6.1 Preamble

A ***bus*** is a set of wires/cables designed to transfer all bits of a w-bit word from a specified source to a specified destination. The source & destination are typically registers. So, a bus is a communication pathway connecting two or more registers within the system modules namely CPU, Memory, I/O etc.

So it can be said that multiple registers (CPU, Memory, I/O registers etc.) connect to the bus and at any time, a signal transmitted by a single register is available for reception by one or all other registers attached to the bus. Only one register can send data via the bus at any time. Information transmitted through the bus contains source & destination addresses. Only the destination register will read and accept the data while the other registers will reject it.

A bus structure consists of a set of common lines, one for each bit of a register, through which binary information is transferred one bit at a time. A bus may be unidirectional i.e. capable of transmitting data in one direction only, or it may be bi-directional i.e. capable of transmitting data in both direction.

Example with Block Diagram

The following diagram shows how information among various registers is transmitted via the bus.

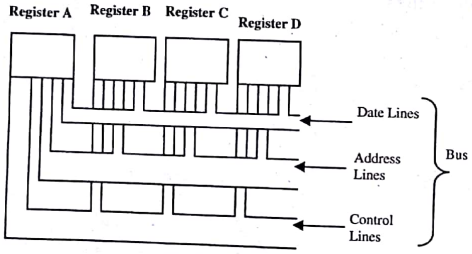


Fig: 1

Information transfer between two registers via a bus is often broken down into a sequence of two 'register' transfers:

Example

To transfer information from register A to register C in the above fig 1:

1st transfer: BUS ← Register A

i.e. information is transmitted from Register A to the bus.

2nd transfer: Register C ← BUS

i.e. information transmitted from the bus to the Register C.

So, with the help of two 'register' transfers, information from one register can be transmitted to another via a bus.

A bus is thus a communication pathway connecting two or more registers.

Multiple registers (CPU, Memory, I/O registers etc.) connect to the bus and at any time, a signal transmitted by a single register is available for reception by one or all other registers attached to the bus.

Information transmitted through the bus contains source & destination addresses. Suppose, in the fig 1, Register A sends an information via the bus to Register C. The information must contain the address of Register C (i.e. the destination address). All the registers connected to the bus will receive the information but registers, other than Register C, will reject the information as soon as they will find that their addresses are not matching the destination address given in the information. So, only Register C will read and accept the information.

Interfacing

An interface between two entities is a device that converts or translates the properties or characteristics of one entity to the corresponding properties or characteristics of another entity.

So a bus and CPU or bus and memory or bus and I/O unit may be the two entities of an interface (e.g. tristate buffer acts as an interface).

6.2 Internal and External Buses

The bus can be of internal and external types

(a) Internal bus: This runs within the CPU. Internal bus connects all the registers within the CPU.

Diagram

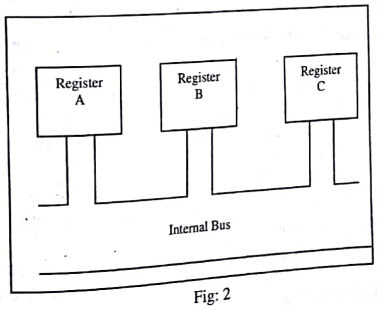


Fig: 2

(b) External / System bus: This runs outside the CPU i.e. the external bus connects all the three subsystems (i.e. connects all the registers within the CPU, Memory and I/O unit) of a digital computer. This is a set of shared communication lines via which the registers inside the Input-Output processors & the CPU share a common access path to main Memory registers.

Diagram

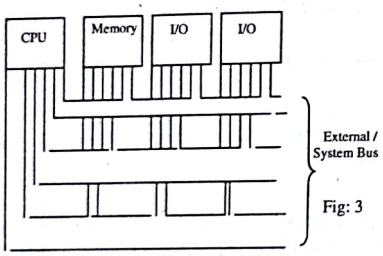


Fig: 3

6.3 Shared and Dedicated Buses

Shared Bus

Here several registers can be connected via a single common bus with the restriction that only one register can send data at any time.

