CprE 381: Computer Organization and Assembly Level Programming

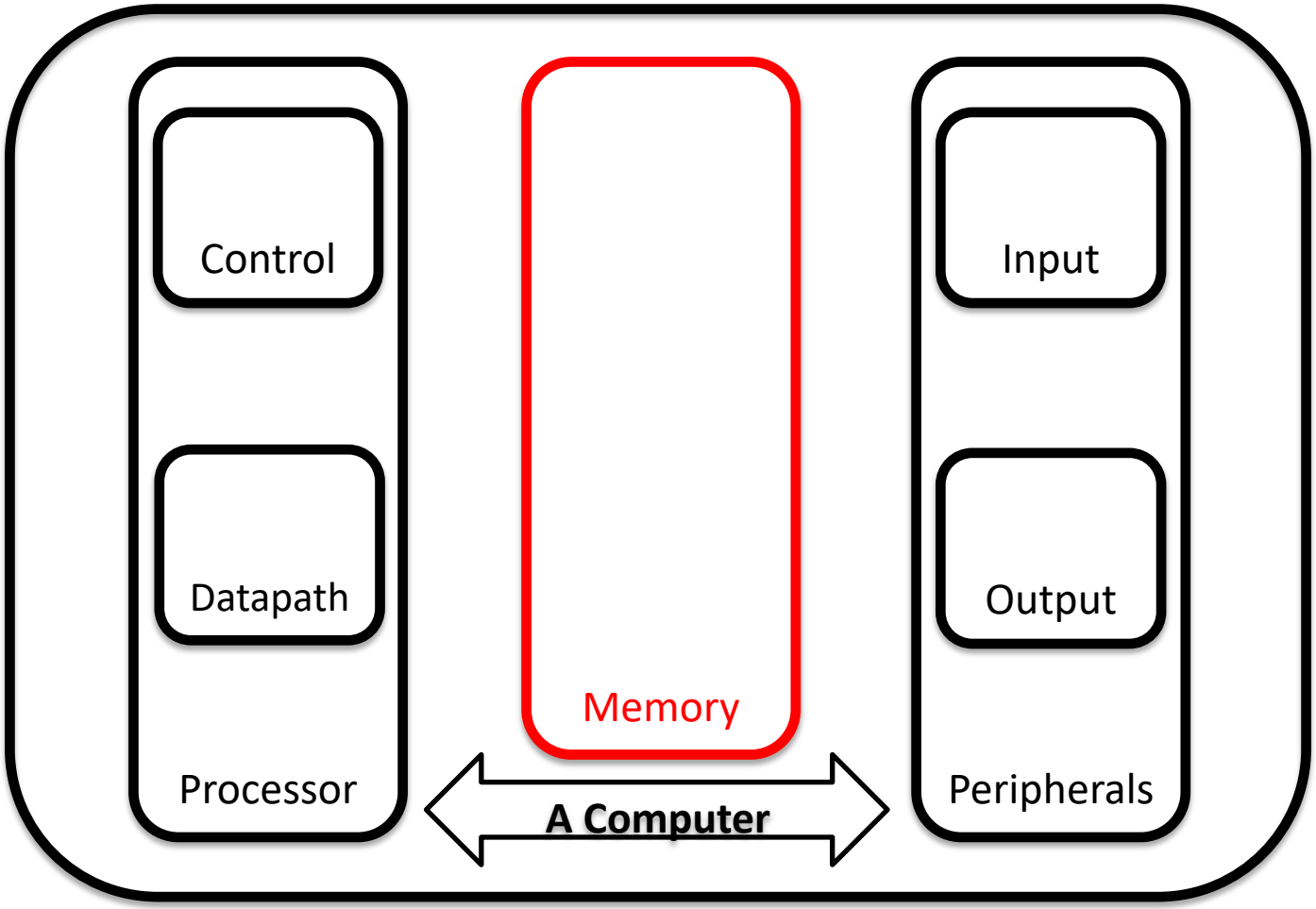
Memory and Storage

Henry Duwe
Electrical and Computer Engineering
Iowa State University

Administrative

- Exam 2
  - Grades by tomorrow morning
  - Raw score will be posted to Canvas
- HW8:
  - Problem 1: redo exam question on which you lost the most points (total, not fraction)
  - Problem 2: pipeline hazard-detection test program
- Project Part 3a
  - Checkpoint demo in a week
  - **WARNING**: Much easier than Part 3b!

