03-Boolean Algebra Part 1

Text: Unit 2

ECEGR/ISSC 201
Digital Operations and Computations
Winter 2011

Overview

- Introduction
- Basic Operations
- Switch Realizations
- Boolean Expressions
- Truth Tables

Introduction

- Consider this logical statement:
  - *The light is to turn on if button A and button B are pushed or if button A is pushed and button B is not pushed or if button C is pushed*
- Is the light on if button A is not pushed, B is pushed and C is not pushed?

Introduction

- Consider this logical statement:
  - *The light is to turn on if button A and button B are pushed or if button A is pushed and button B is not pushed or if button C is pushed*
- Is the light on if button A is not pushed, B is pushed and C is not pushed?
  - No
- In fact, we can rewrite the statement as
  - *The light turns on if button A or button C are pushed*
  - This is an easier expression to implement

Introduction

- We are interested in examining mathematical logical expressions in order to reduce, reorder or find equivalence
- Specifically, we are interested in logic in which the variables assume one of two values (such as a switch)
- The mathematics of this logic is known as Boolean Algebra

Introduction

- What is a Boole?
- George Boole (1815-1864)

Boolean Algebra

- Boolean variables, such as *A*, *B*, *X* or *Y*
  - Input(s) or output(s) of a switching circuit
  - Symbols 1 or 0 (not binary numbers) are used to represent different values
  - Could also use T and F or Up and Down

Basic Operations

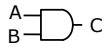
- The basic operations are:
  - AND
  - OR
  - Complement (NOT)

AND

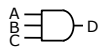
- Consider two inputs A and B and output C
- AND: compares A and B, if A = B = 1 (both 1), then C=1; otherwise C=0
- Written as AB, A(B), (A)B, A x B, A • B
- Example: AB = C

AND

- Symbolic gate representation:



- Possible to AND more than two inputs:
- ABC = D



- AND is logical or Boolean multiplication

AND

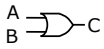
A	B	C = AB
0	0	0
0	1	0
1	0	0
1	1	1

OR

- Consider two inputs A and B and output C
- OR: compares A and B, if A or B is 1, then C=1; otherwise C=0
- Written as A+B = C

OR

- Symbolically:



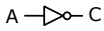
- Possible for more than two inputs to be OR'ed
- A+B+C=D
- OR is logical or Boolean addition

OR

A	B	C = A+B
0	0	0
0	1	1
1	0	1
1	1	1

Complement

- Consider input A and output C
- Complement: inverts the input; if A = 1 then C =0; if A = 0, then C = 1
- Written as A' = C, also known as NOT
- Symbolically:



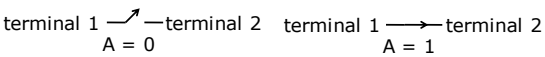
- NOT is logical or Boolean inversion

Complement

A	C = A'
0	1
1	0

Switch Realization

- Switches can be used to model AND and OR statements
- Each variable is a switch
- Each switch has two terminals
- If variable is equal to 1, then the switch is closed, if it is 0, then it is open
- If terminal 1 is connected to terminal 2, then the output is 1, otherwise it is zero
- NOT is represented by a normally closed switch

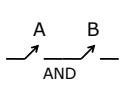


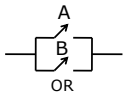
Switch Realization

- Which is the AND?
- Which is the OR?

Switch Realization

- Which is the AND?
- Which is the OR?



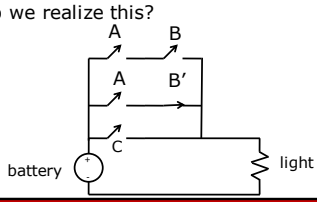


Switch Realization

- The light is to turn on if button A and button B are pushed or if button A is pushed and button B is not pushed or if button C is pushed
- "Pushed" means variable equals 1
- How do we realize this?

Switch Realization

- The light is to turn on if button A and button B are pushed or if button A is pushed and button B is not pushed or if button C is pushed
- "Pushed" means variable equals 1
- How do we realize this?

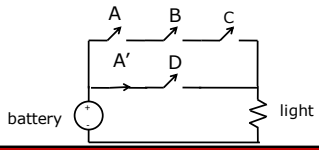


Example

- Draw the switch realization of the following statement:
  - The light should be lit if button A and button B and button C are pushed, or if button A is not pushed and button D is pushed

Example

- Draw the switch realization of the following statement:
  - The light should be lit if button A and button B and button C are pushed, or if button A is not pushed and button D is pushed



Boolean Expressions

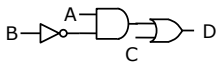
- AND, OR and NOT can be combined into expressions to represent complex logical functions
- Examples:
  - AB' + C = F
  - [A(C+D)]' +BE =F

Boolean Expressions

- Order of operations when parenthesis are not present
  - Complement
  - AND
  - OR
- Example: AB' + C =D
  - First: Complement B
  - Second: AB'
  - Third: AB' + C

Boolean Expressions

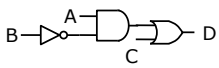
- Using gates: AB' + C =D



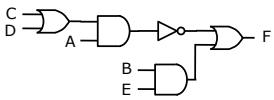
- Write the gate equivalent of [A(C+D)]' +BE =F

Boolean Expressions

- Using gates: AB' + C =D

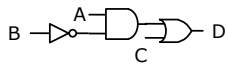


- Write the gate equivalent of [A(C+D)]' +BE =F



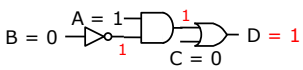
Evaluation of Boolean Expressions

- Boolean expressions are evaluated by substituting a 1 or 0 for each variable
- Example: what is D if A = 1, B = 0 and C = 0?
  - AB' + C =D



Evaluation of Boolean Expressions

- Boolean expressions are evaluated by substituting a 1 or 0 for each variable
- Example: what is D if A = 1, B = 0 and C = 0?
  - AB' + C =D



- Each variable or complement of a variable is known as a *literal*
- Each literal is a gate input

Truth Tables

- A truth table specifies the output for every possible combination of inputs
- Intermediate results can be included
- For A' + B = C

inputs		intermediate	output
A	**B**	**A'**	**C = A'+B**
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

optional intermediate result

Example

- Complete the Truth Table for AB′ + C

A	B	C	B′	AB′	D= AB′ + C
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1		
1	0				
1					

Example

- Complete the Truth Table for AB′ + C

A	B	C	B′	AB′	D= AB′ + C
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	1

Truth Tables

- Truth table of an expression with n variables will have 2^n rows
- Two expressions that have the same output for every combination of inputs are equivalent expressions

Truth Tables

- Example: are the following equal?
  - AB′ + C
  - (A+C)(B′+C)

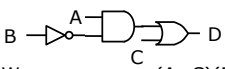
A	B	C	AB′ + C	(A+C)(B′+C)
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

Yes!

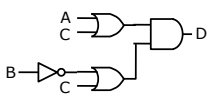
Why?

Truth Tables

- (A+C)(B′+C) = AB′ + C
- Recall: AB′ + C



- We can express (A+C)(B′+C) as



Truth Tables

- Since the expressions are equal, which should we construct?

Truth Tables

- Since the expressions are equal, which should we construct?
- AB′ + C requires three gates
- (A+C)(B′+C) requires four gates
- AB′ + C is easier to construct
- There is benefit in seeking ways to reduce expressions