Digital Fundamentals

Thomas L. Floyd

Functions of Combinational Logic
Chapter 6

Half-Adders

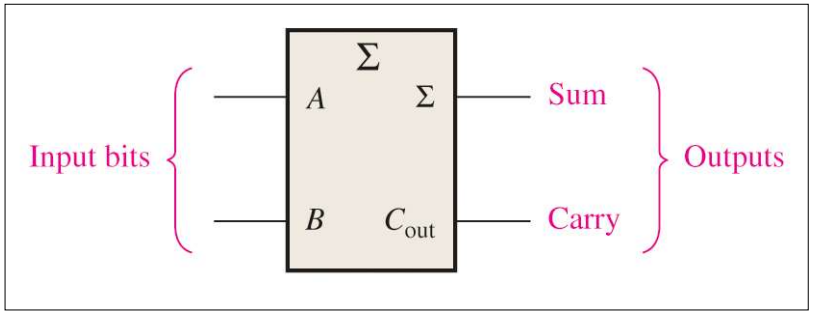
Input: A, B (binary)

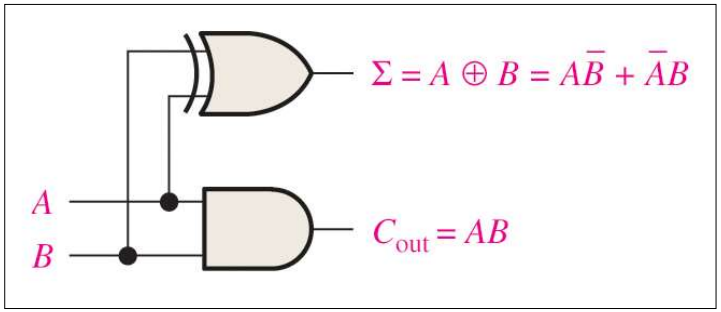
Output: Sum, Carry-out (binary)

TABLE 5–1 • Half-adder truth table.

A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Σ = sum
C_{out} = output carry
A and B = input variables (operands)





Full-Adders

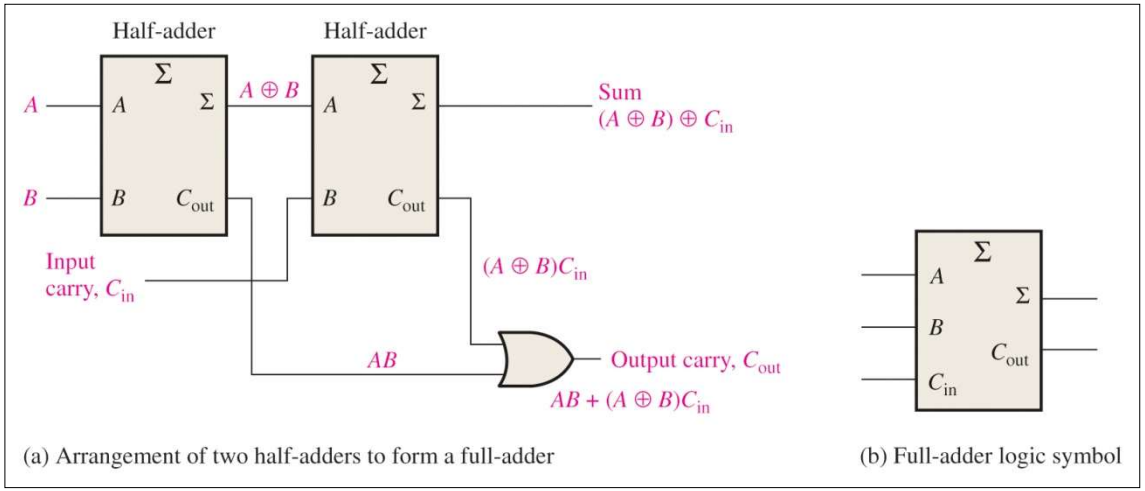
Input: A, B, Carry-in (binary)

Output: Sum, Carry-out (binary)

TABLE 5–2 • Full-adder truth table.

A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

C_{in} = input carry, sometimes designated as *CI*
C_{out} = output carry, sometimes designated as *CO*
Σ = sum
A and B = input variables (operands)

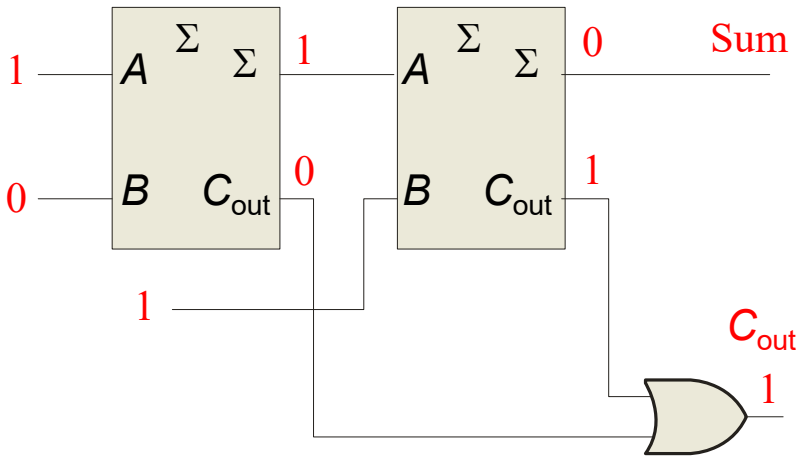


(a) Arrangement of two half-adders to form a full-adder

(b) Full-adder logic symbol

Ch.6 Summary

Full-Adders



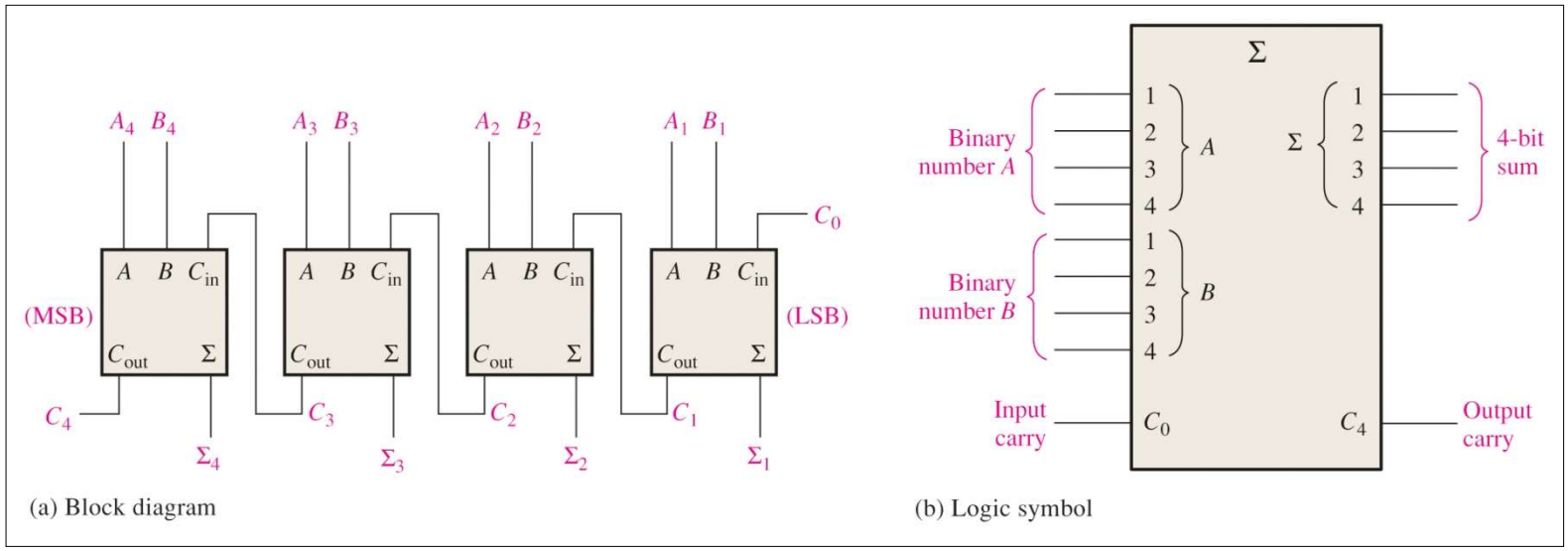
The first half-adder has inputs of 1 and 0; therefore the Sum =1 and the Carry out = 0.

