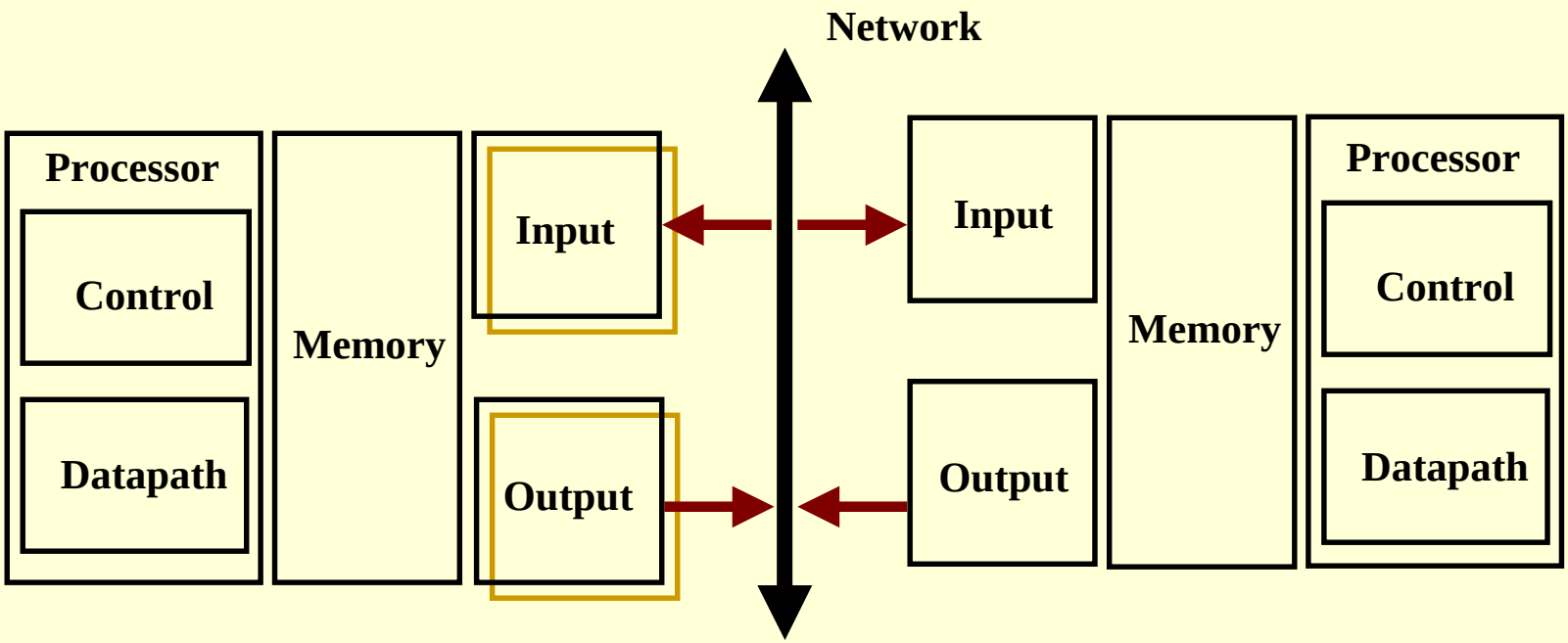
Input/Output



Exceptions and Interrupts

Terminology isn't consistent. We'll define:

- Exceptions: any unexpected change in control flow

  Branches and jumps are not considered exceptions.

- Exceptions are classified as either:
  - Internal Interrupts: Exception caused by something in the execution pipeline.
    - arithmetic overflow
    - illegal instruction
    - divide by zero
    - user program invoking the OS
  - External Interrupts: ones caused by something else.
    - I/O device signals completion to CPU
    - memory parity error
    - timer signal
    - low battery warning

Handling exceptions

On an exception, the hardware needs to:

1. Save the PC (allows the OS look at the offending instructions and to later resume execution)
2. Record the nature of the exception/interrupt
3. Transfer control to exception handler (in OS)

Two ways to do 2 & 3:

- Write cause of exception in a status register; pass control to the exception handler.
- Use vectored interrupt: transfers control to a different location for each possible type of interrupt/exception

User versus System operation

- User-level programs are restricted.
  - Only program state instructions can be executed
    - No I/O, no ability to set the page table register, etc.
    - Memory references are virtual addresses to user's VM.
  - If a user program needs to do I/O, it signals the OS
    - Typically by raising an exception
- System programs run in supervisor (aka kernel) mode
  - Can execute privileged instructions for I/O, etc.
- To accomplish I/O, hardware needs ability for:
  - OS to tell I/O device what to do
  - I/O devices to notify OS (e.g. "I'm done" or "I failed")
  - Data to be transferred from device to memory

