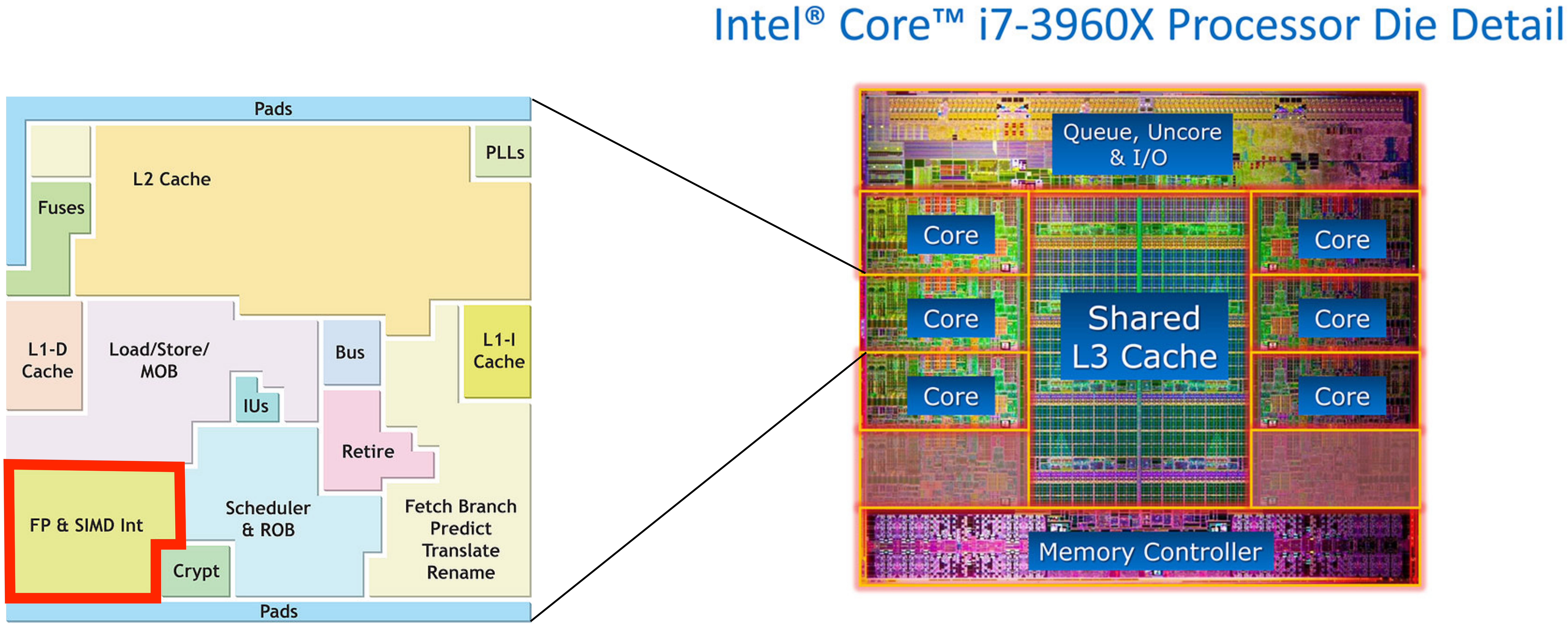
Lecture 3: Single-processor Computing Summary

