EE 2720

Handout #12.

EE 2720

# • Combinational-Circuit Analysis.

## Definition of analysis.

– Given a logic diagram of a logic circuit:
  - Find out what the circuit does.
  - Find out it's Boolean function.
  - Obtain a formal description of the circuit's logic function.

Once we have a formal description of the circuits logic function, a number of other operations are possible:

- We can determine the behavior of the circuit for various input combinations.
- We can manipulate an algebraic description (describing the function of the circuit) to obtain different circuit structures for the logic function; (for example realizing the circuit using NOT, AND, OR gates, or only NAND gates or only NOR gates etc; we have seen that before).
- We can transform an algebraic description into a standard form corresponding to an available circuit structure. For example, a sum-of-products expression corresponds directy to the circuit structure used in PLDs (programmable logic devices; we might study them if time allows).
- We can use the results of the analysis of a circuit to analyze a larger circuit that includes this circuit. In other words, we can decompose a large circuit that might be difficult to be analyzed into

smaller circuits, analyze the smaller circuits and put the entire picture together.

If you are given a logic diagram for a combinational logic circuit, there are several ways to get a formal description of the circuit's logic function. The most primitive functional description is the truth table. A very primitive way to derive the truth table for the circuit under analysis is to use only the basic axioms of switching algebra. The truth table will be the final result. If the circuit has $n$ inputs, there are $2^n$ input combinations. For each input combination, determine all the gate outputs produced by that input, propagating information from the circuit inputs to the circuit outputs.

Consider for example the logic circuit of figure 1 below.

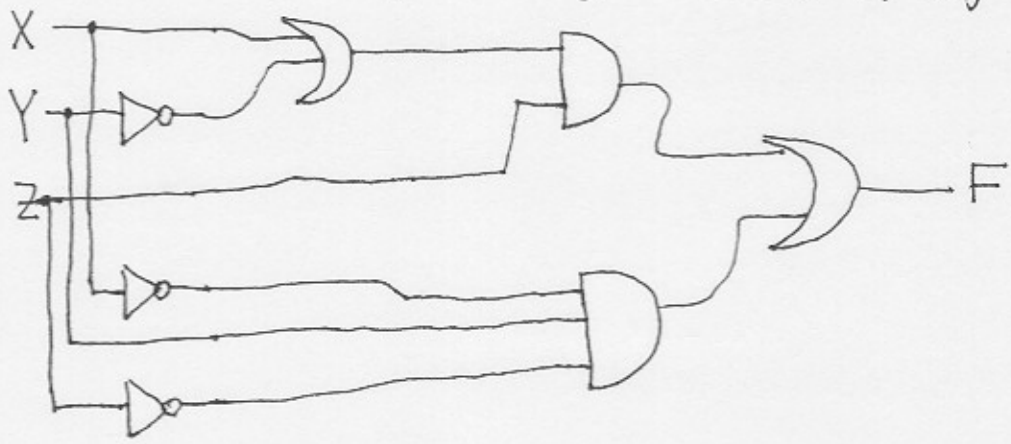


Figure 1; (A logic circuit with three inputs).

I will now derive the truth table for the logic circuit of fig. 1 using the primitive way that I described above (the exhaustive approach). The result is shown on the next page.

