

# EK307 – Electric Circuit Theory – Fall 2009

## Supplemental Notes on Digital Logic and Systems\*

### DIGITAL CIRCUITS AND APPLICATIONS

Digital signals are binary in nature, in that they have the ability of taking on values in one of two well-defined ranges. We shall see below that the set of basic operations that can be performed on digital signals is quite small, and can easily be mastered by the beginner. Furthermore, the behavior of *any* digital system, up to and including the most sophisticated digital computer, can be represented by appropriate combinations of digital variables and the digital operations from this small set. Finally, digital integrated circuits with input-output characteristics that correspond to each of the basic digital operations are inexpensive and easy to use. Thus, it is possible for the beginning student to design and successfully construct a wide variety of digital circuits, including circuits that perform logic and arithmetic functions, circuits for data storage and transmission, and circuits that interface between peripheral equipment and the small, commercial general-purpose digital computers known affectionately as “mini-computers”.<sup>1</sup>

---

### ALGEBRAIC REPRESENTATIONS OF DIGITAL VARIABLES

This chapter is concerned with digital system variables that take on only two values (binary variables). We conventionally denote these values as “0” and “1,” and then use a special set of rules called *Boolean algebra* to summarize the various ways in which digital variables can be combined. This algebra and much of the notation are adopted directly from mathematical logic. Thus, “logic variable” or “logic operation” are commonly used in place of “digital variable” or “digital operation.”

*Definition of the AND operation.* Given two input variables,  $A$  and  $B$ , and an output variable  $C$ , the expression

$$C = A \text{ AND } B \quad (16.1)$$

means

$$C = 1 \quad \text{if} \quad A = 1 \quad \text{AND} \quad B = 1 \quad (16.2a)$$

otherwise

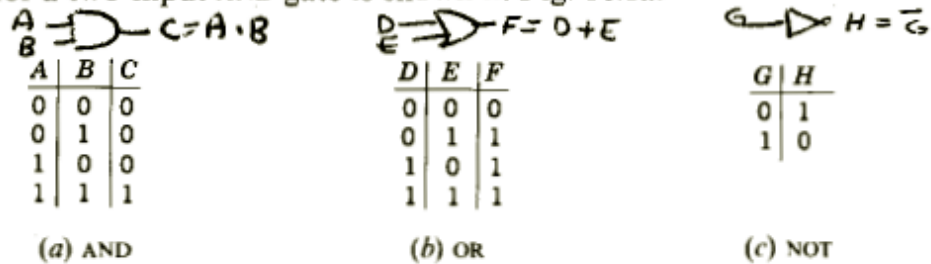
$$C = 0 \quad (16.2b)$$

---

\*All reproductions in this course packet fall under the Fair Use Doctrine of US Copyright

---

A circuit that performs the AND operation is called an AND gate. The logic symbol for a two-input AND gate is shown in Fig. 16.1a.



**Figure 16.1**  
Logic symbols and function tables for AND, OR, and NOT.

A dot is used as a shorthand for the AND operation, so that Eq. 16.1 may be written

$$C = A \cdot B \tag{16.3}$$

The dot is often omitted simplifying Eq. 16.3 further.

$$C = AB \tag{16.4}$$

One nice feature of digital operations is that the complete set of input-output variable values can be written down. Figure 16.1a shows such a *function table*, corresponding to Eq. 16.3, which lists all possible combinations of input variables *A* and *B* together with the corresponding output variable *C*. From this function table we see that in algebraic terms the AND operation is a form of multiplication, with these manipulation rules:

$$\begin{aligned} 0 \cdot 0 &= 0 \\ 0 \cdot 1 &= 1 \cdot 0 = 0 \\ 1 \cdot 1 &= 1 \end{aligned} \tag{16.5}$$

*Definition of the OR operation.* Given two input variables *D* and *E*, and an output variable *F*, the expression

$$F = D \text{ OR } E \tag{16.6}$$

means

$$\begin{aligned} F = 1 & \quad \text{if} \quad D = 1 \\ & \quad \text{OR} \quad E = 1 \\ & \quad \text{OR} \quad \text{both} \quad D = 1 \quad \text{and} \quad E = 1 \end{aligned} \tag{16.7}$$

The + sign is used as a shorthand for OR, and is *never* omitted in algebraic expressions. Thus, Eq. 16.6 is written algebraically as

$$F = D + E \quad (16.8)$$

Figure 16.1b shows the logic symbol used for the two-input OR gate together with the corresponding function table. Algebraically, the OR operation is a special form of addition performed according to these rules:

$$\begin{aligned} 0 + 0 &= 0 \\ 0 + 1 &= 1 + 0 = 1 \\ 1 + 1 &= 1 \end{aligned} \quad (16.9)$$

Note that the last manipulation,  $1 + 1 = 1$ , differs from the ordinary arithmetic use of the + sign.

