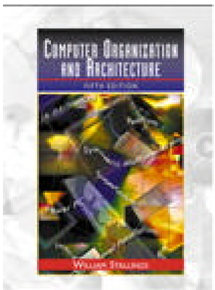
Campus de Gualtar
4710-057 Braga

UNIVERSIDADE DO MINHO
ESCOLA DE ENGENHARIA

Departamento de
Informática



Computer Organization and Architecture

5th Edition, 2000

by William Stallings

Table of Contents

II. THE COMPUTER SYSTEM.

3. System Buses. (29-Jan-01)

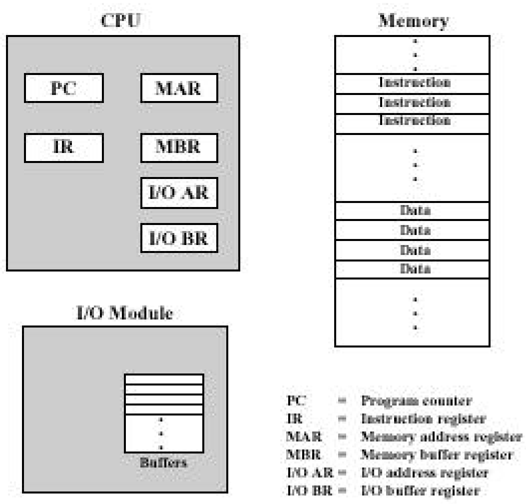
System Buses

Interconnecting Basic Components

Computer Components (3.1)

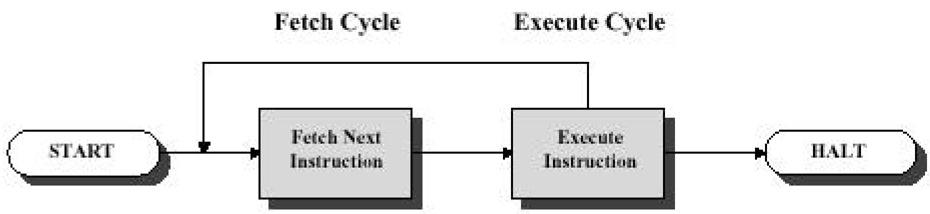
- The von Neumann architecture is based on three key concepts:
    - Data and instructions are stored in a single read-write memory
    - The contents of this memory are addressable by location, without regard to the type of data contained there
    - Execution occurs in a sequential fashion (unless explicitly modified) from one instruction to the next
- Two approaches to programming
    - hardwired programming - constructing a configuration of hardware logic components to perform a particular set of arithmetic and logic operations on a set of data
    - software - a sequence of codes or instructions, each of which supply the necessary control signals to a general-purpose configuration of control and logic functions (which may themselves be hardwired programs)
- Other components needed
    - I/O Components - a means to:
        - accept data and instructions in some form, and convert to an internal form of signals
        - report results
    - Main memory
        - distinguished from external storage/peripherals
        - a place to temporarily store both:
            - instructions - data interpreted as codes for generating control signals
            - data - data upon which computations are performed

- Interactions among Computer Components
    - Memory Address Register - specifies address for next read or write
    - Memory Buffer Register - contains data to be written into or receives data read from memory
    - I/O address register - specifies a particular I/O device
    - I/O buffer register - used for exchange of data between an I/O module and CPU (or memory)
    - Memory module - a set of locations
        - with sequentially numbered addresses
        - each holds a binary number that can be either an instruction or data



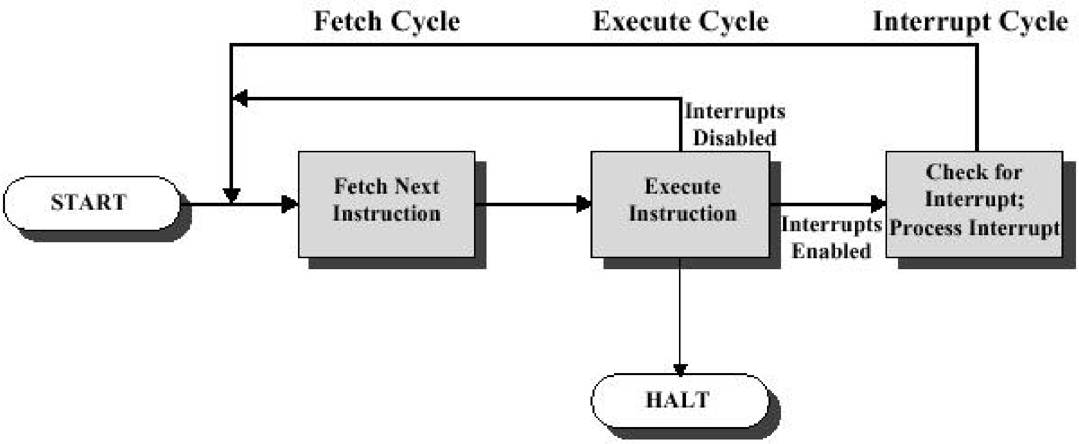
Computer Function (3.2)

