EXTRA EXERCISES: WEEK 2

EXERCISE 1

Minimize the following Boolean expressions using the properties and postulates of Boolean algebra:

1. $f(a,b,c) = \overline{b \cdot c} \cdot \overline{a \cdot \bar{b} \cdot \bar{c}} \cdot \overline{a \cdot b \cdot \bar{c}}$

2. $f(a,b,c,d) = a.b.c.d + \bar{a}.\bar{b} + a.b.\bar{c} + a.\bar{b} + a.\bar{c}$

(From [4] (see bibliography). With permission)

Some previous considerations:

1) First clarification: Given a certain Boolean expression, normally multiple minimized versions of it exists. ***There is no single answer to the problem of minimizing a Boolean function*.**

2) The second point we should bear in mind is: what is a minimal expression? The definition of "minimal" depends on the environment in which the function is used. For example, in the "classical" implementation of digital circuits using low scale integration chips (chips that typically contain between 4-6 logic gates), the most expensive components are logic gates, which means that we should obtain the expression requiring the minimum number of them. Otherwise, if we are implementing the circuit designing an integrated circuit, the most valuable resource will be the silicon area; therefore in this case the number of connection lines may be as important as the number of gates used.

We are going to use a mathematical definition, a kind of agreement between the number of logic gates and the number of inputs to each gate. This last factor is related to the number of connections. Our target will be to obtain a Boolean expression that contains the ***minimum number of 2-operand operations***. As an example, the expression (1) of the problem statements has ten 2-operand operations as we can see in the following schema:

$$f(a,b,c) = \underbrace{\underbrace{\overline{b.c}}_{2}.\underbrace{\overline{a.\bar{b}.\bar{c}}}_{3}.\underbrace{\overline{a.b.\bar{c}}}_{3}}_{10}$$

Bearing this in mind, let's minimize the following expressions:

1. $f(a,b,c) = \overline{b \cdot c} \cdot \overline{a \cdot \bar{b} \cdot \bar{c}} \cdot \overline{a \cdot b \cdot \bar{c}}$

The previous expression could seem a little bit confusing if the order in which the Boolean operations are executed is not clear: The "negation" or "complement" operation has priority over the logical product, and the logical product has priority over the logical addition. Bearing this in mind we may write the expression to be minimized in a clearer way:

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

Applying De Morgan's law to the parentheses:

$f(a,b,c) = (\bar{b} + \bar{c}).(\bar{a} + b + c).(\bar{a} + \bar{b} + c)$

Multiplying the last two parentheses:

$(\bar{a} + b + c).(\bar{a} + \bar{b} + c) = \bar{a}.\bar{a} + \bar{a}.\bar{b} + \bar{a}.c + \bar{a}.b + b.\bar{b} + b.c + \bar{a}.c + \bar{b}.c + c.c$

Considering that $x.x = x$ and $x.\bar{x} = 0$, the previous expression may be written as:

$(\bar{a} + b + c).(\bar{a} + \bar{b} + c) = \bar{a}.\bar{a} + \bar{a}.\bar{b} + \bar{a}.c + \bar{a}.b + \cancel{b.\bar{b}} + b.c + \bar{a}.c + \bar{b}.c + c.c$

$= \bar{a} + \bar{a}.\bar{b} + \bar{a}.c + \bar{a}.b + b.c + \bar{b}.c + c$

The terms $\bar{a}.\bar{b}$, $\bar{a}.c$, $\bar{a}.b$ are absorbed by $\bar{a}$ and can be deleted. Moreover, the terms $b.c$, $\bar{b}.c$ are absorbed by c:

$\bar{a} + \cancel{\bar{a}.\bar{b}} + \cancel{\bar{a}.c} + \cancel{\bar{a}.b} + \cancel{b.c} + \cancel{\bar{b}.c} + c = \bar{a} + c$

Then:

$f(a,b,c) = (\bar{b} + \bar{c}).(\bar{a} + c) = \bar{a}.\bar{b} + \bar{b}.c + \bar{a}.\bar{c} + c.\bar{c} = \bar{a}.\bar{b} + \bar{b}.c + \bar{a}.\bar{c}$

At this point we may believe that the expression can't be further reduced, but it can. To do so, we multiply the term $\bar{a}.\bar{b}$ by $(c + \bar{c})$ (that may be done because $c + \bar{c} = 1$):

