CMPE110 Lecture 21
Input / Output (I/O) 3

Heiner Litz

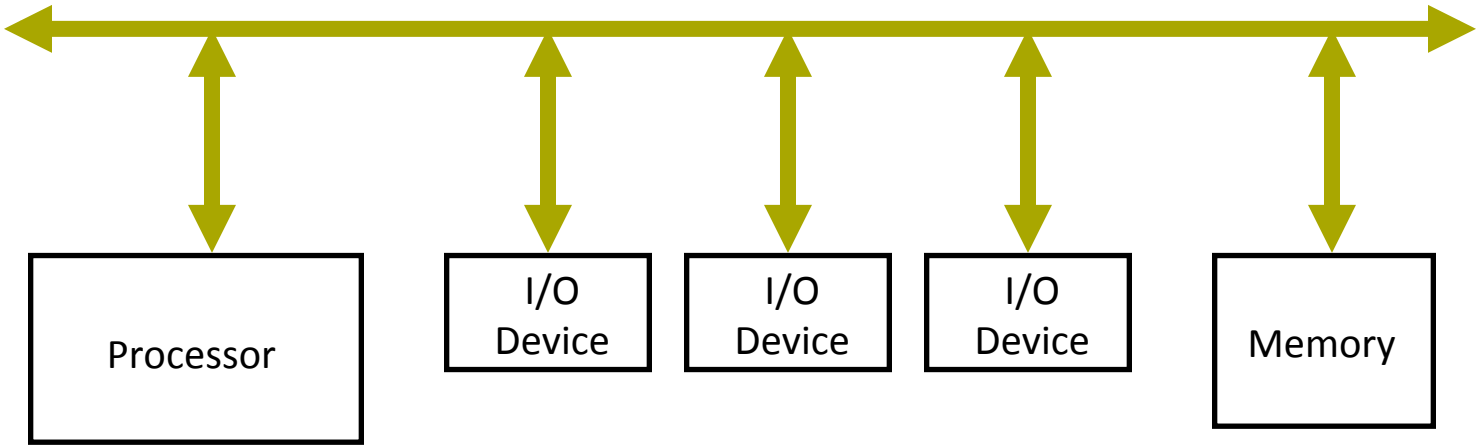
https://canvas.ucsc.edu/courses/12652

Announcements

Review

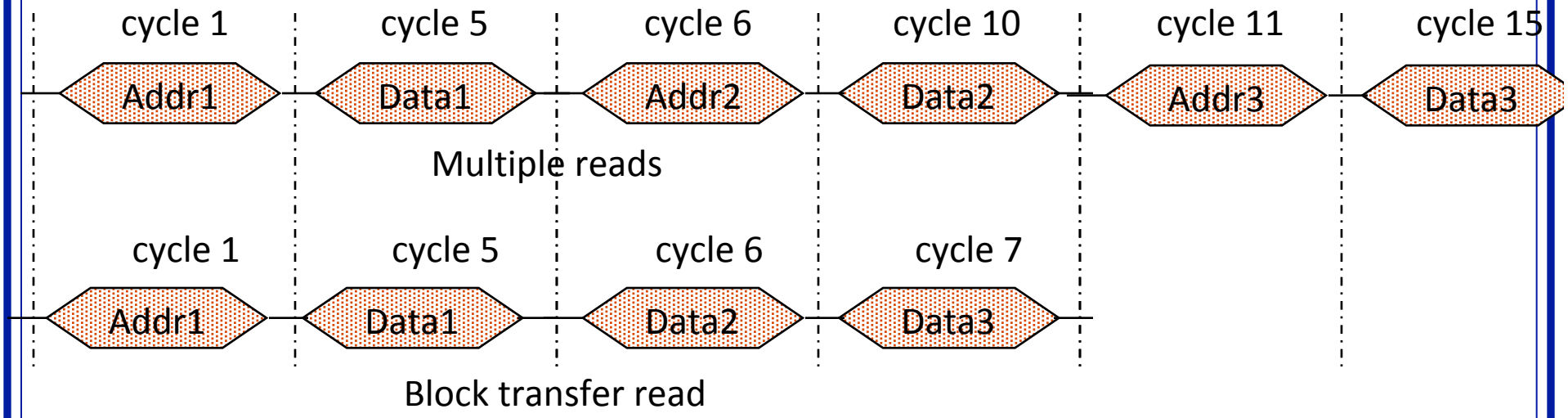
Bus

- A shared communication link that connects multiple devices
  - Single set of wires connects multiple “subsystems”
  - As opposed to a point to point link which only connects two components
- Wires connect in parallel
  - E.g., 32 bit bus has 32 wires of data
  - And a few additional wires for control