The second half-adder has inputs of 1 and 1; therefore the Sum = 0 and the Carry out = 1.

The OR gate has inputs of 1 and 0, therefore the final carry out = 1.

Parallel Adders

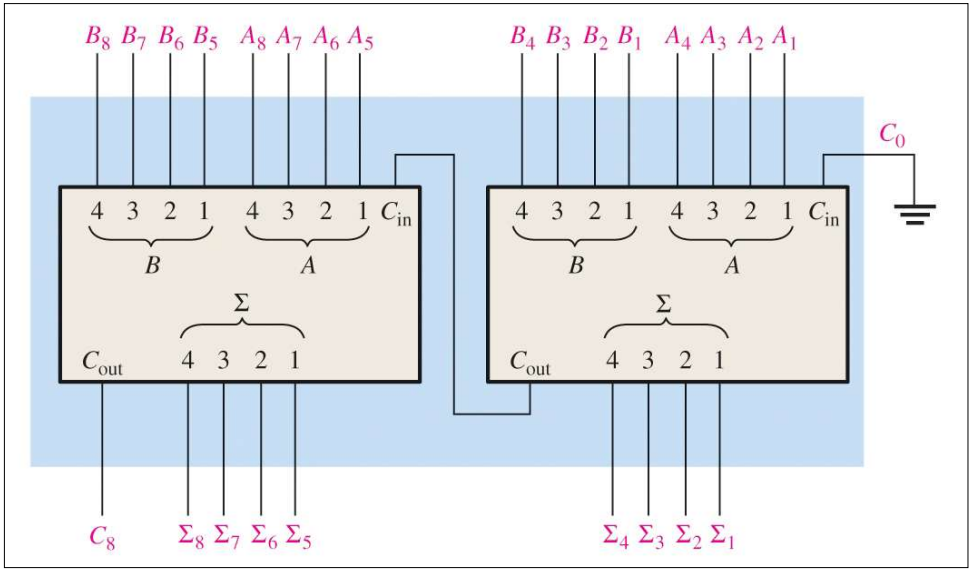
Full adders are combined into parallel adders that can add binary numbers with multiple bits. A 4-bit adder is shown.



The output carry (C_4) is not ready until it propagates through all of the full adders. This is called *ripple carry*, which delays the addition process.

Adder Expansion

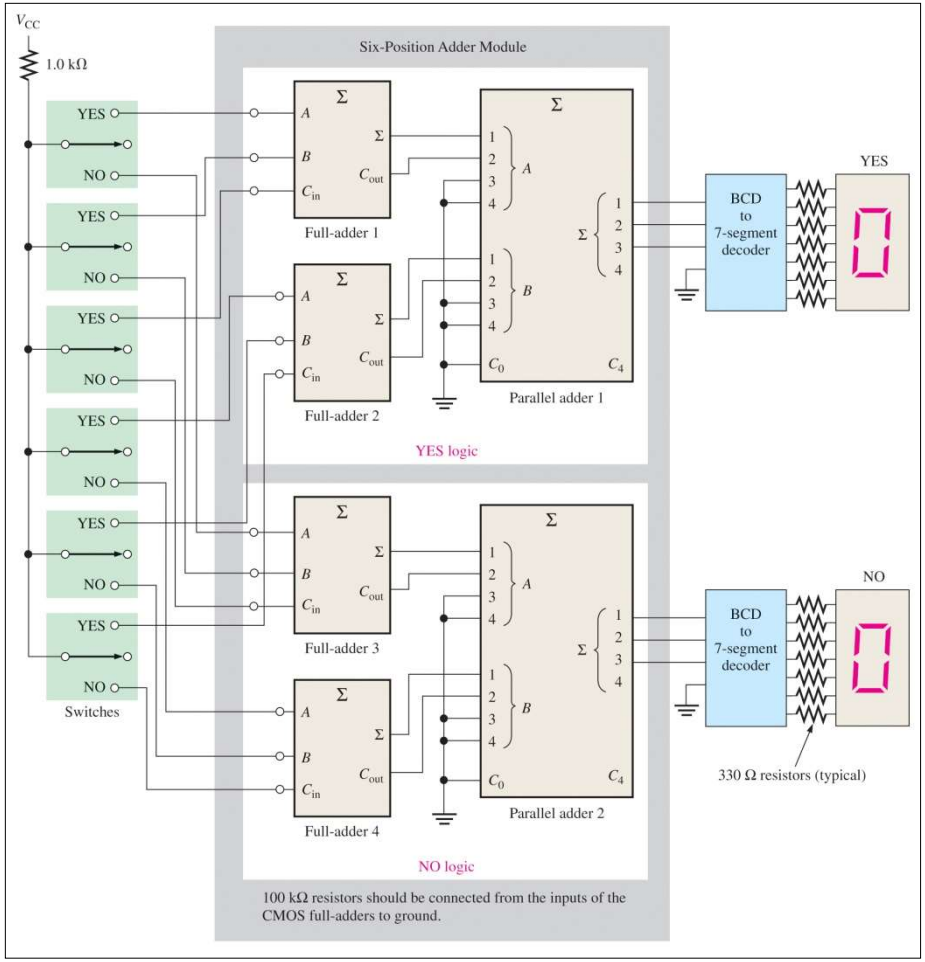
Two four-bit adders can be **cascaded** to form an 8-bit adder as shown.



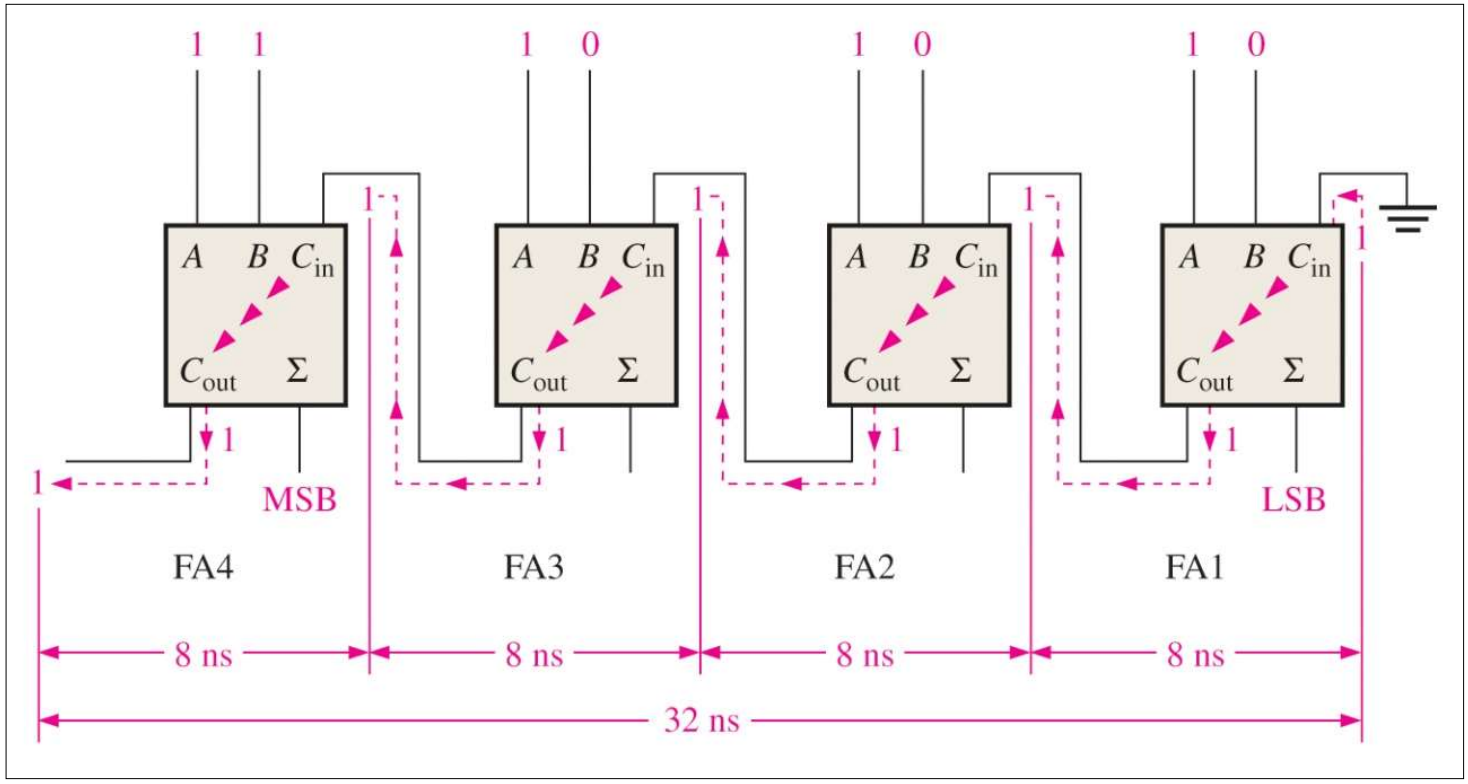
The carry-in (C_0) pin on the lower-order adder is grounded and the carry-out pin is connected to the C_0 pin of the higher-order adder.

