Secure Multi-Party Computation

Assignment 1 - Truth-table and circuits for sign activation

Or Dinar
Liad Ackerman

06.01.2024

Exercise 1

Write the Truth-table for the function specified in Equation 2. The Truth-table specifies the function values for all possible assignments to $\overrightarrow{a}, \overrightarrow{x} \in \{0, 1, 2, 3\}$

Equation 2:

$$f_{a,4}(x) = \begin{cases} 1 & \text{if } ax \geq 4 \\ 0 & \text{otherwise} \end{cases} \qquad for \quad a \in \{0, 1, 2, 3\}$$

The Truth-table for given equation will look like the following table:

	x=0	***x=1***	***x=2***	***x=3***
a=0	***0***	***0***	***0***	***0***
a=1	***0***	***0***	***0***	***0***
a=2	***0***	***0***	***1***	***1***
a=3	***0***	***0***	***1***	***1***

Figure 1: Truth-table for equation 2.

Exercise 2

Write a Boolean circuit computing the function specified in Equation 3, with XOR and AND gates.

Equation 3:

$$f_{\overrightarrow{a},4}(x_1, x_2) = \begin{cases} 1 & \text{if } a_1x_1 + a_2x_2 \geq 4 \\ 0 & \text{otherwise} \end{cases} \qquad for \quad \overrightarrow{a} \in \{0,1,2,3\}^2$$

In order to construct a Boolean circuit that represents *equation* 3, we first need to construct Boolean circuits for multiplying 2-bit numbers, and for adding of 4-bit numbers.

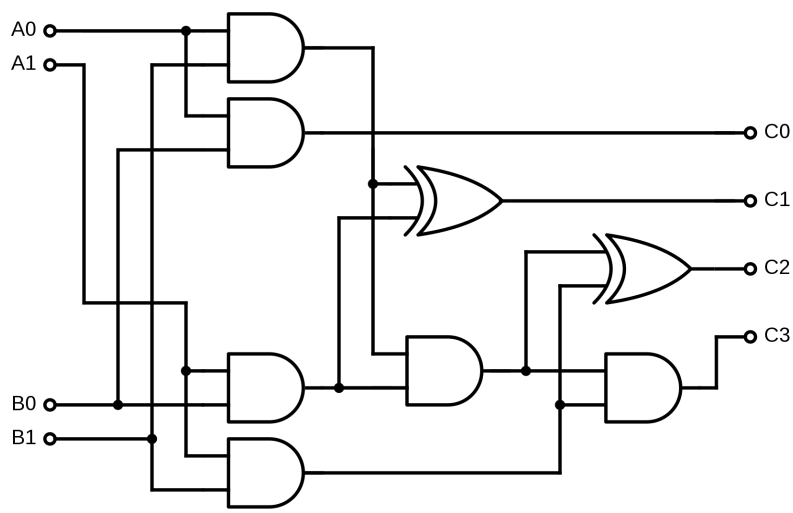

Figure 2: A multiplication Boolean circuit for multiplying 2-bit numbers, as seen in *Binary Multiplier - Wikipedia*[1].

The product of 2-bit numbers A and B, whereas A_0, A_1 and B_0, B_1 are the bits of A and B respectively, Is the number C whereas C_0, C_1, C_2, C_3 are the bits of C.
We will refer to the Boolean circuit from Figure 2 as 2-bit MULTI.

In order to make an addition Boolean circuit for two 4-bit numbers we will have to use OR gates. Since we can not use OR gates we will represent them using only AND and XOR gates. We'll use the Truth-tables of AND and XOR to infer how the wanted Boolean circuit will look like.

AND	0	1
0	0	0
1	0	1

XOR	0	1
0	0	1
1	1	0

Figure 3: Truth-tables for XOR and AND gates.

OR	0	1
0	0	1
1	1	1

Figure 4: Truth-table for OR gates.

The following Boolean circuit has the same Truth-table as an OR gate:

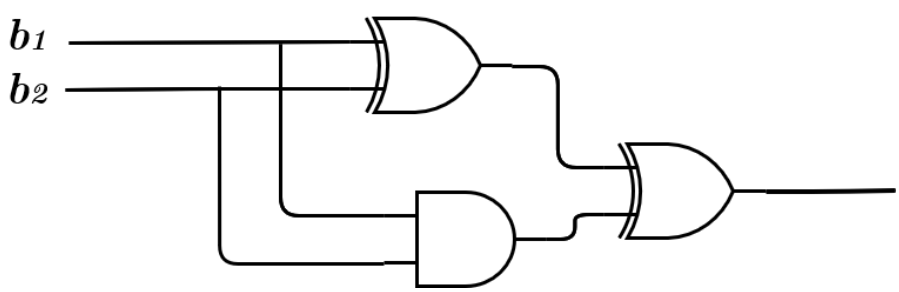

Figure 5: OR gate represented by AND and XOR gates only

Now that we can use OR gates we can create an addition boolean circuit that will be referred to as 4-bit ADDER. See below:

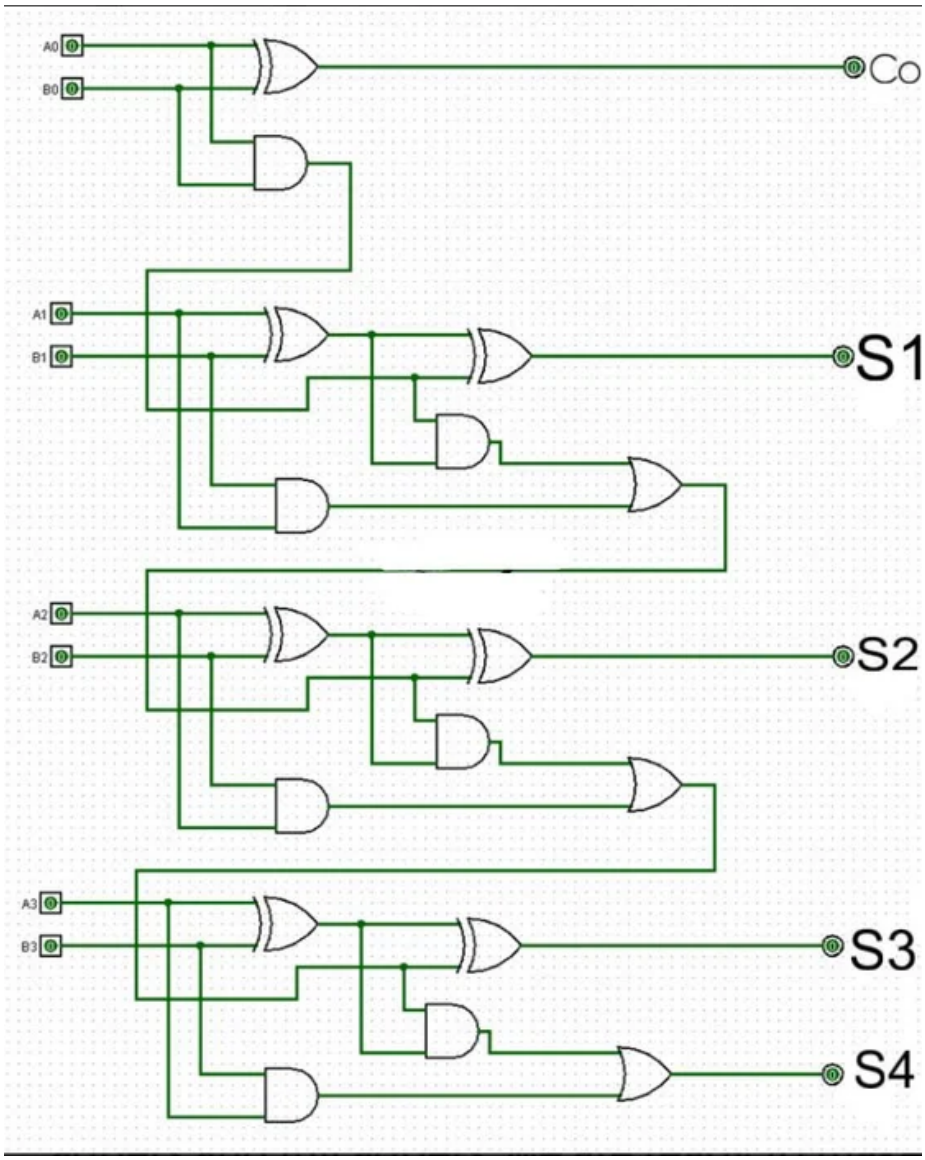

Figure 6: Addition boolean circuit as seen in *instructables.com − 4 Bit Binary Adder* [2].

Observation: a binary number is greater or equal than 4 (in decimal) when either the 3rd or 4th bits are 1, meaning we will take the product of all calculations and run the 3rd, 4th and the carry out bit through an OR gate, that will guarantee us the product is greater or equal to 4.

