10.4 READ ONLY MEMORY (ROM)

A read only memory (ROM) is a semiconductor memory device used to store the information permanently. It performs only read operation and does not have a write capability. A ROM is programmed for a particular purpose during the manufacturing process and the user cannot alter its function. ROM circuits are typically used to provide the computer with resident programs and key operating functions needed to *boot* the operating system of the computer.

The ROM is a combinational logic circuit. It includes both the decoder and the OR gates within a single IC package. In order to minimize the number of address lines, decoders are used. The address of the desired line is given in binary. The connections between the output of the decoder and the input of the OR gates can be specified for each particular configuration. The ROM is used to implement complex combinational circuits within one IC package or as permanent storage package for binary information. The binary information must be specified by the designer and is then embedded in the unit to form the required interconnection pattern.

10.4.1 Architecture of ROM

The block diagram of a ROM is shown in Fig. 10.2. It consists of n address lines and m output lines. Each bit combination of the address variables is called an *address*. Each bit combination that comes out of the output lines is called a *data word*. Hence, the number of bits per word is equal to the number of output lines, m; an address is essentially a binary number that denotes one of the 2^n memory locations. An output word can be selected by a unique address; since there are 2^n distinct addresses in a ROM, there are 2^n distinct words that are said to be stored in the unit. The word available on the output lines at any instant depends on the address value applied to the input lines. A ROM is characterized by the number of words 2^n and the number of bits per word m; this terminology is used because of the similarity between the read only memory and the random access memory.

