

# **COM/BLM 376**

# **Computer**

# **Architecture**

## **Chapter 5 Internal Memory**

Asst. Prof. Dr. Gazi Erkan BOSTANCI

ebostanci@ankara.edu.tr

Slides are mainly based on

Computer Organization and Architecture: Designing for Performance by William  
Stallings, 9th Edition, Prentice Hall

# Outline

## Internal Memory

1. Semiconductor Main Memory
2. Error Correction

# Semiconductor Main Memory

In earlier computers, the most common form of random-access storage for computer main memory employed an array of doughnut-shaped ferromagnetic loops referred to as *cores*.

The advent of, and advantages of, microelectronics has long since vanquished the magnetic core memory.

Today, the use of semiconductor chips for main memory is almost universal. Key aspects of this technology are explored in this chapter.

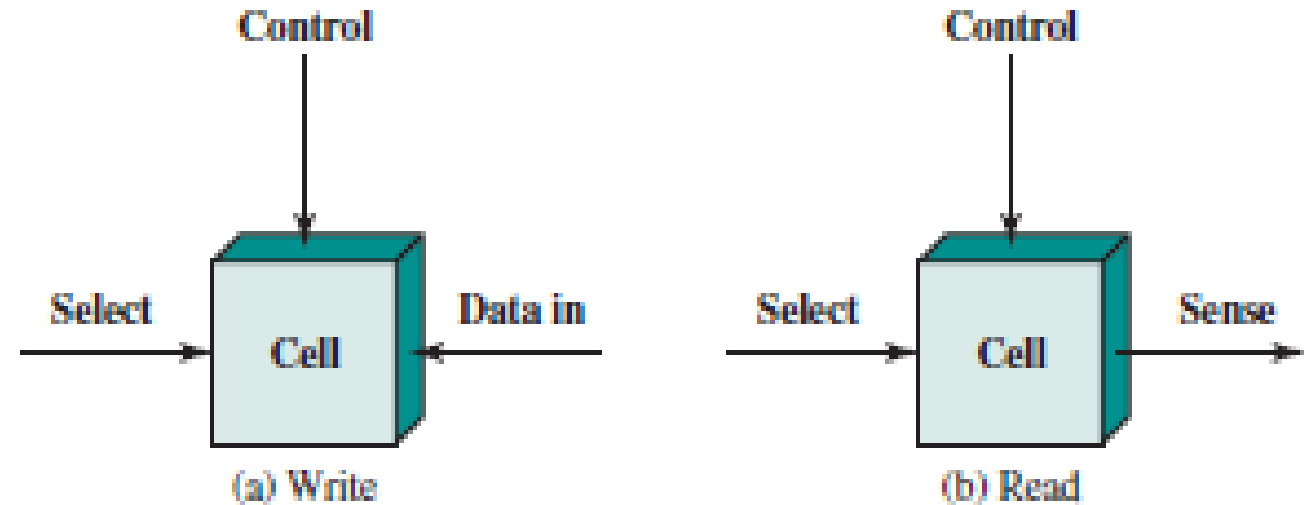
## Organization

The basic element of a **semiconductor memory** is the memory cell. Although a variety of electronic technologies are used, all semiconductor memory cells share certain properties:

- They exhibit two stable (or semistable) states, which can be used to represent binary 1 and 0.
- They are capable of being written into (at least once), to set the state.
- They are capable of being read to sense the state.

Figure depicts the operation of a memory cell. Most commonly, the cell has three functional terminals capable of carrying an electrical signal.

The select terminal, as the name suggests, selects a memory cell for a read or write operation.



The control terminal indicates read or write. For writing, the other terminal provides an electrical signal that sets the state of the cell to 1 or 0. For reading, that terminal is used for output of the cell's state.

## DRAM and SRAM

All of the memory types that we will explore here are random access. That is, individual words of memory are directly accessed through wired-in addressing logic.

Table lists the major types of semiconductor memory. The most common is referred to as *random-access memory* (RAM).

This is, in fact, a misuse of the term, because all of the types listed in the table are random access. One distinguishing characteristic of memory that is designated as RAM is that it is possible both to read data from the memory and to write new data into the memory easily and rapidly.

Both the reading and writing are accomplished through the use of electrical signals.

# Semiconductor Memory Types

Memory Type	Category	Erasure	Write Mechanism	Volatility
Random-access memory (RAM)	Read-write memory	Electrically, byte-level	Electrically	Volatile
Read-only memory (ROM)	Read-only memory	Not possible	Masks	Nonvolatile
Programmable ROM (PROM)			Electrically	
Erasable PROM (EPROM)	UV light, chip-level			
Electrically Erasable PROM (EEPROM)	Electrically, byte-level			
Flash memory	Electrically, block-level			

The other distinguishing characteristic of RAM is that it is volatile. A RAM must be provided with a constant power supply. If the power is interrupted, then the data are lost. Thus, RAM can be used only as temporary storage.

RAM technology is divided into two technologies: dynamic and static.  
(DRAM and SRAM)



## ***DYNAMIC RAM***

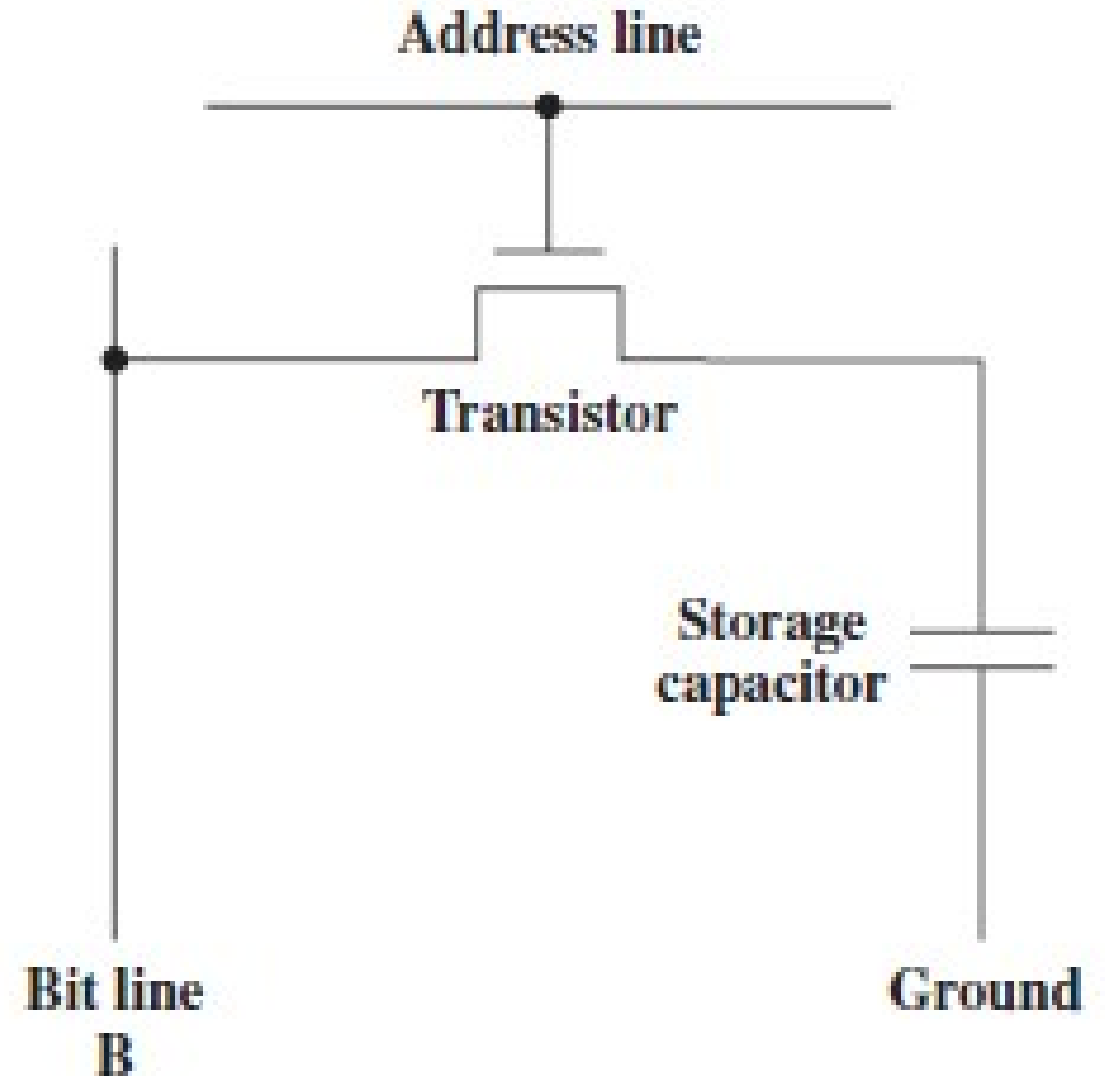
A **dynamic RAM (DRAM)** is made with cells that store data as charge on capacitors.