Remember the System View!

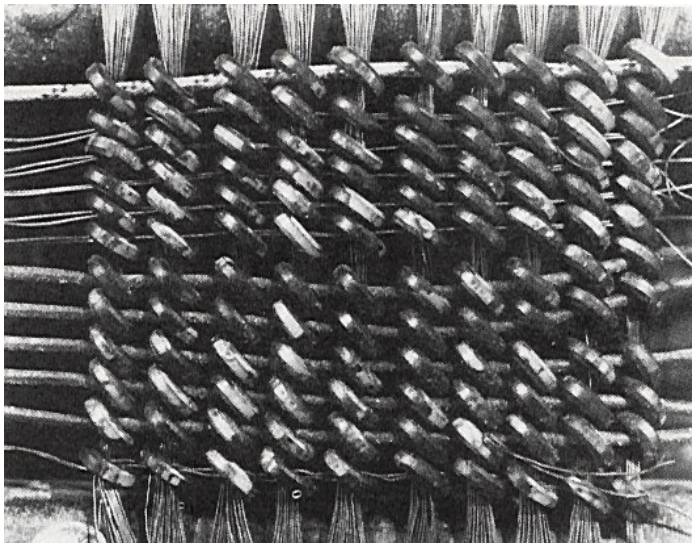


Memory + I/O

- We've already seen how to make a fast processor. How can we supply the CPU with enough data to keep it busy?
- The 3rd part of CprE 381 focuses on memory and input/output issues, which are frequently bottlenecks that limit the performance of a system.
- We'll start off by looking at memory systems for the next few weeks:
  - Real memory
  - How caches can dramatically improve the speed of memory accesses.
  - How virtual memory provides security and ease of programming
  - How processors, memory and peripheral devices can be connected

In the Beginning

- In the earliest electronic computing memory was very hard
  - Memory devices used dots on a CRT screen, and stuff that was even worse!
  - In late 40's core memory was invented
    - This was the dominant memory until the mid 70's
    - Threaded by hand!



[http://www.columbia.edu/acis/history/core.html]

- Early semiconductor memory
  - Static memory
    - 256 bits
  - Dynamic memory
    - MOS
    - 1K bits in 1970, i1103

DRAMs

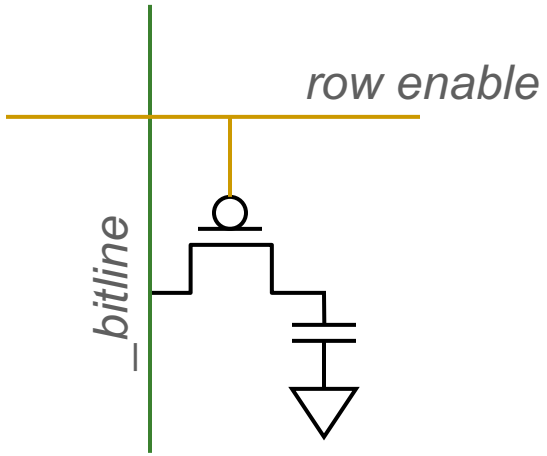
- In the late 70's a type of memory became cheapest/bit
  - DRAM – dynamic random access memory
- It was built in MOS technology
  - So it was reasonably fast
- Processors could just access that memory when they needed it
  - Processors were slower than memory

- But as technology scaled:
  - Memories got denser and a little faster
  - Processors got a lot faster

