High-Performance Scientific Computing
Lecture 3: OpenCL

MATH-GA 2011 / CSCI-GA 2945 · September 19, 2012

Today

HW2

Chips for Throughput

Synchronization

Admin Bits

- New here? Please send email

Admin Bits

- New here? Please send email
- Started looking for a final project yet?

Admin Bits

- New here? Please send email
- Started looking for a final project yet?
- HW1 not found $\rightarrow$ email

Admin Bits

- New here? Please send email
- Started looking for a final project yet?
- HW1 not found $\rightarrow$ email
- Grading

Admin Bits

- New here? Please send email
- Started looking for a final project yet?
- HW1 not found $\rightarrow$ email
- Grading
- Overall pace

Admin Bits

- New here? Please send email
- Started looking for a final project yet?
- HW1 not found $\rightarrow$ email
- Grading
- Overall pace
- HW3 out on the weekend

Outline

HW2 problem 2

Demo time

OpenMP sync primitives

- Critical section

OpenMP sync primitives

- Critical section
- Locks

OpenMP sync primitives

- Critical section
- Locks
- Atomics

OpenMP sync primitives

- Critical section
- Locks
- Atomics
  - Update: `x++;`

OpenMP sync primitives

- Critical section
- Locks
- Atomics
  - Update: `x++;`
  - Capture: `v = x++;`

OpenMP sync primitives

- Critical section
- Locks
- Atomics
  - Update: `x++;`
  - Capture: `v = x++;`
  - Structured: v = x; x —= expr; ("Test-and-set")

OpenMP sync primitives

- Critical section
- Locks
- Atomics
  - Update: `x++;`
  - Capture: `v = x++;`
  - Structured: v = x; x —= expr; ("Test-and-set")
  - Compare-and-swap (not in OpenMP)

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?
  - What is an orphaned directive?

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?
  - What is an orphaned directive?
  - What is close nesting?

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?
  - What is an orphaned directive?
  - What is close nesting?
  - What is a 'dynamic extent' of a region?

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?
  - What is an orphaned directive?
  - What is close nesting?
  - What is a 'dynamic extent' of a region?
- May a worksharing region be closely nested inside another one?

OpenMP corner case pop quiz 1

- May OpenMP directives be nested?
  - What is an orphaned directive?
  - What is close nesting?
  - What is a 'dynamic extent' of a region?
- May a worksharing region be closely nested inside another one?
- What happens if I nest two `critical` regions of the same name?

OpenMP corner case pop quiz 2

- Corresponding getter function for `omp_set_num_threads()`?

OpenMP corner case pop quiz 2

- Corresponding getter function for `omp_set_num_threads()`?
- Relation between `omp_set_dynamic()` and `schedule(dynamic)`?

OpenMP corner case pop quiz 2

- Corresponding getter function for `omp_set_num_threads()`?
- Relation between `omp_set_dynamic()` and `schedule(dynamic)`?
- What is wrong with this statement?

OpenMP corner case pop quiz 2

- Corresponding getter function for `omp_set_num_threads()`?
- Relation between `omp_set_dynamic()` and `schedule(dynamic)`?
- What is wrong with this statement?

> *A barrier region may not be closely nested inside a worksharing region.* *(from the OpenMP tutorial)*

OpenMP corner case pop quiz 2

- Corresponding getter function for `omp_set_num_threads()`?
- Relation between `omp_set_dynamic()` and `schedule(dynamic)`?
- What is wrong with this statement?

  A barrier region may not be closely nested inside a worksharing region. *(from the OpenMP tutorial)*

- What threads does a `barrier` bind to?

OpenMP corner case pop quiz 2

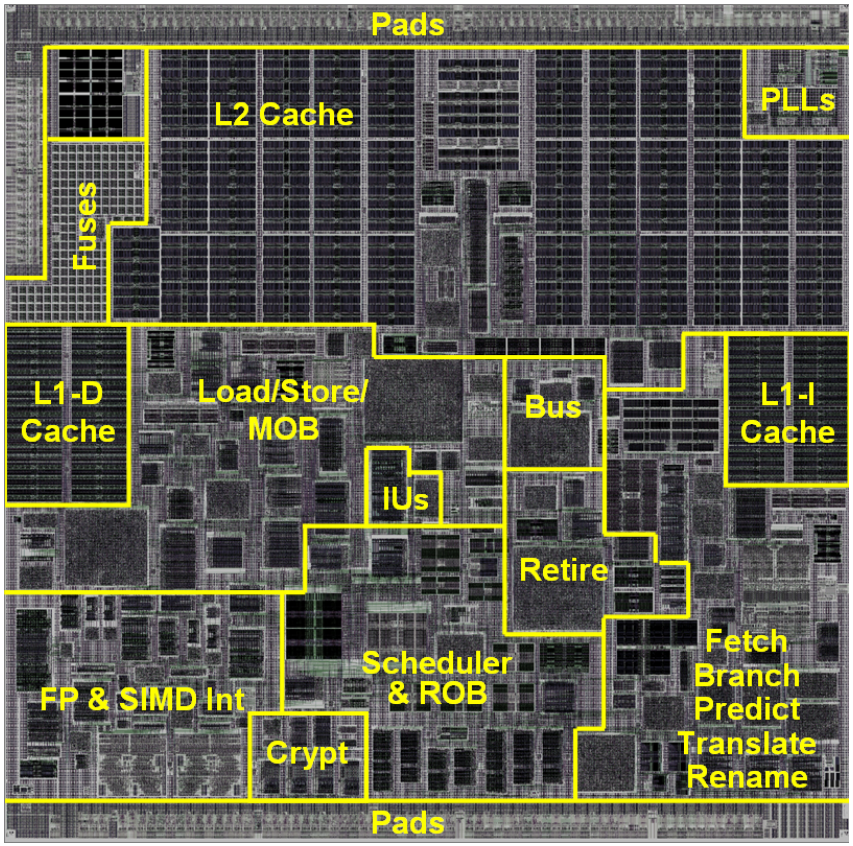
- Corresponding getter function for `omp_set_num_threads()`?
- Relation between `omp_set_dynamic()` and `schedule(dynamic)`?
- What is wrong with this statement?

  > *A barrier region may not be closely nested inside a worksharing region.* *(from the OpenMP tutorial)*

- What threads does a `barrier` bind to?
- What threads does a `critical` region bind to?

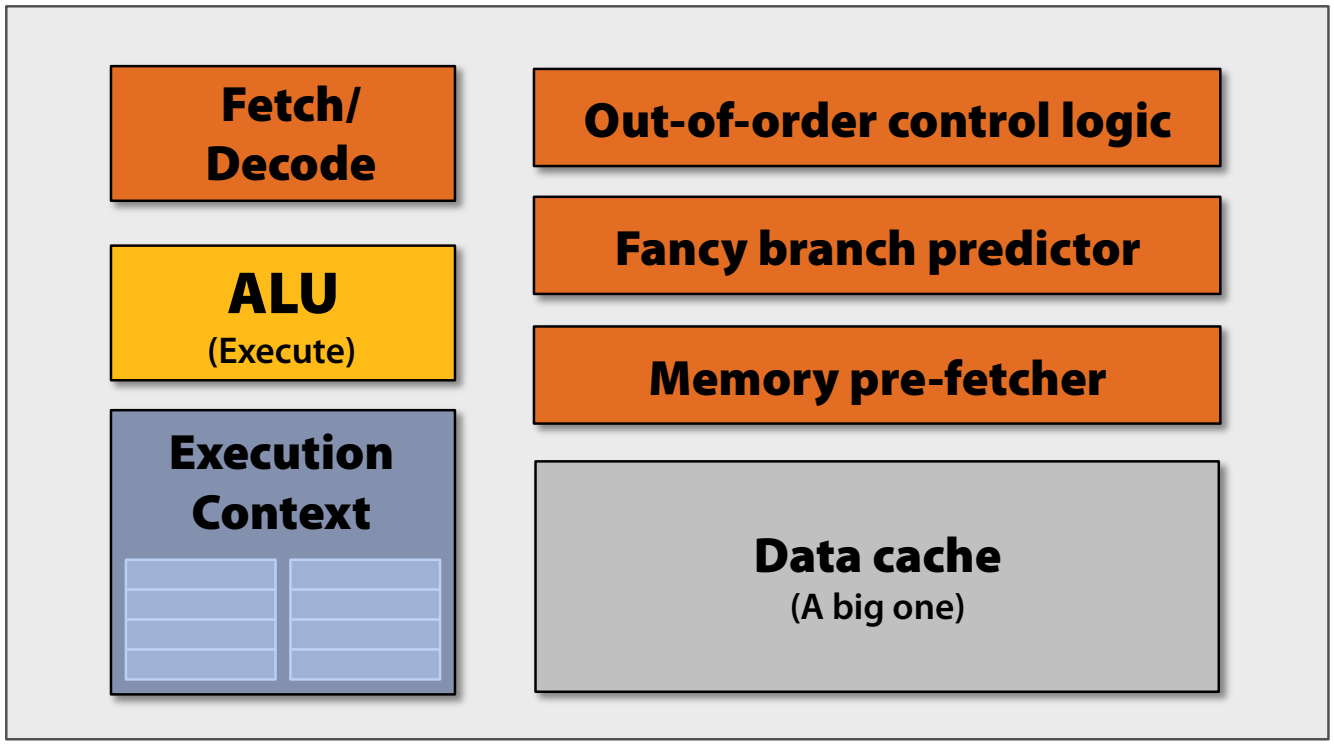
Outline

CPU Chip Real Estate



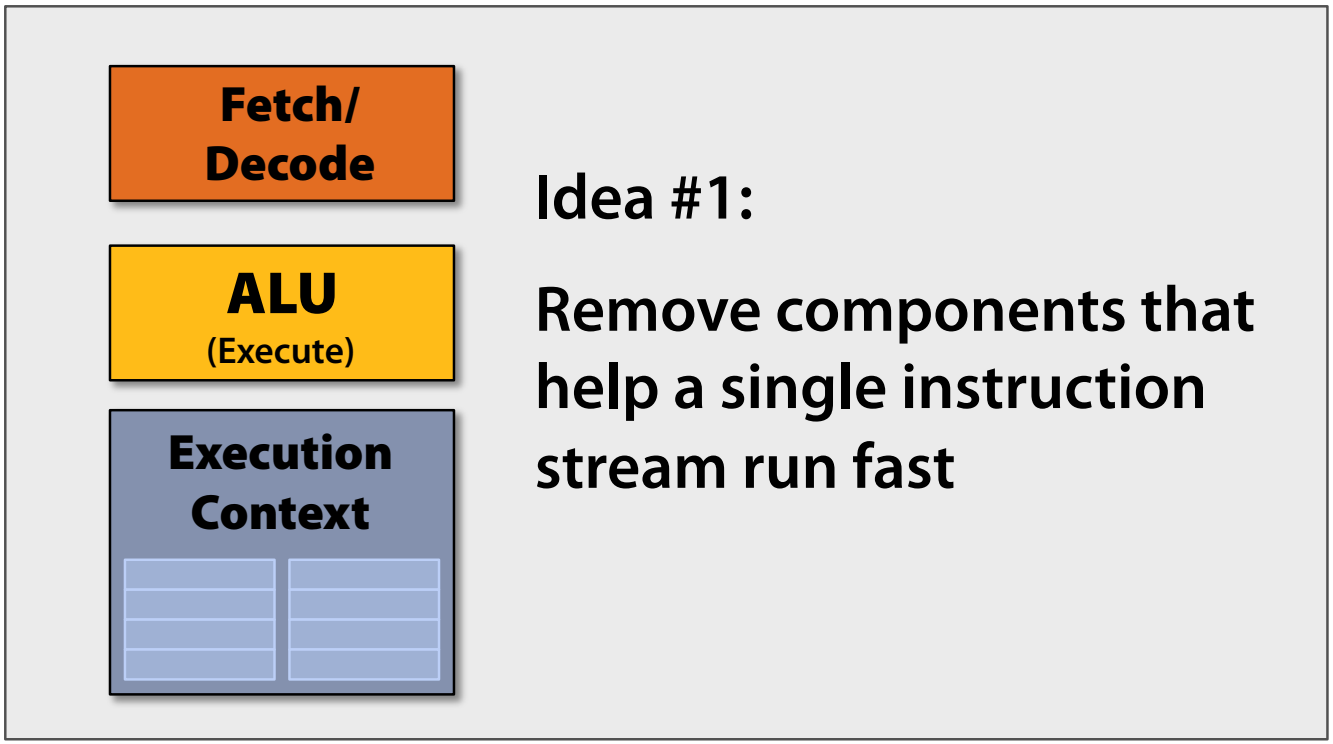
Die floorplan: VIA Isaiah (2008).
65 nm, 4 SP ops at a time, 1 MiB L2.

