Supplement to

Logic and Computer Design Fundamentals 3rd Edition[1]

CMOS CIRCUITS

Selected topics not covered in the third edition of *Logic and Computer Design Fundamentals* are provided here for optional coverage and for self-study if desired. This material fits well with the desired coverage in some programs but not may not fit within others due to time constraints or local preferences. This supplement, referenced in Chapter 2 as a part of the coverage Other Gate Types, presents a functional view of the implementation of gates as CMOS electronic circuits. It is particularly appropriate for coverage by electrical or computer engineering students.

So far we have dealt largely with implementing logic circuits in terms of gates. In this supplement, we briefly explore implementing the gates themselves using CMOS technology. In addition, we study how structures other than primitive logic gates can be implemented directly in terms of electronic elements called transistors. CMOS implementation is important because we often design CMOS logic from Boolean equations directly to the transistor level, skipping the logic gate level.

Switch Models for CMOS Transistors

CMOS technology employs two types of transistor: *n-channel* and *p-channel*. The two differ in the characteristics of the semiconductor materials used in their implementation and in the mechanism governing the conduction of a current through them. Most important to us, however, is the difference in behavior of the two types of transistor. We will model this behavior using switches controlled by voltages corresponding to logic 0 and logic 1. Such a model ignores complex electronic devices and captures only logical behavior.

The symbol for an n-channel transistor is shown in Figure 1(a). The transistor has three terminals: the gate (G), the source (S), and the drain (D), as shown in Figure 1(b). The voltage applied between G and S determines whether a path for current to flow exists between D and S. If a path exists, we say that the transistor is

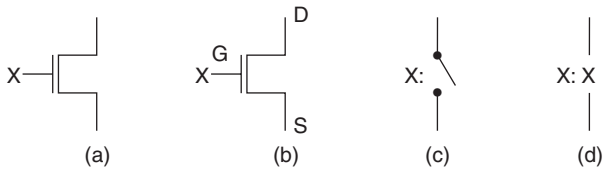


FIGURE 1
Symbol and Switch Model for n-Channel Transistor

ON, and if a path does not exist, we say that the transistor is OFF. The n-channel transistor is ON if the applied gate-to-source voltage is H and OFF if the applied voltage is L. Here we will make the usual assumption that a 1 represents the H voltage range and a 0 represents the L voltage range.

The notion of whether a path for current to flow exists is easily modeled by a switch, as shown in Figure 1(c). The switch consists of two fixed terminals corresponding to the S and D terminals of the transistor. In addition, there is a movable contact that, depending on its position, determines whether the switch is open or closed. The position of the contact is controlled by the voltage applied to the gate terminal G. Since we are looking at logic behavior, this control voltage is represented on the symbol by the input variable X on the gate terminal. For an n-channel transistor, the contact is open (no path exists) for the input variable X equal to 0 and closed (a path exists) for the input variable X equal to 1. Such a contact is traditionally referred to as being *normally open*, that is, open without a positive voltage applied to activate or close it. Figure 1(d) shows a shorthand notation for the n-channel switch model with the variable X applied. This notation represents the fact that a path between S and D exists for X equal to 1 and does not exist for X equal to 0.

The symbol for a p-channel transistor is shown in Figure 2(a). In Figure 2(b), the positions of the source S and drain D are seen to be interchanged relative to their positions in the n-channel transistor. The voltage applied between the gate G and the source S determines whether a path exists between the drain and source. Note in Figure 2(a) that the negation indicator or bubble appears as a part of the symbol. This is because, in contrast to the behavior of an n-channel transistor, a path exists between S and D in the p-channel transistor for input variable X equal to 0 (at value L) and does not exist for input variable X equal to 1 (at value H). This behavior is represented by the model in Figure 2(c), which has a *normally*

