Microprocessors

Dr Mridul Gupta

Assistant Professor

Dept. of ECE

UNIT-2

Memory

The Design and Operation of Memory

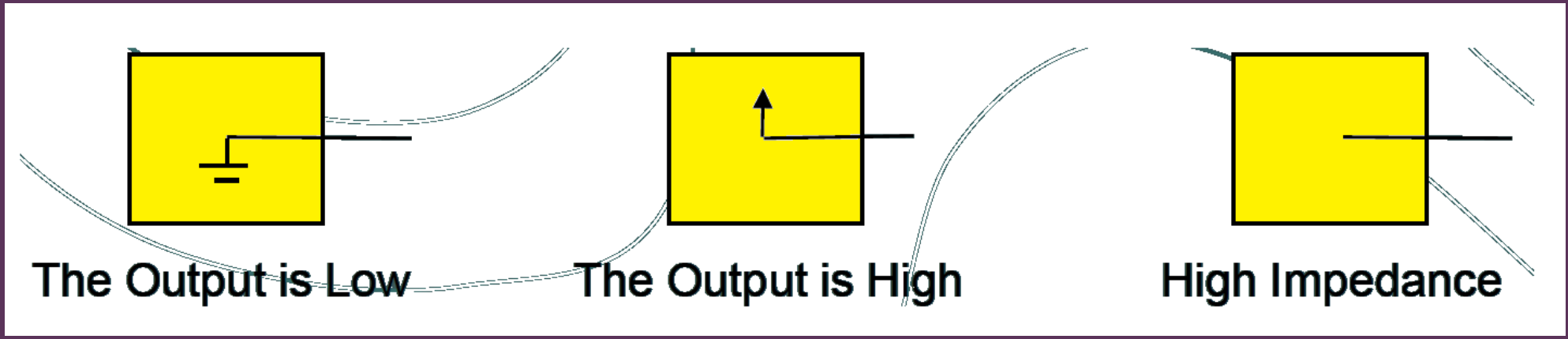
- Memory in a microprocessor system is where information (data and instructions) is kept. It can be classified into two main types:
  - Main memory (RAM and ROM)
  - Storage memory (Disks , CD ROMs, etc.)
- The simple view of RAM is that it is made up of registers that are made up of flip-flops (or memory 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.

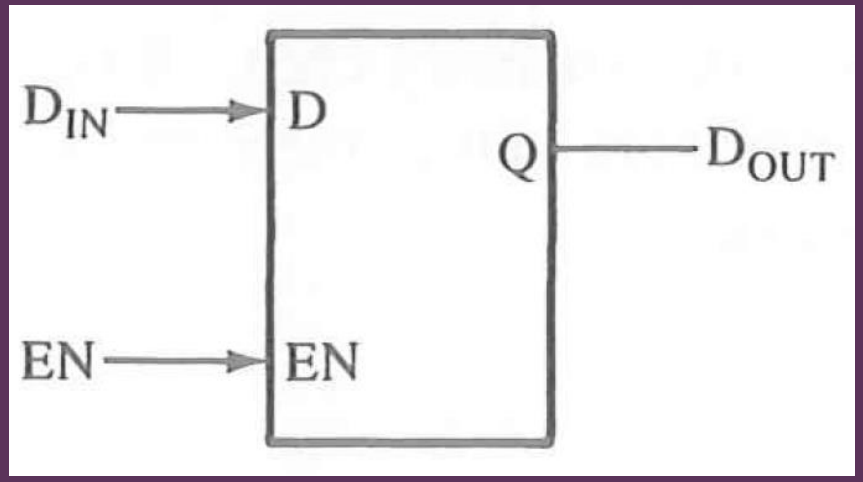


Tri-State Buffers

- 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.

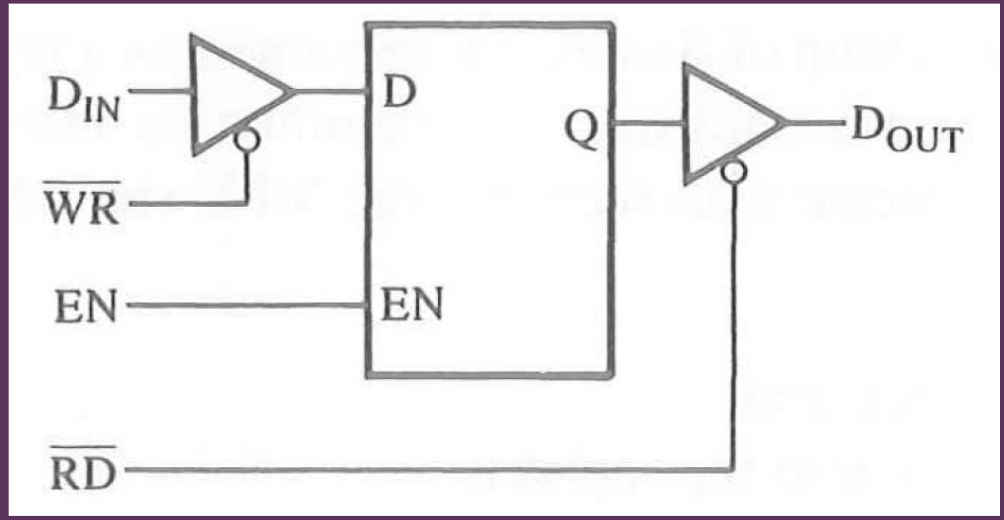
The Basic Memory Element

- 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.