As in ordinary algebra, parentheses may be used in Boolean expressions to group terms and give precedence to operations. If there are no parentheses, the AND functions in an equation are evaluated first.

*Definition of the NOT operation.* In some situations, the opposite value of a particular variable is required. In Boolean algebra, the opposite value of a variable is called the *complement* of that variable, and is denoted by a bar

drawn over the variable in question. The complement operation is summarized below using variable  $G$  as an example.

$$\begin{aligned} \text{If } G = 1 & \quad \text{then} \quad \bar{G} = 0 \\ \text{If } G = 0 & \quad \text{then} \quad \bar{G} = 1 \end{aligned} \quad (16.10)$$

The logic operation that produces the complement is called *inversion*, or the *NOT operation*. The logic symbol and function table for an inverter is shown in Fig. 16.1c.

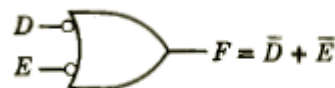
*The composite operations NAND and NOR.* Two combinations of basic operations arise so often that they are given individual names and logic symbols. The NOR operation is the complement of the OR operation (the name is simply a contraction of “NOT OR”), and is defined by

$$C = \overline{(A + B)} \quad (16.13)$$

Two equivalent symbols for the NOR gate, representing Eqs. 16.13 and 16.14 respectively, are shown in Fig. 16.2a along with the NOR function table. Note that the small circle adjacent to the input or output of the basic gate symbols produces the INVERSION of the variable in each case.

The complement of the AND operation is called the NAND operation (from “NOT AND”), and is defined by the two equivalent forms

$$F = \overline{D \cdot E} \tag{16.15}$$



A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

(a) NOR

D	E	F
0	0	1
0	1	1
1	0	1
1	1	0

(b) NAND

**Figure 16.2**

The principal importance of NOR and NAND is that they are the simplest logic functions to construct in integrated circuit form. Thus, while it may be easier for the beginner to learn to “think” with OR and AND, he should also practice thinking with NOR and NAND as these functions are likely to be used in the final circuit realization. Also, it is possible to synthesize all of the logic functions using only NOR gates or only NAND gates.

## A Simple Example

Let us formulate a simple everyday situation in terms of digital variables and Boolean operations. Suppose you are driving home and become thirsty for a hot drink. You see a diner ahead and pull in. Let us develop a Boolean equation for whether or not you obtain a drink.

The first step is to assign variables for the problem:

The diner is open for business  $\Rightarrow D = 1$ , or simply  $D$

( $\Rightarrow$  means implies)

The diner sells coffee  $\Rightarrow C$

The diner sells tea  $\Rightarrow T$

You get a drink  $\Rightarrow X$

The next step is to use AND and OR operations to construct this Boolean equation:

$$X = (C + T) \cdot D \quad (16.17)$$

Obtain drink  $\leftarrow$  sells coffee OR sells tea AND open for business

or omitting the dot

$$X = (C + T)D \quad (16.18)$$

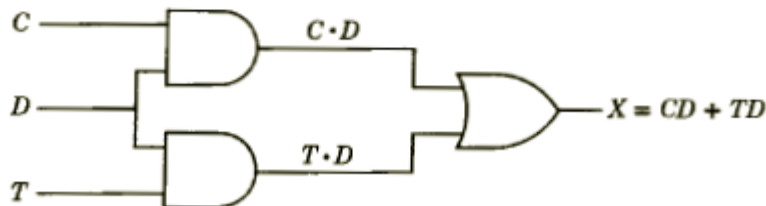
which is equivalent to

$$X = CD + TD \quad (16.19)$$

The final step is to construct a *logic flow diagram* using gate symbols. Two possible logic flow diagrams, corresponding to Eqs. 16.18 and 16.19, are shown in Fig. 16.3. As an illustration of how only NOR gates or only NAND gates can be used to synthesize any function, two additional implementations of this same example are shown in Fig. 16.4. Notice that when both inputs of the two-input NOR gate are connected together, as in Fig. 16.4a, the gate becomes an inverter.



