CS221: Logic Design

Instructors:
Dr. Ahmed Shalaby http://bu.edu.eg/staff/ahmedshalaby14#
Dr. Fatma Sakr

Digital Fundamentals

CHAPTER 6
Functions of Combinational Logic

Combinational Logic - Basic Adders

Half-Adder

Simple Binary Addition

0 + 0 = 0 Zero plus zero equals zero

0 + 1 = 1 Zero plus one equals one

1 + 0 = 1 One plus zero equals one

1 + 1 = 10 One plus one equals zero with a carry of one

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

Combinational Logic - Basic Adders

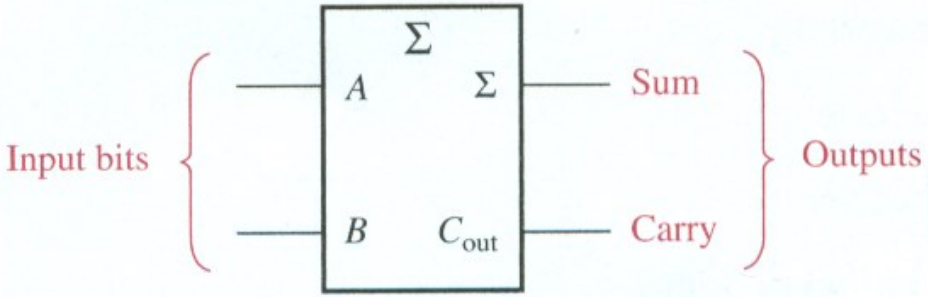
Half-Adder

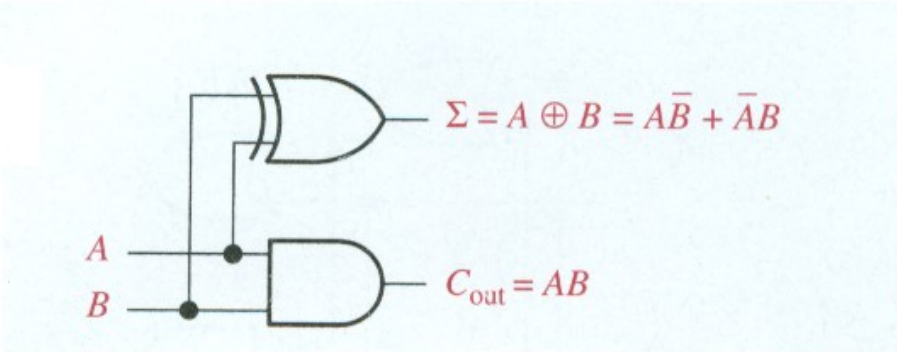
Basic rules of binary addition are performed by a **half adder**, which has two binary inputs (*A* and *B*) and two binary outputs (Carry out and Sum).

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)

Half-Adder Truth Table





Combinational Logic - Basic Adders

Full-Adder

By contrast, a **full adder** has three binary inputs (A, B, and Carry in) and two binary outputs (Carry out and Sum).

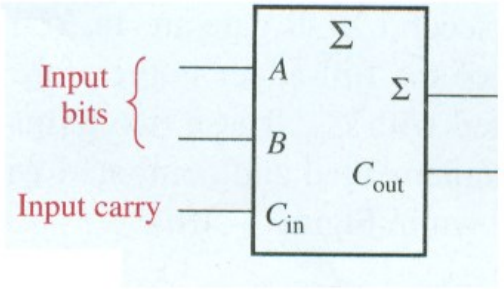
A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

C_{in} = input carry, sometimes designated as CI

C_{out} = output carry, sometimes designated as CO

Σ = sum

A and B = input variables (operands)



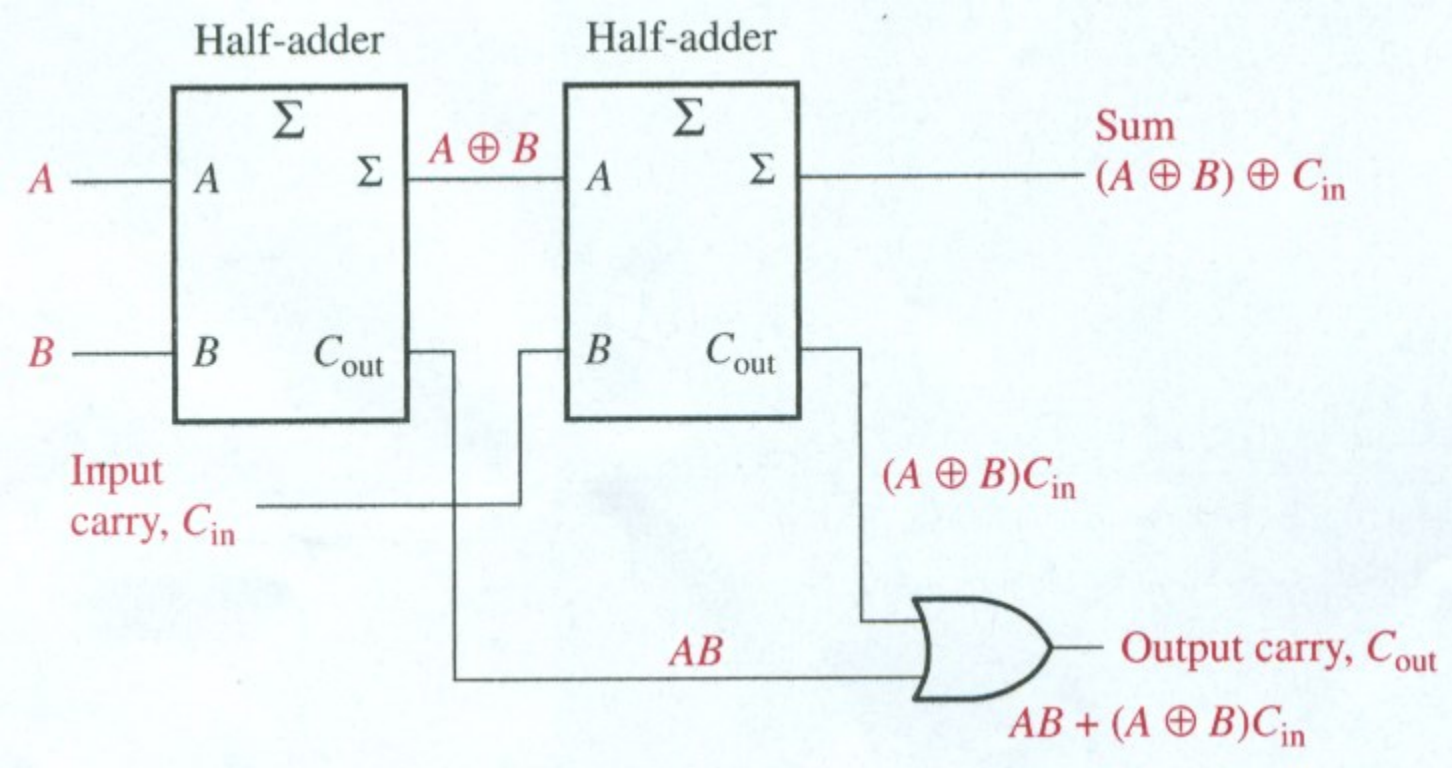
$$\Sigma = (A \oplus B) \oplus C_{in}$$

$$C_{out} = AB + (A \oplus B)C_{in}$$

Combinational Logic - Basic Adders

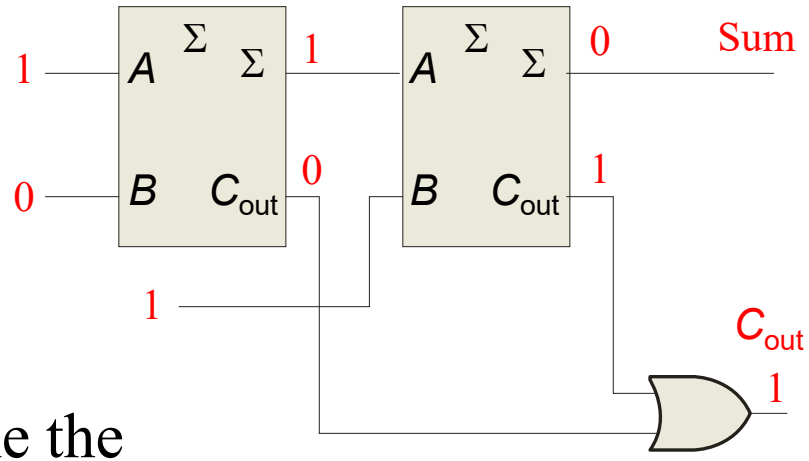
Full-Adder

A full-adder can be constructed from two half adders as shown:



Combinational Logic - Basic Adders

Full-Adder



Example

For the given inputs, determine the intermediate and final outputs of the full adder.

Solution

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

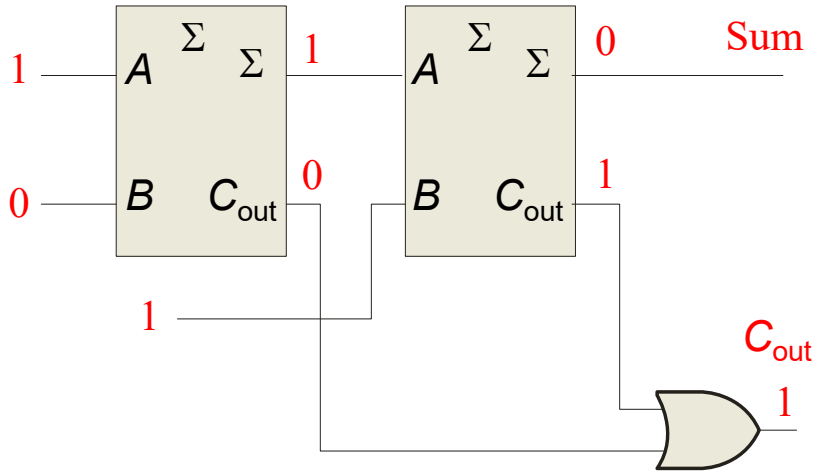
The second half-adder has inputs of 1 and 1; therefore the Sum = 0 and the Carry out = 1.

Combinational Logic - Basic Adders

Full-Adder

Notice that the result from the previous example can be read directly on the truth table for a full adder.