"CPU-style" Cores



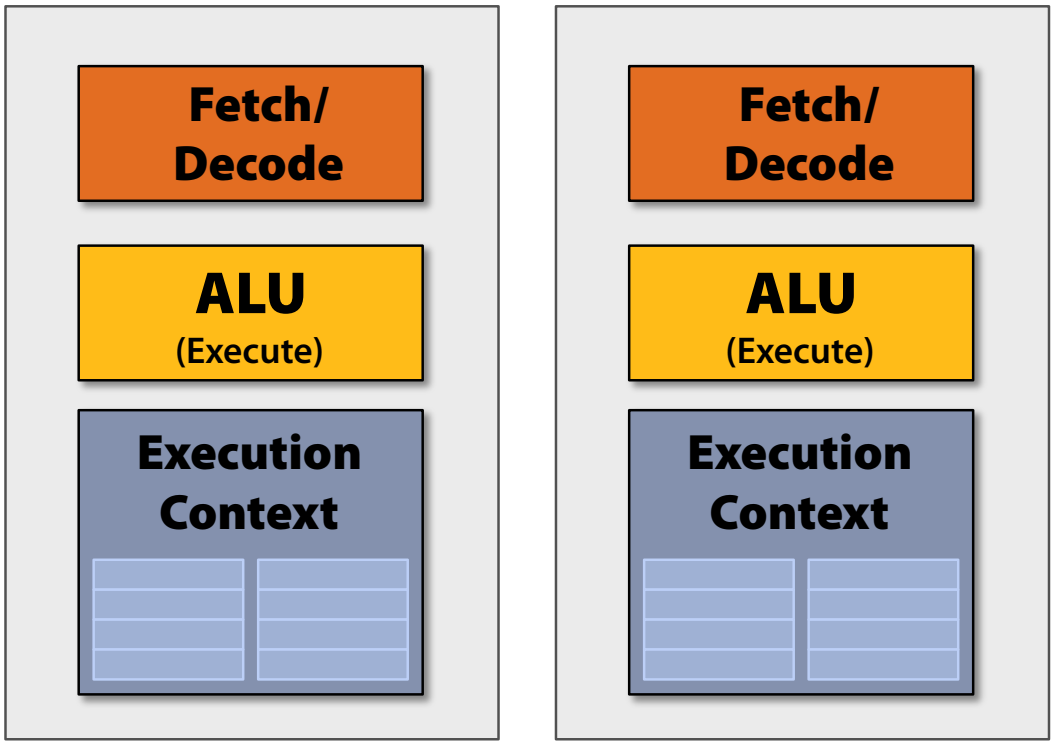
Credit: Kayvon Fatahalian (Stanford)

Slimming down



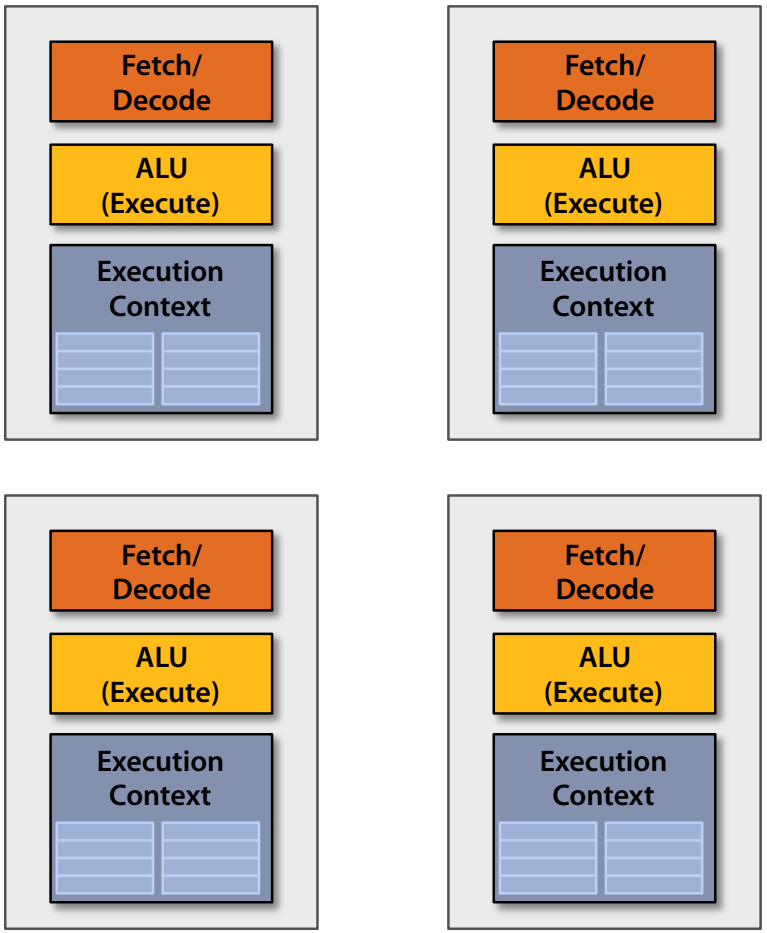
Credit: Kayvon Fatahalian (Stanford)

More Space: Double the Number of Cores



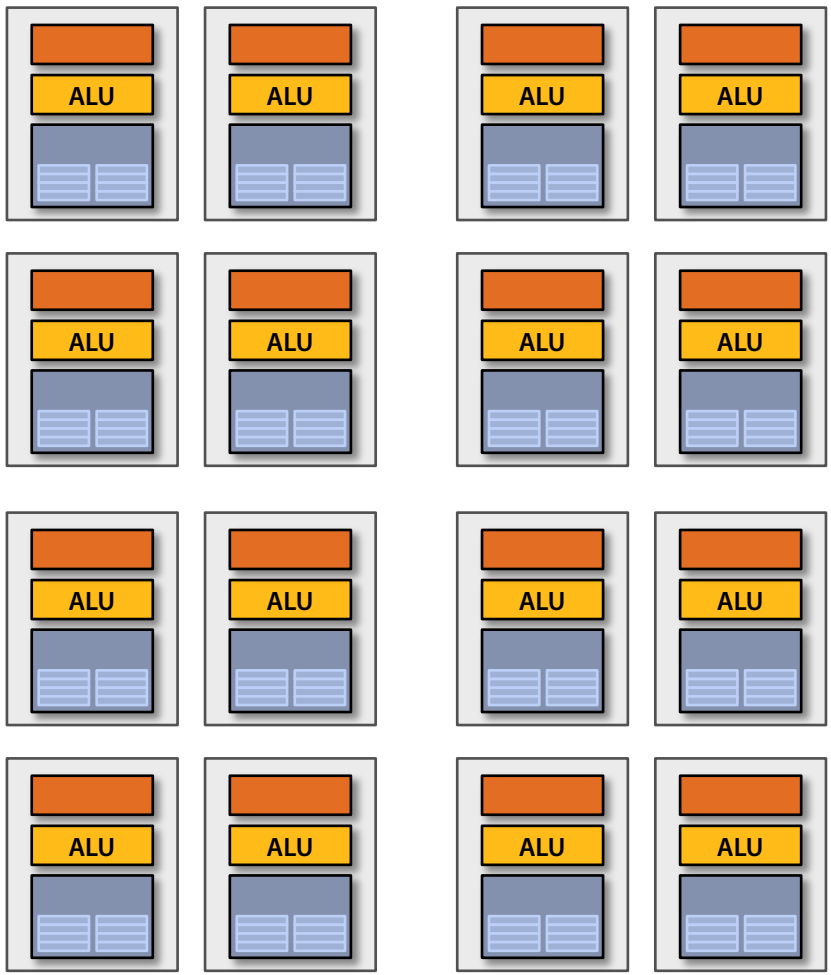
Credit: Kayvon Fatahalian (Stanford)

. . . again



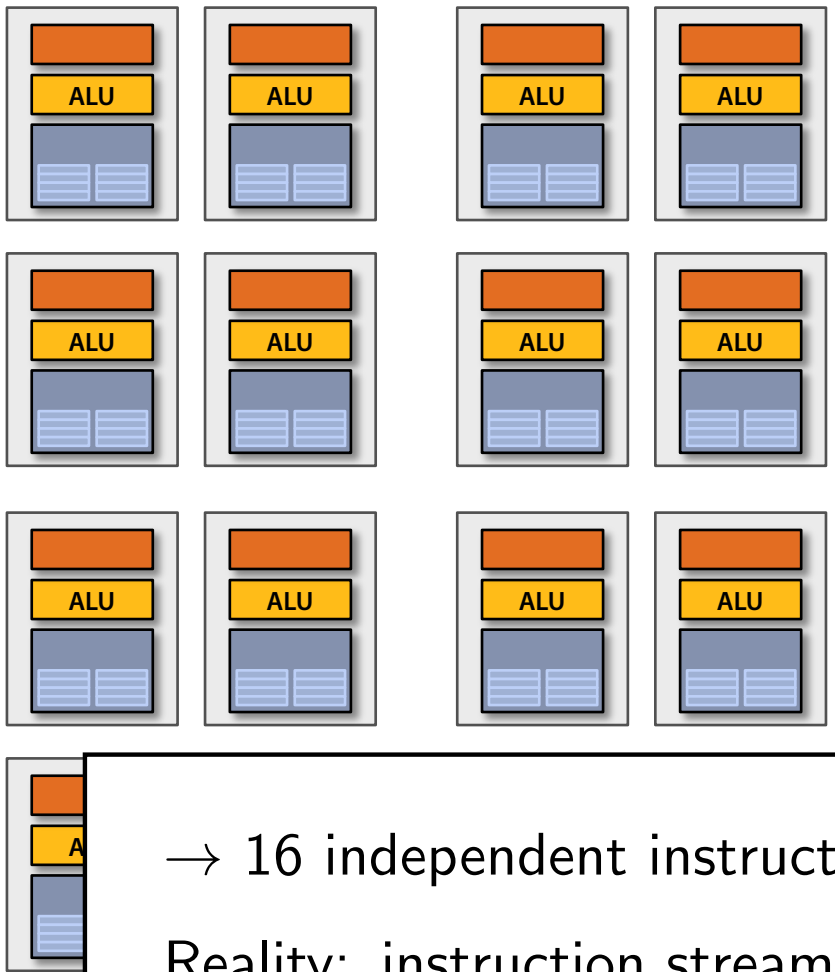
Credit: Kayvon Fatahalian (Stanford)

. . . and again



Credit: Kayvon Fatahalian (Stanford)

