EECS 42 Introduction to Digital Electronics
Andrew R. Neureuther

Lecture # 10 Prof. King: Basic Digital Blocks

- **20 Min Quiz**
  Basic Circuit Analysis and Transients
- **Logic Functions, Truth Tables**
- **Circuit Symbols, Logic from Circuit**

Schwarz and Oldham 11.1, 11.2 393-402

Midterm 10/2: Lectures # 1-9: 4 Topics – See slide 2
Length/Credit Review TBA

http://inst.EECS.Berkeley.EDU/~ee42/

Logical Expressions

Standard logic notation :

AND: "dot" Examples: $X = A \cdot B$; $Y = A \cdot B \cdot C$

OR : "+ sign" Examples: $W = A+B$; $Z = A+B+C$

NOT: "bar over symbol for complement" Example: $Z = \overline{A}$

With these basic operations we can construct any logical expression.

Order of operation: NOT, AND, OR (note that negation of an expression is performed after the expression is evaluated, so there is an implied parenthesis, e.g. $\overline{A \bullet B}$ means $\overline{(A \bullet B)}$.

First Midterm Exam: Topics

- Basic Circuit Analysis (KVL, KCL)
- Equivalent Circuits and Graphical Solutions for Nonlinear Loads
- Transients in Single Capacitor Circuits
- Node Analysis Technique and Checking Solutions

Exam is in class 9:40-10:45 AM, Closed book, Closed notes, Bring a calculator, Paper provided

Logic Function Example

- Boolean Expression: $H = (A \cdot B \cdot C) + T$

This can be read H=**1** if (A **and** B **and** C are **1) or** T is **1,** or

H is true if all of A,B,and C are true, or T is true, or

The voltage at node H will be high if the input voltages at nodes A, B and C are high or the input voltage at node T is high

Logic Functions

Logic Expression : To create logic values we will define **"True"** , as **Boolean 1** and **"False"** , as **Boolean 0**.

Moreover we can associate a logic variable with a circuit node. Typically we associate logic 1 with a high voltage (e.g. 2V) and and logic 0 with a low voltage (e.g. 0V).

Example: The logic variable H is true (H=**1**) if (A **and** B **and** C are **1) or** T is true (logic **1**), where all of A,B,C and T are also logical variables**.**

Logic Statement: H = 1 if A and B and C are 1 or T is 1.

We use "dot" to designate logical "and" and "+" to designate logical or in switching algebra. So how can we express this as a Boolean Expression?

Boolean Expression: $H = (A \cdot B \cdot C) + T$

Note that there is an order of operation, just as in math, and AND is performed before OR. Thus the parenthesis are not actually required here.

Logic Function Example 2

You wish to express under which conditions your burglar alarm goes off (**B=1**):

If the "Alarm Test" button is pressed (**A=1**)

OR if the Alarm is Set (**S=1**) AND { the door is opened (**D=1**) OR the trunk is opened (**T=1**)}

Boolean Expression: $B = A + S(D + T)$

This can be read **B=1 if A = 1 or S=1 AND (D OR T =1),** i.e.

B=1 if {A = 1} or {S=1 AND (D OR T =1)}

or

B is true IF {A is true} OR {S is true AND D OR T is true}

or

The voltage at node H will be high if {the input voltage at node A is high} OR {the input voltage at S is high and the voltages at D and T are high}

Evaluation of Logical Expressions with "Truth Tables"

Truth Table for Logic Expression $H = (A \cdot B \cdot C) + T$

A	B	C	T	H
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	1
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Some Important Logical Functions

- **"AND"** $A \cdot B$ (or $A \cdot B \cdot C$)
- **"OR"** $A + B$ (or $A + B + C + D \ldots$)
- **"INVERT" or "NOT"** not A (or $\overline{A}$)
- **"not AND" = NAND** $\overline{AB}$ (only 0 when A and $B = 1$)
- **"not OR" = NOR** $\overline{A+B}$ (only 1 when $A = B = 0$)
- **exclusive OR = XOR** $A \oplus B$ (only 1 when A, B differ)
  i.e., $A + B$ **except** $A \cdot B$