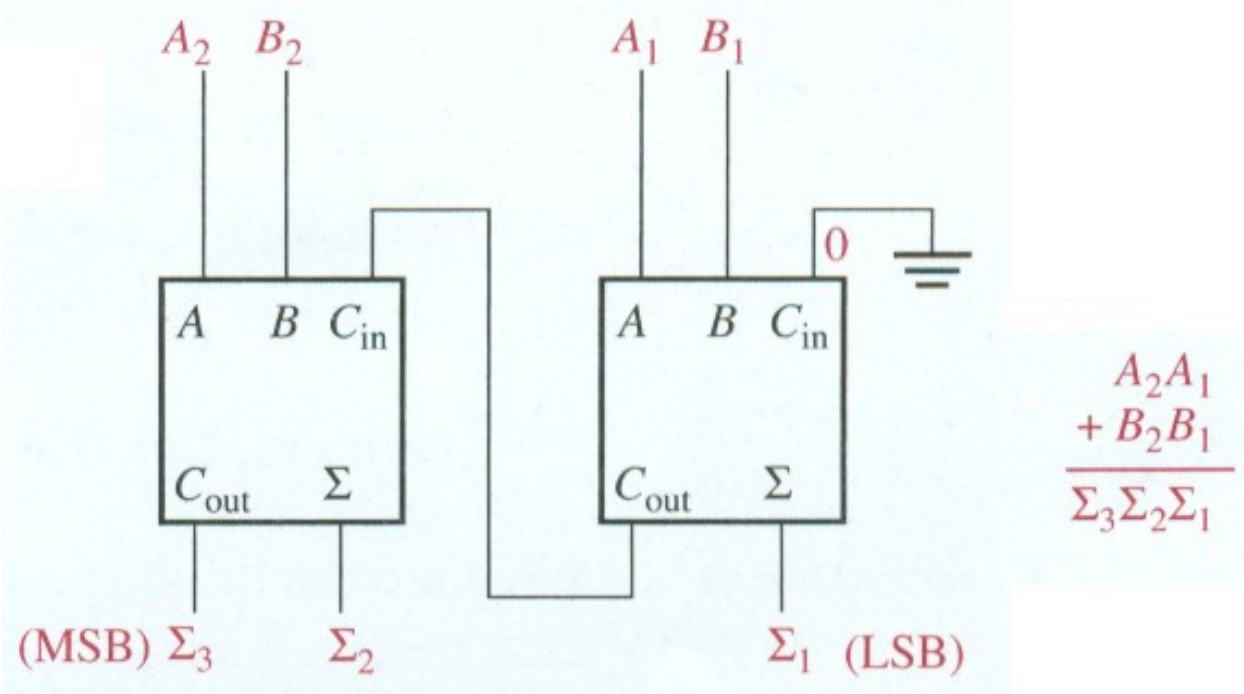
Inputs			Outputs	
A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Combinational Logic - Parallel Binary Adders

Parallel Adders

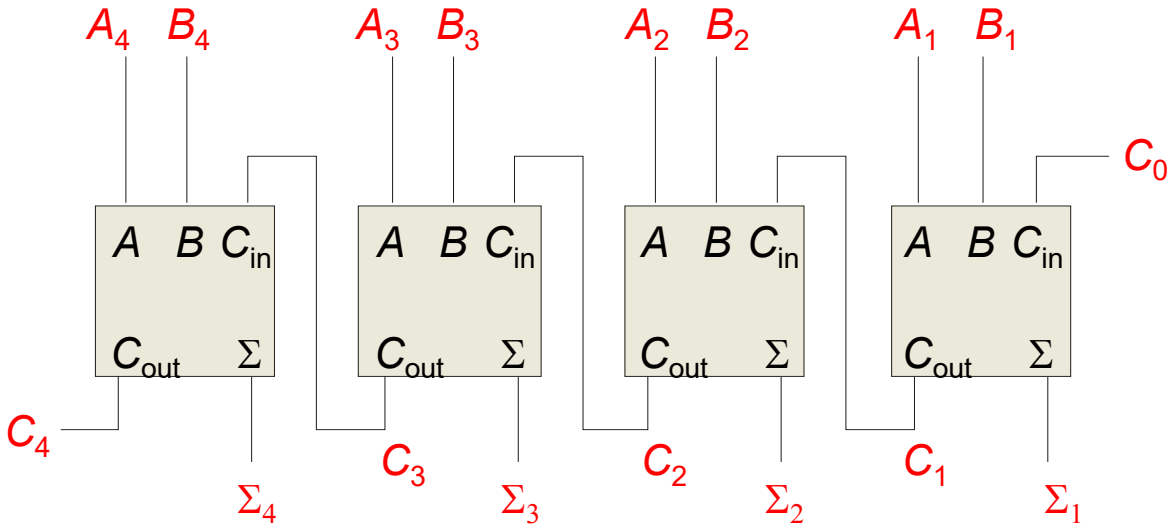
Full adders are combined into parallel adders that can add binary numbers with multiple bits. A 2-bit adder is shown.



Combinational Logic - Parallel Binary Adders

Parallel Adders

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

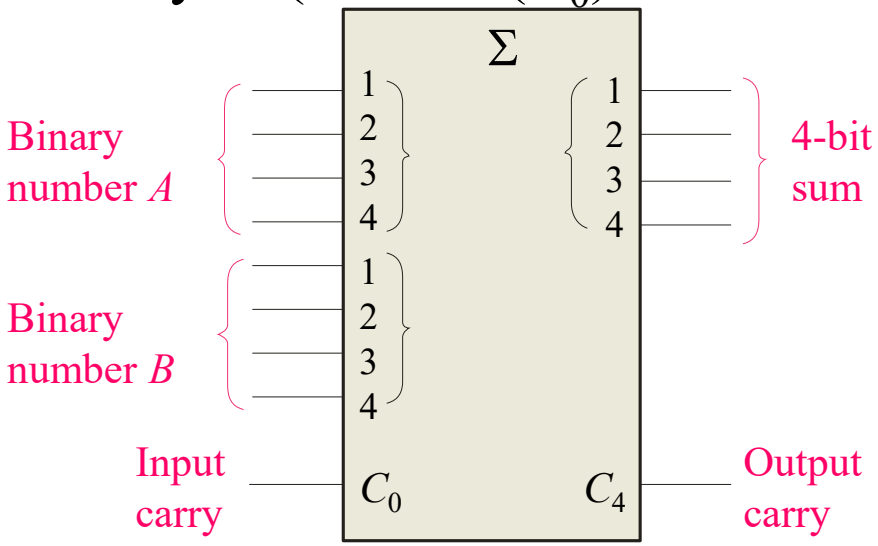


The output carry (C_4) is not ready until it propagates through all of the full adders. This is called *ripple carry*, delaying the addition process.

Combinational Logic - Parallel Binary Adders

Parallel Adders

The logic symbol for a 4-bit parallel adder is shown. This 4-bit adder includes a carry in (labeled (C_0) and a Carry out (labeled C_4)

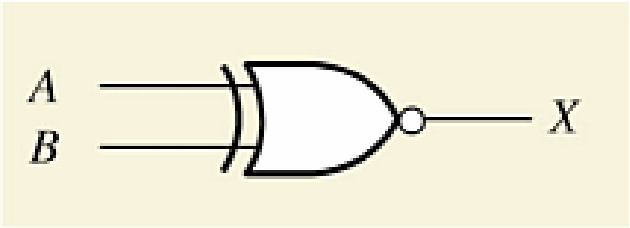


The 74LS283 is an example. It features *look-ahead carry*, which adds logic to minimize the output carry delay. For the 74LS283, the maximum delay to the output carry is 17 ns.

Combinational Logic - Comparators

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.



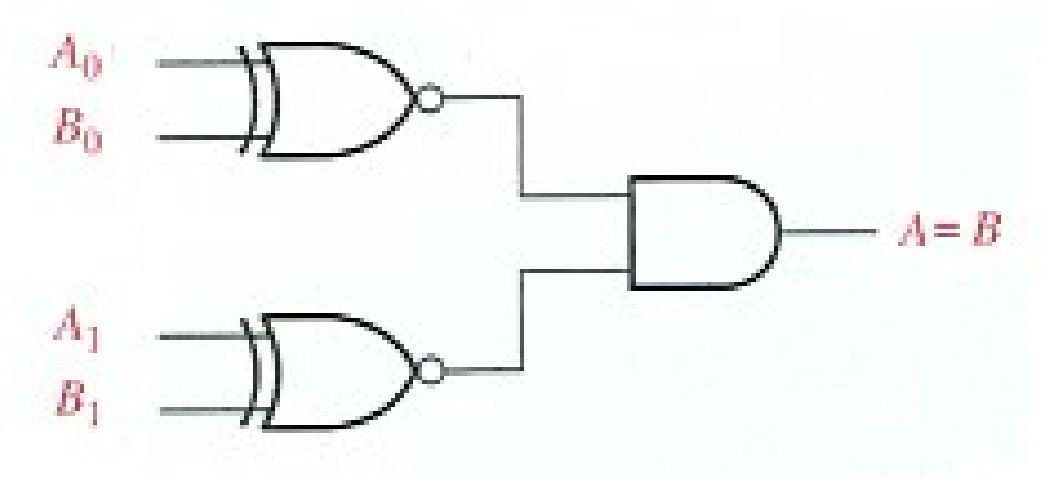
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

The output is 1 when the inputs are equal

Combinational Logic - Comparators

Comparators

2-bit Comparator



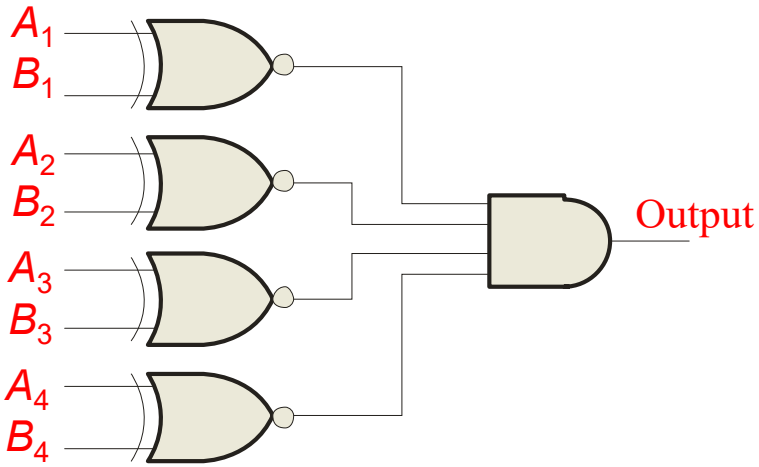
The output is 1 when $A_0 = B_0$ AND $A_1 = B_1$

Combinational Logic - Comparators

Comparators

Example How could you test two 4-bit numbers for equality?

