Memory and I/O Interfacing

MCA 1ST YEAR 2ND SEMESTER 2020

Paper: MCA 203

Dr. Utpal Nandi

Dept. of Computer Science

VIDYASAGAR UNIVERSITY

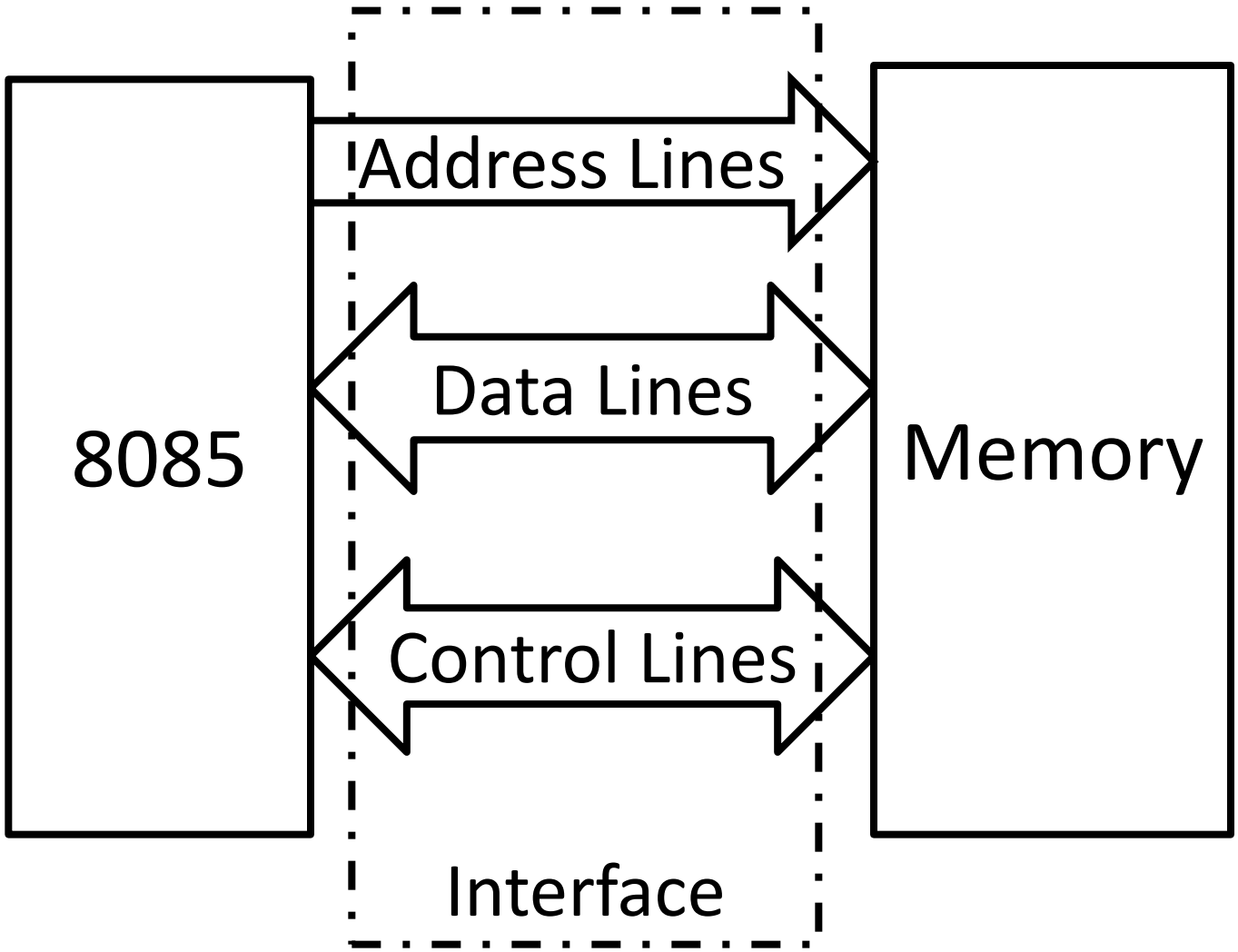
Index

- What is an Interface
- Pins of 8085 used in Interfacing
- Memory – Microprocessor Interface
- I/O – Microprocessor Interface

