

BASIC STRUCTURE OF COMPUTERS

Dr. Surekha Lanka

PhD,MTech,MSc,PGDBM,PGDPM,BSc.

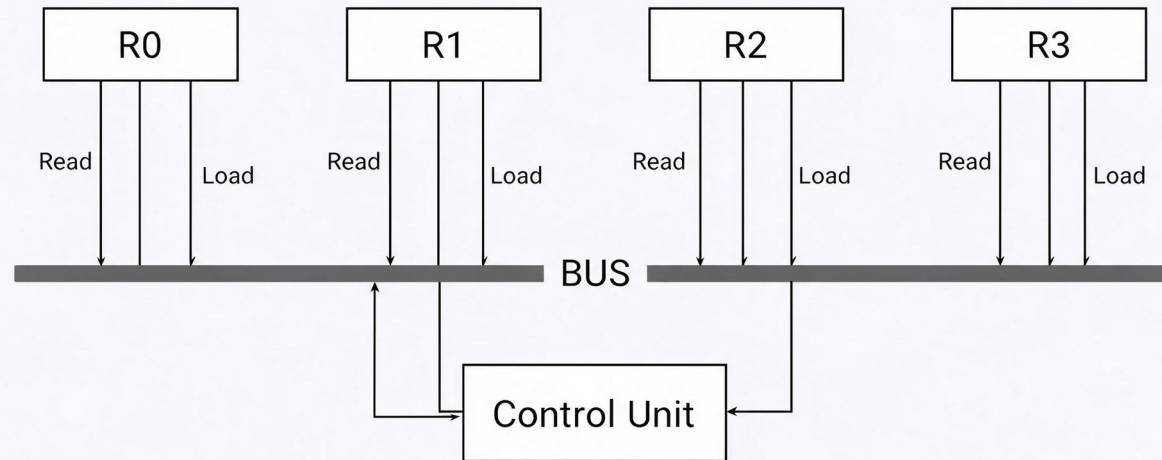
simple common bus system with 4 registers

CPU moves data from one register to another over a shared bus instead of using separate wires for every pair of registers.

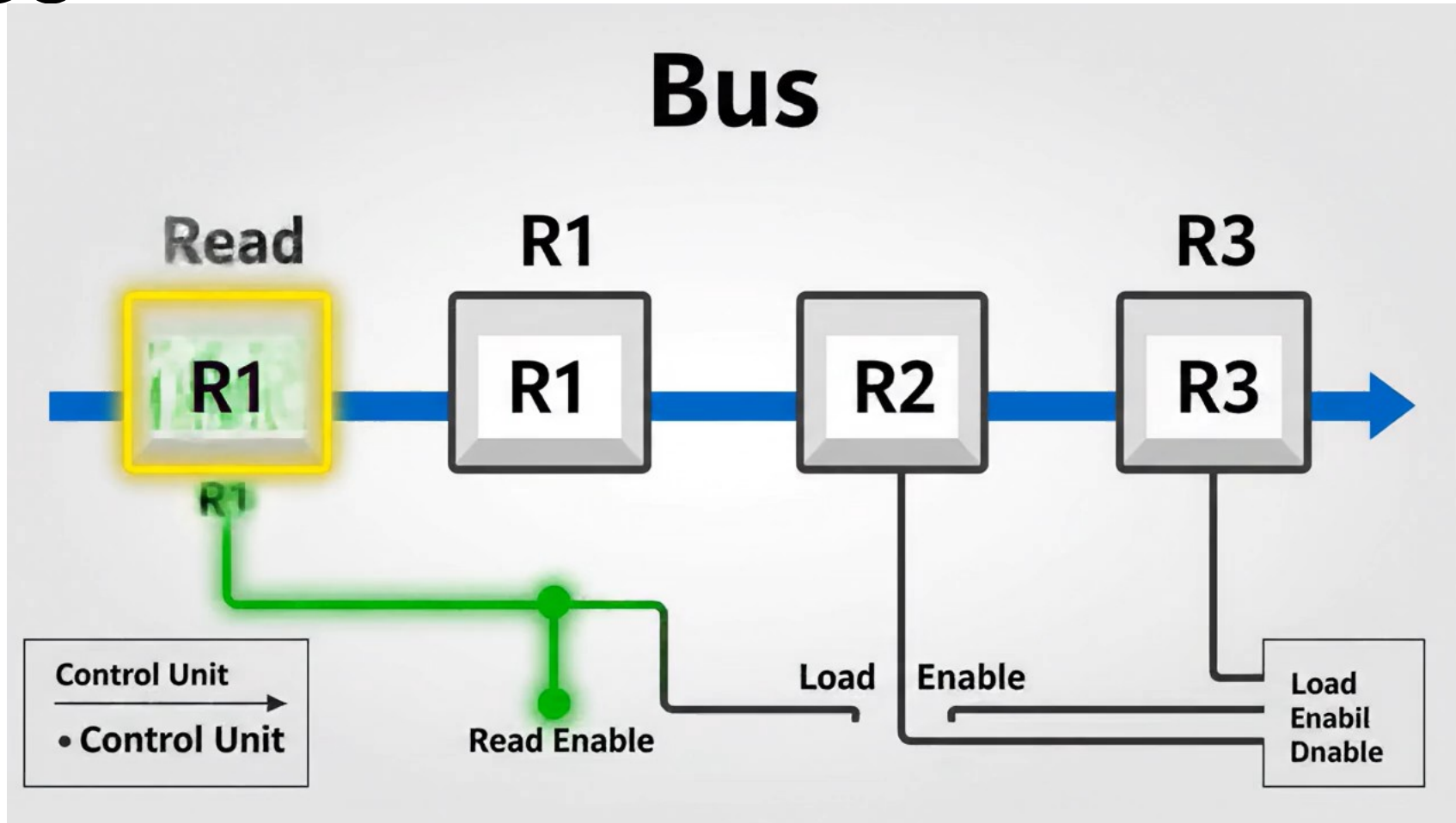
- Four registers, labeled R0, R1, R2, R3
- A horizontal line in the middle representing the **bus**, the shared data pathway.
- A simple **control unit** area that highlights which register is allowed to put data on the bus and which one is allowed to take data from it.
- multiple registers connected to one n-line common bus, with control signals choosing the source and destination.

4- Register Common Bus System

- In a 4-register common bus system, **read** means one register puts its value onto the shared bus, and **load** means another register copies the value from the bus into itself on the next clock edge.



video



In your video, the animation is showing the transfer path: source register → bus → destination register.

What read does

- **Read** is the action that makes a source register drive the bus. Only one register should be enabled to read onto the bus at a time, because the bus is shared.
- Example: if R1 contains 1011 and R1 is selected to read, then R1 places 1011 on the bus.

What load does

- **Load** is the action that makes a destination register accept whatever is currently on the bus. The register with its LD signal enabled stores the bus value at the next clock pulse.
- Example: if R3 has LD = 1 while the bus carries 1011, then R3 copies 1011 into itself.

