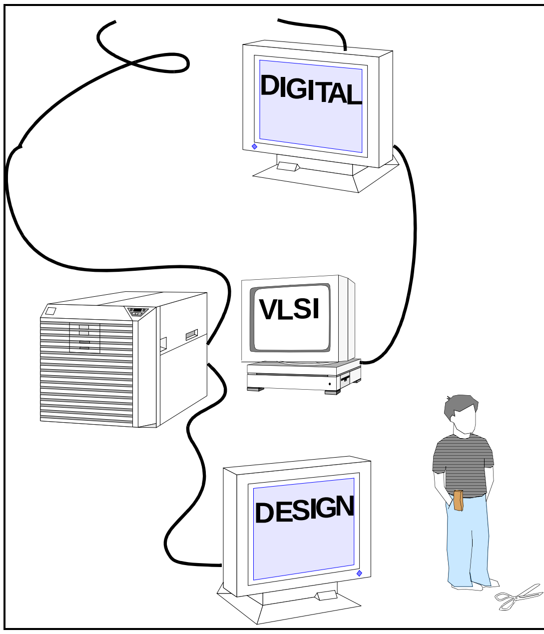
Digital Circuit Engineering



2nd Distributive

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

Simplification

$$YX + X = X$$

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

Absorption

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

DeMorgan Simple

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

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

DeMorgan General

If

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

Then

$$\overline{F} \equiv F(\overline{a}, \overline{b}, \ldots \overline{z}, \cdot, +, 1, 0)$$

Remember to bracket AND terms

Carleton University **2009**

▪

DeMorgan's Theorem

Simple two variable forms

As equations and as gates with inverted inputs

Why Real Gates are NAND/NOR, not AND/OR

Two symbols for NAND; two for NOR

AND-OR designs are easier to think about

NAND-NOR designs must be done to use real gates

Design with AND-OR; Implement with NAND-NOR

Change between them using DeMorgan's Theorem

AND/OR to NAND/NOR Conversions

Generalized DeMorgan's Thorem

Common Errors

DeMorgan's Law

DeMorgan's Simple Form

Used To Find the Inverse of Expressions

DeMorgan's Law

$\overline{A \cdot B} = \overline{A} + \overline{B}$ *(DeM)*

Dual DeMorgan's Law

$\overline{D + E} = \overline{D} \cdot \overline{E}$ *(DeM)*

Inverse

$\overline{\overline{A} + \overline{B}} = A \cdot B$

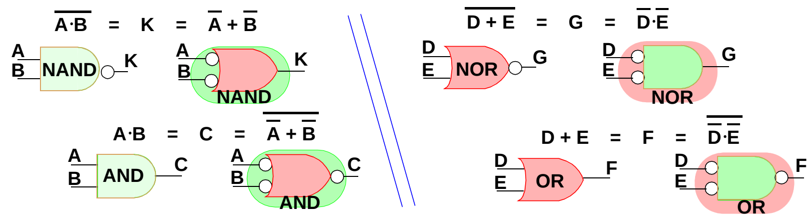
The dual inverse

$\overline{\overline{D} \cdot \overline{E}} = D + E$

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$	AB	$\overline{AB}$
0	0	1	1	1	0	1
0	1	1	0	1	0	1
1	0	0	1	1	0	1
1	1	0	0	0	1	0

D	E	$\overline{D+E}$	$\overline{D}$	$\overline{E}$	$\overline{D}\cdot\overline{E}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

Equivalent graphical forms:



DeMorgan's Laws on Complementing Expressions

A theorem relating NANDs and NORs.

- An OR gate with inverted inputs is equivalent to an AND gate with an inverted output.
- An AND gate with inverted inputs is equivalent to an OR gate with an inverted output.
- Inverting inputs and outputs of an OR makes it an AND.
- Inverting inputs and outputs of an AND makes it an OR.

EXAMPLE

Convert $\overline{(a + b)(a + \overline{c})}$ to an expression with 3 letters and inversion bars only over single letters.

$\overline{(a + b)(a + \overline{c})} = \overline{(a + b)} + \overline{(a + \overline{c})}$ *(DeM1)*

$= (\overline{a}\cdot\overline{b}) + (\overline{a}\cdot\overline{\overline{c}})$ *(DeM2)*

$= \overline{a}\cdot\overline{b} + \overline{a}\cdot c$ *(Clear brackets)*

$= \overline{a}(\overline{b} + c)$ *(D1) xy + xz = x(y+z)*

41.• PROBLEM

Reduce $\overline{\overline{(a+b)} + \overline{\overline{a\cdot b}}}$ to four letters with inversion bars over single letters only.

42.• PROBLEM

Reduce $\overline{d(\overline{de}) + (\overline{de})e}$ to four letters with inversion bars over single letters only.

