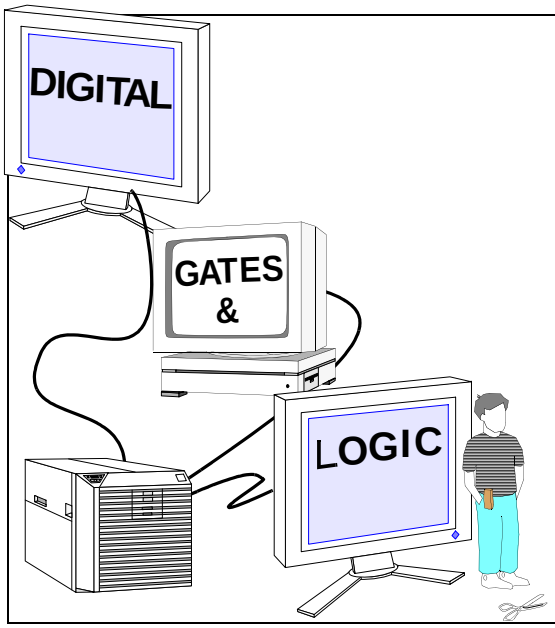
Digital Circuit Engineering



1st Distributive

$XA + XB = X(A + B)$

2nd Distributive

$(X + A)(X + B) = X + AB$

$(X + A)(X + B)(X + C) = X + ABC$

Simplification

$YX + X = X$

$XY + X\overline{Y} = X$

Absorption

$Y + X\overline{Y} = X + Y$

The Most Common Stupid Errors

~~$\overline{X \cdot Y} = \overline{X}\,\overline{Y}$~~

~~$X + 1 = X$~~

Duality

If

$F(a, b, \ldots z, +, \cdot, 1, 0) \equiv G(a, b, \ldots z, +, \cdot, 1, 0)$

Then

$F(a, b, \ldots z, \cdot, +, 0, 1) \equiv G(a, b, \ldots z, \cdot, +, 0, 1)$

Carleton University 2010

■

Digital Gates and Logic

Properties of Digital Signals

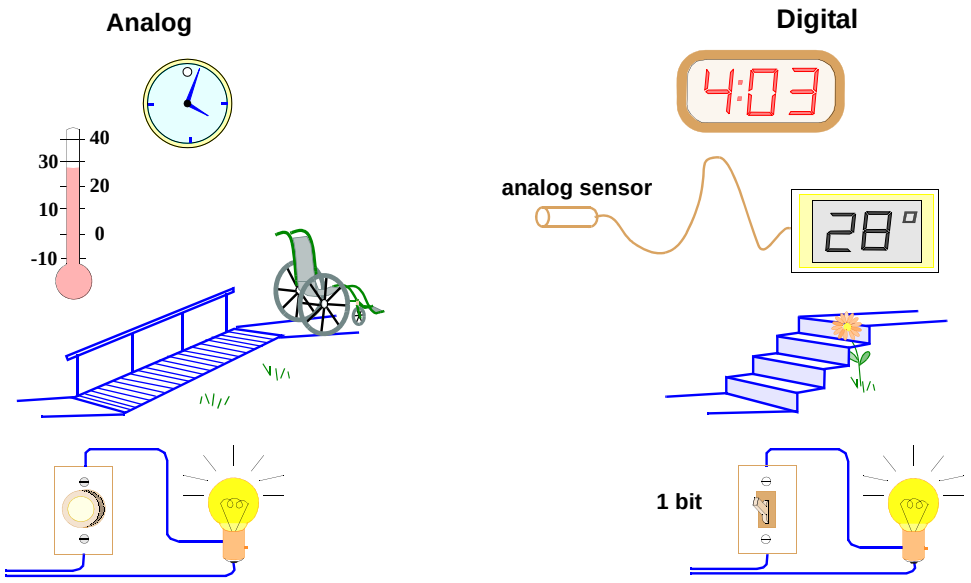
- Compare Analog and Digital

Boolean Algebra

- Gates
- Basic Laws
- Proofs
- Duality
- Simplification, Absorption, Consensus, Swap Rules
- Uses of Duality
- XNORs XORs
- 3-Input gates
- Deriving Circuits from Truth Tables
- Full Adders
- Common Mistakes

General Digital Relevance

Digital Circuits vs. Analog Circuits



General Digital Relevance ■

Digital

Properties of a single digital element.

1. It has two values called, *on-off*, or *high-low*, or *true-false*, or "1"-"0".[1]
2. The value of such an element is often called a *bit*.

Properties of combinations of digital elements

1. The bits can be combined to form *symbols*. These symbols mean whatever people want them to mean.

Examples of the meanings people give symbols.

Binary numbers

0000 means 0	0101 means 6
0001 means 1	0111 means 7
0010 means 2	1000 means 8
0011 means 3	1001 means 9
0100 means 4	1010 means 10
0101 means 5	. . .

ASCII characters

0100,0001 means A
0100,0010 means B
0100,0011 means C
0100,0100 means D
0100,0101 means E
0100,0110 means F
. . .

3-bit two's- complement numbers

100 means -4
101 means -3
110 means -2
111 means -1
000 means 0
001 means 1
010 means 2
011 means 3

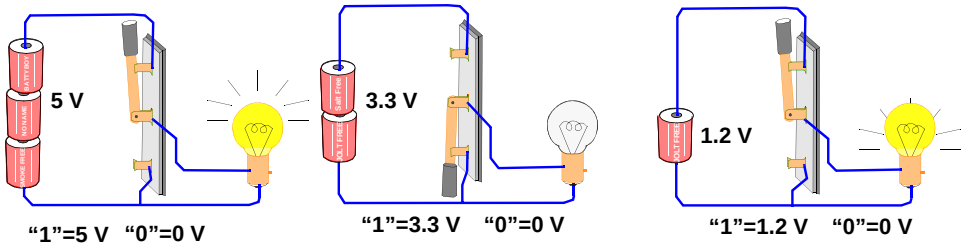
2. Digital values change in steps. Analog values change continuously.
   On an analog thermometer the column goes through all values between temperature changes.
   The digital thermometer changes in steps. The one shown cannot show changes of less than one degree.
3. By using more bits one can make the digital numbers express any accuracy one can get out of the sensor which will be analog. However it is far too easy to make a digital thermometer which displays more digits than can accurately extracted from the analog sensor.

[1] People use a 10 value element all the time. The values are the digits 0,1,2,3,4,5,6,7,8,9.

"1" and "0" are Voltages (usually)

Digital Signals

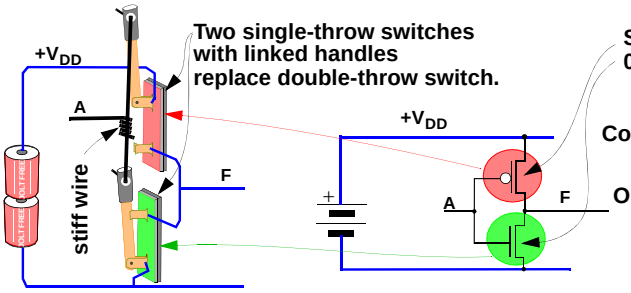
Digital bits ("1" and "0") are represented by a high and a low voltage



For many years a logic "1" was 5 V. A logic "0" was 0 V.

Newer circuits use 3.3, 1.8, 1.5, 1.2, 0.9 or lower voltage as a "1".

Transistor Logic Circuits



Two single-throw switches with linked handles replace double-throw switch.

Switches are silicon transistors typically 0.095 microns long (0.000095 mm)

Control (input) is a logic signal (voltage) "A"

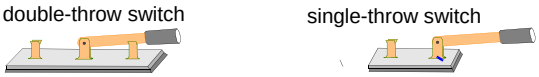
Output is a logic signal (voltage),"F"

Digital Signals

Voltages Represent Signals

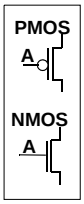
From 1975-90 most digital signals represented a *low*, *false* or "0" value as 0.0 V, a *high*, *true* or "1" signal as 5.0 V. After 1990 lower voltages like 3.3, 2.5. After 2005, 1.8, 1.2 and 0.9 V were used in new integrated circuits designs. Lower voltage circuits have a longer battery life and run cooler, but have other problems like leakage.

Switches



A switches shown above have two positions. They also have a middle position with "both sides off." When transistors are used in pairs as switches, they are never both off, in fact the transistors both partially conduct when A is in the middle position, i.e. A = $V_{DD}/2$. Then F is unpredictable.

Transistor Digital Circuits



In silicon chips, the switches are replaced by transistors. There are two types used in this course:

PMOS transistors which act like closed switches when input A is "0" and open switches when A is "1"

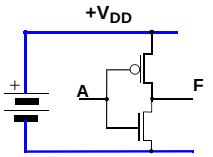
NMOS transistors which act like closed switches when input A is "1" and open switches when A is "0"

The two transistors work like two switches with their handles linked together. When one is **on** the other is **off**.

When the two types of transistors are connected to the same digital input (signal A here), one transistor will act like a closed switch and the other like an open switch.

1-1.• PROBLEM

Complete the table on the right to show how F relates to A.



