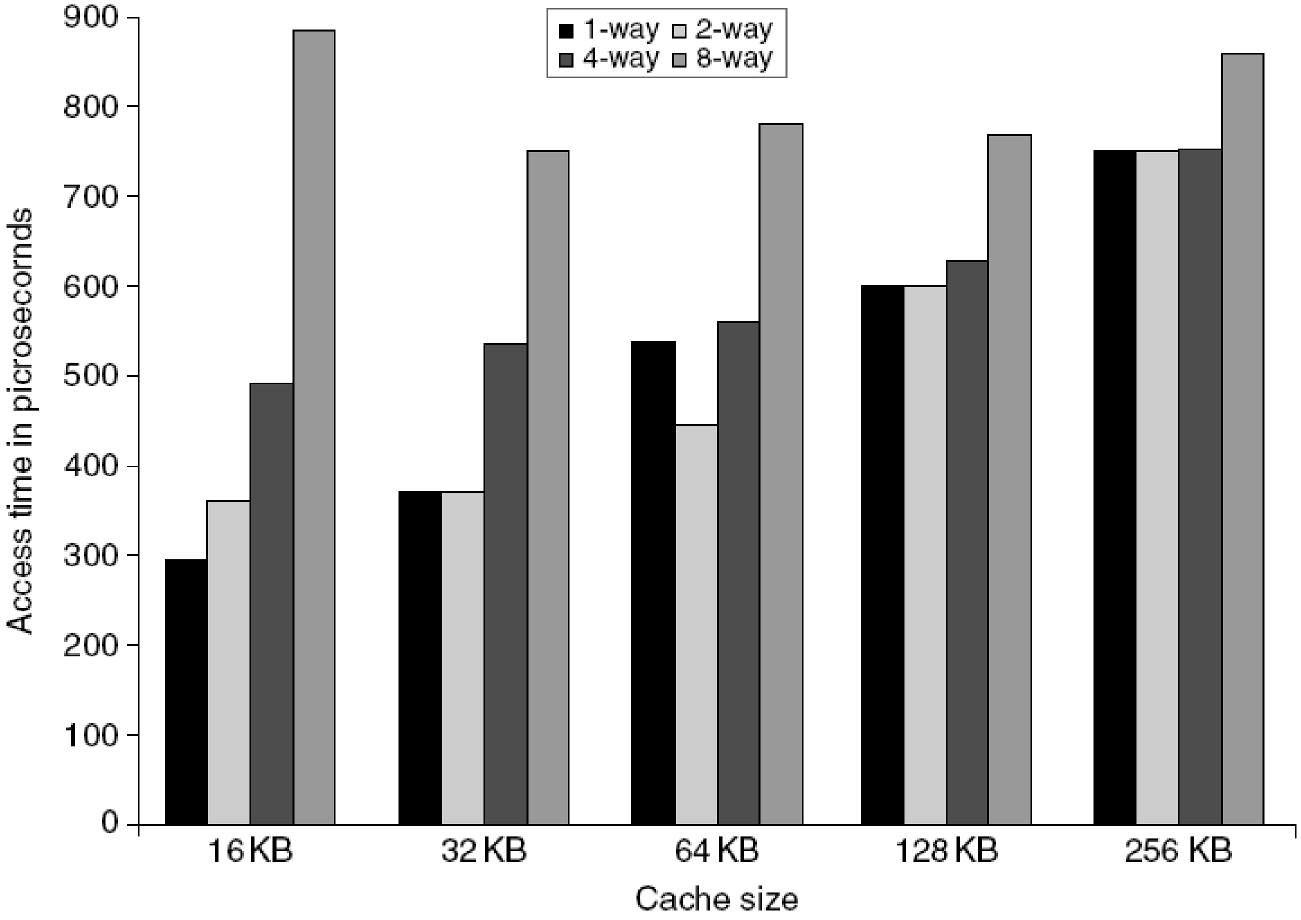
Critical Path?