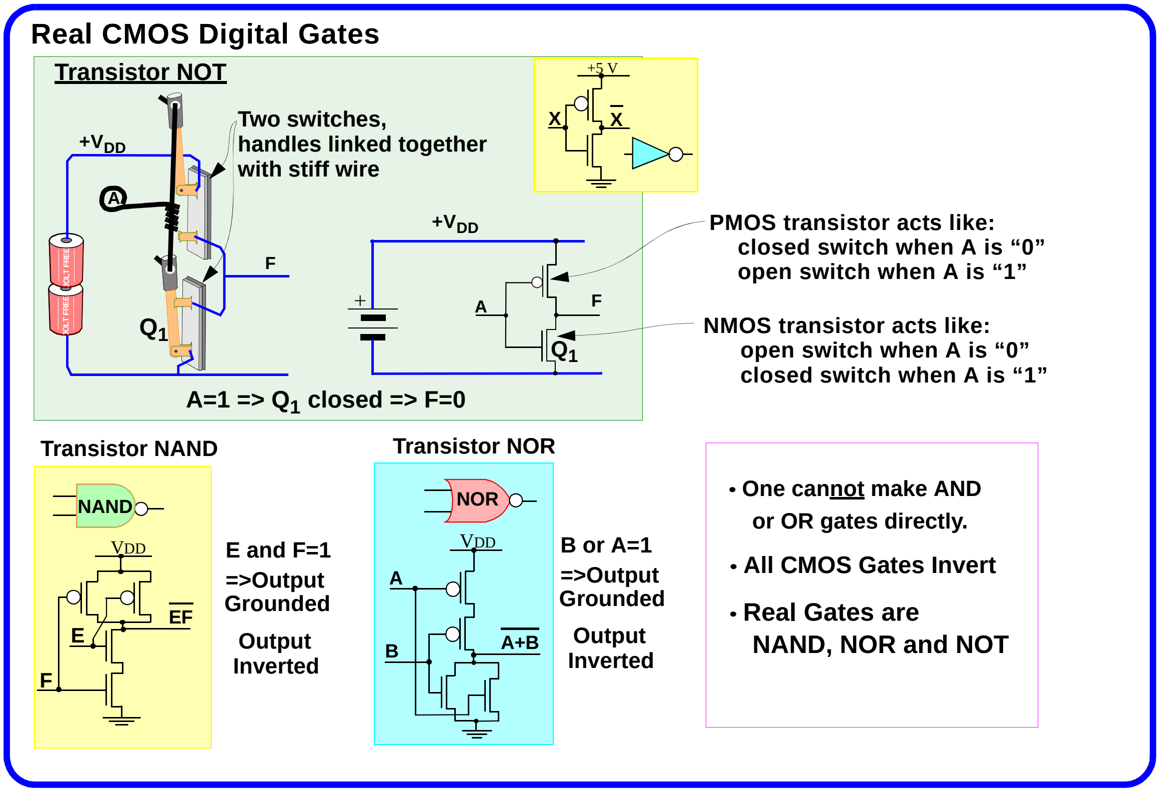
<u>Changing everything into NOT and AND gates</u>

It turns out that any logic circuit can be made from AND and NOT gates. DeMorgan's law can be used to transform the circuits.

43.• PROBLEM

Convert $((rw + t)u + \overline{r})\overline{t}$ into a function with only AND and NOT operations.

Why Real Gates are NAND/NOR, not AND/OR

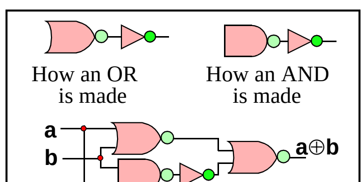


Constructing Gates One Can Build

Making CMOS gates from transistors

CMOS stands for Complementary-symmetry Metal-Oxide-Semiconductor gates. They always have complementary transistors, which means PMOS (turn off with a one input) above the output, and NMOS (turn on with a one input) below the output.

The correct one inputs turn the lower NMOS transistors on, which pulls the output down to zero thus inverting the output.



How an OR is made

How an AND is made

One way of making an XOR
See problem 41.•.

The Easily Constructed CMOS Gates

NANDs with 2, 3 or 4 inputs, NORs with 2, 3 or 4 inputs, and NOTs.

Gates constructed from other gates

To avoid all the extra inverters (NOTs) real circuits are designed to use NANDs and NORs instead of ANDs and ORs.

Thinking

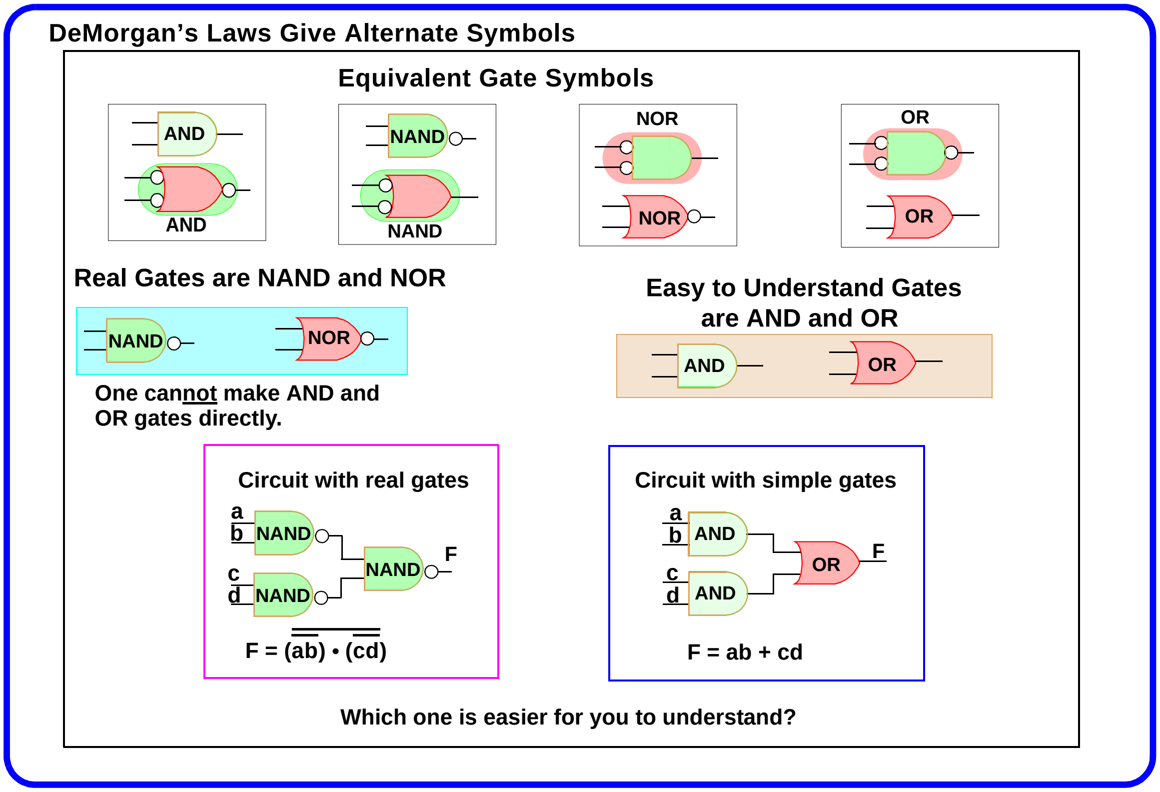
- Thinking in NAND-NOR logic is difficult. Just look at an industrial schematic used extensively for maintenance. The margin will be full of “1”s and “0”s pencilled in by users.
- Converting AND-OR into NAND-NOR is a straightforward mechanical process.
- Much less error prone than doing logic with inverted signals[1] and NAND-NOR logic.