So, here the number of buses and thus the number of cables required are much less. Cost of cables required, hence, is much less. However, shared buses do not permit simultaneous data transfers between different pair of registers, so this leads to a loss of performance compared to dedicated buses. To implement shared buses, more complicated logic circuits are needed.

Figure 1, shows a single shared bus between four registers A, B, C and D.

Dedicated Bus

It has a unique source and a unique destination i.e. within each pair of registers there is a pair of dedicated bus for both way transmission of information.

If 'n' registers are interconnected by buses in all possible ways, then the number of dedicated buses required is 'n (n-1)'.

Here the number of wires required is more. The number of wires required however varies with the number of registers to be connected. More the number of registers to be connected, more the number of wires needed.

As the number of system components and thus the number of buses to be connected increases, the number of pin requirements and the cost of cables along with the cost of complicated circuits also increases.

Here the simultaneous transfer of information between different pair of devices is possible.

Diagram

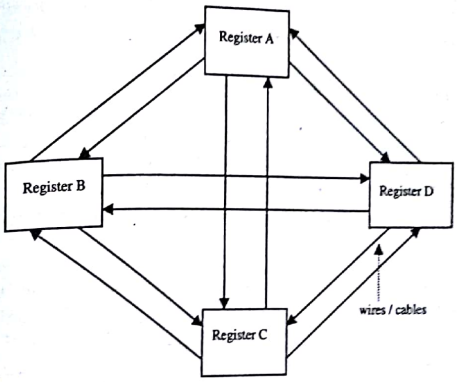


Fig: 4

Diagram of dedicated buses connecting the registers

In the fig 4, A, B, C and D are the registers, connected both way by buses, so that information can be transmitted between them in either direction, sequentially.

Here the number of dedicated buses required are n (n-1) = 4(4-1)=4*3=12, where 'n' is the number of registers.

However this concept of 'dedicated bus' are not used nowadays. In modern technologies, buses are always shared in nature.

Differences

	Shared Bus		DEDICATED BUS
1.	A single common bus shared by multiple registers.	1.	Within each register pair, there is a pair of dedicated bus.
2.	Number of cables required is much less.	2.	Number of cables required is much more.
3.	Less expensive.	3.	More expensive.
4.	More complicated logic circuits are needed.	4.	Less complicated logic circuits are needed.
5.	Less transmission speed.	5.	More transmission speed.
6.	Only one register can send data at any time.	6.	Simultaneous information transfer between different register pairs is possible.
7.	This concept is in use nowadays.	7.	Not used in modern day technologies.

6.4 Bus Master and Bus Slaves

Buses help in carrying out inter-register or intersystem data communication. An important design issue in this type of communication is the selection process by which a register gains access to the bus.

The particular unit (or register) that acts as the main bus in control unit is known as the ***bus master***. It supervises the use of the bus by the other units (or registers) known as the ***bus slaves***. So if there are multiple units connected to the bus, then one unit acts as the bus master and the others are known as the bus slaves. Only a bus master can initiate data transfer via the bus while bus slaves can only respond to the commands issued by a bus master. CPU is the usual bus master and memory and I/O units are the bus slaves.

Bus Initiator

A bus master is also called an ***initiator*** as it initiates data transfer on the bus by issuing commands.

Target

The devices, the bus master addresses to, are called the **target** (or slaves).

Bus Arbitration

Sometimes several units may generate requests for bus access simultaneously. In such cases the bus controller (bus master) must have a method for selecting one of the units that can use the bus at that particular time. This selection process is called the ***bus arbitration***. The three main arbitration schemes are daisy chaining, polling and independent requesting.

6.5 Control Signals

Signals controlling the information transfer through the bus are known as control signals. As only one register can transfer information through the bus at a time, these signals decide which register, out of the many registers, connected to the bus, will get access to the bus i.e. data of which register will be sent through the bus.

