MICROPROCESSOR MEMORY & I/O DEVICES

LECTURE 3

THE DESIGN AND OPERATION OF MEMORY

Memory in a microprocessor system is where information (data and instructions) is kept.

It can be classified into two main types:

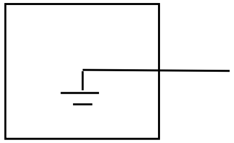
- Main memory (RAM and ROM)
- Storage memory (Disks , CD ROMs, etc.)
- The simple view of RAM is that it is made up of registers that are made up of flip-flops (or memory elements).
- The number of flip-flops in a “memory register” determines the size of the memory word.
- ROM on the other hand uses diodes instead of the flip-flops to permanently hold the information.

ACCESSING INFORMATION IN MEMORY

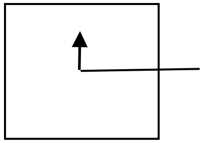
- For the microprocessor to access (Read or Write) information in memory (RAM or ROM), it needs to do the following:
  - Select the right memory chip (using part of the address bus).
  - Identify the memory location (using the rest of the address bus).
  - Access the data (using the data bus).

TRI-STATE BUFFERS

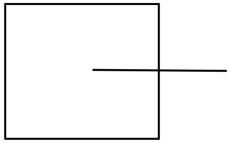
- An important circuit element that is used extensively in memory.
- This buffer is a logic circuit that has three states:
  - Logic 0, logic1, and high impedance.
  - When this circuit is in high impedance mode it looks as if it is disconnected from the output completely.



The Output is Low



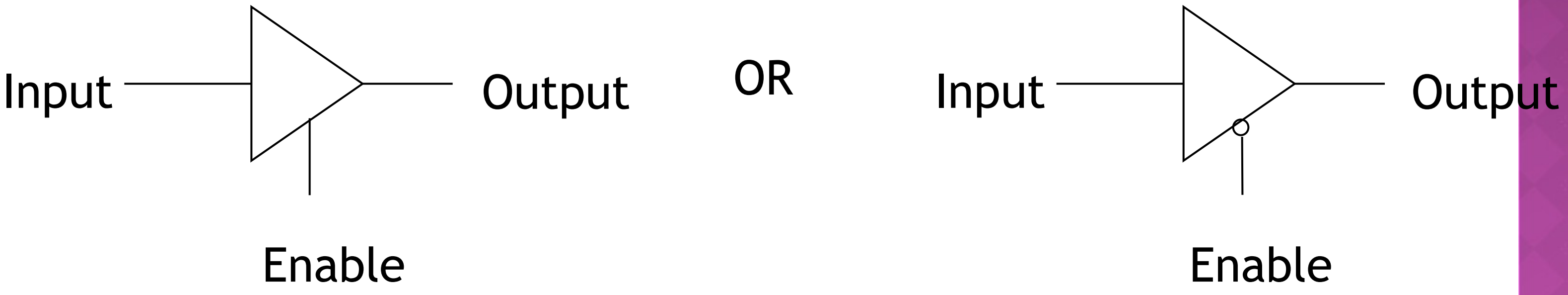
The Output is High



High Impedance

THE TRI-STATE BUFFER

- This circuit has two inputs and one output.
  - The first input behaves like the normal input for the circuit.
  - The second input is an “**enable**”.
    - If it is set high, the output follows the proper circuit behavior.
    - If it is set low, the output looks like a wire connected to nothing.

