Threads and Concurrency

ECE 373

Threads of Execution

- Thread – smallest unit of processing that can be scheduled by the OS
  - UNIX-like systems are typically process oriented
  - WinNT-like systems are thread oriented

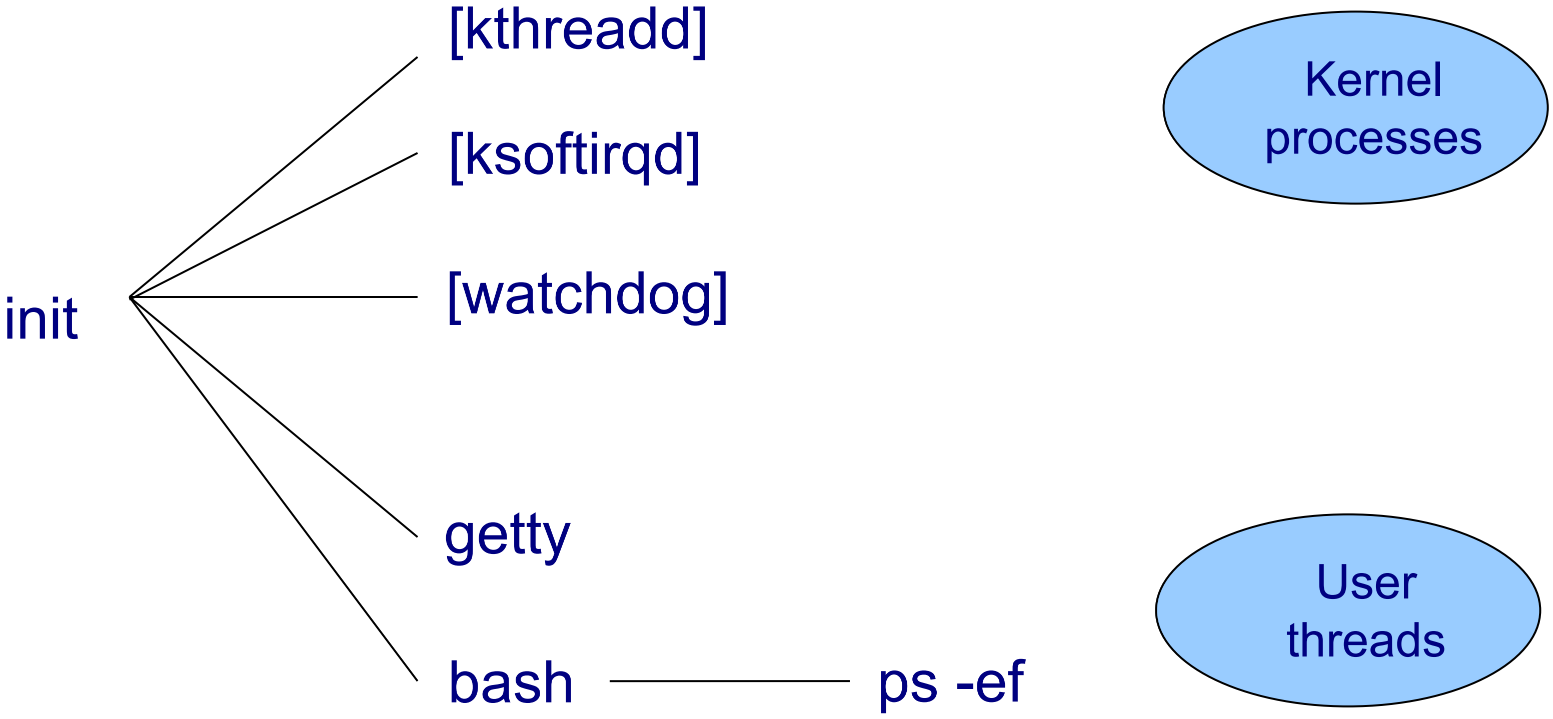
- Single CPU core
  - single line of instructions at any one time

