Direct Memory Access

Disk and Network transfers: awkward timing:

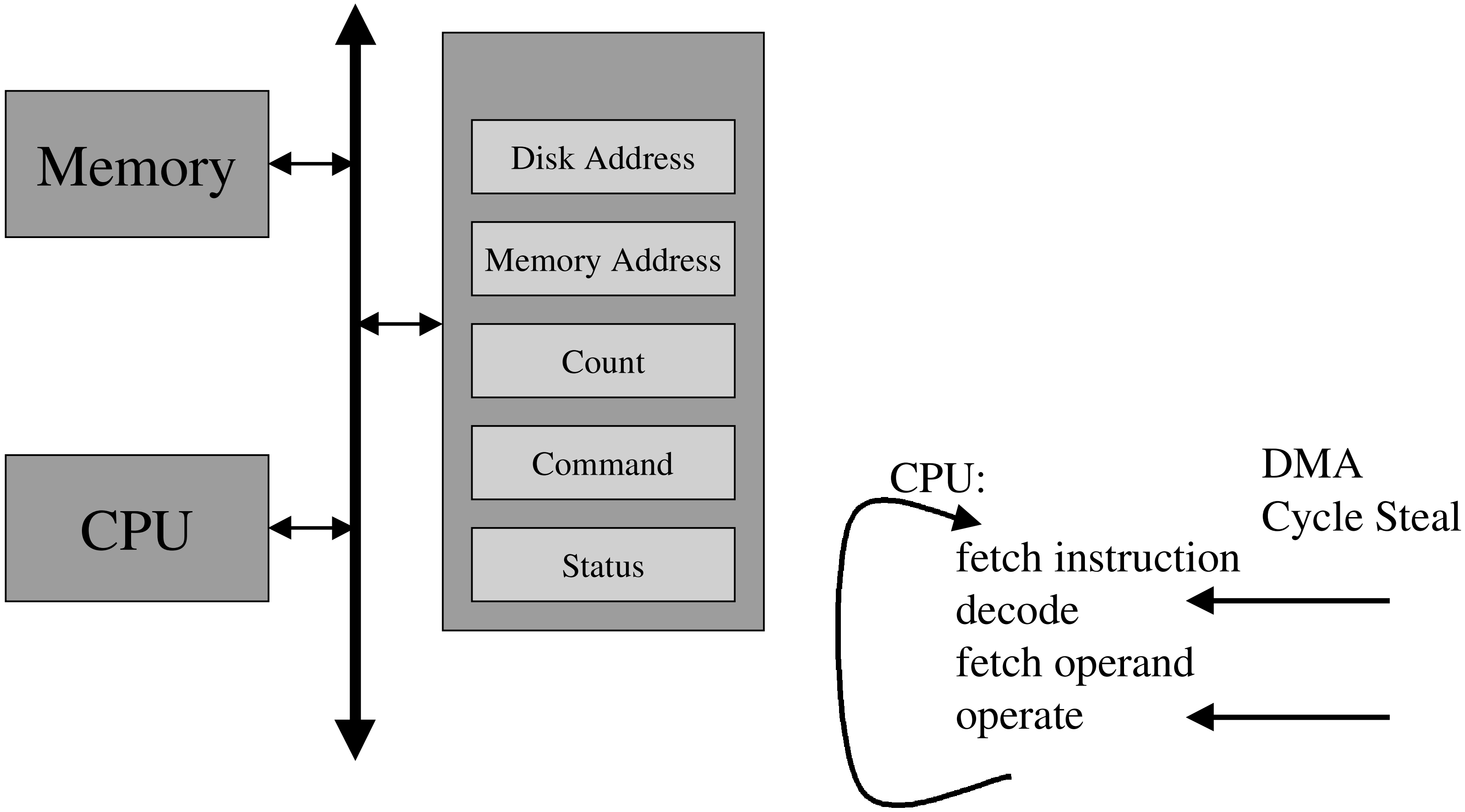
- Latency: milliseconds
- Transfer rate: megabytes per second

Too fast for interrupts

Too slow for polling

- (cope with latency?)

DMA Interface



Advanced DMA Controllers

- “Command Chaining”
- “Buffer Chaining”
- “Scatter/Gather”
- Interrupts

Memory Mapped I/O

Interface may appear as memory location

- e.g. write to disk interface control register initiates transfer
- e.g. write pixels to *frame buffer* - changes appearance of screen

n.b. protection issues

Timers

- Usually generate interrupts for:
  - Time of Day (real clock chip?)
  - timeslice for fair use of CPU
  - alarm clock service for applications
  - monitoring/profiling
    - Application
    - OS
  - Network & Device timeouts
- Typically 50-1000 interrupts per sec

Exceptions

- e.g.
  - Overflow interrupt
  - Odd address accessed
  - Invalid memory address used
  - Undefined instruction executed
- Similar to interrupts (save PC/PSR; set PRIV, goto handler) via vector table
- Synchronous to current running program

How does Application enter OS?

- e.g. application wishes to delete file, or send network packet
- Why not by procedure call instruction?
- Use Trap or Software Interrupt instruction, causing exception
- Pass parameters?

Interrupts in RISC processors

- (e.g. MIPS, Itanium)
- Interrupts are rare, complex activities
  - saving state in memory
  - looking up vector table
- So, implement in software
  - Shadow registers/first level ISR/save GPR/interrupt cause/specific ISR/pre-emption