The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, dynamic RAMs require periodic charge refreshing to maintain data storage.

The term *dynamic* refers to this tendency of the stored charge to leak away, even with power continuously applied.

Figure shows a typical DRAM structure for an individual cell that stores 1 bit. The address line is activated when the bit value from this cell is to be read or written.

The transistor acts as a switch that is closed (allowing current to flow) if a voltage is applied to the address line and open (no current flows) if no voltage is present on the address line.



For the write operation, a voltage signal is applied to the bit line; a high voltage represents 1, and a low voltage represents 0. A signal is then applied to the address line, allowing a charge to be transferred to the capacitor.

For the read operation, when the address line is selected, the transistor turns on and the charge stored on the capacitor is fed out onto a bit line and to a sense amplifier. The sense amplifier compares the capacitor voltage to a reference value and determines if the cell contains a logic 1 or a logic 0. The readout from the cell discharges the capacitor, which must be restored to complete the operation.

Although the DRAM cell is used to store a single bit (0 or 1), it is essentially an analog device. The capacitor can store any charge value within a range; a threshold value determines whether the charge is interpreted as 1 or 0.

## ***STATIC RAM***

In contrast, a **static RAM (SRAM)** is a digital device that uses the same logic elements used in the processor. In a SRAM, binary values are stored using traditional flip-flop logic-gate configurations.

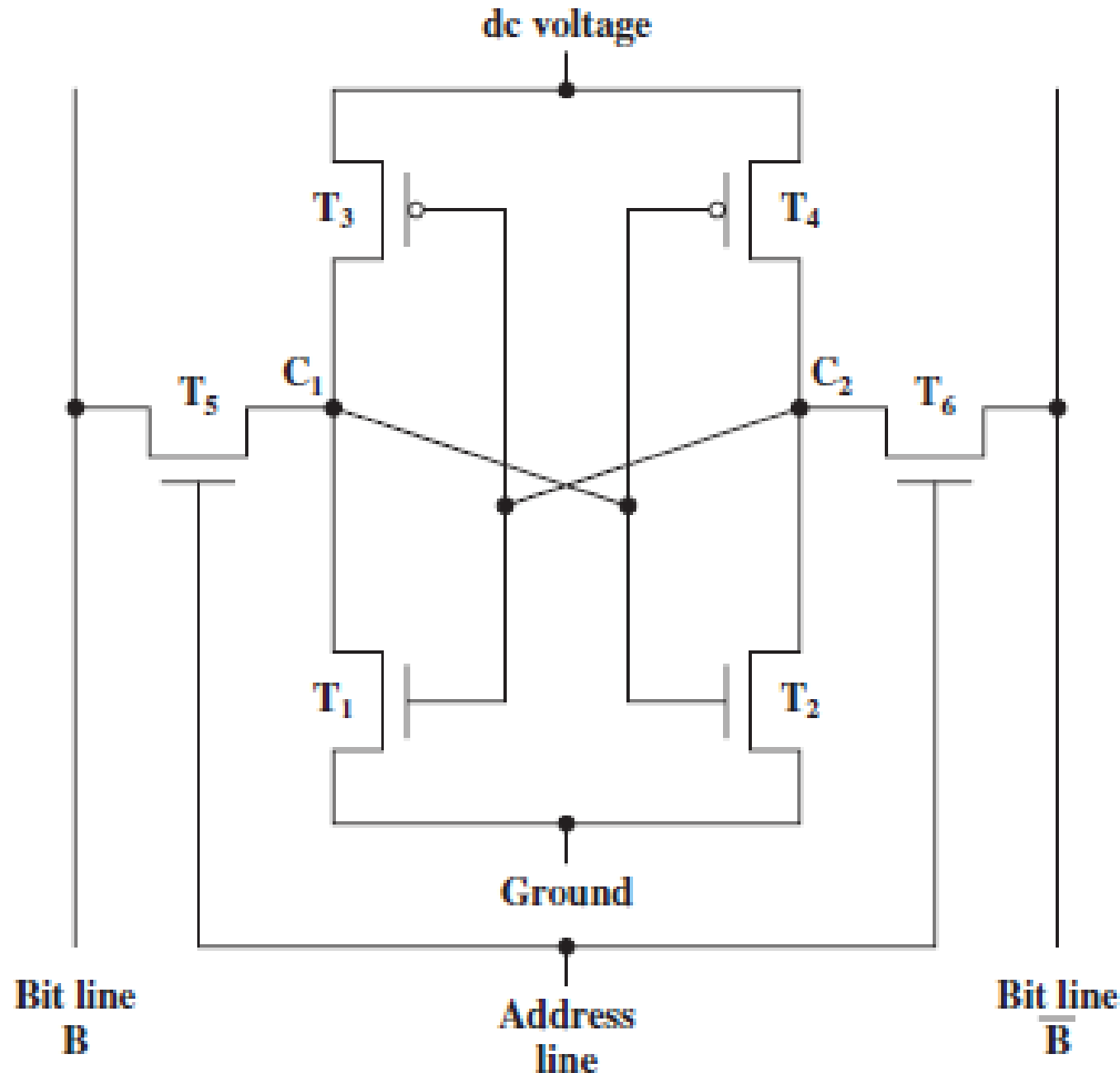
A static RAM will hold its data as long as power is supplied to it.

Figure shows a typical SRAM structure for an individual cell. Four transistors (T1, T2, T3, T4) are cross connected in an arrangement that produces a stable logic state.

In logic state 1, point C1 is high and point C2 is low; in this state, T1 and T4 are off and T2 and T3 are on.

In logic state 0, point C1 is low and point C2 is high; in this state, T1 and T4 are on and T2 and T3 are off. Both states are stable as long as the direct current (dc) voltage is applied.

Unlike the DRAM, no refresh is needed to retain data.



As in the DRAM, the SRAM address line is used to open or close a switch.

The address line controls two transistors (T5 and T6). When a signal is applied to this line, the two transistors are switched on, allowing a read or write operation.

For a write operation, the desired bit value is applied to line B, while its complement is applied to line . This forces the four transistors (T1, T2, T3, T4) into the proper state.

For a read operation, the bit value is read from line B.

## ***SRAM VERSUS DRAM***

Both static and dynamic RAMs are volatile; that is, power must be continuously supplied to the memory to preserve the bit values.

A dynamic memory cell is simpler and smaller than a static memory cell. Thus, a DRAM is more dense (smaller cells = more cells per unit area) and less expensive than a corresponding SRAM.