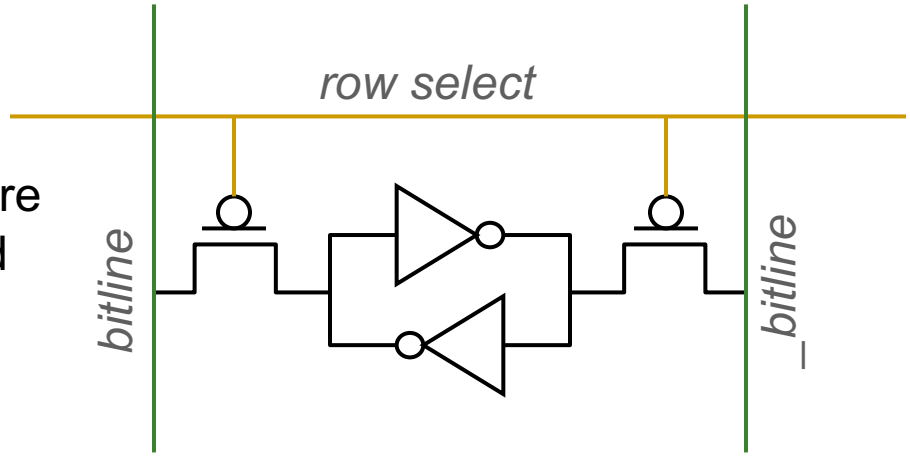
- But before we talk about this problem,
  - Let's review the kinds of memory

Random Access Memory (RAM)

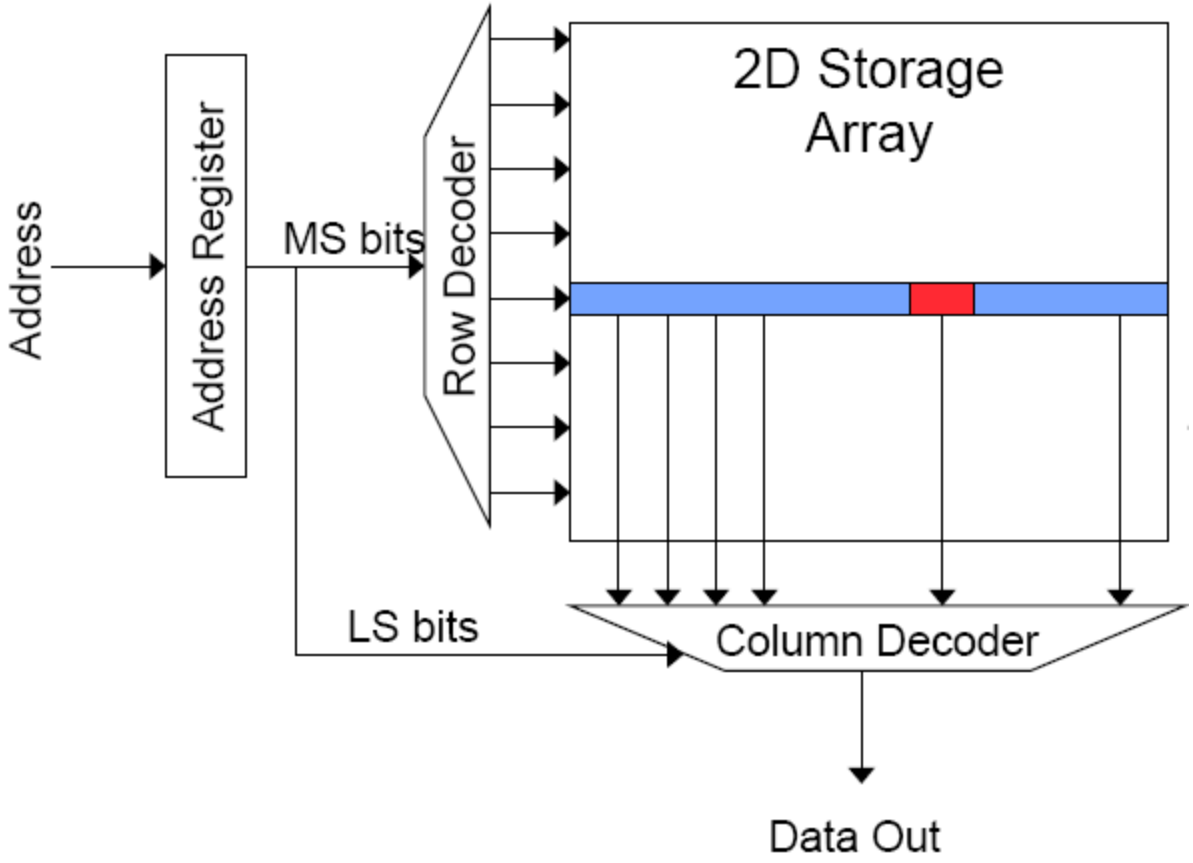
- Dynamic Random Access Memory (DRAM)
  - High density, low power, cheap, but slow
  - Dynamic since data must be "refreshed" regularly ("leaky buckets")
  - Contents are lost when power is lost