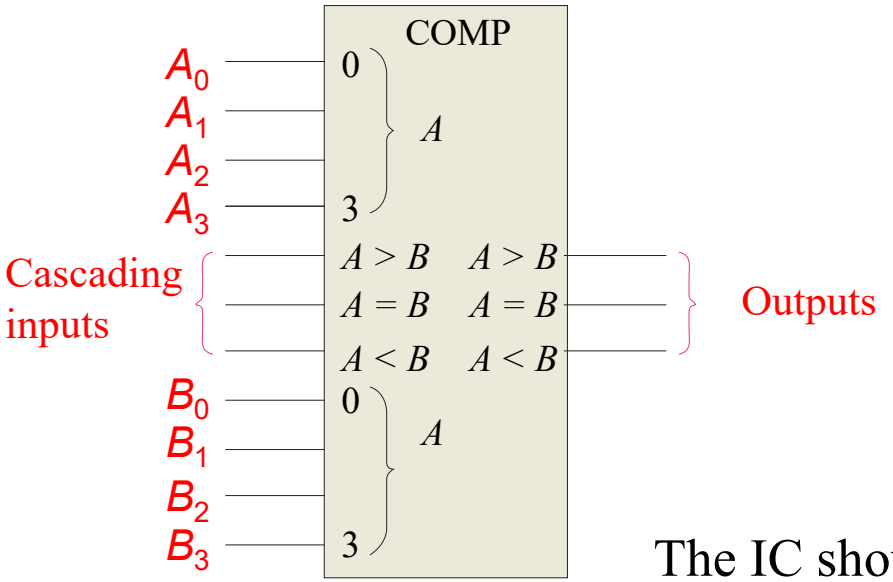
Solution AND the outputs of four XNOR gates.



Combinational Logic - Comparators

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.

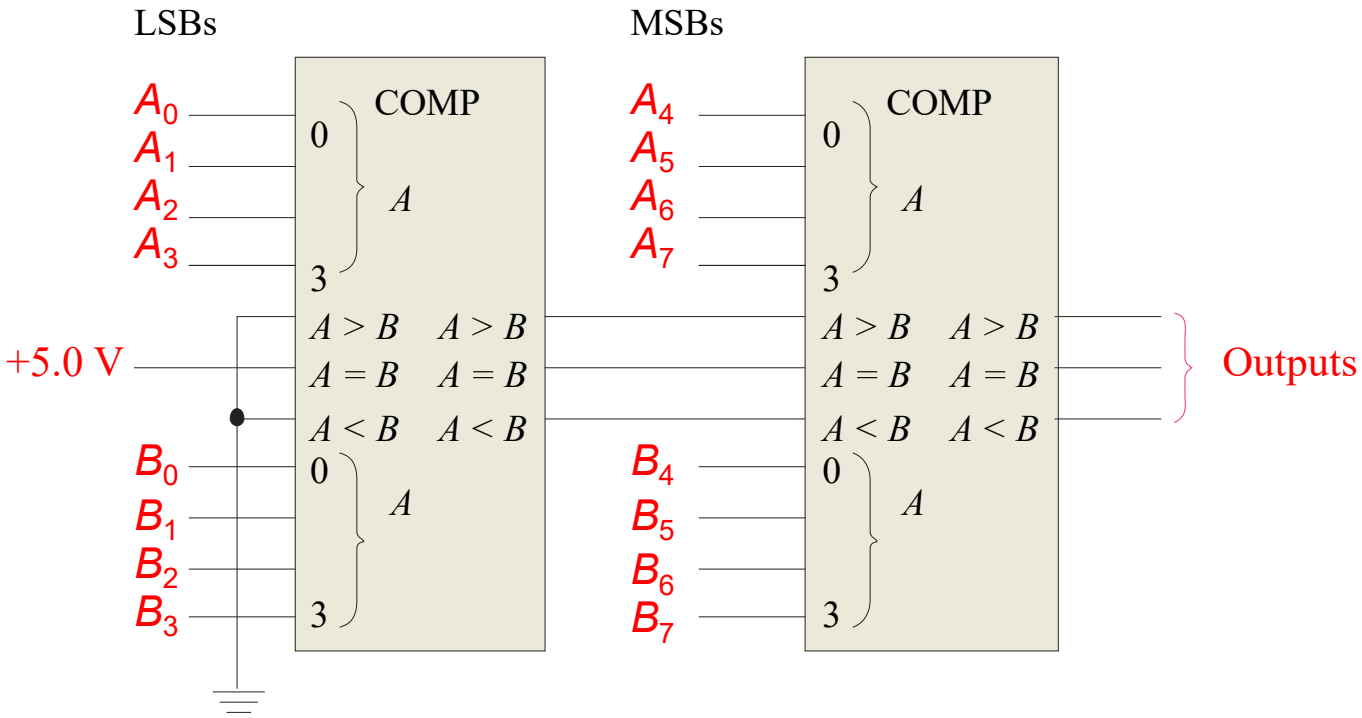


The IC shown is the 4-bit 74LS85.

Combinational Logic - Comparators

Comparators

IC comparators can be expanded using the cascading inputs as shown. The lowest order comparator has a HIGH on the $A = B$.

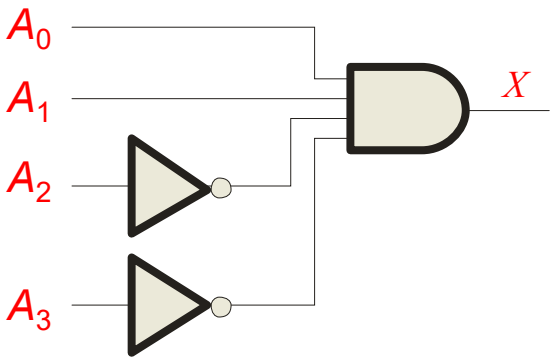


Combinational Logic - Decoders

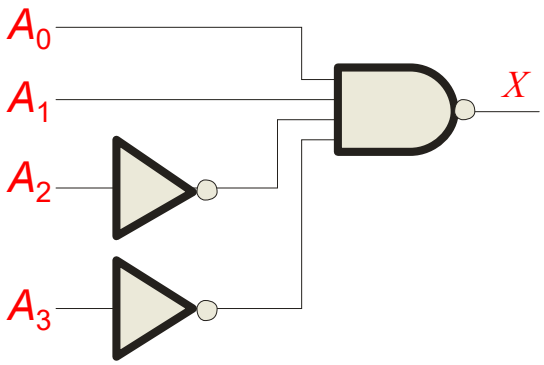
Decoders

A **decoder** is a logic circuit that detects the presence of a specific combination of bits at its input. Two simple decoders that detect the presence of the binary code 0011 are shown.

The first has an active HIGH output; the second has an active LOW output.



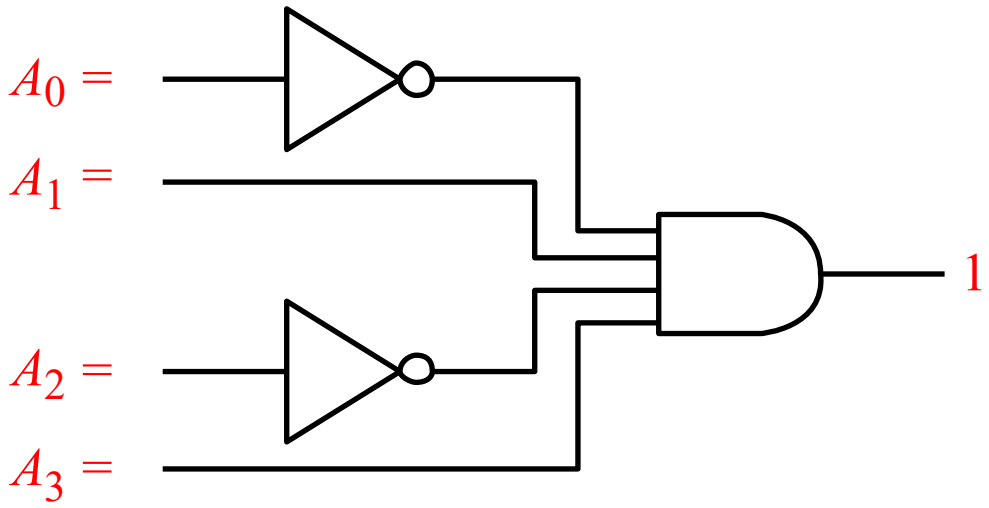
Active HIGH decoder for 0011



Active LOW decoder for 0011

QUIZ

Question Assume the output of the decoder shown is a logic 1. What are the inputs to the decoder?

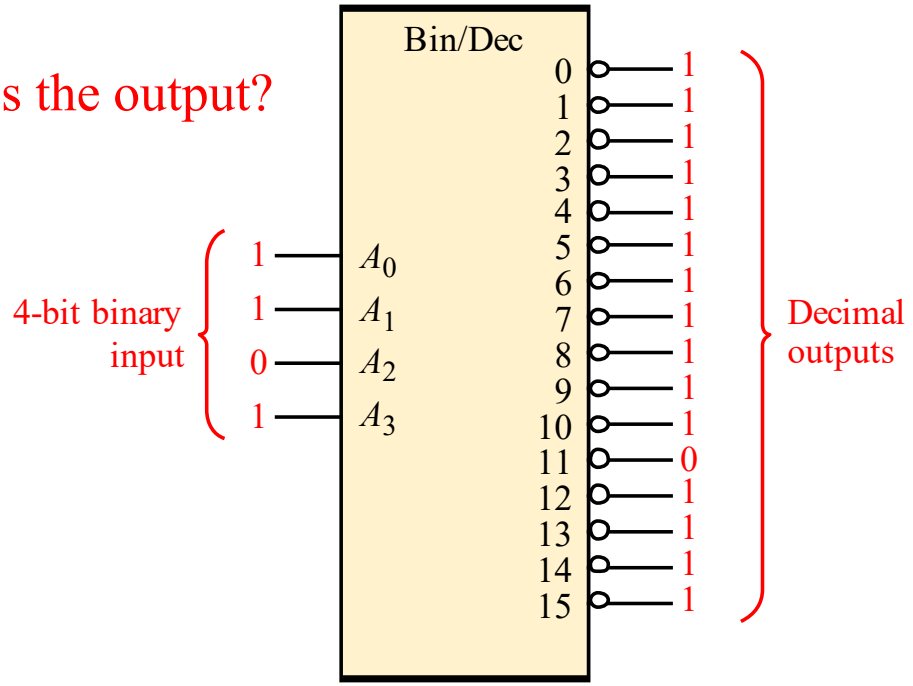


Combinational Logic - Decoders

Decoders

IC decoders have multiple outputs to decode any combination of inputs. For example the binary-to-decimal decoder shown here has 16 outputs – one for each combination of binary inputs.

For the input shown, what is the output?