Example

Suppose, in fig 2, register A can transfer information through the shared bus, only when the control signal permits register A to transmit. At that time no other registers can transmit their information. Similar will be the situation for other registers also.

6.6 Data, Address and Control Lines of a Bus

The lines that constitute a communication bus can be divided into three functional groups:

Data Lines

These lines provide a path for moving data between registers. These are designed to transmit all bits of an n-bit word in parallel. So, they either consist of two sets of n-unidirectional lines or a single set of n-bi-directional lines.

Data lines are collectively called the ***data bus***. This bus of size 'n' is usually a multiple of 8, with n = 8, 16 or 32 etc. separate lines. Since each line can carry only 1 bit at a time, the number of lines determines how many bits can be transferred at a time.

Example with Diagram

Let us now consider an example. In fig 5, there are four registers A, B, C and D with four bits (3, 2, 1, 0) each connected by a common (shared) bus system.

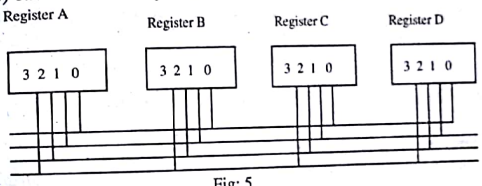


Fig: 5

As each line of the data bus can carry one bit at a time, so the data bus must have ***either*** two sets of 4 unidirectional lines [i.e. 2*(n=4) = 8] or 8 lines, out of which one set (i.e. 4 lines) will transmit data in the forward direction & the other set will transmit data in the reverse direction. ***Otherwise***, the data bus will have 4 (i.e. 1 set) bi-directional lines. Number of bi-directional lines in the data bus = number of bits in each register.

Address Lines

Address lines of the bus are used to transfer the addresses of the source register sending the data and the destination register receiving the data. Address lines are collectively called the ***address bus***.

Example

Consider figure 1. Suppose register A is sending data to register C. So via the address lines of the bus, both the sender's address (i.e. register A) and the receiver's address (i.e. register C) are transmitted to let all the registers sharing the bus identify the sender as well as the receiver of the data.

Control Lines

These lines, transmitting control signals, are used to control the access to the data & address lines i.e. control lines select which register, out of the multiple registers, will transmit its information through the bus.

Also control lines are used to transfer timing signals & status information about the registers in the system as well as to indicate the type of information present on the data lines.
Since all registers share the data & address lines, there must be means of controlling their use, which is done by the control lines.

Example

Fig 1, shows the control lines of the shared bus among the 4 registers.

6.7 Data / Address Multiplexing

Sometimes data lines may be used to transfer data as well as addresses. This is termed as ***data/address multiplexing***.

Use

This may be done to decrease the cost of the bus (i.e. less number of cables are required) and also to decrease the number of external connections (pins). This concept is generally common in case of data transfers via an I/O bus i.e. while the CPU is interfacing with the I/O units but not used in case of memory interfacing.

6.8 Synchronous and Asynchronous Bus Transfer

(A) Synchronous bus transfer

In synchronous data transfer over the bus each item is transferred during a time slot (clock cycle) known to both the source and destination units. Hence the bus interface circuits of both units are synchronized.
Synchronization can be achieved by connecting both source and destination units to a common clock source, which is feasible only over very short distances.

Advantages

Transfer rate is much fast and comparatively less complex circuitry needed.

Disadvantages

Here data-transfer rates are largely determined by the slowest units in the system, so some devices may not be able to communicate at their maximum rate.

(B) Asynchronous bus transfer

Here each item transferred, is accompanied by a control signal that indicates its presence to the destination unit i.e. whenever the source unit sends some information, it also sends a control signal to let the destination unit know that the information has been send. The destination can respond with another control signal to acknowledge receipt of the item. Here there is no concept of a predefined time-slot.
Because each device can generate bus-control signals at its own rate, data-transfer speed varies with the inherent speed of the communicating devices.

This technique is widely used in both local and (especially) long-distance communications.

