1. Registers and Semiconductor Memorie

1.1 What is a microprocessor-based system?

A microprocessor-based system is an electronic digital system that utilizes a microprocessor as its central processing unit (CPU). Microcomputer in one among many microprocessor-based systems It includes three essential elements:

1. Microprocessor(s);
2. Input/output (I/O) devices (e.g., keyboard, mouse, display, storage devices);
3. Storage memories (RAM/ROM);

which are organized through a communication pathway called a bus, as illustrated in the Figure 1.1.

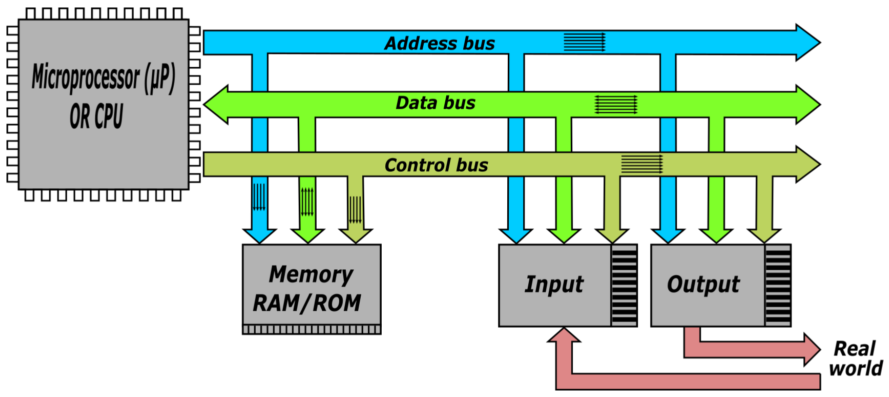


Figure 1.1: Key components of a microprocessor-based system

The functions of these elements can be summarized as follows:

1. Microprocessor
   - reads instructions from memory.
   - communicates with all peripherals (memory and I/Os) using the system bus.

- controls the timing of information flow.
- performs the computing tasks specified in a program

2. The memory
   - stores binary information, called instruction and data.
   - provides the instructions and data to the microprocessor on request.
   - stores results and data for the microprocessor.
3. The input device
   - enters data and instructions under the control of a program such as monitor program.
4. The output device
   - accepts data from the microprocessor as specified in a program.
5. The bus
   - carries bits between the microprocessor and memory and I/Os.

1.2 From transistor to registre/memory

1.2.1 Transistor

What is a transistor ? When an n-type or p-type semiconductor is connected between two p-type or n-type semiconductors, respectively, two PN junctions are formed. These two junction devices are known as PNP or NPN transistors, as shown in Figure 1.2.

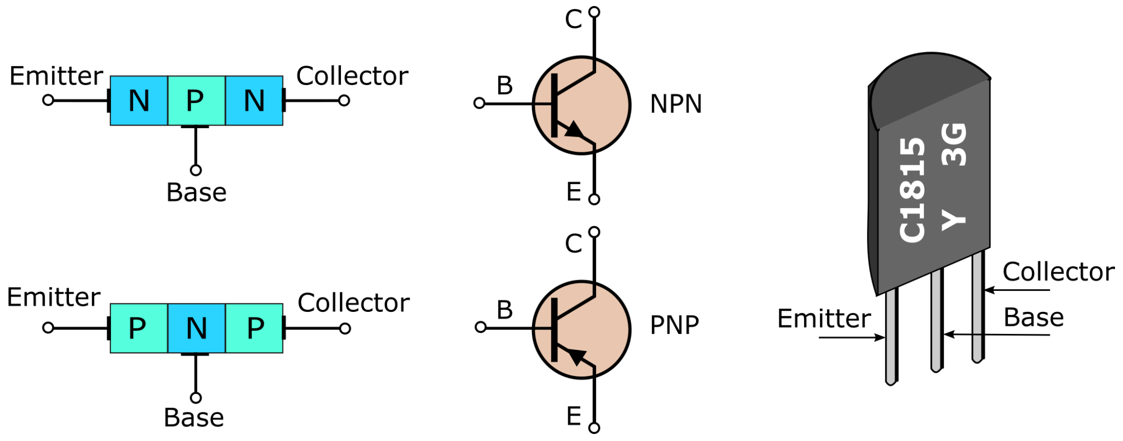


Figure 1.2: NPN and PNP transistor

The transistor is widely considered one of the greatest inventions of the 20th century, it is used in nearly every electric device (microprocessor, memroy, radio, ...) and enormous range of applications such as embedded systems, digital circuits and control systems, its main uses are:

- **Amplification**: Transistors are commonly used in analog domains to amplify electronic signals, as shown in Figure 1.4. For example, in microphones. In this operating mode, the transistor is functioning in the active region illustrated in Figure 1.3.
- **Switching**: Transistors can act as switches, enabling the creation of binary signals (0s and 1s), which serve as the foundation of digital domains and are essential components of all digital logic circuits, as illustrated in Figure 1.5. In this mode, the

transistor operates in the cut-off region when it is open and in the saturation region when it is closed, as demonstrated in Figure 1.3.

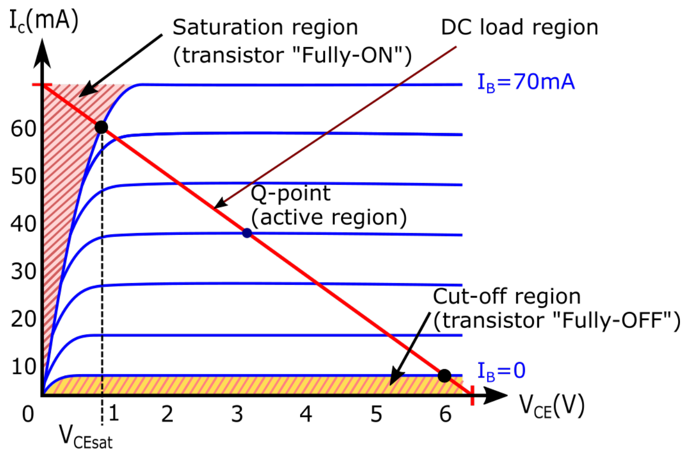


Figure 1.3: Operating regions of a transistor based on the NPN transistor output characteristics

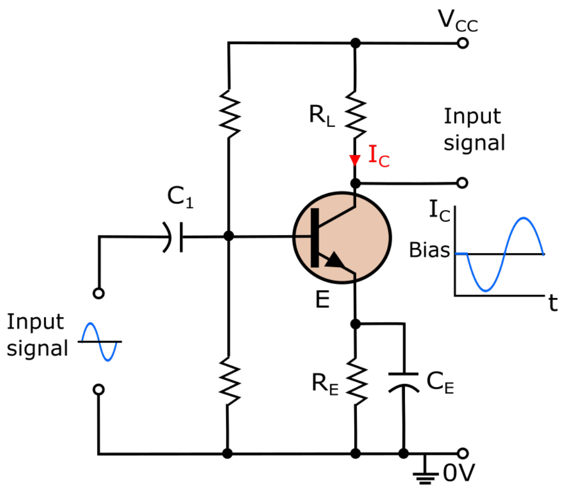


Figure 1.4: NPN transistor working in the active region as a simple Class A amplifier

1.2.2 Logic gates and transistor

In mathematics, there are arithmetic operations (addition, subtraction, division, and multiplication: +, -, /, x), and a second type of operations that apply exclusively to binary numbers (0 and 1). These are known as logical operations and are used to perform logical functions. These operations are divided into two types:

- Fundamental logical operations: NOT, AND, OR;
- Derived operations: NAND, NOR, XOR, XNOR

.