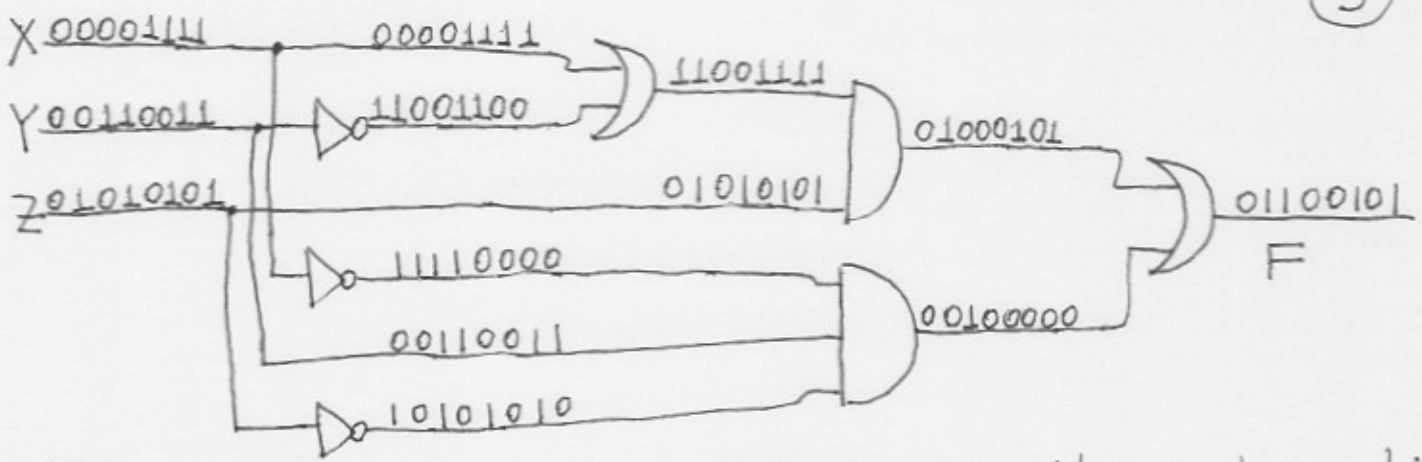


Figure 2; (Gate outputs created by all input combinations for the circuit of fig.1. Here I applied all8 input combinations XYZ 000, 001, 010, ..., 111).

We can now get the truth table for the logic circuit of fig.1 by looking at fig. 2. The truth table is shown in table 1 below

| X | Y | Z | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Table 1; (truth table for the logic circuit of figure 1).

After we get the truth table for the circuit, we can write a logic expression for this circuit, for example the canonical sum or canonical product.

For example the canonical sum for the circuit of fig. 1 is (we get it from table 1)

$$F = \Sigma_{X,Y,Z}(1,2,5,7) = X' \cdot Y' \cdot Z + X' \cdot Y \cdot Z' + X \cdot Y' \cdot Z + X \cdot Y \cdot Z$$

The canonical product for the same circuit is

$$F = \Pi_{X,Y,Z}(0,3,4,6) = (X+Y+Z) \cdot (X+Y'+Z') \cdot (X'+Y+Z) \cdot (X'+Y'+Z)$$

Analyzing a logic circuit by deriving its truth table using the primitive way that I described above is not a good idea. The number of input combinations of a logic circuit grows exponentially with the number of inputs, so this exhaustive approach can become very time consuming; (don't do it). Instead, one can use an algebraic approach; (we have done this many times in previous handouts). What you do is start at the inputs and find logic expressions of the circuit's outputs. This is shown below in figure 3 for the logic circuit of figure 1.