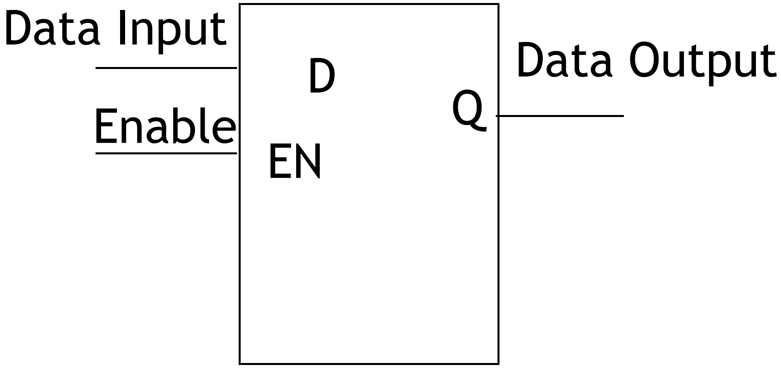


Input Output OR Input Output

Enable Enable

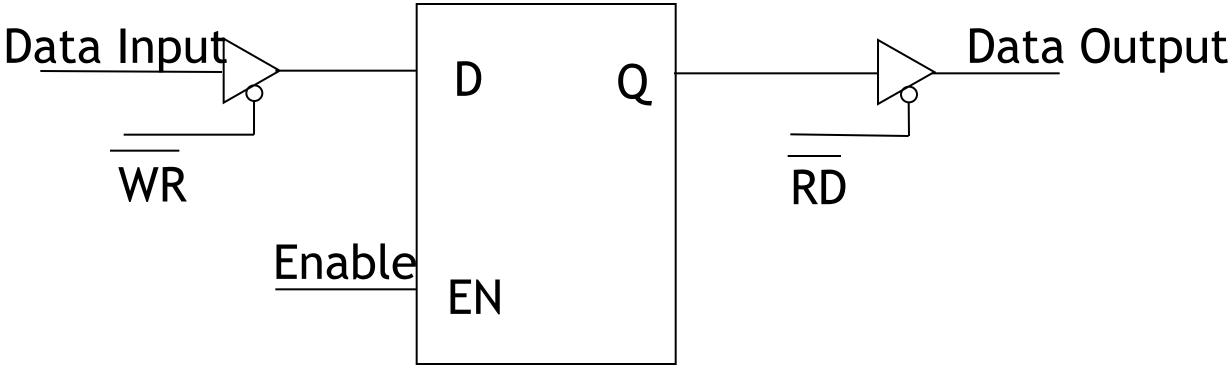
THE BASIC MEMORY ELEMENT

- The basic memory element is similar to a D latch.
- This latch has an input where the data comes in. It has an enable input and an output on which data comes out.



THE BASIC MEMORY ELEMENT

- However, this is not safe.
  - Data is always present on the input and the output is always set to the contents of the latch.
  - To avoid this, tri-state buffers are added at the input and output of the latch.

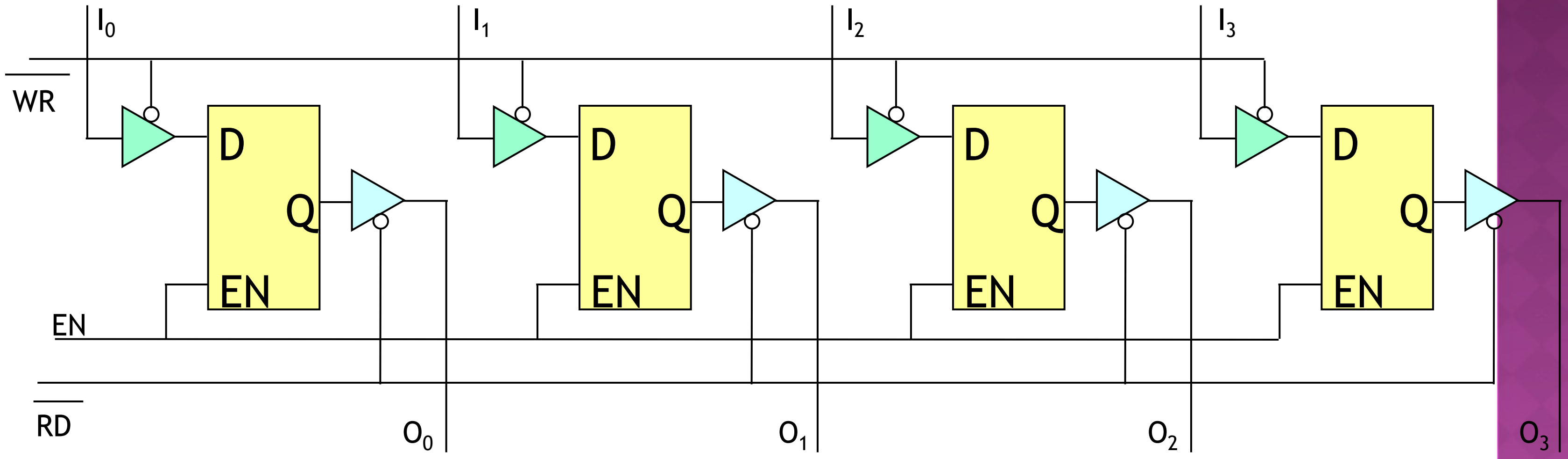


THE BASIC MEMORY ELEMENT

- The WR signal controls the input buffer.
  - The bar over WR means that this is an active low signal.
  - So, if WR is 0 the input data reaches the latch input.
  - If WR is 1 the input of the latch looks like a wire connected to nothing.
- The RD signal controls the output in a similar manner.