. . . and again

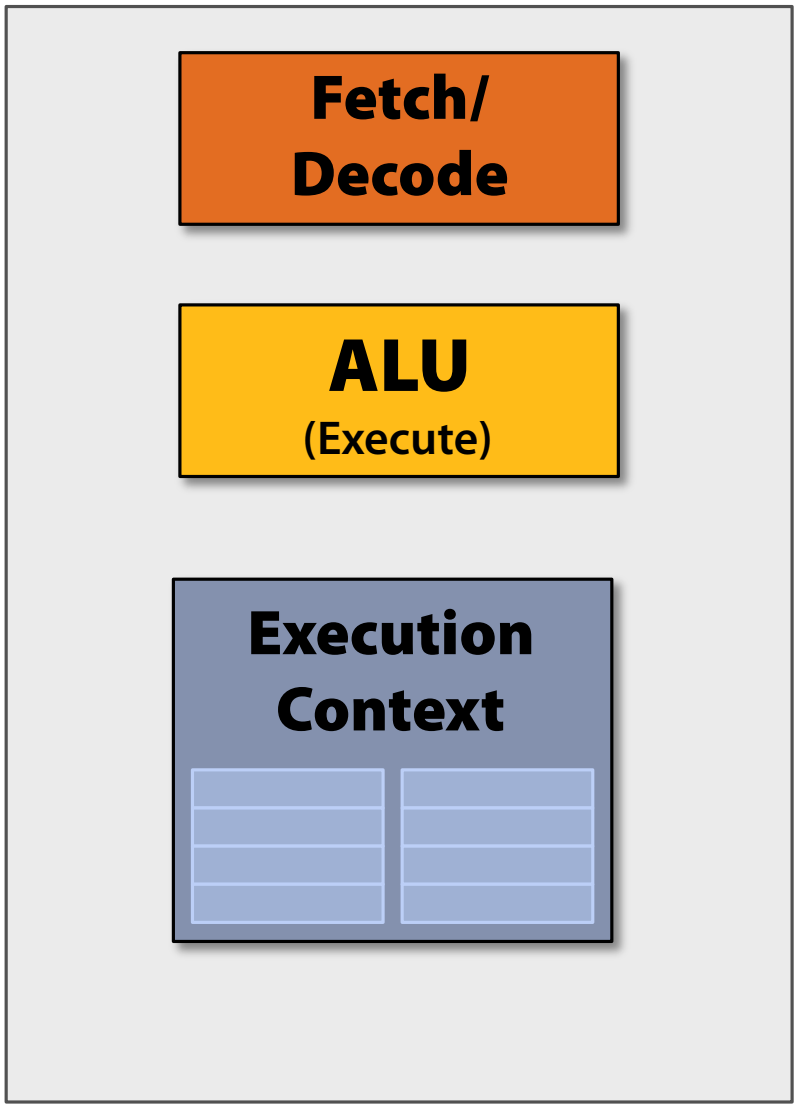


→ 16 independent instruction streams

Reality: instruction streams not actually very different/independent

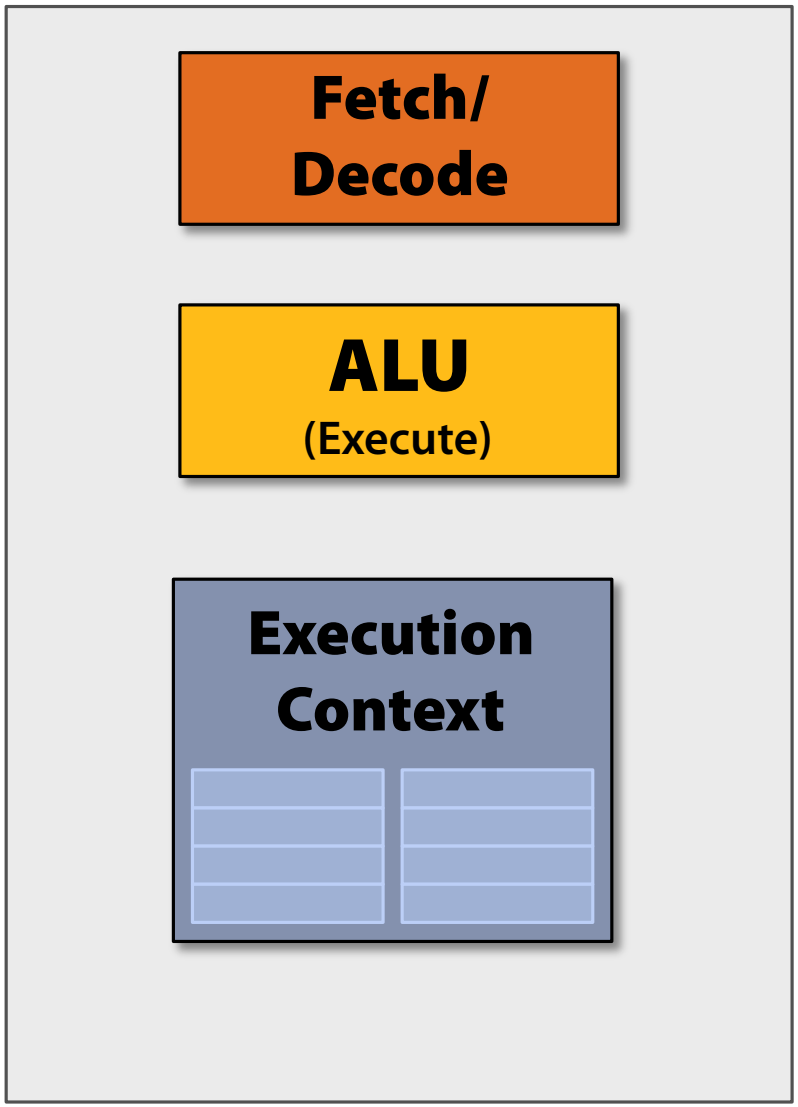
Credit: Kayvon Fatahalian (

Saving Yet More Space



Credit: Kayvon Fatahalian (Stanford)

Saving Yet More Space



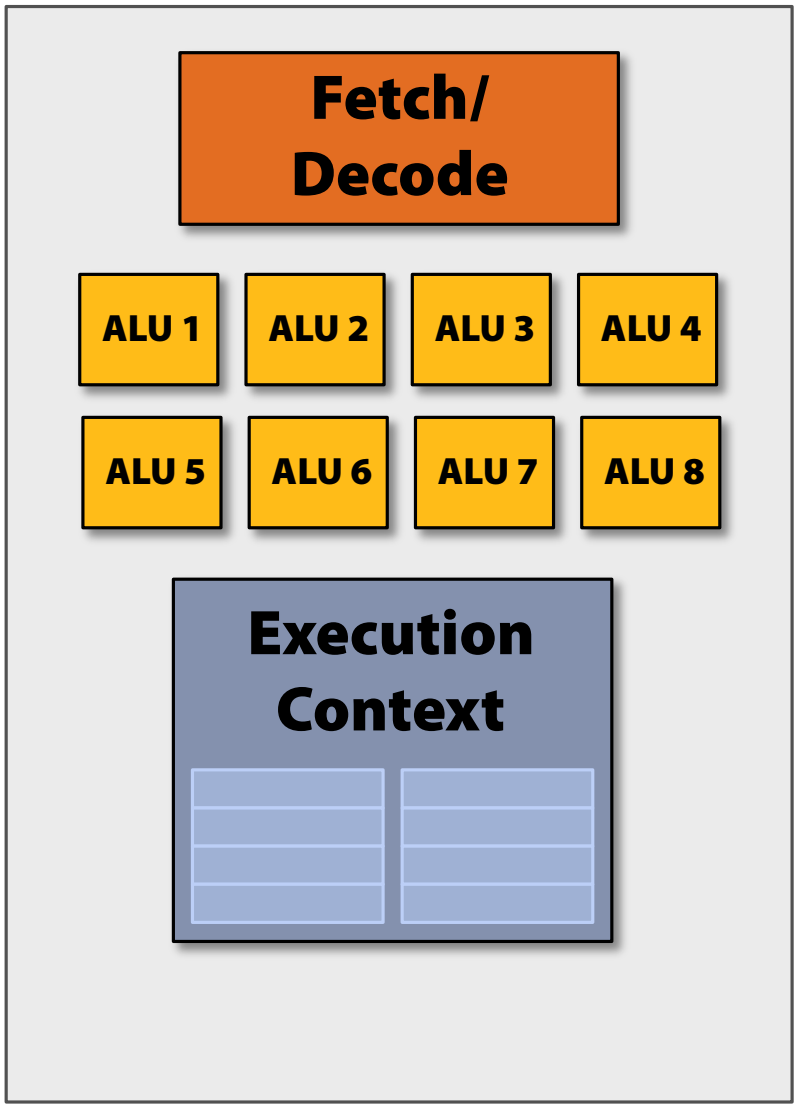
Credit: Kayvon Fatahalian (Stanford)

Idea #2

Amortize cost/complexity of managing an instruction stream across many ALUs

→ **SIMD**

Saving Yet More Space



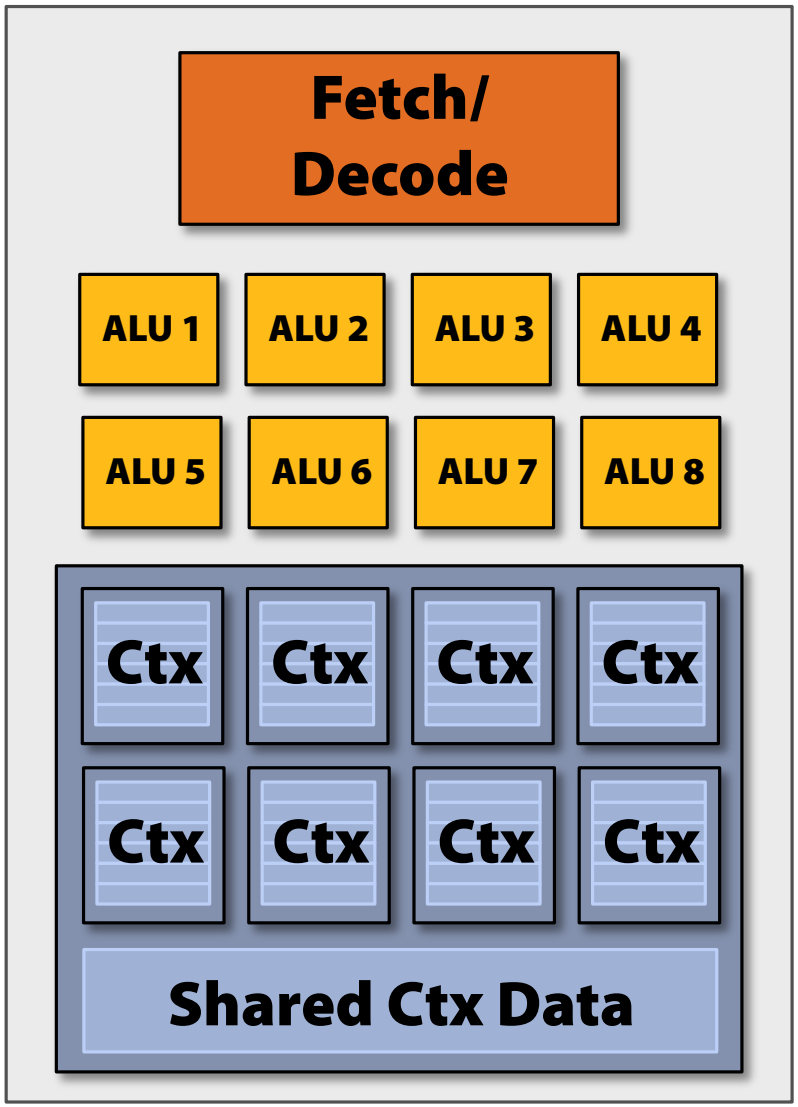
Credit: Kayvon Fatahalian (Stanford)

