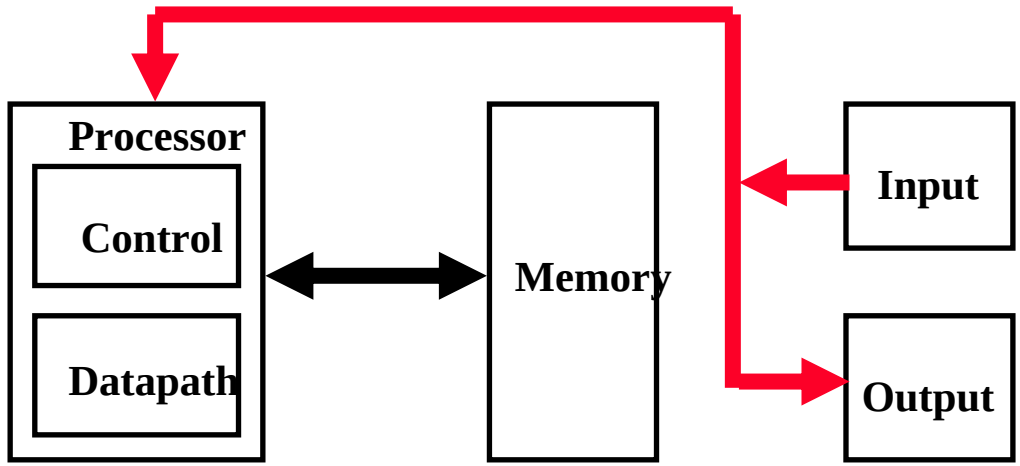
Computer Architecture
Bus Design

What is a bus?

- Slow vehicle that many people ride together
  - well, true...
- A bunch of wires...

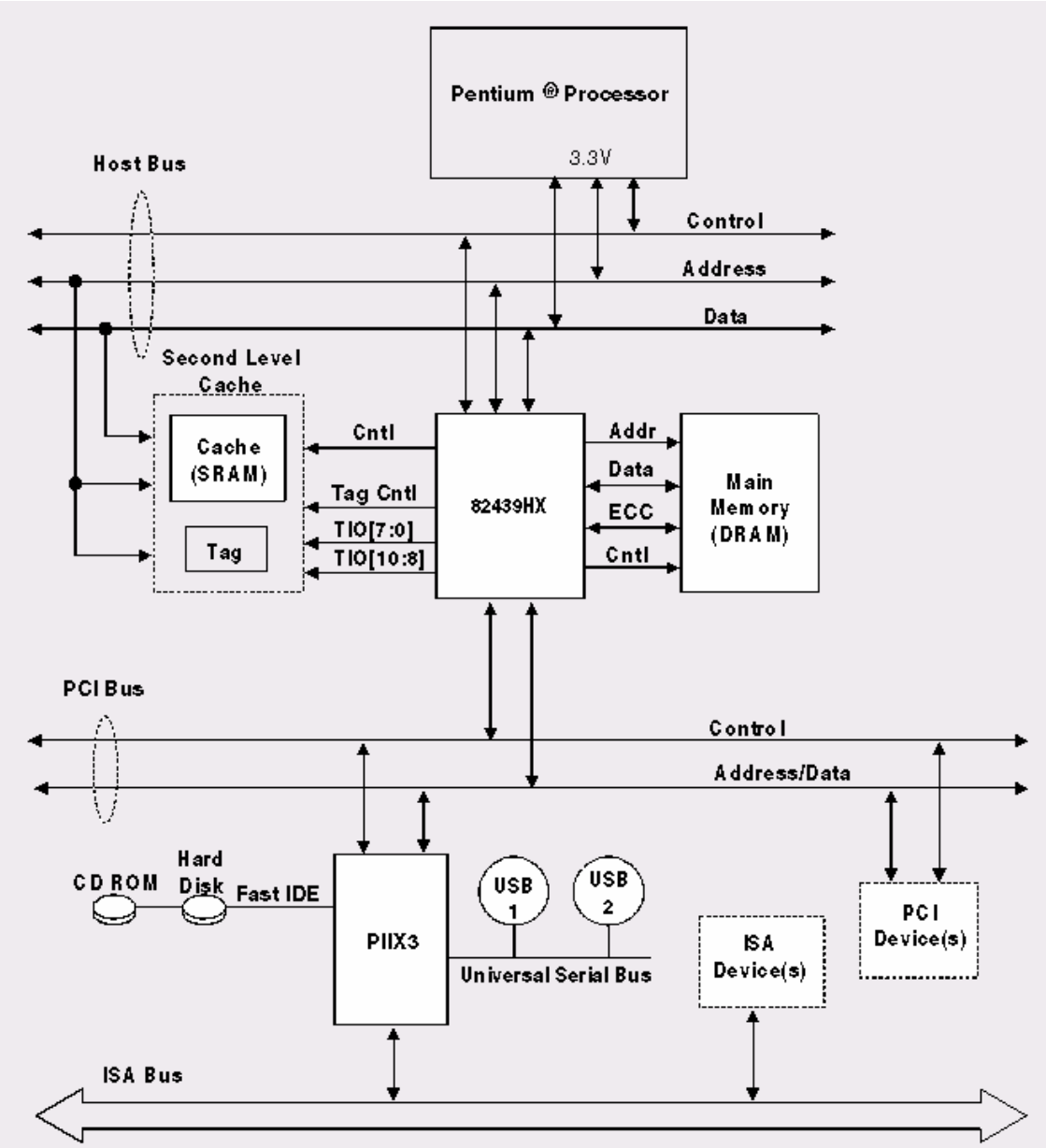
A Bus is:

- shared communication link
- single set of wires used to connect multiple subsystems



- **A Bus is also a fundamental tool for composing large, complex systems**
  - **systematic means of abstraction**