As demonstrated in Figure 1.5, transistors can function as switches at the nanometer scale. Because of this capability and their speed, they were initially used to create logic gates, which are physical devices that perform logical operations. This, in turn, led to the development of memories, microcontrollers, and microprocessors.

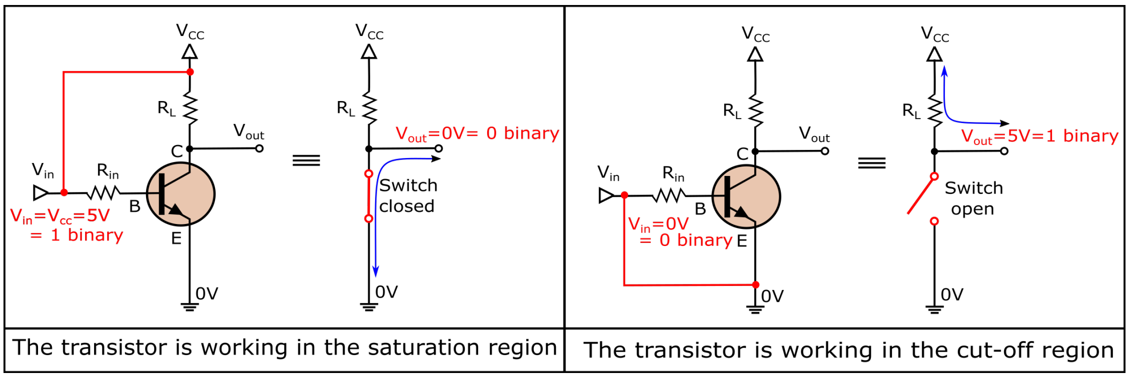


Figure 1.5: NPN transistor working as switch

Figure 1.6 depicts the equations, symbols, and truth tables of the NOT, NAND, and NOR logic operations. It also illustrates how transistors operating in switching mode are used to construct logic gates for these operations. Similarly, logic gates for the remaining operations, AND, OR, XOR, and XNOR, can also be constructed using transistors."

1.2.3 Combinational and sequential Logic

Combinational logic refers to circuits based on logic gates where the outputs depend solely on the inputs, without any feedback elements. However, in digital electronics, there is a need for memory, counters, etc. where the outputs depend on both their current inputs and their previous outputs. Circuits following this principle are classified as sequential logic. Most of these circuits are constructed using combinational circuits and memory elements known as 'flip-flops' (Figure 1.7).

1.2.4 Sequential Logic and flip-flops

Flip-flops are fundamental components in sequential logic circuits, used to store and manipulate binary data. The RS flip-flop is the most common among all; it is constructed using basic logic gates such as NAND gates or NOR gates. As depicted in Figure 1.8, an RS flip-flop consists of three inputs (R for Reset or Q=1, CLK, and S for Set or Q=0) and two complementary outputs (Q and $\bar{Q}$). When Set "S" is active (S=1), the output "Q" would be low (Q=0), and "$\bar{Q}$" would be high ($\bar{Q} = 0$). While when Reset "R" is active (R=1), the output "Q" would be high (Q=1), and "$\bar{Q}$" would be low ($\bar{Q} = 0$).

Using the truth table of the NAND gate provided in Figure 1.6, it can be found that the various cases of uncontrolled RS flip-flops (Figure 1.6(a)) are as listed in Table 1.1:

Table 1.1: Truth table of RS flip-flop

S	R	Q	$\bar{Q}$
0	0	Not used	
0	1	1	0
1	0	0	1
1	1	Memory (as before)	

Note: *Memory for the fourth case means that when the inputs take the new values S=1*

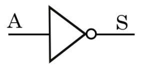
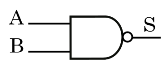
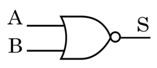
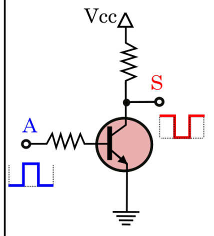
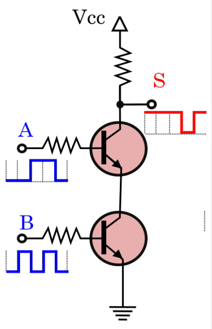
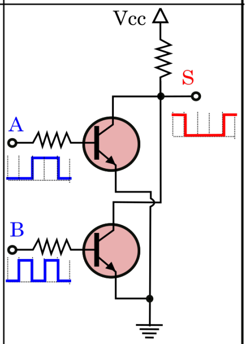
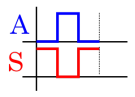
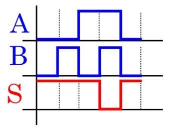
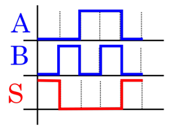
Gate	NOT	NAND	NOR
Equation	$S = \overline{A}$	$S = \overline{A.B}$	$S = \overline{A+B}$
Symbol			
Truth table	A | S 0 | 1 1 | 0	A | B | S 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	A | B | S 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0
Transistor-based logic gate			
Timing diagram			

Figure 1.6: Equations, symbols, truth tables, and transistor-based gates for NOT, NAND, and NOR logic operations

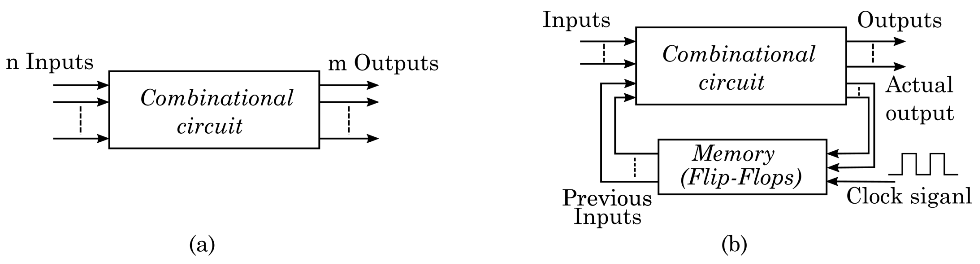


Figure 1.7: Combinational (a) and sequential (b) logic principles

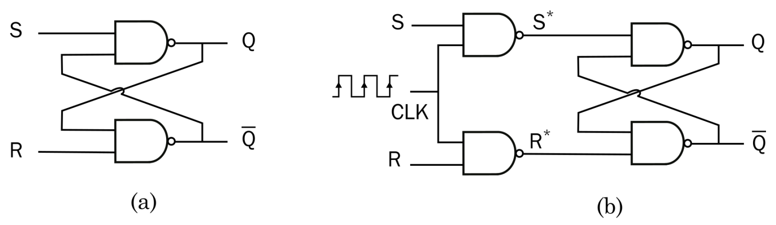


Figure 1.8: Uncontrolled (a) and controlled (b) input RS flip-flop

and R=1, after the set or reset cases (S=1/0 and R=0/1), the outputs maintain their set or reset values. This means the RS flip-flop stores the previous values

To prevent accidental changes to the inputs of the RS flip-flop and ensure that they change only when desired, two new NAND gates are added along with the clock signal, as shown in Figure 1.8(b). The truth table of the controlled RS flip-flop is given in Table 1.2.

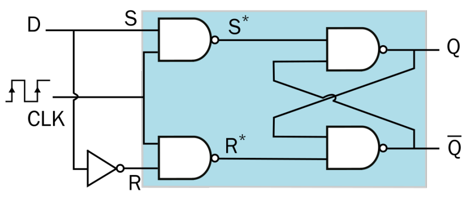
Table 1.2: Truth table of RS flip-flop

CLK	S	R	Q	$\bar{Q}$
0	x	x	Memory	
1	0	0	Not used	
1	0	1	0	1
1	1	0	1	0
1	1	1	Memory	

Note: *The same circuit as shown in Figure 1.8(b) can act as a* ***Latch****, when the CLK input of that circuit is used as the Enable or level-triggering signal. In other words, the circuit will operate when this input is high.*

1.2.5 D flip-flop

To prevent the occurrence of the “not used” case in the RS flip-flop (S=0 and R=0), an inverter (NOT gate) is connected between the SET and RESET inputs. This modification results in a new type of flip-flop called a D flip-flop (Figure 1.9).