Giving Commands to I/O Devices

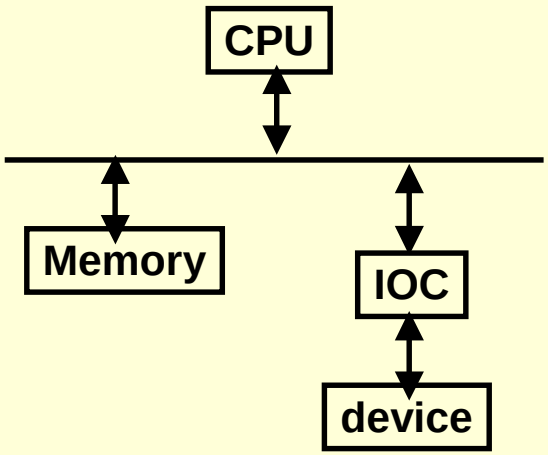
- Two methods are used to address I/O devices:
  - Special I/O instructions
  - Memory-mapped I/O
- Special I/O instructions specify a device number and a command
- Memory-mapped I/O:
  - Each I/O device is given a portion of the real address space

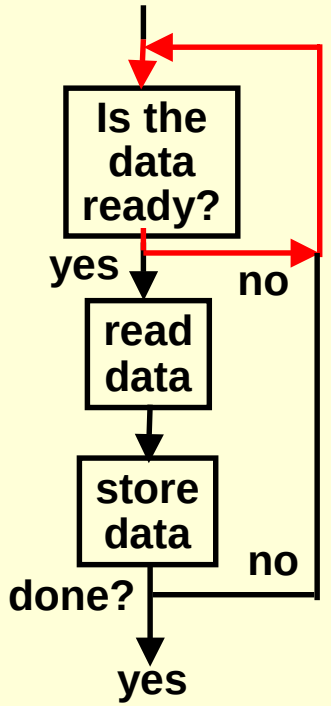
    To prevent chaos, these addressed aren't mapped to users' virtual memory.
  - Read and writes to those addresses (by the OS) are interpreted as commands by the I/O device

I/O Device Notifying the OS

- The OS needs to know when:
  - The I/O device has completed an operation
  - The I/O operation has encountered an error
- This can be accomplished in two different ways:
  - Polling:
    - The I/O device put information in a status register
    - The OS periodically check the status register
  - I/O Interrupt:
    - Whenever an I/O device needs attention from the processor, it issues an interrupt to the processor.

Polling



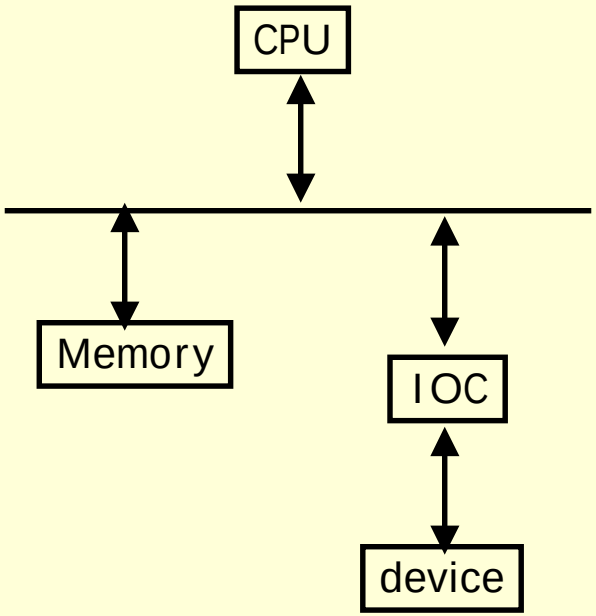


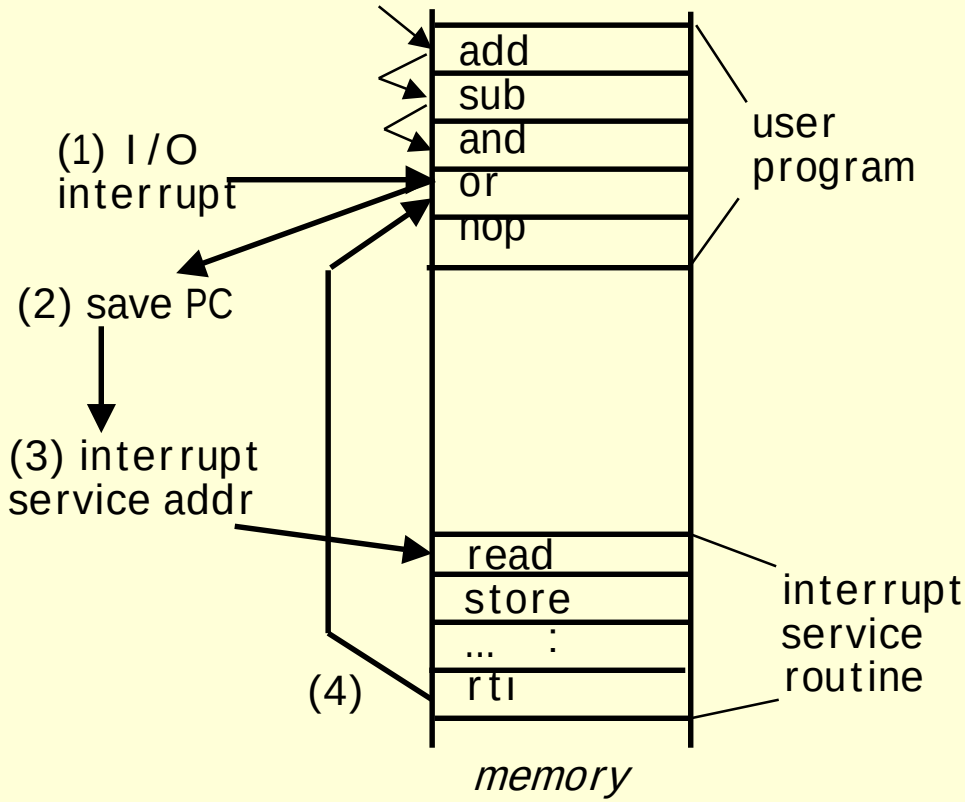
busy wait loop is an inefficient way to use the CPU unless the device is very fast!