A	PMOS	NMOS	F
"0"	closed		
"1"			

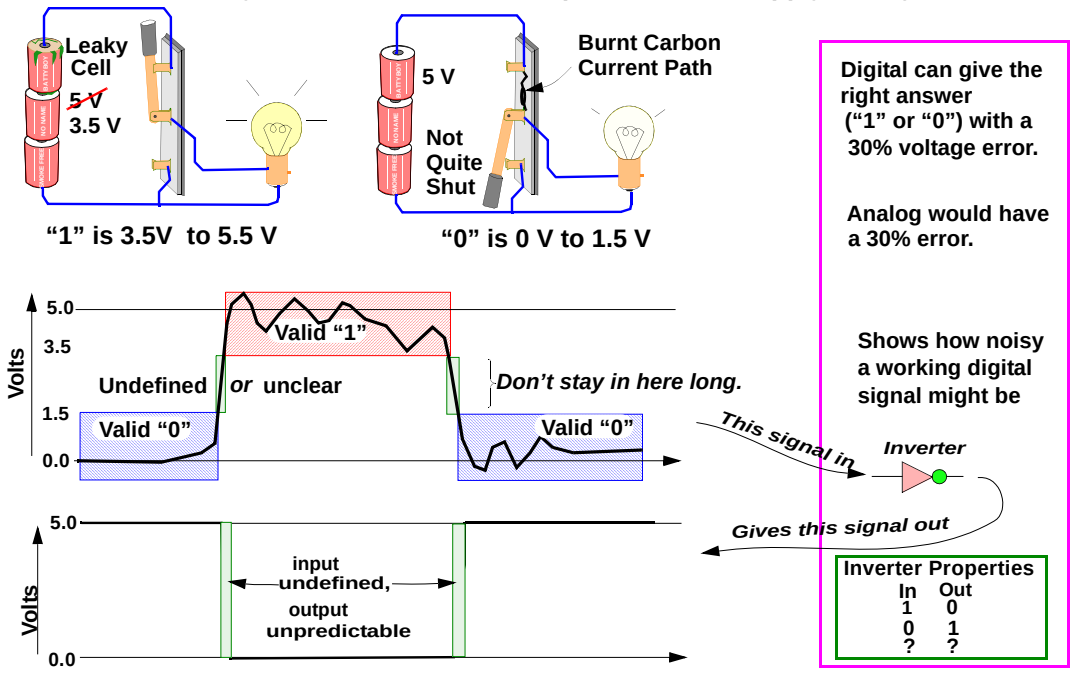
Microns

1 micron = 10^{-6} meters

Properties of Digital Signals

Digital Signals are Very Tolerant

1" and "0" Not Very Sensitive to Noise, Temperature, and Supply Voltage.



Properties of Digital Signals

Digital Less Sensitive to The Electronic Environment

For common (CMOS) digital circuits, the signal may deviate by up to 30% of the supply voltage from the ideal "1" (supply voltage), and "0" (0 V) without causing a bit to be read erroneously.

With analog signals, a 30% error in the signal is a 30% error!

In the *undefined* region, another circuit might read the signal as either 1 or 0.

A typical digital signal

For circuits in the lab, which run from a 5 V supply, a "1" will be a steady signal within a few percent of 5.0 V, and a "0" will be almost exactly 0.00 V. There an AND gate with two 3.6V inputs would see two logical "1" inputs and would give out a 5 V output.

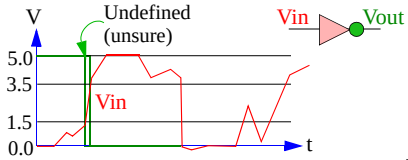
However for signals in noisy environments, such as microprocessor, logic signals may be quite noisy. They still work because of the 30% tolerance on the waveforms.

1-2.•PROBLEM

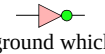
Complete the table shown. Assume CMOS circuits

Supply voltage	5.0V	3.3V	1.2V
Logical 1	Over 3.5 V		
Undefined input, could be read as 0 or 1 or in between	3.5 to 1.5V		
Logical 0			

1-3.•PROBLEM

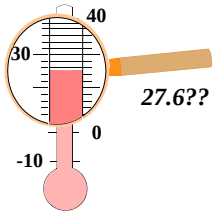


Read ahead and find what the inverter gate  does. Then complete the plot Vout on the graph with Vin.

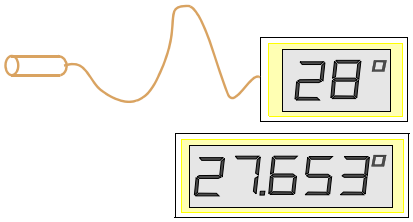
The inverter  has connections to a power supply and ground which are not shown.

Analog vs Digital

Accuracy



0.1% is very good accuracy for analog



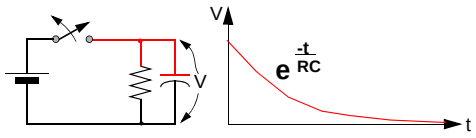
For digital, add more circuitry to get more accuracy.

Subject to accuracy of analog sensor input.

Long Term Storage

Analog circuits store data as voltage on a capacitor.

Max storage time is perhaps 30 minutes



Digital circuits store data in memory

Max storage time is years.

RAM

FLASH Memory

Accuracy

Analog circuits are made of parts with relatively low tolerance. 5% is common. It is very hard to make components accurate to more than 0.1%. Analog voltages can be no more accurate than the components they are built from.

The accuracy of numbers in digital circuits depend on the number of bits. One can increase this indefinitely by adding more bits.

In the digital thermometer shown, the ultimate accuracy is the resolving power of the temperature sensor.

Long-Term Storage

The easiest way to remember a voltage is to store it on a capacitor. Capacitors leak charge which makes the voltage slowly decrease. It is very hard to store a voltage for more than a few minutes.
This is not to say it cannot be done. There are devices called *electrets* that are permanently charged capacitors. They are made by melting the capacitor dielectric and letting it solidify while charged. Also electrons can be injected into the S_iO_2 dielectric of silicon transistrors, for long-term storage.
However these are far less convenient than conventional “leaky” capacitor storage.

Digital memory is very convenient. RAM (random access memory) will remember as long as the power is on. Flash memory which is an electrically erasable memory which can remember with the power off.

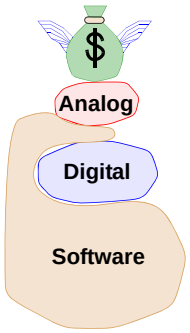
Analog vs Digital

Other Properties, Digital vs Analog

Speed

Fastest circuits are analog.

The highest frequency circuits in your cell-phone and TV are analog.



Design Cost/Time

- **Analog designers have to worry about:**
  noise, power-supply variation, cross-talk between-wires, inaccurate values, temperature variation of component values, ground bounce, clock feed-through, . . .
  To digital designers, these are 2nd order effects.
- **Digital chips work on "first silicon"[1] over 90% of the time.**
  For analog chips this is more like 60%
- **Analog designers need several years experience.**
  Digital designers are productive soon after graduation.
- **There are many more people doing digital design than analog.**
- **Analog circuits are rapidly being redesigned in digital.**

Digital (Underneath) Is Analog

Analog problems become especially important in digital for:
- very fast circuits.|
- very low supply voltages (1.0 V)
- very large circuits (5 million gates per chip)."

[1] First silicon" is the first actual circuit that is built as an integrated circuit. The circuits are simulated, but some problems can slip through.

Other Properties, Digital vs Analog

Speed

The fastest circuits are analog. However every year more and more of these circuits are being redesigned in digital as the speed at which digital circuits can run increases.

Design Cost/Time

Crosstalk is a signal on one wire getting mixed with the signal on another wire by travelling through the small capacitance between the wires.
Ground bounce is when the low voltage return of the power supply bounces above zero volts. It is usually due to inductance in the power supply leads.

Digital circuits suffer from these problems, but they cause no trouble if the interference is less than 30% of the "1" level. It is only when "1" is mistaken for "0" that digital circuits have problems.

Because one can often ignore such problems digital design is usually easier to do than analog.
The designs can be done faster.
They are more likely to be error free.

Digital (Underneath) Is Analog

At very high speeds very small stray capacitance and inductance, which were negligible at lower speeds, become important. Then analog problems have to be considered, and high-speed digital design becomes hard.

When circuits have many million gates per chip a lot of nondigital problems become important. These chips are completely designed in software, because people could never keep track of all the detail.

What makes a good digital designer on large circuits is knowing where the software may give untrustworthy results.

Boolean Algebra

The algebra of 1 and 0

Created by
George Boole 1847
Claude Shannon 1939

Two Valued Algebra

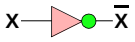
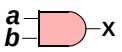
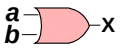
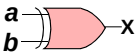
Values are called "1" and "0"
or True and False
or High and Low

Variables.

A variable X can have only two values, 0 or 1.

