

### Chapter 2: Memory Hierarchy Design (Part 3)

Introduction  
 Caches  
 Main Memory (Section 2.2)  
 Virtual Memory (Section 2.4, Appendix B.4, B.5)

### Memory Technologies

Dynamic Random Access Memory (DRAM)  
 Optimized for density, not speed  
 One transistor cells  
 Multiplexed address pins  
 Row Address Strobe (RAS)  
 Column Address Strobe (CAS)  
 Cycle time > access time  
 Destructive reads  
 Must refresh every few ms  
 Access every row  
 Sold as dual inline memory modules (DIMMs)

### Memory Technologies, cont.

Static Random Access Memory (SRAM)  
 Optimized for speed, then density  
 Typically 6 transistors per cell  
 Separate address pins  
 Static  $\Rightarrow$  No Refresh  
 Greater power dissipation than DRAM  
 Access time = cycle time

### DRAM Organization

DIMM  
 Rank  
 Bank  
 Array  
 Row buffer

### DRAM Organization

Rank: chips needed to respond to a single request

Assume 64 bit data bus

For 8 bit DRAM, need 8 chips in a rank

For 4 bit DRAM, need 16 chips in a rank

Can have multiple ranks per DIMM

Bank: A chip is divided into multiple independent banks for **pipelined** access

Array: A bank consists of many arrays, 1 array per bit of output, for **parallel** access

Row buffer: A "**cache**" that preserves the last row read from a bank

### DRAM Organization

See figure 1.5 in

**The Memory System: You Can't Avoid It, You Can't Ignore It, You Can't Fake It**

**By Bruce Jacob**

**Synthesis Lectures on Computer Architecture, Morgan & Claypool**

**Series editor: Mark Hill**

Downloadable from U of I accounts

### Internals of a DRAM Array

See Figure 1.6 of the synthesis lecture

Steps to access a bit

Pre-charge bit lines

Activate row: turn on word line for the row, brings data to sense amps

Column read: send subset of data (columns)

(Restore data)

### DRAM Optimizations – Page Mode

Unoptimized DRAM

First read entire row

Then select column from row

Stores entire row in a buffer

Page Mode

Row buffer acts like an SRAM

By changing column address, random bits can be accessed *within a row*.

### DRAM Optimizations – Synchronous DRAM

Previously, DRAM had asynchronous interface  
 Each transfer involves handshaking with controller  
 Synchronous DRAM (SDRAM)  
   Clock added to interface  
   Register to hold number of bytes requested  
     Send multiple bytes per request  
 Double Data Rate (DDR)  
   Send data on rising and falling edge of clock

### Simple Main Memory

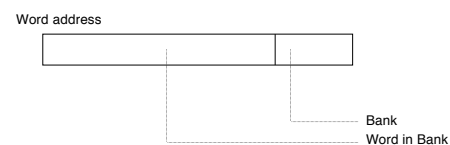
Consider a memory with these parameters:  
 1 cycle to send address  
 6 cycles to access each word  
 1 cycle to send word back to CPU/Cache  
 What's the miss penalty for a 4word block?  
 $(1 + 6 \text{ cycles} + 1 \text{ cycle}) \times 4 \text{ words}$   
 $= 32 \text{ cycles}$   
 How can we speed this up?

### Wider Main Memory

Make the memory wider  
   Read out 2 (or more) words in parallel  
 Memory parameters:  
   1 cycle to send address  
   6 cycles to access each *doubleword*  
   1 cycle to send doubleword back to CPU/Cache  
 Miss penalty for a 4 word block:  
 $(1 + 6 \text{ cycles} + 1 \text{ cycle}) \times 2 \text{ doublewords}$   
 $= 16 \text{ cycles}$   
 Cost  
   Wider bus  
   Larger expansion size

### Interleaved Main Memory

Organize memory in banks  
 Subsequent words map to different banks  
 Word A in bank  $(A \bmod M)$   
 Within a bank, word A in location  $(A \text{ div } M)$



How many banks to include?

Production year	Chip size	DRAM type	Best case access time (no precharge)			Precharge needed
			RAS time (ns)	CAS time (ns)	Total (ns)	Total (ns)
2000	256M bit	DDR1	21	21	42	63
2002	512M bit	DDR1	15	15	30	45
2004	1G bit	DDR2	15	15	30	45
2006	2G bit	DDR2	10	10	20	30
2010	4G bit	DDR3	13	13	26	39
2016	8G bit	DDR4	13	13	26	39

Figure 2.4 Capacity and access times for DDR SDRAMs by year of production. Access time is for a random memory word and assumes a new row must be opened. If the row is in a different bank, we assume the bank is precharged; if the row is not open, then a precharge is required, and the access time is longer. As the number of banks has increased, the ability to hide the precharge time has also increased. DDR4 SDRAMs were initially expected in 2014, but did not begin production until early 2016.