CMSE 822: Parallel Computing
Prof. Sean M. Couch

Anatomy of a Computation

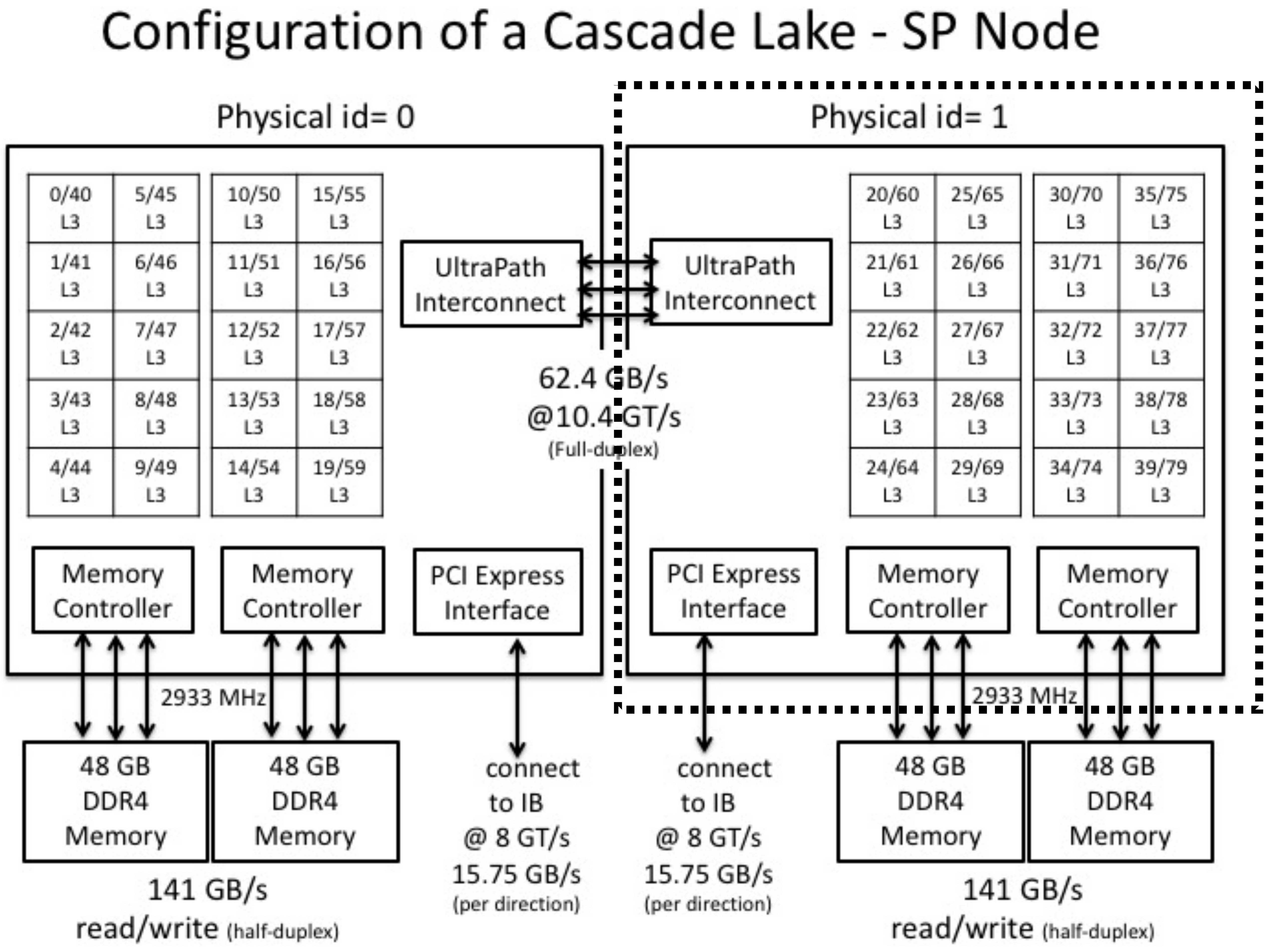
A CPU



Single-CPU computing is parallel!

Anatomy of a Computation

A Node



Configuration of a Cascade Lake - SP Node

Physical id= 0

0/40 L3	5/45 L3	10/50 L3	15/55 L3
1/41 L3	6/46 L3	11/51 L3	16/56 L3
2/42 L3	7/47 L3	12/52 L3	17/57 L3
3/43 L3	8/48 L3	13/53 L3	18/58 L3
4/44 L3	9/49 L3	14/54 L3	19/59 L3

UltraPath Interconnect

Memory Controller

Memory Controller

PCI Express Interface

2933 MHz

48 GB DDR4 Memory

48 GB DDR4 Memory

141 GB/s read/write (half-duplex)

connect to IB @ 8 GT/s 15.75 GB/s (per direction)

62.4 GB/s @10.4 GT/s (Full-duplex)

Physical id= 1

20/60 L3	25/65 L3	30/70 L3	35/75 L3
21/61 L3	26/66 L3	31/71 L3	36/76 L3
22/62 L3	27/67 L3	32/72 L3	37/77 L3
23/63 L3	28/68 L3	33/73 L3	38/78 L3
24/64 L3	29/69 L3	34/74 L3	39/79 L3

UltraPath Interconnect

PCI Express Interface

Memory Controller

Memory Controller

2933 MHz

connect to IB @ 8 GT/s 15.75 GB/s (per direction)