- Multiple CPU cores
  - multiple simultaneous threads of execution

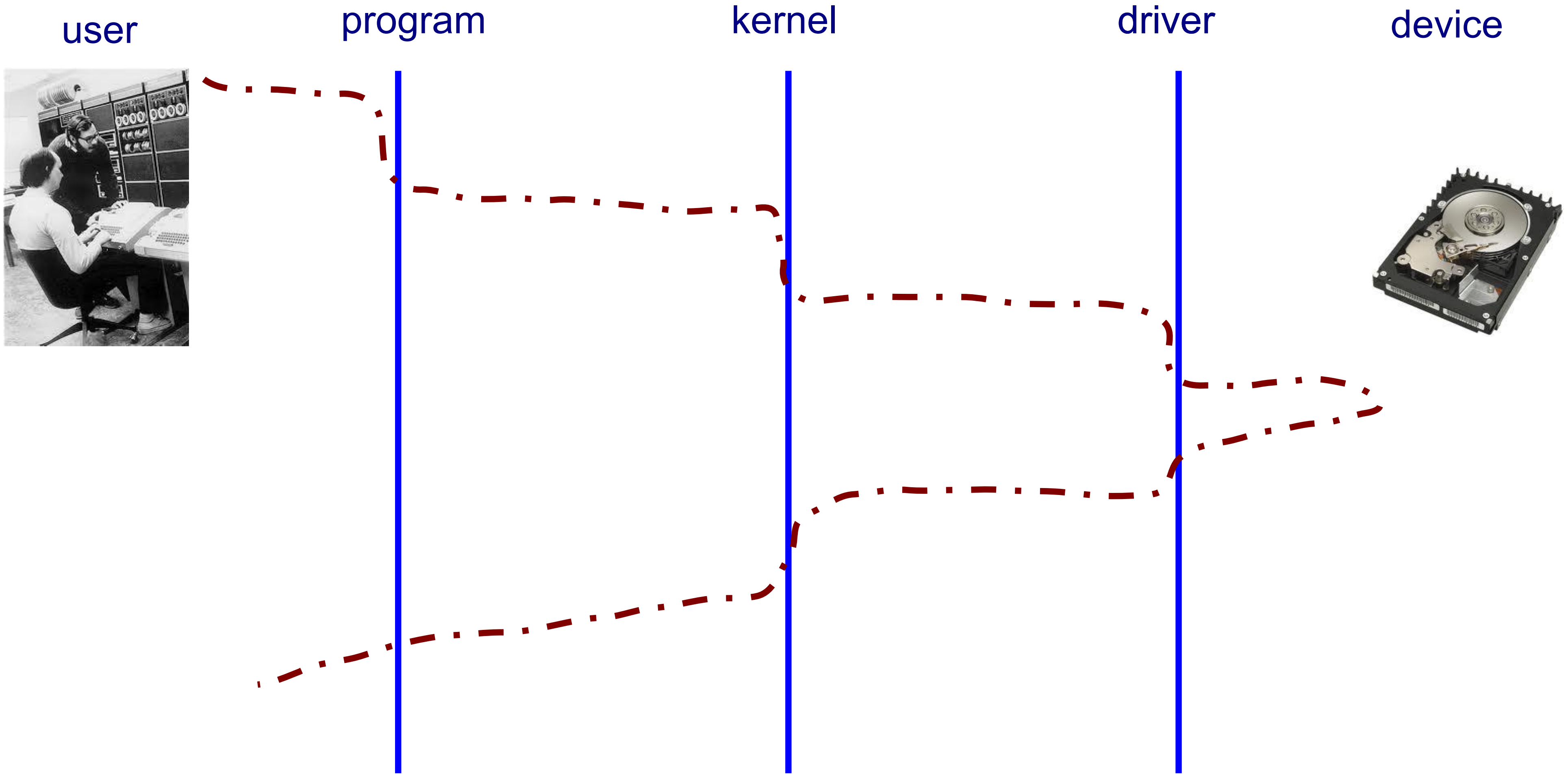
Threads

- Kernel threads
  - Jobs the kernel itself is doing for internal [projects]
  - All have access to the same kernel data
- User context
  - Threads running user jobs, might be running kernel code to service system calls from user code
  - Collected into specific user processes, see only the individual process space data
- Interrupts
  - Not really full threads, but in the mix
- See ps -ef