$f(a,b,c) = \bar{a}.\bar{b}.(c + \bar{c}) + \bar{b}.c + \bar{a}.\bar{c} = \bar{a}.\bar{b}.c + \bar{a}.\bar{b}.\bar{c} + \bar{b}.c + \bar{a}.\bar{c}$

The term $\bar{a}.\bar{b}.c$ is absorbed by $\bar{b}.c$ because $\bar{a}.\bar{b}.c + \bar{b}.c = \bar{b}.c.(\bar{a} + 1) = \bar{b}.c$; and for the same reason the term $\bar{a}.\bar{b}.\bar{c}$ is absorbed by $\bar{a}.\bar{c}$. Finally:

$f(a,b,c) = \bar{b}.c + \bar{a}.\bar{c}$

(The minimal expression requires three 2-operand operations. This expression can be implemented using two 2-input AND gates, one 2-input OR gate, and tree inverters)

2. $f(a,b,c,d) = a.b.c.d + \bar{a}.\bar{b} + a.b.\bar{c} + a.\bar{b} + a.\bar{c}$

We can take out $a.b$ as a common factor on both the first and the third product terms and $\bar{b}$ form the second and the fourth ones. This may be done because of the distributive property:

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

As we know, $x + \bar{x}.y = x + y$. Applying this property to $c.d + \bar{c}$, and bearing in mind that $\bar{a} + a = 1$, the previous expression may be expressed as:

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

We may apply the property $x + \bar{x}.y = x + y$ again to the two first terms. If we consider that $x = \bar{b}$ and $y = a.(d + \bar{c})$, then:

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

Finally, applying the distributive property to the term $a.(d + \bar{c})$, the function may be expressed as:

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

In this case we started with a Boolean expression requiring nine 2-operand operations and in the end we obtain an equivalent one that only requires four 2-operand operations. The simplified version may be implemented using two 2-input AND gates, one 3-input OR gate (or two 2-input OR gates) and 2 inverters.

EXERCICE 2

Draw the circuits that directly implement the following Boolean functions (Note: ***Do not simplify*** the functions before drawing them):

1. $f(a,b,c) = \overline{b \cdot c} \cdot \overline{a \cdot \bar{b} \cdot \bar{c}} \cdot \overline{a \cdot b \cdot \bar{c}}$

2. $f(a,b,c,d) = a.b.c.d + \bar{a}.\bar{b} + a.b.\bar{c} + a.\bar{b} + a.\bar{c}$

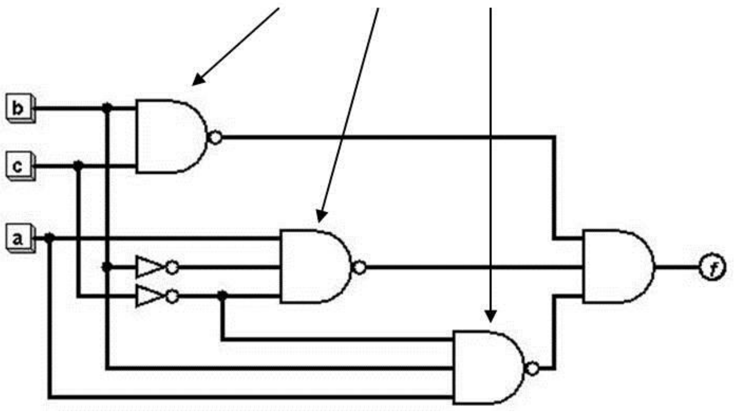
1. $f(a,b,c) = \overline{b \cdot c} \cdot \overline{a \cdot \bar{b} \cdot \bar{c}} \cdot \overline{a \cdot b \cdot \bar{c}}$

To see the function more clearly we will write it as:

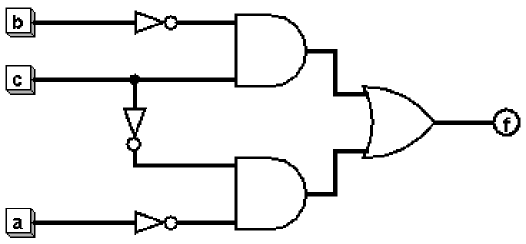
$$f(a,b,c) = \left(\overline{b \cdot c}\right) \cdot \left(\overline{a \cdot \bar{b} \cdot \bar{c}}\right) \cdot \left(\overline{a \cdot b \cdot \bar{c}}\right)$$

