CHAPTER 10

SEMICONDUCTOR MEMORIES

10.1. Introduction

Semiconductor memory arrays capable of storing large quantities of digital information are essential to all digital systems. The amount of memory required in a particular system depends on the type of application, but, in general, the number of transistors utilized for the information (data) storage function is much larger than the number of transistors used in logic operations and for other purposes. The ever-increasing demand for larger data storage capacity has driven the fabrication technology and memory development towards more compact design rules and, consequently, toward higher data storage densities. Thus, the maximum realizable data storage capacity of single-chip semiconductor memory arrays approximately doubles every two years. On-chip memory arrays have become widely used subsystems in many VLSI circuits, and commercially available single-chip read/write memory capacity has reached 64 megabits. This trend toward higher memory density and larger storage capacity will continue to push the leading edge of digital system design.

The area efficiency of the memory array, i.e., the number of stored data bits per unit area, is one of the key design criteria that determine the overall storage capacity and, hence, the memory *cost per bit*. Another important issue is the memory access time, i.e., the time required to store and/or retrieve a particular data bit in the memory array. The access time determines the memory *speed*, which is an important performance criterion of the memory array. Finally, the static and dynamic *power consumption* of the memory array is a significant factor to be considered in the design, because of the increasing importance of low-power applications. In the following, we will investigate different types of MOS memory arrays and discuss in detail the issues of area, speed, and power consumption for each circuit type.

Memory circuits are generally classified according to the type of data storage and the type of data access. *Read-Only Memory* (ROM) circuits allow, as the name implies, only the retrieval of previously stored data and do not permit modifications of the stored information contents during normal operation. ROMs are *non-volatile* memories, i.e., the data storage function is not lost even when the power supply voltage is off. Depending on the type of data storage (data write) method, ROMs are classified as mask-programmed ROMs, Programmable ROMs (PROM), Erasable PROMs (EPROM), and Electrically Erasable PROMs (EEPROM).

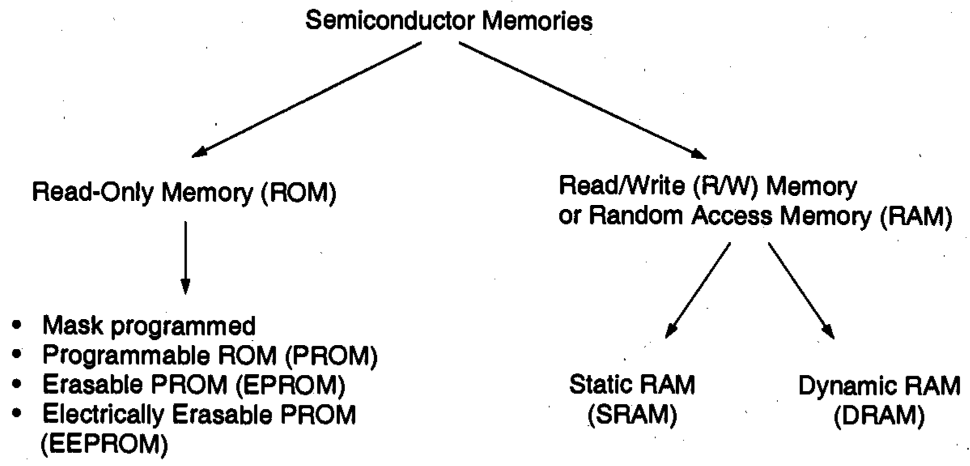


Figure 10.1. Overview of semiconductor memory types.

Read-write (R/W) memory circuits, on the other hand, must permit the modification (writing) of data bits stored in the memory array, as well as their retrieval (reading) on demand. This requires that the data storage function be *volatile*, i.e., the stored data are lost when the power supply voltage is turned off. The read-write memory circuit is commonly called *Random Access Memory* (RAM), mostly due to historical reasons. Compared to sequential-access memories such as magnetic tapes, any cell in the R/W memory array can be accessed with nearly equal access time. Based on the operation type of individual data storage cells, RAMs are classified into two main categories: *Static* RAMs (SRAM) and *Dynamic* RAMs (DRAM). Figure 10.1 shows an overview of the different memory types and their classifications.

A typical memory array organization is shown in Fig. 10.2. The data storage structure, or *core*, consists of individual memory cells arranged in an array of horizontal rows and vertical columns. Each *cell* is capable of storing one bit of binary information. Also, each memory cell shares a common connection with the other cells in the same row, and another common connection with the other cells in the same column. In this structure, there are 2^N rows, also called *word lines*, and 2^M columns, also called *bit lines*. Thus, the total number of memory cells in this array is 2^M x 2^N.

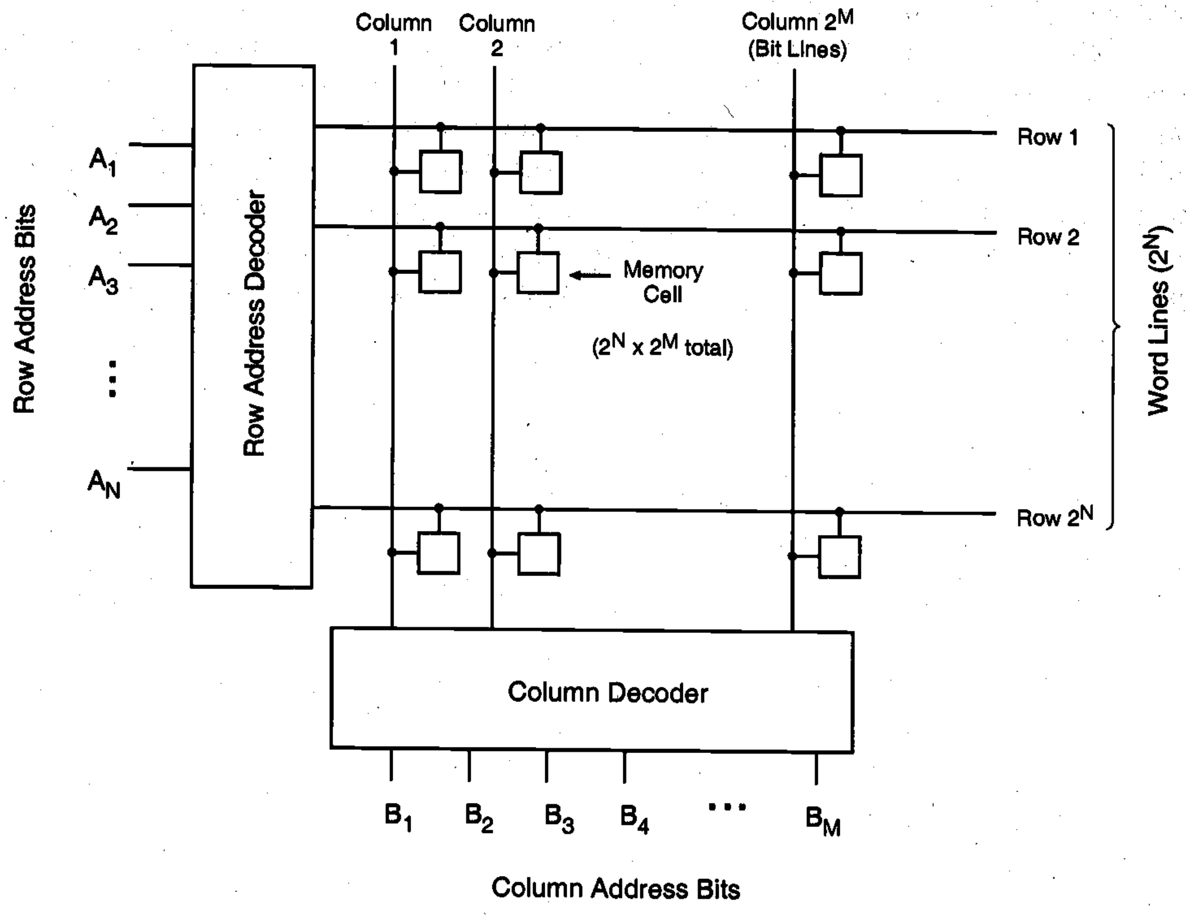
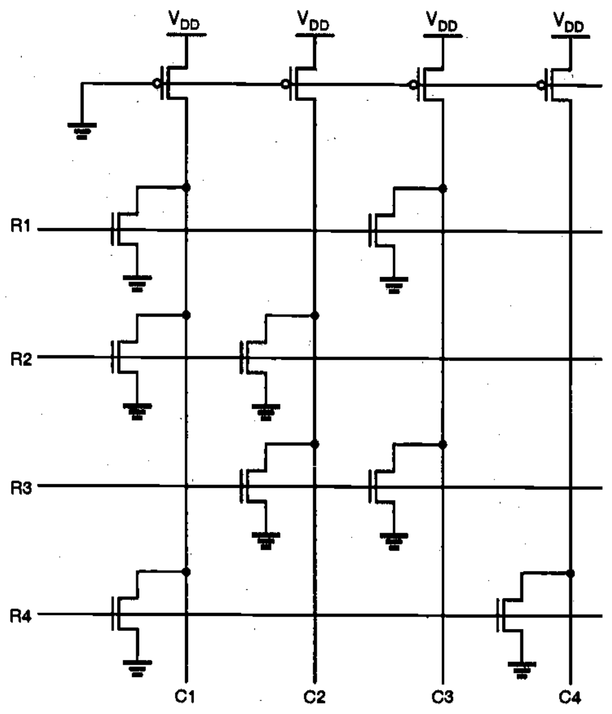


Figure 10.2. Typical random-access memory array organization.

To access a particular memory cell, i.e., a particular data bit in this array, the corresponding bit line *and* the corresponding word line must be activated (selected). The row and column selection operations are accomplished by row and column *decoders*, respectively. The row decoder circuit selects one out of 2^N word lines according to an N-bit row address, while the column decoder circuit selects one out of 2^M bit lines according to an M-bit column address. Once a memory cell or a group of memory cells are selected in this fashion, a data read and/or a data write operation may be performed on the selected single bit or multiple bits on a particular row. The column decoder circuit serves the double duties of selecting the particular columns and routing the corresponding data content in a selected row to the output.

We can see from this simple discussion that individual memory cells can be accessed for data read and/or data write operations in random order, independent of their physical locations in the memory array. Thus, the array organization examined here is called a *Random Access Memory* (RAM) structure. Notice that this organization can be used for both read-write memory arrays and read-only memory arrays. In the following sections, however, we will use the acronym RAM specifically for read-write memories, because it is the universally accepted abbreviation for this particular type of memory array.

The read-only memory array can also be seen as a simple combinational Boolean network which produces a specified output value for each input combination, i.e., for each address. Thus, storing binary information at a particular address location can be achieved by the presence or absence of a data path from the selected row (word line) to the selected column (bit line), which is equivalent to the presence or absence of a device at that particular location. In the following, we will examine two different implementations for MOS ROM arrays. Consider first the 4-bit x 4-bit memory array shown in Fig. 10.3. Here, each column consists of a pseudo-nMOS NOR gate driven by some of the row signals, i.e., the word lines.



R1	R2	R3	R4	C1	C2	C3	C4
1	0	0	0	0	1	0	1
0	1	0	0	0	0	1	1
0	0	1	0	1	0	0	1
0	0	0	1	0	1	1	0

Figure 10.3. Example of a 4-bit x 4-bit NOR-based ROM array.

As described in the previous section, only one word line is activated (selected) at a time by raising its voltage to V_{DD}, while all other rows are held at a low voltage level. If an active transistor exists at the cross point of a column and the selected row, the column voltage is pulled down to the logic low level by that transistor. If no active transistor exists at the cross point, the column voltage is pulled high by the pMOS load device. Thus, a logic "1"-bit is stored as the absence of an active transistor, while a logic "0"-bit is stored as the presence of an active transistor at the crosspoint. To reduce static power consumption, the pMOS load transistors in the ROM array shown in Fig. 10.3 can also be driven by a periodic precharge signal, resulting in a *dynamic* ROM.

In actual ROM layout, the array can be initially manufactured with nMOS transistors at every row-column intersection. The "1"-bits are then realized by omitting the drain or source connection, or the gate electrode of the corresponding nMOS transistors in the final metallization step. Figure 10.4 shows four nMOS transistors in a NOR ROM array, forming the intersection of two metal bit lines and two polysilicon word lines. To save silicon area, the transistors in every two adjacent rows are arranged to share a common ground line, also routed in n-type diffusion. To store a "0"-bit at a particular address location, the drain diffusion of the corresponding transistor must be connected to the metal bit line via a metal-to-diffusion contact. Omission of this contact, on the other hand, results in a stored "1"-bit.

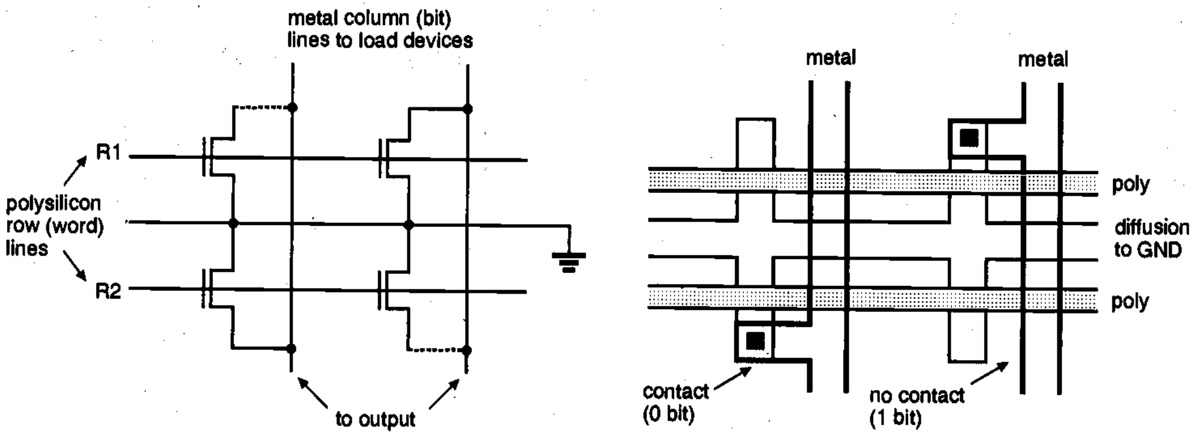


Figure 10.4. Layout example of a NOR ROM array.

Figure 10.5 shows a larger portion of the ROM array, except for the pMOS load transistors connected to the metal columns. Here, the 4-bit x 4-bit ROM array shown in Fig. 10.3 is realized using the contact-mask programming methodology described above. Note that only 8 of the 16 nMOS transistors fabricated in this structure are actually connected to the bit lines via metal-to-diffusion contacts. In reality, the metal column lines are laid out directly on top of diffusion columns to reduce the horizontal dimension of the ROM array.