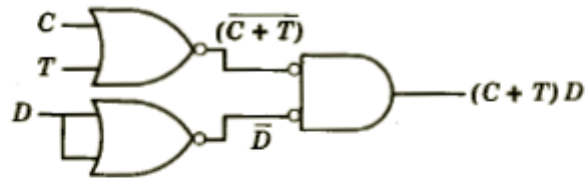
(a) Implementation of Eq. 16.18



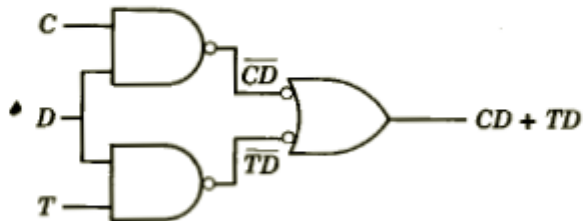
(b) Implementation Eq. 16.19

**Figure 16.3**

Logic flow diagrams for text example using AND and OR gates.



(a) NOR only



(b) NAND only

**Figure 16.4**

Implementation of text example using (a) only NOR gates and (b) only NAND gates.


## SUMMARY OF GATE CHARACTERISTICS

### 1) The NOT Gate

Symbol	Boolean Equation	Truth Table						
		Input    Output						
	$X = \bar{A}$	<table border="1"> <tr> <td>A</td> <td>X</td> </tr> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </table>	A	X	0	1	1	0
A	X							
0	1							
1	0							

Because the NOT gate has only one input, the truth table has two rows. Moreover the output inverts the logic level of the input. In addition to the overhead bar shows above (read as "X = A-bar"), notation for logical inversion includes the following !A, /A, -A, A\*.


2) **The NAND Gate**

Symbol	Boolean Equation	Truth Table																
		Inputs	Output															
	$Y = \overline{A B}$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Y</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	B	Y	0	0	1	0	1	1	1	0	1	1	1	0	
A	B	Y																
0	0	1																
0	1	1																
1	0	1																
1	1	0																

The behavior of a NAND gate can be summarized as follows: The output is LOW only when **all** the inputs are HIGH. If one or more inputs are LOW (false or logic 0), the output will be HIGH. Comparing the truth table for the NAND gate with that of the AND gate, you will find out that each output of a NAND gate is exactly the opposite (inverted) logic value of the corresponding output of an AND gate. In fact, a NAND gate is functionally equivalent to an AND gate cascaded with a NOT gate as shown below.



3) **The NOR Gate**

Symbol	Boolean Equation	Truth Table																
		Inputs	Output															
	$Z = \overline{A + B}$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	0	
A	B	Z																
0	0	1																
0	1	0																
1	0	0																
1	1	0																

As seen from the above truth table, the output of a NOR gate is HIGH only when **all** the inputs are LOW. If one or more of the inputs are HIGH, then the output is LOW.

Similarly, a NOR gate can be constructed using an OR gate cascaded with a NOT gate. In other words, a NOR gate is functionally equivalent to an OR gate followed by an inverter.

## Digital Systems and Information Representation

Almost all digital systems utilize circuits with *binary logic*. In binary digital logic, **data** is represented by a voltage (or occasionally a current) switching between one of two possible levels, a low level called a logic 0, and a high level called a logic 1. This process is termed “asserted positive logic.” The system could be implemented using the opposite convention where the high level is considered a logic 0 and the low level a logic 1, and such a system is said to use “asserted negative logic.” Positive logic is by far the most common.

The single unit of digital information, the *bit*, is represented by the transmission of a 1 or a 0. The bit has been found to be a very useful means of quantizing information. Information theory, a subspecialty within the field of communications, has developed expressions for calculating the number of equivalent bits of information contained in various signals.

Each bit conveys to us whether the information or value is in the top half or lower half of its possible range. For example, if we consider a signal,  $v$ , which we know has values in the range  $0 < v < 2$ , one bit of information would tell us the following: if the voltage is in the range  $0 < v < 1$ , then the bit status could be 0, and if the voltage is in the range  $1 < v < 2$ , the bit status would be 1.