Clock	D	Q	$\bar{Q}$	Description
↓»0	X	Q	$\bar{Q}$	No change
↑»1	0	0	1	Store 0
↑»1	1	1	0	Store 1

Figure 1.9: Schematic and truth table of the D flip-flop

The “D” in D flip-flop stands for Data. Therefore, a D flip-flop stores a single bit of data, it is characterized by a clock input, a data input, and a Q output. It is an edge-triggered device, transferring input data to Q on either the rising or falling edge of the clock signal (Figure 1.10).

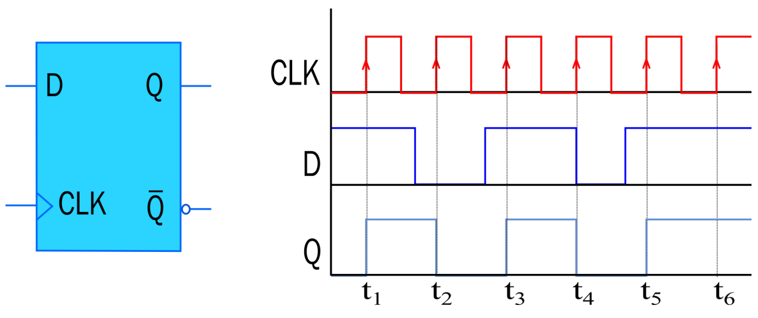


Figure 1.10: Block schematic and timing diagram of the D flip-flop

1.2.6 D flip-flop : latch and clocked

A latch is a D flip-flop used as a memory device for storing one bit of data. However, the main difference between them is that the flip-flop is an edge-triggered circuit, while the latch is a level-triggered circuit, A typical example of a latch (7475) and a D flip-flop (7474) is shown in Figure 1.11.

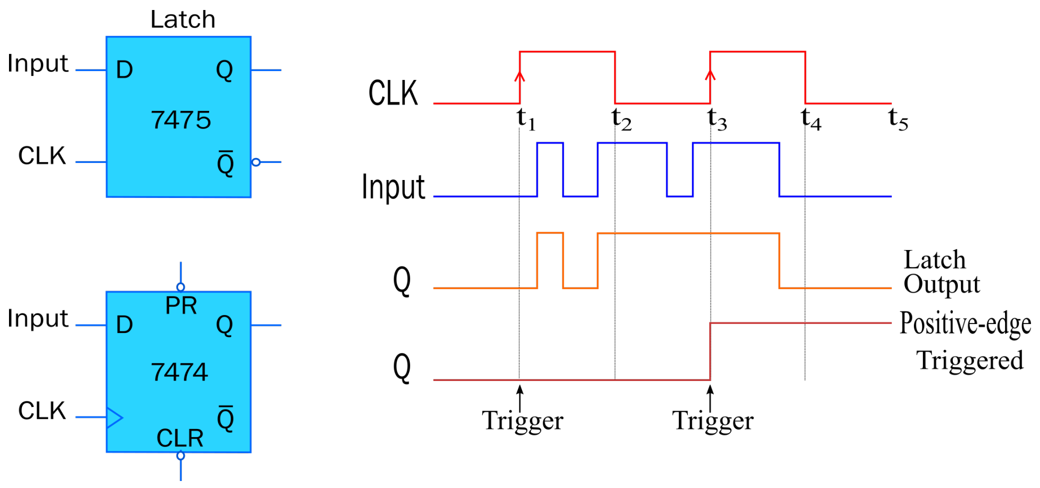


Figure 1.11: Output Waveforms of Latch and Positive-Edge-Triggered Flip-Flop

1.2.7 Logic devices for interfacing

In digital electronics, it is common for several components to share the same bus lines in bus-based systems. For example, in a computer system, multiple devices such as the CPU, memory, and peripherals may be connected to the same data bus. While the bus can be used by one component at a time, it is crucial to use interfacing devices to control the flow of data on that bus. Among these devices, there are:

Tri-state devices

Tri-state devices are logic components with three states: logic 1, logic 0, and high impedance. As depicted in Figure 1.12, a tri-state logic device includes a third line called Enable. When activated (set to 1), the input is routed to the output. Conversely, when disabled, the logic device enters the high-impedance state, which is essentially a theoretically open circuit.

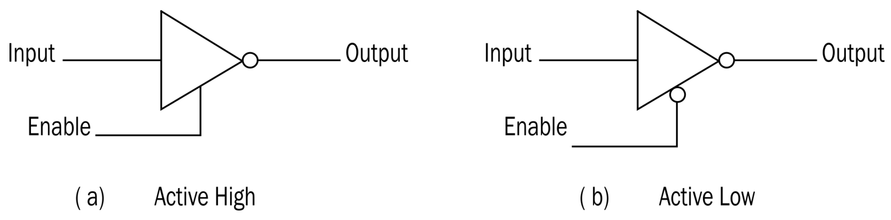


Figure 1.12: Tri-State Inverter with (a) Active High and (b) Active Low Enable Lines

Buffer

The buffer is a logic circuit that amplifies the current or power. It is used to increase the driving capability of a logic circuit and is also known as a driver (See Figure 1.13.

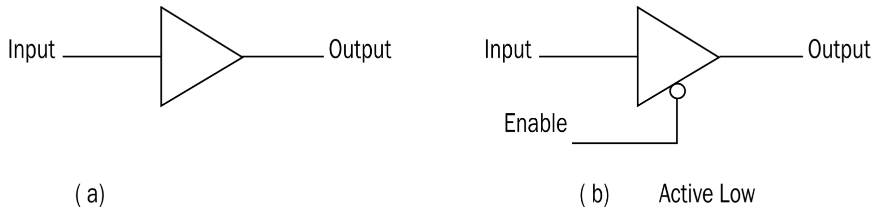


Figure 1.13: Buffer (a) and Tri-State Buffer (b)

1.3 Flip-Flop or Latch as Storage Element

What is memory? It is a circuit that can store bits—high or low, generally voltage levels or capacitive charges representing 1 and 0. A flip-flop or a latch is a basic element of memory. To write or store a bit in the latch, we need an input data bit (D_{IN}) and an enable signal (EN), as shown in Figure 1.14(a). To avoid unintentional input and control the availability of the output, one tri-state buffer is used at the input and another at the output, as shown in Figure 1.14(b). Now, it can be written to the latch by enabling the input buffer and read from it by enabling the output buffer. The write signal is referred to as $\overline{WR}$, and the read signal as $\overline{RD}$. This latch, capable of storing one binary bit, is called a binary cell.

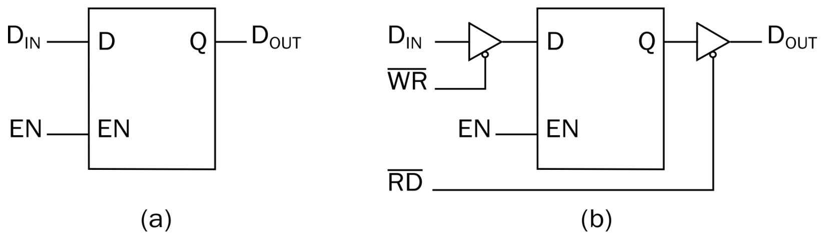


Figure 1.14: Latch as Storage Element: Basic Latch (a) and Latch with Two Tri-State Buffers (b)

1.4 Registers

A register is a set of memory cells or latches grouped together, used to store combinations of bits called words (e.g., 0011, 0010, 1100, 11110000, 11001100). Figure 1.15(a) shows four cells grouped together forming a 4-bit register that can store a memory word of four bits. The register comprises:

- Input lines
- Output lines
- Enable signal (EN)
- Write signal $\overline{WR}$
- Read signal $\overline{RD}$

Figure 1.15(b) and (c) show a simplified block diagram of the 4-bit register. To obtain a register with an 8-bit memory word, we need to group 8 memory cells together.