These Notes

- If the logic is important ANDs and ORs will be used.
- If the gate design is important, as when we talk about CMOS gates, NANDs and NORs will be used.

1. If “flash” is a signal name, “$\overline{\text{flash}}$” is an inverted signal.

DeMorgan; Two Symbols for NAND, Two for NOR



DeMorgan's Theorem

The two expressions for the circuit with real gates and the circuit with simple gates, are equivalent

$$ab + cd = F = \overline{\overline{(ab)} \cdot \overline{(cd)}}$$

Use High-True And-Or Signals for Thinking

Thinking

- Thinking in nand-nor logic is difficult. Just look at any industrial schematic used extensively for maintenance. The margin will be full of "1"s and "0"s pencilled in by users.
- Converting between and-or and nand-nor is a straightforward mechanical process.
- Much less error prone than doing logic with asserted low signals and nand-nor logic.

<u>These Notes</u>

- If the logic is important ANDs and ORs will be used.
- If the gate design is important, as when we talk about CMOS gates, NANDs and NORs will be used.

44.• PROBLEM

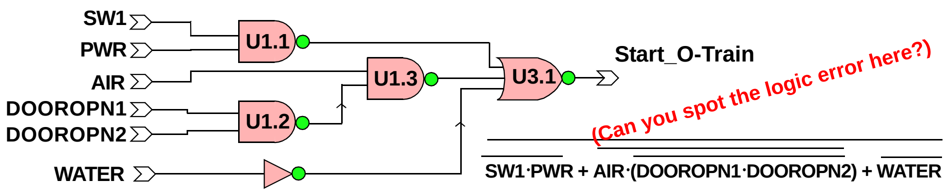
Prove, using DeMorgan's Theroem(s), that $ab + cd = \overline{\overline{(ab)} \cdot \overline{(cd)}}$

Start the O-Train; Use NAND-NOR or AND-OR?

Circuits Using NANDs and NORs are Hard to Follow

Don’t Think Using NANDs and NORs.

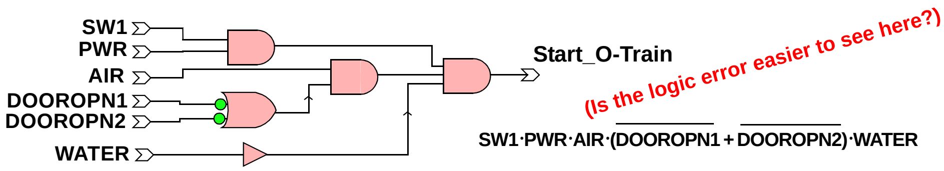
Confusing multiple logic inversions.
Does water stop or start the train?
Is SW1 a start or stop switch?



Circuits Using ANDs and ORs are Easy to Follow

Do Thinking Part of Design with AND/OR; Convert After Thinking is Done

The same circuit made with ANDs and ORs.
The final equation can be written from inspection of the circuit.



Starting The O-Train[1]

The AND-OR logic is much clearer. It takes air, power, water to start the train. Also two doors must not be open. However it takes 4 gates and 6 inverters to implement the AND-OR circuit made by adding inverters to NANDs and NORs. The less clear NAND-NOR circuit does the same logic, with 4 gates and 1 inverter.

NAND-NOR Logic is Confusing to Humans

Multiple negatives are confusing

If your teacher said, “Never will I not, not give you a A in Digital Circuits,” it would take you some careful reading to determine your mark.

Diagrams with real gates

This is the kind of diagram one would use to build a circuit.

It has easily manufacturable gates, i.e. NOT, NAND and NOR.

The numbers U1.1 etc. are gate numbers that would physically identify the gate on a layout diagram showing where each part was.

Easy to read diagrams

You should be able to spot the logic error.