On the other hand, a DRAM requires the supporting refresh circuitry. For larger memories, the fixed cost of the refresh circuitry is more than compensated for by the smaller variable cost of DRAM cells. Thus, DRAMs tend to be favored for large memory requirements.

A final point is that SRAMs are somewhat faster than DRAMs. Because of these relative characteristics, SRAM is used for cache memory (both on and off chip), and DRAM is used for main memory.



## Types of ROM

As the name suggests, a **read-only memory** (ROM) contains a permanent pattern of data that cannot be changed. A ROM is nonvolatile; that is, no power source is required to maintain the bit values in memory.

While it is possible to read a ROM, it is not possible to write new data into it. An important application of ROMs is microprogramming.

Other potential applications include:

- Library subroutines for frequently wanted functions
- System programs
- Function tables

For a modest-sized requirement, the advantage of ROM is that the data or program is permanently in main memory and need never be loaded from a secondary storage device.

A ROM is created like any other integrated circuit chip, with the data actually wired into the chip as part of the fabrication process. This presents two problems:

- The data insertion step includes a relatively large fixed cost, whether one or thousands of copies of a particular ROM are fabricated.
- There is no room for error. If one bit is wrong, the whole batch of ROMs must be thrown out.

When only a small number of ROMs with a particular memory content is needed, a less expensive alternative is the **programmable ROM (PROM)**.

Like the ROM, the PROM is **nonvolatile** and may be written into only once. For the PROM, the writing process is performed electrically and may be performed by a supplier or customer at a time later than the original chip fabrication.

Special equipment is required for the writing or “programming” process. PROMs provide flexibility and convenience.

The optically **erasable programmable read-only memory (EPROM)** is read and written electrically, as with PROM. However, before a write operation, all the storage cells must be erased to the same initial state by exposure of the packaged chip to ultraviolet radiation.

Erase is performed by shining an intense ultraviolet light through a window that is designed into the memory chip. This erasure process can be performed repeatedly; each erasure can take as much as 20 minutes to perform. Thus, the EPROM can be altered multiple times and, like the ROM and PROM, holds its data virtually indefinitely.

For comparable amounts of storage, the EPROM is more expensive than PROM, but it has the advantage of the multiple update capability.

A more attractive form of read-mostly memory is **electrically erasable programmable read-only memory (EEPROM)**. This is a read-mostly memory that can be written into at any time without erasing prior contents; only the byte or bytes addressed are updated.

The write operation takes considerably longer than the read operation, on the order of several hundred microseconds per byte.

The EEPROM combines the advantage of nonvolatility with the flexibility of being updatable in place, using ordinary bus control, address, and data lines. EEPROM is more expensive than EPROM and also is less dense, supporting fewer bits per chip.

Another form of semiconductor memory is **flash memory** (so named because of the speed with which it can be reprogrammed). First introduced in the mid-1980s, flash memory is intermediate between EPROM and EEPROM in both cost and functionality. Like EEPROM, flash memory uses an electrical erasing technology.

An entire flash memory can be erased in one or a few seconds, which is much faster than EPROM. In addition, it is possible to erase just blocks of memory rather than an entire chip.

Flash memory gets its name because the microchip is organized so that a section of memory cells are erased in a single action or “flash.”

## Chip Logic

In the memory hierarchy as a whole, we saw that there are trade-offs among speed, capacity, and cost. These trade-offs also exist when we consider the organization of memory cells and functional logic on a chip.

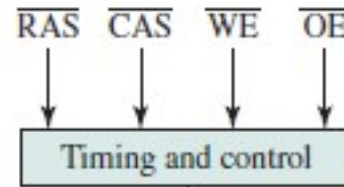
For semiconductor memories, one of the key design issues is the number of bits of data that may be read/written at a time. At one extreme is an organization in which the physical arrangement of cells in the array is the same as the logical arrangement (as perceived by the processor) of words in memory.

The array is organized into  $W$  words of  $B$  bits each. For example, a 16-Mbit chip could be organized as 1M 16-bit words. At the other extreme is the so-called 1-bit-per-chip organization, in which data are read/written 1 bit at a time.

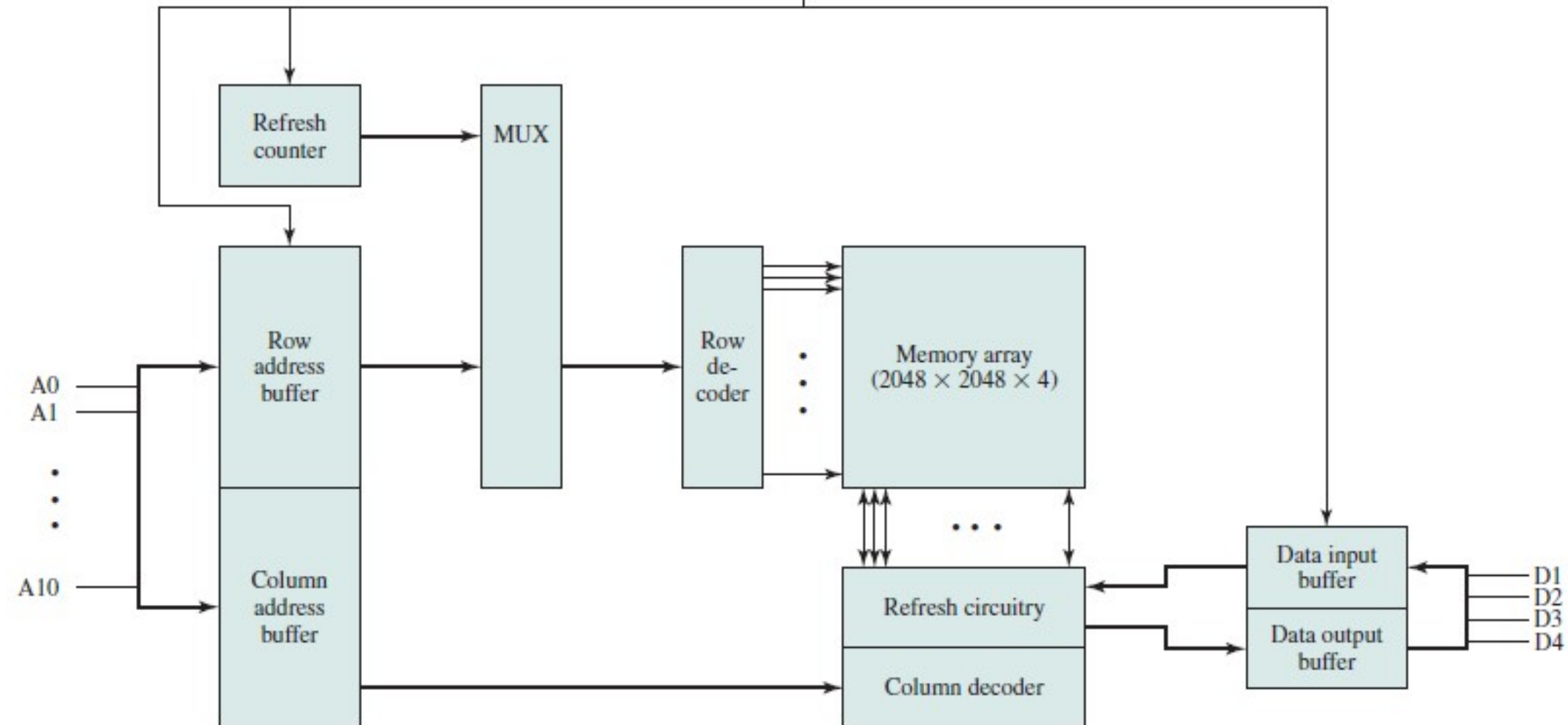
We will illustrate memory chip organization with a DRAM; ROM organization is similar, though simpler.