A different NOR ROM layout implementation is based on deactivation of the nMOS transistors by raising their threshold voltages through channel implants. Figure 10.6 shows the circuit diagram of a NOR ROM array in which every two rows of nMOS transistors share a common ground connection, and every drain diffusion contact to the metal bit line is shared by two adjacent transistors. In this case, all nMOS transistors are already connected to the column lines (bit lines); therefore, storing a "1"-bit at a particular location by omitting the corresponding drain contact is not possible. Instead, the nMOS transistor corresponding to the stored "1"-bit can be deactivated, i.e., permanently turned off, by raising its threshold voltage *above* the V_{OH} level through a selective channel implant during fabrication.

The alternative layout of the 4-bit x 4-bit bit ROM array example (Fig. 10.3), which is based on implant-mask programming, is shown in Fig. 10.7. Note that in this case, each threshold voltage implant signifies a stored "1"-bit, and all other (non-implanted) transistors correspond to stored "0"-bits. Since each diffusion-to-metal contact in this structure is shared by two adjacent transistors, the implant-mask ROM layout can yield a higher core density, i.e., a smaller silicon area per stored bit, compared to the contact-mask ROM layout.

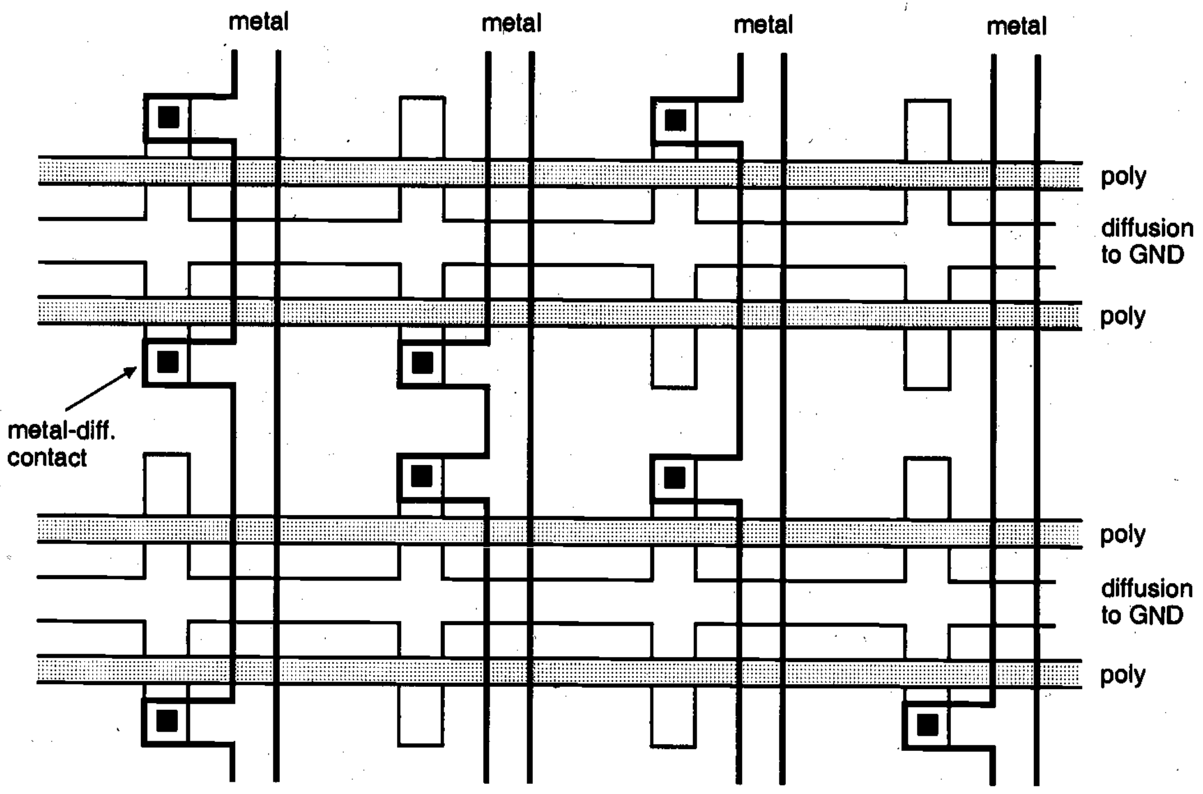


Figure 10.5. Layout of the 4-bit x 4-bit NOR ROM array example shown in Fig. 10.3.

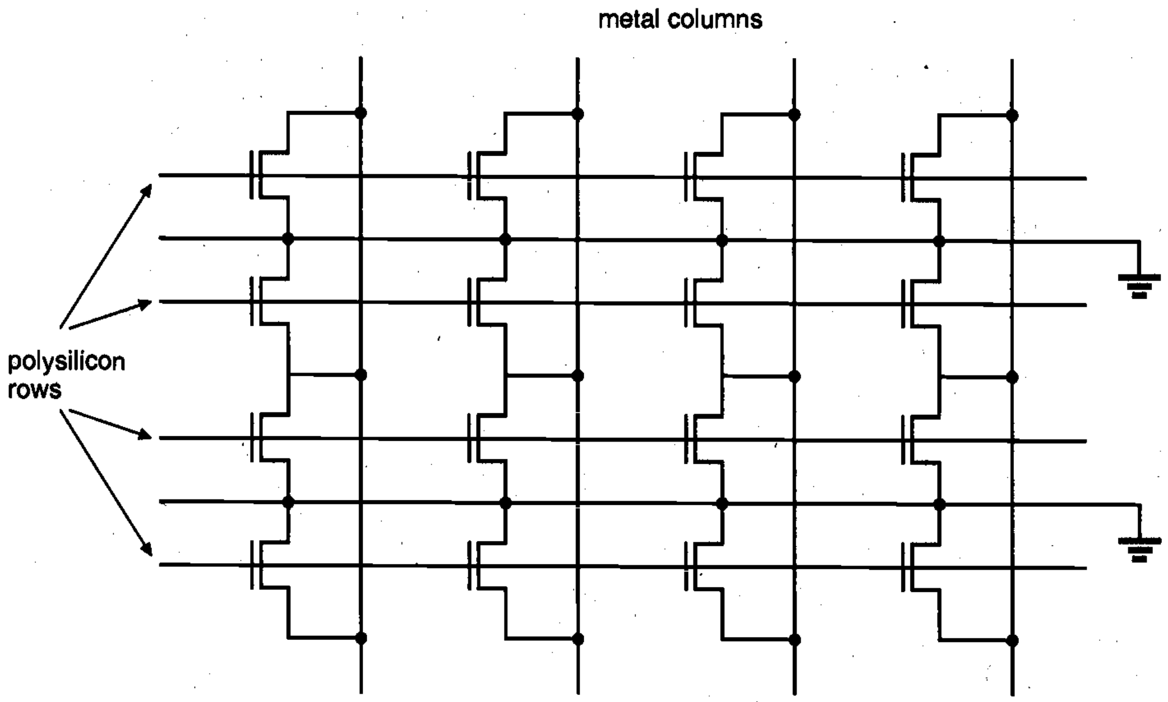


Figure 10.6. Arrangement of the nMOS transistors in the implant-mask programmable NOR ROM array. Every metal-to-diffusion contact is shared by two adjacent devices.

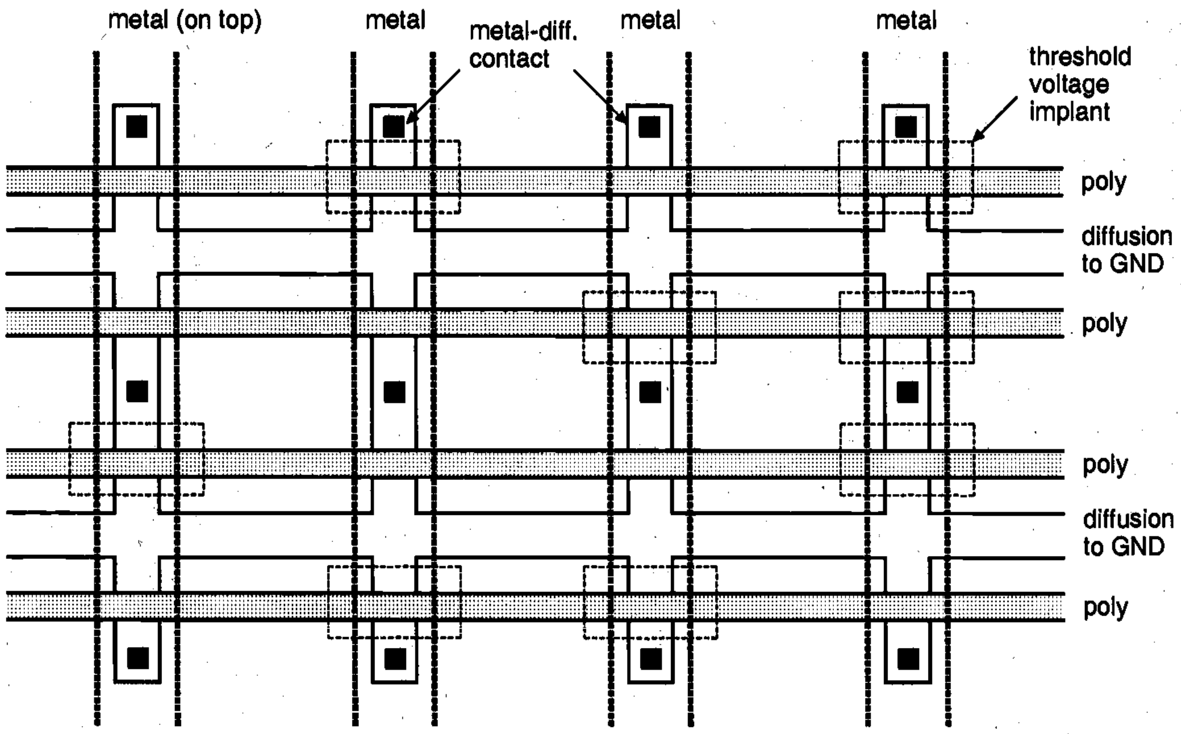
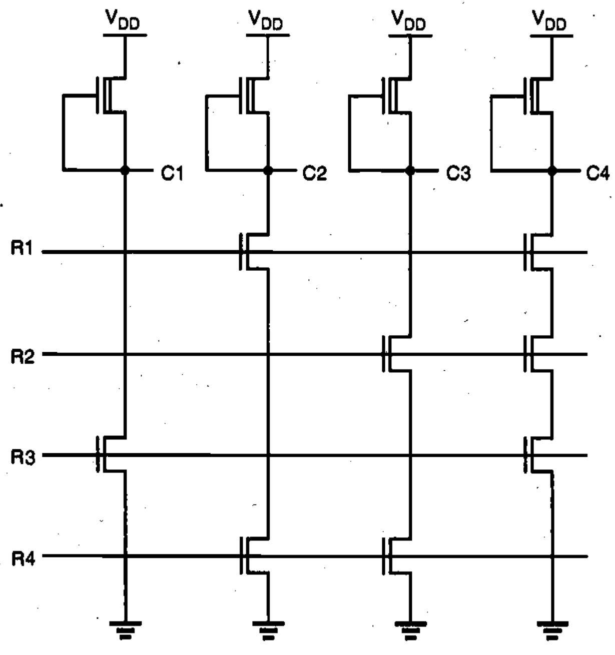


Figure 10.7. Layout of the 4-bit x 4-bit NOR ROM array example shown in Fig. 10.3. The threshold voltages of "1"-bit transistors are raised above V_{DD} through implant.

Next, we will examine a significantly different ROM array design, which is also called a NAND ROM (Fig. 10.8). Here, each bit line consists of a depletion-load NAND gate, driven by some of the row signals, i.e., the word lines. In normal operation, all word lines are held at the logic-high voltage level except for the selected line, which is pulled *down* to logic-low level. If a transistor exists at the crosspoint of a column and the selected row, that transistor is turned off and the column voltage is pulled high by the load device. On the other hand, if no transistor exists (shorted) at that particular crosspoint, the column voltage is pulled low by the other nMOS transistors in the multi-input NAND structure. Thus, a logic "1"-bit is stored by the presence of a transistor that can be deactivated, while a logic "0"-bit is stored by a shorted or normally on transistor at the crosspoint.

As in the NOR ROM case, the NAND-based ROM array can be fabricated initially with a transistor connection present at every row-column intersection. A "0"-bit is then stored by lowering the threshold voltage of the corresponding nMOS transistor at the cross point through a channel implant, so that the transistor remains on regardless of the gate voltage (i.e., the nMOS transistor at the intersection becomes a depletion-type device). The availability of this process step is also the reason why depletion-type nMOS load transistors are used instead of pMOS loads in the example shown above. Figure 10.9 shows a sample 4-bit x 4-bit layout of the implant-mask NAND ROM array. Here, vertical columns of n-type diffusion intersect at regular intervals with horizontal rows of polysilicon, which results in an nMOS transistor at each intersection point. The transistors with threshold voltage implants operate as normally-on depletion devices, thereby providing a continuous current path regardless of the gate voltage level. Since this structure has no contacts embedded in the array, it is much more compact than the NOR ROM array. However, the access time is usually slower than the NOR ROM, due to multiple series-connected nMOS transistors in each column. An alternative layout method for the NAND ROM array is not to place the nMOS transistors at "0"-bit locations, as in the case of the PLA (Programmable Logic Array) layout generation. In this case, the missing transistor is simply replaced by a metal line, instead of using a threshold voltage implant at that location.



R1	R2	R3	R4	C1	C2	C3	C4
0	1	1	1	0	1	0	1
1	0	1	1	0	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	0	1	1	0

Figure 10.8. A 4-bit x 4-bit NAND-based ROM array.

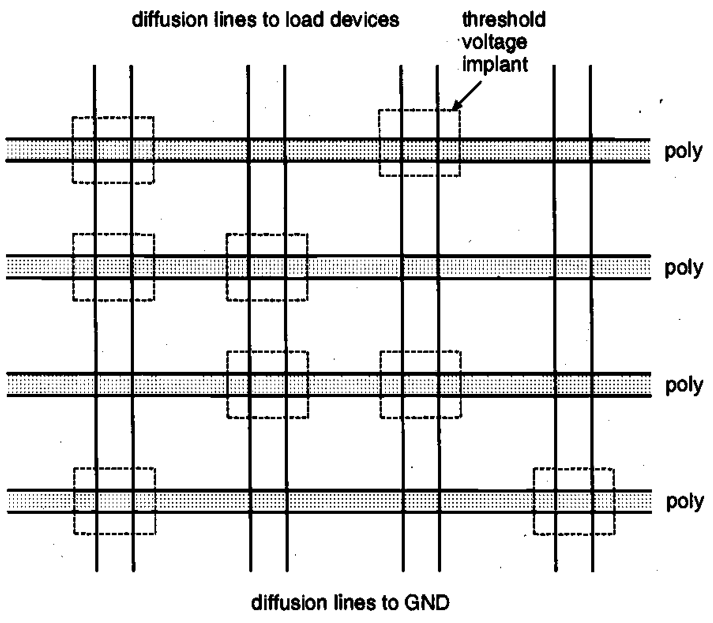
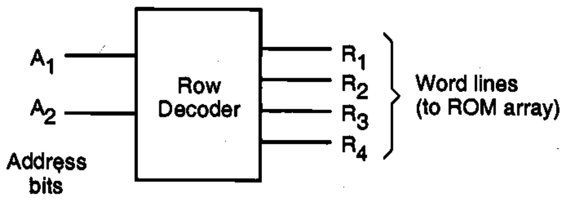


Figure 10.9. Implant-mask layout of the NAND ROM array example in Fig. 10.8. The threshold voltages of "0"-bit transistors are lowered below 0 V through implant.