Idea #2

Amortize cost/complexity of managing an instruction stream across many ALUs

→ **SIMD**

Saving Yet More Space



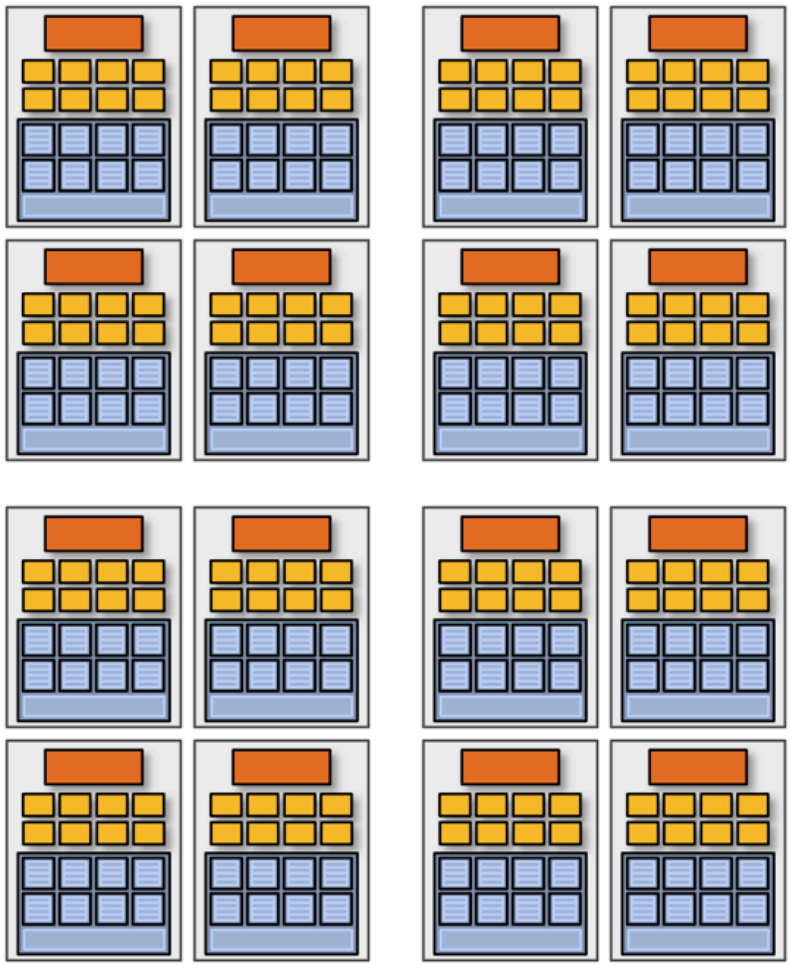
Credit: Kayvon Fatahalian (Stanford)

Idea #2

Amortize cost/complexity of managing an instruction stream across many ALUs

→ **SIMD**

Gratuitous Amounts of Parallelism!



Credit: Kayvon Fatahalian (Stanford)

Gratuitous Amounts of Parallelism!

Example:

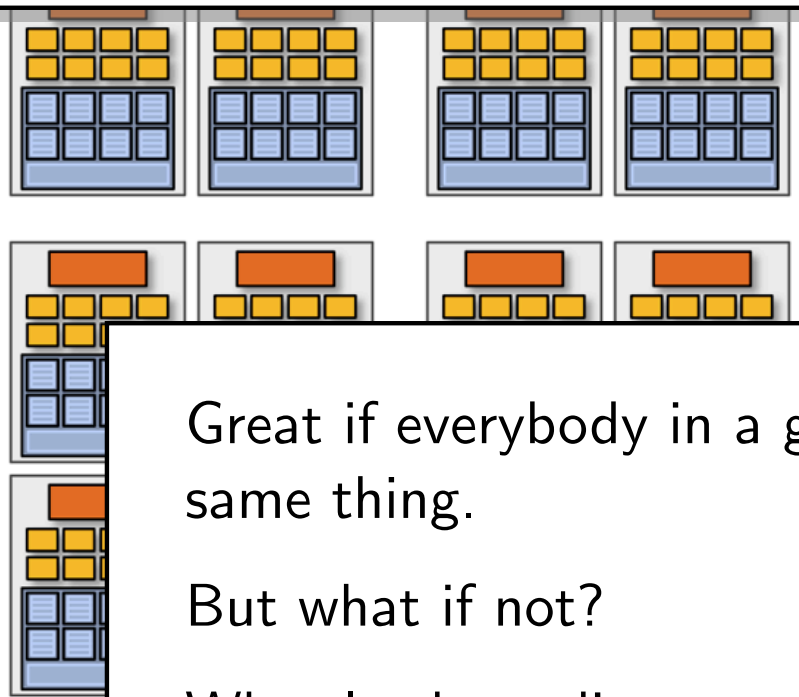
128 instruction streams in parallel
16 independent groups of 8 synchronized streams

Credit: Kayvon Fatahalian (Stanford)

Gratuitous Amounts of Parallelism!

Example:

128 instruction streams in parallel
16 independent groups of 8 synchronized streams



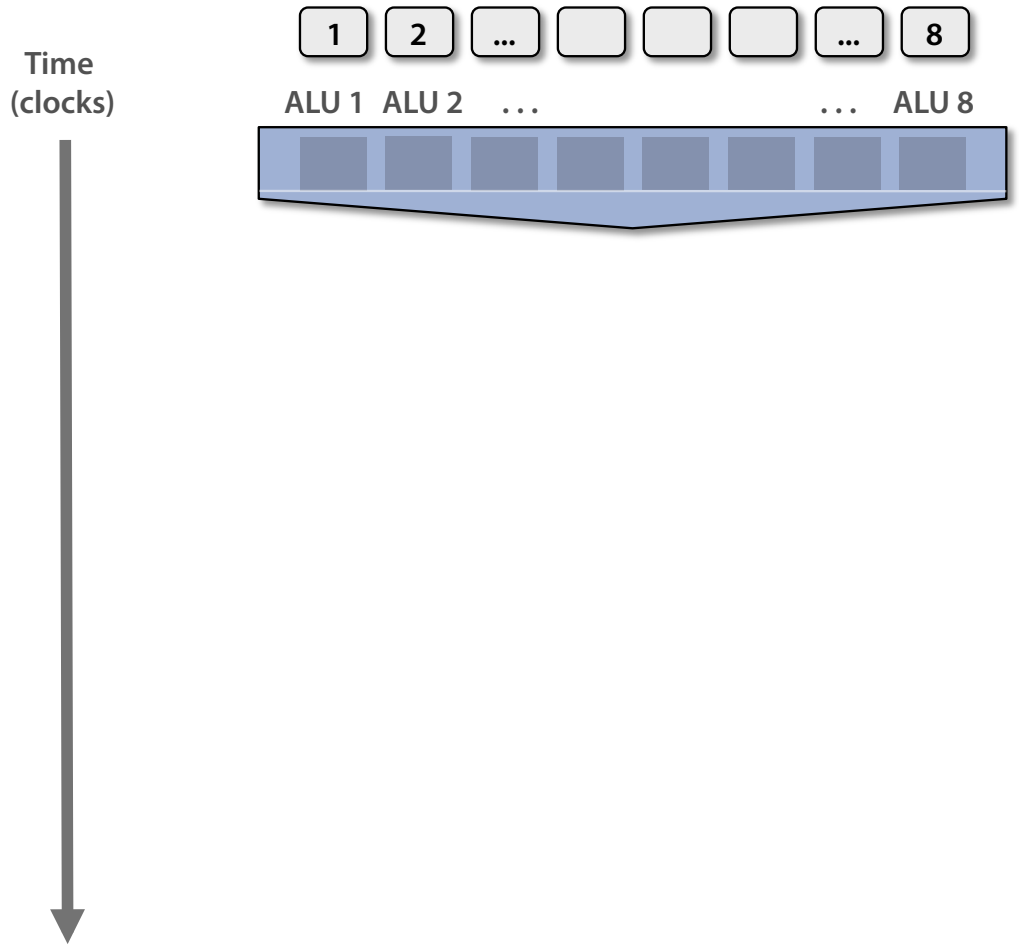
Great if everybody in a group does the same thing.

But what if not?

What leads to divergent instruction streams?

Credit: Kayvon Fatahalian (

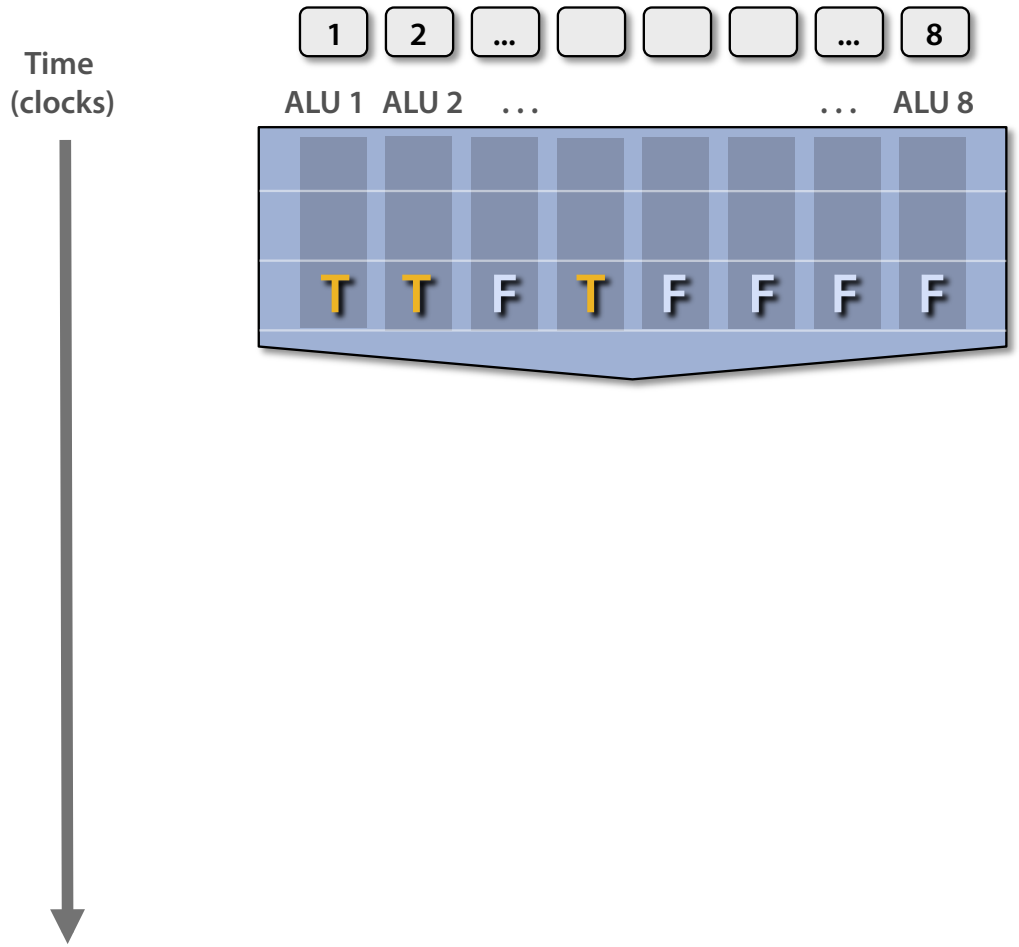
Branches




```
<unconditional
shader code>

if (x > 0) {
    y = pow(x, exp);
    y *= Ks;
    refl = y + Ka;
} else {
    x = 0;
    refl = Ka;
}

<resume unconditional
shader code>
```

