

Appendix A

Digital Logic

Objectives

- To know the Boolean algebra
- To know the use of different types of logic gate
- To know how to construct combinational circuits

Boolean Operators

P	Q	NOT P	P AND Q	P OR Q	P XOR Q	P NAND Q	P NOR Q
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	1	0
1	1	0	1	1	0	0	0

Basic Identities of Boolean Algebra

Basic Postulates		
$A \cdot B = B \cdot A$	$A + B = B + A$	Commutative Laws
$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A + B) \cdot (A + C)$	Distributive Laws
$1 \cdot A = A$	$0 + A = A$	Identity Elements
$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$	Inverse Elements
Other Identities		
$0 \cdot A = 0$	$1 + A = 1$	
$A \cdot A = A$	$A + A = A$	
$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$	Associative Laws
$\overline{\bar{A} \cdot \bar{B}} = \bar{A} + \bar{B}$	$\overline{\bar{A} + \bar{B}} = \bar{A} \cdot \bar{B}$	DeMorgan's Theorem

Basic Logic Gates

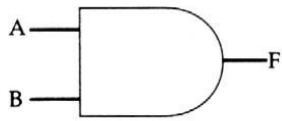
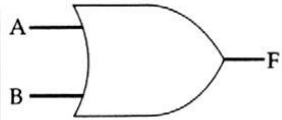
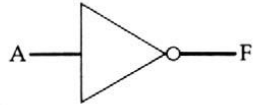
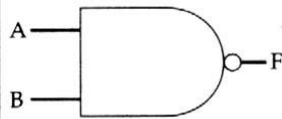
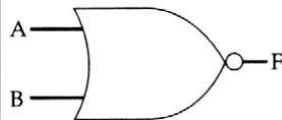
Name	Graphic Symbol	Algebraic Function	Truth Table															
AND		$F = A \cdot B$ or $F = AB$	<table border="1" style="display: inline-table;"> <thead> <tr> <th>A</th> <th>B</th> <th>F</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	F	0	0	0	0	1	0	1	0	0	1	1	1
A	B	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = A + B$	<table border="1" style="display: inline-table;"> <thead> <tr> <th>A</th> <th>B</th> <th>F</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</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>1</td></tr> </tbody> </table>	A	B	F	0	0	0	0	1	1	1	0	1	1	1	1
A	B	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NOT		$F = \bar{A}$ or $F = A'$	<table border="1" style="display: inline-table;"> <thead> <tr> <th>A</th> <th>F</th> </tr> </thead> <tbody> <tr><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </tbody> </table>	A	F	0	1	1	0									
A	F																	
0	1																	
1	0																	
NAND		$F = (\overline{AB})$	<table border="1" style="display: inline-table;"> <thead> <tr> <th>A</th> <th>B</th> <th>F</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	F	0	0	1	0	1	1	1	0	1	1	1	0
A	B	F																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR		$F = (\overline{A + B})$	<table border="1" style="display: inline-table;"> <thead> <tr> <th>A</th> <th>B</th> <th>F</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	F	0	0	1	0	1	0	1	0	0	1	1	0
A	B	F																
0	0	1																
0	1	0																
1	0	0																
1	1	0																

Figure A.1 Basic Logic Gates.