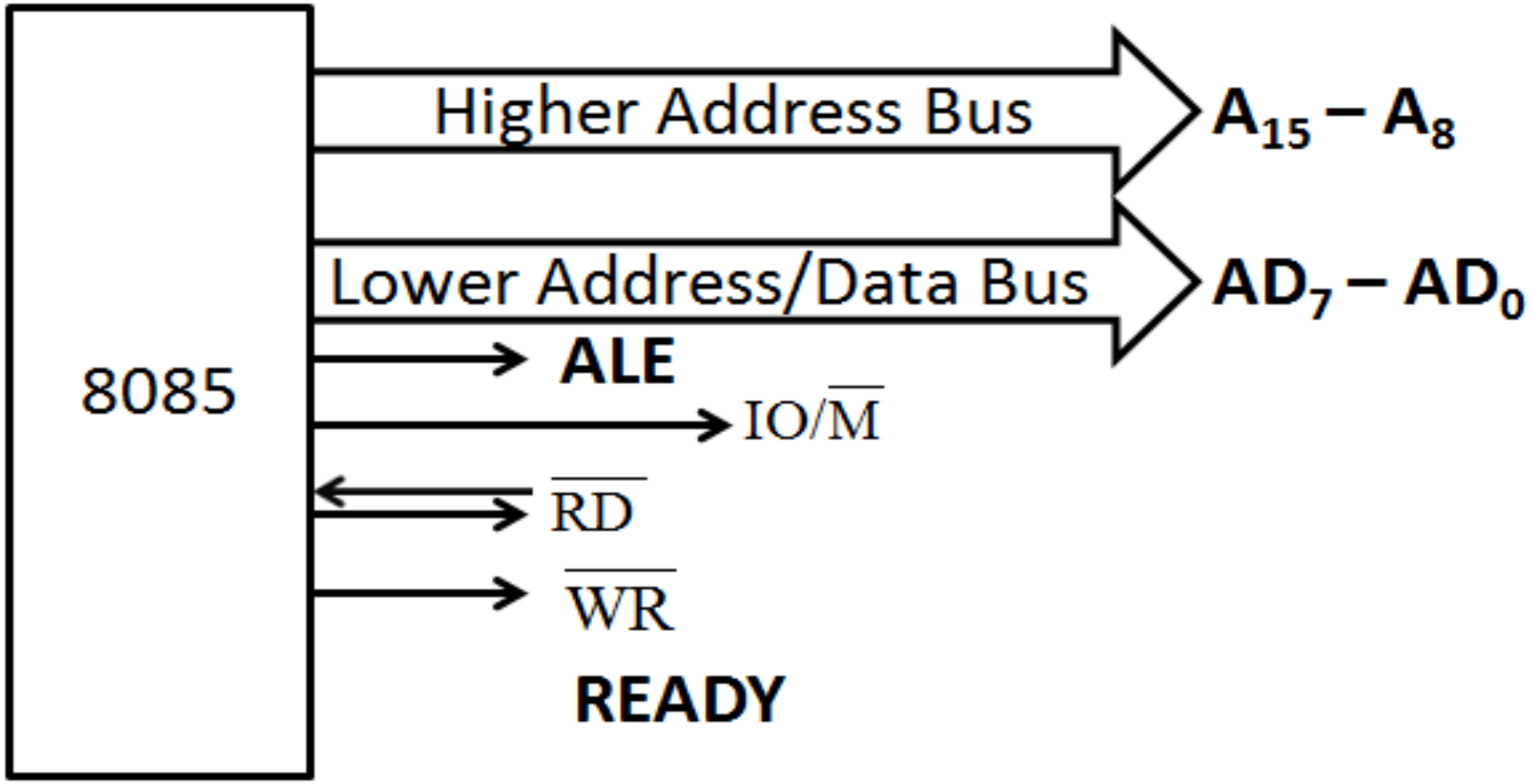
What is an Interface

- An **interface** is a concept that refers to a point of interaction between components, and is applicable at the level of both hardware and software.
- This allows a component, (such as a **graphics card** or an **Internet browser**), to function independently while using interfaces to communicate with other components via an **input/output system** and an **associated protocol**.

Example Block Diagram



8085 Interfacing Pins

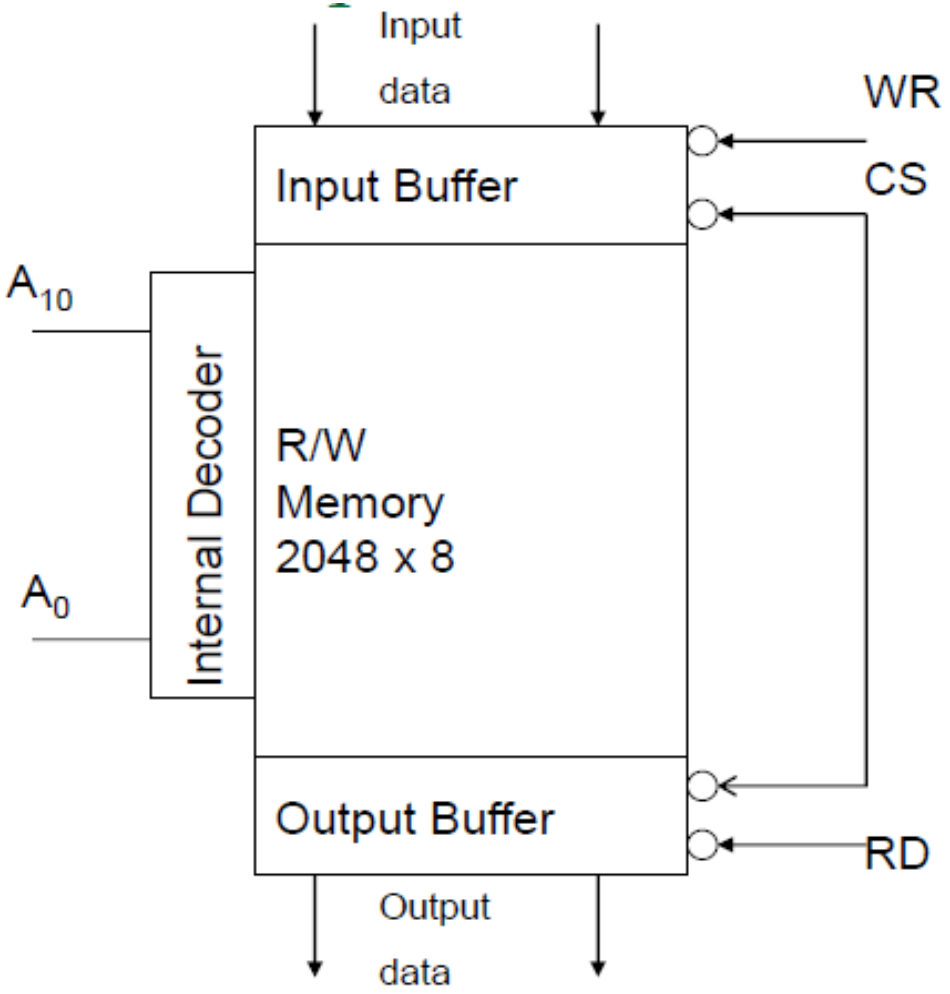


Terminology and Operations

- Memory are made up of (**registers**).
- Each register consists of one storage location
- Each location consists of an **address**
- The number of storage locations from few hundreds to several mega or giga locations
- The total number of memory storage is called memory **capacity** and measured in Bytes
- Each register consists of storage element (FF, capacitor for semiconductor)
- A storage element is called **cell**
- The data could be read from or written to memory

Memory Structure and its requirements

- As mentioned earlier, read/write memories consist of an array of registers, in which each register has unique address
- The size of the memory is N x M as shown below where N is the number of registers and M is the word length, in number of bits[1]