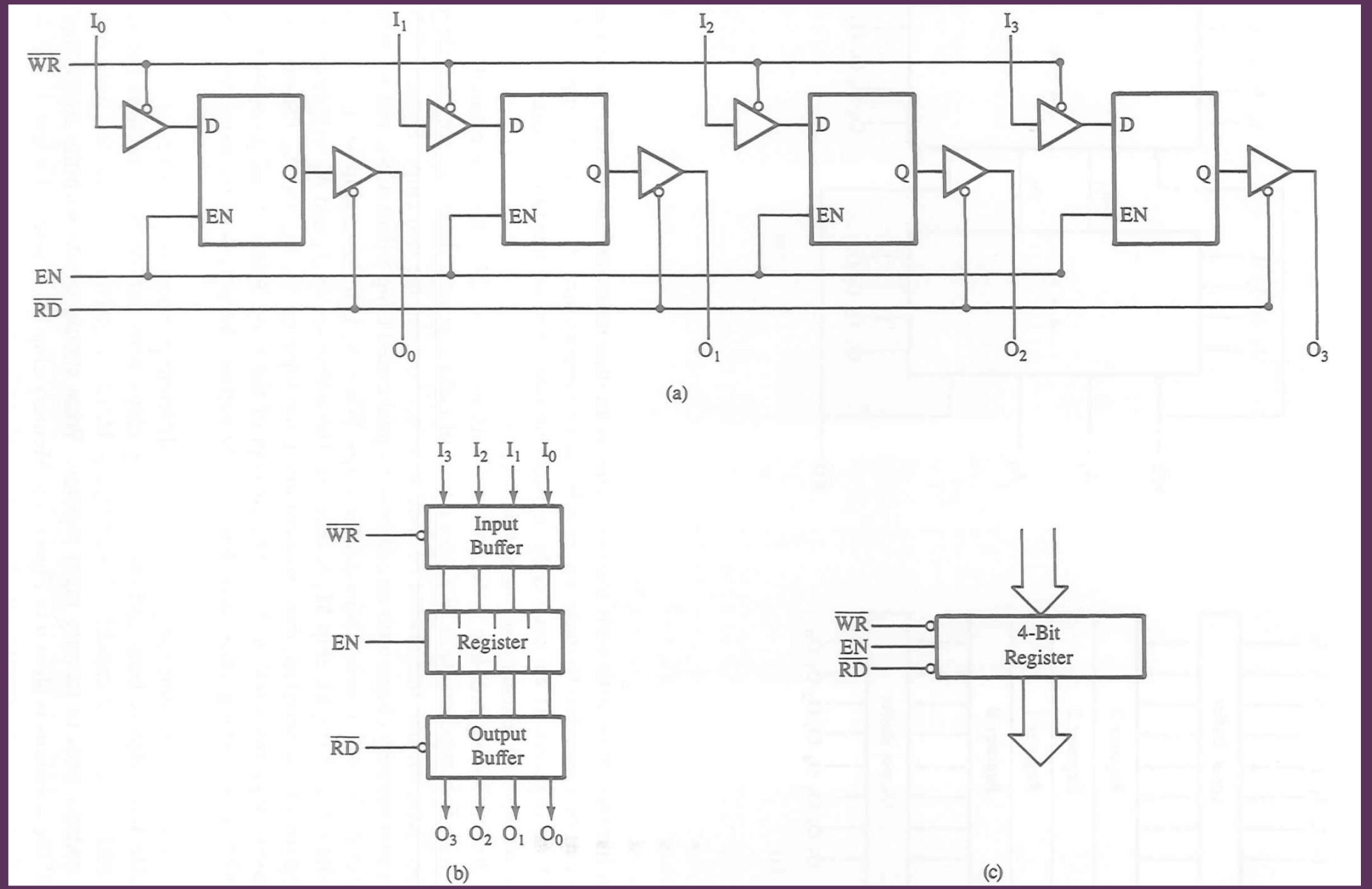


The Basic Memory Element

- 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.

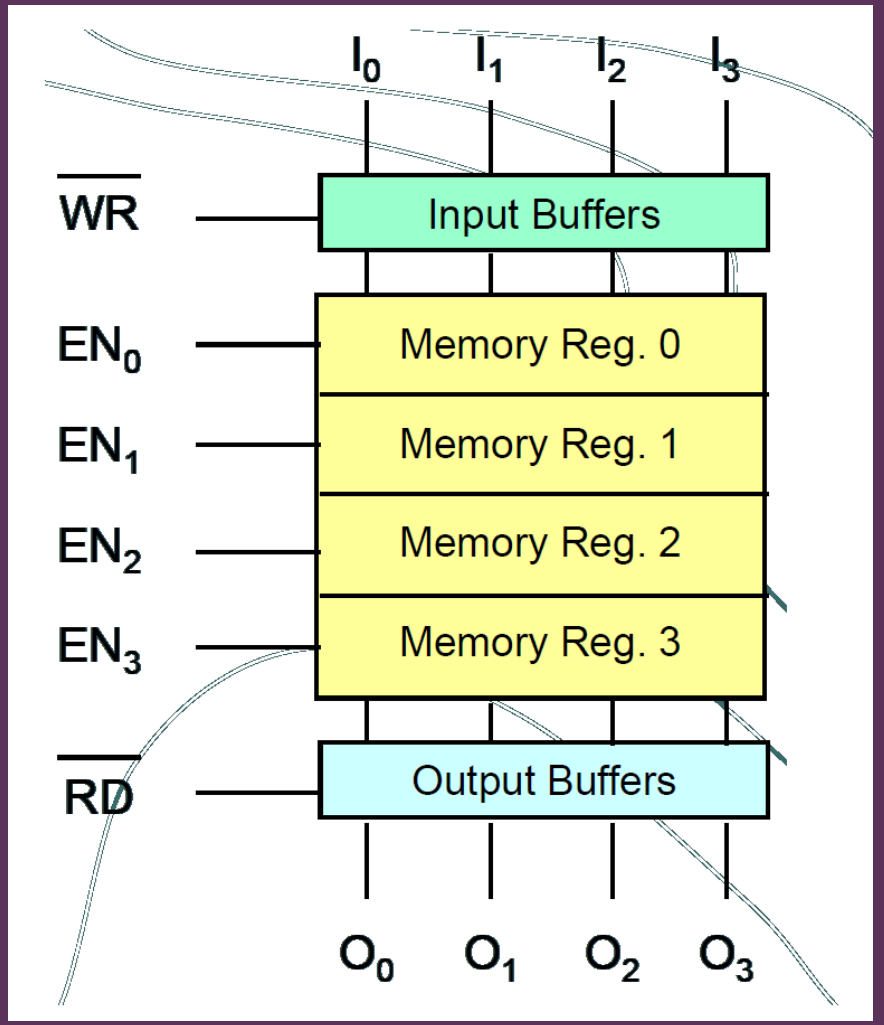
A Memory "Register"

