

Digital Design & Computer Arch.

Lecture 18c: Decoupled Access-Execute

Prof. Onur Mutlu

ETH Zürich

Spring 2023

4 May 2023

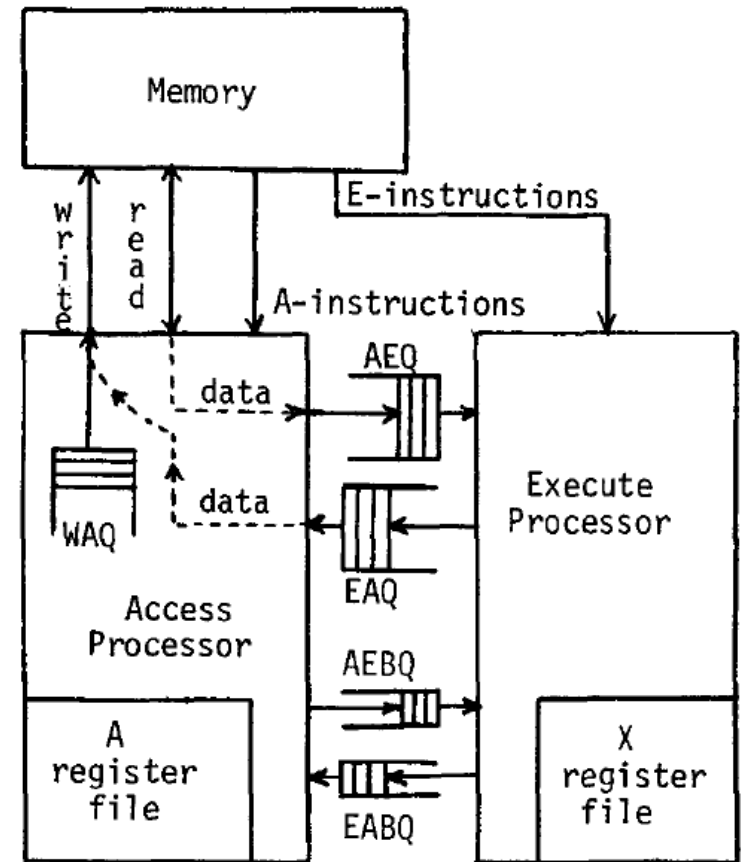
Approaches to (Instruction-Level) Concurrency

- Pipelining
- Fine-Grained Multithreading
- Out-of-order Execution
- Dataflow (at the ISA level)
- Superscalar Execution
- VLIW
- Systolic Arrays
- Decoupled Access Execute
- SIMD Processing (Vector and array processors, GPUs)

Decoupled Access/Execute (DAE)

Decoupled Access/Execute (DAE)

- Motivation: **Tomasulo's algorithm too complex** to implement
 - 1980s before Pentium Pro
- Idea: Decouple operand access and execution via **two separate instruction streams that communicate via ISA-visible queues**.
- Smith, “[Decoupled Access/Execute Computer Architectures](#),” ISCA 1982, ACM TOCS 1984.



Decoupled Access/Execute (II)

- Compiler generates two instruction streams (A and E)
 - Synchronizes the two upon control flow instructions (using branch queues)

```

q = 0.0
Do 1 k = 1, 400
1  x(k) = q + y(k) * (r * z(k+10) + t * z(k+11))
    
```

Fig. 2a. Lawrence Livermore Loop 1 (HYDRO EXCERPT)

	A7 ← -400	. negative loop count
	A2 ← 0	. initialize index
	A3 ← 1	. index increment
	X2 ← r	. load loop invariants
	X5 ← t	. into registers
loop:	X3 ← z + 10, A2	. load z(k+10)
	X7 ← z + 11, A2	. load z(k+11)
	X4 ← X2 *f X3	. r*z(k+10)-flt. mult.
	X3 ← X5 *f X7	. t * z(k+11)
	X7 ← y, A2	. load y(k)
	X6 ← X3 +f X4	. r*z(x+10)+t*z(k+11))
	X4 ← X7 *f X6	. y(k) * (above)
	A7 ← A7 + 1	. increment loop counter
	x, A2 ← X4	. store into x(k)
	A2 ← A2 + A3	. increment index
	JAM loop	. Branch if A7 < 0

Fig. 2b. Compilation onto CRAY-1-like architecture

<u>Access</u>	<u>Execute</u>
.	
.	
.	
AEQ ← z + 10, A2	X4 ← X2 *f AEQ
AEQ ← z + 11, A2	X3 ← X5 *f AEQ
AEQ ← y, A2	X6 ← X3 +f X4
A7 ← A7 + 1	EAQ ← AEQ *f X6
x, A2 ← EAQ	.
A2 ← A2+ A3	.
.	.
.	
.	

