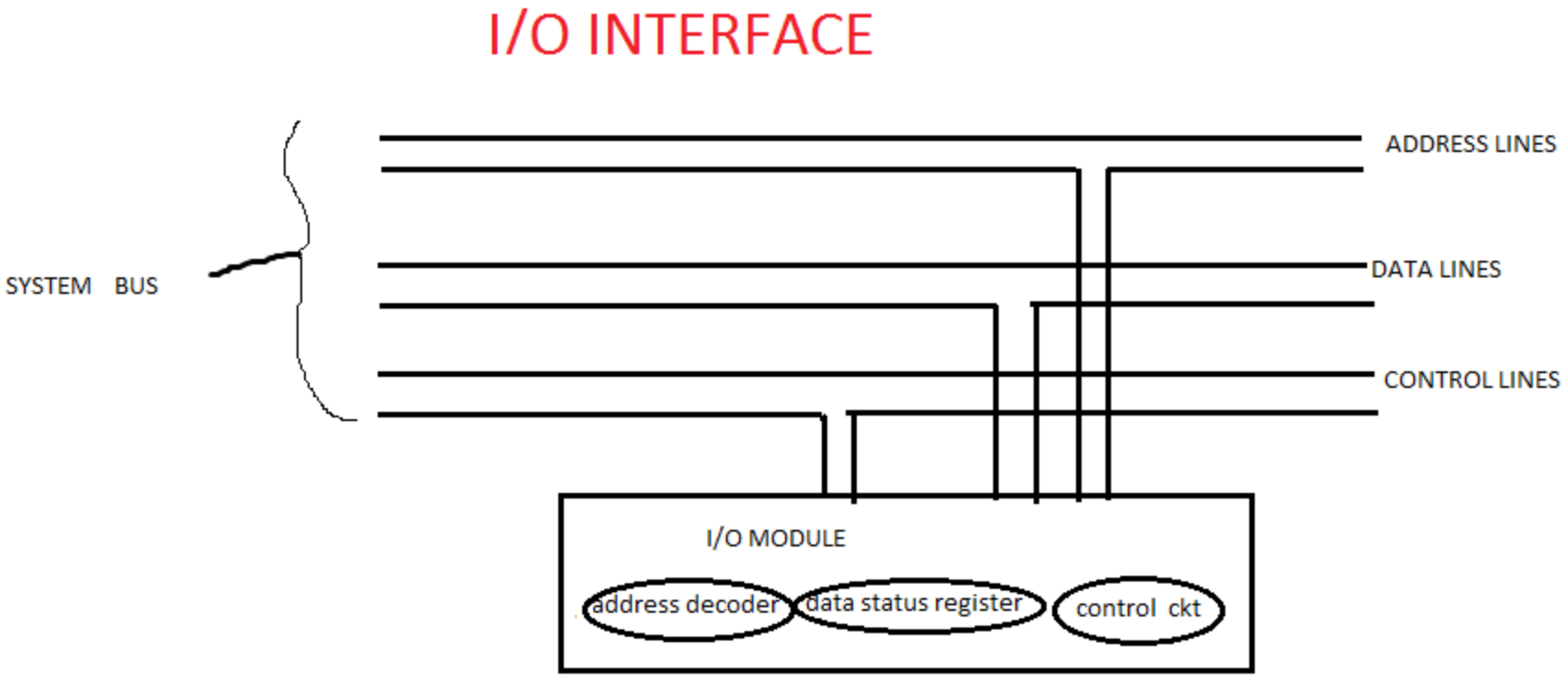
MODULE 3

Input Output Organization

Accessing I/O Devices:-

- Most modern computers use single bus arrangement for connecting I/O devices to CPU & Memory
- The bus enables all the devices connected to it to exchange information
- Bus consists of 3 set of lines : Address, Data & Control Lines
- Processor places a particular address (unique for an I/O Dev.) on address lines
- Device which recognizes this address responds to the commands issued on the Control lines
- Processor requests for either Read / Write
- The data will be placed on Data lines

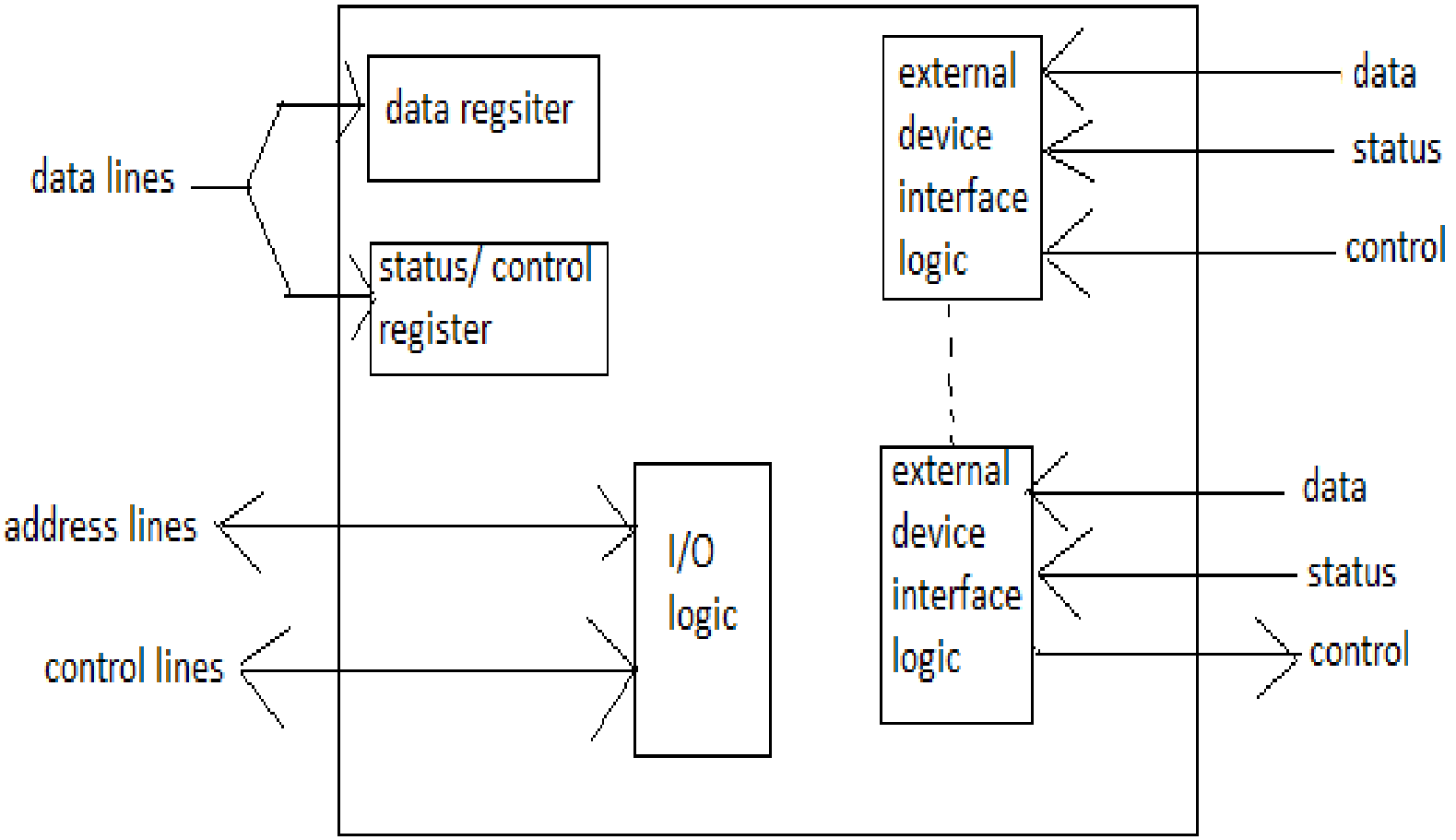
I/O INETRFACE:-



I/O INETRFACE:-

- It is a special hardware used to connect the I/O device to the bus , i.e, I/O interface.
- The I/O module has 3 section:-
- (a)**Address decoder:-** It enables the device to recognize its address when this address appears on the address lines.
- (b)**Data register:-** The data register holds the data being transferred to or from the processor.
- The status register contains information relevant to the operation of the I/O module.
- * Both the data and status register are connected to the data