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



A group of Memory Registers

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



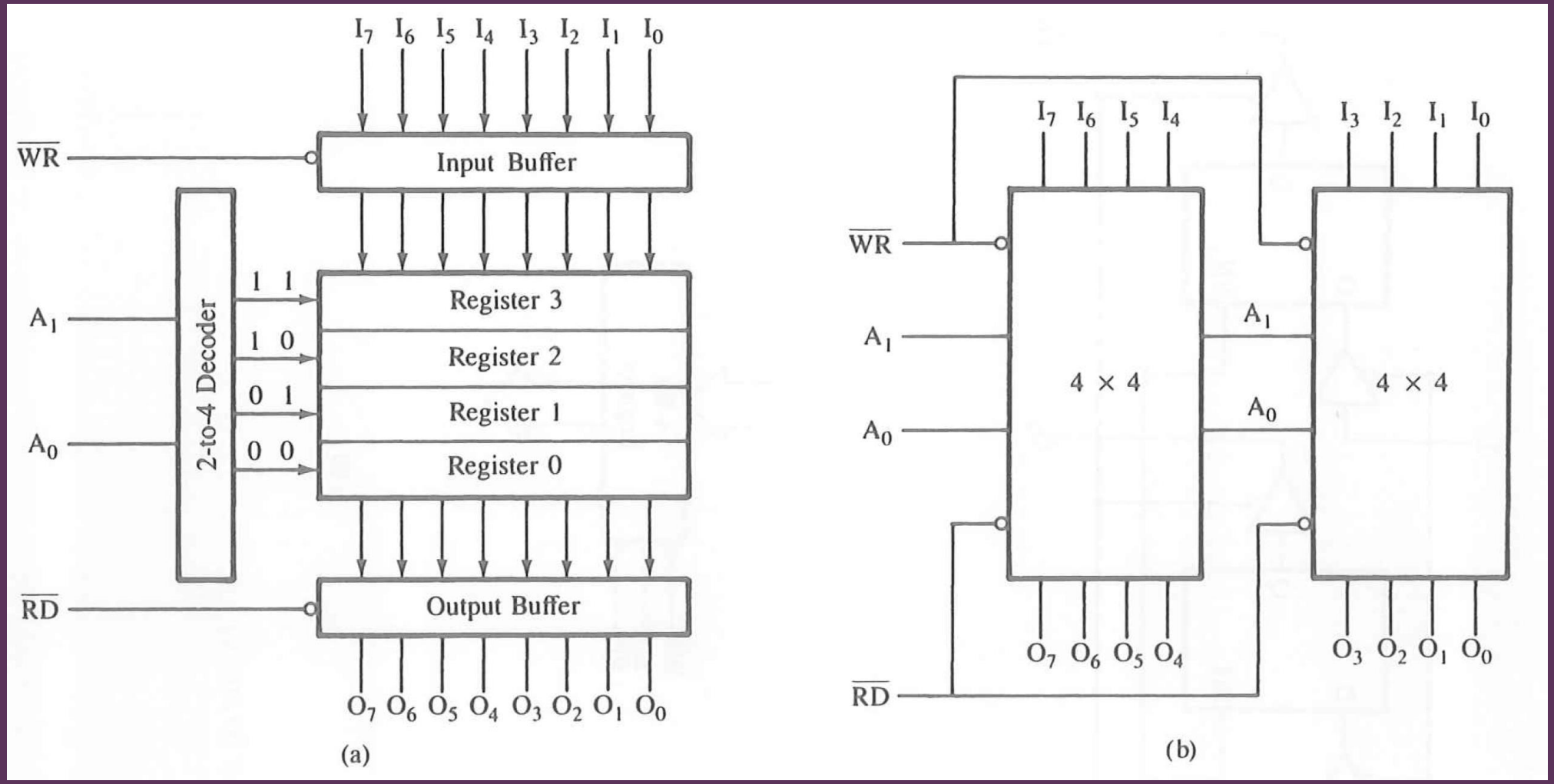
A group of Memory Registers

- 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.

- Let us design **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].**

A group of Memory Registers

(a) 4x8-bit Register; (b) 2 chips of 4-bit Registers



The 8085 and Memory

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

  2^{16} Bytes = 64kB 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 necessary CS inputs for the memory chips to be used**.

Chip Select (Example-1)

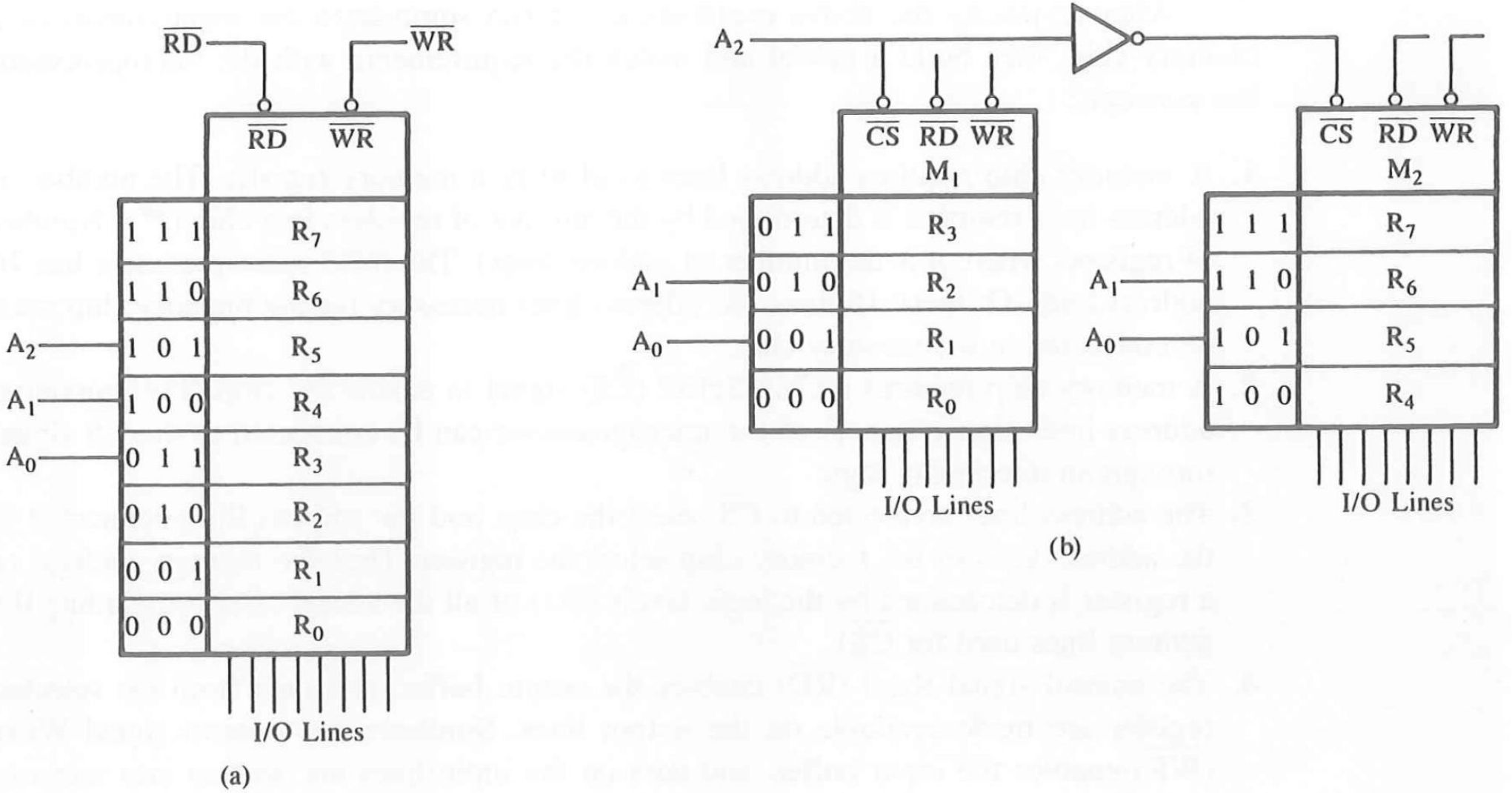


FIGURE 3.7
Two Memory Chips with Four Registers Each and Chip Select

Chip Select (Example-2)