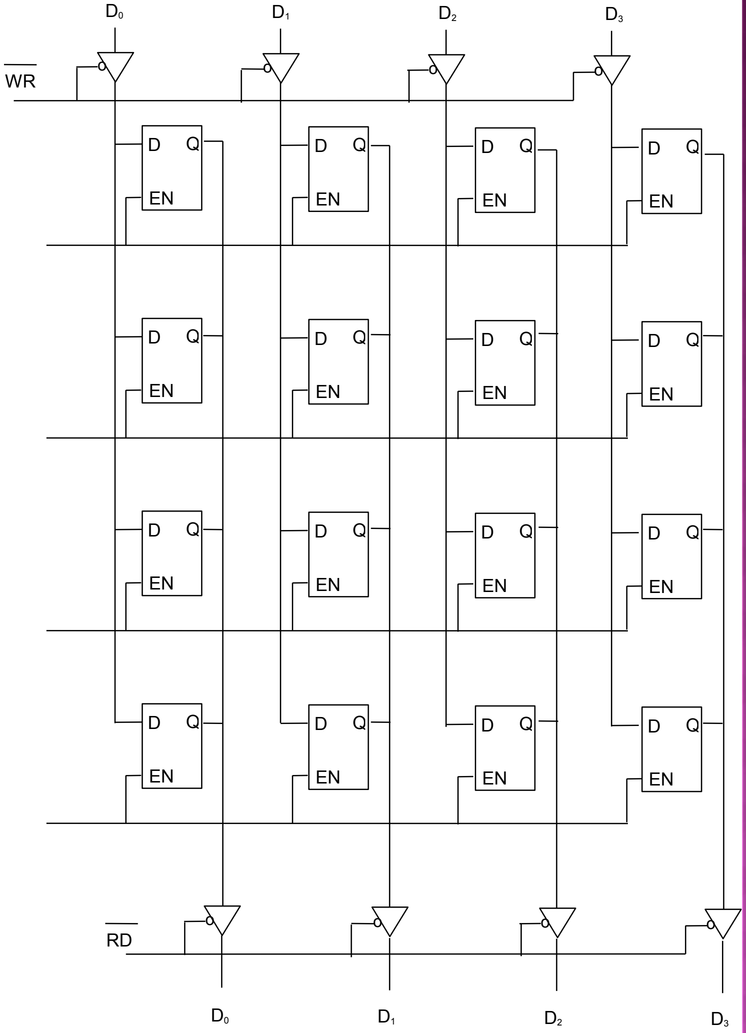
A MEMORY "REGISTER"

- If we take four of these latches and connect them together, we would have a 4-bit memory register



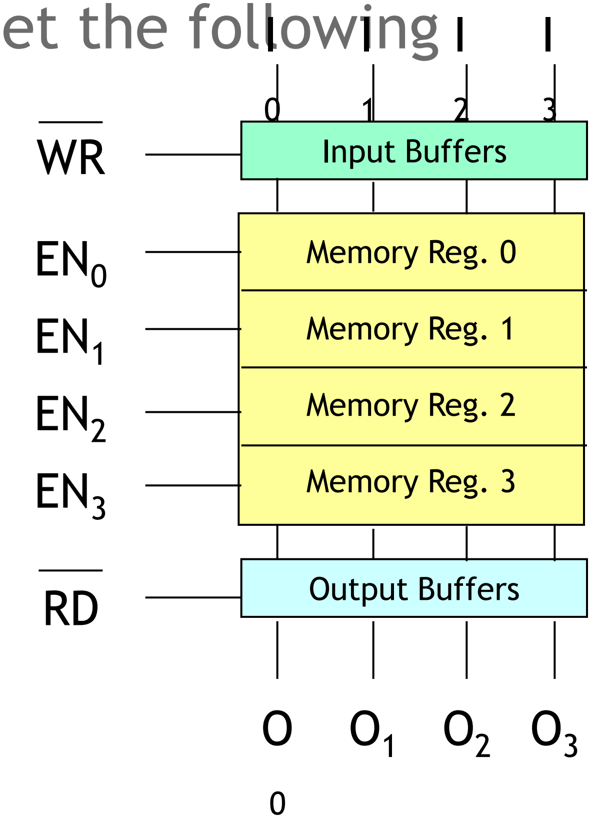
A GROUP OF MEMORY REGISTERS

- Expanding on this scheme to add more memory registers we get the diagram to the right.