- Processing required for a single instruction is called an instruction cycle
- Simple POV (Point-Of-View): 2 steps



  - Fetch - CPU reads an instruction from a location in memory
    - Program counter (PC) register keeps track of which instruction executes next
    - Normally, CPU increments PC after each fetch
    - Fetched instruction is loaded into the instruction register (IR)
  - Execute - CPU executes the instruction
    - May involve several operations
    - May utilize previously changed state of CPU and (indirectly) other devices
    - General categories:
      - CPU-Memory: Data may be transferred from CPU to memory or vice-versa
      - CPU-IO: Data may be transferred between CPU and an I/O module
      - Data Processing: CPU may perform some arithmetic or logic operation on the data
      - Control: An instruction may specify that the sequence of execution be altered
- More complex instructions
  - May combine these categories
  - May perform more than one reference to memory
  - May specify I/O operation instead of memory reference
  - May specify an operation to be performed on a vector of numbers or a string of characters
- Expanded execution cycle
  - Instruction Address Calculation (iac) - determine the address of the next instruction
  - Instruction Fetch (if)
  - Instruction Operation Decoding (iod) - analyze op to determine op type and operands
  - Operand Address Calculation (oac)
  - Operand Fetch (of)
  - Data Operation (do) - perform indicated op
  - Operand Store (os) - write result into memory or out to I/O
- Interrupts
  - Mechanism by which other modules may interrupt the normal processing of the CPU
  - Classes
    - Program - as a result of program execution
    - Timer - generated by hardware timer
    - I/O - to signal completion of I/O or error
    - Hardware failure

- Instruction cycle with interrupts

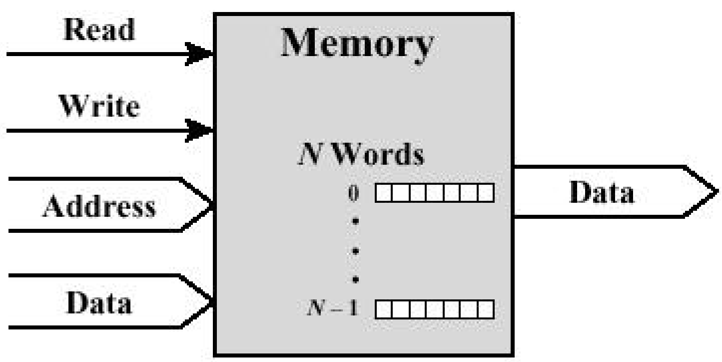


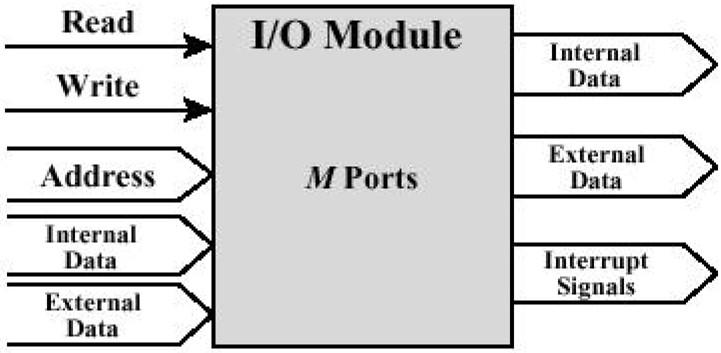
- When an interrupt signal is generated, the processor:
    - Suspends execution of the current program and saves its context (such as PC and other registers)
    - Sets PC to starting address of an interrupt handler routine
- Multiple interrupts
    - Can be handled by disabling some or all interrupts. Disabled interrupts generally remain pending and are handled sequentially
    - Can be handled by prioritizing interrupts, allowing a higher priority interrupt to interrupt one of lower priority