Now we will turn our attention to the circuit structures of row and column address decoders, which select a particular memory location in the array, based on the binary row and column addresses. A row decoder designed to drive a NOR ROM array must, by definition, select one of the 2^N word lines by raising its voltage to V_{OH}. As an example, consider the simple row address decoder shown in Fig. 10.10, which decodes a two-bit row address and selects one out of four word lines by raising its level.



A_1	A_2	R_1	R_2	R_3	R_4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Figure 10.10. Row address decoder example for 2 address bits and 4 word lines.

A most straightforward implementation of this decoder is another NOR array, consisting of 4 rows (outputs) and 4 columns (two address bits and their complements). Note that this NOR-based decoder array can be built just like the NOR ROM array, using the same selective programming approach (Fig. 10.11). The ROM array and its row decoder can thus be fabricated as two adjacent NOR arrays, as shown in Fig. 10.12.

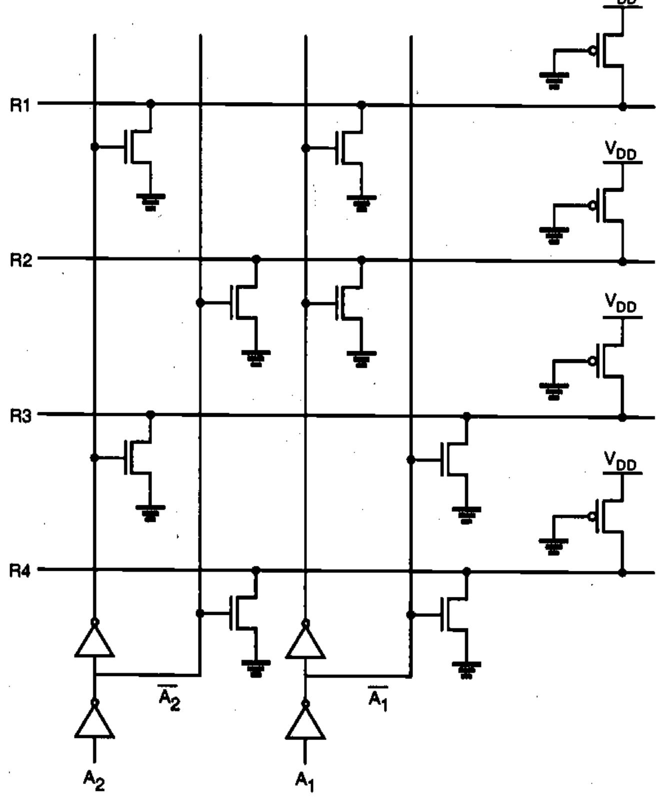


Figure 10.11. NOR-based row decoder circuit for 2 address bits and 4 word lines.

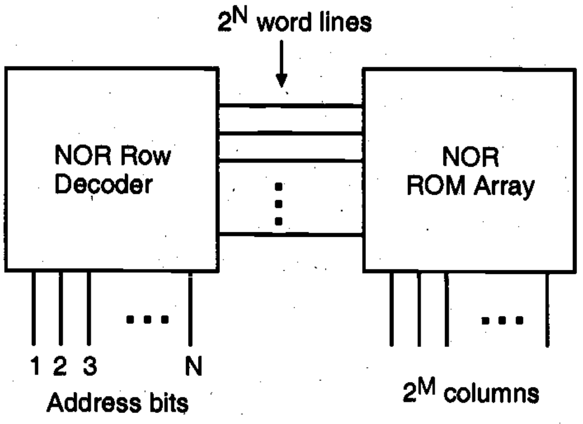


Figure 10.12. Implementation of the row decoder circuit and the ROM array as two adjacent NOR planes.

A row decoder designed to drive a NAND ROM, on the other hand, must lower the voltage level of the selected row to logic "0" while keeping all other rows at a logic-high level. This function can be implemented by using an N-input NAND gate for each of the row outputs. The truth table of a simple address decoder for four rows and the double NAND-array implementation of the decoder and the ROM are shown in Fig. 10.13. As in the NOR ROM case, the row address decoder of the NAND ROM array can thus be realized using the same layout strategy as the memory array itself.

A_1	A_2	R_1	R_2	R_3	R_4
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

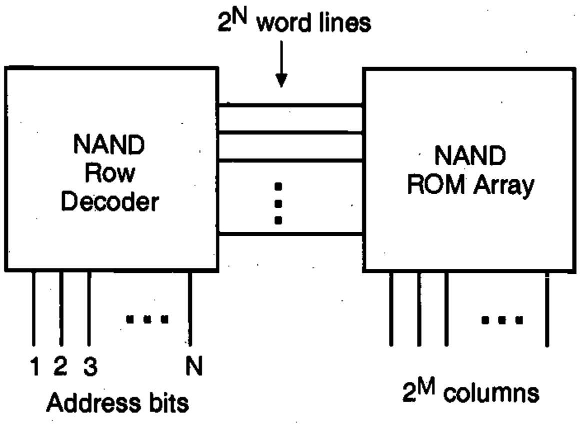


Figure 10.13. Truth table of a row decoder for the NAND ROM array, and implementation of the row decoder circuit and the ROM array as two adjacent NAND planes.

The column decoder circuitry is designed to select one out of 2^M bit lines (columns) of the ROM array according to an M-bit column address, and to route the data content of the selected bit line to the data output. A straightforward but costly approach would be to connect an nMOS pass transistor to each bit-line (column) output, and to selectively drive one out of 2^M pass transistors by using a NOR-based column address decoder, as shown in Fig. 10.14. In this arrangement, only one nMOS pass transistor is turned on at a time, depending on the column address bits applied to the decoder inputs. The conducting pass transistor routes the selected column signal to the data output. Similarly, a number of columns can be chosen at a time, and the selected columns can be routed to a *parallel* data output port.

Note that the number of transistors required for this column decoder implementation is $2^M(M+1)$, i.e., 2^M pass transistors for each bit line and $M\,2^M$ transistors for the decoder circuit. This number can quickly become excessive for large M, i.e., for a large number of bit lines.

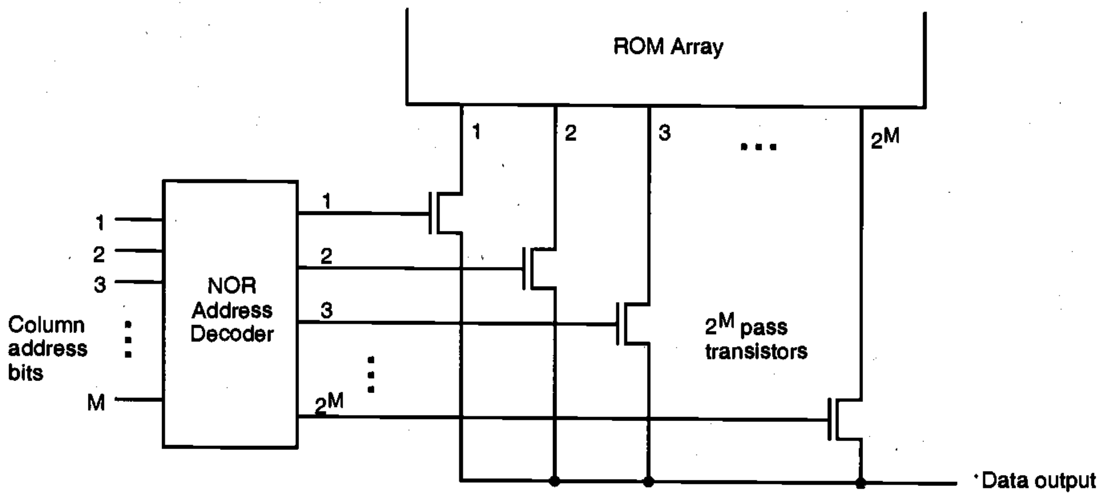


Figure 10.14. Bit-line (column) decoder arrangement using a NOR address decoder and nMOS pass transistors for every bit line.

An alternative design of the column decoder circuit is to build a binary selection tree consisting of consecutive stages, as shown in Fig. 10.15. In this case, the pass transistor network is used to select one out of every two bit lines at each stage (level), whereas the column address bits drive the gates of the nMOS pass transistors. Notice that a NOR address decoder is not needed for this decoder tree structure, thereby reducing the number of transistors significantly although it requires M additional inverters ($2M$ transistors) for complementing column address bits. The example shown in Fig. 10.15 is a column decoder tree for eight bit lines, which requires three column address bits (and their complements) to select one of the eight columns.

One drawback of the decoder tree approach is that the number of series-connected nMOS pass transistors in the data path is equal to the number of column address bits, M. This situation can cause a long data access time, since the decoder delay time depends on the equivalent series resistance of the decoder branch that directs the column data to the output. To overcome this constraint, column address decoders can be built as a combination of the two structures presented here, i.e., consisting of relatively shallow, partial tree decoders and of additional selection circuits similar to that shown in Fig. 10.14.

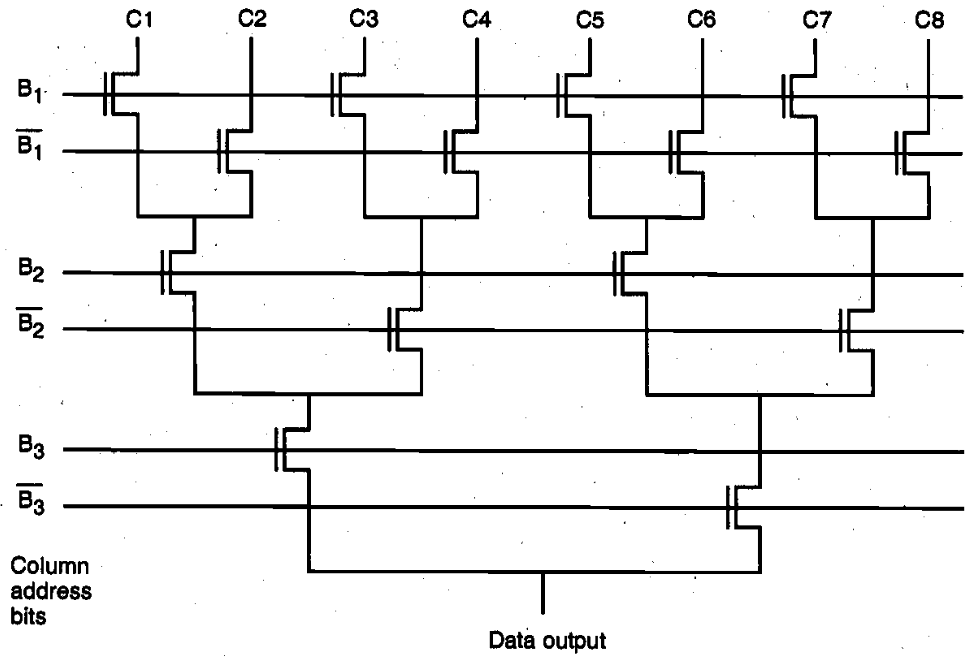


Figure 10.15. Column decoder circuit for eight bit lines, implemented as a binary tree decoder which is driven directly by the three column address bits.

Example 10.1.

In the following example, the design of a 32-kbit NOR ROM array will be considered, and the relevant design issues related to access time analysis will be examined.

A 32-kbit ROM array consists of 2^{15} = 32,768 individual memory cells, which are arranged into a certain number of the rows and columns, as already explained in the beginning of this chapter. Note that the *sum* of row address bits and the column address bits must be equal to 15 in a 32-kbit array; the actual number of rows and columns in the memory array may be determined according to this and other constraints, as will be demonstrated in the following.

Assume that the ROM array under consideration has 7 row address bits and 8 column address bits, which results in a memory array of 128 rows and 256 columns. A section of the memory cell layout is shown in Fig. 10.16, in which the programming is done by an implant-mask to adjust the threshold voltages of those transistors which are to be left inactive (cf. Fig. 10.7). To provide a compact layout, the drain contact shown here is actually shared by two adjacent transistors, and the metal bit line (column) runs directly on top of the diffusion columns shown in this example. To simplify the mask view, the metal column line is not shown in Fig. 10.16.

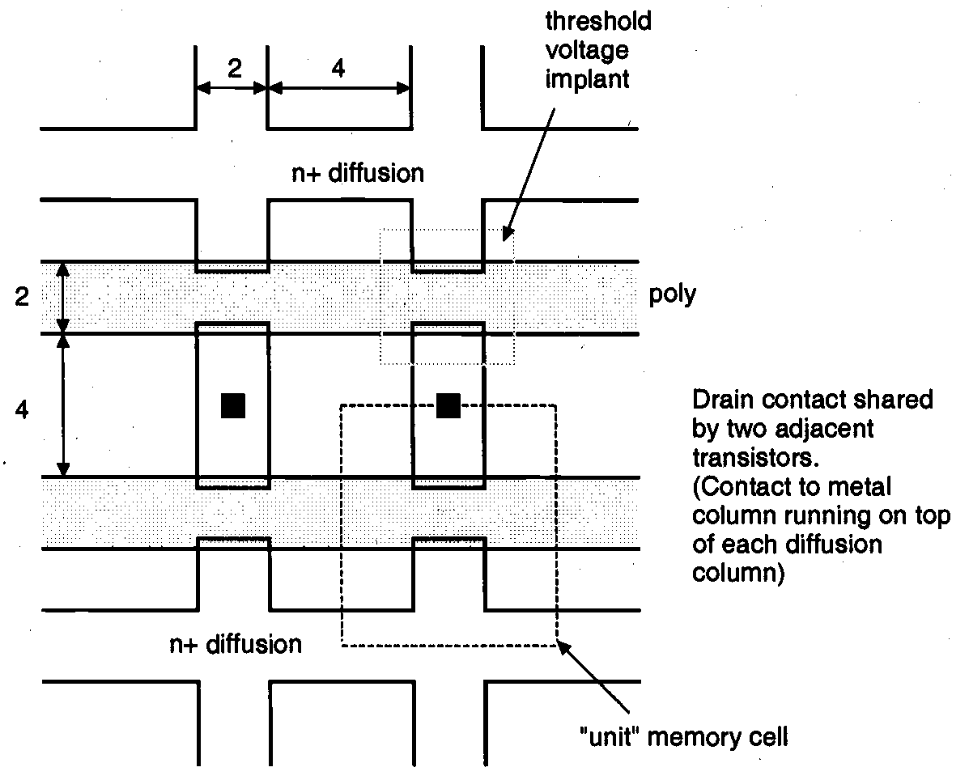

threshold voltage implant
2
4
n+ diffusion
2
poly
4
Drain contact shared by two adjacent transistors. (Contact to metal column running on top of each diffusion column)
n+ diffusion
"unit" memory cell

Figure 10.16. Simplified mask layout of the implant-mask programmable NOR ROM array considered in the example. All dimensions are in micrometers ($W = 2$ μm, $L = 1.5$ μm).