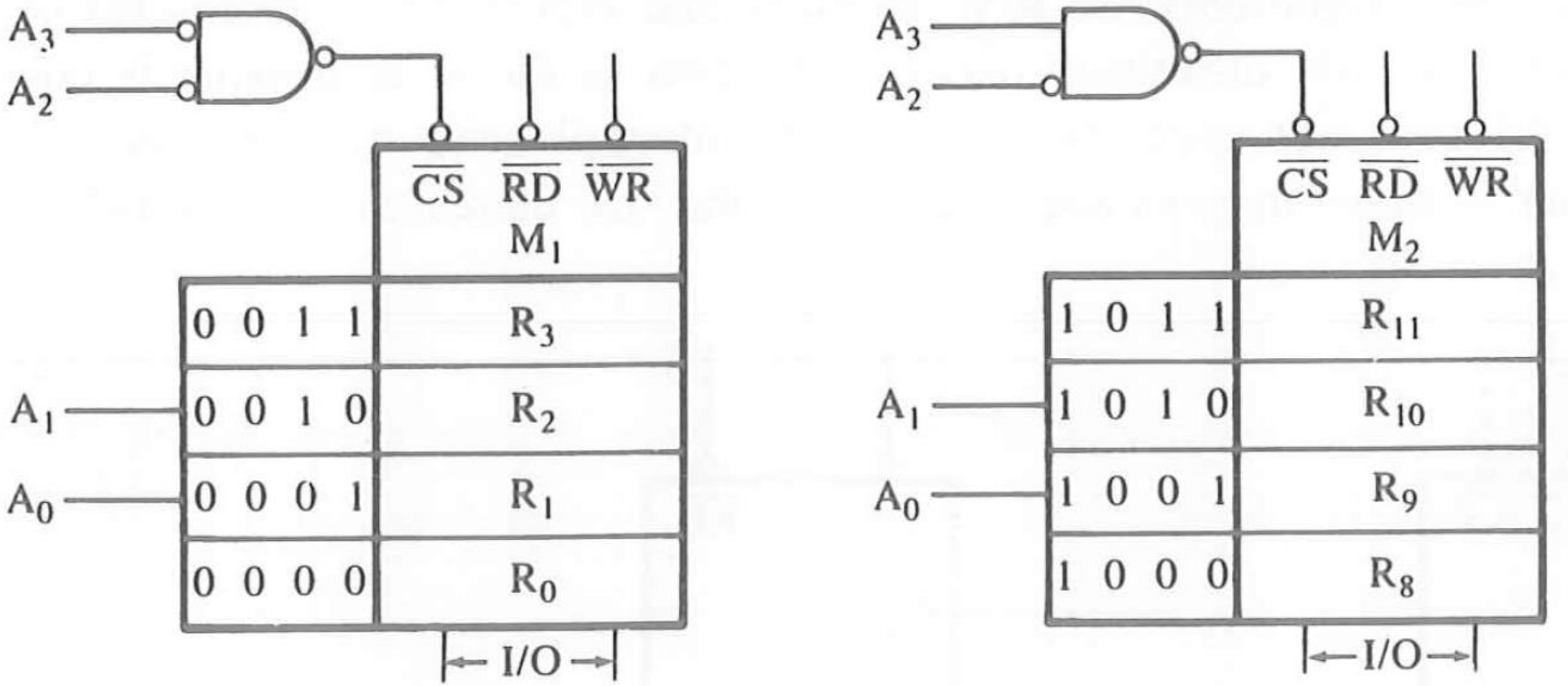


FIGURE 3.8
Addressing Eight Registers with Four Address Lines

A Typical Memory Chip

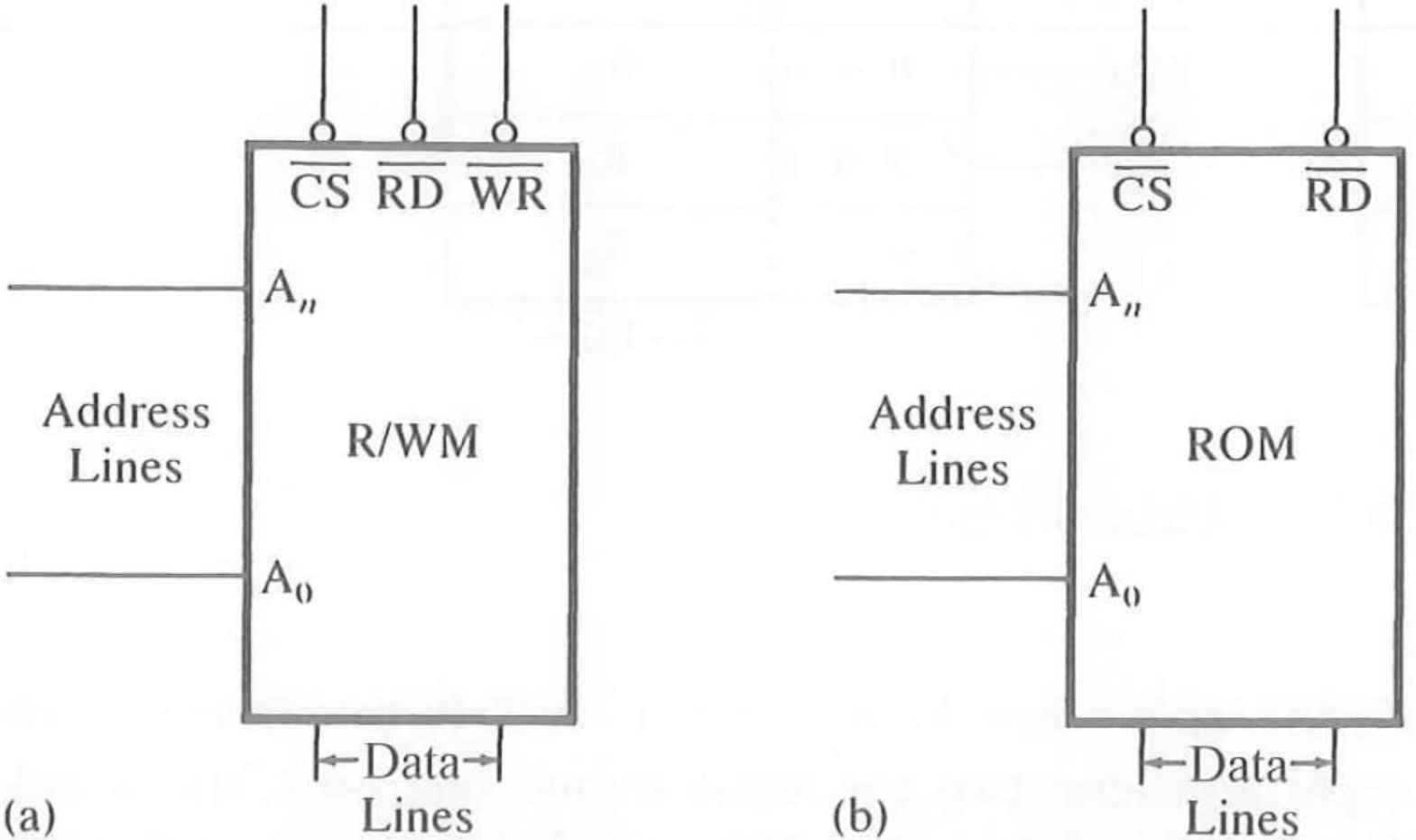
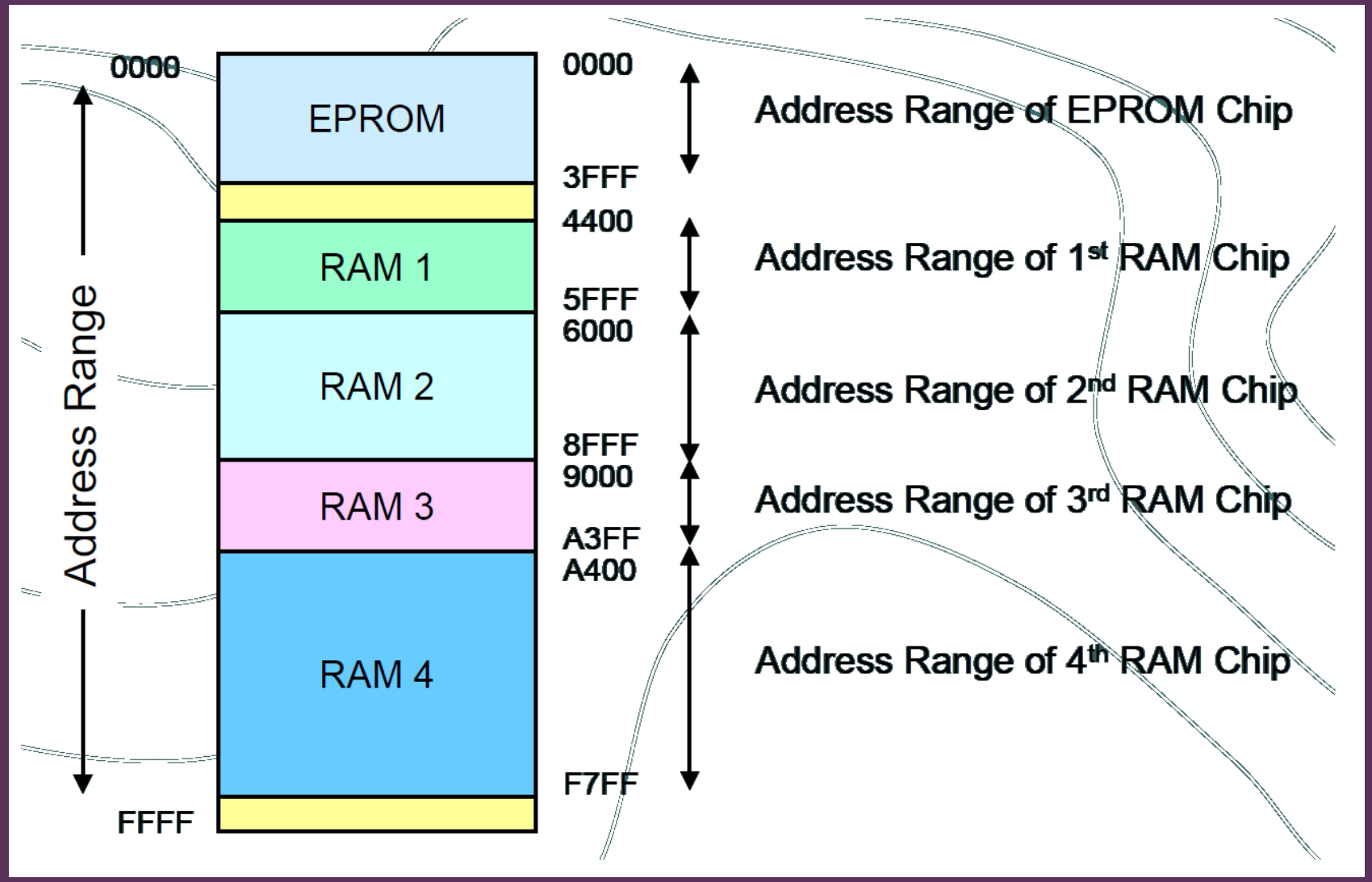