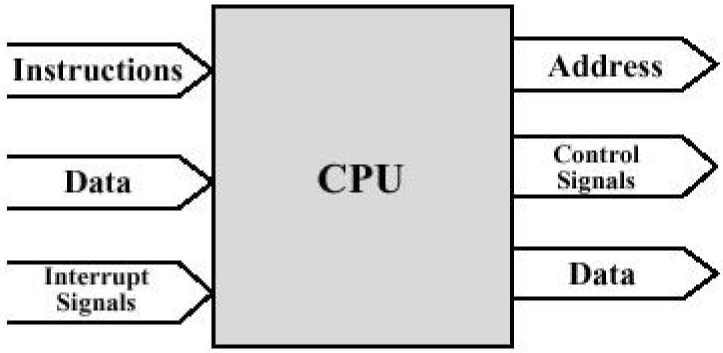
- Physical Interrupts
    - Interrupts are represented as one or more lines in the system bus
        - One line: polling - when line goes high, CPU polls devices to determine which caused interrupt
        - Multiple lines: addressable interrupts - combination of lines indicates both interrupt and which device caused it. Ex. 386 based architectures use 4 bit interrupts, allowing IRQ's 0-15 (with an extra line to signal pending)

Interconnection Structures (3.3)

- The collection of paths connecting the various modules of a computer (CPU, memory, I/O) is called the interconnection structure.



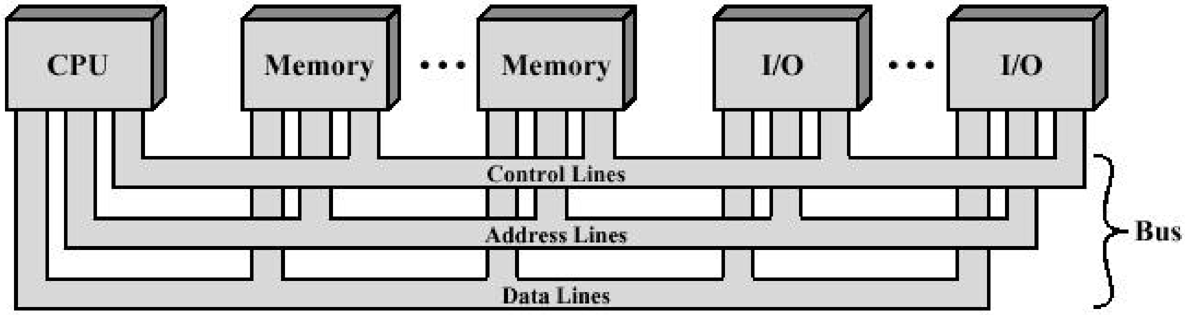




- It must support the following types of transfers:
    - Memory to CPU
    - CPU to Memory
    - I/O to CPU
    - CPU to I/O
    - I/O to or from Memory - using Direct Memory Access (DMA)

Bus Interconnection (3.4)

- A bus is a shared transmission medium
    - Must only be used by one device at a time
    - When used to connect major computer components (CPU, memory, I/O) is called a system bus
- Three functional groups of communication lines