Applications of NAND and NOR Gates

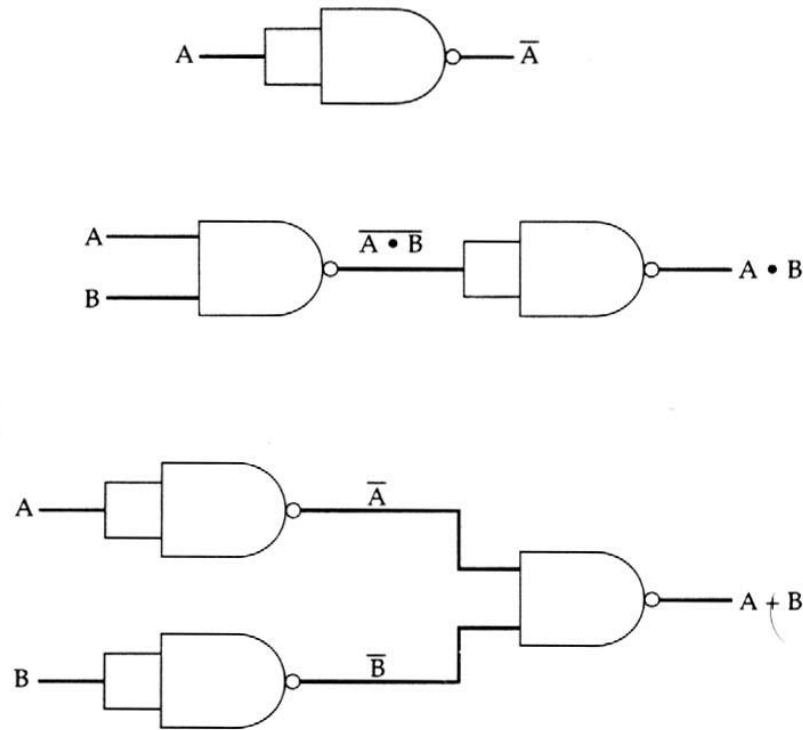


Figure A.2 The Use of NAND Gates.

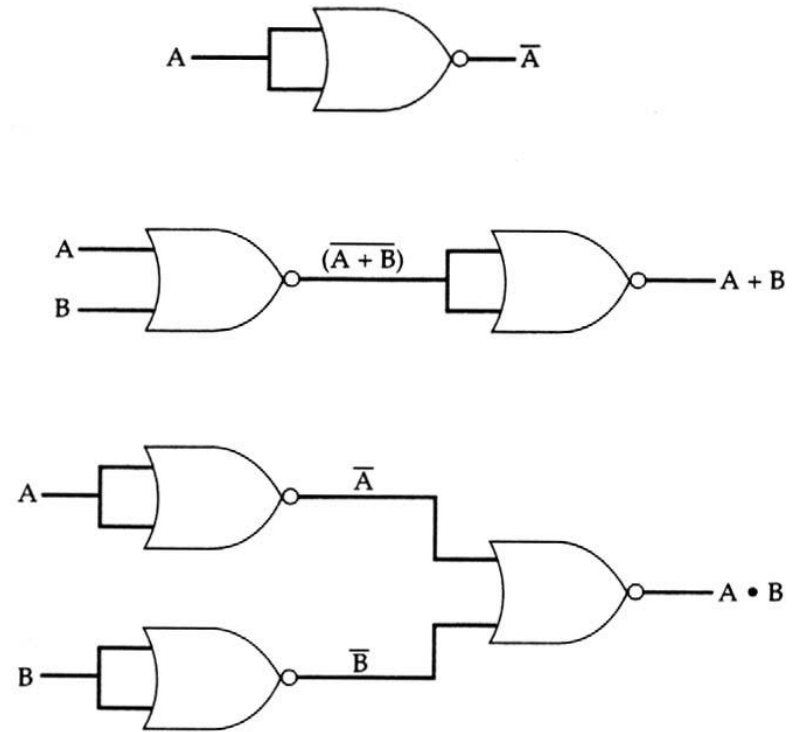


Figure A.3 The Use of NOR Gates.

Combination Circuits

$$F = A'BC' + A'BC + ABC'$$

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

Combination Circuits

$$F = A'BC' + A'BC + ABC'$$

(sum of products)

$$F = (A+B+C) \cdot (A+B+C') \cdot (A'+B+C) \cdot (A'+B+C')$$

(product of sums)