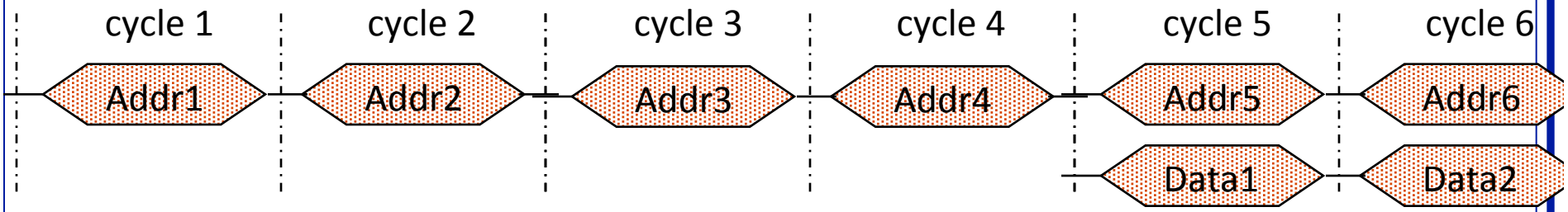
Increasing Bus Bandwidth

- Block transfers
  - Transfer multiple words in back-to-back bus cycles
  - Only one address needs to be sent at the start
  - Bus is not released until the last word is transferred
  - Costs: Increased complexity and response time for pending requests



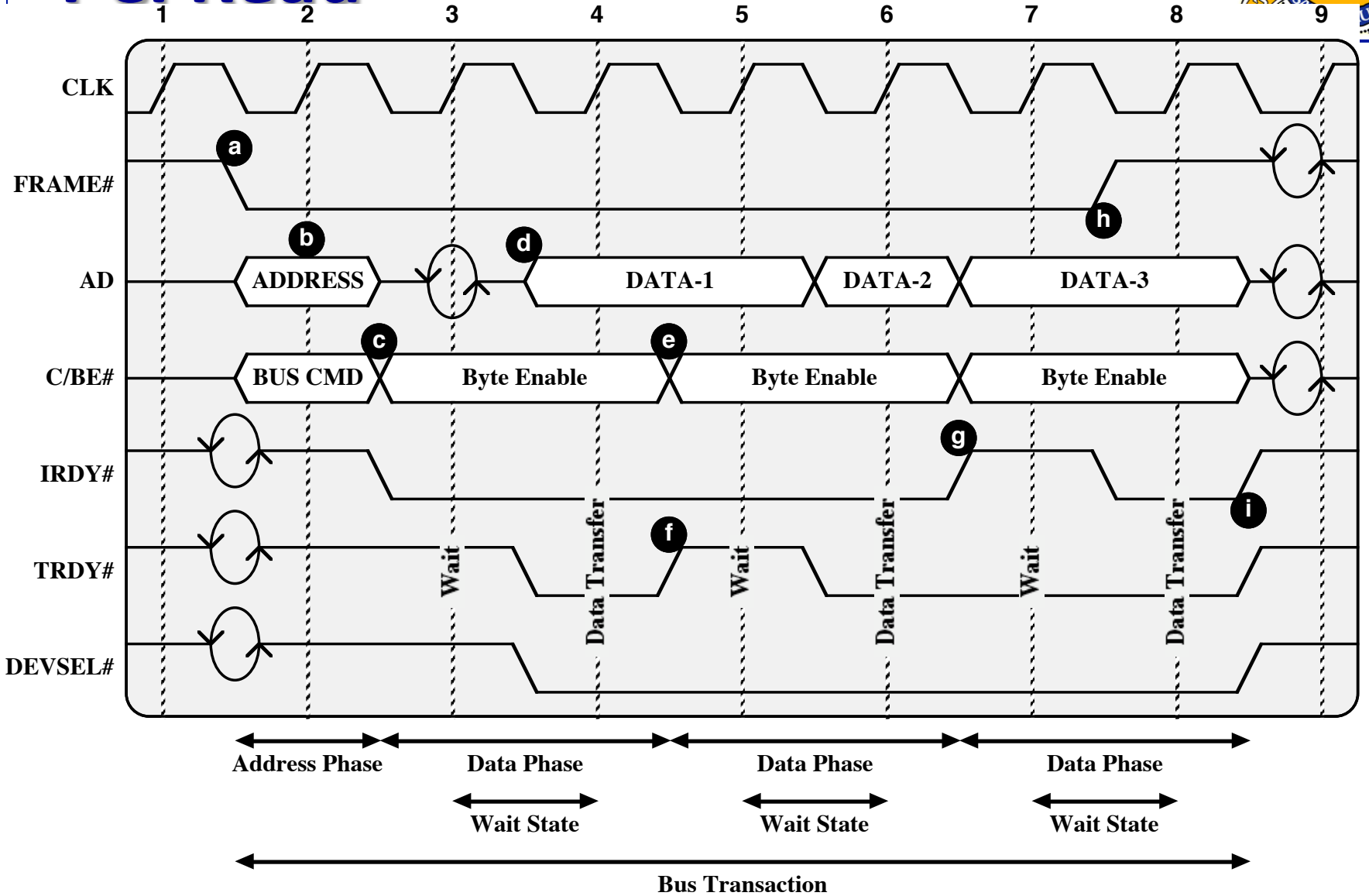
Increasing Bus Bandwidth

- Split transaction "pipelining the bus"
  - Free the bus during time between request and data transfer
  - Costs: Increased complexity and higher potential latency