This is very coarse information about this signal,  $v$ . We originally knew its value was between zero and two, and now we have started to “pin it down” by breaking the total range into two zones. We use this first bit, the *most significant bit* (MSB), to tell us whether we are in the upper 50% or lower 50% of the entire range.

We can add a second bit to tell us whether we are in the top half or lower half of the smaller range just selected by the previous bit. We are now capable of quantifying the signal to within four possible “zones” using the following correspondence:

Range	Bit status	
	MSB	LSB
$0.0 < v < 0.5$	0	0
$0.5 < v < 1.0$	0	1
$1.0 < v < 1.5$	1	0
$1.5 < v < 2.0$	1	1

Where MSB = most significant bit and LSB = least significant bit.

We could now add a third bit which would give us a resolution of eight possible ranges, or zones. Note that in the binary system, the number of zones of resolution,  $Z$ , is related to the number of bits,  $n$ , by the expression

$$Z = 2^n \quad (7.5)$$

The series of bits that represent a particular analog level is termed a digital *word*. If this series of bits is presented at the same time via parallel conductors, that is, a parallel bus, we receive a parallel word; if they appear one after the other via a single conductor, that is, a serial data bus, then we receive a serial word. Eight-bit word segments are normally referred to as a *byte*.



**EXAMPLE**

Let us determine the voltage resolution that can be achieved if a signal with a maximum value of 12 volts and minimum value of 0 volts is represented by 10 bits.

The number of zones defined by 10 bits is:

$$Z = 2^n = 2^{10} = 1024 \text{ zones}$$

$$\text{Resolution} = \frac{12.0}{1024} = 0.0117 \text{ volts/zone}$$

**D7.3.** A 4-bit parallel data bus can transmit 8 bits of information in 40 microseconds. If the same technology were used to transmit data serially over a single conductor, how many bits could be transmitted in the same 40 microseconds?

*Ans:* 2 bits.

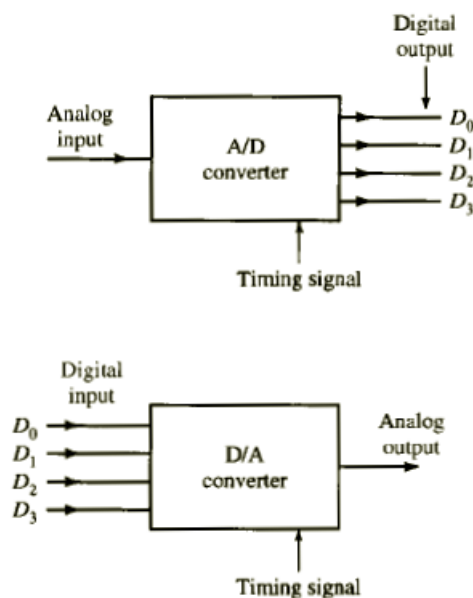
**D7.4.** A signal that can have values between  $-10$  to  $+10$  volts is represented by a 12-bit digital word; what is the voltage resolution obtainable with this system?

*Ans:* 4.9 mV.

**A/D and D/A Converters**

Electronic circuits, which are designed to automatically convert an analog voltage into a digital or binary representation of that voltage, are called *Analog to Digital Converters*, or *A/D Converters*. Circuits that convert a digital word into an analog voltage level are called *Digital to Analog Converters*, or *D/A converters*. Both converters are shown symbolically in Fig. 7.9. A timing signal is used to specify the point in time at which a sample is taken.

An 8-bit A/D converter, for example, converts a continuously variable voltage within a specified range at its input, into an 8-bit digital word at its output which represents the



**Figure 7.9** A/D and D/A converters.

quantized value of the input voltage at a specific point in time. Therefore, we are told which of 256 zones the input voltage is contained within ( $2^8 = 256$ ) at the time of the sample. A/D and D/A converters are available from many manufacturers integrated on a single IC chip.

## 15.4 ANALOG-TO-DIGITAL INTERFACING

A complete treatment of digital circuits must also include a discussion of digital-to-analog (D/A) and analog-to-digital (A/D) conversion. Although physical measurement usually involves analog variables, most data collection, information transmission, and signal analysis are performed digitally. Analog-to-digital and digital-to-analog circuits provide the all-important interfaces between the analog and the digital worlds.