Evaluation of Logical Expressions with "Truth Tables"

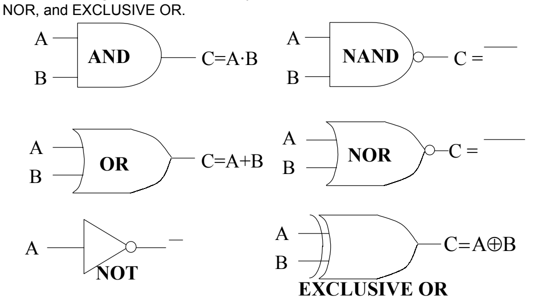
The Truth Table completely describes a logic expression

In fact, we will use the Truth Table as the fundamental meaning of a logic expression.

Two logic expressions are equal if their truth tables are the same

Logic Gates

These are circuits that accomplish a given logic function such as "OR". We will shortly see how such circuits are constructed. Each of the basic logic gates has a unique symbol, and there are several additional logic gates that are regarded as important enough to have their own symbol. The set is: AND, OR, NOT, NAND, NOR, and EXCLUSIVE OR.



The Important Logical Functions

The most frequent (i.e. important) logical functions are implemented as electronic "building blocks" or "gates".

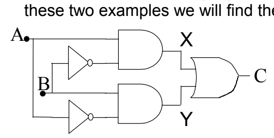
We already know about **AND** , **OR** and **NOT** What are some others:

Combination of above: inverted AND = **NAND,**
inverted OR **= NOR**

And one other basic function is often used: the "EXCLUSIVE OR" … which logically is "or except not and"

Logic Circuits

With a combination of logic gates we can construct any logic function. In these two examples we will find the truth table for the circuit.



It is helpful to list the intermediate logic values (at the input to the OR gate). Let's call them X and Y.

Now we complete the truth tables for X and Y, and from that for C. (Note that X and Y and finally C = X + Y)

A	B	X	Y	C
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Interestingly, this is the same truth table as the EXCLUSIVE OR

EECS 42 Introduction to Digital Electronics
Andrew R. Neureuther

Lecture # 10 Prof. King: Basic Digital Blocks

- **20 Min Quiz**
  Basic Circuit Analysis and Transients
- **Logic Functions, Truth Tables**
- **Circuit Symbols, Logic from Circuit**

Schwarz and Oldham 11.1, 11.2 393-402

Midterm 10/2: Lectures # 1-9: 4 Topics – See slide 2
Length/Credit Review TBA

http://inst.EECS.Berkeley.EDU/~ee42/

Logical Expressions

Standard logic notation :

AND: "dot" Examples: X = A · B ; Y = A · B · C

OR : "+ sign" Examples: W = A+B ; Z = A+B+C

NOT: "bar over symbol for complement" Example: Z = $\overline{A}$

With these basic operations we can construct any logical expression.

Order of operation: NOT, AND, OR (note that negation of an expression is performed after the expression is evaluated, so there is an implied parenthesis, e.g. $\overline{A \bullet B}$ means $\overline{(A \bullet B)}$.

First Midterm Exam: Topics

- Basic Circuit Analysis (KVL, KCL)
- Equivalent Circuits and Graphical Solutions for Nonlinear Loads
- Transients in Single Capacitor Circuits
- Node Analysis Technique and Checking Solutions

Exam is in class 9:40-10:45 AM, Closed book, Closed notes, Bring a calculator, Paper provided

Logic Function Example

- Boolean Expression: H = (A · B · C) + T

This can be read H=**1** if (A **and** B **and** C are **1) or** T is **1,** or

H is true if all of A,B,and C are true, or T is true, or

The voltage at node H will be high if the input voltages at nodes A, B and C are high or the input voltage at node T is high

Logic Functions

Logic Expression : To create logic values we will define **"True"** , as **Boolean 1** and **"False"** , as **Boolean 0**.