Logic Diagram for RAM

Example

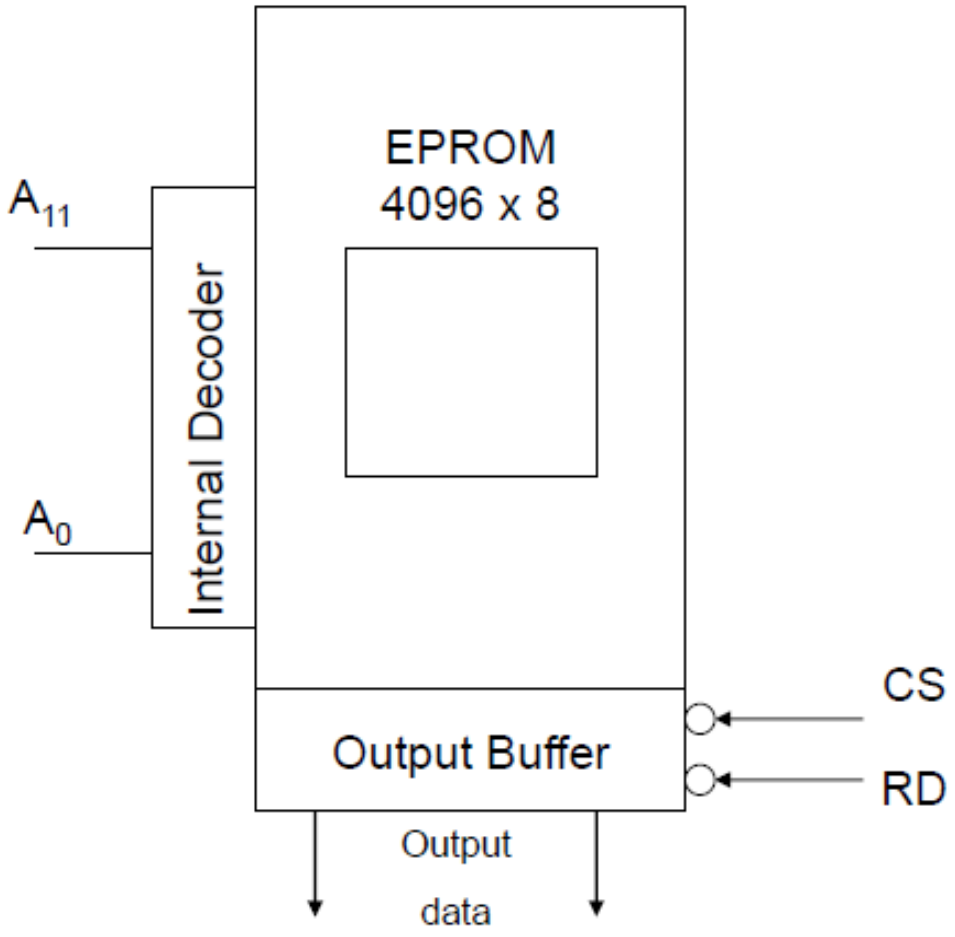
- If memory is having 12 address lines and 8 data lines, then Number of registers/ memory locations (capacity) = 2N= 212
- = 4096
- Word length = M bit
- = 8 bit
- Example 2: if memory has 8192 memory locations, then it has 13 address lines. (How?)

Table summarizes capacity with address

Memory Capacity	Address lines required
1k = 1024 memory locations	10
2k = 2048 memory locations	11
4k = 4096 memory locations	12
8k = 8192 memory locations	13
16k = 16384 memory locations	14
32k = 32768 memory locations	15
64k = 65536 memory locations	16

EPROM layout

- Shows the logic diagram of typical EPROM (Erasable Programmable Read-Only Memory) with 4096 (4k) registers
- It has 12 address lines (A0-A11), one chip select (CS), one Read control signal.
- No WR signal, why?



Basic Memory Interfacing with 8085

- For interfacing memory devices to µp 8085, keep the following points in your mind:
- µp 8085 can access 64KB memory since address bus is 16-bit.[1]
- Generally EPROM (or EPROMs) is used as a program memory and RAM (or RAMs) as data memory.[2]
- The capacity of program memory and data memory depends on the application.
- Is is not always necessary to select 1 EPROM and 1 RAM. We can have multiple EPROMs and multiple RAMs as per the rquirement of application

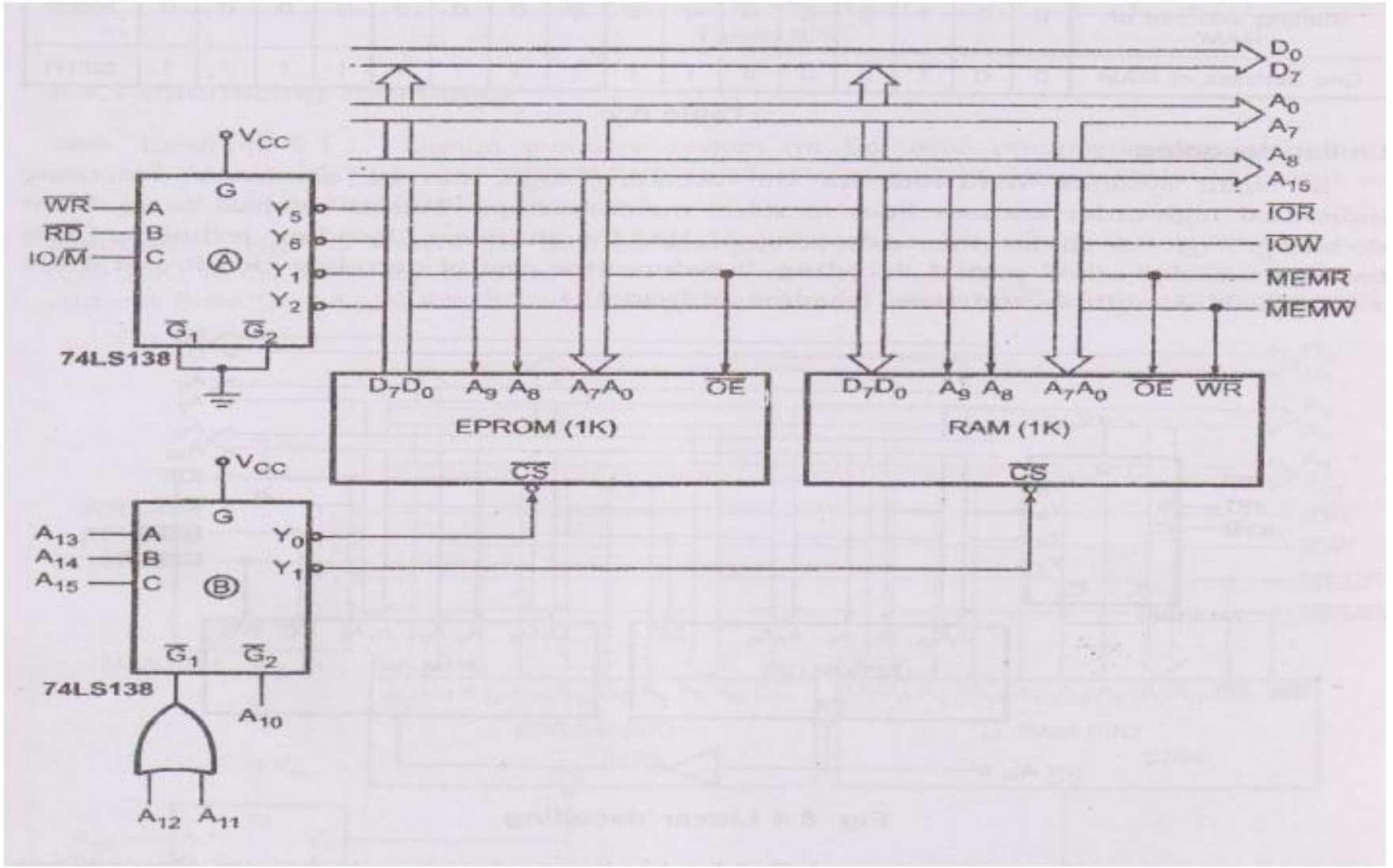
Example

- We have to implement 32 KB of program memory and 4KB EPROMs are available. In this case we can connect 8 EPROMs in parallel.[1]
- We can place EPROM/RAM anywhere in full 64 KB address space. But program memory (EPROM) should be located from address 0000 H.[2]
- It is not always necessary to locate EPROM and RAM in consecutive memory address.[3]
- The memory interfacing requires to:
  - Select the chip
  - Identify the register
  - Enable the appropriate buffer.
- µp system includes memory and I/O devices.
- It is important to note that µp can communicate (read/write) with only one device at a time, so address decoding needed.