Other relevant parameters of the structure are:

$\mu_n C_{ox} = 20$ μA/V^2
$C_{ox} = 3.47$ μF/cm^2
Poly sheet resistance = 20 Ω/square

First, we calculate the row resistance and row capacitance per bit, i.e., per memory cell. It is assumed that the row capacitance of each memory cell is dictated primarily by the thin oxide capacitance of the nMOS transistor, and that the polysilicon capacitance outside the active region is negligible. The row resistance associated with each memory cell, on the other hand, is calculated by summing the number of polysilicon squares (in this case, three) over the unit cell.

$$C_{row} = C_{ox} \cdot W \cdot L = 10.4 \text{ fF/bit}$$

$$R_{row} = (\text{\# of squares}) \times (\text{Poly sheet resistance}) = 60 \ \Omega/\text{bit}$$

We note that each polysilicon row (word line) in this memory array is actually a distributed RC transmission line. Figure 10.17 depicts a few components of this structure, which ultimately affects the row access time of the memory array in consideration. It can be seen that once the row is selected by the row address decoder, i.e., the row voltage is forced to a logic-high level at one end of the word line, the gate voltage of the last (256th) transistor in that row will rise *last* because of the RC line delay. The propagation delay time of the gate voltage of the 256th transistor in the row determines the row access time, t_{row} (Fig. 10.18).

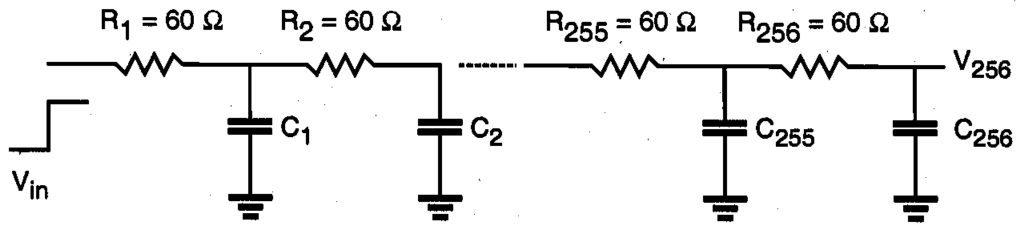


Figure 10.17. RC transmission line representation of the polysilicon word line.

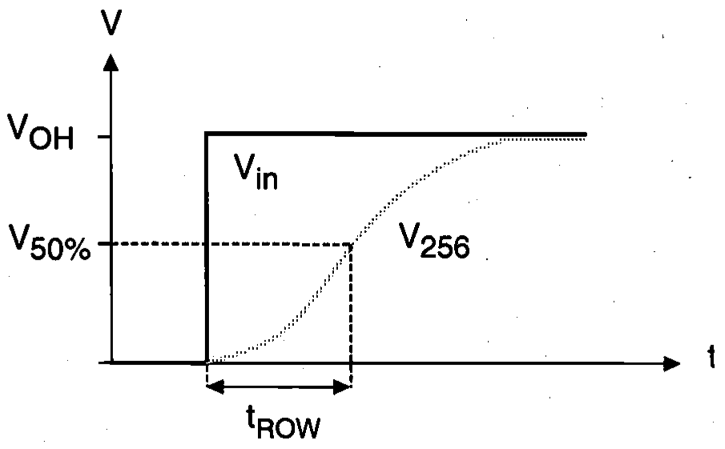


Figure 10.18. Definition of the row access time in a memory array with 256 columns.

By neglecting the signal propagation delay associated with the row address decoder circuit, and assuming that the row (word line) is driven by an ideal step voltage waveform, the row access time can be approximated by using the following empirical formula,

$$t_{row} \approx 0.38 \cdot R_T \cdot C_T = 15.53 \text{ ns}$$

where

$$R_T = \sum_{all\ columns} R_i = 15.36 \text{ k}\Omega$$

$$C_T = \sum_{all\ columns} C_i = 2.66 \text{ pF}$$

A more accurate RC delay value can be calculated by using the *Elmore time constant* for RC ladder circuits, as follows.

$$t_{row} = \sum_{k=1}^{256} R_{jk} C_k = 20.52 \text{ ns} \qquad \text{where} \qquad R_{jk} = \sum_{j=1}^{k} R_j$$

The row access time t_{row} is the time delay associated with selecting and activating one of the 128 word lines in this ROM array.

To calculate the column access time, we need to consider one of the 128-input NOR gates, which represent the bit lines in this ROM structure. The pseudo-nMOS NOR gate corresponding to each column is designed by using a pMOS load transistor, in which the (W/L) ratio is (4/1.5) (Fig. 10.19).

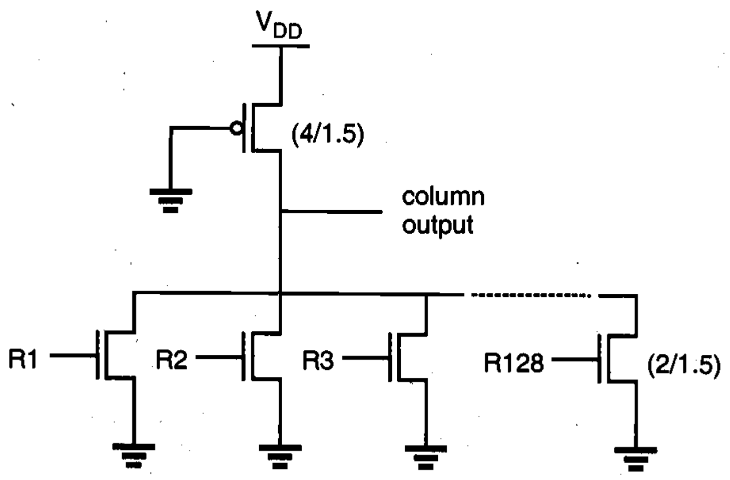


Figure 10.19. 128-input NOR gate representation of a column (bit line) in the ROM array.

The combined column capacitance loading the output node of the 128-input NOR gate can be approximated by adding up the parasitic capacitances of each driver transistor.

$$C_{column} = 128 \times \left(C_{gd,driver} + C_{db,driver} \right) \approx 1.5 \text{ pF}$$

where

$$C_{gd,driver} + C_{db,driver} = 0.0118 \text{ pF/word line}$$

Since only one word line (row) is activated at a time by the row address decoder, the NOR gate representing the column can actually be reduced to the inverter shown in Fig. 10.20. To calculate the column access time, we must consider the worst-case signal propagation delay τ_{PHL} (for falling output voltage) for this inverter. By using the propagation delay formula (6.22) in Chapter 6, the worst-case column access time can thus be calculated as t_{column} = 18 ns. Note that the propagation delay for rising output signals τ_{PLH} is not considered here because the bit line (column) is precharged high *before* each row access operation.

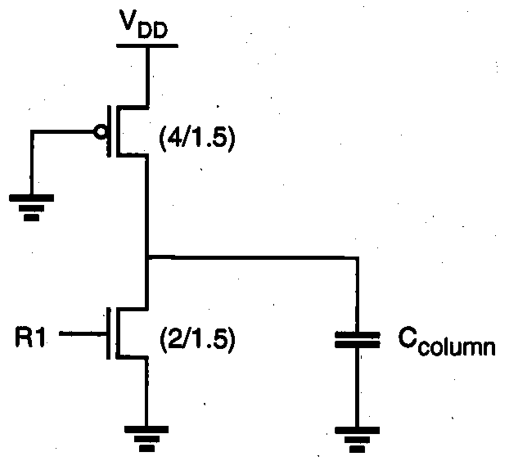


Figure 10.20. Equivalent inverter circuit representing the bit line (column). Note that only one word line (row) is activated at a time.

The total access time for this ROM array consisting of 128 rows and 256 columns is found by adding the row and column access times, as $t_{access} = 38.5$ ns.

At this point, we may also consider a different arrangement for the ROM array, which consists of 256 rows and 128 columns, i.e., using 8 row address bits and 7 column address bits. Since the number of columns in this arrangement is half of the number of columns in the previous example, the total row resistance and row capacitance values will be approximately half of those obtained for the 256-column arrangement. Consequently, the row access time for this structure will be *one-fourth* of the row access time found for the 256-column ROM array, i.e., approximately 5 ns. The number of rows in the new arrangement, on the other hand, is 256, which results in a column capacitance approximately *twice* that in the previous case. Thus, the column access time of the 256-by-128 memory array will be twice as large, approximately 36 ns. In conclusion, we find that the total access time of the 128-by-256 array examined initially is shorter than that of the 256-by-128 array.

10.3. Static Read-Write Memory (SRAM) Circuits

As already explained in Section 10.1, read-write (R/W) memory circuits are designed to permit the modification (writing) of data bits to be stored in the memory array, as well as their retrieval (reading) on demand. The memory circuit is said to be *static* if the stored data can be retained indefinitely (as long as a sufficient power supply voltage is provided), without any need for a periodic refresh operation. We will examine the circuit structure and the operation of simple SRAM cells, as well as the peripheral circuits designed to read and write the data.

The data storage cell, i.e., the 1-bit memory cell in static RAM arrays, invariably consists of a simple latch circuit with two stable operating points (states). Depending on the preserved state of the two-inverter latch circuit, the data being held in the memory cell will be interpreted either as a logic "0" or as a logic "1." To access (read and write) the data contained in the memory cell via the bit line, we need at least one switch, which is controlled by the corresponding word line, i.e., the row address selection signal (Fig. 10.21(a)). Usually, two complementary access switches consisting of nMOS pass transistors are implemented to connect the 1-bit SRAM cell to the complementary bit lines (columns). This can be likened to turning the car steering wheel with both left and right hands in complementary directions.

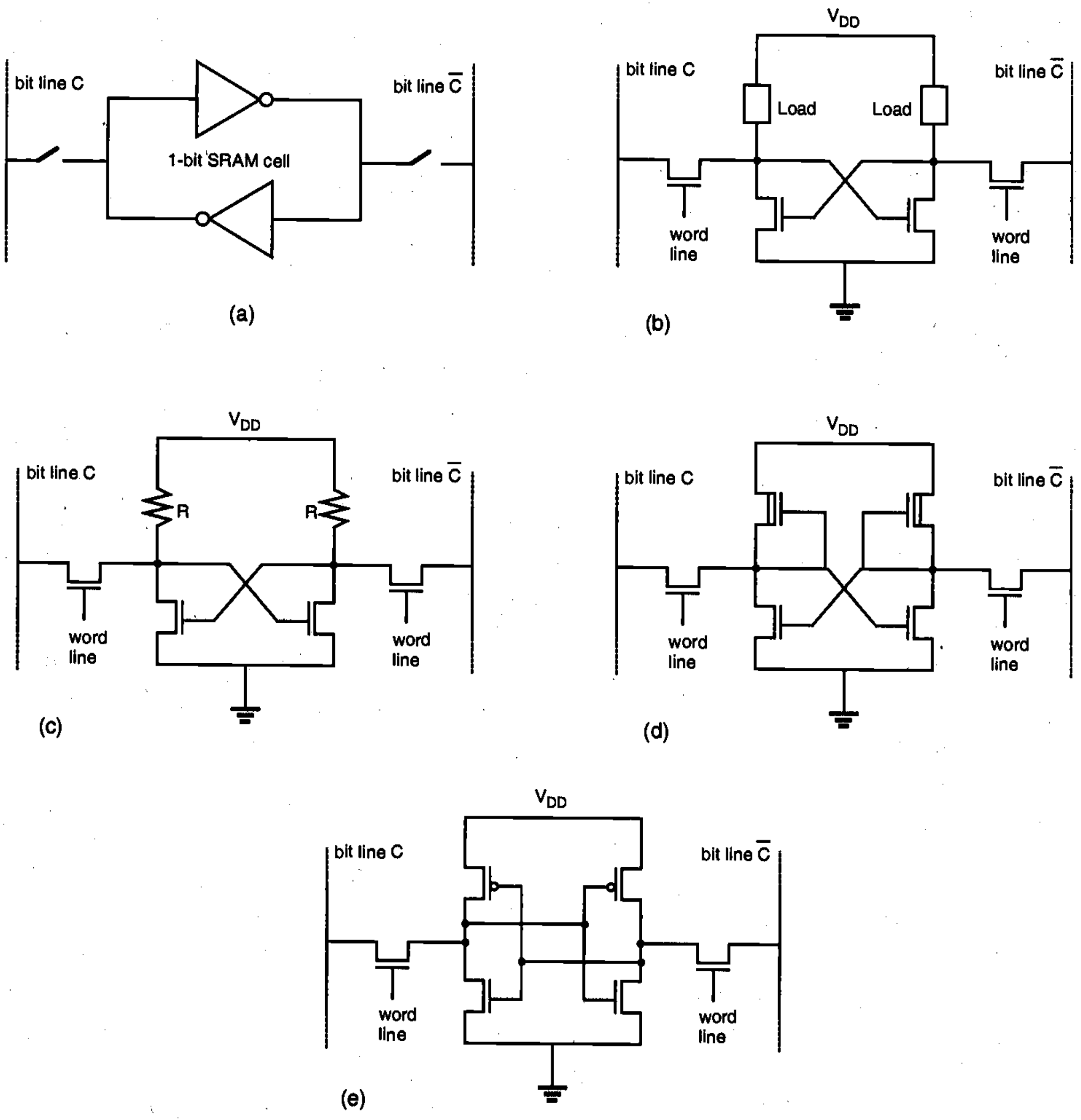


Figure 10.21. Various configurations of the static RAM cell. (a) Symbolic representation of the two-inverter latch circuit with access switches. (b) Generic circuit topology of the MOS static RAM cell. (c) Resistive-load SRAM cell. (d) Depletion-load nMOS SRAM cell. (e) Full CMOS SRAM cell.

Figure 10.21(b) shows the generic structure of the MOS static RAM cell, consisting of two cross-coupled inverters and two access transistors. The load devices may be polysilicon resistors, depletion-type nMOS transistors, or pMOS transistors, depending on the type of the memory cell. The pass gates acting as data access switches are enhancement-type nMOS transistors.