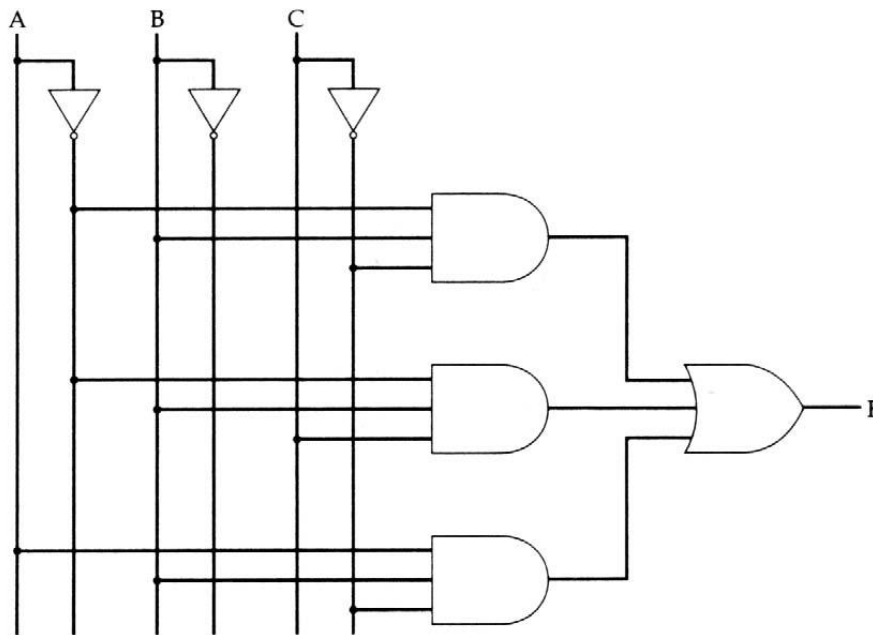


Figure A.4 Sum-of-Products Implementation of Table A.3

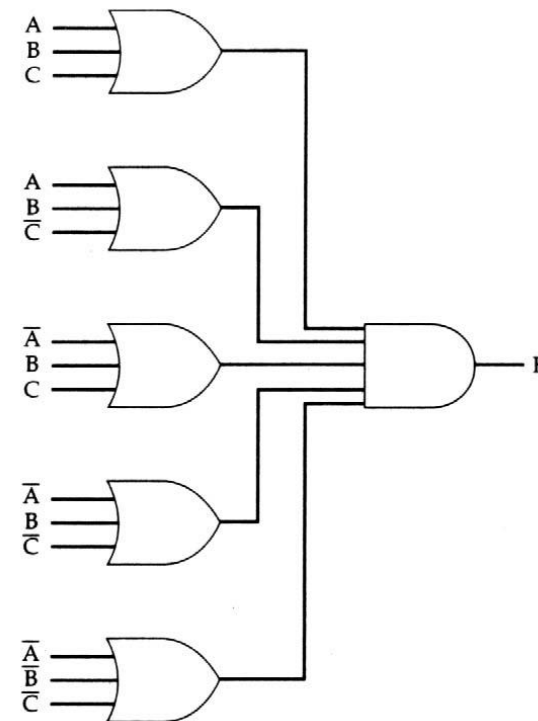


Figure A.5 Product-of-Sums Implementation of Table A.3

Simplification

$$F = A'BC' + A'BC + ABC'$$

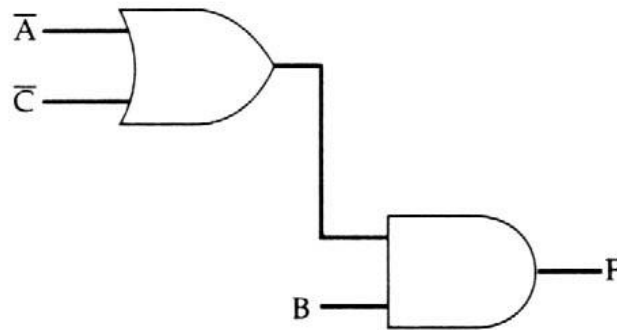
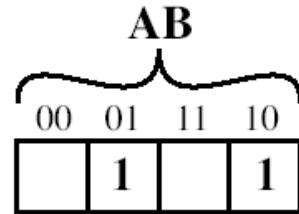
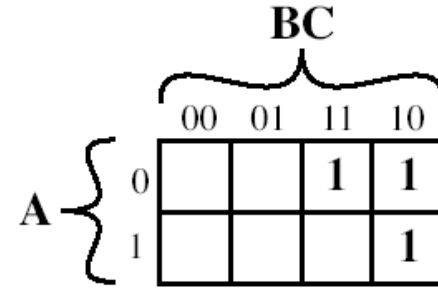


Figure A.6 Simplified Implementation of Table A.3.