The final Boolean circuit will be the following:

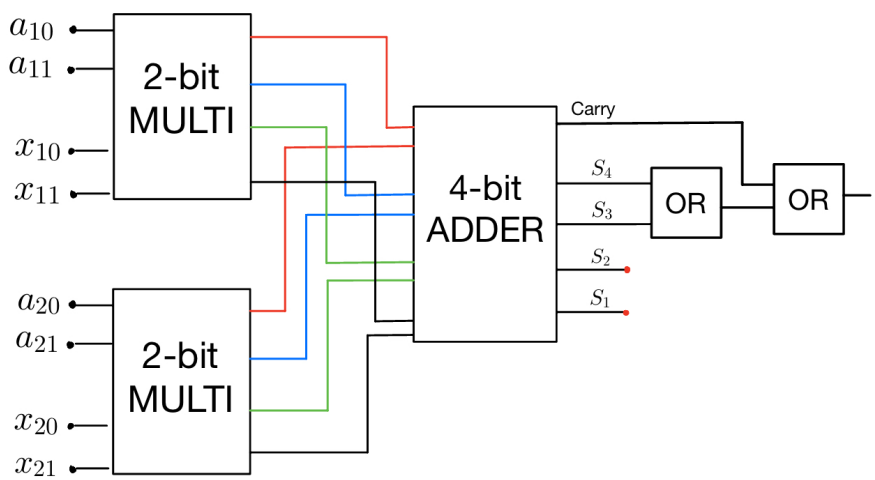

Figure 7: Boolean circuit that represents *Equation* 3.

Exercise 3

In this exercise we are being asked to create an arithmetic circuit under the field $GF(11)$, where we have addition and multiplication gates mod 11. Equation is still equation (3):

Equation 3:

$$f_{\overrightarrow{a},4}(x_1, x_2) = \begin{cases} 1 & \text{if } a_1x_1 + a_2x_2 \geq 4 \\ 0 & \text{otherwise} \end{cases} \qquad for \quad \overrightarrow{a} \in \{0,1,2,3\}^2$$

Note: The gates in the circuit are closed under the field GF(11), which means and addition gate is mod 11 and so is the multiplication gate.
In the circuit, first, we add the multiplications of a_1x_1 and a_2x_2 like so:

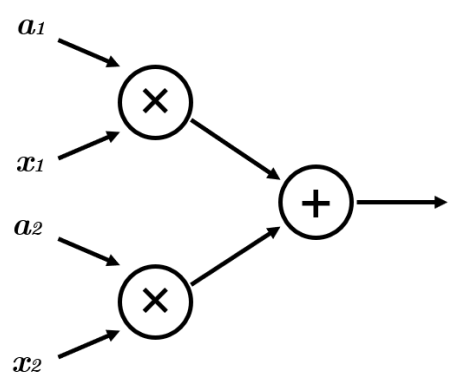

Figure 8: Adding the product of a_1x_1 and a_2x_2

Now we have a value that we can compare to 4 (mod 11), we will denote it as SUM. After discussing the issue with Dr. Akavia we reached the conclusion that we can use Fermat's little theorem in order to compare both sides of the equation. The claim of the theorem is the following:

$$a^{p-1} \equiv \begin{cases} 0 \quad (mod \quad p) & \text{if } a = 0 \\ 1 \quad (mod \quad p) & \text{otherwise} \end{cases} \qquad for\ p\ is\ prime$$

Furthermore, we can say that for every $k \in GF(11)$:
$(SUM - k)^{10} \equiv 0 \quad (mod \quad 11)$ if and only if SUM = k. That means we can implement an arithmetic circuit to compare two numbers in a field, therefore we can compare SUM to each of the values in $\{4, 5, 6, 7, 8, 9, 10\}$.
Note that $-k \equiv 11 - k \ (mod\ 11)$. That means that instead of preforming $(SUM + (-k))\ for\ k \in \{4, 5, 6, 7, 8, 9, 10\}$, we can preform $(SUM + k)$ $for\ k \in \{7, 6, 5, 4, 3, 2, 1\}$. The implementation of such circuit is the following:

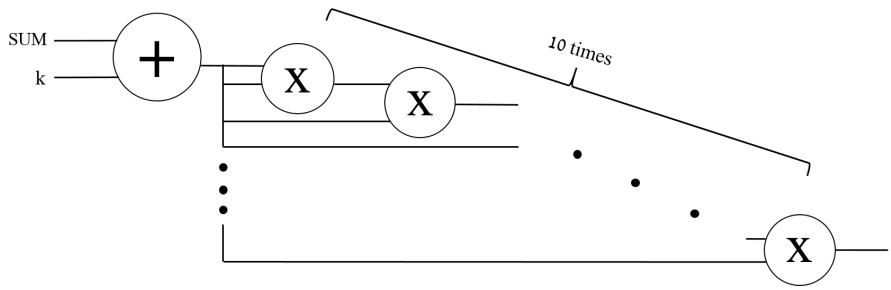

Figure 9: Implementation of $(SUM + k)^{10}$. It returns 0 if $SUM = k$, and 1 otherwise.