Critical Path

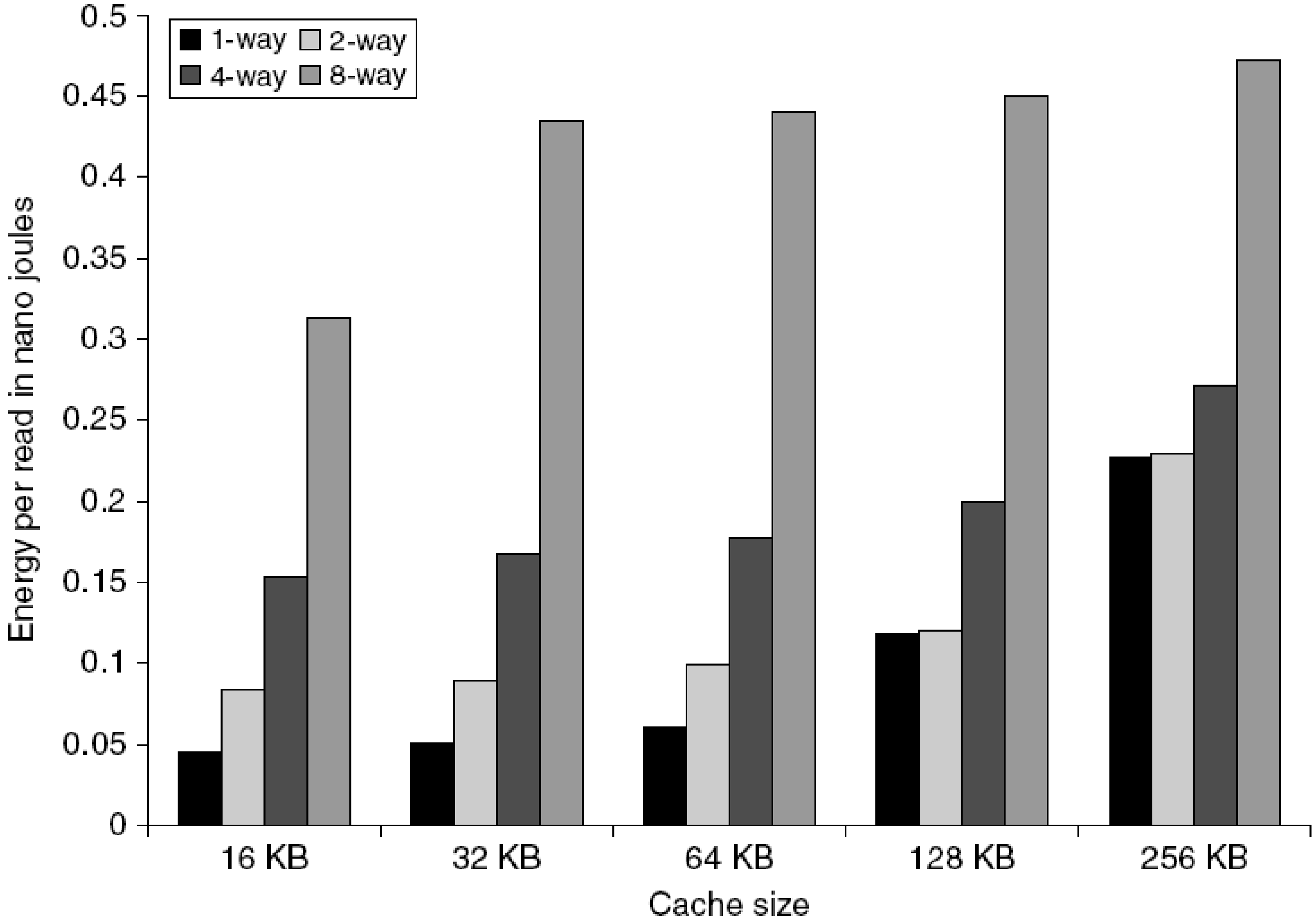


L1 Size and Associativity



- Access time vs. size and associativity

L1 Size and Associativity



- Energy per read vs. size and associativity

Small and simple first level caches

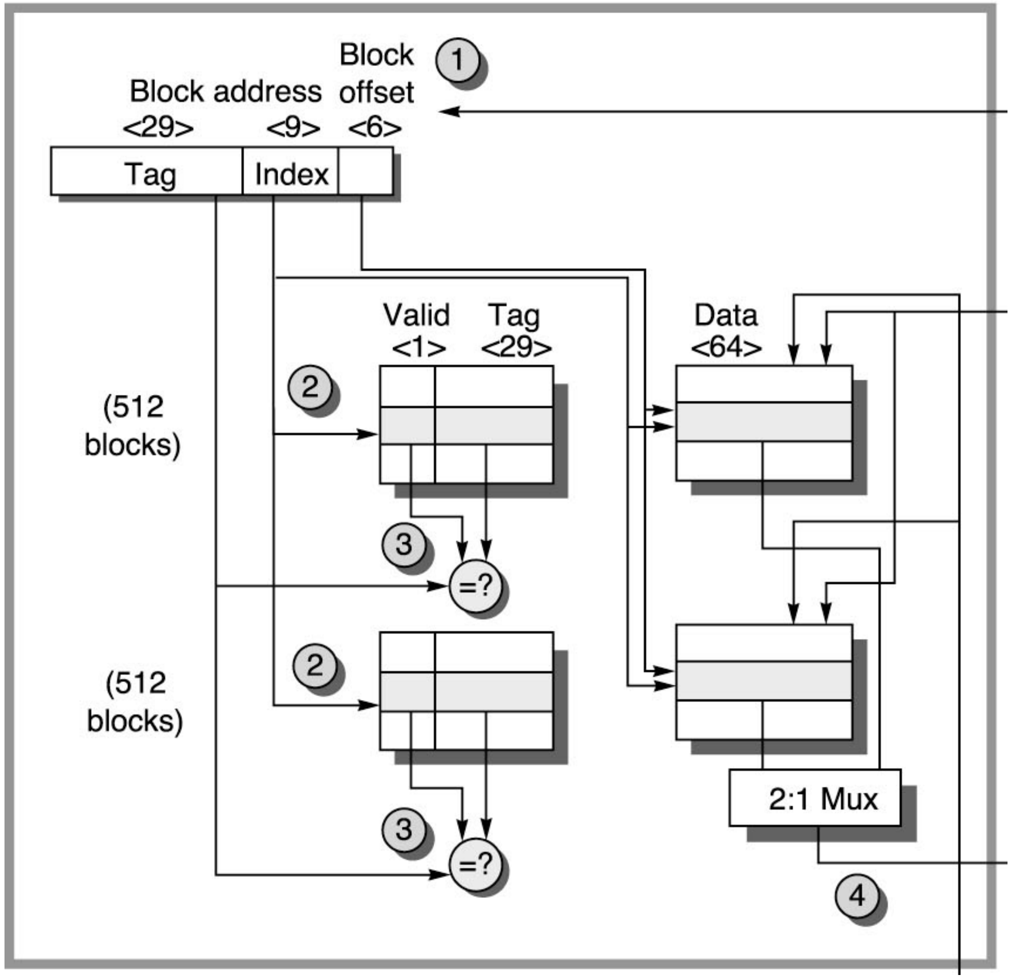
- Critical timing path:
  - addressing tag memory, then
  - comparing tags, then
  - selecting correct set of data
- Direct-mapped caches can overlap tag compare and transmission of data
- Lower associativity reduces power because fewer cache lines are accessed

Advanced Optimizations

Way Prediction

- To improve hit time, predict the way to pre-set mux
  - Mis-prediction gives longer hit time
  - Prediction accuracy
    - > 90% for two-way
    - > 80% for four-way
    - I-cache has better accuracy than D-cache
  - First used on MIPS R10000 in mid-90s
  - Used on ARM Cortex-A8
- Extend to predict block as well
  - "Way selection"
  - Increases mis-prediction penalty

Way Prediction-Way Selection



Pipelining Cache

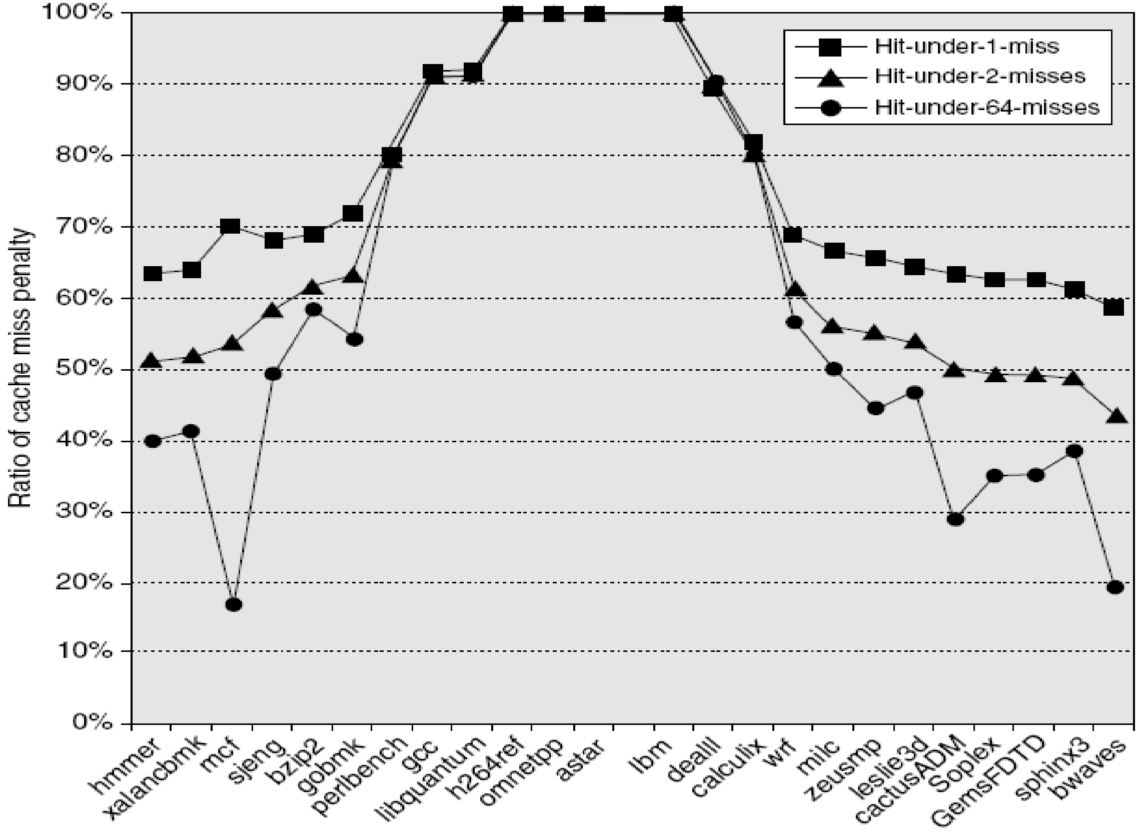
- Pipeline cache access to improve bandwidth
  - Examples:
    - Pentium: 1 cycle
    - Pentium Pro – Pentium III: 2 cycles
    - Pentium 4 – Core i7: 4 cycles

- Makes it easier to increase associativity
- Increases branch mis-prediction penalty

Advanced Optimizations

Nonblocking Caches

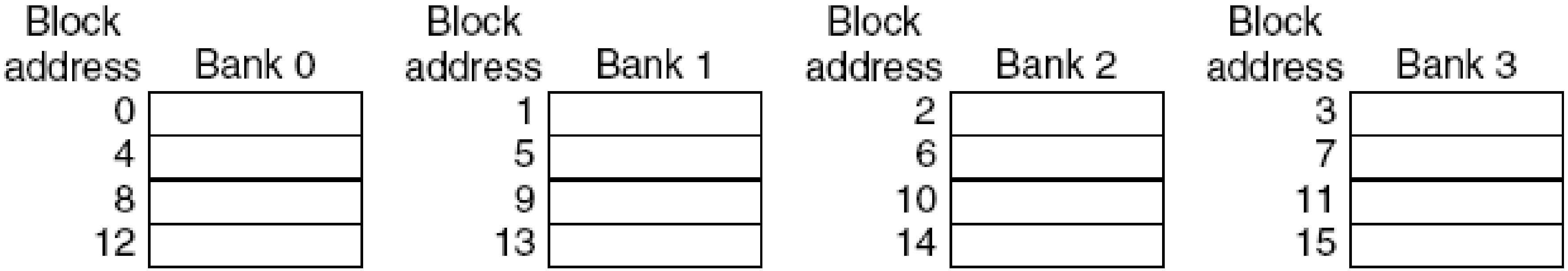
- Allow hits before previous misses complete
  - “Hit under miss”
  - “Hit under multiple miss”
- L2 must support this
- In general, processors can hide L1 miss penalty but not L2 miss penalty
- Rely on MSHRs
- (miss handle registers)