The direct implementation of the function using AND, OR and INV is quite simple and self-explanatory

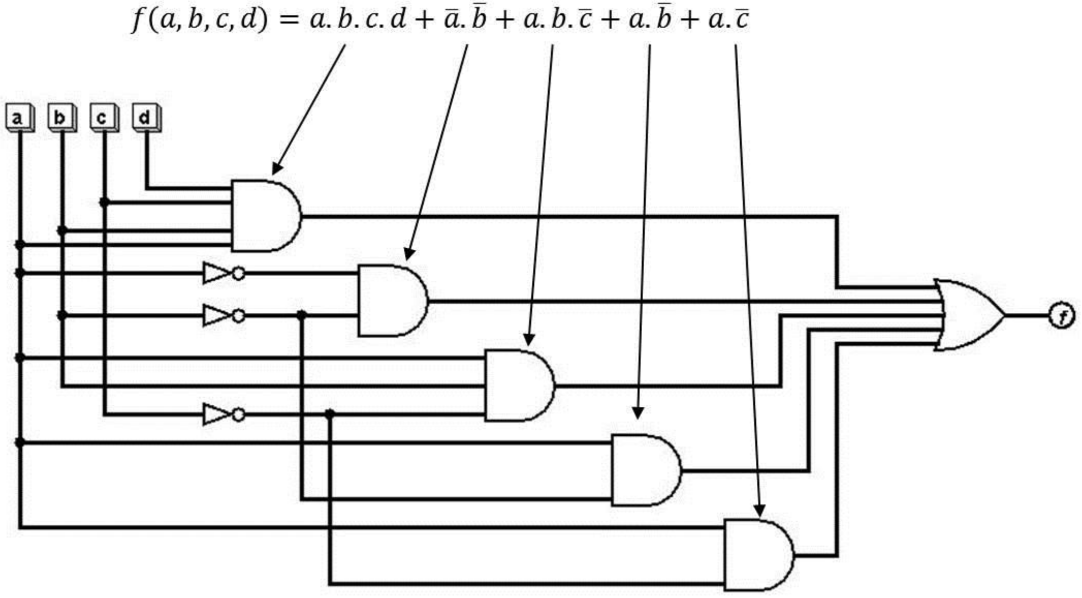
$$f(a,b,c) = \overline{(b.c)}.\overline{(a.\bar{b}.\bar{c})}.\overline{(a.b.\bar{c})}$$



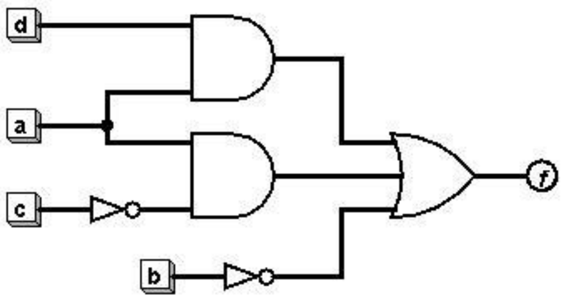
In the first exercise we have minimized this same Boolean expression, and we deduced that it was equivalent to $f(a,b,c) = \bar{b}.c + \bar{a}.\bar{c}$. This means that the previous circuit is equivalent to the following one, which only needs two 2-input AND gates, one 2-input OR, and 3 INV:



2. $f(a,b,c,d) = a.b.c.d + \bar{a}.\bar{b} + a.b.\bar{c} + a.\bar{b} + a.\bar{c}$

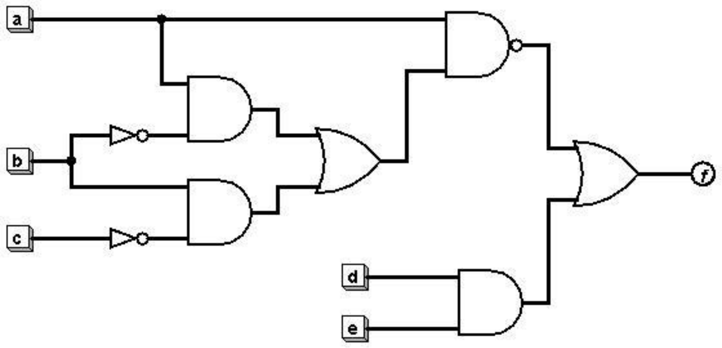

$f(a,b,c,d) = a.b.c.d + \bar{a}.\bar{b} + a.b.\bar{c} + a.\bar{b} + a.\bar{c}$
a
b
c
d
f