Combinational Logic - Decoders

Decoders

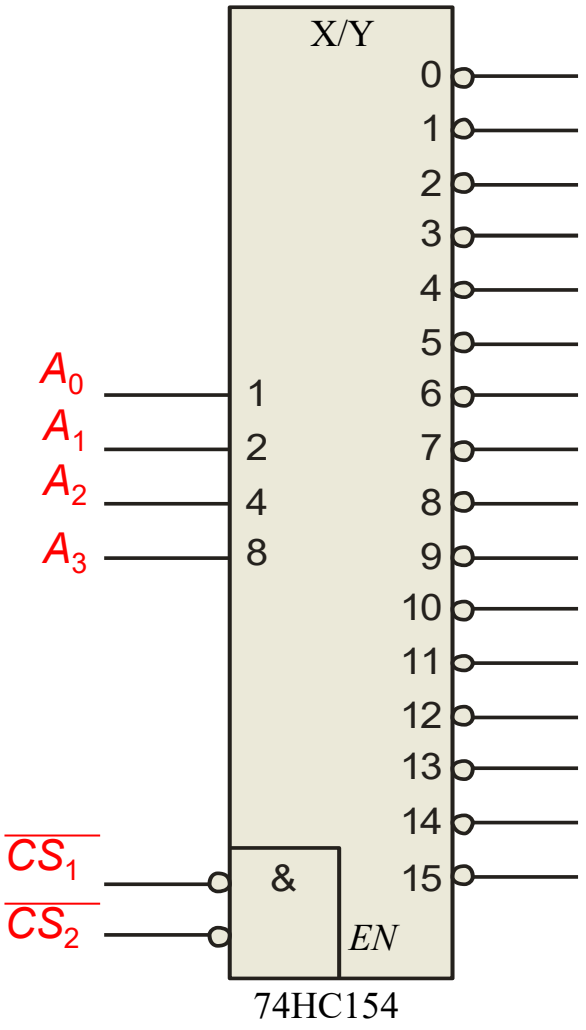
BINARY INPUTS				DECODING	OUTPUTS															
A_3	A_2	A_1	A_0	FUNCTION	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	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
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
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
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
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
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
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
1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
1	0	1	0	$A_3\overline{A}_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	0	1	1	$A_3\overline{A}_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	0	0	$A_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	0	1	$A_3A_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	1	1	0	$A_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	1	1	1	$A_3A_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Combinational Logic - Decoders

Decoders

A specific integrated circuit decoder is the 74HC154 (shown as a 4-to-16 decoder).

It includes two active LOW chip select lines which must be at the active level to enable the outputs.

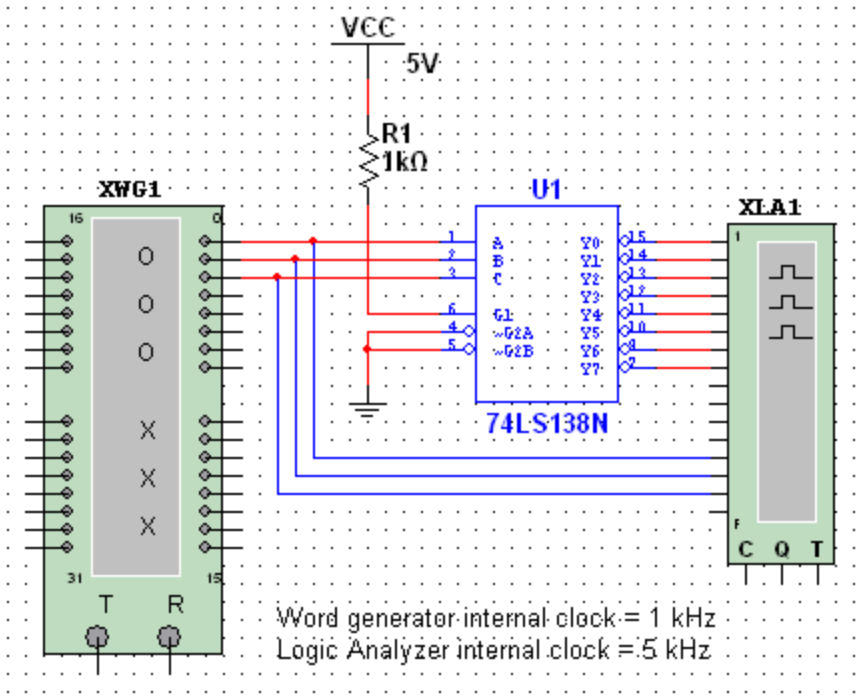


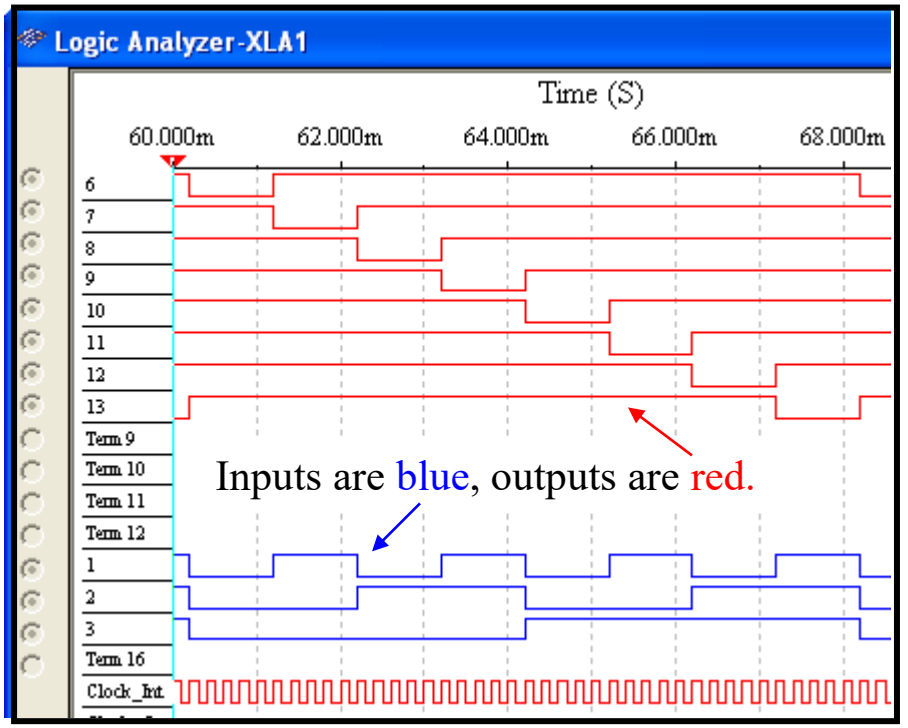
Combinational Logic - Decoders

Decoders

The 74LS138 is a 3-to-8 decoder with three chip select inputs (two active LOW, one active HIGH).

In this Multisim circuit, the word generator (XWG1) is set up as an up counter. The logic analyzer (XLA1) compares the input and outputs of the decoder.



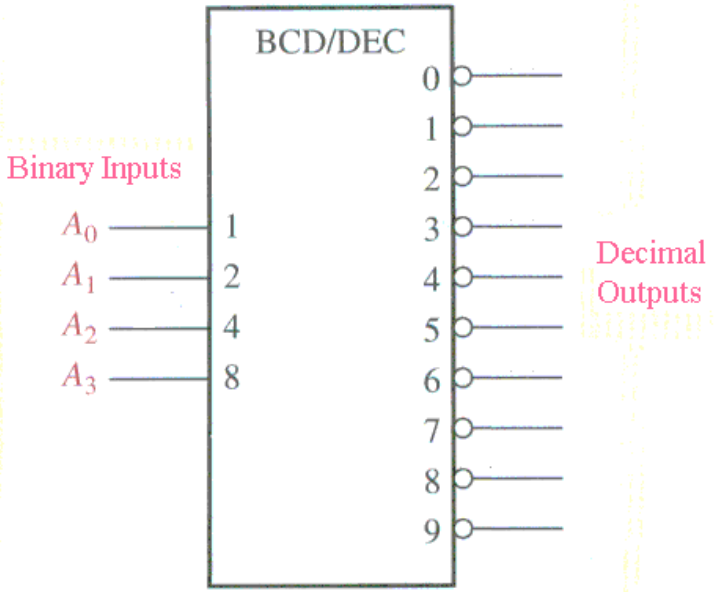


Combinational Logic - Decoders

Decoders

BCD-to-decimal decoders accept a binary coded decimal input and activate one of ten possible decimal digit indications.

DECIMAL DIGIT	BCD CODE A_3	A_2	A_1	A_0	DECODING FUNCTION
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$

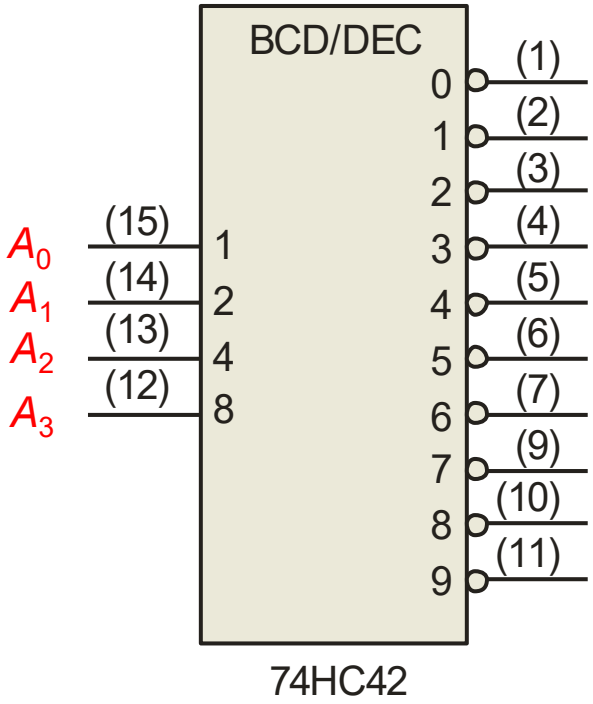


Combinational Logic - Decoders

Decoders

Example

Assume the inputs to the 74HC42 decoder are the sequence 0101, 0110, 0011, and 0010. Describe the output.