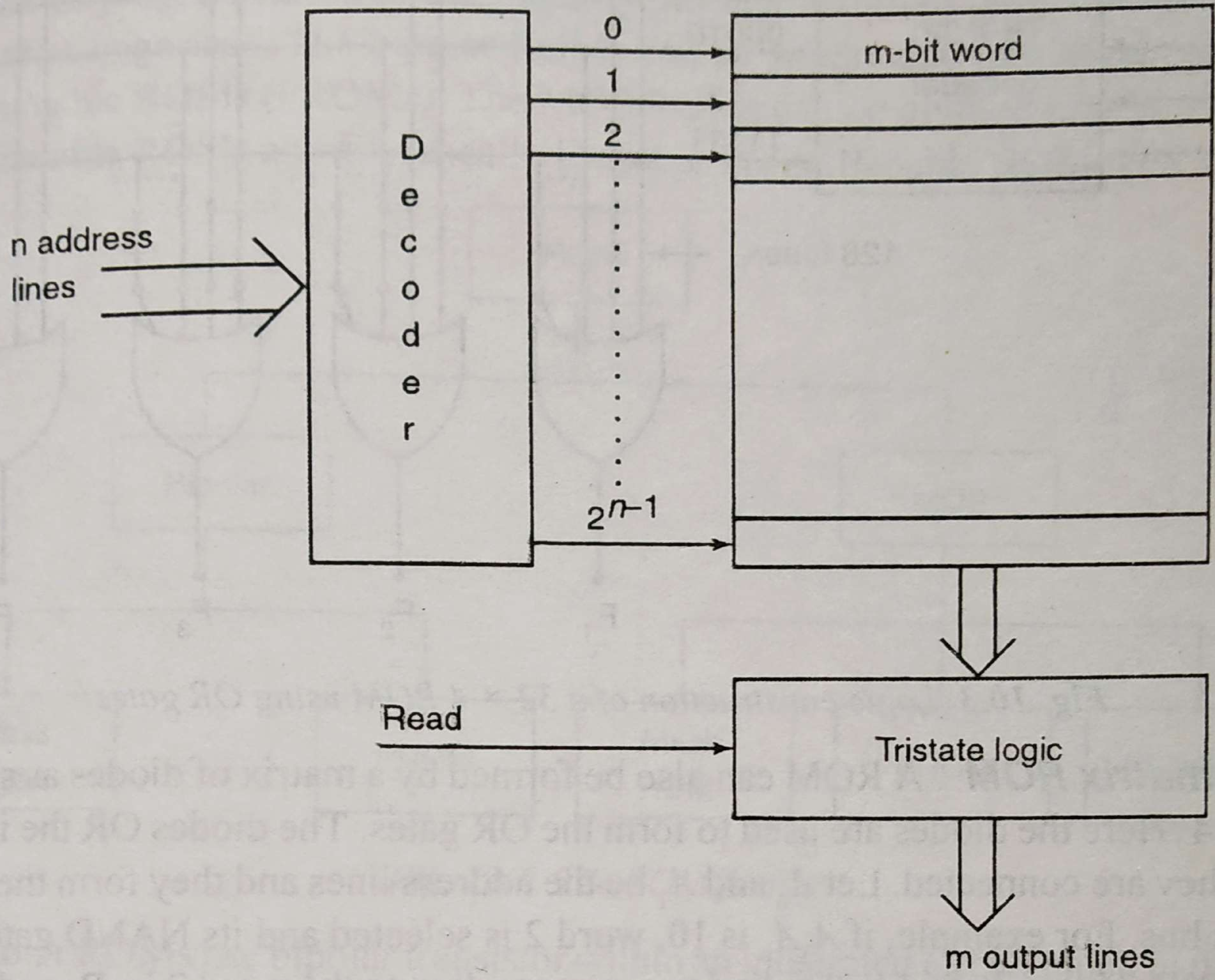

Fig. 10.2 Block diagram of ROM

Consider a 32 × 8 ROM. This unit consists of 32 words of 8 bits each. This means that there are eight (m) output lines and 32 distinct words in the unit, each of which may be applied to the output lines. There are only five (n) address lines in a 32 × 8 ROM because $2^5 = 32$; with five address variables, we can specify 32 memory locations. For each address, a unique word is selected. Thus, if the input address is 00010, word number 2 is selected and it appears on the output lines. If the input address is 10011, word number 19 is selected and applied to the output lines. Totally 32 addresses can select 32 words.

A ROM is sometimes specified by the total number of bits it contains, which is $2^n \times m$. For example, a 2048-bit ROM may be organised as 512 words of 4 bits each. This means that the unit has four (m) output lines and nine (n) address lines to specify 512 ($= 2^9$) words. The number of bits stored in the unit is $512 \times 4 = 2048$. The capacity of ROM varies from 256 bits to 256 KB. They are well suited to LSI manufacturing process.

32 × 4 ROM using OR gates Internally, the ROM is a combinational circuit with AND gates connected as *decoders*, and the number of OR gates is equal to the number of output lines in the unit. The internal logic construction of a 32 × 4 ROM is shown in Fig. 10.3. The five input variables are decoded into 32 ($= 2^5$) lines by means of 32 AND gates and 5 inverters. Each one of the 32 addresses selects one and only one output of the decoder. The 32 outputs of the decoder are connected through fuses to each OR gate. Only four of these fuses are shown in Fig. 10.3, but actually each OR gate has 32 inputs and each input of the OR gate goes through a fuse that can be blown as desired.

Address
input
A0
A1
A2
A3
A4
5 × 32
decoder
0
1
2
31
00000
00001
00010
11111
128 fuses
F1
F2
F3
F4

***Fig. 10.3** Logic construction of a 32 × 4 ROM using OR gates*

10.4.2 Types of ROM

Semiconductor ROMs are manufactured with *bipolar technology* or with *MOS technology*. ROMs can be classified into several types, which differ as to how information is written or *programmed* into the memory storage locations. Fig. 10.5 shows how ROMs are categorised. The bipolar ROMs can be subdivided into Mask ROMs and Programmable ROMs (PROMs). The MOS ROMs can be divided into Mask ROMs, Programmable ROMs and Electrically Erasable Programmable ROMs (EEPROMs).