Again, in the first exercise we have simplified this Boolean function, this time as$f(a,b,c,d) = a.d + a.\bar{c} + \bar{b}$. So, the previous circuit is equivalent to the following one, which only requires two 2-input AND gates, one 3-input OR gate and 2 INV:

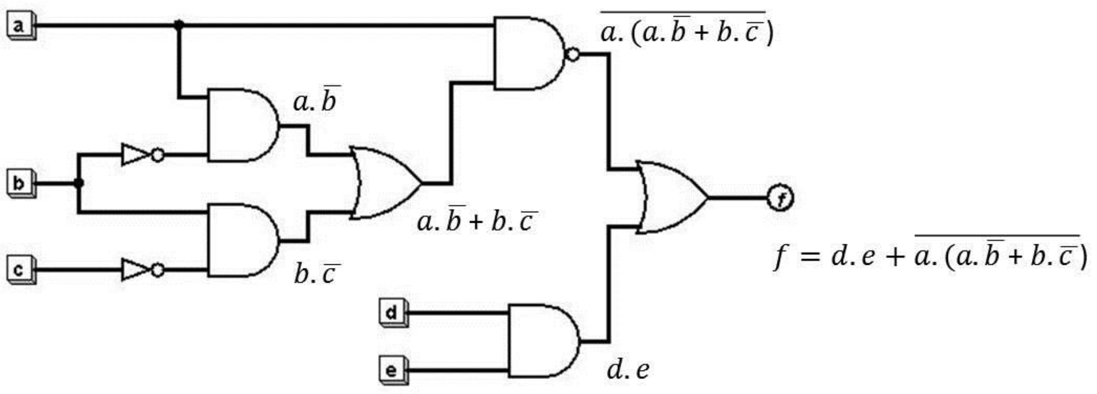


EXERCICE 3

Given the circuit from the following figure: (1) Write the equivalent Boolean function, (2) simplify the function using the properties and postulates of the Boolean algebra and (3) draw the final circuit.



(1)


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

So,

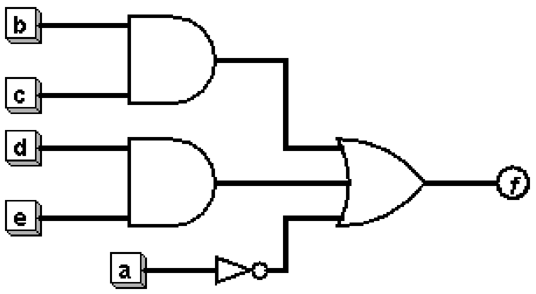
$$f = d.e + a.(a.\overline{b} + b.\overline{c})$$

(2) Let's simplify the expression …

$$f = d.e + \overline{a.(a.\overline{b} + b.\overline{c})} = d.e + \overline{a.\overline{b} + a.b.\overline{c}} = d.e + (\overline{a} + b).(\overline{a} + \overline{b} + c);$$

$$f = d.e + \overline{a} + b.c$$

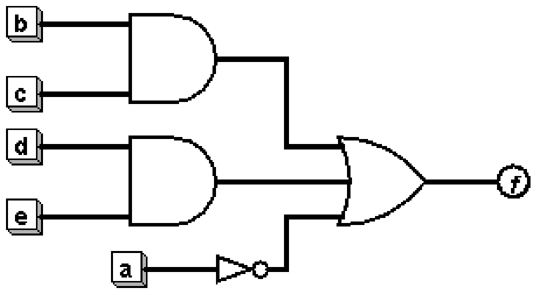
(3)



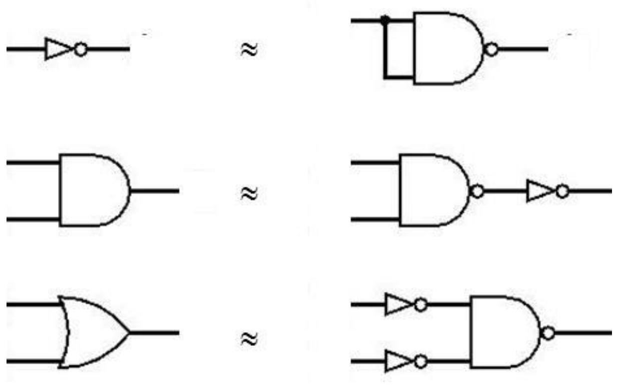
EXERCISE 4

Exercises 4 and 5 show how circuits could be built using only a combination of NAND or NOR gates. This may be useful in certain situations.