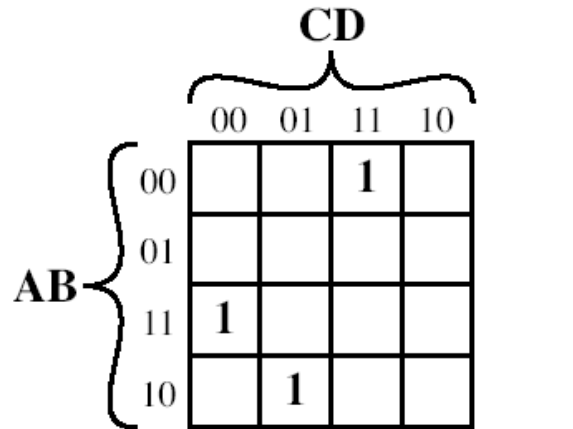
Simplification using Karnaugh Maps



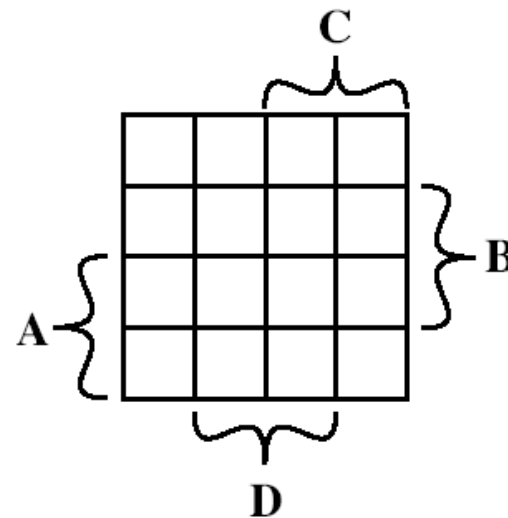
(a) $F = A\bar{B} + \bar{A}B$



(b) $F = \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C}$



(c) $F = \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + AB\bar{C}\bar{D}$



(d) Simplified Labeling of Map

Simplification using Karnaugh Maps

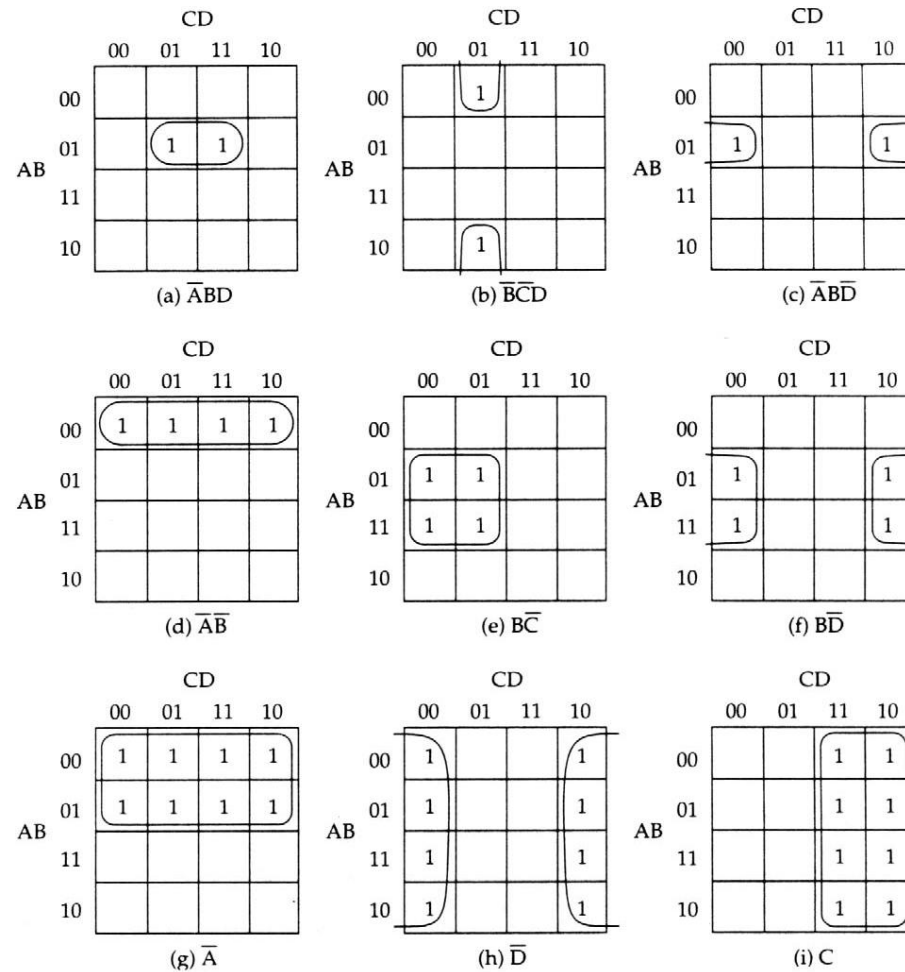


Figure A.8 The Use of Karnaugh Maps.