Simple example

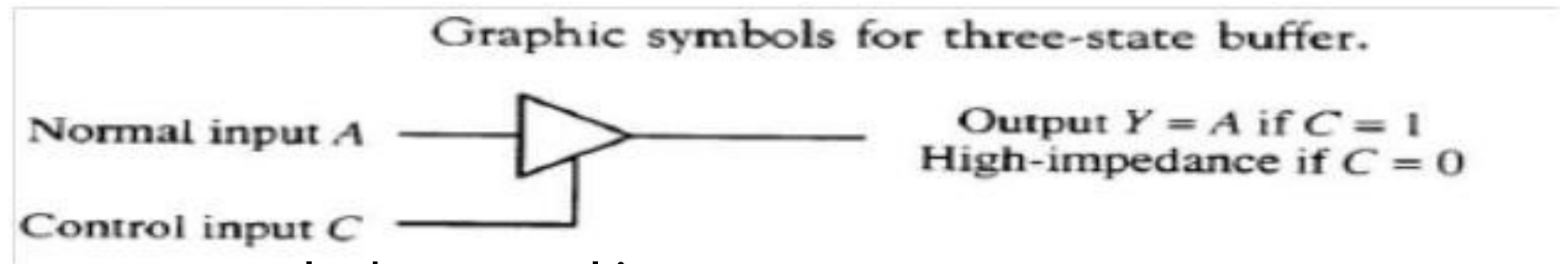
- Suppose you want to transfer data from R1 to R3:
- Enable R1 to **read** onto the bus.
- Put R1's contents on the bus, for example 1011.
- Enable R3's **load** input.
- On the next clock pulse, R3 stores 1011.
- So the bus is just the shared highway, **read** means "send data out," and **load** means "copy data in".

A three-state gate

A three-state gate is a digital circuit that exhibits three states

- Two of the states are signals equivalent to logic 1 and 0
- The third state is a *high-impedance* state – this behaves like an open circuit, which means the output is disconnected and does not have a logic significance

The three-state buffer gate has a normal input and a control input which determines the output state



- With control 1, the output equals the normal input
- With control 0, the gate goes to a high-impedance state
- This enables a large number of three-state gate outputs to be connected with wires to form a common bus line without endangering loading effects.

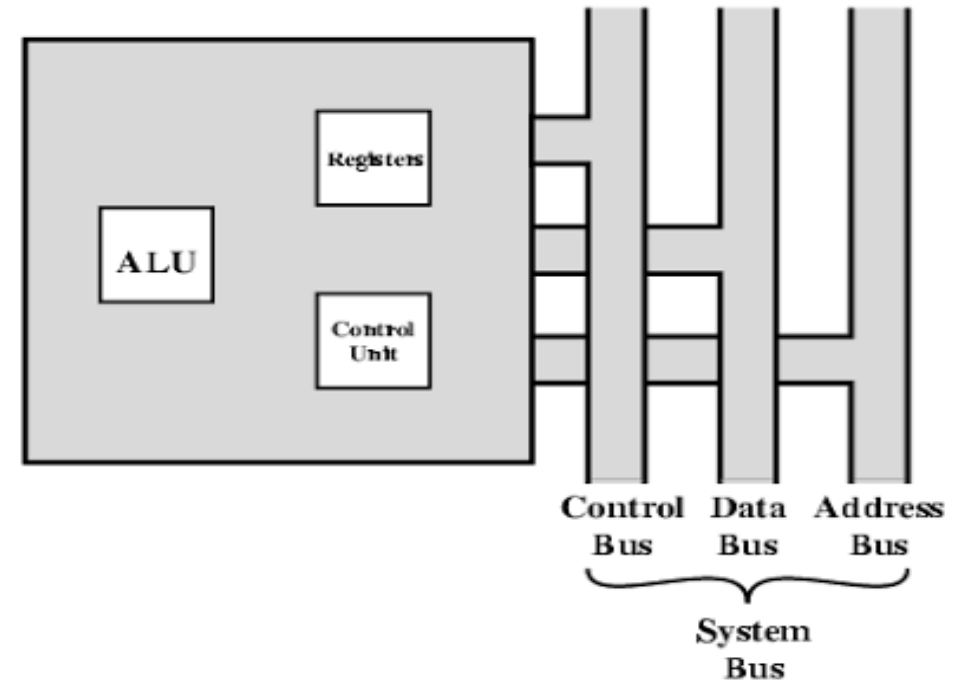
BASIC STRUCTURE OF COMPUTERS

CPU Organization

- CPU Structure and Function Processor Organization

- Things a CPU must do:

- - Fetch Instructions
- - Interpret Instructions
- - Fetch Data
- - Process Data
- - Write Data



- A small amount of internal memory, called the registers, is needed by the CPU to fulfil these requirements

REGISTER TRANSFER LANGUAGE

- The symbolic notation used to describe the micro operation transfers among registers is called a Register transfer language.
- Register transfer language
 - A symbolic language
 - A convenient tool for describing the internal organization of digital computers
 - Can also be used to facilitate the design process of digital systems.

REGISTER TRANSFER

- Registers are designated by capital letters, sometimes followed by numbers (e.g., A, R13, IR).
- Often the names indicate function:
 - MAR - memory address register
 - PC - program counter
 - IR - instruction register
- Information transfer from one register to another is designated in symbolic form by means of a replacement operator. $R2 \leftarrow R1$
- In this case the contents of register R1 are copied (loaded) into register R2 and contents of R1 remains same.

BLOCK DIAGRAM OF A REGISTER

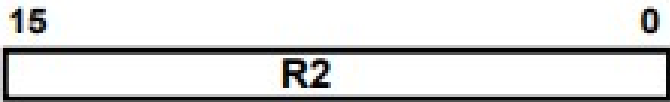


Register R

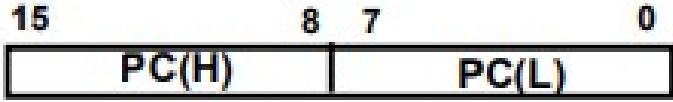


Showing individual bits

The individual flip-flops in an n-bit register are numbered in sequence from 0 to n-1 (from the right position toward the left position) R 1 7 6 5 4 3 2 1 0