An Adder Application

A voting system



Ripple Carry Adder

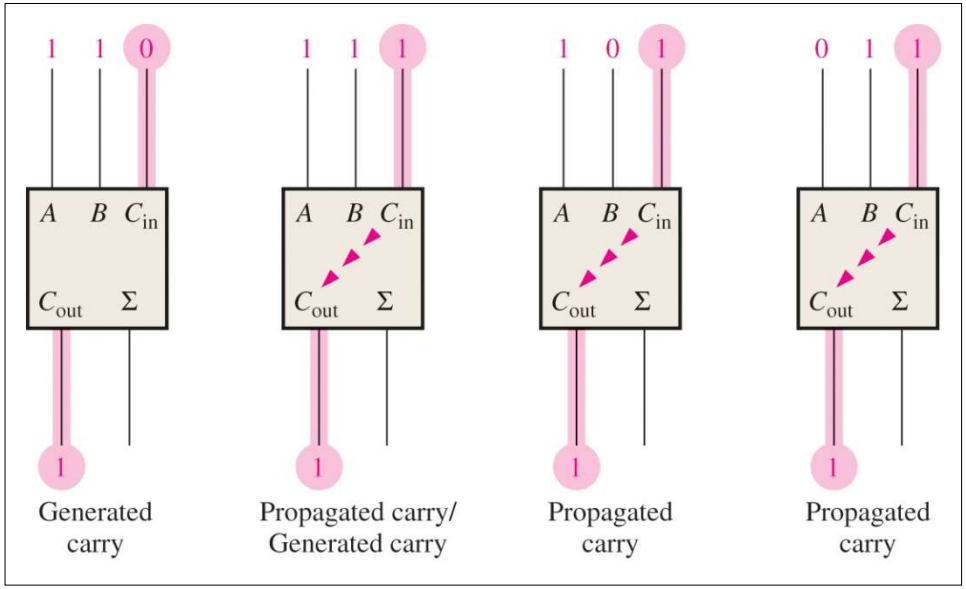


Look-Ahead Carry Adder

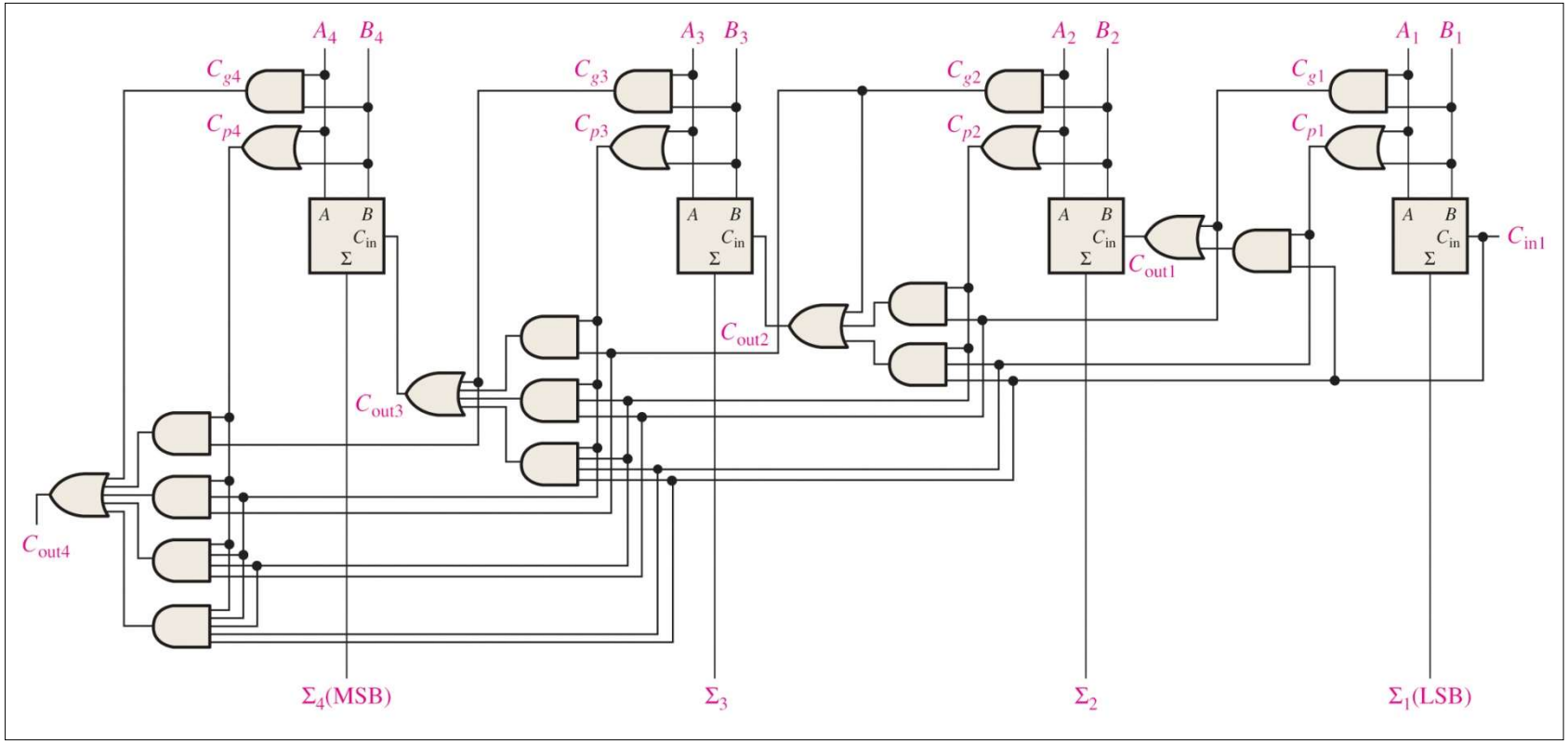
A **look-ahead carry adder** anticipates the carry

Carry generation: When a carry signal is produced internally (i.e., a carry-in is not involved).

Carry propagation: When an input carry signal is involved in producing a carry out signal.