but checks for I/O completion can be dispersed among computation intensive code

- Advantage:
  - Simple: the processor is in control and does all the work
- Disadvantage:
  - Polling overhead can consume a lot of CPU time

Interrupt Driven Data Transfer

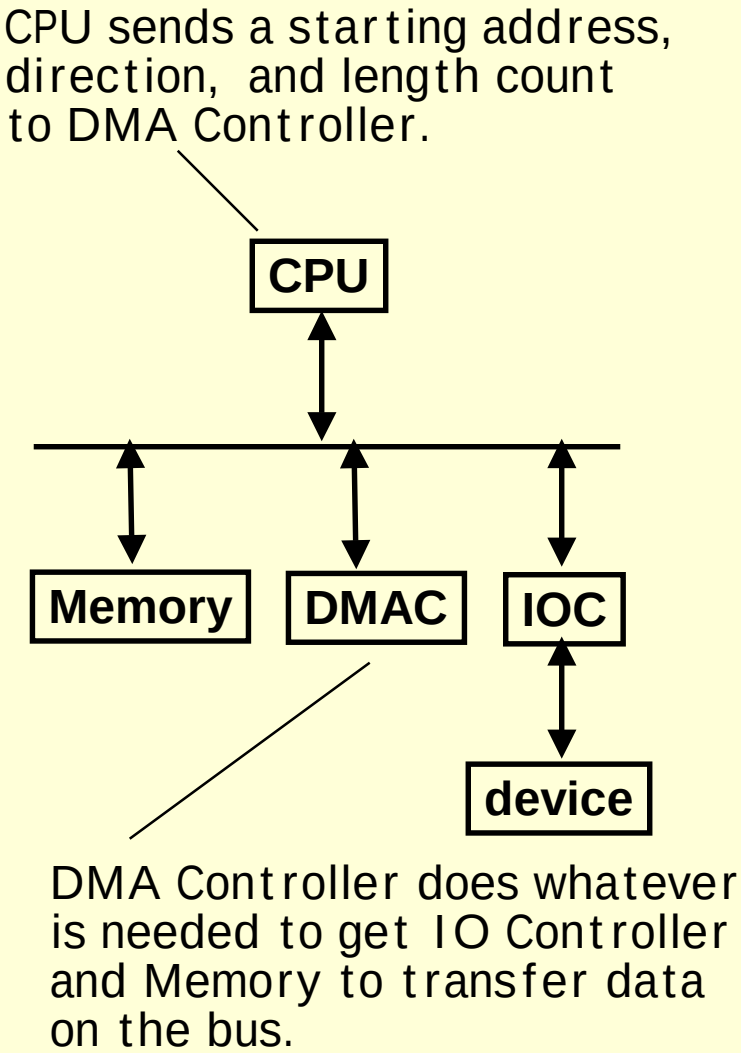




- Advantage:
  - User program is only halted during actual transfer
- Disadvantage:
  - Special hardware is needed to
    - Cause an interrupt (I/O device)
    - Detect an interrupt (processor)

Transferring Data to/form Memory

- Direct Memory Access (DMA):
  - External to the CPU
  - DMA controller is a bus master
  - Transfer blocks of data to or from memory without CPU intervention



Network technologies

- Local Area Network (LAN)
  - Ethernet is most common LAN technology
    - 10 Mbit/sec, 100 Mbit/s ("fast Ethernet") and 1 Gbit/s ("Gigabit Ethernet") versions
    - Originally, used "Aloha" protocol
      - If line is quiet, sender dumps message on wire
      - If message got garbled with another message, all senders wait a random amount of time and retry
      - Limits throughput to about 70% peak
    - Today, often controlled by switches
- Wide Area Network (WAN)
  - IP (Internet Protocol) used on Internet.
    - TCP layer on top of IP resends lost packets, etc
  - ATM (Asynchronous Transfer Protocol)
    - Another protocol for long distances; 155 Mbit/sec to 2.5 Gbit/sec