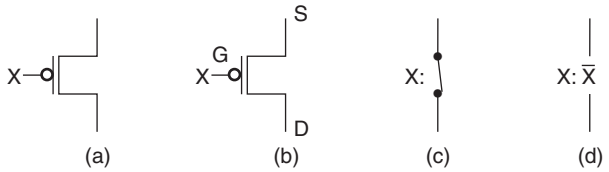


FIGURE 2
Symbol and Switch Model for p-Channel Transistor

closed contact through which a path exists for X equal to 0. No path exists through the contact for X equal to 1. In addition, the shorthand notation of the p-channel switch model with variable X applied is given in Figure 2(d). Since a 0 on input X causes a path to exist through the switch and a 1 on X produces no path, the literal shown on the switch is $\overline{X}$ instead of X.

Networks of Switches

A network made up of switches that model transistors can be used to design CMOS logic. The network implements a function F if there is a path through the network for F equal to 1 and no path through the network for F equal to 0. A simple network of p-channel transistor switch models is shown in Figure 3(a). The function G_1 implemented by this network can be determined by finding the input combinations for which a path exists through the network. In order for the path to exist through G_1, both switches must be closed; that is, the path exists for $\overline{X}$ and $\overline{Y}$ both 1. This implies that $X = 0$ *and* $Y = 0$. Thus, the function G_1 of the network is $\overline{X} \cdot \overline{Y} = \overline{X + Y}$, in other words, the NOR function. In Figure 3(b), for function G_2, a path exists through the n-channel switch model network if either switch is closed, that is, for $X = 1$ *or* $Y = 1$. Thus, the function G_2 is $X + Y$.

In general, switches in series give an AND function and switches in parallel give an OR function. (The function for the preceding network that models p-channel transistors is a NOR function because of the complementation of the variables and the application of DeMorgan's law.) By using these network functions to produce paths in a circuit that attach logic 1 (H) or logic 0 (L) to an output, we can implement a logic function on the output, as discussed next.

Fully Complementary CMOS Circuits

The subfamily of CMOS circuits that we will now consider has the general structure shown in Figure 4(a). Except during transitions, there is a path to the output of the circuit F either from the power supply +V (logic 1) or from ground (logic 0). Such a circuit is called *static* CMOS. In order to have a static circuit, the transistors must implement networks of switches for both function F and function $\overline{F}$. In other words, both the 0's and the 1's of the function F must be implemented with paths

