EN1014 Electronic Engineering

Combinational Logic Building Blocks

Chamira U. S. Edussooriya
B.Sc.Eng. (Moratuwa), M.A.Sc. (UVic), Ph.D. (UVic)

Department of Electronic and Telecommunication Engineering
University of Moratuwa

Some of the tables and figures included in this presentation have been extracted from
Digital Design: With an Introduction to the Verilog HDL (M. M. Mano and M. D. Ciletti, Prentice Hall, 2012)
and
Digital Systems: Principles and Applications (R. J. Tocci, N. S.Widmer and G. L. Moss, Prentice Hall, 2007)

Introduction

- There are several combinational circuits, classified as standard components, that are employed extensively in the design of digital systems.

Introduction

- There are several combinational circuits, classified as standard components, that are employed extensively in the design of digital systems.
- In this lecture, we learn the most important standard components:
  - adders
  - multipliers
  - comparators
  - encoders and decoders
  - multiplexers and demultiplexers

Introduction

- There are several combinational circuits, classified as standard components, that are employed extensively in the design of digital systems.
- In this lecture, we learn the most important standard components:
  - adders
  - multipliers
  - comparators
  - encoders and decoders
  - multiplexers and demultiplexers
- These standard components are available in integrated circuits as medium-scale integration (MSI) circuits.
- They are also used as standard cells in complex very large-scale integration (VLSI) circuits.

Binary Adder-Subtractor

- A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers.

Binary Adder-Subtractor

- A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers.
- We first consider the design of a half adder, which performs the addition of two bits.

Binary Adder-Subtractor

- A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers.
- We first consider the design of a half adder, which performs the addition of two bits.
- Next, we design a full adder, which performs the addition of three bits (two significant bits and a previous carry).

Binary Adder-Subtractor

- A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers.
- We first consider the design of a half adder, which performs the addition of two bits.
- Next, we design a full adder, which performs the addition of three bits (two significant bits and a previous carry).
- An n-bit binary adder can be constructed by cascading n full adders, with the output carry from each full adder connected to the input carry of the next full adder.

Binary Adder-Subtractor *cont'd*

- A four-bit binary ripple carry (or carry-ripple) adder.

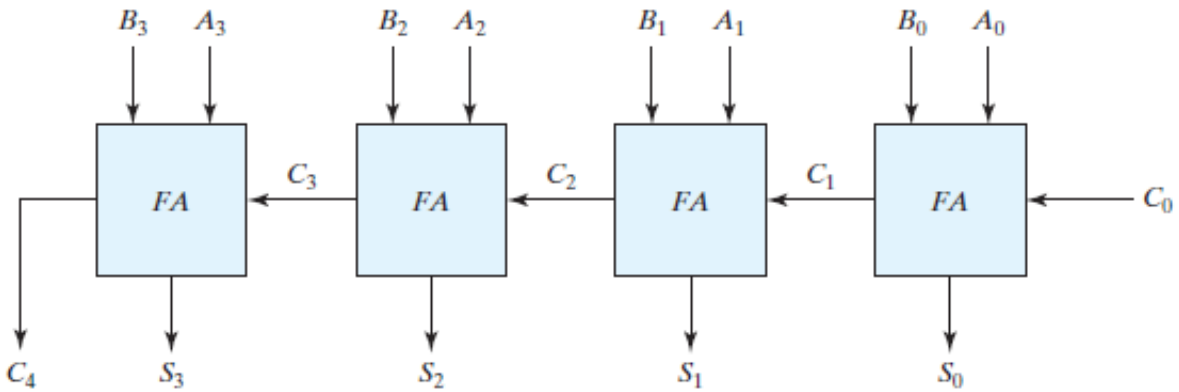


FIGURE 4.9
Four-bit adder

Binary Adder-Subtractor *cont'd*

- A four-bit binary ripple carry (or carry-ripple) adder.