Figure below also indicates the inclusion of refresh circuitry. All DRAMs require a refresh operation.





**RAS: Row Address Select**  
**CAS: Column Address Select**



A simple technique for refreshing is, in effect, to disable the DRAM chip while all data cells are refreshed. The refresh counter steps through all of the row values.

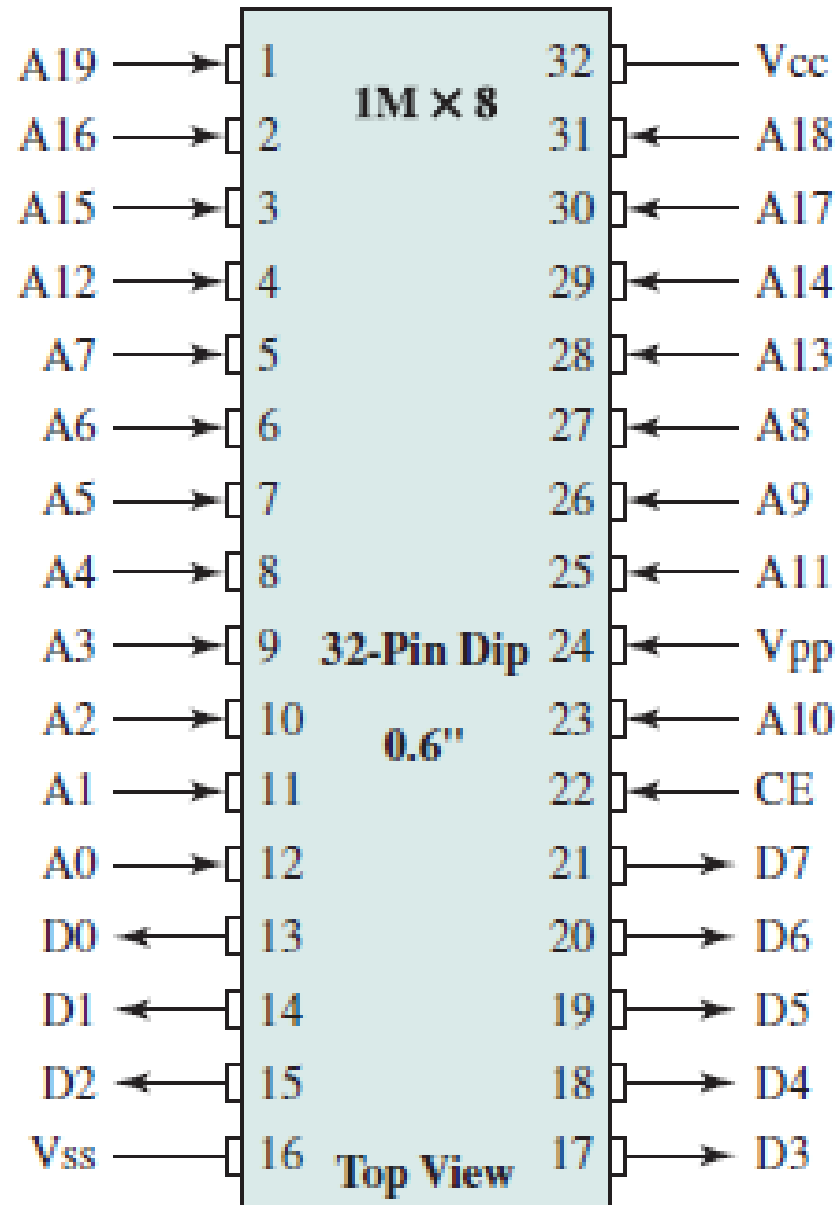
For each row, the output lines from the refresh counter are supplied to the row decoder and the RAS line is activated. The data are read out and written back into the same location. This causes each cell in the row to be refreshed.

## Chip Packaging

As mentioned earlier, an integrated circuit is mounted on a package that contains pins for connection to the outside world.

Figure below shows an example EPROM package, which is an 8-Mbit chip organized as 1M x 8. In this case, the organization is treated as a one-word-per-chip package. The package includes 32 pins, which is one of the standard chip package sizes. The pins support the following signal lines:

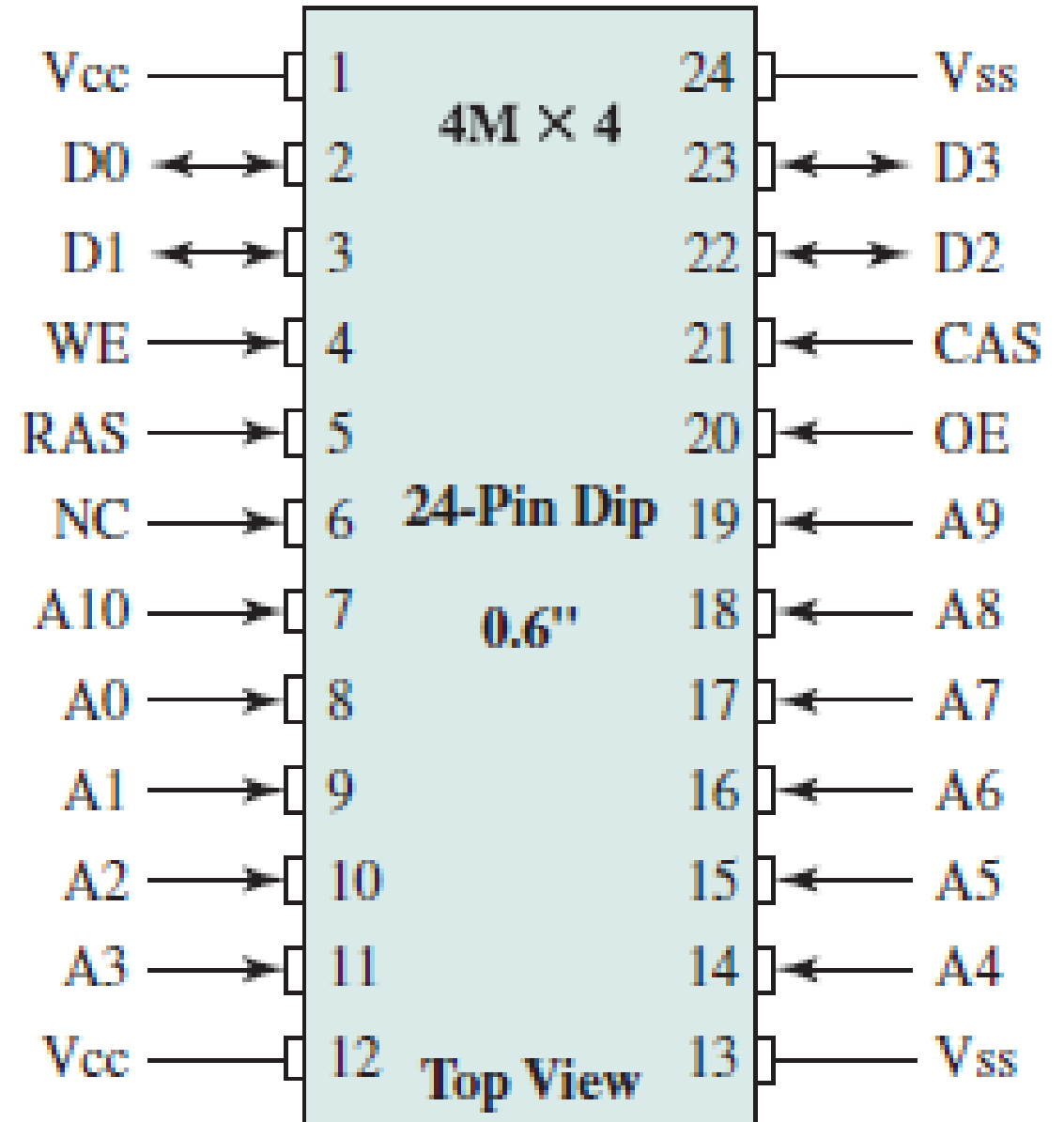
- The address of the word being accessed. For 1M words, a total of 20 ( $2^{20} = 1\text{M}$ ) pins are needed (A0–A19).
- The data to be read out, consisting of 8 lines (D0–D7).
- The power supply to the chip (Vcc).
- A ground pin (Vss).
- A chip enable (CE) pin. Because there may be more than one memory chip, each of which is connected to the same address bus, the CE pin is used to indicate whether or not the address is valid for this chip. The CE pin is activated by logic connected to the higher-order bits of the address bus (i.e., address bits above A19). The use of this signal is illustrated presently.
- A program voltage (Vpp) that is supplied during programming (write operations).