Advantages

If the inherent speed of the communicating devices are not of the same order (i.e. one device is much fast than the other device) then the data transfer speed over the bus varies. In synchronous bus transfer this is a problem as some devices may not be able to communicate at their maximum rate and this problem reduces the bus performance. However owing to the handshaking concept, this problem is handled effectively in the asynchronous bus transfers where each communicating device can transfer as per their own transfer rate.

Disadvantages

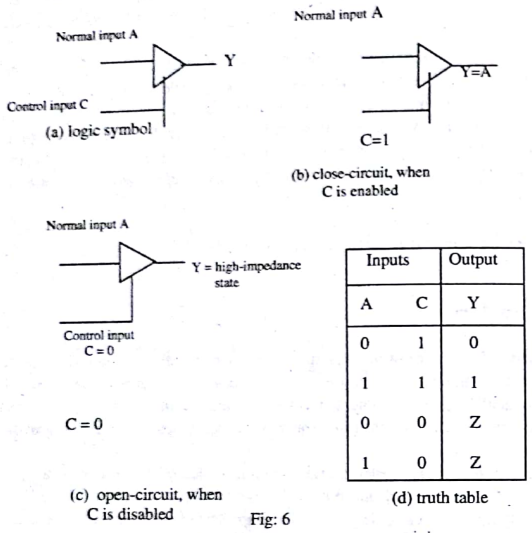
Due to more complex bus-control circuitry, cost is much more.

6.9 Tri-State Buffers

Bus system constructed with multiplexers has some serious disadvantages. If more number of registers are connected by the bus, multiplexers cannot support that load and voltages drop. Also as all registers (connected to the bus) draw some current from the bus even if they are not transmitting at that time, the bus voltage drops and thus the transfer becomes unreliable.
So to counter these problems a device (or a digital circuit) called ***tristate buffer*** is used. It exhibits three states, two inputs and one output. The output can be in one of the three states (signal values) namely, ***logic 0***, ***logic 1*** and a ***'high-impedance'*** state. While 0 & 1 typically correspond to two electrical voltage levels, e.g. 0 & 5V, the 'high-impedance' state represents the state of a line that is electrically disconnected from all voltage sources i.e. an open-circuited line.
This high-impedance state is very useful in building the bus in the computer because the registers that are interconnected through the bus (i.e. communicate through the bus) do not draw any current through the bus unless they either transmit or receive data over the bus lines.

Functioning

The tri-state buffer has three output states namely logic 0, logic 1 and a 'high-impedance' state. It has two inputs (figure a). The inputs A (normal input) and C (control input) are ordinary binary signals that assume only the values 0 & 1; the output Y can assume the values 0, 1 & high-impedance.
The special input line C, called the 'output enable' or 'control input' when set to 1 (figure b), the output Y = A i.e. acts as a close-circuited line with what given in the input A, comes to the output Y (i.e. if A = 0 then Y = 0 or if A = 1 then Y = 1) but if C = 0, it disables the output line Y by placing it in the high-impedance state.



Inputs		Output
A	C	Y
0	1	0
1	1	1
0	0	Z
1	0	Z

(d) truth table

Fig: 6

Need for Tri-State Buffers

Tristate buffer eliminates the problems with the multiplexers because of their high-impedance state. Use of tristate buffers allow a large number of registers to get connected to the bus without the bus transfer becoming unreliable.

Also a bus system built with tristate buffers is much cheaper compared to that built using multiplexers.

Multiple registers (CPU, Memory, I/O registers etc.) connect to the bus and at any time, a signal transmitted by a single register is available for reception by one or all other registers attached to the bus. Only one register can send data via the bus at any time. Information transmitted through the bus contains source & destination addresses. Only the destination register will read and accept the data while the other registers will reject it.

Related Questions & Answers

Question 1

With the help of a diagram show the bus system for data transfer among 4 registers using multiplexers.

Answer:

A bus system is used for transferring information between multiple registers. Control signals determine which register the bus selects during each particular information transfer i.e. as only one register can use or transfer information via the bus at a time hence with the help of the control signals, the particular register to transfer is selected.