48 GB DDR4 Memory

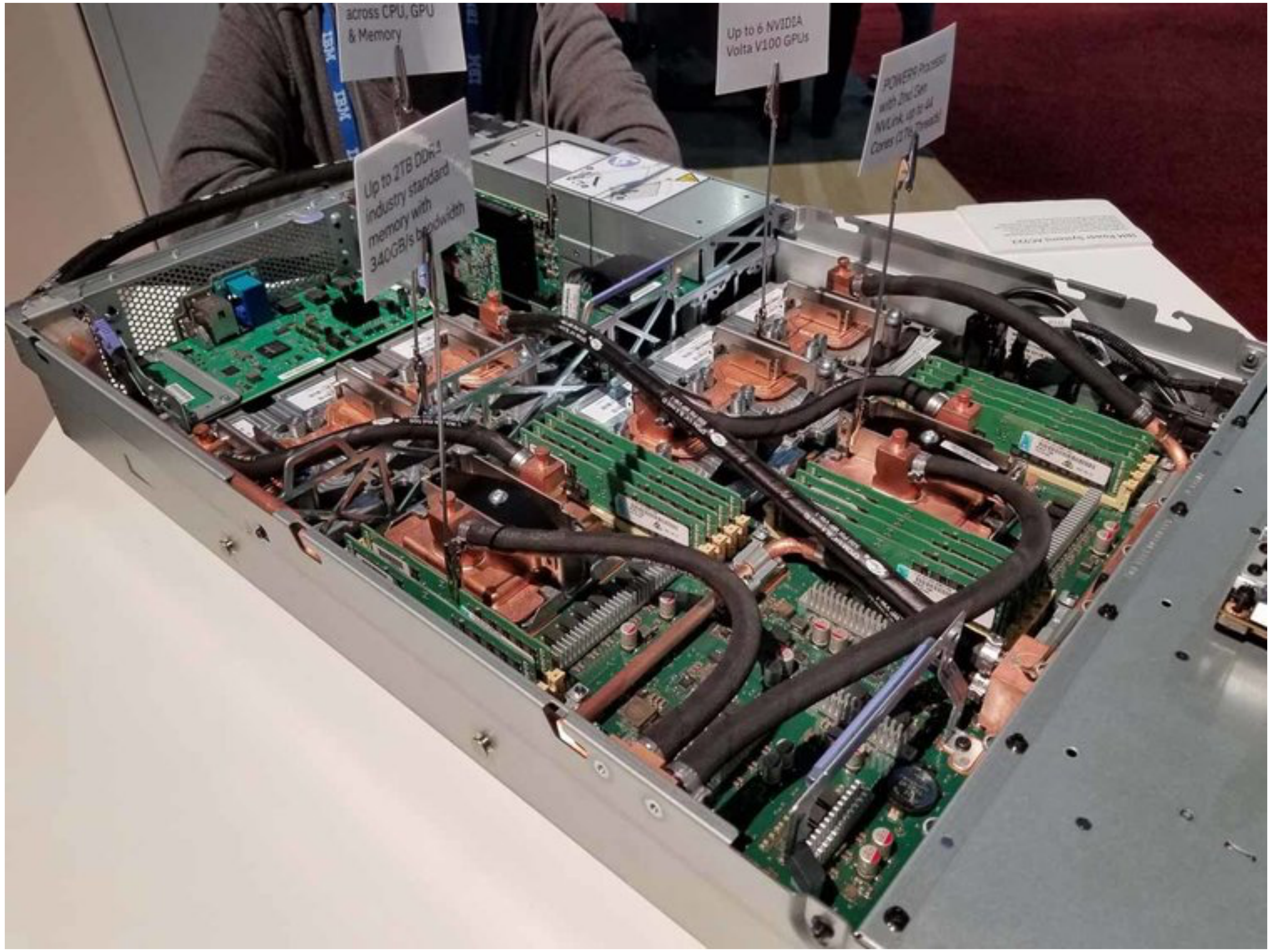
48 GB DDR4 Memory

141 GB/s read/write (half-duplex)

“socket”