Moreover we can associate a logic variable with a circuit node. Typically we associate logic 1 with a high voltage (e.g. 2V) and and logic 0 with a low voltage (e.g. 0V).

Example: The logic variable H is true (H=**1**) if (A **and** B **and** C are **1) or** T is true (logic **1**), where all of A,B,C and T are also logical variables**.**

Logic Statement: H = 1 if A and B and C are 1 or T is 1.

We use "dot" to designate logical "and" and "+" to designate logical or in switching algebra. So how can we express this as a Boolean Expression?

Boolean Expression: H = (A · B · C) + T

Note that there is an order of operation, just as in math, and AND is performed before OR. Thus the parenthesis are not actually required here.

Logic Function Example 2

You wish to express under which conditions your burglar alarm goes off (**B=1**):

If the "Alarm Test" button is pressed (**A=1**)

OR if the Alarm is Set (**S=1**) AND { the door is opened (**D=1**) OR the trunk is opened (**T=1**)}

Boolean Expression: B = A + S(D + T)

This can be read **B=1 if A = 1 or S=1 AND (D OR T =1),** i.e.

B=1 if {A = 1} or {S=1 AND (D OR T =1)}

or

B is true IF {A is true} OR {S is true AND D OR T is true}

or

The voltage at node H will be high if {the input voltage at node A is high} OR {the input voltage at S is high and the voltages at D and T are high}

Evaluation of Logical Expressions with "Truth Tables"

Truth Table for Logic Expression $H = (A \cdot B \cdot C) + T$

A	B	C	T	H
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	1
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Some Important Logical Functions

- **"AND"** $A \cdot B$ (or $A \cdot B \cdot C$)
- **"OR"** $A + B$ (or $A+B+C+D\ldots$)
- **"INVERT" or "NOT"** not A (or $\overline{A}$)
- **"not AND" = NAND** $\overline{AB}$ (only 0 when A and $B=1$)
- **"not OR" = NOR** $\overline{A+B}$ (only 1 when $A=B=0$)
- **exclusive OR = XOR** $A \oplus B$ (only 1 when A, B differ)
  i.e., $A+B$ **except** $A \cdot B$

Evaluation of Logical Expressions with "Truth Tables"

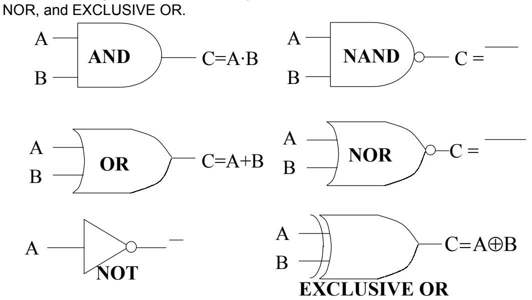
The Truth Table completely describes a logic expression

In fact, we will use the Truth Table as the fundamental meaning of a logic expression.

Two logic expressions are equal if their truth tables are the same

Logic Gates

These are circuits that accomplish a given logic function such as "OR". We will shortly see how such circuits are constructed. Each of the basic logic gates has a unique symbol, and there are several additional logic gates that are regarded as important enough to have their own symbol. The set is: AND, OR, NOT, NAND, NOR, and EXCLUSIVE OR.



The Important Logical Functions

The most frequent (i.e. important) logical functions are implemented as electronic "building blocks" or "gates".

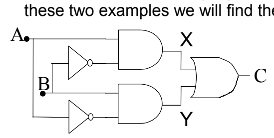
We already know about **AND** , **OR** and **NOT** What are some others:

Combination of above: inverted AND = **NAND,**
inverted OR **= NOR**

And one other basic function is often used: the "EXCLUSIVE OR" … which logically is "or except not and"

Logic Circuits

With a combination of logic gates we can construct any logic function. In these two examples we will find the truth table for the circuit.



It is helpful to list the intermediate logic values (at the input to the OR gate). Let's call them X and Y.

Now we complete the truth tables for X and Y, and from that for C. (Note that X and Y and finally C = X + Y)

A	B	X	Y	C
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Interestingly, this is the same truth table as the EXCLUSIVE OR