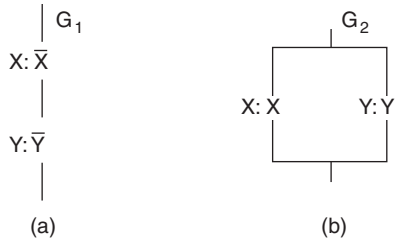


FIGURE 3
Example of Switch Model Networks

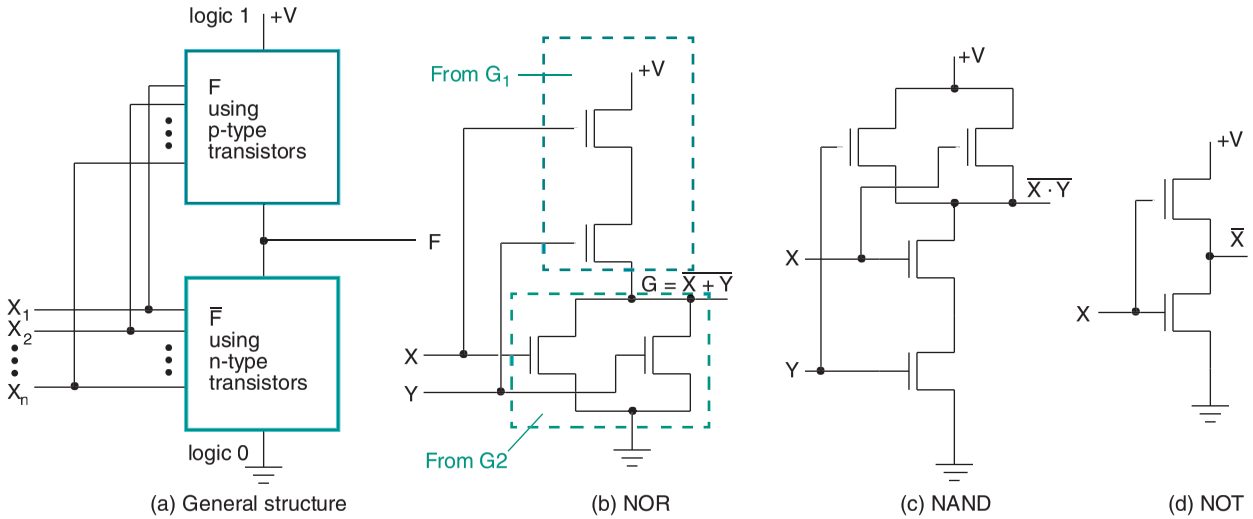


FIGURE 4
Fully Complementary CMOS Gate Structure and Examples

through networks. The switch network implementing F is constructed using p-channel transistors and connects the circuit output to logic 1. P-channel transistors are used because they conduct logic-1 values better than logic-0 values. The switch network implementing $\overline{F}$ is constructed using n-channel transistors and connects the circuit output to logic 0. Here n-channel transistors are used because they conduct logic-0 values better than logic-1 values. Note that the same input variables enter both the p-channel and n-channel switch networks.

To illustrate a fully complementary circuit, we use transistors corresponding to the networks G_1 and G_2 from Figure 3(a) and (b) as the p-channel implementation of G and the n-channel implementation of $\overline{G}$, respectively, in Figure 4(b). A path exists through G_1 for $\overline{X+Y} = 1$, which means that a path exists in Figure 4(b) from logic 1 to the circuit output, making $G = 1$ for $\overline{X+Y} = 1$. This provides the 1's on the output for the function G. A path exists through G_2 for $X + Y = 1$, which means that a path exists in Figure 4(b) from logic 0 to the output for $X + Y = \overline{\overline{X+Y}} = 1$. This path makes $G = 0$ for the complement of $\overline{X+Y}$. Thus, the n-channel circuit implements $\overline{G}$. This provides the 0's on the output for function G. Since both the 1's and 0's are provided for G, we can say that the circuit output $G = \overline{X+Y}$, which is a NOR gate. This is the standard static CMOS implementation for a NOR.

Since the NAND is just the dual of the NOR, we can implement the CMOS NAND by simply replacing the $+$ by $\cdot$ in the equations for G_1 and G_2. In terms of the switch network, the dual of switches in series is switches in parallel and vice versa. This duality applies to the transistors that are modeled as well, giving the NAND implementation in Figure 4(c). The final gate in Figure 4(d) is the implementation of the NOT.

Note that all of the circuits in Figure 4 implement inverting functions under DeMorgan's laws. This inversion property is characteristic of CMOS gates. In fact, as we look at a general design procedure, we will assume that all functions are implemented as $F = \overline{\overline{F}}$. This avoids working directly with p-channel switches, which involve complementing variables. Thus, we will design the n-channel network for $\overline{F}$ and take the dual to get the p-channel network for F. For functions more complex than NAND, NOR, and NOT, the resulting circuits are called *complex gates* and are designed in accordance with the following procedure for function F:

1. Find and simplify the complement of F, $\overline{F}$.
2. Implement $\overline{F}$ as a switch network using n-channel switch models.
3. Connect the n-channel switch network between ground and output F, and convert the switches to n-channel transistors.
4. Take the dual of the n-channel switch network for $\overline{F}$, and replace the n-channel switch models with p-channel switch models, keeping switch inputs unchanged.
5. Connect the p-channel switch network between +V and the output F, and convert the switches to p-channel transistors.