~~X = 2~~
~~X = -1~~
~~X = 3+j6~~
X = 1
X = 0

Operations

Symbol	Schematic	Truth Table	Compact Truth Table
Complement, Not, or Inverse $\overline{X}$ is the opposite of X $\overline{X}$		X | $\overline{X}$ 0 | 1 1 | 0	Value of X for a=0, b=1
AND X is 1 if *a* AND *b* are both 1. $a \cdot b$ ab		a b | X 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1	a | b: 0 1 / 0: 0 0 / 1: 0 1
OR X is 1 if *a* OR *b* or both are 1. $a + b$		a b | X 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1	a | b: 0 1 / 0: 0 1 / 1: 1 1
X-OR, XOR, Exclusive OR X is 1 if exactly one of *a* or *b* is 1. $a \oplus b$		a b | X 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0	a | b: 0 1 / 0: 0 1 / 1: 1 0

Variables

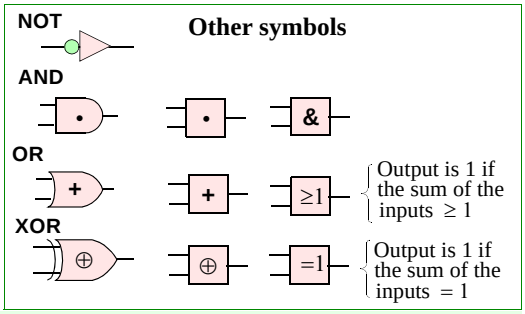
A Boolean variable can have the value "1" or "0". Later we will give them a "don't care" value of "-" or "*d*". Where "*d*" means the variable could have either a "0" or a "1" and it will not change the result of interest.

Gates

The operators shown can be built with simple transistor circuits called gates. The circuit for a NOT gate is shown under See "Digital Signals" on page 4. These gates are connected together to construct digital circuits.

Other Notations

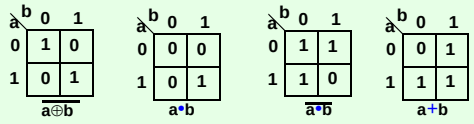
NOT	$\overline{x}$, ¬ x, !x, ~x (Matlab, Verilog)
AND	a·b, a∩b, a^b, a&b (Matlab, Verilog)
OR	a + b, a∪b, a∨d, a|b (Matlab, Verilog)
X-OR	a ⊕ b, a+b, a^b (Verilog)



Compact Way to Make a Truth Table

The conventional table shown in *a*) can be replaced with *b*) called a map. In the map, the inputs are around the outside, and the value of the output is put in the interior squares.

This "map" form of table can be made for other gates.



a	b	a⊕b
0	0	0
0	1	1
1	0	1
1	1	0

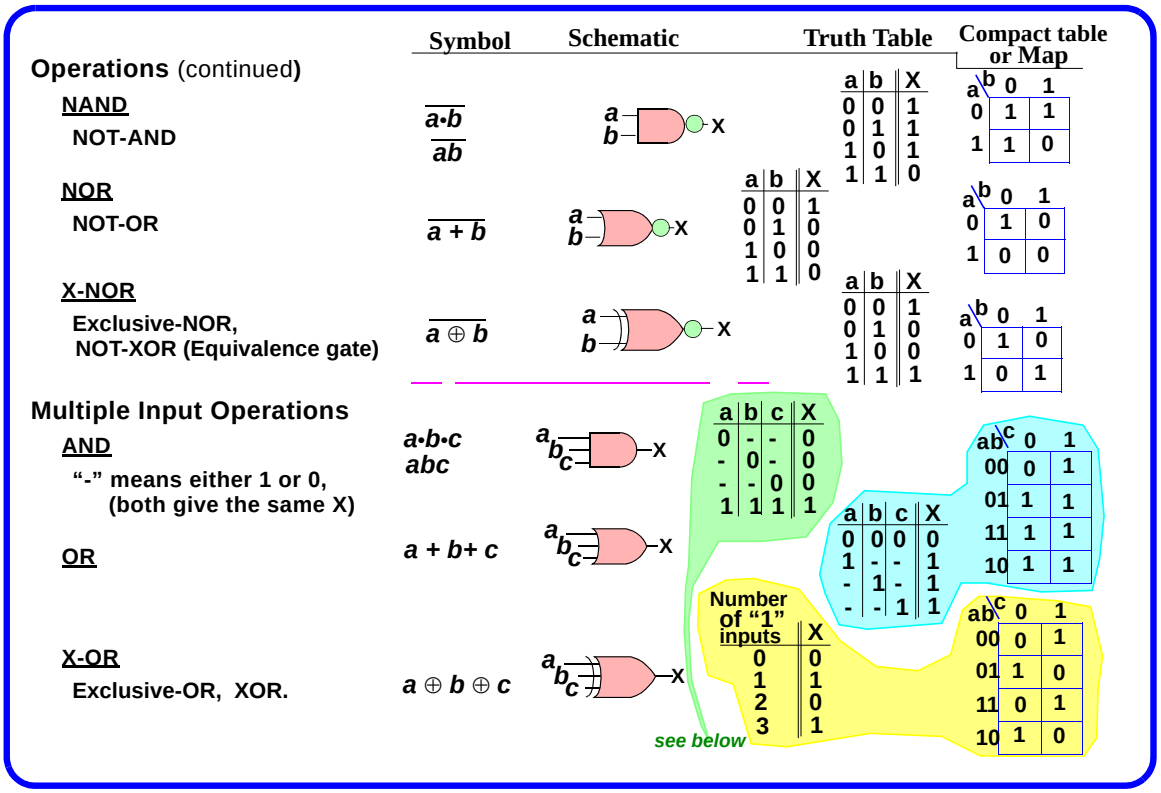
a) Conventional table

a \ b	0	1
0	0	1
1	1	0

a ⊕b

b) Map

Boolean Algebra; Gates



Boolean Operations

Common Mistake

$\overline{a \cdot b}$ is **not** the same as $\overline{a} \cdot \overline{b}$

1-4.•PROBLEM

Is $\overline{a+b}$ the same as $\overline{a} + \overline{b}$? If not, give a counter example.

X-NOR or "Equivalence" gate

This gate gives a "1" output if the two inputs are equal. Thus it is sometimes called an equivalence or coincidence gate.

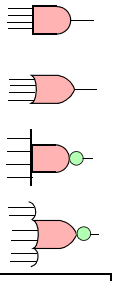
The compact table (map) for the 3-input AND

ab \ c	0	1
00	0	0
01	0	0
11	0	1
10	0	0

Multiple Input Operations

N-input Gates

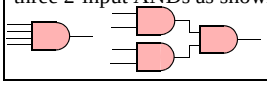
In theory a gate can have any number of inputs. For transistor circuits, the gates become very slow with more than four inputs. However theoretical circuits will often have more than four inputs. They can easily be converted to smaller gates. The figure shows how lines may be added to the gate symbol to give the inputs a bit more spacing on a diagram.



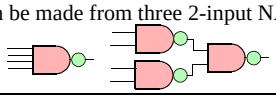
With transistors it is easier to build NAND and NOR than AND and OR. However people make fewer mistakes if the design is done with AND and OR. It is easy to convert an AND/OR circuit to NAND/NOR as a final design step.

1-5.•PROBLEM .

A 4-input AND can be made from three 2-input ANDs as shown.



Prove or disprove that a 4-input NAND can be made from three 2-input NANDs



All Possible 2-Input Gates

a b	0	a•$\overline{b}$	a•b	a	$\overline{a}$•b	a⊕b	b	a+b
0 0	0	0	0	0	0	0	0	0
0 1	0	0	0	0	1	1	1	1
1 1	0	0	1	1	0	0	1	1
1 0	0	1	0	1	0	1	0	1
Symbol								
Relation	0	a>b			a<b	a≠b		

a b	$\overline{a+b}$	$\overline{b}$	$\overline{a⊕b}$	a+$\overline{b}$	$\overline{a}$	$\overline{a•b}$	$\overline{a}$+b	1
0 0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1
1 1	0	0	1	1	0	0	1	1
1 0	0	1	0	1	0	1	0	1
Symbol								V_{DD}
Relation			a=b	a≥b			b≥a	1

Gates easily made from transistors are NAND, NOR and NOT.

Possible Gates

Common gates

AND, OR, NAND, NOR, XOR and XNOR six of the 16 possible gates.
Z(the output) = 1, 0, a, b, $\overline{a}$, or $\overline{b}$ are another six.
The final four are a$\overline{b}$, $\overline{a}$b, $\overline{a}$ + b, and a + $\overline{b}$

A buffer gate 

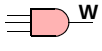
Logically the buffer gate is like a piece of wire. What goes in comes out. However the buffer has gain. If a weak "1", say 4 V on a system with a 5 V supply, is sent into the buffer it will boost the "1" to 5 V.

1-6.•PROBLEM

How many different types of 3-input gates are there? Look at how many possible input combinations there are.

1-7.•PROBLEM

a	b	c	W	X	Y	Z
0	0	0	0	0	1	0
0	0	1	0	1	0	0
0	1	1	0	1	0	0
0	1	0	0	1	0	1
1	0	0	0	1	0	0
1	0	1	0	1	0	0
1	1	1	1	1	0	0
1	1	0	0	1	0	0



The truth tables for four 3-input gates are shown on the left.
By extrapolating the 2-input symbols, sketch the symbol for the gate that gives the output X, and the gate that gives Y.

The solution for W is shown.