A typical DRAM pin configuration is shown in Figure, for a 16-Mbit chip organized as 4M x 4.

There are several differences from a ROM chip. Because a RAM can be updated, the data pins are input/output.

The write enable (WE) and output enable (OE) pins indicate whether this is a write or read operation.



Because the DRAM is accessed by row and column, and the address is multiplexed, only 11 address pins are needed to specify the 4M row/column combinations ( $2^{11} \times 2^{11} = 2^{22} = 4\text{M}$ ).

The functions of the row address select (RAS) and column address select (CAS) pins were discussed previously.

Finally, the no connect (NC) pin is provided so that there are an even number of pins.

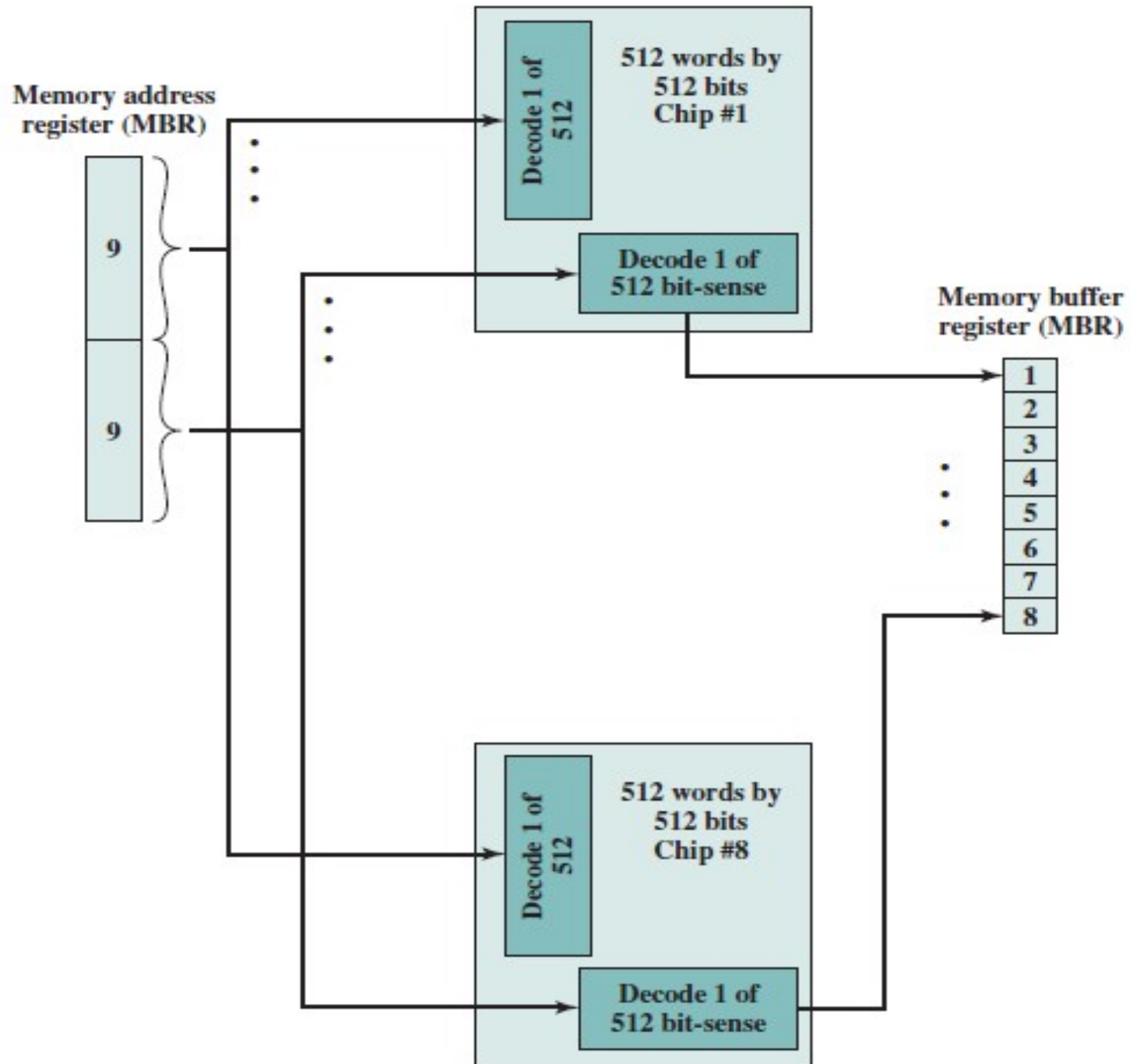
## Module Organization

If a RAM chip contains only 1 bit per word, then clearly we will need at least a number of chips equal to the number of bits per word.

As an example, Figure below shows how a memory module consisting of 256K 8-bit words could be organized. For 256K words, an 18-bit address is needed and is supplied to the module from some external source (e.g., the address lines of a bus to which the module is attached).

The address is presented to 8 256K x 1-bit chips, each of which provides the input/output of 1 bit.