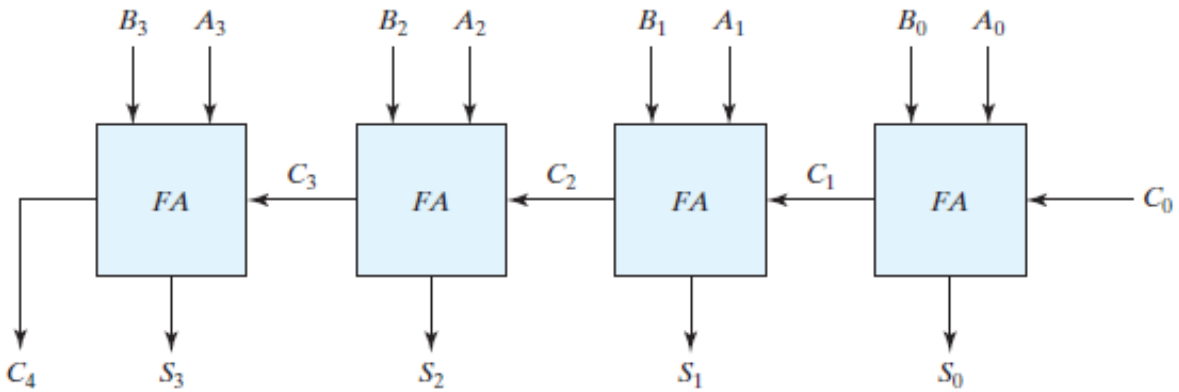


FIGURE 4.9
Four-bit adder

- An *n*-bit binary subtractor can be constructed by using an *n*-bit binary adder and *n* NOT gates, with the input carry being 1.

Binary Adder-Subtractor *cont'd*

- A four-bit binary ripple carry (or carry-ripple) adder.

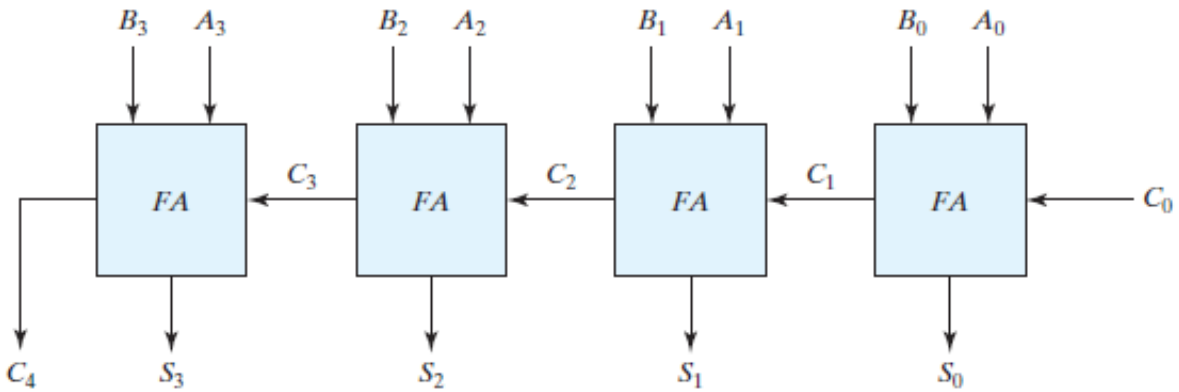


FIGURE 4.9
Four-bit adder

- An *n*-bit binary subtractor can be constructed by using an *n*-bit binary adder and *n* NOT gates, with the input carry being 1.
- The addition and the subtraction operations can be combined into one circuit with one common binary adder by
  - including an exclusive-OR gate with each full adder
  - employing an control input to select the operation.

Binary Adder-Subtractor *cont'd*

- A four-bit binary adder-subtractor with overflow detection.