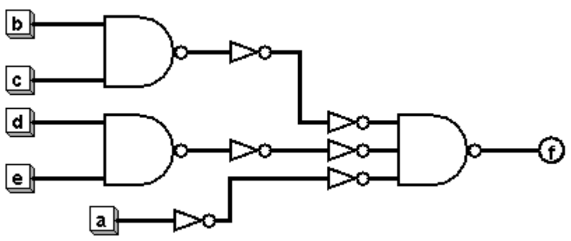
Implement the following circuit using only NAND gates and inverters.



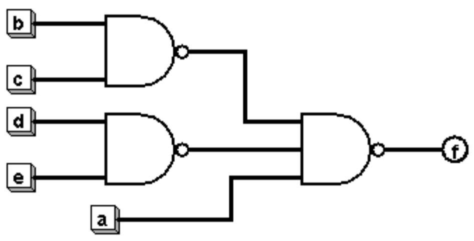
In lesson 2.3 we have seen that NAND gates are universal modules, meaning that any Boolean function can be implemented with only NAND gates. This is very easy to prove: we have seen that the logical addition, the logical product, and the negation, may be implemented using NAND gates as follows:



Replacing the gates in our circuit by the NAND equivalents gives:

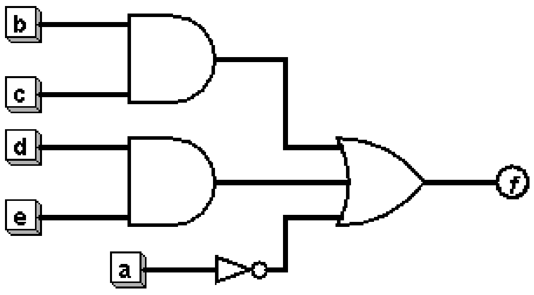


The circuit may be reduced taking into account that two inverters placed in line may always be deleted because to invert anything twice implies leaving it the same:

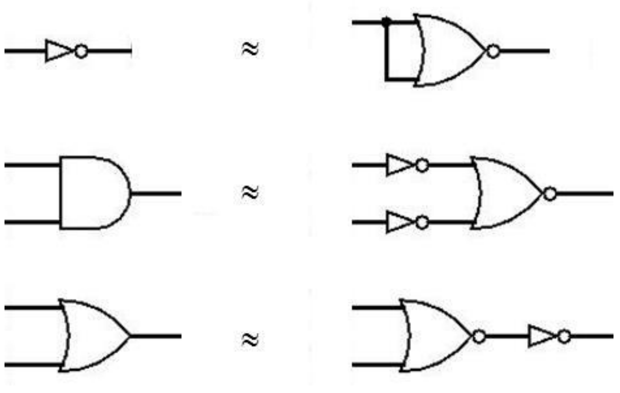


EXERCISE 5

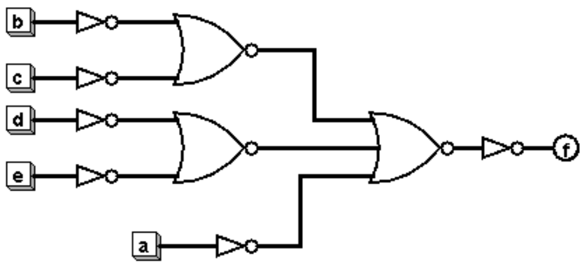
Implement the following circuit using only NOR gates and inverters.



The solution to the problem follows exactly the same steps as the previous exercise, with a few changes. The equivalences between AND, OR, INV and NOR logic gates are the following:



Solution:



Extension:

As we can see, the circuit implemented with NOR gates is more complex than the one built with NAND gates in the previous exercise.