## 256-KByte Memory Organization





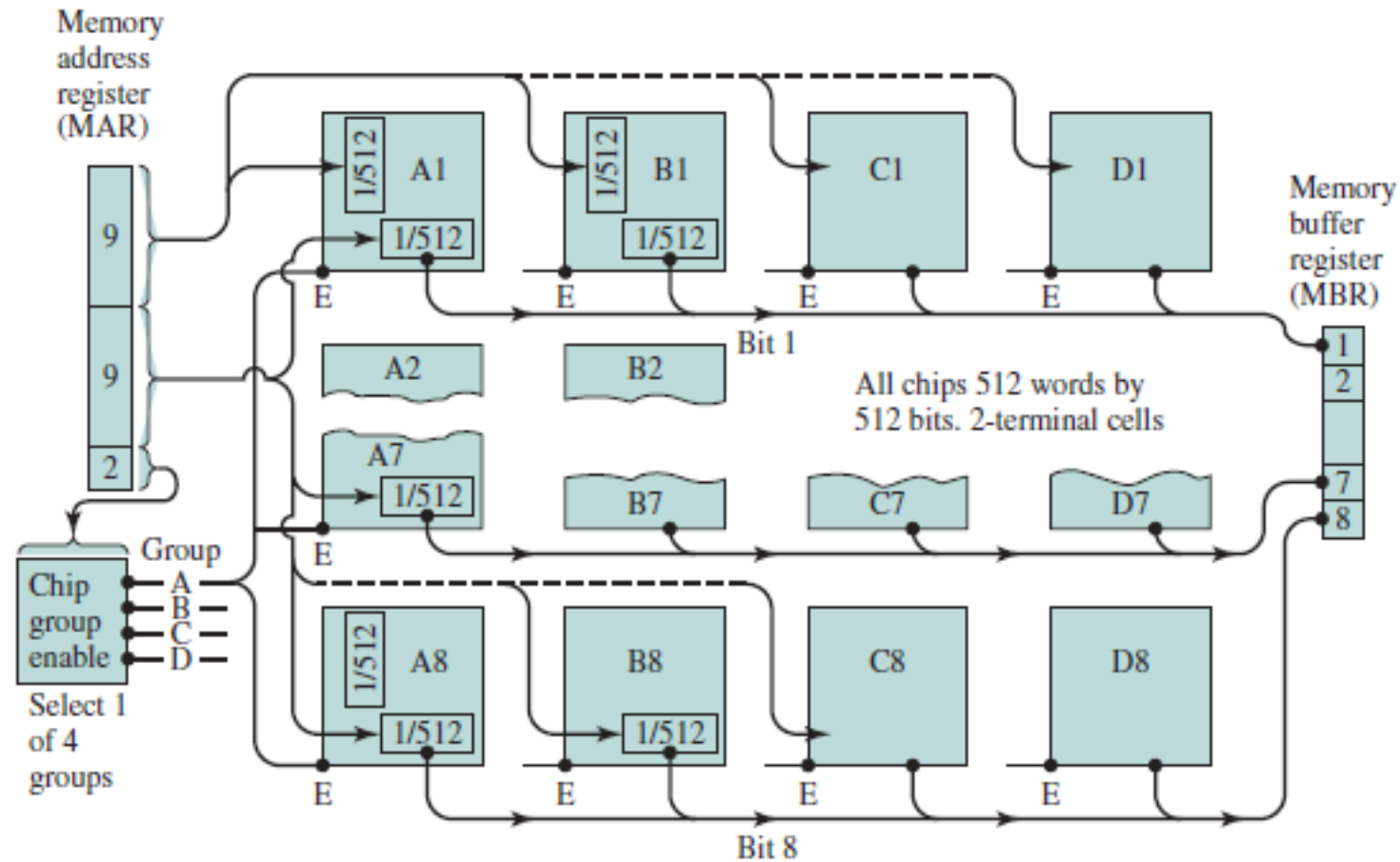
This organization works as long as the size of memory equals the number of bits per chip.

In the case in which larger memory is required, an array of chips is needed.

Figure shows the possible organization of a memory consisting of 1M word by 8 bits per word. In this case, we have four columns of chips, each column containing 256K words arranged as in Figure above.

For 1M word, 20 address lines are needed. The 18 least significant bits are routed to all 32 modules.

The high-order 2 bits are input to a group select logic module that sends a chip enable signal to one of the four columns of modules.



## Interleaved Memory

Main memory is composed of a collection of DRAM memory chips. A number of chips can be grouped together to form a *memory bank*.

It is possible to organize the memory banks in a way known as interleaved memory. Each bank is independently able to service a memory read or write request, so that a system with  $K$  banks can service  $K$  requests simultaneously, increasing memory read or write rates by a factor of  $K$ .

If consecutive words of memory are stored in different banks, then the transfer of a block of memory is speeded up.

# ERROR CORRECTION

A semiconductor memory system is subject to errors. These can be categorized as hard failures and soft errors.

A **hard failure** is a permanent physical defect so that the memory cell or cells affected cannot reliably store data but become stuck at 0 or 1 or switch erratically between 0 and 1. Hard errors can be caused by harsh environmental abuse, manufacturing defects, and wear.

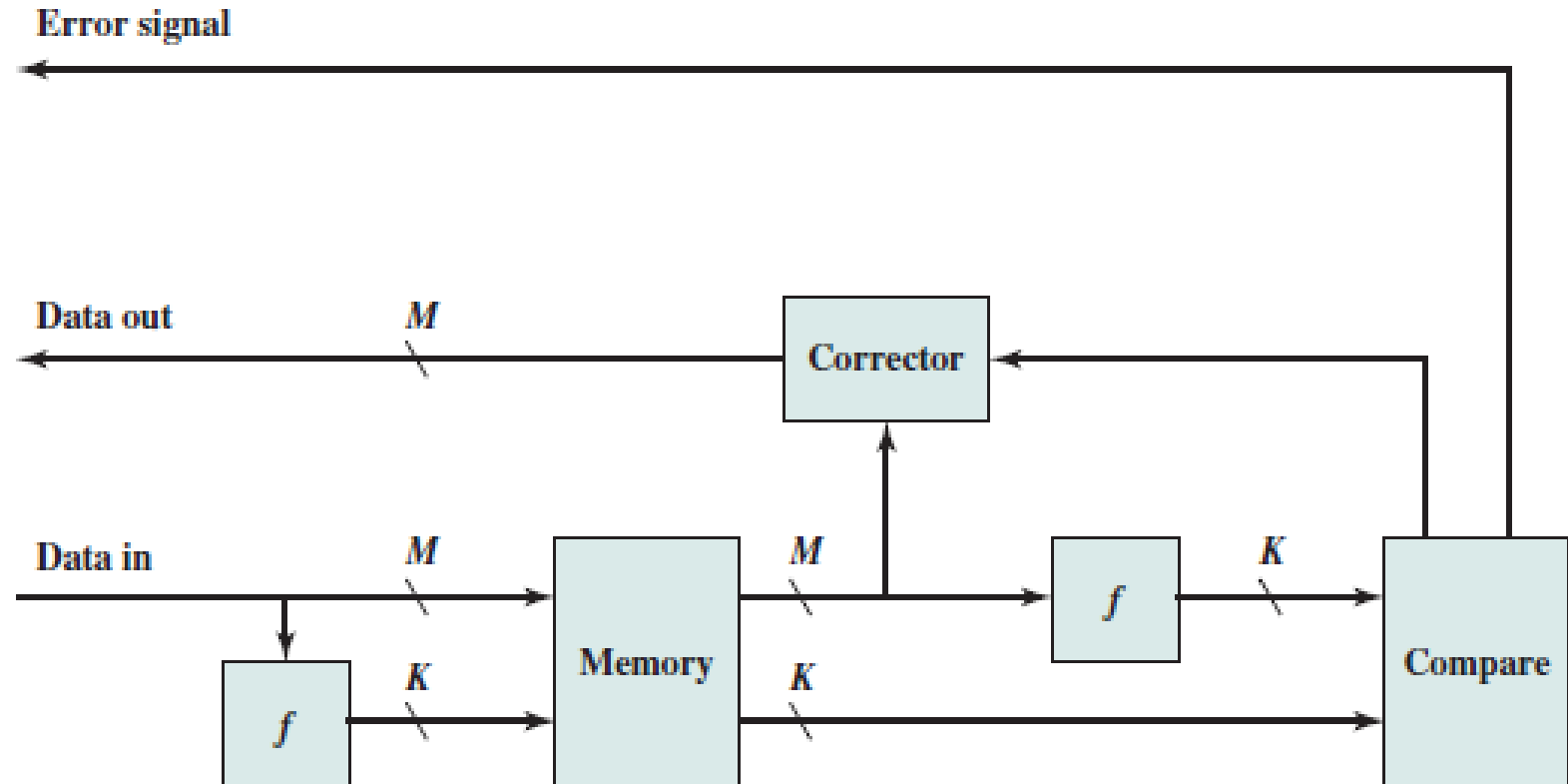
A **soft error** is a random, nondestructive event that alters the contents of one or more memory cells without damaging the memory. Soft errors can be caused by power supply problems or alpha particles. These particles result from radioactive decay and are distressingly common because radioactive nuclei are found in small quantities in nearly all materials.

Both hard and soft errors are clearly undesirable, and most modern main memory systems include logic for both detecting and correcting errors.