Numbering of bits



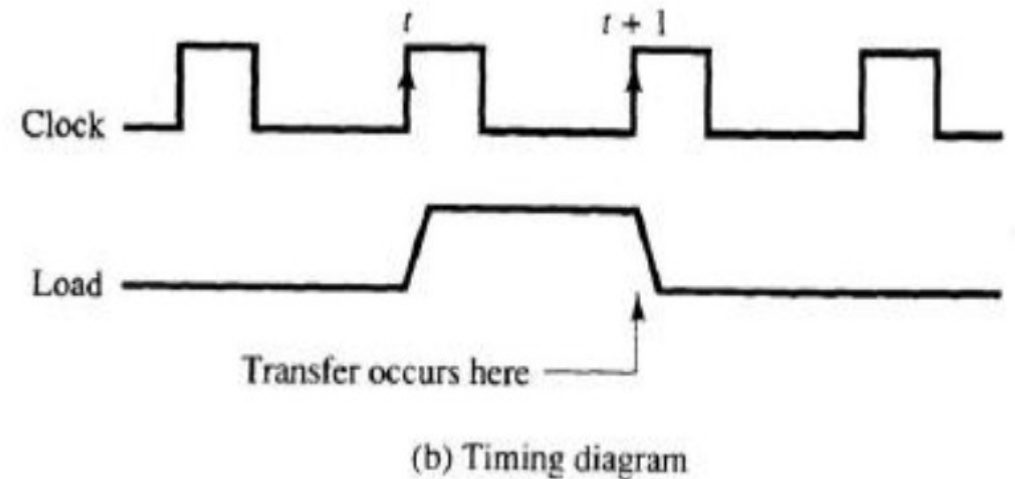
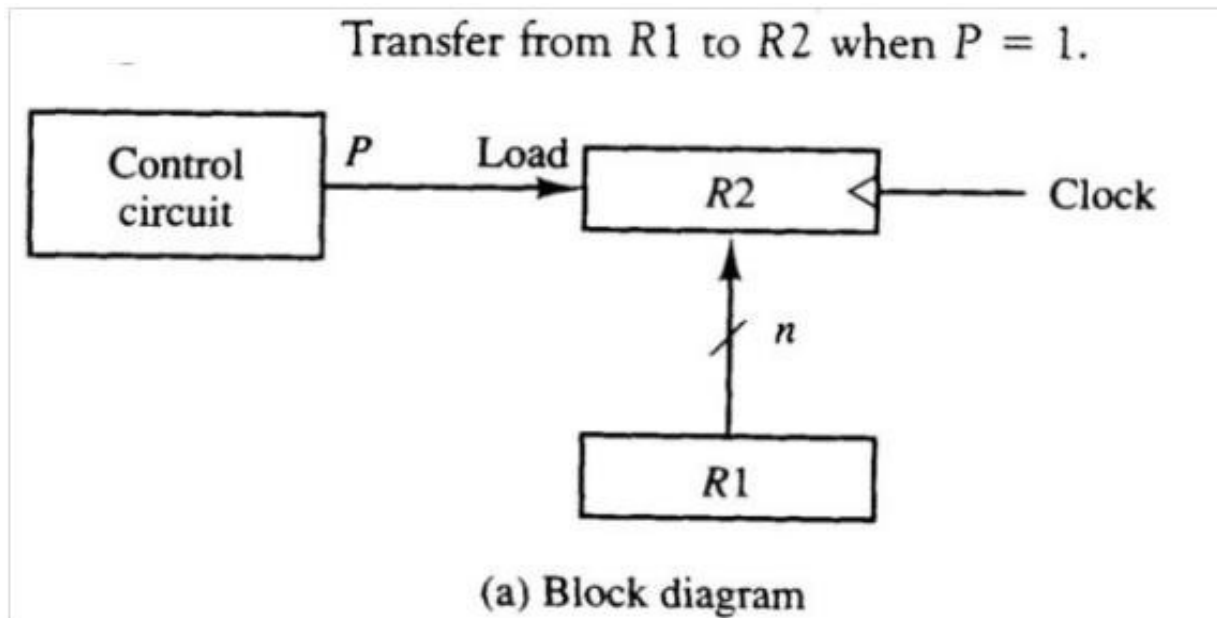
Subfields (Divided into two parts)

15 0 PC Numbering of bits 15 Upper byte PC(H) 8 7 0 PC(L) Lower byte Partitioned into two parts

HARDWARE IMPLEMENTATION OF CONTROLLED TRANSFERS

- Designate information transfer from one register to another by $R2 \leftarrow R1$
- This statement implies that the hardware is available
- The outputs of the source must have a path to the inputs of the destination
- The destination register has a parallel load capability
- If the transfer is to occur only under a predetermined control condition, designate it by
 - *If* ($P = 1$) *then* ($R2 \leftarrow R1$) or,
 - $P: R2 \leftarrow R1$,
 - Where P is a control function that can be either 0 or 1
 - Every statement written in register transfer notation implies the presence of the required hardware construction

- It is assumed that all transfers occur during a clock edge transition
- All micro operations written on a single line are to be executed at the same time T : $R2 \leftarrow R1, R1 \leftarrow R2$
- The same clock controls the circuits that generate the control function and the destination register



Basic Symbols for Register Transfers

Symbol	Description	Examples
Letters (and numerals)	Denotes a register	<i>MAR, R2</i>
Parentheses ()	Denotes a part of a register	<i>R2(0–7), R2(L)</i>
Arrow $\leftarrow$	Denotes transfer of information	<i>R2 $\leftarrow$ R1</i>
Comma ,	Separates two microoperations	<i>R2 $\leftarrow$ R1, R1 $\leftarrow$ R2</i>

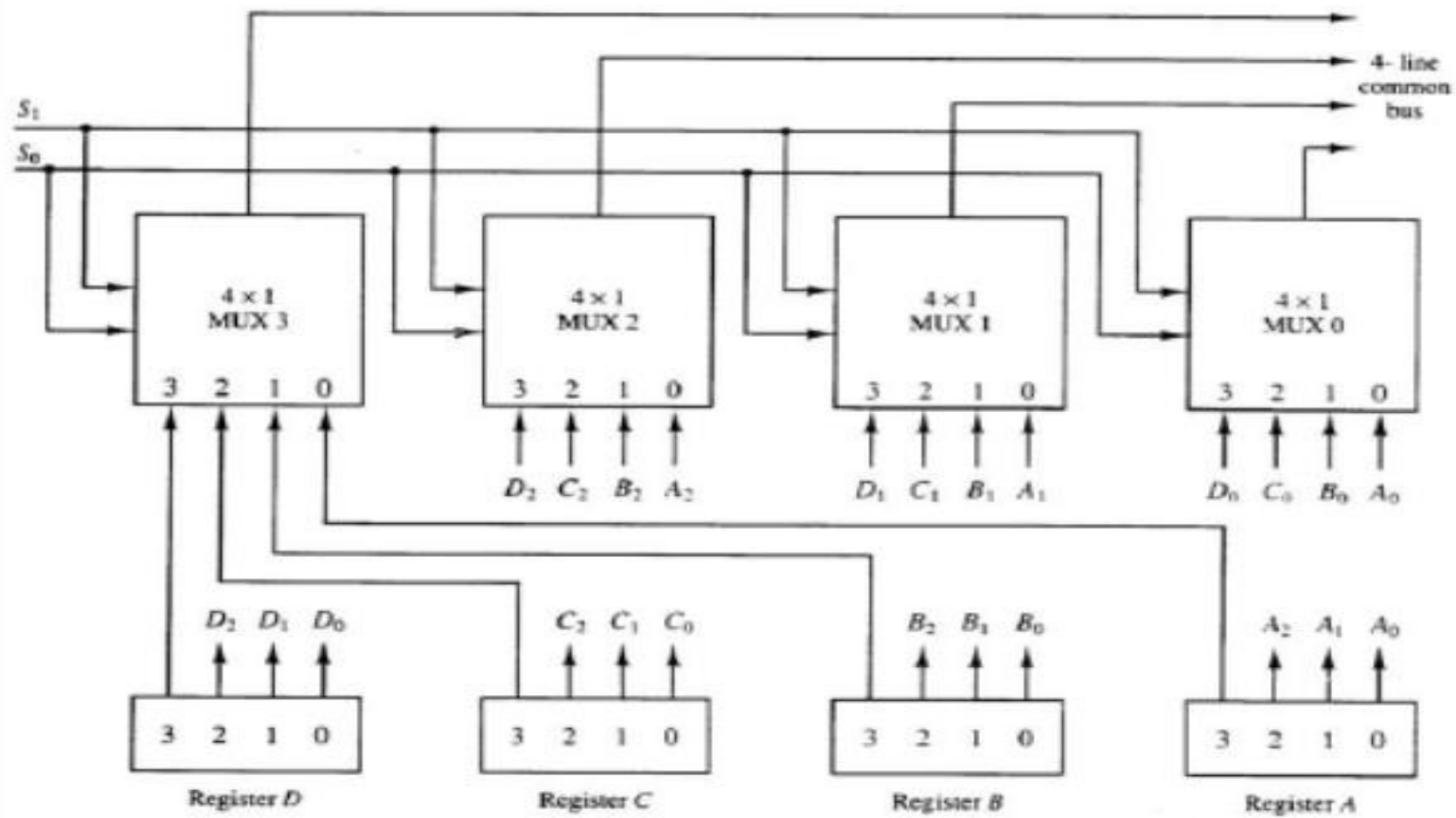
SIMULTANEOUS OPERATIONS

- If two or more operations are to occur simultaneously,
- they are separated with commas P: R3 <-R5, MAR <- IR
- Here, if the control function P = 1, load the contents of R5 into R3, and at the same time (clock), load the contents of register IR into register MAR

Bus and Memory Transfers

- Bus is a path(of a group of wires) over which information is transferred, from any of several sources to any of several destinations
- Rather than connecting wires between all registers, a common bus is used
- A bus structure consists of a set of common lines, one for each bit of a register
- Control signals determine which register is selected by the bus during each transfer
- Multiplexers can be used to construct a common bus
- Multiplexers select the source register whose binary information is then placed on the bus
- The select lines are connected to the selection inputs of the multiplexers and choose the bits of one register .