Split transaction bus with separate Address and Data wires

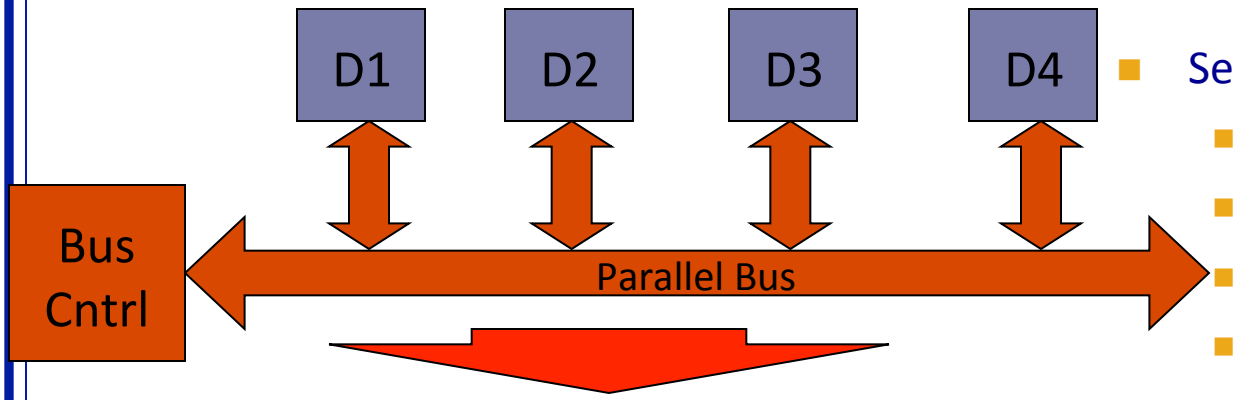
PCI Read


1
2
3
4
5
6
7
8
9
CLK
FRAME#
AD
C/BE#
IRDY#
TRDY#
DEVSEL#
a
b
c
d
e
f
g
h
i
ADDRESS
DATA-1
DATA-2
DATA-3
BUS CMD
Byte Enable
Byte Enable
Byte Enable
Wait
Data Transfer
Wait
Data Transfer
Wait
Data Transfer
Address Phase
Data Phase
Data Phase
Data Phase
Wait State
Wait State
Wait State
Bus Transaction

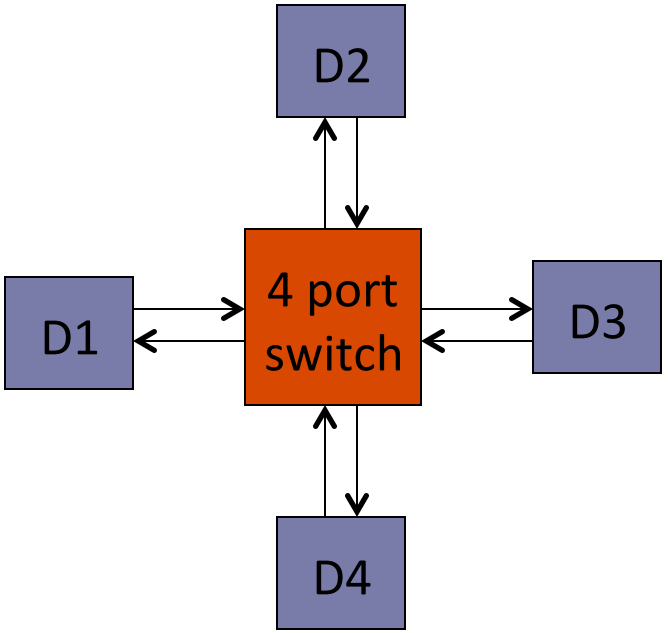
Buses Revisited

- Advantages:
  - Simple
  - Broadcast
  - Serialization
    - Where is this useful? Tip: think multi-core
- Drawbacks: many ways of saying it's slow
  - Only one transaction at a time
  - Electrically ugly
    - Stubs and discontinuities limit operating speed
  - Global closed-loop handshakes

Alternative: Point-to-Point Links & Switches



- Serial point-to-point advantages
  - Faster links
  - Fewer chip package pins
  - Higher performance
  - Switch keeps arbitration on chip



- Many bus standards are moving to serial, point to point
- 3GIO, PCI-Express(PCI)
- Serial ATA (IDE hard disk)
- AMD Hypertransport, Intel QPI versus Intel Front Side Bus (FSB)