Fig. 2c. Access and execute programs for straight-line section of loop

Decoupled Access/Execute (III)

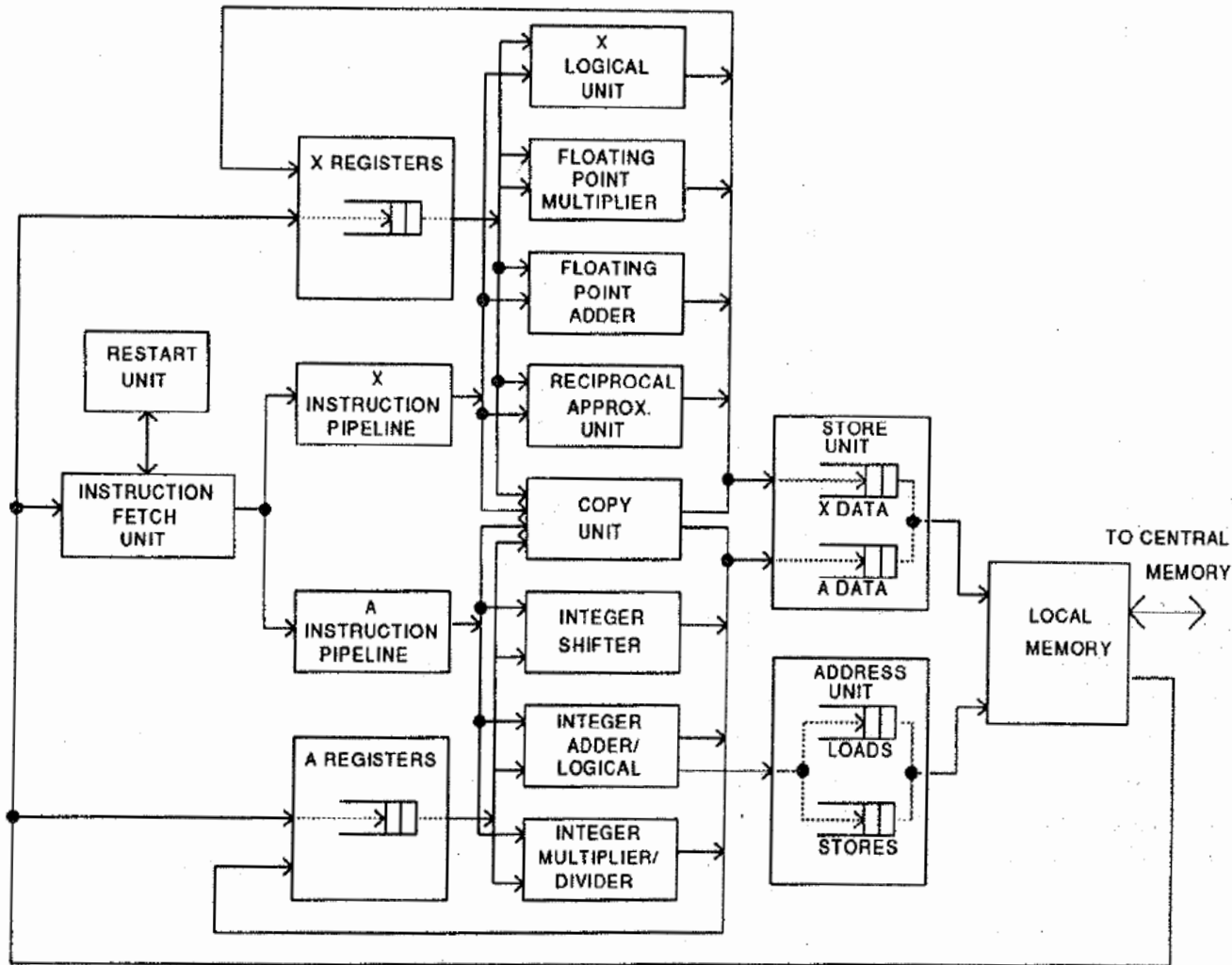
■ Advantages:

- + Execute stream can run ahead of the access stream and vice versa
 - + If A is waiting for memory, E can perform useful work
 - + If A hits in cache, it supplies data to lagging E
 - + Queues reduce the number of required registers
- + Limited out-of-order execution without wakeup/select complexity

■ Disadvantages:

- **Compiler support** to partition the program and manage queues
 - Determines the amount of decoupling
- Branch instructions require synchronization between A and E
- Multiple instruction streams (can be done with a single one, though)

Astronautics ZS-1




- Single stream steered into A and X pipelines
- Each pipeline in-order
- Smith et al., “[The ZS-1 central processor](#),” ASPLOS 1987.
- Smith, “[Dynamic Instruction Scheduling and the Astronautics ZS-1](#),” IEEE Computer 1989.