1-8.• PROBLEM

Sketch the gate that gives Z. This is harder you might want to check the page "Common Three-Input Gates," Slide 24

Boolean Algebra; Laws

Basic Laws

Basic Laws with Zeros and Ones		2nd Form (Dual)		
X + 0 = X	*(B1)* or just *(B)*	X•1 = X	*(B2)*	Use B for any of these four
X + 1 = 1	*(B3*	X•0 = 0	*(B4*	
Idempotent				
A variable is unchanged by operating with itself.				
X + X = X	*(I1)*	X•X = X	*(I2)*	*or (I) for either*
Double Negative				
$(\overline{\overline{X}}) = X$	*(N^2)*	*No 2nd form*		
Negation Laws				
$X + \overline{X} = 1$	*(N1)*	$X \cdot \overline{X} = 0$	*(N2)*	*or (N) for either*
Commutative Laws				
X + Y = Y + X	*(C1)*	X•Y = Y•X	*(C2)*	*or (C) for either*
Associative Laws				
X + (Y + Z) = (X + Y) + Z	*(A1)*	X•(Y•Z) = (X•Y)•Z	*(A2)*	*or (A) for either*
Distributive Laws				
(X + Y) Z = XZ + YZ	*(D1)*	X Y + Z = (X + Z)(Y + Z)	*(D2)*	*or (Dstrange)*

Remember This

Basic Laws

These laws are fairly obvious from the description of the gates (operators). The only strange on is the second distributive law *(D2)*. This one has to be memorized because it is very useful as well as strange.

We will treat all these laws as axioms except (D2). This will be proven given (D1).

For Mathematicians

All of the above laws can be proven by using only a few basic axioms.

For example one set of basic axioms suggested by Wakerly[1], is:

X is either 0 or 1.

If X = 0, $\overline{X} = 1$;	If X = 1, $\overline{X} = 0$.	
0·0 = 0,	1·0 = 0 = 0·1,	1·1 = 1
0+0 = 0,	1+0 = 1 = 0+1,	1+1 = 1

We will use these axioms to prove X + X = X *(I1)*

X must be either 0 or 1.
If X=0, then X+X = 0+0 = 0; thus X+X = X for X=0.
If X=1, then X+X = 1+1 = 1, again X+X = X for X=1.

Since these are the only two possible values of X, we have proven X + X = X

- qed[2] -

Keep Your Proofs Reasonably Rigorous

For the proofs on the next pages we write one side, say the left-hand side (LHS) at the top. Then we work on it until we can change it to the other, the RHS. Do not pretend they are equal and operate on both sides at once.

1. Wakerly, John F., *Digital Design Principles and Practices, 3rd. Ed.*, Prentice Hall, Englewood Cliffs NJ., 1998

2. *quod erat demonstrandum*

Boolean Algebra

Using Boolean Algebra

Prove: $AB + B = B$ — **Simplification law** *(S)*

Proof:

LHS= $AB + B = AB + B\cdot 1$	*from (B)* $X \cdot 1 = X$
$= (A + 1)B$	*from (D1)* $(X + Y)Z = XZ + YZ$
$= (1)B$	*from (B)* $X + 1 = 1$
$= B$ $= RHS$	*from (B)* $X \cdot 1 = X$

Prove: $AB + C = (A + C)(B + C)$ — **Second distributive law** *(D2)*

Proof:

RHS = $(A + C)(B + C) = Q(B + C)$	*let* $Q = (A + C)$
$= QB + QC$	*from (D1)* $(X + Y)Z = XZ + YZ$
$= (A + C)B + (A + C)C$	since $Q = (A + C)$
$= AB + CB + AC + CC$	*from (D1)*
$= AB + CB + AC + C$	*from (I)* $XX = X$
$= AB + CB + C$	*use (S)* $XY + X = X$
$= AB + C = LHS$	*use (S)* $XY + X = X$

Simplify: $A\overline{B} + AB + \overline{A}B$

$A\overline{B} + AB + \overline{A}B = A\overline{B} + AB + AB + \overline{A}B$	*from (I)* $X + X = X$
$= A(\overline{B} + B) + (A + \overline{A})B$	*from (D1)* $(X + Y)Z = XZ + YZ$
$= A(1) + (1)B$	from *(N)* $X + \overline{X} = 1$
$= A + B$	*from (B)* $X \cdot 1 = X$

Examples and Problems

Simplification Rule

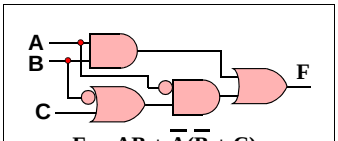
$x + xy = x$ is one of the most useful rules, and fairly easy to remember.

1-9.•EXAMPLE

Simplify: $F = \overline{A}\cdot\overline{B}\cdot\overline{C} + AB\overline{C} + \overline{A}BC + \overline{A}\cdot\overline{B}C + ABC$ Reduce to five letters (literals).

$\overline{A}\cdot\overline{B}\cdot\overline{C} + AB\overline{C} + ABC + \overline{A}\cdot\overline{B}C + \overline{A}BC = AB\overline{C} + ABC + \overline{A}\cdot\overline{B}\cdot\overline{C} + \overline{A}\cdot\overline{B}C + \overline{A}BC$	*(A)* (Rearrange terms)
$= AB(\overline{C} + C) + \overline{A}\cdot\overline{B}(\overline{C} + C) + \overline{A}BC$	*(D1)* (ax +ay=a(x + y))
$= AB + \overline{A}\cdot\overline{B} + \overline{A}BC$	*(N), (B)* x1=1
$= AB + \overline{A}(\overline{B} + BC)$	*(D1)*
$= AB + \overline{A}(\overline{B} + \overline{B}C + BC)$	**(S)** (b +bc=b)
$= AB + \overline{A}(\overline{B} + C(\overline{B} + B)$	*(D1), (B)*
$= AB + \overline{A}(\overline{B} + C)$	*(N)* $(\overline{b} + b = 1)$

Draw a gate level circuit which implements the result. (Done on the right)



$F = AB + \overline{A}(\overline{B} + C)$

1-10.•PROBLEM Simplify $AB + BCD + BC + A\overline{C}$
Reduce to four letters (literals)[1].
Draw a gate level circuit.

1-11.•PROBLEM Simplify $(A + \overline{B}C)(B + C)(B + \overline{C})$

1-12.•PROBLEM Prove $(\overline{A} + C)(B + C)(B + \overline{A}) = \overline{A}B + BC + C\overline{A}$

[1] Reducing to 5 letters is easy. Reducing to 4 letters is easy if you look ahead in theses notes and use a Karnaugh map.

Boolean Algebra: Exhaustive Proofs

Good for a small number of inputs (up to 4 or 5)

Make a table

a) List all possible inputs

b) Calculate all values of the left hand side

c) Calculate all values of the right hand side

d) See if they agree

This only works because A, B, and C have only 2 values. Its no good if they represent *integers* or *real* numbers.

Alternate Proof

Second distributive law:

$(A + C)(B + C) = AB + C$

Proof: Use a truth table and prove for all cases

			LHS		RHS
ABC	(A + C)	(B + C)	(A+C)(B+C)	AB	AB + C
000	0	0	0	0	0
001	1	1	1	0	1
011	1	1	1	0	1
010	0	1	0	0	0
100	1	0	0	0	0
101	1	1	1	0	1
111	1	1	1	1	1
110	1	1	1	1	1

The two sides are equal for all combinations of ABC.

Trying All Combinations

The method is good for proofs, with not very many variables.

It is not good for simplification.

Exhaustive Problems

1-13.•PROBLEM WITH XORS

Prove that $A \oplus \overline{B} = \overline{A} \oplus B = \overline{A \oplus B}$

1-14.•PROBLEM Prove that $\overline{C} + BC = \overline{C} + B$

Prove it (a) algebraically and (b) with a truth table.

1-15.•PROVE

$Ab + bc + c\overline{A} = Ab + c\overline{A}$ Consensus theorem

This is hard to prove algebraically.[1]

[1] Hint $bc = bcA + bc\overline{A}$.

Boolean Algebra

Exhaustive Proof Using Maps (Compact Tables)

$(A + C)(B + C) = AB + C$

Proof: Using the compact tables

ABC	(A + C)	(B + C)	LHS (A+C)(B+C)	AB	C	RHS AB + C
000	0	0	0	0	0	0
001	1	1	1	0	0	1
011	1	1	1	0	0	1
010	0	1	0	0	0	0
100	1	0	0	0	1	0
101	1	1	1	0	1	1
111	1	1	1	1	1	1
110	1	1	1	1	1	1

The two sides are equal for all combinations of ABC.

(A + C)

BC \ A	0	1
00	0	1
01	1	1
11	1	1
10	0	1

(B + C)

BC \ A	0	1
00	0	0
01	1	1
11	1	1
10	1	1

(A+C)(B+C)

BC \ A	0	1
00	0	0
01	1	1
11	1	1
10	0	1

AB

BC \ A	0	1
00	0	0
01	0	0
11	0	1
10	0	1

C

BC \ A	0	1
00	0	0
01	1	1
11	1	1
10	0	1

AB + C

BC \ A	0	1
00	0	0
01	1	1
11	1	1
10	0	1

With 3 inputs, table has 8 entries
With 5 inputs, table has 32 entries

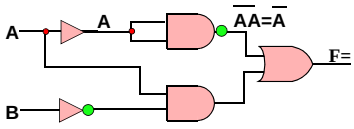
With 4 inputs, table has 16 entries
With 6 inputs, table has 64 entries

With 7 inputs, 128 entries, algebraic proofs start to look easier.