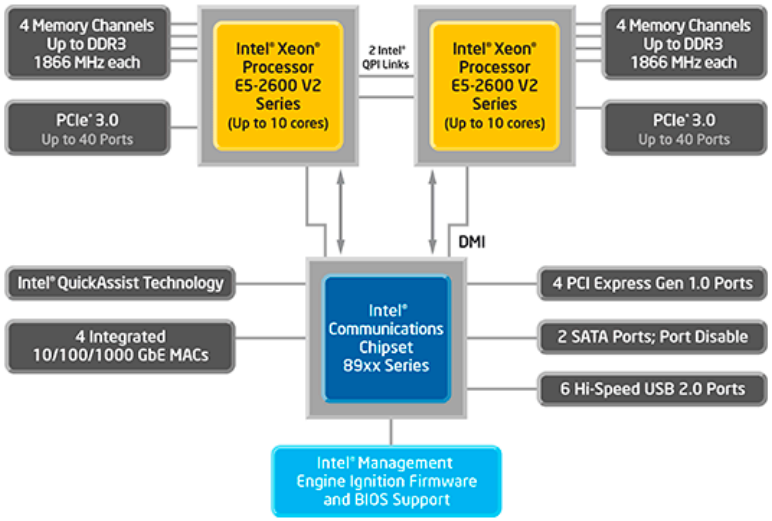
PCI vs. PCI Express

- Same bus protocol
  - Same driver software
- PCI
  - 32–64 shared wires
  - Frequency: 33MHz – 133 MHz
  - Bandwidth: 132 MB/s – 1 GB/s
- PCI Express
  - 1-16 wires per direction
  - Frequency: 2.5GT/s (PCIe 2.0: 5GT/s, PCIe 3.0: 8GT/s, PCIe 4.0: 16GT/s)
  - Bandwidth: 250 MB/s per direction per wire (500MB/984MB/1669MB)
- PCI Express advantage
  - ~100x pin bandwidth
  - Multiple links (lanes) for more bandwidth

Buses Vs Switched Interconnects

- Both useful, both used
- Busses used mostly on-chip
  - Mostly when connecting very small number of components
  - But also off-chip sometimes, e.g. DDR-x for main memory
- Switch-based interconnects used on and off-chip
  - Particularly when connecting many components
  - But topology can also be simple (e.g., rings)

Hierarchies of Interconnects



- Modern systems have hierarchies of buses/interconnects
  - Each chip tracks address maps
  - Memory-mapped I/O requests forwarded down the hierarchy until they reach the proper device
    - Potentially with translation along the way

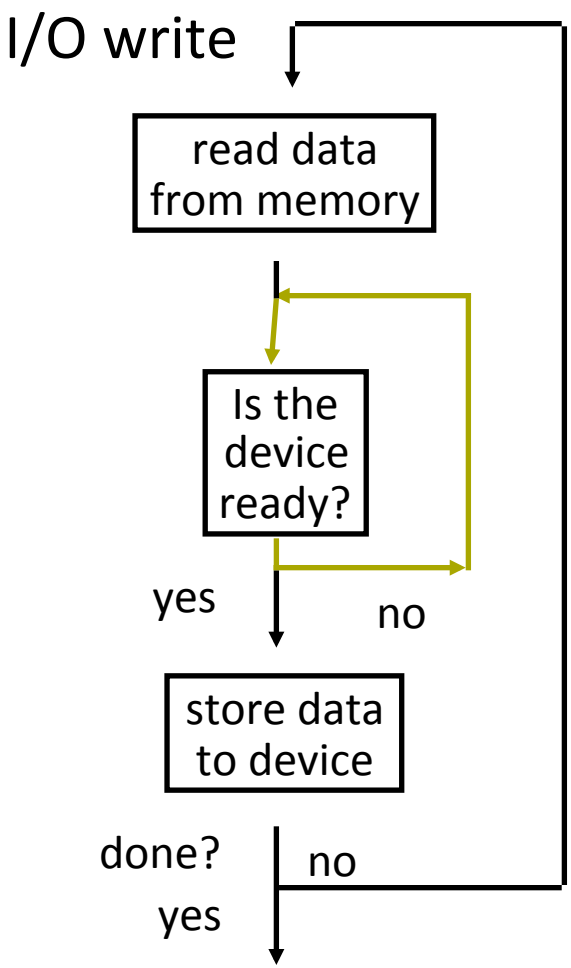
I/O Notification

- How does the OS or our program know that something interesting happened in the I/O device?
  - E.g., a packet arrived on the network interface
  - E.g., an error occurred while reading the disk