Address Decoding techniques

- There are two main techniques:
  - Absolute decoding/ Full Decoding
  - Linear decoding / Partial Decoding
- Absolute Decoding:
- All the higher address lines are decoded to select the memory chip, and the memory chip is selected only for the specified logic level on these high-order address, no other logic levels can select the chip.
- The following figure shows the memory interface with absolute decoding. This addressing technique is normally used in large memory systems.

Absolute Decoding Technique



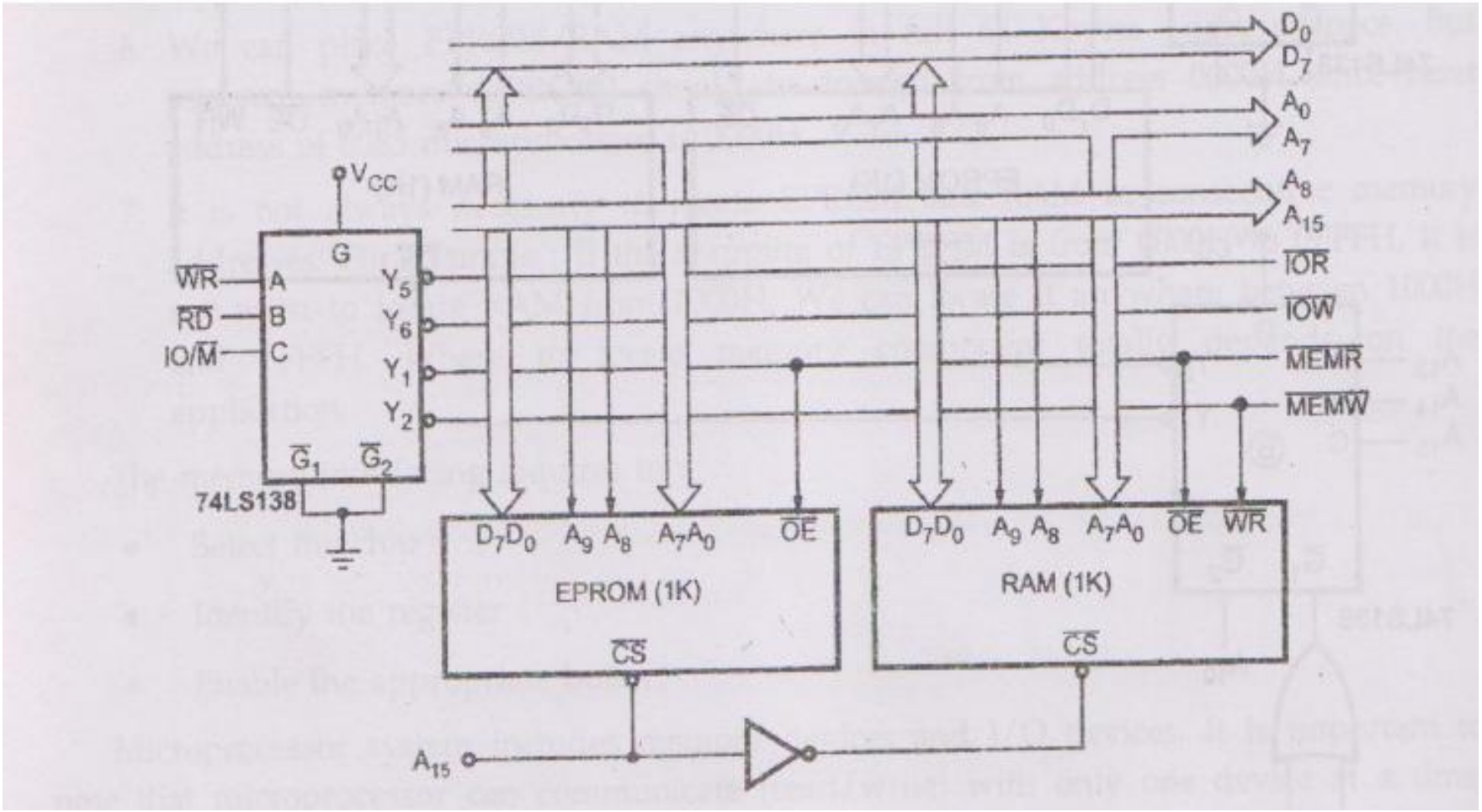
Memory map

Memory ICs	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Address
Starting address of EPROM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
End address of EPROM	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	03FFH
Starting address of RAM	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000H
End address of RAM	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	23FFH

Linear Decoding

- In small systems, h/w for the decoding logic can be eliminated by using individual high-order address lines to select memory chips.
- This is referred to as linear decoding.
- The figure below shows the addressing of RAM with linear decoding technique.
- This technique is also called partial decoding.
- It reduces the cost of the decoding cct., but it has a drawback of multiple address (shadow addresses)

Linear Decoding



What about memory map?

Memory ICs	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Address
Starting address of EPROM	0	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0000H
End address of EPROM	0	X	X	X	X	X	1	1	1	1	1	1	1	1	1	1	03FFH
Starting address of RAM	1	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	8000H
End address of RAM	1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	1	83FFH

Exhaustive/ Absolute/Full Decoding

- In this type of scheme all the 16 bits of the 8085 address bus are used to select a particular location in memory chip.
- Advantages:
  - Complete Address Utilization
  - Ease in Future Expansion
  - No Bus Contention, as all addresses are unique.
- Disadvantages
  - Increased hardware and cost.
  - Speed is less due to increased delay.

Partial /Linear Decoding

- In this scheme minimum number of address lines are used as required to select a memory location in chip.
- Advantages:
  - Simple, Cheap and Fast.
- Disadvantages:
  - Unutilized space & fold back (multiple mapping).
  - Bus Contention.
  - Difficult future expansion.

Interfacing I/O Devices

- Using I/O devices data can be transferred between the microprocessor and the outside world.
- This can be done in groups of 8 bits using the entire data bus. This is called parallel I/O.
- The other method is serial I/O where one bit is transferred at a time using the SID and SOD pins on the Microprocessor.

Types of Parallel Interface

- There are two ways to interface 8085 with I/O devices in parallel data transfer mode:
  - Memory Mapped IO
  - IO Mapped IO

Memory Mapped IO

- It considers them like any other memory location.
  - They are assigned a 16-bit address within the address range of the 8085.
  - The exchange of data with these devices follows the transfer of data with memory. The user uses the same instructions used for memory.

IO Mapped IO

- It treats them separately from memory.
  - I/O devices are assigned a “port number” within the 8-bit address range of 00H to FFH.
  - The user in this case would access these devices using the IN and OUT instructions only.