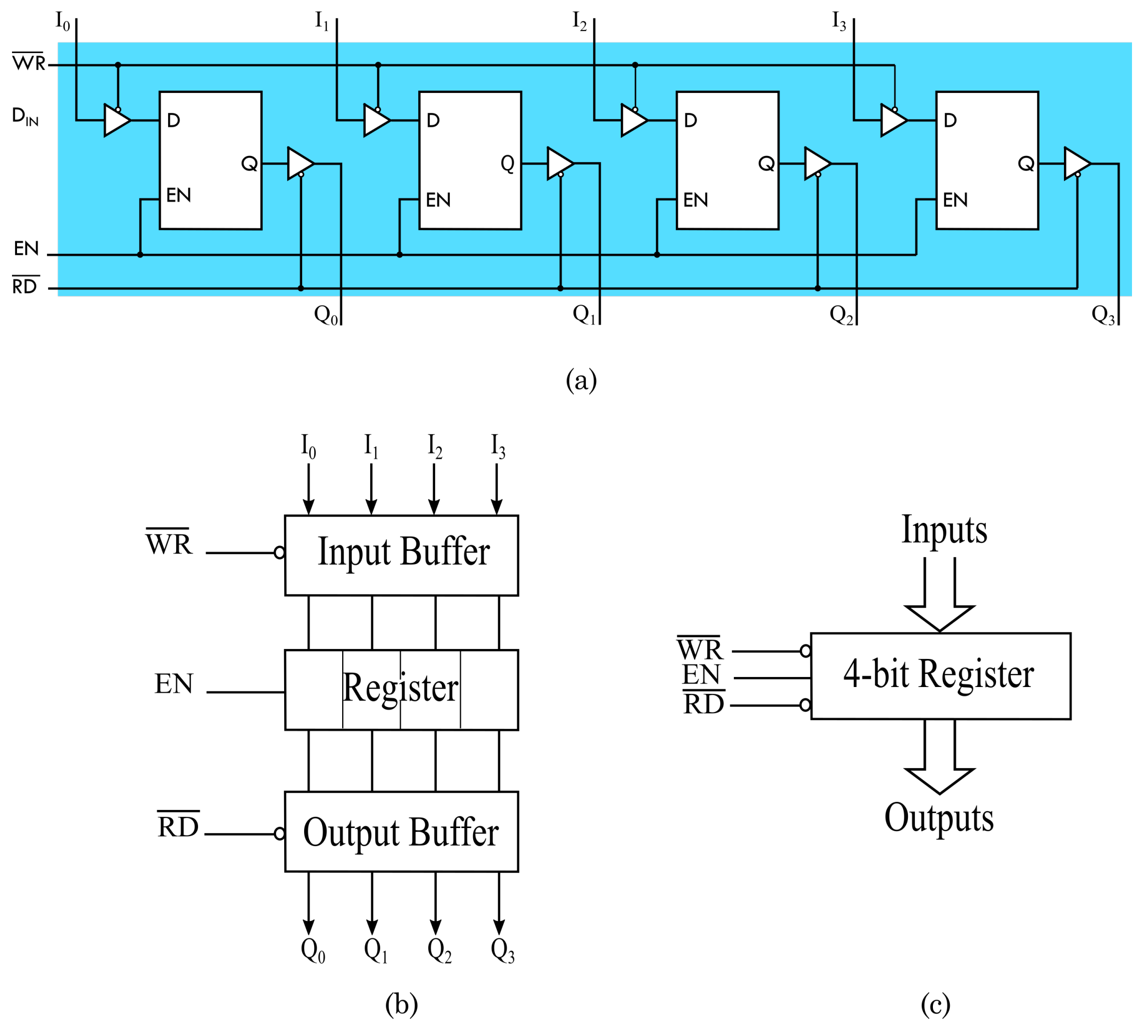


Figure 1.15: Four Latchs as a 4-Bit Registers (a) and Block Diagrams of a 4-Bit Register (b and c)

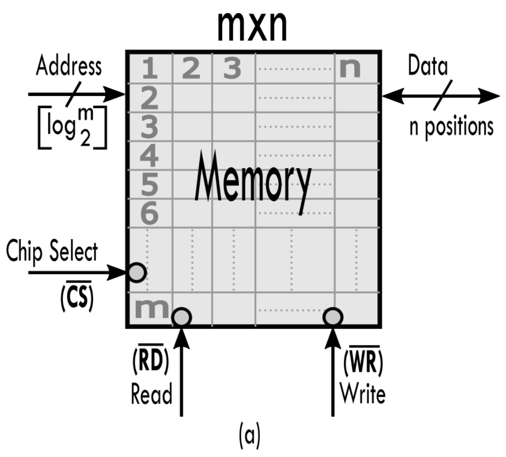
1.5 Memory

1.5.1 Definition

Memory is an essential component of a microprocessor system; it stores instructions and data for the microprocessor. It is classified into primary (main) memory and storage memory. The primary memory has two types: Read/Write memory (R/WR) and Read-Only Memory (ROM).

The R/W memory is composed of registers grouped together. These registers are arranged in a sequence and identified by binary numbers called memory addresses. As shown in Figure 1.16(a), the word size of the register is referred to as *n* and can be 4-bit, 8-bit, 16-bit, etc. The number of registers, referred to as *m*, can vary from a few in some memories to hundreds of thousand in others. To communicate with memory, the microprocessor should be able to:

- ✓ select the chip using $\overline{CS}$ signal ;
- ✓ identify the register through the address bus, and
- ✓ read from or write to the register through the data bus using $\overline{RD}$ or $\overline{WR}$ signal.



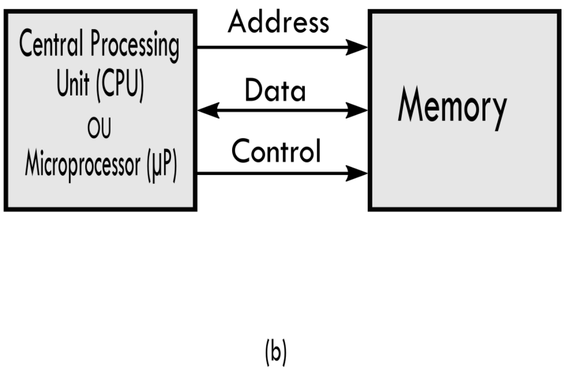


Figure 1.16: Input/Output Signals of a Memory (a) and Its Connection to a Microprocessor (b)

Note: The small bubble at the input of read, write, or chip select signals (Figure 1.17(a) indicates that the signal is active low. It is activated when it equals binary '0'.

An example of memory consists of four registers with an 8-bit word size, input and output buffers, and a 2-to-4 decoder to identify and enable one of the registers, as shown in Figure 1.17.

