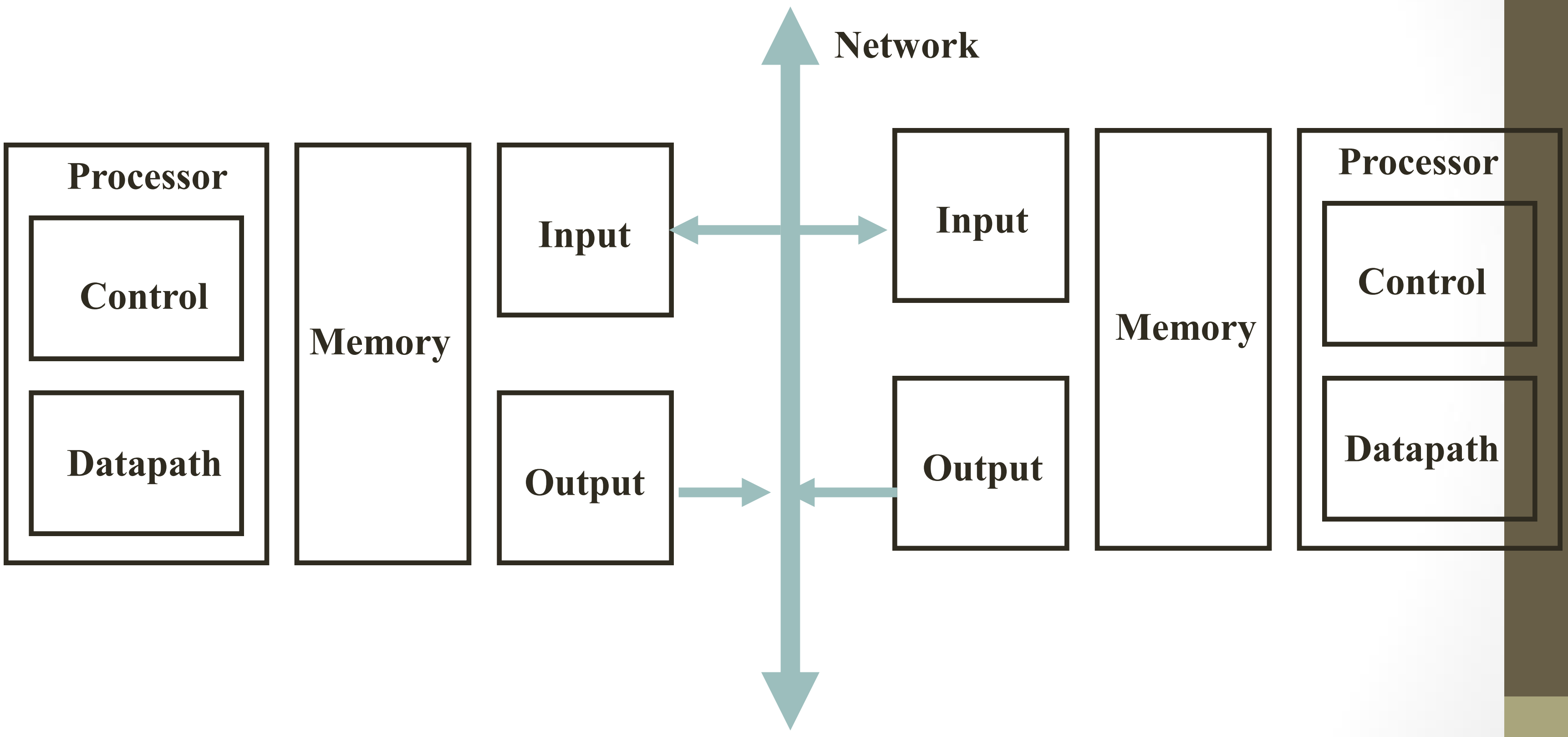
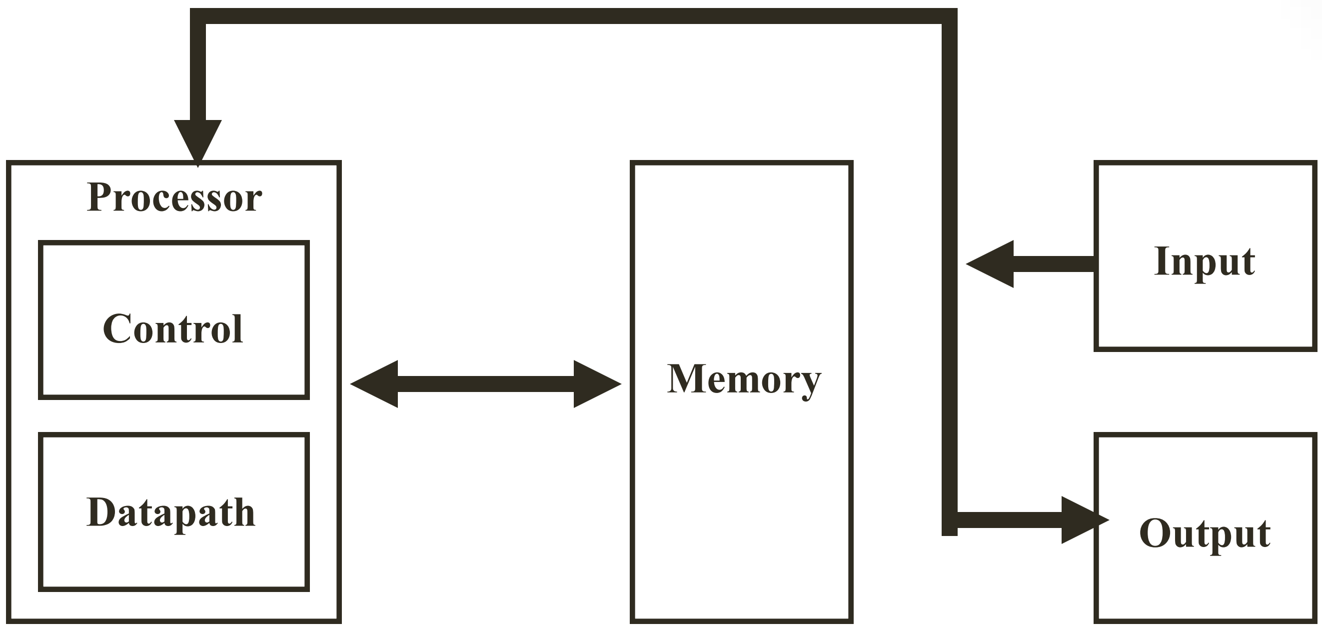
- **How to connect I/O to the rest of the computer?**