Circuit Problems

1-16.•PROBLEM: FOR CIRCUIT SHOWN

Simplify:-



1-17.•PROBLEM

Simplify to 4 literals (letters)-

AD +AE + AF

1-18.•PROBLEM

Implement:-

ABCD + ABCE + ABCF

with one multi-input OR and one multi-input AND.

1-19.•PROBLEM

Simplify -

(A + B)(A + C)(A + D)

1-20.•PROBLEM

Implement:-

(A+B+C+D)(A+B+C+E)(A+B+C+F)

with one multi-input OR and one multi-input AND[1]

[1] Hint; try Problem 1-19.• first. It is very similar.

Duality

Duality Principle

Take any valid Boolean identity

Substitute $\bullet \Longleftrightarrow +$ **and** $0 \Longleftrightarrow 1$

The result is another valid Boolean identity.

Practical detail

Put brackets around all AND terms before doing the substitution.

Example

$x(y + z) = xy + xz$ *Valid identity; (D1)*

$x(y + z) = (xy) + (xz)$ *Put brackets around the ANDed terms*

$x+(yz) = (x + y)(x + z)$ *Substitute* $\bullet \Longleftrightarrow +$

Result is (D2), 2nd distributive law, another valid identity.

Example

$x + 1 = 1$ *Valid identity; (B)*

$x \cdot 0 = 0$ *Substitute* $\bullet \Longleftrightarrow +$ **and** $0 \Longleftrightarrow 1$

Result is *(B)*, a valid identity.

WARNING

Never Never Never! **say an expression is equal to its dual.**

Duality Principle

This is a principle not an equality

It is **WRONG WRONG WRONG** to say an expression equals its dual[1].

Example:

abc is the dual of **a+b+c**

But clearly **abc** ≠ **a+b+c**

Do not do this:

F = a(bc + e) + def

= [a((bc) + e)] + (dqf) put brackets around ANDs *This is still equal to F*

= [a+((b+c)e)](q + e + f) take dual *You cannot just write in another equal sign*

F_{dual} = [a+((b+c)e)](q + e + f) take dual *It is correct to call this F_{dual}*

1-21.•PROBLEM

Two expressions for XNOR are: $\bar{a}\bullet\bar{b} + a\bullet b = (a+\bar{b})(\bar{a}+b)$

Find the dual identity.

[1] There are special cases when they are equal, but you had better be able to prove it if you put in that equal sign

Duality: Duals of Rules

Half the Basic Laws are Duals of the Other Half

Laws with Zeros and Ones			Dual Form	
(B1)	X + 0 = X	• ⟺ + and 0 ⟺ 1	X•1 = X	*(B2)*
(B3)	X + 1 = 1	• ⟺ + and 0 ⟺ 1	X•0 = 0	*(B4)*

or just *(B)* for any of these four

Indempotent			
(I)	X + X = X	• ⟺ +	X•X = X *(I)*
Double Negative			
	$(\overline{\overline{X}}) = X$		*Self dual*
Negation Laws			
(N1)	$X + \overline{X} = 1$	• ⟺ +	$X \cdot \overline{X} = 0$ *(N2)*
Commutative Laws			
	X + Y = Y + X	• ⟺ +	X•Y = X•Y *(C)*
Associative Laws			
	X + (Y + Z) = (X + Y) + Z	• ⟺ +	X•(Y•Z) = (X•Y)•Z *(A)*
Distributive Laws			
(D1)	(X + Y) Z = XZ + YZ	• ⟺ +	X Y + Z = (X + Z)(Y + Z) *(D2)*

Duals

Self Duals

Some formulas are their own duals.

Examples

$(\overline{\overline{x}}) = x$

The three input majority (carry) circuit obeys:

ab + bc +ca = (a + b)(b + c)(c + a)

Taking the dual merely interchanges the right and left sides.

1-22.•PROBLEM

Prove ab + bc + ca = (a + b)(b + c)(c + a)

1-23.•PROBLEM

Show that the $\overline{x}A + xB = (x + A)(\overline{x} + B)$, later called the "swap rule", is self dual, if one substitutes x for $\overline{x}$ in the dual rule.

General property of self-dual functions

Self dual functions have $\overline{f(a,b,c)} = f(\overline{a},\overline{b},\overline{c})$

There are 2^{n-1} self-dual function with n inputs, among them the carry and add circuits.[1]

[1] CMOS gates of self-dual functions can have symmetric NMOS and PMOS parts. This is useful in designing the insides of gates.

Duality: Duals of Rules

Other Rules and Their Duals

Simplification Rules *(S1, S2)*

X + (X•Y) = X • ⟺ + **X•(X+Y) = X**

Proof:

LHS = X + (X•Y)
= X•1 + X•Y *(B2) X •1 = X*
= X(1 + Y) *(D1) XZ + XY = X(Z + Y)*
= X = *RHS* *(B2,B3) 1•X=X, X + 1 = 1,*

Proof: (*Each line is the dual of the line on the left*)

LHS = X•(X+Y)
= (X+0)•(X+Y) *(B1) X + 0 = X*
= (X+0•Y) *(D2) (X + Z)(X + Y) = X + ZY*
= X = *RHS* *(B1,B4) X + 0 = X, 0•Y = 0*

If You Prove Half of the Boolean Algebra Theorems
Duality Will Give You The Other Half:

- **(S1) and (S2) are duals;**
- **Each line in their respective proofs are duals.**
- **If one proof references a rule (like D1), the dual proof will reference the dual rule (D2).**

Memorize

X + X•anything = X

The dual gives you X(X + anything) = X

Simplification Rule

Simplification x + xy = x (*S1*) • ⟺ + x(x + y) =x (*S2*)

People who complain they can't finish their exam usually don't know the time they can save using this rule. Look for terms like:

a + **a**x
abc + **abc**xy
a + **a**bcdefgh
xy + ac(t+m)(ab+x)**xy**

Examples:

abcd + abcde = abcd (*S1*) Simplification
abe + abc + e + bc = e + bc (*S1*) Simplification, applied twice

1-24.•PROBLEM
Simplify:
$x + \bar{a}xy + abx(z+uv) + axy$

1-25.•PROBLEM:
Simplify $x\bar{y} + xz\bar{y}$

1-26.•PROBLEM:
Simplify $x + xzy + \bar{x} + x\bar{y}z + \bar{x}\bar{y}\bar{z} + y\bar{z}$ Designed to mislead

Duality: Duals of Rules

Other Rules and Their Duals

Simplification Rules *(S1, S2)* *(For comparison)*

$X + (X \cdot Y) = X$ $\quad \cdot \Longleftrightarrow + \quad$ $X \cdot (X + Y) = X$

Absorption Rules *(Ab1, Ab2)*

$X + (\overline{X} \cdot Y) = X + Y$ $\quad \cdot \Longleftrightarrow + \quad$ $X \cdot (\overline{X} + Y) = (X \cdot Y)$

Proof:

LHS $= X + (\overline{X} \cdot Y)$ $= (X + X) \cdot (X + Y)$ $= 1 \cdot (X + Y)$ $= X + Y \quad = RHS$	*(D2)* $X + UY = (X + U)(X + Y)$ *(N1)* $X + \overline{X} = 1$ *(B2)* $1 \cdot Z = Z$

Proof: (*Each line is the dual of the line on the left*)

LHS $= X \cdot (\overline{X} + Y)$ $= X\overline{X} + XY)$ $= 0 + XY$ $= X \cdot Y \quad = RHS$	*(D1)* $(X + Y)Z = XZ + YZ$ *(N2)* $X\overline{X} = 0$ *(B1)* $0 + Z = Z$

Note:

- **(Ab1) and (Ab2) are duals.**
- **Each line in their respective proofs are duals.**
- **Note when the proof for (Ab1) used (D1), the proof for (Ab2) used the dual rule (D2).**

Simplification and Absorption

These two simple useful rules can save you a lot of work when doing problems. Look for them!

Simplification $x + xy = x$ *(S1)*

A second rule, which is also almost as easy to use, is the absorption rule.

Absorption $x + (\overline{x}y) = x + y$ *(Ab1)*

Examples:

$abcd + \overline{a}bcde = bcd(a + e) = abcd + bcde$ *(Ab1)* $a + (\overline{a}y) = a + y$

$abe + abcd\overline{e} = ab(e + cd\overline{e}) = ab(e + cd)$ (*Ab1*) Absorption

$abe + abc + \overline{e} + bc = ab + abc + \overline{e} + bc = ab + \overline{e} + bc$ (*Ab1*) Absorption and (*S1*) Simplification

1-27.•PROBLEM:

Simplify $x\overline{y} + xzy$

1-28.•PROBLEM:

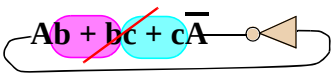
Simplify[1] $x\overline{y} + xzy + \overline{z}y$

[1] Apply (Ab1) to the second and third terms, get $x + \overline{z}y$. Applying (Ab1) to the first and second terms doesn't do as much.