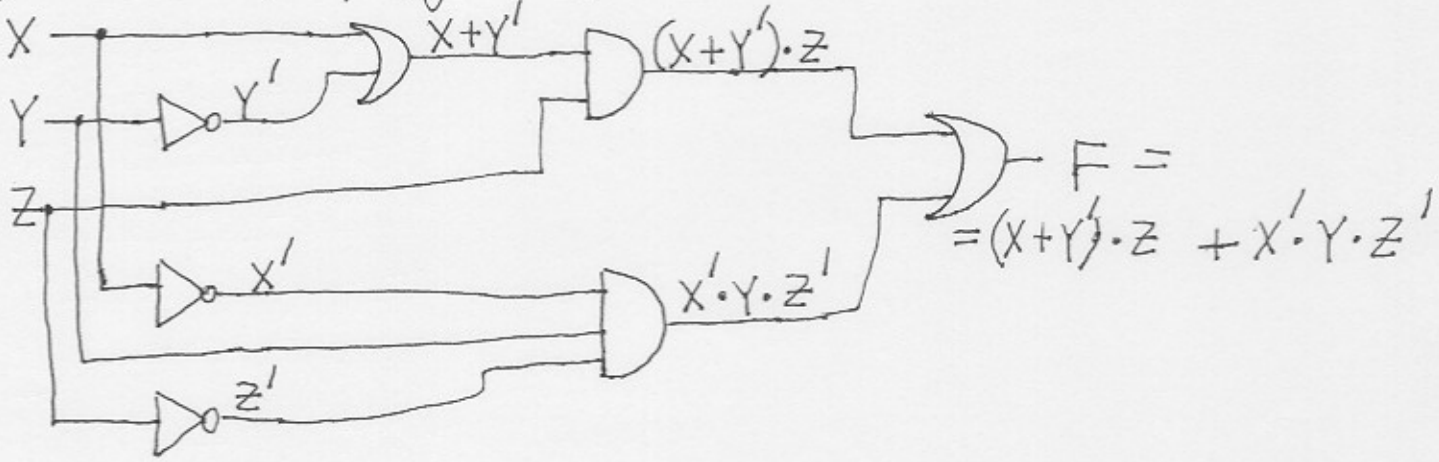


Figure 3; (the same logic circuit of figure 1 with logic expressions).

The output of the above logic circuit is

$$F = (X+Y') \cdot Z + X' \cdot Y \cdot Z' \quad (1)$$

You can transform the previous circuit to many other forms. For example by multiplying out F of eq.(1) you can get an equivalent AND-OR logic circuit. Multiplying out we get

$F = (X+Y') \cdot Z + X' \cdot Y \cdot Z' = X \cdot Z + Y' \cdot Z + X' \cdot Y \cdot Z'$ or

$$F = X \cdot Z + Y' \cdot Z + X' \cdot Y \cdot Z' \quad (2)$$

Equation (2) suggests figure 4 below.

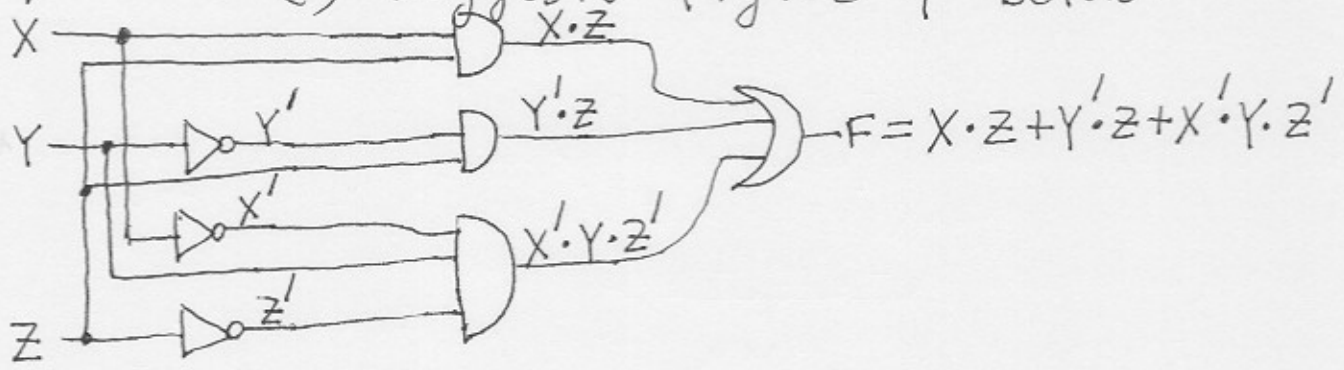


Figure 4; (equivalent logic circuit to this of fig 3. This is an AND-OR logic circuit).