I/O MODULE BLOCK DIAGRAM



MODES OF DATA TRANSFER:-

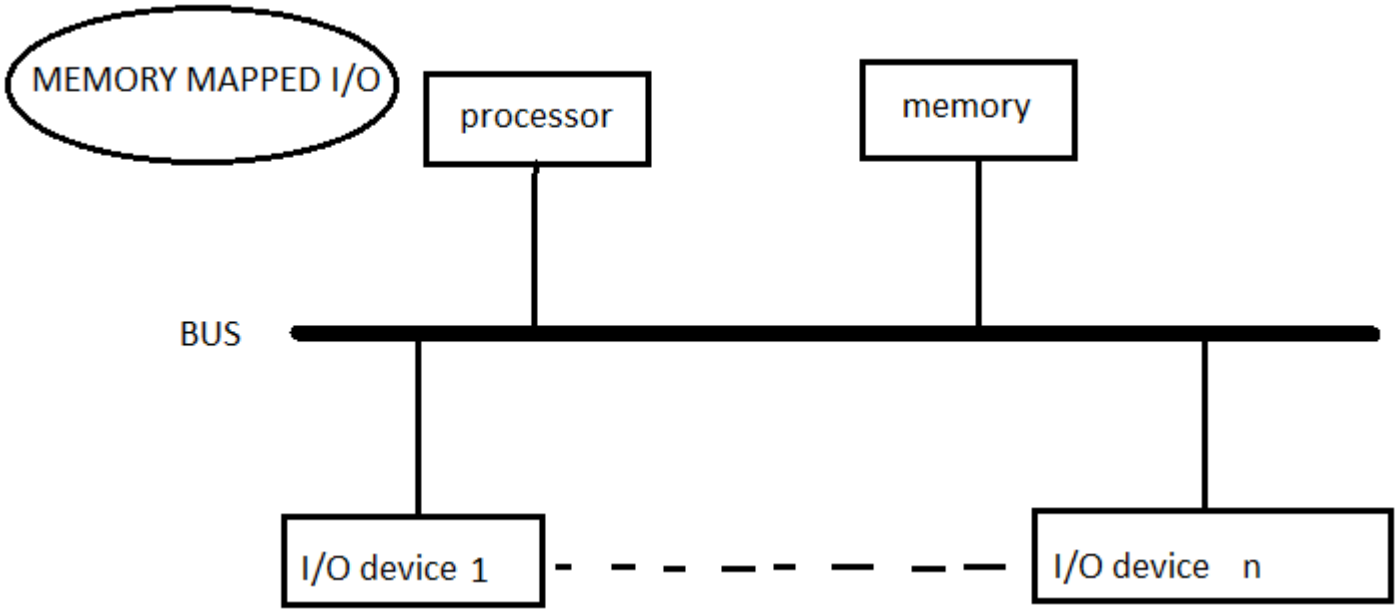
1.Programmed I/O Mode:-

- In programmed I/O mode data are exchanged between the processor and the I/O module. When a processor is executing a program and encounters an instruction relating to I/O, it executes that instruction by issuing a command to that appropriate I/O module. With programmed I/O the I/O module will perform the requested action and then set the appropriate bit in the I/O status register .
- The I/O module takes no further action to alert the processor (it doesn't interrupt the processor).
- The I/O commands issued by the processor to the I/O module

✓ Test
✓ Control
✓ Read
✓ Write

2.Memory Mapped I/O:-

- There is a single address space for memory location and I/O devices.(the address space is shared)
- With memory mapped I/O a single read line a single write line are needed on the bus. The bus may be equipped with memory read and write plus Input and output command lines.
- Now the command lines specifies whether the address refers to memory location or an I/O device.
- Most CPU uses memory mapped I/O.
- Always CPU assigns address to memory some of memory space is stolen and assigned to I/O device.
- It deals with fewer address lines.



3.Interrupt Driven I/O:-

- In this method the program issues an I/O command and then continues to execute untill it is interrupted by the I/O hardware to signal the end of I/O operation.
- Here the program enters a wait loop in which it repeatedly checks the device status. During this process the processor is not performing any useful computation.
- There are many situations where tasks can be performed while waiting for an I/O device to be ready, to allow this the I/O device should alert the processor when it becomes ready. It can be done by sending a hardware signal called an **interrupt.**
- The routine executed in response to an interrupt request is called **Interrupt Service Routine**(ISR).
- The processor first completes execution of instruction then it loads the program counter(pc) with the address of 1st instruction of ISR.

4.Direct Memory Access:-

- In this method the input and output devices read/write information from the main memory without interference of the CPU through the system bus.
- For I/O transfer, processor determines the status of I/O by:

✓Polling

✓Waiting for interrupt signal.

- Considerable Overhead is incurred in above I/O transfer processing.
- By DMA approach , large blocks of data at high speed can be sent between external device and main memory.

DMA CONTROLLER:-

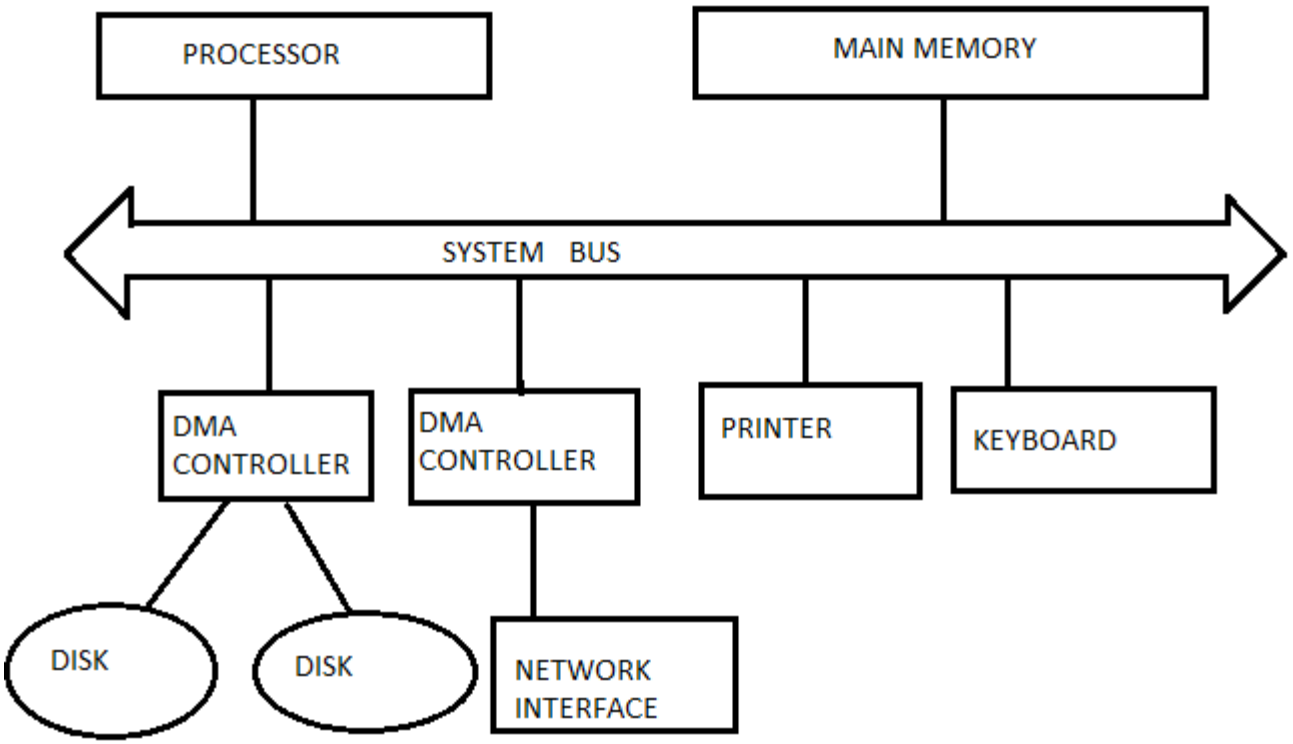
- It allows the data transfer between I/O device and memory.
- DMA controllers acts a processor but it is controlled by the CPU. To initiate the transfer of a block of words, the processor sends the following data to controller
- Starting address of memory block

✓The word count

✓Control to specify the mode of transfer such as read or write.

✓A control to start the DMA transfer.

- The DMA controller performs the required I/O operation and send a interrupt to the processor upon completition.



Use Of DMA controller in Computer System

- DMA device have higher priority than processor over BUS control.
- Types of DMA Transfer:-

✓Cycle stealing

✓ Brust mode

- **Cycle Stealing:-**
- DMA controller ‘steals’ memory cycles from the processor though processor originates most memory access.
- **Burst mode:-**
- The DMA controller may be given exclusive access to the main memory to tranasfer a block of data without interruption.
- **Conflicts Of DMA**:-
- ->Conflict between processor and DMA
- ->Two DMA Controller try to access the BUS at same time to access the main memory

Interrupts

- Interrupts are external events that require the processor's attention.
  - Peripherals and other I/O devices may need attention.
  - Timer interrupts to mark the passage of time.
- These situations are not errors.
  - They happen normally.
  - All interrupts are recoverable:
    - The interrupted program will need to be resumed after the interrupt is handled.
- It is the operating system's responsibility to do the right thing, such as:
  - Save the current state.
  - Find and load the correct data from the hard disk
  - Transfer data to/from the I/O device.