### 15.4.1 Digital-to-Analog Conversion

A digital-to-analog (D/A) converter produces a single analog output voltage from a multibit digital input. The decoding can be performed using a variety of D/A algorithms. One common algorithm produces an analog output proportional to a fixed reference voltage, as determined by the equation

$$v_{\text{OUT}} = \frac{n}{2^N - 1} V_{\text{REF}} \quad (15.25)$$

In this equation,  $N$  is the number of bits in the digital input word, and  $n$  is the decimal integer represented by the input bits that are set to 1. The voltage  $V_{\text{REF}}$  can be either positive or negative. Other D/A converters, which produce output voltages over a bipolar voltage range, employ the two's complement decoding scheme, to be described in Section 15.4.3.

---

**EXAMPLE 15.1** The input to a 10-bit D/A converter with a reference voltage of 5 V is (00 1001 0001). Find the resulting analog output.

#### **Solution**

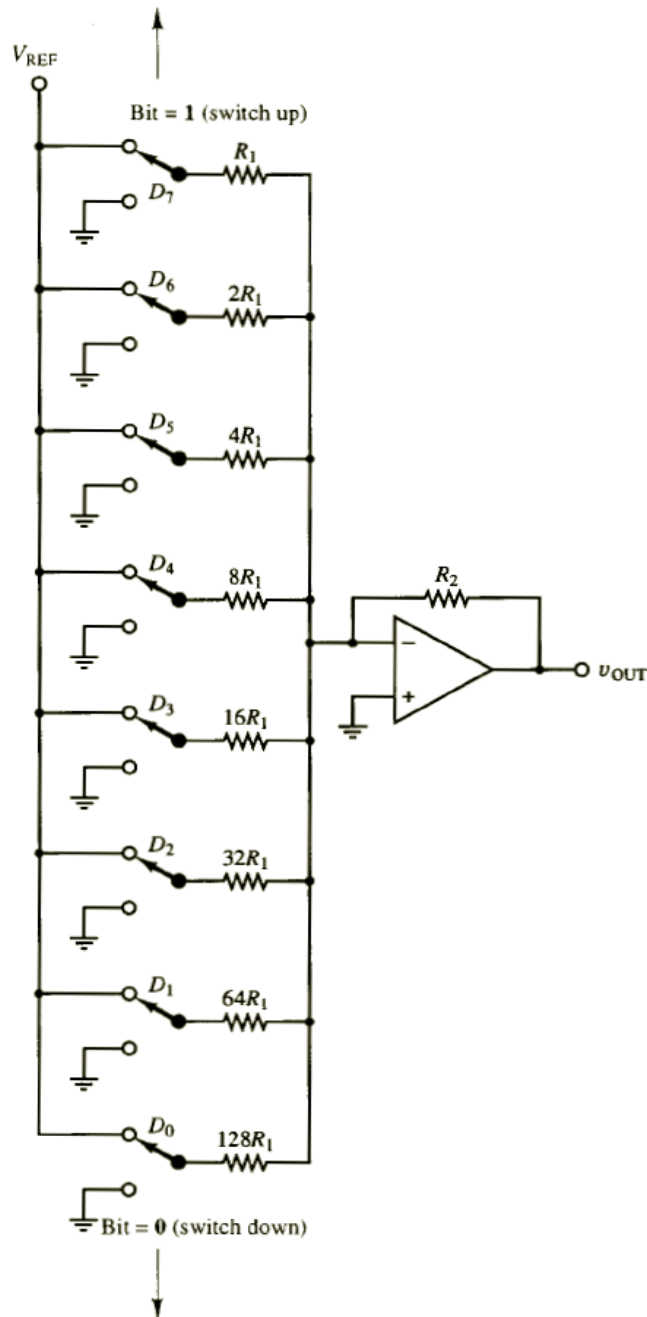
The decimal integer  $n$  represented by the specified digital input is  $128 + 16 + 1 = 145$ . In this case,  $N = 10$  and  $2^{10}$  is equal to 1024; hence the output given by Eq. (15.25) becomes

$$v_{\text{OUT}} = \frac{145}{1023} (5 \text{ V}) = 0.709 \text{ V} \quad (15.26)$$

#### **Summing Amplifier D/A Converter**

One circuit that can perform D/A conversion using the algorithm of Eq. (15.25) utilizes the op-amp summation amplifier of Chapter 2. In a summing converter, the digital input bits to be decoded control the parallel input nodes of a summation amplifier, as depicted in Fig. 15.30. The input terminals of this circuit are connected to either ground or to  $V_{\text{REF}}$  by a set of two-position switches labeled  $D_0$  through  $D_7$ . The status of each switch is determined by the value of its corresponding digital input bit. If a given input bit is equal to 1, the switch represented by the bit is connected to  $V_{\text{REF}}$ ; if the bit is equal to 0, the switch is connected to ground. In practice, switches  $D_0$  through  $D_7$  are made from MOSFET devices driven by the digital input signals. An 8-bit converter is shown in Fig. 15.30; a converter with any number of bits could be made by changing the number of input channels.