Figure illustrates in general terms how the process is carried out.

When data are to be written into memory, a calculation, depicted as a function  $f$ , is performed on the data to produce a code.

Both the code and the data are stored. Thus, if an  $M$ -bit word of data is to be stored and the code is of length  $K$  bits, then the actual size of the stored word is  $M + K$  bits.



When the previously stored word is read out, the code is used to detect and possibly correct errors. A new set of  $K$  code bits is generated from the  $M$  data bits and compared with the fetched code bits. The comparison yields one of three results:

- No errors are detected. The fetched data bits are sent out.
- An error is detected, and it is possible to correct the error. The data bits plus **error correction** bits are fed into a corrector, which produces a corrected set of  $M$  bits to be sent out.
- An error is detected, but it is not possible to correct it. This condition is reported.

Codes that operate in this fashion are referred to as **error-correcting codes**. A code is characterized by the number of bit errors in a word that it can correct and detect.

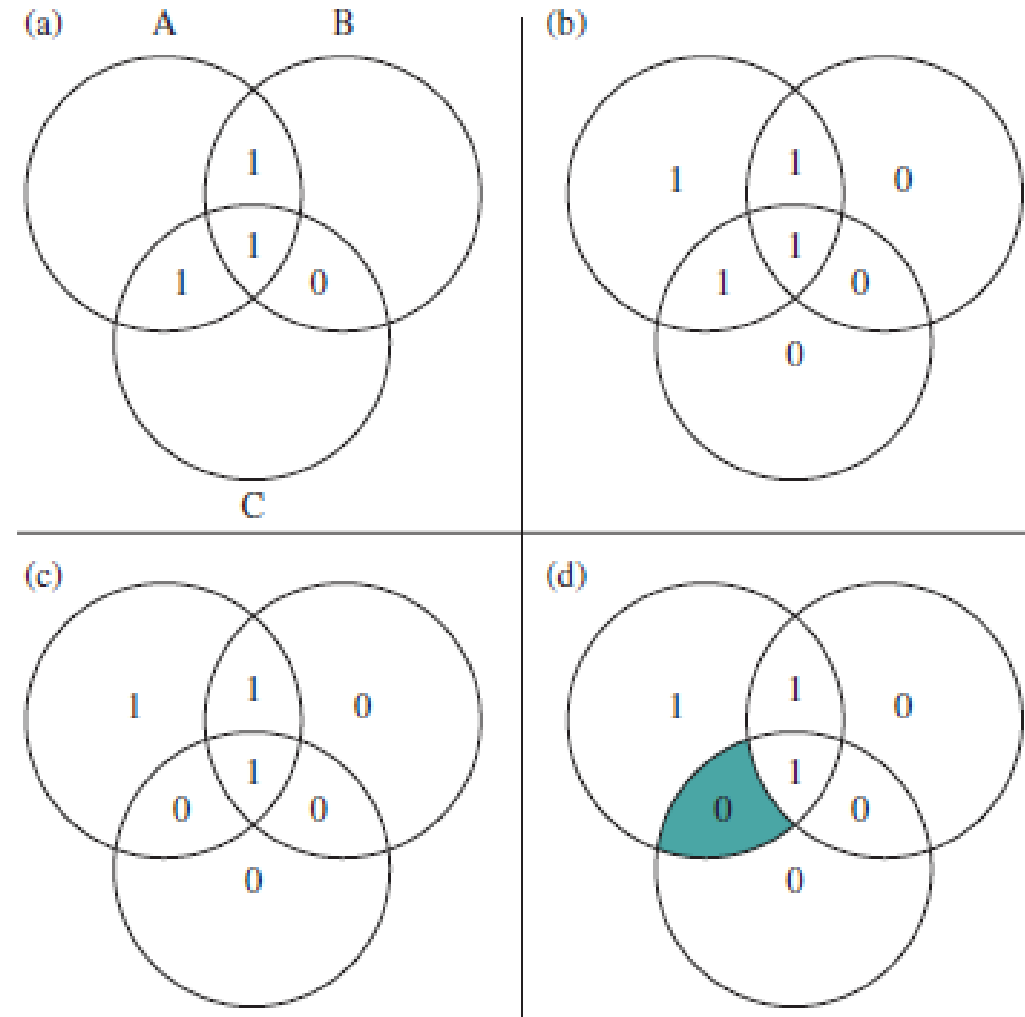
The simplest of the error-correcting codes is the **Hamming code** devised by Richard Hamming at Bell Laboratories.

# Hamming Error-Correcting Code

Figure uses Venn diagrams to illustrate the use of this code on 4-bit words ( $M = 4$ ). With three intersecting circles, there are seven compartments.

We assign the 4 data bits to the inner compartments (a). The remaining compartments are filled with what are called *parity bits*. Each parity bit is chosen so that the total number of 1s in its circle is even (b). Thus, because circle A includes three data 1s, the parity bit in that circle is set to 1.

Now, if an error changes one of the data bits (c), it is easily found. By checking the parity bits, discrepancies are found in circle A and circle C but not in circle B. Only one of the seven compartments is in A and C but not B. The error can therefore be corrected by changing that bit.





To clarify the concepts involved, we will develop a code that can detect and correct single-bit errors in 8-bit words.

To start, let us determine how long the code must be.

Referring to the error-correcting code function logic diagram above, the comparison logic receives as input two  $K$ -bit values. A bit-by-bit comparison is done by taking the exclusive-OR of the two inputs.

The result is called the *syndrome word*. Thus, each bit of the **syndrome** is 0 or 1 according to if there is or is not a match in that bit position for the two inputs.

The syndrome word is therefore  $K$  bits wide and has a range between 0 and  $2^K - 1$ . The value 0 indicates that no error was detected, leaving  $2^K - 1$  values to indicate, if there is an error, which bit was in error.

Now, because an error could occur on any of the  $M$  data bits or  $K$  check bits, we must have

$$2^K - 1 \geq M + K$$

This inequality gives the number of bits needed to correct a single bit error in a word containing  $M$  data bits. For example, for a word of 8 data bits ( $M = 8$ ), we have

- $K = 3$ :  $2^3 - 1 < 8 + 3$
- $K = 4$ :  $2^4 - 1 > 8 + 4$

Thus, eight data bits require four check bits. The first three columns of Table lists the number of check bits required for various data word lengths.

	Single-Error Correction		Single-Error Correction/ Double-Error Detection	
Data Bits	Check Bits	% Increase	Check Bits	% Increase
8	4	50	5	62.5
16	5	31.25	6	37.5
32	6	18.75	7	21.875
64	7	10.94	8	12.5
128	8	6.25	9	7.03
256	9	3.52	10	3.91

For convenience, we would like to generate a 4-bit syndrome for an 8-bit data word with the following characteristics:

- If the syndrome contains all 0s, no error has been detected.
- If the syndrome contains one and only one bit set to 1, then an error has occurred in one of the 4 check bits. No correction is needed.
- If the syndrome contains more than one bit set to 1, then the numerical value of the syndrome indicates the position of the data bit in error. This data bit is inverted for correction.

To achieve these characteristics, the data and check bits are arranged into a 12-bit word as depicted in Figure below. The bit positions are numbered from 1 to 12. Those bit positions whose position numbers are powers of 2 are designated as check bits. The check bits are calculated as follows, where the symbol  $\oplus$  designates the exclusive-OR operation:

$$\begin{aligned}
 C1 &= D1 \oplus D2 \oplus D4 \oplus D5 \oplus D7 \\
 C2 &= D1 \oplus D3 \oplus D4 \oplus D6 \oplus D7 \\
 C4 &= D2 \oplus D3 \oplus D4 \oplus D8 \\
 C8 &= D5 \oplus D6 \oplus D7 \oplus D8
 \end{aligned}$$

Bit position	12	11	10	9	8	7	6	5	4	3	2	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check bit					C8				C4		C2	C1

Each check bit operates on every data bit whose position number contains a 1 in the same bit position as the position number of that check bit.