Note: We have done the above task of translating a logic circuit into an AND-OR logic circuit before. So this is not new to you. By now it should be trivial.

Similarly, by factoring F of eq.(1) we can get an equivalent OR-AND circuit, equivalent to the logic circuit of figure 3; (we have done the above task of translating a logic circuit into an OR-AND logic circuit before. So this is not new to you).

Starting from eq.(1) and factoring we get

$$F = \underbrace{(X+Y')}_{A} \cdot \underbrace{Z}_{B} + \underbrace{X' \cdot Y}_{A'} \cdot \underbrace{Z'}_{B'} = A \cdot B + A' \cdot B' = A' \cdot B' + A \cdot B =$$

$= \Sigma_{A,B}(0,3) = \Pi_{A,B}(1,2) = (A+B')\cdot(A'+B) =$

$= (X+Y'+Z')\cdot(X'\cdot Y+Z) = (X+Y'+Z')\cdot(Z+X'\cdot Y) =$

$= (X+Y'+Z')\cdot(Z+X')\cdot(Z+Y)$ (by applying theorem T8').

$= (X+Y'+Z')\cdot(X'+Z)\cdot(Y+Z)$ or

$F = (X+Y'+Z')\cdot(X'+Z)\cdot(Y+Z)$ (3)

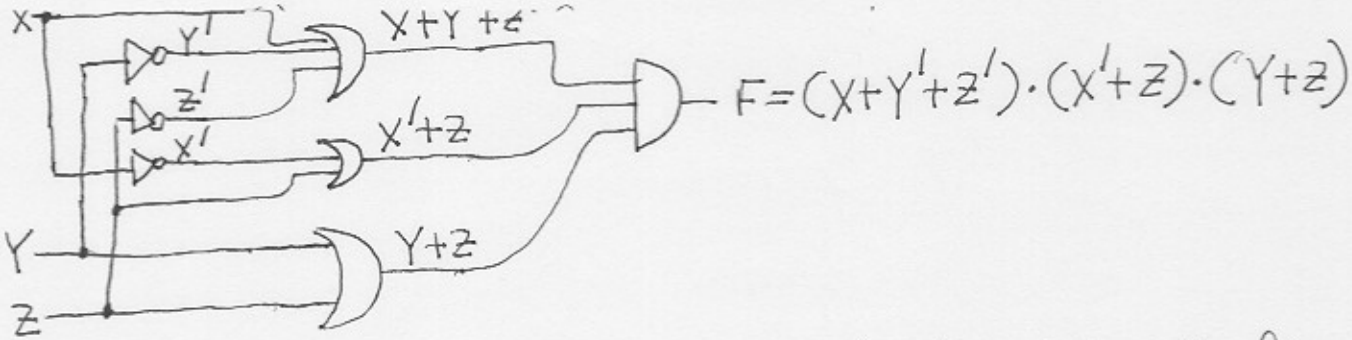


Figure 5; (equivalent logic circuit to this of fig. 3. This is an OR-AND logic circuit).

Note: We can also provide equivalent realizations for the logic circuit of fig 3. using only NAND gates, using only OR and NAND gates, using only NOR and OR gates, using only NOR gates, using only AND and NOR gates or using only NAND and AND gates. This is true for any logic circuit. We have done this before. See handout # 10 for details; (These six extra realizations are on top (extra) of the AND-OR and OR-AND that we provided in figures 4 and 5).

One last example of logic circuit analysis follows. The circuit is shown by figure 6 on next page and includes NAND and NOR gates.