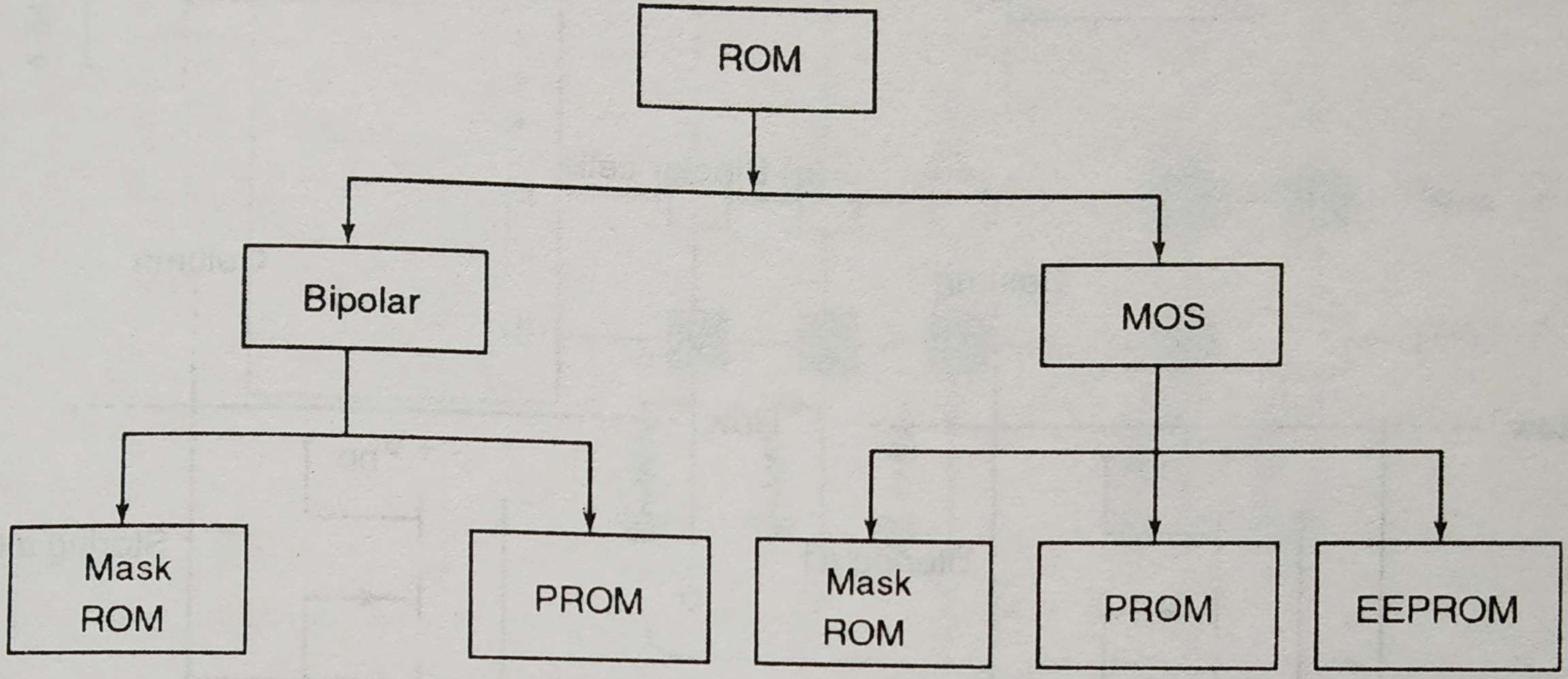

Fig. 10.5 *The ROM family*

Bipolar ROMs use bipolar transistor within an integrated circuit mucli in the same way as the diode matrix. The bipolar ROMs can be programmed. MOS ROM is constructed using MOSFETs. In a MOS cell, the silicon gate is completely insulated from the *n*-type substrate. The MOS can also be programmed in two ways. One is by placing charge on the gate at the time of manufacture.

10.4.3 Programming Mechanisms

ROMs may be programmed in two different ways. The first is called *mask programming* and is done by the manufacturer during the last fabrication process of the unit. The second type of ROM is *programmable read only memory* (PROM).

Mask-programmed ROM Mask ROM is permanently programmed during the manufacturing process. It is actually programmed by the manufacturer according to the customer's applications. Once the memory is programmed, it cannot be changed. Most IC ROMs utilize the presence or absence of a transistor connection at a ROW/COLUMN junction to represent 1 or 0. The particular 1s or 0s are obtained by providing a mask in the last fabrication step. A photographic negative called a mask is used to control the electrical interconnections on the chip.

10.4.4 Organization of a Simple ROM

A 16 × 8-bit simplified ROM array is shown in Fig. 10.7. This ROM is organized into 16 addresses, each of which stores 8 data bits. The total capacity of this ROM is 128 bits. The dark squares in this diagram represent stored 1s by means of base-connected transistor or gate-connected MOSFET, and the light squares represent stored 0s. When a 4-bit binary address is applied to the address inputs, the corresponding ROW line becomes HIGH. This HIGH is connected to the COLUMN line through the transistor at each junction where 1 is stored. The column line stays LOW at each cell where 0 is stored because of the terminating resistor. The COLUMN lines form the data output. Thus, the eight data bits stored in the selected ROW appear on the output lines.