Example: Pentium System Organization

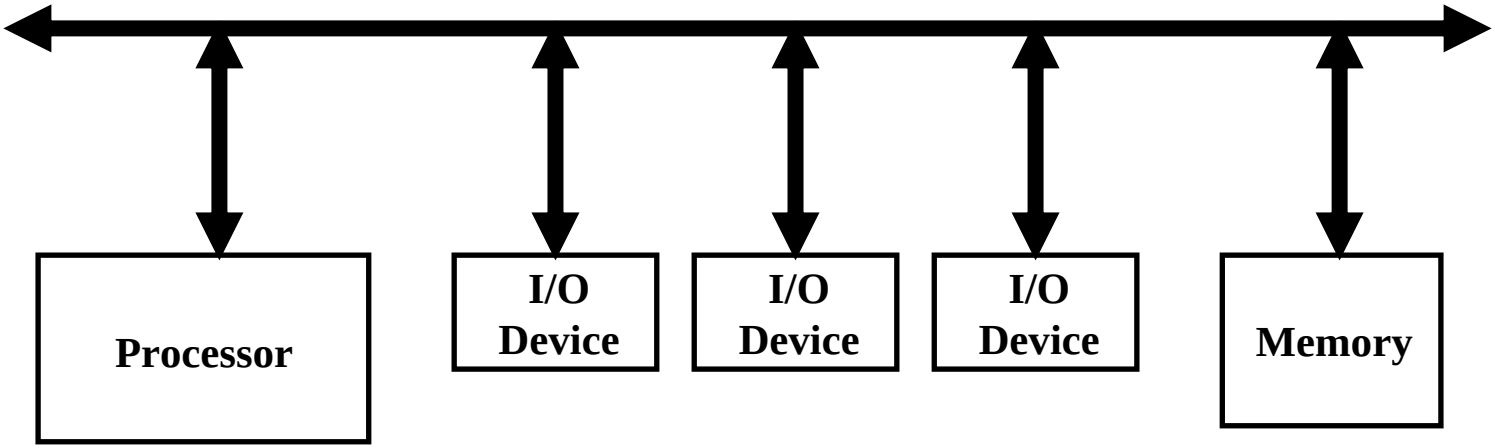


Processor/Memory Bus

PCI Bus

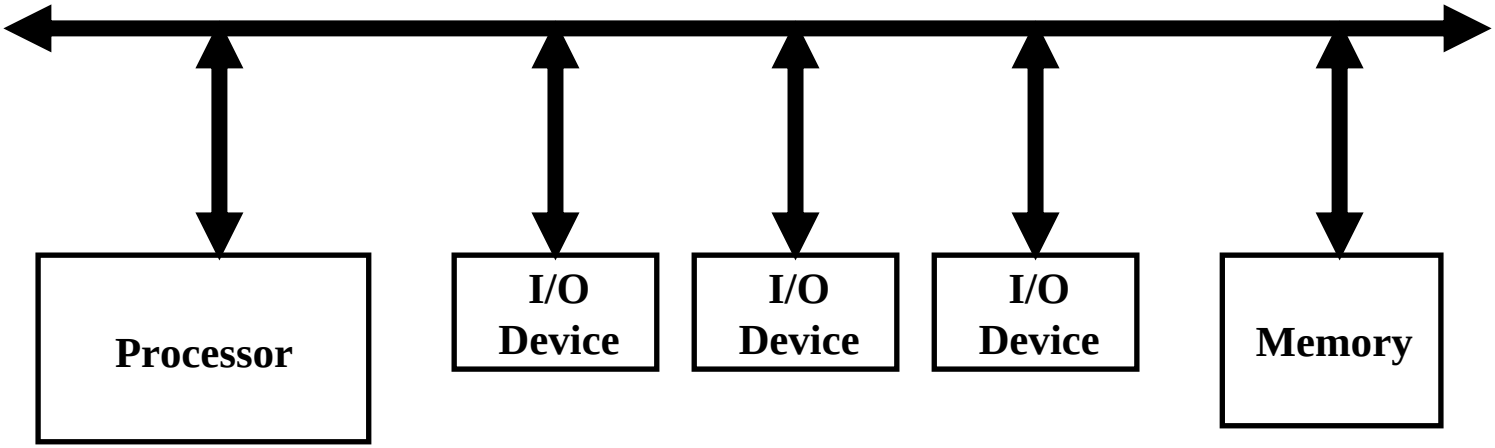
I/O Busses

Advantages of Buses



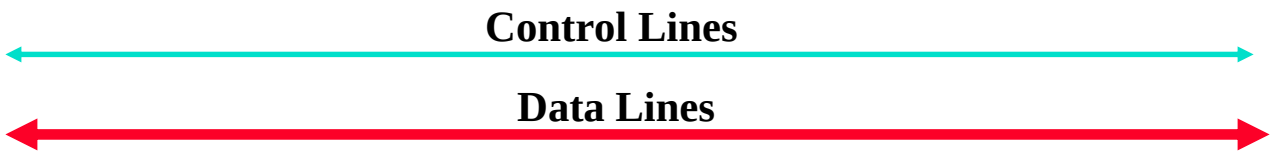
- **Versatility:**
  - New devices can be added easily
  - Peripherals can be moved between computer systems that use the same bus standard
- **Low Cost:**
  - A single set of wires is shared in multiple ways
- **Manage complexity by partitioning the design**

Disadvantage of Buses



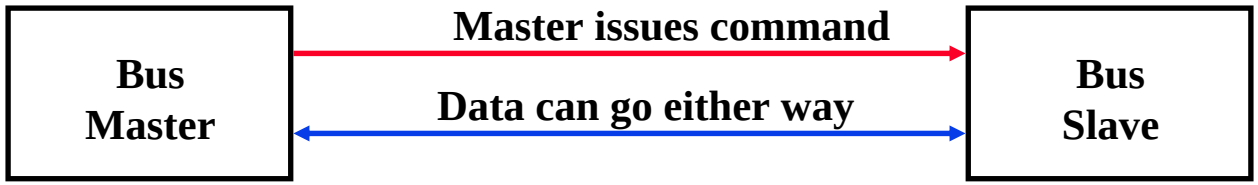
- It creates a communication bottleneck
  - The bandwidth of that bus can limit the maximum I/O throughput
- The maximum bus speed is largely limited by:
  - The length of the bus
  - The number of devices on the bus
  - The need to support a range of devices with:
    - Widely varying latencies
   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

Master versus Slave

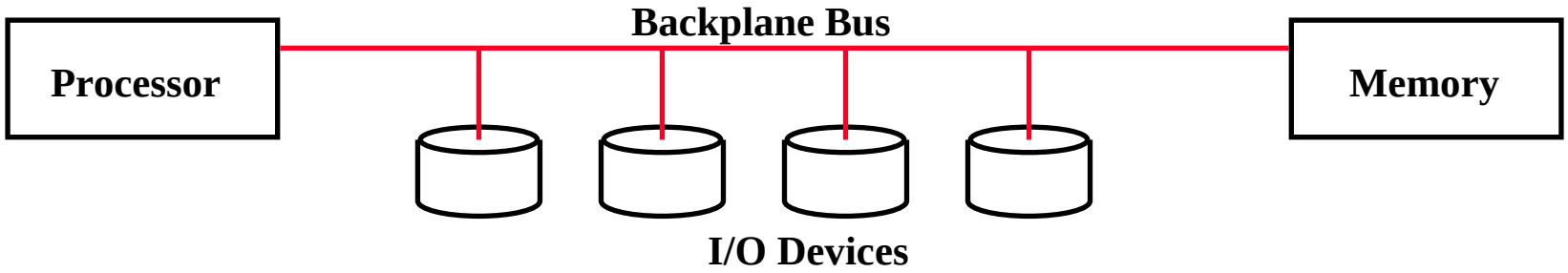


- A bus transaction includes two parts:
  - Issuing the command (and address) – request
  - Transferring the data – action
- Master is the one who starts the bus transaction by:
  - issuing the command (and address)
- Slave is the one who responds to the address by:
  - Sending data to the master if the master ask for data
  - Receiving data from the master if the master wants to send data

Types of Buses

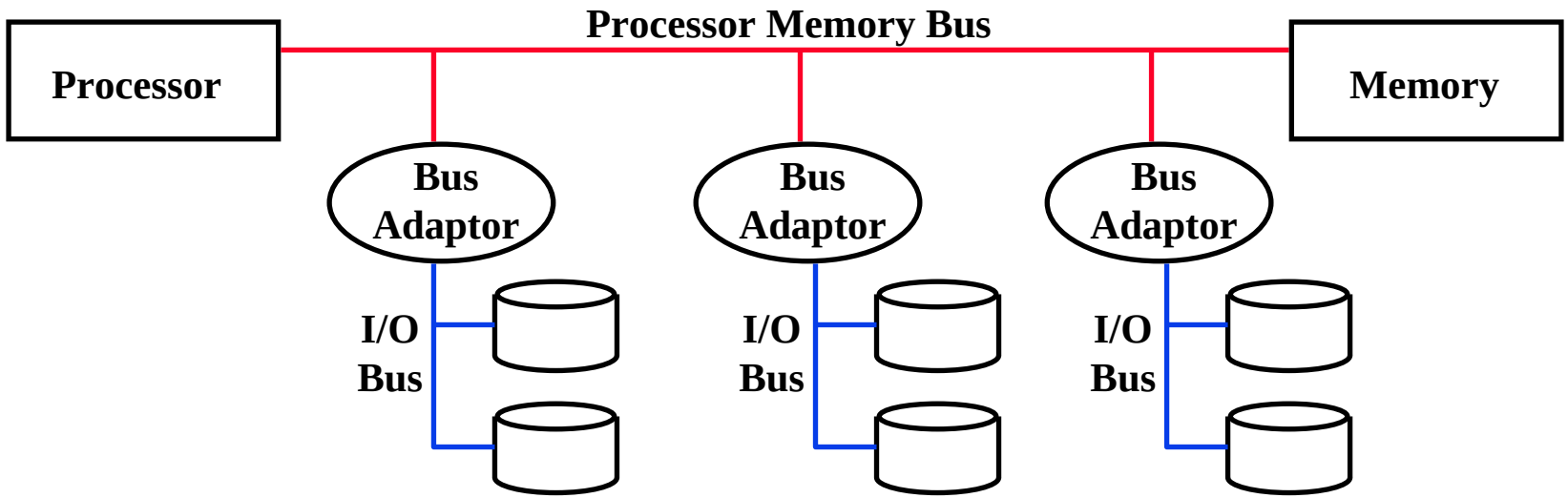
- **Processor-Memory Bus (design specific)**
  - Short and high speed
  - Only need to match the memory system
    - Maximize memory-to-processor bandwidth
  - Connects directly to the processor
  - Optimized for cache block transfers
- **I/O Bus (industry standard)**
  - Usually is lengthy and slower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