Bus system for four registers.



TRANSFER FROM BUS TO A REGISTER

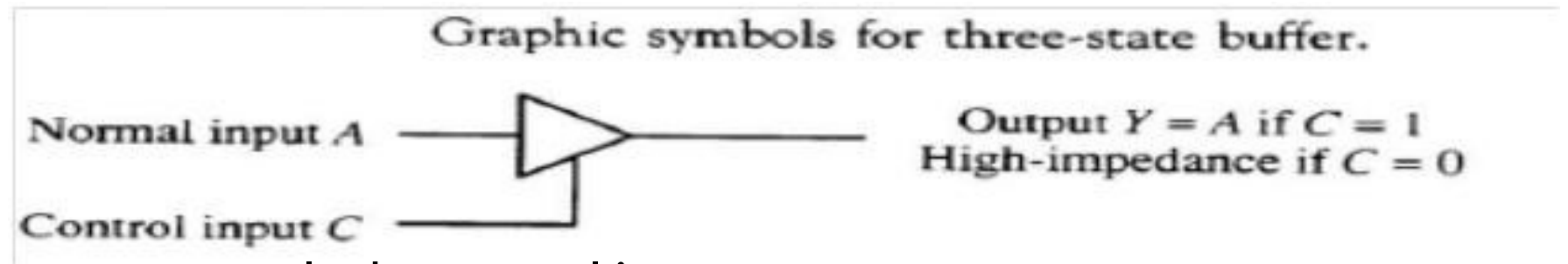
- In general, a system will multiplex k registers of n bits each to produce an n -line common bus
- This requires n multiplexers – one for each bit
- The size of each multiplexer must be $k \times 1$
- The number of select lines required is $\log k$
- • To transfer information from the bus to a register, the bus lines are connected to the inputs of all destination registers and the corresponding load control line must be activated
- Rather than listing each step as
- $BUS \leftarrow C, R1 \leftarrow BUS,$
- use $R1 \leftarrow C$, since the bus is implied
- Instead of using multiplexers, *three-state gates* can be used to construct the bus system

TRANSFER FROM BUS TO A REGISTER

A three-state gate is a digital circuit that exhibits three states

- Two of the states are signals equivalent to logic 1 and 0
- The third state is a *high-impedance* state – this behaves like an open circuit, which means the output is disconnected and does not have a logic significance

The three-state buffer gate has a normal input and a control input which determines the output state



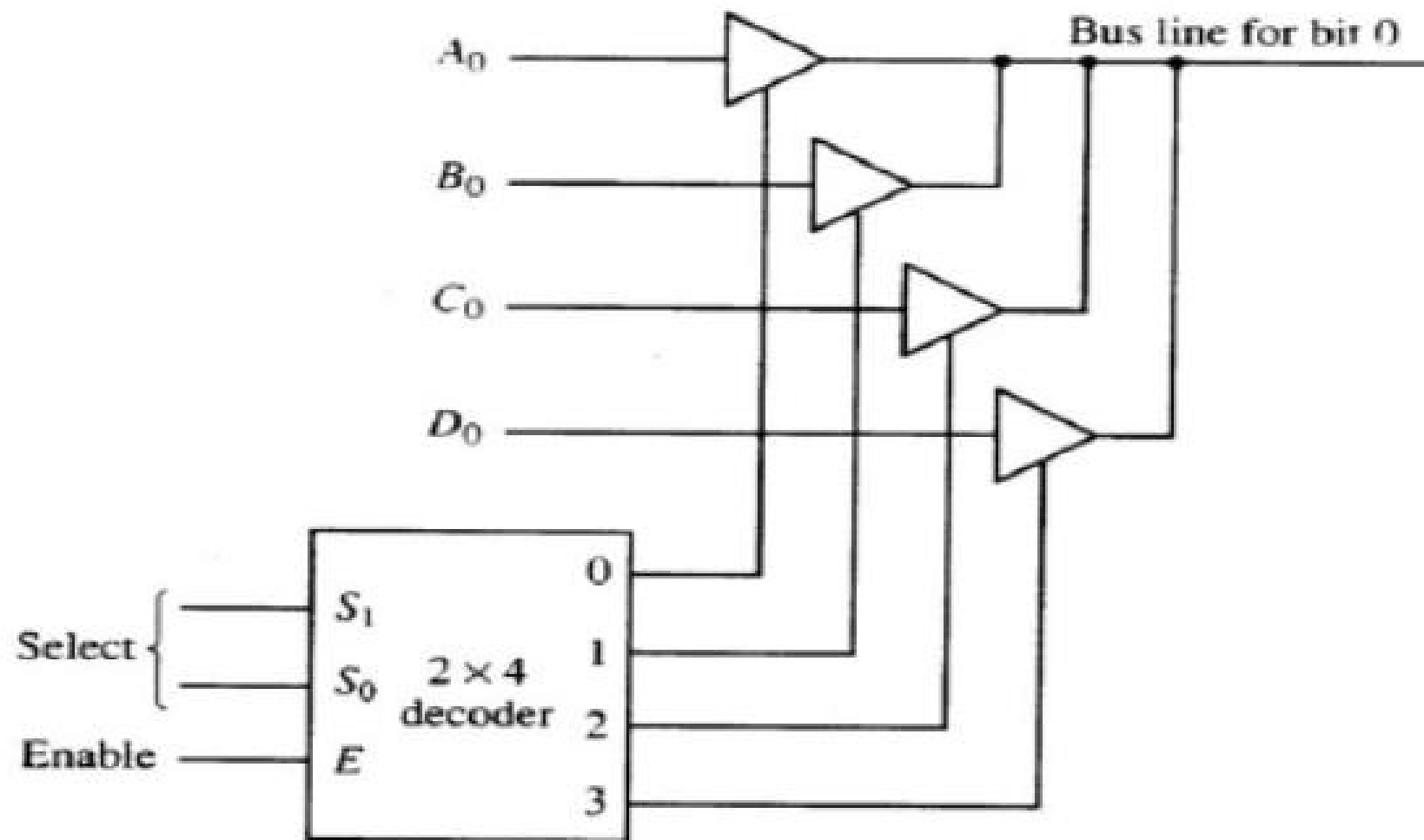
- With control 1, the output equals the normal input
- With control 0, the gate goes to a high-impedance state
- This enables a large number of three-state gate outputs to be connected with wires to form a common bus line without endangering loading effects.