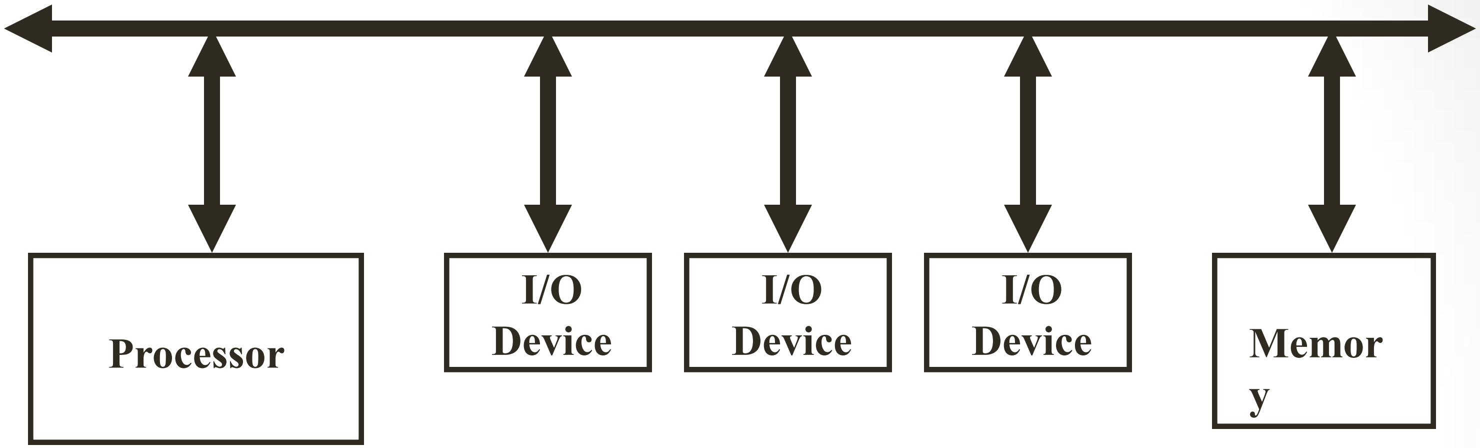


Buses: Connecting I/O to Processor and Memory



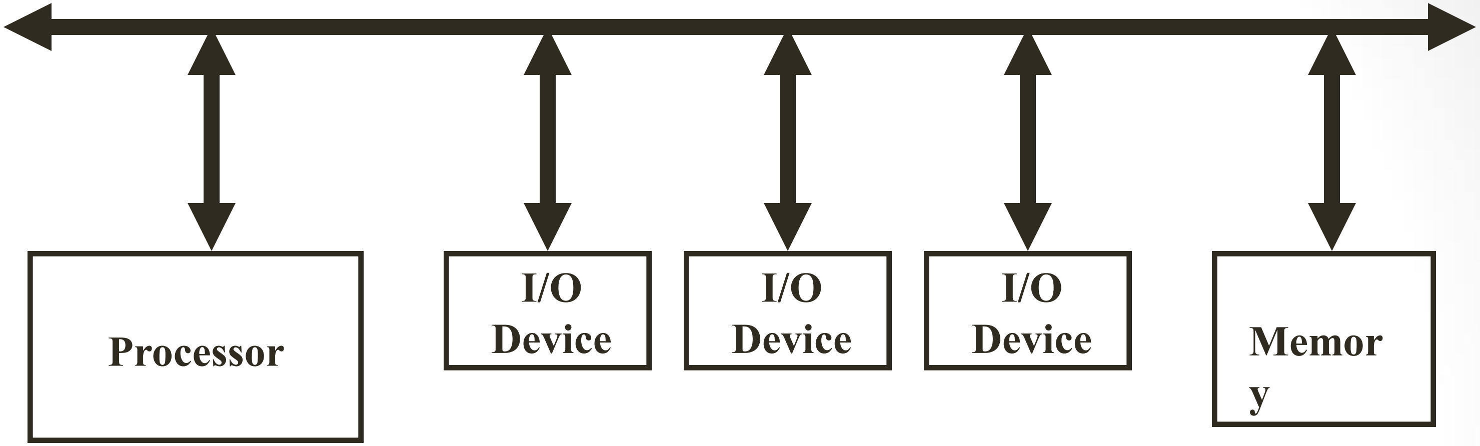
- A bus is a shared communication link
- It uses one set of wires to connect multiple subsystems

Advantages of Buses



- Versatility:
  - New devices can be added easily
  - Peripherals can be moved between computer systems that use the same bus standard
- Low Cost:
  - A single set of wires is shared in multiple ways

Disadvantages of Buses

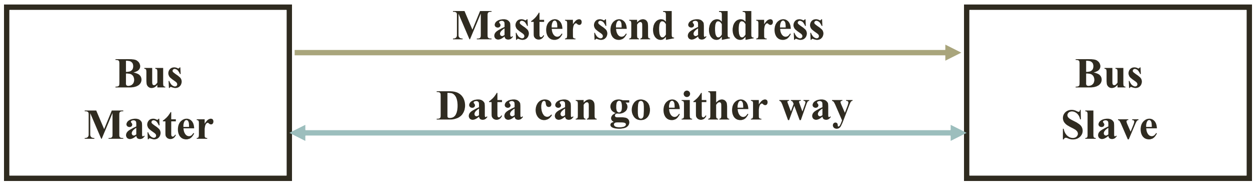


- It creates a communication bottleneck
  - The bandwidth of that bus can limit the maximum I/O throughput
- The maximum bus speed is largely limited by:
  - The length of the bus
  - The number of devices on the bus
  - The need to support a range of devices with:
    - Widely varying latencies
    - Widely varying data transfer rates

The General Organization of a Bus

- Control lines:
  - Signal requests and acknowledgments
  - Indicate what type of information is on the data lines
- Data lines carry information between the source and the destination:
  - Data and Addresses
  - Complex commands
- A bus transaction includes two parts:
  - Sending the address
  - Receiving or sending the data