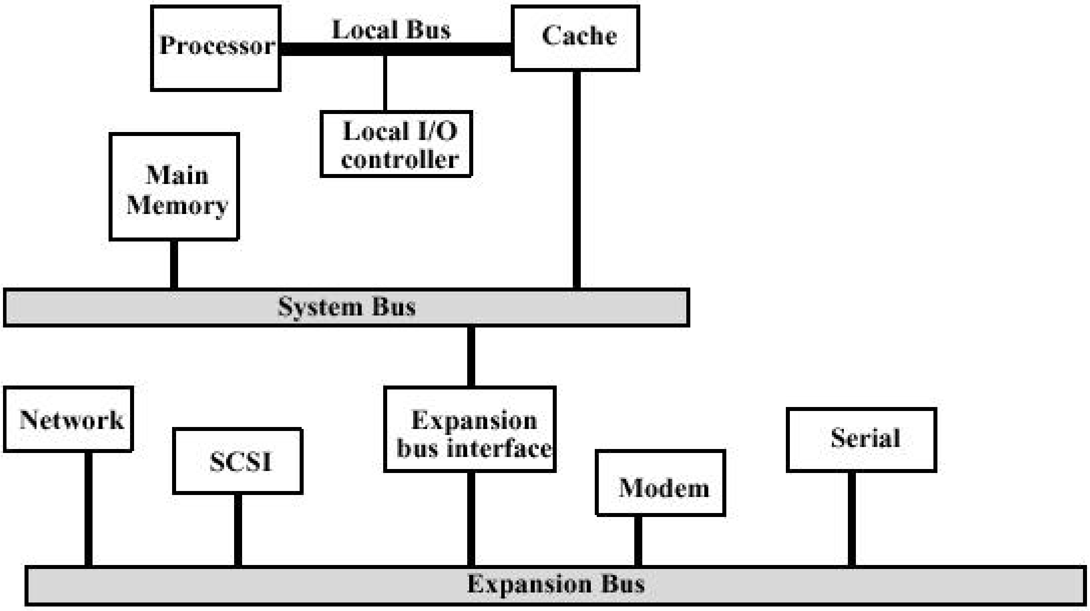
- Data lines (data bus) - move data between system modules
  - Width is a key factor in determining overall system performance
- Address lines - designate source or destination of data on the data bus
  - Width determines the maximum possible memory capacity of the system (may be a multiple of width)
  - Also used to address I/O ports. Typically:
    - high-order bits select a particular module
    - lower-order bits select a memory location or I/O port within the module
- Control lines - control access to and use of the data and address lines. Typical control lines include:
  - Memory Read and Memory Write
  - I/O Read and I/O Write
  - Transfer ACK
  - Bus Request and Bus Grant
  - Interrupt Request and Interrupt ACK
  - Clock
  - Reset

- If one module wishes to send data to another, it must:
  - Obtain use of the bus
  - Transfer data via the bus
- If one module wishes to request data from another, it must:
  - Obtain use of the bus
  - Transfer a request to the other module over control and address lines
  - Wait for second module to send data
- Typical physical arrangement of a system bus
  - A number of parallel electrical conductors
  - Each system component (usually on one or more boards) taps into some or all of the bus lines (usually with a slotted connector)
  - System can be expanded by adding more boards
  - A bad component can be replaced by replacing the board where it resides