I/O Notification with Polling

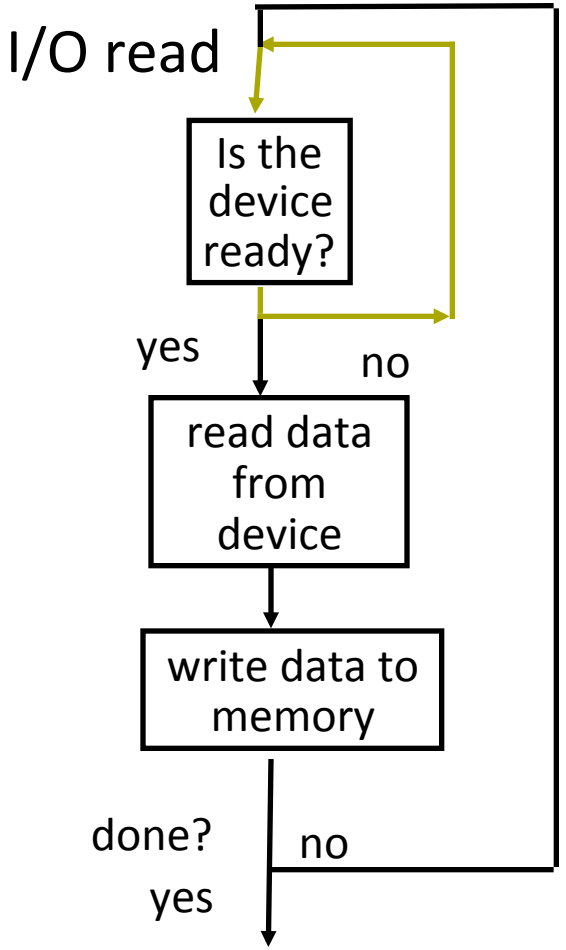
- The OS periodically checks the status register
  - I/O device places information in a status register
- Advantages
  - Simple to implement
  - Processor is in control and does the work
- Disadvantage
  - Polling overhead and data transfer consume CPU time
  - Difficult to determine how frequently to poll

I/O with Polling



Busy wait loop not an efficient way to use the CPU unless the device is very fast!