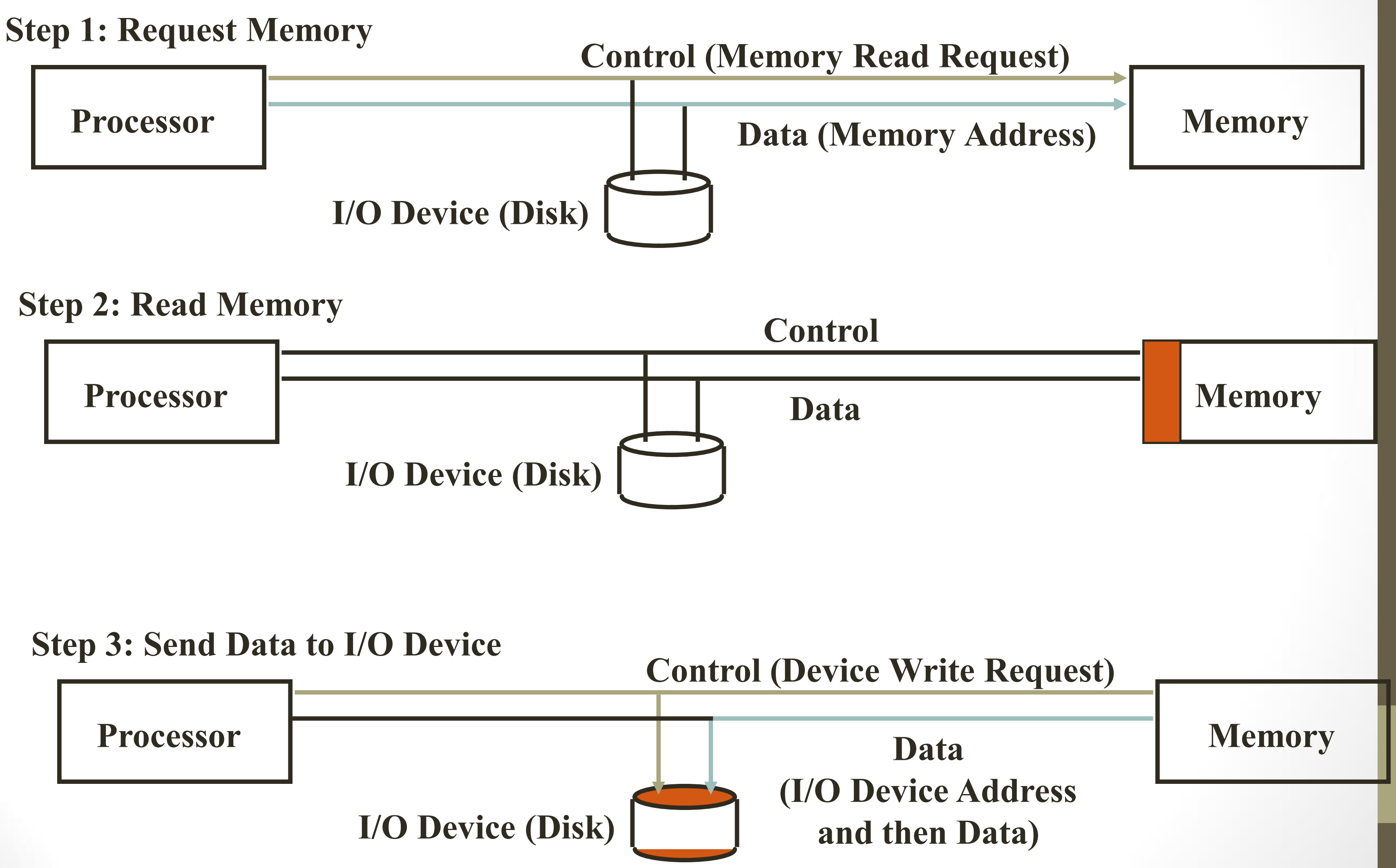
Master versus Slave



- A bus transaction includes two parts:
  - Sending the address
  - Receiving or sending the data
- Master is the one who starts the bus transaction by:
  - Sending the address
- Salve is the one who responds to the address by:
  - Sending data to the master if the master ask for data
  - Receiving data from the master if the master wants to send data

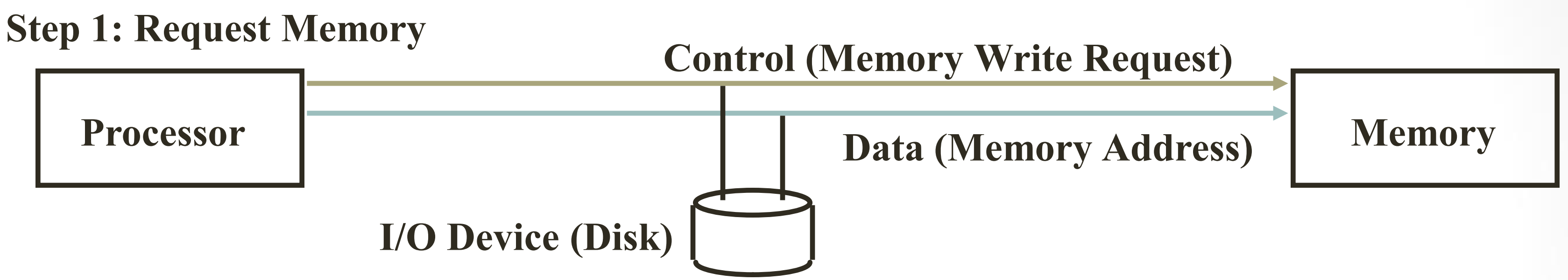
Output Operation

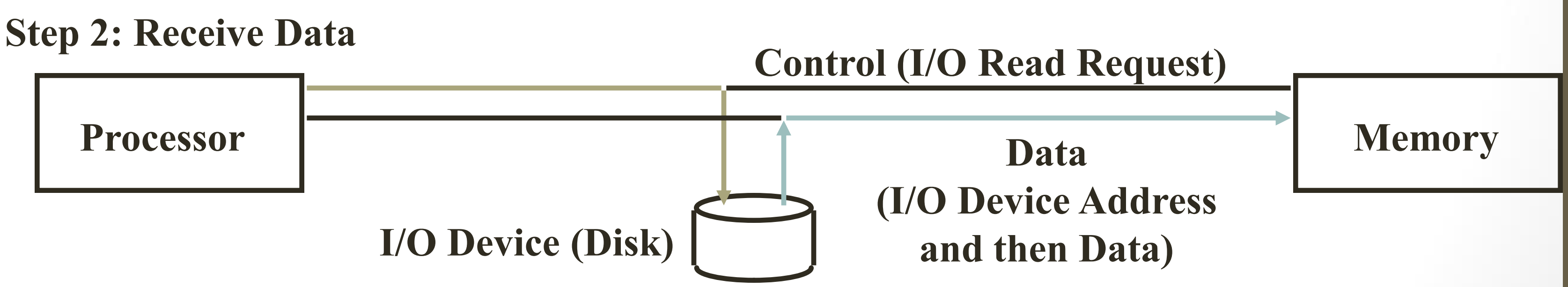
- Output is defined as the Processor sending data to the I/O device:



Input Operation

- **Input is defined as the Processor receiving data from the I/O device:**

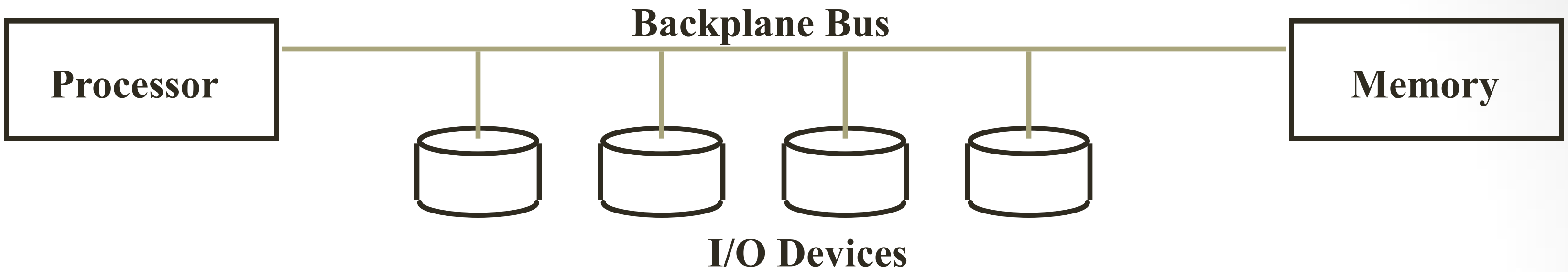




Types of Buses

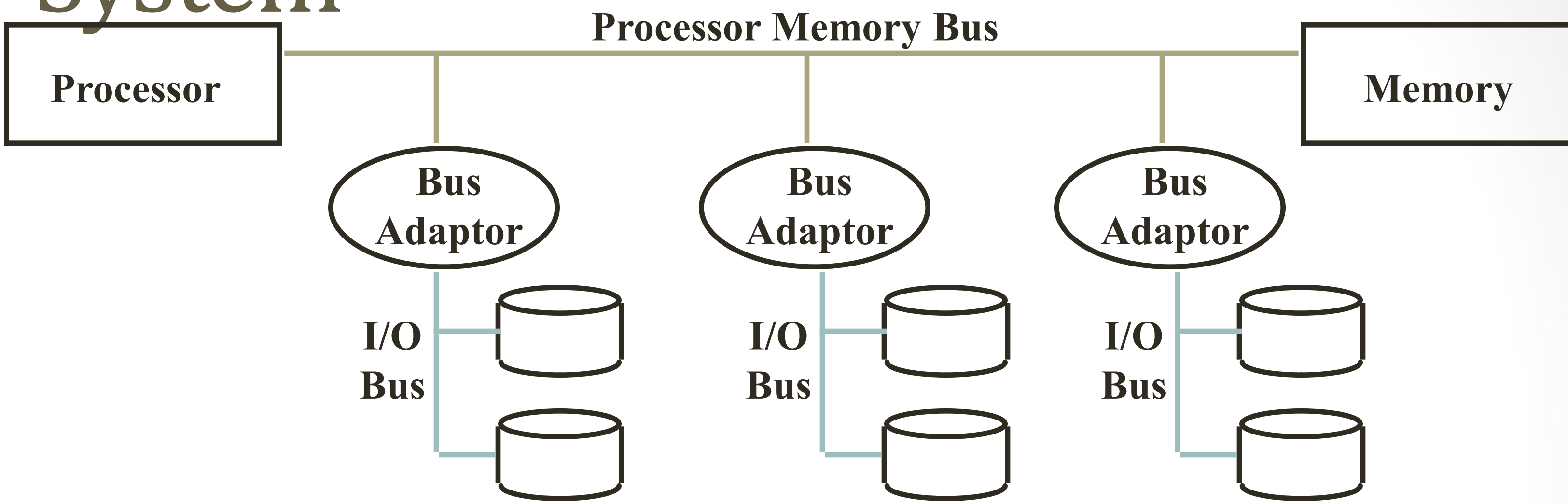
- Processor-Memory Bus (design specific)
  - Short and high speed
  - Only need to match the memory system
    - Maximize memory-to-processor bandwidth
  - Connects directly to the processor
- I/O Bus (industry standard)
  - Usually is lengthy and slower
  - Need to match a wide range of I/O devices
  - Connects to the processor-memory bus or backplane bus
- Backplane Bus (industry standard)
  - Backplane: an interconnection structure within the chassis
  - Allow processors, memory, and I/O devices to coexist
  - Cost advantage: one single bus for all components

A Computer System with One Bus: Backplane Bus



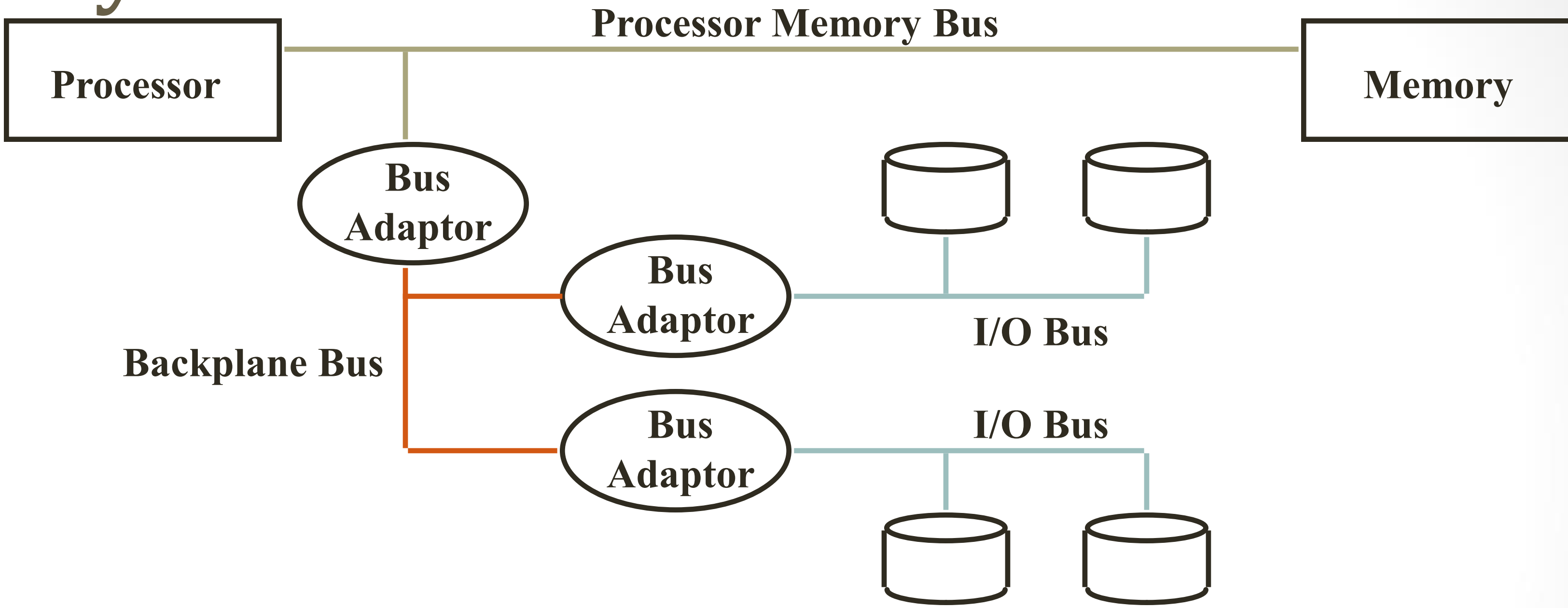
- A single bus (the backplane bus) is used for:
  - Processor to memory communication
  - Communication between I/O devices and memory
- Advantages: Simple and low cost
- Disadvantages: slow and the bus can become a major bottleneck
- Example: IBM PC