Linux process tree



Thread simple



user

program

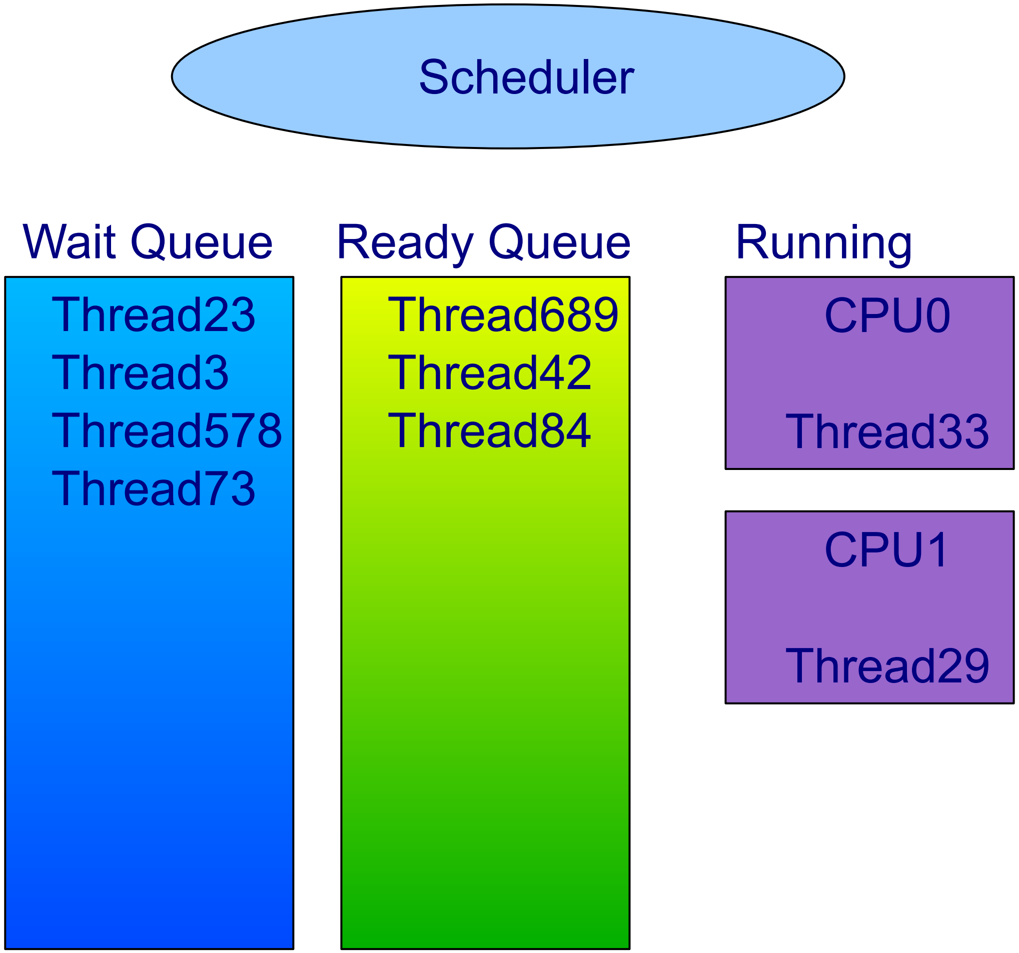
kernel

driver

device

OS Scheduler

- Chooses which to process/thread to run next on which CPU
  - Ready queue
  - Wait queue
  - Thread priority
  - Time slice
  - Preemption
  - CPU core affinity
  - Etc