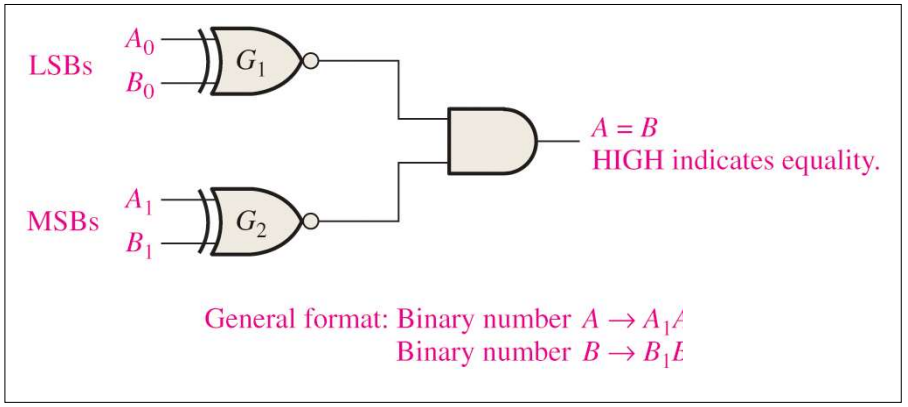
A Four-Stage Look-Ahead Carry Adder



Comparators

The function of a comparator is to compare the magnitudes of two binary numbers to determine the relationship between them. In the simplest form, a comparator can test for equality using XNOR gates.

How could you test two 2-bit numbers for equality?

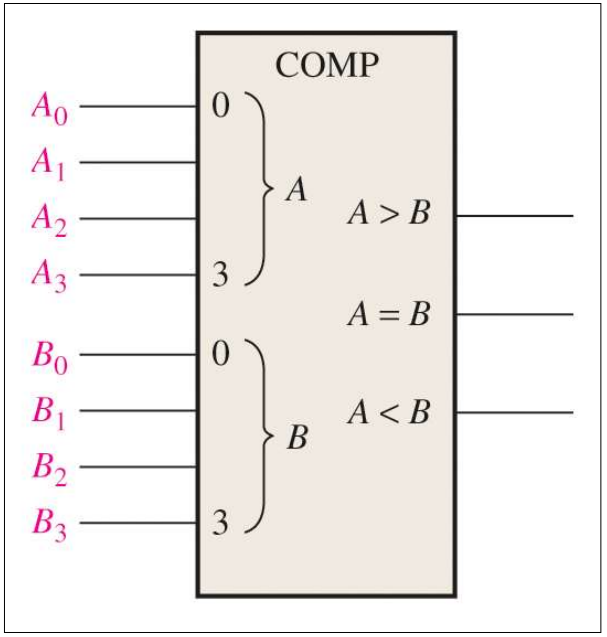


AND the outputs of two XNOR gates

Comparators

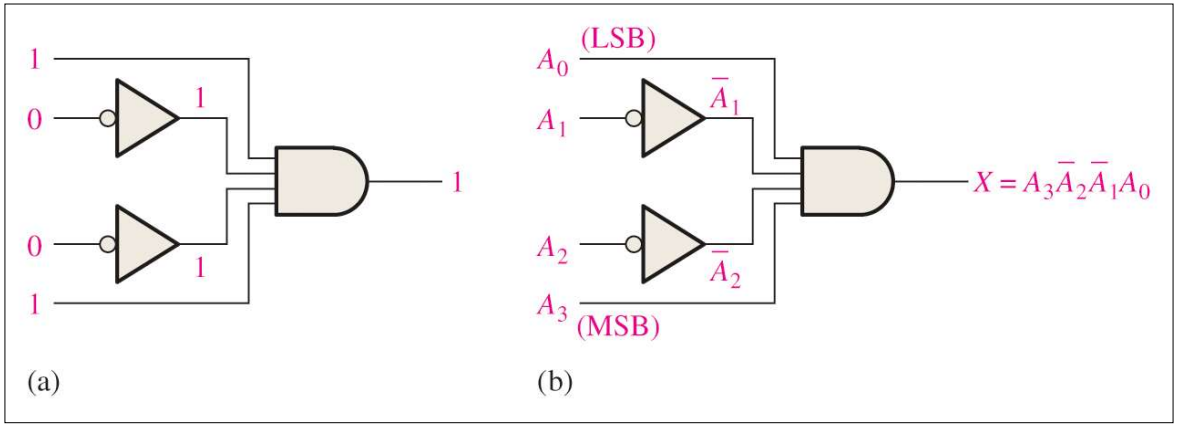
IC comparators provide outputs to indicate which of the numbers is larger or if they are equal. The bits are numbered starting at 0, rather than 1 as in the case of adders.

Cascading inputs are provided to expand the comparator to larger numbers.



Decoders

A **decoder** is a logic circuit that detects the presence of a specific combination of bits at its input. A simple decoder that detects the presence of the binary code 1001 is shown.

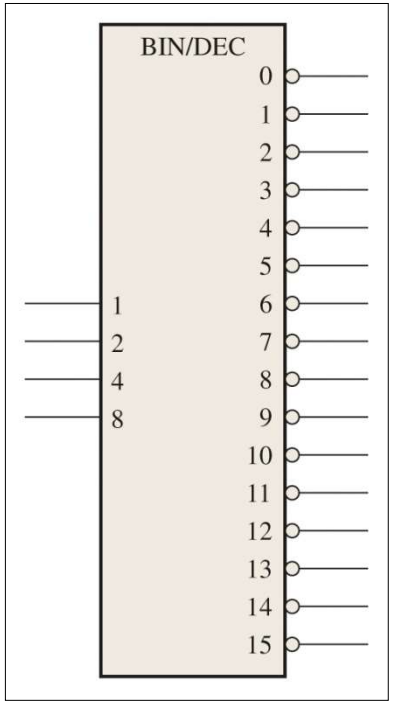


The circuit on the left has an active HIGH output for the inputs shown; the circuit on the right shows the logic expressions for the various gate outputs.

Decoders

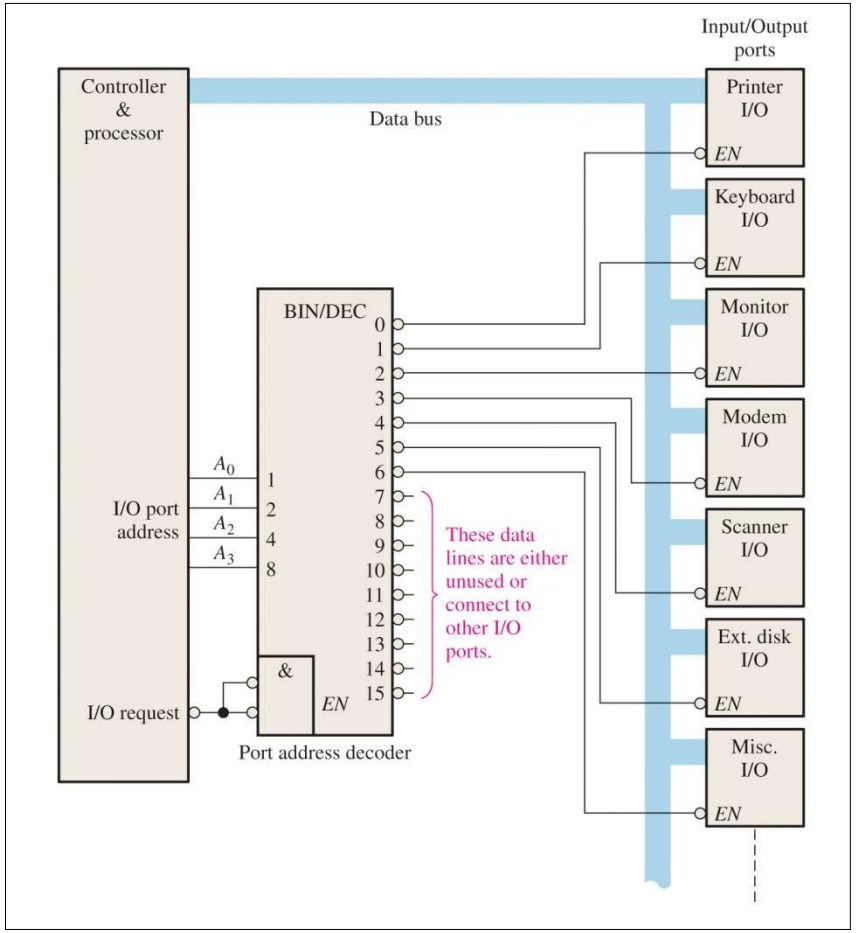
IC decoders have multiple outputs to decode any combination of inputs