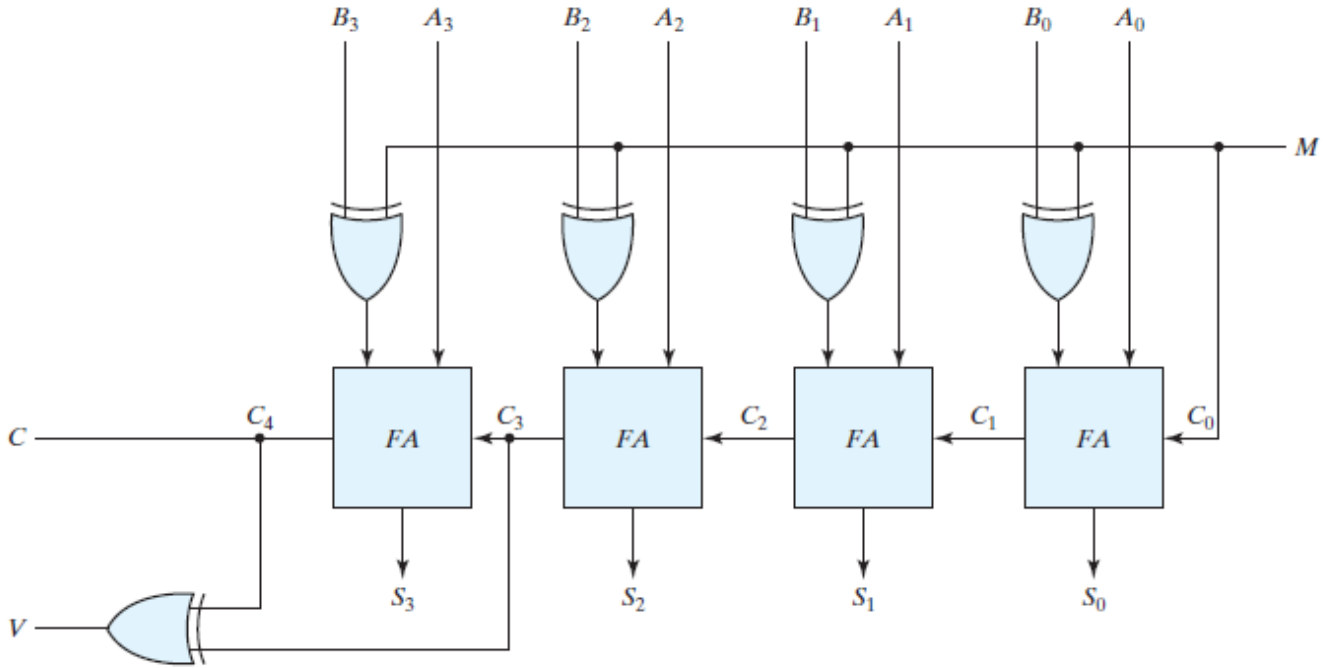


FIGURE 4.13
Four-bit adder–subtractor (with overflow detection)

- When $M = 0$, $B \oplus 0 = B$ and $C_0 = 0 \Rightarrow$ circuit performs addition.

Binary Adder-Subtractor *cont'd*

- A four-bit binary adder-subtractor with overflow detection.

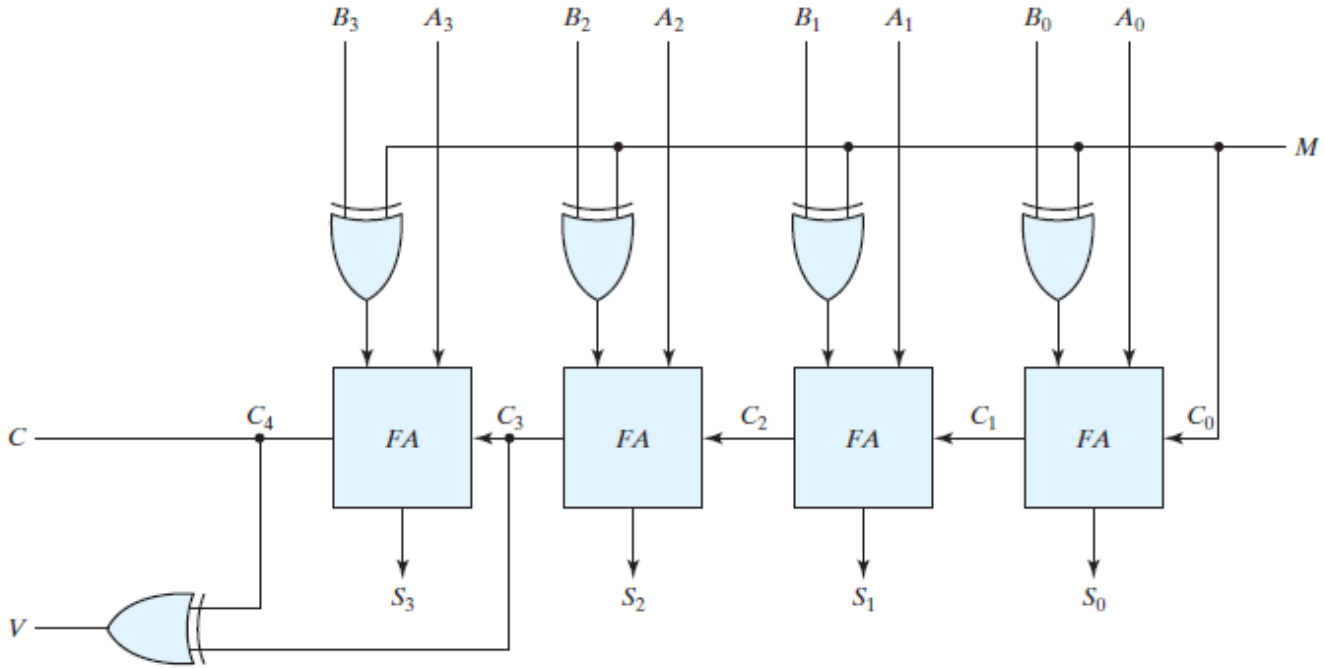


FIGURE 4.13
Four-bit adder–subtractor (with overflow detection)