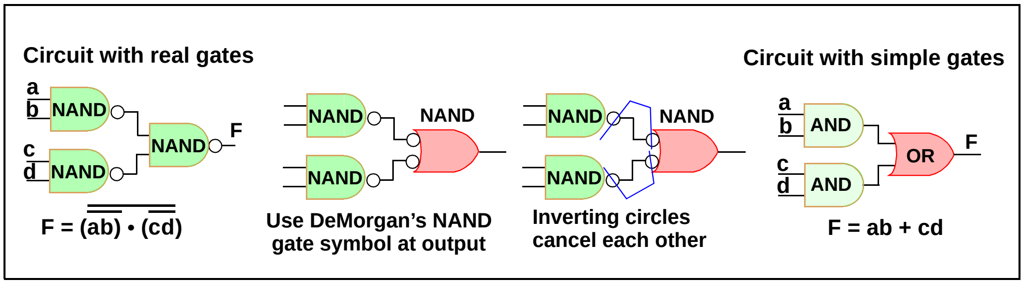
$\overline{\text{DOOROPN1}} + \overline{\text{DOOROPN2}}$

Should you be able to start the train with one door open?

[1] The O-Train is an Ottawa interurban train, which stops right outside the Engineering Building at Carleton University.

Changing Real Gates into Simple Gates

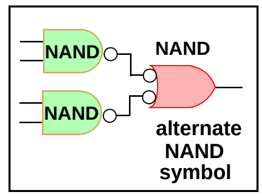
DeMorgan's Law Transfers NAND/NOR Gates Into AND/OR Gates



- Circuits meant for <u>understanding</u> the logic use ANDs and ORs.
  - Draw your circuits with ANDs and ORs.
- Circuits for <u>construction</u> are drawn with NANDs and NORs.

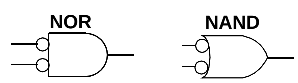
Draw construction diagrams by:
- transforming the understandable circuit into the real circuit
- using the DeMorgan alternate symbols.

- <u>Compromise</u> drawings have NANDS and NORs with the circles are arranged to cancel each other.



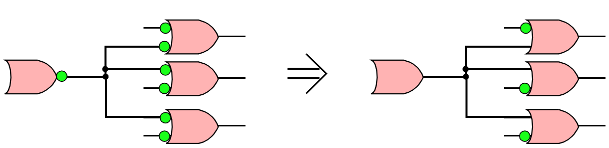
Transforming NAND-NOR Diagrams into AND-OR Diagrams

- Diagrams in this course will be drawn with ANDs and ORs as much as possible.
- Diagrams for construction or maintainance, that want to show exactly what gates were used, will be drawn with NANDs and NORs. This is particularly true of older diagrams.
- A compromise method, which is almost as easy to follow, but shows the real gates as used, is to make alternate gates with the alternate symbols, the ones with the inverting circles on the inputs.



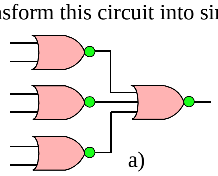
<u>NOTE:</u>

One output circle cancels all the input circles it feeds.



45.• PROBLEM

Transform this circuit into simple gates.

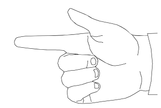


a)

Design in AND and OR Gates

Don't do initial design in NANDs and NORs.

Design in AND/OR;
Convert After Thinking is Done



Rational

- **People think best with AND and OR.**
- **Multiple inversions are very confusing**
- **There is seldom a logical reason to invert except at circuit inputs.**
- **Conversion AND/OR $\Rightarrow$ NAND/NOR is easy using DeMorgan's form of gates.**
  Do it after the thinking is done.

Design With Conceptually Easy Logic

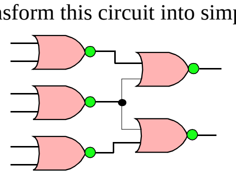
In the post 2000 era, people usually think about the logic. The details of construction are done automatically. This means you will normally do AND-OR type logic.

Think with AND-OR
Build with NAND-NOR

DeMorgan's Law Transforms Real Gates to Simple Gates

46.• PROBLEM

Transform this circuit into simple gates.



AND/OR to NAND/NOR Conversions

Using DeMorgan in Graphical Form

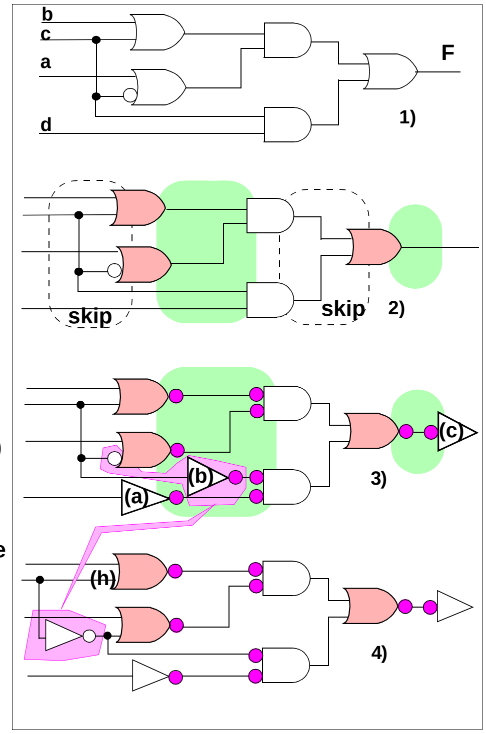
Example:

1. **Start with AND/OR circuit**
   F = (a + c)(b + c) + cd

2. **Select alternate connection layers.**
   (Every second layer of connecting wires between layers of gates.)
   One end of wires may be inputs or outputs.