Solution

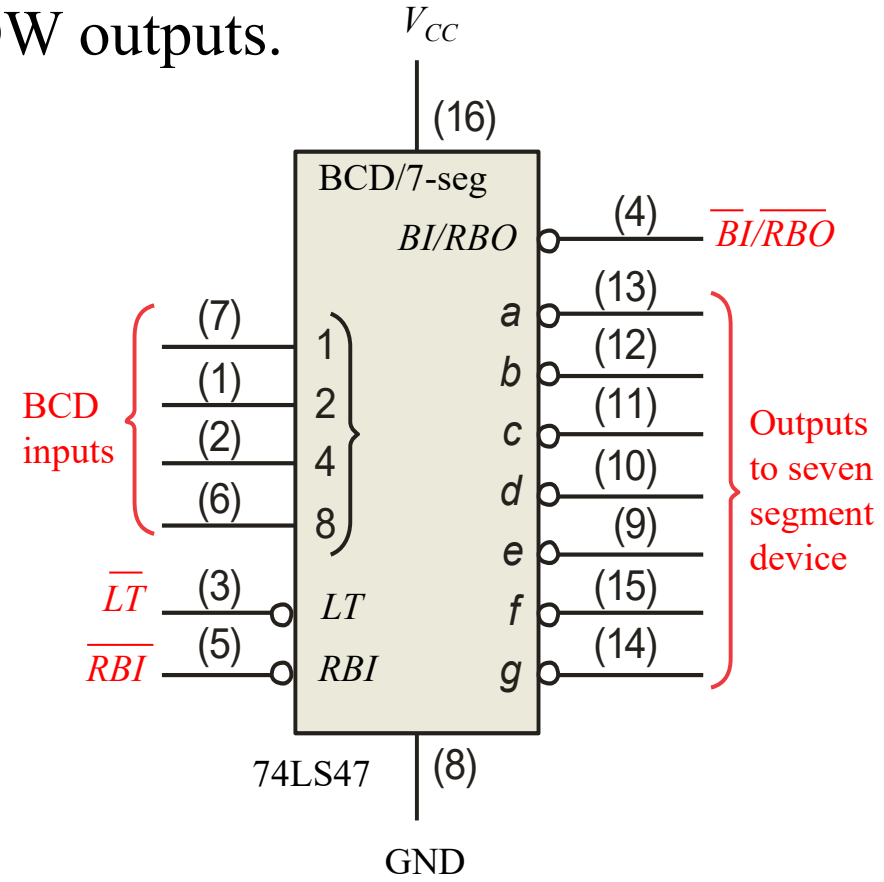
All lines are HIGH except for one active output, which is LOW. The active outputs are 5, 6, 3, and 2 in that order.

Combinational Logic - Decoders

BCD Decoder/Driver

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

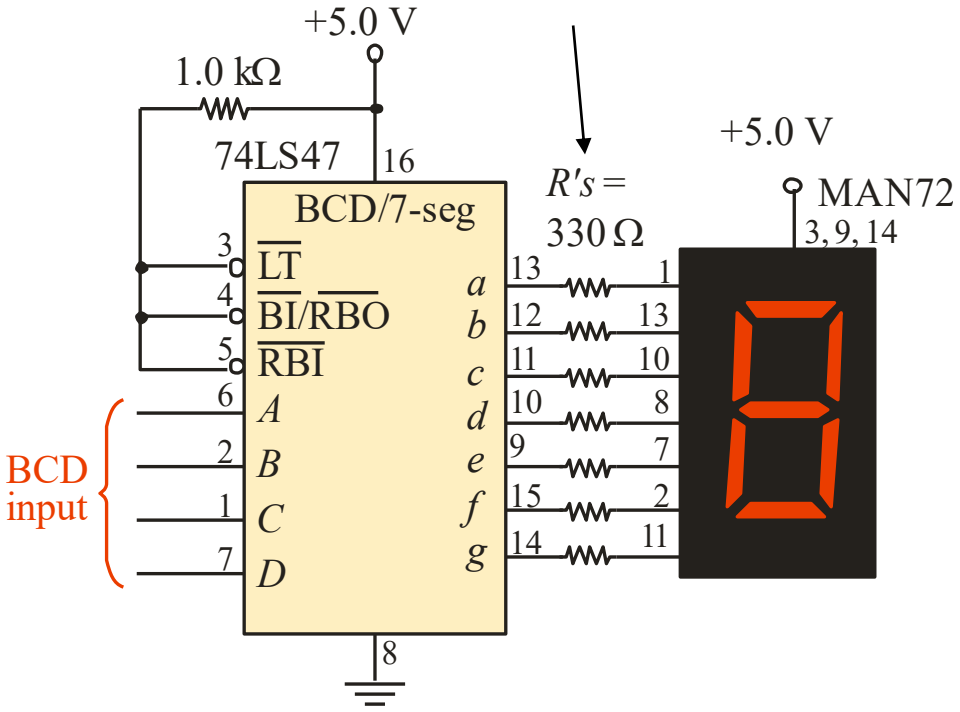
The *a*-*g* outputs are designed for much higher current than most devices (hence the word driver in the name).



Combinational Logic - Decoders

BCD Decoder/Driver

Here the 7447A is an connected to an LED seven segment display. Notice the current limiting resistors, required to prevent overdriving the LED display.



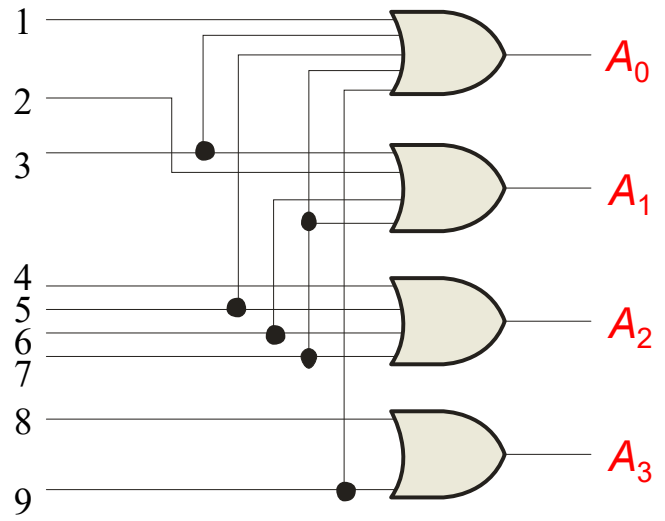
Combinational Logic - Encoders

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.



Combinational Logic - Encoders

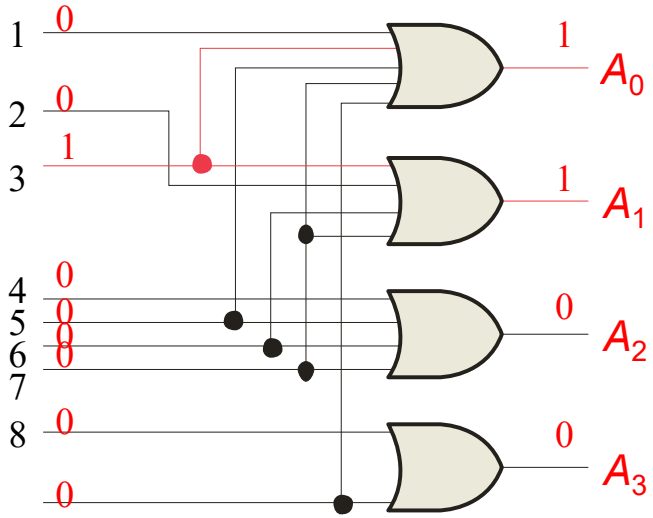
Encoders

Example

Show how the decimal-to-BCD encoder converts the decimal number 3 into a BCD 0011.

Solution

The top two OR gates have ones as indicated with the red lines. Thus the output is 0111.



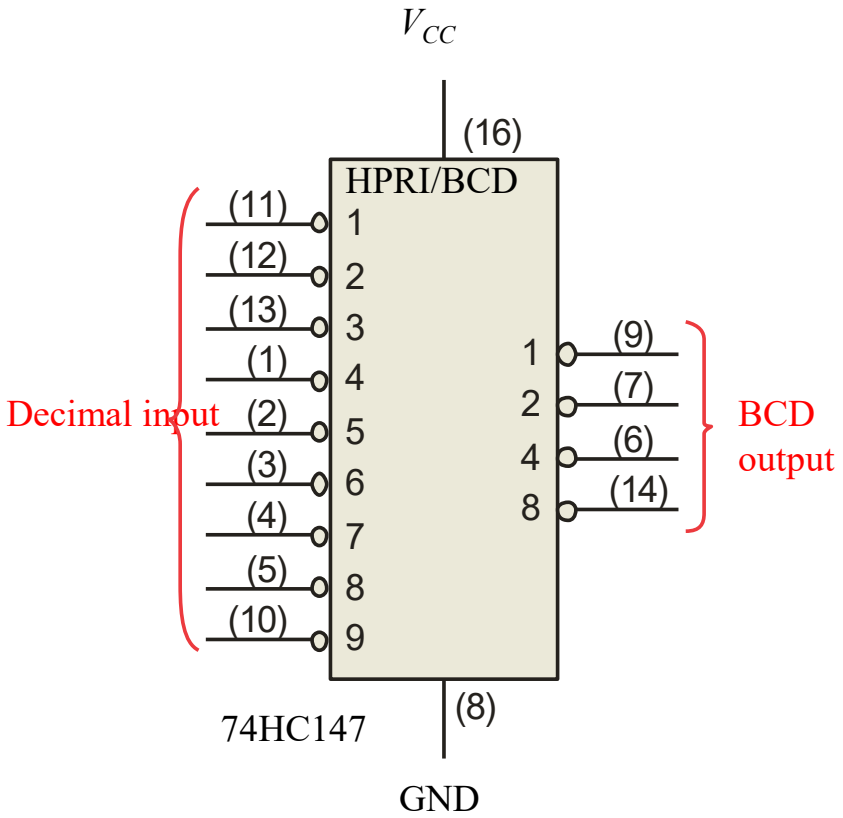
Combinational Logic - Encoders

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 offers additional flexibility in that it is a **priority encoder**.