The use of resistive-load inverters with undoped polysilicon resistors in the latch structure typically results in a significantly more compact cell size, compared with the other alternatives (Fig. 10.21(c)). This is true since the resistors can be stacked on top of the cell (using double-polysilicon technology), thereby reducing the cell size to four transistors, as opposed to the six-transistor cell topologies. If multiple polysilicon layers are available, one layer can be used for the gates of the enhancement-type nMOS transistors, while another level is used for load resistors and interconnects.

In order to attain acceptable noise margins and output pull-up times for the resistive-load inverter, the value of the load resistor has to be kept relatively low, as already examined in Section 5.2. On the other hand, a high-valued load resistor is required in order to reduce the amount of standby current being drawn by each memory cell. Thus, there is a trade-off between the high resistance required for low power and the requirement to provide wider noise margins and high speed. The power consumption issue will be addressed later in more detail.

The six-transistor depletion-load nMOS SRAM cell shown in Fig. 10.21(d) can be easily implemented with one polysilicon and one metal layer, and the cell size tends to be relatively small, especially with the use of buried metal-diffusion contacts. The static characteristics and the noise margins of this memory cell are typically better than those of the resistive-load cell. The static power consumption of the depletion-load SRAM cell, however, makes it an unsuitable candidate for high-density SRAM arrays.

The full CMOS SRAM cell shown in Fig. 10.21(e) achieves the lowest static power dissipation among the various circuit configurations presented here. In addition, the CMOS cell offers superior noise margins and switching speed as well. The comparative advantages and disadvantages of the CMOS static RAM cell will be investigated in depth later in this section.

SRAM Operation Principles

Figure 10.22 shows a typical four-transistor resistive-load SRAM cell widely used in high-density memory arrays, consisting of a pair of cross-coupled inverters. The two stable operating points of this basic latch circuit are used to store a one-bit piece of information; hence, this pair of cross-coupled inverters make up the central component of the SRAM cell. To perform read and write operations, we use two nMOS pass transistors, both of which are driven by the row select signal, *RS*. Note that the SRAM cell shown in Fig. 10.22 is accessed via two bit lines or columns, instead of one. This complementary column arrangement allows for a more reliable operation.

When the word line (*RS*) is not selected, i.e., when the voltage level of line *RS* is equal to logic "0," the pass transistors M3 and M4 are turned off. The simple latch circuit consisting of two cross-connected inverters preserves one of its two stable operating points; hence, data is being *held*. At this point, consider the two columns, *C* and $\overline{C}$. If all word lines in the SRAM array are inactive, the relatively large column capacitances are

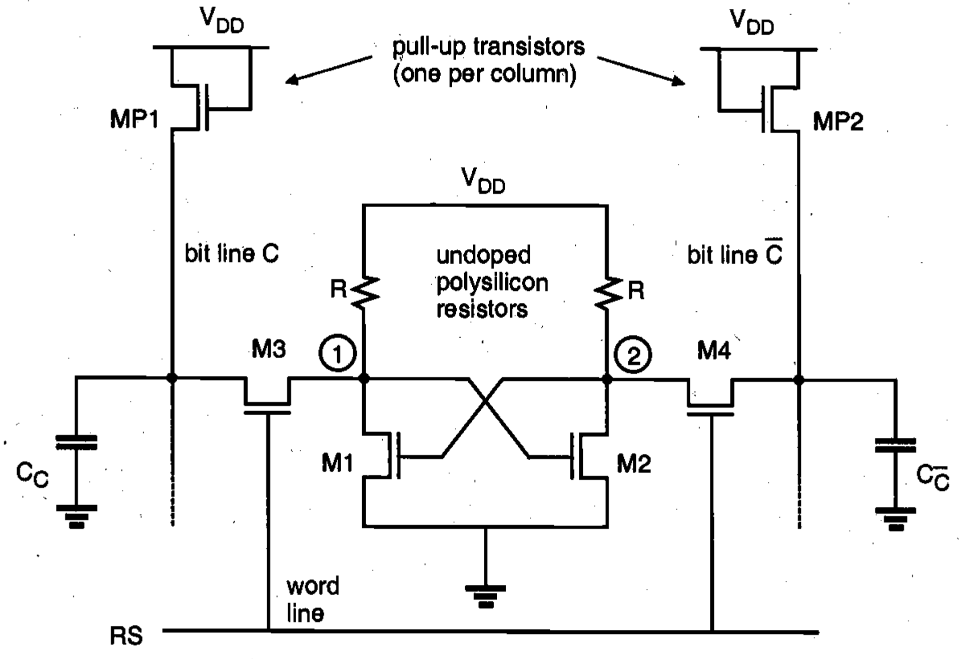


Figure 10.22. Basic structure of the resistive-load SRAM cell, shown with the column pull-up transistors.

charged-up by the column pull-up transistors, MP1 and MP2. Since both transistors operate in saturation, the steady-state voltage level V_C for both columns is determined by the following relationship:

$$V_{DD} - V_C = V_{T0} + \gamma\left(\sqrt{|2\phi_F| + V_C} - \sqrt{|2\phi_F|}\right) \tag{10.1}$$

Assuming $V_{DD} = 5$ V, $V_{T0} = 1$ V, $|2\phi_F| = 0.6$ V, and $\gamma = 0.4$ V$^{1/2}$, this voltage level is found to be approximately equal to 3.5 V. Note that the voltage levels of the two complementary bit lines (columns) are equal during this phase.

Now assume that we select the memory cell by raising its word line voltage to logic "1," hence, the pass transistors M3 and M4 are turned on. Once the memory cell is selected, four basic operations may be performed on this cell.

a) Write "1" operation:	The voltage level of column $\overline{C}$ is forced to logic-low by the data-write circuitry. The driver transistor M1 turns off. The voltage V_1 attains a logic-high level, while V_2 goes low.
b) Read "1" operation:	The voltage of column C retains its precharge level while the voltage of column $\overline{C}$ is pulled down by M2 and M4. The data-read circuitry detects the small voltage difference ($V_C > V_{\overline{C}}$) and amplifies it as a logic "1" data output.

c) Write "0" operation: The voltage level of column C is forced to logic-low by the data-write circuitry. The driver transistor M2 turns off. The voltage V_2 attains a logic-high level, while V_1 goes low.

d) Read "0" operation: The voltage of column $\overline{C}$ retains its precharge level while the voltage of column C is pulled down by M1 and M3. The data-read circuitry detects the small voltage difference $(V_C < V_{\overline{C}})$ and amplifies it as a logic "0" data output.

Typical voltage waveforms associated with the word line RS and the two pseudo-complementary bit lines are shown qualitatively in Fig. 10.23. Note that the voltage difference between the two columns during a read operation may be only a few hundred millivolts, which must be detected by the data-read circuitry. The reason for this is that the two nMOS transistors in series (e.g., M1 and M3 for read "0") pulling down the column during the read phase cannot discharge the large column capacitance quickly.

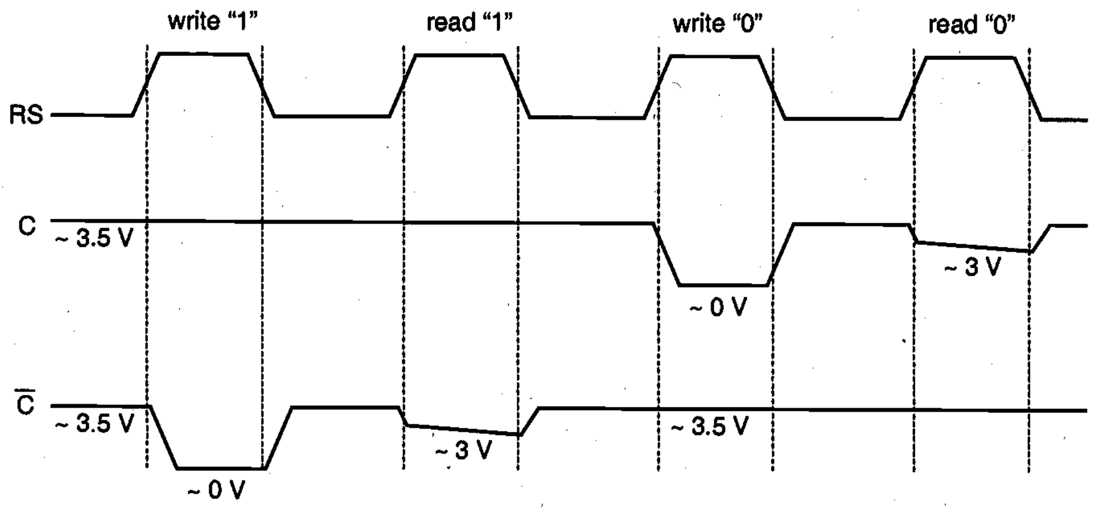


Figure 10.23. Typical row and column voltage waveforms of the resistive-load SRAM shown in Fig. 10.22 during read and write.

Power Consumption

To estimate the standby power consumption of the static read-write memory cell, assume that a "1"-bit is stored in the cell. This means that the driver transistor M1 is turned off, while the driver transistor M2 is conducting, resulting in $V_1 = V_{OH}$ and $V_2 = V_{OL}$. In this circuit, one of the load resistors will always conduct a non-zero current and, consequently, consume steady-state power. The amount of the standby power consumption is ultimately determined by the value of the load resistor.

Large resistance values and a smaller cell size can be achieved by using lightly-doped or undoped polysilicon for the load resistors, which has a typical sheet resistivity of 10 MΩ per square or higher. The added process complexity for implementing resistive-load SRAM cells using undoped poly resistors is usually worth the advantage of low-power operation, which is manifested by a standby current of less than 10 nA per cell. With a load resistance value of $R = 100\ \text{M}\Omega$, for example, the standby power dissipation of the resistive SRAM cell shown in Fig. 10.22 becomes

$$P_{stand-by} = V_{DD} \cdot \frac{\mu_n C_{ox}}{2} \cdot \left(\frac{W}{L}\right)_{load} \cdot \left|V_{T,load}\right|^2 \tag{10.2}$$

If we consider that a typical memory array contains a large number of memory cells, each consuming a non-zero standby power, the significance of the power consumption problem in static memory arrays becomes obvious.

Full CMOS SRAM Cell

A low-power SRAM cell may be designed simply by using cross-coupled CMOS inverters instead of the resistive-load nMOS inverters. In this case, the stand-by power consumption of the memory cell will be limited to the relatively small leakage currents of both CMOS inverters. The possible drawback of using CMOS SRAM cells, on the other hand, is that the cell area tends to increase in order to accommodate the n-well for the pMOS transistors and the polysilicon contacts.

The circuit structure of the full CMOS static RAM cell is shown in Fig. 10.24, along with the pMOS column pull-up transistors on the complementary bit lines. The basic operation principle of the CMOS SRAM cell is identical to that of the resistive-load nMOS cell examined earlier. The most important advantage of this circuit topology is that the static power dissipation is even smaller; essentially, it is limited by the leakage current of the pMOS transistors. A CMOS memory cell thus draws current from the power supply only during a switching transition. The low standby power consumption has certainly been a driving force for the increasing prominence of high- density CMOS SRAMs.

Other advantages of CMOS SRAM cells include high noise immunity due to larger noise margins, and the ability to operate at lower power supply voltages than, for example, the resistive-load SRAM cells. The major disadvantages of CMOS memories historically were larger cell size, the added complexity of the CMOS process, and the tendency to exhibit "latch-up" phenomena. With the widespread use of multi-layer polysilicon and multi-layer metal processes, however, the area disadvantage of the CMOS SRAM cell has been reduced significantly in recent years. Considering the undisputable advantages of CMOS for low-power and low-voltage operation, the added process complexity and the required latch-up prevention measures do not present a substantial barrier against the implementation of CMOS cells in high density SRAM arrays. Figure 10.25 compares typical layouts of the four-transistor resistive-load SRAM cell and the six-transistor full CMOS SRAM cell.

Note that unlike the nMOS column pull-up devices used in resistive-load SRAM, the pMOS column pull-up transistors shown in Fig. 10.24 allow the column voltages to reach full V_{DD} level. To further reduce power consumption, these transistors can also be driven by a periodic precharge signal, which activates the pull-up devices to charge-up column capacitances.

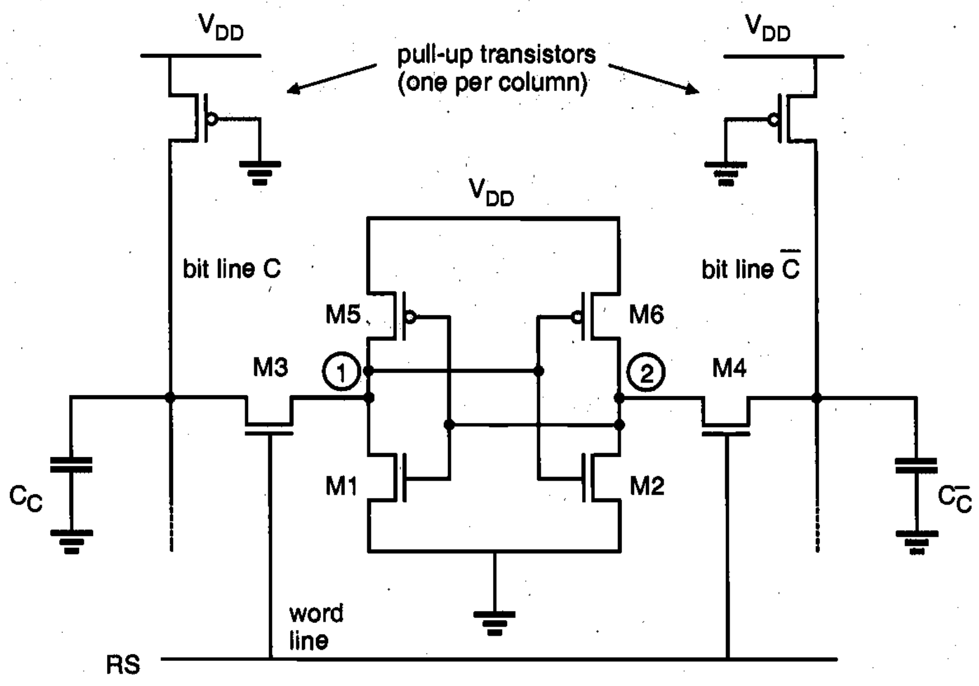
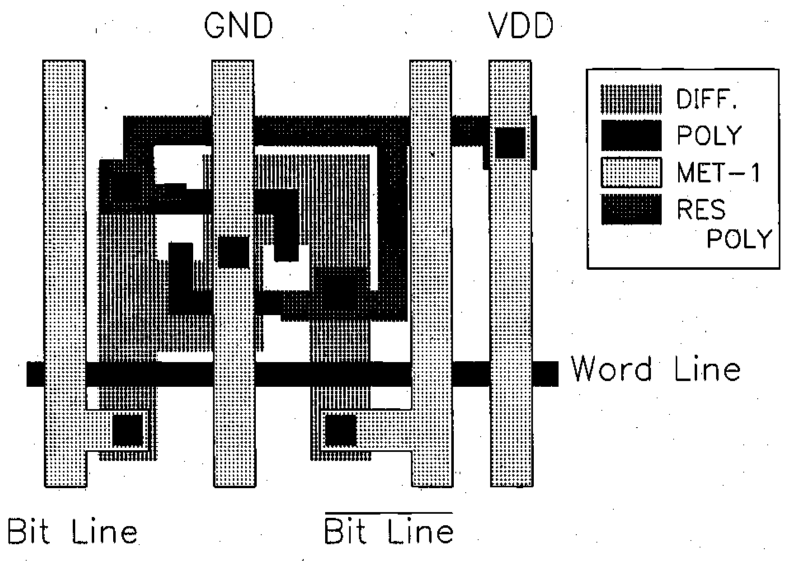


Figure 10.24. Circuit topology of the CMOS SRAM cell.