Multibanked Caches

- Organize cache as independent banks to support simultaneous access
  - ARM Cortex-A8 supports 1-4 banks for L2
  - Intel i7 supports 4 banks for L1 and 8 banks for L2

- Interleave banks according to block address



Block address	Bank 0	Block address	Bank 1	Block address	Bank 2	Block address	Bank 3
0		1		2		3	
4		5		6		7	
8		9		10		11	
12		13		14		15	

Advanced Optimizations

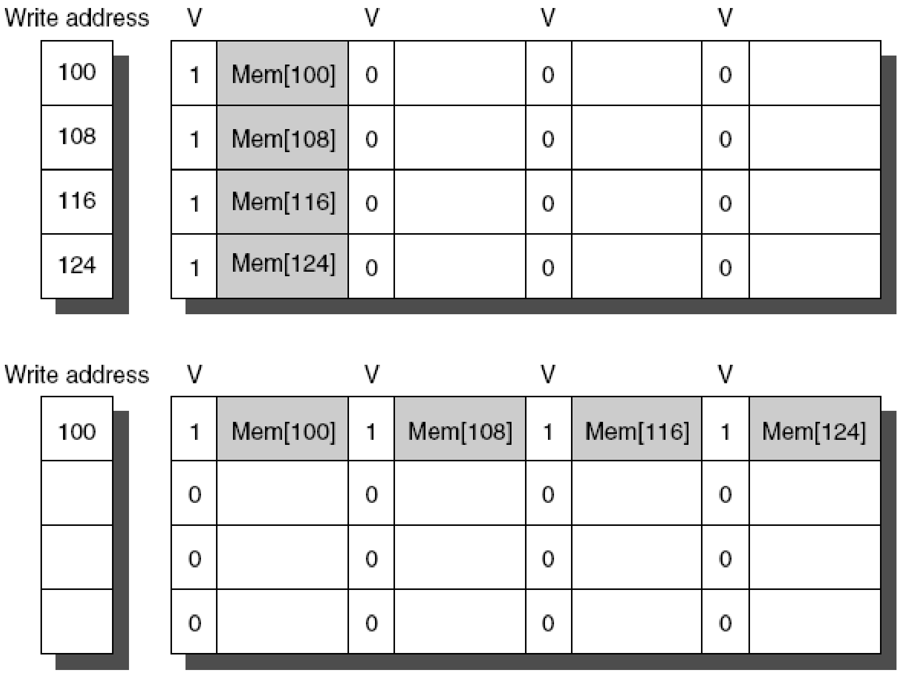
Critical Word First, Early Restart

- Critical word first
  - Request missed word from memory first
  - Send it to the processor as soon as it arrives
- Early restart
  - Request words in normal order
  - Send missed word to the processor as soon as it arrives

- Effectiveness of these strategies depends on block size and likelihood of another access to the portion of the block that has not yet been fetched

Merging Write Buffer

- When storing to a block that is already pending in the write buffer, update write buffer
- Reduces stalls due to full write buffer



Write address	V		V		V		V	
100	1	Mem[100]	0		0		0	
108	1	Mem[108]	0		0		0	
116	1	Mem[116]	0		0		0	
124	1	Mem[124]	0		0		0	

•No write buffering

Write address	V		V		V		V	
100	1	Mem[100]	1	Mem[108]	1	Mem[116]	1	Mem[124]
	0		0		0		0	
	0		0		0		0	
	0		0		0		0	

•Write buffering

Compiler Optimizations

- Loop Interchange
  - Swap nested loops to access memory in sequential order
- Blocking
  - Instead of accessing entire rows or columns, subdivide matrices into blocks
  - May require more memory accesses but improves locality of accesses

Loop blocking: temporal locality

- Poor code

```
for (k=0; k<NUM_ITERATIONS; k++)
   for (i=0; i<NUM_ELEMS; i++)
      X[i] = f(X[i]);
```

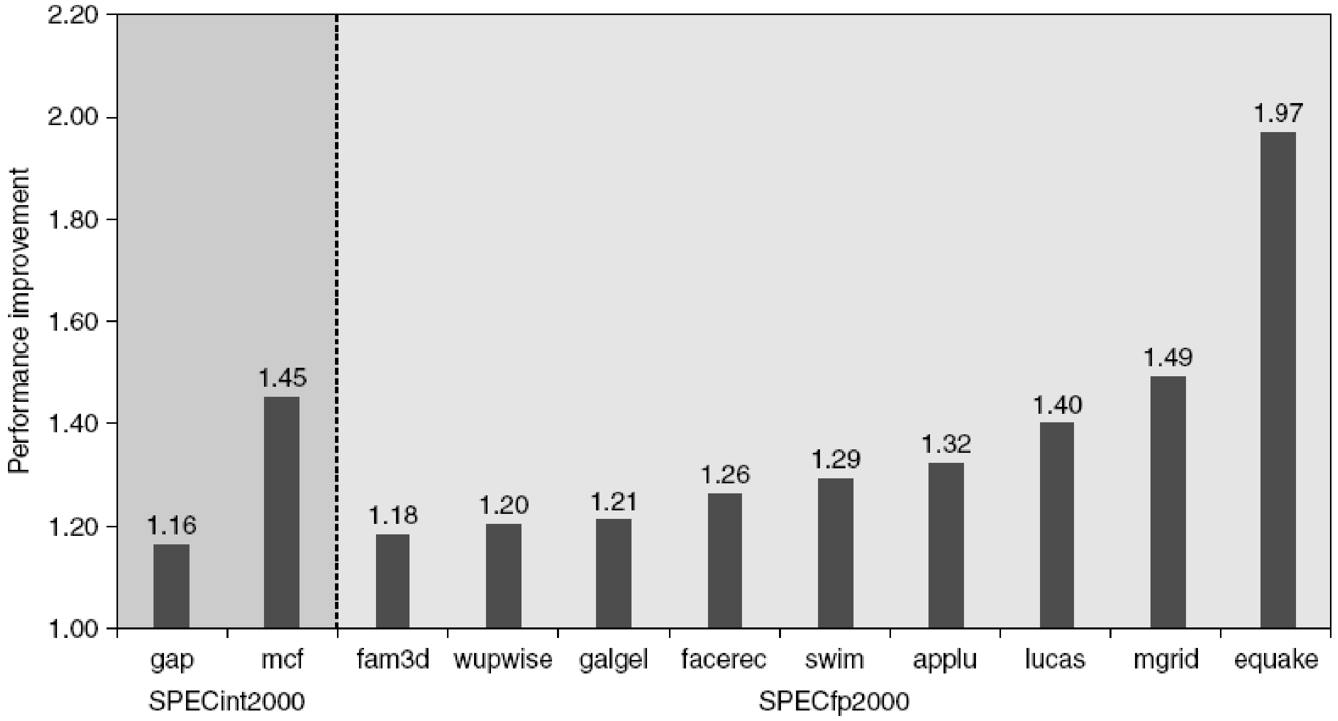
- Better code
- Cut array into CACHE_SIZE chunks
- Run all phases on one chunk, proceed to next

```
for (i=0; i<NUM_ELEMS; i+=CACHE_SIZE)
   for (k=0; k<NUM_ITERATIONS; k++)
      for (j=0; j<CACHE_SIZE; j++)
         X[i+j] = f(X[i+j]);
```

– Assumes you know **CACHE_SIZE, do you?**

Hardware Prefetching

- Fetch two blocks on miss (include next sequential block)