Processor may be inefficient way to transfer data

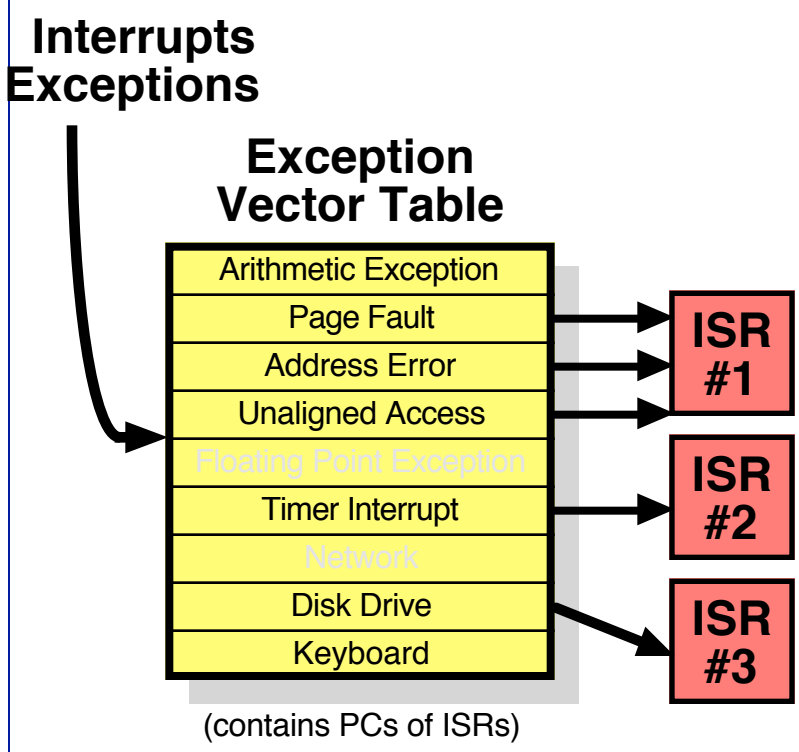


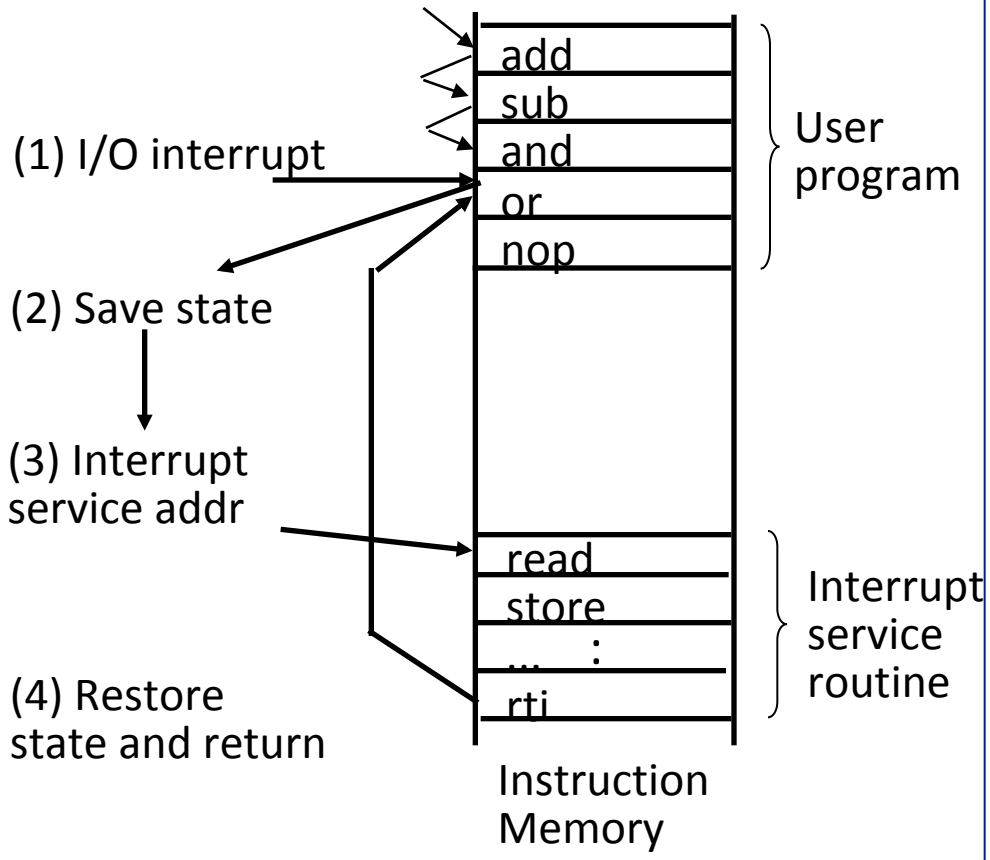
I/O Notification with Interrupts

- When I/O device needs attention, it *interrupts* the processor
  - Just like an exception but due to an external source and asynchronous
- Cause register helps OS find the I/O device
  - Or vector interrupts in the case of x86
- Interrupts may have different priorities
  - Ex.: Network = high priority, keyboard = low priority

- Advantage
  - Execution is only halted during actual transfer
- Disadvantage
  - Software overhead of interrupt processing

I/O Interrupts





Polling Vs. Interrupts

- Polling works great when
  - A device has a fixed I/O rate (e.g., keyboard or mouse)
  - The rate of interrupts is too high
  - The latency of interrupts is unacceptable
- Interrupts work best when
  - When devices can initiate their own I/O events
  - When I/O rate or latency is unpredictable
    - E.g. disk access
  - The overhead (CPU cycles) of polling is unacceptable

Large Data Transfers

- Simple approach: use ld/st instructions
  - SW moves *all* data between memory and I/O addresses
  - Simple and flexible, but wastes processor time

- Need a solution to allow data transfer to happen *without* the processor's involvement

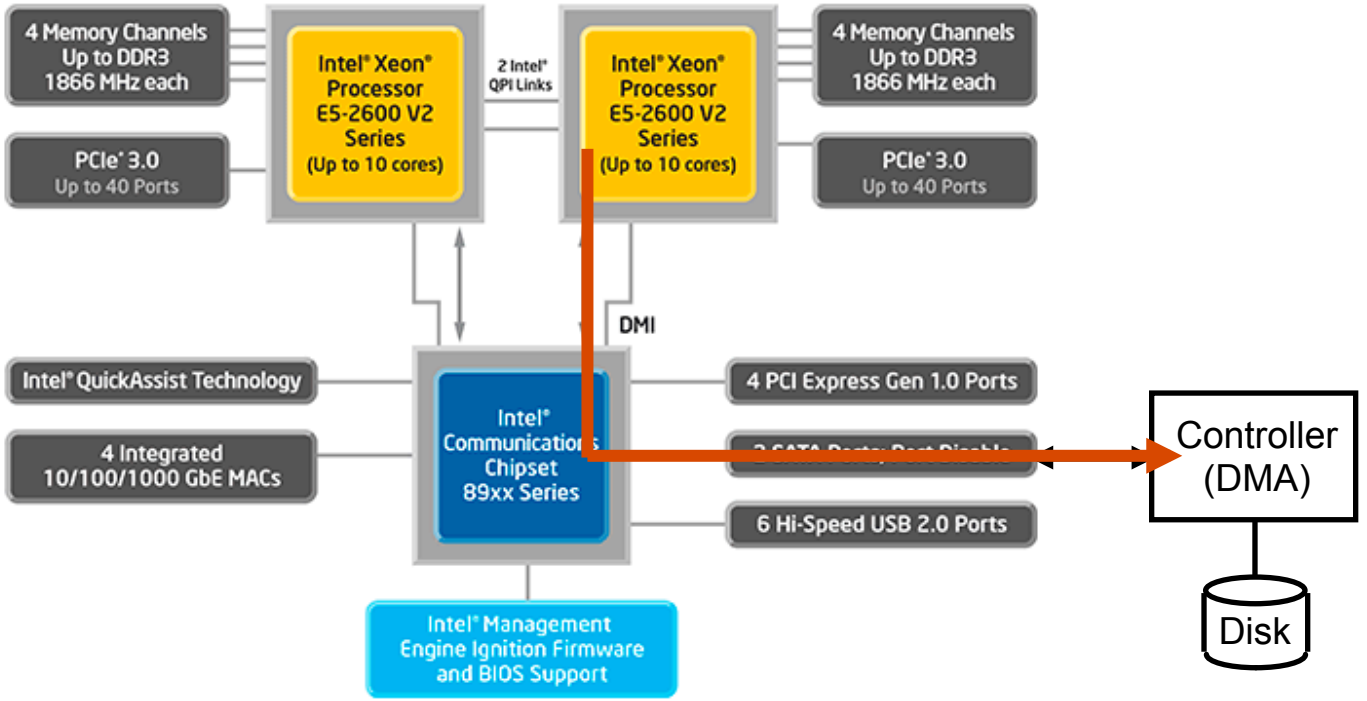
Direct Memory Access (DMA)