Step 4 is different than expected: we take the dual of the n-channel switch network to get the p-channel switch network. Recall that the difference between the dual and the complement is that, in taking the complement, all literals in the expression are complemented. We need not do this complementing, however, since the complement of the variables is automatically taken by replacing the n-channel switch models with p-channel switch models. Taking the dual of a function means replacing AND with OR and OR with AND. For a switch network, this corresponds to taking switches or subnetworks that are in parallel and placing them in series and taking switches or subnetworks that are in series and placing them in parallel.

We illustrate this design procedure with an example.

• EXAMPLE 1

Design of a Complex Gate for $F = A\overline{B} + AC + B\overline{C}$

In step 1 of the foregoing procedure, we place F on a map and, from the map, use the 0's to obtain a sum-of-products expression for $\overline{F}$ and the 1's to obtain a product-of-sums expression for $\overline{F}$:

$$\overline{F} = \overline{A}\,\overline{B} + \overline{A}C \qquad \overline{F} = \overline{A}(\overline{B} + C)$$

Since the product-of-sums expression has fewer literals (three instead of four), we select it for use in the next step.

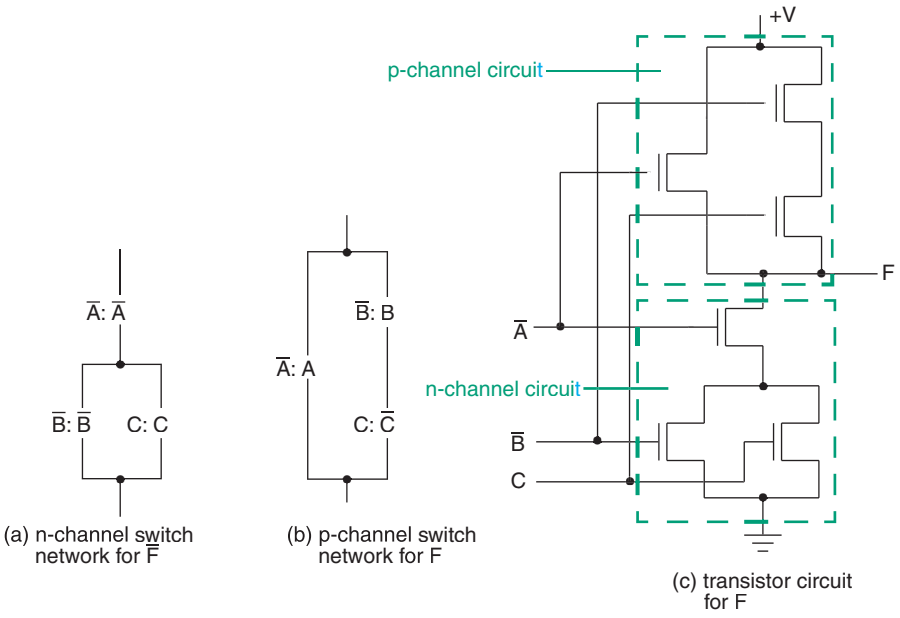


FIGURE 5
Networks and Circuit for Example 1, $F = A + B\overline{C}$

In step 2, we find an n-channel switch model network for $\overline{F}$. From the product-of-sums expression, $\overline{A}$ is ANDed with the term $\overline{B}+C$. Thus, we place a switch with input $\overline{A}$ in series with a network implementing $\overline{B}+C$, as shown in Figure 5(a). We implement $\overline{B}+C$ with a switch having input $\overline{B}$ in parallel with a switch having input C. Checking step 2, the final network in Figure 5(a) has a path through it for $A = 0$ and either $B = 0$ or $C = 1$ or both.

The network from Figure 5(a) is converted into the corresponding n-channel transistor circuit between ground and the output F, as given in the lower part of Figure 5(c), completing step 3.

Next, in step 4, we are to take the dual of the n-channel switch network from step 2. First, we take the switches B and C that are in parallel in Figure 5(a) and place them in series. Then we take A, which is in series with the parallel combination of B and C and place it in parallel with the series combination of B and C. Finally, we replace the n-channel switch models with p-channel switch models, keeping the complementation on the input values to the switches unchanged. The circuit in Figure 5(b) results.