Anatomy of a Computation

A Node



Summit, ORNL

Anatomy of a Computation

A Node

Summit, ORNL

Anatomy of a Computation

Node-to-node Interconnect

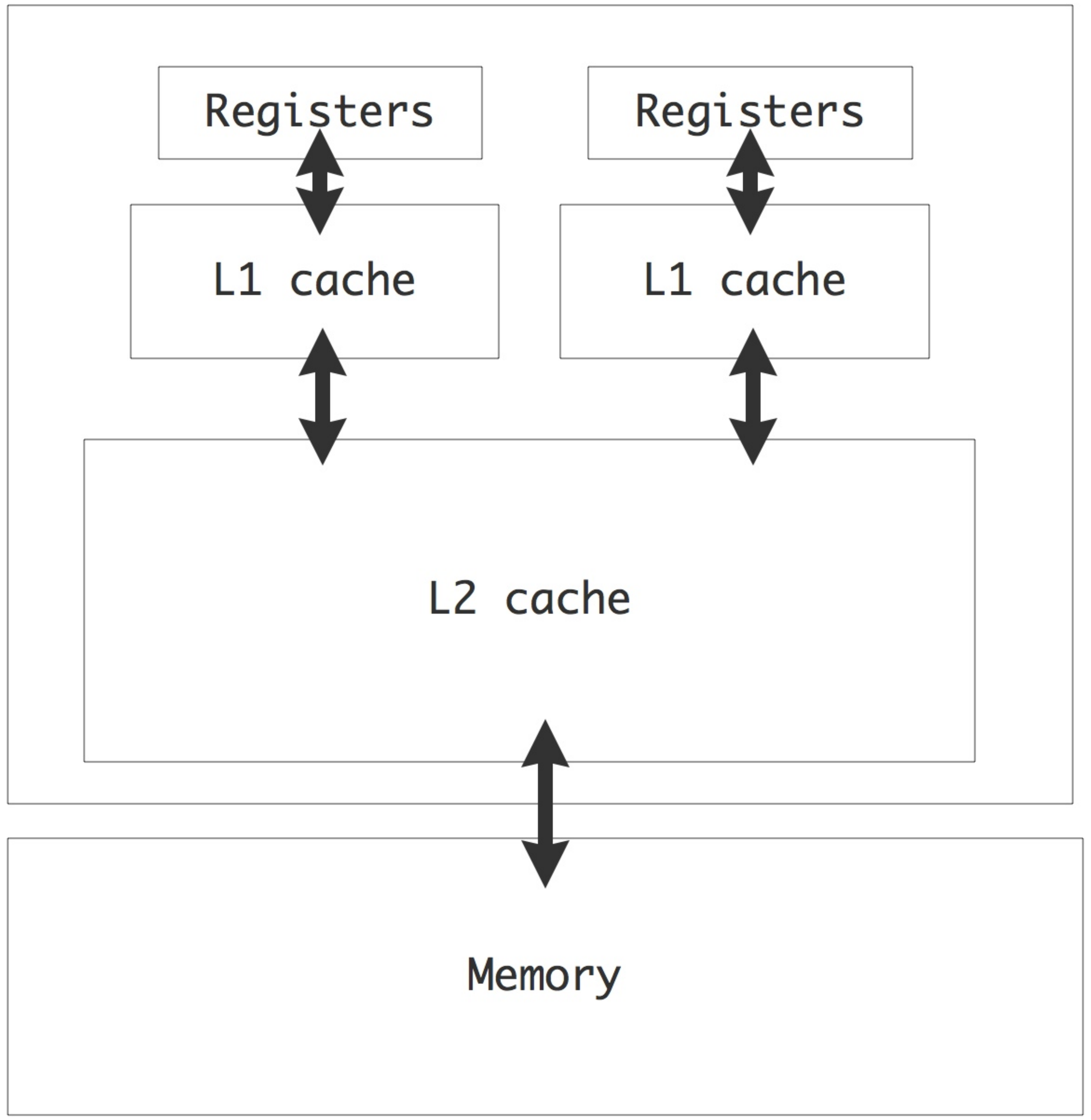
Anatomy of a Computation

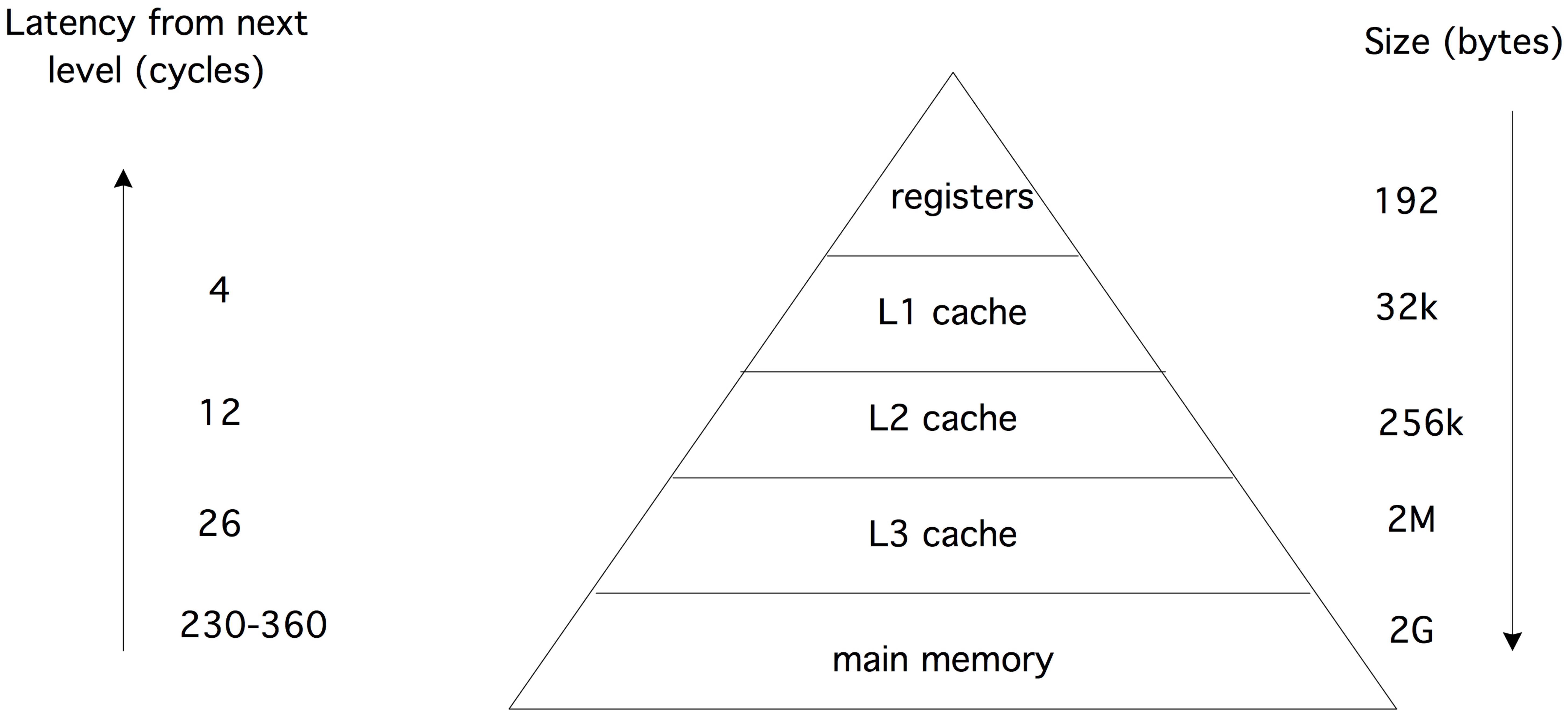
Cluster

Summit, ORNL

Memory hierarchy