**Figure 15.30**  
 Eight-bit D/A  
 converter made  
 from summation  
 amplifier. This  
 converter  
 essentially  
 implements the  
 decoding algorithm  
 given by  
 Eq. (15.25) but  
 creates a negative  
 output due to the  
 inversion of the  
 summation  
 amplifier.



The D/A decoding performed by this circuit is realized by selecting proper values for the resistors. Specifically, beginning with the most significant bit, the input resistor of each successive channel must be made two times larger than its predecessor as the weighting of the corresponding binary bit is decreased. The arrangement is illustrated in Fig. 15.30, where, for example, the input resistor to  $D_7$  has value  $R_1$ , the input resistor to  $D_6$  is  $2R_1$ , the resistor to  $D_5$  is  $4R_1$ , and so on. This ordering of resistor values allows the bit with the highest weighting (the  $D_7$  bit) to amplify  $V_{REF}$  by the largest gain, and the bit with the smallest weighting (the  $D_0$  bit) to amplify  $V_{REF}$  by the smallest gain. An 8-bit converter like the one shown in Fig. 15.30 has input resistors that vary between the values  $R_1$  and  $128R_1$ . Given this geometrical progression of resistor values, the

output of the circuit can be expressed by the equation

$$v_{OUT} = -\frac{R_2}{R_1} \left( D_7 + \frac{D_6}{2} + \frac{D_5}{4} + \dots + \frac{D_1}{64} + \frac{D_0}{128} \right) V_{REF} \quad (15.27)$$

The variables  $D_0$  through  $D_7$  in this equation represent integers with values of **1** or **0**, as determined by the status of the eight digital input bits.

For the general case of an  $N$ -bit converter, Eq. (15.27) becomes

$$v_{OUT} = -\frac{R_2}{R_1} \sum_{n=0}^{N-1} \frac{D_n}{2^K} V_{REF} \quad (15.28)$$

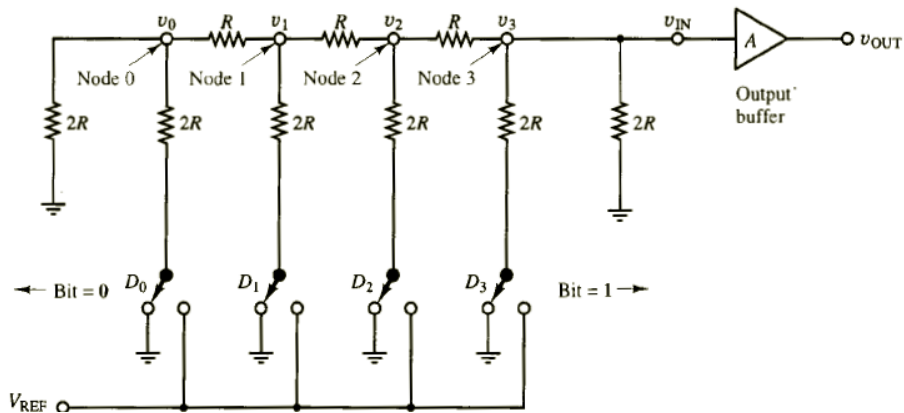
where  $K = 2^{N-1-n}$ .

The principal disadvantage of the summing D/A converter shown in Fig. 15.30 is its sensitivity to the input resistor values. These resistors must be fabricated to close tolerances over a wide range of values. If the converter is fabricated on an integrated circuit, adequate control of resistor values may not be possible.

### Resistive Ladder Converter

The D/A converter shown in Fig. 15.31 is called a *resistive ladder converter*. Its central component is a ladder network in which horizontally drawn resistors have a value of  $R$  and vertically drawn resistors have a value of  $2R$ . This network is sometimes called an  $R-2R$  network. The output buffer in Fig. 15.31 (usually an op-amp voltage follower) minimizes the current drawn from the ladder network, thereby reducing the loading at the  $v_3$  node (node 3). For simplicity, a 4-bit converter is shown in Fig. 15.31. The number of bits could be increased to any number by extending the length of the ladder network.