© 2019 Elsevier Inc. All rights reserved.

13

Standard	I/O clock rate	M transfers/s	DRAM name	MB/s/DIMM	DIMM name
DDR1	133	266	DDR366	2128	PC2100
DDR1	150	300	DDR400	2400	PC2400
DDR1	200	400	DDR600	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	1333	2666	DDR4-2666	21,300	PC21300

Figure 2.5 Clock rates, bandwidth, and names of DDR DRAMs and DIMMs in 2016. Note the numerical relationship between the columns. The third column is twice the second, and the fourth uses the number from the third column in the name of the DRAM chip. The fifth column is eight times the third column, and a rounded version of this number is used in the name of the DIMM. DDR4 saw significant first use in 2016.

© 2019 Elsevier Inc. All rights reserved.

14

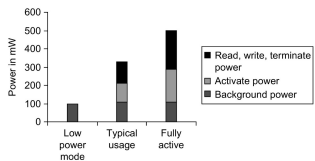


Figure 2.6 Power consumption for a DDR3 SDRAM operating under three conditions: low-power (shutdown) mode, typical system mode (DRAM is active 30% of the time for reads and 15% for writes), and fully active mode, where the DRAM is continuously reading or writing. Reads and writes assume bursts of eight transfers. These data are based on a Micron 1.5V 2GB DDR3-1066, although similar savings occur in DDR4 SDRAMs.

© 2019 Elsevier Inc. All rights reserved.

15

### Other Technologies

#### Graphics Data RAMS (GDDR)

Wider (32 bits), higher clock, connect directly to GPUs (soldered to board vs. DIMMs)

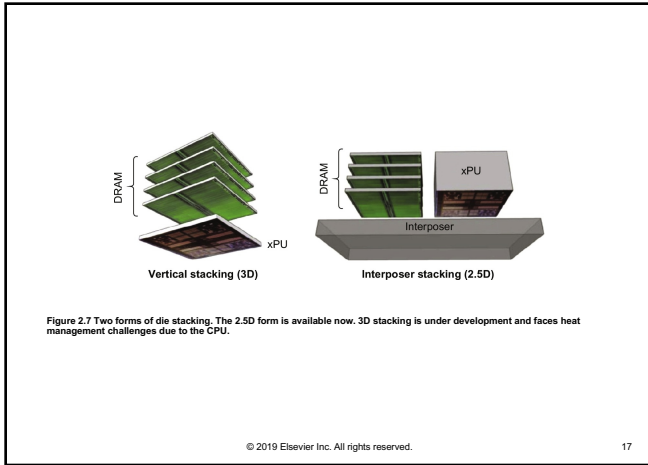
#### Die stacked DRAMs / 3D / High Bandwidth Memory (HBM)

#### Nonvolatile memory (later)

Flash

Phase change

Reliability: Parity, ECC, chipkill



### Virtual Memory

User operates in a virtual address space, mapping between virtual space and main memory is determined at runtime

**Original Motivation**

- Avoid overlays
- Use main memory as a cache for disk

**Current motivation**

- Relocation
- Protection
- Sharing
- Fast startup

Engineered differently than CPU caches

- Miss access time  $O(1,000,000)$
- Miss access time  $\gg$  miss transfer time

### Virtual Memory, cont.

Blocks, called *pages*, are 512 to 16K bytes.

**Page placement**

- Fully-associative -- avoid expensive misses

**Page identification**

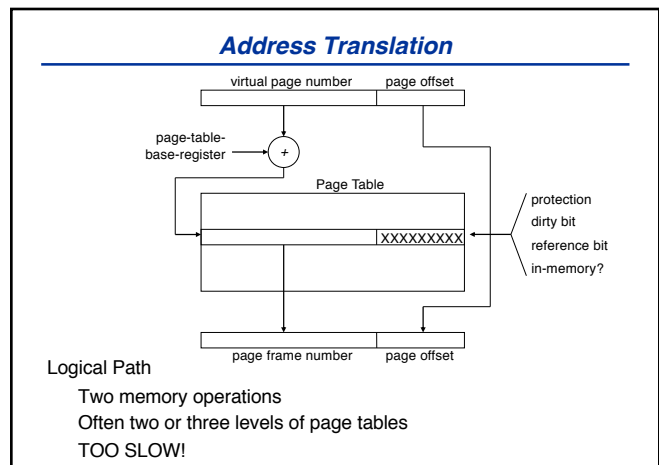
- Address translation -- virtual to physical address
- Indirection through one or two page tables
- Translation cached in translation buffer