DECIMAL DIGIT	BINARY INPUTS A_3	A_2	A_1	A_0	DECODING FUNCTION	OUTPUTS 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1



A Decoder Application

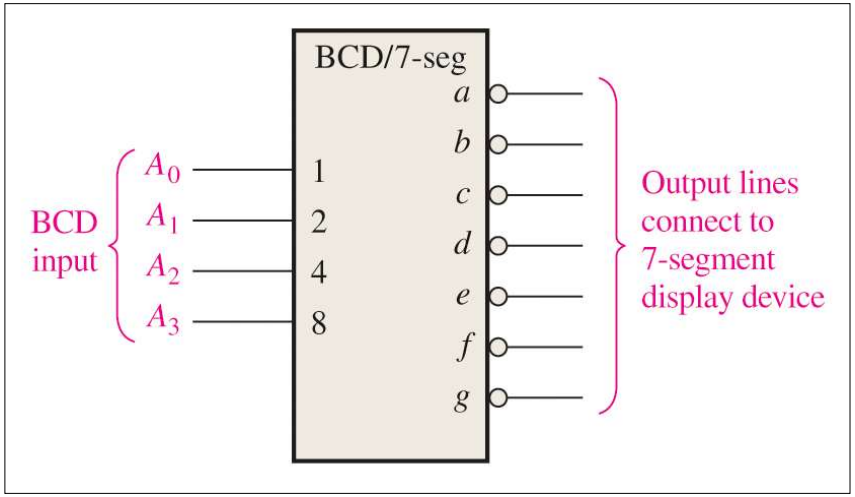
A simplified I/O Port System



BCD Decoder/Driver

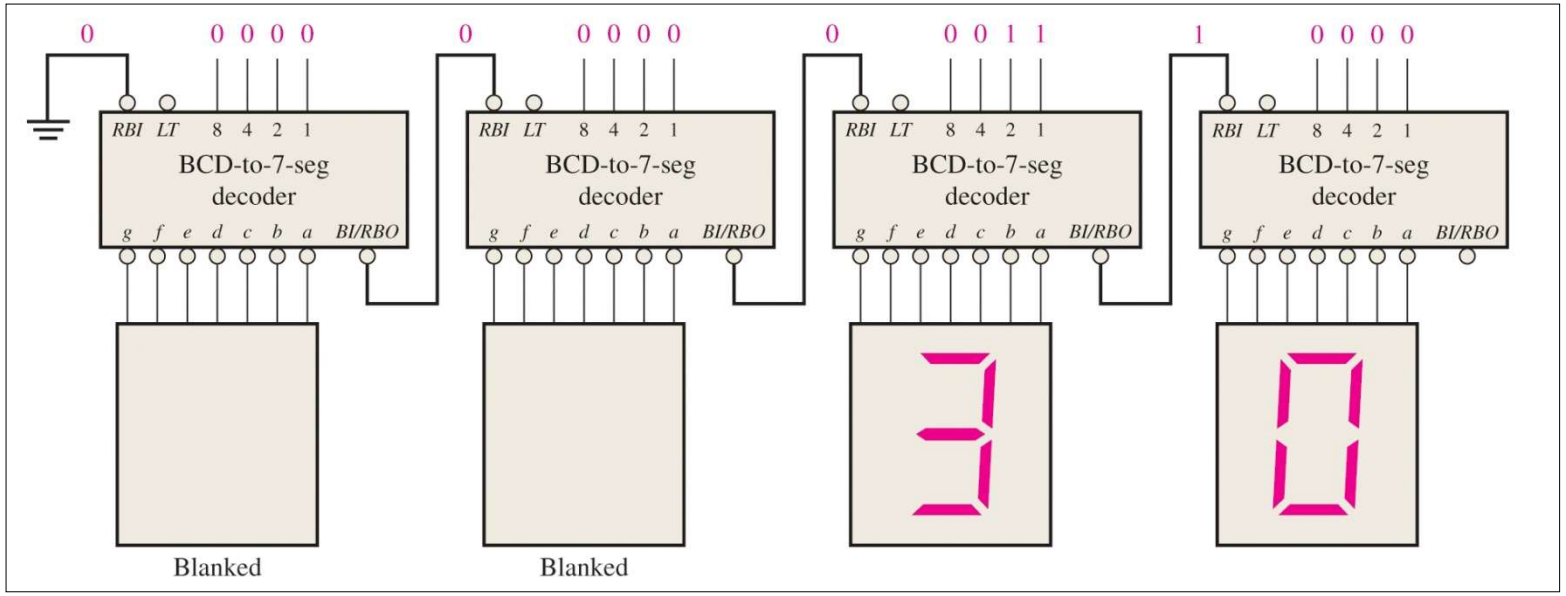
Another useful decoder is the 74LS47. This is a BCD-to-seven segment display with active LOW outputs.

The *a-g* outputs are designed for much higher current than most devices (hence the word driver in the component's name).



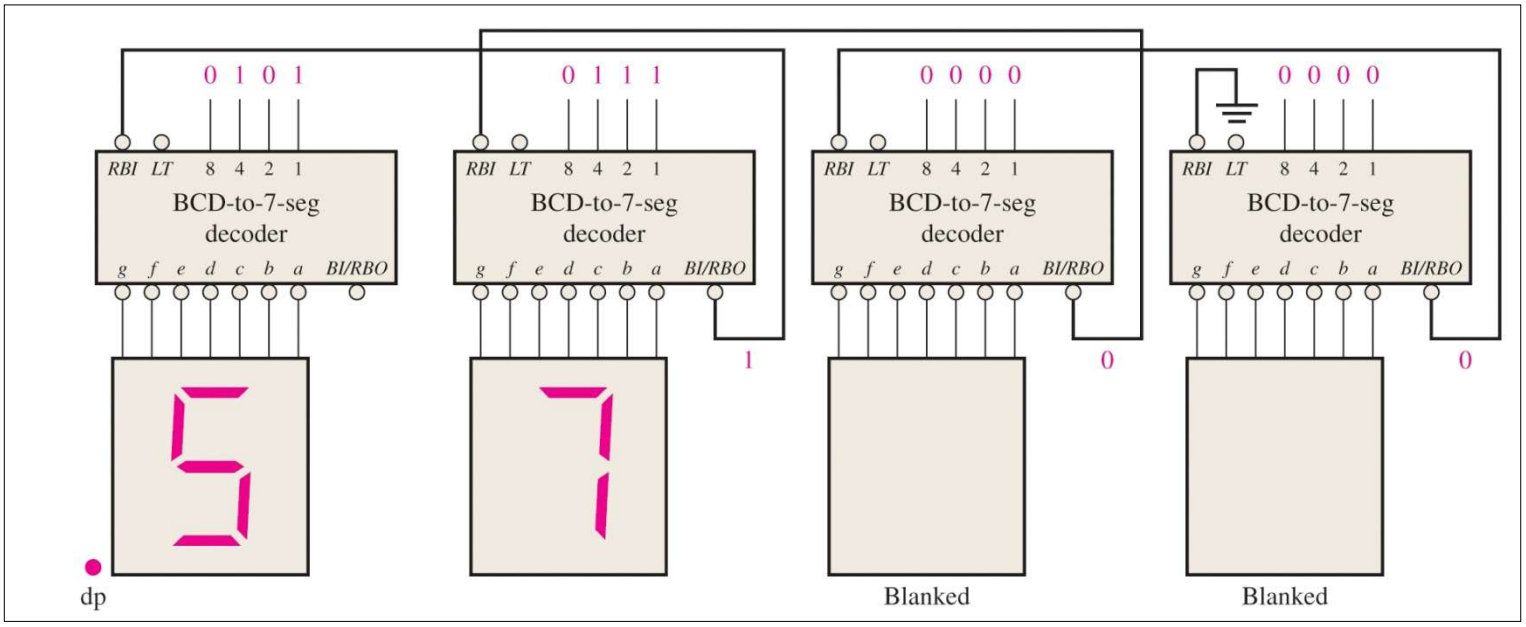
Leading Zero Suppression

The 74LS47 features leading zero suppression, which blanks unnecessary leading zeros but keeps significant zeros as illustrated here. The $\overline{BI}/\overline{RBO}$ output is connected to the $\overline{RBI}$ input of the next decoder.