Bus line with 3 state buffers

- Decoders are used to ensure that no more than one control input is active at any given time
- This circuit can replace the multiplexer in Figure 4.3
- To construct a common bus for four registers of n bits each using three-state buffers, we need n circuits with four buffers in each
- Only one decoder is necessary to select between the four registers
- Designate a memory word by the letter M
- It is necessary to specify the address of M when writing memory transfer operations

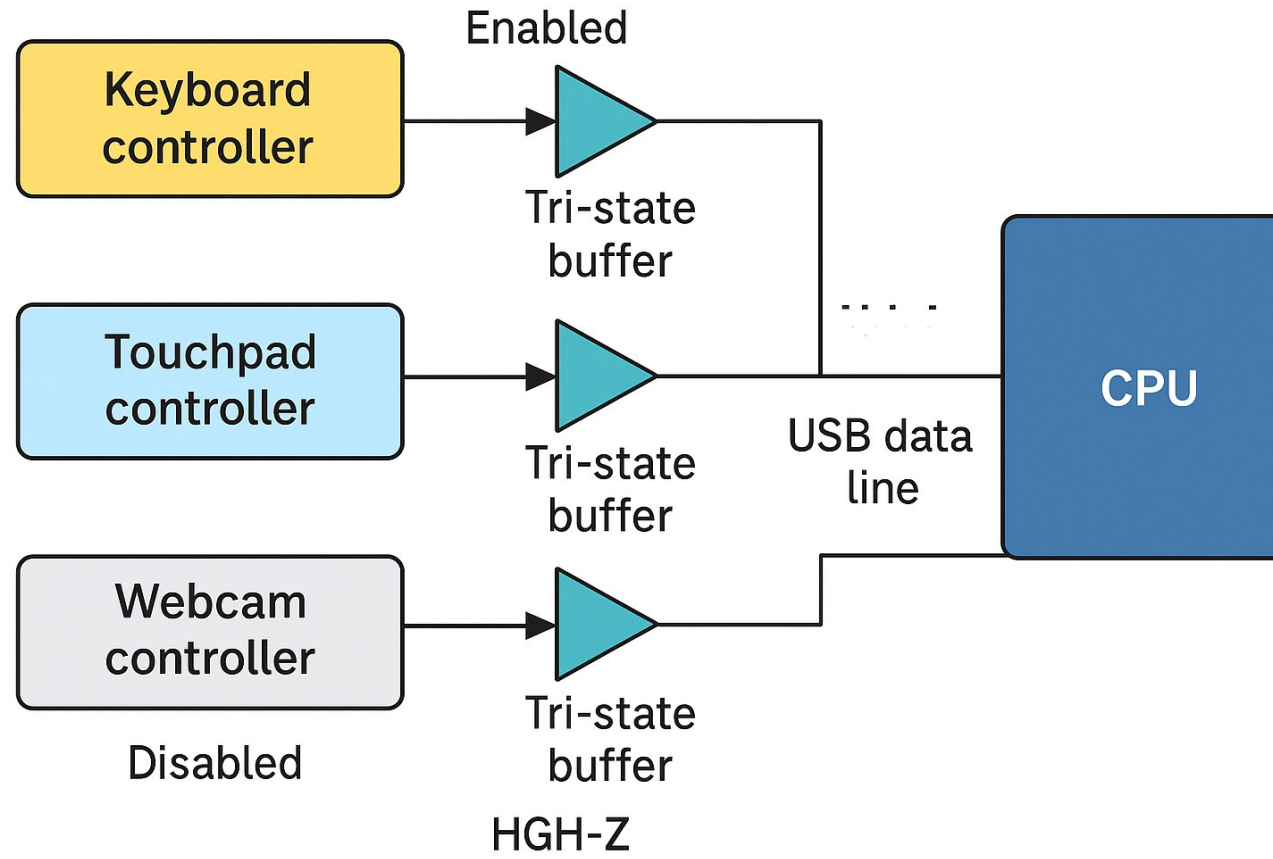
Bus line with 3 state buffers

- Designate the address register by AR and the data register by DR
- The read operation can be stated as:
 - Read: $DR \leftarrow M[AR]$
- The write operation can be stated as:
 - Write: $M[AR] \leftarrow R1$



Bus line with three state-buffers.

Real Scenario for the Tri-state Buffer



Quiz Time

- **1. Why do USB devices like keyboard, touchpad, and webcam need tri-state buffers when sharing one data line?**
- A. To increase their power consumption
- B. To ensure only one device sends data at a time
- C. To make the USB cable longer
- D. To convert digital data to analog signals

- **1. Why do USB devices like keyboard, touchpad, and webcam need tri-state buffers when sharing one data line?**
- A. To increase their power consumption
- B. To ensure only one device sends data at a time
- C. To make the USB cable longer
- D. To convert digital data to analog signals

- **2. What happens when a device's tri-state buffer is in Hi-Z (high-impedance) mode?**
- A. It forces a constant 1 onto the bus
- B. It forces a constant 0 onto the bus
- C. It disconnects from the USB data line
- D. It sends random data

- **2. What happens when a device's tri-state buffer is in Hi-Z (high-impedance) mode?**
- A. It forces a constant 1 onto the bus
- B. It forces a constant 0 onto the bus
- C. It disconnects from the USB data line
- D. It sends random data

- **3. In the scenario, the keyboard is enabled and sending data. What must the other devices (touchpad, webcam) do?**
- A. Send data at the same time
- B. Switch OFF their power
- C. Enter Hi-Z state to avoid bus conflict
- D. Increase their clock speed

- **3. In the scenario, the keyboard is enabled and sending data. What must the other devices (touchpad, webcam) do?**
- A. Send data at the same time
- B. Switch OFF their power
- **C. Enter Hi-Z state to avoid bus conflict**
- D. Increase their clock speed

- **4. What problem would occur if two USB devices send data on the same line simultaneously without tri-state control?**
- A. The CPU will ignore both devices
- B. The USB line will become faster
- C. Data collision or short circuit
- D. The devices will automatically synchronize

- **4. What problem would occur if two USB devices send data on the same line simultaneously without tri-state control?**
- A. The CPU will ignore both devices
- B. The USB line will become faster
- C. **Data collision or short circuit**
- D. The devices will automatically synchronize