- Pentium 4 Pre-fetching
- Stream prefetchers: 8

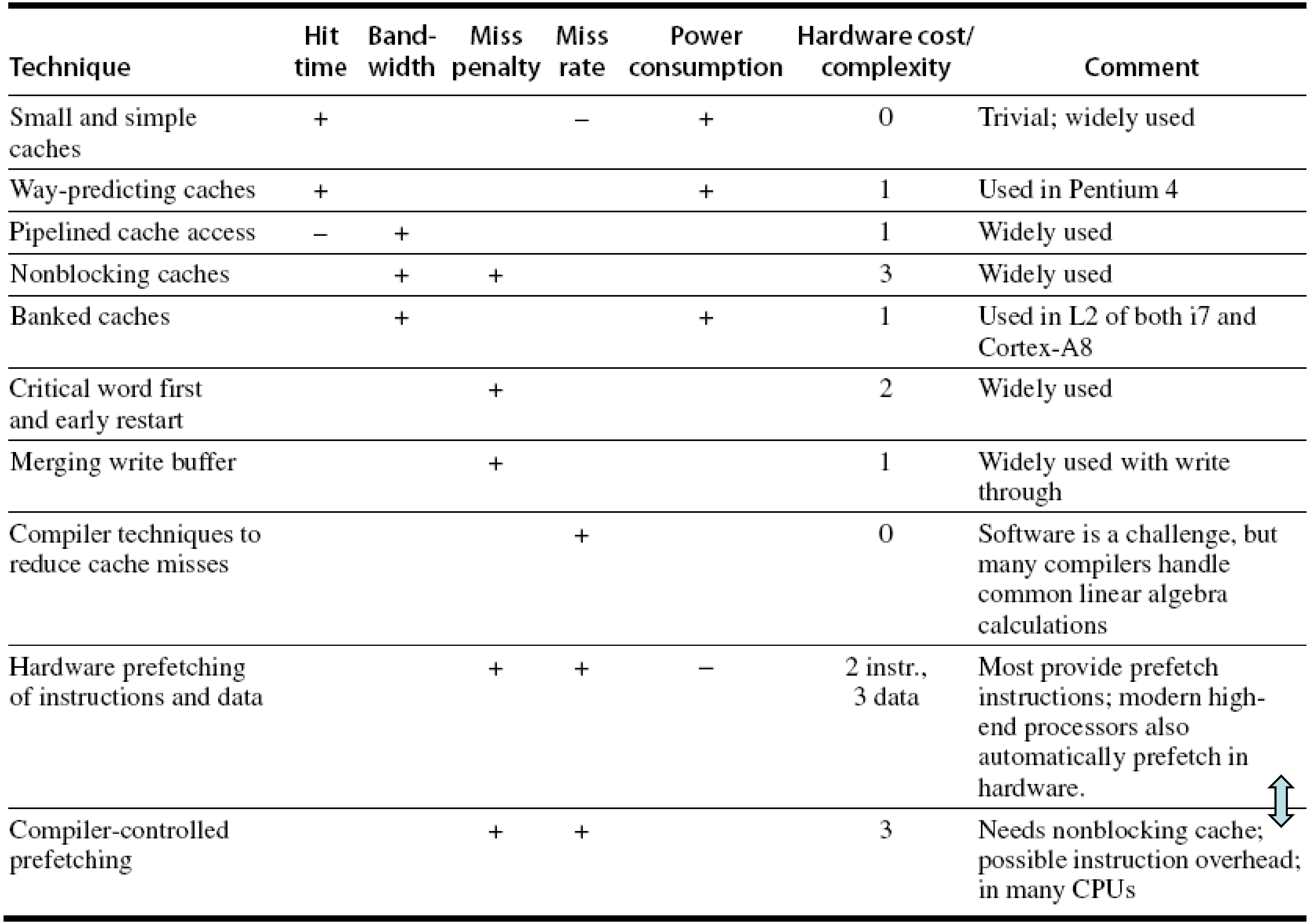
Compiler Prefetching

- Insert prefetch instructions before data is needed
- Non-faulting: prefetch doesn’t cause exceptions

- Combine with loop unrolling and other sw optimizations

Advanced Optimizations

Summary



Technique	Hit time	Band-width	Miss penalty	Miss rate	Power consumption	Hardware cost/ complexity	Comment
Small and simple caches	+			–	+	0	Trivial; widely used
Way-predicting caches	+				+	1	Used in Pentium 4
Pipelined cache access	–	+				1	Widely used
Nonblocking caches		+	+			3	Widely used
Banked caches		+			+	1	Used in L2 of both i7 and Cortex-A8
Critical word first and early restart			+			2	Widely used
Merging write buffer			+			1	Widely used with write through
Compiler techniques to reduce cache misses				+		0	Software is a challenge, but many compilers handle common linear algebra calculations
Hardware prefetching of instructions and data			+	+	–	2 instr., 3 data	Most provide prefetch instructions; modern high-end processors also automatically prefetch in hardware.
Compiler-controlled prefetching			+	+		3	Needs nonblocking cache; possible instruction overhead; in many CPUs

Advanced Optimizations

Memory Technology

- Performance metrics
  - Bandwidth (bytes/s)
  - Access time
    - Time between read request and when desired word arrives
  - *Memory Cycle time*
    - Minimum time between unrelated requests to memory
- DRAM used for main memory, SRAM used for cache

Memory Technology

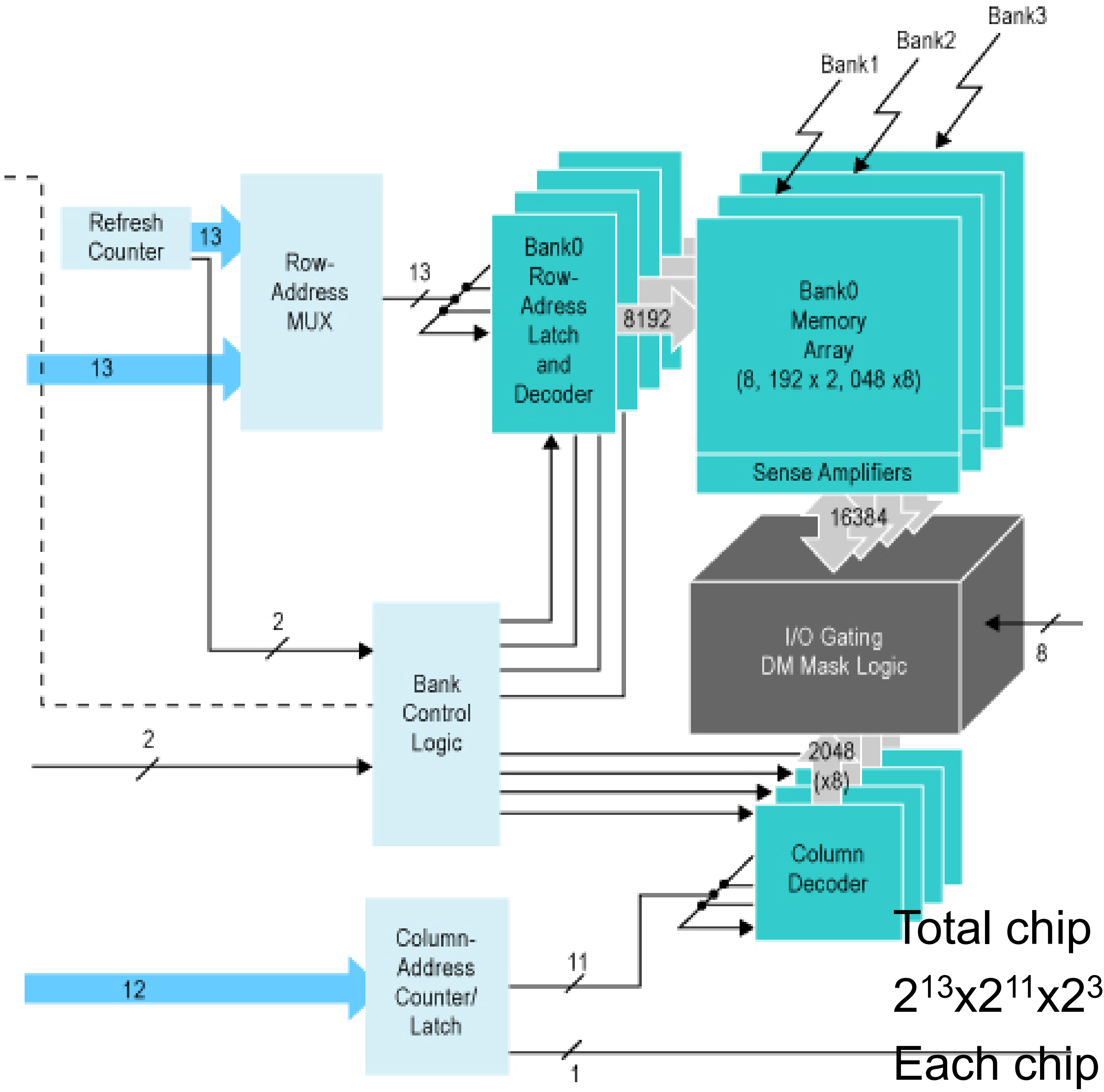
- SRAM
  - Requires low power to retain bit
  - Requires 6 transistors/bit
    - Some processors used 8 to be able to operate at lower V
- DRAM
  - Must be re-written after being read (reads destructive)
  - Must also be periodically refreshed
    - Every ~ 32-64 ms
    - Each row can be refreshed simultaneously
  - One transistor/bit
  - Address lines are multiplexed:
    - Upper half of address: row access strobe (RAS)
    - Lower half of address: column access strobe (CAS)