Trailing Zero Suppression

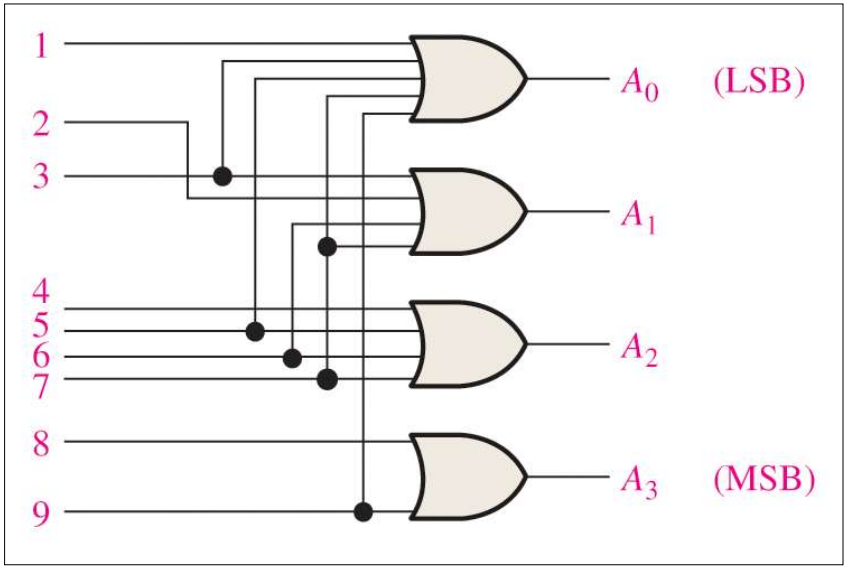
Trailing zero suppression blanks unnecessary trailing zeros to the right of the decimal point as illustrated here. The *RBI* input is connected to the *BI/RBO* output of the following decoder.



Encoders

An **encoder** accepts an active logic level on one of its inputs and converts it to a coded output, such as BCD or binary.

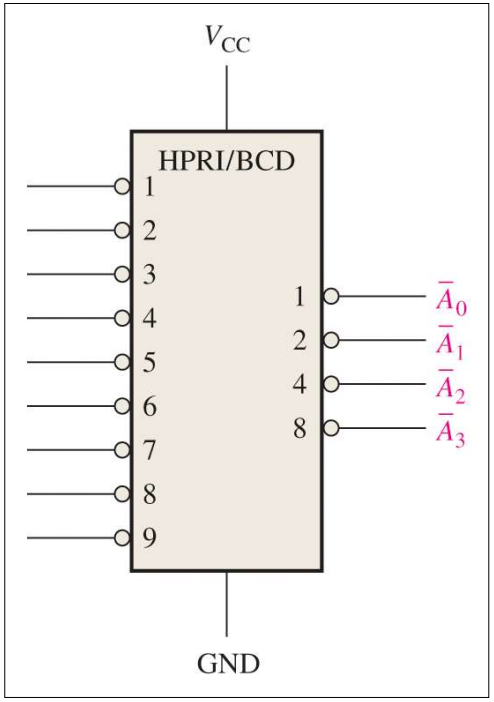
The decimal to BCD is an encoder with an input for each of the ten decimal digits and four outputs that represent the BCD code for the active digit. The basic logic diagram is shown. There is no zero input because the outputs are all LOW when the input is zero.



Encoders

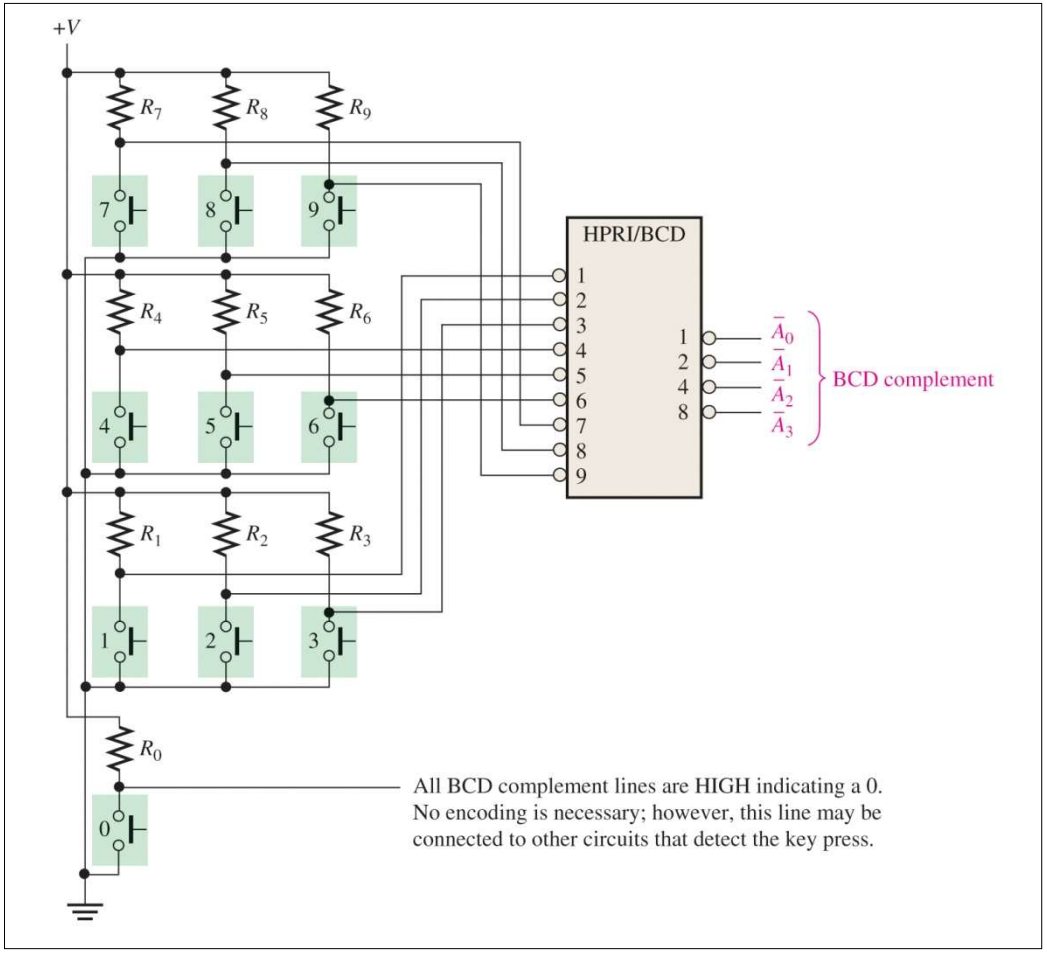
The 74HC147 is an example of an IC encoder. It is has ten active-LOW inputs and converts the active input to an active-LOW BCD output.

This device is a **priority encoder**. This means that if more than one input is active, the component responds to the highest numbered input.