The memory capacity can be expressed in:

- Bit;
- Byte: 1 byte = 8 bits;
- Kilobyte (KB): 1 kilobyte = 1024 bytes = 2^{10} bytes;
- Megabyte (MB): 1 megabyte = 1024 KB = 2^{20} bytes;
- Gigabyte (GB): 1 gigabyte = 1024 MB = 2^{30} bytes;
- Terabyte (TB): 1 terabyte = 1024 GB = 2^{40} bytes;

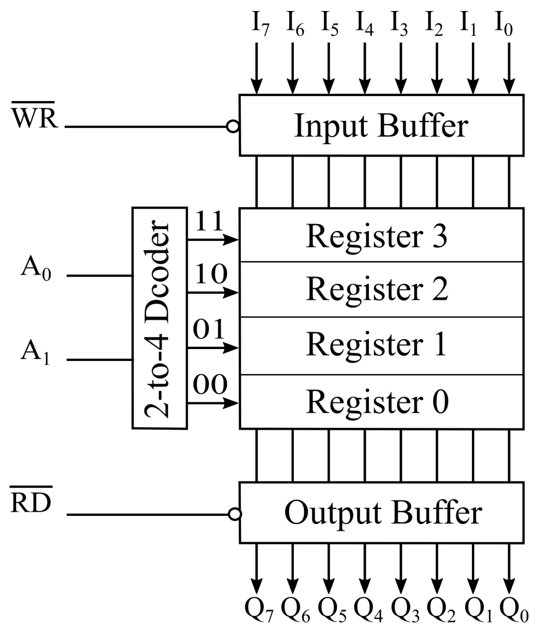


Figure 1.17: Memory with 4 Registers of 8-Bit Word Size

1.5.2 Memory access time

Memory access time is the duration it takes for a character in memory to be transferred to or from the microprocessor. In other words, it is the time it takes to read from or write to memory.

1.6 Classification of Memories

Memories can be classified based on its fabrication materials into several types, each with distinct characteristics. Here are common classifications based on fabrication materials:

- **Semiconductor Memory:** Memory cells are constructed using semiconductor materials, typically silicon. Generally used is as the main memory element of a microcomputer-based system and is used to store program and data (RAM, ROM, Flash memory), it is very fast but of small sizes.
- **Magnetic Memory:** Memory stores data by manipulating magnetic fields. Information is recorded and read through changes in magnetic alignment (Hard disk drives (HDDs), magnetic tapes, ...). It is less fast but has high storage capacity
- **Optical Memory:** Memory that uses light to read and write data. Information is stored as variations in the optical properties of the material (Compact Discs (CDs), Digital Versatile Discs (DVDs)).

1.6.1 Semiconductor memories

In this type of memory, each binary data bit is stored in a memory cell made of one or more transistors. Figure 1.18 shows the various types of this memory. Firstly, a distinction is made between ROM (Read-Only Memories) and RWM (Read & Write Memories). Secondly, it is categorized based on whether it belongs to the non-volatile or volatile class, i.e., whether the contents are retained or lost when power is turned off. Then, a further distinction can be made for volatile RWM devices, including other types of memory structures used for specific purposes, such as FIFO, LIFO, shift registers, and those with random access (RAMs).

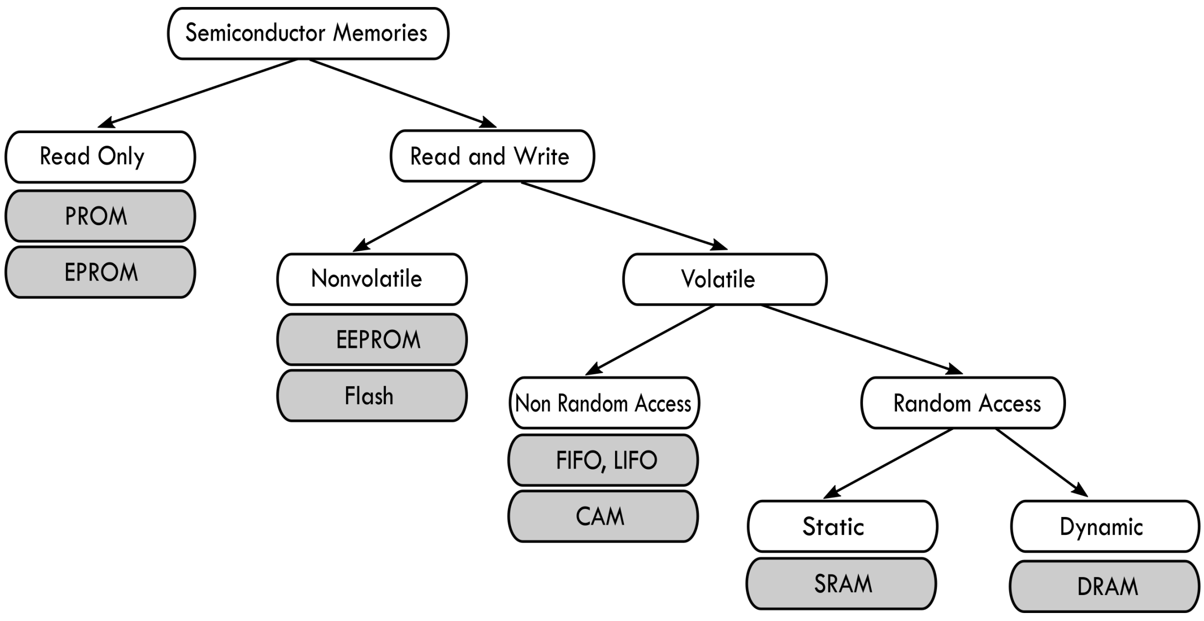


Figure 1.18: Classification of Semiconductor Memories

- **PROM** (Programmable Read Only Memory): The PROMs are one time programmable. The user can program then with special PROMs programmer. The PROMs programmer selectively burns the fuses according to the bit pattern to be stored.
- **EPROM** (Erasable PROM): EPROMs can be programmed by the user using a special EPROM programmer and erased by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes.
- **EEPROM** (Electrically Erasable PROM): EEPROM has a special chip erase mode by which the entire chip can be erased in 10 ms, and it can be erased and reprogrammed with the device right in the circuit.
- **Flash memory**: It is a type of EEPROM with the ability to write and erase data in blocks, rather than one byte at a time. It is used in various electronic devices, including USB drives, memory cards, solid-state drives (SSDs), and some types of embedded systems.

1.6.2 Random Access Memory (RAM)

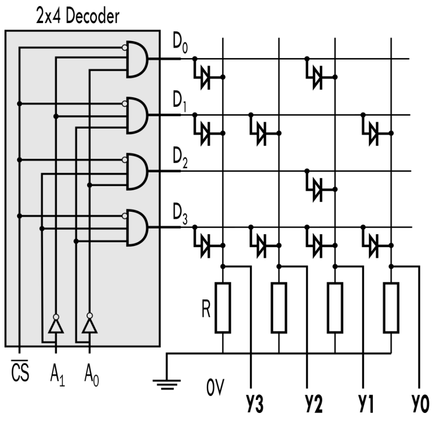
RAM is a type of volatile computer memory that is used to store data and machine code currently being used and processed by a computer. It serves as the immediate working space for the processor due to its quick read and write access. It can be classified as either static RAM or dynamic RAM.

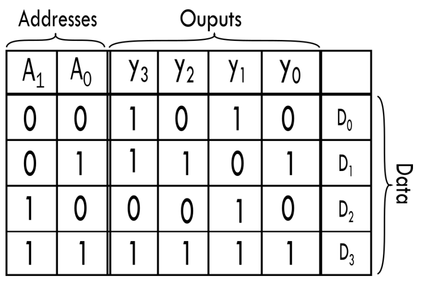
- **Static RAM (SRAM)**: Memory cell of SRAM is typically implemented using a D flip-flop. It is more expensive to manufacture and consumes more power compared to DRAM. However, it does not require constant refreshing. SRAM is a fast memory, responding to the CPU with very low latency time. Therefore, it is commonly used for cache memory.