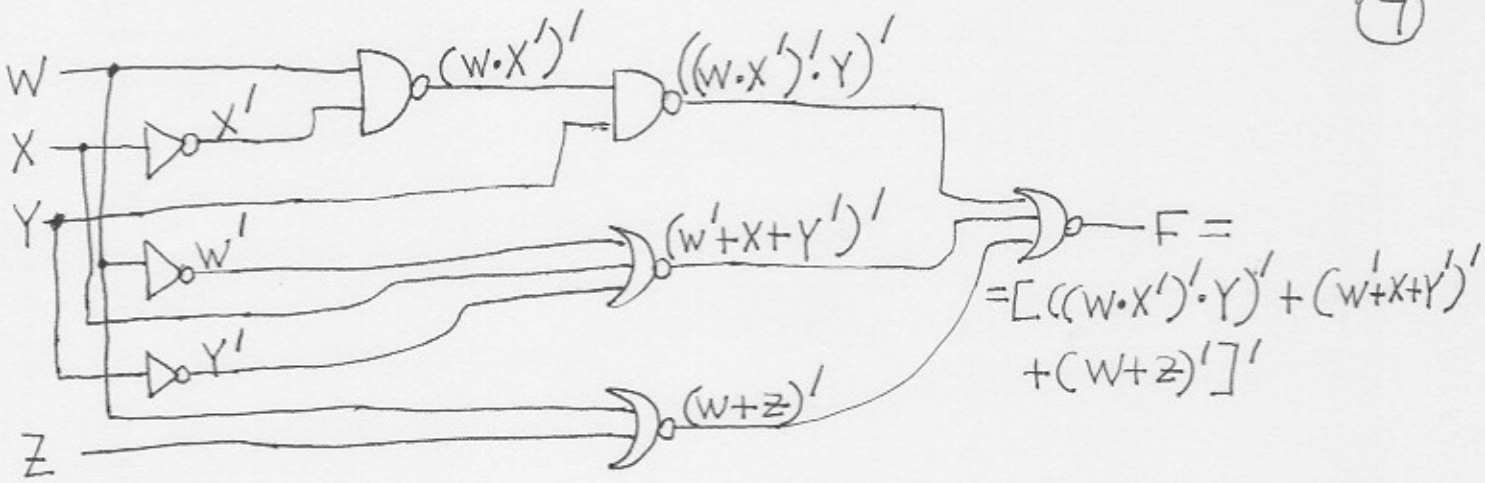


Figure 6; (a logic circuit including NAND and NOR gates).

The analysis of the above circuit of figure 6 is a little bit difficult because of the complemented expressions produced by the NAND and NOR gates. However, the output of the circuit F can be simplified by using the generalized DeMorgan's theorem; (we have done such things before, so again nothing new; redundant work). Here is how.

$$F=[((W\cdot X')'\cdot Y)'+(W'+X+Y')'+(W+Z)']'=$$
$$=[((W'+X)\cdot Y)'+W\cdot X'\cdot Y+W'\cdot Z']'=$$
$$=[(W'+X)'+Y'+W\cdot X'\cdot Y+W'\cdot Z']'=$$
$$=(W\cdot X'+Y'+W\cdot X'\cdot Y+W'\cdot Z')'=$$
$$=(W\cdot X')'\cdot(Y')'\cdot(W\cdot X'\cdot Y)'\cdot(W'\cdot Z')'=$$
$$=(W'+X)\cdot Y\cdot(W'+X+Y')\cdot(W+Z). \quad (4)$$

Now F of eq. (4) is much more simplified than before and the analysis of the logic circuit is easier.

The same simplification can be obtained by applying the graphical approach which by now you know very well. Two figures follow on the next page. They are figures 7 and 8 and the circuits represented by them are equivalent to the circuit of fig. 6.