An Encoder Application

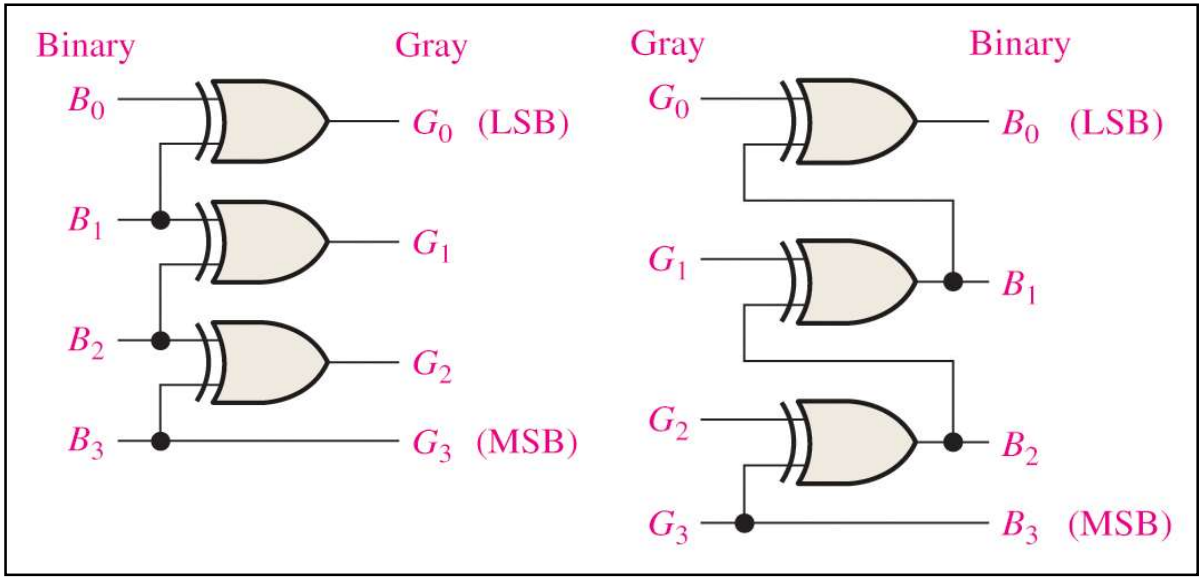
A keypad encoder



Code Converters

There are various code converters that change one code to another. Two examples are the four bit binary-to-Gray converter and the Gray-to-binary converter.

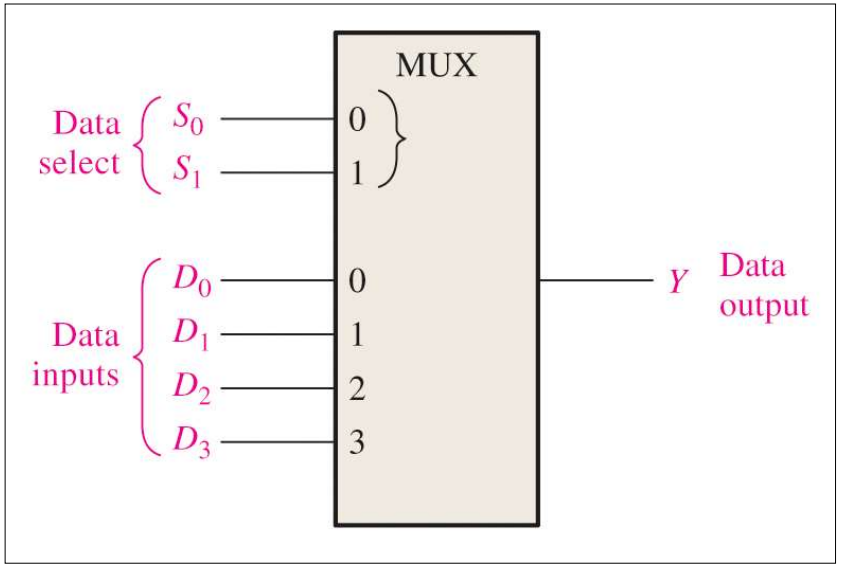
Show the conversion of binary 0111 to Gray and back to binary.



Multiplexers

A multiplexer (MUX) selects one of several data (*D*) inputs and routes data from that input to the output. The data line that is selected is determined by the select (*S*) inputs.

The multiplexer shown has two select (S) inputs that are used to select one of four data (D) inputs.

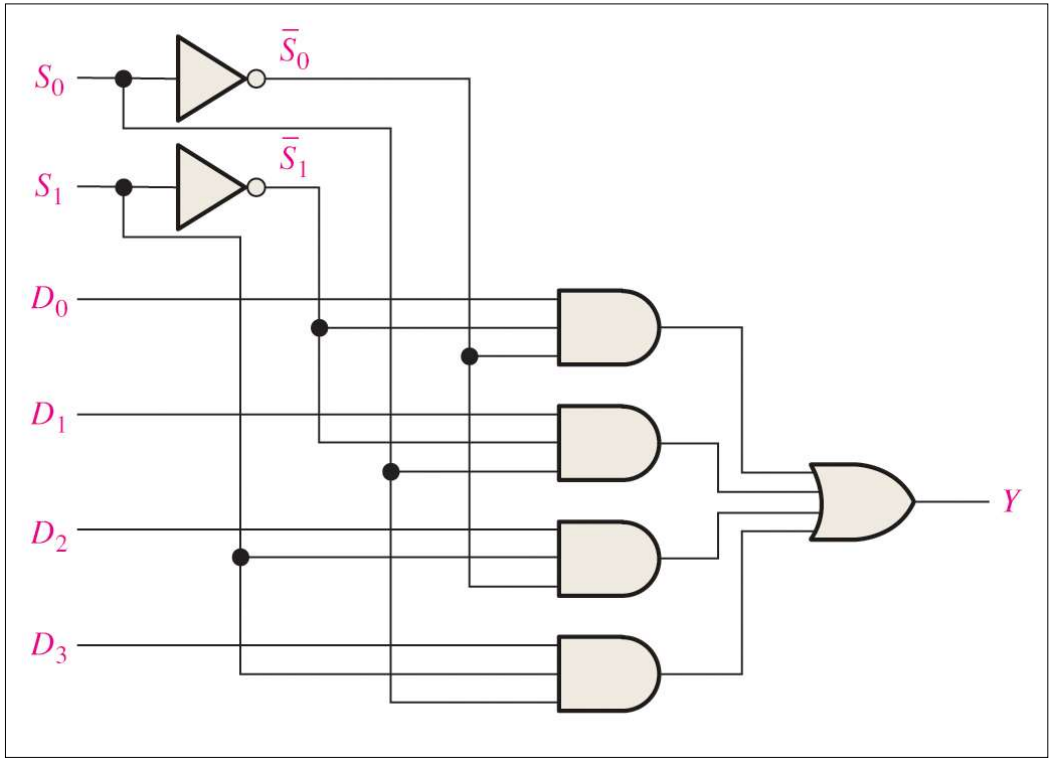


Which data line is selected if $S_1S_0 = 10$?