- Interrupts are provided primarily as a way to improve processing efficiency
- the most common classes of interrupts

Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
I/O	Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.
Hardware failure	Generated by a failure such as power failure or memory parity error.

Program flow of control without and with interrupts

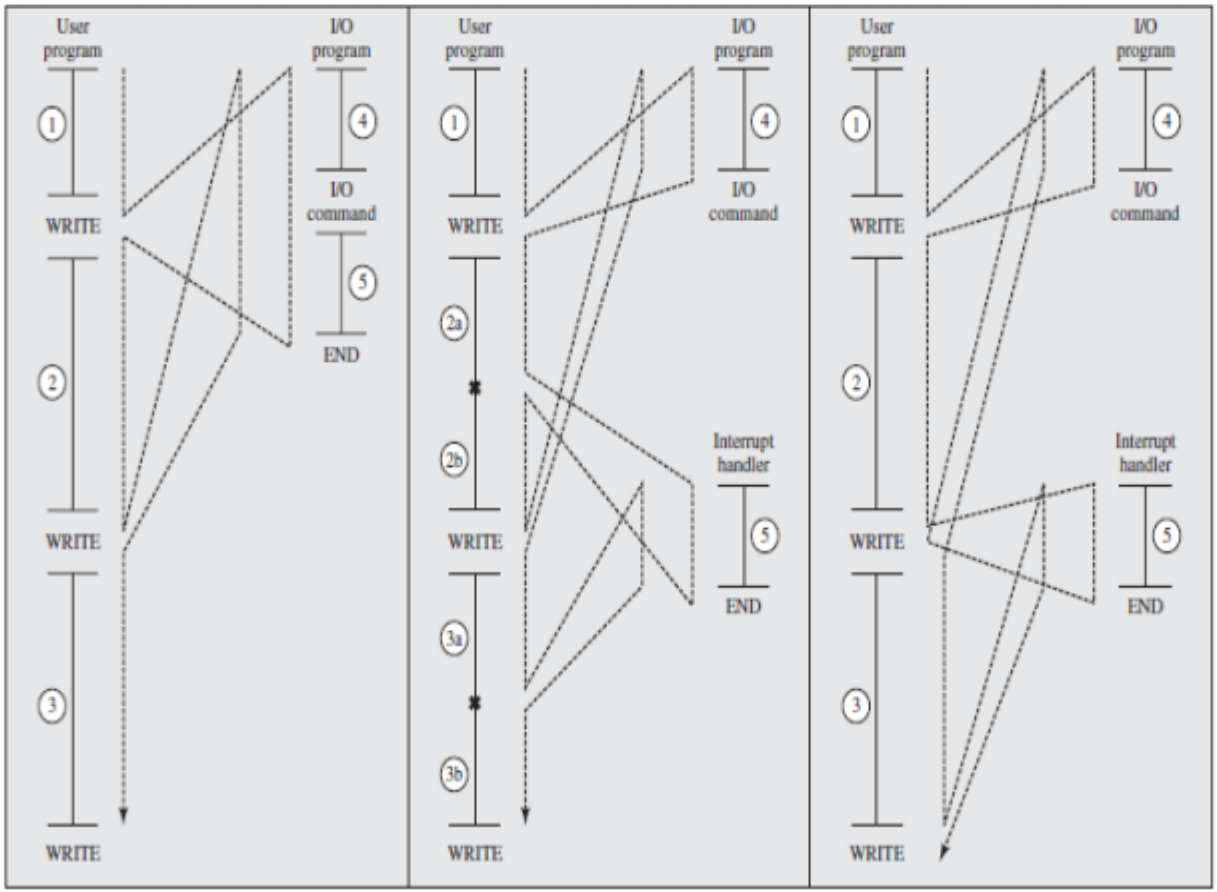


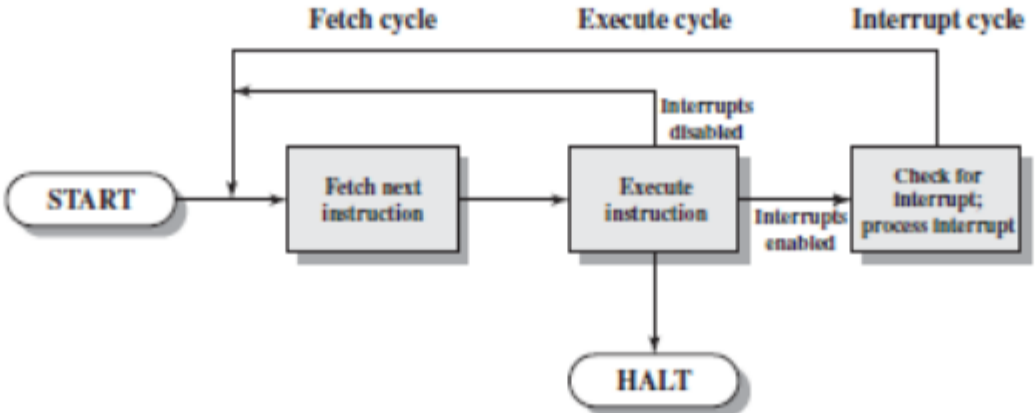
Figure 1.9 Program Flow of Control Without and With Interrupts

INTERRUPTS AND THE INSTRUCTION CYCLE

- With interrupts, the processor can be engaged in executing other instructions while an I/O operation is in progress.
- When the external device becomes ready to accept more data from the processor,—the I/O module for that external device sends an interrupt request signal to the processor. The processor responds by suspending operation of the current program, branching off to a program to service that particular I/O device, known as an interrupt handler, and resuming the original execution after the device is serviced.

- From the point of view of the user program, an interrupt is just that: an interruption of the normal sequence of execution. When the interrupt processing is completed, execution resumes

Instruction cycle with interrupts



- In the interrupt cycle, the processor checks to see if any interrupts have occurred. If no interrupts are pending, the processor proceeds to the fetch cycle and fetches the next instruction of the current program. If an interrupt is pending, the processor does the following:
- It suspends execution of the current program being executed and saves its context
- It sets the program counter to the starting address of an interrupt handler routine. The processor now proceeds to the fetch cycle and fetches the first instruction in the interrupt handler program, which will service the interrupt. When the interrupt handler routine is completed, the processor can resume execution of the user program at the point of interruption