Scheduler

Wait Queue

Thread23
Thread3
Thread578
Thread73

Ready Queue

Thread689
Thread42
Thread84

Running

CPU0

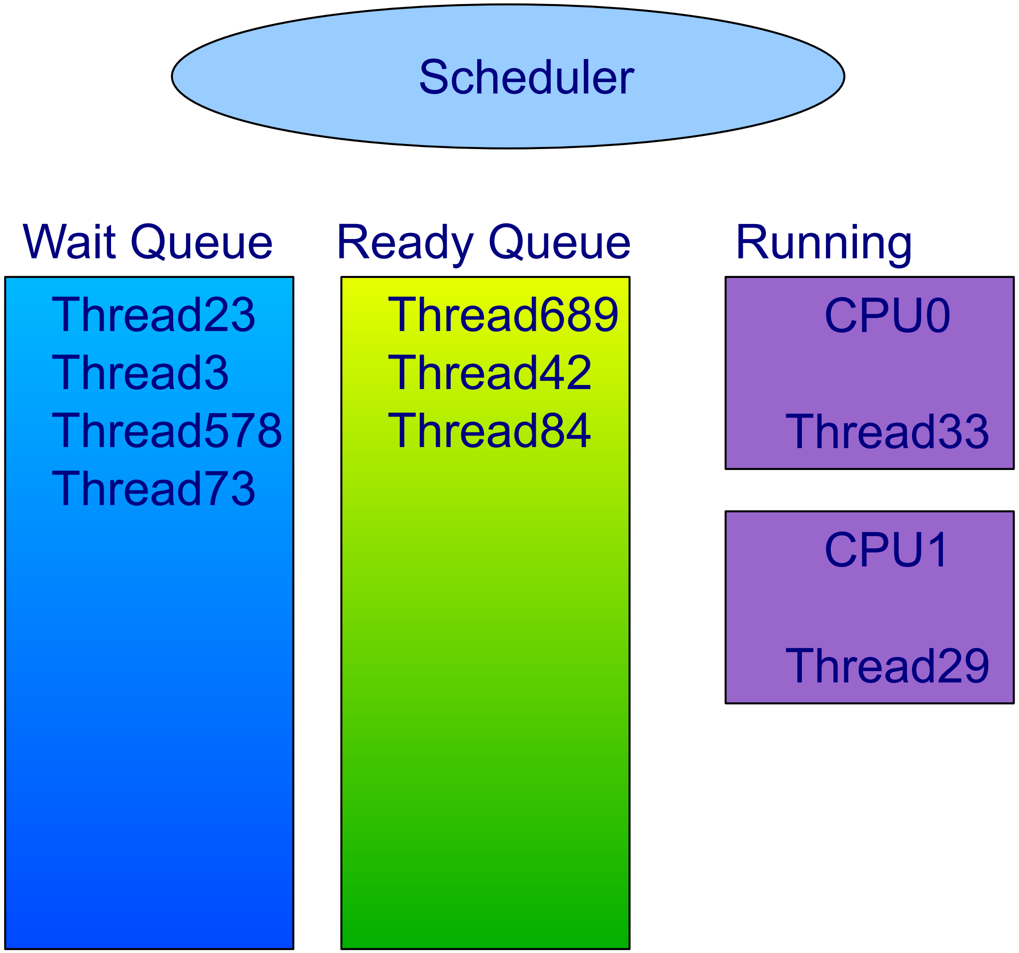
Thread33

CPU1

Thread29

OS Scheduler

- Chooses which to process/thread to run next on which CPU
  - Ready queue
  - Wait queue
  - Thread priority
  - Time slice
  - Preemption
  - CPU core affinity
  - Etc