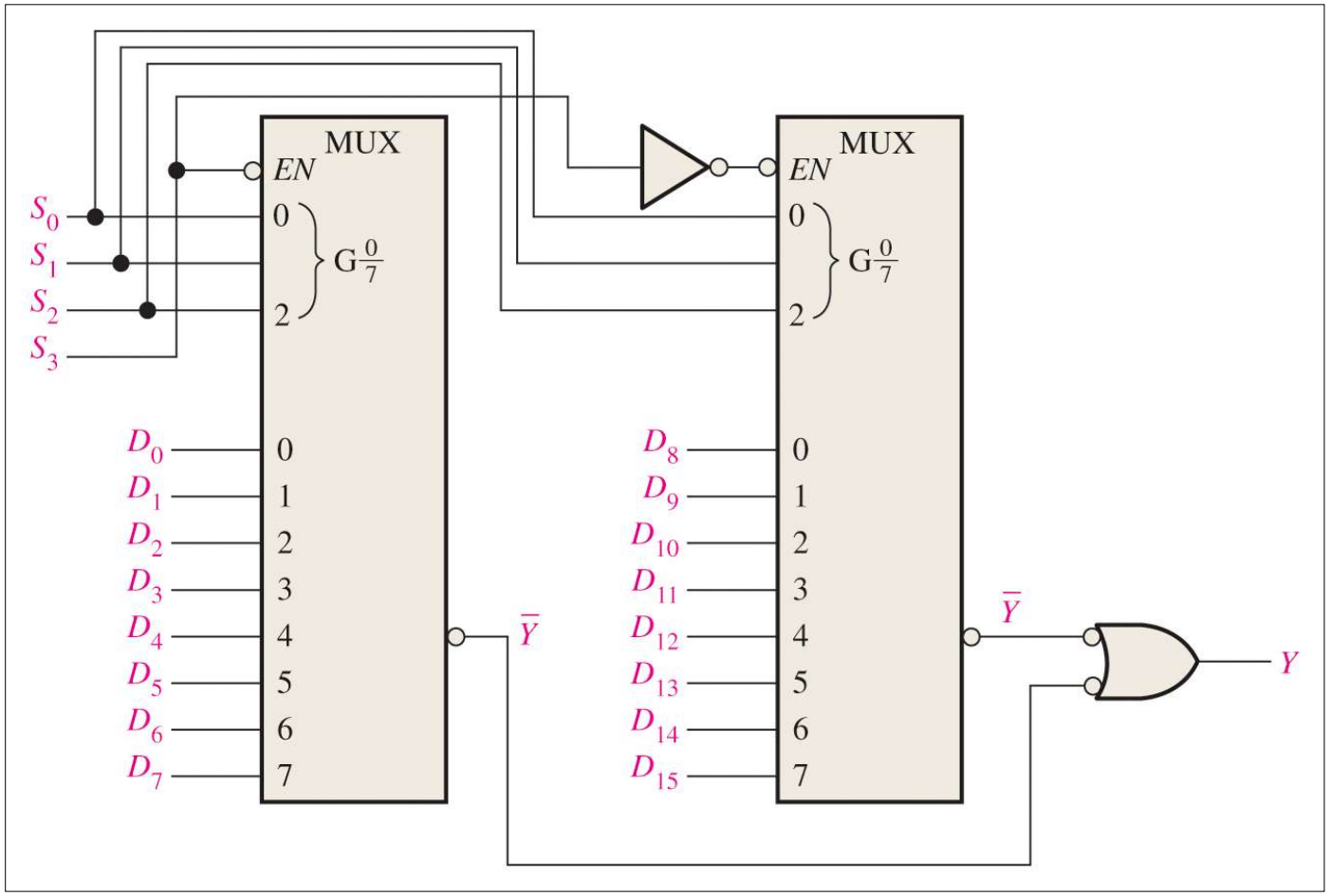
The select input (10) connects data line 2 to the output.

Multiplexers

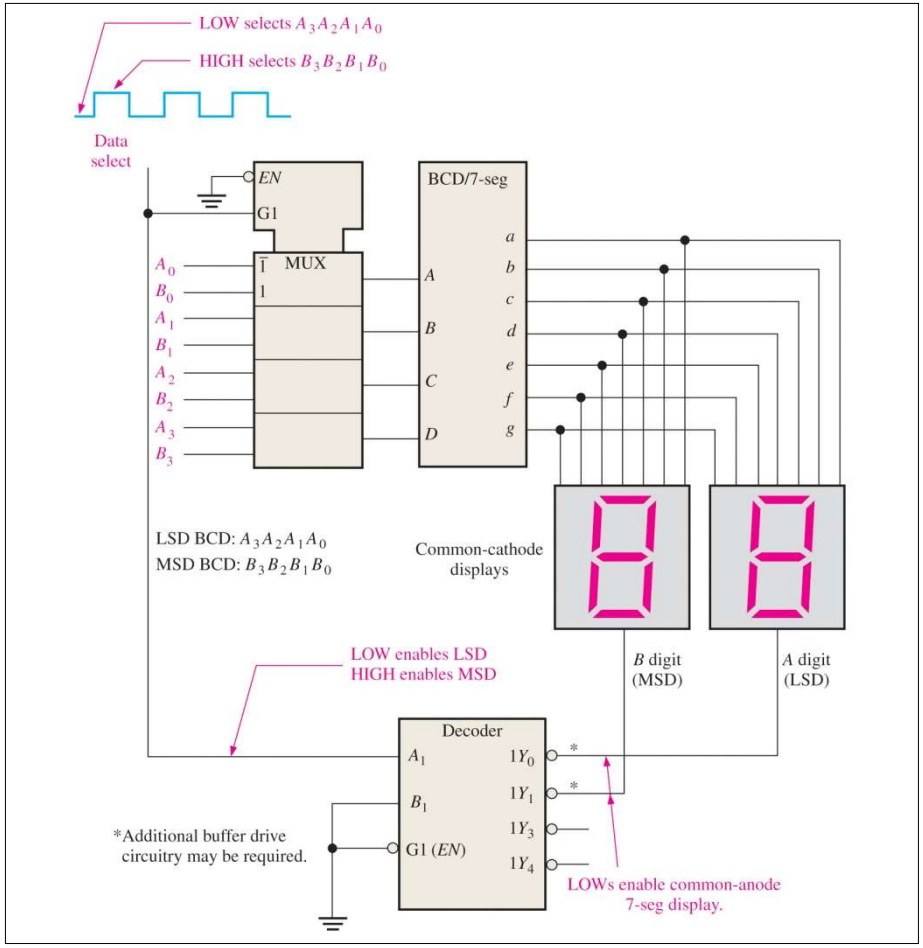
Here is the logic diagram for a 4-input multiplexer.



A 16-Bit Multiplexer



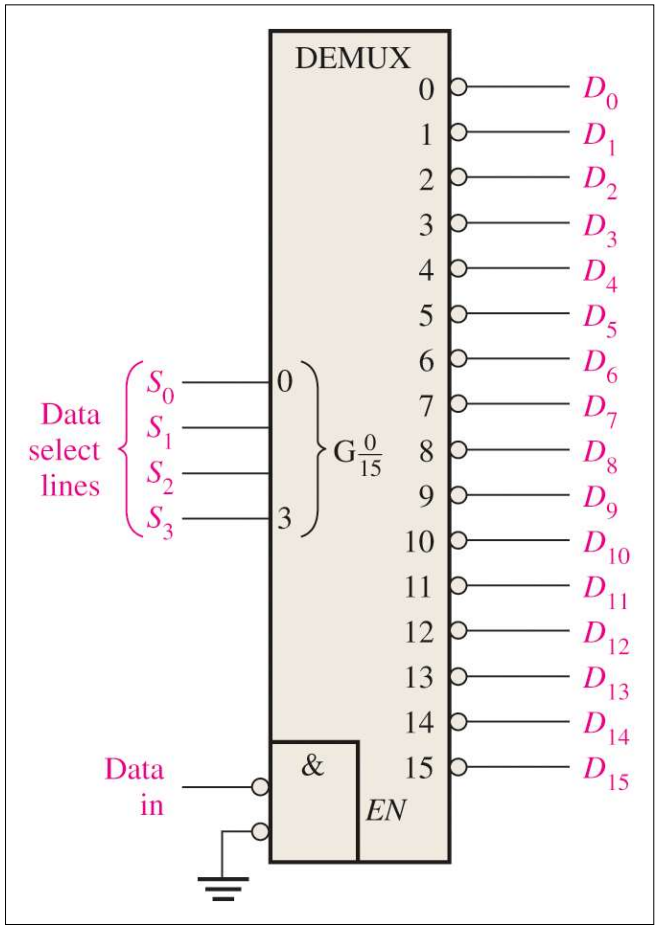
A 7-Segment Display Multiplexer



Demultiplexers

A demultiplexer (DEMUX) performs the opposite function from a MUX. It switches data from one input line to two or more data lines depending on the select inputs.

Data is applied to one of the data input pin, and routed to the selected output line depending on the select variables. Note that the outputs are active-LOW.



Parity Generators/Checkers

Parity is an error detection method that uses an extra bit appended to a group of bits to force them to be either odd or even. In even parity, the total number of ones is even; in odd parity the total number of ones is odd.

The ASCII letter S is 1010011. Show the parity bit for the letter S with odd and even parity.

S with odd parity = 11010011

S with even parity = 01010011

Parity Generators/Checkers

A 9-bit parity checker/generator can be used to generate a parity bit or to check an incoming data stream for even or odd parity.

Checker: The even output will normally be HIGH if the data lines have even parity; otherwise it will be LOW. Likewise, the odd output will normally be HIGH if the data lines have odd parity; otherwise it will be LOW.

Generator: To generate even parity, the parity bit is taken from the odd parity output. To generate odd parity, the output is taken from the even parity output.