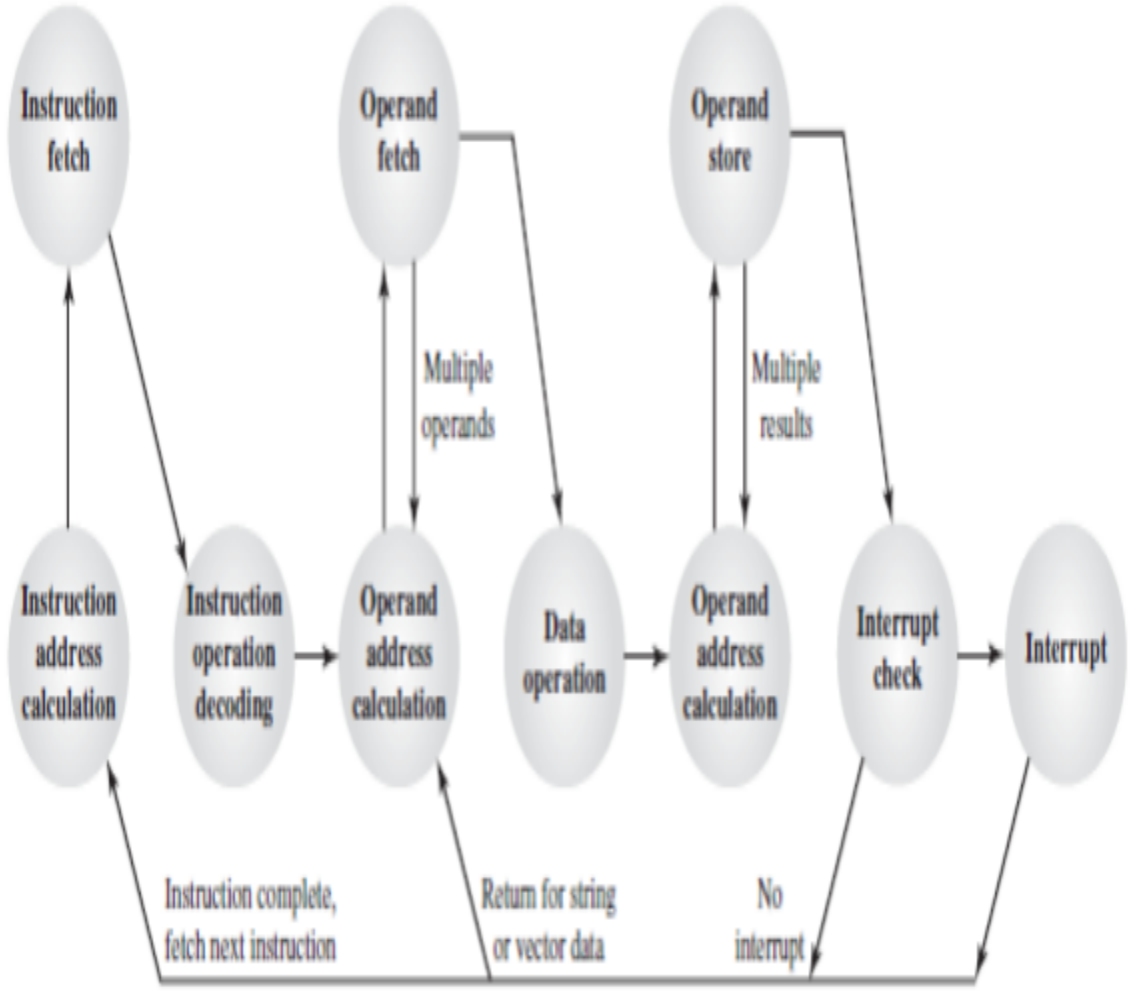
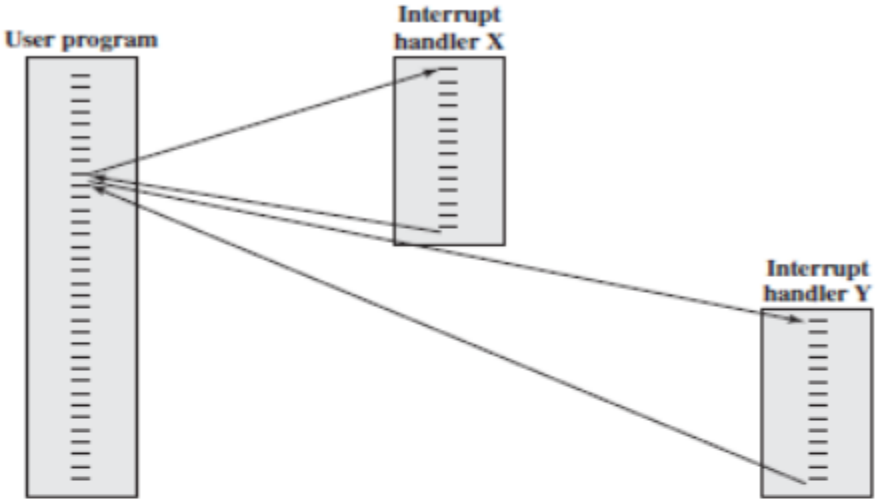


Figure 1.14 Instruction Cycle State Diagram With Interrupts

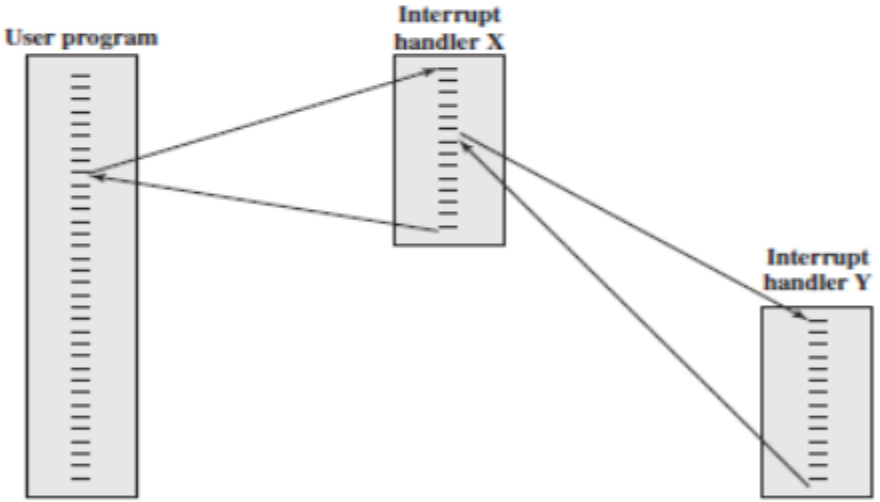
MULTIPLE INTERRUPTS

- Multiple interrupts can occur. Two approaches can be taken to dealing with multiple interrupts.
- The first is to disable interrupts while an interrupt is being processed. A disabled interrupt simply means that the processor can and will ignore that interrupt request signal. Thus, when a user program is executing and an interrupt occurs, interrupts are disabled immediately. After the interrupt handler routine completes, interrupts are enabled before resuming the user program and the processor checks to see if additional interrupts have occurred. This approach is nice and simple, as interrupts are handled in strict sequential order

- The drawback to the preceding approach is that it does not take into account relative priority or time-critical needs A second approach is to define priorities for interrupts and to allow an interrupt of higher priority to cause a lower-priority interrupt handler to be itself interrupted



(a) Sequential interrupt processing



Nested interrupt processing

Buses

- A bus is a communication pathway connecting two or more devices. A key characteristic of a bus is that it is a shared transmission medium. Multiple devices connect to the bus, and a signal transmitted by any one device is available for reception by all other devices attached to the bus. If two devices transmit during the same time period, their signals will overlap and become garbled. Thus, only one device at a time can successfully transmit.

- Typically, a bus consists of multiple communication pathways, or lines. Each line is capable of transmitting signals representing binary 1 and binary 0. An 8-bit unit of data can be transmitted over eight bus lines. A bus that connects major computer components (processor, memory, I/O) is called a system bus.

Bus structure

Bus Structure

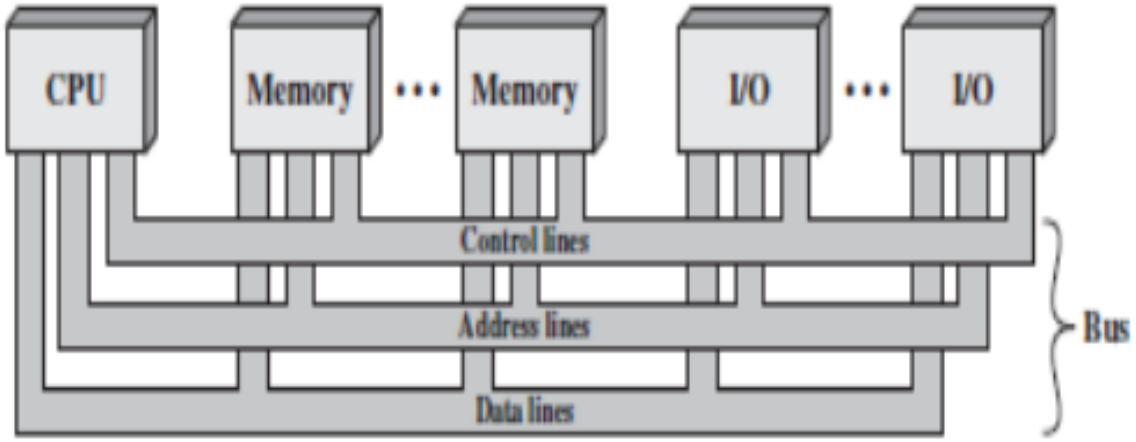


Figure 1.17Bus Interconnection Schemes

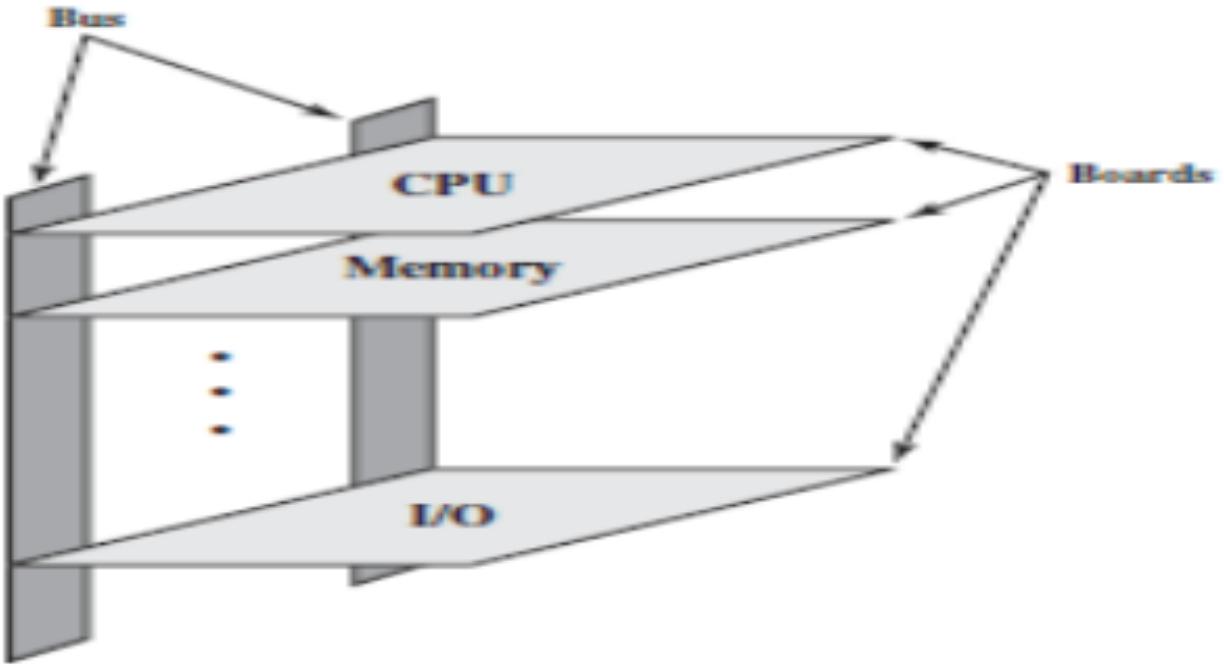
- On any bus the lines can be classified into three functional groups
- data, address, and control lines. In addition, there may be power distribution lines that supply power to the attached modules
- The data lines provide a path for moving data among system modules. These lines, collectively, are called the data bus. The address lines are used to designate the source or destination of the data on the data bus. For example, on an 8-bit address bus, address 01111111 and below might reference locations in a memory module (module 0) with 128 words of memory, and address 10000000 and above refer to devices attached to an I/O module (module 1). The control lines are used to control the access to and the use of the data and address lines. Control signals transmit both command and timing information among system modules.