FIGURE 3.9
R/W Memory Model (a) and ROM Model (b)

Memory Map and Addresses

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



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.
- The address lines from a microprocessor can be classified into two types:
  - **High-Order**
    - **Used for memory chip selection**
  - **Low-Order**
    - **Used for location selection** within a memory chip.

Address Range of a Memory Chip

- This classification is highly dependent on the memory system design.

Example 3.1	Illustrate the memory address range of the chip with 256 bytes of memory, shown in Figure 3.10(a), and explain how the range can be changed by modifying the hardware of the Chip Select $\overline{CS}$ line in Figure 3.10(b).

Address Range of a Memory Chip

Figure 3.10(a) will range from 0000H to 00FFH, as shown below.

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	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	= 0000H
Chip Enable or Chip Select								1	1	1	1	1	1	1	1	= 00FFH
								Register Select								

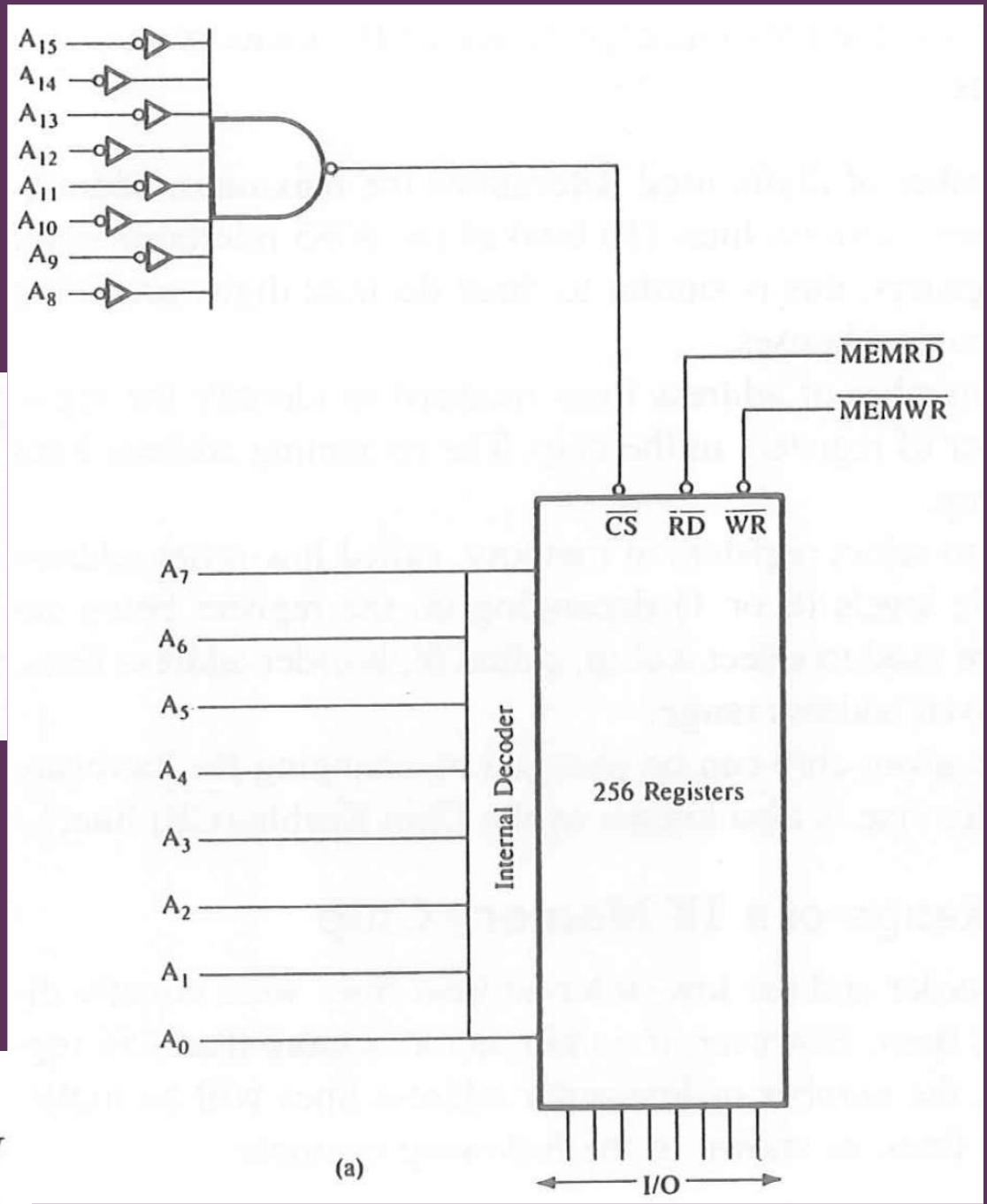
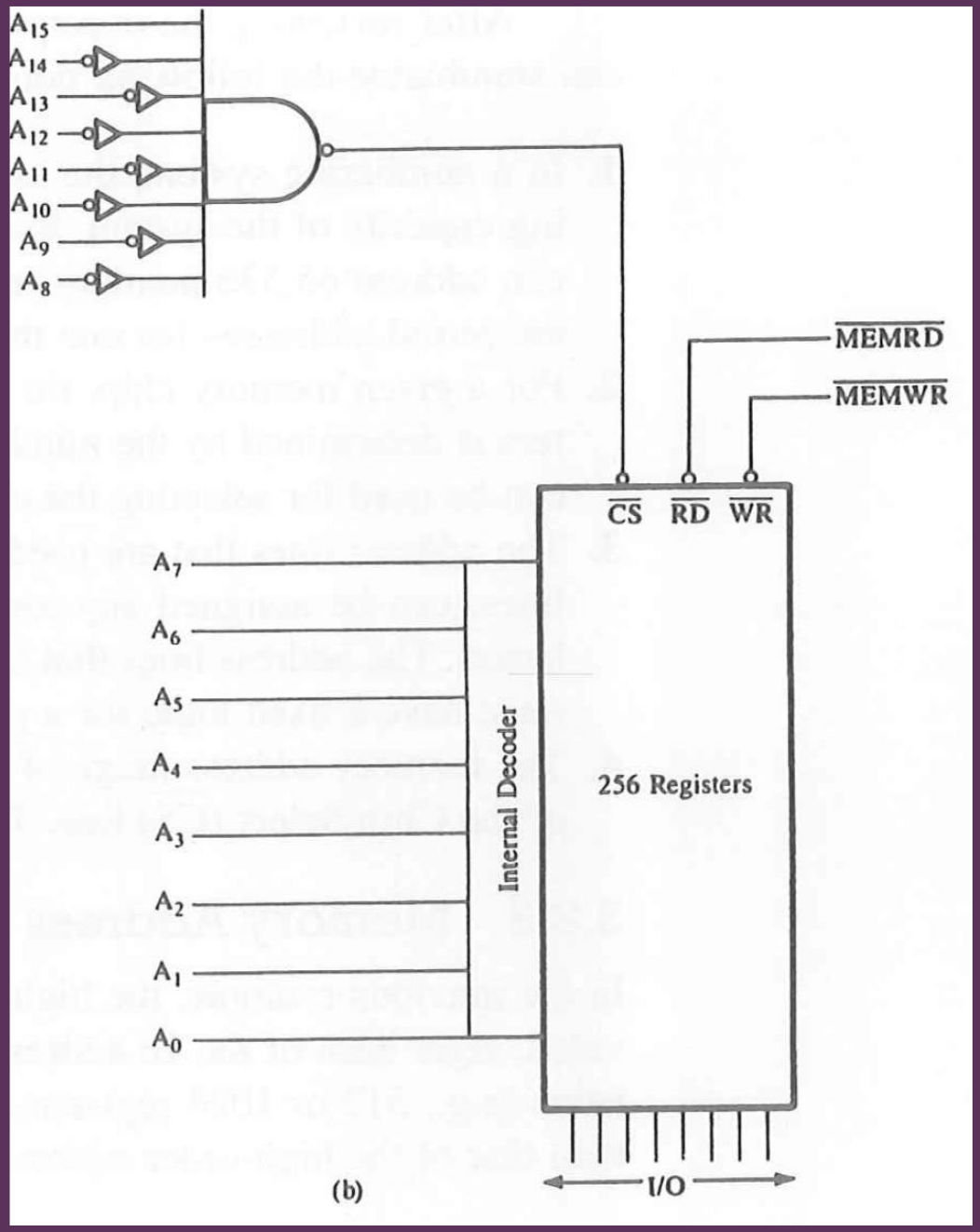