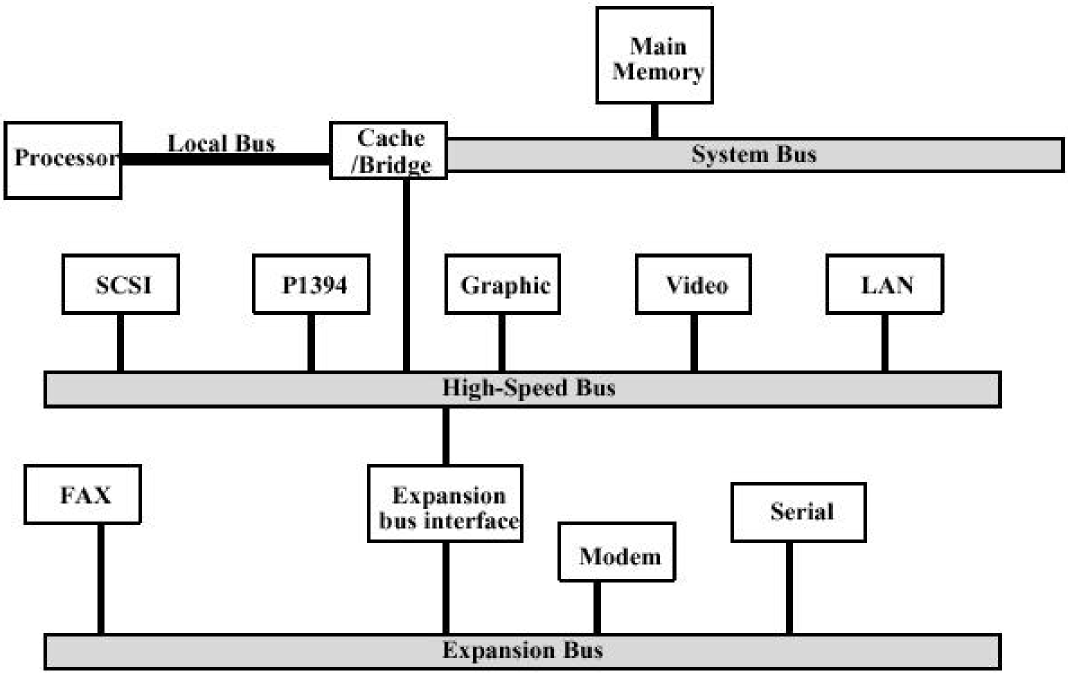
Multiple Bus Hierarchies

- A great number of devices on a bus will cause performance to suffer
  - Propagation delay - the time it takes for devices to coordinate the use of the bus
  - The bus may become a bottleneck as the aggregate data transfer demand approaches the capacity of the bus (in available transfer cycles/second)
- Traditional Hierarchical Bus Architecture
  - Use of a cache structure insulates CPU from frequent accesses to main memory
  - Main memory can be moved off local bus to a system bus
  - Expansion bus interface
    - buffers data transfers between system bus and I/O controllers on expansion bus
    - insulates memory-to-processor traffic from I/O traffic

- Traditional Hierarchical Bus Architecture Example

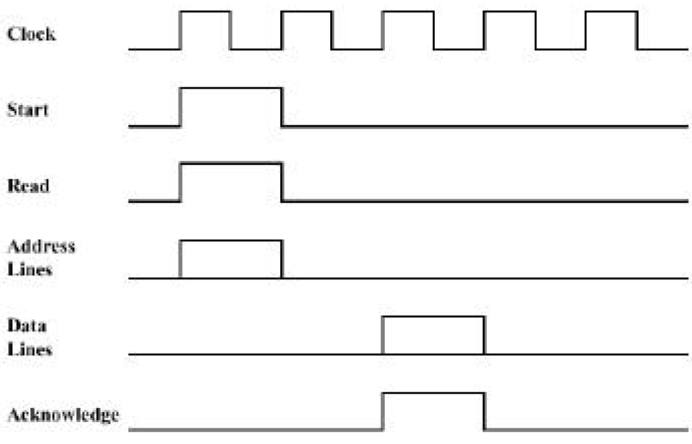


- High-performance Hierarchical Bus Architecture
  - Traditional hierarchical bus breaks down as higher and higher performance is seen in the I/O devices
  - Incorporates a high-speed bus
    - specifically designed to support high-capacity I/O devices
    - brings high-demand devices into closer integration with the processor and at the same time is independent of the processor
    - Changes in processor architecture do not affect the high-speed bus, and vice-versa
  - Sometimes known as a mezzanine architecture
- High-performance Hierarchical Bus Architecture Example



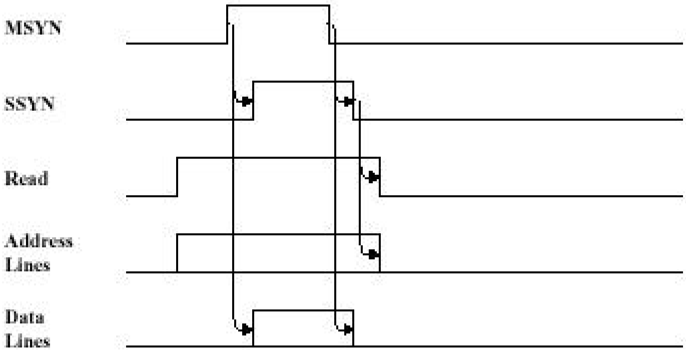
Elements of Bus Design

- Bus Types
    - Dedicated - a line is permanently assigned either to one function or to a physical subset of computer components
    - Multiplexed
        - Time multiplexing - using the same lines for multiple purposes (different purposes at different times)
            - Uses fewer lines, saving space and cost
            - BUT more complex circuitry required in each module
            - BUT potential reduction in performance
- Physical dedication - the use of multiple buses, each of which connects to only a subset of modules, with an adapter module to connect buses and resolve contention at the higher level
- Method of Arbitration - determining who can use the bus at a particular time
    - Centralized - a single hardware device called the bus controller or arbiter allocates time on the bus
    - Distributed - each module contains access control logic and the modules act together to share the bus
    - Both methods designate one device (either CPU or an I/O module) as master, which may initiate a data transfer with some other device, which acts as a slave.
- Timing
    - Synchronous Timing
        - Bus includes a clock line upon which a clock transmits a regular sequence of alternating 1's and 0's of equal duration
        - A single 1-0 transmission is referred to as a clock cycle or bus cycle
        - All other devices on the bus can read the clock line, and all events start at the beginning of a clock cycle