A Two-Bus System



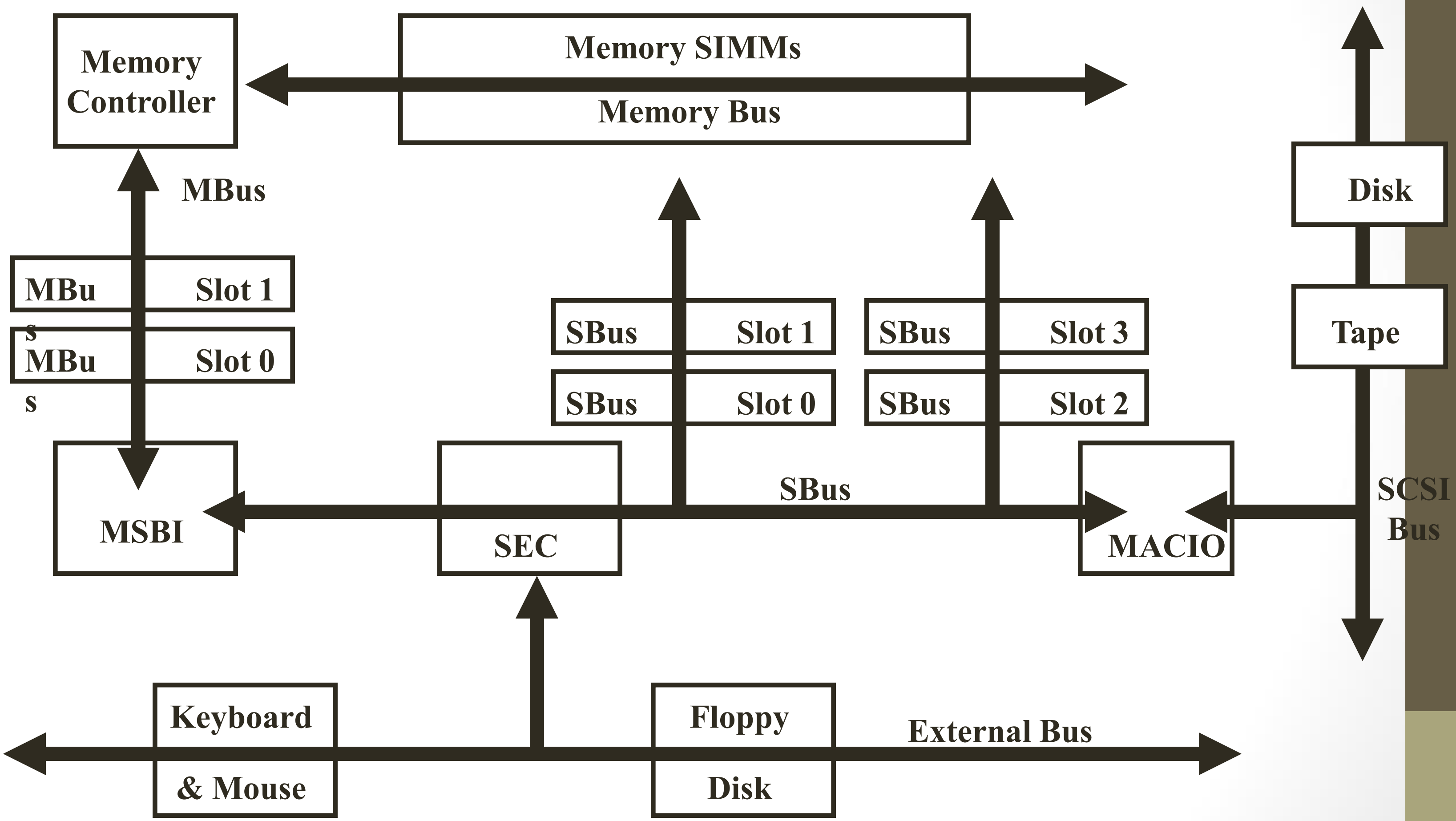
- I/O buses tap into the processor-memory bus via bus adaptors:
  - Processor-memory bus: mainly for processor-memory traffic
  - I/O buses: provide expansion slots for I/O devices
- Apple Macintosh-II
  - NuBus: Processor, memory, and a few selected I/O devices
  - SCCI Bus: the rest of the I/O devices

A Three-Bus System

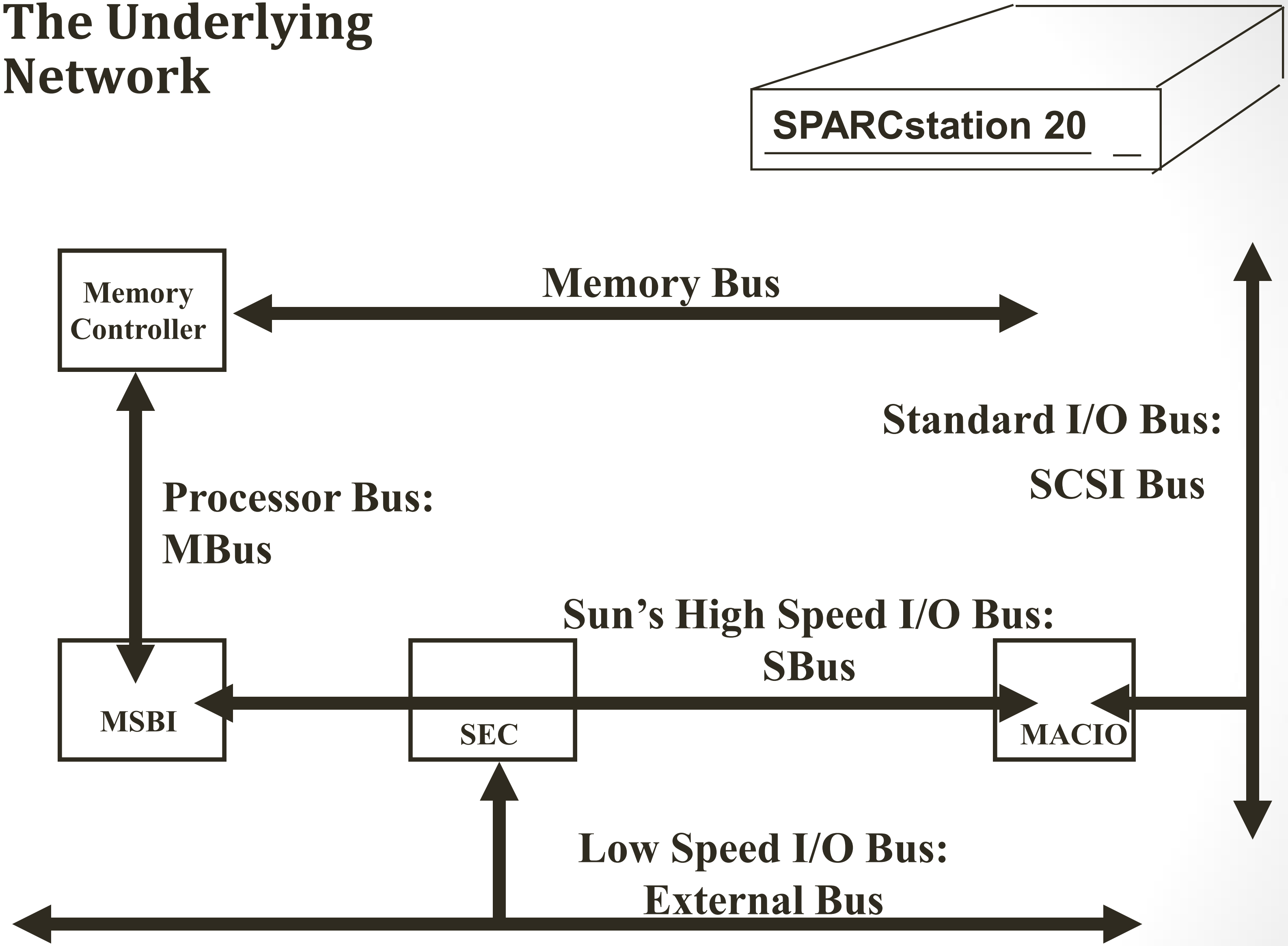


- A small number of backplane buses tap into the processor-memory bus
  - Processor-memory bus is used for processor memory traffic
  - I/O buses are connected to the backplane bus
- Advantage: loading on the processor bus is greatly reduced