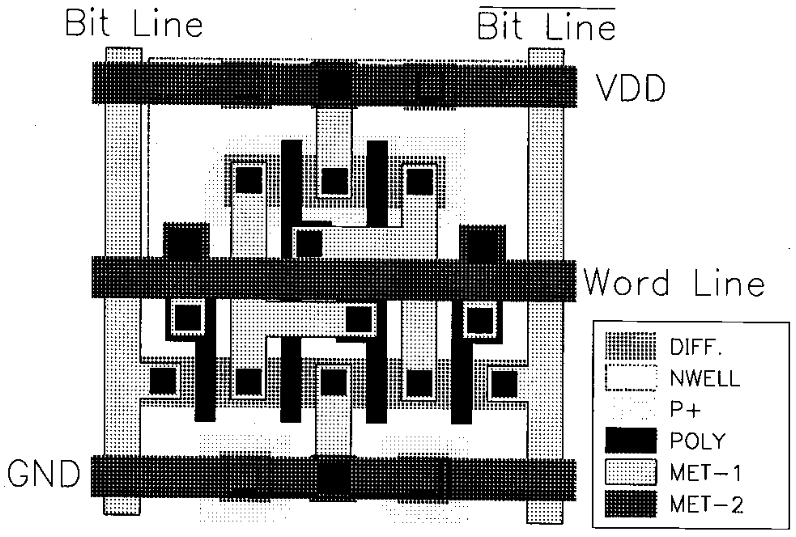
CMOS SRAM Cell Design Strategy

To determine the (W/L) ratios of the transistors in a typical CMOS SRAM cell as shown in Fig. 10.24, a number of design criteria must be taken into consideration. The two basic requirements which dictate the (W/L) ratios are: (a) the data-read operation should

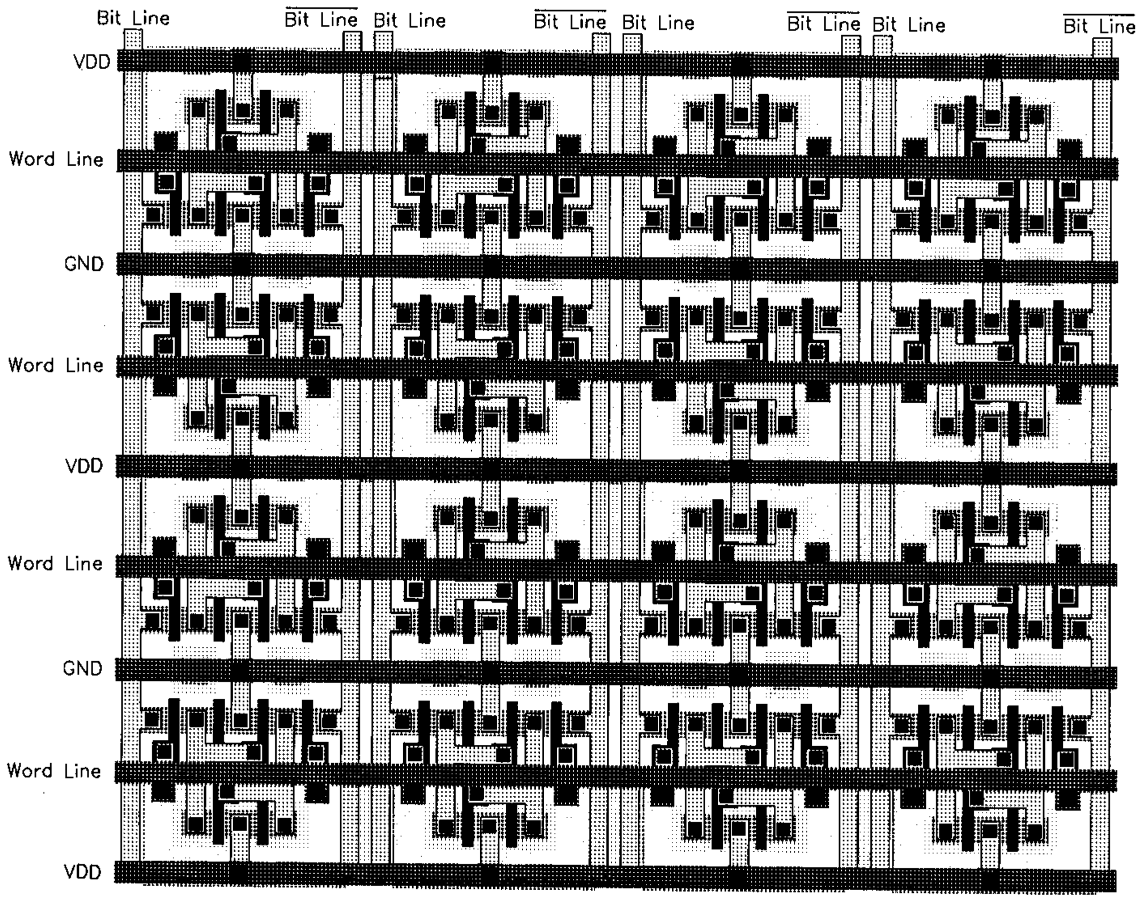


(a)

Figure 10.25. (a) Layout of the resistive-load SRAM cell.



(b)


Bit Line
$\overline{\text{Bit Line}}$
Bit Line
$\overline{\text{Bit Line}}$
Bit Line
$\overline{\text{Bit Line}}$
Bit Line
$\overline{\text{Bit Line}}$
VDD
Word Line
GND
Word Line
VDD
Word Line
GND
Word Line
VDD

(c)

Figure 10.25. (continued) (b) Layout of the CMOS SRAM cell. (c) Layout of a 4-bit x 4-bit SRAM array, consisting of 16 CMOS SRAM cells.

not destroy the stored information in the SRAM cell, and (b) the cell should allow modification of the stored information during the data-write phase. Consider the data-read operation first, assuming that a logic "0" is stored in the cell. The voltage levels in the CMOS SRAM cell at the beginning of the "read" operation are depicted in Fig. 10.26. Here, the transistors M2 and M5 are turned off, while the transistors M1and M6 operate

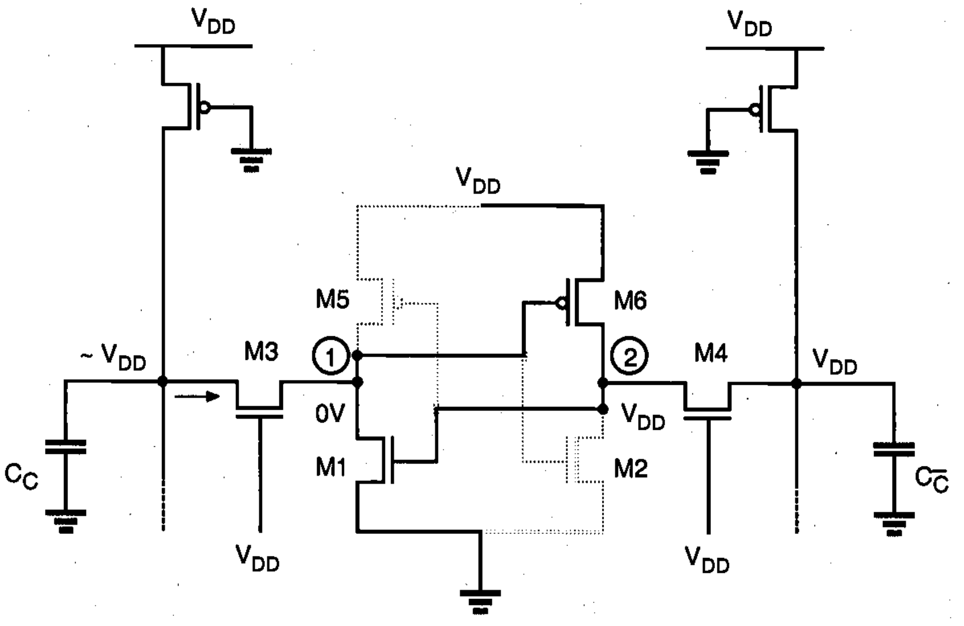


Figure 10.26. Voltage levels in the SRAM cell at the beginning of the "read" operation.

in the linear mode. Thus, the internal node voltages are $V_1 = 0$ and $V_2 = V_{DD}$ *before* the cell access (or pass) transistors M3 and M4 are turned on. The active transistors at the beginning of the data-read operation are highlighted in Fig. 10.26.

After the pass transistors M3 and M4 are turned on by the row selection circuitry, the voltage level of column $\overline{C}$ will not show any significant variation since no current will flow through M4. On the other half of the cell, however, M3 and M1 will conduct a nonzero current and the voltage level of column C will begin to drop slightly. Note that the column capacitance C_C is typically very large; therefore, the amount of decrease in the column voltage is limited to a few hundred millivolts during the read phase. The data-read circuitry to be examined later in this chapter is responsible for detecting this small voltage drop and amplifying it as a stored "0." While M1 and M3 are slowly discharging the column capacitance, the node voltage V_1 will increase from its initial value of 0 V. Especially if the (W/L) ratio of the access transistor M3 is large compared to the (W/L) ratio of M1, the node voltage V_1 may exceed the threshold voltage of M2 during this process, forcing an unintended change of the stored state. The key design issue for the data-read operation is then to guarantee that the voltage V_1 does not exceed the threshold voltage of M2, so that the transistor M2 remains turned off during the read phase, i.e.,

$$V_{1,max} \leq V_{T,2} \qquad (10.3)$$

We can assume that after the access transistors are turned on, the column voltage V_C remains approximately equal to V_{DD}. Hence, M3 operates in saturation while M1 operates in the linear region.

$$\frac{k_{n,3}}{2}\left(V_{DD}-V_1-V_{T,n}\right)^2 = \frac{k_{n,1}}{2}\left(2\left(V_{DD}-V_{T,n}\right)V_1-V_1^2\right) \tag{10.4}$$

Combining this equation with (10.3) results in:

$$\frac{k_{n,3}}{k_{n,1}} = \frac{\left(\frac{W}{L}\right)_3}{\left(\frac{W}{L}\right)_1} < \frac{2\left(V_{DD}-1.5\,V_{T,n}\right)V_{T,n}}{\left(V_{DD}-2\,V_{T,n}\right)^2} \tag{10.5}$$

The upper limit of the aspect ratio found above is actually more conservative, since a portion of the drain current of M3 will also be used to charge-up the parasitic node capacitance of node (1). To summarize, the transistor M2 will remain in cut-off during the read "0" operation if condition (10.5) is satisfied. A symmetrical condition also dictates the aspect ratios of M2 and M4.

Now consider the write "0" operation, assuming that a logic "1" is stored in the SRAM cell initially. Figure 10.27 shows the voltage levels in the CMOS SRAM cell at the beginning of the data-write operation. The transistors M1 and M6 are turned off, while the transistors M2 and M5 operate in the linear mode. Thus, the internal node voltages are $V_1 = V_{DD}$ and $V_2 = 0$ V *before* the cell access (or pass) transistors M3 and M4 are turned on.

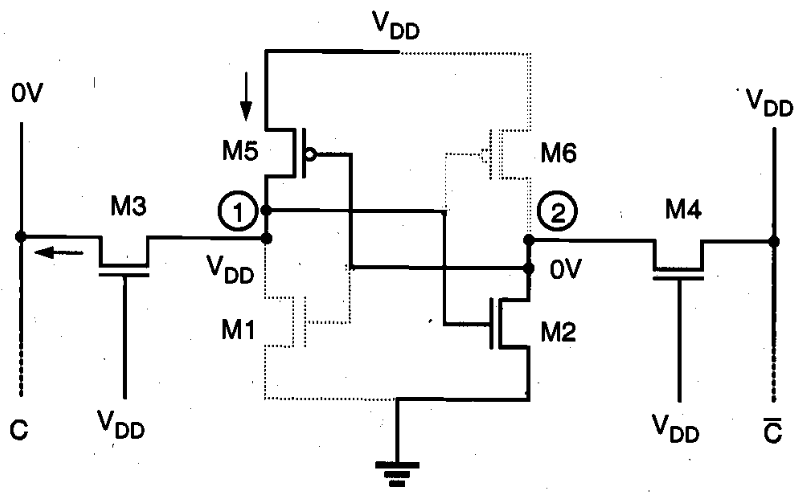


Figure 10.27. Voltage levels in the SRAM cell at the beginning of the "write" operation.

The column voltage V_C is forced to logic "0" level by the data-write circuitry; thus, we may assume that V_C is approximately equal to 0 V. Once the pass transistors M3 and M4 are turned on by the row selection circuitry, we expect that the node voltage V_2 remains *below* the threshold voltage of M1, since M2 and M4 are designed according to condition (10.5). Consequently, the voltage level at node (2) would not be sufficient to turn on M1. To change the stored information, i.e., to force V_1 to 0 V and V_2 to V_{DD}, the node voltage V_1 *must be reduced below* the threshold voltage of M2, so that M2 turns off first. When $V_1 = V_{T,n}$, the transistor M3 operates in the linear region while M5 operates in saturation.

$$\frac{k_{p,5}}{2}\left(0-V_{DD}-V_{T,p}\right)^2 = \frac{k_{n,3}}{2}\left(2\left(V_{DD}-V_{T,n}\right)V_{T,n}-V_{T,n}^{\,2}\right) \qquad (10.6)$$

Rearranging this condition results in:

$$\frac{k_{p,5}}{k_{n,3}} < \frac{2\left(V_{DD}-1.5\,V_{T,n}\right)V_{T,n}}{\left(V_{DD}+V_{T,p}\right)^2}$$

$$\frac{\left(\frac{W}{L}\right)_5}{\left(\frac{W}{L}\right)_3} < \frac{\mu_n}{\mu_p}\cdot\frac{2\left(V_{DD}-1.5\,V_{T,n}\right)V_{T,n}}{\left(V_{DD}+V_{T,p}\right)^2} \qquad (10.7)$$

To summarize, the transistor M2 will be *forced* into cut-off mode during the write "0" operation if condition (10.7) is satisfied. This will guarantee that M1 subsequently turns on, changing the stored information. Note that a symmetrical condition also dictates the aspect ratios of M6 and M4.