IO mapped IO V/s Memory Mapped IO

Memory Mapped IO

- IO is treated as memory.
- 16-bit addressing.
- More Decoder Hardware.
- Can address 2^{16}=64k locations.
- Less memory is available.

IO Mapped IO

- IO is treated IO.
- 8- bit addressing.
- Less Decoder Hardware.
- Can address 2^{8}=256 locations.
- Whole memory address space is available.

IO mapped IO V/s Memory Mapped IO

- Memory Mapped IO
- Memory Instructions are used.
- Memory control signals are used.
- Arithmetic and logic operations can be performed on data.
- Data transfer b/w register and IO.

- IO Mapped IO
- Special Instructions are used like IN, OUT.
- Special control signals are used.
- Arithmetic and logic operations can not be performed on data.
- Data transfer b/w accumulator and IO.

The interfacing of output devices

- Output devices are usually slow.
- Also, the output is usually expected to continue appearing on the output device for a long period of time.
- Given that the data will only be present on the data lines for a very short period (microseconds), it has to be latched externally.

The interfacing of output devices

- To do this the external latch should be enabled when the port's address is present on the address bus, the IO/M signal is set high and WR is set low.
- The resulting signal would be active when the output device is being accessed by the microprocessor.
- Decoding the address bus (for memory-mapped devices) follows the same techniques discussed in interfacing memory.

Interfacing of Input Devices

- The basic concepts are similar to interfacing of output devices.
- The address lines are decoded to generate a signal that is active when the particular port is being accessed.
- An IORD signal is generated by combining the IO/M and the RD signals from the microprocessor.

Interfacing of Input Devices

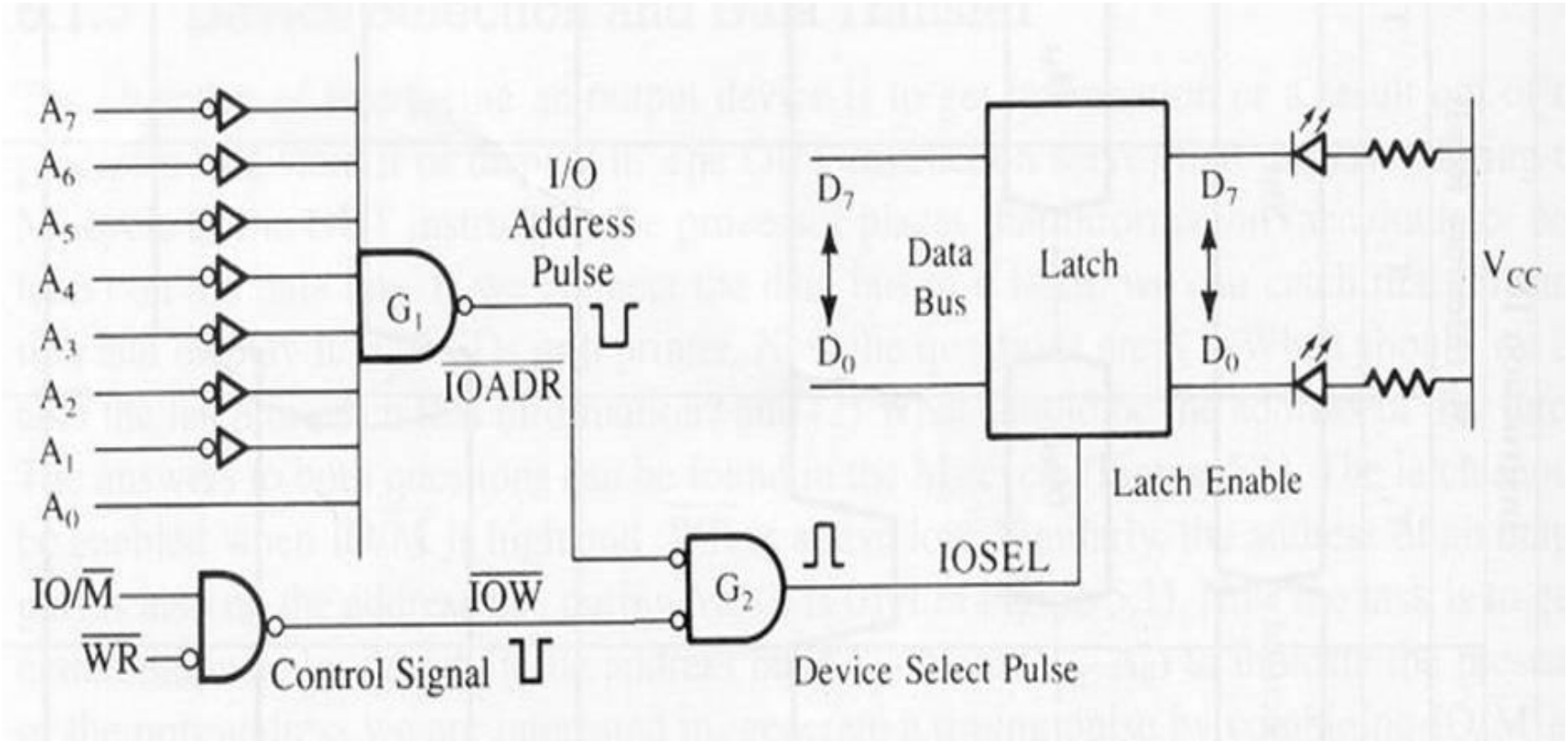
- A tri-state buffer is used to connect the input device to the data bus.
- The control (Enable) for these buffers is connected to the result of combining the address signal and the signal IORD.

I/O Peripherals Interface

- The objective of interfacing I/O peripherals:
  - is to obtain information or results from process.
  - to store, process or display.
- The instructions IN and OUT perform this operation.
- The following examples shows the process of instruction:

```
2050      D3     OUT   01H
2051      01
```

I/O Peripherals Interface



Note: In IO interfacing, only one segment of the address bus (low or high addresses) is sufficient (both segment have same address).