Timing signals indicate the validity of data and address information. Command signals specify operations to be performed. Typical control lines include

- Memory write: Causes data on the bus to be written into the addressed location
- Memory read: Causes data from the addressed location to be placed on the bus
- I/O write: Causes data on the bus to be output to the addressed I/O port
- I/O read: Causes data from the addressed I/O port to be placed on the bus
- Transfer ACK: Indicates that data have been accepted from or placed on the bus
- Bus request: Indicates that a module needs to gain control of the bus
- Bus grant: Indicates that a requesting module has been granted control of the bus
- Interrupt request: Indicates that an interrupt is pending
- Interrupt ACK: Acknowledges that the pending interrupt has been recognized
- Clock: Is used to synchronize operations
- Reset: Initializes all modules

- The operation of the bus is as follows. If one module wishes to send data to another, it must do two things:
- (1) obtain the use of the bus, and
- (2) transfer data via the bus. If one module wishes to request data from another module, it must
- (1) obtain the use of the bus, and
- (2) transfer a request to the other module over the appropriate control and address lines. It must then wait for that second module to send the data.

The classic physical arrangement of a bus

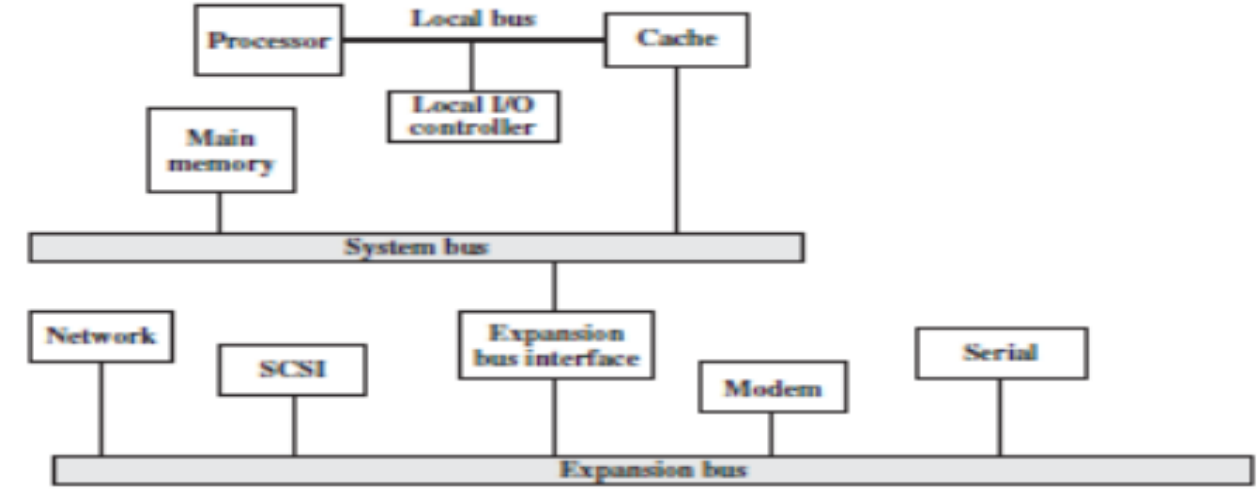


- In this example, the bus consists of two vertical columns of conductors. Each of the major system components occupies one or more boards and plugs into the bus at these slots. Thus, an on-chip bus may connect the processor and cache memory, whereas an on-board bus may connect the processor to main memory and other components. This arrangement is most convenient. A small computer system may be acquired and then expanded later (more memory, more I/O) by adding more boards. If a component on a board fails, that board can easily be removed and replaced.

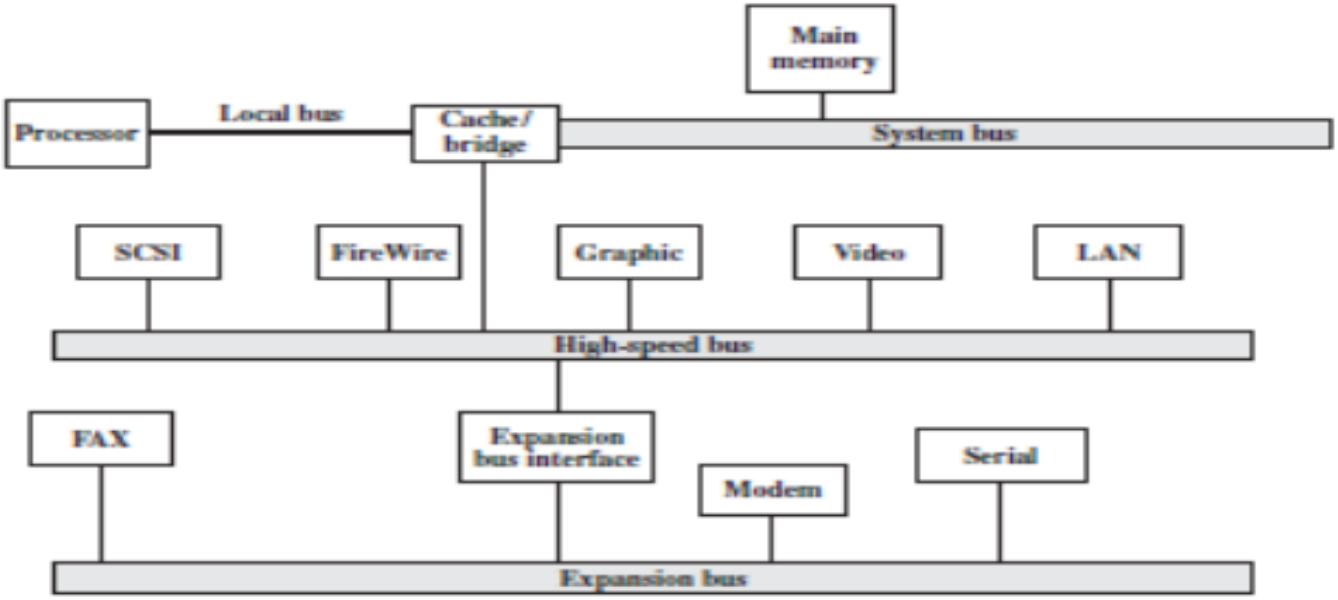
Multiple-Bus Hierarchies

- In general, the more devices attached to the bus, the greater the bus length and hence the greater the propagation delay.
- . The bus may become a bottleneck as the aggregate data transfer demand approaches the capacity of the bus.
- Most computer systems use multiple buses, generally laid out in a hierarch

- There is a local bus that connects the processor to a cache memory and that may support one or more local devices The cache memory is connected to a system bus to which all of the main memory modules are attached. It is possible to connect I/O controllers directly onto the system bus. A more efficient solution is to make use of one or more expansion buses for this purpose. This arrangement allows the system to support a wide variety of I/O devices and at the same time insulate memory-to-processor traffic from I/O traffic.



(a) Traditional bus architecture



(b) High-performance architecture

Figure 1.19 Example Bus Confihuration

of Bus Design

Elements of Bus Design

Type	**Bus Width**
Dedicated	Address
Multiplexed	Data
Method of Arbitration	**Data Transfer Type**
Centralized	Read
Distributed	Write
Timing	Read-modify-write
Synchronous	Read-after-write
Asynchronous	Block

Elements of Bus Design

- There are a few design elements that serve to classify and differentiate buses.
- BUS TYPES Bus lines can be separated into two generic types: dedicated and multiplexed.