Arithmetic Micro operations

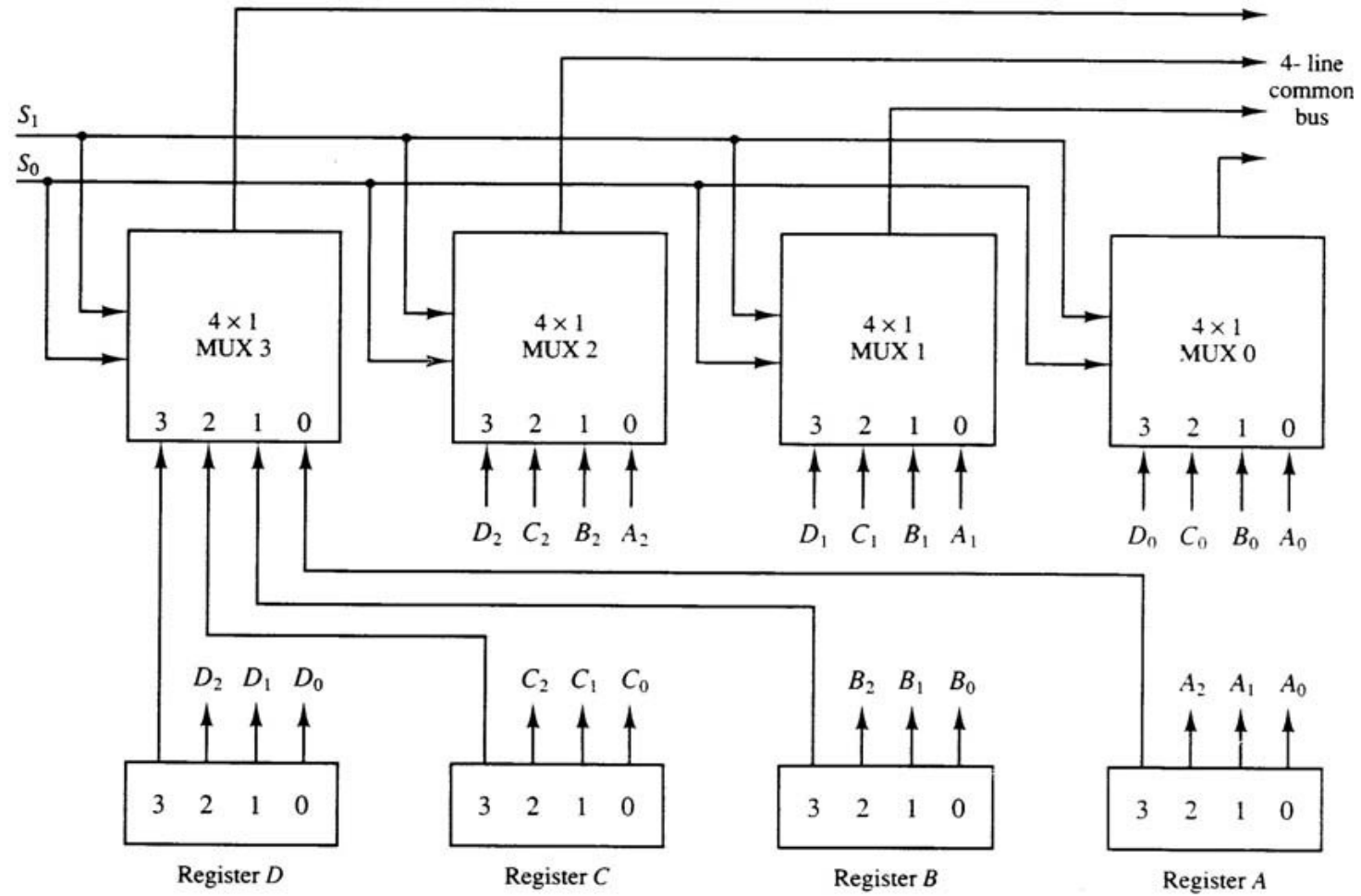
- There are four categories of the most common micro operations:
- Register transfer: transfer binary information from one register to another
- Arithmetic: perform arithmetic operations on numeric data stored in registers
- Logic: perform bit manipulation operations on non-numeric data stored in registers
- Shift: perform shift operations on data stored in registers
- The basic arithmetic micro operations are addition, subtraction, increment, decrement, and shift

Arithmetic Micro operations

- Example of addition: $R3 \leftarrow R1 + R2$
- Subtraction is most often implemented through complementation and addition
- Example of subtraction: $R3 \leftarrow R1 + \overline{R2} + 1$ (strikethrough denotes bar on top – 1's complement of R2)
- Adding 1 to the 1's complement produces the 2's complement .
- Adding the contents of R1 to the 2's complement of R2 is equivalent to subtracting.

- Multiply and divide are not included as micro operations
- A micro operation is one that can be executed by one clock pulse
- Multiply (divide) is implemented by a sequence of add and shift micro operations (subtract and shift)
- To implement the add micro operation with hardware, we need the registers that hold the data and the digital component that performs the addition
- A full-adder adds two bits and a previous carry
- A binary adder is a digital circuit that generates the arithmetic sum of two binary numbers of any length
- A binary adder is constructed with full-adder circuits connected in cascade
- An n -bit binary adder requires n full-adders

Figure 4-3 Bus system for four registers.



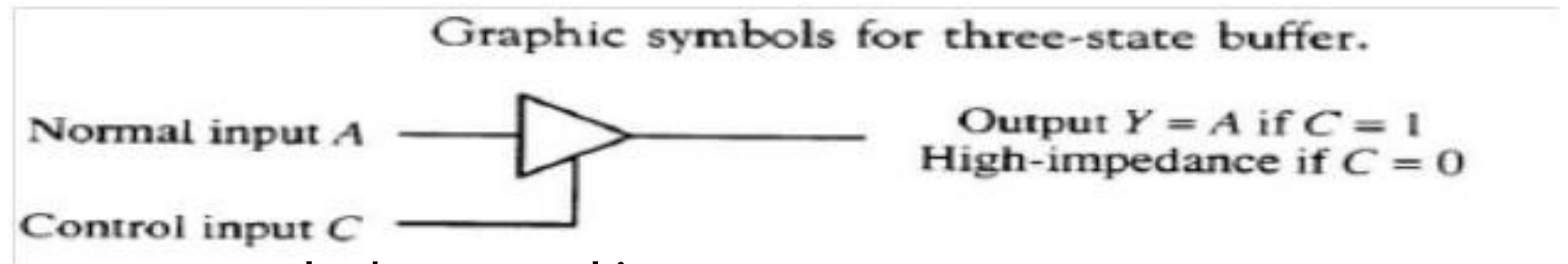
BASIC STRUCTURE OF COMPUTERS

TRANSFER FROM BUS TO A REGISTER

A three-state gate is a digital circuit that exhibits three states

- Two of the states are signals equivalent to logic 1 and 0
- The third state is a *high-impedance* state – this behaves like an open circuit, which means the output is disconnected and does not have a logic significance

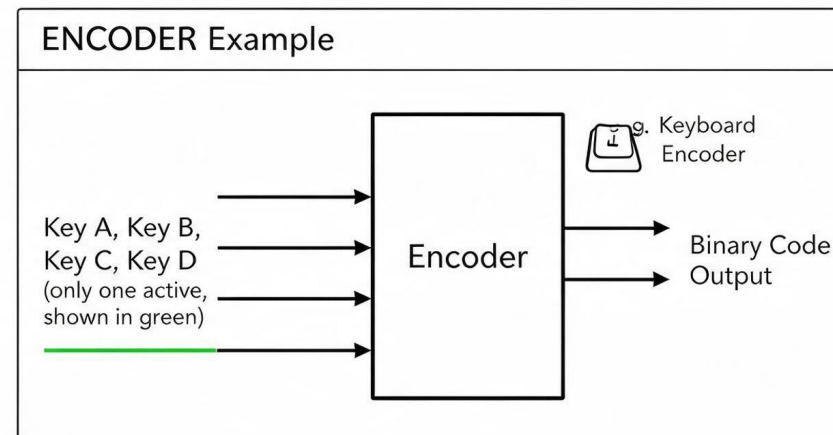
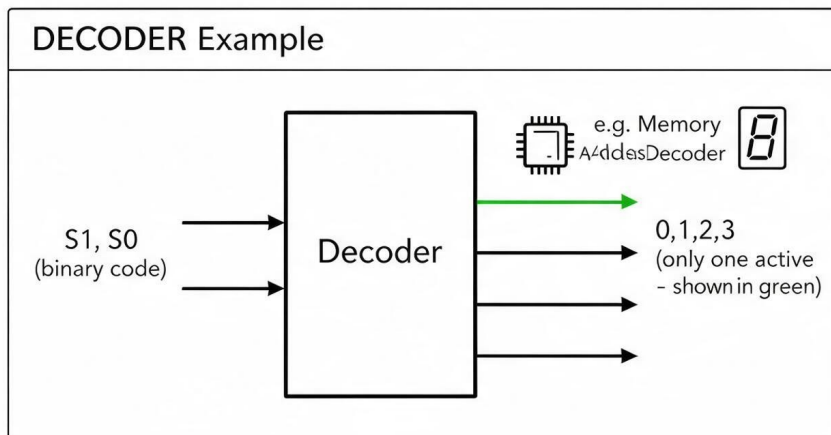
The three-state buffer gate has a normal input and a control input which determines the output state



- With control 1, the output equals the normal input
- With control 0, the gate goes to a high-impedance state
- This enables a large number of three-state gate outputs to be connected with wires to form a common bus line without endangering loading effects.