- A custom engine for data movement
  - Transfer blocks of data to or from memory without CPU intervention
  - DMA engines are available in processor chips and I/O chips
  - DMA engine acts as a bus master in bus-based systems
- DMA engine interface
  - The DMA engine is an I/O device itself
  - Registers or memory to describe transfers
    - “From”, “to”, and “length” registers
    - Can have multiple of those or a queue
  - Registers for status and control

DMA Steps

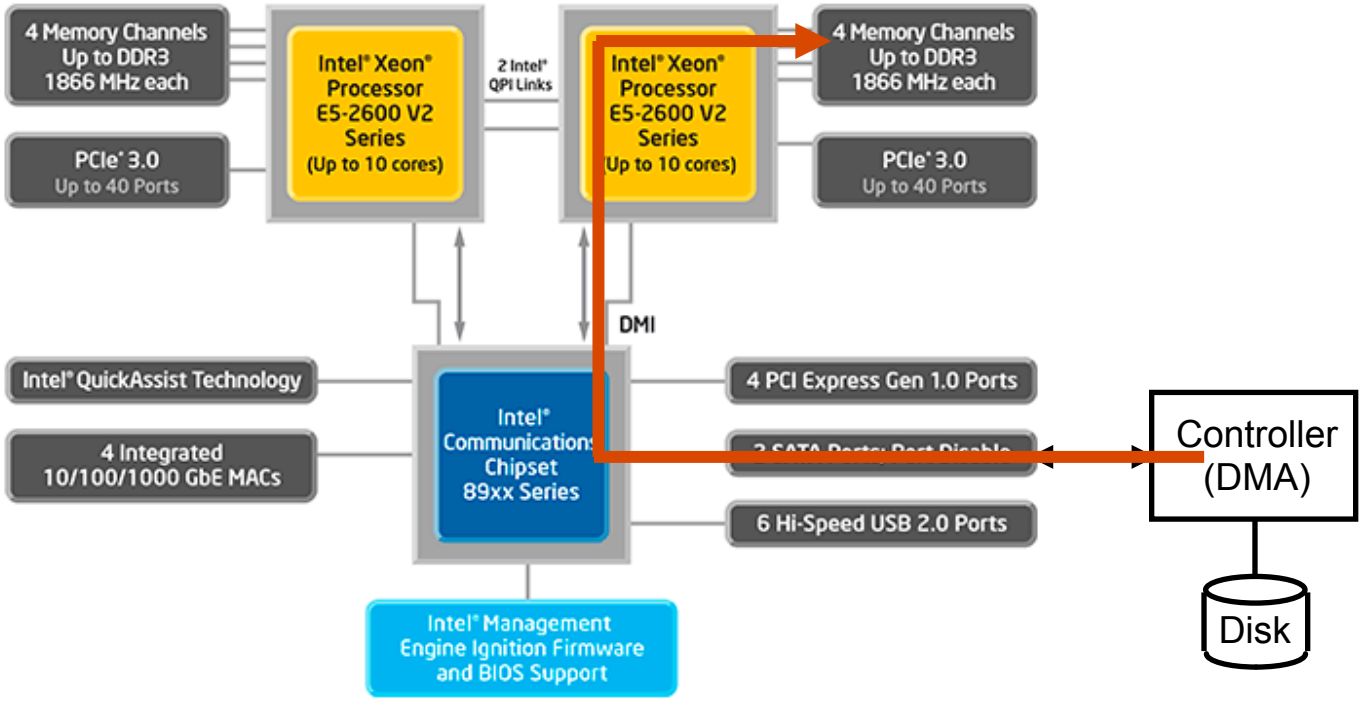
- Processor sets up DMA by supplying
  - Identity of the device and the operation (read/write)
  - The memory address for source/destination
  - The number of bytes to transfer
- DMA engine starts by arbitrating for bus and then starting transfer when data is ready
- Notify processor when transfer is complete or on error
  - Typically with interrupt

Reading a Disk Sector (1)