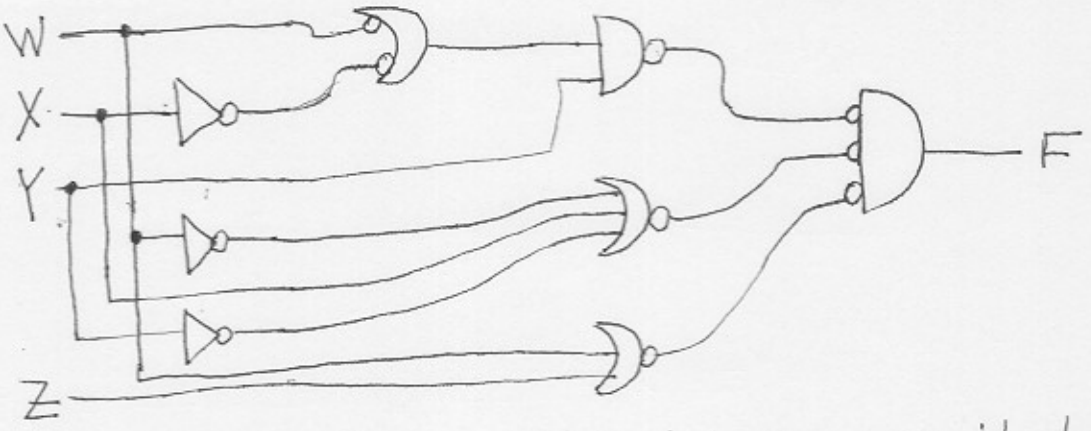


Figure 7; (one equivalent logic circuit to the logic circuit of figure 6).

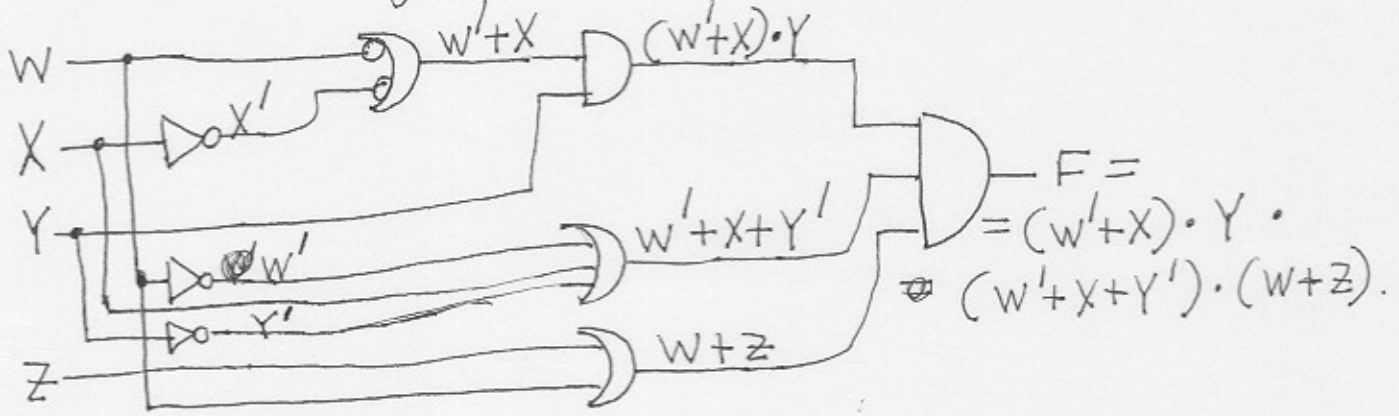


Figure 8; (another equivalent logic circuit to the logic circuit of figure 6).

As seen, the output F of the circuit of figure 8 is the same as this of eq. (4). We obtained the same result with the graphical approach as applying DeMorgan's theorem but with less effort.

Note: By multiplying out or factoring F of eq. (4), (which is the same as the output of the circuit of fig. 8), we can get equivalent AND-OR or OR-AND realizations of the logic circuit; (we have studied this subject before, so do it if you want). Besides these, we can get six more equivalent circuit realizations as mentioned in the note below figure 5 on page 6 of this handout; (again we did this before in handout#10).

Conclusion: This concludes the presentation of the subject of combinational circuit analysis. To conclude, we basically didn't learn too many new things in this handout. Almost all the topics covered in this handout were materials that were basically presented before in previous handouts.