Chapter 10 Memory and Storage

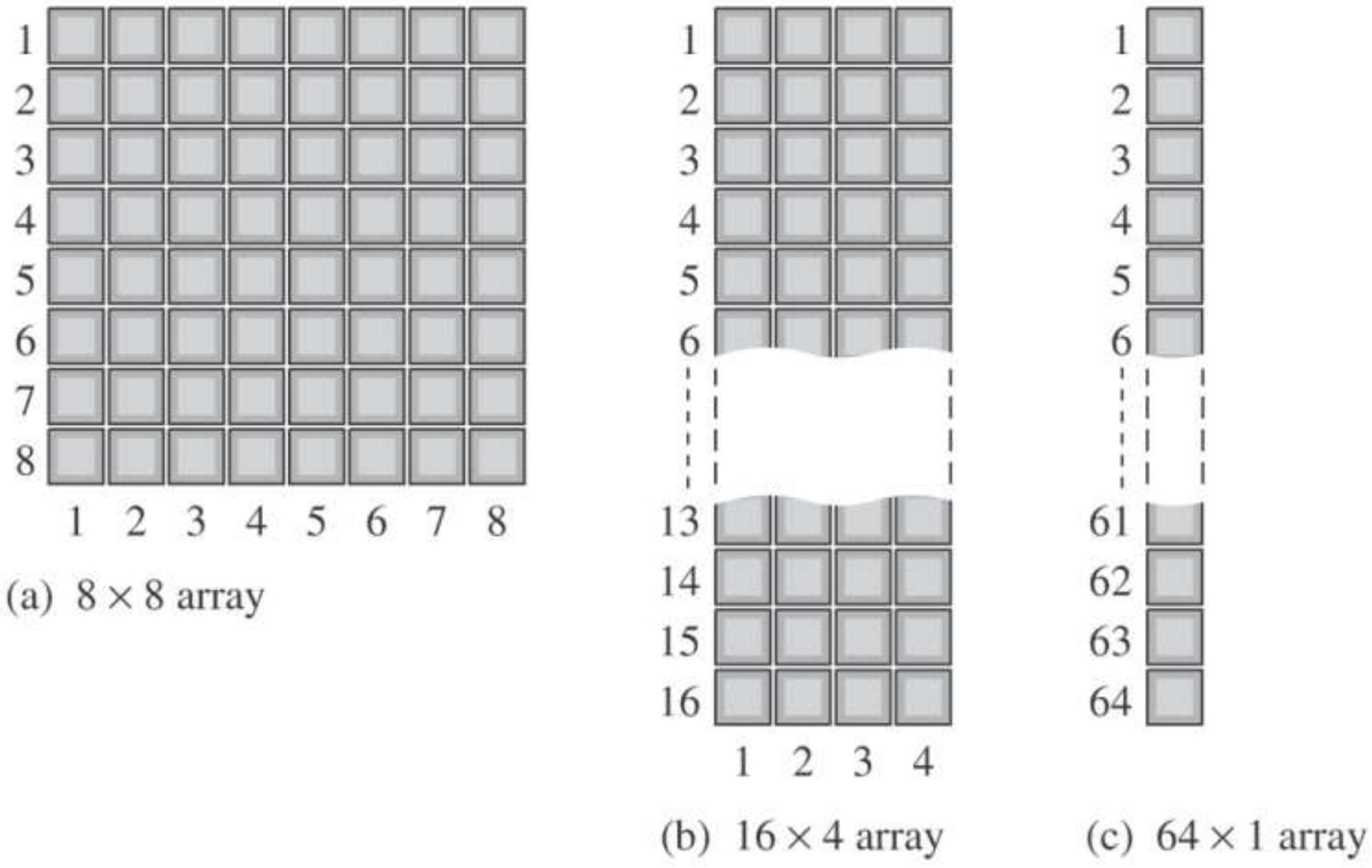
Outline

- Fundamentals of Semiconductor Memory
- Random-Access Memories (RAM)
- Read-Only Memories (ROM)
- Programmable ROMs (PROMs)
- Flash Memories
- Memory Expansion
- Special Types of Memories

10.1 Fundamentals of Semiconductor Memory

- Units of binary data
  - Bits: the smallest unit of binary data
  - Bytes: 8-bit units
  - Nibbles: 4-bit units
  - Words: a complete unit of information; one or more bytes

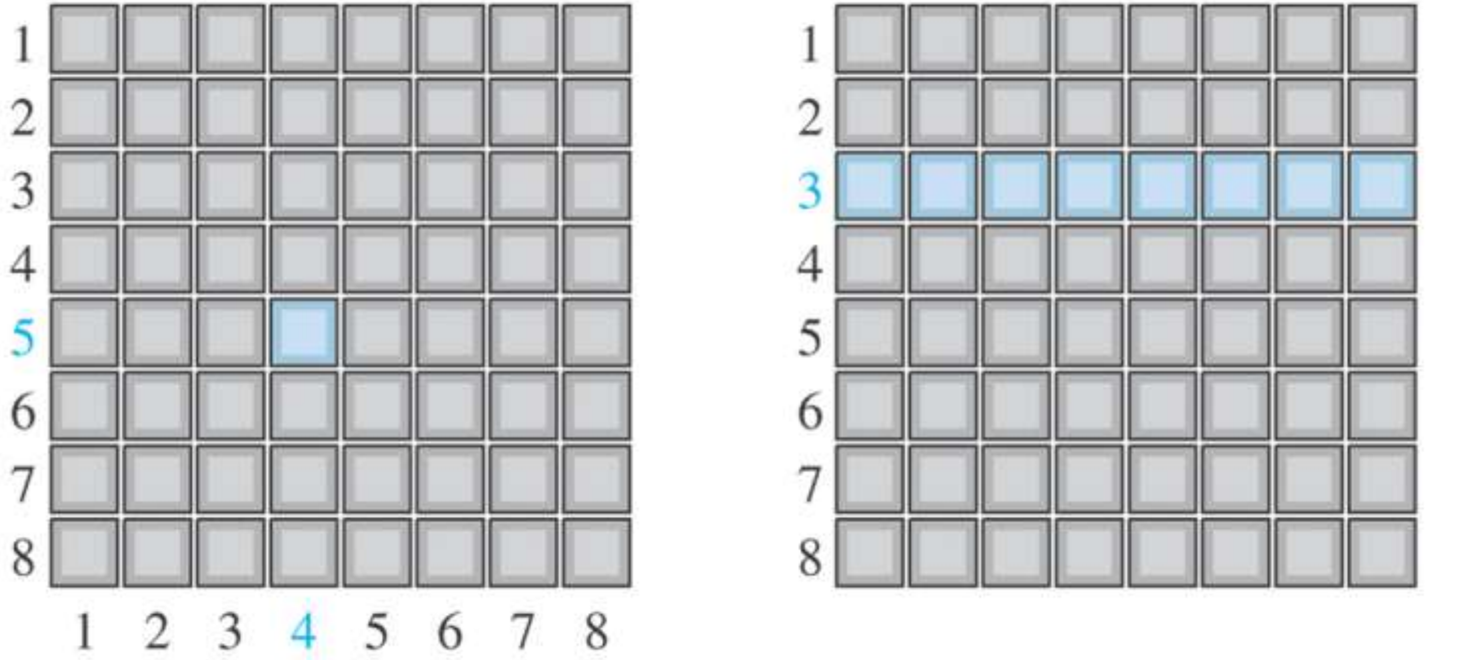
Figure 10–1 A 64-cell memory array organized in three different ways.



Notes

- A memory is identified by the number of words it can store times the word size
  - 16K×8 : 16384 words of eight bits each
- The actual number of words is always a power of 2
  - 2^{16}=65536

Memory Address and Capacity

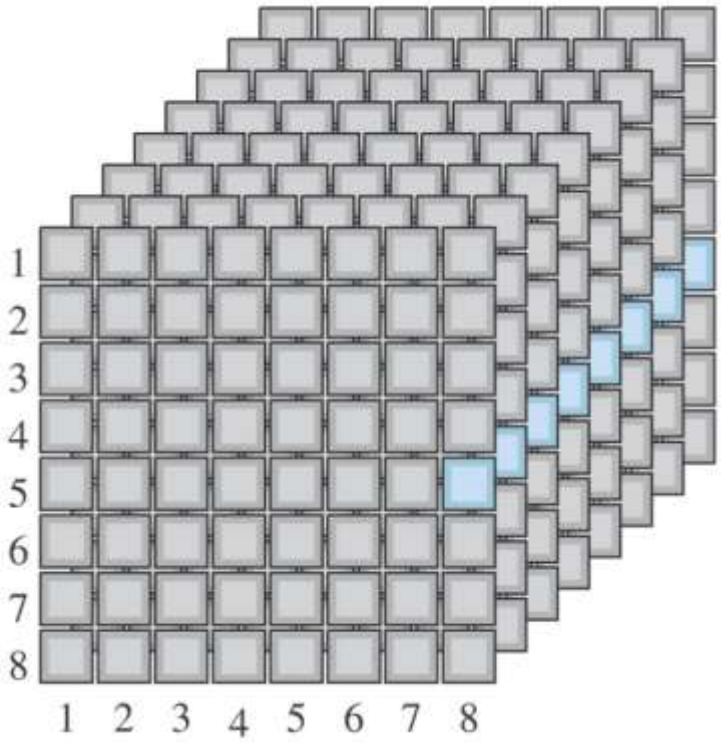


(a) The address of the blue bit is row 5, column 4.

(b) The address of the blue byte is row 3.

Figure 10–2 Examples of memory address in a 2-dimensional array.

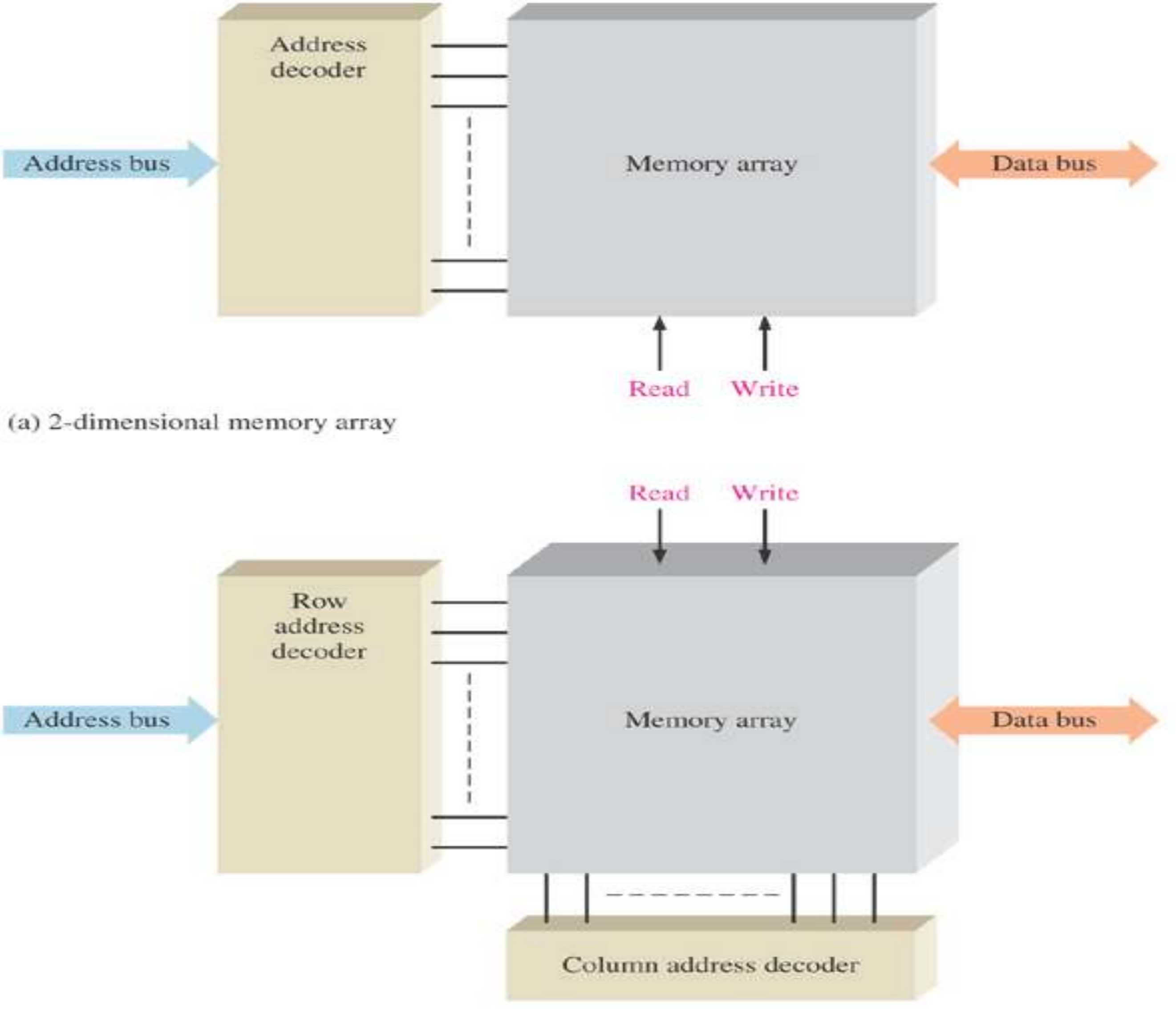
Figure 10–3 Example of memory address in a 3-dimensional array.



The address of the blue byte is row 5, column 8.

- The location of a unit of data in a memory array is called its **address**.
- The **capacity** of a memory: the total number of data units that can be stored.