✓A dedicated bus line is permanently assigned either to one function or to a physical subset of computer components. Physical dedication refers to the use of multiple buses, each of which connects only a subset of modules.

✓The potential advantage of physical dedication is high throughput, because there is less bus contention.A disadvantage is the increased size and cost of the system. Address and data information may be transmitted over the same set of lines using an Address Valid control line.

✓ At the beginning of a data transfer, the address is placed on the bus and the Address Valid line is activated. The address is then removed from the bus, and the same bus connections are used for the subsequent read or write data transfer. This method of using the same lines for multiple purposes is known as time multiplexing.

✓The advantage of time multiplexing is the use of fewer lines, which saves space and, usually, cost. The disadvantage is that more complex circuitry is needed within each module.

METHOD OF ABITRATION:

- The various methods can be roughly classified as being either centralized or distributed. In a centralized scheme, a single hardware device, referred to as a bus controller or arbiter, is responsible for allocating time on the bus.
- In a distributed scheme, there is no central controller. Rather, each module contains access control logic and the modules act together to share the bus. With both methods of arbitration, the purpose is to designate either the processor or an I/O module, as master.
- The master may then initiate a data transfer (e.g., read or write) with some other device, which acts as slave for this particular exchange.

TIMING Buses

- use either synchronous timing or asynchronous timing. With synchronous timing, the occurrence of events on the bus is determined by a clock. A single 1–0 transmission is referred to as a clock cycle or bus cycle and defines a time slot
- With asynchronous timing, the occurrence of one event on a bus follows and depends on the occurrence of a previous event.
- Synchronous timing is simpler to implement and test. However, it is less flexible than asynchronous timing. With asynchronous timing, a mixture of slow and fast devices, using older and newer technology, can share a bus

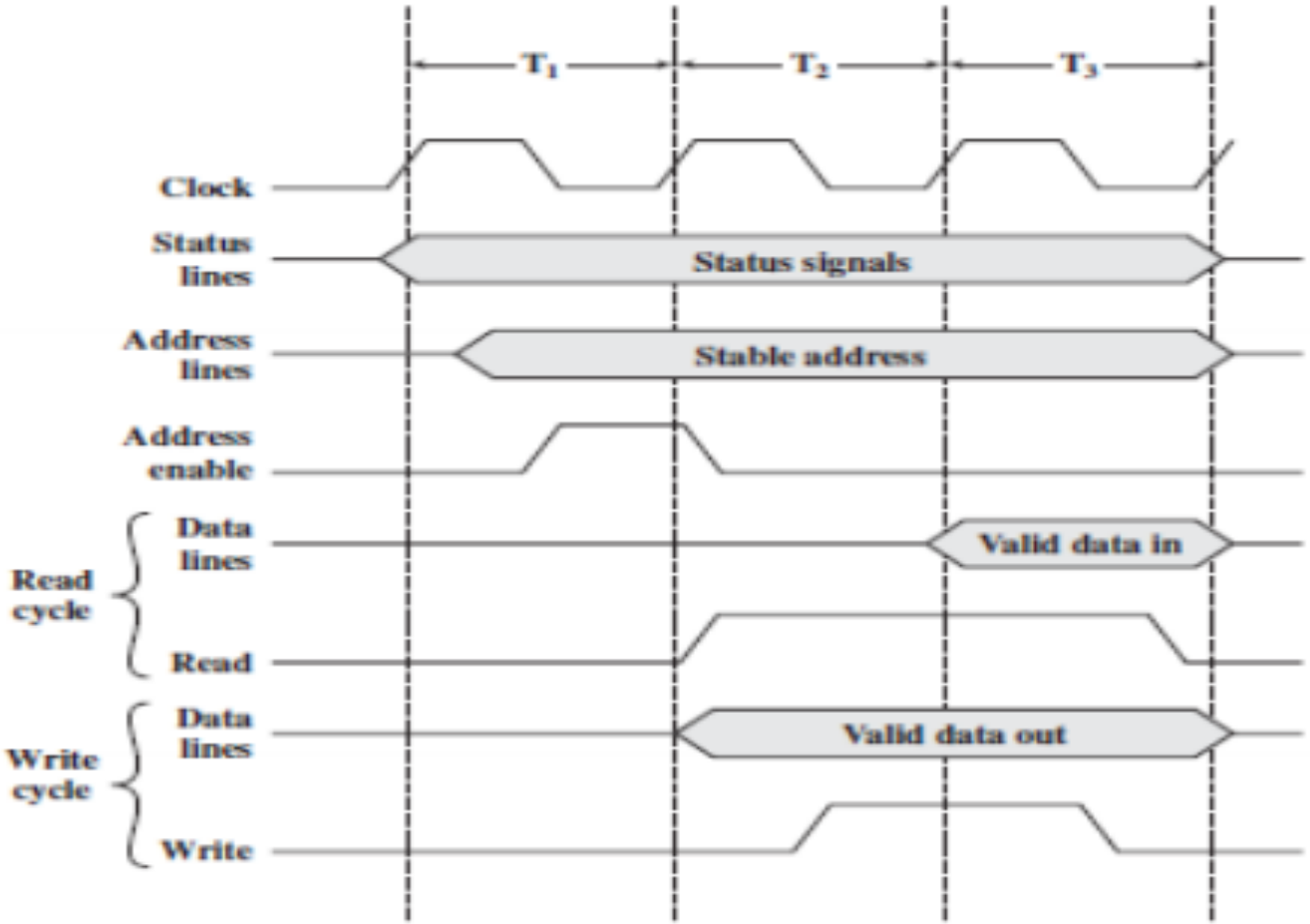
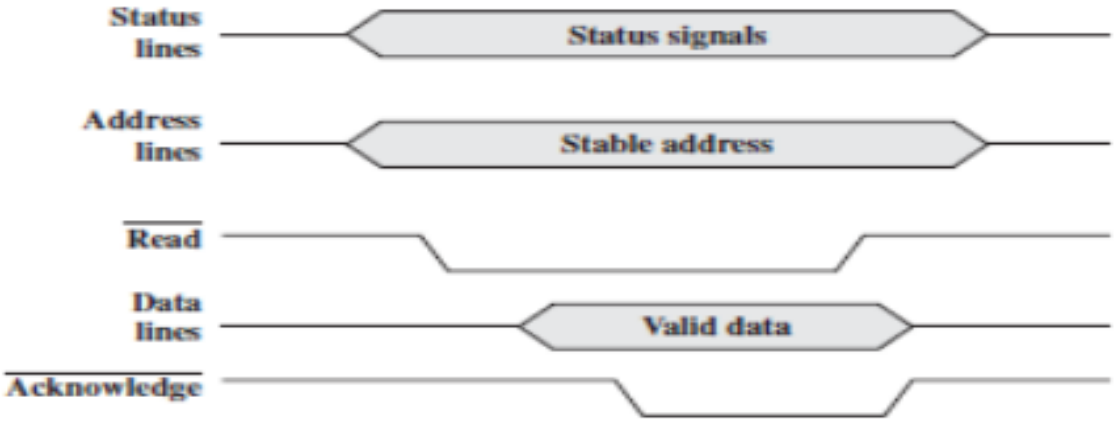
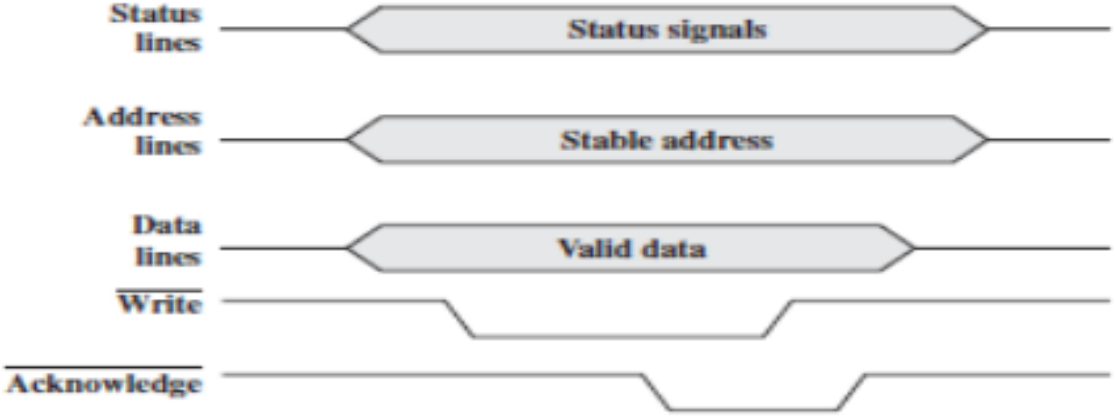


Figure 1.20 Timing of Synchronous Bus Operations



(a) System bus read cycle



(b) System bus write cycle

Figure 1.21 Timing of Asynchronous Bus Operations

BUS WIDTH

- The width of the data bus has an impact on system performance: The wider the data bus, the greater the number of bits transferred at one time. The width of the address bus has an impact on system capacity: the wider the address bus, the greater the range of locations that can be referenced.