The network in Figure 5(b) is converted to the p-channel transistor circuit between +V and F, as given in the upper part of Figure 5(c), completing the circuit in step 5. •

We can use any Boolean expression for $\overline{F}$, although by minimizing the number of literals as much as possible, as is done in this example, we minimize the number of transistors in the circuit. In the actual design of these transistor circuits, it is also necessary to take electronic considerations into account. For example, in most cases, no path through one of the networks can contain more than four or five transistors in series. This clearly limits the functions that can be implemented in a single complex gate.

Transmission Gates

Besides primitive and complex gates, there is one additional transistor circuit frequently used in CMOS logic. This circuit is the transmission gate (TG). It has its own symbol and is often included in gate-level logic circuit diagrams. A transmission gate is used as an electronic switch for making a connection between two points in a circuit. It consists of an n-channel transistor and a p-channel transistor in parallel, as shown in Figure 6(a). The two types of transistor are used because the p-channel transistor passes 1 (H) well and the n-channel transistor passes 0 (L) well. Figure 6(b) is the switch model for the transmission gate. Here X is the input, Y is the output, and the two terminals C and $\overline{C}$ are control inputs. If $C = 1$ (H) and $\overline{C} = 0$ (L), there is a path between X and Y for the signal to pass through. If $C = 0$ and $\overline{C} = 1$, there is no path, and the circuit behaves like an open switch. The IEEE symbol for the transmission gate is given in Figure 6(c). Normally, the control inputs are connected through an inverter, as shown in Figure 6(d), so that C and $\overline{C}$ are the complements of each other.

Transmission gates are particularly useful for performing selection functions. A TG-based circuit that selects one of two values A and B to apply to an output F

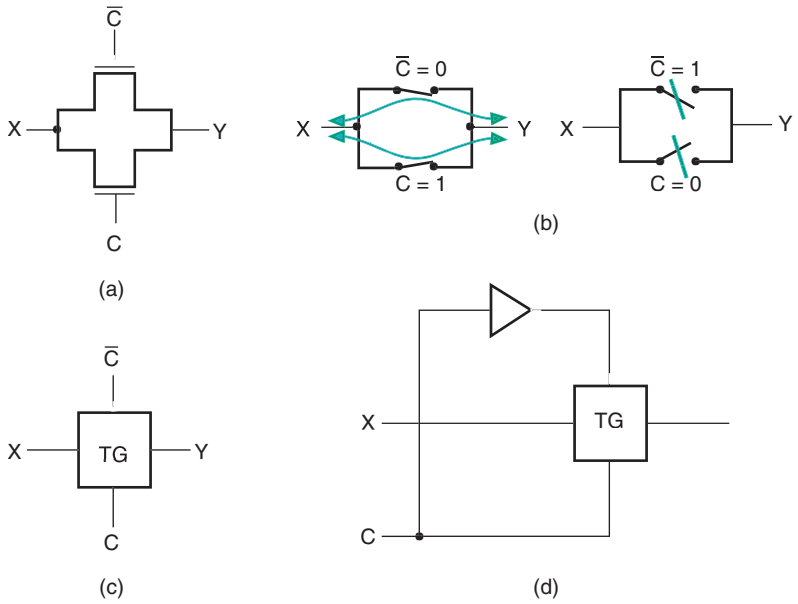


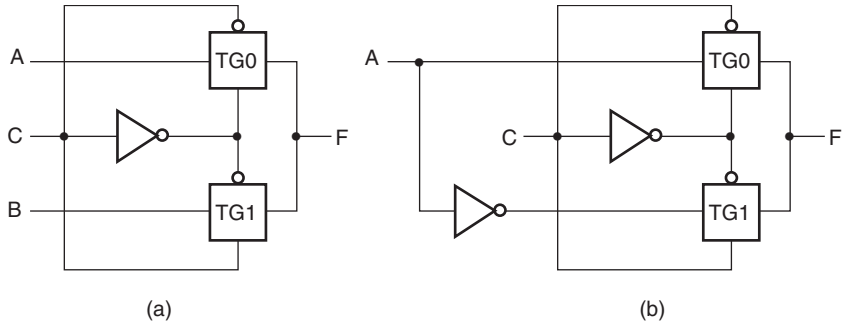
FIGURE 6
Transmission Gate (TG)