Example Architecture:
The SPARCstation 20



The Underlying Network

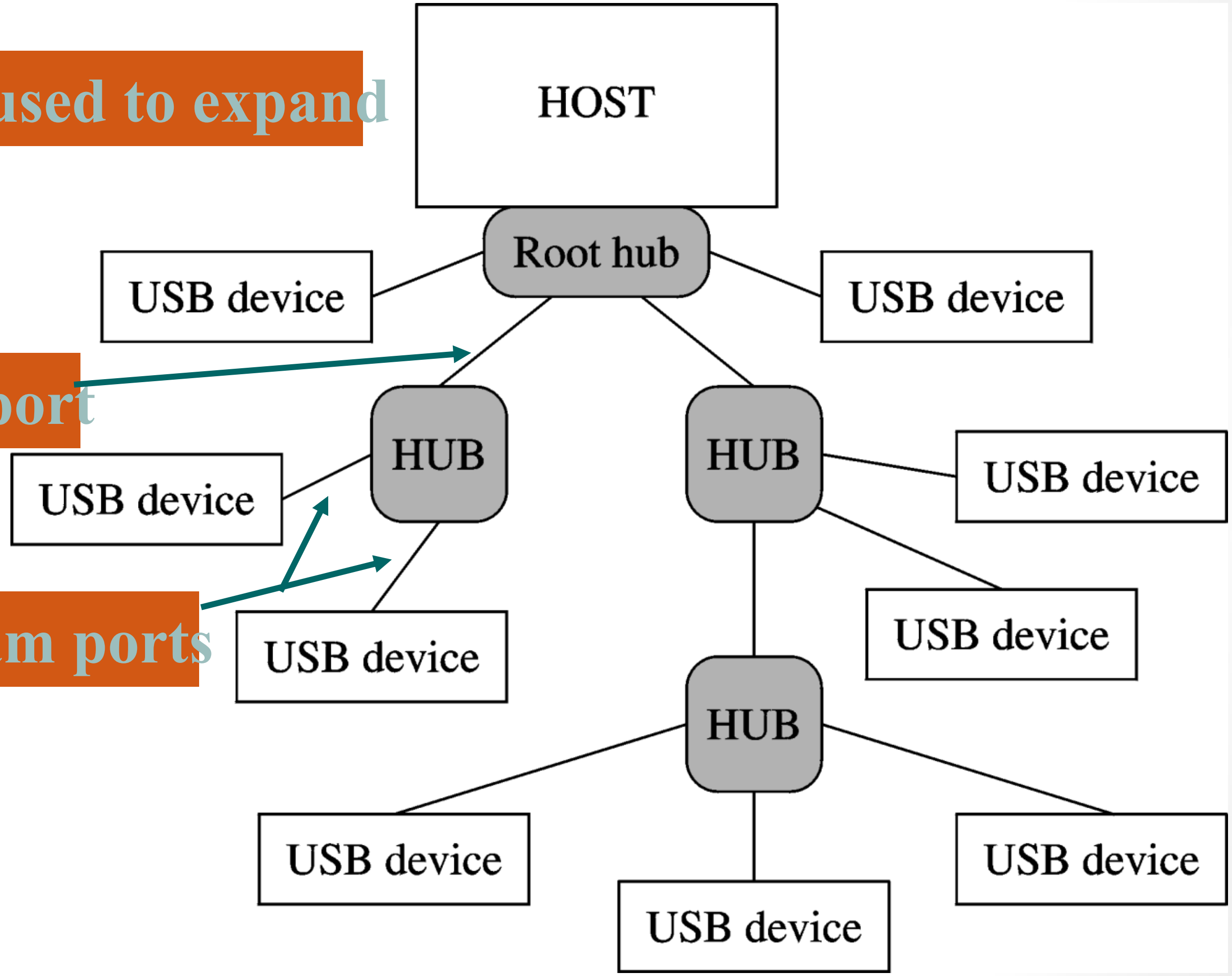


USB

- Universal Serial Bus
  - Originally developed in 1995 by a consortium including
    - Compaq, HP, Intel, Lucent, Microsoft, and Philips
  - USB 1.1 supports
    - Low-speed devices (1.5 Mbps)
    - Full-speed devices (12 Mbps)
  - USB 2.0 supports
    - High-speed devices
      - Up to 480 Mbps (a factor of 40 over USB 1.1)
    - Uses the same connectors
      - Transmission speed is negotiated on device-by-device basis

USB (cont'd)

Hubs can be used to expand



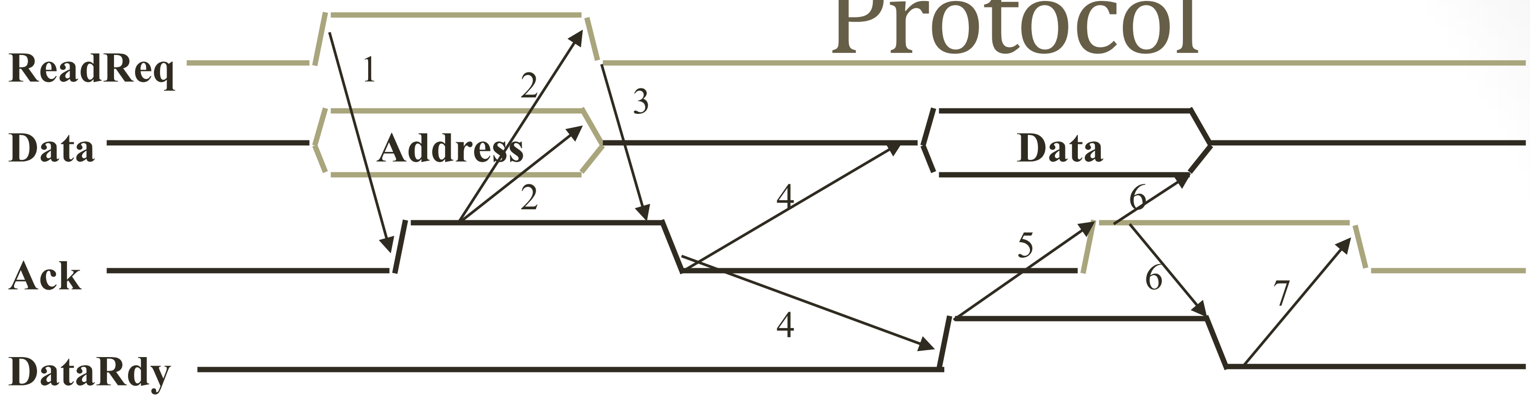
Upstream port

Downstream ports

Synchronous and Asynchronous Bus

- Synchronous Bus:
  - Includes a clock in the control lines
  - A fixed protocol for communication that is relative to the clock
  - Advantage: involves very little logic and can run very fast
  - Disadvantages:
    - Every device on the bus must run at the same clock rate
    - To avoid clock skew, they cannot be long if they are fast
- Asynchronous Bus:
  - It is not clocked
  - It can accommodate a wide range of devices
  - It can be lengthened without worrying about clock skew
  - It requires a handshaking protocol

A Handshaking Protocol



- Three control lines
  - ReadReq: indicate a read request for memory
    Address is put on the data lines at the same line
  - DataRdy: indicate the data word is now ready on the data lines
    Data is put on the data lines at the same time
  - Ack: acknowledge the ReadReq or the DataRdy of the other party

Increasing the Bus Bandwidth

- Separate versus multiplexed address and data lines:
  - Address and data can be transmitted in one bus cycle if separate address and data lines are available
  - Cost: (a) more bus lines, (b) increased complexity
- Data bus width:
  - By increasing the width of the data bus, transfers of multiple words require fewer bus cycles
  - Example: SPARCstation 20's memory bus is 128 bit wide
  - Cost: more bus lines
- Block transfers:
  - Allow the bus to transfer multiple words in back-to-back bus cycles
  - Only one address needs to be sent at the beginning
  - The bus is not released until the last word is transferred
  - Cost: (a) increased complexity
    (b) decreased response time for request