- Static Random Access Memory (SRAM)
  - Lower density, (about 1/10 density of DRAM)
    - This means it is going to cost more
  - Static since data is held "forever" and do not require a refresh
    - But only when the power is on
  - Fast access time, often 2 to 10 times faster than DRAM



Memory Bank Organization and Operation



- Read access sequence:

1. Decode row address & drive word-lines
2. Selected bits drive bit-lines
   - Entire row read
3. Amplify row data
4. Decode column address & select subset of row
   - Send to output
5. Precharge bit-lines for next access

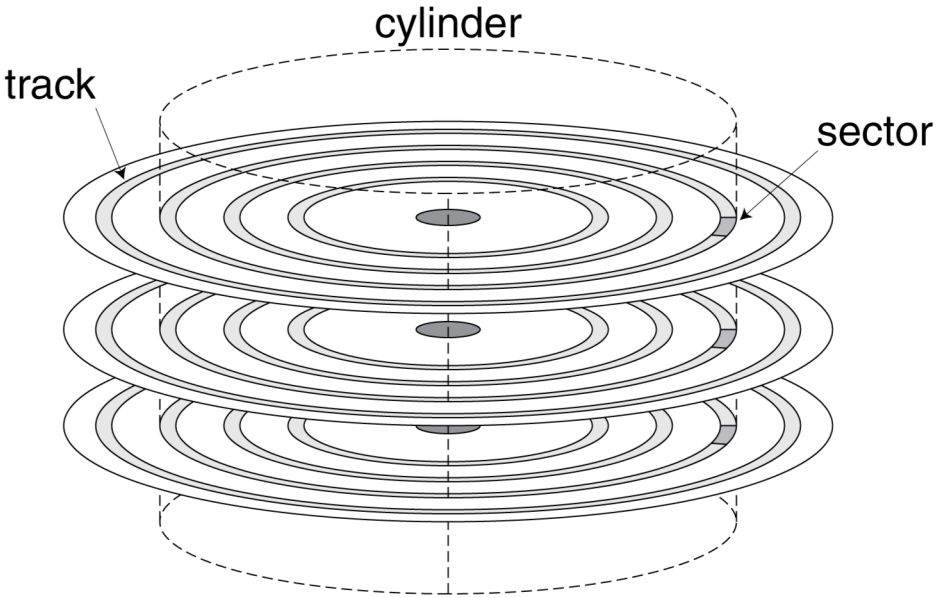
Other Solid State Storage

- There really is even cheaper storage
  - That does not need power to remember data
  - Solid state storage - hard drives!
  - Today this costs << $1/GB

- The access to these type of devices is different
  - Block access
    - Mechanically move an arm to the right track
    - Wait for the data you want to rotate under the head
  - Very slow access times (ms)
    - Reasonable transfer rates

Disk Storage

- Nonvolatile, rotating magnetic storage



Disk Sectors and Access

- Each sector records
  - Sector ID
  - Data (512 bytes, 4096 bytes proposed)
  - Error correcting code (ECC)
    - Used to hide defects and recording errors
  - Synchronization fields and gaps
- Access to a sector involves
  - Queuing delay if other accesses are pending
  - Seek: move the heads
  - Rotational latency
  - Data transfer
  - Controller overhead

Disk Access Example

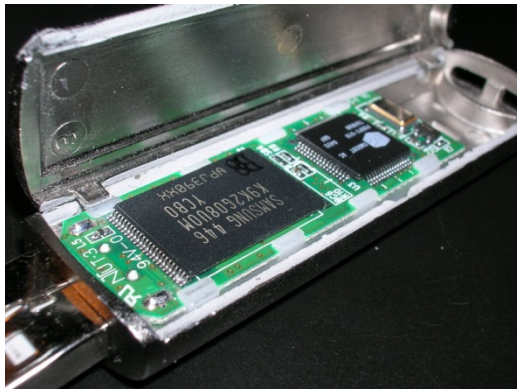
- Given
  - 512B sector, 15,000rpm, 4ms average seek time, 100MB/s transfer rate, 0.2ms controller overhead, idle disk
- Average read time
  - 4ms seek time
    + ½ / (15,000/60) = 2ms rotational latency
    + 512 / 100MB/s = 0.005ms transfer time
    + 0.2ms controller delay
    = 6.2ms
- If actual average seek time is 1ms
  - Average read time = 3.2ms