    - Asynchronous Timing
        - The occurrence of one event on a bus follows and depends on the occurrence of a previous event
        - Allows system to take advantage of advances in device performance by having a mixture of slow and fast devices, using older and newer technology, sharing the same bus

  - BUT harder to implement and test than synchronous timing

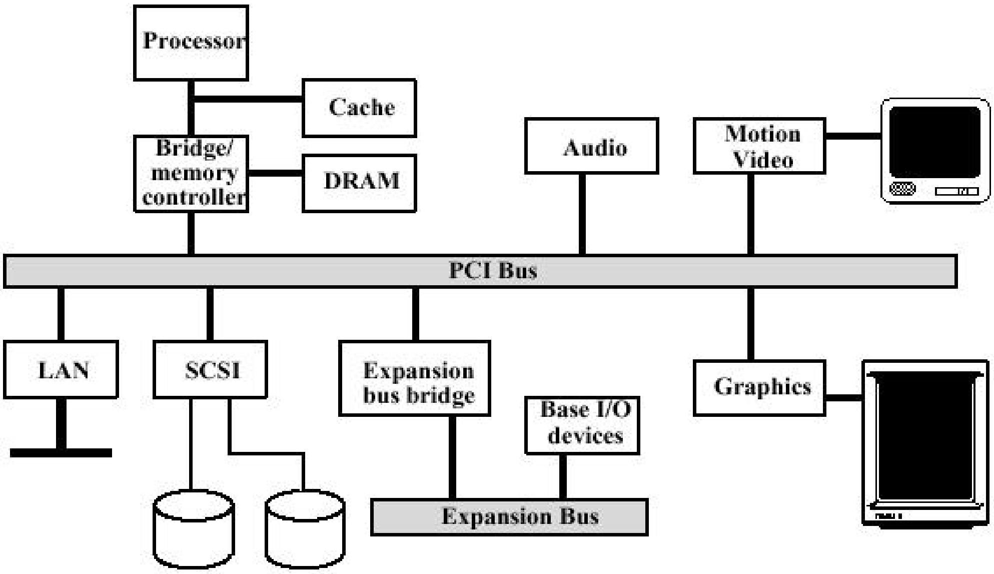


- Bus Width
  - Data bus: wider = better performance
  - Address bus: wider = more locations can be referenced
- Data Transfer Type
  - All buses must support write (master to slave) and read (slave to master) transfers
- Combination operations
  - Read-modify-write
    - a read followed immediately by a write to the same address.
    - Address is only broadcast once, at the beginning of the operation
    - Indivisible, to prevent access to the data element by other potential bus masters
    - Principle purpose is to protect shared memory in a multiprogramming system
  - Read-after-write - indivisible operation consisting of a write followed immediately by a read from the same address (for error checking purposes)
- Block data transfer
  - one address cycle followed by n data cycles
  - first data item to or from specified address
  - remaining data items to or from subsequent addresses

PCI (3.5)

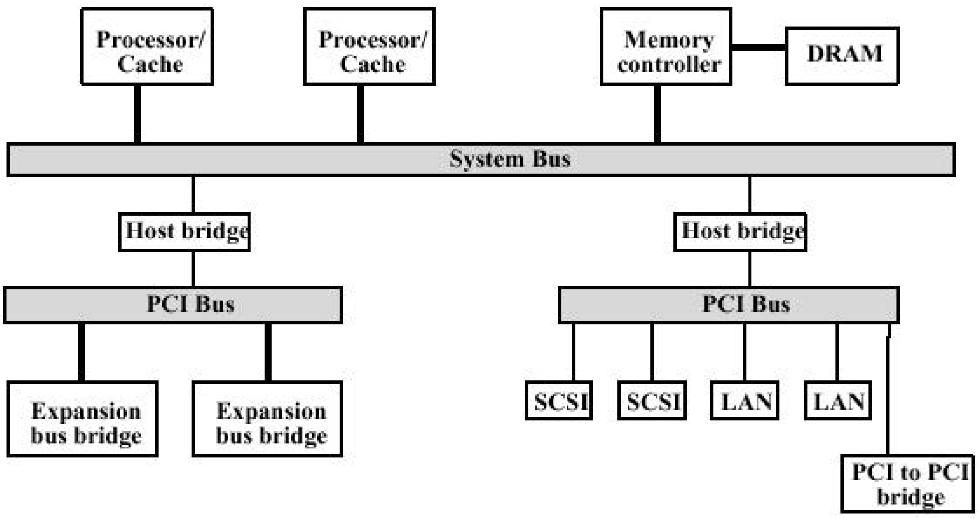
- PCI = Peripheral Component Interconnect
  - High-bandwidth
  - Processor independent
  - Can function as a mezzanine or peripheral bus
- Current Standard
  - up to 64 data lines at 33Mhz
  - requires few chips to implement
  - supports other buses attached to PCI bus
  - public domain, initially developed by Intel to support Pentium-based systems
  - supports a variety of microprocessor-based configurations, including multiple-processors
  - uses synchronous timing and centralized arbitration