FIGURE 3.10
Memory Maps: 256 Bytes of Memory

Address Range of a Memory Chip

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	
1	0	0	0	0	0	0	0	= 80H

The memory address range in Figure (b) will be 8000H to 80FFH.



Address Range of a Memory Chip

Example 3.2	Explain the memory address range of 1K (1024 × 8) memory shown in Figure 3.11 and explain the changes in the addresses if the hardware of the $\overline{CS}$ line is modified.

Address Range of a Memory Chip

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	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	= 0000H
Chip Select Logic						1	1	1	1	1	1	1	1	1	1	= 03FFH

The memory address range in Figure will be 0000H to 03FFH.

By combining the high-order and low-order address lines, we can specify the complete memory address range of a given chip.

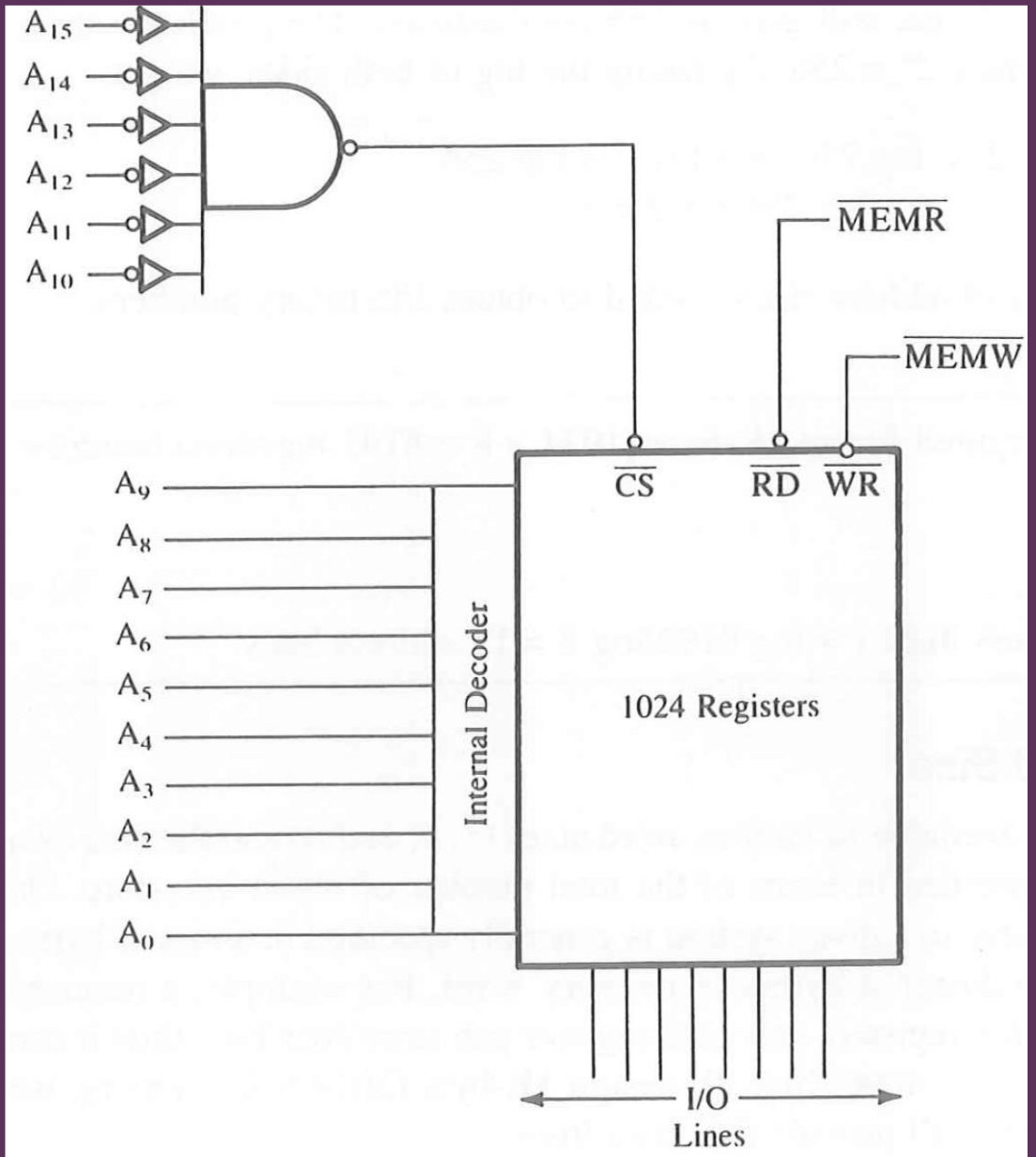


FIGURE 3.11
Memory Address Range: 1024 Bytes of Memory

Address Range of a Memory Chip

Now give the memory map and interfacing diagram for connecting this 1KB memory along with a 256 Byte memory in the 8085 microprocessor based system.