- **Dynamic RAM (DRAM)**: Dynamic RAM stores the data as a charge on the capacitor. Its disadvantage is its needs to refreshing of charge on the capacitor after every few milliseconds. This complicates the system design, since it requires the extra hardware to control refreshing of dynamic RAMs. It is generally used for main computer memory.

1.6.3 Internal Structure of ROM

The input/output lines of a 4x4-bit ROM are arranged as illustrated in Figure 1.19.





Addresses		Ouputs				
A_1	A_0	y_3	y_2	y_1	y_0	Data
0	0	1	0	1	0	D_0
0	1	1	1	0	1	D_1
1	0	0	0	1	0	D_2
1	1	1	1	1	1	D_3

Figure 1.19: Internal structure, addressing, and reading of a 4x4-bit ROM memory

Figure 1.20 illustrates the schematic diagram and the chip of a 128 KB EPROM (Am29F010).

Pins discription:

A_0-A_{16} : 17 Addresses
DQ_0-DQ_7 : 8 Data Inputs-Outputs
CE : Chip Enable
OE : Output Enable
WE : Write Enable
NC : Pin not connected
Vcc : +5V
Vss : Device ground

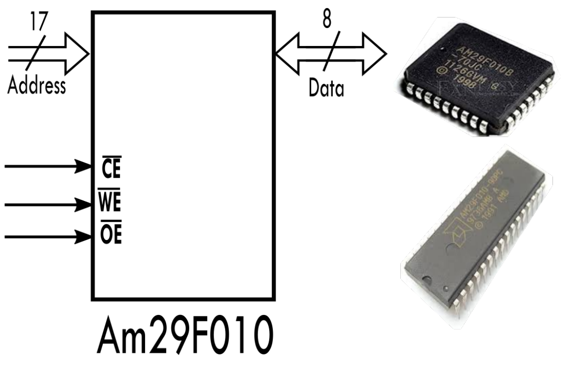


Figure 1.20: Schematic Diagram and Chip of 128 KB EPROM

1.6.4 Timing Diagram for ROM Read Operation

The Figure 1.21 shows the timing diagram of a ROM memory read operation, starting with the CPU (Central Processing Unit) sending the address of the desired data to the ROM and ending with the ROM retrieving the corresponding data stored at that address and providing it as an output.

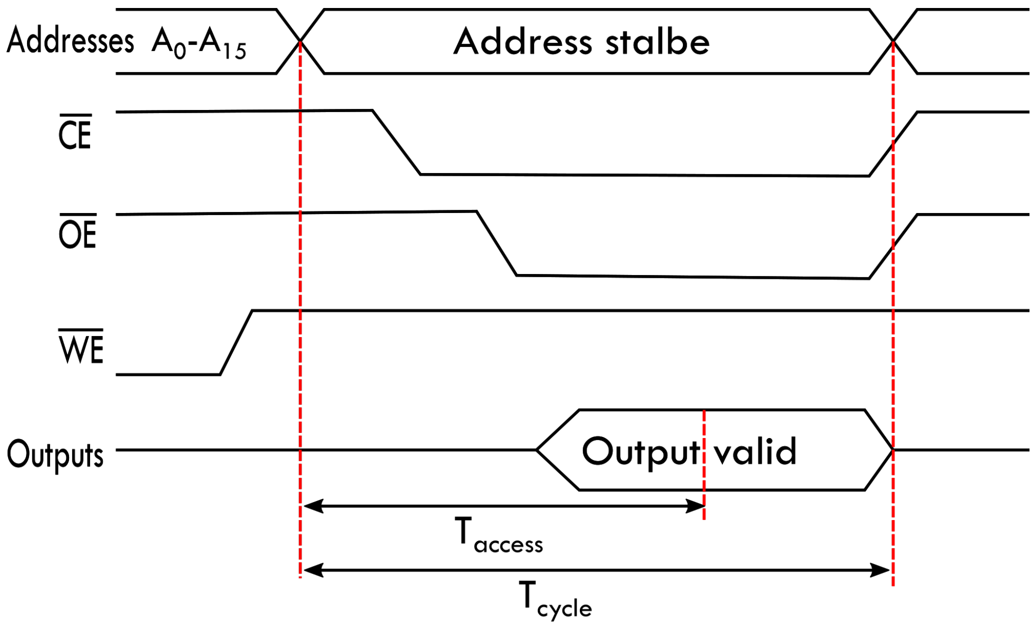


Figure 1.21: Timing Diagram for ROM Read Operation

1.6.5 Internal Structure of RAM

The illustration in Figure 1.22 depicts the internal organization of a typical 256-byte RAM. It includes a 2D-array of 1-bit storage cells (D flip-flop), an 8x256 address decoder, and input and output amplifiers.

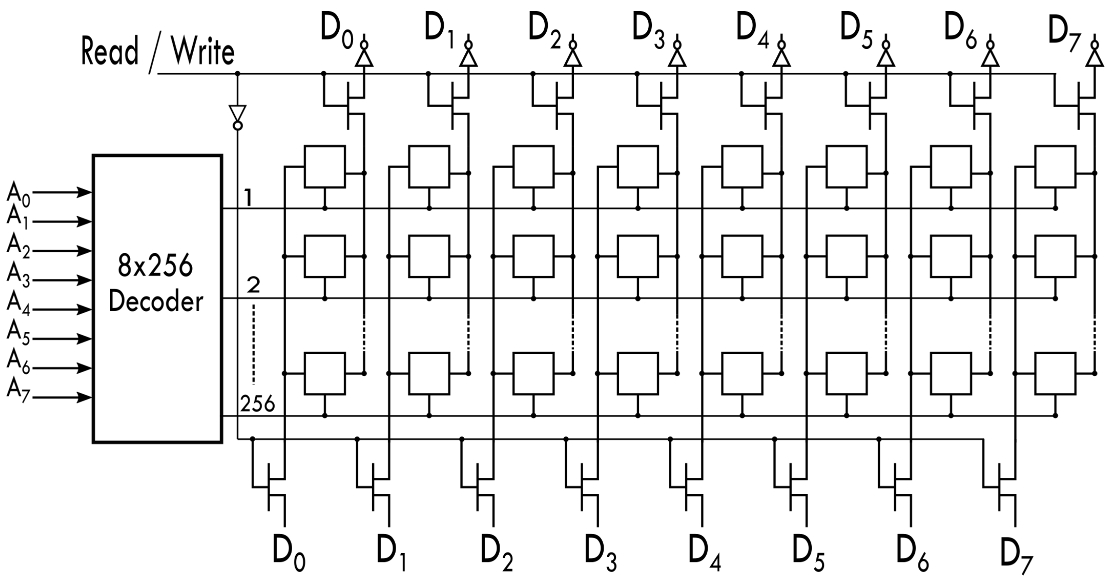


Figure 1.22: Internal Structure of a 256-byte RAM

Figure 1.23 illustrates the schematic diagram and the chip of a 2KB Static RAM CMOS (6116). This memory can retain stored information with a power consumption of 5 micro-watts.

Pin Discription:

A_0-A_{10}: 11 addresses pins

D_0-D_7: 8 data pins

CS: ship selection pin

OE: output enable pin

$\overline{W}$/R: operation selction pin for Write or Read

Vcc: +5V

GND: 0V)

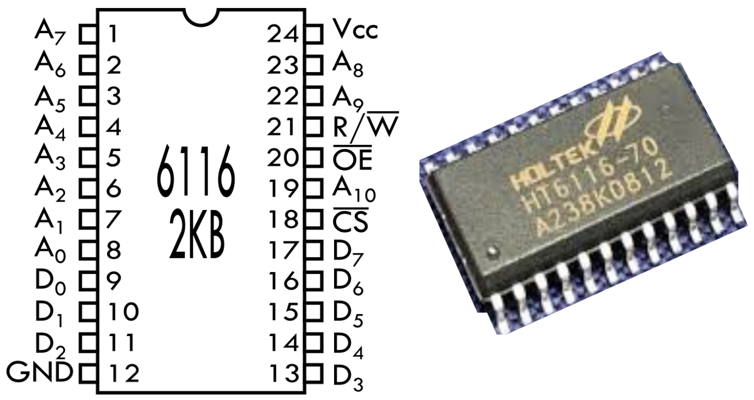


Figure 1.23: Schematic Diagram and Chip of 2KB CMOS RAM

1.6.6 Timing Diagram for RAM Read/Write Operation

The Figure 1.24 depicts the timing diagram for reading and writing to a RAM memory.