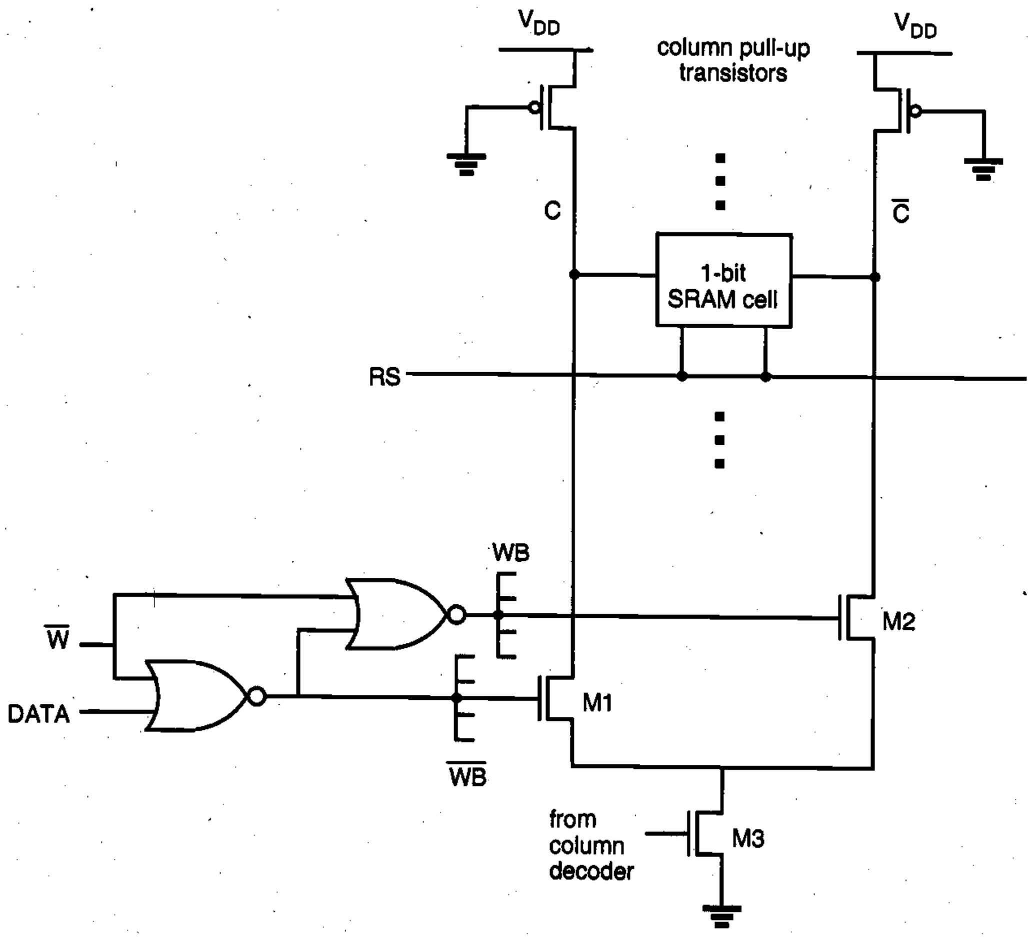
SRAM Write Circuitry

As already discussed in the preceding section, a "write" operation is performed by forcing the voltage level of either column (bit line) to a logic-low level. To accomplish this task, a low-resistance, conducting path must be provided from each column to the ground, which can be selectively activated by the data-write signals. A simplified view of the SRAM "write" circuitry designed for this operation is shown in Fig. 10.28. Here, the nMOS transistors M1 and M2 are used to pull down the two column voltages, while the transistor M3 completes the conducting path to ground. Note that M3 is driven by the column address decoder circuitry, i.e., M3 turns on only when the corresponding column address is selected. The column pull-down transistors, on the other hand, are driven by two pseudo-complementary control signals, *WB* and $\overline{WB}$. The "write-enable" signal *W* (active low) and the data to be written (*DATA*) are used to generate the control signals, as shown in the table in Fig. 10.28.

The nMOS pull-down transistors M1 and M2, as well as the column selection transistor M3 must have sufficiently large (W/L) ratios so that the column voltages can be forced to almost 0 V level during a "write" operation. Also note that the data input circuitry consisting of two NOR2 gates can be shared by several columns, assuming that one column is activated, i.e., selected by the column address decoder, at any given time.

SRAM Read Circuitry

During the "data read" operation in the SRAM array, the voltage level on either one of the columns drops slightly after the pass transistors are turned on by the row address decoder circuit. In order to reduce the read access time, the "read" circuitry must detect



$\overline{W}$	*DATA*	*WB*	$\overline{WB}$	*Operation*
0	1	1	0	M1 is off, M2 is on → $V_{\overline{C}}$ low
0	0	0	1	M1 is on, M2 is off → V_C low
1	X	0	0	M1 and M2 are off → both columns remain high

Figure 10.28. Data write circuitry associated with one column-pair in an SRAM array.

a very small voltage difference between the two complementary columns, and amplify this difference to produce a valid logic output level. A simple source-coupled differential amplifier can be used for this task, as shown in Fig. 10.31. Here, the drain currents of the two complementary nMOS transistors are:

$$I_{D,1} = \frac{k_n}{2}\left(V_C - V_x - V_{T,1}\right)^2 \tag{10.8}$$

$$I_{D,2} = \frac{k_n}{2}\left(V_{\overline{C}} - V_x - V_{T,2}\right)^2 \tag{10.9}$$

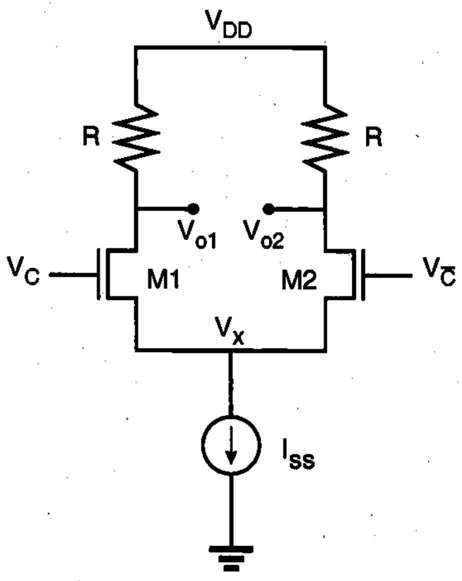


Figure 10.29. Simple source-coupled differential amplifier circuit for "read" operation.

Small-signal analysis of the circuit yields the differential gain of this circuit as

$$\frac{\partial\left(V_{o1} - V_{o2}\right)}{\partial\left(V_C - V_{\overline{C}}\right)} = -R \cdot g_m \tag{10.10}$$

where

$$g_m = \frac{\partial I_D}{\partial V_{GS}} = \sqrt{2 k_n I_D}$$

The differential gain of the amplifier can be increased significantly by using active loads instead of resistors and by using *cascode* configuration, i.e., an intermediary common-gate stage between the common-source transistors and the load transistors. Finally, the differential output of the cascode stage must be converted into single-ended output, by using a level-shifter and buffer stage. Although the "data read" circuitry described above is capable of detecting small voltage differences between the two complementary bit lines (columns), other types of efficient sense amplifiers are also implemented with CMOS technology, as will be examined in the following.

The architecture of the output "read" circuitry is driven primarily by the constant demand for high access speed and high integration density. We must recognize first that the full CMOS SRAM cell has a natural speed advantage which is derived from using both active pull-up and active pull-down devices in the latch circuits. The CMOS rise and fall times are short and symmetrical, whereas the nMOS latch circuit has a short output fall time but a larger rise time because of its high-resistance load. Both the depletion-load nMOS SRAM cell and the resistive-load nMOS SRAM cell, hence, have slower average switching speed compared to that for the full CMOS cell.

Apart from the type of the SRAM cell, the *precharging* of bit lines also plays a significant role in the access time. In an unclocked SRAM array, data from the accessed cell develops a voltage difference on the bit lines. This voltage difference is then detected and amplified to drive the output buffer. When another cell on the same column is accessed next, one that contains data opposite to the data contained in the previously accessed cell, the output has to switch first to an equalized state and then to the opposite logic state. Since the capacitance on the bit lines is quite large, the time required for switching the differential from one state to the other becomes a significant portion of the overall access time. The access time penalty associated with this procedure can be substantially reduced by the *equalization* of bit lines prior to each new access. Equalization can be done when the memory array is deselected, i.e., between two access cycles.

Fast Sense Amplifiers

The availability of the CMOS technology for the manufacturing of high-density SRAM arrays, either with full-CMOS or resistive-load nMOS memory cells, also allows us to implement efficient sense amplifier structures with pMOS current mirrors. Figure 10.30 shows a single-stage differential current-mirror amplifier, which is typically used as a front-end (input stage) in operational amplifiers.

In this circuit, the gates of the two nMOS transistors M1 and M2 are connected to the bit lines. Their substrate terminals are tied to their respective source terminals in order to remove the substrate-bias effect. Notice that each bit line is represented by a large parasitic capacitance. The nMOS transistor M3 is a long-channel device which acts as a current source for both branches, and is controlled by a clock signal. The output inverter is not a part of the differential amplifier, but it is used to drive the output node.

Before the beginning of a "read" operation, the two bit lines (columns) are pulled up for equalization, as discussed earlier. The CLK signal is low during this phase, so that the nMOS transistor M3 remains off. Since both M1 and M2 conduct, the common source node is pulled up, and the output node of the amplifier also goes high. Therefore, the output of the inverter is at a logic-low level initially.

Once a memory cell is selected for the "read" operation, the voltage on one of the complementary bit lines will start to drop slightly. At the same time as the row selection signal, the CLK signal driving M3 is also turned on. If the stored data on the selected SRAM cell forces the bit line *C* to decrease slightly, transistor M1 turns off, and the output voltage of the differential amplifier drops immediately. Consequently, the output voltage of the inverter goes high. Otherwise, if the stored data on the selected memory cell forces the bit line $\overline{C}$ to drop slightly, M2 turns off. Thus, the voltage level at the output node of the differential amplifier remains high in this case, and the inverter also preserves its logic-low output level.

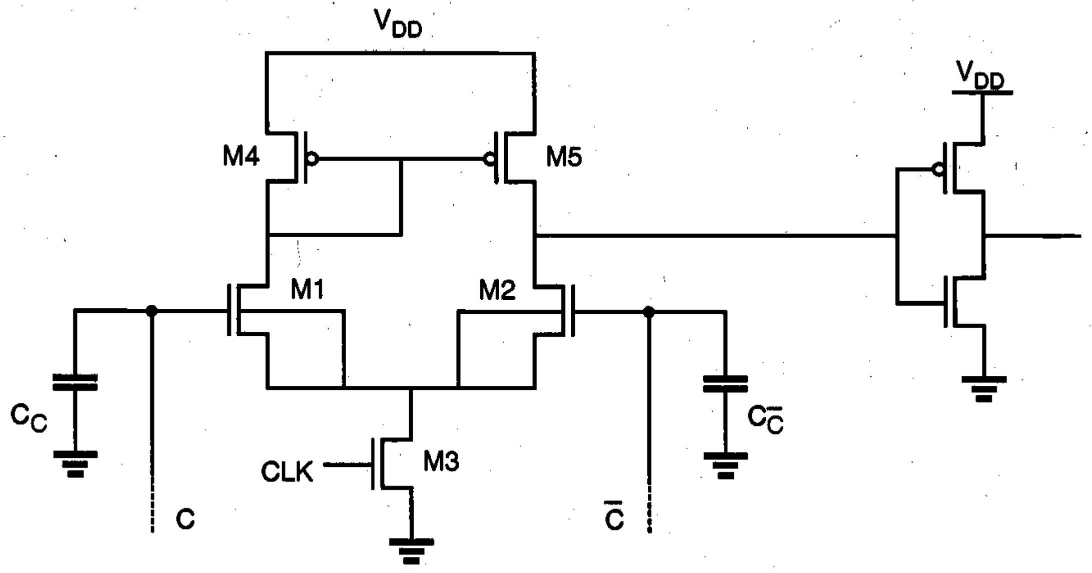


Figure 10.30. Differential current-mirror amplifier to speed up the memory read access time.

In most high-density SRAM chips, two- or three-stage current-mirror differential sense amplifiers are implemented to further improve the "read" access speed. An example of a two-stage CMOS sense amplifier is shown in Fig. 10.31. Here, the first stage consists of two *complementary* differential sense amplifiers, both of which receive the column (bit line) voltages at their inputs. Since they are operated in an anti-symmetrical configuration, the output voltages of the two sense amplifiers must change into opposite directions. The complementary outputs of the first-stage amplifiers are applied to the input terminals of a second-stage differential amplifier, which generates the output signal.

The dynamic behavior of the one-stage sense amplifier is compared with the dynamic behavior of the two-stage sense amplifier via SPICE simulation, in Fig. 10.32. Here, one of the column voltages (V_C) starts to drop slightly, while the other column voltage remains constant (not shown in the figure). It can be seen that the output of the one-stage sense amplifier responds to this change with a delay of about 10 ns, whereas the output delay of the two-stage sense amplifier is only about 1 ns.

As mentioned earlier, in the sense amplifier circuit examined here (Fig. 10.30), the substrate terminals of M1 and M2 are connected to the common source node instead of the ground, in order to avoid threshold voltage variations due to the substrate-bias effect. This is possible using twin-tub, silicon-on-sapphire (SOS), or p-well fabrication processes. In an n-well CMOS process, on the other hand, where all nMOS transistors are formed over the common substrate, the circuit cannot be implemented.