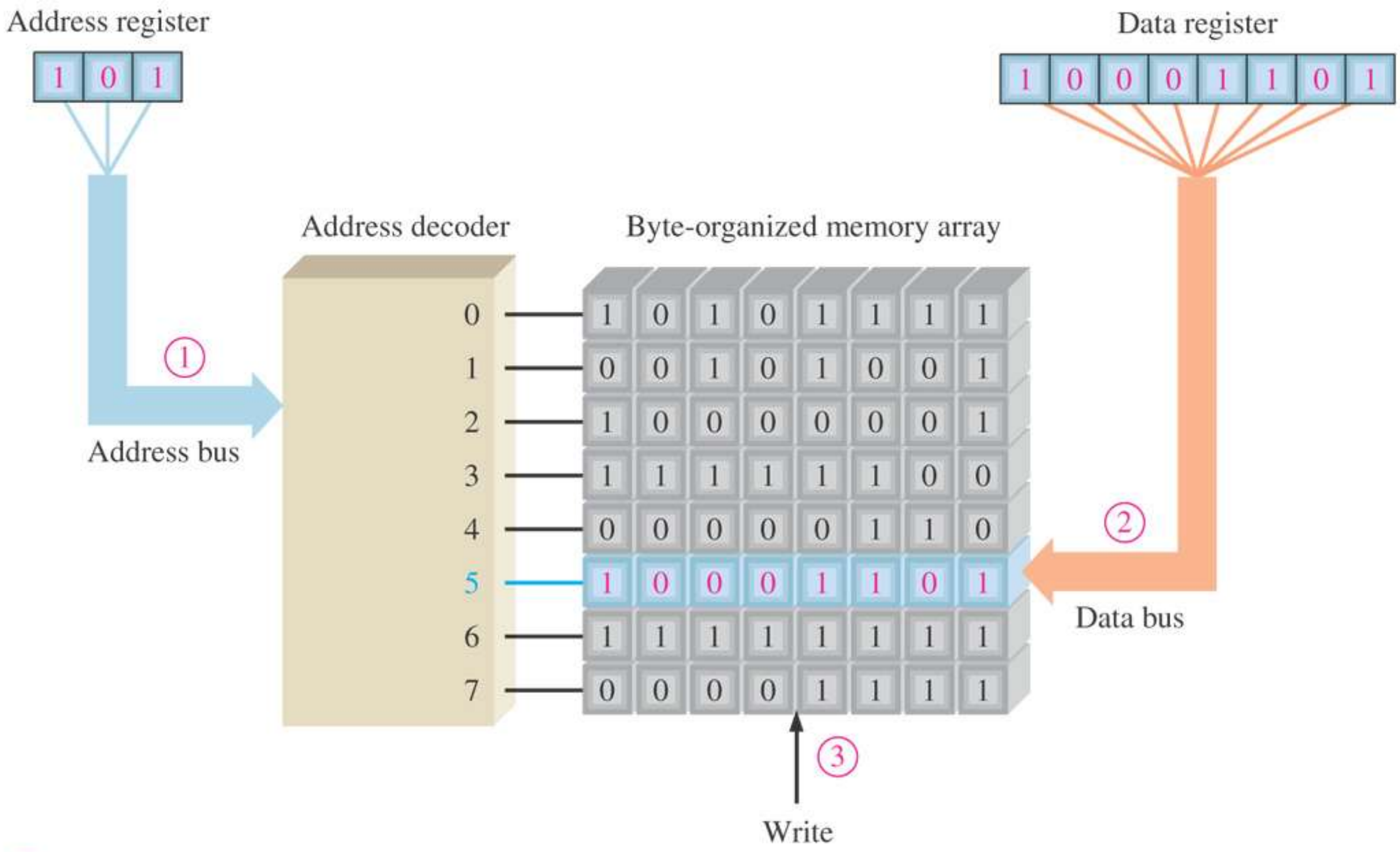
Basic memory operation



(a) 2-dimensional memory array

(b) 3-dimensional memory array

Write Operation

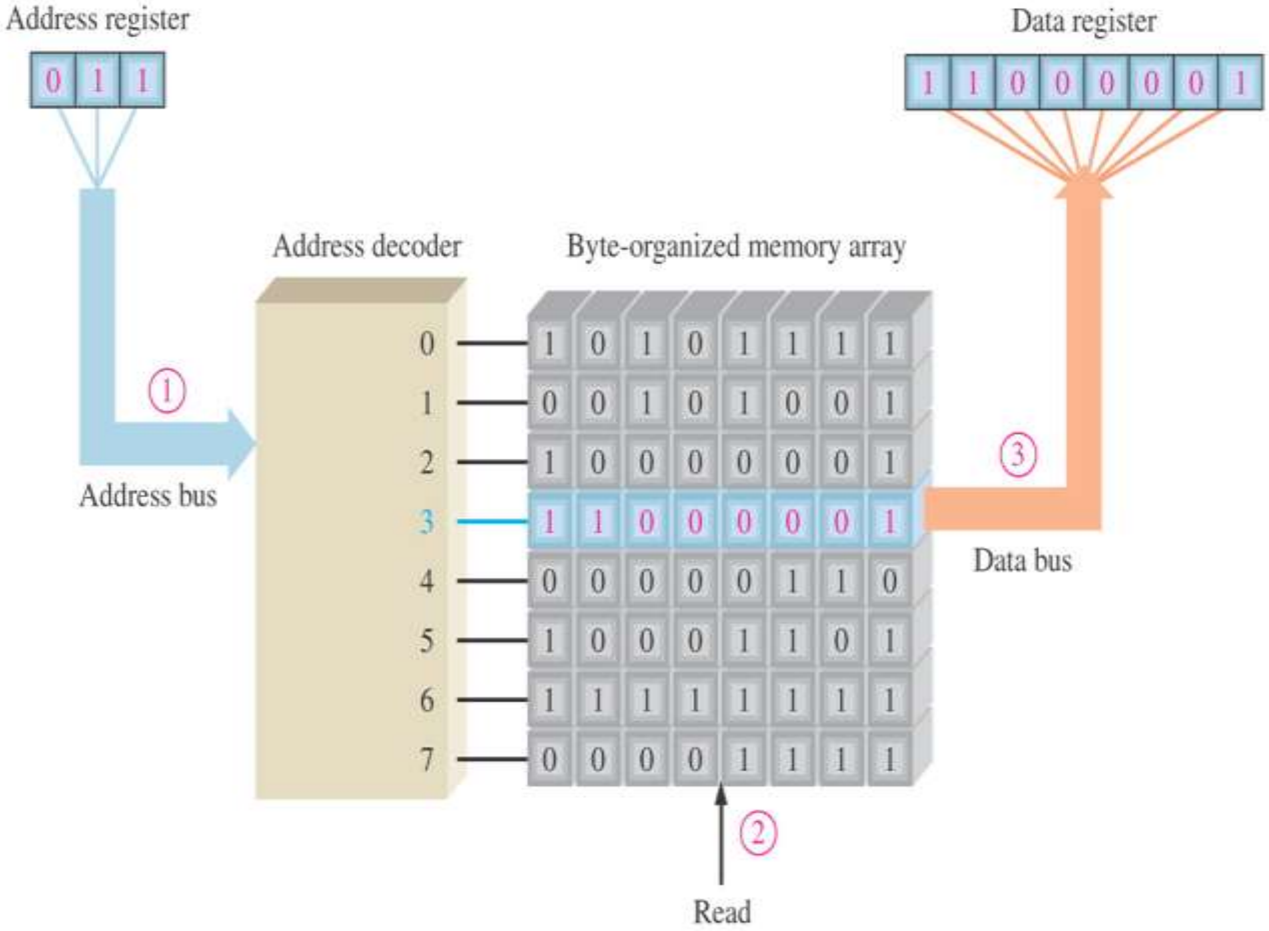


(1) Address code 101 is placed on the address bus and address 5 is selected.

(2) Data byte is placed on the data bus.

(3) Write command causes the data byte to be stored in address 5, replacing previous data.

Read Operation



0	1	0	1	0	1	1	1	1
1	0	0	1	0	1	0	0	1
2	1	0	0	0	0	0	0	1
3	1	1	0	0	0	0	0	1
4	0	0	0	0	0	1	1	0
5	1	0	0	0	1	1	0	1
6	1	1	1	1	1	1	1	1
7	0	0	0	0	1	1	1	1

1. Address code 011 is placed on the address bus and address 3 is selected.
2. Read command is applied.
3. The contents of address 3 is placed on the data bus and shifted into data register.
   The contents of address 3 is not erased by the read operation.

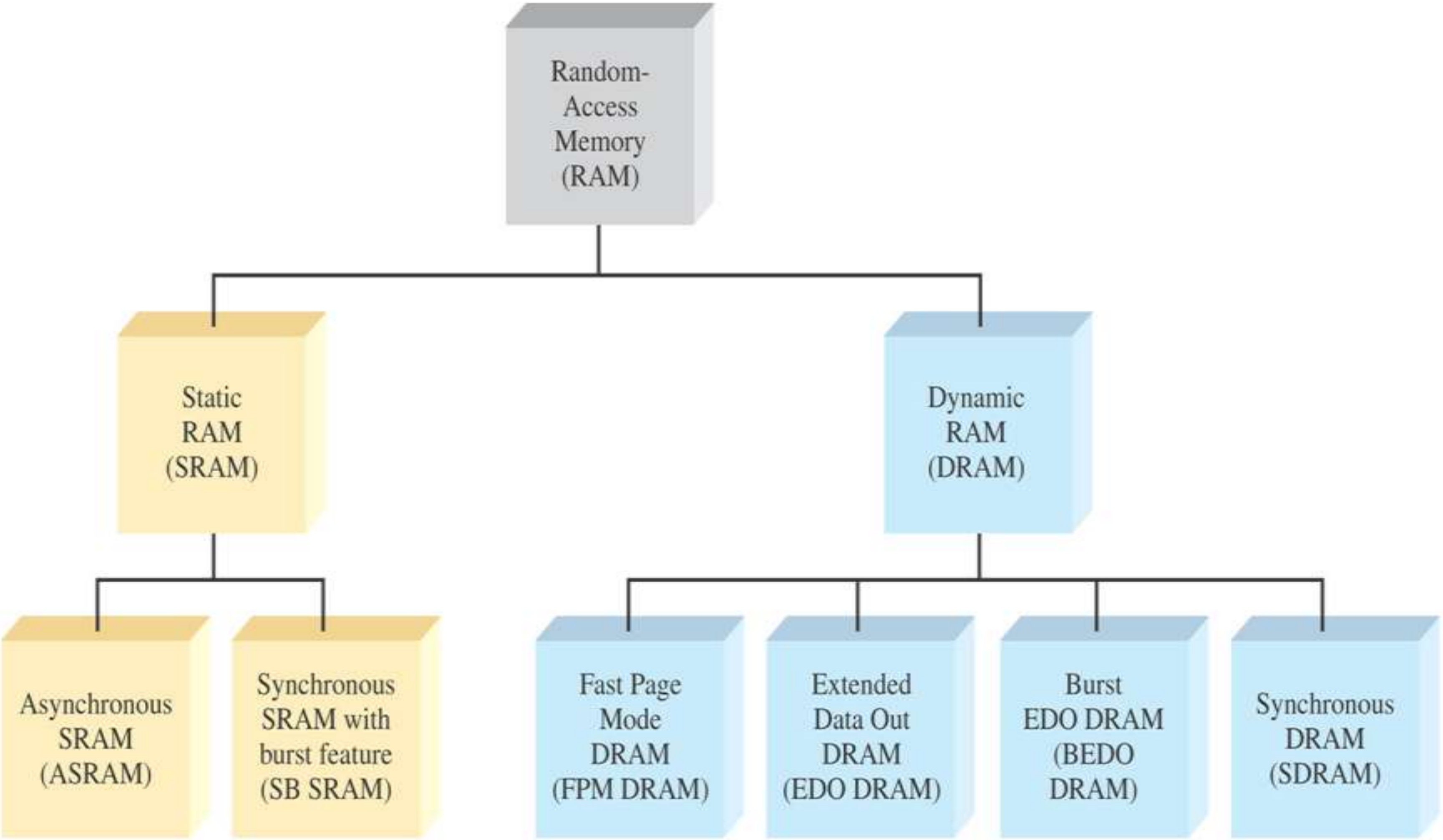
10.2 Random-Access Memories (RAMs)

- RAMs are read/write memories in which data can be written into or read from any selected address in any sequence.
- Two categories of RAM
  - Static RAM (SRAM)
    - use latches as storage element
    - Store data indefinitely as long as dc power is applied
  - Dynamic RAM (DRAM)
    - Use capacitors as storage elements
    - Need being recharged by a process called refreshing

SRAMs and DRAMs

- Data can be read much faster from SRAMs than from DRAMs
- DRAMs can store much data than SRAMs for a given physical size and cost
- CPU Cache: SRAM, Computer Memory: DRAM

Figure 10–7 The RAM family.



Random-Access Memory (RAM)

Static RAM (SRAM)

Dynamic RAM (DRAM)

Asynchronous SRAM (ASRAM)

Synchronous SRAM with burst feature (SB SRAM)

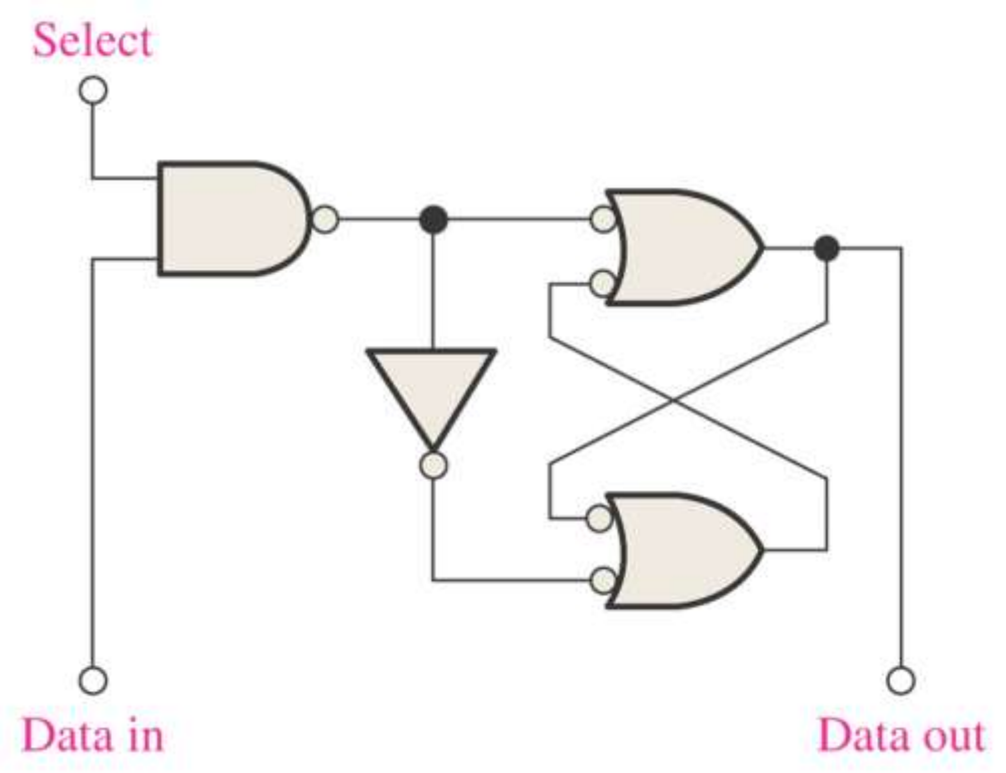
Fast Page Mode DRAM (FPM DRAM)

Extended Data Out DRAM (EDO DRAM)

Burst EDO DRAM (BEDO DRAM)

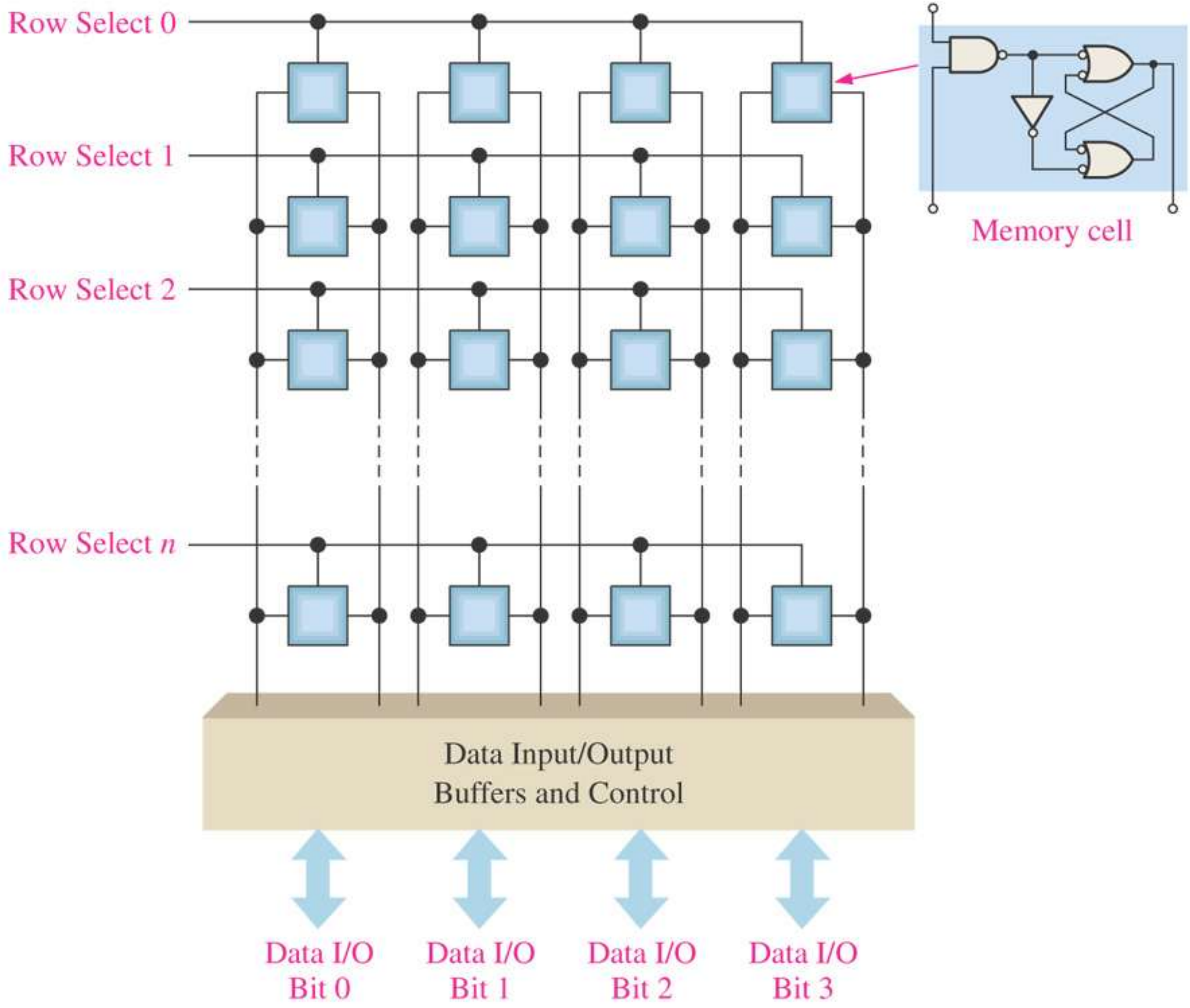
Synchronous DRAM (SDRAM)

Static RAMs (SRAMs)



A typical SRAM latch memory cell.

Basic SRAM array.



Logic diagram for an asynchronous 32k x 8 SRAM.