I/O Peripherals Interface

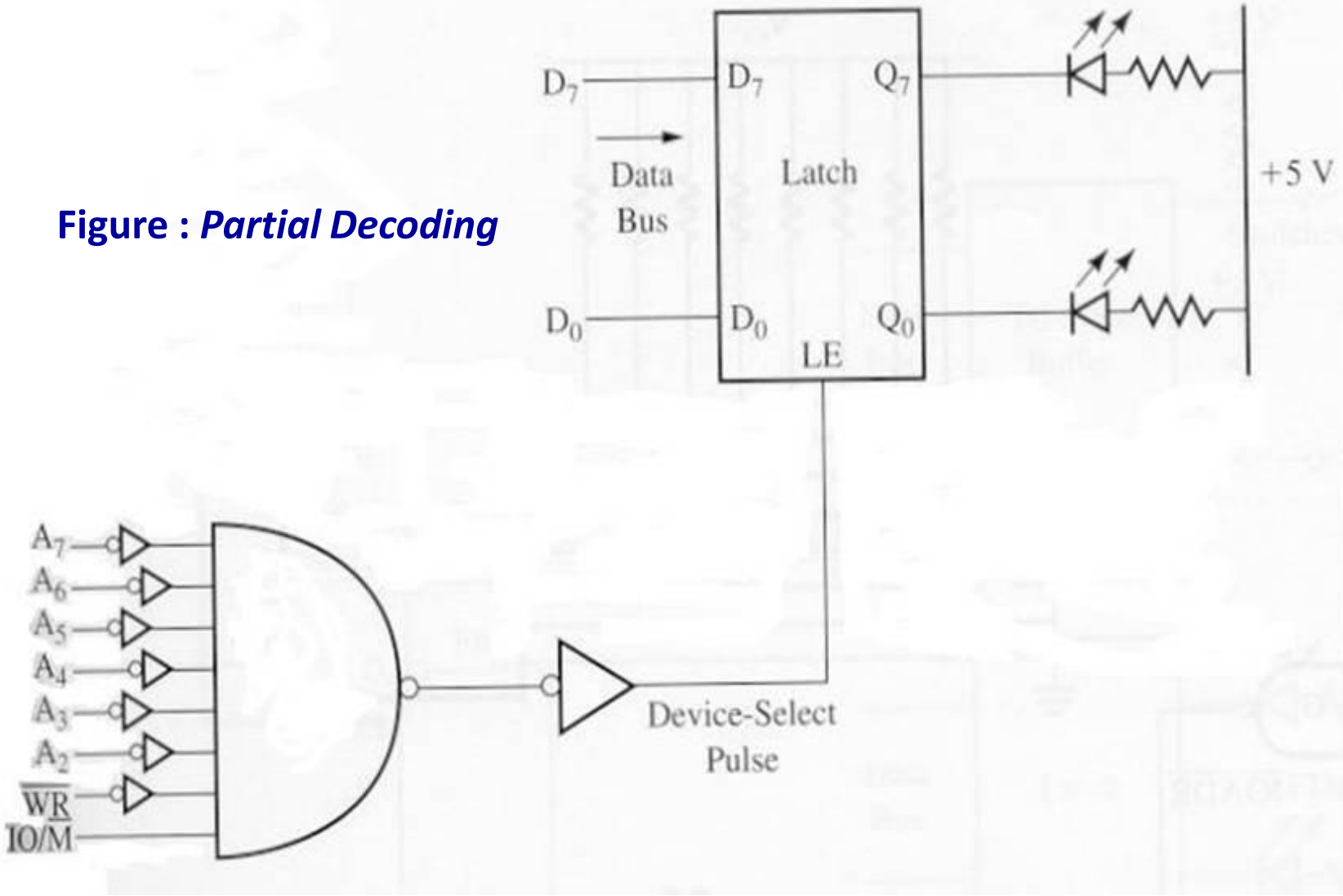
- Figure 2 shows one of the way to decode address lines to obtain output address 01H.
- The line address A_7-A_0 is connected to eight NAND gates function as decoder.
- The line A_0 is connected directly, A_7-A_1 are connected through inverter gates.
- The G_2 gate is combined with gate G_1 and IO/$\overline{W}$ control signal to generate select signal I/O whenever both signals are active low.

Multiple output address, I/O interface

- Figure 2 have shown the technique to decode I/O output address in *absolute decoding* technique.
- There is another technique which is cost saving - partial decoding.
- This technique gives the flexibility to user as to use more than one addresses to one output or input device.

Multiple output address, I/O interface

Figure : ***Partial Decoding***



Multiple output address, I/O interface

- The address lines A_1 and A_0 are unused.
- Depending on the logic input given at address lines A1 and A0;
  - the output addresses: 00h, 01h, 02h or 03h, which refer to the same output device.
- The multiple address is normally used in a small system;
  - OK if those addresses are not being used by any other system, input or output devices.