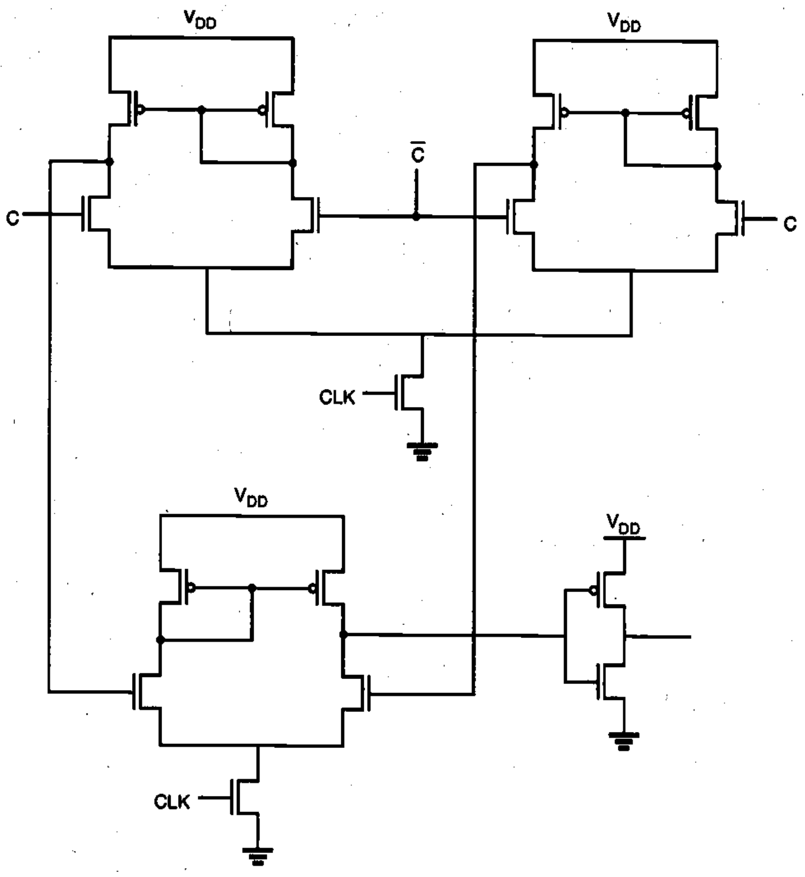


Figure 10.31. Two-stage differential current-mirror sense amplifier circuit.

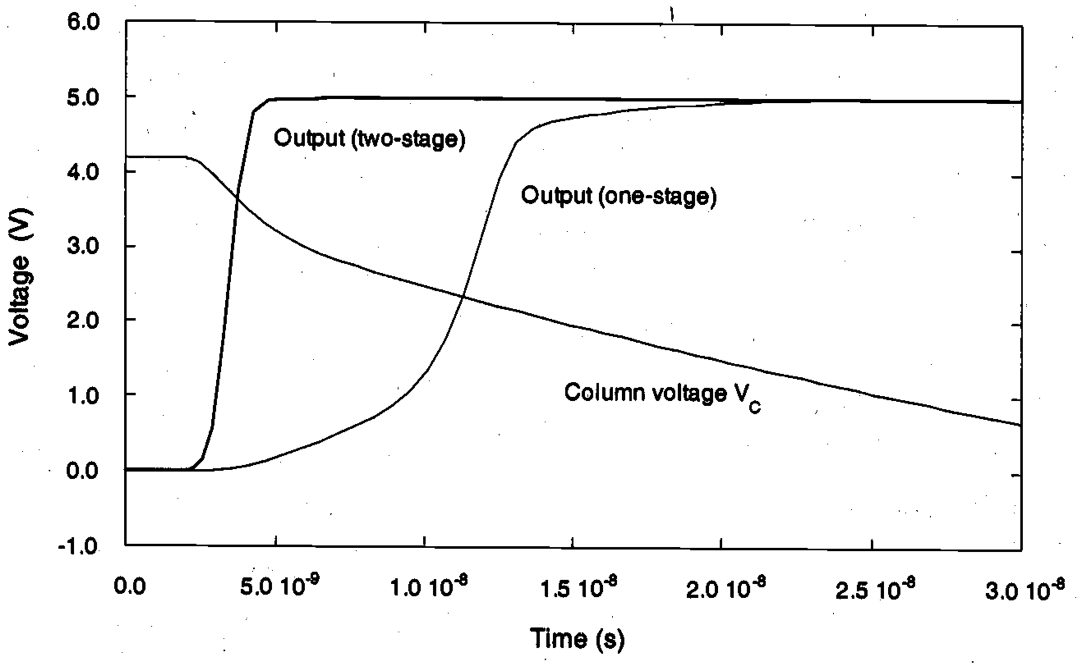


Figure 10.32. Typical dynamic response characteristics of the one-stage sense amplifier and the two-stage sense amplifier circuits.

Another simple circuit topology for sense amplifiers is the cross-coupled latch, shown in Fig. 10.33. Assume that both columns (bit lines) are being pulled up during the precharge cycle and that the voltage on the bit line C starts to drop slightly when the SRAM cell access transistors are activated. Consequently, when the transistor M3 is turned on, the voltage at node $\overline{C}$ is higher than the voltage at node C. Therefore, M1 turns on first and further pulls down the potential at node C. This makes it more difficult for M2 to conduct. Eventually, M2 turns off completely, and M1 keeps conducting, so that the bit line C is discharged through M1 and M3.

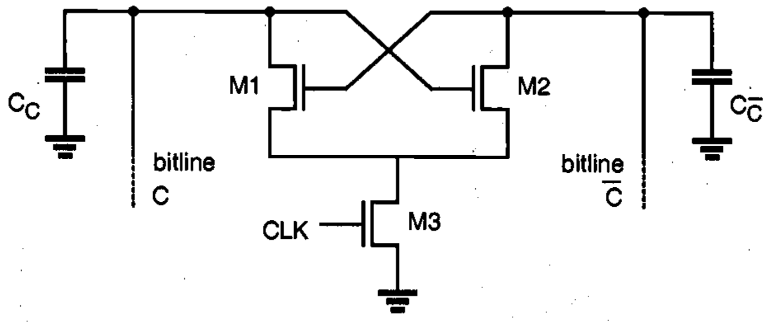


Figure 10.33. Cross-coupled nMOS sense amplifier circuit.

Note that the operation of this circuit is distinctly different from that of the differential amplifier circuit. The cross-coupled sense amplifier does not generate an output voltage level which corresponds to the polarity of the voltage difference between the two bit lines, but it rather *amplifies* the small voltage difference already existing between the bit lines. This voltage difference must still be translated into a logic level, by using a buffer stage. We will see the implementation of a very similar structure also in the dynamic RAM (DRAM) sense amplifiers.

In most SRAM arrays, the cross-coupled sense amplifier circuit is used in conjunction with the differential sense amplifier (Fig. 10.30) examined earlier. In this case, the cross-coupled amplifier serves as a front-end structure to amplify the small voltage difference between the two bit lines, whereas the current-mirror amplifier detects the voltage difference and generates the output level. Figure 10.34 shows the complete circuit diagram of a CMOS SRAM column with one representative CMOS memory cell, the "data write" circuitry, the cross-coupled sense amplifier, and the differential (main) sense amplifier circuit. Note that the main amplifier is connected to the two complementary bit lines via pass transistors, which are driven by the "read select" signal. This configuration enables the use of one main sense amplifier to read the data out of several columns, one at a time.

Dual-Port Static RAM Arrays

In some cases, the memory array may have to be accessed simultaneously by multiple processors, or by one processor and another peripheral device. This could result in a timing conflict called "contention," which can be resolved only by having one of the processors wait until the SRAM is free. The added wait state, however, significantly reduces the advantages of the high-speed processor. The dual-port RAM architecture is implemented in systems in which a main memory array must serve multiple high-speed processors and peripheral devices with minimum delay.

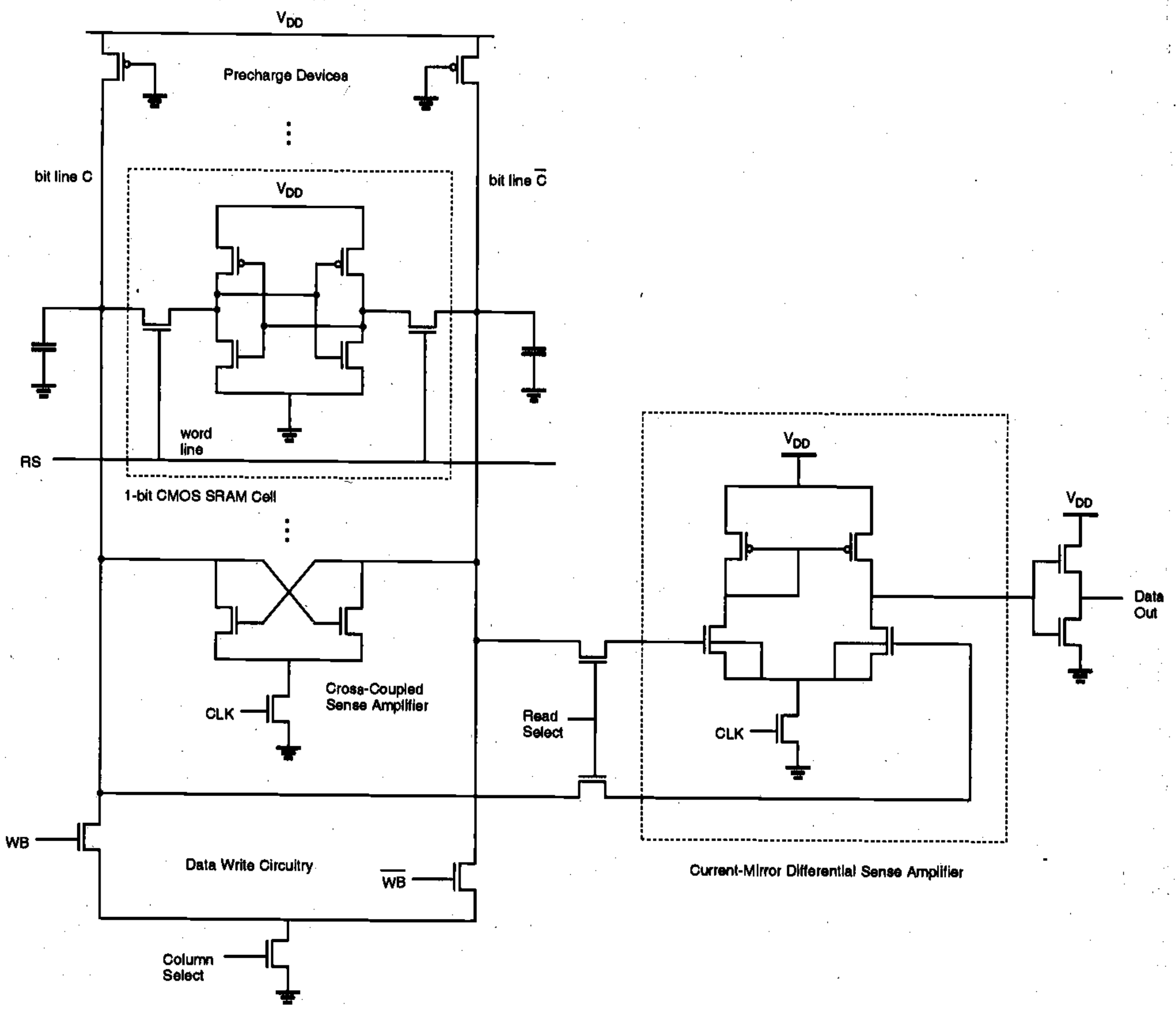


Figure 10.34. Complete circuit diagram of a CMOS static RAM column with data write and data read circuitry.

The ideal dual-port SRAM allows simultaneous access to the same location in the memory array, by using two independent sets of bit lines and associated access switches for each memory cell. The circuit structure of a typical CMOS dual-port SRAM cell is shown in Fig. 10.35. Here, "word line 1" is used to access one set of complementary bit lines (bit line 1), while "word line 2" allows access to the other set of bit lines (bit line 2). The capability of simultaneous access eliminates wait states for the processors during "data read" operations. However, contention may still occur if both external processors accessing the same memory location simultaneously attempt to write data onto the accessed cell, or if one of the processors attempts to read data while the other processor

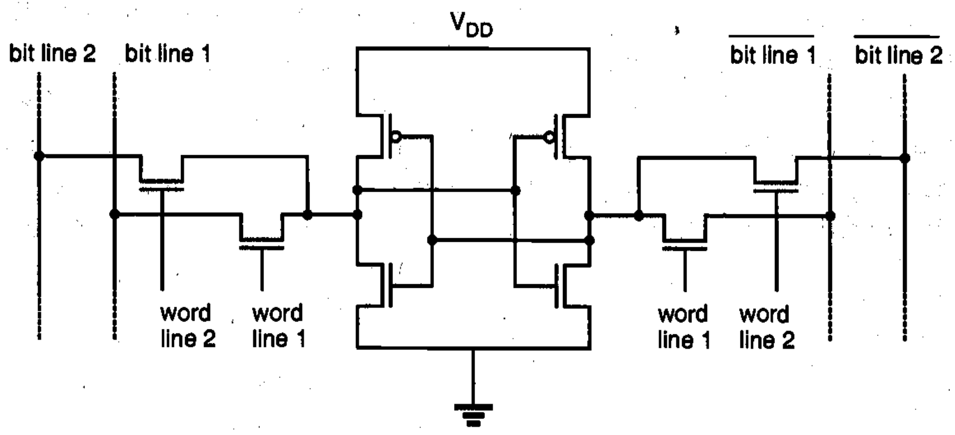


Figure 10.35. Circuit diagram of the CMOS dual-port SRAM cell.

writes data onto the same cell. In most cases, overlapping operations to the same memory location can be eliminated by a contention arbitration logic. It can either allow contention to be ignored and both operations to proceed, or it can arbitrate and delay one port until the operation on the other port is completed.

10.4. Dynamic Read-Write Memory (DRAM) Circuits

All of the static RAM cells examined in the previous section consist of a two-inverter latch circuit, which is accessed for "read" and "write" operations via two pass transistors. Consequently, the SRAM cells require four to six transistors per bit, and four to five lines connecting to each cell, including the power and ground connections. To satisfy these requirements, a substantial silicon area must be reserved for each memory cell. In addition, most SRAM cells have non-negligible standby (static) power dissipation, with the exception of the full CMOS SRAM cell.

As the trend for high-density RAM arrays forces the memory cell size to shrink, alternative data storage concepts must be considered to accommodate these demands. In a *dynamic* RAM cell, binary data is stored simply as charge in a capacitor, where the presence or absence of stored charge determines the value of the stored bit. Note that the data stored as charge in a capacitor cannot be retained indefinitely, because the leakage currents eventually remove or modify the stored charge. Thus, all dynamic memory cells require a periodic refreshing of the stored data, so that unwanted modifications due to leakage are prevented before they occur.

The use of a capacitor as the primary storage device generally enables the DRAM cell to be realized on a much smaller silicon area compared to the typical SRAM cell. Notice that even as the binary data is stored as charge in a capacitor, the DRAM cell must have access devices, or switches, which can be activated externally for "read" and "write" operations. But this requirement does not significantly affect the area advantage over the SRAM cell, since the cell access circuitry is usually very simple. Also, no static power is dissipated for storing charge on the capacitance. Consequently, dynamic RAM arrays can achieve higher integration densities than SRAM arrays. Note that a DRAM array also