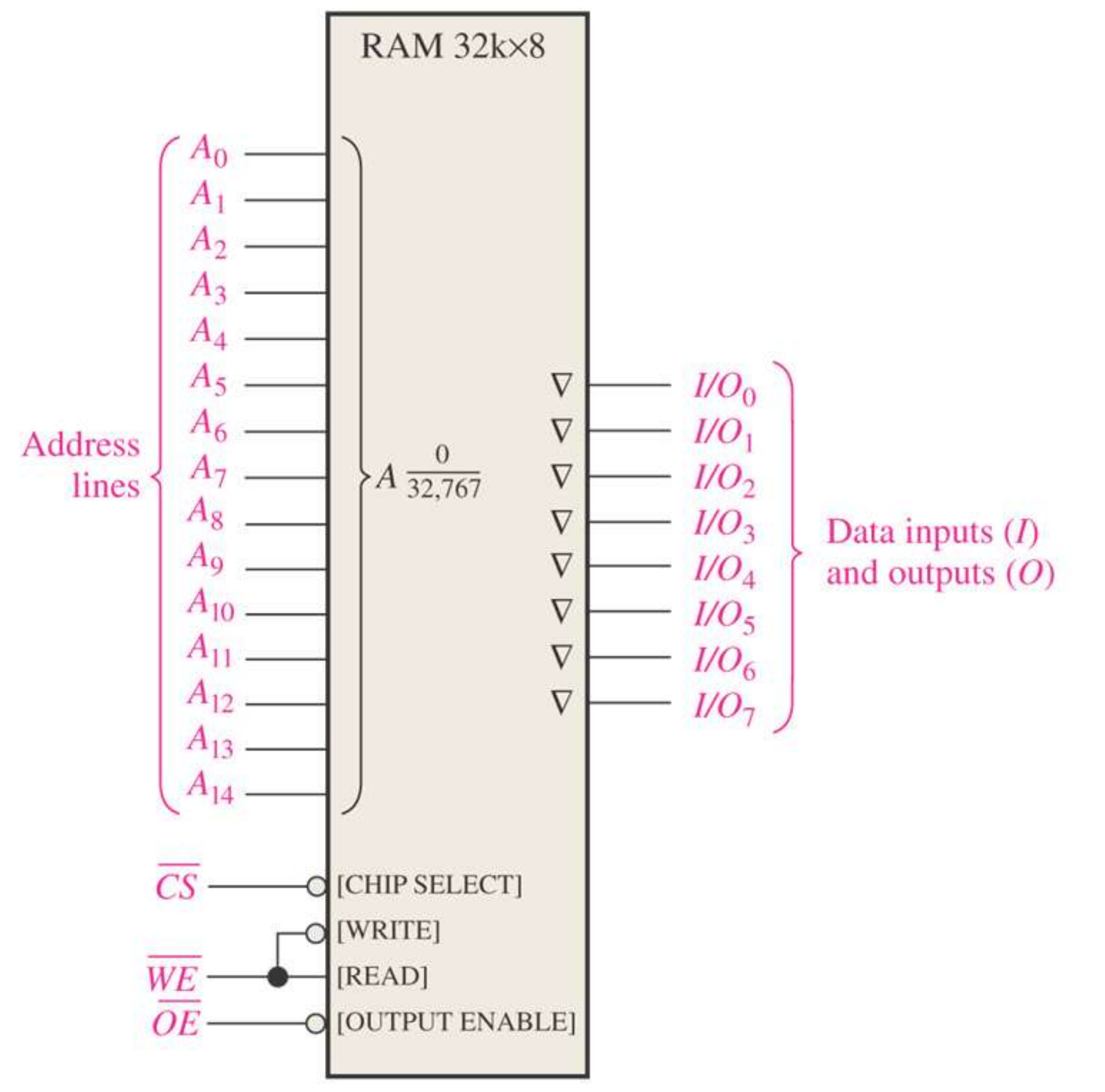
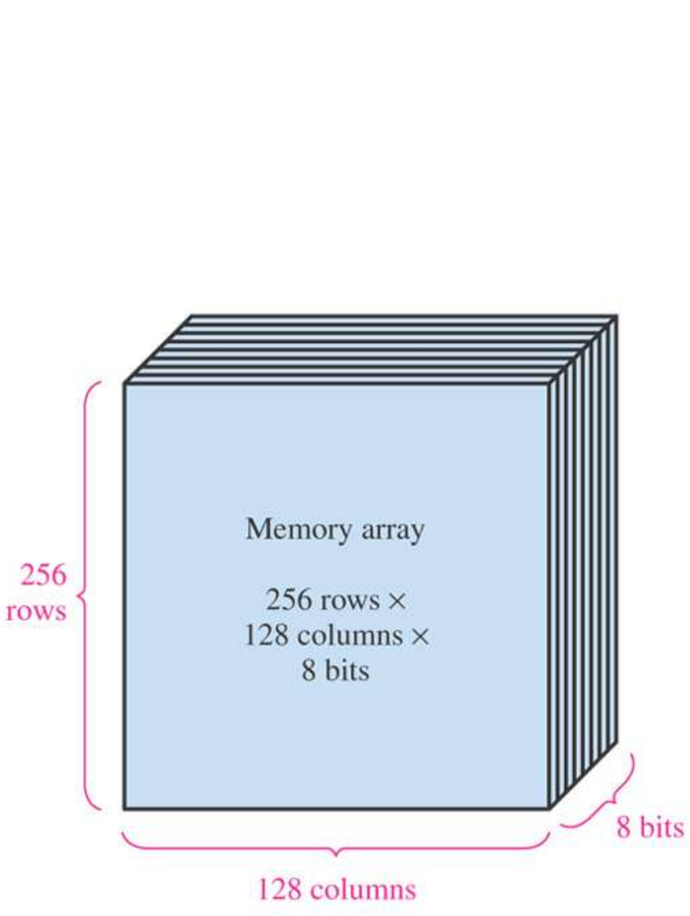
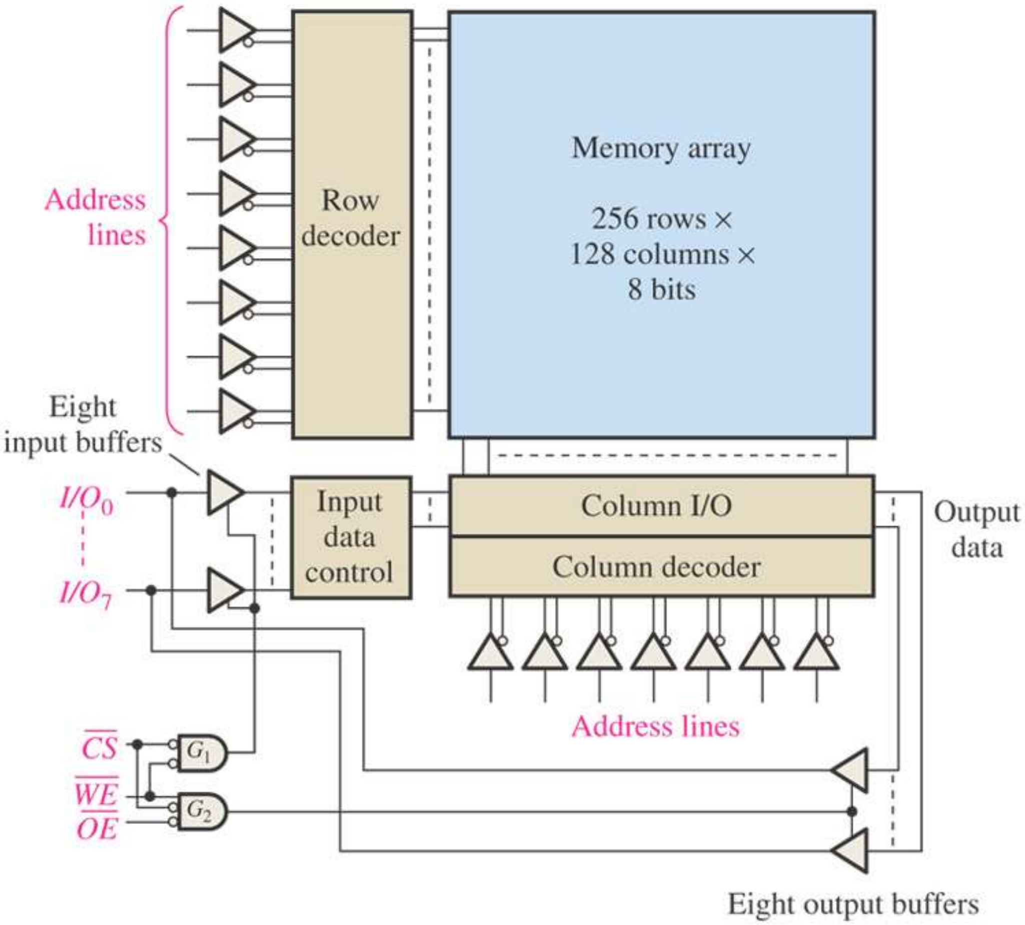


Figure 10–11 Basic organization of an asynchronous 32k x 8 SRAM.

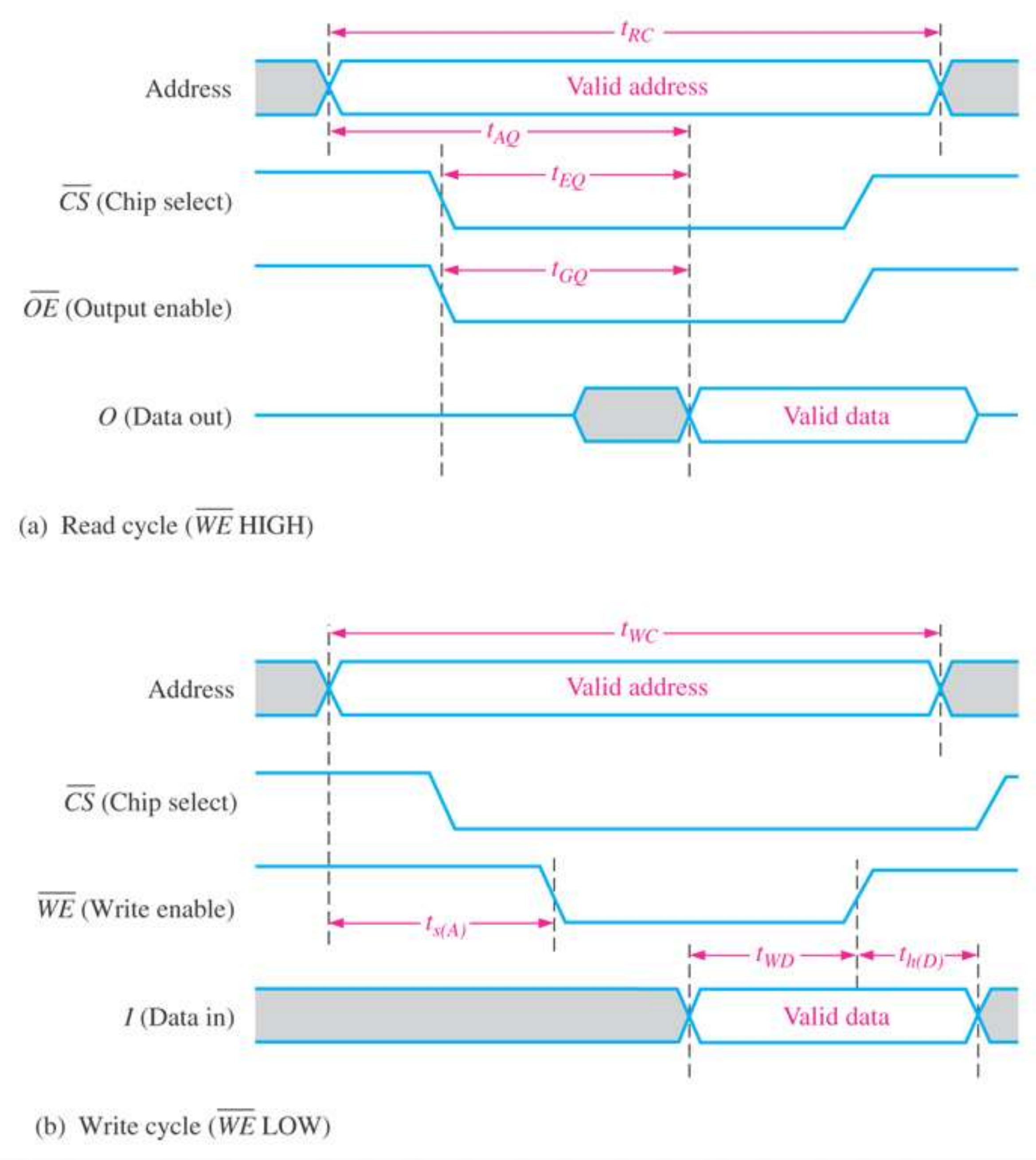


(a) Memory array configuration



(b) Memory block diagram

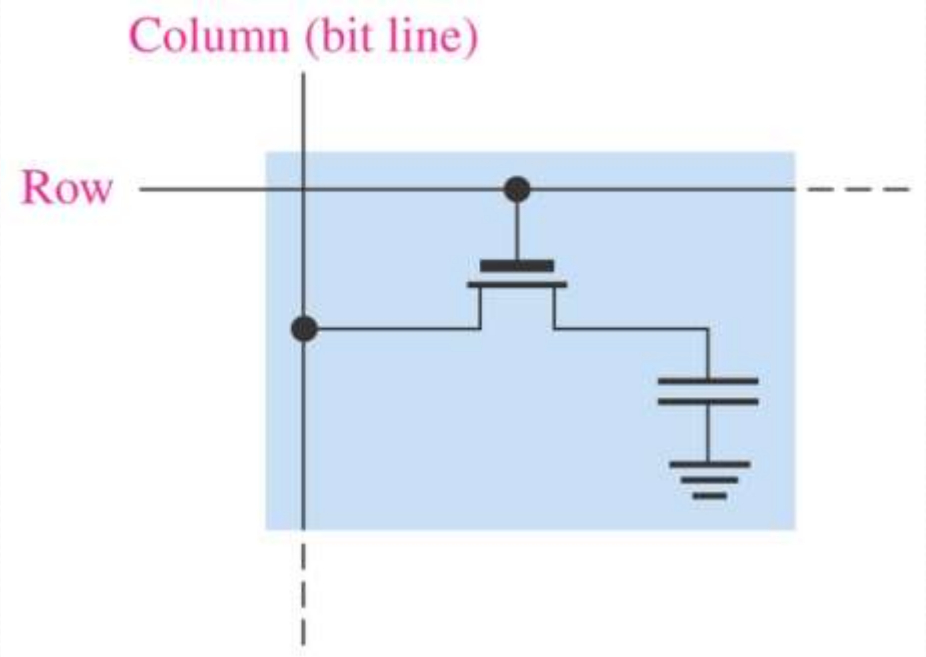
Figure 10–12 Basic read and write cycle timing for the SRAM in Figure 10–11.



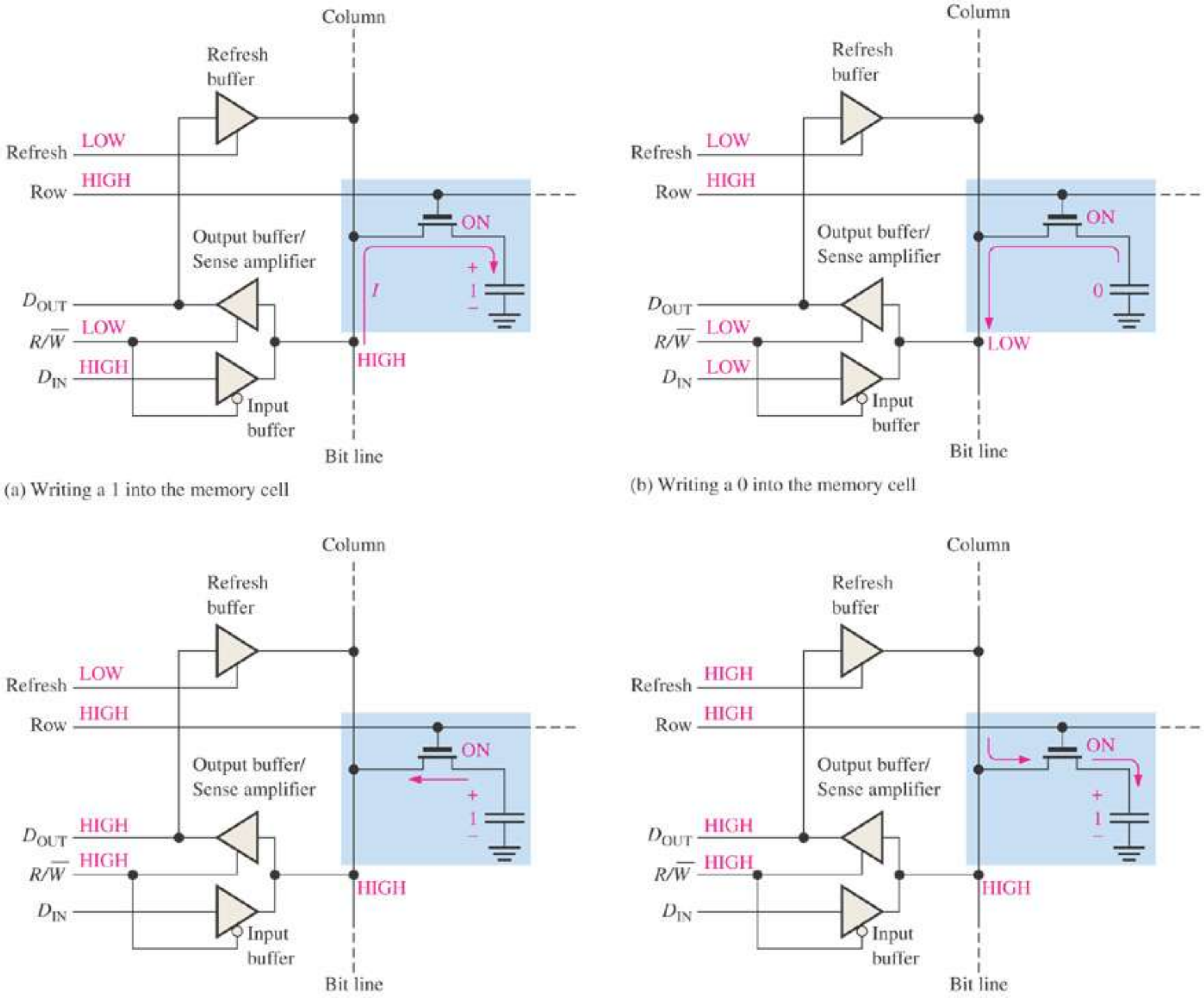
(a) Read cycle ($\overline{WE}$ HIGH)

(b) Write cycle ($\overline{WE}$ LOW)

Dynamic RAM (DRAM) Memory Cells



Basic operation of a DRAM cell.