The next available memory is to map 256 Byte chip is:

0 0 0 0 0 1 0 0 || 0 0 0 0 0 0 0 0 → 0400
0 0 0 0 0 1 0 0 || 1 1 1 1 1 1 1 1 → 04FF

Memory Address Lines

- In the previous examples, the address lines of the memory chips were given.
- However, we need to know the relationship between the number of registers in a memory chip and the number of address lines.
- **Each address line can assume only two logic states (0 and 1); therefore, we need to find the power of 2**.

E.g. For a chip with 256 registers (x represents the number of address lines):

Find x where $2^x = 256$. By taking the log of both sides, we get:

$$\log 2^x = \log 256 \rightarrow x \log 2 = \log 256$$
$$x = \log 256 / \log 2 = 8$$

Memory Address Lines

- E.g

Calculate the address lines required for an 8K-byte ($1024 \times 8 = 8192$ registers) memory chip.

Number of address lines $x = \log 8192/\log 2 = 13$ address lines

Memory word size

- Memory devices (chips) are available in various word sizes (1, 4 and 8) and **the size of a memory chip is generally specified in terms of the total number of bits it can store**.
- On the other hand, **the memory size in a given system is generally specified in terms of bytes**.
- Therefore, it is necessary to design a byte-size memory word.
- E.g., a memory chip size 1024x4 has 1024 registers and each register can store four bits; thus it can store a total of 4096 (1024x4=4096) bits.
- To design 1K-byte (1024x8) memory, we will need two chips; each chip will provide four data lines.

Memory word size

- E.g

Calculate the number of memory chips needed to design 8K-byte memory if the memory chip size is 1024×1.

The chip 1024×1 has 1024 (1K) registers and each register can store one bit with one data line. We need eight data lines for byte-size memory. Therefore, eight chips are necessary for 1K-byte memory. For 8K-byte memory, we will need 64 chips. We can arrive at the same answer by dividing 8K-byte by $1K \times 1$ as follows:

$$8192 \times 8 \div 1024 \times 1 = 64$$

Basic Interfacing Concept

The approach to designing an interfacing circuit for an I/O device is determined primarily by the instructions to be used for data transfer. An I/O device can be interfaced with the 8085 microprocessor either as a peripheral I/O or as a memory-mapped I/O. In the peripheral I/O, the instructions IN/OUT are used for data transfer, and the device is identified by an 8-bit address. In the memory-mapped I/O, memory-related instructions are used for data transfer, and the device is identified by a 16-bit address.

I/O with 8-bit Address (Peripheral-Mapped I/O)

- The 8085 microprocessor has a separate 8-bit addressing scheme (I/O space) for I/O devices.
- This is known as peripheral-mapped I/O (also known as I/O-mapped I/O).

- The 8 address lines can have 256 combination of addresses; thus the MPU can identify 256 input and 256 output devices with addresses ranging from 00H to FFH.

I/O with 8-bit Address (Peripheral-Mapped I/O)

- The steps in communication with an I/O device are similar to those in communicating with memory and can be summarized as follows:
  1. The MPU places an 8-bit address on the address bus, which is decoded by external decode logic.
  2. The MPU sends a control signal (I/O Read or I/O Write) and enables the I/O device.
  3. Data are transferred using the data bus.

I/O with 8-bit Address (Peripheral-Mapped I/O)

The 8085 microprocessor has two instructions for data transfer between the processor and the I/O device: IN and OUT. The instruction IN (Code DB) inputs data from an input device (such as a keyboard) into the accumulator, and the instruction OUT (Code D3) sends the contents of the accumulator to an output device such as an LED display. These are 2-byte instructions, with the second byte specifying the address or the port number of an I/O device.

I/O with 8-bit Address (Peripheral-Mapped I/O)

Opcode	Operand	Description
OUT	8-bit Port Address:	This is a two-byte instruction with the hexadecimal opcode D3, and the second byte is the port address of an output device. This instruction transfers (copies) data from the accumulator to the output device.

Typically, to display the contents of the accumulator at an output device (such as LEDs) with the address, for example, 01H, the instruction will be written and stored in memory as follows:

Memory Address	Machine Code	Mnemonics		Memory Contents	
2050	D3	OUT 01H	; 2050 →	1 1 0 1 0 0 1 1	= D3H
2051	01		; 2051 →	0 0 0 0 0 0 0 1	= 01H

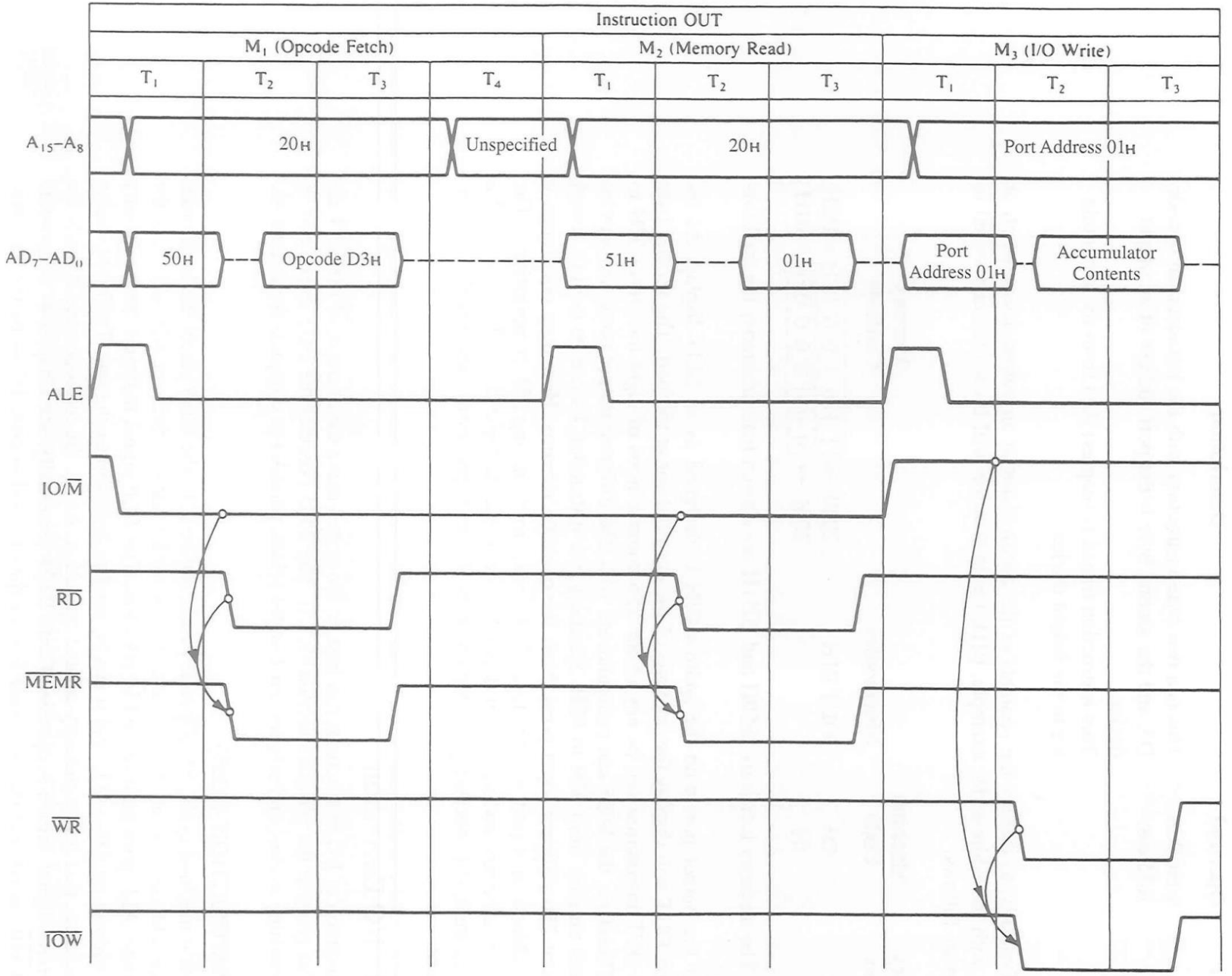


FIGURE 5.1
8085 Timing for Execution of OUT Instruction

I/O with 8-bit Address (Peripheral-Mapped I/O)