- When $M = 0$, $B \oplus 0 = B$ and $C_0 = 0 \Rightarrow$ circuit performs addition.
- When $M = 1$, $B \oplus 1 = \bar{B}$ and $C_0 = 1 \Rightarrow$ circuit performs subtraction.

Binary Multiplier

- A two-bit by two-bit binary multiplier can be implemented using two half adders and four AND gates.

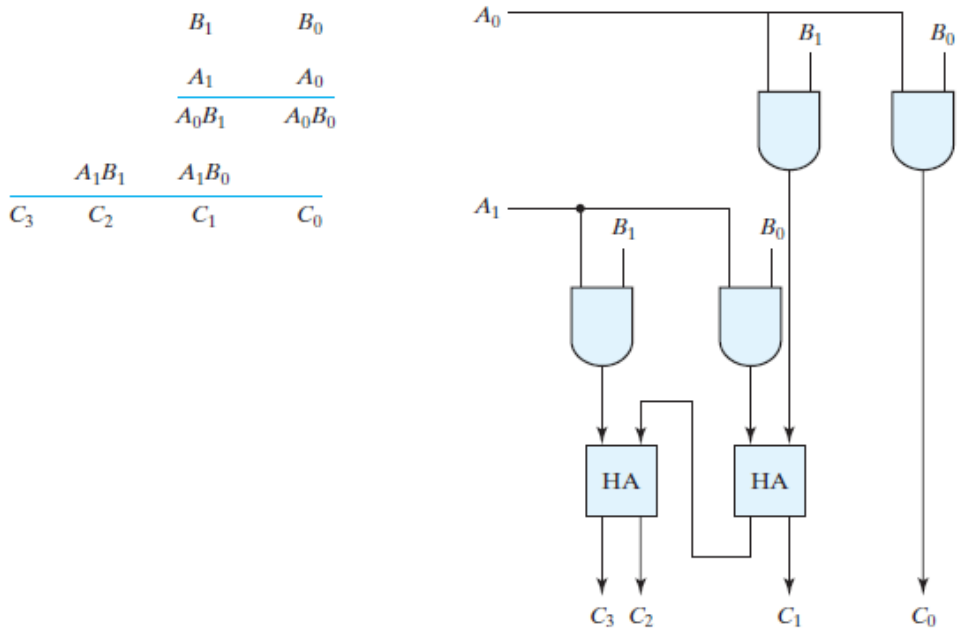


FIGURE 4.15
Two-bit by two-bit binary multiplier

Binary Multiplier

- A two-bit by two-bit binary multiplier can be implemented using two half adders and four AND gates.