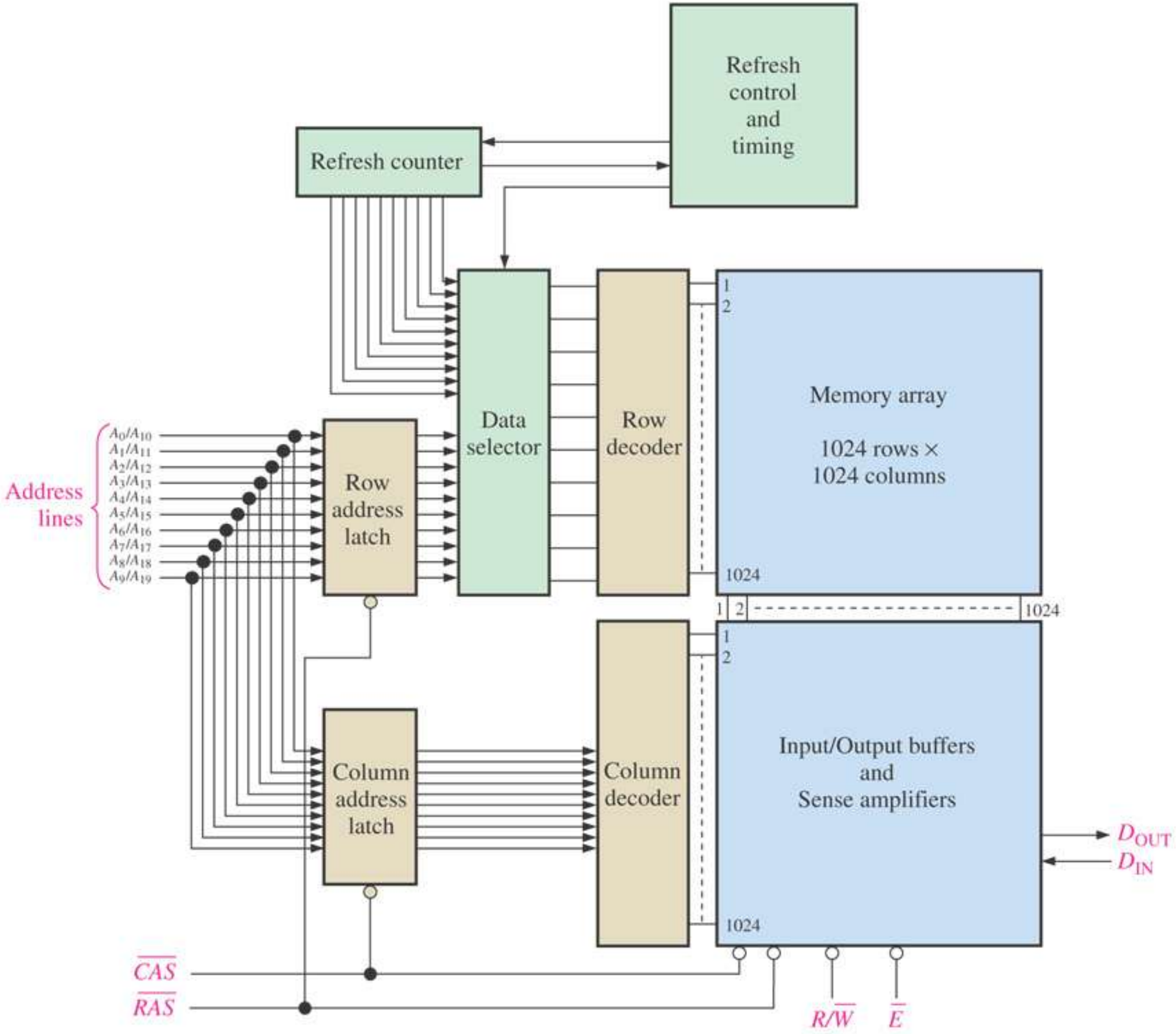
(a) Writing a 1 into the memory cell

(b) Writing a 0 into the memory cell

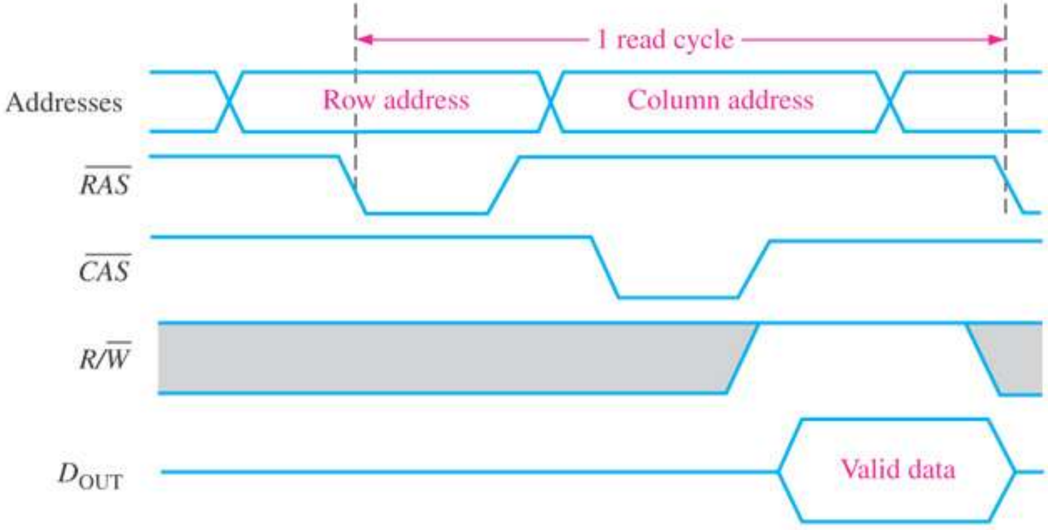
(c) Reading a 1 from the memory cell

(d) Refreshing a stored 1

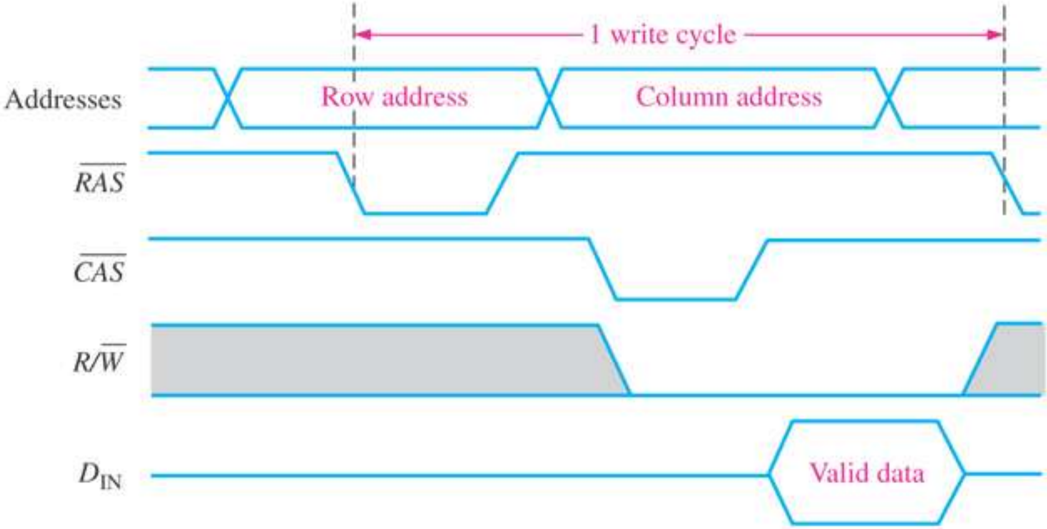
Simplified block diagram of a 1M x 1 DRAM.



Normal read and write cycle timing.



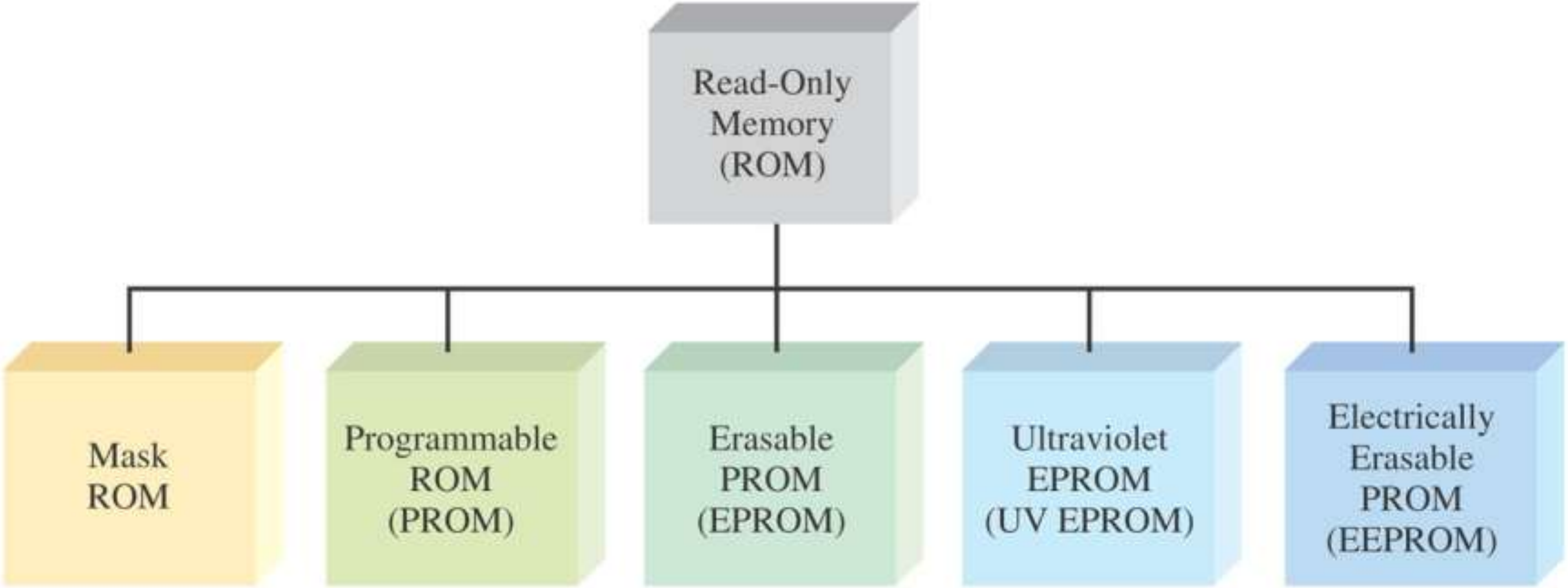
(a) Read cycle



(b) Write cycle

10.3 Read-Only Memories (ROMs)

- Store data permanently or semi-permanently
- Stored data can be read from the memory
- Stored data can not be changed at all or cannot be changed without specialized equipment.



Read-Only Memory (ROM)

Mask ROM

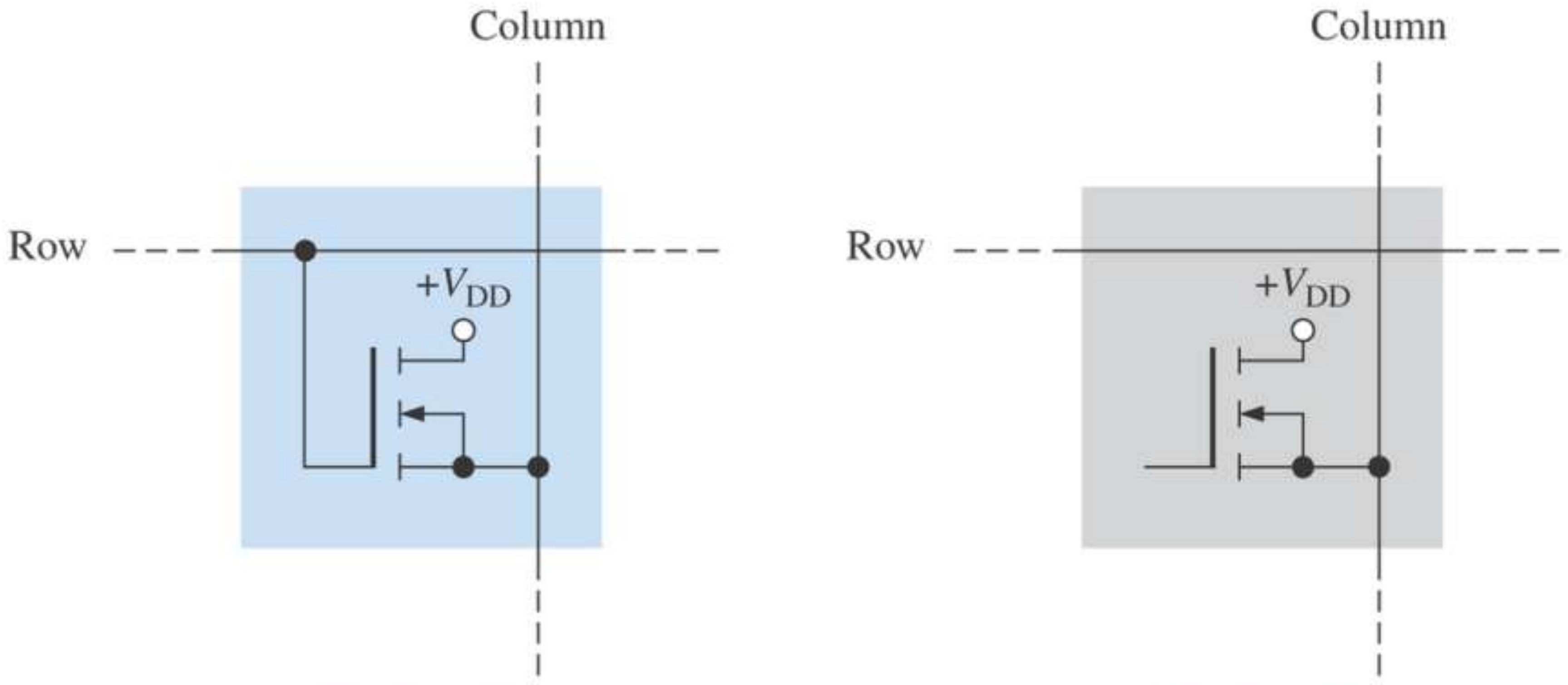
Programmable ROM (PROM)

Erasable PROM (EPROM)

Ultraviolet EPROM (UV EPROM)

Electrically Erasable PROM (EEPROM)

ROM cells



Storing a 1

Storing a 0

A representation of a 16 x 8-bit ROM array.

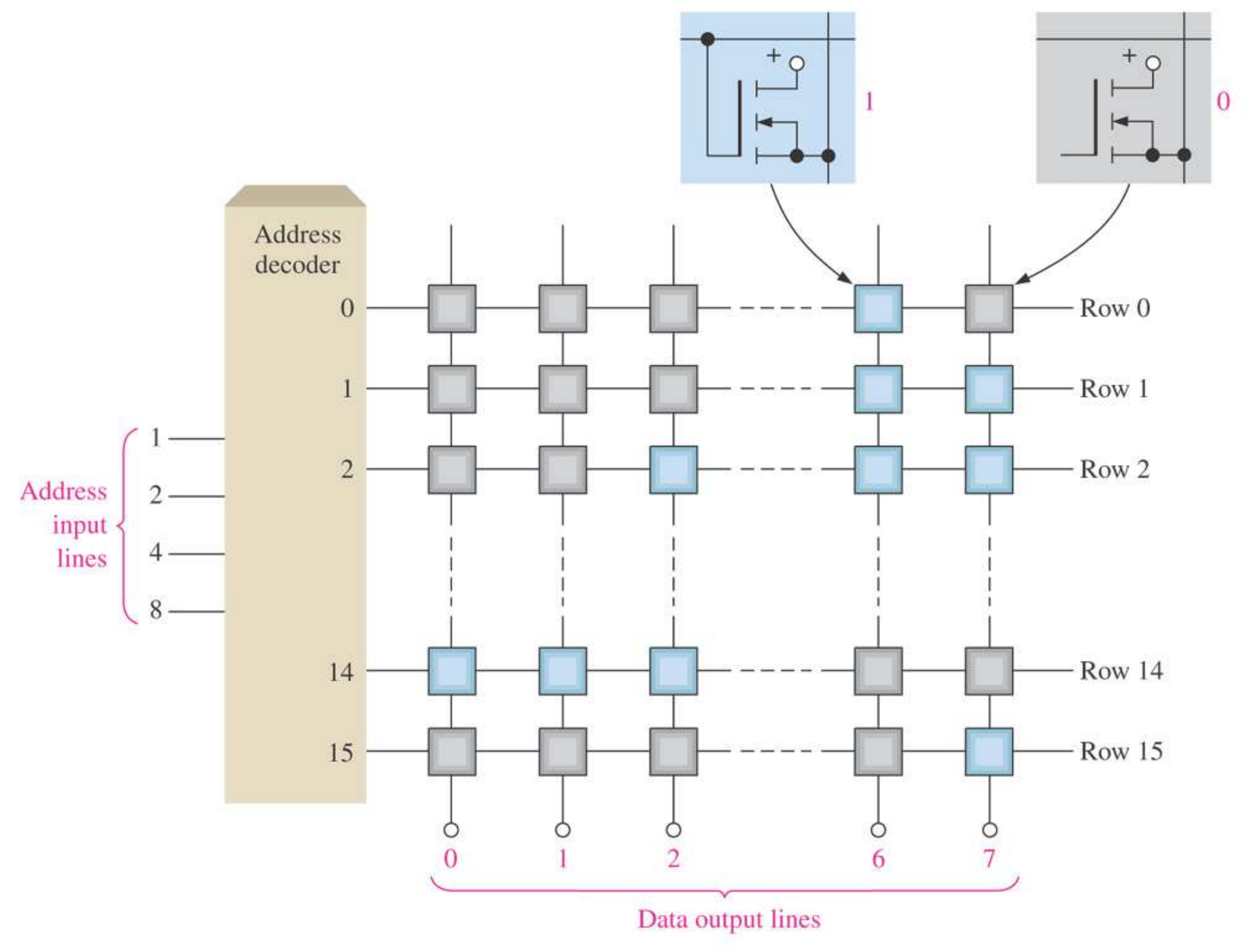


Figure 10–26 A 256 x 4 ROM logic symbol. The $A\frac{0}{255}$ designator means that the 8-bit address code selects addresses 0 through 255.

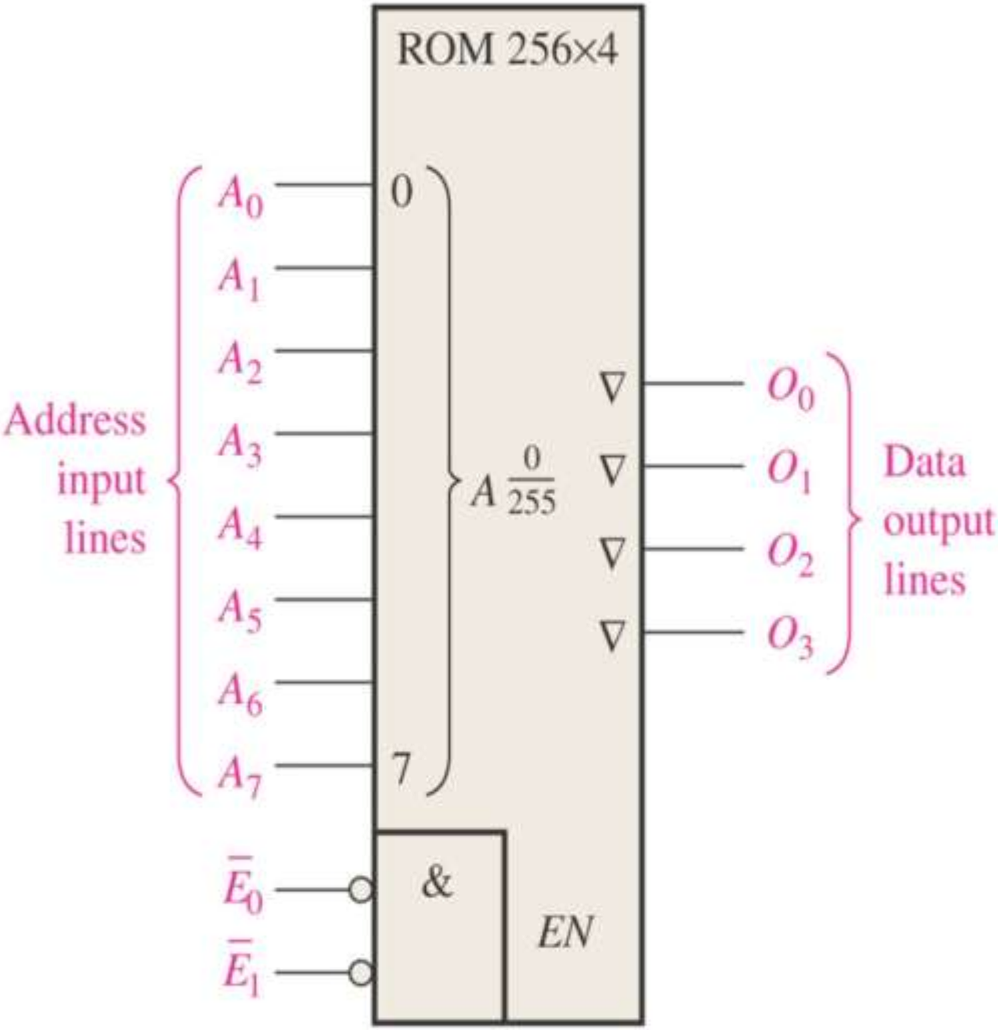


Figure 10–27 A 1024-bit ROM with a 256 x 4 organization based on a 32 x 32 array.