The fig 7, below shows the bus system for four registers.

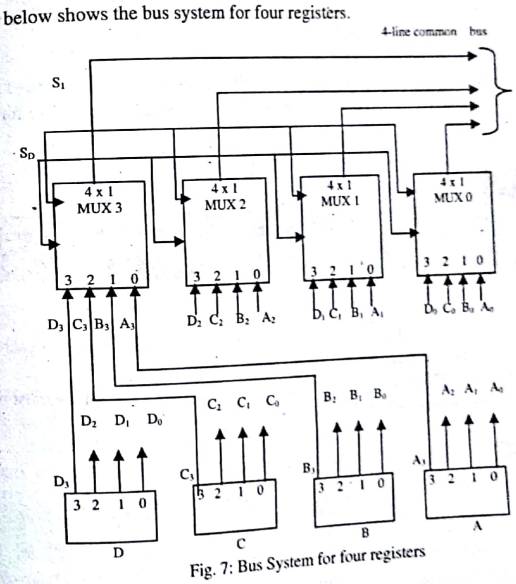


Fig. 7: Bus System for four registers

Four registers, A, B, C and D are connected with the help of the 4-line common bus. Four multiplexers (4 x 1) are used to select the particular register to transfer the information. Suppose each register has 4 bits of data, numbered 0 through 3. Each multiplexer along

with four data inputs (0 through 3), also has two selection inputs (S_1 and S_0). The selection lines choose the four bits of one register and transfer them into the four-line common bus. The 0-bits of all registers are connected to MUX 0, 1-bits of all registers are connected to MUX 1 and so on. So depending on the control inputs, the registers are selected accordingly.

The function table for the bus will show how the system works.

S_1	S_0	Register selected
0	0	A
0	1	B
1	0	C
1	1	D

When $S_1S_0 = 00$, the 0 data inputs of all multiplexers are selected and applied to the outputs that form the bus. Hence the contents of register A are send to the bus lines as all outputs of register A are connected to 0 data inputs of the multiplexers.

Similarly when $S_1S_0 = 01$, the 1 data inputs of all multiplexers are transmitted to the bus and hence the contents of register B are send to the bus. Similar is the case for the other registers.

Question 2

Suppose a common bus system is to be constructed for 8 registers each having 32 bits. How many multiplexers, data input lines and selection lines are needed for that?

Answer:

The number of multiplexers needed to construct the bus = number of bits in each register. Hence 32 multiplexers are needed.

Number of data input lines = number of registers. Hence 8 data input lines must be there.

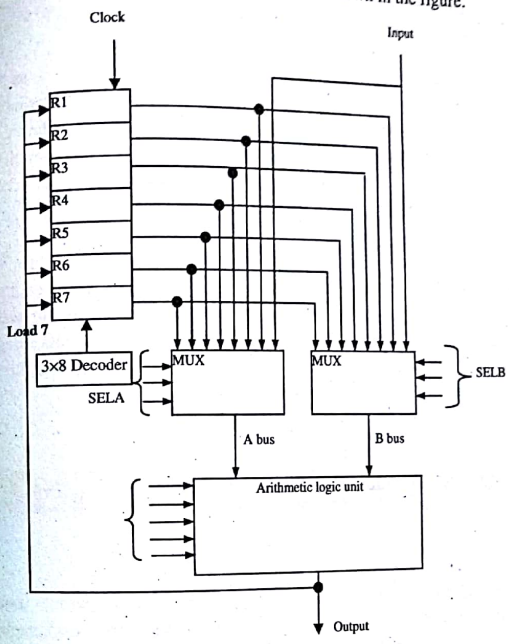
Number of selection lines = 3 (as $2^3 = 8$), since there are 8 registers.

Question 3

Draw the block diagram of a General Register organization & explain how $R_1 \leftarrow R_2 + R_3$ operation is being performed by the control unit by providing the binary selection variables to different selector inputs, where R_1, R_2 & R_3 are three general purpose Registers. Assume the suitable encoding for addition operation. **[WBUT 2003]**

Answer:

The block diagram of a general register system is shown in the figure.