Encoder and Decoder

- **Decoder:** Converts an n-bit binary code into one active output out of 2^n lines — e.g., a 2×4 decoder uses 2 select bits to activate exactly 1 of 4 outputs.
- **Encoder:** Converts one active input out of 2^n lines into a compact n-bit binary code — the reverse operation of a decoder.

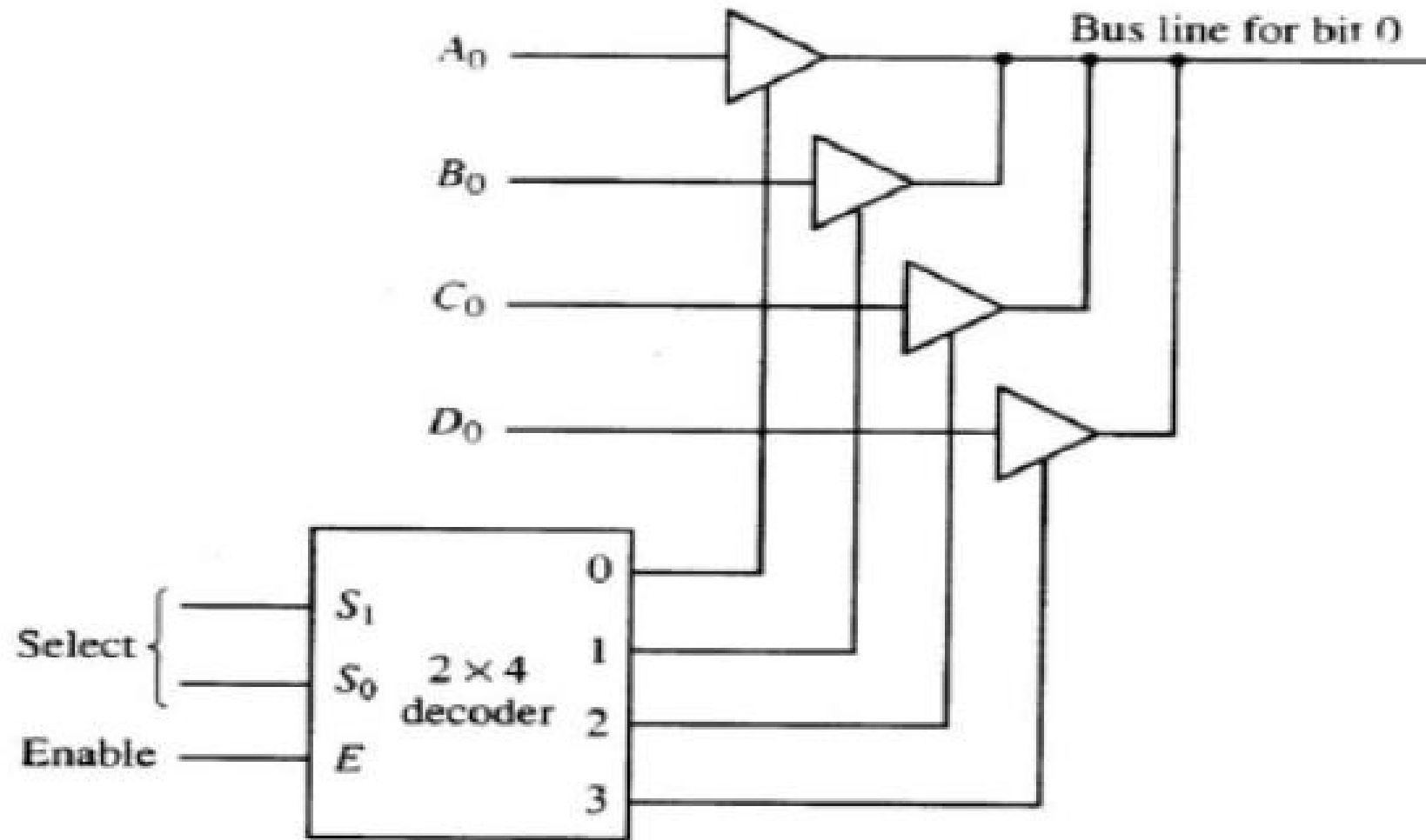


Bus line with 3 state buffers

- To construct a common bus for four registers of n bits each using three-state buffers, we need n circuits with four buffers in each
- Only one decoder is necessary to select between the four registers
- Designate a memory word by the letter M
- It is necessary to specify the address of M when writing memory transfer operations

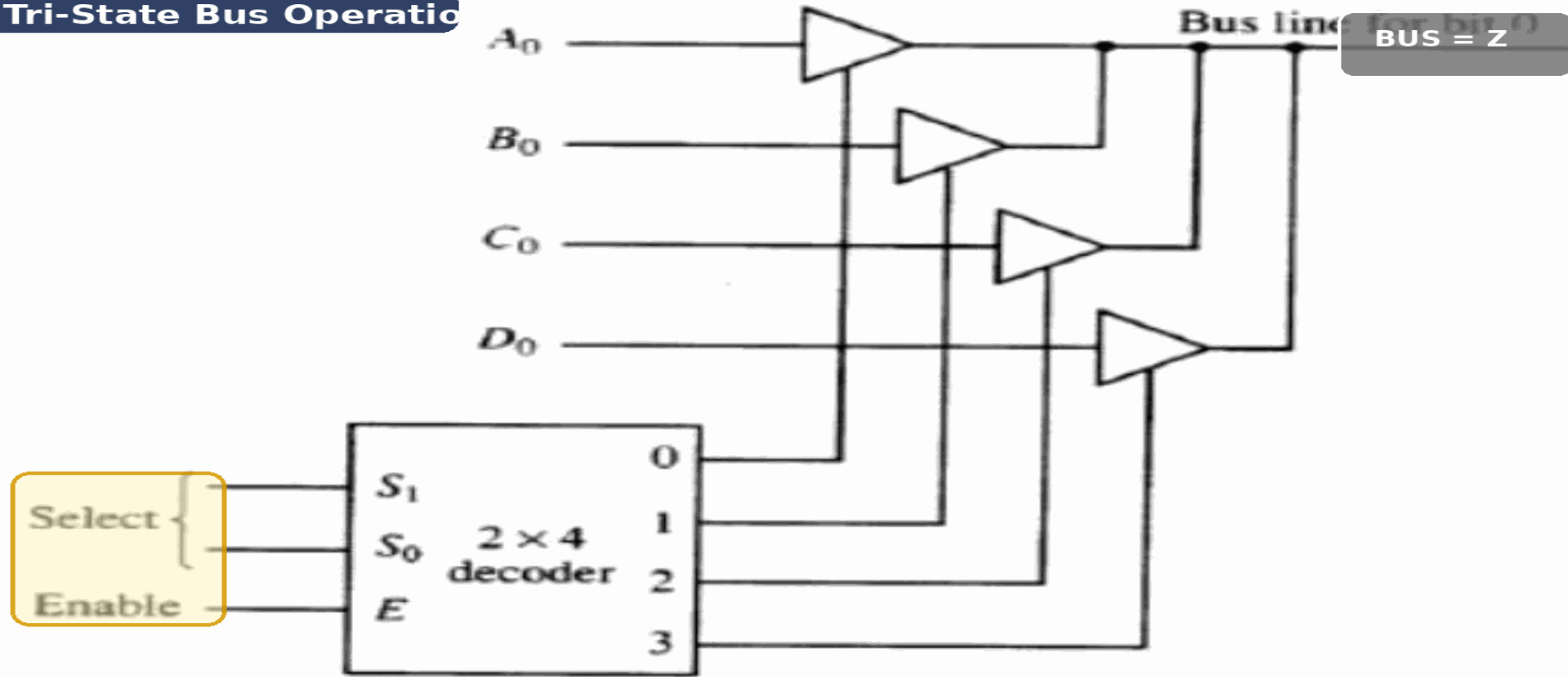
Bus line with 3 state buffers

- Designate the address register by AR and the data register by DR
- The read operation can be stated as:
 - Read: $DR \leftarrow M[AR]$
- The write operation can be stated as:
 - Write: $M[AR] \leftarrow R1$



Bus line with three state-buffers.