Scheduler

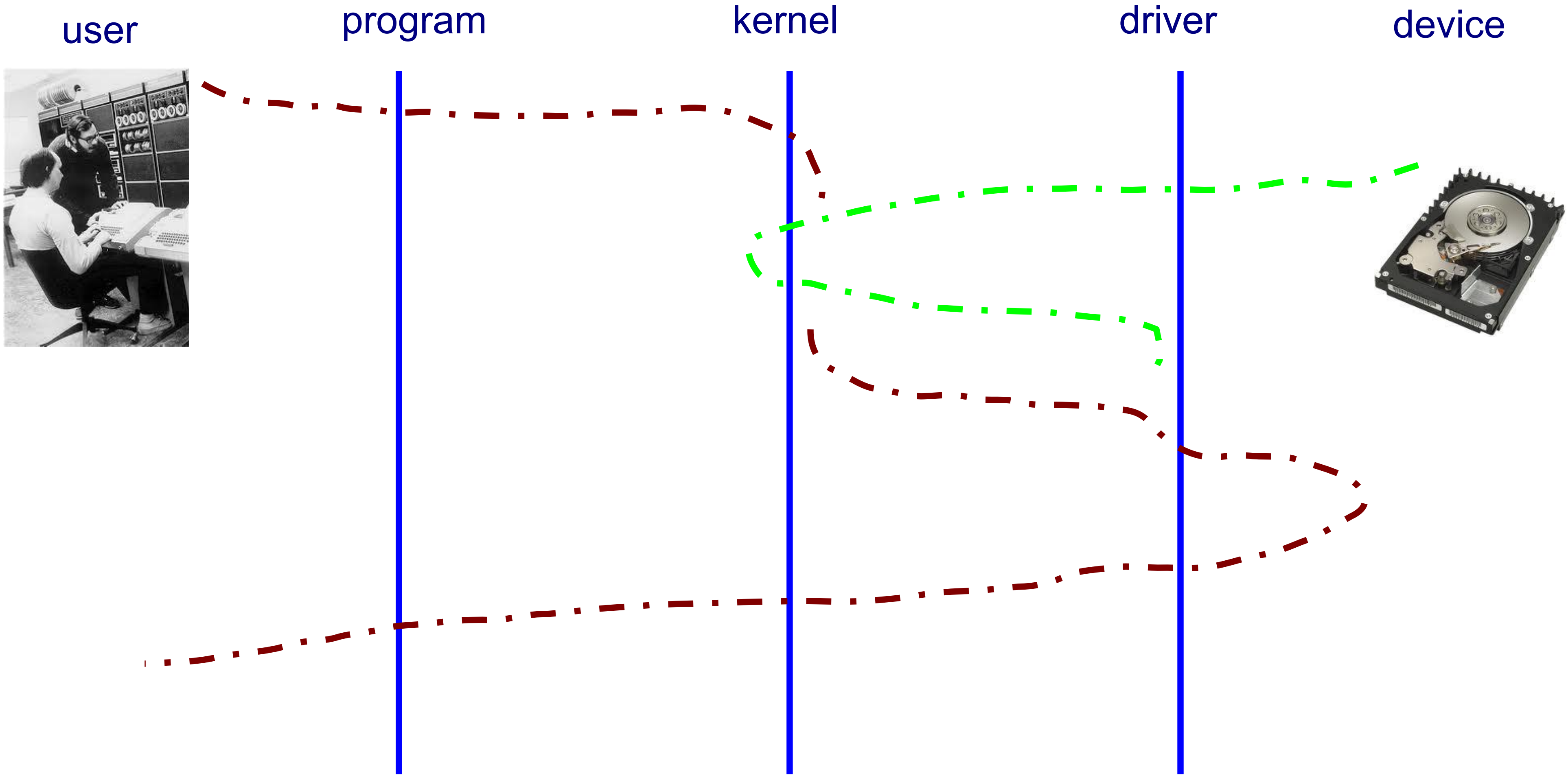
Wait Queue
Thread23
Thread3
Thread578
Thread73