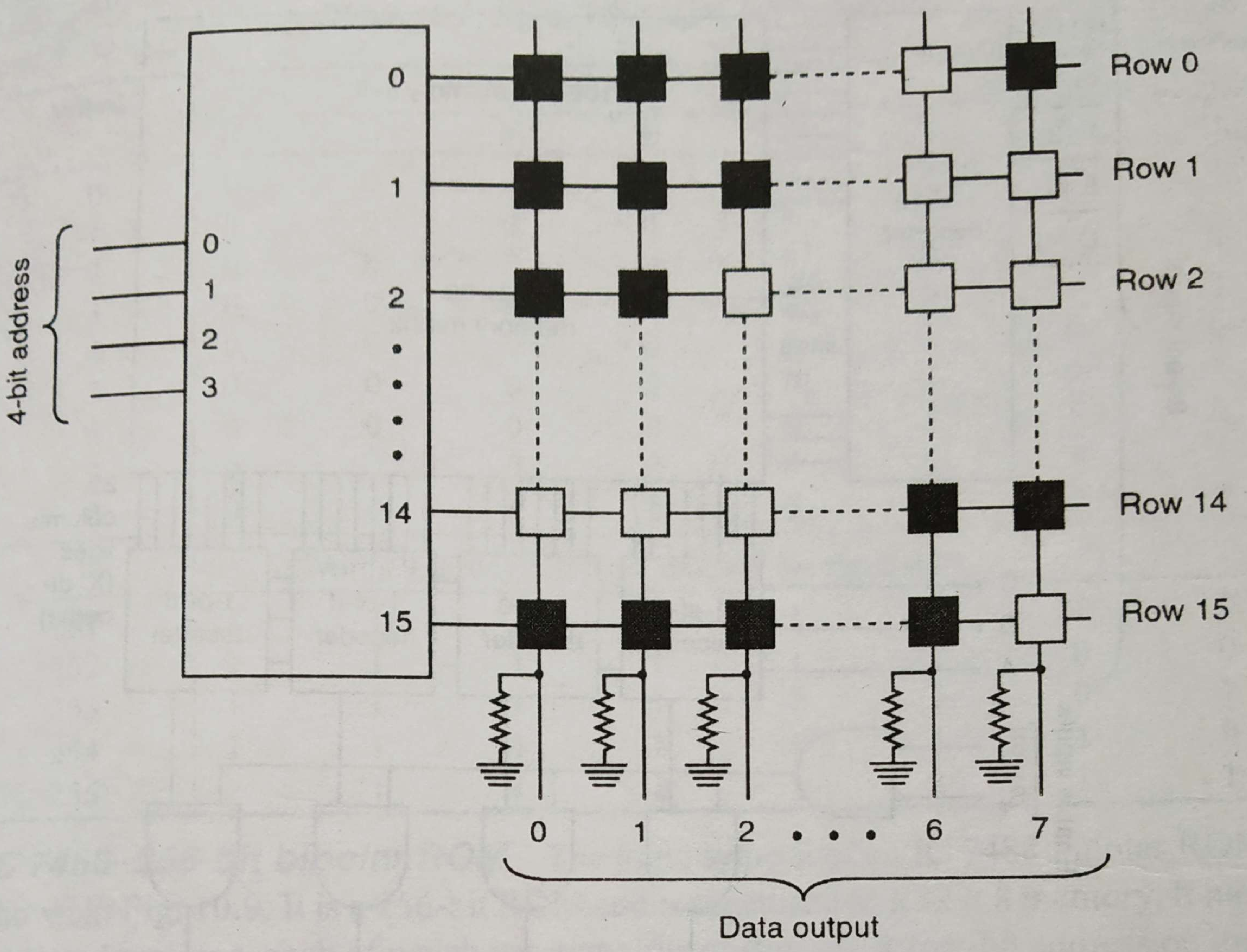

Fig. 10.7 A 16 × 8-bit ROM array

10.4.5 ROM ICs

Some of the ROM ICs are discussed in the following sections.