Simplification Karnaugh Maps

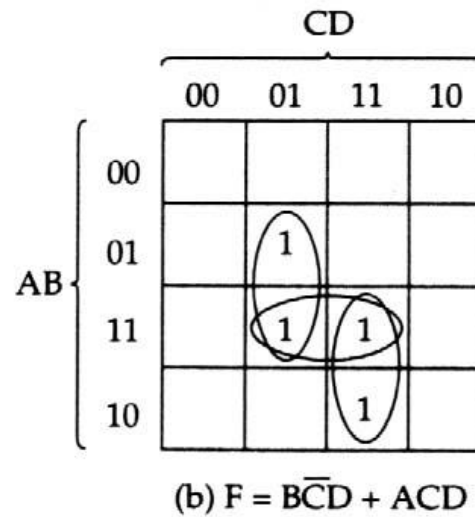
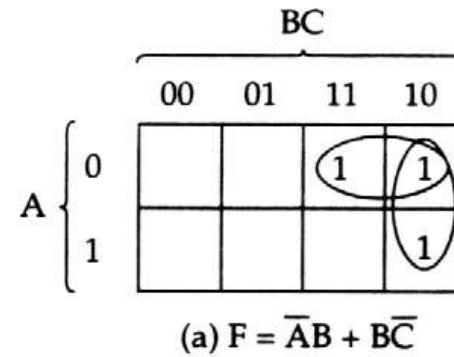


Figure A.9 Overlapping Groups.

Simplification Karnaugh Maps

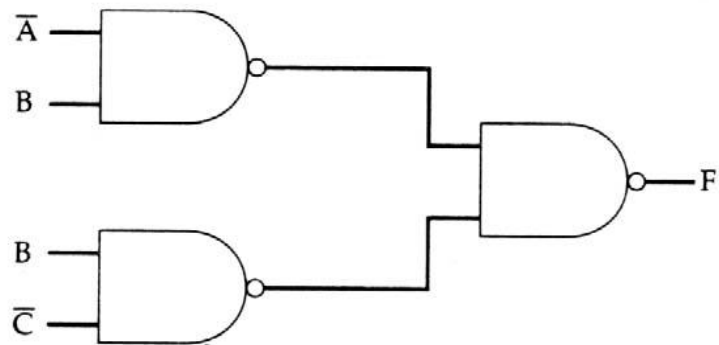


Figure A.11 NAND Implementation of Table A.3.

Multiplexer

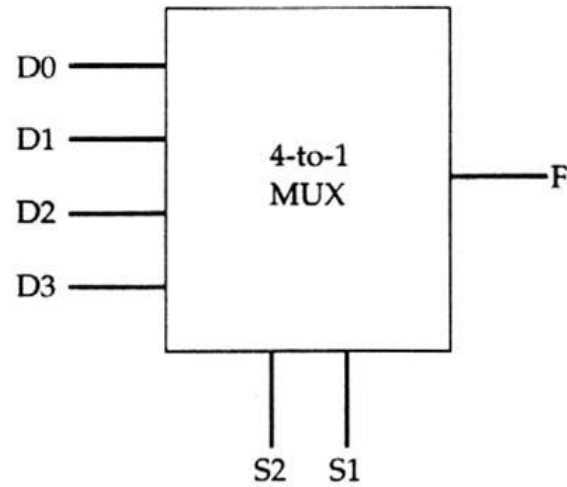


Figure A.12 4-to-1 Multiplexer Representation.