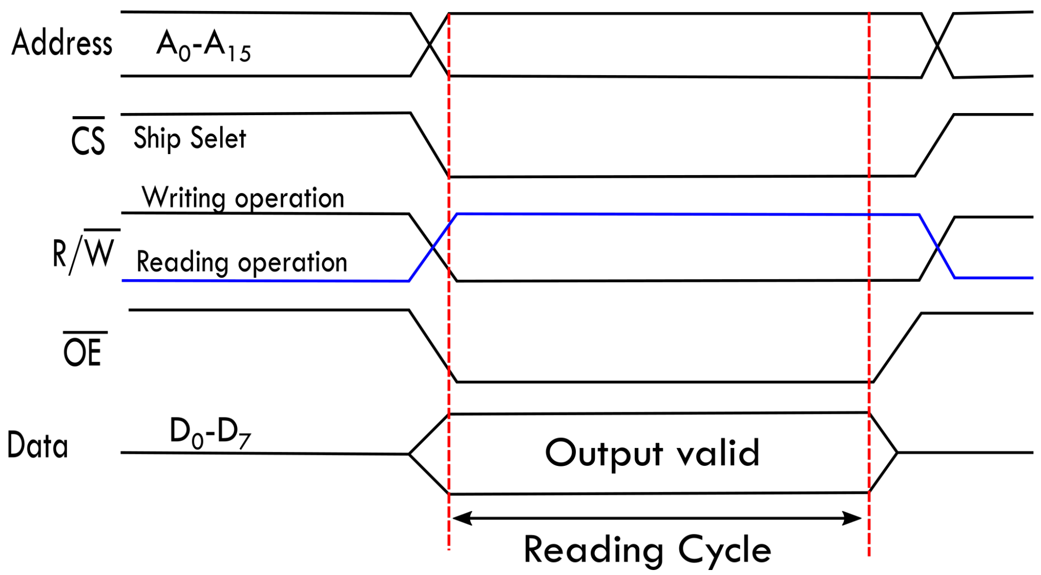


Figure 1.24: Schematic Diagram and Chip of 2KB CMOS RAM

1.7 Memory Chip Configuration for Expanding Memory Size

As each microprocessor-based system application requires a specific memory size, it becomes necessary to combine multiple memory chips to increase either the memory word size or the total number of words

1.7.1 Expanding Memory Size by Increasing the Number of Words

Assuming we have memory chips of 64 KB each and aim to design a memory size of 256 KB, we need to use 4 chips of 64 KB (256/64 = 4). To address these chips, for each memory chip of 64 K registers, we require 16 address lines obtained as follows: $64K = 2^6 \times 2^{10} = 2^{16}$. Therefore, the 16 low-order address bus bits $(A_0 - A_{15})$ will be used to identify 65536 registers (64x1024 = 65536) within one of the 4 chips, while the two high-order address lines $(A_{16} - A_{17})$ will determine the chip number to be selected through the use of a 2x4 decoder. Figure 1.25 illustrates the addressing of the assembled 4 chips.

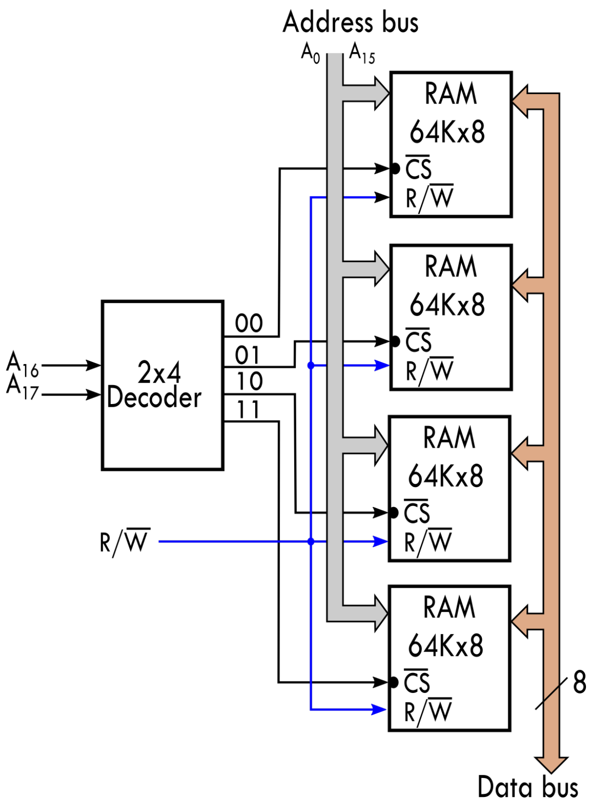


Figure 1.25: Memory Chip Assembly to Increase the Number of Words

1.7.2 Expanding Memory Size by Increasing the Word Size

To design a 64K memory with a 16-bit word size based on 64K chips with an 8-bit word size, we use two 64K x 8-bit chips, and they should be addressed simultaneously. Since 64K requires 16 address lines, as seen in the previous example, the 16 address bus bits $(A_0 - A_{15})$ are connected to the address pins of both chips simultaneously. The data output pins of the first and the second chips are considered the highest-order bits and the lowest-order bits of the resulting memory, respectively, as illustrated in Figure **??**.

1.8 Memory Map and Addresses

A memory map is a pictorial representation in which memory devices are located across the entire range of addresses. Assuming a microprocessor with a 16-bit address bus and an 8-bit data bus, how do we address a memory with a capacity below the maximum addressable capacity of that microprocessor, which is 65536 registers (2^{16})?

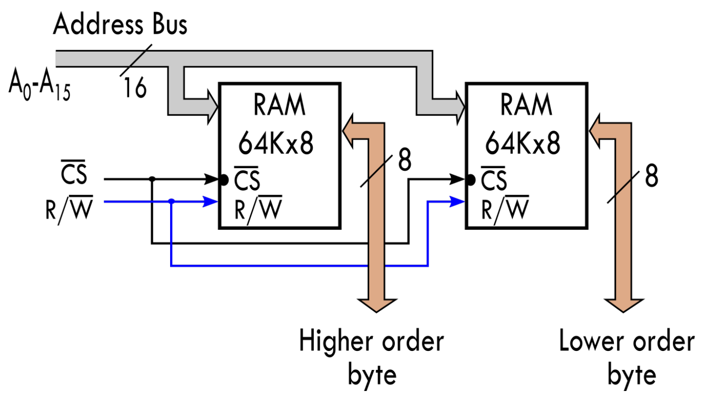


Figure 1.26: Memory Chip Assembly to Increase the Number of Words

Example 1

In this example, we want to create a memory map for a microprocessor-based system that features a microprocessor with a 16-bit address bus and an 8-bit data bus, along with an 8 KB memory (8K x 8-bit) assembled using 8 chips of 2K x 4 bits.

Number of Chips: To create an 8K memory using a 2K memory chip, we need to connect 4 chips in parallel.

On the other hand, to increase the word size to 8 bits, we need to connect two rows of 4 chips of 4 bits in series.