Credit: Kayvon Fatahalian (Stanford)

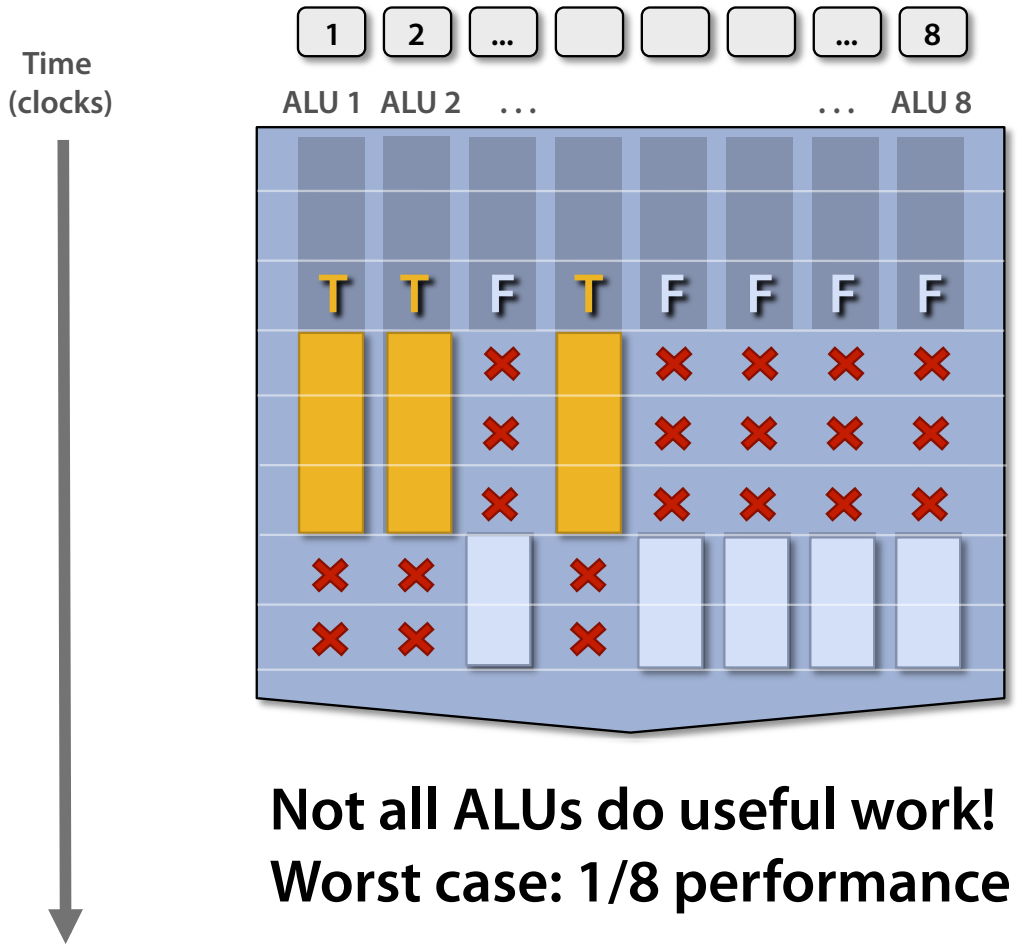
Branches




```
<unconditional
shader code>

if (x > 0) {
    y = pow(x, exp);
    y *= Ks;
    refl = y + Ka;
} else {
    x = 0;
    refl = Ka;
}

<resume unconditional
shader code>
```

Credit: Kayvon Fatahalian (Stanford)

Branches

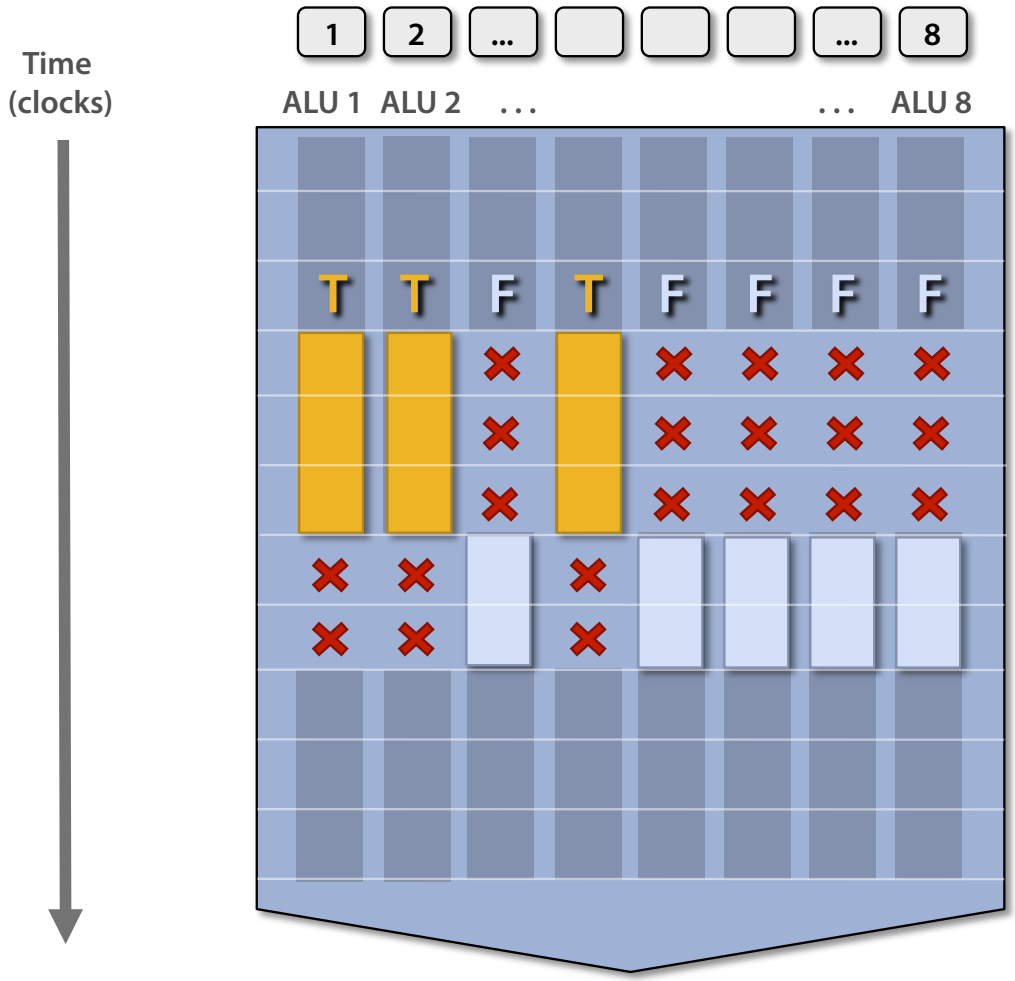


Not all ALUs do useful work!
Worst case: 1/8 performance

```
<unconditional
shader code>

if (x > 0) {
    y = pow(x, exp);
    y *= Ks;
    refl = y + Ka;
} else {
    x = 0;
    refl = Ka;
}

<resume unconditional
shader code>
```

Credit: Kayvon Fatahalian (Stanford)

Branches



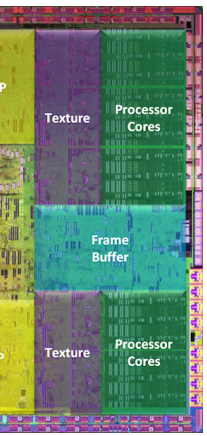

```
<unconditional
shader code>

if (x > 0) {
    y = pow(x, exp);
    y *= Ks;
    refl = y + Ka;
} else {
    x = 0;
    refl = Ka;
}

<resume unconditional
shader code>
```

Credit: Kayvon Fatahalian (Stanford)

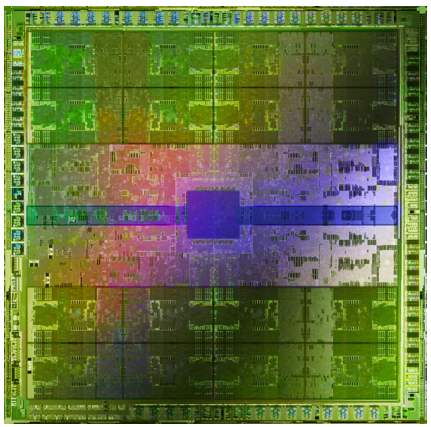
Recent Processor Architecture

- Commodity chips
- "Infinitely" many cores
- "Infinite" vector width
- Must hide memory latency
  ($\rightarrow$ ILP, SMT)

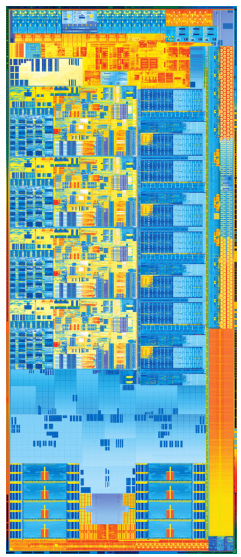
- Compute bandwidth
  $\gg$ Memory bandwidth
- Bandwidth only achievable
  by *homogeneity*



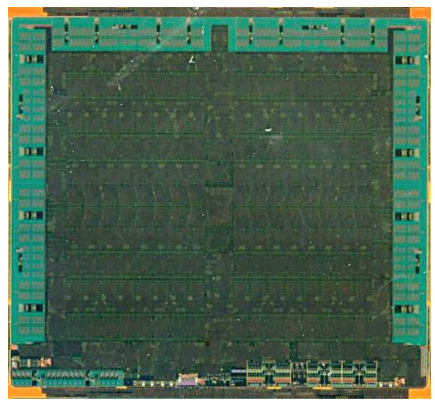
T200
08)



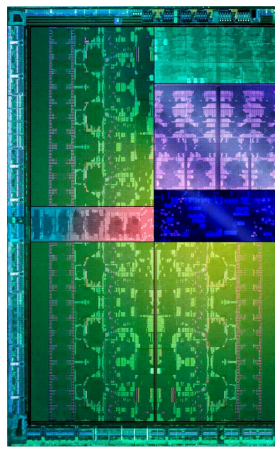
Nv Fermi
(2010)



Intel IVB
(2012)



AMD Tahiti
(2012)

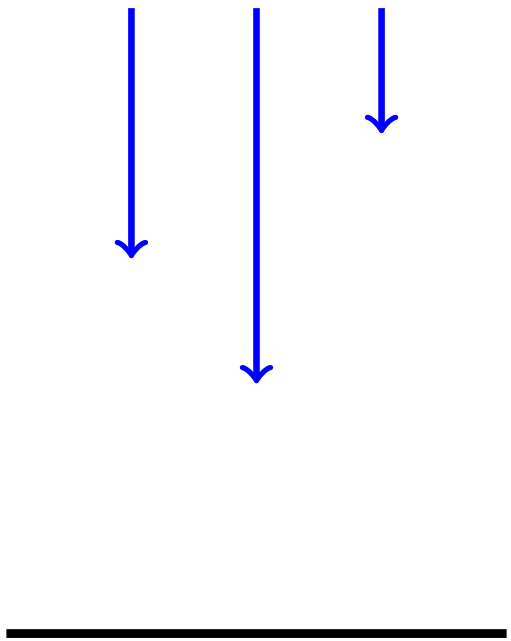


Nv GK1
(2012

Outline

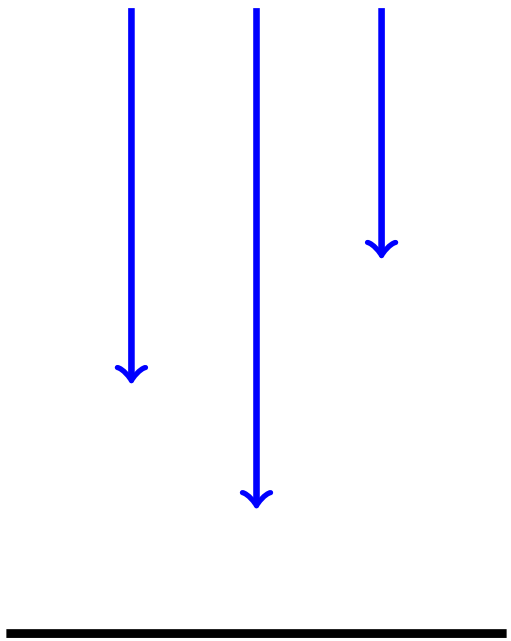
Synchronization

What is a Barrier?