DATA TRANSFER TYPE Finally, a bus supports various data transfer types, as illustrated in Figure

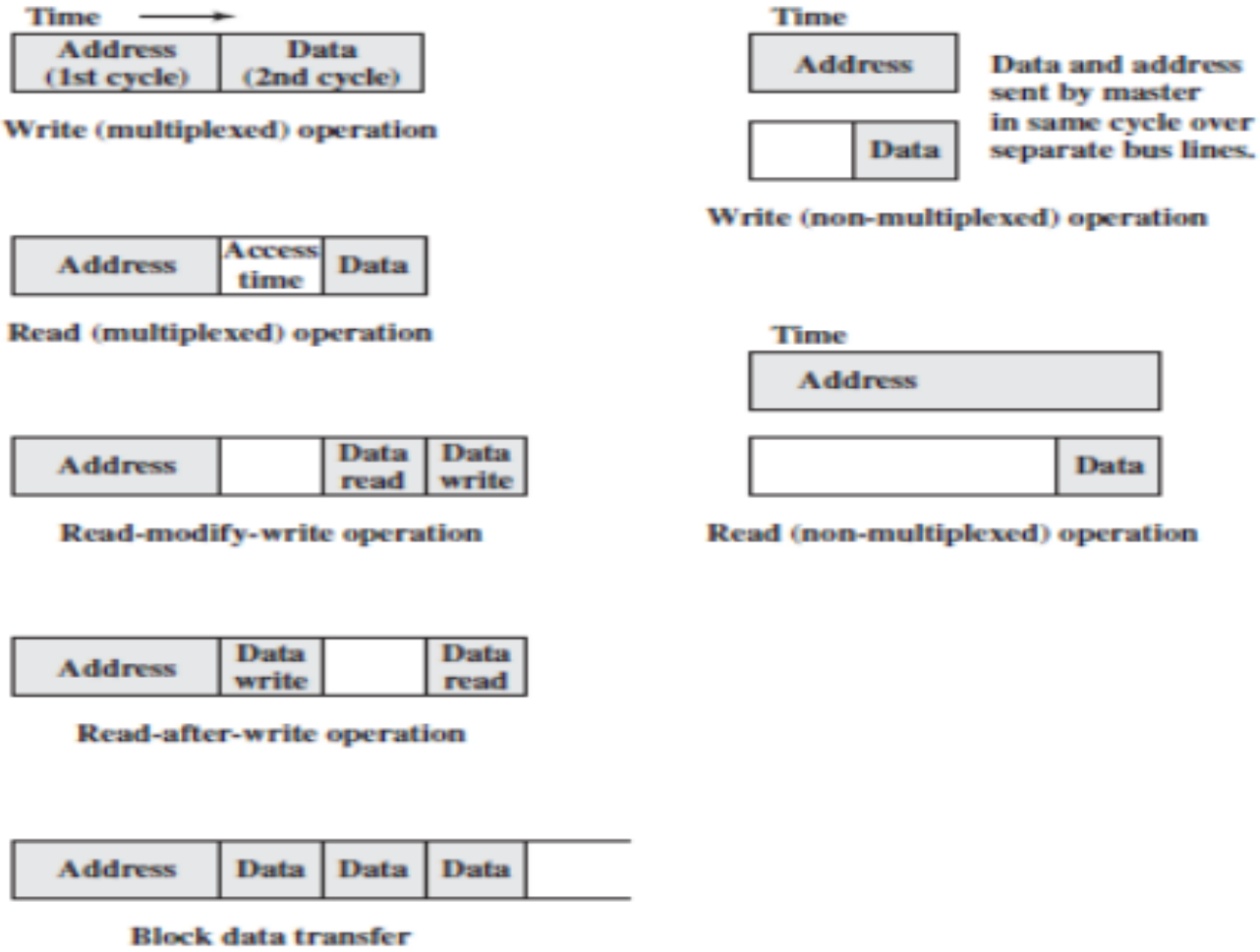


Figure 1.22 Bus Data Transfer Types

- In the case of a multiplexed address/data bus, the bus is first used for specifying the address and then for transferring the data. For a read operation, there is typically a wait while the data are being fetched from the slave to be put on the bus. For either a read or a write, there may also be a delay if it is necessary to go through arbitration to gain control of the bus for the remainder of the operation.
- In the case of dedicated address and data buses, the address is put on the address bus and remains there while the data are put on the data bus. For a write operation, the master puts the data onto the data bus as soon as the address has stabilized and the slave has had the opportunity to recognize its address. For a read operation, the slave puts the data onto the data bus as soon as it has recognized its address and has fetched the data.

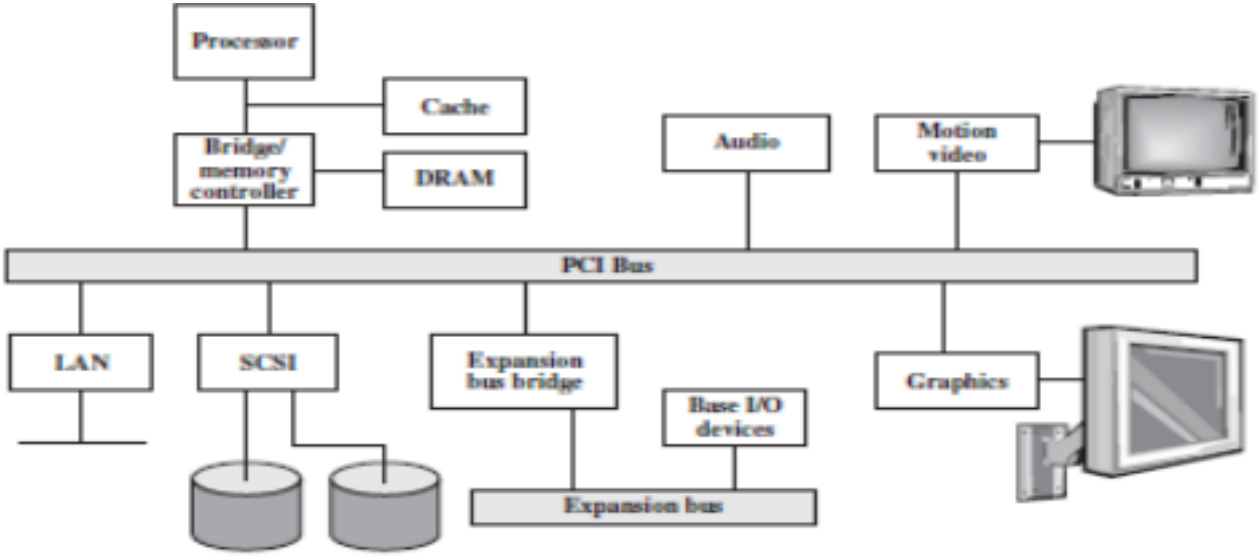
- A read–modify–write operation is simply a read followed immediately by a write to the same address
- Read-after-write is an indivisible operation consisting of a write followed immediately by a read from the same address.
- Some bus systems also support a block data transfer. The first data item is transferred to or from the specified address; the remaining data items are transferred to or from subsequent addresses

PCI (PHERIPHERAL COMPONENT INTERCONNECT)

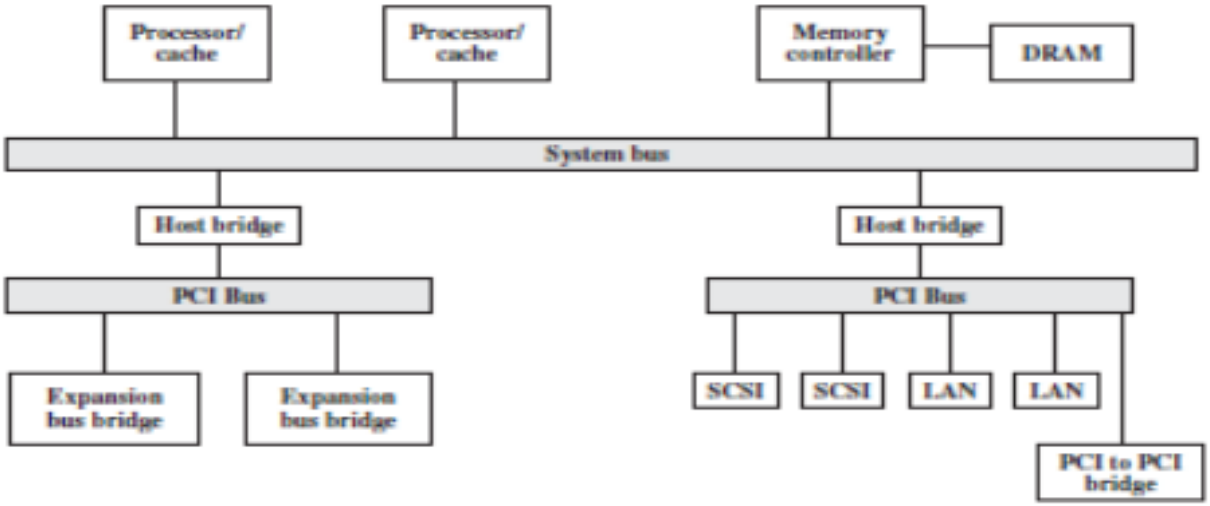
- The peripheral component interconnect (PCI) is a popular high-bandwidth, processorindependent bus that can function as a peripheral bus. The current standard allows the use of up to 64 data lines at 66 MHz, for a raw transfer rate of 528 MByte/s, or 4.224 Gbps. It requires very few chips to implement and supports other buses attached to the PCI bus.