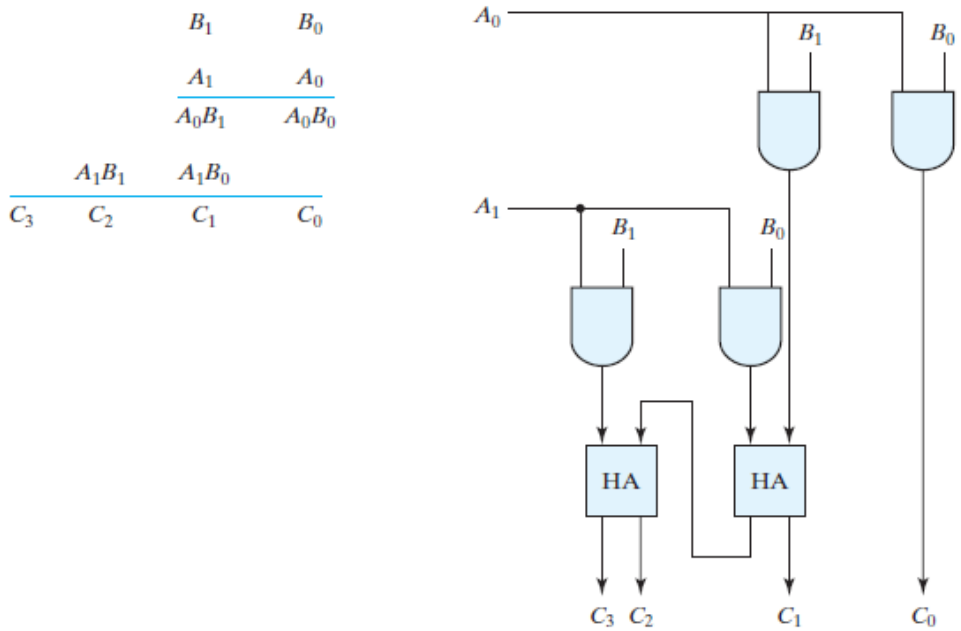


FIGURE 4.15
Two-bit by two-bit binary multiplier

- Home work:
  - Design a two-bit by two-bit multiplier using a truth table and K-maps.
  - Implement the circuit using only NAND and NOT gates.

Binary Multiplier *cont'd*

- For a *j*-bit by *k*-bit multiplier, we need $j \times k$ AND gates and $(k-1)$ *j*-bit adders to produce a product of $(j+k)$ bits.

Binary Multiplier *cont'd*

- For a *j*-bit by *k*-bit multiplier, we need $j \times k$ AND gates and $(k-1)$ *j*-bit adders to produce a product of $(j+k)$ bits.
- Example: four-bit by three-bit multiplier.

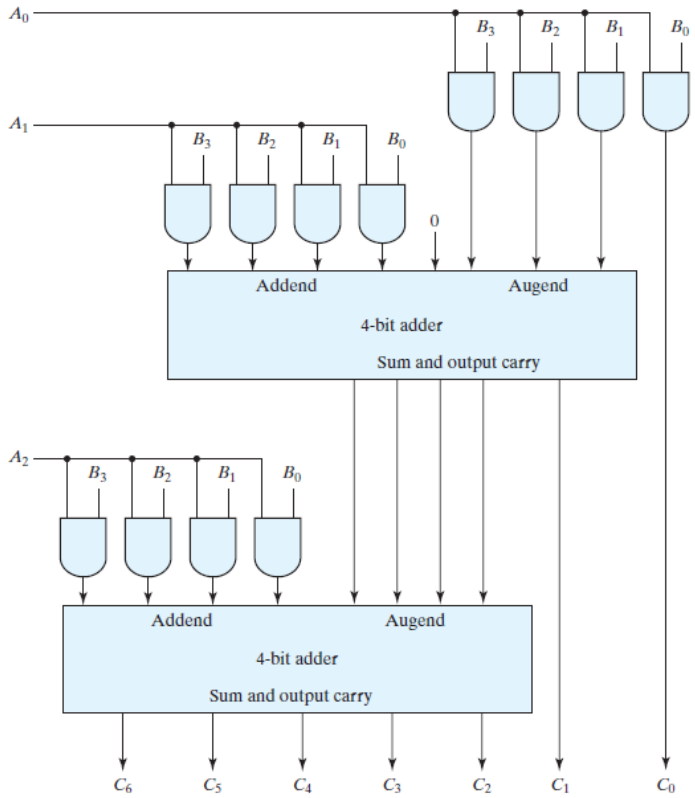


FIGURE 4.16
Four-bit by three-bit binary multiplier

Magnitude Comparator

- A magnitude comparator compares two numbers a and b and determines whether $a > b$, $a = b$ or $a < b$.

Magnitude Comparator

- A magnitude comparator compares two numbers a and b and determines whether $a > b$, $a = b$ or $a < b$.
- The combinational circuit for comparing two n-bit numbers has 2^{2n} entries in the truth table; 64 entries even for $n = 3$.

Magnitude Comparator

- A magnitude comparator compares two numbers a and b and determines whether $a > b$, $a = b$ or $a < b$.
- The combinational circuit for comparing two n-bit numbers has 2^{2n} entries in the truth table; 64 entries even for $n = 3$.
- So, we follow an algorithmic approach to design a magnitude comparator.