A Computer System with One Bus: Backplane Bus



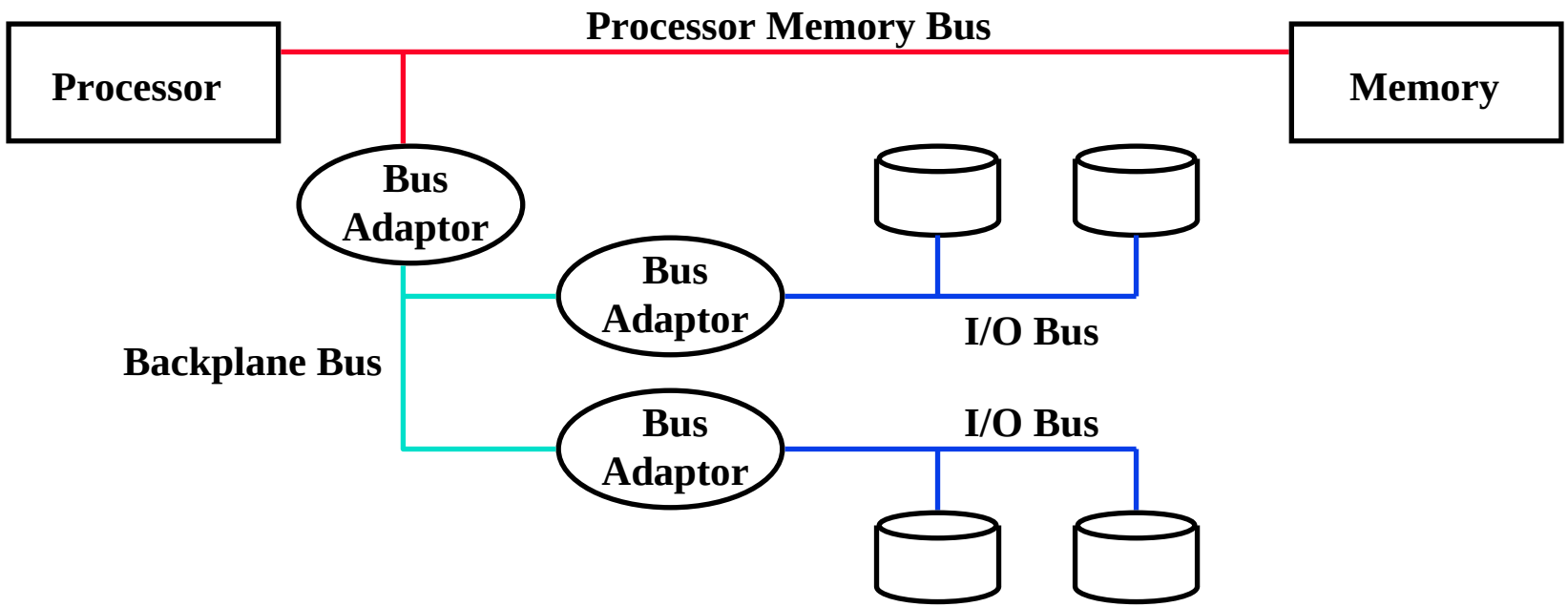
- A single bus (the backplane bus) is used for:
  - Processor to memory communication
  - Communication between I/O devices and memory
- Advantages: Simple and low cost
- Disadvantages: slow and the bus can become a major bottleneck
- Example: IBM PC - AT

A Two-Bus System



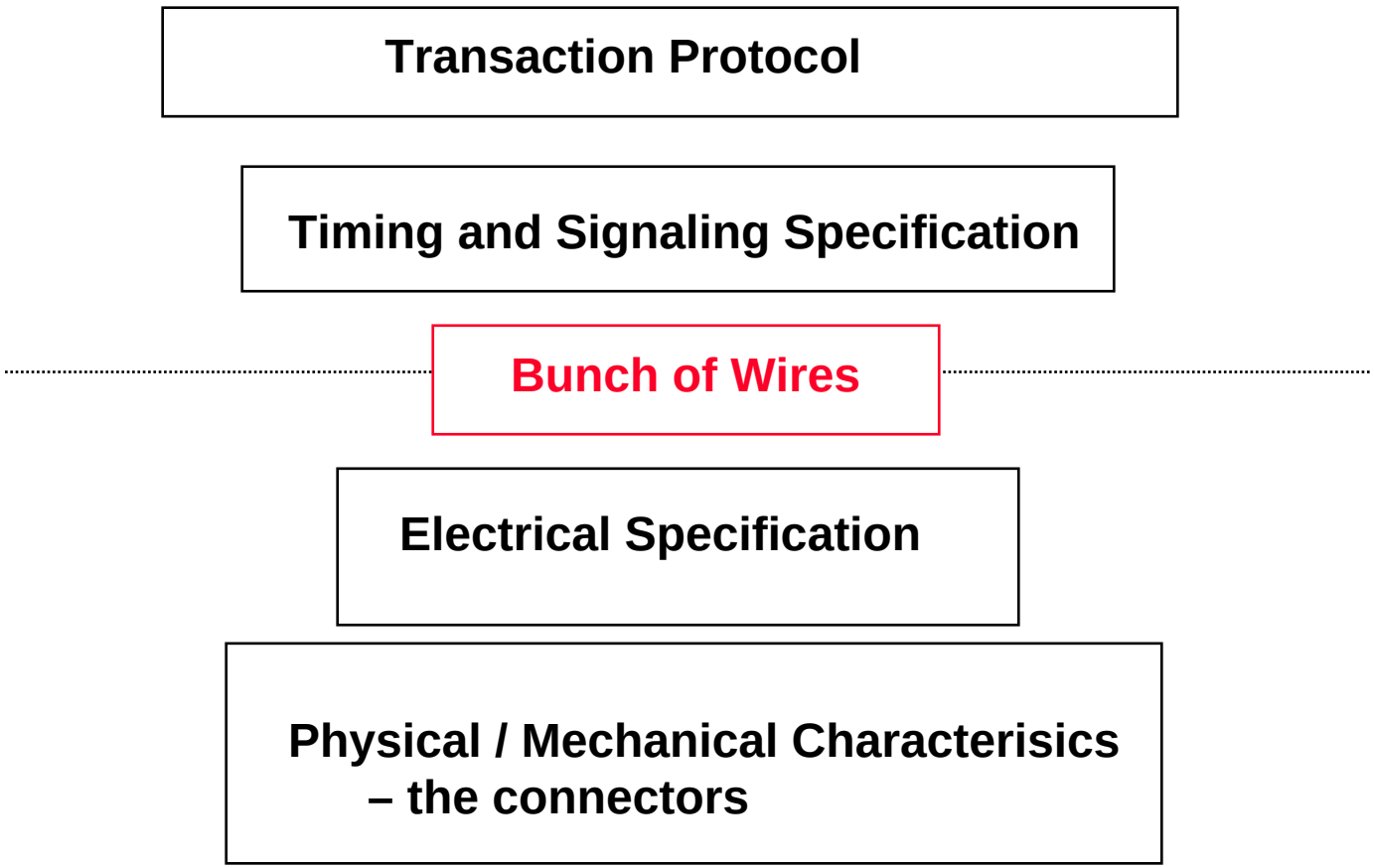
- I/O buses tap into the processor-memory bus via bus adaptors:
  - Processor-memory bus: mainly for processor-memory traffic
  - I/O buses: provide expansion slots for I/O devices
- Apple Macintosh-II
  - NuBus: Processor, memory, and a few selected I/O devices
  - SCCI Bus: the rest of the I/O devices

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

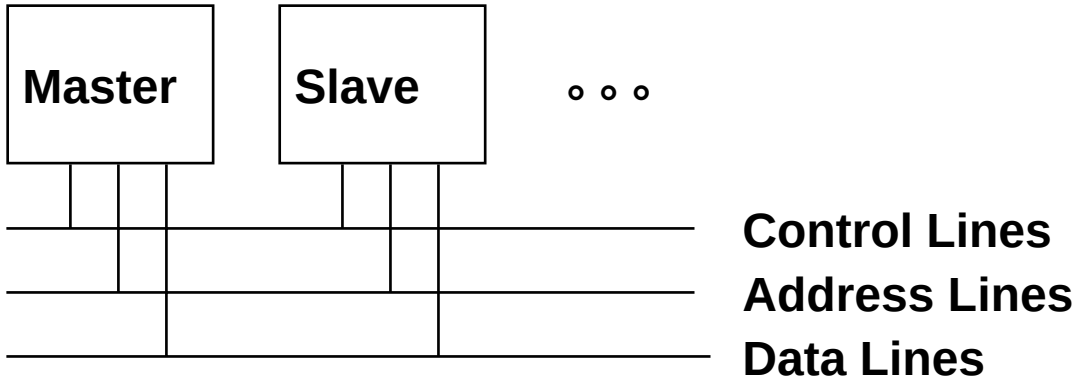
What defines a bus?



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

Busses so far



Multibus: 20 address, 16 data, 5 control, 50ns Pause

Bus Master: has ability to control the bus, initiates transaction

Bus Slave: module activated by the transaction

Bus Communication Protocol: specification of sequence of events and timing requirements in transferring information.

Asynchronous Bus Transfers: control lines (req, ack) serve to orchestrate sequencing.

Synchronous Bus Transfers: sequence relative to common clock.