Disk Performance Issues

- Manufacturers quote average seek time
  - Based on all possible seeks
  - Locality and OS scheduling lead to smaller actual average seek times
- Smart disk controller allocate physical sectors on disk
  - Present logical sector interface to host
  - SCSI, ATA, SATA
- Disk drives include caches
  - Prefetch sectors in anticipation of access
  - Avoid seek and rotational delay

Flash Storage

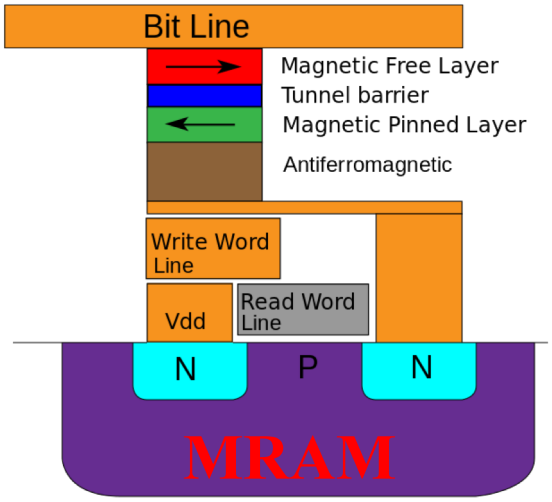
- Nonvolatile semiconductor storage
  - 100× – 1000× faster than disk
  - Smaller, lower power, more robust
  - But more $/GB (between disk and DRAM)

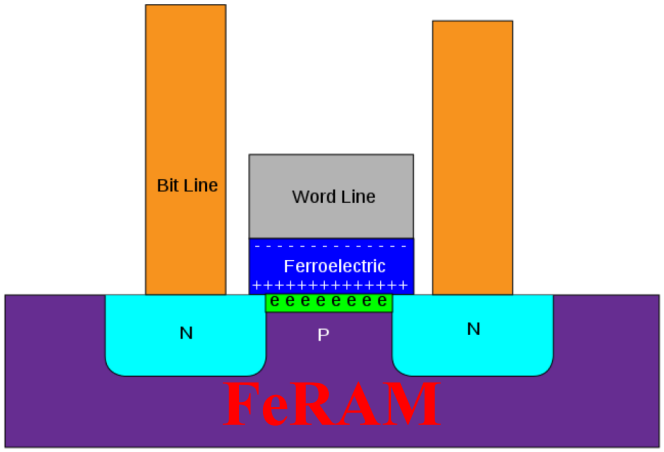


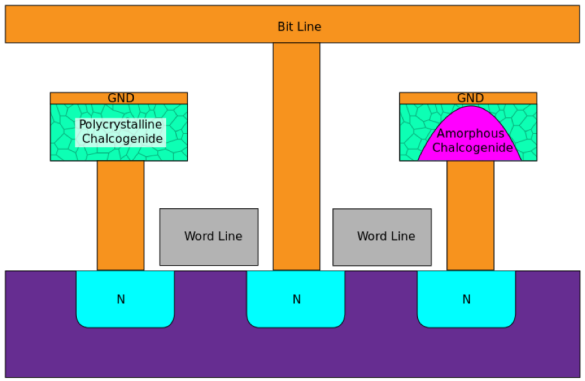
Flash Types

- NOR flash: bit cell like a NOR gate
  - Random read/write access
  - Used for instruction memory in embedded systems
- NAND flash: bit cell like a NAND gate
  - Denser (bits/area), but block-at-a-time access
  - Cheaper per GB
  - Used for USB keys, media storage, …
- Flash bits wears out after 1000's of accesses
  - Not suitable for direct RAM or disk replacement
  - Wear leveling: remap data to less used blocks