Ready Queue
Thread689
Thread42
Thread84

Running
CPU0
Thread33
CPU1
Thread29

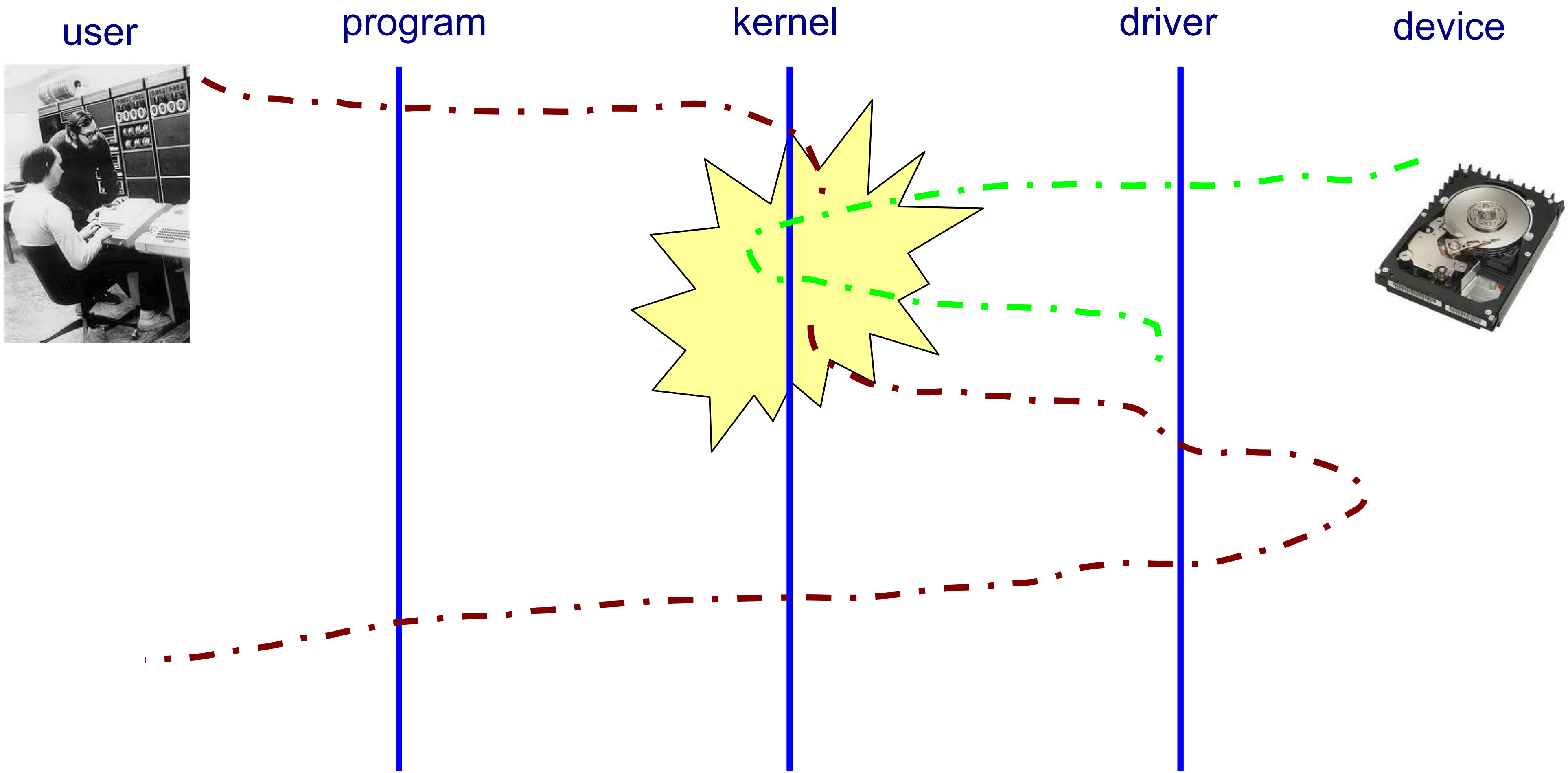
- Interrupts mess with scheduler plans

Thread, interrupted



Concurrency and conflict

- Multiple threads could hit data at same time



user

program

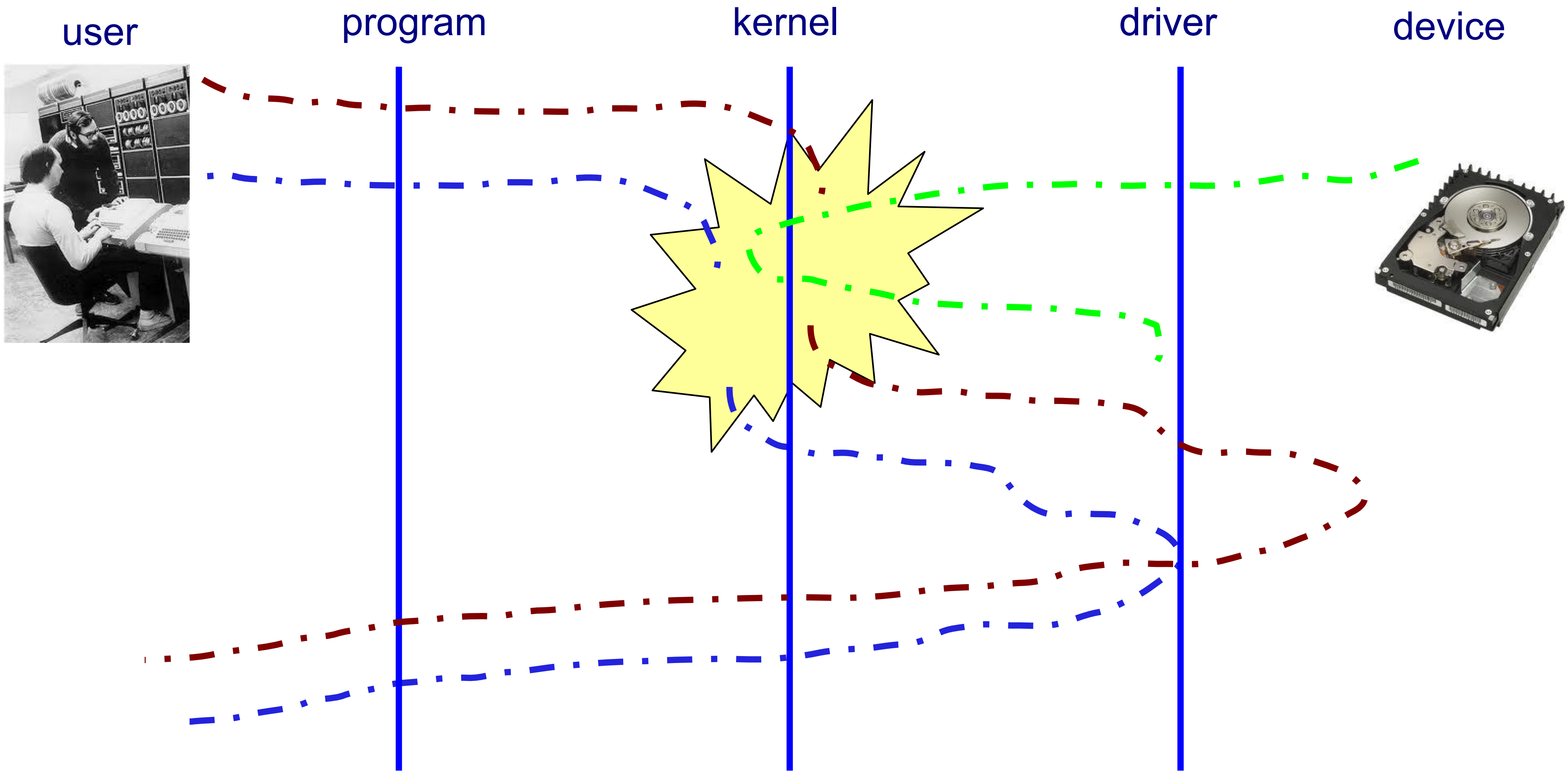
kernel

driver

device

Concurrency and conflict

- Multiple CPU threads could hit data at same time



Order matters

Instance 1	Instance 2	Value
read very_important_count		5
add 5 + 1 = 6		6
write very_important_count		6
	read very_important_count	6
	add 6 + 1 = 7	7
	write very_important_count	7

Order matters

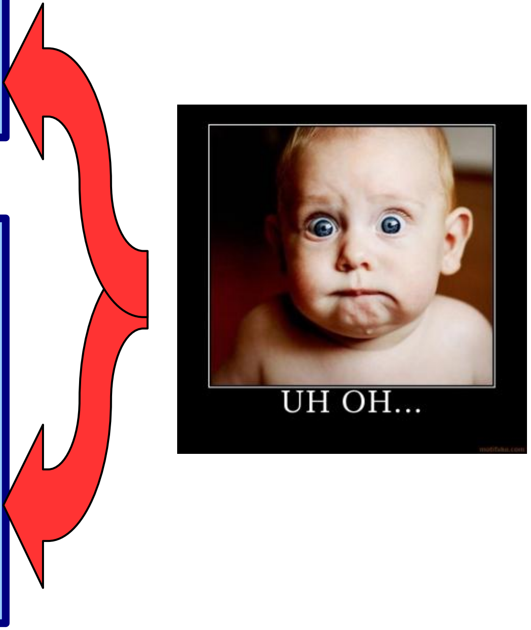
Instance 1	Instance 2	Value
read very_important_count		5
add 5 + 1 = 6		6
write very_important_count		6
	read very_important_count	6
	add 6 + 1 = 7	7
	write very_important_count	7

Instance 1	Instance 2	Value
read very_important_count		5
	read very_important_count	5
add 5 + 1 = 6		6
	add 5 + 1 = 6	6
write very_important_count		6
	write very_important_count	6

Order matters

Instance 1	Instance 2	Value
read very_important_count		5
add 5 + 1 = 6		6
write very_important_count		6
	read very_important_count	6
	Add 6 + 1 = 7	7
	write very_important_count	7

Instance 1	Instance 2	Value
read very_important_count		5
	read very_important_count	5
add 5 + 1 = 6		6
	Add 5 + 1 = 6	6
write very_important_count		6
	write very_important_count	6



Concurrency and conflict

- Multiple threads could hit data at same time

- Coordination needed
  - Completions
  - Semaphore
  - Atomic action – increment, decrement
  - Mutex
  - Spin lock
  - RCU

Reading Hints

- LDD3 – Chapter 5: Concurrency

- ELDD – Chapter 2: Concurrency, pgs 39-48
- ELDD – Chapter 3: Kernel Facilities