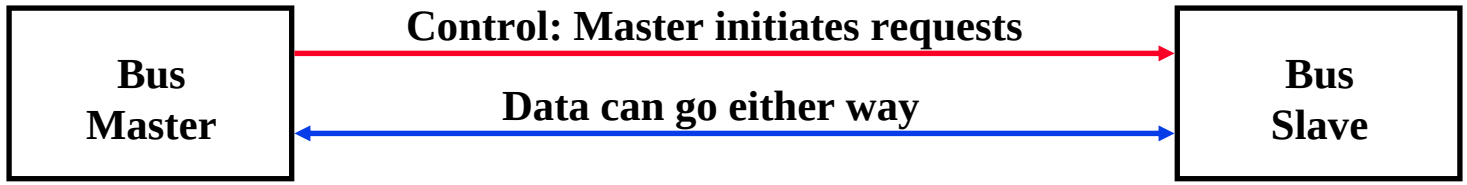
Administrative Issues

- Read P&H Chapter 8
- Intel “field trip” Friday 11/21
  - bus at 7 am !

Bus Transaction

- Arbitration
- Request
- Action

Arbitration: Obtaining Access to the Bus

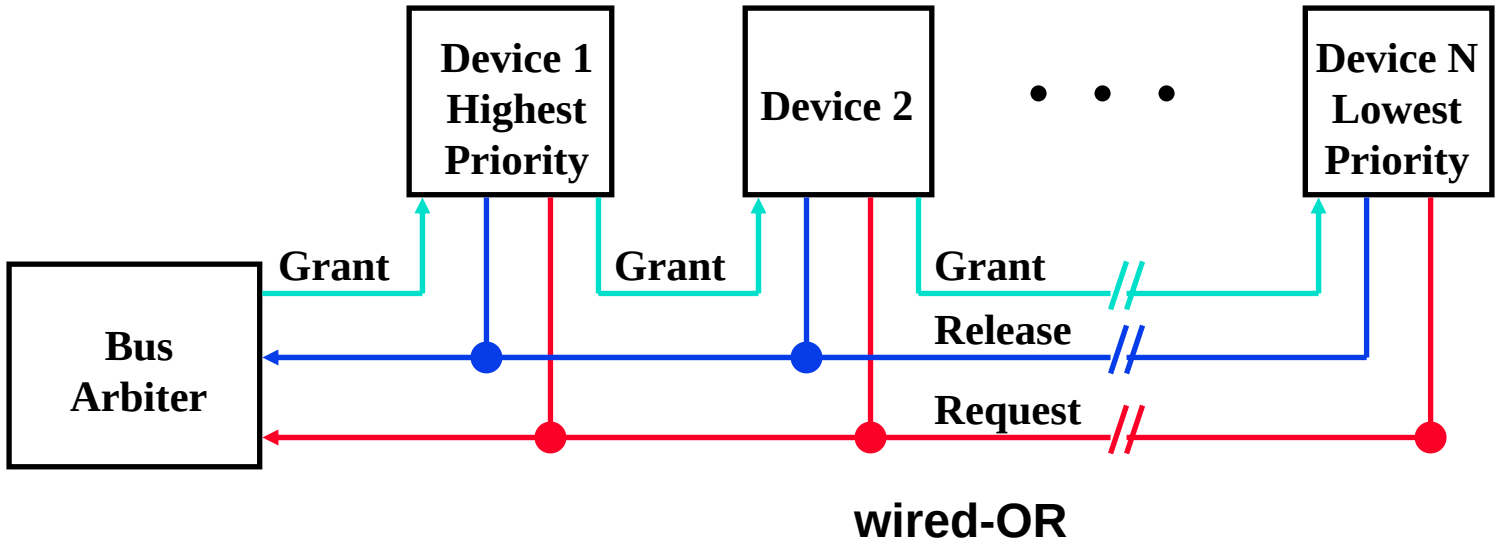


- One of the most important issues in bus design:
  - How is the bus reserved by a devices that wishes to use it?
- Chaos is avoided by a master-slave arrangement:
  - Only the bus master can control access to the bus:
    It initiates and controls all bus requests
  - A slave responds to read and write 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:**
  - Daisy chain arbitration: single device with all request lines.
  - Centralized, parallel arbitration: see next-next slide
  - 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.

The Daisy Chain Bus Arbitrations Scheme

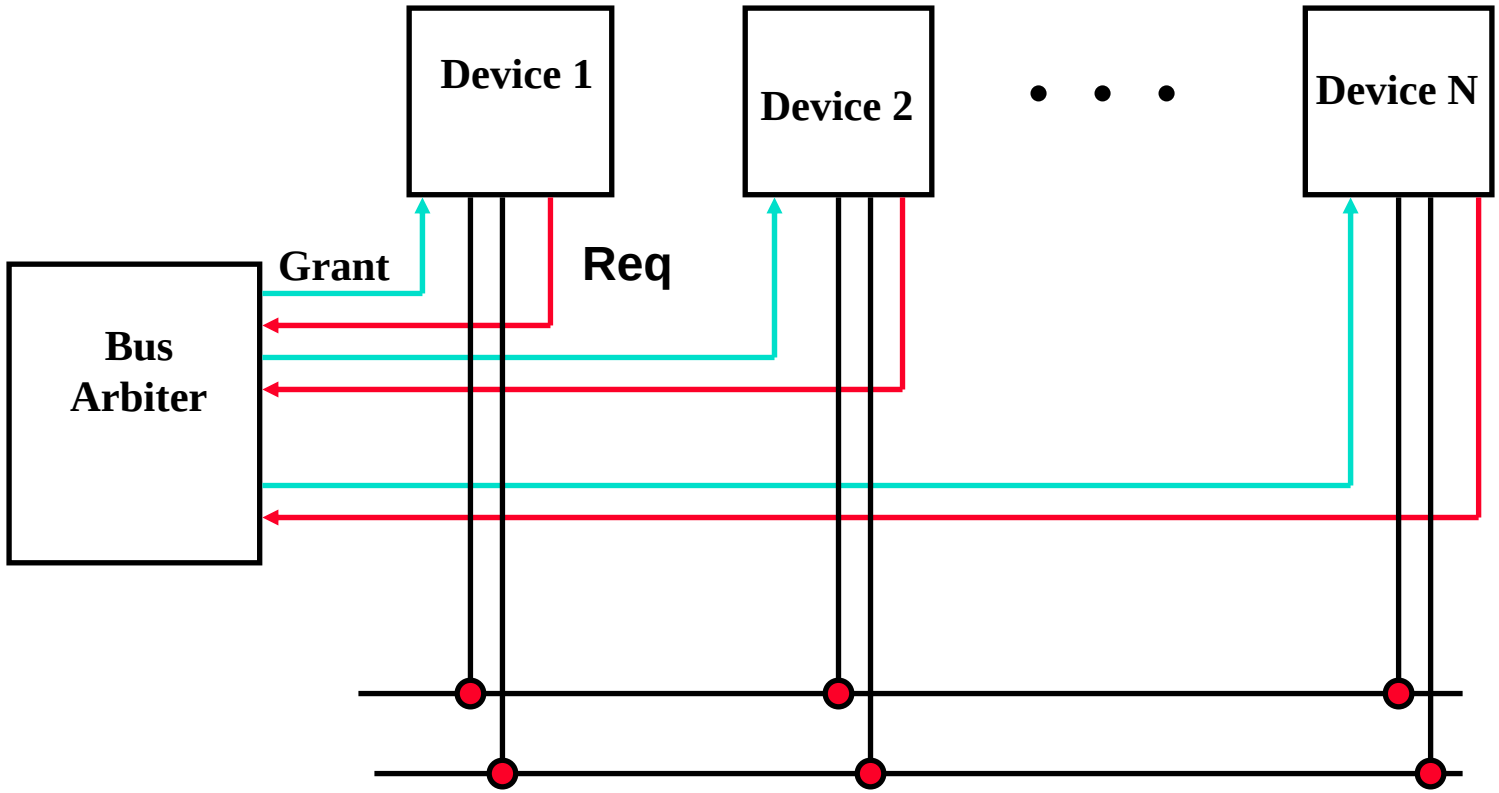


° **Advantage: simple**

° **Disadvantages:**

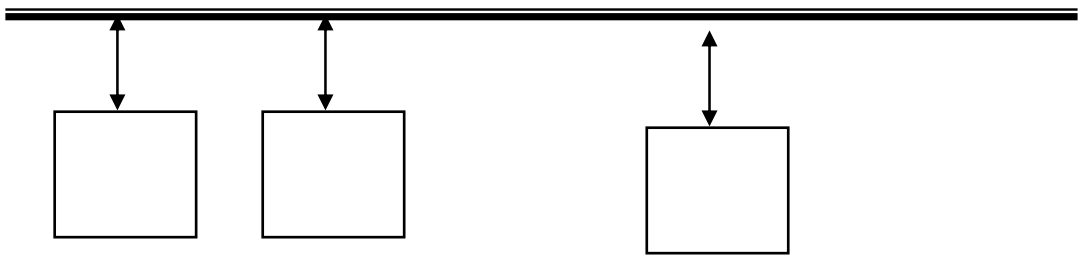
- **Cannot assure fairness:**
  A low-priority device may be locked out indefinitely
- **The use of the daisy chain grant signal also limits the bus speed**