Consensus & Swap Rules and Their Duals

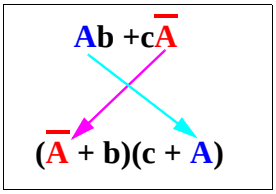
Other Dual Rules

Consensus Theorems *(Con1, Con2)*



$Ab + bc + c\overline{A} = Ab + c\overline{A}$ • $\Longleftrightarrow$ + $(A + b)(b + c)(c + \overline{A}) = (A + b)(c + \overline{A})$

LHS = $Ab + bc + c\overline{A}$
= $Ab + bc1 + c\overline{A}$ *(B) x•1 = x*
= $Ab + bc(A + \overline{A}) + c\overline{A}$ *(N) $\overline{x}$ +x = 1*
= $Ab + bcA + bc\overline{A} + c\overline{A}$ *(D1)*
= $Ab + Abc + c\overline{A}b + c\overline{A}$ *(C) xz=zx*
= $Ab + c\overline{A}$ = *RHS* *(S1) x+xy=x*



Swap *(Swap)* Rule

$(\overline{A} + b)(A + c) = Ab + \overline{A}c$ • $\Longleftrightarrow$ + $\overline{A}b + Ac = (A + b)((\overline{A} + c)$

LHS = $(\overline{A} + b)(A + c)$
= $\overline{A}A + \overline{A}c + bA + bc$ *(D1) multiply out*
= $\overline{A}c + bc + bA$ *(N) $\overline{A}$•A = 0*
= $\overline{A}c + bA$ = *RHS* *(Con1) Consensus*

Swap and Consensus

The consensus rule is often hard to see. Fortunately, Karnaugh maps, which we do later, will allow one to forget consensus for simple (up to 4 variable) problems.

1-29.• PROBLEM: Simplify
$ABCE + CE\overline{D} + ABD$

1-30.•PROBLEM: Simplify

a) $(C+E)X + (A+B)(C+E) + (A+B)\overline{X}$

b) Simplify
$(\overline{C+E})X + (A+B)(C+E) + (A+B)X$

1-31.•PROBLEM

Complete the proof of *(Con2)*.

1-32.•PROBLEM
Why is the dual of the swap rule not another rule?

1-33.•PROBLEM: Find the **Error** in this students simplification of $(\overline{C+E})X + (A+B)X + (A+B)(C+E)$

$(C+\overline{E})X + (A+B)X + (A+B)(C+E)$
$= (\overline{C+E})X + [(A+B) + X(C+E)]$ using D2
$= (\overline{C+E})X + X(C+E) + (A+B)$ by commuting
$= X[(\overline{C+E}) + (C+E)] + (A+B)$ using D1
$= X + (A+B)$ since $a + \overline{a} = 1$

Uses of Duality: Summary

Uses of Duality

1. Learn half the rules, remember them all!

2. Do hard algebra (usually things with brackets) in a more familiar way

Example

Simplify $F = (\bar{a} + d)(a + b)(a + d)$

How would you do this?

Start

$F = (\bar{a} + d)(a + b)(a + d)$

Take the dual

$F_{dual} = \bar{a}d + ab + ad$

$= (\bar{a} + a)d + ab$

$= (1)d + ab$

$= d + ab$

(D1) *(N)* $x+\bar{x} = 1$ *(B)* $1x = x$

By taking the dual, use the familiar distributive law. Otherwise use the strange D2

Put in brackets ready to take reverse dual

$F_{dual} = d + (ab)$ **put in brackets**

Take dual back

$F = d(a + b)$ **Valid reduced identity.**

Thus $(\bar{a} + d)(a + b)(a + d) = d(a + b)$

Uses of Duality

Factoring in algebra is difficult. Using duality makes it easier because the dual is already factored. On the dual, one multiplies out, which is much easier for most people. At the end of multiplying out, the dual is in factored form. Taking the dual of the dual takes us back to the original expression, only now it is in factored form.

This works because-

if one has a valid identity its dual is a valid identity.

Another example which has a simpler dual.

Simplify $X + (\overline{X} \cdot Y)$

Most people do not see how to start the above. It may help to take the dual.

Take the dual $X \cdot (\overline{X} + Y)$

This multiplies out to $X\overline{X} + Y = Y$

A Third Use for Duality

Finding new formulas, as in PROBLEM 1-38.▪ where another new formula for XOR is found by taking the dual of a new formula for XNOR.

1-34.▪ PROBLEM: Common error when calculating the dual

$F = a + b(c + d)$ (*take dual*)

$F_{dual} = ab + (cd)$

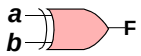
What is wrong with the above dual calculation?

Applications of XOR, XNOR

XOR, X-OR, Exclusive OR

F is 1 if exactly one of *a* or *b* is 1.

$a \oplus b$



a	b	F
0	0	0
0	1	1
1	0	1
1	1	0

a \ b	0	1
0	0	1
1	1	0

Commutative

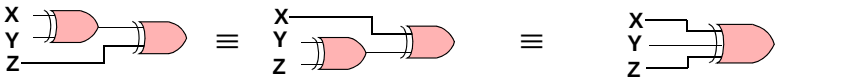
$X \oplus Y = Y \oplus X$

Associative

$X \oplus (Y \oplus Z) = (X \oplus Y) \oplus Z$

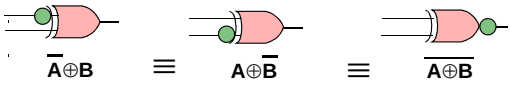
No brackets needed

Thus one can write $X \oplus Y \oplus Z$



Useful Properties

Inverting any lead changes XOR ⇒XNOR



A	B	A⊕B	$\overline{B}$	$A \oplus \overline{B}$
0	0	0	1	1
0	1	1	0	0
1	0	1	1	0
1	1	0	0	1

Making one input "1" makes XOR an inverter"

c	x	c⊕x
0	0	0=x
0	1	1=x
1	0	1=$\overline{x}$
1	1	0=$\overline{x}$

XOR and XNOR

Distributive Law:

only law like D1 holds

$(X \oplus Y)Z = (XZ) \oplus (YZ)$ (D1x)

Law like D2 does not

~~$(X \oplus Y) \oplus Z = (X \oplus Z)(Y \oplus Z)$~~

1-35.•PROVE THAT: $(X \oplus Y)Z = XZ \oplus YZ$

1-36.•PROVE THAT: $\overline{A} \oplus B = A \oplus \overline{B} = \overline{A \oplus B}$

1-37.•PROBLEM

The formula for XNOR is given on the next page as $\overline{a} \cdot \overline{b} + a \cdot b$

Prove: $\overline{a} \cdot \overline{b} + a \cdot b = (a + \overline{b})(\overline{a} + b)$

1-38.•PROBLEM

Use duality on Problem 1-21.• to find a similar alternate form for $\overline{a} \cdot b + a \cdot \overline{b}$, the XOR formula.

1-39.•PROBLEM

Use the swap rule on $\overline{a} \cdot b + a \cdot \overline{b}$ to find an alternate way to make an XOR out of AND, OR and NOT gates. Sketch the circuit

Deriving Formulas From Truth Tables

The Formulas for XOR and XNOR

a	b	(a⊕b)	$\overline{(a \oplus b)}$
0	0	0	1 $\bar{a}\cdot\bar{b}$
0	1	1 $a\cdot\bar{b}$	0
1	0	1 $\bar{a}\cdot b$	0
1	1	0	1 $a\cdot b$

Equation for when (a⊕b) is "1"

$(a \oplus b) = \bar{a}\cdot b + a\cdot\bar{b}$

(a⊕b) is "0" for all other combinations

Equation for when $\overline{(a \oplus b)}$ is "1"

$\overline{(a \oplus b)} = \bar{a}\cdot\bar{b} + a\cdot b$

1. Write out the Truth Table
2. Look for where the result is "1".
3. Write down the letters corresponding to the inputs
   Thus a "0" in the "a" column is written as "$\bar{a}$".
   a "1" in the "a" column is written as "a".
4. AND the letters to make a term like $\bar{a}\cdot b$.
5. The equation is the OR of all the terms where the output is "1".

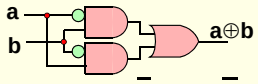
XOR-XNOR FORMULA

$(a \oplus b) = \bar{a}\cdot b + a\cdot\bar{b}$

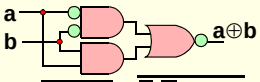
$\overline{(a \oplus b)} = \bar{a}\cdot\bar{b} + a\cdot b$

Actual XOR Gates

These are complex combinations of AND, OR, NAND and NOR.



$(a \oplus b) = \bar{a}\cdot b + a\cdot\bar{b}$



$(a \oplus b) = \overline{(a \oplus b)} = \bar{a}\cdot\bar{b} + a\cdot b$

Obtaining Formulas from Truth Tables

This method will give formulas in what is called OR of ANDs or (SUM of PRODUCTs) form.
These formulas are letters or their inverses ANDed together and the resulting terms ORed together[1], like

$a\bar{b}c + acd + \bar{c}f + ace + ...$

We need only find all the terms that make the final answer "1".
If no term make the formula "1", then it defaults to "0".