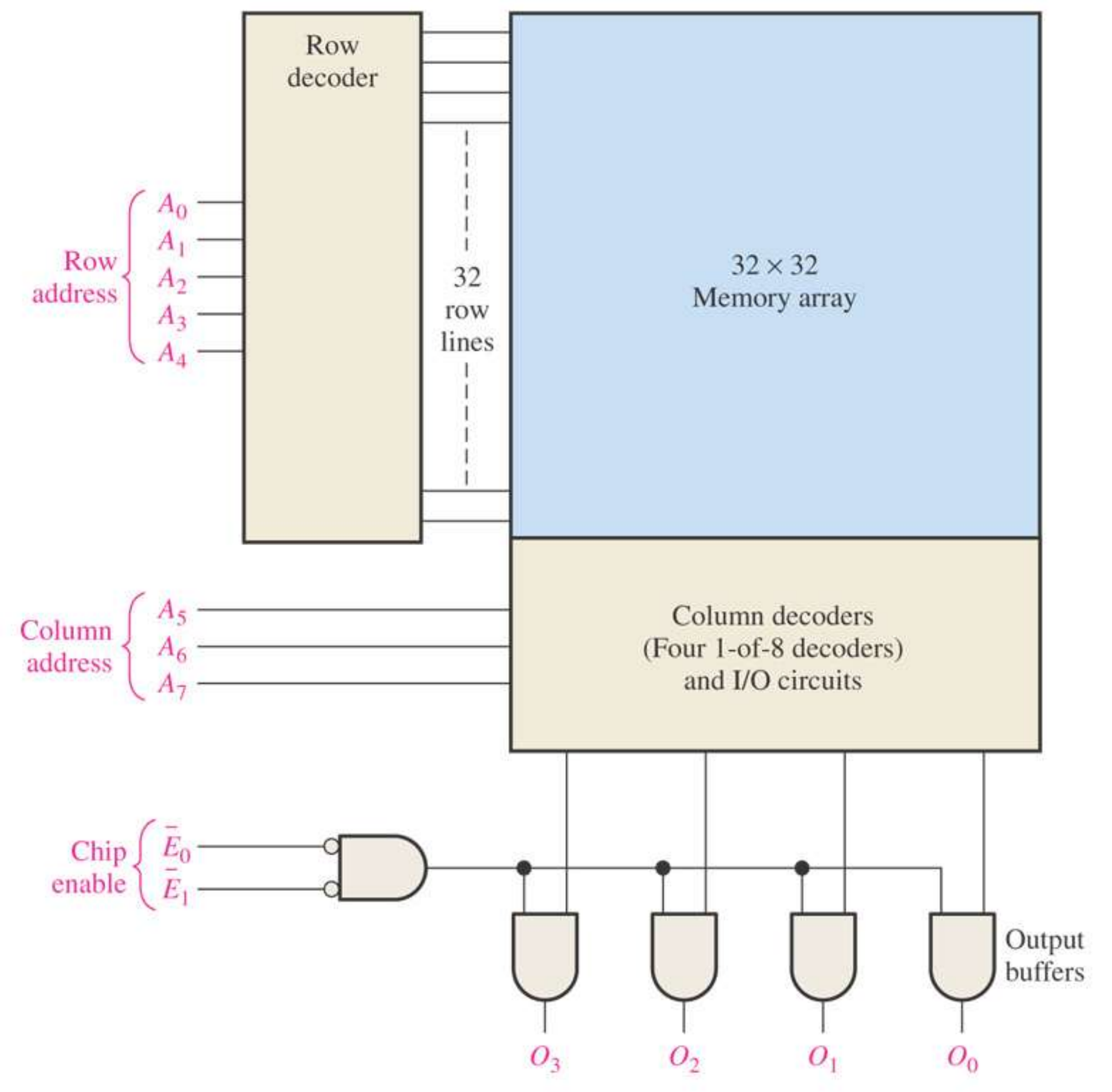
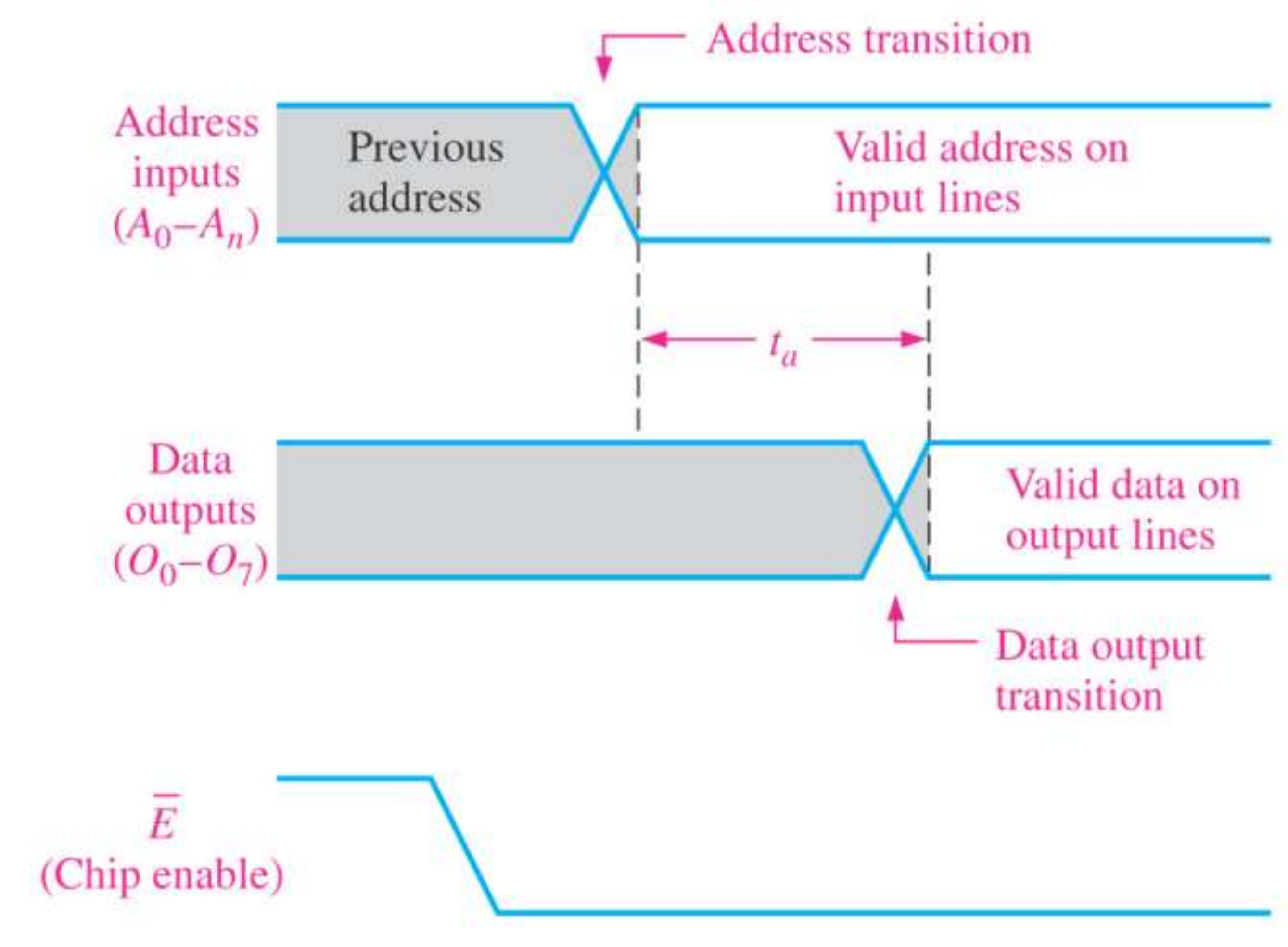


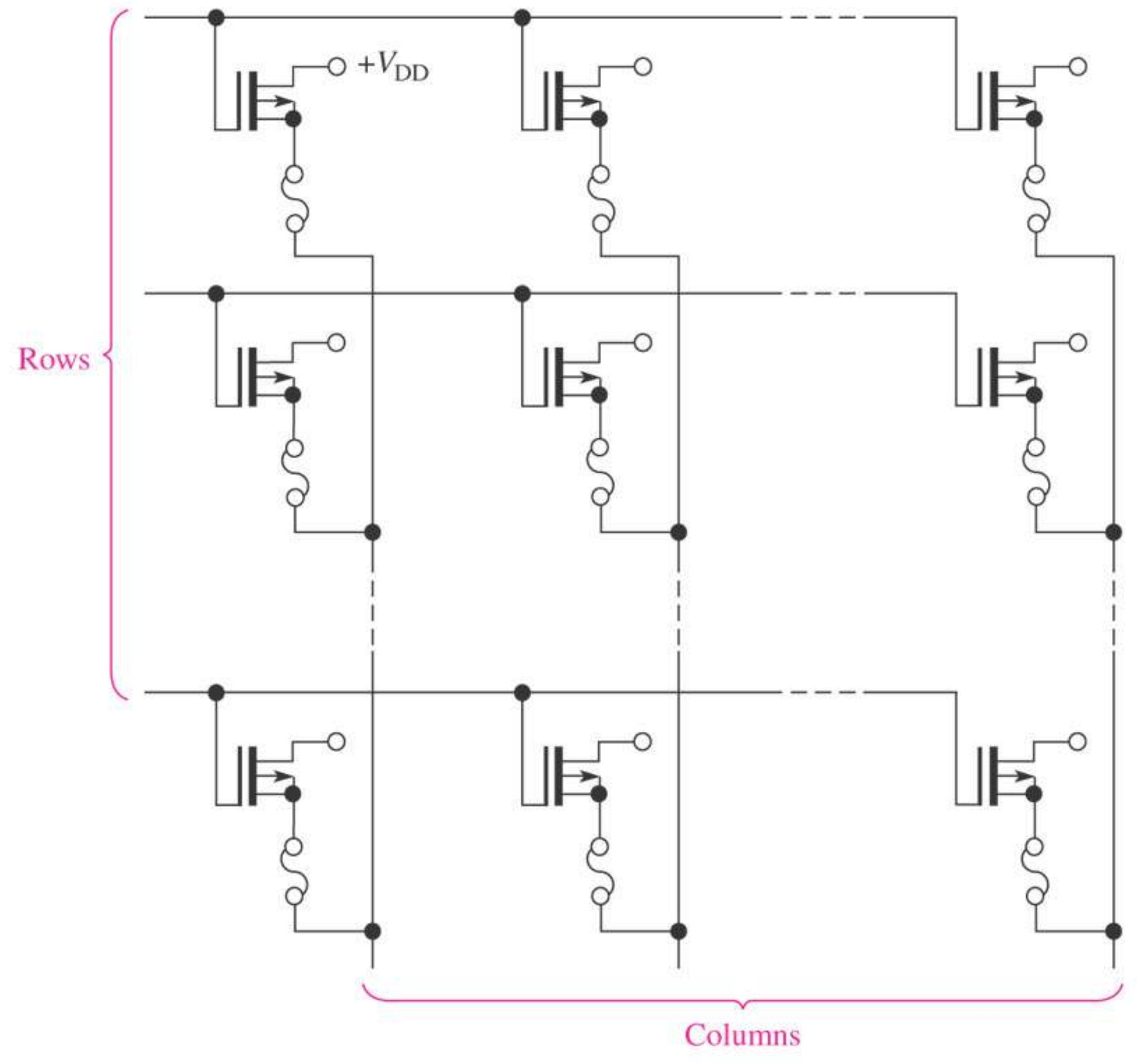
Figure 10–28 ROM access time (t_a) from address change to data output with chip enable already active.


Address transition
Address inputs (A_0–A_n)
Previous address
Valid address on input lines
t_a
Data outputs (O_0–O_7)
Valid data on output lines
Data output transition
$\bar{E}$ (Chip enable)

10.4 Programmable ROMs

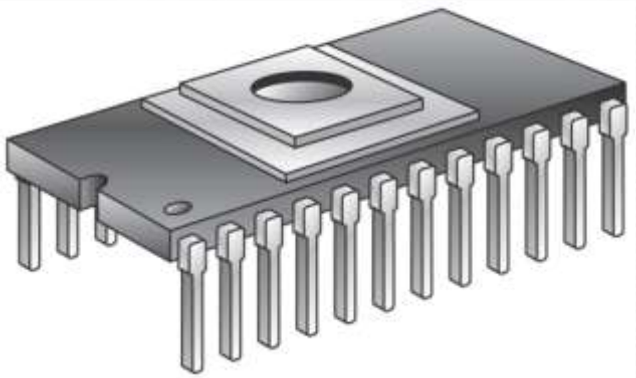
- PROMs
- EPROMs
- EEPROMs

Figure 10–29 MOS PROM array with fusible links. (All drains are commonly connected to V_{DD}.)

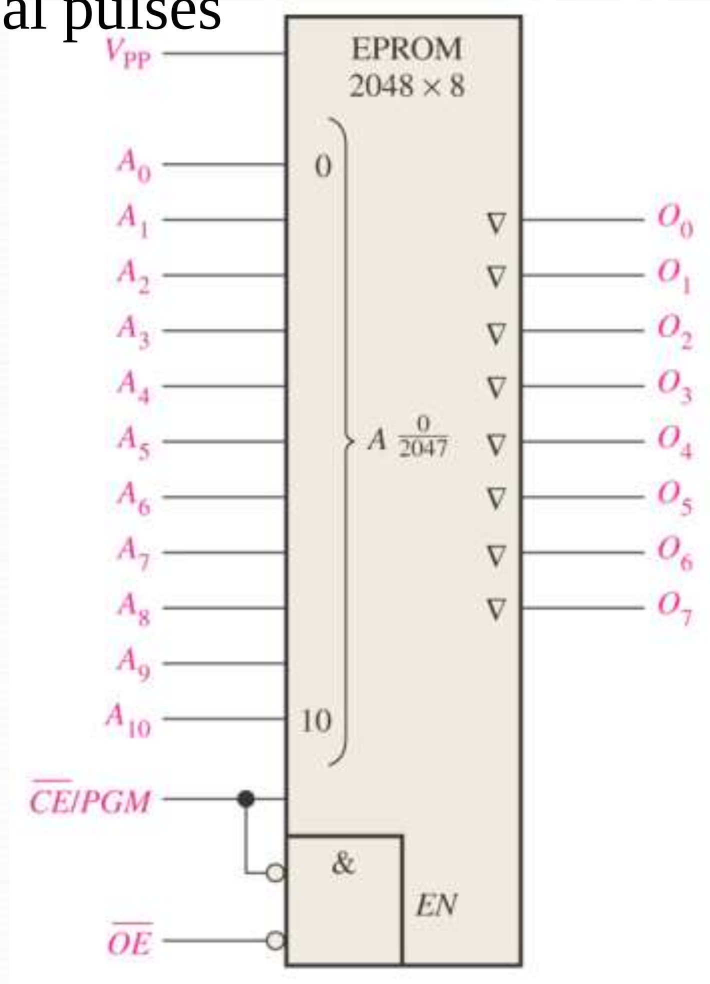


After the programming process, the link is left intact for a stored ‘1’.