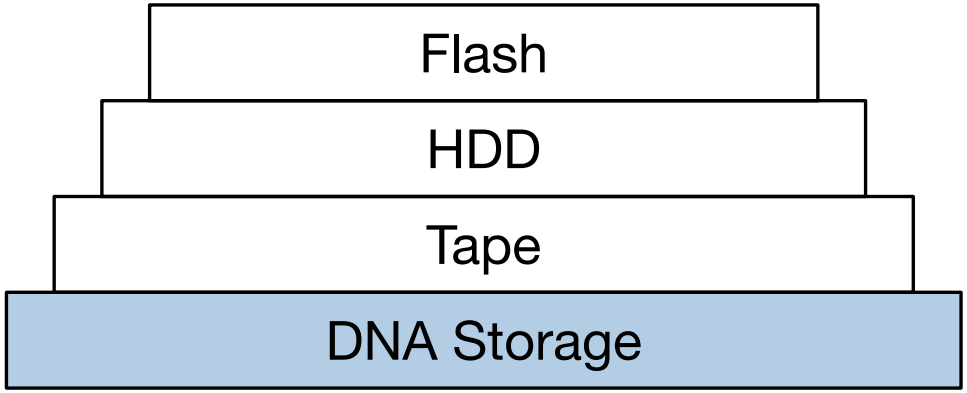
Other Technologies







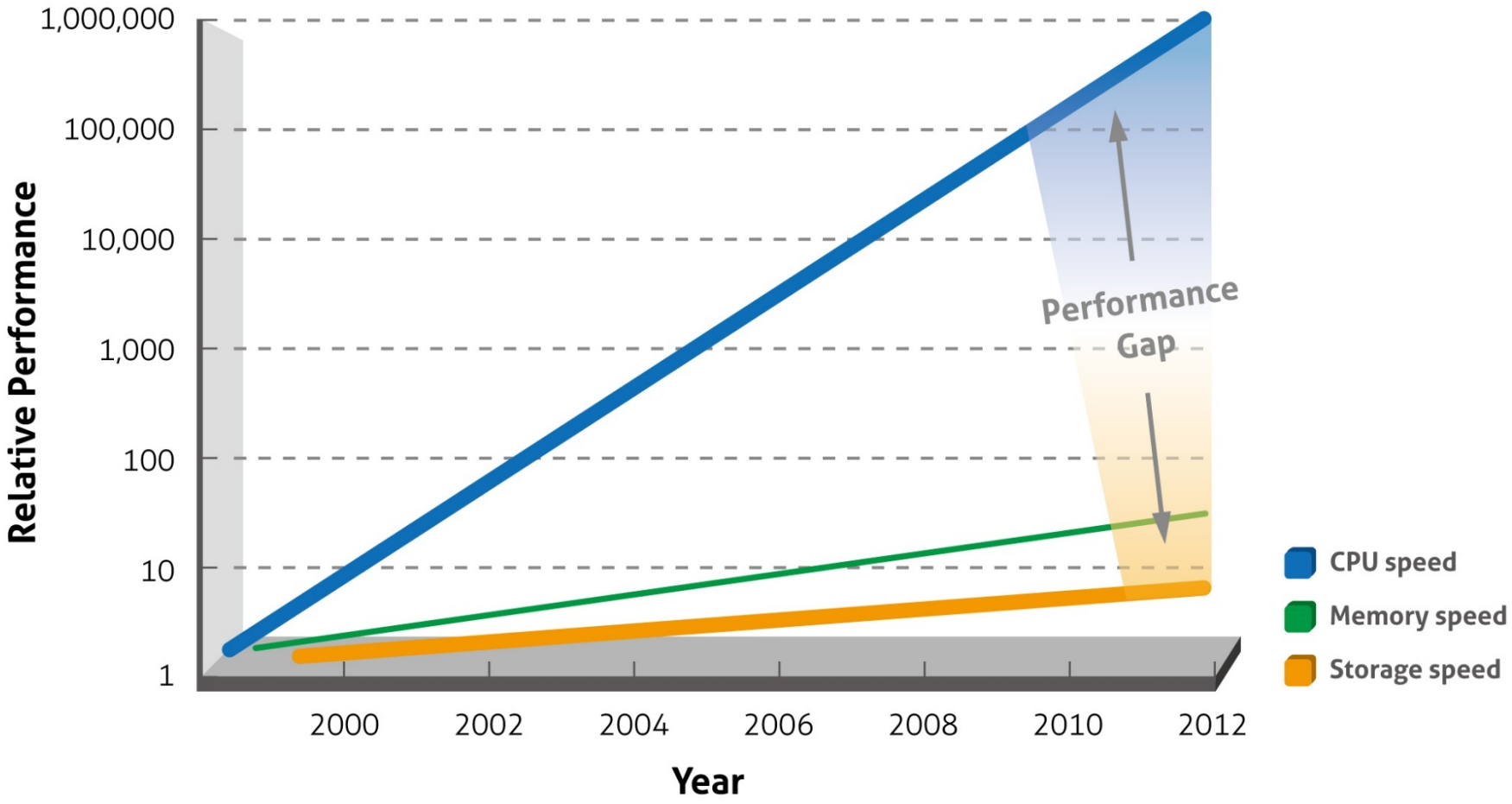
[https://en.wikipedia.org/wiki/Non-volatile_random-access_memory]