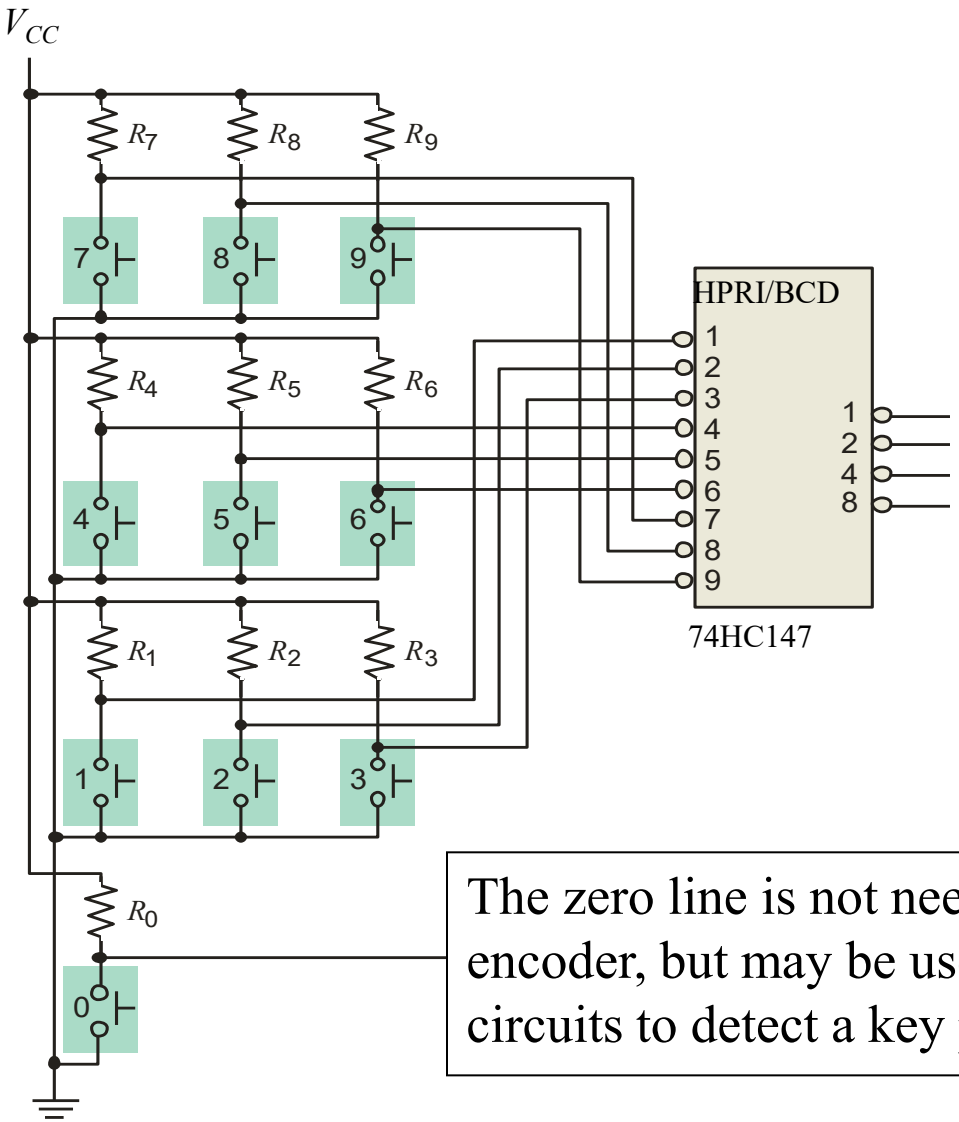
This means that if more than one input is active, the one with the highest order decimal digit will be active.



Combinational Logic - Encoders

Encoders

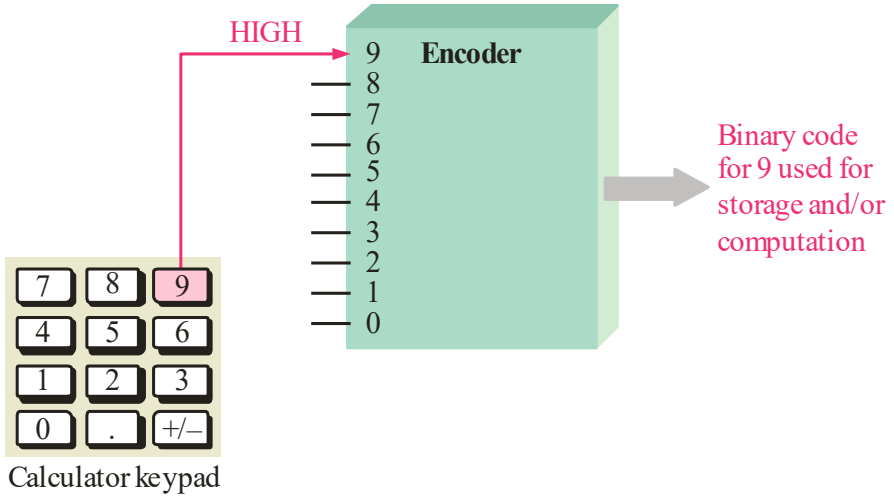
Keyboard encoder



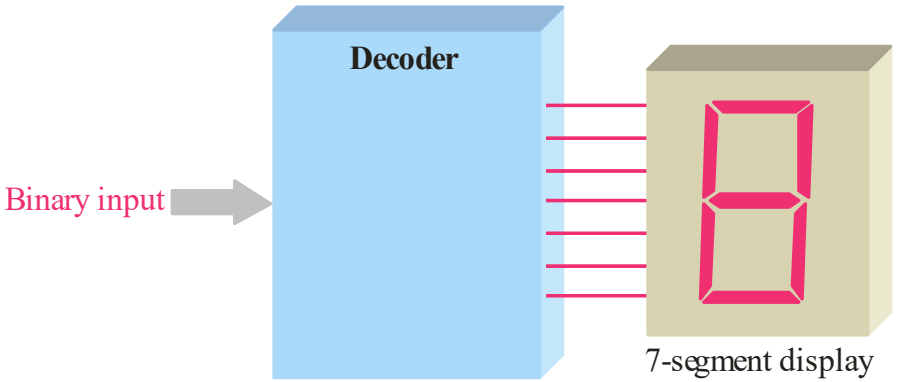
The zero line is not needed by the encoder, but may be used by other circuits to detect a key press.

Combinational Logic - Decoder/Encoders

Keypad Encoder



Seven Segment Decoder



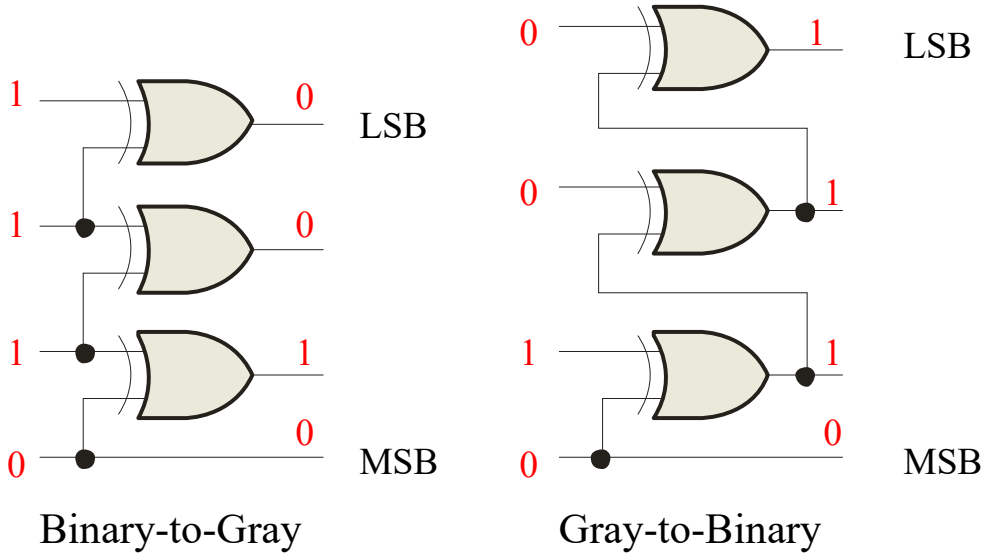
Combinational Logic – Code Converter

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.

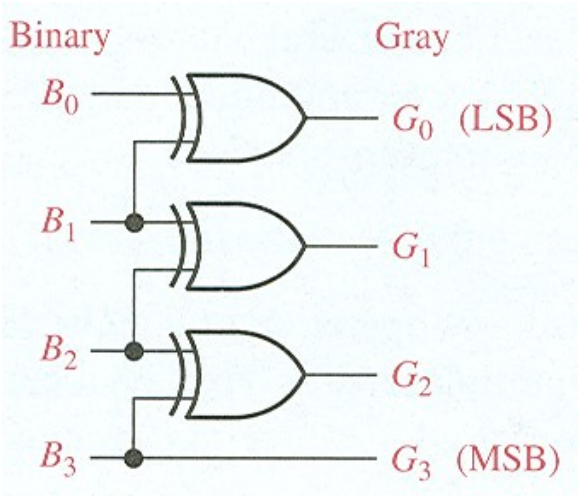
Example Show the conversion of binary 0111 to Gray and back.

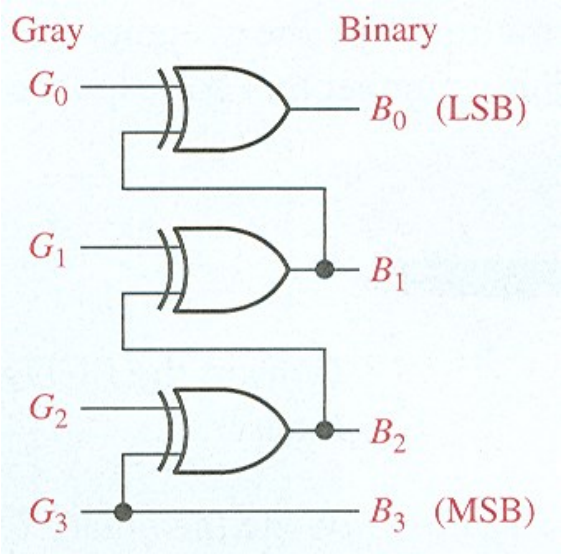
Solution



Combinational Logic – Code Converter

Code converters





BINARY	GRAY CODE
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Combinational Logic – Multiplexer

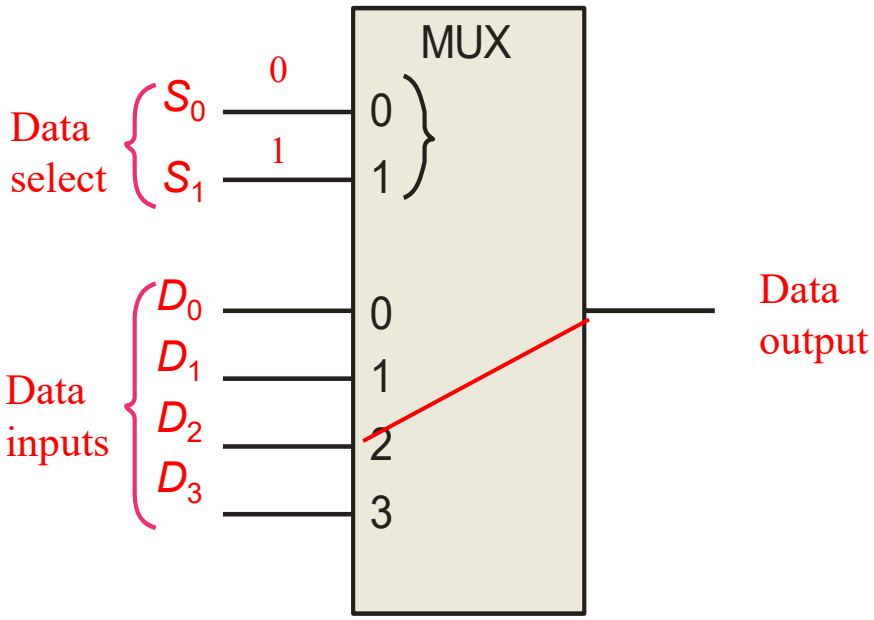
Multiplexers

A multiplexer (MUX) selects one data line from two or more input lines and routes data from the selected line to the output. The particular data line that is selected is determined by the select inputs.