EPROM : Erasable PROM (by ultraviolet)
EEPROM: Erasable by electrical pulses

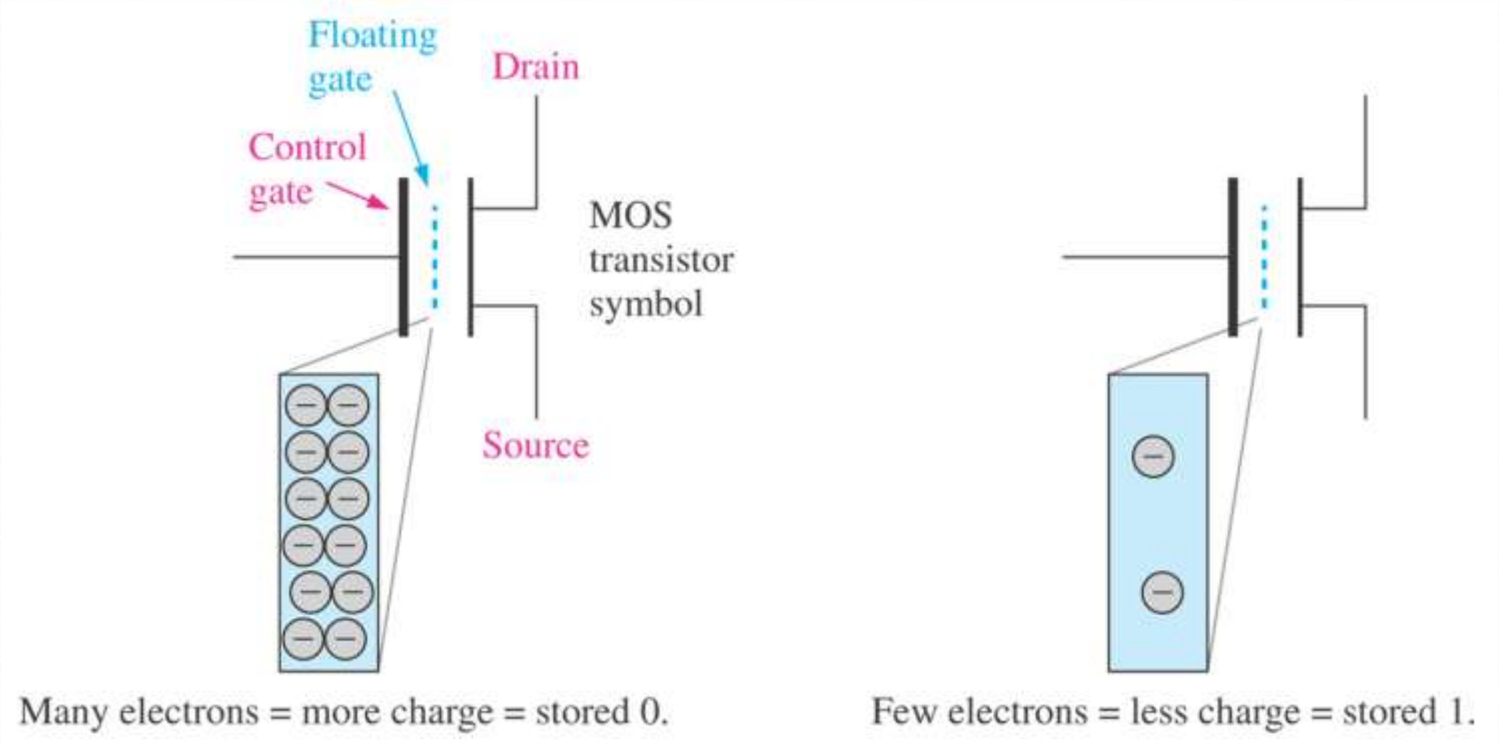


Ultraviolet erasable PROM package



The logic symbol for a 2048 x 8 UV EPROM

10.5 Flash Memories

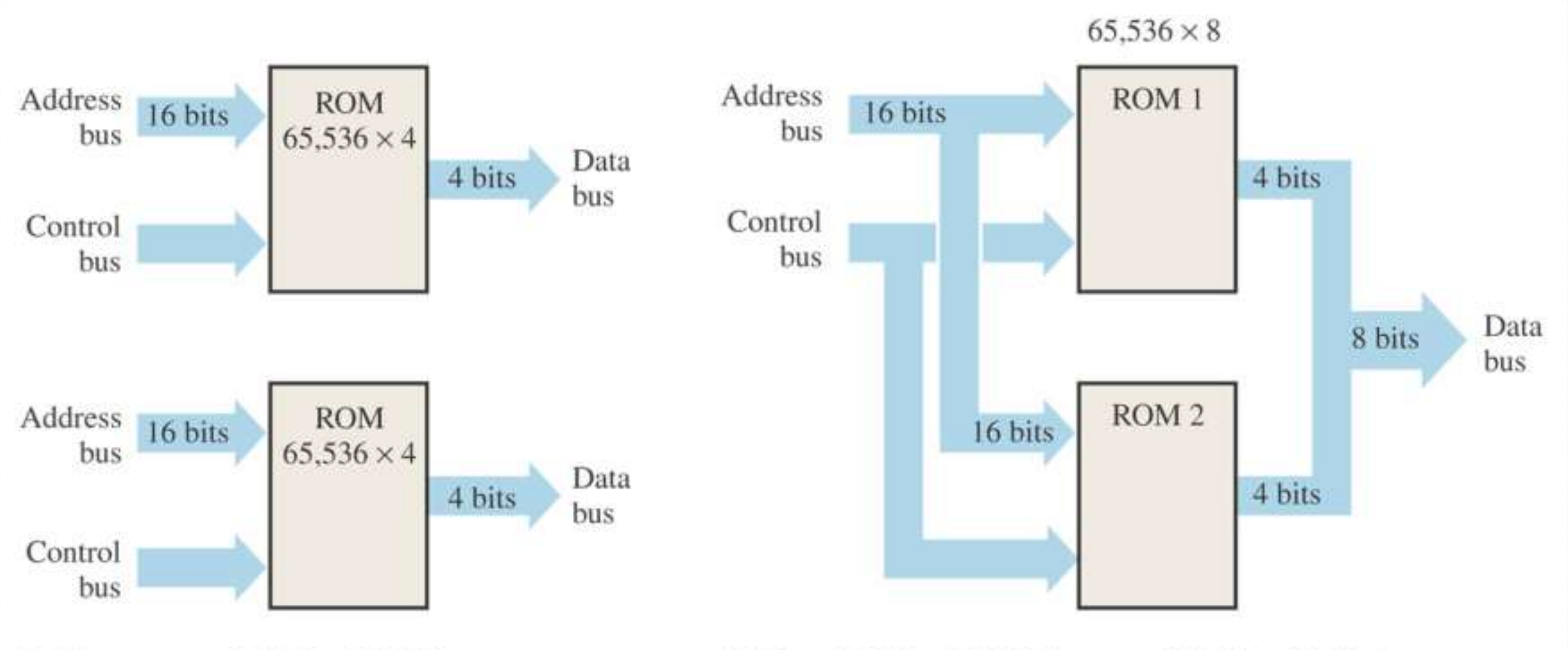

Floating gate
Drain
Control gate
MOS transistor symbol
Source
Many electrons = more charge = stored 0.
Few electrons = less charge = stored 1.

The storage cell in a flash memory.

10.6 Memory Expansion

- Word-length expansion
- Word-capacity expansion

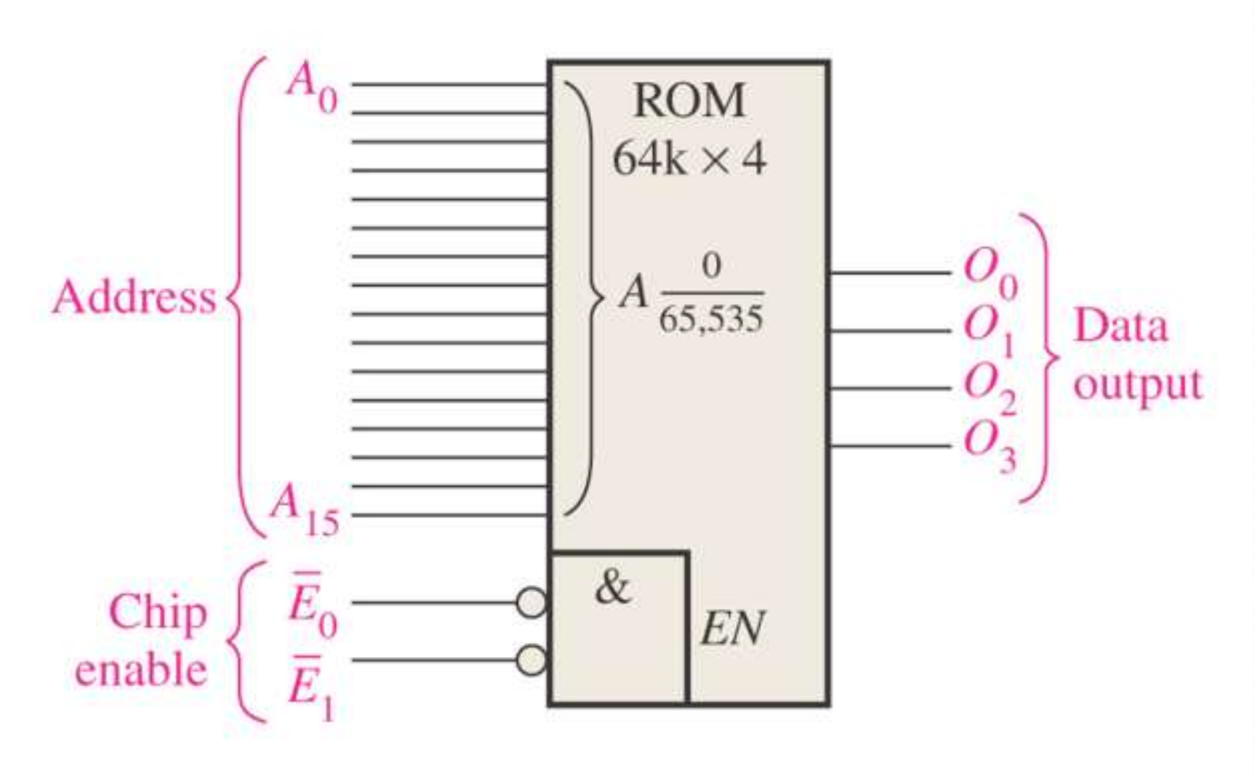
Figure 10–38 Expansion of two 65,536 x 4 ROMs into a 65,536 x 8 ROM to illustrate ***word-length expansion.***

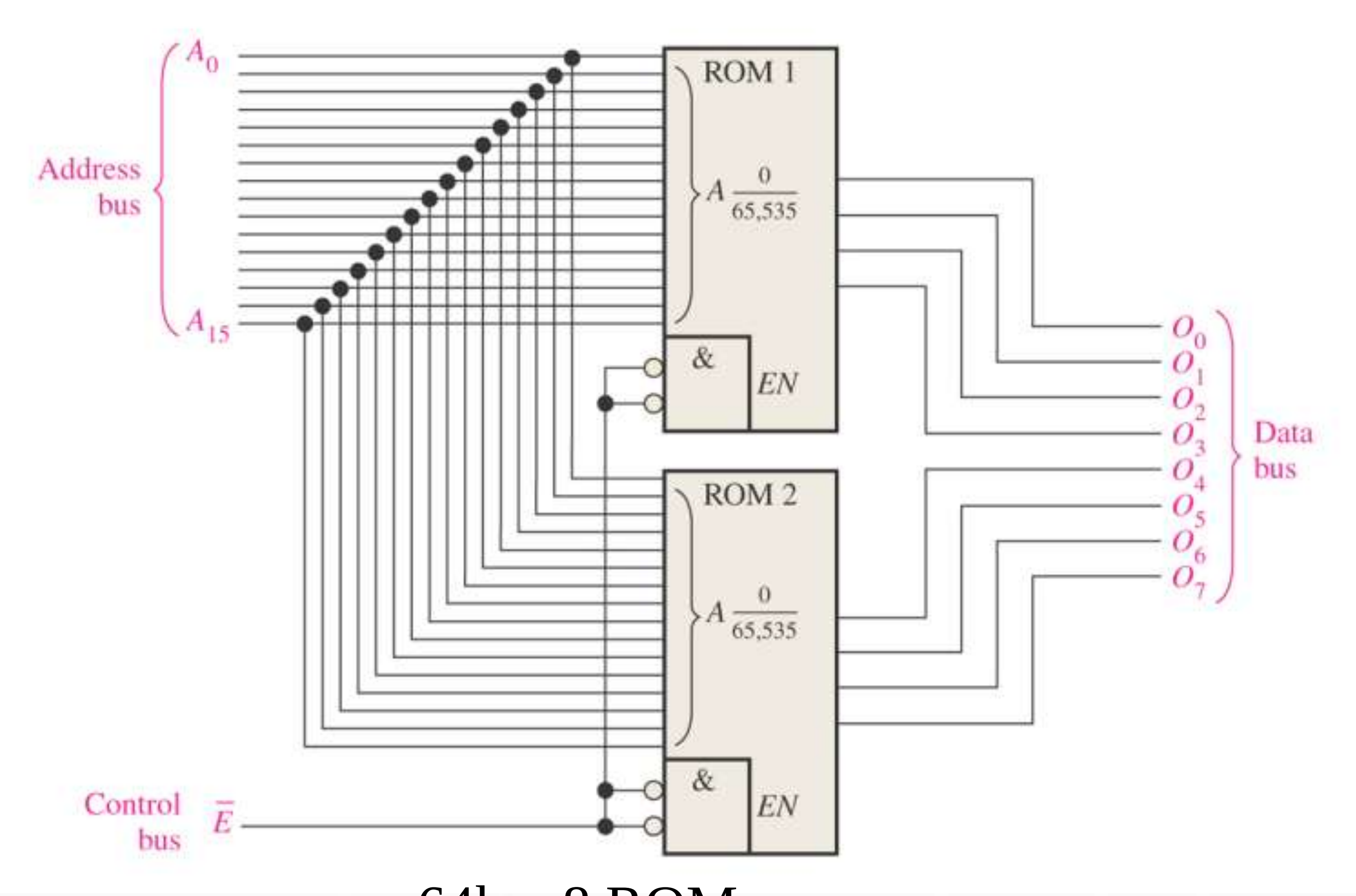


(a) Two separate 65,536 × 4 ROMs

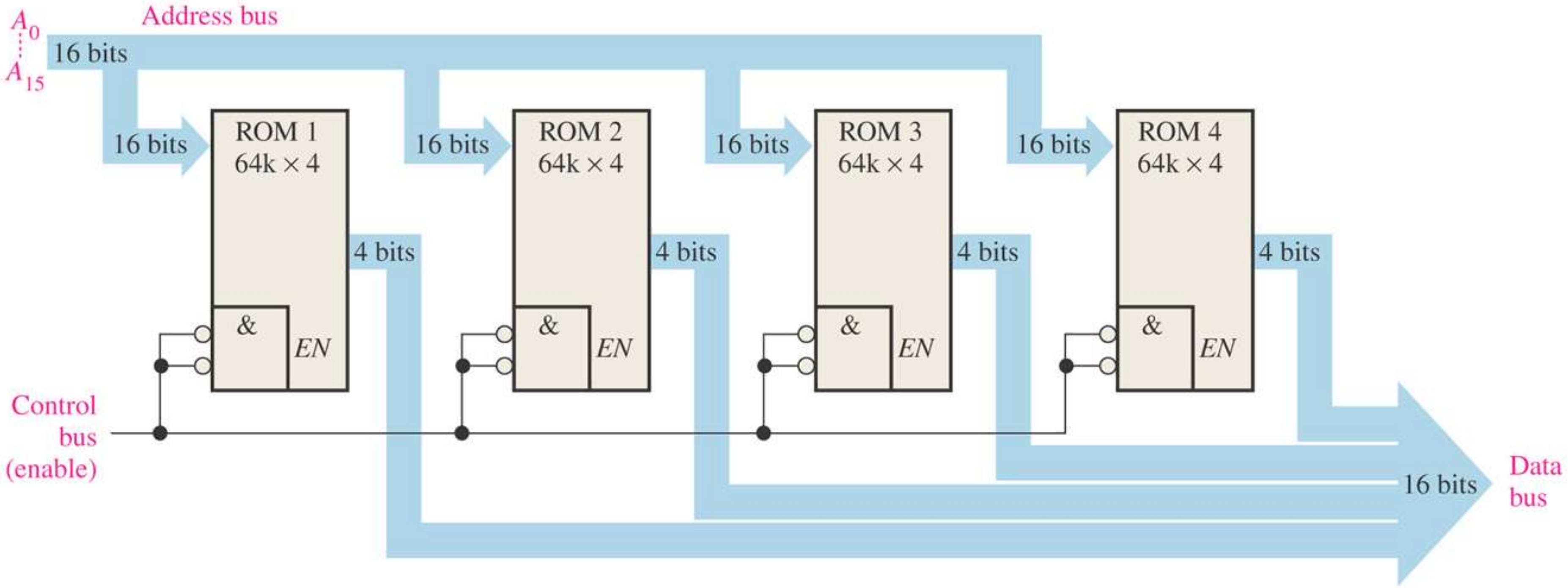
(b) One 65,536 × 8 ROM from two 65,536 × 4 ROMs

Example: Expand the 65536 x 4 ROM to form a 64k x 8 ROM and a 64k x 16 ROM



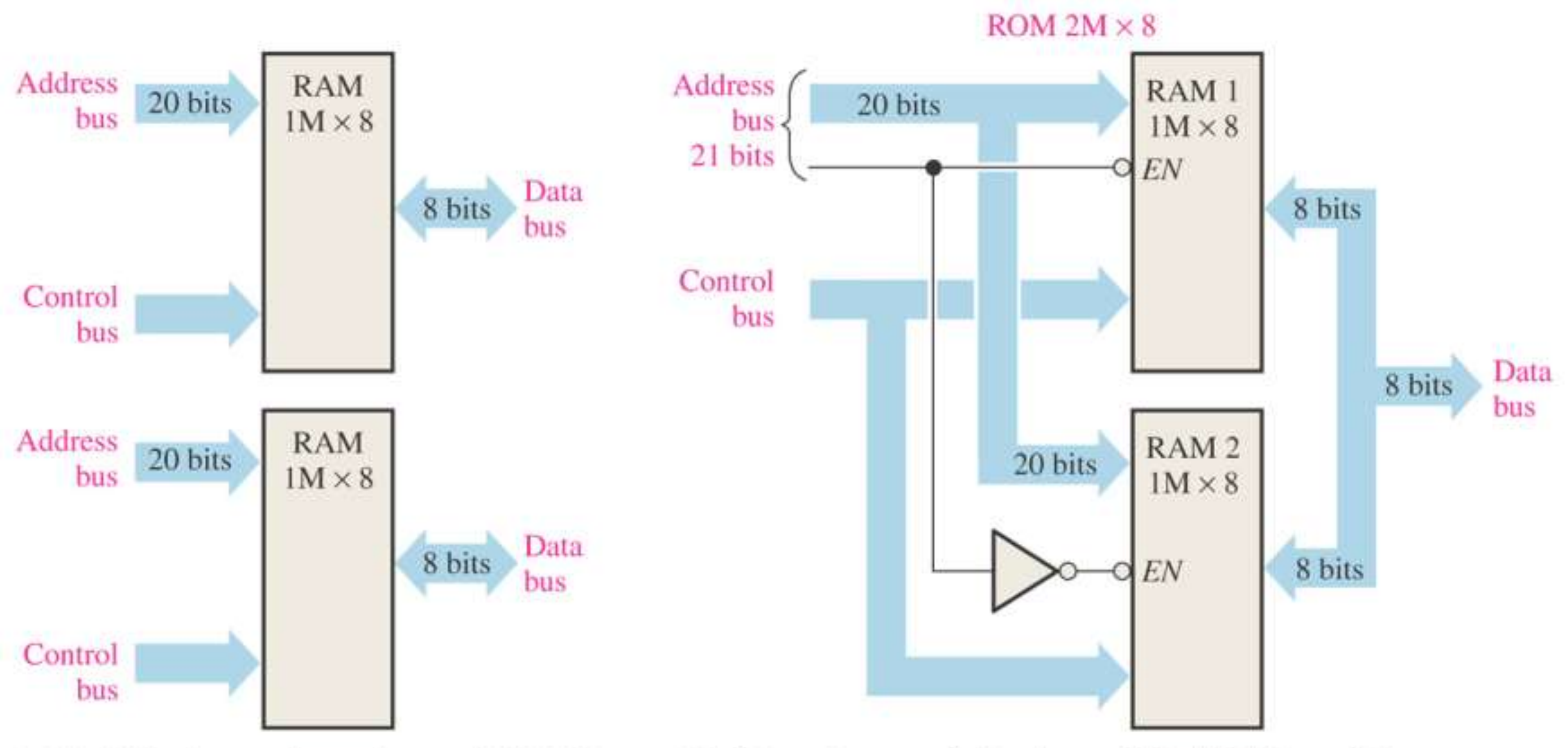

A_0
Address bus
A_{15}
ROM 1
$A \frac{0}{65{,}535}$
&
EN
ROM 2
$A \frac{0}{65{,}535}$
&
EN
Control bus
$\overline{E}$
O_0
O_1
O_2
O_3
O_4
O_5
O_6
O_7
Data bus