is shown in Figure 7(a). If $C = 0$, then a path exists through TG0 connecting F to A, and no path exists through TG1. If $C = 1$, then a path exists through TG1 connecting F to B, and no path exists through TG0. In Chapter 3, we will find that selection circuits, such as this one, are called multiplexers. So we call this circuit a transmission gate–based multiplexer.

By making $B = \overline{A}$ for the selector, an exclusive-OR gate can be constructed with two transmission gates and two inverters, as shown in Figure 7(b). Input C controls the paths in the transmission gates, and input A provides the output for F. If input C is equal to 1, a path exists through transmission gate TG1 connecting F to $\overline{A}$, and no path exists through TG0. If input C is equal to 0, a path exists through TG0 connecting F to A, and no path exists through TG1. Thus, the output F is connected to A. This results in the exclusive-OR truth table, as indicated in Figure 7(c).

REFERENCES

1. WESTE, N. H. E., AND ESHRAGHIAN, K. *Principles Of CMOS VLSI Design: A Systems Perspective*, 2nd ed. Reading, MA: Addison-Wesley, 1993.
2. RABAEY, J. M., CHANDRAKASAN, A., AND NIKOLIC, B. *Digital Integrated Circuits*: *A Design Perspective.* 2nd ed. Upper Saddle River, NJ: Pearson Education, Inc., 2003.
3. WOLF, W. *Modern VLSI Design*: *Systems on Silicon,* 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1998.



(a) (b)

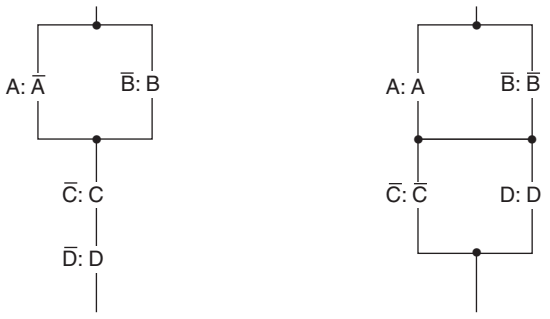
A	C	TG1	TG0	F
0	0	No path	Path	0
0	1	Path	No path	1
1	0	No path	Path	1
1	1	Path	No path	0

(c)

FIGURE 7
Selector and Exclusive-OR Constructed with Transmission Gates

PROBLEMS

1. Find the Boolean function that corresponds to the closed paths through each of the given switch model networks in Figure 8.



(a) (b)

FIGURE 8
Switch Networks for Problem 1

2. Implement each of the following Boolean functions as closed paths through (1) an n-channel switch model network and (2) a p-channel switch model network. In each case, use a minimum number of transistors.
(a) $F(X, Y, Z) = YZ + \overline{X}Z + \overline{X}Y\overline{Z}$
(b) $F(A, B, C, D) = A\overline{D} + A\overline{B} + BD + BC$

3. Find the CMOS complex gate circuit for each of the following functions:
(a) $F(A, B, C, D) = (A + \overline{C})(\overline{A} + C)(B + \overline{D})(\overline{B} + D)$
(b) $W, X, Y, Z) = \Sigma m(4, 7, 9, 11, 12, 13, 14, 1, l(W, X, Y, Z) = \Sigma m(3, 10$

4. Find a multiple-level NAND circuit for F in Problem 3(a), and compare the number of transistors used with the number used in that problem. A NAND gate with n inputs uses $2n$ transistors.

5. Construct an exclusive-NOR circuit with two NOT gates and two transmission gates.