Loop Unrolling to Eliminate Branches

```
for (int i = 0; i < N; i++){  
  
    A[i] = A[i] + B[i];  
  
}
```

```
for (int i = 0; i < N; i+=4){  
  
    A[i]   = A[i]   + B[i];  
    A[i+1] = A[i+1] + B[i+1];  
    A[i+2] = A[i+2] + B[i+2];  
    A[i+3] = A[i+3] + B[i+3];  
  
}
```



- Idea: Replicate loop body multiple times within an iteration
- + Reduces loop maintenance overhead
 - Induction variable increment or loop condition test
- + Enlarges basic block (and analysis scope)
 - Enables code optimization and scheduling opportunities
- What if iteration count not a multiple of unroll factor? (need extra code to detect this)
- Increases code size

A Modern DAE Example: Pentium 4

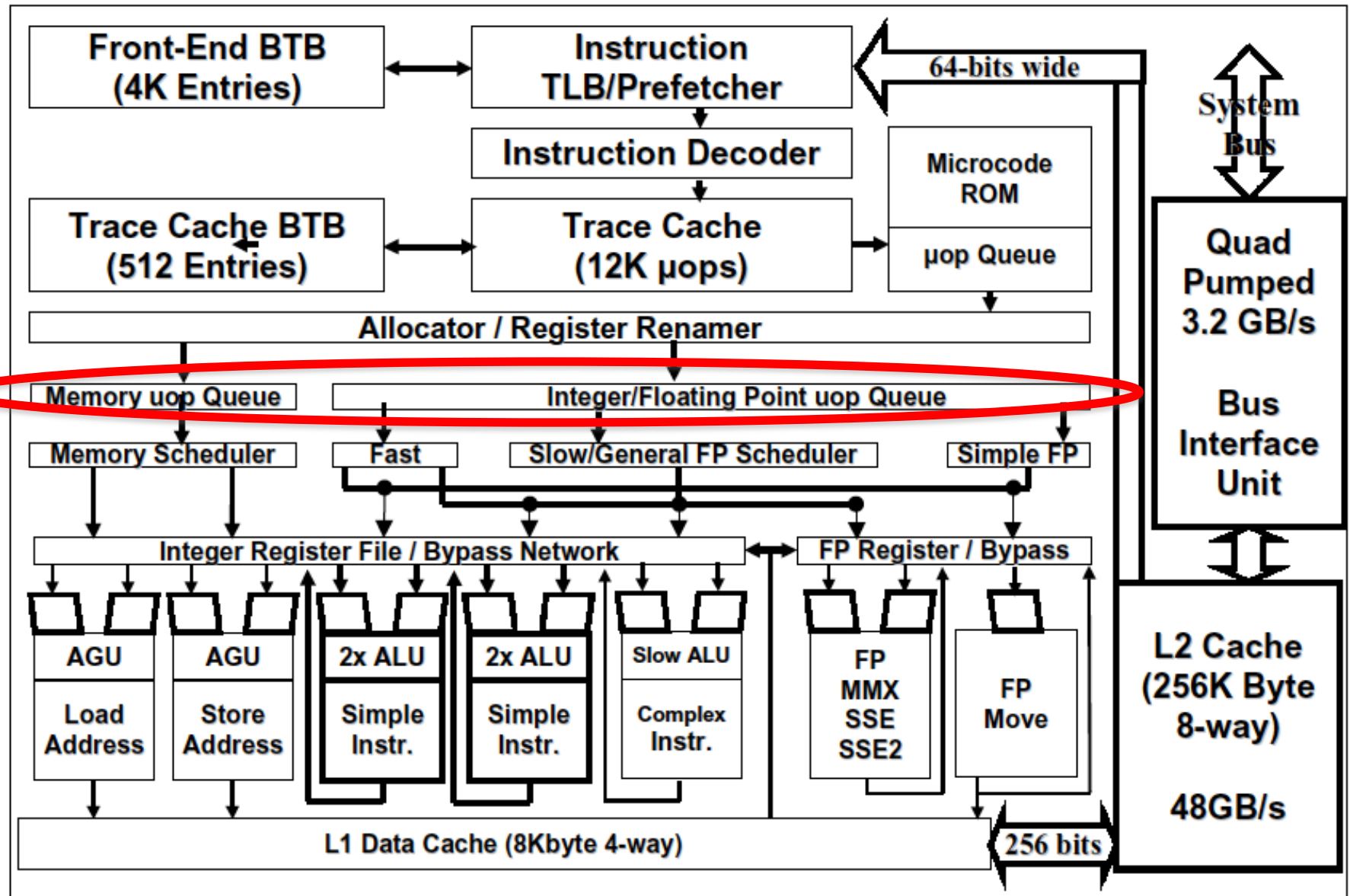
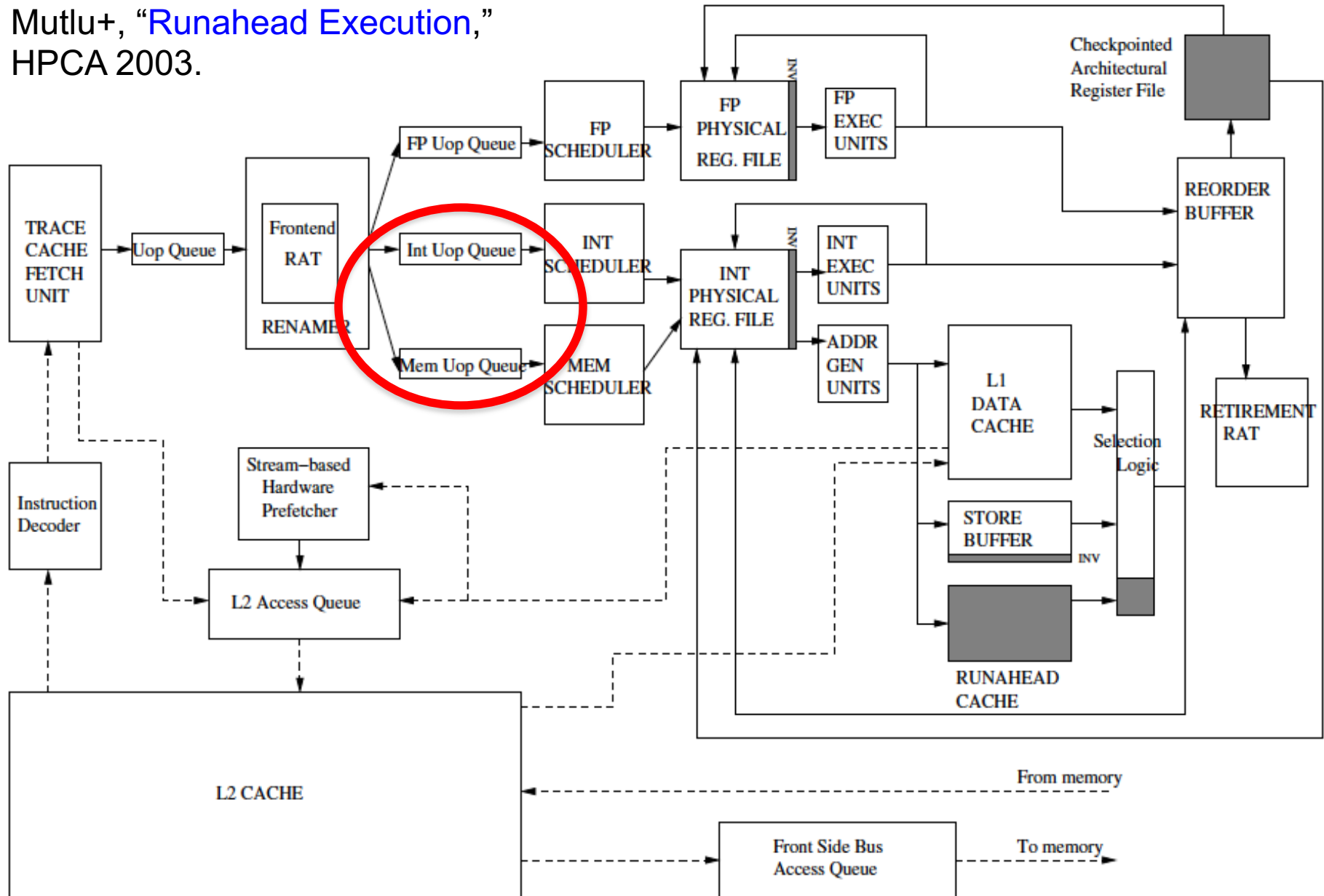


Figure 4: Pentium[®] 4 processor microarchitecture

Intel Pentium 4 Simplified

Mutlu+, “Runahead Execution,”
HPCA 2003.



Approaches to (Instruction-Level) Concurrency

- Pipelining
- Fine-Grained Multithreading
- Out-of-order Execution
- Dataflow (at the ISA level)
- Superscalar Execution
- VLIW
- Systolic Arrays
- Decoupled Access Execute
- SIMD Processing (Vector and array processors, GPUs)

Digital Design & Computer Arch.

Lecture 18c: Decoupled Access-Execute

Prof. Onur Mutlu

ETH Zürich

Spring 2023

4 May 2023