Magnitude Comparator

- A magnitude comparator compares two numbers a and b and determines whether $a > b$, $a = b$ or $a < b$.
- The combinational circuit for comparing two n-bit numbers has 2^{2n} entries in the truth table; 64 entries even for $n = 3$.
- So, we follow an algorithmic approach to design a magnitude comparator.
- Consider the comparison of two four-bit numbers $a = a_3 a_2 a_1 a_0$ and $b = b_3 b_2 b_1 b_0$.
  - $a = b$ if $a_i = b_i$ for $i = 0, 1, 1, 2, 3$

Magnitude Comparator

- A magnitude comparator compares two numbers a and b and determines whether $a > b$, $a = b$ or $a < b$.
- The combinational circuit for comparing two n-bit numbers has 2^{2n} entries in the truth table; 64 entries even for $n = 3$.
- So, we follow an algorithmic approach to design a magnitude comparator.
- Consider the comparison of two four-bit numbers $a = a_3 a_2 a_1 a_0$ and $b = b_3 b_2 b_1 b_0$.
  - $a = b$ if $a_i = b_i$ for $i = 0, 1, 1, 2, 3$
  - To find whether $a > b$ or $a < b$, we inspect the relative magnitudes of pairs of significant digits, starting from the most significant position.

Magnitude Comparator *cont'd*

- Example: four-bit magnitude comparator.

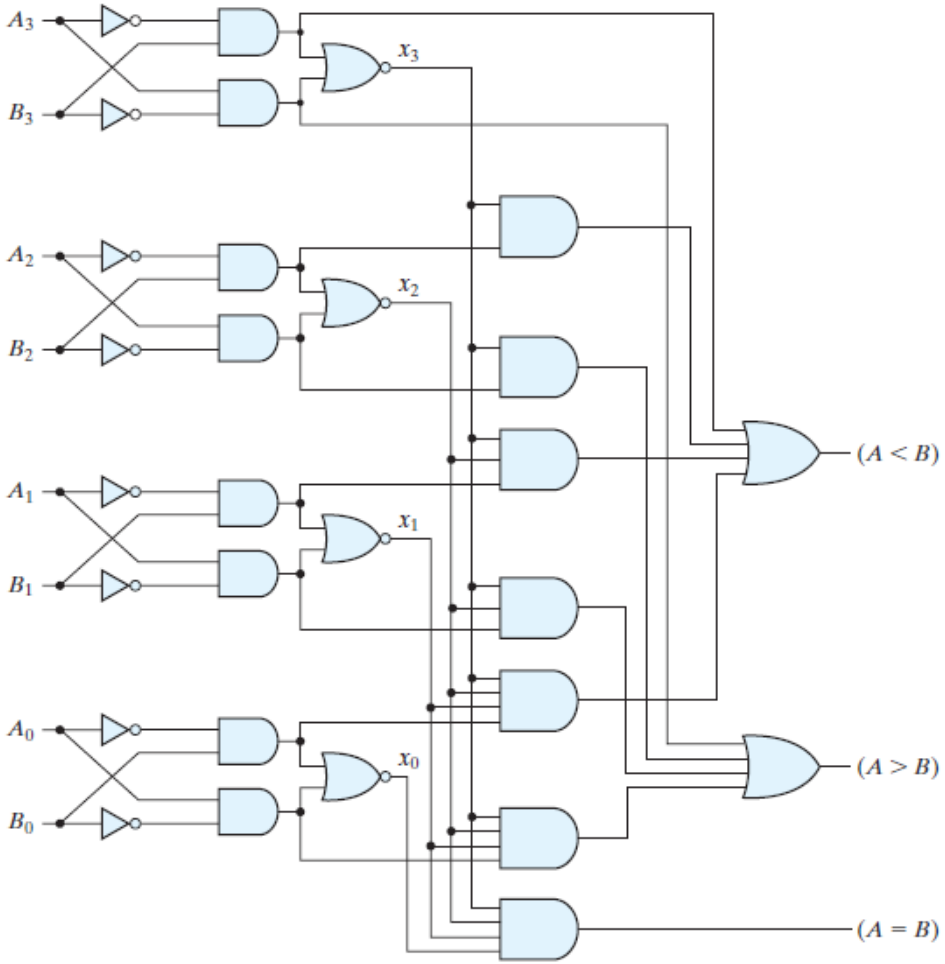


FIGURE 4.17
Four-bit magnitude comparator

Decoder

- A decoder, generally called n-to-m-line decoder, converts binary information from n input lines to m unique output lines, where $m \leq 2^n$.

Decoder

- A decoder, generally called n-to-m-line decoder, converts binary information from n input lines to m unique output lines, where $m \leq 2^n$.
- Example: 3-to-8-line decoder.

Table 4.6
Truth Table of a Three-to-Eight-Line Decoder

Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Decoder *cont'd*

- An n-to-m-line decoder can be implemented using n NOT gates and m AND gates.