In both cases the circuit implements a Boolean function, $f = d.e + \bar{a} + b.c$, that is expressed as a ***sum-of-products.***

Any Boolean function expressed as a sum-of-products is easier to implement using NAND gates than NOR gates (easier means that fewer gates are required). In the same way, Boolean functions expressed as products-of-sums are easier to implement using NOR gates than NAND gates.

As an example, the function of the circuit in the problem statement may be expressed as a product of sums applying the distributive function:

$$x + y.z = (x + y).(x + z)$$

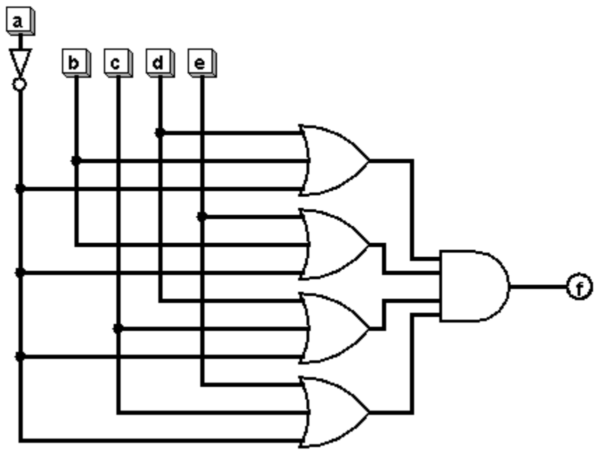
Using $x = \bar{a} + b.c,\ \ y = d,\ \ z = e$, we would have:

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

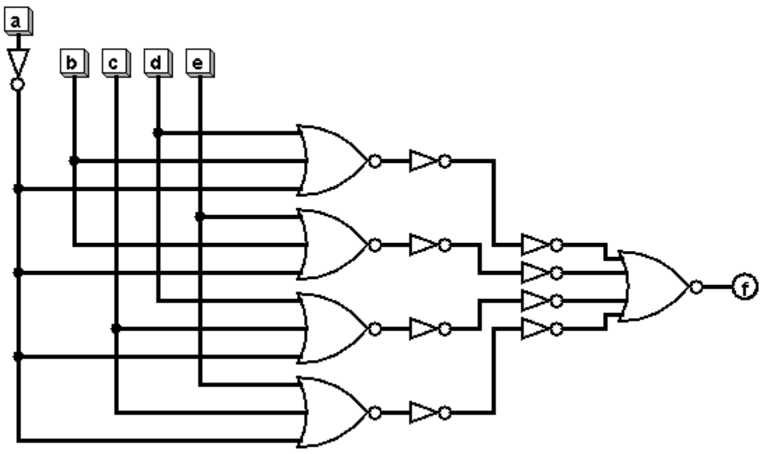
If we repeat the same process for the two parentheses, using $x = \bar{a} + d,\ \ y = b,\ \ z = c$ for the first one and $x = \bar{a} + e,\ \ y = b,\ \ z = c$ for the second, we would obtain the original function expressed as a product-of-sums:

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

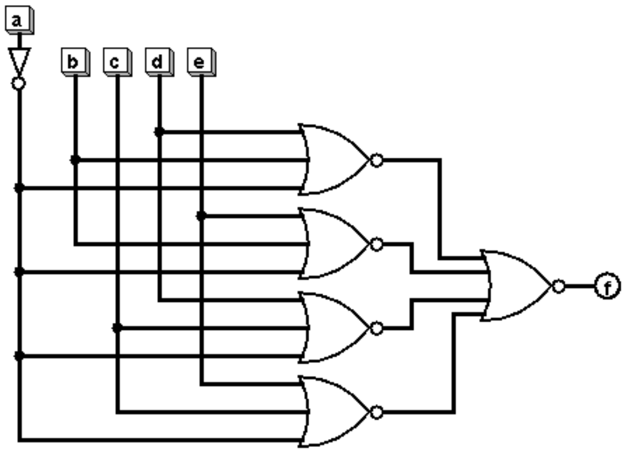
The previous function may be drawn as:



Now, let's do the same as before and replace AND and OR gates with their equivalent NOR gates:



If we delete the series of two inline inverters:



As we can see, the final circuit consists of four 3-input NOR gates and one 4-input NOR gate plus 1 inverter (6 gates in total) while the original circuit needed two 2-input and one 3-input NOR gates plus 6 inverters; that is, 9 logic gates in total.