IC 74187–1024-bit ROM A typical MSI/TTL ROM IC 74187 is shown in Fig.10.8. It is a 16-pin, 1024-bit, 256 words of 4 bits each ROM. Although the 256 × 4 organization of this device implies that there are 256 rows and 4 columns in the memory, the memory cells are organized in a 32 × 32 matrix. Word selection of this memory is accomplished by means of an 8-bit address or select lines with the *A* being the least significant bit, progressing alphabetically through the select inputs to *H*, which is the most significant bit. The word-select table of this ROM is described in Table 10.1.

The working of ROM is as follows: Five of the eight address lines (*H* to *D*) are decoded by the ROW (*Y*) decoder to select one of the 32 rows. Three of the eight address lines (*C* to *A*) are decoded by the COLUMN (*X*) decoders to select four of the 32 columns. The *X* decoder consists of 1-of-8 decoders as shown in Fig. 10.8(a). The pinout diagram of 74187 is shown in Fig. 10.8(b).

When an eight-bit address code (*A* to *H*) is applied, a four-bit word appears on the data outputs (Y_4, Y_3, Y_2, Y_1), when the ENABLE inputs $\overline{S}_1$ and $\overline{S}_2$ are LOW.

H (15)
G (1)
F (2)
E (3)
D (4)
Select inputs
1-of-32
decoder
(Y)
32
row
lines
1024-bit memory cell
32 × 32
memory matrix
32
column
lines
(X de-
coder)
C
B
A
1-of-8
decoder
1-of-8
decoder
1-of-8
decoder
1-of-8
decoder
Enable inputs
$\overline{S}_1$
$\overline{S}_2$
(9)
(10)
(11)
(12)
Y_4
Y_3
Y_2
Y_1
Select
inputs
Enable inputs
Data outputs
V_{CC}
H
$\overline{S}_2$
$\overline{S}_1$
16
15
14
13
12
11
10
9
H
ME
Y_1
Y_2
Y_3
G
IC 74187
Y_4
F
E
D
A
B
C
1
2
3
4
5
6
7
8
GND
Select inputs

(a) Block diagram

(b) Pinout diagram

Fig. 10.8 IC 74187–1024-bit ROM

10.4.6 ROM Access Time

The ROM access time, t_a, is the time from the application of a valid address code on the address inputs until the appearance of valid output data. It can also be measured from the activation of *chip select* ($\overline{CS}$) to the occurrence of valid output data when a valid address is already on the inputs. A typical diagram that illustrates ROM access time is shown in Fig. 10.11.