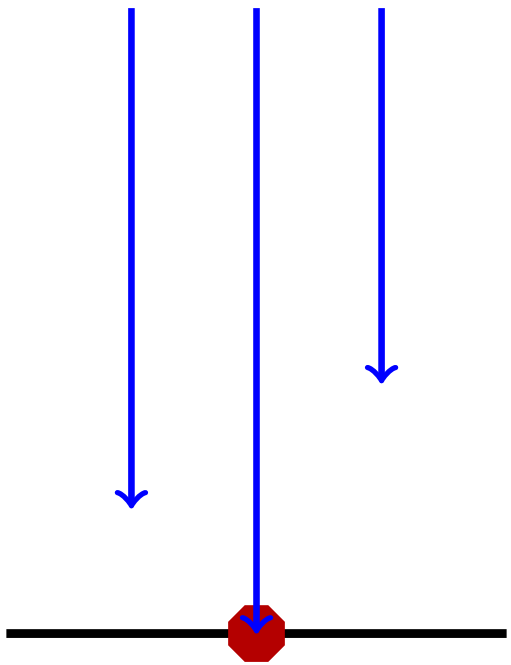
Synchronization

What is a Barrier?



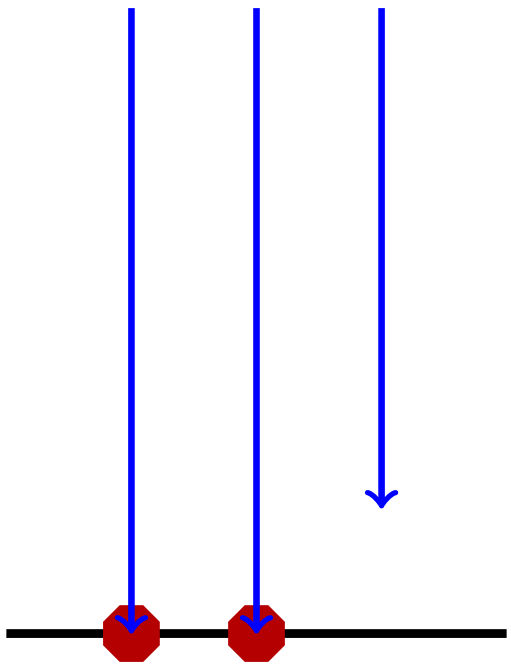
Synchronization

What is a Barrier?



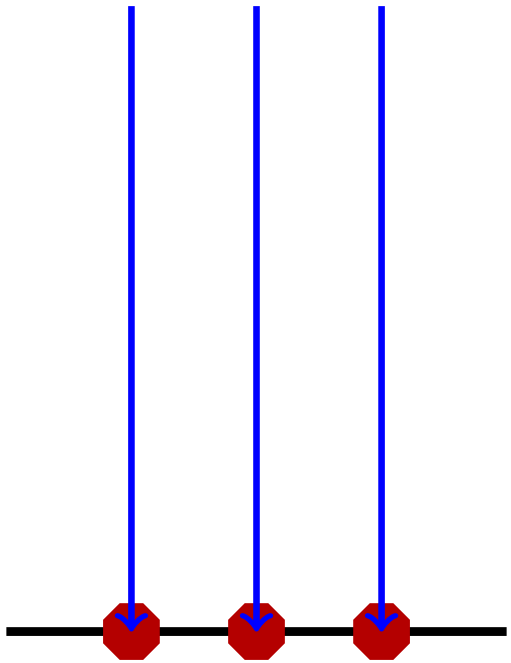
Synchronization

What is a Barrier?



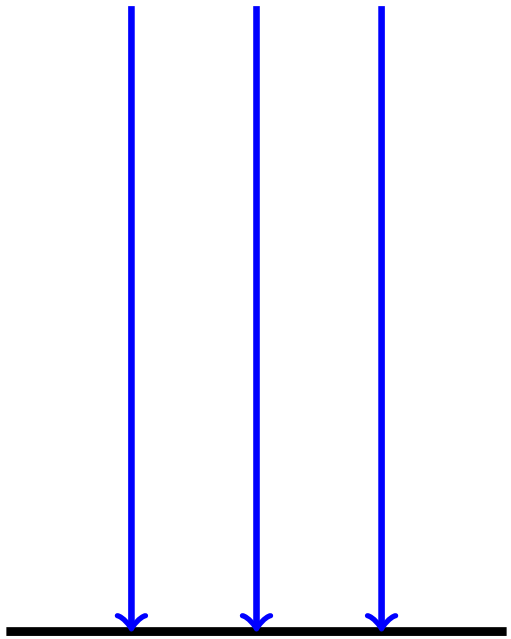
Synchronization

What is a Barrier?



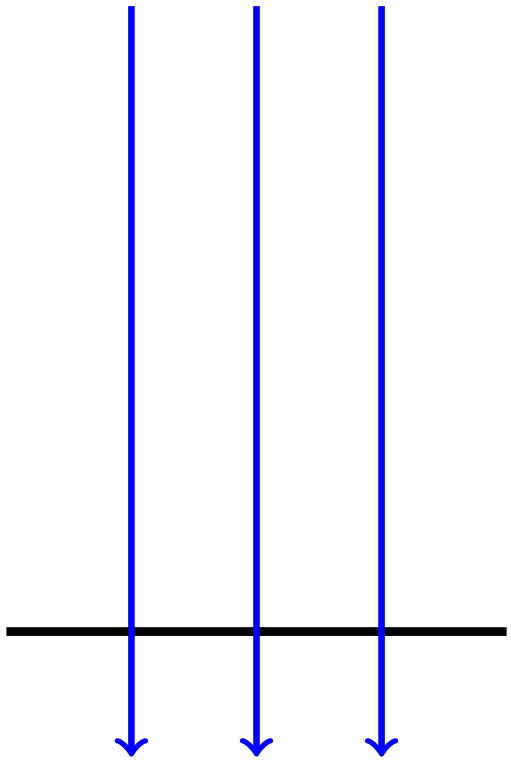
Synchronization

What is a Barrier?



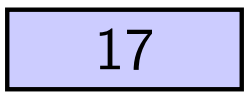
Synchronization

What is a Barrier?



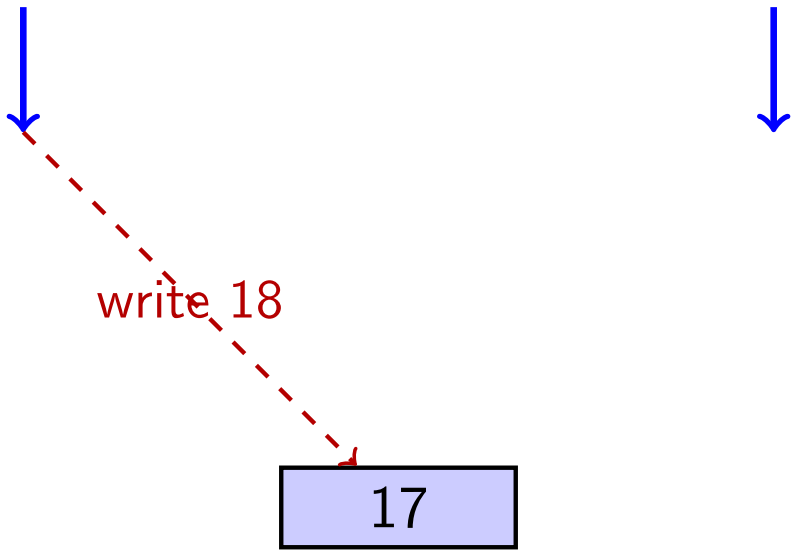
Synchronization

What is a Memory Fence?



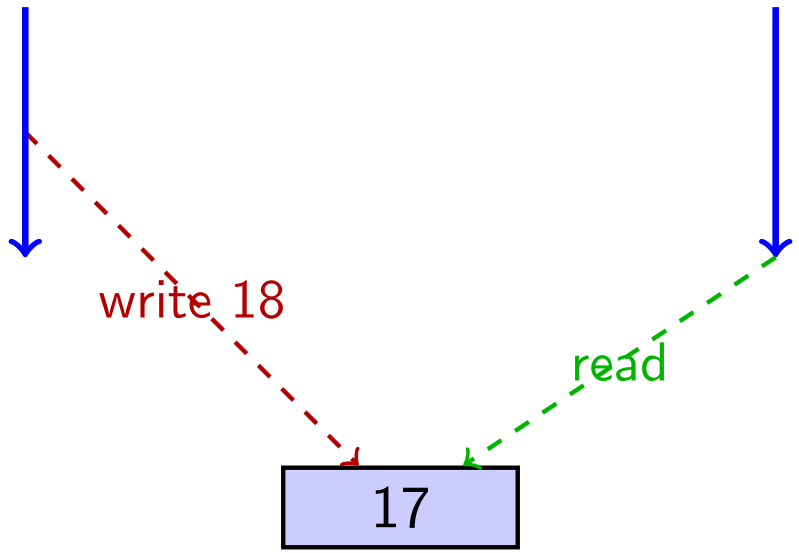
Synchronization

What is a Memory Fence?



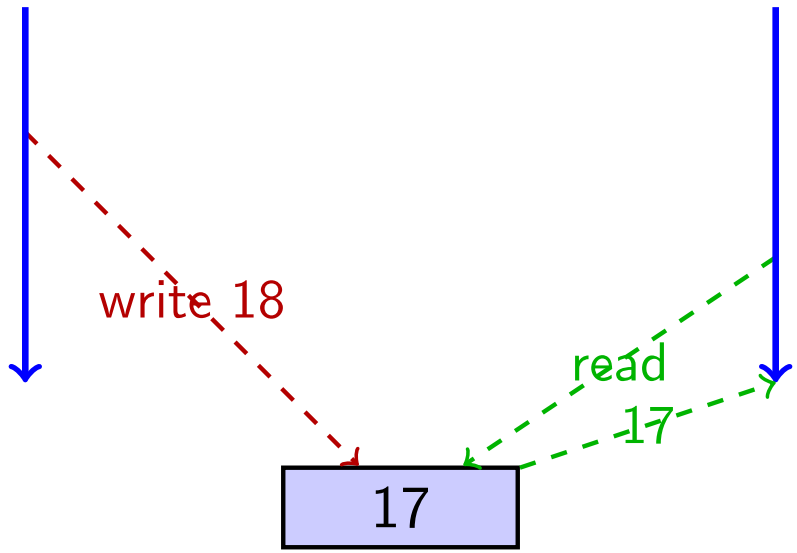
Synchronization

What is a Memory Fence?