Memory Technology

- Amdahl:
  - Memory capacity should grow linearly with processor speed
  - Unfortunately, memory capacity and speed has not kept pace with processors

- Some optimizations:
  - An access opens a whole row
  - Multiple accesses to same row
  - Synchronous DRAM
    - Added clock to DRAM interface
    - Burst mode with critical word first
  - Wider interfaces
  - Double data rate (DDR)
  - Multiple banks on each DRAM device

DRAM Device Structure



Total chip size

2^{13}x2^{11}x2^{3}x4

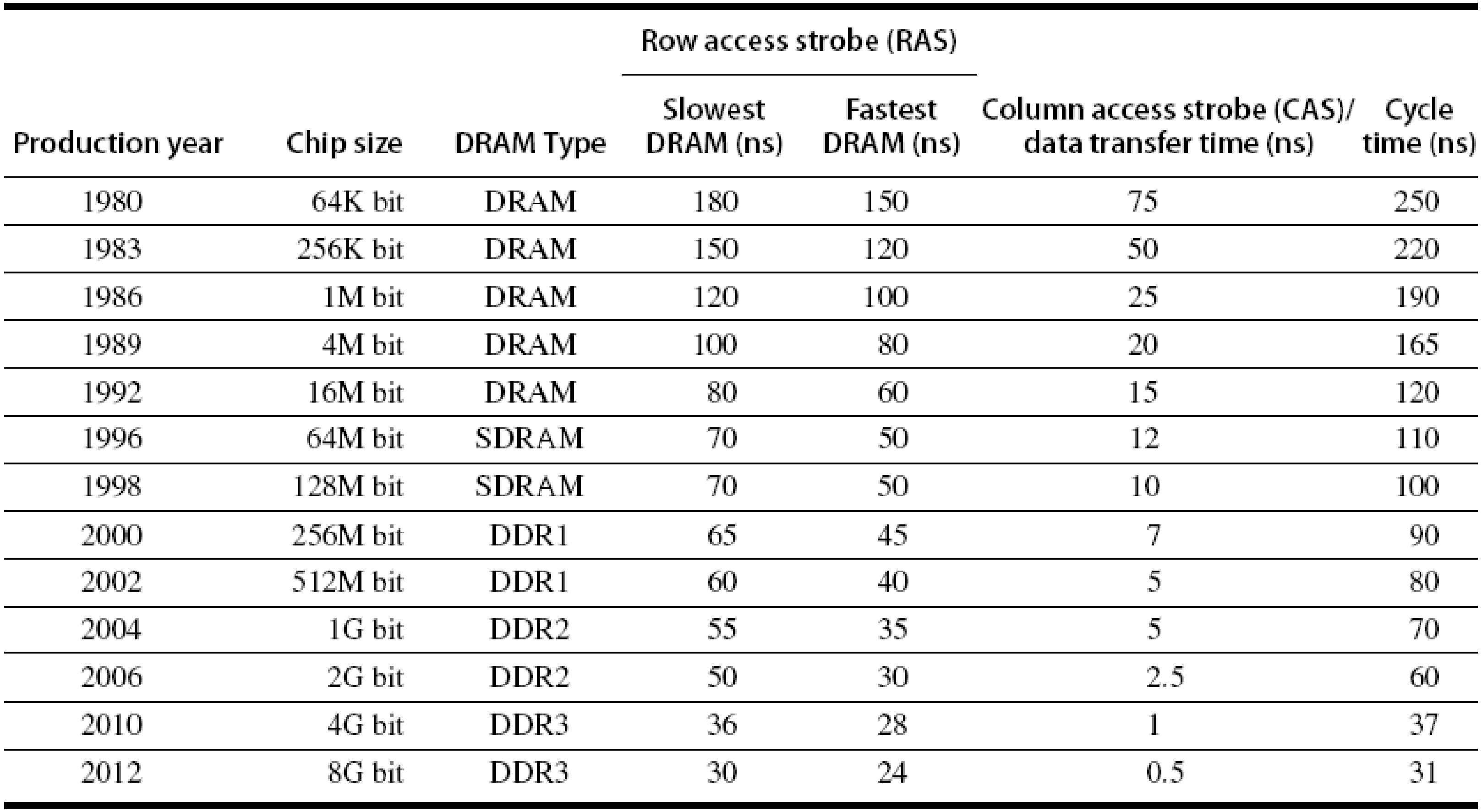
Each chip contributes 4 or 8 bits

DIMM typically 16x4, 8x8

Memory Optimizations

- Bank Level Parallelism
- When a row is accessed saved in a row-buffer (row = memory page)
- Memory controller can exploit row-buffer locality by scheduling memory requests to save memory page
  - Goal to improve over simply first come first served