A GROUP OF MEMORY REGISTERS

- If we represent each memory location (Register) as a block we get the following



THE DESIGN OF A MEMORY CHIP

- Using the RD and WR controls we can determine the direction of flow either into or out of memory. Then using the appropriate Enable input we enable an individual memory register.

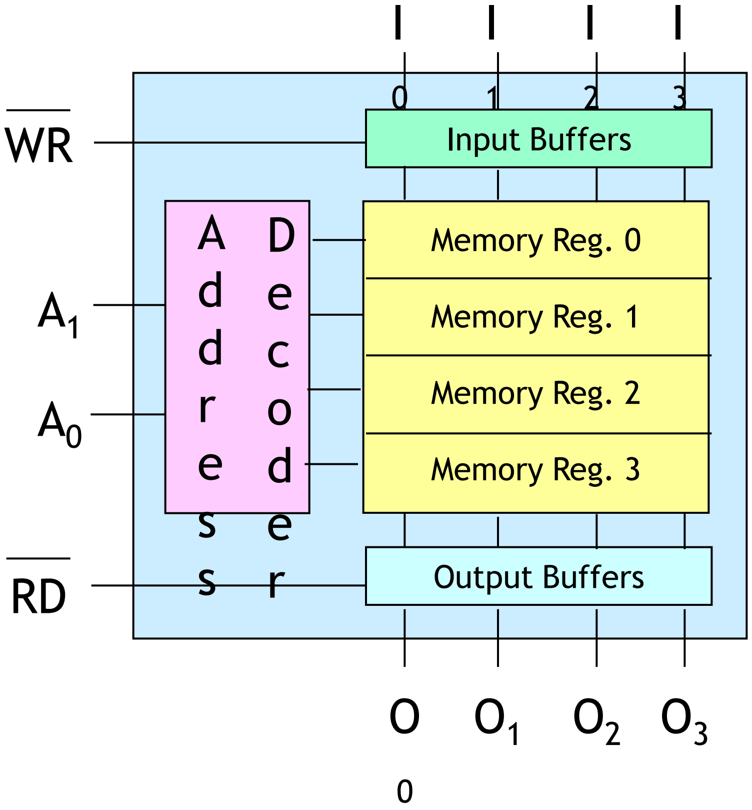
- What we have just designed is a memory with 4 locations and each location has 4 elements (bits). This memory would be called 4 X 4 [Number of location X number of bits per location].

THE ENABLE INPUTS

- How do we produce these enable line?
  - Since we can never have more than one of these enables active at the same time, we can have them encoded to reduce the number of lines coming into the chip.
  - These encoded lines are the address lines for memory.