Registers R_1 through R_7 (general purpose CPU registers) communicate with each other through a common bus system. The output of each register, as has been shown in the figure, is connected to two multiplexers (MUX) thus forming buses A and B, which are the inputs to the common ALU unit. As there are seven registers in the system, hence each MUX has three selection inputs, which select data from one particular register at a time to be sent o the common bus for transmission. The particular arithmetic or logic operation to be performed is selected in the ALU. Now information for a particular register comes from the output bus. The particular register, for which the information is meant, is selected by a decoder, which activates one register at a time to receive inputs from the outside bus.

The operation $R_1 \leftarrow R_2 + R_3$ is to be performed, which means that the contents of R_2 and R_3 registers to be added and the result to be kept in register R_1. To perform this operation, binary selection variables must be provided by the control to all concerned selector inputs, like:

MUX A selector (SELA): This will activate register R_2 to place its contents in the bus A.

MUX B selector (SELB): This will activate the register R_3 to place its contents in bus B.

ALU operation selector (OPR): Out of the many ALU operations that can be performed, the arithmetic addition line is selected for performing the A + B operation.

Decoder destination selector (SELD): This transfer the contents of the result of addition from the ALU unit via the output bus to store in the register R_1.

Generations of all the four control selection variables are done in the control unit and they must be available at the beginning of a clock cycle. In one clock cycle, contents of the two registers comes to the ALU unit via the MUX, desired operation is done in the ALU unit, result goes o the output bus, and then into the inputs of the destination register. In the next clock cycle, the binary information from the output bus is placed in to the destination register R_1.

Question 4

What are the disadvantages of using multiplexers?

Answer:

The ***disadvantages*** of using multiplexers in constructing a bus system are:

(a) As there is a limit to the number of inputs (fan-in) and number of outputs (fan-out) that a normal logic gate can support, hence if more number of registers are connected by the bus, multiplexers cannot support that load and voltages drop.

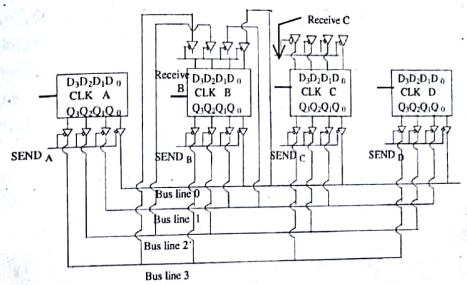
(b) Only one register, called the ***bus master***, transmits data over the bus at one time but any number of registers called ***bus slaves*** can receive this data from the bus (though only the destination register retains the data while others reject it). Now when multiplexers are used, each receiving register draws some current from the (i.e. even the registers that are not transmitting any data also draws some current) bus and the total current drawn from the bus by all the receivers together may greatly exceed the current outputting capacity of the bus and hence bus voltage may drop considerably and make the bus transfer unreliable.

Question 5

Draw the logic diagram of a common bus which connects 4 registers of 4-bit each using tristate buffers. **[WBUT 2004, 2010]**

Answer:

Consider the diagram below. A, B, C & D are the 4 registers connected through the 4-lines bus system. Both the inputs (D-ends of the flip-flops) & outputs (Q-end of the flip-flops) are connected through the bus system. Tristate buffers are there at both the sending & receiving ends of the registers.



Bus system Constructed with Tristate Buffers

Fig: 8

Suppose that register D wants to send data to register B through the bus. The steps to be performed are:

(a) First of all, a communication path is to be set-up between the sending (register D) & the receiving (register B) registers. So, in order to do this, it is needed to set the control inputs C of all the Tristate buffers of both the D and the B registers to 1 (i.e. the control inputs of all the Tristate buffers of both the registers are enabled). This will activate the data lines of the bus & thus set the communication path between the register D & register B.