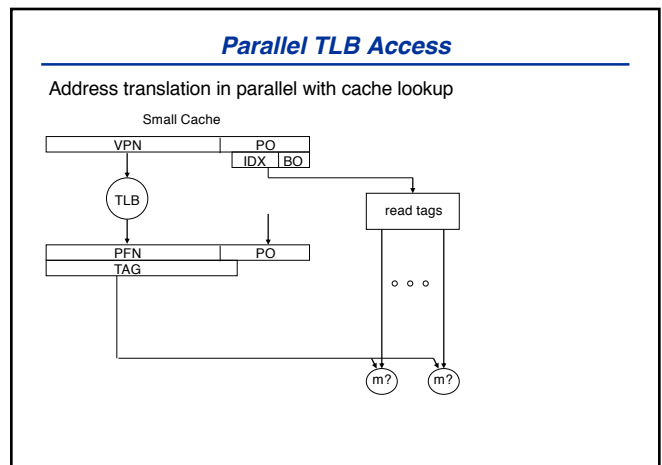
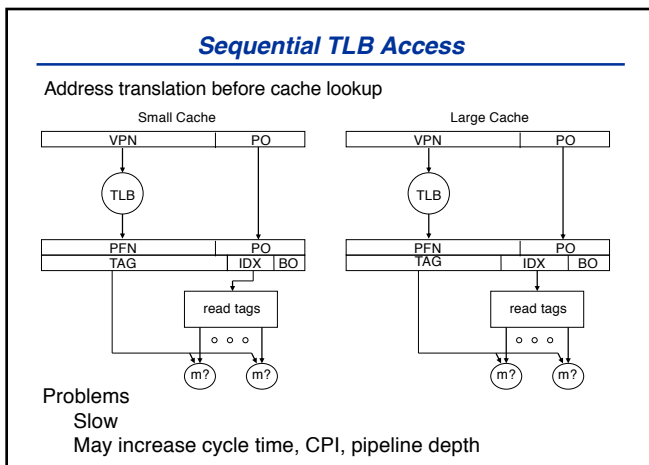
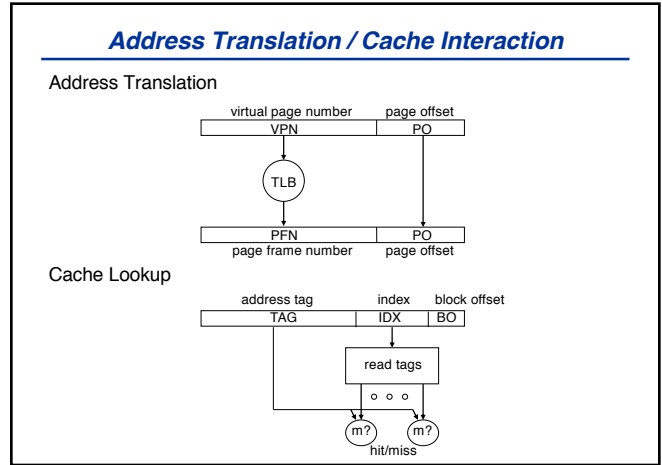
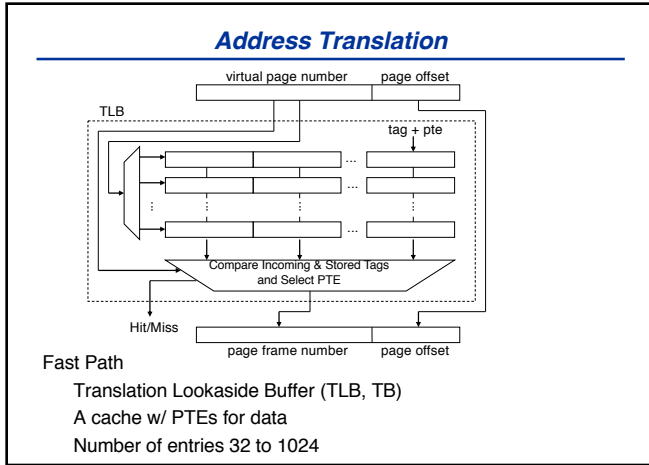
**Page replacement**