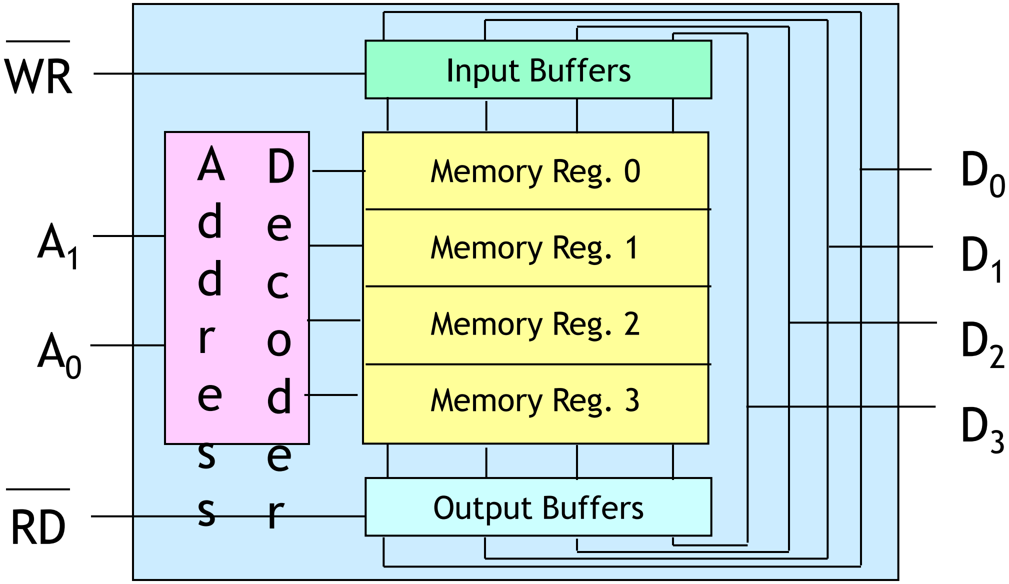
THE DESIGN OF A MEMORY CHIP

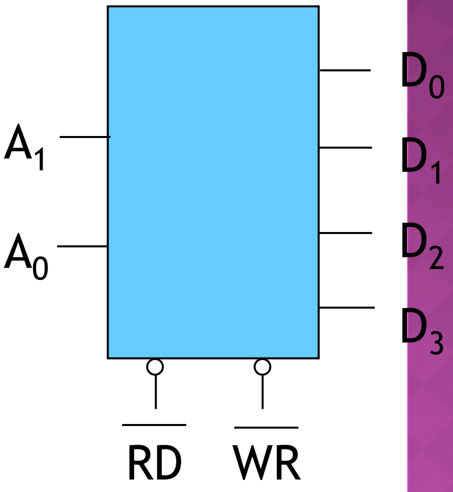
- So, the previous diagram would now look like the following:



THE DESIGN OF A MEMORY CHIP

- Since we have tri-state buffers on both the inputs and outputs of the flip flops, we can actually use one set of pins only.
  - The chip would now look like this:





THE STEPS OF WRITING INTO MEMORY

- What happens when the programmer issues the STA instruction?
  - The microprocessor would turn **on** the WR control (WR = 0) and turn **off** the RD control (RD = 1).
  - The address is applied to the address decoder which generates a **single** Enable signal to turn on **only one** of the memory registers.
  - The data is then applied on the data lines and it is stored into the enabled register.

DIMENSIONS OF MEMORY

- Memory is usually measured by two numbers: its length and its width (Length X Width).
  - The length is the total number of locations.
  - The width is the number of bits in each location.

- The length (total number of locations) is a function of the number of address lines.

  $$\# \text{ of memory locations} = 2^{(\# \text{ of address lines})}$$

  - So, a memory chip with 10 address lines would have

    $$2^{10} = 1024 \text{ locations (1K)}$$

  - Looking at it from the other side, a memory chip with 4K locations would need

    $$\text{Log}_2\ 4096 = 12 \text{ address lines}$$

THE 8085 AND MEMORY

- The 8085 has 16 address lines. That means it can address

  2^{16} = 64K memory locations.

  - Then it will need 1 memory chip with 64 k locations, or 2 chips with 32 K in each, or 4 with 16 K each or 16 of the 4 K chips, etc.

- how would we use these address lines to control the multiple chips?

CHIP SELECT

- Usually, each memory chip has a CS (Chip Select) input. The chip will only work if an active signal is applied on that input.