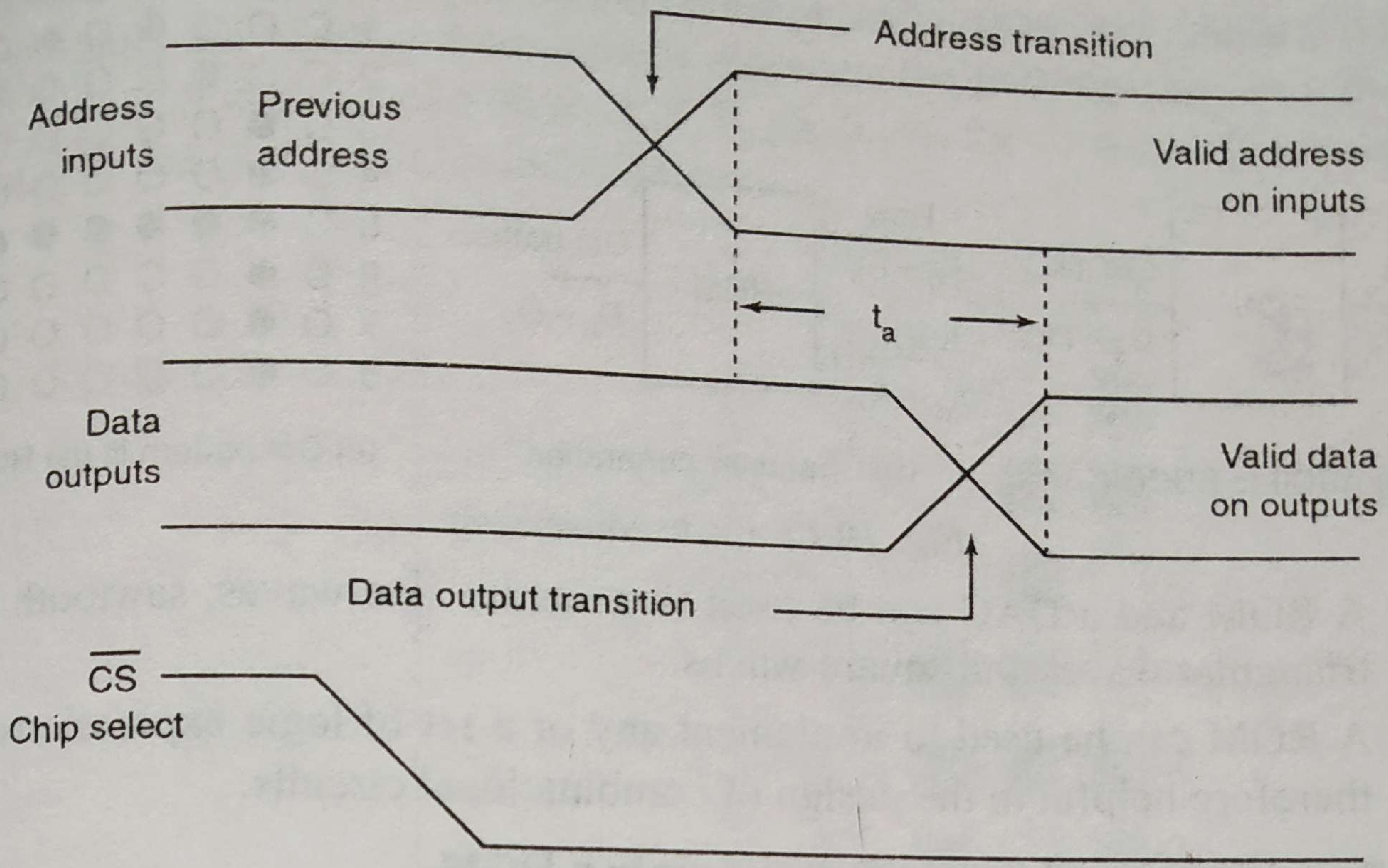

Fig. 10.11 Diagram of ROM access time

10.4.7 Applications of ROM

1. ROMs are used for a variety of tasks within a digital system. They can be used as a direct substitute for any random logic of AND, OR and NOT gates.
2. ROMs are used to store bootstrap program that loads operating system program available in secondary memory and language interpreters in personal and business computers and to store, monitor or control programs in microcomputer and microprocessor based systems like electronic games, electronic cash registers, electronic scales and microcomputer controlled automobile fuel injection.
3. A very significant application of MOS ROM is for character generation. This includes display control for moving billboards and light-emitting diode arrays. The TMS 4100 is a family of MOS ROMs organised to function primarily as a character generator. TMS 4100 family ROMs with a capacity of 2,240 bits, mounted in a 28-pin package with the following two different organisations, are available:
   (i) 64 words of 35 bits (5 by 7)
   (ii) 32 words of 70 bits (5 by 14)
4. ROMs can also be used for code conversion. In a ROM for ASCII to EBCDIC conversion as shown in Fig. 10.12(a), if ASCII data is placed on the address bus, the data bus will contain extended binary-coded decimal interchange code (EBCDIC). ROMs can also convert an ASCII data and row number into a dot pattern as shown in Fig. 10.12(b) & (c). Address lines A_0 to A_3 are used to select the row of dots, and address lines $A_4 - A_{10}$ are used to select the character that is to be displayed.

ASCII
$A_0 - A_7$
ROM
EBCDIC
$D_0 - D_7$
Row
$A_0 - A_3$
ASCII
$A_4 - A_{10}$
ROM
Dot pattern
$D_0 - D_7$
0 ○ ○ ○ ○ ● ○ ○ ○
1 ○ ○ ○ ● ○ ● ○ ○
2 ○ ○ ● ○ ○ ○ ● ○
3 ○ ● ○ ○ ○ ○ ○ ●
4 ○ ● ○ ○ ○ ○ ○ ●
5 ○ ● ● ● ● ● ● ●
6 ○ ● ○ ○ ○ ○ ○ ●
7 ○ ● ○ ○ ○ ○ ○ ●
8 ○ ● ○ ○ ○ ○ ○ ●

(a) ASCII to EBCDIC (b) Character generation (c) Dot pattern in the ROM

Fig. 10.12 Code conversion

5. A ROM and a DAC can be used to generate sine waves, sawtooth waves, triangular waves and square waves.
6. A ROM can be used to implement any or a set of logic expressions and is therefore helpful in the design of combinational circuits.