- Typical Desktop System



Note: Bridge acts as a data buffer so that the speed of the PCI bus may differ from that of the processor's I/O capability.

- Typical Server System



Note: In a multiprocessor system, one or more PCI configurations may be connected by bridges to the processor's system bus.

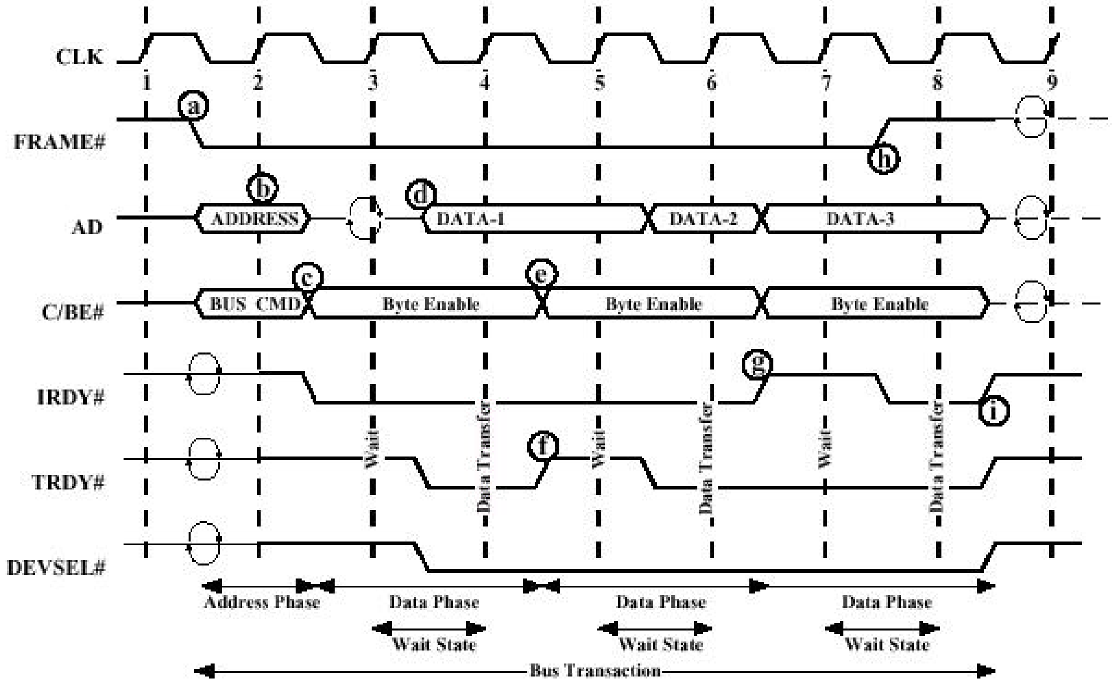
- Bus Structure
  - o 50 mandatory signal lines, divided into the following groups:
    - System Pins - includes clock and reset
    - Address and Data Pins - 32 time-multiplexed lines for addresses and data, plus lines to interpret and validate these
    - Interface Control Pins - control timing of transactions and provide coordination among initiators and targets
    - Arbitration Pins - not shared, each PCI master has its own pair to connect to PCI bus arbiter
    - Error Reporting Pins - for parity and other errors