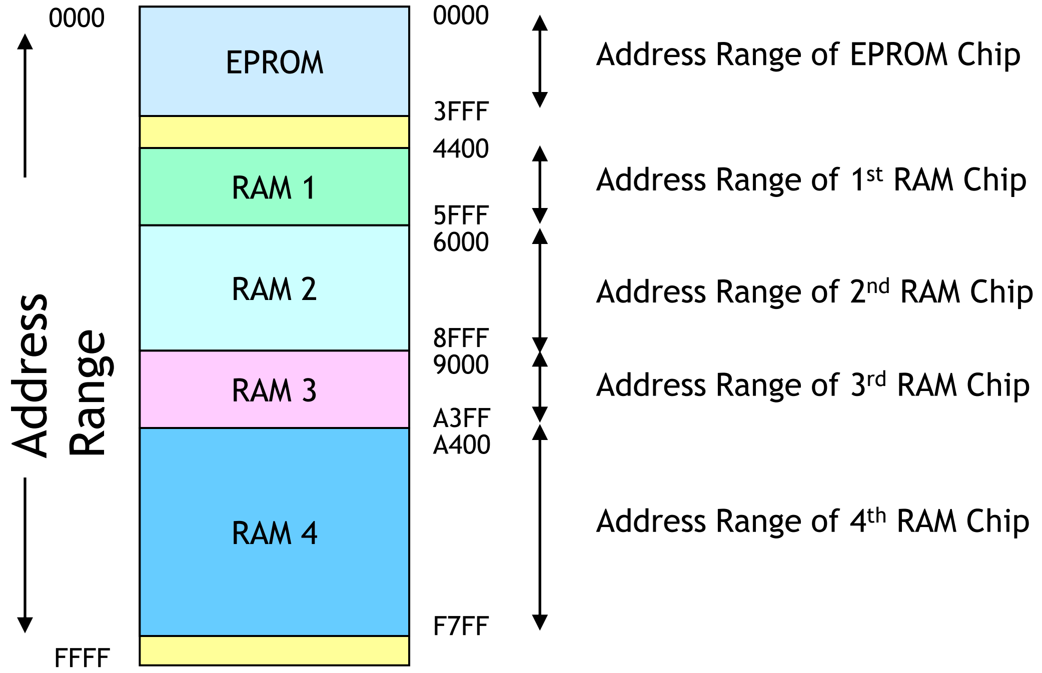
- To allow the use of multiple chips in the make up of memory, we need to use a number of the address lines for the purpose of "chip selection".
  - These address lines are decoded to generate the 2^n necessary CS inputs for the memory chips to be used.

CHIP SELECTION EXAMPLE

- Assume that we need to build a memory system made up of 4 of the 4 X 4 memory chips we designed earlier.
- We will need to use 2 inputs and a decoder to identify which chip will be used at what time.
- The resulting design would now look like the one on the following slide.

MEMORY MAP AND ADDRESSES

- The memory map is a picture representation of the address range and shows where the different memory chips are located within the address range.



0000
Address Range
FFFF
EPROM
RAM 1
RAM 2
RAM 3
RAM 4
0000
3FFF
4400
5FFF
6000
8FFF
9000
A3FF
A400
F7FF
Address Range of EPROM Chip
Address Range of 1st RAM Chip
Address Range of 2nd RAM Chip
Address Range of 3rd RAM Chip
Address Range of 4th RAM Chip

ADDRESS RANGE OF A MEMORY CHIP

- The address range of a particular chip is the list of all addresses that are mapped to the chip.

  - An example for the address range and its relationship to the memory chips would be the Post Office Boxes in the post office.
    - Each box has its unique number that is assigned sequentially. (memory locations)
    - The boxes are grouped into groups. (memory chips)
    - The first box in a group has the number immediately after the last box in the previous group.

INPUT AND OUTPUT DEVICES

Microprocessor need to Identify I/O devices with binary number.

IO devices can be interfaced:

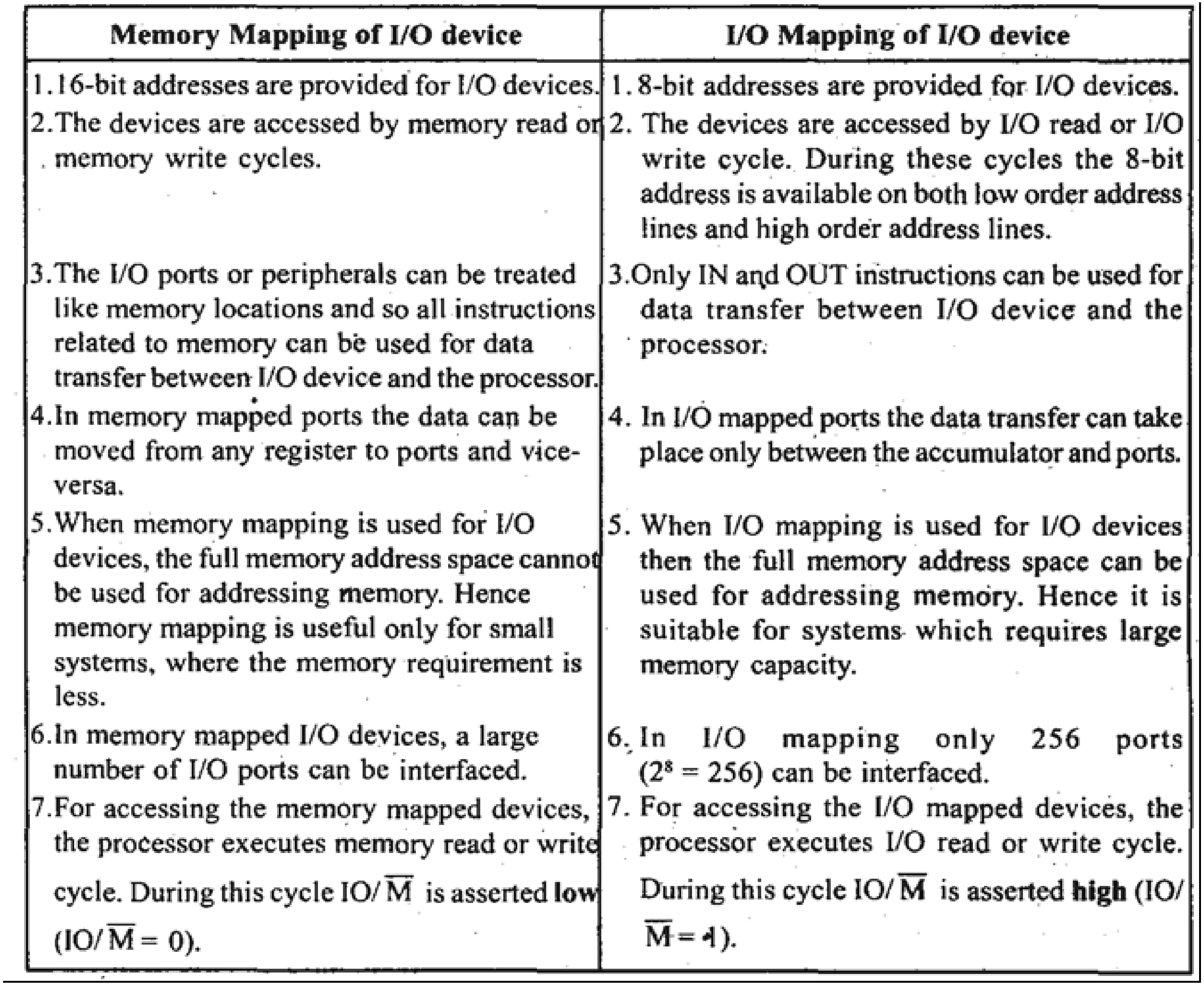
●**Memory-Mapped I/O** (using addresses from memory space)

● Device is identified by 16-bit address (Space ranges from 0000H –FFFFH

●**Standard I/O mapped or isolated I/O mapping /Peripheral Mapped I/O** has separate numbering scheme for I/O devices

● Instructions IN/OUT are used for data transfer

● Device is identified by 8-bit address (Space ranges from 00H –FFH)



Memory Mapping of I/O device	I/O Mapping of I/O device
1.16-bit addresses are provided for I/O devices.	1. 8-bit addresses are provided for I/O devices.
2.The devices are accessed by memory read or memory write cycles.	2. The devices are accessed by I/O read or I/O write cycle. During these cycles the 8-bit address is available on both low order address lines and high order address lines.
3.The I/O ports or peripherals can be treated like memory locations and so all instructions related to memory can be used for data transfer between I/O device and the processor.	3.Only IN and OUT instructions can be used for data transfer between I/O device and the processor.
4.In memory mapped ports the data can be moved from any register to ports and vice-versa.	4. In I/O mapped ports the data transfer can take place only between the accumulator and ports.
5.When memory mapping is used for I/O devices, the full memory address space cannot be used for addressing memory. Hence memory mapping is useful only for small systems, where the memory requirement is less.	5. When I/O mapping is used for I/O devices then the full memory address space can be used for addressing memory. Hence it is suitable for systems which requires large memory capacity.
6.In memory mapped I/O devices, a large number of I/O ports can be interfaced.	6. In I/O mapping only 256 ports ($2^8 = 256$) can be interfaced.
7.For accessing the memory mapped devices, the processor executes memory read or write cycle. During this cycle IO/ $\overline{M}$ is asserted **low** (IO/ $\overline{M}$ = 0).	7. For accessing the I/O mapped devices, the processor executes I/O read or write cycle. During this cycle IO/ $\overline{M}$ is asserted **high** (IO/ $\overline{M}$ = 1).