a 64k x 8 ROM


Address bus
A_0
A_{15}
16 bits
16 bits
ROM 1
64k × 4
16 bits
ROM 2
64k × 4
16 bits
ROM 3
64k × 4
16 bits
ROM 4
64k × 4
4 bits
4 bits
4 bits
4 bits
&
EN
&
EN
&
EN
&
EN
Control
bus
(enable)
16 bits
Data
bus

a 64k x 16 ROM

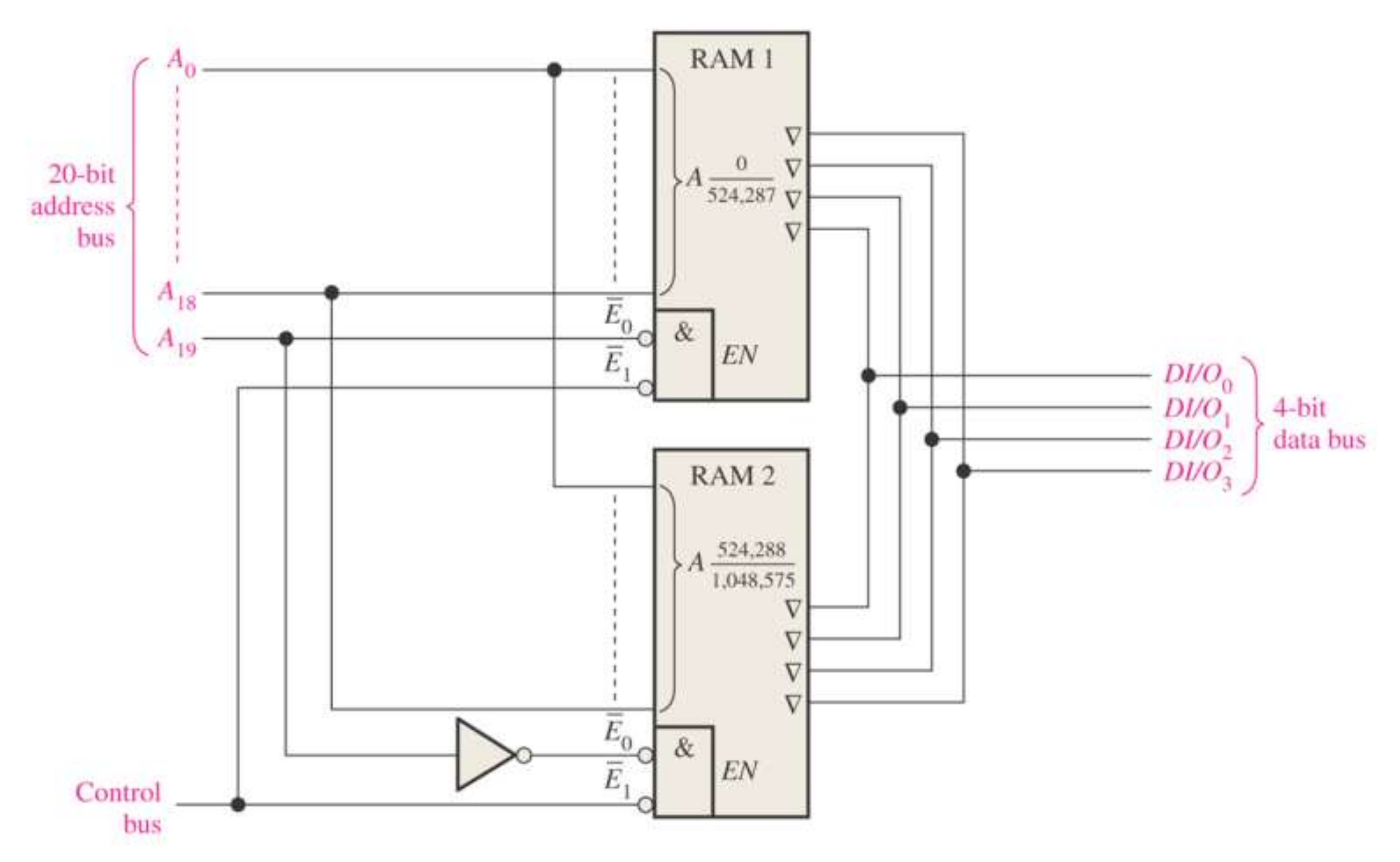
Word-capacity Expansion



(a) Individual memories each store 1,048,576 8-bit words

(b) Memories expanded to form a 2M × 8 RAM requiring a 21-bit address bus

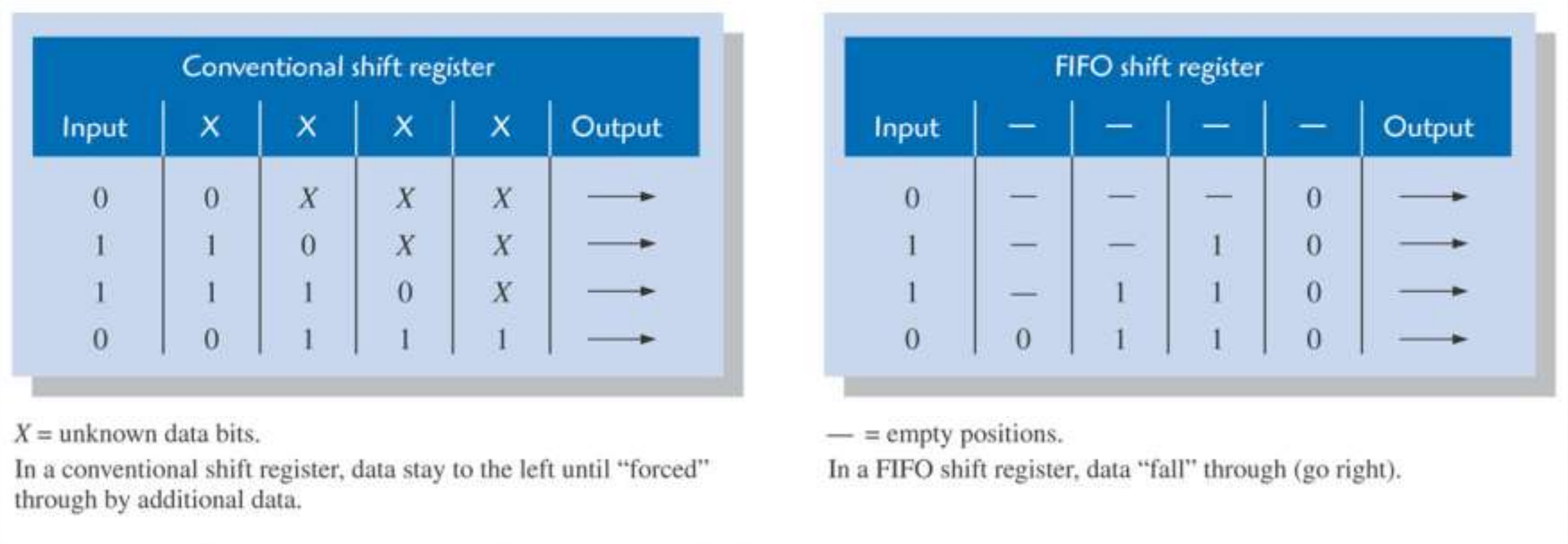
Use 512k x 4 RAM to implement a 1M x 4 memory



10.7 Special Types of Memories

- FIFO memories
- LIFO memories

Comparison of conventional shift register and FIFO register operation



Conventional shift register

Input	X	X	X	X	Output
0	0	*X*	*X*	*X*	→
1	1	0	*X*	*X*	→
1	1	1	0	*X*	→
0	0	1	1	1	→

X = unknown data bits.

In a conventional shift register, data stay to the left until “forced” through by additional data.

FIFO shift register

Input	—	—	—	—	Output
0	—	—	—	0	→
1	—	—	1	0	→
1	—	1	1	0	→
0	0	1	1	0	→

— = empty positions.

In a FIFO shift register, data “fall” through (go right).

Figure 10–49 Block diagram of a typical FIFO serial memory.

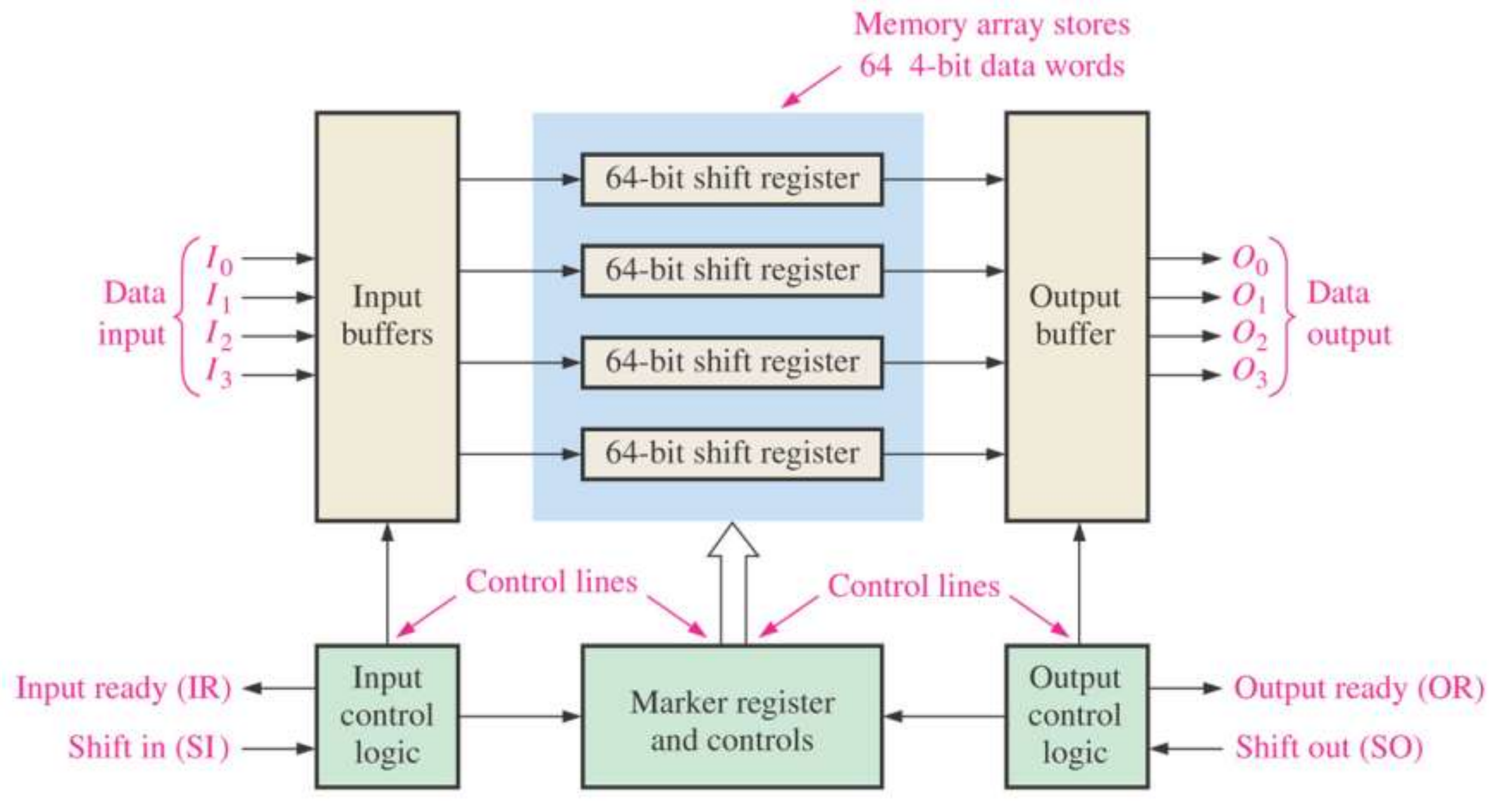
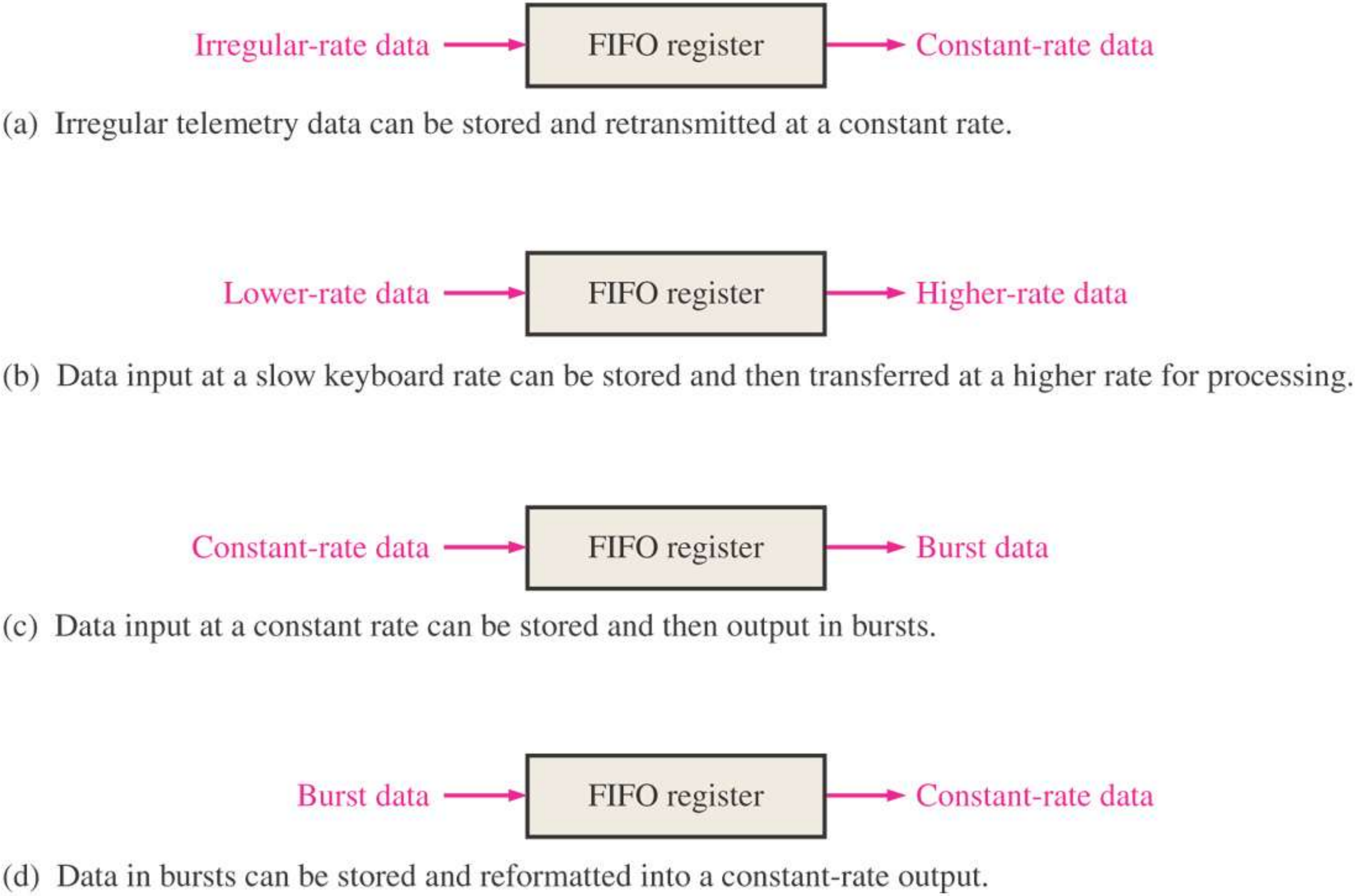


Figure 10–50 Examples of the FIFO register in data-rate buffering applications.



Irregular-rate data → FIFO register → Constant-rate data

(a) Irregular telemetry data can be stored and retransmitted at a constant rate.

Lower-rate data → FIFO register → Higher-rate data

(b) Data input at a slow keyboard rate can be stored and then transferred at a higher rate for processing.

Constant-rate data → FIFO register → Burst data

(c) Data input at a constant rate can be stored and then output in bursts.

Burst data → FIFO register → Constant-rate data

(d) Data in bursts can be stored and reformatted into a constant-rate output.

Register Stack (LIFO: Last In First Out)

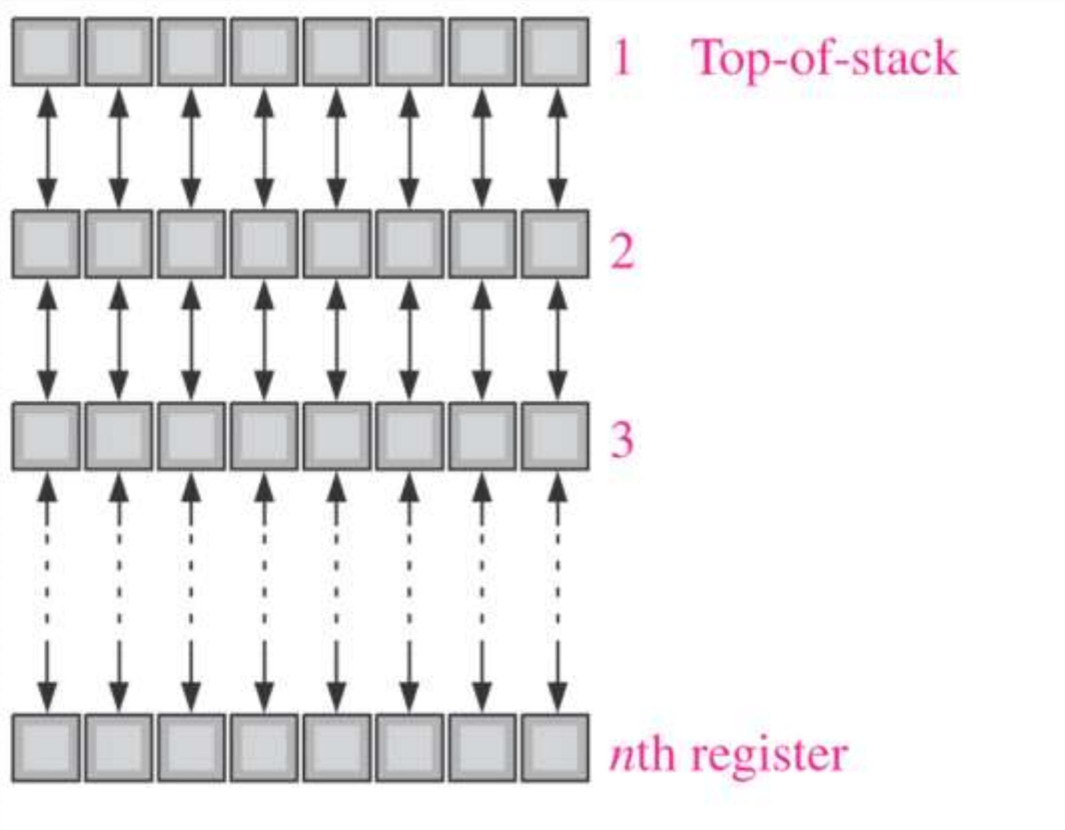


Figure 10–52 Simplified illustration of pushing data onto the stack.