Now, we can implement this operation seven times, one for each $k \in \{7, 6, 5, 4, 3, 2, 1\}$. As said before - the calculation in Figure 9 returns 0 if $SUM = k$, and 1 otherwise. which means that after adding the return value of each of the seven operations (we will refer it as $TOTAL$), we should get $TOTAL = 7$ if $SUM \neq -k$ for every $k \in \{7, 6, 5, 4, 3, 2, 1\}$ and 6 otherwise (Because there can be at most one equality). The above can be written as the equation:

$$TOTAL \equiv \begin{cases} 7 & \text{if } SUM < 4 \\ 6 & SUM \geq 4 \end{cases}$$

Which brings us closer to the solution. We would like to map $7 \mapsto 0$ and $6 \mapsto 1$. One way we can do that, is by adding 5 to TOTAL (which maps $7 \mapsto 0$ and $6 \mapsto 10$) and using Fermat's little theorem again to conclude that:

$$a^{10} \equiv \begin{cases} 0 \pmod{11} & \text{if } a = 0 \\ 1 \pmod{11} & \text{if } a = 10 \end{cases}$$

In conclusion - we need to return $(TOTAL + 5)^{10}$. The implementation of such circuit is the following:

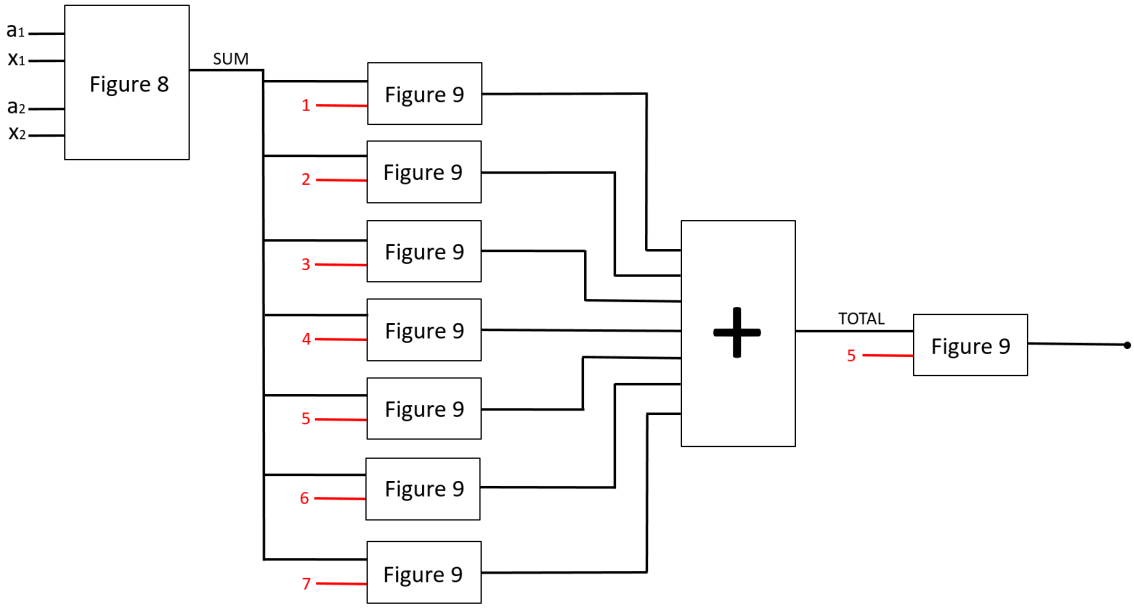

Figure 10: Arithmetic circuit that represents *Equation* 3.

Exercise 4

i Circuit Depth The maximal length of an input is the largest path an input can traverse the circuit in order to get to the end of it, technically all paths are the same so we will calculate the path from the input to the output. The depth of Figure 8 is 3. The depth of figure 9 is 11, and it appears twice in a path. Therefore the depth is $3 + 11 + 1 + 11 =$**26 gates**.

ii Circuit Size We will sum the total amount of gates in the circuit to conclude the size, in figure 9 we have 10 multiplication gates and one addition gate, The figure appears 8 times. in figure 8 we have 3 gates. And we also have the addition gate at the end. All in all we have $11 \times 8 + 3 + 1 =$ **92 gates**.

iii x-depth The multiplicative depth is the amount of multiplications in a maximal path. The amount of multiplications in Figure 8 is 1, and the amount of multiplications in Figure 9 is 10, therefore the x-depth is $1 + 10 + 10 =$ **21 gates**.

iv # MULT The amount of multiplication gates in the arithmetic circuit, since there are 8 Figure 9's (10 multiplication gates in each), and one Figure 8 (2 multiplication gates), the amount of multiplication gates is $10 \times 8 + 1 \times 2 =$ **82 multiplication gates**.

1 Bibliography

[1] - https://en.wikipedia.org/wiki/Binary_multiplier - Binary multiplier - Wikipedia.
[2] - https://www.instructables.com/4-Bit-Binary-Adder-FINAL-PROJECT/ - 4 Bit Binary Adder.
[3] - Dr. Akavia's proposal for using Fermat's little Theorem.