Thus, data bit positions 3, 5, 7, 9, and 11 (D1, D2, D4, D5, D7) all contain a 1 in the least significant bit of their position number as does C1; bit positions 3, 6, 7, 10, and 11 all contain a 1 in the second bit position, as does C2; and so on. Looked at another way, bit position  $n$  is checked by those bits  $C_i$  such that .

For example, position 7 is checked by bits in position 4, 2, and 1; and  $7 = 4 + 2 + 1$ .

# Example

Assume that the 8-bit input word is 00111001, with data bit D1 in the rightmost position. The calculations are as follows:

$$C1 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$C2 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$C4 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$$

$$C8 = 1 \oplus 1 \oplus 0 \oplus 0 = 0$$

- Suppose now that data bit 3 sustains an error and is changed from 0 to 1. When the check bits are recalculated, we have

$$C1 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$C2 = 1 \oplus 1 \oplus 1 \oplus 1 \oplus 0 = 0$$

$$C4 = 0 \oplus 1 \oplus 1 \oplus 0 = 0$$

$$C8 = 1 \oplus 1 \oplus 0 \oplus 0 = 0$$

- When the new check bits are compared with the old check bits, the syndrome word is formed:

	C8	C4	C2	C1
	0	1	1	1
$\oplus$	0	0	0	1
	0	1	1	0



The result is 0110, indicating that bit position 6, which contains data bit 3, is in error.

Figure illustrates the preceding calculation.

<b>Bit position</b>	12	11	10	9	8	7	6	5	4	3	2	1
<b>Position number</b>	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
<b>Data bit</b>	D8	D7	D6	D5		D4	D3	D2		D1		
<b>Check bit</b>					C8				C4		C2	C1
<b>Word stored as</b>	0	0	1	1	0	1	0	0	1	1	1	1
<b>Word fetched as</b>	0	0	1	1	0	1	1	0	1	1	1	1
<b>Position number</b>	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
<b>Check bit</b>					0				0		0	1

The data and check bits are positioned properly in the 12-bit word. Four of the data bits have a value 1 (shaded in the table), and their bit position values are XORed to produce the Hamming code 0111, which forms the four check digits. The entire block that is stored is 001101001111.

Suppose now that data bit 3, in bit position 6, sustains an error and is changed from 0 to 1. The resulting block is 001101101111, with a Hamming code of 0001. An XOR of the Hamming code and all of the bit position values for nonzero data bits results in 0110.

The nonzero result detects an error and indicates that the error is in bit position 6.

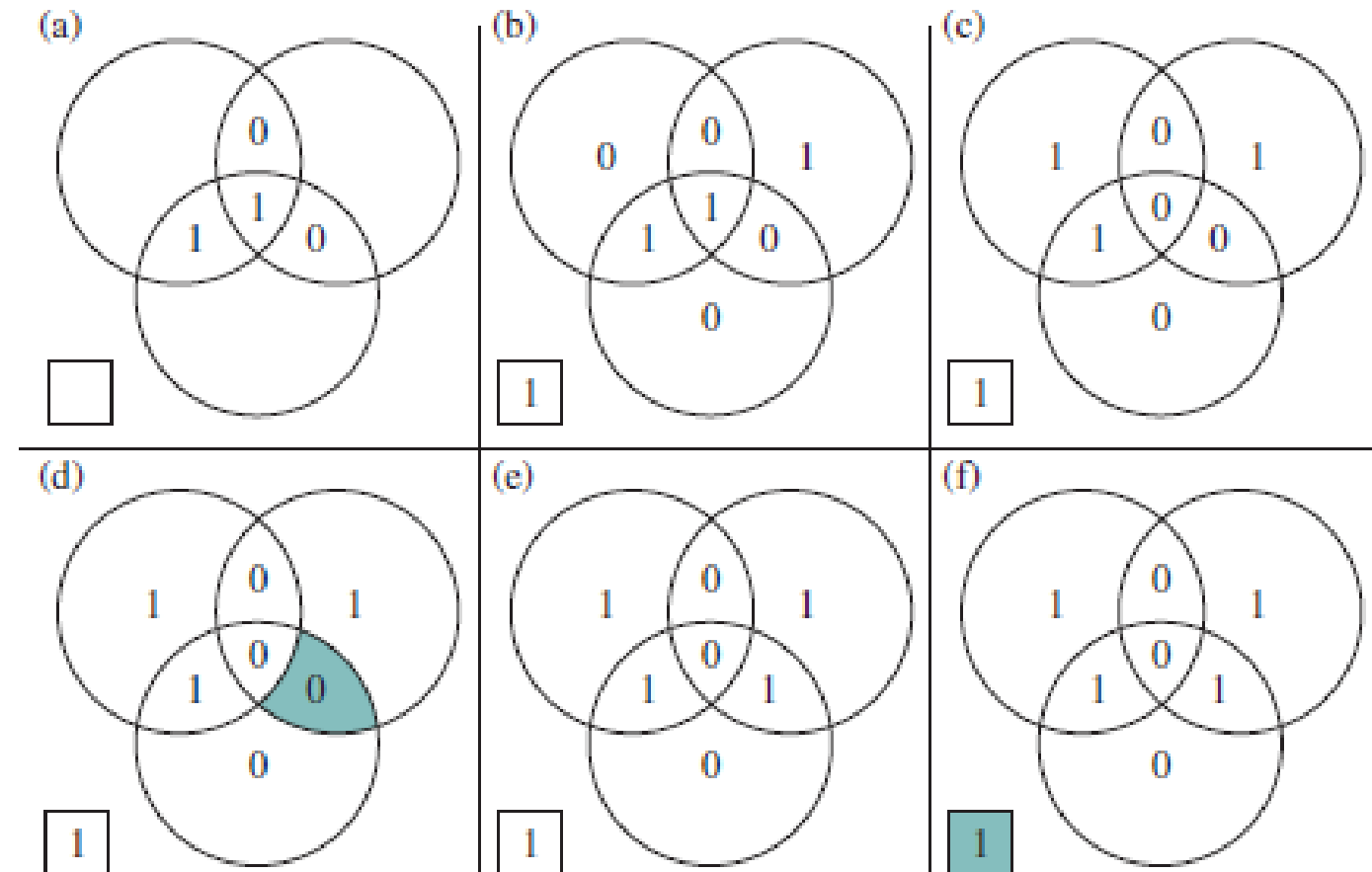
The code just described is known as a **single-error-correcting (SEC) code**.

More commonly, semiconductor memory is equipped with a **single-error-correcting, double-error-detecting (SEC-DED) code**. As Table 5.2 shows, such codes require one additional bit compared with SEC codes.

Figure illustrates how such a code works, again with a 4-bit data word.

The sequence shows that if two errors occur (c), the checking procedure goes astray (d) and worsens the problem by creating a third error (e).

To overcome the problem, an eighth bit is added that is set so that the total number of 1s in the diagram is even. The extra parity bit catches the error (f).



An error-correcting code enhances the reliability of the memory at the cost of added complexity. With a 1-bit-per-chip organization, an SEC-DED code is generally considered adequate.

For example, the IBM 30xx implementations used an 8-bit SEC-DED code for each 64 bits of data in main memory. Thus, the size of main memory is actually about 12% larger than is apparent to the user.

The VAX computers used a 7-bit SEC-DED for each 32 bits of memory, for a 22% overhead. A number of contemporary DRAMs use 9 check bits for each 128 bits of data, for a 7% overhead.