Tri-State Bus Operation



Bus line with three state-buffers.

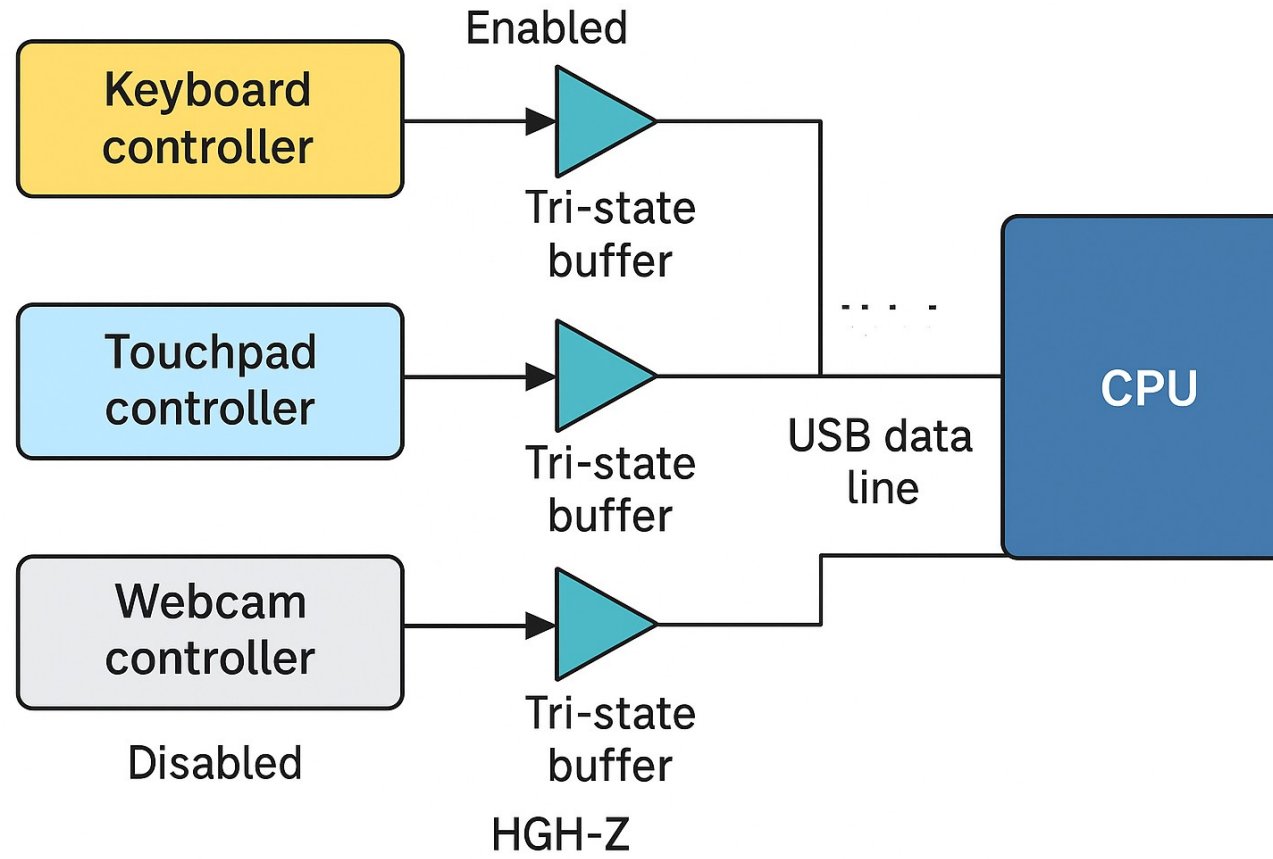
Enable $E = 0$ $S_1 = X$ $S_0 = X$ Active input: None Bus value = Z

Enable = 0: all tri-state buffers are OFF; the bus is high impedance (Z).

Example input values: $A_0=1$, $B_0=0$, $C_0=1$, $D_0=0$. Only one buffer is enabled at a time.

E	S_1S_0	Selected	Bus
0	XX	None	Z
1	00	A_0	A_0
1	01	B_0	B_0
1	10	C_0	C_0

Real Scenario for the Tri-state Buffer



Quiz Time

- **1. Why do USB devices like keyboard, touchpad, and webcam need tri-state buffers when sharing one data line?**
- A. To increase their power consumption
- B. To ensure only one device sends data at a time
- C. To make the USB cable longer
- D. To convert digital data to analog signals

- **1. Why do USB devices like keyboard, touchpad, and webcam need tri-state buffers when sharing one data line?**
- A. To increase their power consumption
- B. To ensure only one device sends data at a time
- C. To make the USB cable longer
- D. To convert digital data to analog signals

- **2. What happens when a device's tri-state buffer is in Hi-Z (high-impedance) mode?**
- A. It forces a constant 1 onto the bus
- B. It forces a constant 0 onto the bus
- C. It disconnects from the USB data line
- D. It sends random data

- **2. What happens when a device's tri-state buffer is in Hi-Z (high-impedance) mode?**
- A. It forces a constant 1 onto the bus
- B. It forces a constant 0 onto the bus
- C. It disconnects from the USB data line
- D. It sends random data

- **3. In the scenario, the keyboard is enabled and sending data. What must the other devices (touchpad, webcam) do?**
- A. Send data at the same time
- B. Switch OFF their power
- C. Enter Hi-Z state to avoid bus conflict
- D. Increase their clock speed

- **3. In the scenario, the keyboard is enabled and sending data. What must the other devices (touchpad, webcam) do?**
- A. Send data at the same time
- B. Switch OFF their power
- **C. Enter Hi-Z state to avoid bus conflict**
- D. Increase their clock speed

- **4. What problem would occur if two USB devices send data on the same line simultaneously without tri-state control?**
- A. The CPU will ignore both devices
- B. The USB line will become faster
- C. Data collision or short circuit
- D. The devices will automatically synchronize

- **4. What problem would occur if two USB devices send data on the same line simultaneously without tri-state control?**
- A. The CPU will ignore both devices
- B. The USB line will become faster
- C. **Data collision or short circuit**
- D. The devices will automatically synchronize

Arithmetic Micro operations

- There are four categories of the most common micro operations:
- Register transfer: transfer binary information from one register to another
- Arithmetic: perform arithmetic operations on numeric data stored in registers
- Logic: perform bit manipulation operations on non-numeric data stored in registers
- Shift: perform shift operations on data stored in registers
- The basic arithmetic micro operations are addition, subtraction, increment, decrement, and shift