Recitation 14

All good things…

Today’s agenda

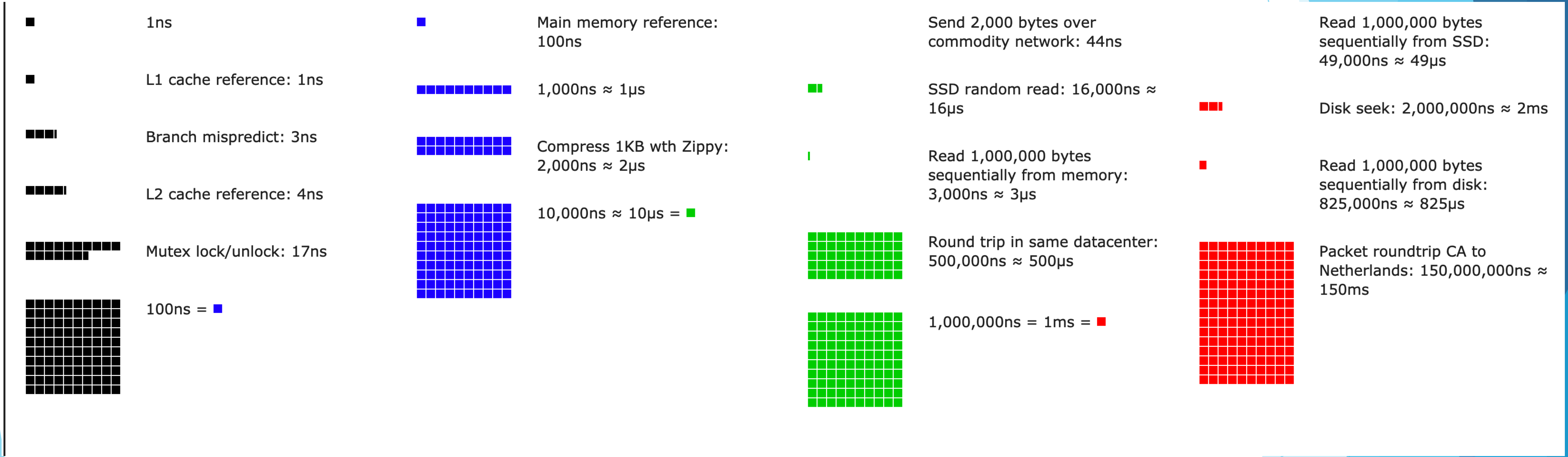
- We will discuss in recitation
  - Mock Exam
  - Caches
  - Virtual Memory
- For homework tonight
  - R14
    - Answer some questions
  - Check ALL of your Recitation grades!!! Email me with any concerns ASAP!!!

Caches

Memory

- Main memory is SLOW!!!

From “Latency Numbers Every Programmer Should Know”



1ns

L1 cache reference: 1ns

Branch mispredict: 3ns

L2 cache reference: 4ns

Mutex lock/unlock: 17ns

100ns = 🟦

Main memory reference: 100ns

1,000ns ≈ 1μs

Compress 1KB wth Zippy: 2,000ns ≈ 2μs

10,000ns ≈ 10μs = 🟩

Send 2,000 bytes over commodity network: 44ns

SSD random read: 16,000ns ≈ 16μs

Read 1,000,000 bytes sequentially from memory: 3,000ns ≈ 3μs

Round trip in same datacenter: 500,000ns ≈ 500μs

1,000,000ns = 1ms = 🟥

Read 1,000,000 bytes sequentially from SSD: 49,000ns ≈ 49μs

Disk seek: 2,000,000ns ≈ 2ms

Read 1,000,000 bytes sequentially from disk: 825,000ns ≈ 825μs

Packet roundtrip CA to Netherlands: 150,000,000ns ≈ 150ms

Memory

- Main memory is SLOW!!!
  - Slow enough that it would be a real bottleneck
- To get around being slow, we use caching
  - Basically, we store some data in really fast data that is IN the CPU

Memory

- Main memory is SLOW!!!
  - Slow enough that it would be a real bottleneck
- To get around being slow, we use caching
  - Basically, we store some data in really fast data that is IN the CPU

Caching

- There are a few different schemes and concepts
  - Associativity
    - Where can data be stored in the cache?
    - Direct Cache, Fully Associative, Set Associative
  - Replacement/Eviction Policy
- What do we store in the cache?
  - The data, part of the address, and some information on if something is there or not
- Have multiple levels of cache

Virtual Memory

Virtual Addresses

- Each virtual address is mapped to some physical address
- The translation occurs using a “page table”
- Each virtual address contains a page offset, which maps to a single byte within a page
- Each virtual address contains one or more page indexes into a number of tables

Page Tables

- Page tables map virtual page numbers to physical page numbers
- Page tables contain a number bits to mark whether an entry is present in memory along with the mapping to a physical page number
- The physical page number is the upper bits for the physical memory address
  - Along with the page offset it makes up the physical address

Multilevel Page Tables

- We would need to store many mappings if our addresses are large
- Instead, we can have multiple layers of page tables
- We split the virtual address into several parts, where each part of the address is an index into a table
- Each table entry tells us the physical page number of the next table

The TLB

- Having to access memory to translate memory addresses would be very sad
- Instead, we use the TLB, which is a cache for recently translated addresses
- On TLB miss, we translate the address normally, and then cache the result