Table A.7 4-to-1 Multiplexer Truth Table

S2	S1	F
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Multiplexer Implementation

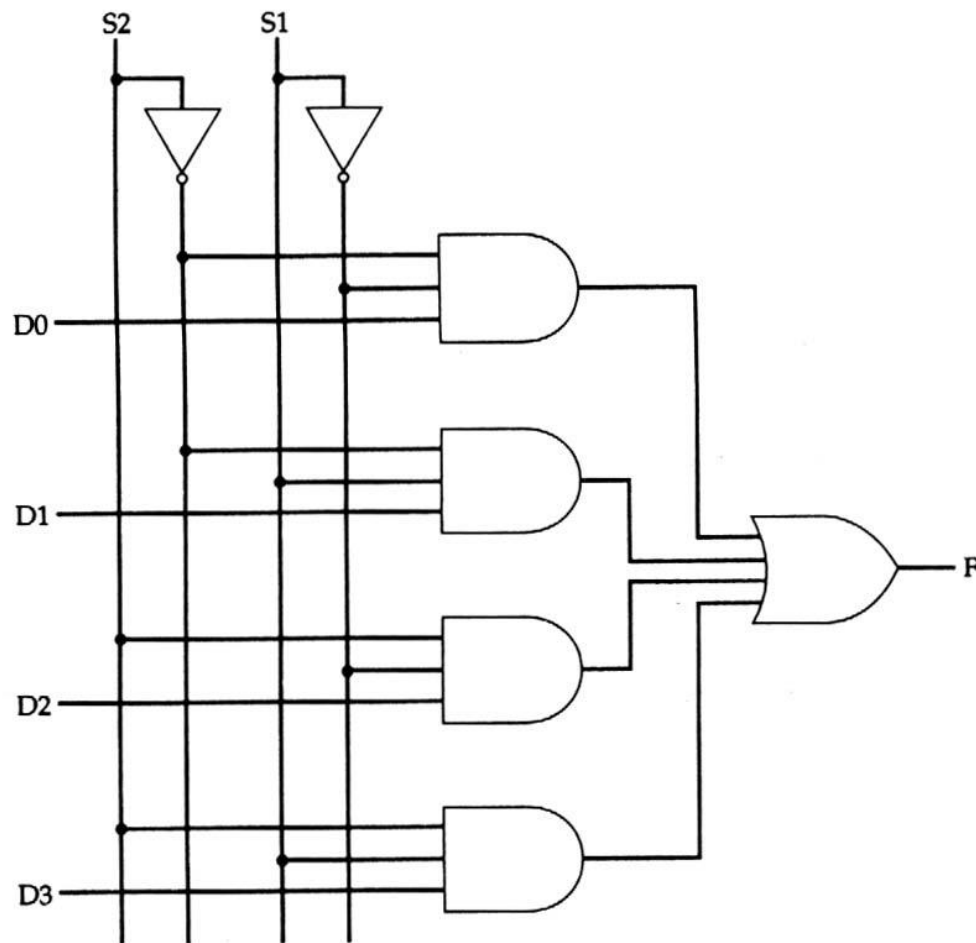


Figure A.13 Multiplexer Implementation.