Decoder *cont'd*

- An n-to-m-line decoder can be implemented using n NOT gates and m AND gates.
- Example: Circuit of a 3-to-8-line decoder.

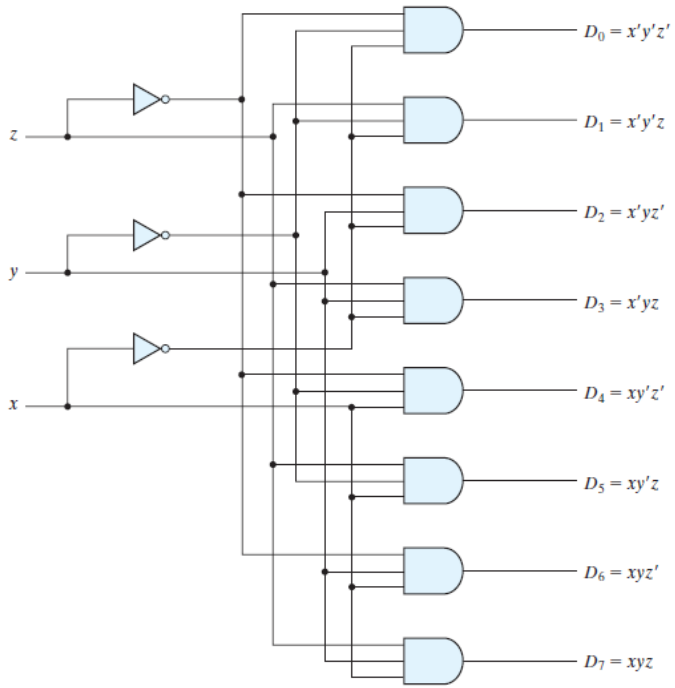


FIGURE 4.18
Three-to-eight-line decoder

Decoder *cont'd*

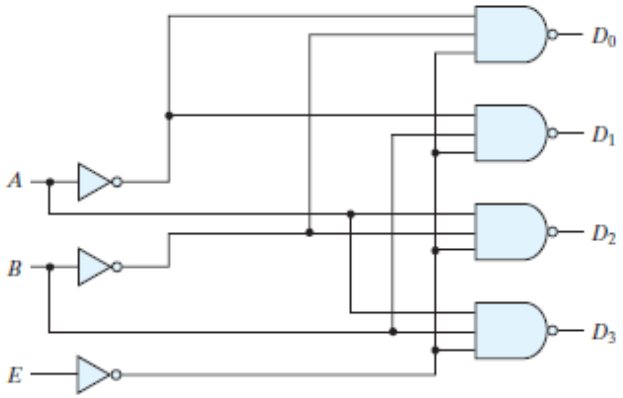
- Some decoders are implemented with NAND gates instead of AND gates, where the decoder minterms are generated in their complemented form.

Decoder *cont'd*

- Some decoders are implemented with NAND gates instead of AND gates, where the decoder minterms are generated in their complemented form.
- Some decoders include one or more enable inputs to control the circuit operation.

Decoder *cont'd*

- Some decoders are implemented with NAND gates instead of AND gates, where the decoder minterms are generated in their complemented form.
- Some decoders include one or more enable inputs to control the circuit operation.
- Example: 2-to-4-line decoder with an enable input (active low).



(a) Logic diagram

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	1

(b) Truth table

FIGURE 4.19
Two-to-four-line decoder with enable input

Decoder *cont'd*

- Decoders can be used to implement Boolean functions.

Decoder *cont'd*

- Decoders can be used to implement Boolean functions.
- Example:
  - Implement the following Boolean function using a 3-to-8-line decoder: $F(x, y, z) = \Sigma(0, 2, 3, 5, 7)$.

Decoder *cont'd*

- Decoders can be used to implement Boolean functions.
- Example:
  - Implement the following Boolean function using a 3-to-8-line decoder: $F(x, y, z) = \Sigma(0, 2, 3, 5, 7)$.
- Home work:
  - Implement the following Boolean function using two 2-to-4-line decoders with active-high enable inputs: $F(x, y, z) = \Sigma(0, 2, 3, 5, 7)$.

Encoder

- An encoder performs the inverse operation of a decoder.

Encoder

- An encoder performs the inverse operation of a decoder.
- Example: 8-to-3-line encoder.

Table 4.7
Truth Table of an Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Encoder

- An encoder performs the inverse operation of a decoder.
- Example: 8-to-3-line encoder.