IN INSTRUCTION

The 8085 instruction set includes the instruction IN to read (copy) data from input devices such as switches, keyboards, and A/D data converters. This is a two-byte instruction that reads an input device and places the data in the accumulator. The first byte is the opcode, and the second byte specifies the port address. Thus, the addresses for input devices can range from 00H to FFH. The instruction is described as

IN 8-bit This is a two-byte instruction with the hexadecimal opcode DB, and the second byte is the port address of an input device.

This instruction reads (copies) data from an input device and places the data byte in the accumulator.

I/O with 8-bit Address (Peripheral-Mapped I/O)

To read switch positions, for example, from an input port with the address 84H, the instructions will be written and stored in memory as follows:

Memory Address	Machine Code	Mnemonics		Memory Contents	
2065	DB	IN 84H	; 2065 →	1 1 0 1 1 0 1 1	= DBH
2066	84		; 2066 →	1 0 0 0 0 1 0 0	= 84H

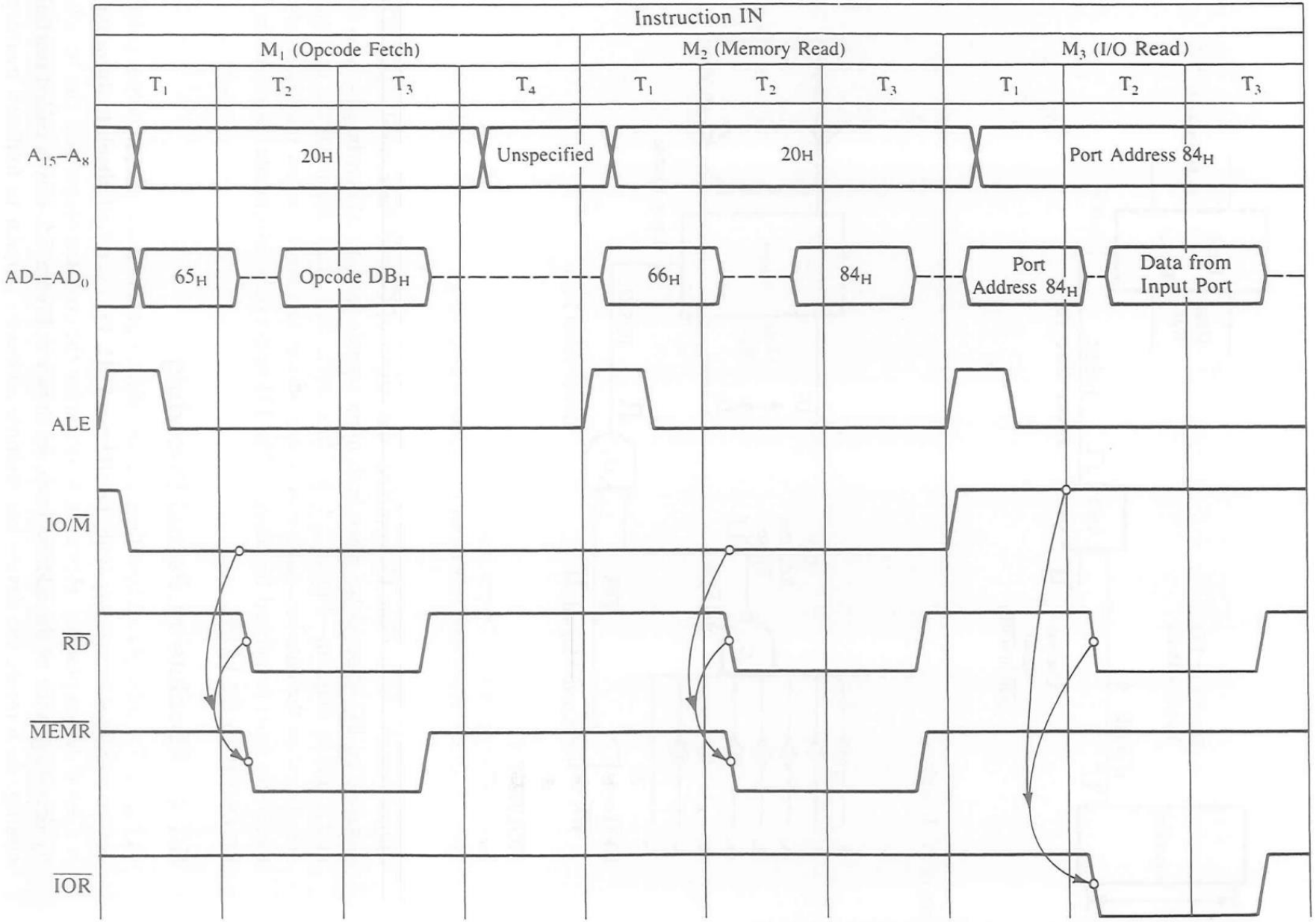


FIGURE 5.2
8085 Timing for Execution of IN Instruction

I/O with 16-bit Address (Memory-Mapped I/O)

- In this type of I/O, the MPU uses 16 address lines to identify an I/O device; an I/O is connected as if it is a memory register.
- The 8085 microprocessor uses 16-bit address bud for identifying and accessing memory registers. Ranging from 000H to FFFH.
- This is known as memory-mapped I/O.

To transfer data between the MPU and I/O devices, memory-related instructions (such as LDA, STA, etc.)* and memory control signals ($\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$) are used. The microprocessor communicates with an I/O device as if it were one of the memory locations.

Comparison of Memory-Mapped I/O and Peripheral I/O

Characteristics	Memory-Mapped I/O	Peripheral I/O
1. Device address	16-bit	8-bit
2. Control signals for Input/Output	$\overline{MEMR}$/$\overline{MEMW}$	$\overline{IOR}$/$\overline{IOW}$
3. Instructions available	Memory-related instructions such as STA; LDA; LDAX; STAX; MOV M,R: ADD M; SUB M; ANA M: etc.	IN and OUT
4. Data transfer	Between any register and I/O	Only between I/O and the accumulator
5. Maximum number of I/Os possible	The memory map (64K) is shared between I/Os and system memory	The I/O map is independent of the memory map; 256 input devices and 256 output devices can be connected
6. Execution speed	13 T-states (STA,LDA) 7 T-states (MOV M,R)	10 T-states
7. Hardware requirements	More hardware is needed to decode 16-bit address	Less hardware is needed to decode 8-bit address
8. Other features	Arithmetic or logical operations can be directly performed with I/O data	Not available