Address lines: To determine the number of address lines to address 4K of memory space, we must solve the following equation:

$$Log_2^m = 2K = 2^{11} \Longrightarrow m = 11\, lines \tag{1.1}$$

Figure 1.27 shows the memory map of an 8KB memory achieved by connecting 2K-4bit memory chips in series and in parallel.

Example 2

In this example, we will design a microprocessor-based system with a 16-bit address bus, an 8-bit data bus, an 8KB ROM, and a 16KB RAM. The memory chips for RAM and ROM are each 4KB. The ROM address should start from address 0000H, and the RAM address should start from address 8000H (Figure **??**).

Note: The unused address space between the two memories allows for future expansion of the ROM.

"If we neglect the "folding" issue, memory addresses will be calculated using the following rule:"

$$\mathrm{FA} = \mathrm{SA} + \mathrm{Cap(B)} - 1 \tag{1.2}$$

With:

FA: Final Address;

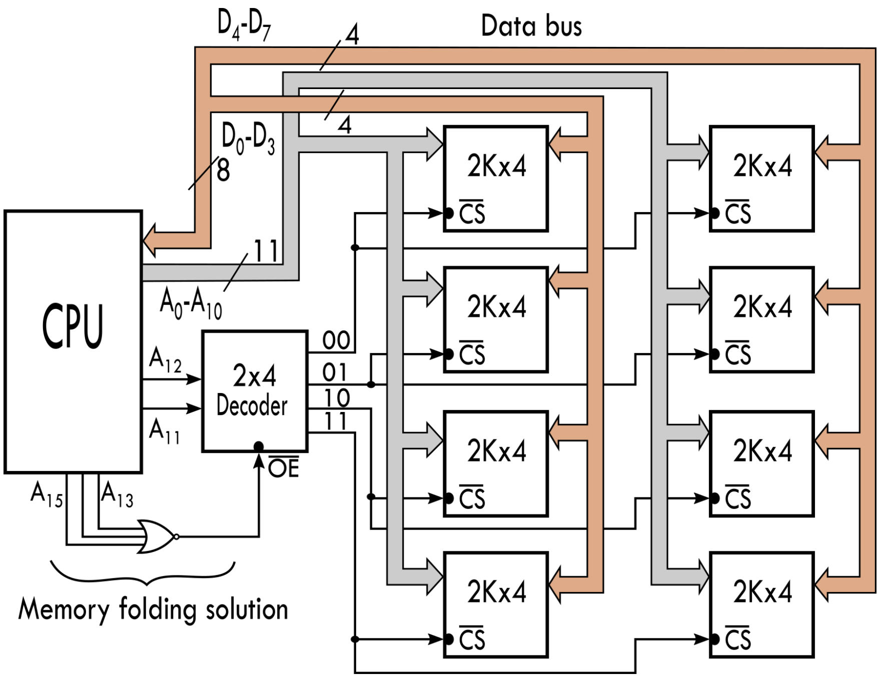


Figure 1.27: Memory map of example I

SA: Starting Address;

Cap(B)= Memory Capacity in Bytes;

ROM1:

$\text{FA} = \text{SA} + \text{Cap(B)} - 1$

$\text{FA} = 0000\text{H} + 4\text{K} - 1$

We have $4\text{KB} = 4\text{xK} = 2^2\text{x}2^{10} = 2^{12} = (1000000000000)_2 = (1000)_\text{H}$

$\text{FA} = 1000\text{H} - 1$

$\text{FA} = 0\text{FFFH}$

ROM2:

$\text{FA} = \text{SA} + \text{Cap(B)} - 1$

$\text{FA} = 1000\text{H} + 4\text{K} - 1$

$\text{FA} = 1000\text{H} + 1000\text{H} - 1$

$\text{FA} = 1000\text{H} + 0\text{FFFH}$

$\text{FA} = 1\text{FFFH}$

RAM1:

$$\mathrm{FA} = \mathrm{SA} + \mathrm{Cap(B)} - 1$$

$$\mathrm{FA} = 8000\mathrm{H} + 4\mathrm{K} - 1$$

$$\mathrm{FA} = 8000\mathrm{H} + 1000\mathrm{H} - 1$$

$$\mathrm{FA} = 8000\mathrm{H} + 0\mathrm{FFFH}$$

$$\mathrm{FA} = 8\mathrm{FFFH}$$

RAM2:

$$\mathrm{FA} = \mathrm{SA} + \mathrm{Cap(B)} - 1$$

$$\mathrm{FA} = 9000\mathrm{H} + 4\mathrm{K} - 1$$

$$\mathrm{FA} = 9000\mathrm{H} + 1000\mathrm{H} - 1$$

$$\mathrm{FA} = 9000\mathrm{H} + 0\mathrm{FFFH}$$

$$\mathrm{FA} = 9\mathrm{FFFH}$$

RAM3:

$$\mathrm{FA} = \mathrm{SA} + \mathrm{Cap(B)} - 1$$

$$\mathrm{FA} = \mathrm{A000H} + 4\mathrm{K} - 1$$

$$\mathrm{FA} = \mathrm{A000H} + 1000\mathrm{H} - 1$$

$$\mathrm{FA} = \mathrm{A000H} + 0\mathrm{FFFH}$$

$$\mathrm{FA} = \mathrm{AFFFH}$$

RAM4:

$$\mathrm{FA} = \mathrm{SA} + \mathrm{Cap(B)} - 1$$

$$\mathrm{FA} = \mathrm{B000H} + 4\mathrm{K} - 1$$

$$\mathrm{FA} = \mathrm{B000H} + 1000\mathrm{H} - 1$$

$$\mathrm{FA} = \mathrm{B000H} + 0\mathrm{FFFH}$$

$$\mathrm{FA} = \mathrm{BFFFH}$$

So, the address ranges of the six memories can be summarized as follows:

- **ROM1:**0000 – 0FFF
- **ROM2:**1000 – 1FFF
- **RAM1:**8000 – 8FFF

- **RAM2:**9000 – 9FFF
- **RAM3:**A000 – AFFF
- **RAM4:**B000 – BFFF

Figure 1.28 illustrates the configuration of the system of Example 2 , which includes 2 ROM chips and 4 RAM chips, each with a capacity of 4KB.

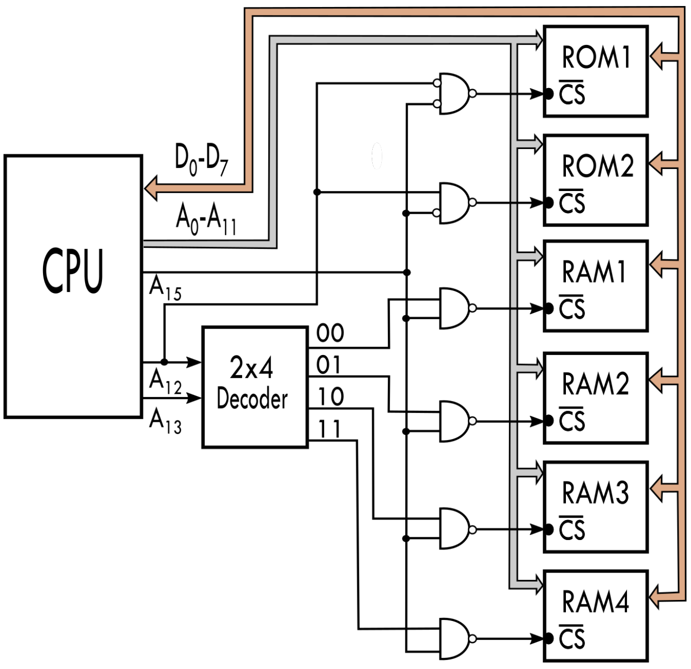


Figure 1.28: Design of a microprocessor-based system consisting of ROMs and RAMs