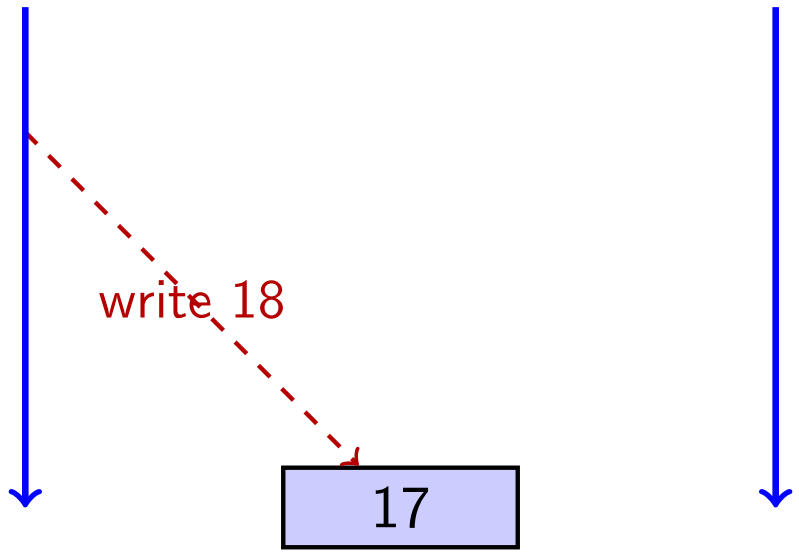
Synchronization

What is a Memory Fence?



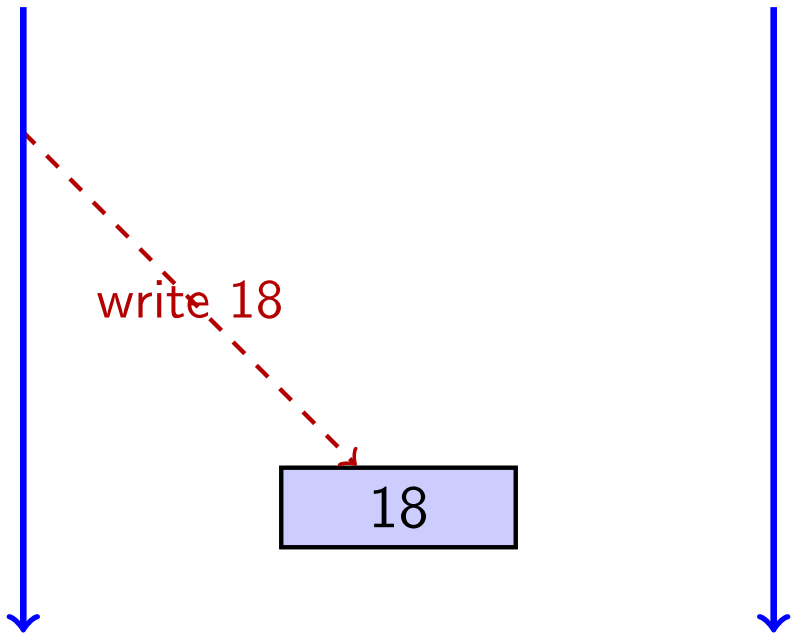
Synchronization

What is a Memory Fence?



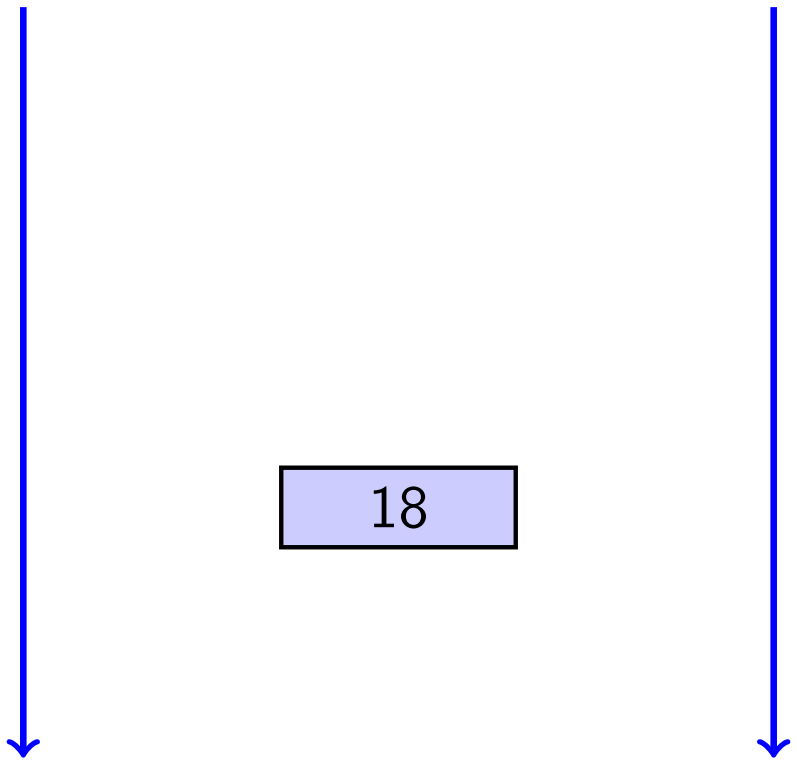
Synchronization

What is a Memory Fence?



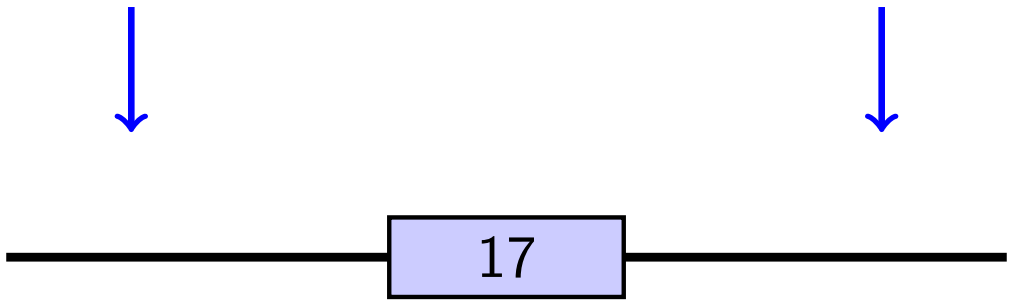
Synchronization

What is a Memory Fence?



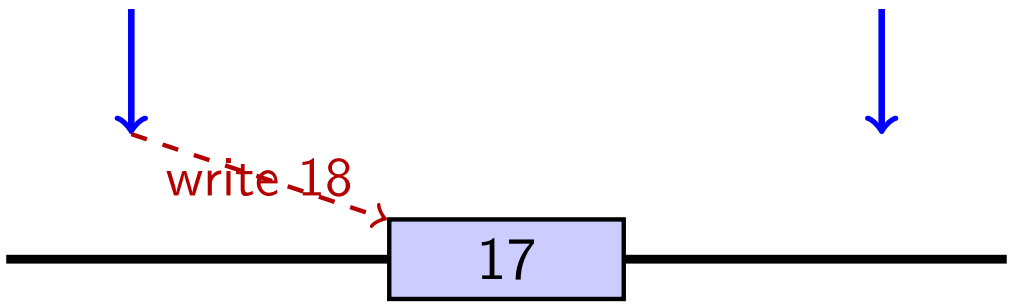
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



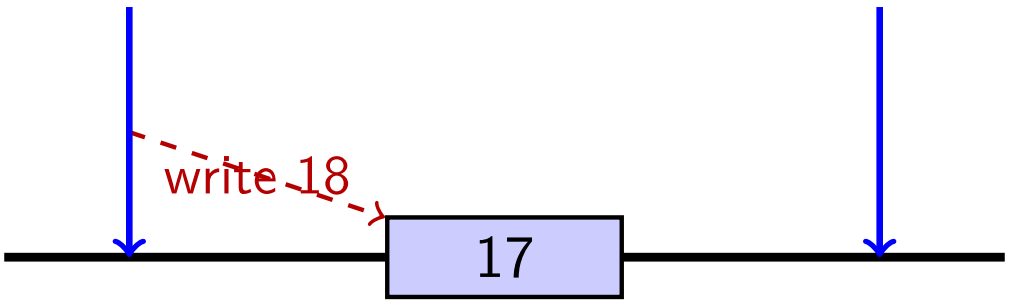
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



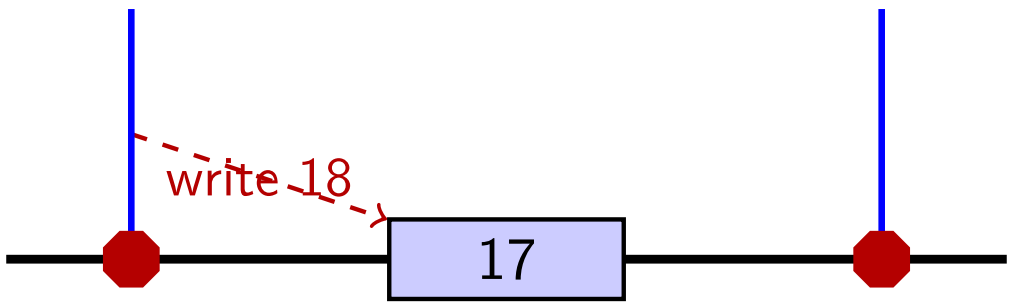
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



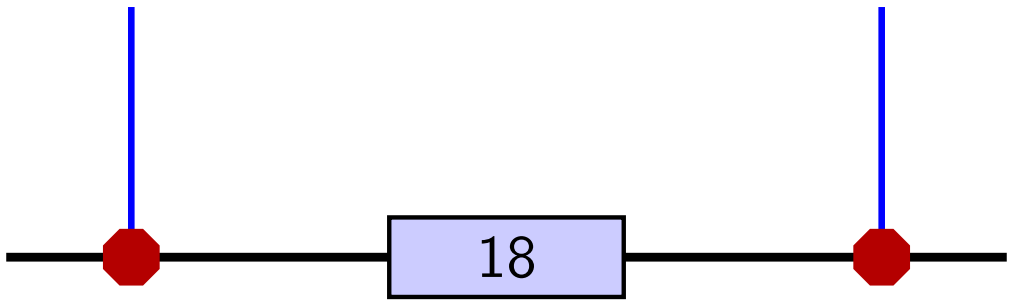
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



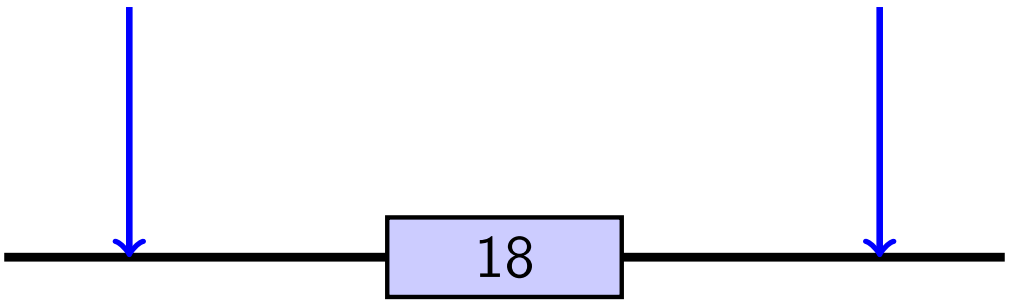
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