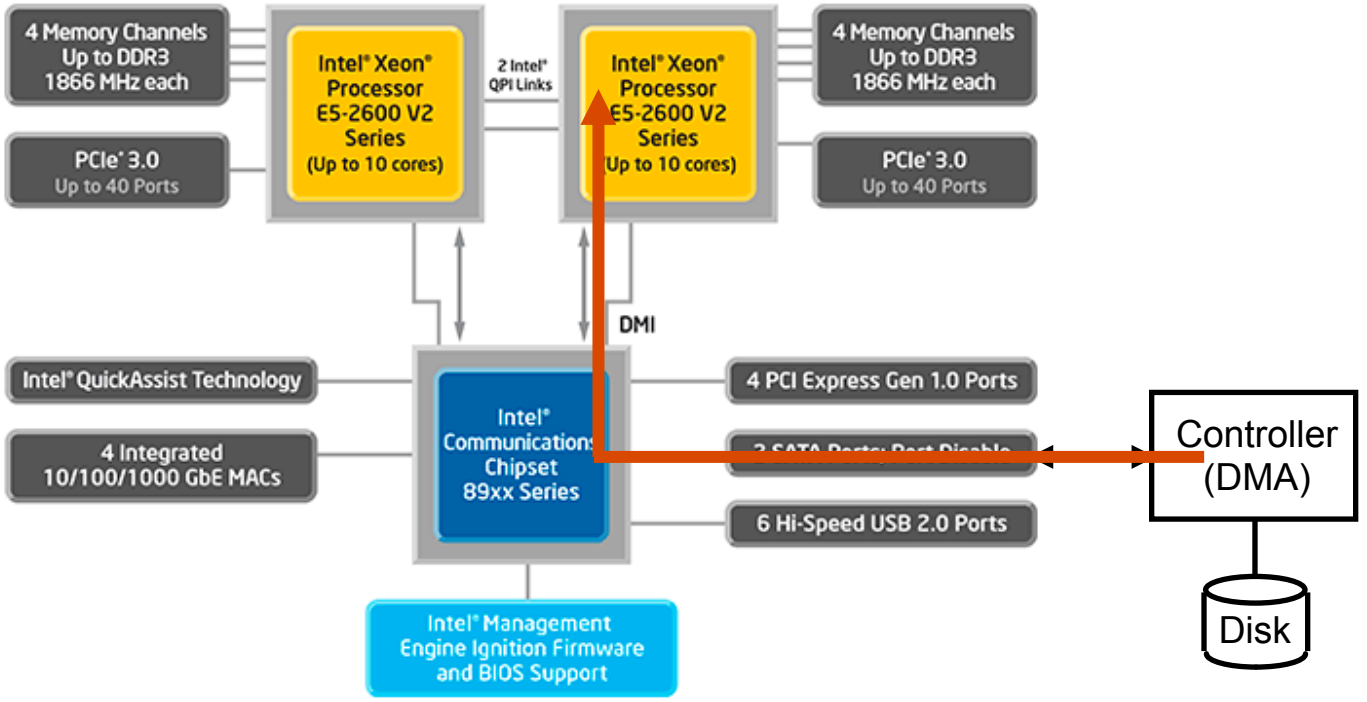
- CPU initiates a disk read by writing a command, logical block number, and a destination memory address to disk controller control registers.

Reading a Disk Sector (2)



- Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory (Disk is Master).

Reading a Disk Sector (3)



- When the DMA transfer completes, the disk controller notifies the CPU with an *interrupt*

DMA Issues (1): Memory Pinning

- For DMA with physical addresses, memory pages must be pinned in DRAM - Why?

- OS should not page to disks pages involved with pending I/O

DMA Issues (2): Memory Translation

- If DMA uses physical addresses
  - Memory access across physical page boundaries may not correspond to contiguous virtual pages (or even the same application!)
- Solution 1: ≤ 1 page per DMA transfer
- Solution 1+: chain series of 1-page requests provided by the OS
  - Single interrupt at the end of the last DMA request in the chain
- Solution 2: DMA engine uses virtual addresses
  - Multi-page DMA requests are now easy
  - A TLB is necessary for the DMA engine
    - Who manages the TLB?

DMA Issues (3): Cache Coherence

- The data involved in DMA may reside in processor cache
  - If updated by I/O: must update or invalidate "old" cached copy
  - If read by I/O: must read latest value, which may be in the cache
  - Only a problem with write-back caches

- Another version of the "cache coherence" problem
  - Same problem in multi-core systems

DMA & Coherence

- Solution 1: OS flushes the cache before I/O reads or forces write-backs before I/O writes
  - Flush/write-back may involve selective addresses or whole cache
  - Can be done in software or with hardware (ISA) support
- Solution 2: Route memory accesses for I/O through the cache
  - Search the cache for copies and invalidate or write-back as needed
  - This hardware solution may impact performance negatively
    - While searching cache for I/O requests, it is not available to processor
  - Multi-level, inclusive caches make this easier
    - Processor searches L1 cache mostly (until it misses)
    - I/O requests search L2 cache mostly (until it finds a copy of interest)