Memory Optimizations

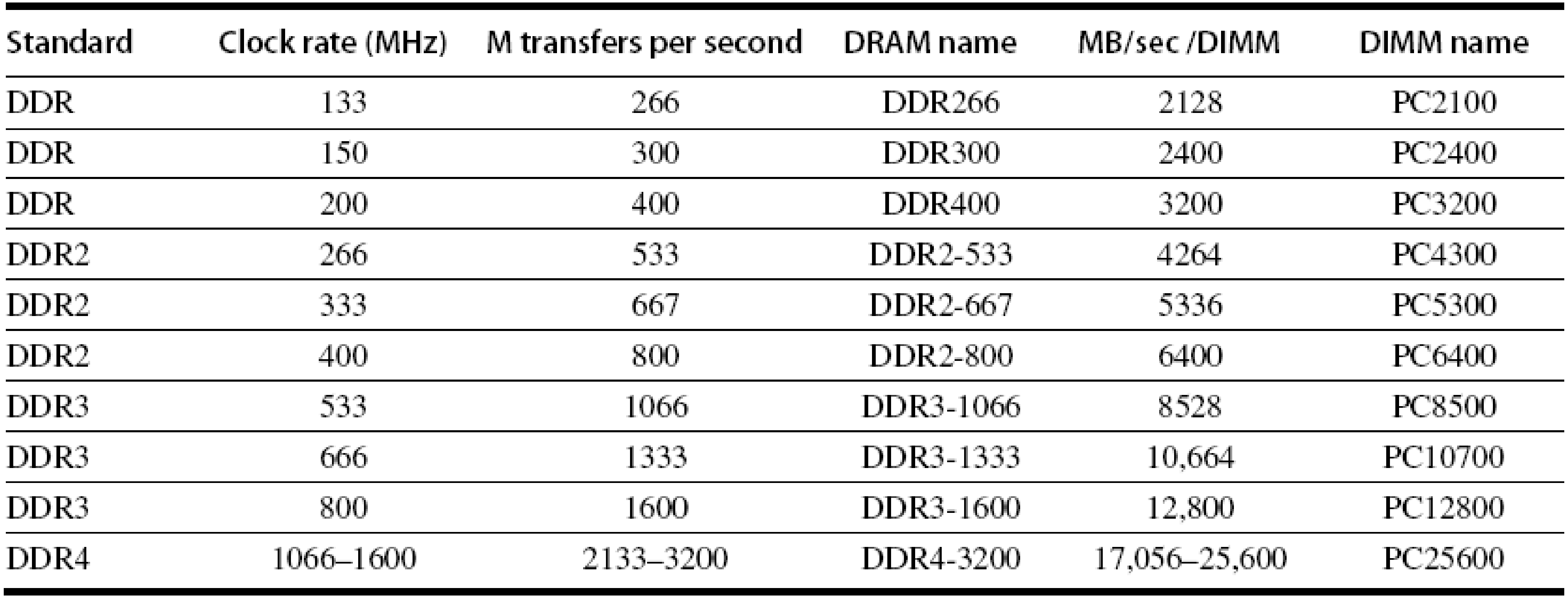


			Row access strobe (RAS)			
Production year	Chip size	DRAM Type	Slowest DRAM (ns)	Fastest DRAM (ns)	Column access strobe (CAS)/ data transfer time (ns)	Cycle time (ns)
1980	64K bit	DRAM	180	150	75	250
1983	256K bit	DRAM	150	120	50	220
1986	1M bit	DRAM	120	100	25	190
1989	4M bit	DRAM	100	80	20	165
1992	16M bit	DRAM	80	60	15	120
1996	64M bit	SDRAM	70	50	12	110
1998	128M bit	SDRAM	70	50	10	100
2000	256M bit	DDR1	65	45	7	90
2002	512M bit	DDR1	60	40	5	80
2004	1G bit	DDR2	55	35	5	70
2006	2G bit	DDR2	50	30	2.5	60
2010	4G bit	DDR3	36	28	1	37
2012	8G bit	DDR3	30	24	0.5	31

Memory Technology

Memory Technology

Memory Optimizations



Standard	Clock rate (MHz)	M transfers per second	DRAM name	MB/sec /DIMM	DIMM name
DDR	133	266	DDR266	2128	PC2100
DDR	150	300	DDR300	2400	PC2400
DDR	200	400	DDR400	3200	PC3200
DDR2	266	533	DDR2-533	4264	PC4300
DDR2	333	667	DDR2-667	5336	PC5300
DDR2	400	800	DDR2-800	6400	PC6400
DDR3	533	1066	DDR3-1066	8528	PC8500
DDR3	666	1333	DDR3-1333	10,664	PC10700
DDR3	800	1600	DDR3-1600	12,800	PC12800
DDR4	1066–1600	2133–3200	DDR4-3200	17,056–25,600	PC25600

Memory Optimizations

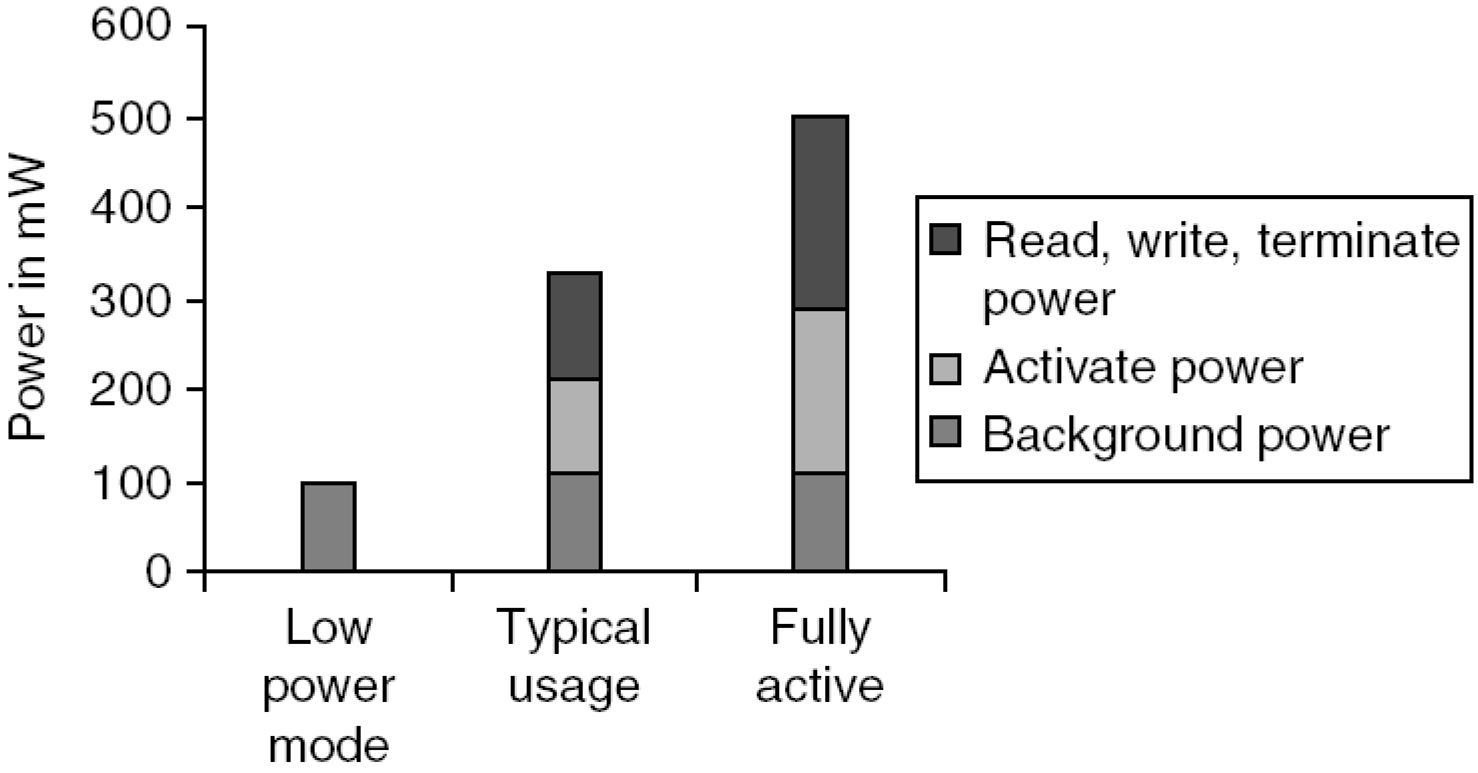
- DDR:
  - DDR2
    - Lower power (2.5 V -> 1.8 V)
    - Higher clock rates (266 MHz, 333 MHz, 400 MHz)
  - DDR3
    - 1.5 V
    - 800 MHz
  - DDR4
    - 1-1.2 V
    - 1600 MHz
- GDDR5 is graphics memory based on DDR3

Memory Optimizations

- Graphics memory:
  - Achieve 2-5 X bandwidth per DRAM vs. DDR3
    - Wider interfaces (32 vs. 16 bit)
    - Higher clock rate
      - Possible because they are attached via soldering instead of socketted DIMM modules

- Reducing power in SDRAMs:
  - Lower voltage
  - Low power mode (ignores clock, continues to refresh)

Memory Power Consumption



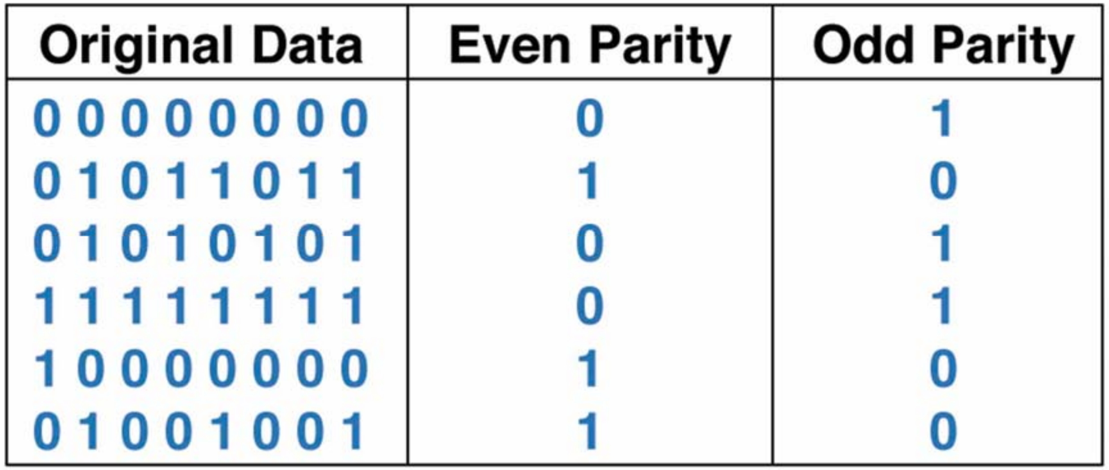
Flash Memory

- Type of EEPROM
- Must be erased (in blocks) before being overwritten
- Non volatile
- Limited number of write cycles (wearout 100K)
- Cheaper than SDRAM, more expensive than disk
- Slower than SRAM, faster than disk