Question

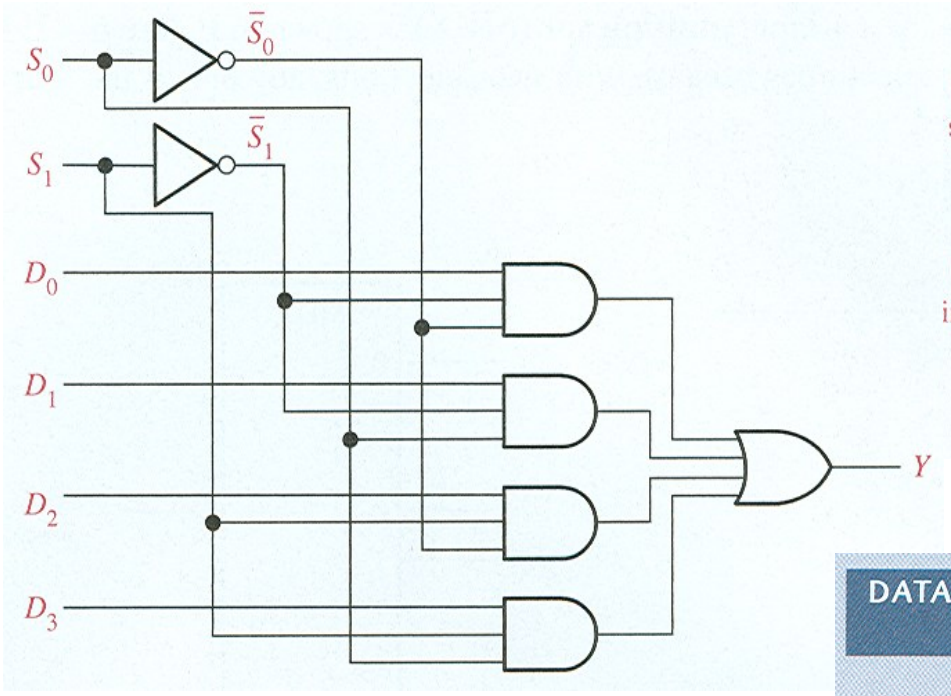
Which data line is selected if $S_1S_0 = 10$?

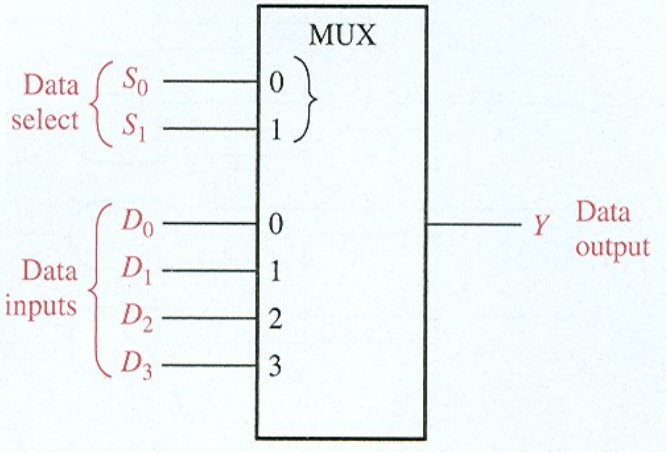
D_2



Combinational Logic – Multiplexer

Multiplexers





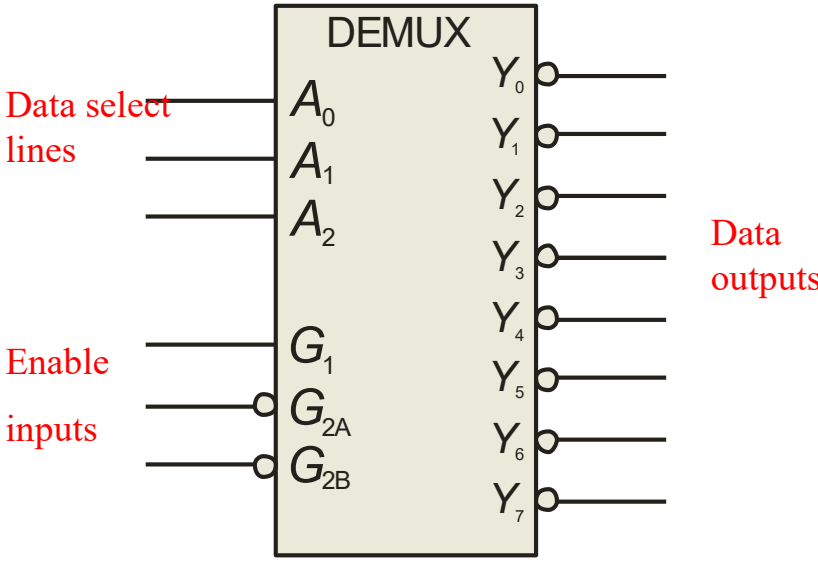
DATA-SELECT INPUTS S_1	S_0	INPUT SELECTED
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Combinational Logic – 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.

The 74LS138 was introduced previously as a decoder but can also serve as a DEMUX. When connected as a DEMUX, data is applied to one of the enable inputs, and routed to the selected output line depending on the select variables. Note that the outputs are active-LOW as illustrated in the following example…

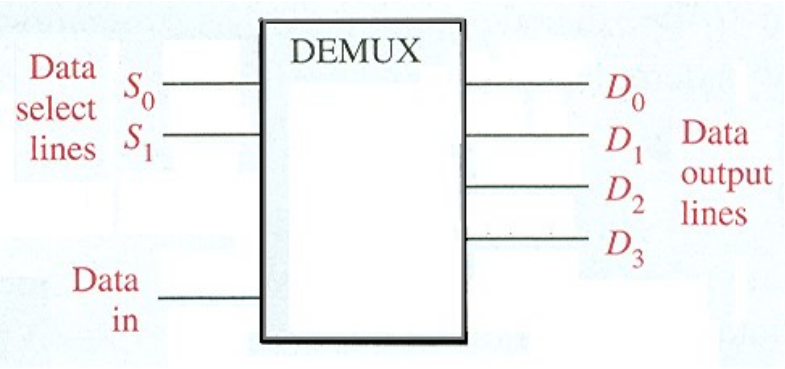


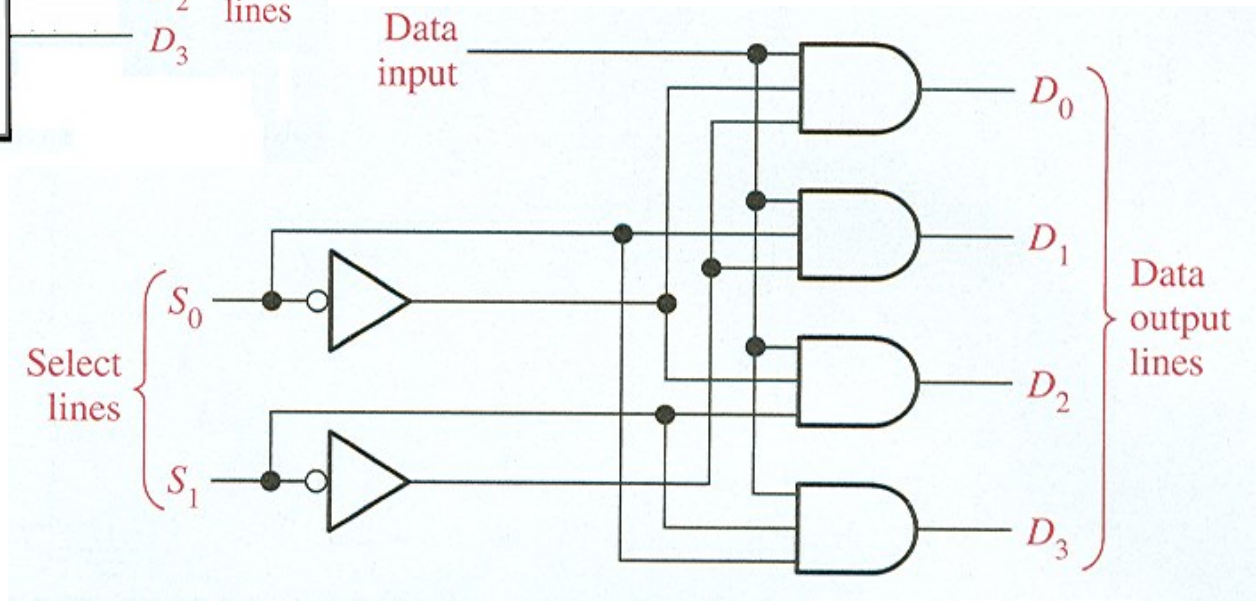
74LS138

Combinational Logic – DeMultiplexer

Demultiplexers

DATA-SELECT INPUTS S_1	S_0	OUTPUT SELECTED
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3