	Access Time	*Durability*
Flash	ms	~5 yrs
HDD	10s ms	~5 yrs
Tape	minutes	~15-30 yrs
DNA Storage	10s hrs	centuries

[Bornholt et al "A DNA-Based Archival Storage System"]

The Processor-Memory Performance Gap



- Starting to flatten out
- Can we afford to stall the pipeline for this long??

Large and Fast

- Today's computers depend upon large and fast storage systems.
  - Large storage capacities are needed for many database applications, scientific computations with large data sets, video and music, Big Data analytics/Machine Learning, and so forth.
  - Speed is important to keep up with our pipelined CPUs, which may access both an instruction and data in the same clock cycle. Things get become even worse if we move to a superscalar CPU design.

- So far we've assumed our memories can keep up and our CPU can access memory twice in one cycle, but as we'll see that's a simplification.

Storage Trade-off?

In-class Assessment!

Access Code: URTurn

Note: sharing access code to those outside of classroom or using access while outside of classroom is considered cheating

Small or Slow

- Unfortunately there is a tradeoff between speed, cost and capacity

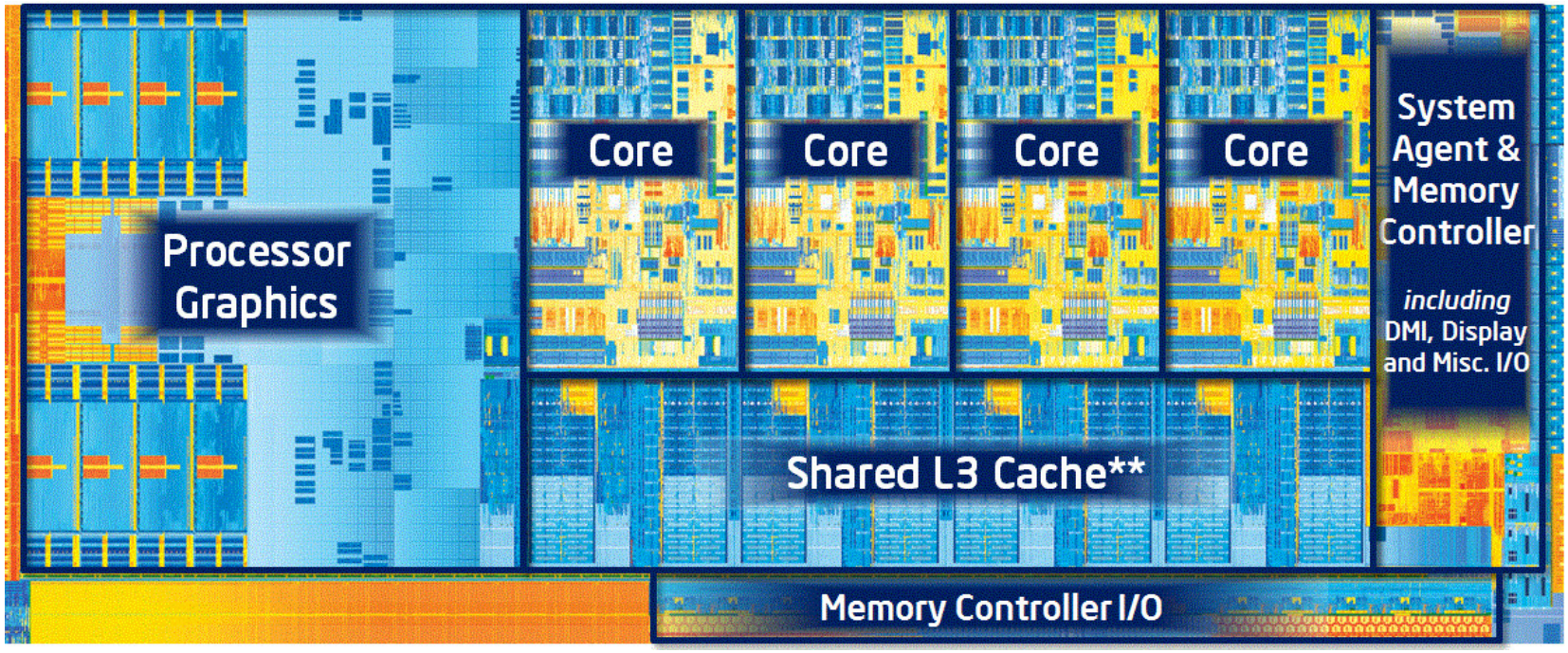
Storage	Speed	Cost	Capacity
Static RAM	Fastest	Expensive	Smallest
Dynamic RAM	Slow	Cheap	Large
Hard disks	Slowest	Cheapest	Largest