3. **Put back-to-back inverting circles on both ends of the leads.**
   Add NOT gates when necessary ((a), (b) and (c))

4. **Moving inverters, or inverting bubbles may make logic simpler.**
   Sometimes inverters on two (or more) input leads should be moved to the output.
   (See next example)

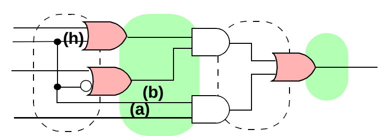


Graphical Conversion AND/OR to NAND/NOR

1. Start with an AND-OR circuit. If you are starting with an equation. draw the circuit.

- The original formula will usually have inversions only on inputs. However there may be inversions anywhere.

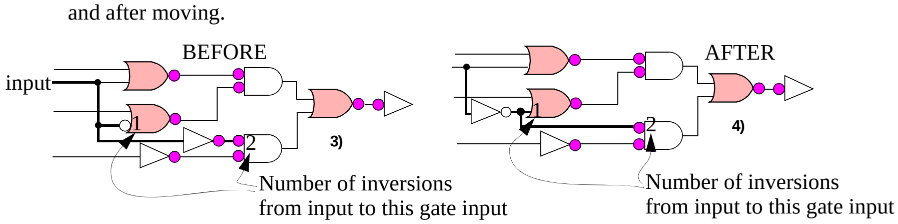
2. The circuit is drawn with a layer of connections which feeds a layer of gates, feeding a layer of connections (green), which feeds another layer of gates, which feeds another layer of connections (green). Some circuits may have more or fewer layers. Select alternate connection layers.



- Some circuits are nicely layered with the first layer feeding a second layer which feeds a third layer etc. However this step is not exact. There are often several legitimate ways to define layers. Note some wires, like (a) and (b) may pass through a gate level (green) without a gate.

3. Put back-to-back inverting circles at both ends of wires going through a connection layer. On leads like (a) and (b), you will have to add an inverter, since there is no gate on which to place the second circle.

4. This step is less automatic. Look for leads where two inverters can be replaced by one. When doing this be careful not to invert a other connections on the same input, like (h). When moving inverting circles make sure that the number of circles between the signal input and all the gate inputs it goes to, are the same before and after moving.



AND/OR to NAND/NOR Conversions

Graphical Conversion (cont.)

4. Copy of step 4 on last slide:
   It has back-to-back bubbles
   and consolidated inverters

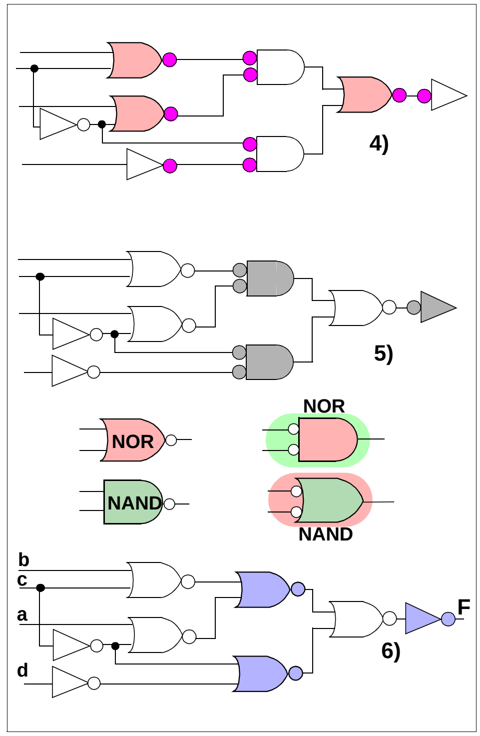
5. Select the unconventional gates.
   The ones with input bubbles.

This is a compromise solution which is:

- fairly readable
- represents real gates

6. Use DeMorgan laws on the unconventional gates.

- Hard to understand the logic
- but good for construction.



On previous slide

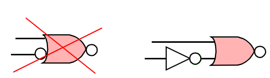
- You must always add two inverting circles to a lead, or none. Never add just one circle, not even if one exists already.

On this slide

5. Look at the gates. Conventional ones have circles on their outputs and are NANDs, NORs or NOTs.
   The unconventional ones have circles on their inputs but are still NANDs, NORs or NOTs.

- This step gives a circuit which is fairly easy to read because one can mentally cancel the back-to-back circles. However it still represents a circuit you can build easily because it does not contain ANDs and ORs.

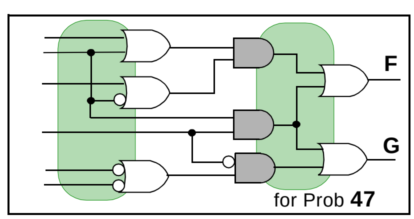
6. If desired, one can replace the DeMorgan forms of NOR and NAND, the ones with circles on the inputs, with the more standard forms with a circle on the output.

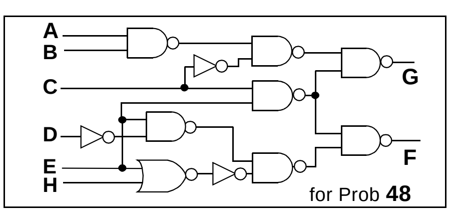


- Since NAND-NOR diagrams represent real gates, do not place a single circle on the input of a gate, as was done in the theoretical AND-OR diagram.

47.• PROBLEM:.

For the circuit on the right, find the NAND-NOR circuit when the connection on the levels shown are selected.