**Figure 15.31**  
Four-bit resistive  
ladder D/A  
converter.



The digitally controlled switches in the circuit connect the lower resistor terminals to either  $V_{REF}$  or ground. Each of the switches is controlled by one bit of the digital input word, with  $D_0$  representing the least significant bit. In practice, the entire converter is fabricated on a single integrated circuit, and the switches are constructed from digitally driven MOSFETs.

An examination of the circuit reveals one of its more interesting characteristics. With all switches connected to ground, each of the numbered nodes 0 through 3 is connected directly to ground via a resistor of value  $2R$ . At the same time, the equivalent resistance to ground seen to either the right or to the left of each numbered node is equal to  $2R$ . For example, the resistance seen to the right of node 3 consists of a single resistor  $2R$ . The resistance seen to the right of node 2 (including the resistance  $2R$  between node 3 and ground) becomes the sum of  $R$  plus the parallel combination of  $2R$  and  $2R$ . Altogether, this combination is equal to  $2R$ . The resistance seen to the right of node 1 then becomes  $R$  plus the parallel combination of  $2R$  and  $2R$ , which again is equal to  $2R$ . This same reasoning can be applied to node 0 and can also be applied when computing the resistance seen to the left of any node.

If the  $n$ th switch is connected to  $V_{REF}$ , with all other switches connected to ground, the voltage  $v_n$  appearing at the node directly above it can be determined by applying the voltage-divider relation, that is,

$$v_n = V_{REF} \frac{2R \parallel 2R}{2R + 2R \parallel 2R} = \frac{V_{REF}}{3} \quad (15.29)$$

The factor  $2R \parallel 2R$  in Eq. (15.29) represents the parallel combination of resistances seen to the right and to the left of the  $n$ th node; the single factor of  $2R$  in the denominator represents the resistance between the  $n$ th switch and its associated node.

The fraction of  $v_n$  that appears as  $v_{IN}$  at the input to the buffer depends on the node's position along the ladder. The node voltage  $v_3$ , for example, is applied directly to the  $v_{IN}$  terminal. The voltage  $v_2$ , however, is attenuated by a voltage divider formed by  $R$  and  $2R \parallel 2R$ , so that  $v_{IN} = v_2/2$ . A similar consideration shows that  $v_1$  is attenuated at node 2 by a voltage divider formed from the resistance  $R$  and the combination  $2R \parallel (R + 2R \parallel 2R)$ . The latter combination is equivalent to a resistance of value  $R$ , so that

$$v_2 = v_1 \frac{R}{R + R} = \frac{v_1}{2} \quad (15.30)$$

The fraction of  $v_1$  that appears at  $v_{IN}$  thus becomes

$$v_{IN} = \frac{v_2}{2} = \frac{v_1}{4} \quad (15.31)$$

Applying this attenuation algorithm to all the nodes in the ladder, together with the superposition principle and Eq. (15.29), yields the total output voltage of the circuit as a function of all the switch values:

$$v_{OUT} = \frac{V_{REF}}{3} \left( D_3 + \frac{D_2}{2} + \frac{D_1}{4} + \frac{D_0}{8} \right) \quad (15.32)$$

Equation (15.32) assumes that  $v_{OUT}$  is equal to  $v_{IN}$ , that is, that the output buffer has unity gain. Each of the variables  $D_0$  through  $D_3$  in this expression is equal to either 0 or 1. As Eq. (15.32) suggests, the ladder network applies appropriate weighting to each digital bit in determining the analog output voltage.

For a ladder converter with  $N$  input bits, Eq. (15.32) can be generalized to the expression

$$v_{OUT} = \frac{V_{REF}}{3} \left( D_{N-1} + \frac{D_{N-2}}{2} + \cdots + \frac{D_1}{2^{N-2}} + \frac{D_0}{2^{N-1}} \right) \quad (15.33)$$

In contrast to the summing converter, which is sensitive to the absolute values of its resistors, the ladder converter is sensitive only to the ratio of the resistors  $R$  and  $2R$ . In the environment of an integrated circuit, resistors with precisely fixed ratios of 2:1 are easily fabricated.