To Find the Formula

Go through the truth table and writes down the terms that make the result "1".
Then OR all these terms together

That is the desired formula.

XOR and XNOR gates

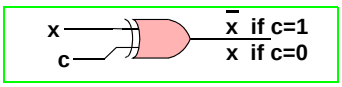
These are more complex to implement than most other two-input gates. There are some simpler transistor circuits that might be discussed in fourth year, but they are still more complex than NAND and NOR.

[1] SUM of PRODUCTS form has no bracketed terms like ab(f + d) or the brackets implied by long inverting bars like $\overline{ab} = \overline{(ab)}$

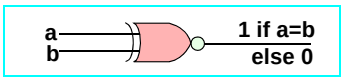
Applications of XOR, XNOR

Applications of XOR/XNOR

Controlled Inverter“



Equality Check“



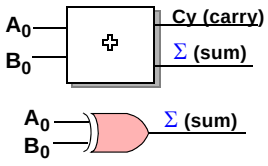
Adder

Adding (“✜“) one-bit numbers.

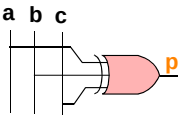
The sum Σ is an XOR gate.

The Carry Cy is an AND gate.

A_0	B_0		Cy	Σ	in dec
0	0	=	0	0	(0)
0	1	=	0	1	(1)
1	0	=	0	1	(1)
1	1	=	1	0	(2)



Odd Parity



Tell if there are an odd number of inputs

“p” = 0 if even number of “1” inputs.

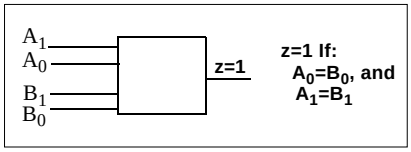
“p” = 1 if odd number of “1” inputs.

	a	b	c	p
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

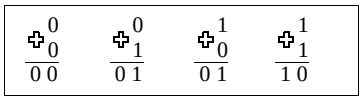
Applications of XOR/XNOR

1-40.•PROBLEM ON COMPARE

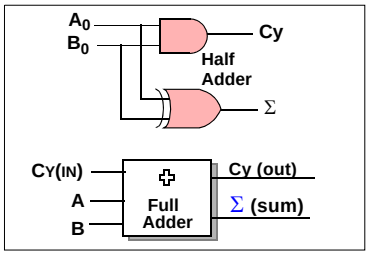
Design a circuit which compares two 2-bit numbers A_1A_0 and B_1B_0. It gives out Z=1 if the two numbers are equal.



The Adder[1]



The four results from adding two bits



This upper circuit is called a half adder because it adds only two bits. A full adder (lower box) can add three bits. The third bit is the carry input from a previous stage.

For adding one-bit numbers, or the first bits of a multibit number, a half added is all right. Otherwise a full adder is needed.

1-41.•PROBLEM ON PARITY

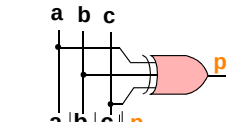
Design an even-parity circuit which gives a “1” output if an even number of inputs are one.

1. We will use ✜ for addition here, since + was used for OR. In many places + is used for both, and you have to figure out which is which.

Applications of XOR and XNOR (Cont)

Error Detection

Parity to find transmission errors

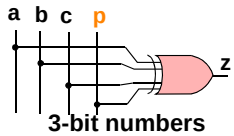


	a	b	c	p
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

"p" = 1 if odd number of "1" inputs.

"z" = 0, if even number of "1" inputs.
Including p as an input to z
makes z always 0.
Except-

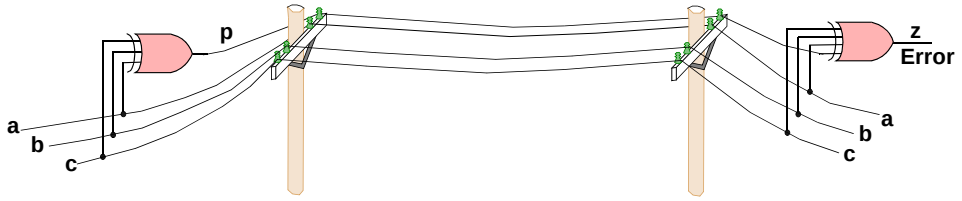
"z" = 1 indicates a bit changed during transmission.



3-bit numbers with parity check

	a	b	c	p	z
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	0
4	1	0	0	1	0
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	0
Bad 1	0	0	1	0	1
Bad 3	0	1	1	1	1
Bad 6	1	1	0	1	1

Transmission Circuit



Using Parity to Check for Transmission Errors

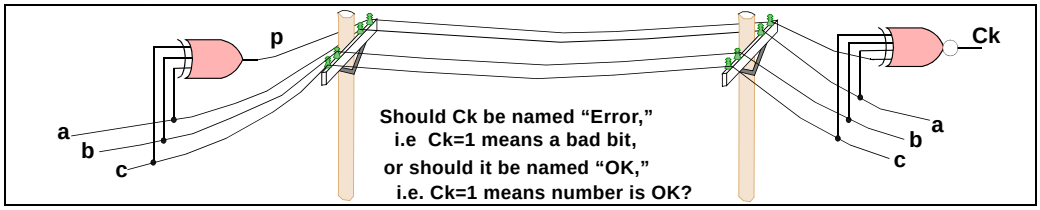
At the input (left side) the parity is calculated for the three input bits (a, b and c) and transmitted as a fourth bit. At the output the parity of all four bits (a, b, c, and p) is calculated.

There should always be an even number of ones (zero is an even number) because the p bit is used to make an even number. If there is an odd number of ones at the output, one bit must have been corrupted during transmission.

1-42.•PROBLEM

If two bits are flipped during transmission? Will the parity error detection circuit shown identify the errors?

1-43.•PROBLEM, PARITY CHECK



Driving Liquid Crystal Displays

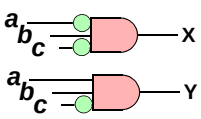
1-44.•PROBLEM, LCD APPLICATION WITH AN XOR AND MANY WORDS. SEE PROB 1-47.• ON PAGE 57.

Common Three-Input Gates

Very Common Three-Input Gates

Select

Identify a particular bit combination.

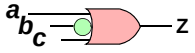


X = 1 when a,b,c = 0,1,0

Y = 1 when a,b,c = 1,1,0

Reject

Reject a particular bit combination.

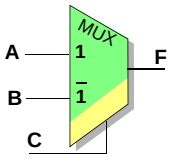


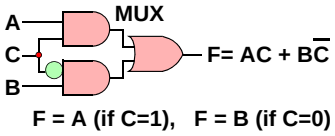
Z = 1 unless a,b,c = 0,1,0

MUX (Multiplexer)

When C=1, A goes thru to F.
When C=0, B goes thru to F.

This is called a 2-input MUX.
The control, C, input isn't counted

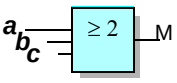


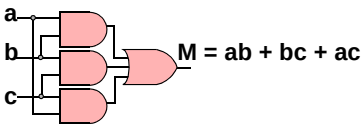


F = A (if C=1), F = B (if C=0)

Majority Gate (Carry)

Z=1 if the majority of the inputs are 1.





Some Useful 3-Input Gates

Gates

A gate is a some logic operator that is small enough to be easily comprehended, and used often enough to be worth giving a special name. Gates may be made from simpler gates. The XOR is usually made from NANDs, NORs and a NOT. The majority gate is shown here as being made from other gates but actually a N-input majority gate could be easily made as a single gate from transistors.[1]

Majority Gate

M = ab + bc + ac + abc

However: abc + ab = ab(c + 1) = ab(1) = ab
Thus the abc term is redundant, and:-

M = ab + bc + ac

Carry Gate

Consider a three-bit adder that adds:-
$Cy_{(out)}, \Sigma = a \oplus b \oplus Cy_{(in)}$

$Cy_{(out)}$ from the table,
is "1" if any two (or three) of a, b and c are "1".

This is the same as for the majority gate.

a	b	Cy(in)	Cy(out)	Σ	Result in Decimal
0	0	0	0	0	0
0	0	1	0	1	1
0	1	1	1	0	2
0	1	0	0	1	1
1	0	0	0	1	1
1	0	1	1	0	2
1	1	1	1	1	3
1	1	0	1	0	2

1-45.•PROBLEM

Design a 4-input circuit which gives an output when 3 or more inputs are "1".

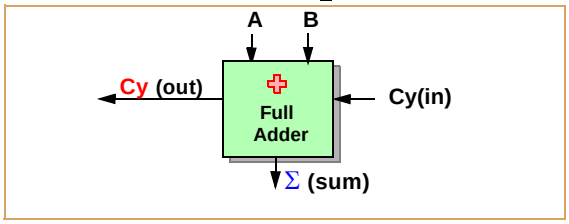
[1] Its output would be inverted.