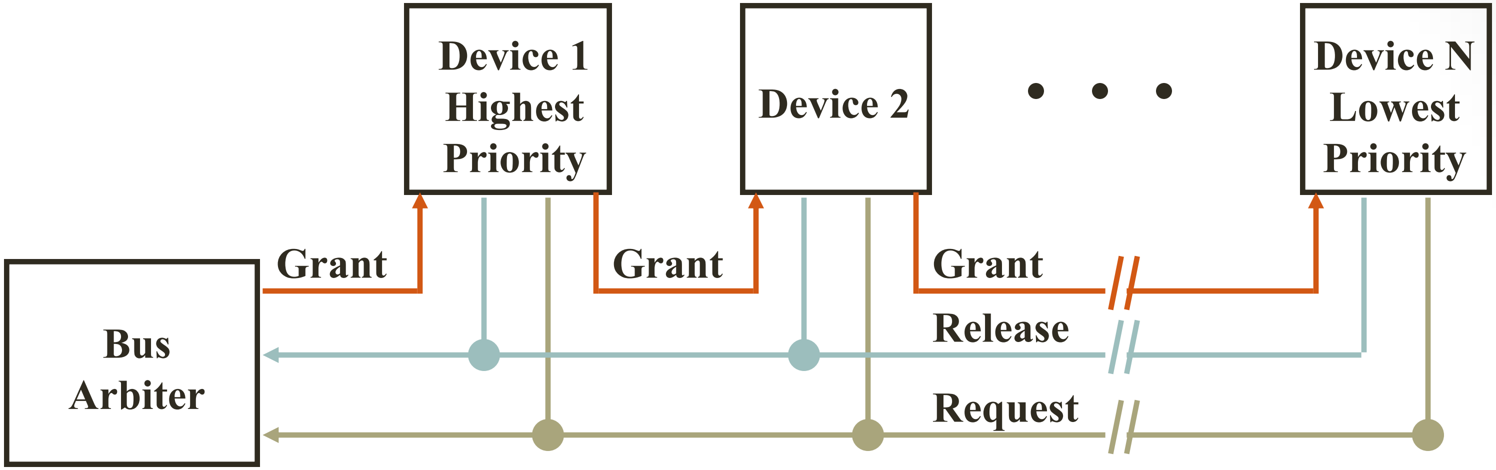
Obtaining Access to the Bus

- One of the most important issues in bus design:
  - How is the bus reserved by a devices that wishes to use it?
- Chaos is avoided by a master-slave arrangement:
  - Only the bus master can control access to the bus:
    It initiates and controls all bus requests
  - A slave responds to read and write requests
- The simplest system:
  - Processor is the only bus master
  - All bus requests must be controlled by the processor
  - Major drawback: the processor is involved in every transaction

Multiple Potential Bus Masters: the Need for Arbitration

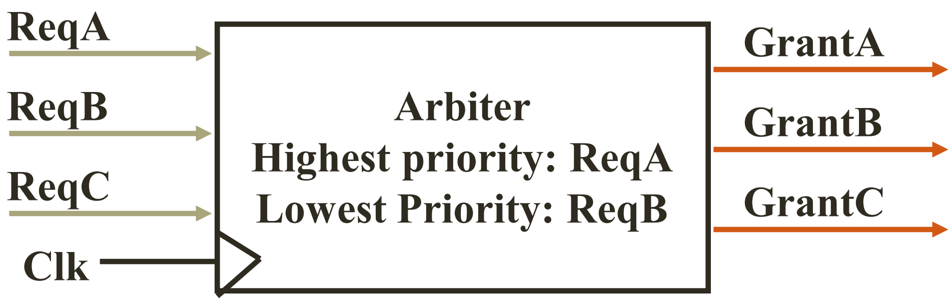
- Bus arbitration scheme:
  - A bus master wanting to use the bus asserts the bus request
  - A bus master cannot use the bus until its request is granted
  - A bus master must signal to the arbiter after finish using the bus
- Bus arbitration schemes usually try to balance two factors:
  - Bus priority: the highest priority device should be serviced first
  - Fairness: Even the lowest priority device should never be completely locked out from the bus
- Bus arbitration schemes can be divided into four broad classes:
  - Distributed arbitration by self-selection: each device wanting the bus places a code indicating its identity on the bus.
  - Distributed arbitration by collision detection: Ethernet uses this.
  - Daisy chain arbitration: see next slide.
  - Centralized, parallel arbitration: see next-next slide