Centralized Parallel Arbitration



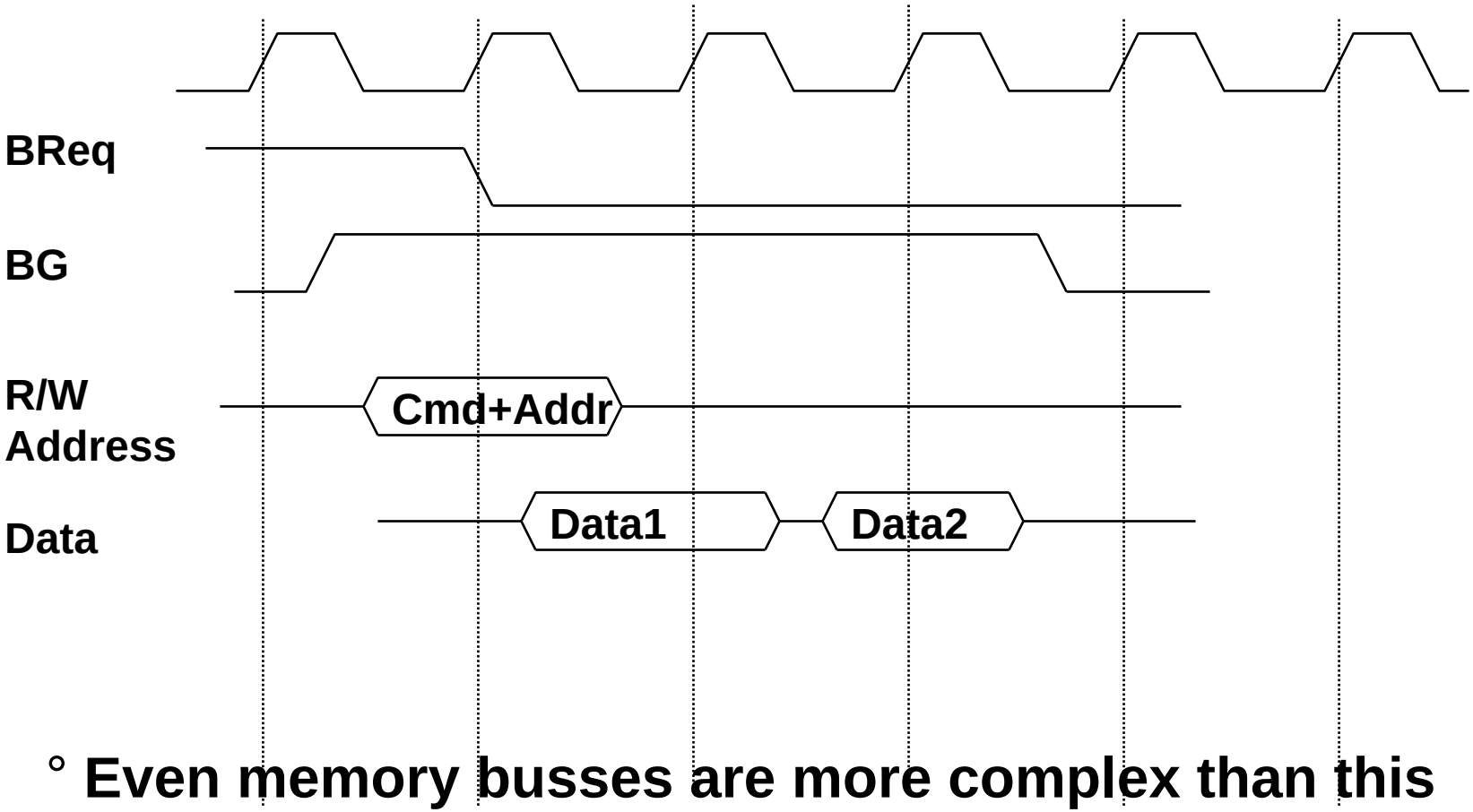
° Used in essentially all processor-memory busses and in high-speed I/O busses

Simplest bus paradigm



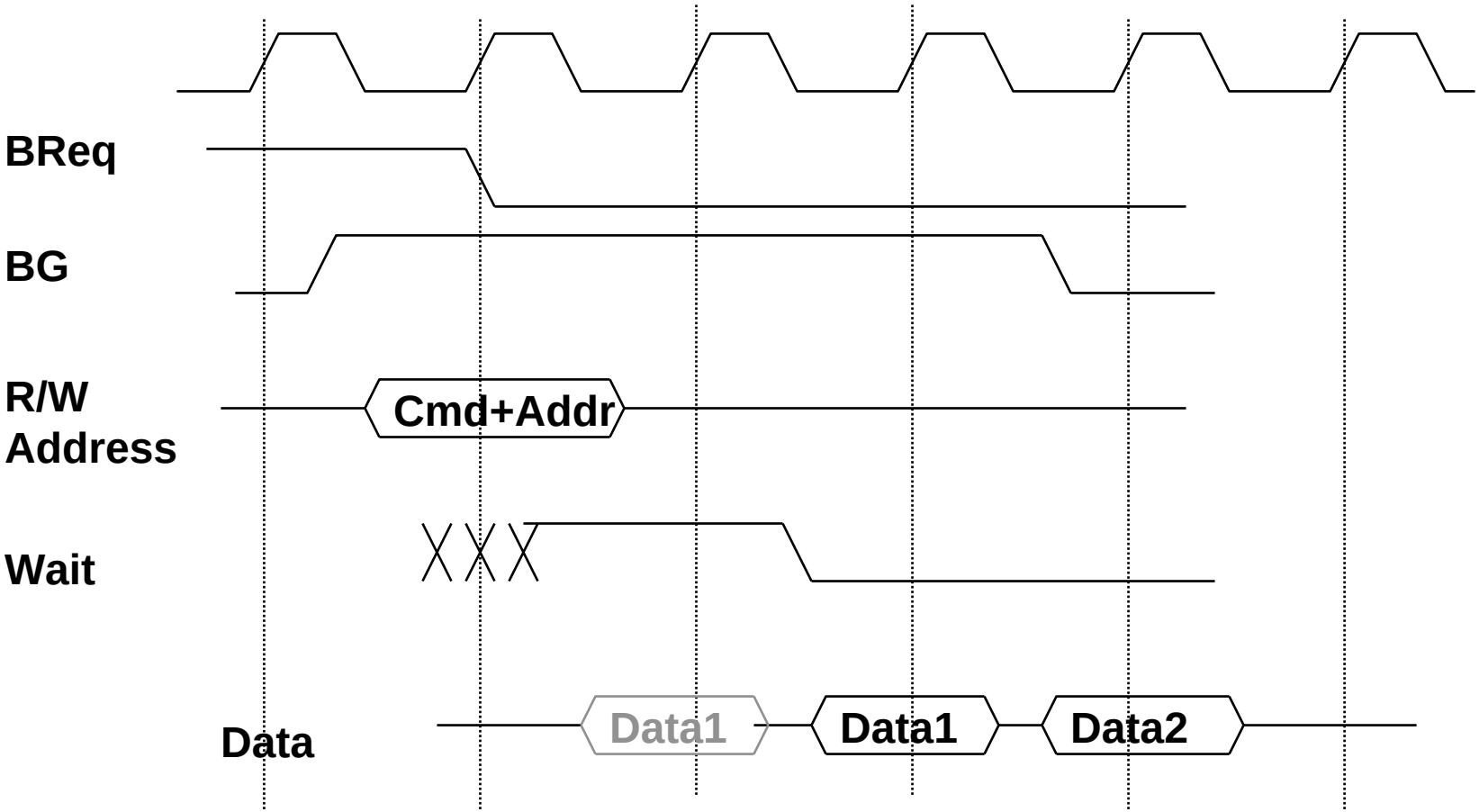
- All agents operate syncronously
- All can source / sink data at same rate
- => simple protocol
  - just manage the source and target

Simple Synchronous Protocol



- Even memory busses are more complex than this
  - memory (slave) may take time to respond
  - it need to control data rate

Typical Synchronous Protocol



- Slave indicates when it is prepared for data xfer
- Actual transfer goes at bus rate