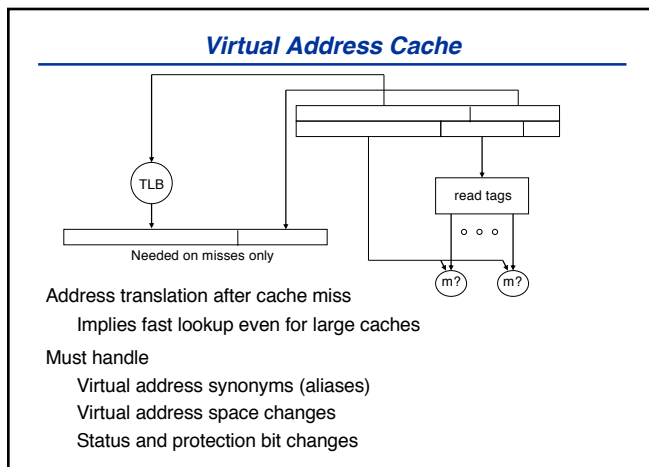
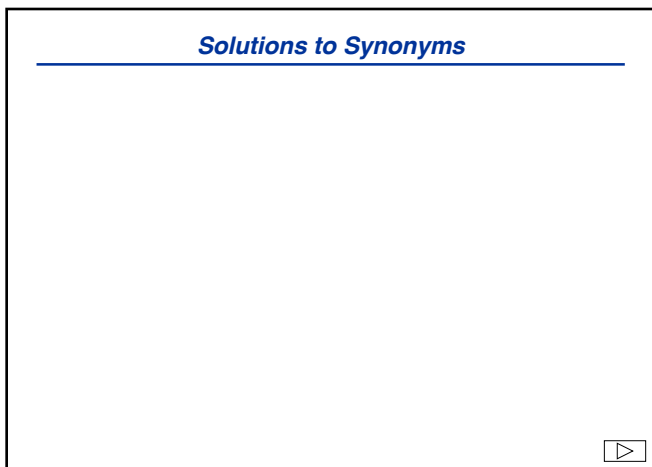
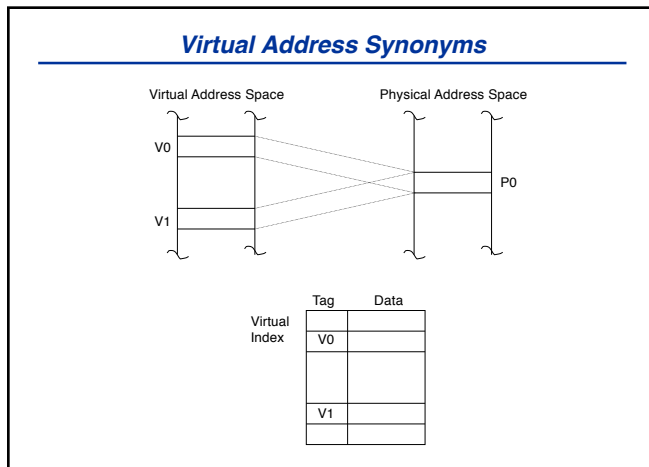
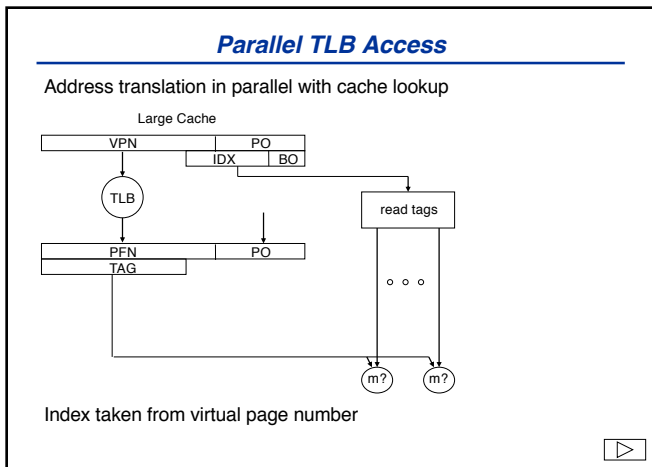
- Approx. LRU

**Write strategy**

- Writeback (with page dirty bit)







### Protection

Goal:

One process should not be able to interfere with the execution of another

Process model

Privileged kernel  
Independent user processes

Primitives vs. Policy

Architecture provides the primitives  
Operating system implements the policy  
Problems arise when hardware implements policy

### Protection Primitives

User vs. Kernel

At least one privileged mode  
Usually implemented as mode bit(s)

How do we switch to kernel mode?

Change mode and continue execution at *predetermined* location

Hardware to compare mode bits to access rights

Access certain resources only in kernel mode

### Protection Primitives, cont.

Base and Bounds

Privileged registers  
 $\text{Base} \leq \text{Address} \leq \text{Bounds}$

Pagelevel protection

Protection bits in page table entry  
Cache them in TLB