PCI

- Intel began work on PCI in 1990 for its Pentium-based systems. The industry association, the PCI Special Interest Group (SIG), developed and further and maintained the compatibility of the PCI specifications. PCI is designed to support a variety of microprocessor-based configurations, including both single- and multiple-processor systems. It makes use of synchronous timing and a centralized arbitration scheme



(a) Typical desktop system



(b) Typical server system

Figure 1.23 Example PCI Configurations

- The 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. In a multiprocessor system (Figure 1.23b), one or more PCI configurations may be connected by bridges to the processor's system bus. Again, the use of bridges keeps the PCI independent of the processor speed yet provides the ability to receive and deliver data rapidly

Bus Structure

- PCI may be configured as a 32- or 64-bit bus. There are 49 mandatory signal lines for PCI which are divided into the following functional groups:
- System pins: Include the clock and reset pins.
- Address and data pins: Include 32 lines that are time multiplexed for addresses and data. The other lines in this group are used to interpret and validate the signal lines that carry the addresses and data.
- Interface control pins: Control the timing of transactions and provide coordination among initiators and targets.
- Arbitration pins: Unlike the other PCI signal lines, these are not shared lines. Rather, each PCI master has its own pair of arbitration lines that connect it directly to the PCI bus arbiter.
- Error reporting pins: Used to report parity and other errors. In addition, the PCI specification defines 51 optional signal lines, divided into the following functional groups:
- Interrupt pins: These are provided for PCI devices that must generate requests for service. As with the arbitration pins, these are not shared lines. Rather, each PCI device has its own interrupt line or lines to an interrupt controller.
- Cache support pins: These pins are needed to support a memory on PCI that can be cached in the processor or another device.
- 64-bit bus extension pins: Include 32 lines that are time multiplexed for ad dresses and data and that are combined with the mandatory address/data lines to form a 64-bit address/data bus.
- JTAG/boundary scan pins: These signal lines support testing procedures

PCI Commands

Bus activity occurs in the form of transactions between an initiator, or master, and a target. When a bus master acquires control of the bus, it determines the type of transaction that will occur next The commands are as follows:

- Interrupt Acknowledge
- Special Cycle
- I/O Read
- I/O Write
- Memory Read
- Memory Read Line
- Memory Read Multiple
- Memory Write
- Memory Write and Invalidate
- Configuration Read
- Configuration Write
- Dual address Cycle

- Interrupt Acknowledge is a read command intended for the device that functions as an interrupt controller on the PCI bus. The Special Cycle command is used by the initiator to broadcast a message to one or more targets. The I/O Read and Write commands are used to transfer data between the initiator and an I/O controller. The memory read and write commands are used to specify the transfer of a burst of data, occupying one or more clock cycles.

Read Command Type	For Cachable Memory	For Noncachable Memory
Memory Read	Bursting one-half or less of a cache line	Bursting 2 data transfer cycles or less
Memory Read Line	Bursting more than one-half a cache line to three cache lines	Bursting 3 to 12 data transfers
Memory Read Multiple	Bursting more than three cache lines	Bursting more than 12 data transfers

- The Memory Write command is used to transfer data in one or more data cycles to memory. The Memory Write and Invalidate command transfers data in one or more cycles to memory. In addition, it guarantees that at least one cache line is written.
- The two configuration commands enable a master to read and update configuration parameters in a device connected to the PCI. The Dual Address Cycle command is used by an initiator to indicate that it is using 64-bit addressing

Data Transfer

- Every data transfer on the PCI bus is a single transaction consisting of one address phase and one or more data phases.

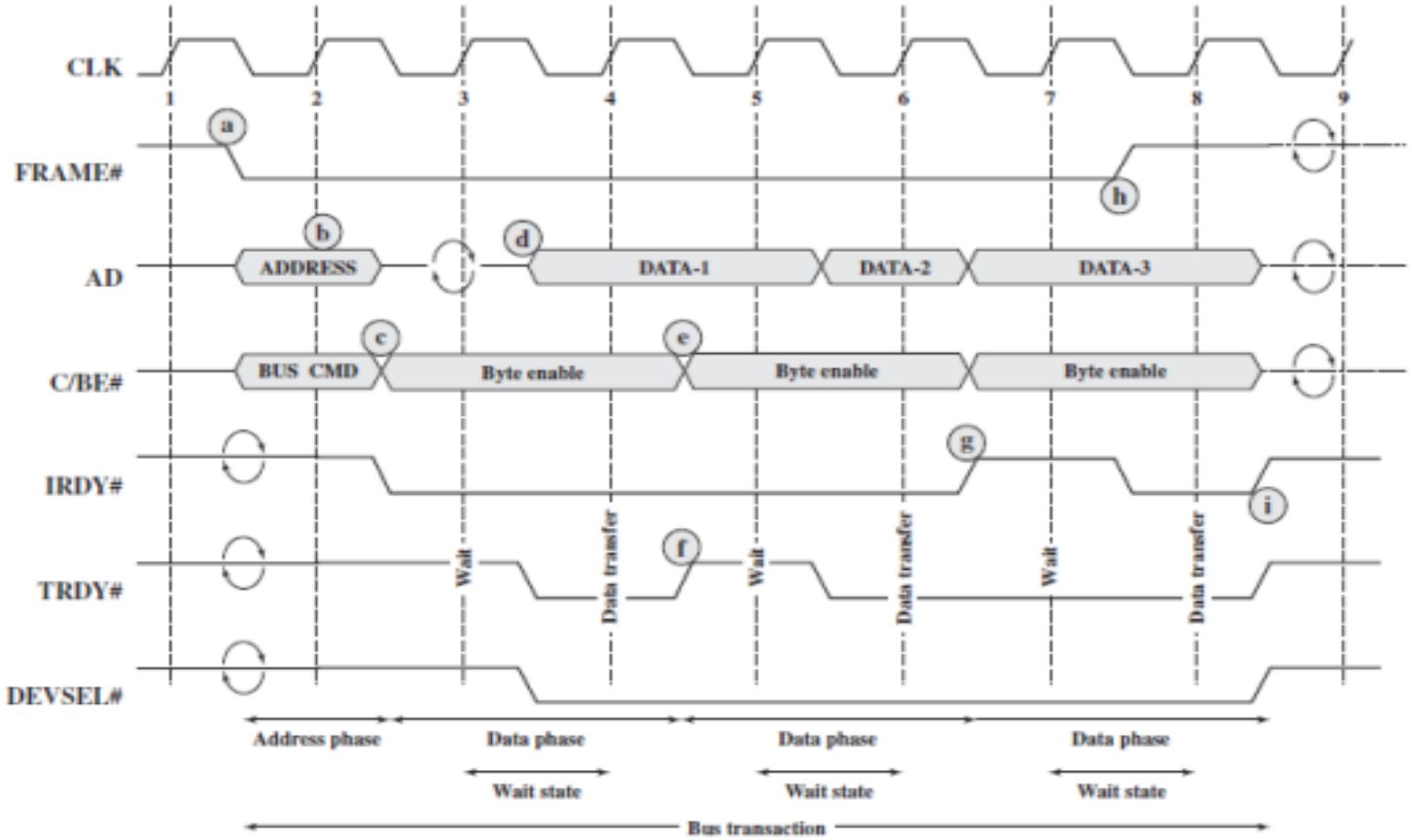


Figure 1.24 PCI Read Operation

Arbitration

- PCI makes use of a centralized, synchronous arbitration scheme in which each master has a unique request (REQ) and grant (GNT) signal. These signal lines are attached to a central arbiter (Figure 1.25) and a simple request–grant handshake is used to grant access to the bus.