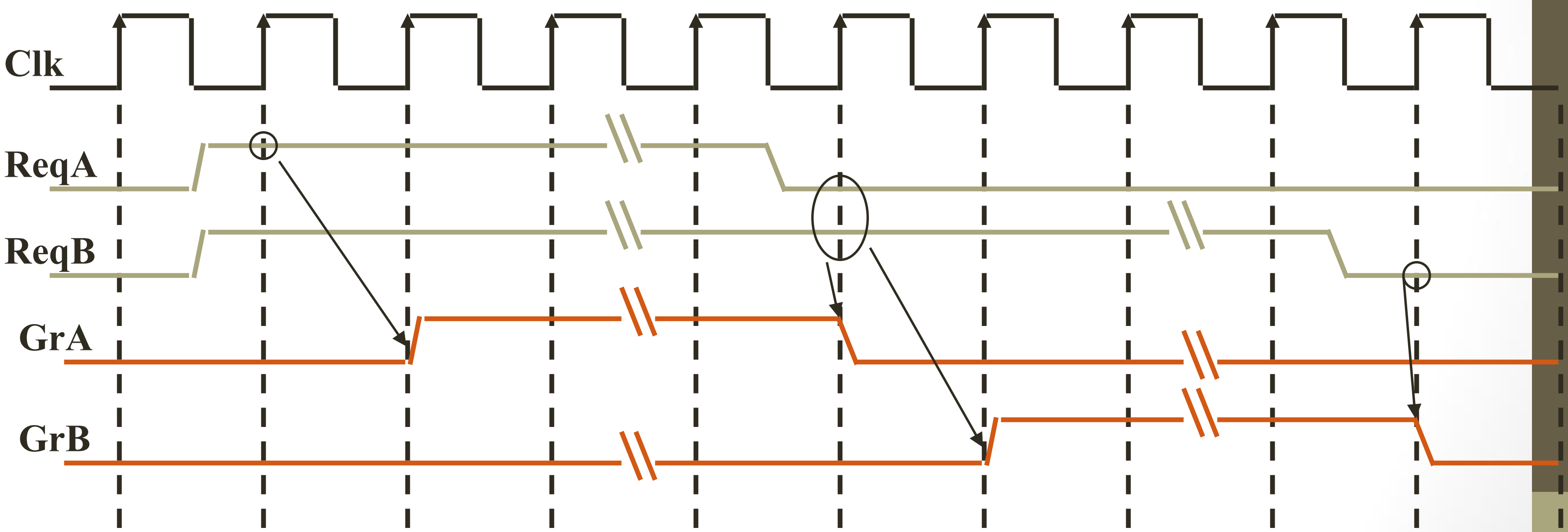
The Daisy Chain Bus Arbitrations Scheme



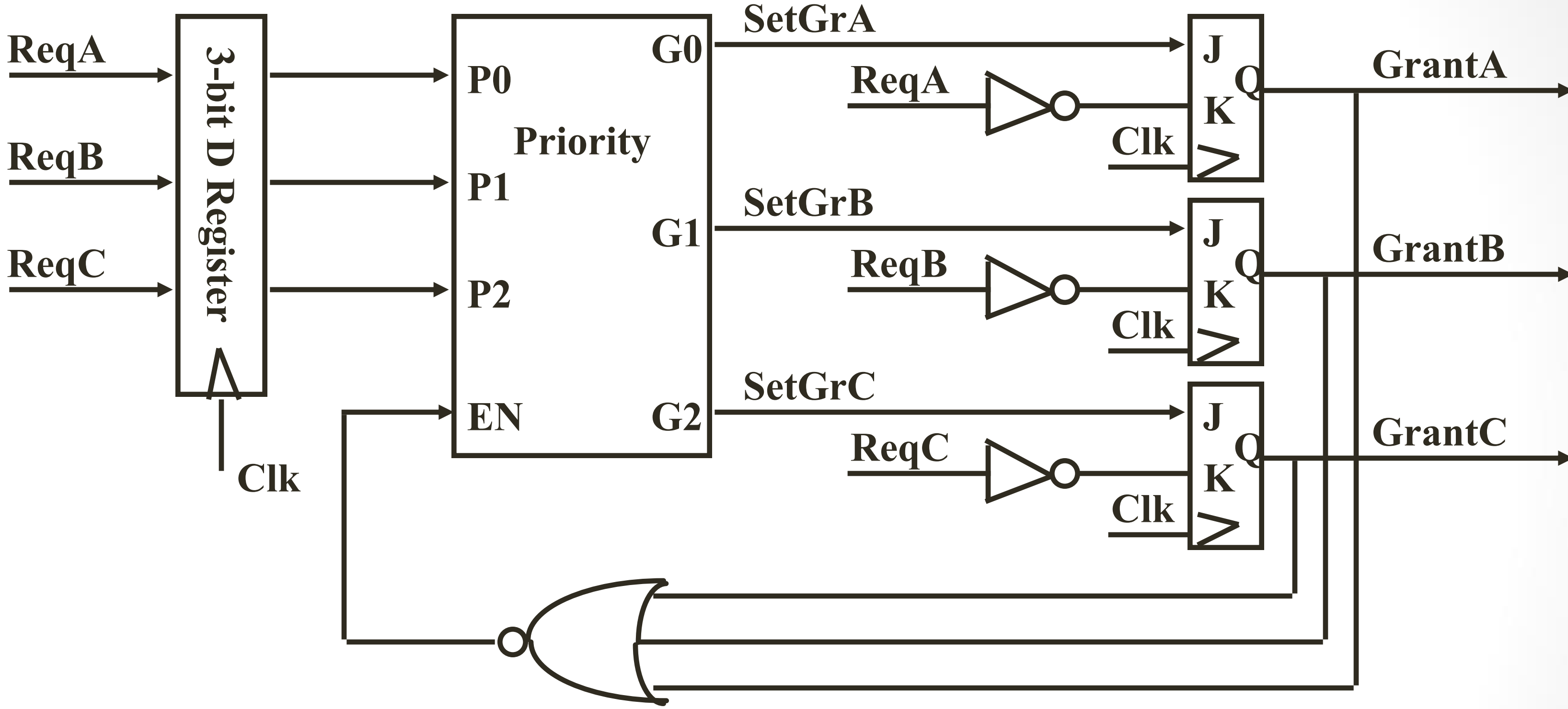
- Advantage: simple
- Disadvantages:
  - Cannot assure fairness:
    A low-priority device may be locked out indefinitely
  - The use of the daisy chain grant signal also limits the bus speed

Centralized Arbitration with a Bus Arbiter

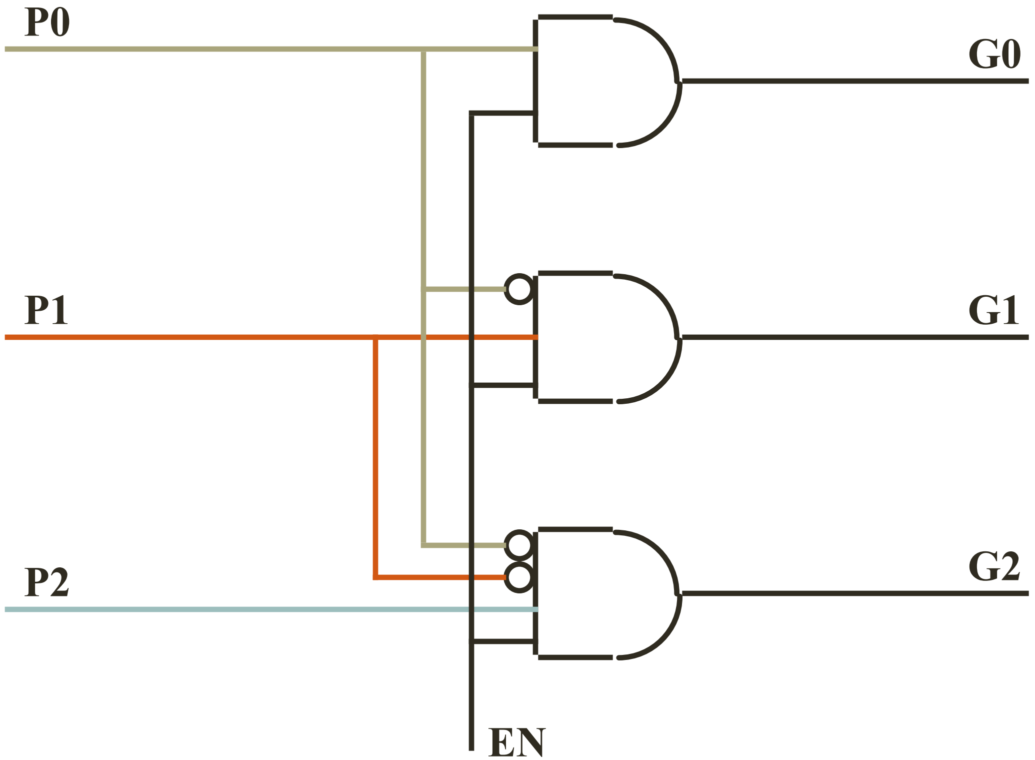




Simple Implementation of a Bus Arbiter



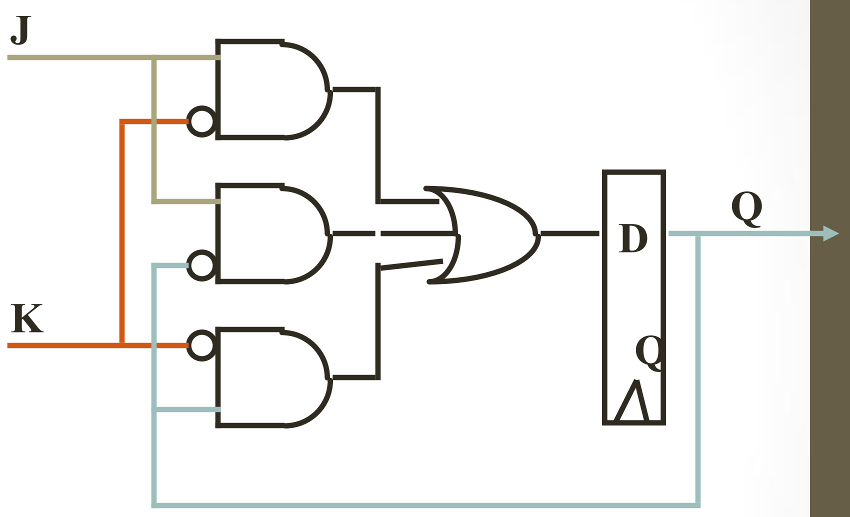
Priority Logic



JK Flip Flop

- JK Flip Flop can be implemented with a D-Flip Flop

J	K	Q(t-1)	Q(t)
0	0	0	0
0	0	1	1
0	1	x	0
1	0	x	1
1	1	0	1
1	1	1	0



Simple Implementation of a Bus Arbiter