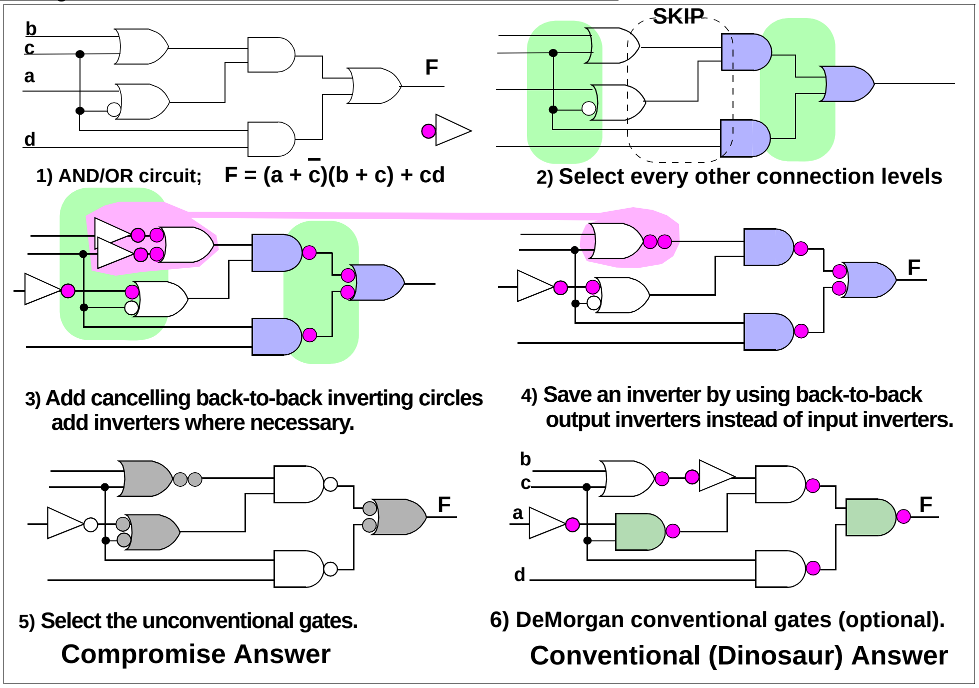
48.• PROBLEM

Find an AND-OR equivalent for the circuit on the left.
This requires working backwards.

AND/OR to NAND/NOR Conversions,

2nd Example of AND/OR to NAND/NOR

Selecting the 1st and 3rd levels, instead of the 2nd and 4th



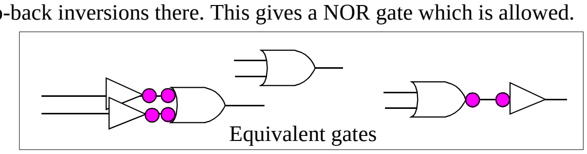
AND/OR ⇐⇒ NAND/NOR Conversion (cont)

This is the same circuit as the previous example.

The difference between the examples starts at step 2

2. the alternate blocks of leads where the inverters are added are that ones marked SKIP in the first example. Compare step 2 here with step 2 on Slide 8.
3. In this step, the two inverters feeding the OR with inverting inputs looks too complex. Throwing out the double circles leaves us with an OR gate. This is not allowed. Go to the output of the OR and add back-to-back inversions there. This gives a NOR gate which is allowed.

   Both the lower gates on the right are equivalent to an OR gate, but one is simpler to implement as real gates.



6. Changing the NORs and NANDs with circles on their inputs, to NORs and NANDs with circles on their outputs, is the final step. Do it if you like such drawings better, or if the drawing standards your company uses require it, or if your boss says to.

- Compare the results with those in steps 5 and 6 of the previous example. This circuit is marginally simpler than the previous result. One NOT gate is saved.

DeMorgan’s Law, the General Form

DeMorgan’s Theorems, General Form

Abstract Notation

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

Action	Example I
a) Take a Boolean expression	$F = \overline{A}\cdot B\cdot C$
b) Bracket all groups of ANDs	$F = \{\overline{A}\cdot B\cdot C\}$
Take dual	
c) Change AND → OR and OR → AND	$F_{dual} = \{\overline{A} + B + C\}$
d) Clean brackets	$F_{dual} = \overline{A} + B + C$
Change dual into inverse	
e) Invert all variables	$\overline{F} = \{A + \overline{B} + \overline{C}\}$

General Form of DeMorgan’s Theorem

The slide shows a simplified version

The true generalized DeMorgan

$$\overline{F(A,B,C,\ldots +, \cdot,0,1)} = F(\overline{A},\overline{B},\overline{C},\ldots ,\cdot, +,1,0)$$

$$F(A,B,C,\ldots +, \cdot,0,1) = \overline{F(\overline{A},\overline{B},\overline{C},\ldots ,\cdot, +,1,0)}$$

The interchange of 0 and 1 was left out on the slide, since designers practically never have 0 or 1 in the expressions they operate on with DeMorgan’s law.

DeMorgan’ general law is very similar to Duality

The obvious difference is the inverting bars.

Another important difference is the application.

- DeMorgan transforms an expression into its inverse.
- Duality takes a valid identity and generates another valid identity.