Deriving Circuits from Truth Tables; Full Adder

Circuits From Truth Tables

The Full Adder

The full-adder adds 3 bits _



a	b	Cy(in)	Cy(out)	Σ	Result, Decimal
0	0	0	0	0	0
0	0	1	0	1	1
0	1	1	1	0	2
0	1	0	0	1	1
1	0	0	0	1	1
1	0	1	1	0	2
1	1	1	1	1	3
1	1	0	1	0	2

a	b	Cy(in)	Cy(out)		Σ	
0	0	0	0		0	$\bar{a}\bar{b}$•Cy
0	0	1	0		1	$\bar{a}\bar{b}$•Cy
0	1	1	1	$\bar{a}$•b•Cy	0	$\bar{a}\bar{b}$•Cy
0	1	0	0		1	$\bar{a}\bar{b}$•Cy
1	0	0	0		1	a$\bar{b}\bar{C}$y
1	0	1	1	a$\bar{b}$•Cy	0	
1	1	1	1	a•b$\bar{C}$y	1	a•b•Cy
1	1	0	1	a•b•$\bar{C}$y	0	

Equation for when Cy(out) is "1"

Cy(out) = $\bar{a}$•b•Cy + a•$\bar{b}$•Cy + a•b•Cy + a•b•$\bar{C}$y

Cy(out) is "0" for all other combinations

Equation for when Σ is "1"

Σ = $\bar{a}\bar{b}$•Cy + $\bar{a}$•b•$\bar{C}$y + a•$\bar{b}\bar{C}$y + a•b•Cy

Deriving Circuits From Truth Tables

Recall we only need consider cases which evaluate to "1".

Boolean formulas default to "0" if they are not "1".

If the output is "1" when a,b,c,d = 1,0,1,0, this gives the term a$\bar{b}$c$\bar{d}$.

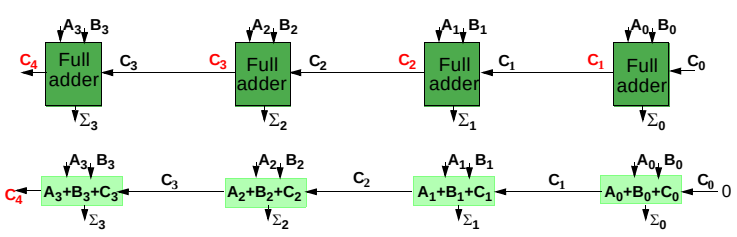
OR together all the terms that give an output of "1", to get the complete formula.

The formulas derived will not be in a reduced form.

Full Adders

The full adder only adds one-bit numbers. To add multibit numbers, one needs several full adders. A 4-bit adder which adds two 4-bit numbers is shown.

4-Bit Adder made from four full adders



The Full Adder (Continued)

Simplify The Full Adder Equations

$Cy(out) = \bar{a}\cdot b\cdot Cy + a\cdot\bar{b}\cdot Cy + a\cdot b\cdot\bar{Cy} + a\cdot b\cdot Cy$

$= \bar{a}\cdot b\cdot Cy + a\cdot\bar{b}\cdot Cy + a\cdot b\cdot\bar{Cy} + a\cdot b\cdot Cy + a\cdot b\cdot Cy$

$= (\bar{a} + a)b\cdot Cy + (\bar{b} + b)a\cdot Cy + a\cdot b(\bar{Cy} + Cy)$

$= (1)b\cdot Cy + (1)a\cdot Cy + a\cdot b(1)$

$= b\cdot Cy + a\cdot Cy + a\cdot b$

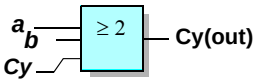
= a majority gate

(I) x + x = x

(D1) $ay \underline{+} by = (a + b)y$

(N) x + x = 1

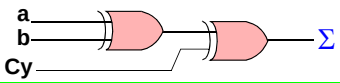
(B) x•1 = x



$\Sigma = \bar{a}\cdot\bar{b}\cdot Cy + \bar{a}\cdot b\cdot\bar{Cy} + a\cdot\bar{b}\cdot\bar{Cy} + a\cdot b\cdot Cy$

Reduce by comparing with the parity circuit on the right.

Reducing algebraically is too much work.
(Work is shown the next slide)



Truth Table For Sum

a	b	Cy(in)	Σ
0	0	0	0
0	0	1	1
0	1	1	0
0	1	0	1
1	0	0	1
1	0	1	0
1	1	1	1
1	1	0	0

Truth Table For Parity

	a	b	c	p
0	0	0	0	0
1	0	0	1	1
3	0	1	1	0
2	0	1	0	1
4	1	0	0	1
5	1	0	1	0
7	1	1	1	1
6	1	1	0	0

Simplifying the Full Adder

Carry Term

This is straightforward

1-46.•PROBLEM

Draw the circuit for a full adder using a majority gate and an exclusive-or gate.

The Full Adder Sum Equation by Algebra

Optional Slide for Algebra Lovers

Simplify the sum equation algebraically

$\Sigma = \overline{a}\bullet\overline{b}\bullet Cy + \overline{a}\bullet b\bullet\overline{Cy} + a\bullet\overline{b}\bullet\overline{Cy} + a\bullet b\bullet Cy$

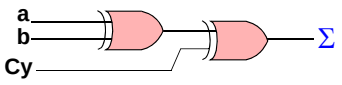
$= \overline{a}\bullet\overline{b}\bullet Cy + a\bullet b\bullet Cy + \overline{a}\bullet b\bullet\overline{Cy} + a\bullet\overline{b}\bullet\overline{Cy}$

$= (\overline{a}\bullet\overline{b} + a\bullet b)\bullet Cy + (\overline{a}\bullet b + a\bullet\overline{b})\overline{Cy}$

$= \overline{(\overline{a}\bullet b + a\bullet\overline{b})}\bullet Cy + (\overline{a}\bullet b + a\bullet\overline{b})\overline{Cy}$

$= \overline{(a \oplus b)}\bullet Cy + (a \oplus b)\bullet\overline{Cy}$

$= (a \oplus b)\oplus Cy$



(C) $x + y = y + x$

(D1) $ay + by = (a + b)y$

(Formula for XNOR) $\overline{a}\bullet\overline{b} + a\bullet b = \overline{\overline{a}\bullet b + a\bullet\overline{b}}$

(Formula for XOR) $\overline{a}\bullet b + a\bullet\overline{b} = (a \oplus b)$

(Formula for XOR) $\overline{u}\bullet Cy + u\bullet\overline{Cy} = (u \oplus Cy)$

$u = (a \oplus b)$

- **Sum Term**

This slide requires being aware of the various forms of XOR and XNOR.

$A\oplus\overline{B} = \overline{A}\oplus B = \overline{A\oplus B}$

$a\oplus b = a\overline{b}+\overline{a}b$

$\overline{a\oplus b} = \overline{a}\bullet\overline{b} + ab$

$\overline{(a\oplus b)}c + (a\oplus b)\overline{c} = \overline{a\oplus b\oplus c}$

Common Mistakes

Common Errors

1. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

2. Saying an expression is equal to its dual.

3. Not using $AB + A = A$ to simplify expressions before using more more complex rules.
   Also not reducing using $A + \overline{A}E = A + E$.
   Simplifying and reducing first may save many lines of algebra.

4. Saying $X + 1 = X$
   Everyone knows better than this, but they still do it.

5. Not placing the brackets around all the AND terms before taking the dual or (coming up soon) before using the generalized DeMorgan's theorem .

   $F = A \cdot (B + \overline{C} \cdot D) + C \cdot \overline{D}$

   Must Add Brackets

   $F = [A \cdot (B + [\overline{C} \cdot D])] + [C \cdot \overline{D}]$

   These are the ones you forget

5. Dropping overbars. On assignments recheck each line, after completion.

   $F = ab + a\overline{b} \cdot \overline{d} + ac$

   $= a(b + b \cdot \overline{d}) + ac$ *by D1, and carelessness*

How not to take the dual:

$F = abce + ab + bce\overline{d}$

$= (a+b+c+f)(a+b)(b+c+e+\overline{d})$ take dual

There are two blunders:

The simplification law should always be used first if applicable.

A function should never never never be equated to its dual.

A better approach

$F = abce + ab + bce\overline{d}$

$= ab + bce\overline{d}$ (S1) $a+ax=a$

$F_{dual} = (a+b)(b+c+e+\overline{d})$ take dual

More Examples

Appendix of Examples and Problems

1-47.• PROBLEM; LIQUID CRYSTAL DISPLAY (LCD)

A *liquid crystal* is a thin layer of polarizing liquid material, sandwiched between two glass plates. Covering the plates is a conductive layer of indium-tin oxide (electrodes) so thin it is transparent. Thus light, polarized in the same direction as the liquid crystal, can pass through. If an electric field is applied between the conductive layers, it can rotate the polarization of the crystal 90°, and shut off the polarized light.

One cannot apply dc to the liquid for more than a few ms or it breaks down, thus one must use ac to control the on and off of an LCD display. Unfortunately the logic circuits, which drive these displays, run from dc, so it must be converted.

The XOR circuit shown, can supply an ac square wave to the electrodes to shut them off with a push button.

Plot the waveforms of:

1) point A with respect to ground, $V_{A.}$

2) point B with respect to ground, $V_{B.}$

3) point A with respect to B, $V_{A\text{-}B.} = V_A\text{-}V_{B.}$

4) indicate the time intervals with "light" and "no light".

If the voltage between the electrodes completely reverses ($V_{A\text{-}B}$ => $-V_{A\text{-}B)}$, the liquid crystals line up in the same direction, and the polarization stays the same.