- Fast memory is too expensive for most people to buy a lot of
- But dynamic memory has a much longer delay than other functional units in a datapath. If every `lw` or `sw` accessed dynamic memory, we'd have to either increase the cycle time or stall frequently
- Here are ***rough*** estimates of some current storage parameters

Storage	Delay	Cost/MB	Capacity
Static RAM	1-10 cycles	~$1	128KB-128MB
Dynamic RAM	100-200 cycles	~$0.005	256MB-512GB
Hard disks	10,000,000 cycles	~$0.00005	512GB-10TB

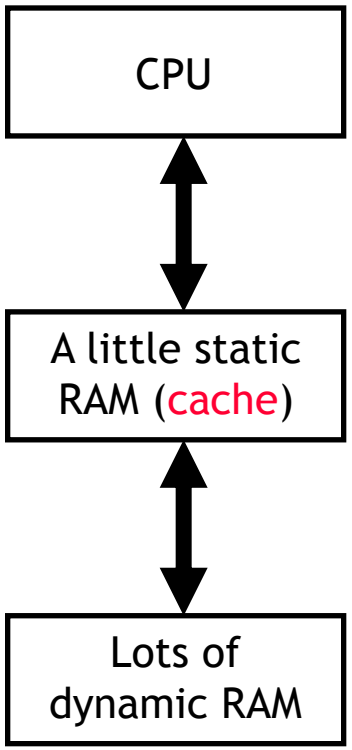
Preview: Cache Introduction

- Today we'll answer the following questions.
  - What are the challenges of building big, fast memory systems?
  - What is a cache?
  - Why caches work? (answer: locality)
  - How are caches organized?
    - Where do we put things -and- how do we find them?



Introducing Caches

- Wouldn't it be nice if we could find a balance between fast and cheap memory?
- We do this by introducing a cache, which is a small amount of fast, expensive memory
  - The cache goes between the processor and the slower, dynamic main memory
  - It keeps a copy of the most frequently used data from the main memory
- Memory access speed increases overall, because we've made the common case faster
  - Reads and writes to the most frequently used addresses will be serviced by the cache
  - We only need to access the slower main memory for less frequently used data



Acknowledgments

- These slides contain material developed and copyright by:
  - Joe Zambreno (Iowa State)
  - Akhilesh Tyagi (Iowa State)
  - David Patterson (UC Berkeley)
  - Mary Jane Irwin (Penn State)
  - Christos Kozyrakis (Stanford)
  - Onur Mutlu (Carnegie Mellon)
  - Krste Asanović (UC Berkeley)
  - Morgan Kaufmann