Decoder

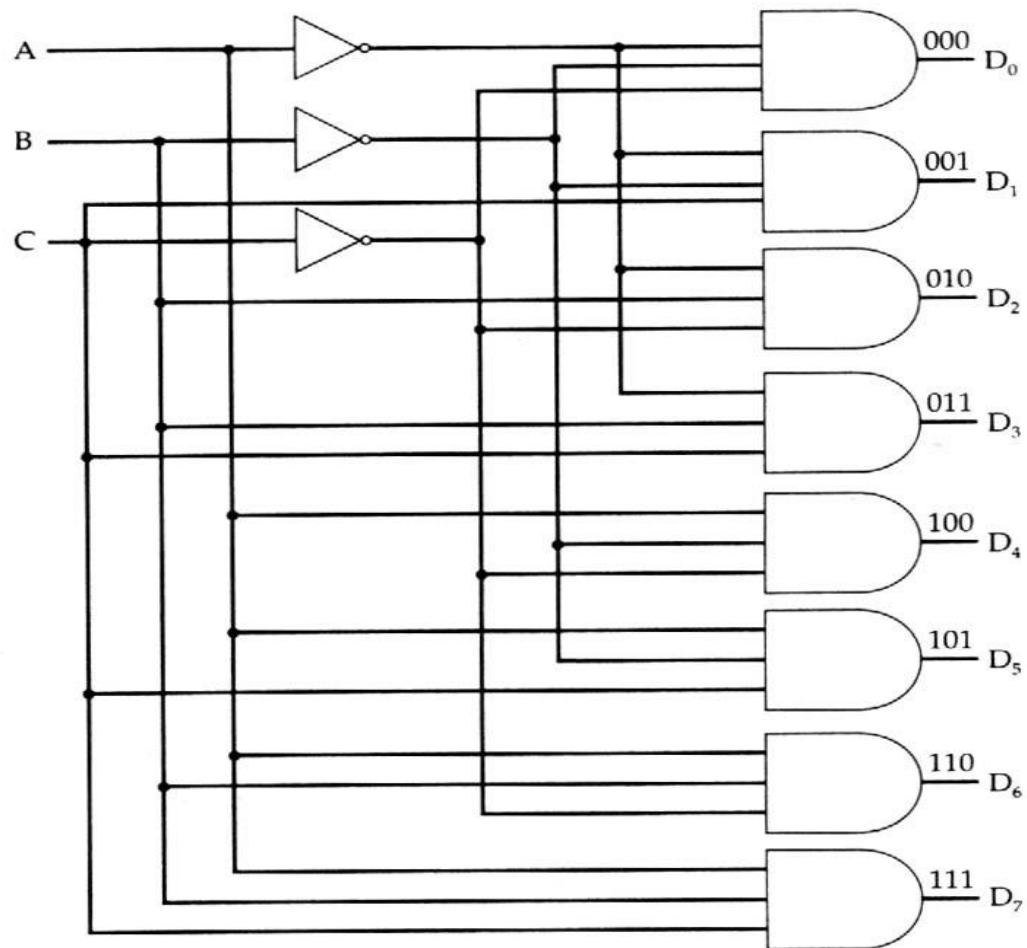


Figure A.15 Decoder with 3 Inputs and $2^3 = 8$ Outputs.

Adder

Table A.9 Binary Addition Truth Tables

(a) Single-Bit Addition

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(b) Addition with Carry Input

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

Full Adder

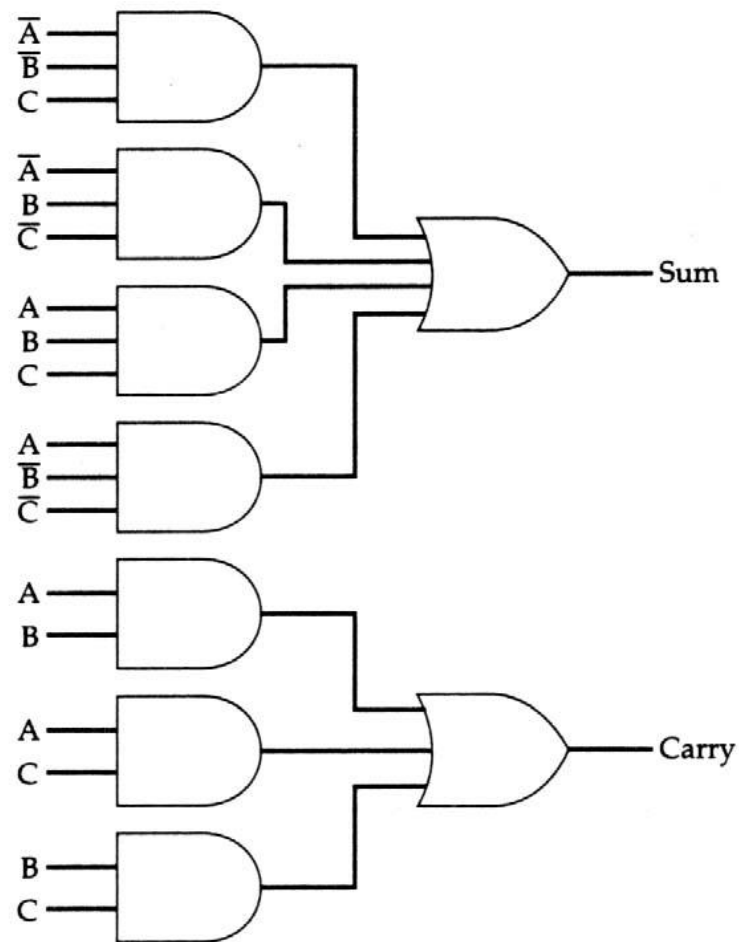


Figure A.22 Implementation of an Adder.