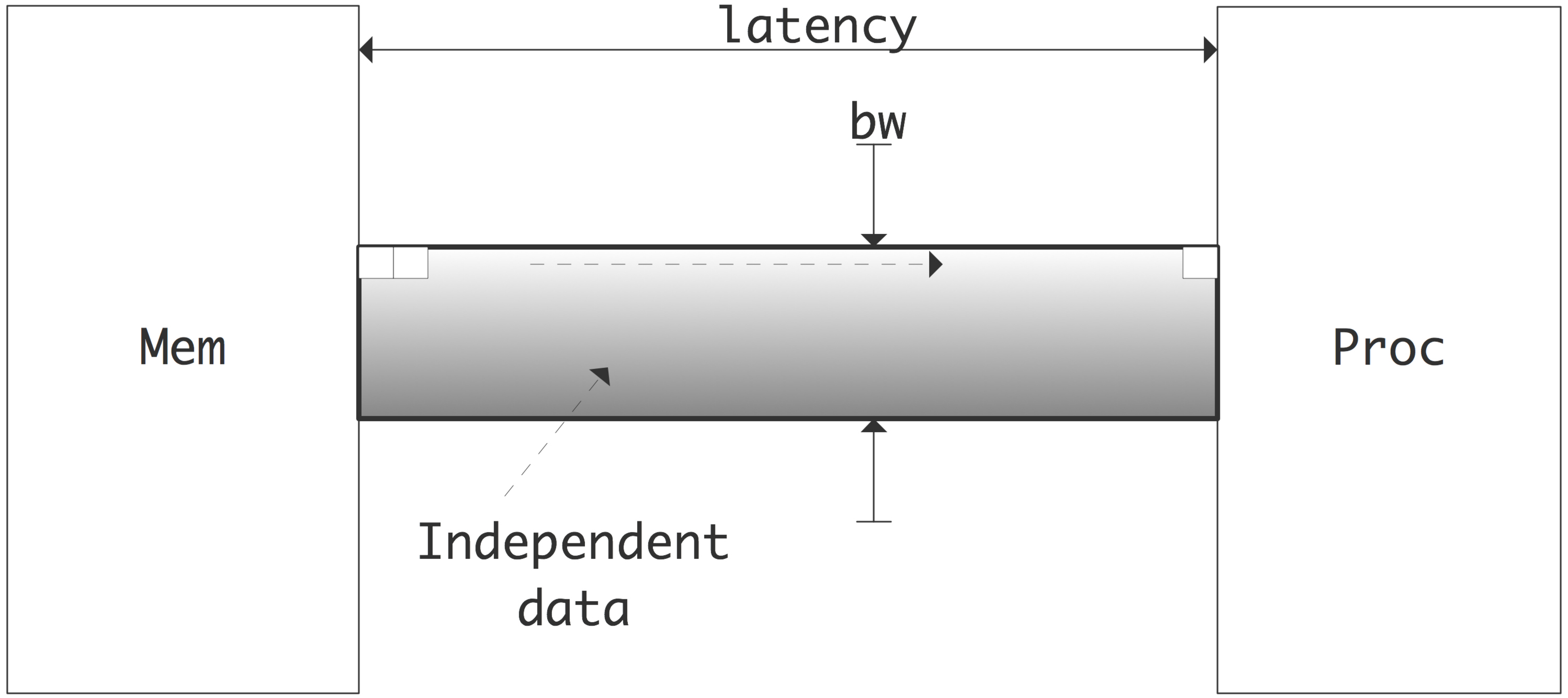
Often the limiter of performance…





Memory hierarchy

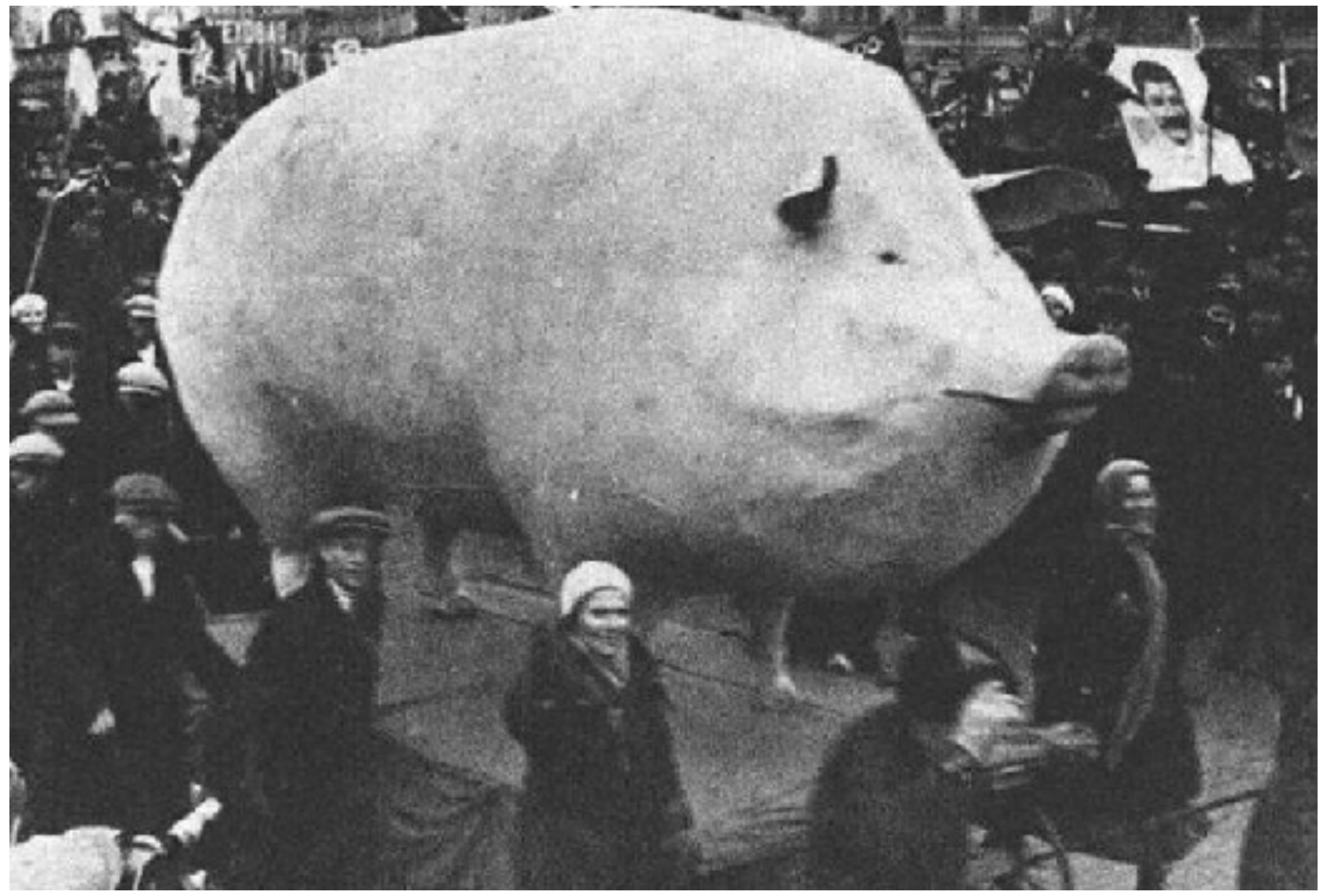
Need to feed the beast (er, CPU)



Little's Law: Concurrency = Bandwidth x Latency

Memory hierarchy

Absolute unit



What is the fundamental unit of memory movement?