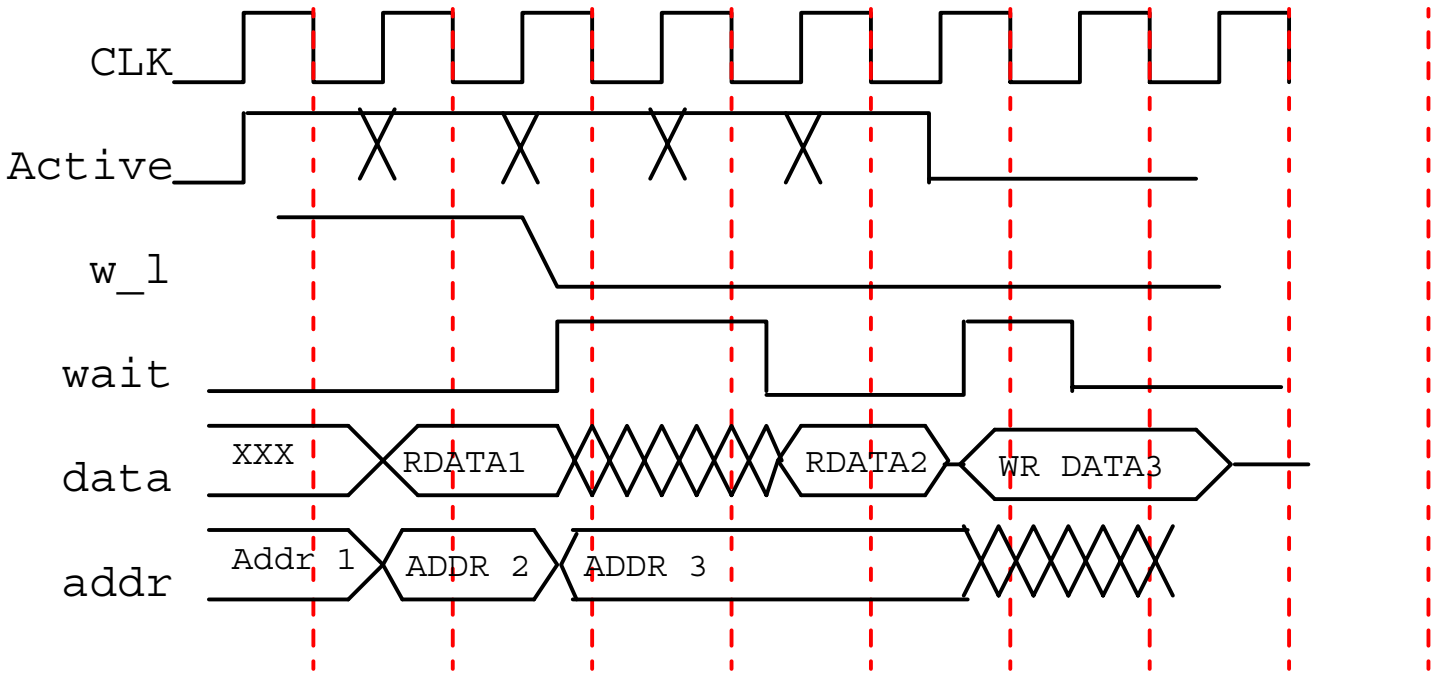
Increasing the Bus Bandwidth

- **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

Pipelined Bus Protocols

Attempt to initiate next address phase during current data phase

Single master example from your lab (proc-to-cache)



Increasing Transaction Rate on Multimaster Bus

- **Overlapped arbitration**
  - perform arbitration for next transaction during current transaction
- **Bus parking**
  - master can holds onto bus and performs multiple transactions as long as no other master makes request
- **Overlapped address / data phases (prev. slide)**
  - requires one of the above techniques
- **Split-phase (or packet switched) bus**
  - completely separate address and data phases
  - arbitrate separately for each
  - address phase yield a tag which is matched with data phase
- **”All of the above” in most modern mem busses**

1993 MP Server Memory Bus Survey: GTL revolution

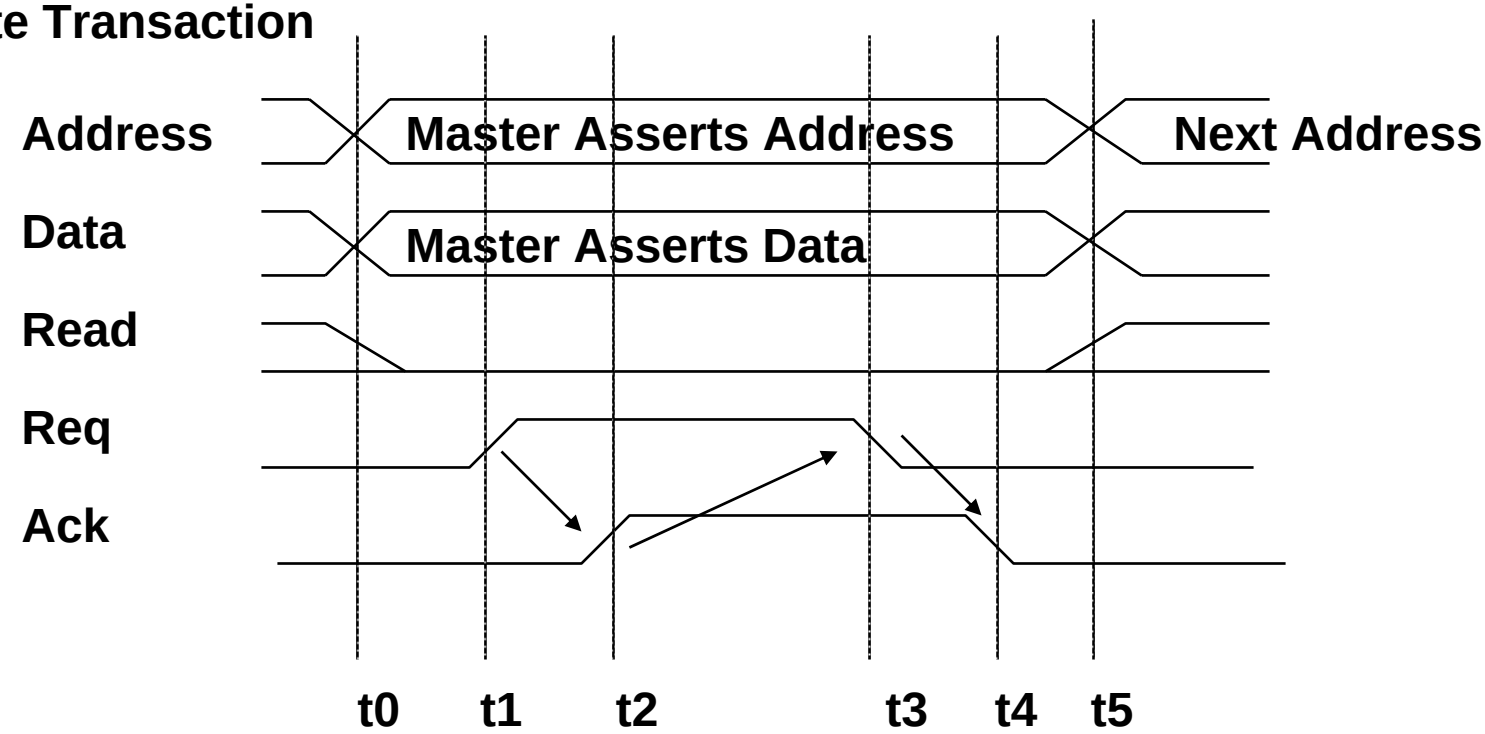
Bus	MBus	Summit	Challenge	XDBus
Originator	Sun	HP	SGI	Sun
Clock Rate (MHz)	40	60	48	66
Address lines	36	48	40	muxed
Data lines	64	128	256	144 (parity)
Data Sizes (bits)	256	512	1024	512
Clocks/transfer		4	5	4?
Peak (MB/s)	320(80)	960	1200	1056
Master	Multi	Multi	Multi	Multi
Arbitration	Central	Central	Central	Central
Slots		16	9	10
Busses/system	1	1	1	2
Length		13 inches	12? inches	17 inches

The I/O Bus Problem

- Designed to support wide variety of devices
  - full set not know at design time
- Allow data rate match between arbitrary speed deviced
  - fast processor – slow I/O
  - slow processor – fast I/O