(b) Once the path is set, it is needed to indicate (activate) the destination register B to receive the data from register D. For this it is needed to give or supply a 'clock pulse' in register B. As soon as the 'clock pulse' is 'high' (i.e. logic 1 state), all the data from register D through the bus will be received by register B.

(c) After this, if the register B wants to 'acknowledge' register D that the data has been received, again the control inputs of all the tristate buffers attached to both the registers will be enabled & thus the path between the 2 registers will be set.

(d) By enabling the clock pulse at register D, the 'acknowledgement' bits from register B through the bus will be received by register D.

(e) Once the transmission is complete, the communication path will be terminated i.e. the control inputs of all the tristate buffers at registers B & D will be disabled (i.e. C = 0 and thus Y = high-impedance) and hence the line between the 2 registers will be an open-circuit one. So no communication is possible between the registers (B & D) through the bus, now.

(f) When the communication between B & D is going on i.e. when the bus lines are allocated or controlled by register B & D, no other registers can transmit or use the bus simultaneously. So, at that time, the control inputs of the tristate buffers of the other registers will be disabled & hence being open-circuited, they won't have any connection with the bus & thus cannot draw any current (leakage current) from the bus till they are enabled.

(Note: The above figure does not show address & control lines).

Question 6

Why do most computers have a Common bus system? **[WBUT 2010]**

Or,

Explain the importance of a common bus system in a computer. **[WBUT 2011]**

Answer:

A bus is a communication pathway connecting two or more registers within the system modules namely CPU, Memory, I/O etc.

Early microcomputer bus systems were essentially a passive backplane connected directly or through buffer amplifiers to the pins of the CPU. Memory and other devices would be added to the bus using the same address and data pins as the CPU itself used, connected in parallel. Communication was controlled by the CPU, which had read and written data from the devices as if they are blocks of memory, using the same instructions, all timed by a central clock controlling the speed of the CPU. A common bus system reduces the number of wire network. This creates an organized and clear connection structure between certain devices. By using latches, same bus can be used for transferring of data or instructions as well. These may lead to simple system structure.

Question 7

What are the advantages of tristate logic circuits in the design of buses?

Answer:

Advantages

(a) They support bi-directional transmission over the bus by allowing the same bus connection to serve as an input port and as an output port at different times.

(b) They greatly increase the fan-in and fan-out limits of the bus lines, permitting very large numbers of devices to be attached to the same line.

(c) Here due to the 'high-impedance' state the registers that are interconnected through the bus (i.e. communicate through the bus) do not draw any current from the bus unless they either transmit or receive data over the bus lines i.e. when the registers are not enabled, they do not draw any current(e.g. leakage current as in a case of multiplexers) from the bus.

Question 8

What is Bus? How many buses are present in computer? **[WBUT 2006]**

Answer: Refer to sections 6.1 and 6.6.

Question 9

What are the probable factors on which choice of a bus design technique depends?

Answer:

The general design technique of a bus depends on choices and trade-offs among the following factors:

(a) The cabling material used.

(b) How simple the device interface is.

(c) The total time (probable) required for a bus transfer.

(d) The ability to accommodate multiple and varied device interfaces that (may) introduce varied amounts of delay in the bus.

(e) How much potential or efficient the bus is in detecting probable errors that can creep in while addressing a nonexistent device.

(f) How much ability the bus has in detecting and addressing errors that may arise due to malfunction of an interface.

Question 10

Write short notes on 'Bus organization using tristate buffer'. [WBUT 2007, 2009]

Answer: Refer to section 6.9.

Question 11

What are the different types of data exchanges that the interconnection structure must support?

Answer:

The following list of data exchanges should be readily supported by the interconnection structure:

a. Memory to processing unit: Data is read from the memory unit by the central processing unit.
b. Processing unit to memory: Data is written to the memory unit by the central processing unit.
c. Input-output unit to processing unit: Data is read from the I/O device by the central processing unit.
d. Processing unit to I/O: Data is written to the I/O device by the central processing unit.
e. I/O unit to the memory and I/O unit from the memory without the processing unit: Data is transferred directly to and from the memory to the I/O device using the direct memory access (DMA) concept by passing the central processing unit.