Table 4.7
Truth Table of an Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

- The encoder can be implemented with OR gates whose inputs are determined directly from the truth table.

Encoder *cont'd*

- A priority encoder is an encoder that includes the priority function.

Encoder *cont'd*

- A priority encoder is an encoder that includes the priority function.
- If two or more inputs of a priority encoder are equal to 1 at the same time, the input having the highest priority will take precedence.

Encoder *cont'd*

- A priority encoder is an encoder that includes the priority function.
- If two or more inputs of a priority encoder are equal to 1 at the same time, the input having the highest priority will take precedence.
- A priority encoder has an additional output, the valid bit.
  - The valid bit is set to 1 when one or more inputs are equal to 1.
  - The valid bit is set to 0 when all inputs are 0.

Encoder *cont'd*

- A priority encoder is an encoder that includes the priority function.
- If two or more inputs of a priority encoder are equal to 1 at the same time, the input having the highest priority will take precedence.
- A priority encoder has an additional output, the valid bit.
  - The valid bit is set to 1 when one or more inputs are equal to 1.
  - The valid bit is set to 0 when all inputs are 0.
- Example: 4-to-2-line priority encoder.

Table 4.8
Truth Table of a Priority Encoder

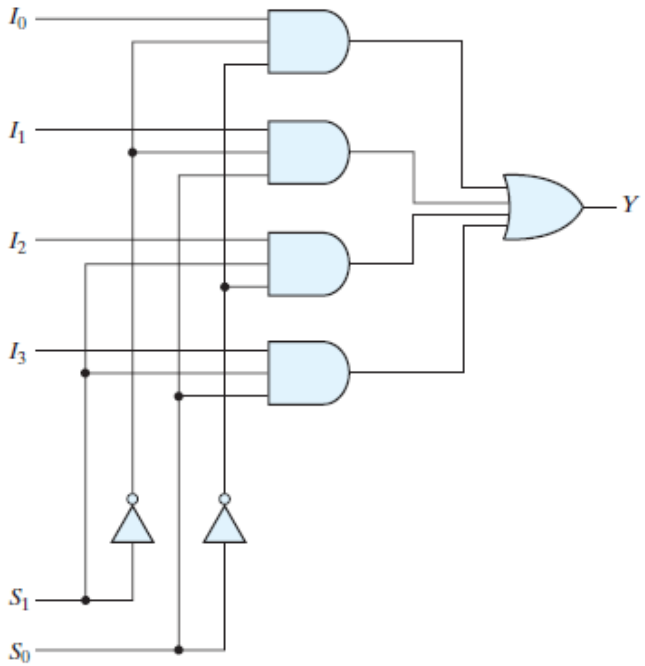
Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

Multiplexer

- A multiplexer selects binary information from one of many input lines and directs it to a single output line.
- Generally, there are 2^n input lines and n selection lines.

Multiplexer

- A multiplexer selects binary information from one of many input lines and directs it to a single output line.
- Generally, there are 2^n input lines and n selection lines.
- Example: 4-to-1-line multiplexer.



(a) Logic diagram

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(b) Function table

FIGURE 4.25
Four-to-one-line multiplexer

Multiplexer *cont'd*

- A Boolean function of n variables can be implemented with a multiplexer having $n-1$ selection inputs.
- In this case, the first $n-1$ variables of the function are connected to the selection inputs and the remaining variable is used for the data inputs.

Multiplexer *cont'd*

- A Boolean function of n variables can be implemented with a multiplexer having $n-1$ selection inputs.
- In this case, the first $n-1$ variables of the function are connected to the selection inputs and the remaining variable is used for the data inputs.
- Example:
  - Implement the following Boolean function using a 4-to-1-line multiplexer: $F(x, y, z) = \Sigma(0, 2, 3, 5, 7)$.
- Home work:
  - Implement the following Boolean function using an 8-to-1-line multiplexer: $F(a, b, c, d) = \Sigma(1, 3, 4, 11, 12, 13, 14, 15)$.

Demultiplexer

- A demultiplexer performs the inverse operation of a multiplexer.
- It connects a single input line to one of many output lines.
- Generally, there are 2^n output lines and n selection lines.

Demultiplexer

- A demultiplexer performs the inverse operation of a multiplexer.
- It connects a single input line to one of many output lines.
- Generally, there are 2^n output lines and n selection lines.
- Example: 1-to-8-line demultiplexer.