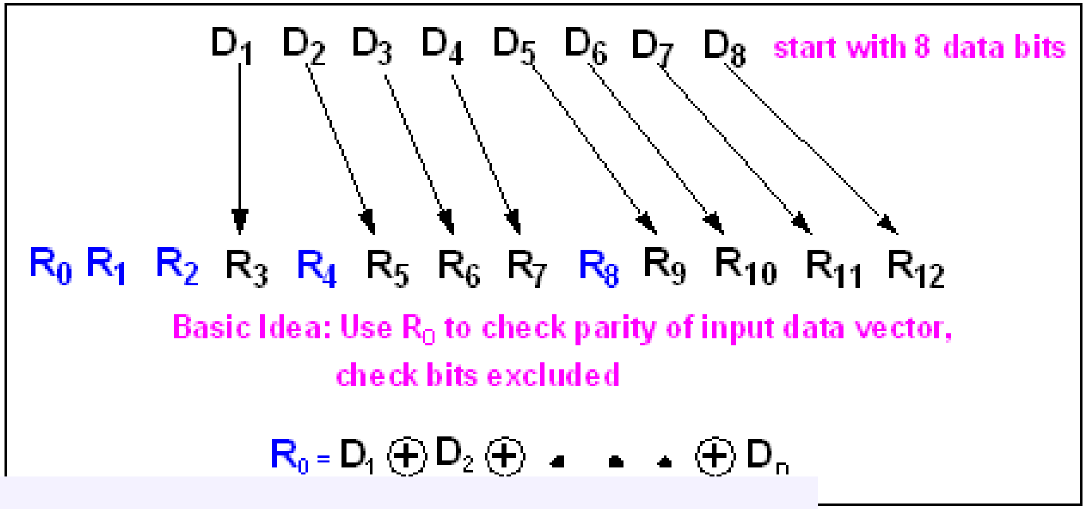
Memory Dependability

- Memory is susceptible to cosmic rays
- *Soft errors*: dynamic errors
  - Detected and fixed by error correcting codes (ECC)
- *Hard errors*: permanent errors
  - Use spare rows to replace defective rows
- Chipkill: a RAID-like error recovery technique
  - Can afford loosing a single device

- Parity
- SECDED ECC
- RAID



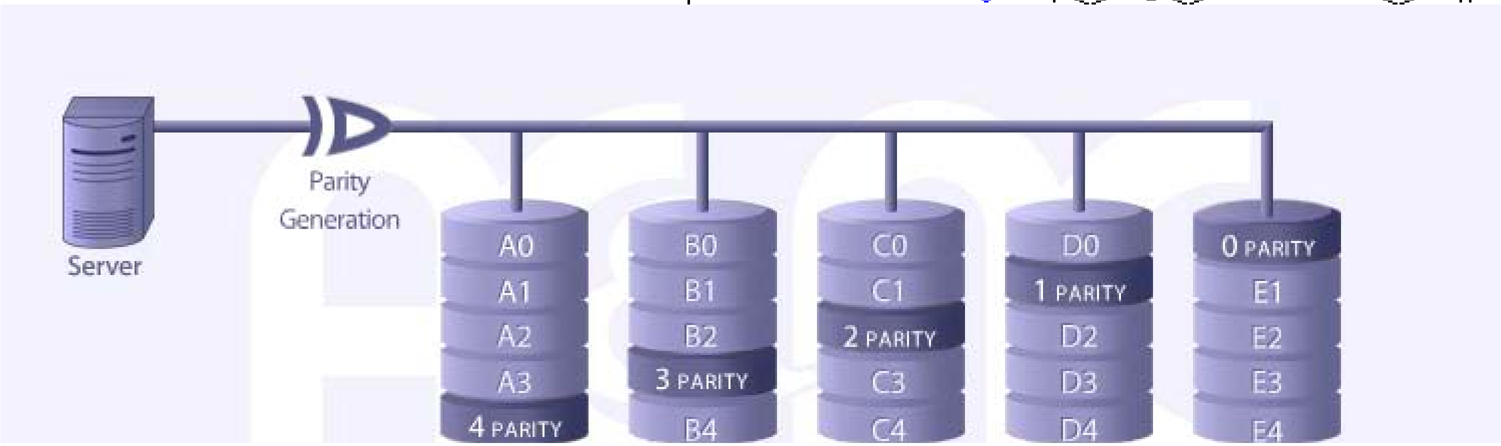
Original Data	Even Parity	Odd Parity
0 0 0 0 0 0 0 0	0	1
0 1 0 1 1 0 1 1	1	0
0 1 0 1 0 1 0 1	0	1
1 1 1 1 1 1 1 1	0	1
1 0 0 0 0 0 0 0	1	0
0 1 0 0 1 0 0 1	1	0


D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8
R_0 R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9 R_{10} R_{11} R_{12}

start with 8 data bits

Basic Idea: Use R_0 to check parity of input data vector, check bits excluded

$$R_0 = D_1 \oplus D_2 \oplus \ldots \oplus D_n$$



Virtual Memory and Virtual Machines

Virtual Memory

- Protection via virtual memory
  - Keeps processes in their own memory space

- Role of architecture:
  - Provide user mode and supervisor mode
  - Protect certain aspects of CPU state
  - Provide mechanisms for switching between user mode and supervisor mode
  - Provide mechanisms to limit memory accesses
    - Fields in Page table and TLB
  - Provide TLB to translate addresses

Virtual Machines

- Supports isolation and security
- Sharing a computer among many unrelated users
- Enabled by raw speed of processors, making the overhead more acceptable