Input Interface

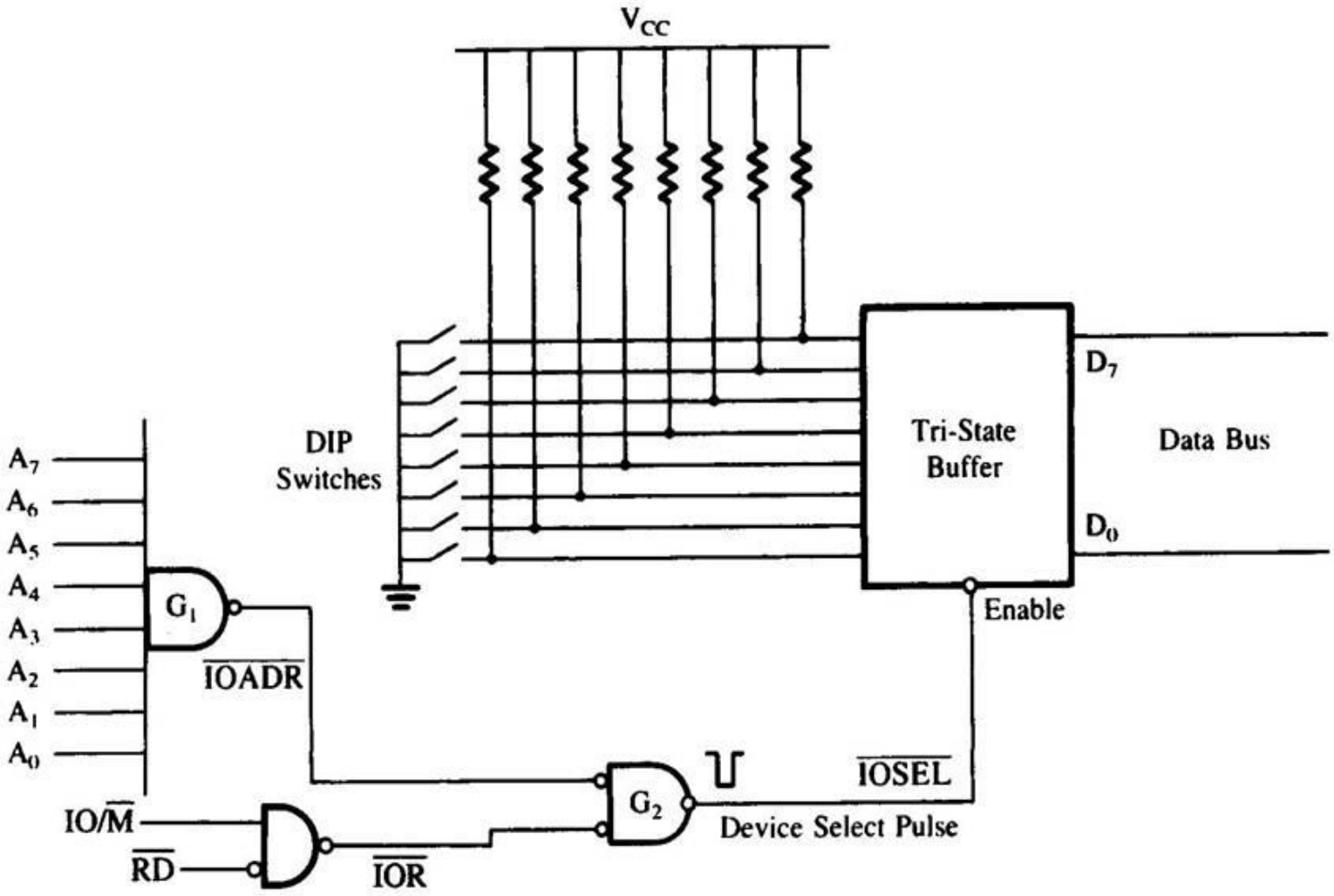
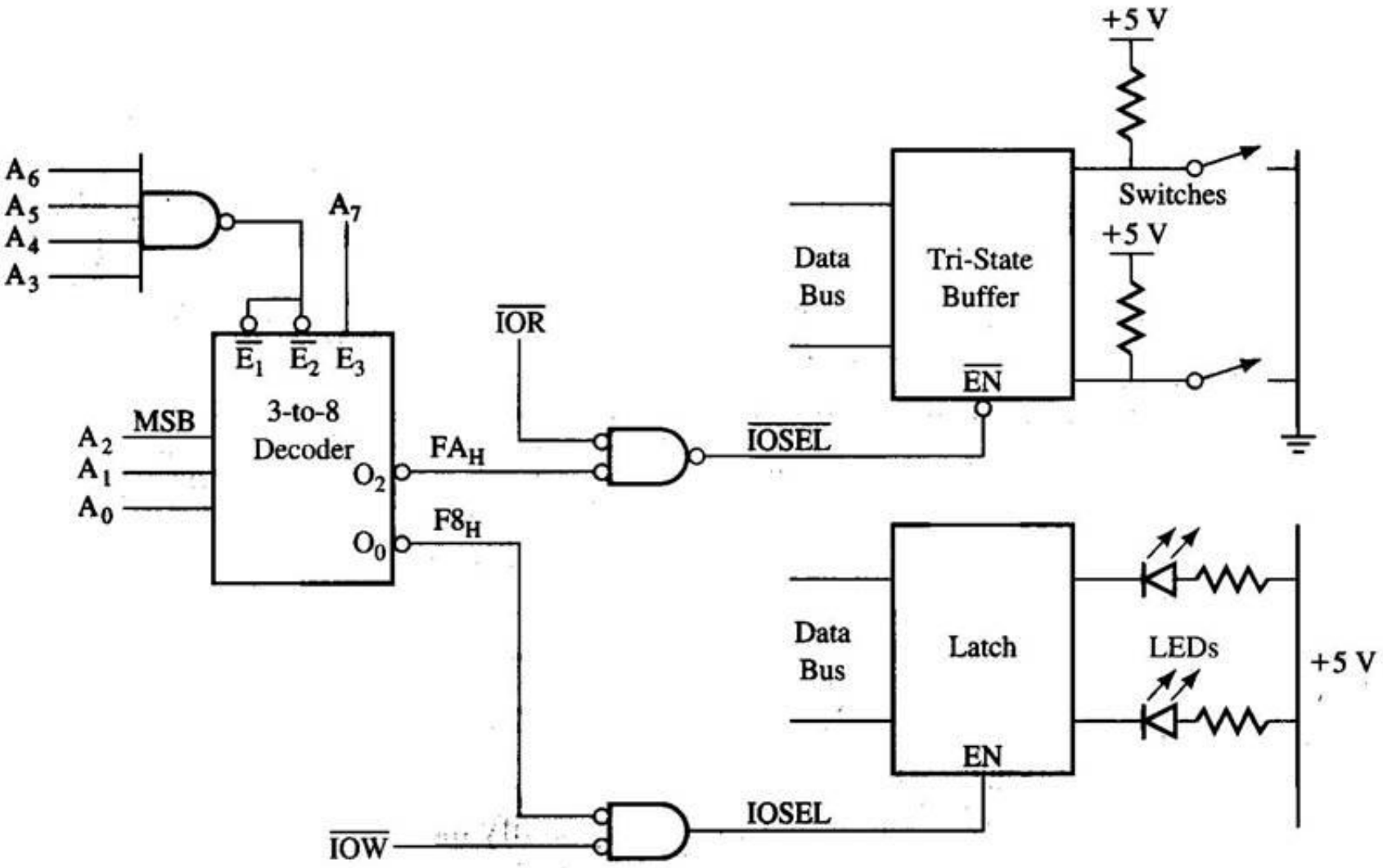


Figure : IN FFH

Input Interface

- The assembly instruction for that circuit in fig. 4 is IN FFH.
- Note: FFH = 1111 1111 binary
- The line address is decoded using NAND gates.
- When address A_7-A_0 is active high (FFH), the output of NAND gate will have an active low signal and then combined with control signals IOR at G_2.
- Suppose the µp run the IN FFH instruction, data at DIP switches will be placed at data bus and copied to accumulator.

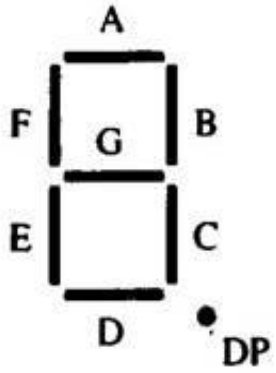
I/O interface using decoder



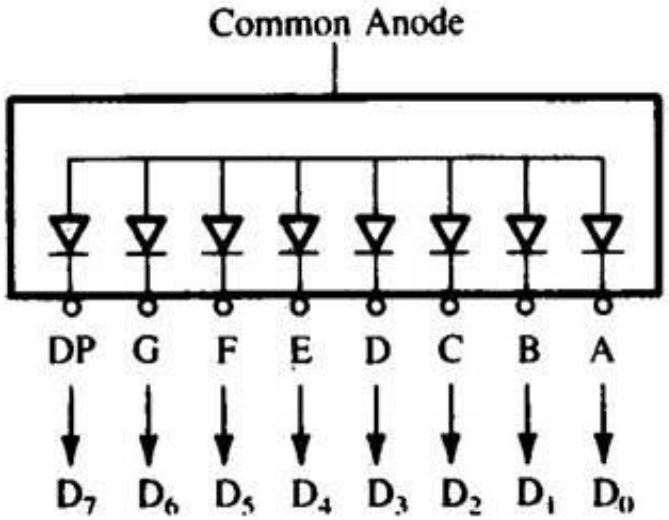
I/O interface using decoder

- Circuit in figure 5 decode input and output device at once using 3-to-8 decoder and four input NAND gates.
- The address lines A_2, A_1 and A_0 are used as inputs to decoder, and the remaining line address A_7 ke A_3 is used to enable the decoder chip.
- The decoder has eight output; therefore we can use the decoder to address **eight** kind of input and output devices.