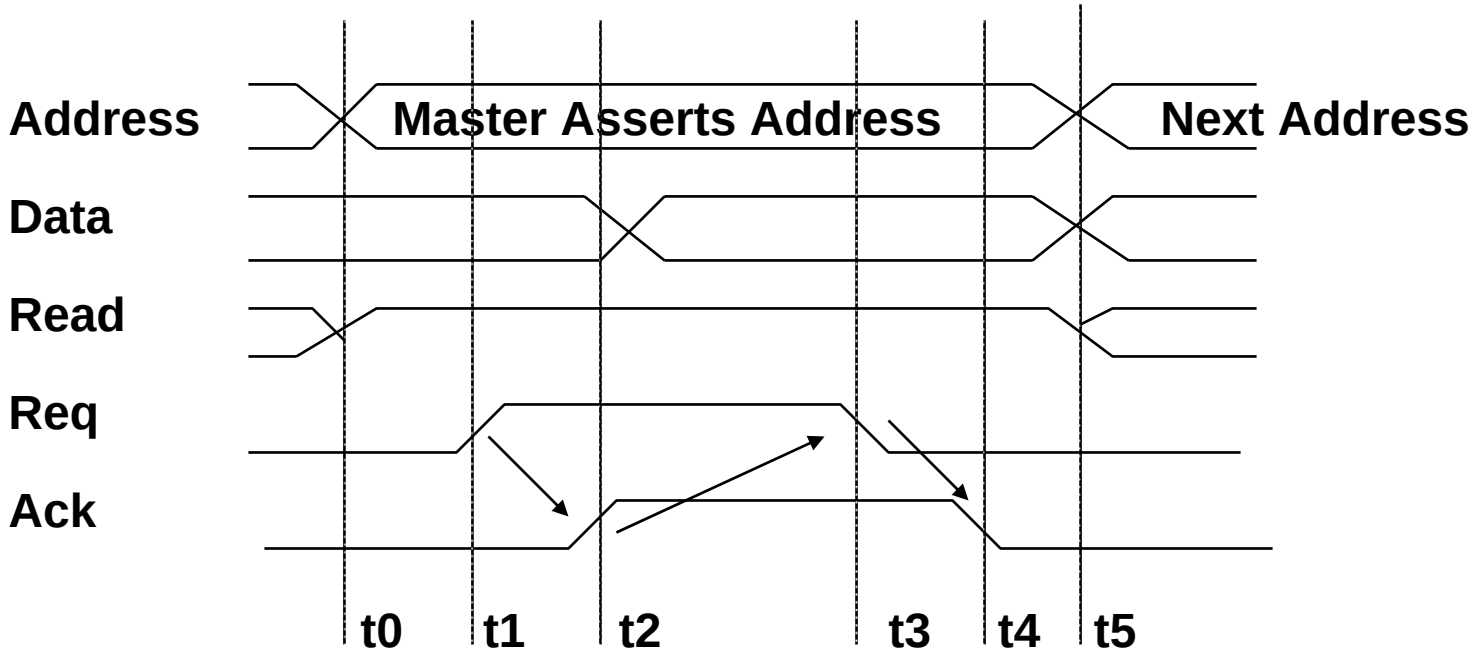
Asynchronous Handshake

Write Transaction



- t0 : Master has obtained control and asserts address, direction, data
- Waits a specified amount of time for slaves to decode target
- t1: Master asserts request line
- t2: Slave asserts ack, indicating data received
- t3: Master releases req
- t4: Slave releases ack

Read Transaction



- t0 : Master has obtained control and asserts address, direction, data
- Waits a specified amount of time for slaves to decode target\
- t1: Master asserts request line
- t2: Slave asserts ack, indicating ready to transmit data
- t3: Master releases req, data received
- t4: Slave releases ack

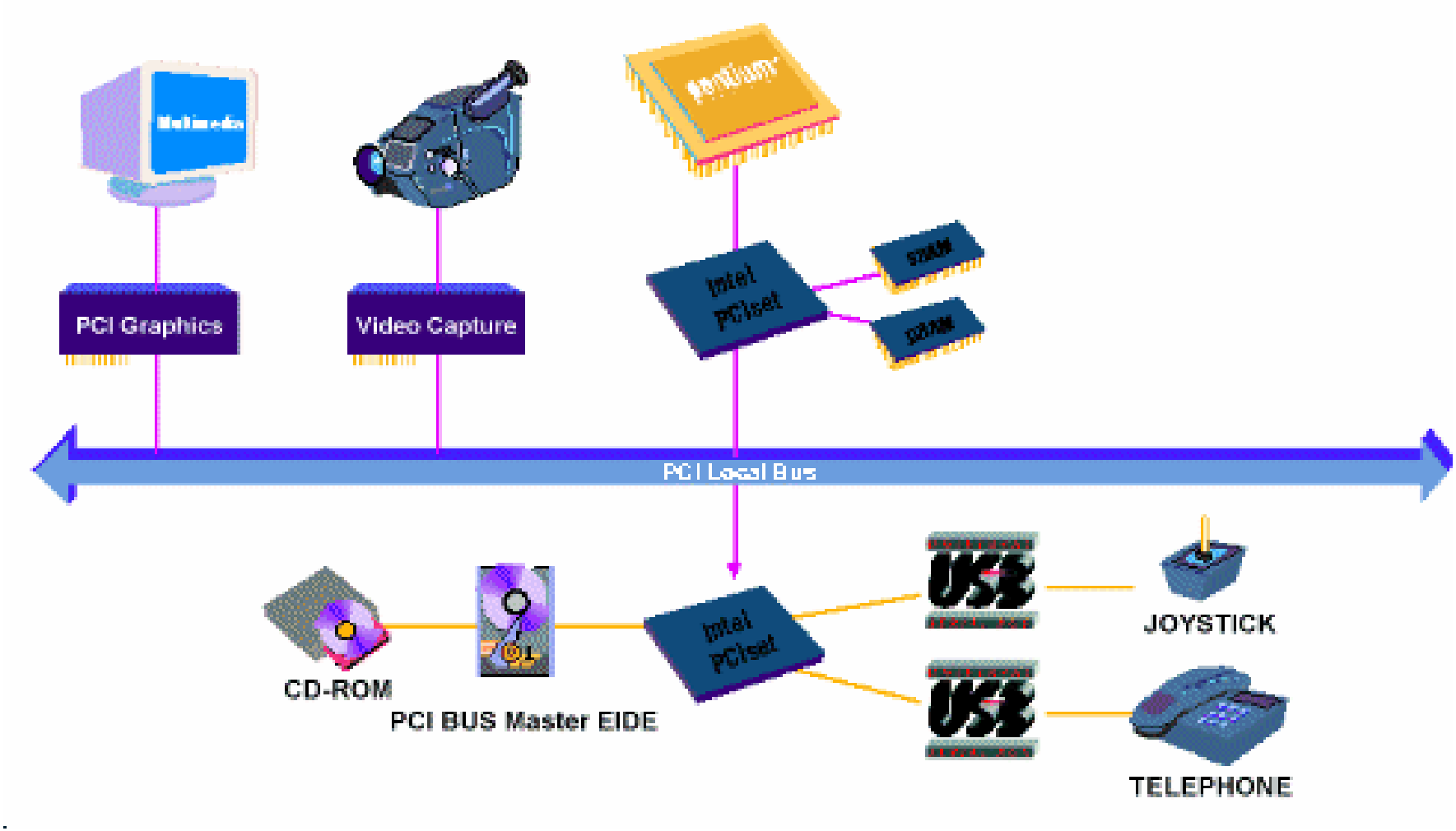
1993 Backplane/IO Bus Survey

Bus	SBus	TurboChannel	MicroChannel	PCI
Originator	Sun	DEC	IBM	Intel
Clock Rate (MHz)	16-25	12.5-25	async	33
Addressing	Virtual	Physical	Physical	Physical
Data Sizes (bits)	8,16,32	8,16,24,32	8,16,24,32,64	8,16,24,32,64
Master	Multi	Single	Multi	Multi
Arbitration	Central	Central	Central	Central
32 bit read (MB/s)	33	25	20	33
Peak (MB/s)	89	84	75	111 (222)
Max Power (W)	16	26	13	25

High Speed I/O Bus

- Examples
  - graphics
  - fast networks
- Limited number of devices
- Data transfer bursts at full rate
- DMA transfers important
  - small controller spools stream of bytes to or from memory
- Either side may need to squelch transfer
  - buffers fill up

Break


PCI Graphics
Video Capture
Intel PCIset
PCI Local Bus
CD-ROM
PCI BUS Master EIDE
Intel PCIset
JOYSTICK
TELEPHONE

PCI Read/Write Transactions

- All signals sampled on rising edge
- Centralized Parallel Arbitration
  - overlapped with previous transaction
- All transfers are (unlimited) bursts
- Address phase starts by asserting FRAME#
- Next cycle “initiator” asserts cmd and address
- Data transfers happen on when
  - IRDY# asserted by master when ready to transfer data
  - TRDY# asserted by target when ready to transfer data
  - transfer when both asserted on rising edge
- FRAME# deasserted when master intends to complete only one more data transfer