Question 12

Why do systems need multiple buses?

Answer:

Most current systems require hierarchically laid multiple buses. This is required to provide expected performances even when large number of devices are connected to the bus. The general reason is, propagation delay proportionately increases with the number of devices connected to the bus and greater the number of devices connected, greater is the delay. Now, during frequent switching of the control of the bus among the devices, the time taken to propagate this switching message among the various devices increases noticeably (for large number of devices) and hence it affects the performance of the entire system.

Also, with large number of devices connected to the bus, the huge number of demands for data transfers slowly approaches the capacity of the bus, creating a bottleneck. Hence, for the two above mentioned reasons, hierarchically laid multiple buses are needed.

Question 13

A digital computer has a common bus system for 16 registers of 32-bits each. The bus is constructed with multiplexers.
(i) How many selection inputs are there in each multiplexer?
(ii) What size of multiplexers are needed?
(iii) How many multiplexers are there in the bus? [WBUT 2010]

Answer: Refer to similar type question number 2.

Question 14

What do you mean by bus contention? Mention the potential hazards of bus contention. How can such a problem be dealt with?

Answer:

Bus contention is a condition that occurs when multiple devices attempt to use the bus (particularly data bus) simultaneously. Such a condition enabling two or more output buffers is more common in a transition period when one device is selected to use the bus and the other device is deselected. For example, suppose one device is trying to place signals or data in the bus even before another device has fully released the bus.

Such a problem can lead to excessive power consumption in the circuitry leading to an erroneous operation damaging the hardware. In the worst case, the bus wiring may get fused.

The problem of bus contention is more common in systems having multiple bus masters (multimaster busses) where each of the bus masters may wish to use the bus at the same time. No such general solution exists till date to resolve with this problem. Commonly used solutions are, either using a centralized arbitrator in the bus contention logic resolving the problem in a fixed finite time or by suitably buffering the outputs of the memory-mapped devices.

Question 15

What is a buffer?

Answer:

A buffer is a logic circuit having one input line and one output line (as shown in the figure below) in which logic 1 input provides logic 1 output (opposite to an inverter). Such a logic circuit generally amplifies the current or power and is also called a driver because of its usefulness in increasing the driving capabilities of a particular logic circuit.

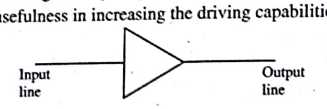


Fig. 9

Question 16

State down the critical characteristics of a bus.

Answer:

The critical characteristics of a bus are as follows:

1. *Width of a bus:* Given by the number of data lines the bus has.
2. *Type of a bus:* Master/Slave type or just bus-master
3. *The speed of the bus:* This is given in MHz and is reciprocal of the bus time
4. *Address space of a bus:* This is given by the number of address lines the bus has
5. *Throughput of a bus:* This is given in MB/sec

Question 17

What are the different performance parameters of a bus?

Answer:

These are as follows:

1. *Bandwidth / Throughput:* This measures the volume of quantity of data that can be transferred through the bus per unit time e.g. quantity of data (in bits) that can be transferred per second.
2. *Failure Rate / Error Rate:* This gives a measure of the probability of failure of a bus.
3. *Cost:* This gives a measure of the overhead that might be required to restart / re-functioning of the bus. The overhead depends either on the number of corrupted

packets (bits) to be resend or on the number of bus cycles (i.e. bus cycles x number of bits per cycle) to be flushed.

Question 18

With respect to the computer bus organization, explain the meaning of an expansion slot.

Answer:

The bus connects the different devices in a computer to the CPU, so that the CPU can successfully control them as and when necessary. An expansion slot enables a new device to be plugged-in to the bus and once plugged-in, the new device is added (i.e. it can communicate) to the entire system. Device drivers are required to be installed, though, for the new device to communicate to get itself identified and communicate to the CPU. So, the entire system does not need to be redesigned every time a new device is added to it.