Seven segment display output interface.



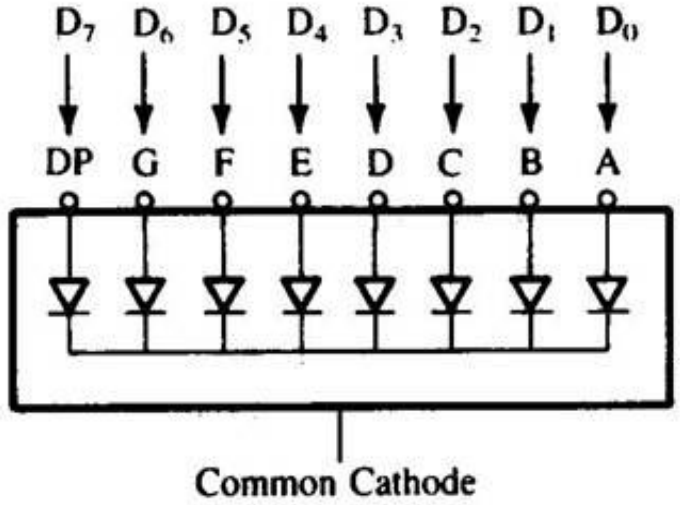
(a)



To Data Lines
Through an Interfacing Device

(b)

From Data Lines
Through an Interfacing Device

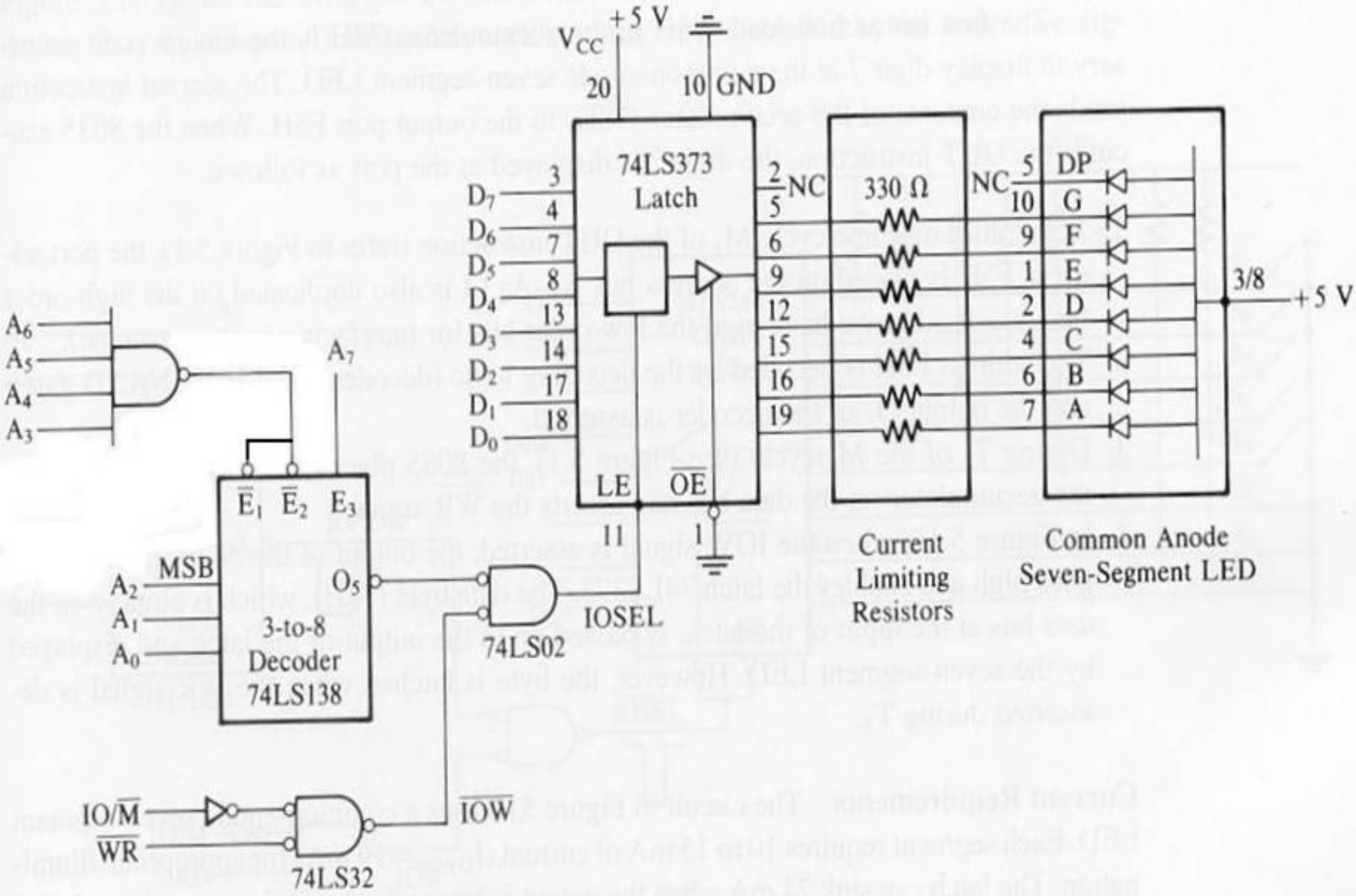


(c)

Seven segment display output interface

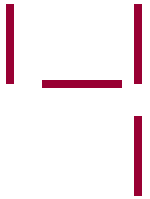
- Fig 6 shows two different type of 7-segment display; *common cathode* and c*ommon anode*.
- 7-segment display consists of a few LEDs and are arranged physically as shown in figure 7a.
- It has seven segment from A to G that normally connected to data bus D_0 to D_6 respectively.
- If decimal point is used, D_7 will be connected to DP; and left unconnected if it is unused.

Seven segment display output interface.



Seven segment display output interface.

- Fig. 7 shows the example to interface seven segment display and address decoder with an address of FDH.
- The common anode display is used therefore 0 logic is needed to activate the segment.
- Suppose to display number 4 at seven segment display, therefore the segment F, G, B and C have to be activated.
- Follows are the instructions to execute it:
  - MVI A, 66H
    OUT FDH



Data lines:	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
Bits:	**X**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	= 66 H
Segments:	NC	G	F	E	D	C	B	A	

Reference Book

"**Microprocessor Architecture, Programming and Applications with 8085**", 5thEdition, Prentice Hall

by

Ramesh S. Goankar

END