PCI Read Transaction

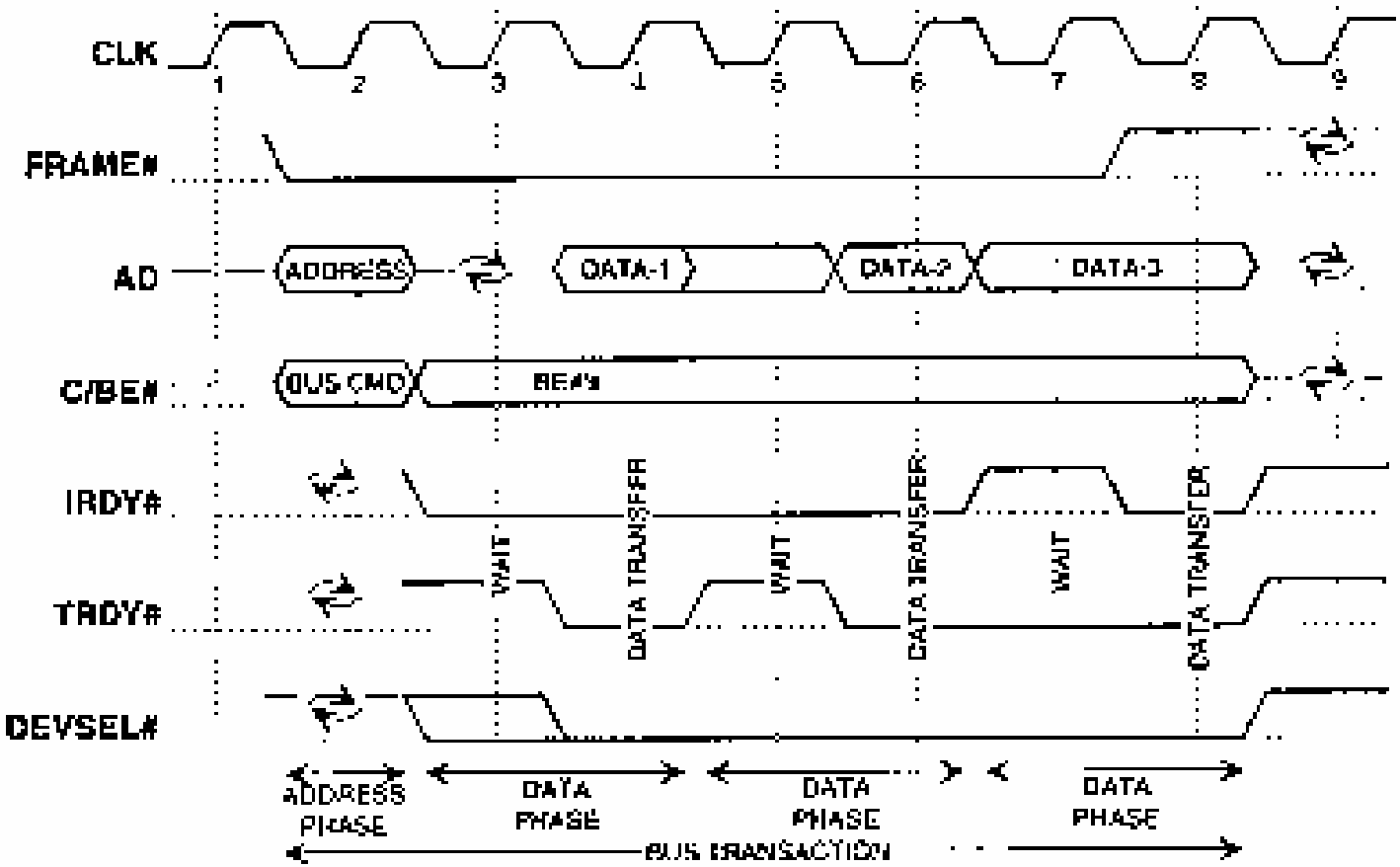


Figure 3-1: Basic Read Operation

– Turn-around cycle on any signal driven by more than one agent

PCI Write Transaction

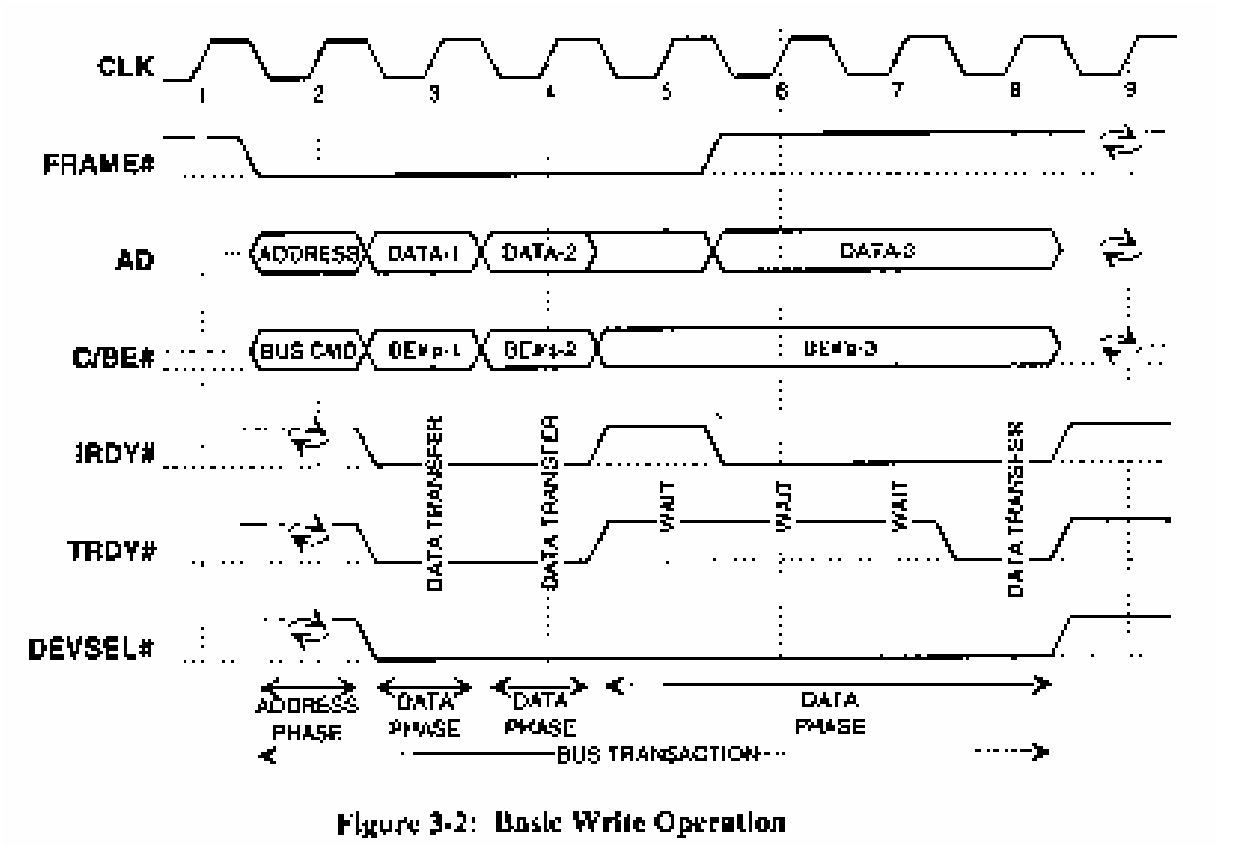


Figure 3-2: Basic Write Operation

PCI Optimizations

- **Push bus efficiency toward 100% under common simple usage**
  - **like RISC**
- **Bus Parking**
  - **retain bus grant for previous master until another makes request**
  - **granted master can start next transfer without arbitration**
- **Arbitrary Burst length**
  - **intiator and target can exert flow control with xRDY**
  - **target can disconnect request with STOP (abort or retry)**
  - **master can disconnect by deasserting FRAME**
  - **arbiter can disconnect by deasserting GNT**
- **Delayed (pended, split-phase) transactions**
  - **free the bus after request to slow device**

Additional PCI Issues

- Interrupts: support for controlling I/O devices
- Cache coherency:
  - support for I/O and multiprocessors
- Locks:
  - support timesharing, I/O, and MPs
- Configuration Address Space

Summary of Bus Options

°*Option*	*High performance*	*Low cost*
°Bus width	Separate address & data lines	Multiplex address & data lines
°Data width	Wider is faster (e.g., 32 bits)	Narrower is cheaper (e.g., 8 bits)
°Transfer size	Multiple words has less bus overhead	Single-word transfer is simpler
°Bus masters	Multiple (requires arbitration)	Single master (no arbitration)
°Clocking	Synchronous	Asynchronous
°Protocol	pipelined	Serial