- o 50 optional signal lines, divided into the following groups:
  - Interrupt Pins - not shared, each PCI device has its own interrupt line or lines to an interrupt controller
  - Cache Support Pins
  - 64-bit Bus Extension Pins - 32 additional time-multiplexed lines for addresses and data, plus lines to interpret and validate these, and to provide agreement between two PCI devices on use of these
  - ITAG/Boundary Scan Pins - support testing procedures from IEEE Standard 149.1

- PCI Commands
  - o issued by the initiator (the master) to the target (the slave)
  - o Use the C/BE lines
  - o Types

    - Interrupt Ack - Memory Read Multiple
    - Special Cycle - Memory Write
    - I/O Read - Memory Write & Invalidate
    - I/O Write - Configuration Read
    - Memory Read - Configuration Write
    - Memory Read Line - Dual Address Cycle

- Data Transfer Example



a. Once a bus master has gained control of the bus, it may begin the transaction by asserting FRAME. This line remains asserted until the initiator is ready to complete the last data phase. The initiator also puts the start address on the address bus, and the read command on the C/BE lines.

b. At the start of clock 2, the target device will recognize its address on the AD lines.

c. The initiator ceases driving the AD bus. A turnaround cycle (indicated by the two circular arrows) is required on all signal lines that may be driven by more than one device, so that the dropping of the address signal will prepare the bus for use by the target device. The initiator changes the information on the C/BE lines to designate which AD lines are to be used for transfer for the currently addressed data (from 1 to 4 bytes). The initiator also asserts IRDY to indicated that it is ready for the first data item.

d. The selected target asserts DEVSEL to indicate that it has recognized its address and will respond. It places the requested data on the AD lines and asserts TRDY to indicate that valid data is present on the bus.

e. The initiator reads the data at the beginning of clock 4 and changes the byte enable lines as needed in preparation for the next read.

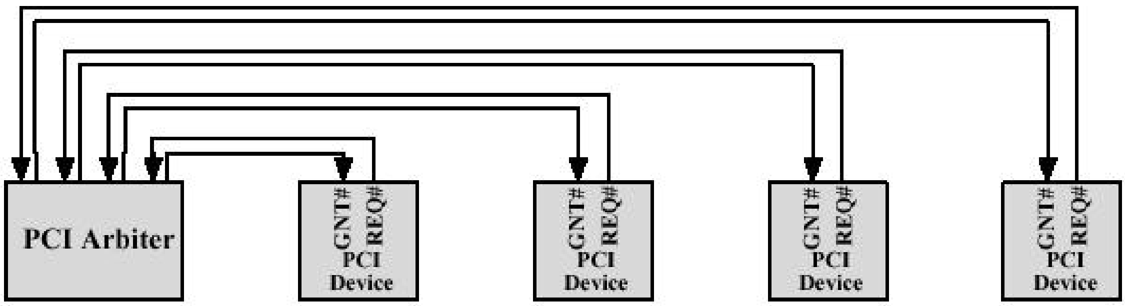
f. In this example, the target needs some time to prepare the second block of data for transmission. Therefore, it deasserts TRDY to signal the initiator that there will not be new data during the coming cycle. Accordingly, the initiator does not read the data lines at the beginning of the 5th clock cycle and does not change byte enable during that cycle. The block of data is read at beginning of clock 6.

g. During clock 6, the target places the 3rd data item on the bus. However, in this example, the initiator is not yet ready to read the data item (e.g., it has a temporary buffer full condition). It therefore deasserts IRDY. This will cause the target to maintain the third data item on the bus for an extra clock cycle.

h. The initiator knows that the 3rd data transfer is the last, and so it deasserts FRAME to signal the target that this is the last data transfer. It also asserts IRDY to signal that it is ready to complete that transfer.

i. The initiator deasserts IRDY, returning the bus to the idle state, and the target deasserts TRDY and DEVSEL.

- Arbitration



  - Centralized
  - Synchronous
  - Each master has a unique request (REQ) and grant (GNT) signal
  - Each master's REQ and GNT is attached to a central arbiter
  - Arbitration algorithm can be any desired, programmed into the arbiter
  - Uses hidden arbitration, meaning that arbitration can take place while other bus transactions are occurring on other bus lines

PCI Enhancements: AGP

- AGP – Advanced Graphics Port

  - Called a port, not a bus because it only connects 2 devices