49.• PROBLEM

Find an expression for $\overline{F}$ that has inverting bars only over single letters,

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

Generalized DeMorgan's Law

Examples

$$\overline{F(A,B,C,\ldots +, \cdot)} = F(\bar{A},\bar{B},\bar{C},\ldots, \cdot, +)$$

Example II

a) Take a Boolean expression $F = \bar{A}\cdot B\cdot C + A\cdot\bar{B}$

b) Bracket all groups of ANDs $\{\bar{A}\cdot B\cdot C\} + \{A\cdot\bar{B}\}$

Take dual

c) Change AND → OR and OR → AND $F_{dual} = \{\bar{A}+B+C\}\cdot\{A+\bar{B}\}$

e) Clean brackets $F_{dual} = (\bar{A}+B+C)\cdot(A+\bar{B})$

Change dual into inverse

e) Invert all variables $\bar{F} = (A+\bar{B}+\bar{C})\cdot(\bar{A}+B)$

Example III

a) Take any Boolean expression $F = [\bar{A}\cdot B\cdot C + D\cdot(A\cdot B + C)]\cdot\bar{A}$

b) Bracket all groups of ANDs $F = \{[\{\bar{A}\cdot B\cdot C\} + \{D\cdot(\{A\cdot B\} + C)\}]\cdot\bar{A}\}$

c) Change AND →OR and OR →AND
Ignore any existing overbars $F_{dual} = \{[\{\bar{A} + B + C\}\cdot\{D + (\{A + B\}\cdot C)\}] + \bar{A}\}$

d) Clean brackets $F_{dual} = \{\bar{A} + B + C\}\cdot\{D + \{A + B\}\cdot C\} + \bar{A}$

e) Invert all variables $\bar{F} = \{A + \bar{B} + \bar{C}\}\cdot\{\bar{D} + \{\bar{A} + \bar{B}\}\cdot\bar{C}\} + A$

Examples using the Generalized DeMorgan's Law

<u>**Be sure to put brackets around the ANDs**</u>

When you do algebra, you automatically do the ANDs before the ORs. The notation is designed that way.

Putting brackets around an OR shows that the OR should be done before the AND.

When you use DeMorgan's law, and you transform
ab + cd into (a+b)(c+d),
the brackets make sure that the variables in the transformed expression are operated on in the same order. That is the ORs that came form ANDs are still done first, as the original ANDs were. Without the brackets
ab + cd => a + bd + d, which is not correct.

By placing the brackets around the ANDs first, you do not get confused during the transformations.

50.• PROBLEM

(a) Convert $\overline{(a + b)(a + \bar{c})}$ to an expression with 3 letters and inversion bars only over single letters, using the Generalized DeMorgan's law.
(b) Compare this method with that of the Example on *Comment on Slide 10* and see if the algebra is shorter.

51.• PROBLEM

Given $H = (ab + \bar{c})g + de(a + b)$, find $\bar{H}$

Generalized DeMorgan's Law

Examples: Generalized DeMorgan's Theorems

Example IV

a) Take a Boolean expression $F = \bar{A}\cdot B\cdot C + \overline{A\cdot B}$

b) Bracket all groups of ANDs $\{\bar{A}\cdot B\cdot C\} + \{\overline{A\cdot B}\}$

c) Change AND → OR and OR → AND
Ignore existing overbars $F_{dual} = \{\bar{A}+B+C\}\cdot\{\overline{A+B}\}$

d) Clean brackets if convenient $F_{dual} = (\bar{A}+B+C)\cdot(\overline{A+B})$

e) Invert all variables $\bar{F} = (A+\bar{B}+\bar{C})\cdot(\overline{\bar{A}+\bar{B}})$

Note:

<u>Ignore any inverting overbars except over single letters</u>

$$F = \overline{\bar{A}\cdot B\cdot\overline{(C + \overline{A\cdot B})}} \Rightarrow F_{dual} = \overline{\{\bar{A}+B\}+\overline{(C\cdot\{\overline{A+B}\})}} \Rightarrow \bar{F} = \overline{\{A+\bar{B}\}+\overline{(\bar{C}\cdot\{\overline{\bar{A}+\bar{B}}\})}}$$

Examples using the Generalized DeMorgan's Law

<u>Common Worry, Intermediate Overbars</u>

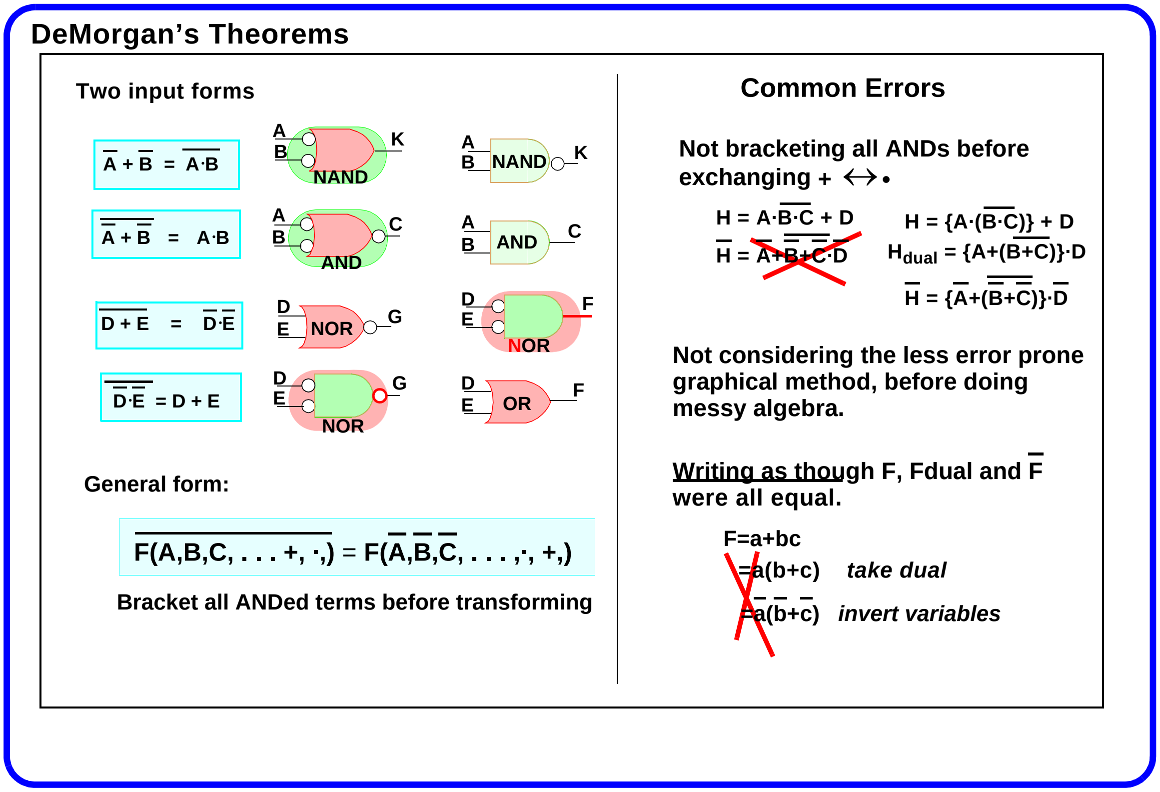
When applying the generalized law, do not cosider any overbars except those on top of single letters. If the overbar is over two or more letters, just carry it through without change.