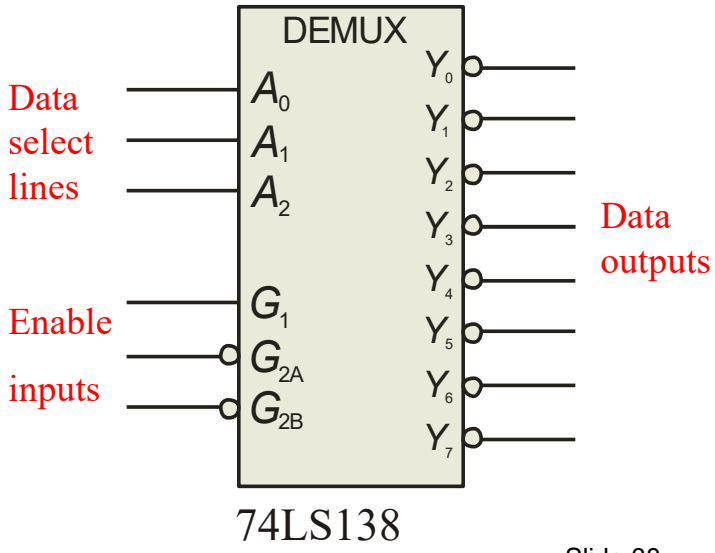
Combinational Logic – DeMultiplexer

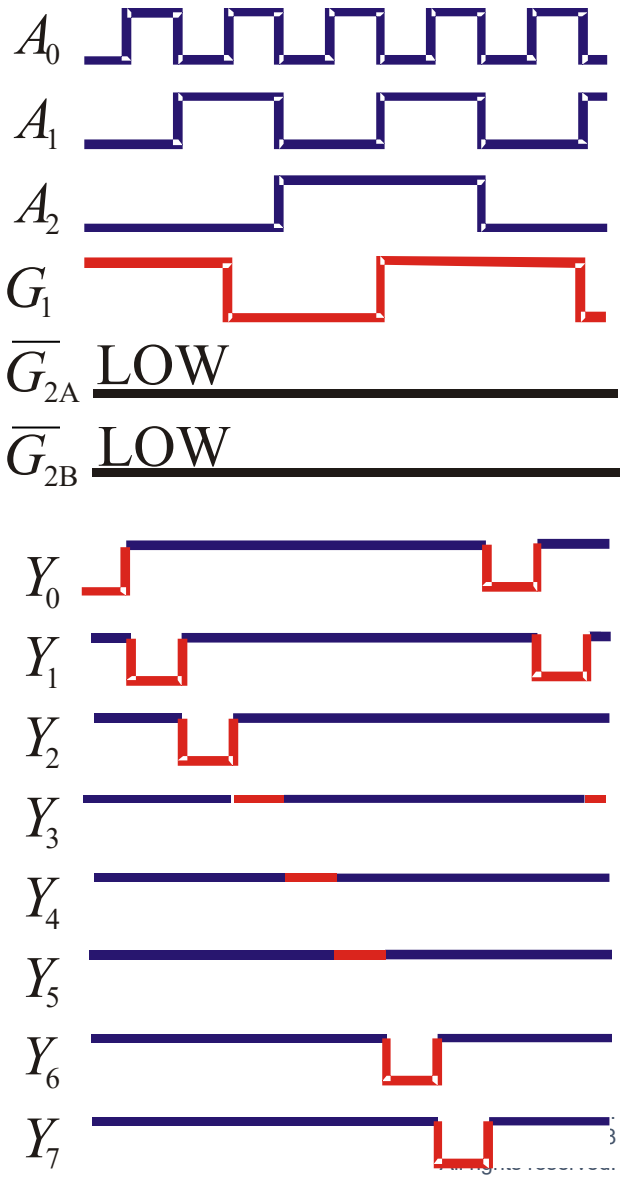
Demultiplexers

Example Determine the outputs, given the inputs shown.

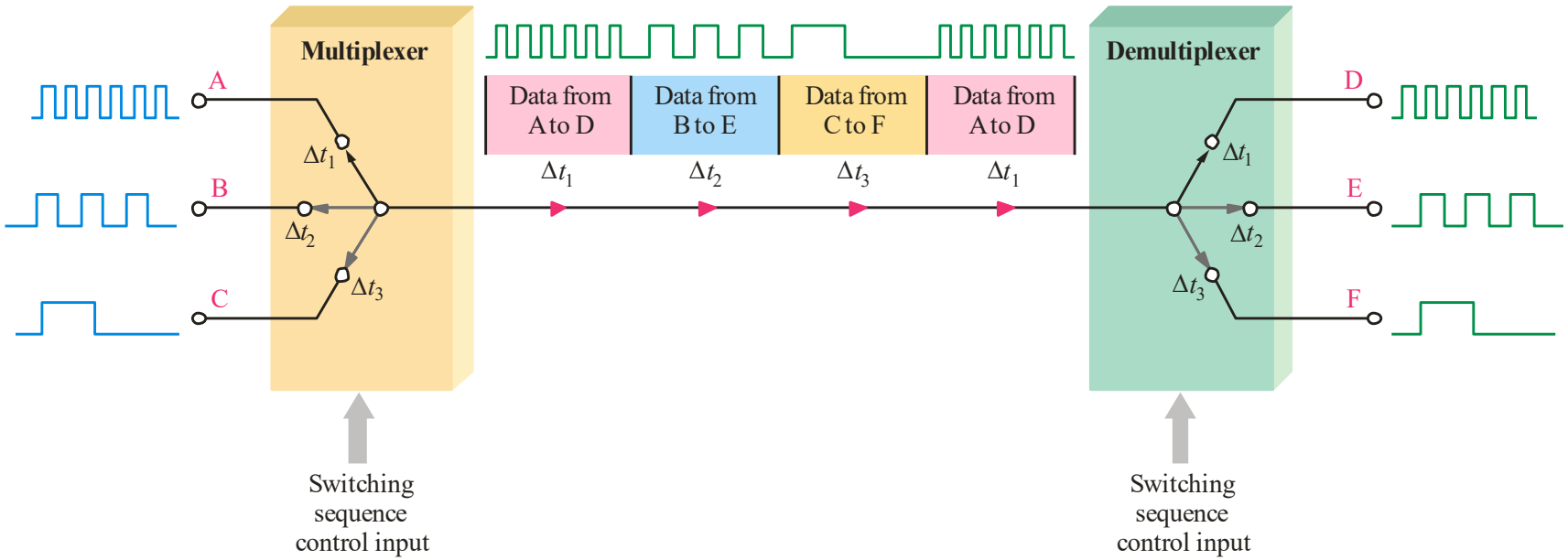
Solution

The output logic is opposite to the input because of the active-LOW convention. (Red shows the selected line).





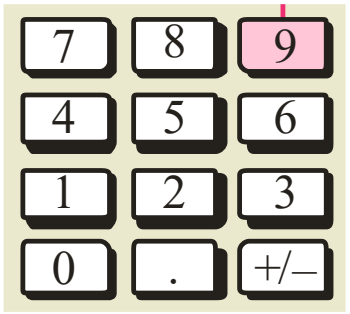
Combinational Logic – Multiplexer/DeMultiplexer



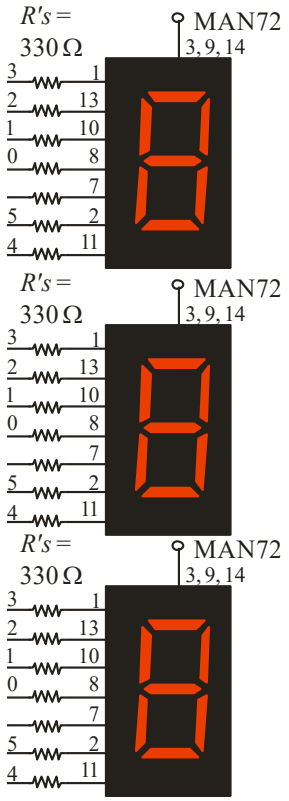
QUIZ

Question

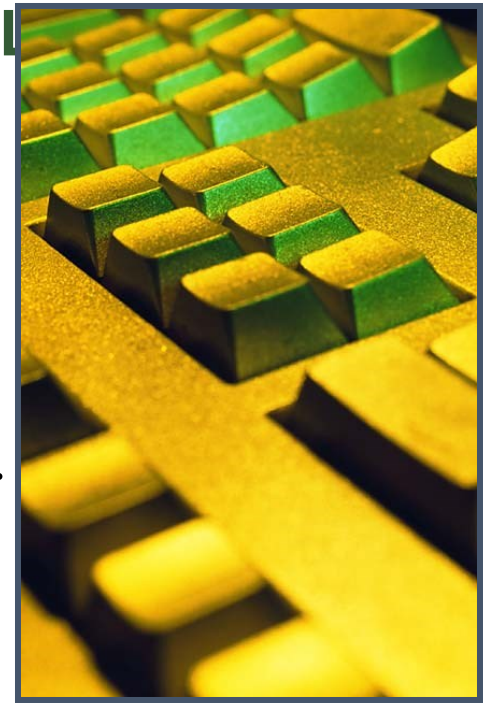
Connect a keypad to three seven-segment displays



Calculator keypad



Functions of Combinational L



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.

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

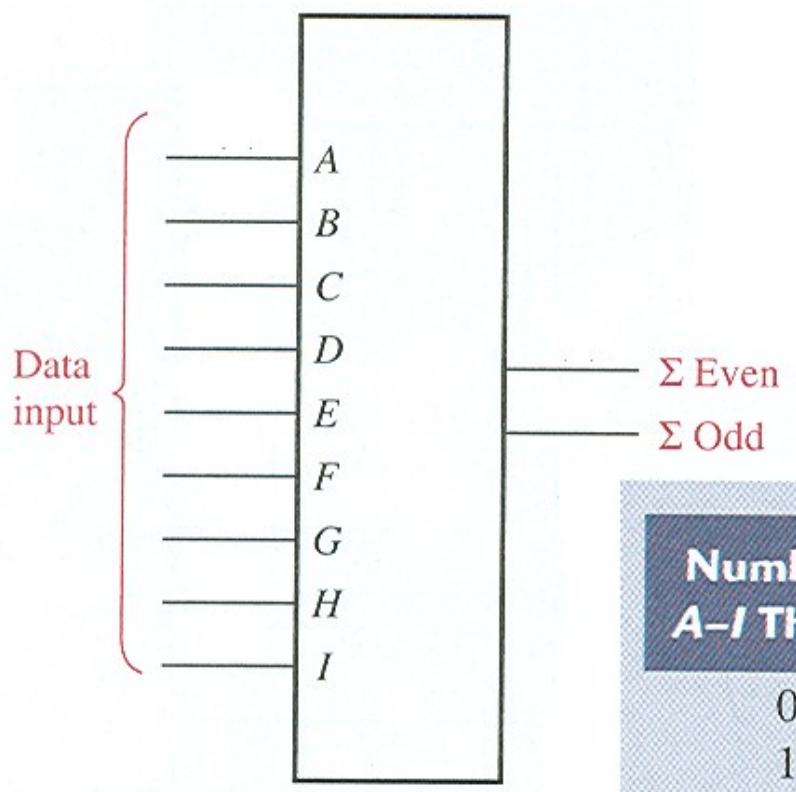
Solution

S with odd parity = 11010011

S with even parity = 01010011

Functions of Combinational Logic

Parity Generators/Checkers



Number of Inputs A–I That Are HIGH	Outputs Σ Even	Σ Odd
0, 2, 4, 6, 8	H	L
1, 3, 5, 7, 9	L	H

Functions of Combinational Logic

Parity Generators/Checkers

The 74LS280 can be used to generate a parity bit or to check an incoming data stream for even or odd parity.

Checker: The 74LS280 can test codes with up to 9 bits. 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.