a. page

b. word

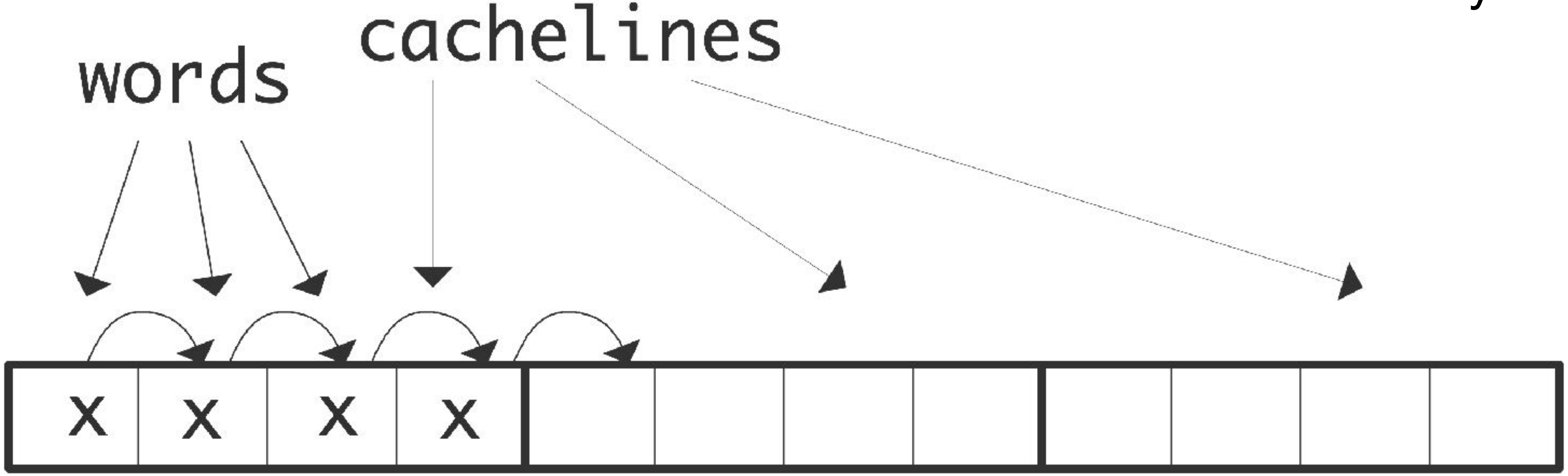
c. line

d. byte

Memory hierarchy

Cache line

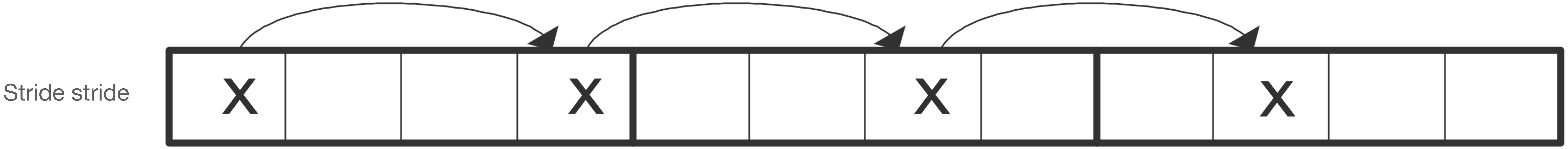
Lines: 64-128 bytes



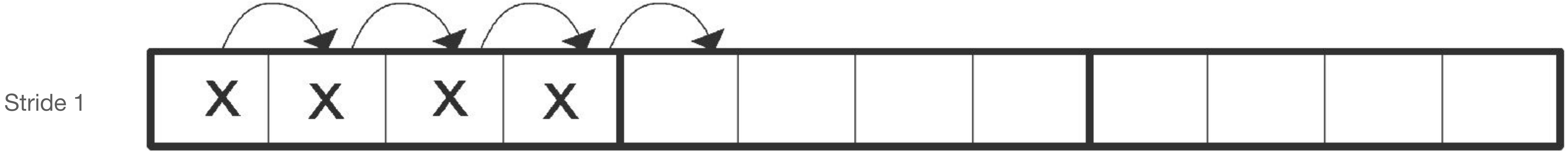
Words: usually 64 bits

Memory hierarchy

Strided access



```
for (i=0; i<N; i+=stride)
  ... = ... x[i] ...
```



```
for (i=0; i<N; i++)
  ... = ... x[i] ...
```

Memory hierarchy

Reuse is key to performance!

- Compulsory cache miss: first time memory is referenced
- Capacity cache miss: cache not big enough to fit problem
- Conflict cache miss: data mapped to same cache location as another
- Invalidation cache miss: another core changed value at memory address

Memory hierarchy

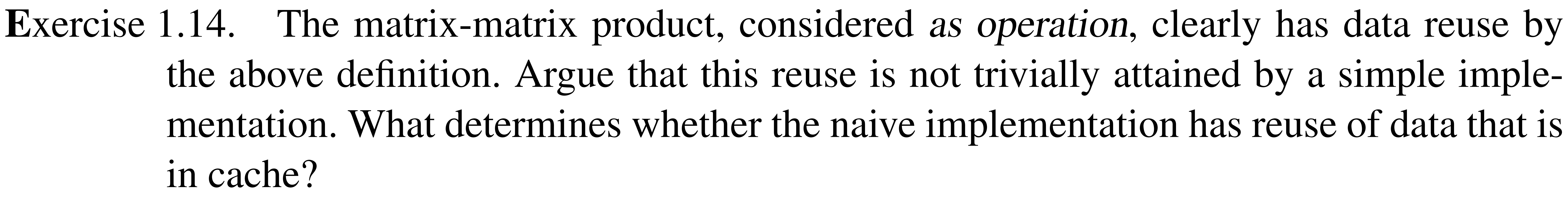
False sharing

```
  local_results = new double[num_threads];
#pragma omp parallel
{
  int thread_num = omp_get_thread_num();
  for (int i=my_lo; i<my_hi; i++)
    local_results[thread_num] = ... f(i) ...
}
global_result = g(local_results)
```

- Cores access and alter data in same *cache line*

Exercise 1.14

Matrix-matrix Multiply



Exercise 1.14. The matrix-matrix product, considered *as operation*, clearly has data reuse by the above definition. Argue that this reuse is not trivially attained by a simple implementation. What determines whether the naive implementation has reuse of data that is in cache?

Caches can only hold a finite amount of data. Once a row of A and a column of B take up more than the size of the cache, their elements will be flushed between iterations of the outer loop.

Project 1

Group work