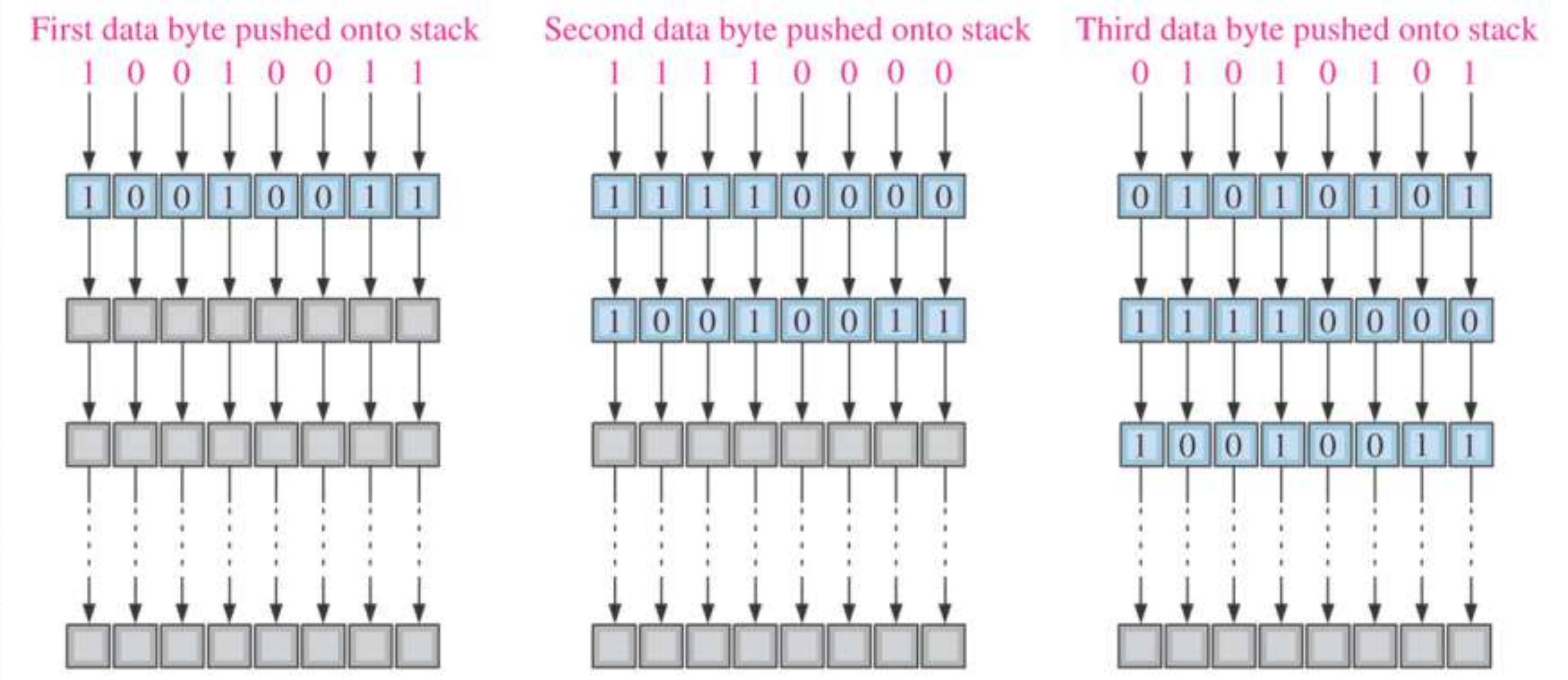
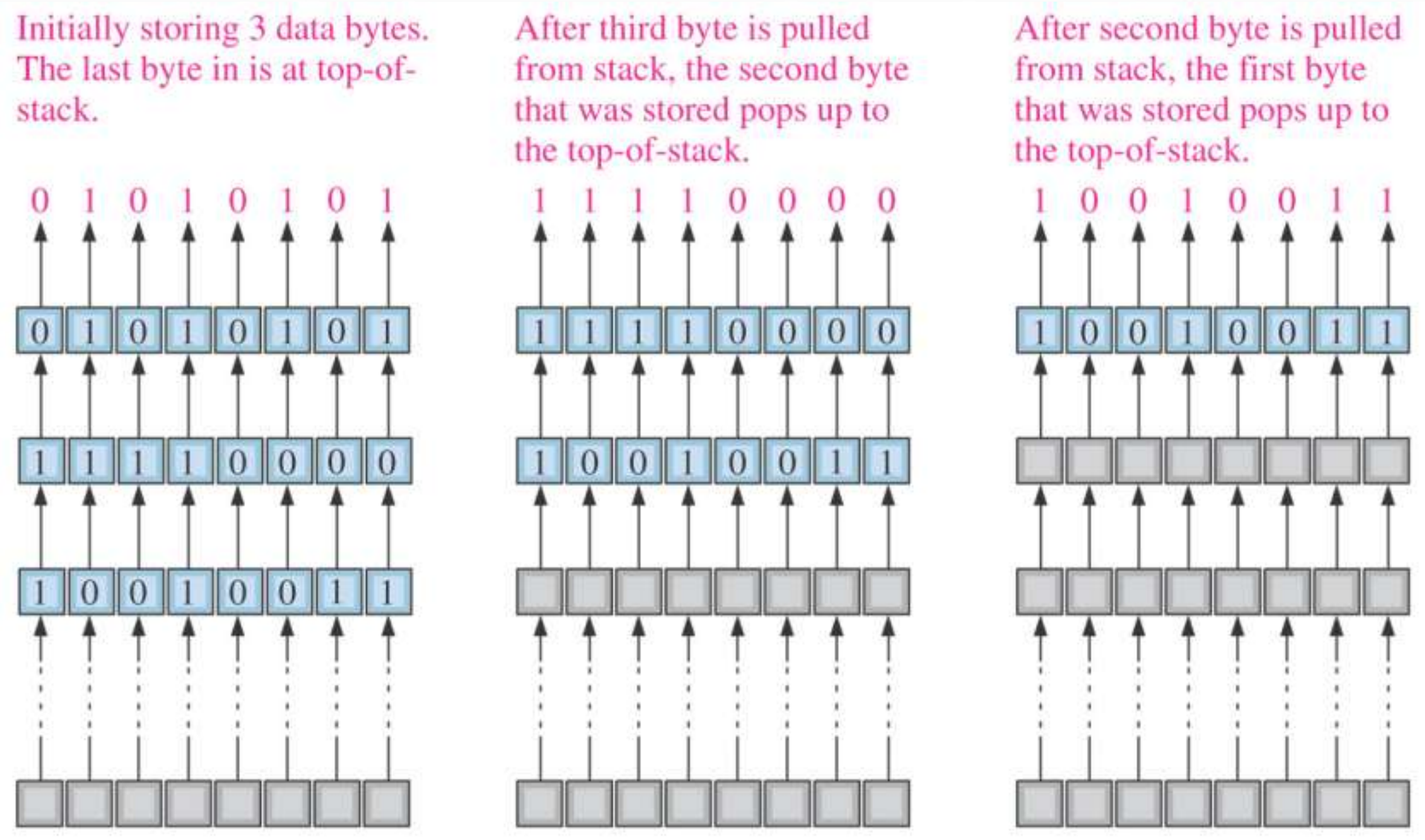


Figure 10–53 Simplified illustration of pulling data from the stack.



Initially storing 3 data bytes. The last byte in is at top-of-stack.

After third byte is pulled from stack, the second byte that was stored pops up to the top-of-stack.

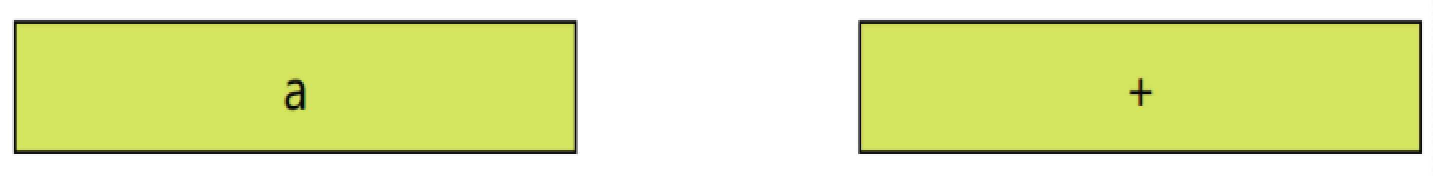
After second byte is pulled from stack, the first byte that was stored pops up to the top-of-stack.

Application: arithmetic expression calculation in computer

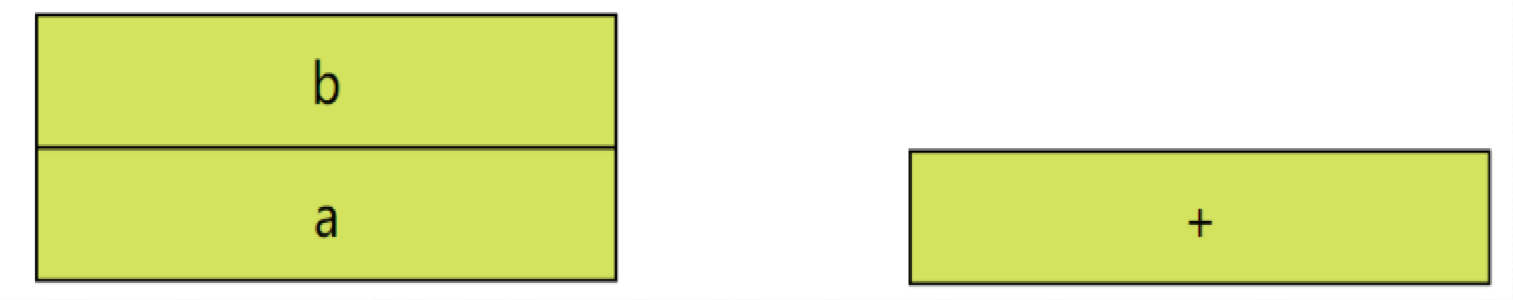
Example: a+b*c/d = ?

number stack **operator stack**

(1)



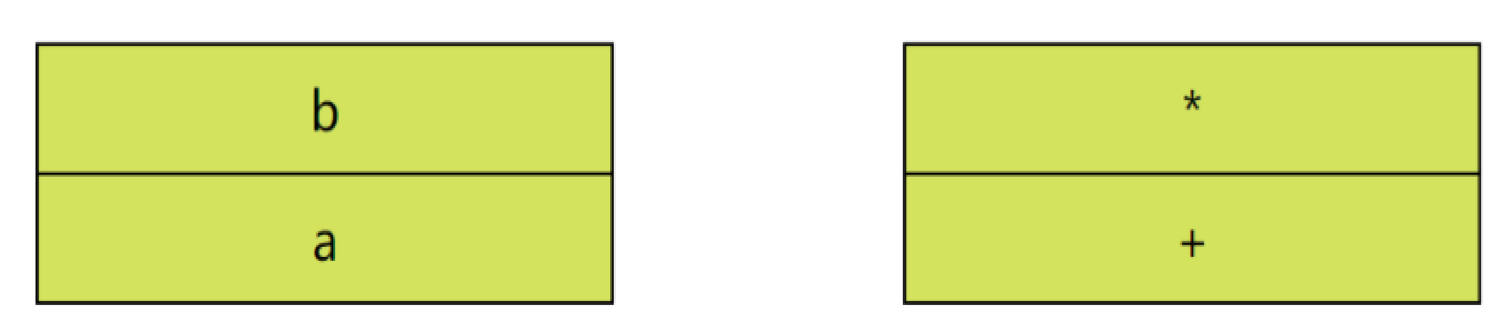
(2)



Example: a+b*c/d = ?

number stack **operator stack**

(3)



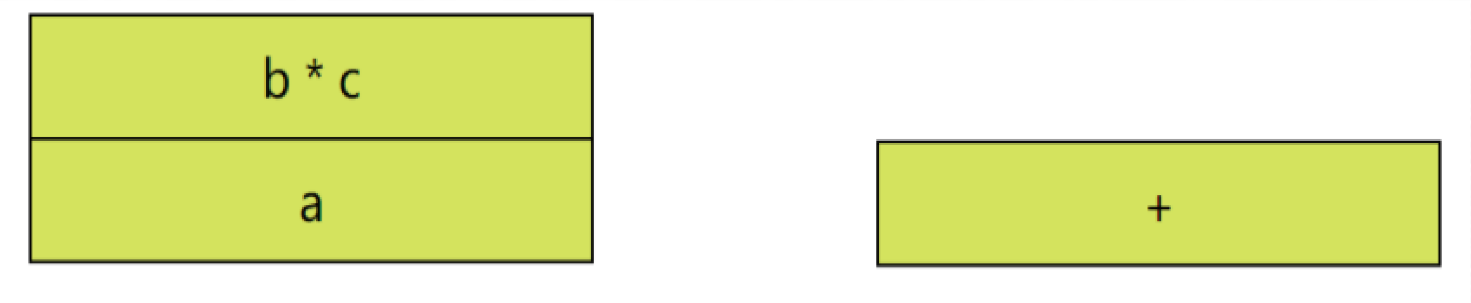
b

a

*

+

(4)



b * c

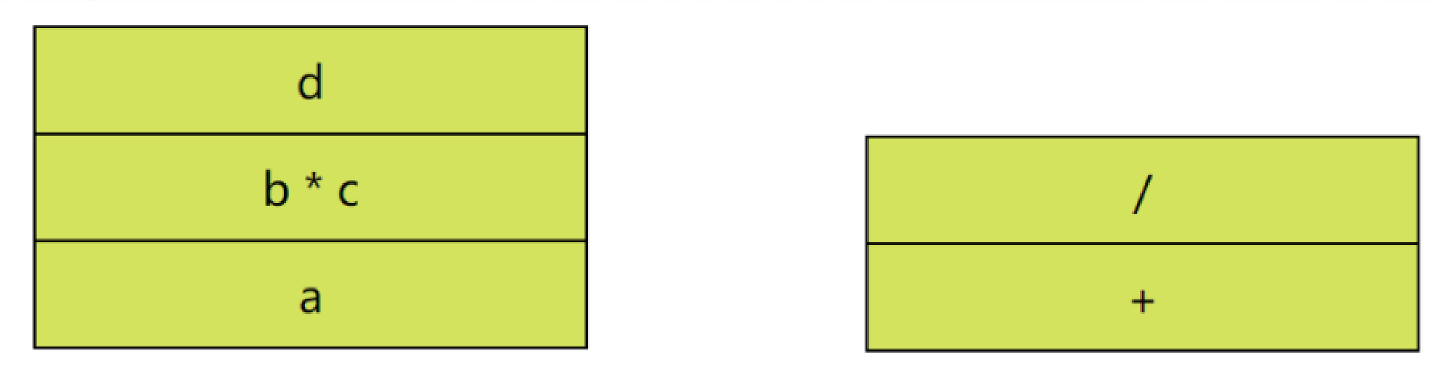
a

+

Example: a+b*c/d = ?

number stack **operator stack**

(5)



d

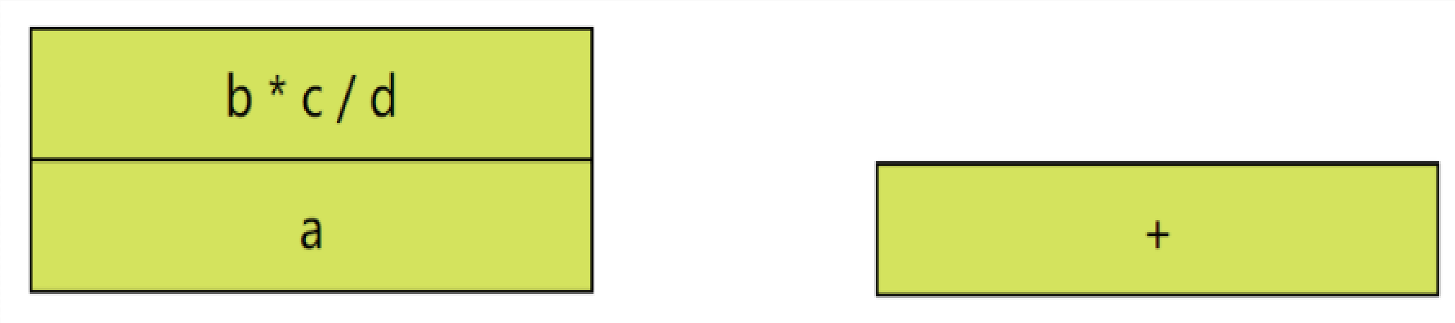
b * c

a

/

+

(6)

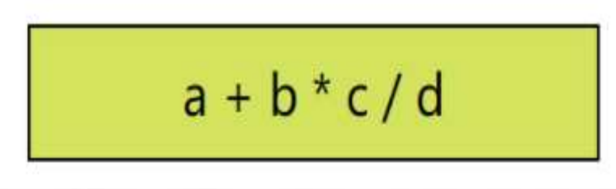


b * c / d

a

+

(7)



a + b * c / d

Summary

- Basics of Semiconductor Memory
- RAM: SRAM and DRAM
- ROM
    - PROM, EPROM, EEPROM, Flash ROM
- Memory Expansion
- FIFO and Stack

HW

- 10th Page 409-411
  - 9
  - 10
  - 11
  - 17

- 11th Page 377-378
  - 9
  - 10
  - 11
  - 18