52.• PROBLEM

Use DeMorgan's general law to remove all but one of the inverting bars from the dinosaur circuit on Slide 10. The expression is

$$f = \overline{\overline{(b+c)\cdot\overline{(\bar{a}\cdot c)}}\cdot\overline{(c\cdot d)}}$$

DeMorgan; Review



OR, XOR, DeMorgan and English http://www.rbs0.com/tw.htm#anchor580

Adapted from Ronald B. Standler's humorous and informative site about English, written by a real engineer and a lawyer.[1] The quoted text is directly from his work.

"*Most legal style manuals devote at least one page of diatribe to the meaning of and, or, and/or. I can only conclude that the intensity of these attorneys' argumentation must compensate for their ignorance!*"

In English, there are two meanings for OR:

1) The **inclusive OR** which means A or B or both. It is better expressed as A *AND/OR* B
2) The **exclusive OR** which means A or B but not both. In English, it also means only one of A, B, C ... or D, as opposed to logic where it means an odd number of the set must be true.

"*It is my impression that most physicists and mathematicians generally use or in the inclusive sense, and most attorneys in the USA generally use or in the exclusive sense.*

The people who write legal style manuals don't seem to notice that their assertion about "or" not only contradicts the mathematical definition that is used in computer data bases, but also is a narrow-minded approach to the English language."

His point is illustrated by a line from the Carleton University Health and Safety documentation.

"Food and beverages are not permitted in the lab..."

Logically, if F=Food, and B=Beverage, this statement means the expression ~(F&B) must be true. Thus coffee with no food, or dry food only, is allowed.

Map of No (F and B)

F \ B	0	1
0	1	1
1	1	0

If one says "no (food or beverage) is permitted in the lab," lawyers would mean ~(F⊕B), however engineers would mean ~(F+B), which is what is desired, provided the brackets are used to avoid ambiguity.

Using DeMorgan's Theorem, one gets ~(F+B) = (~F)&(~B) ie. "No food and no beverages are permitted." Alternately, Sadler suggests a clearer English version, "Neither food nor beverages are permitted."

[1] Sadler's article is much funnier than this adaptation, but he is a copyright lawyer, and I am not going to plagiarize his work here.

Solution to #52.•

$f = \overline{\overline{(b+c)\cdot\overline{(\bar{a}\cdot c)}}\cdot\overline{(c\cdot d)}}$ Let $\mathcal{F} = \bar{f} = \overline{(b+c)\cdot\overline{(\bar{a}\cdot c)}}\cdot\overline{(c\cdot d)}$ Our answer is the inverse of $\mathcal{F}$

$\mathcal{F} = \overline{(b+c)\cdot\overline{(\bar{a}\cdot c)}}\cdot\overline{(c\cdot d)}$

$= \overline{[(b+c)\cdot\overline{(\bar{a}\cdot c)}]}\cdot\overline{(c\cdot d)}$ *Add brackets to be sure we know where the overbar ends*

Take dual

$\mathcal{F}_{dual} = \overline{[(b\cdot c)+\overline{(\bar{a}+c)}]}+\overline{(c+d)}$

Do side calculations using DeMorgan $\overline{[(b\cdot c)+\overline{(\bar{a}+c)}]} = [\overline{(b\cdot c)}\cdot(\bar{a}+c)] = [(\bar{b}+\bar{c})\cdot(\bar{a}+c)]$

$\overline{(c+d)} = \bar{c}\cdot\bar{d}$

$\mathcal{F}_{dual} = [(\bar{b}+\bar{c})\cdot(\bar{a}+c)]+\bar{c}\cdot\bar{d}$

Insert overbars

$\bar{\mathcal{F}} = f = [(b+c)\cdot(a+\bar{c})]+c\cdot d$

Alternate Solution to #52.•

$f = \overline{\overline{(b+c)\cdot\overline{(\bar{a}\cdot c)}}\cdot\overline{(c\cdot d)}}$

DeM $\overline{\bar{x}\cdot\bar{z}} = x+z$

$f = (b+c)\cdot\overline{(\bar{a}\cdot c)}+(c\cdot d)$

DeM $\overline{\bar{x}\cdot\bar{z}} = x+z$

$f = (b+c)\cdot(a+\bar{c})+(c\cdot d)$

Swap $(x+c)(z+\bar{c}) = x\bar{c} + zc$

$f = b\bar{c} + ac +(c\cdot d)$