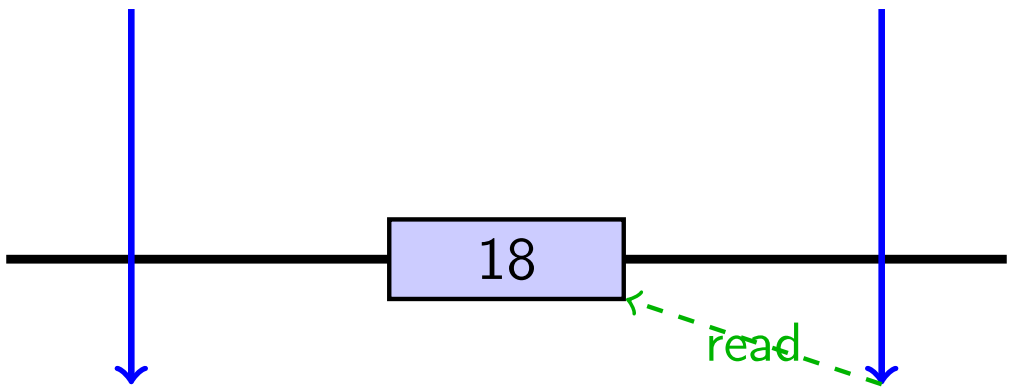
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



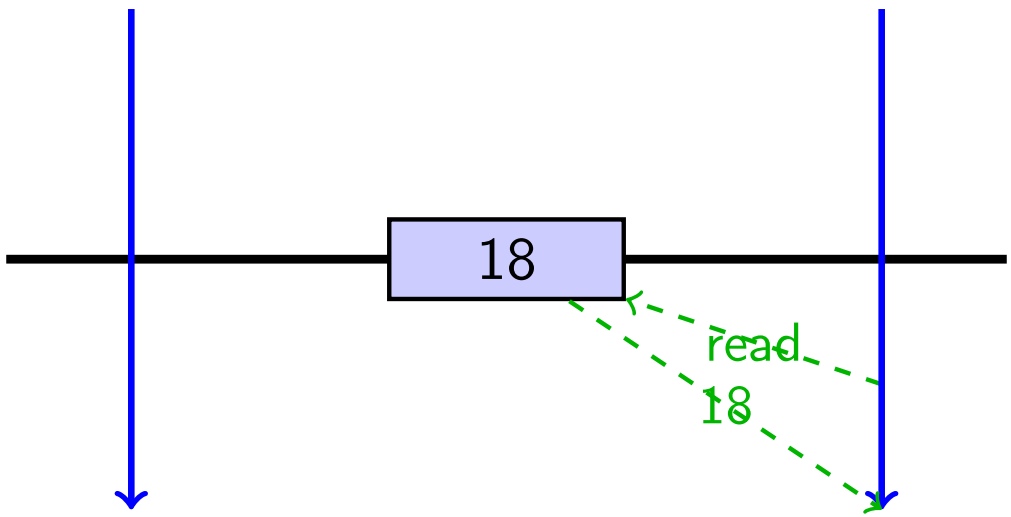
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



Synchronization

What is a Memory Fence? An ordering restriction for memory access.

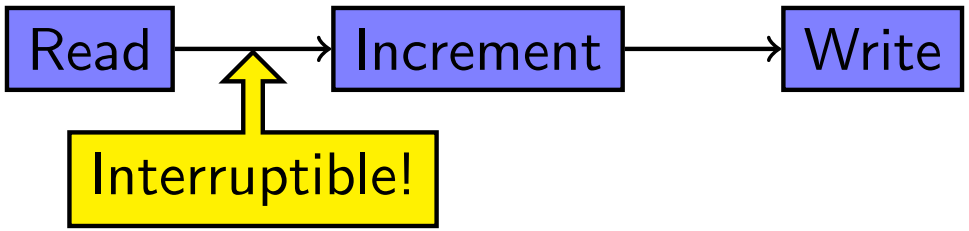


Atomic Operations

Collaborative (inter-block) Global Memory Update:

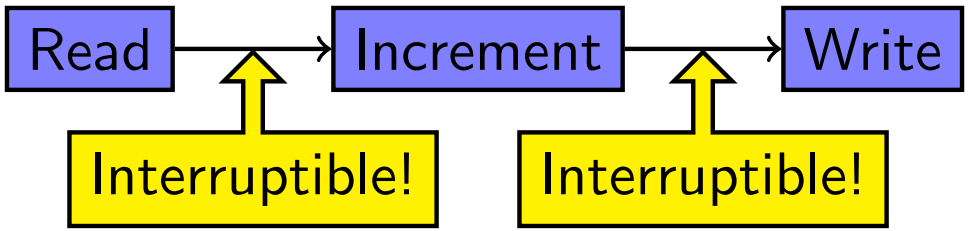
Atomic Operations

Collaborative (inter-block) Global Memory Update:



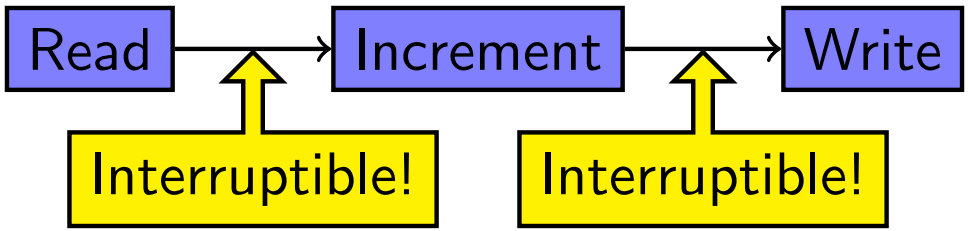
Atomic Operations

Collaborative (inter-block) Global Memory Update:

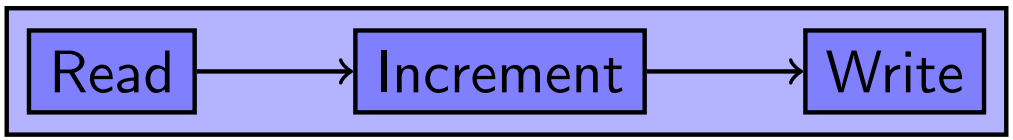


Atomic Operations

Collaborative (inter-block) Global Memory Update:

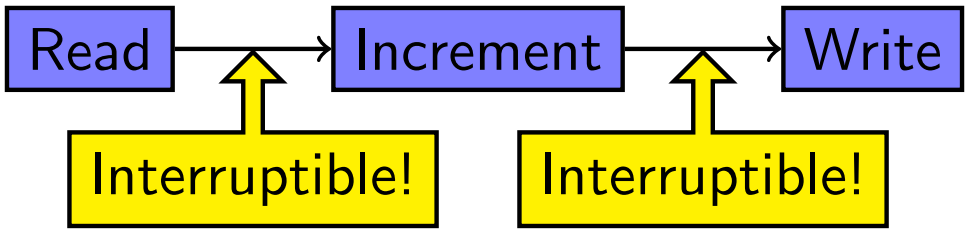


Atomic Global Memory Update:

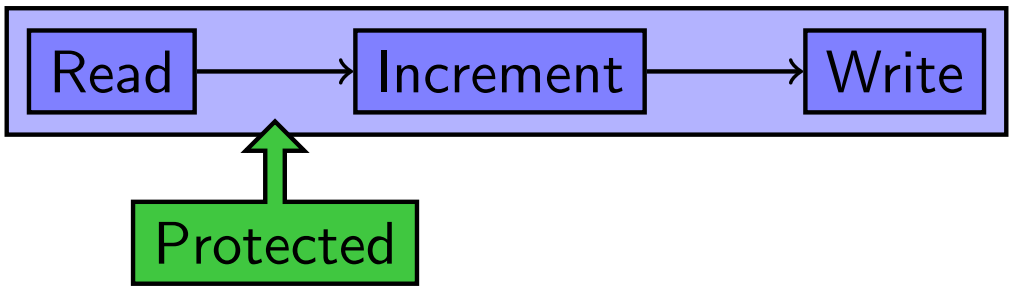


Atomic Operations

Collaborative (inter-block) Global Memory Update:

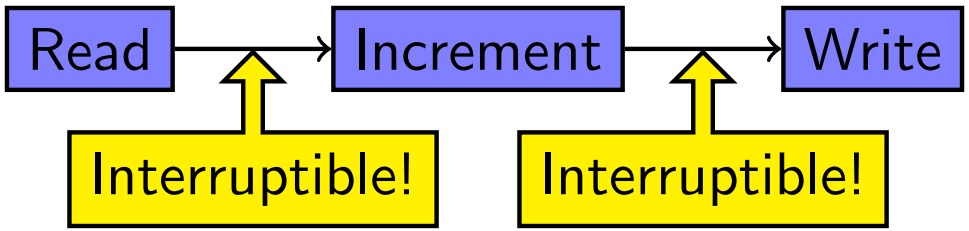


Atomic Global Memory Update:

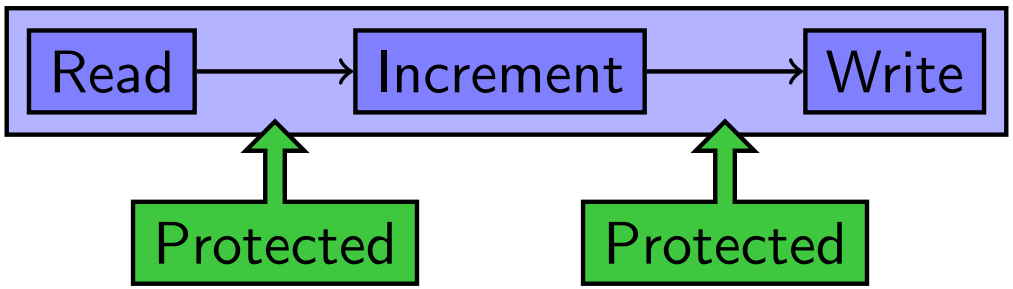


Atomic Operations

Collaborative (inter-block) Global Memory Update:

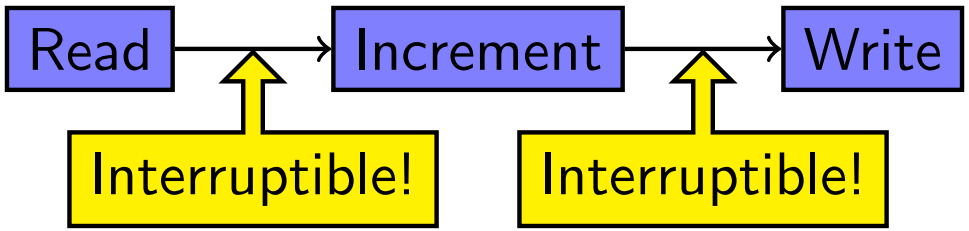


Atomic Global Memory Update:

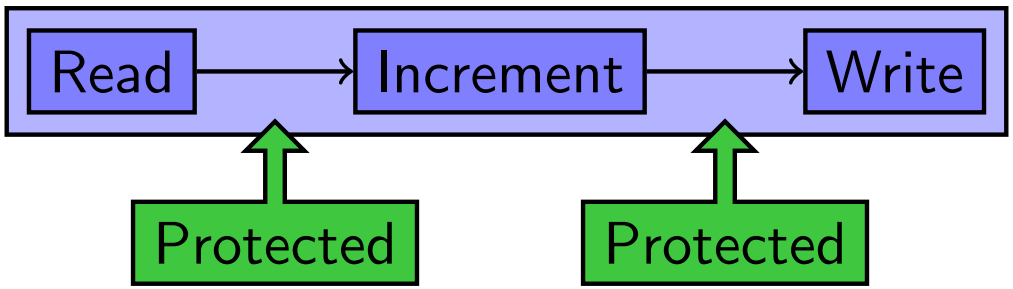


Atomic Operations

Collaborative (inter-block) Global Memory Update:



Atomic Global Memory Update:



How?

atomic_{add,inc,cmpxchg,...}(int *global, int value);

Questions?

?

Image Credits

- Isaiah die shot: VIA Technologies