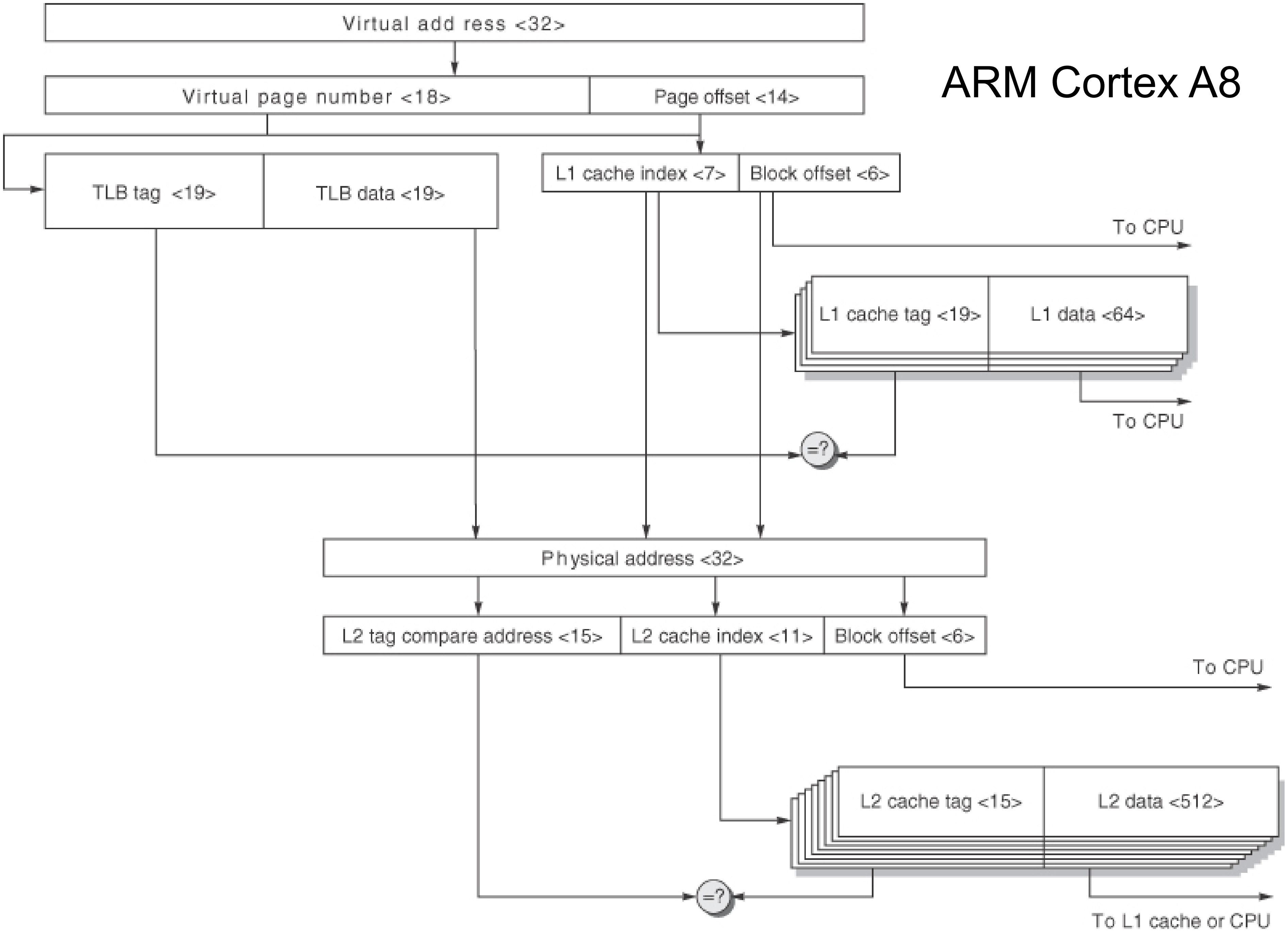
- Allows different ISAs and operating systems to be presented to user programs
  - "System Virtual Machines"
  - SVM software is called "virtual machine monitor" or "hypervisor"
  - Individual virtual machines run under the monitor are called "guest VMs"
    - Gust OS running guest applications

Impact of VMs on Virtual Memory

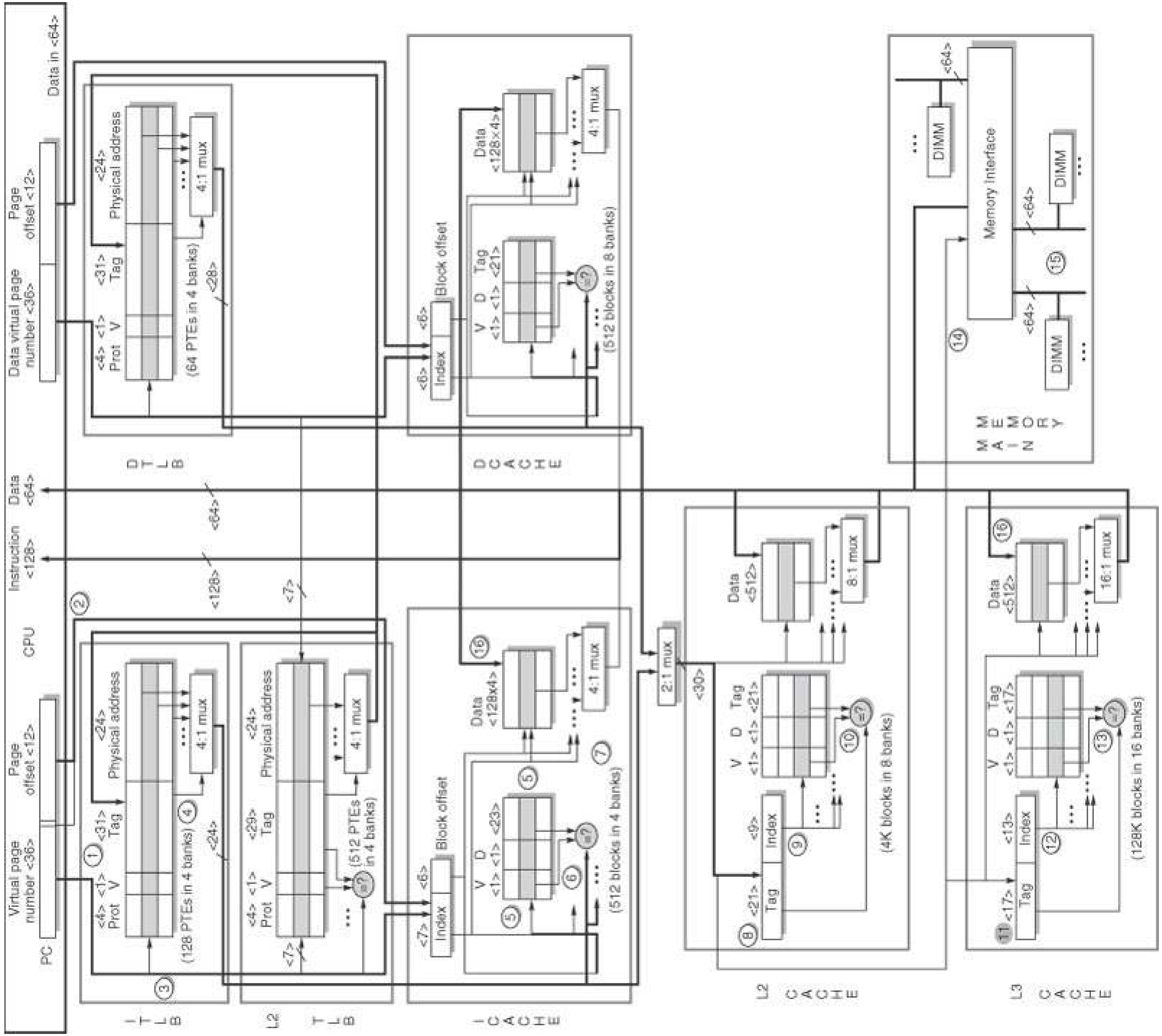
- Each guest OS maintains its own set of page tables
  - VMM adds a level of memory between physical and virtual memory called “real memory”
  - VMM maintains shadow page table that maps guest virtual addresses to physical addresses
    - Requires VMM to detect guest’s changes to its own page table
    - Occurs naturally if accessing the page table pointer is a privileged operation

- Architectural support (ISA changes)
  - Allow direct virtual to physical mapping

ARM Cortex A8



Intel i7



CPU
PC
Virtual page number <36>
Page offset <12>
Data virtual page number <36>
Page offset <12>
Data in <64>
Instruction <128>
Data <64>
<128>
<64>
<7>
I TLB
<4> Prot
<1> V
<31> Tag
<24> Physical address
4:1 mux
(128 PTEs in 4 banks)
<24>
L2 TLB
<4> Prot
<1> V
<29> Tag
<24> Physical address
4:1 mux
(512 PTEs in 4 banks)
=?
I CACHE
<7> Index
<6> Block offset
V <1>
D <1>
<23>
Data <128x4>
4:1 mux
=?
(512 blocks in 4 banks)
D TLB
<4> Prot
<1> V
<31> Tag
<24> Physical address
4:1 mux
(64 PTEs in 4 banks)
<28>
D CACHE
<6> Index
<6> Block offset
V <1>
D <1>
Tag <21>
Data <128×4>
4:1 mux
=?
(512 blocks in 8 banks)
2:1 mux
<30>
L2 CACHE
<21> Tag
<9> Index
V <1>
D <1>
Tag <21>
Data <512>
8:1 mux
=?
(4K blocks in 8 banks)
L3 CACHE
<17> Tag
<13> Index
V <1>
D <1>
Tag <17>
Data <512>
16:1 mux
=?
(128K blocks in 16 banks)
MAIN MEMORY
Memory Interface
DIMM
DIMM
DIMM
<64>
<64>
<64>
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16