Experiment 2: Comibnational Circuits 2, Scrabble

Karrthik Arya, Roll Number: 200020068
EE-214, WEL, IIT Bombay
August 18, 2021

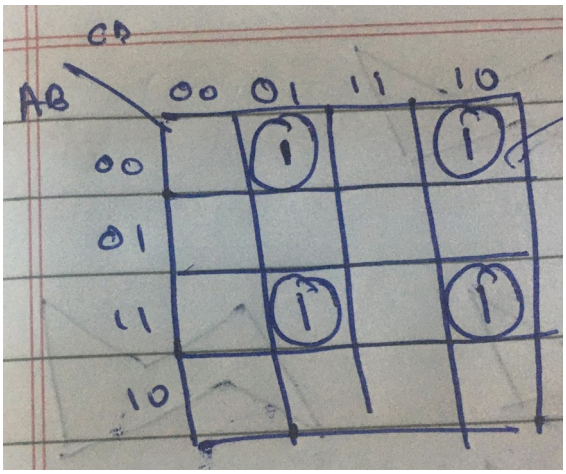
Overview of the experiment:

In this experiment we were asked to make a system to detect alphabets that have 3 points in scrabble. We had taken only the first 16 alphabets and encoded them as binary coded decimals with 4 bits like ‘A’ is as 0000, ‘B’ as 0001, ‘C’ as 0010 and so on.

What I then did was to make the K-map for 4 variables using the tracefile by marking the appropriate terms on the K-map. Then from it I deduced the Boolean expression. Then I went on to rearrange and simplify the expression in the appropriate form using the theorems of Boolean Algebra.

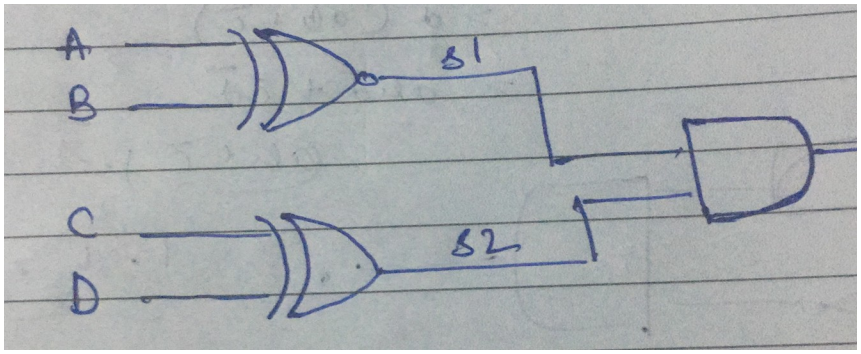
Approach to the experiment:

The first thing I did is to draw the K-map. Here’s the K-map that I drew:



From this I got the boolean expression as ABC’D + ABCD’ + A’B’C’D + A’B’CD’ .
Then I tried to simplify this using Boolean Algebra.
ABC’D + ABCD’+ A’B’C’D + A’B’CD’
= AB(C’D+ CD’) + A’B’(C’D + CD’)
= (AB + A’B’)(C’D + CD’)
Now, AB + A’B’ is XNOR gate and C’D + CD’ is the XOR gate. Sowe finally get
(A XNOR B) AND (C XOR D)

Thus we can design the circuit using these 3 gates. Here's the circuit that I made:



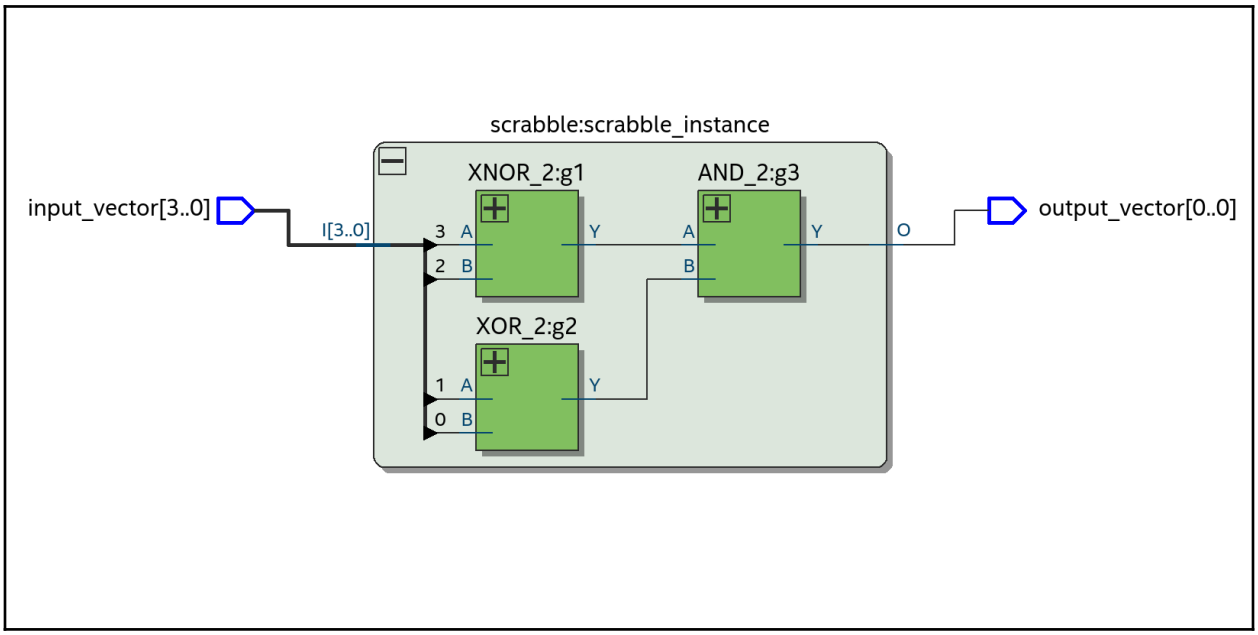
Design document and VHDL code if relevant:

Here's the code for the main architecture my circuit:

```
entity scrabble  is
  port (I: in std_logic_vector(3 downto 0); O: out std_logic);
end entity scrabble;
architecture Struct of scrabble is
  signal s1, s2: std_logic;
begin
  g1: XNOR_2 port map (A => I(3), B => I(2), Y => s1);
  g2: XOR_2 port map (A => I(1), B => I(0), Y => s2);
  g3: AND_2  port map (A => s1, B => s2, Y => O);
end Struct;
```

Here s1 and s2 are the two signals going from the XOR and XNOR gates to AND. I is the input where I(3), I(2), I(1), I(0) are A, B, C, D respectively. I take A XNOR B, C XOR B and then AND the outputs.

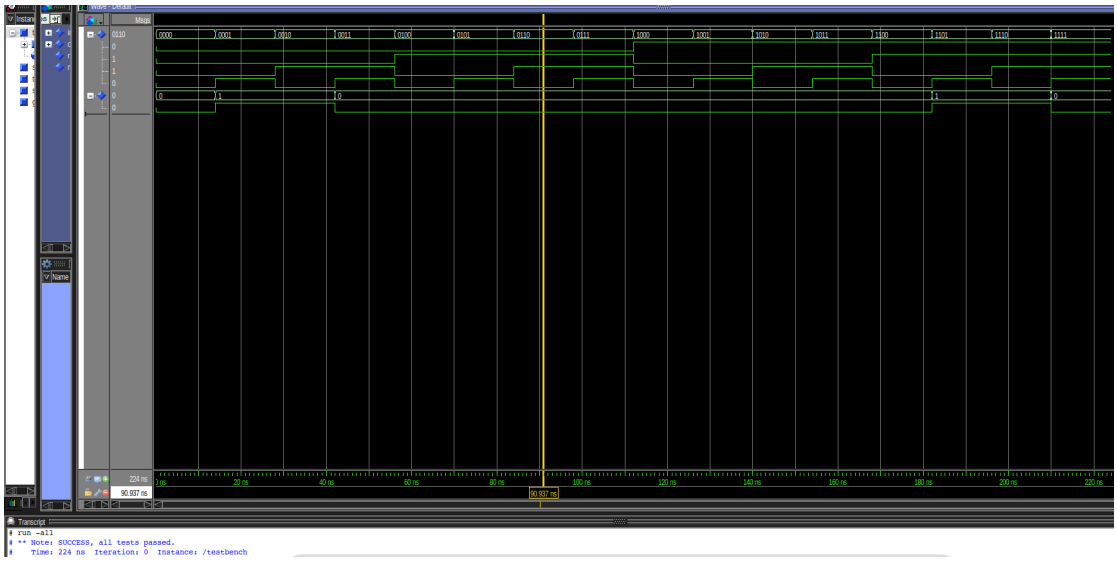
RTL View:



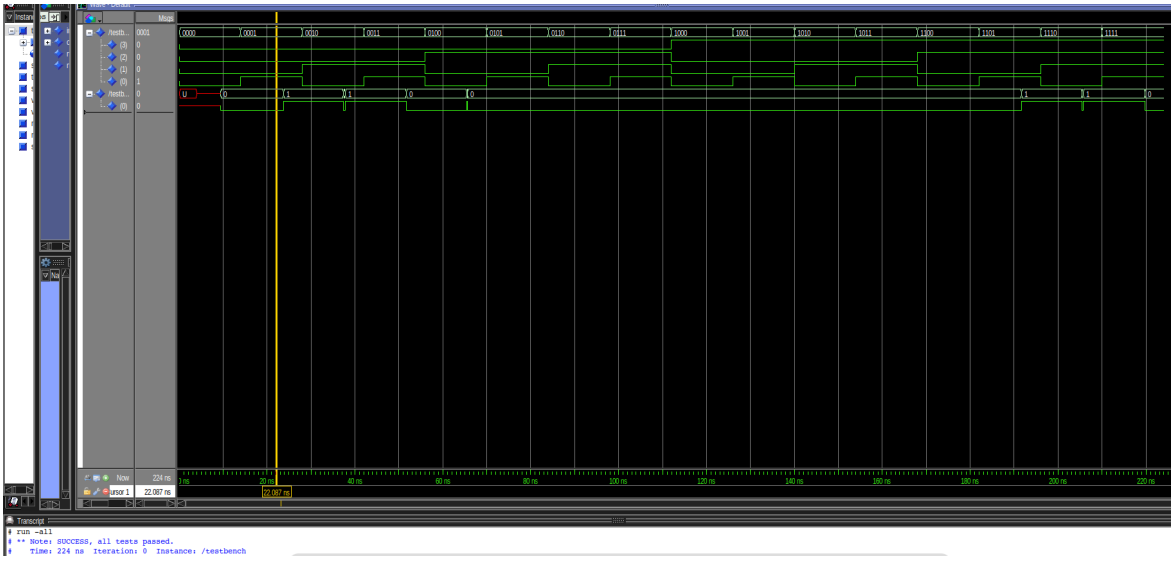
DUT Input/Output Format:

The input is of the form ABCD O 1. Here ABCD is the encoded form of the alpahbets like 0000 for letter “A”, 0001 for letter “B”, 0010 for letter “C” and so on upto 1111 for letter “P”. Here O is the output which should be 1 for 3 point alphabets while 0 for others. The 1 is the mask bit to see whether a test case need to be checked or not.
Here are a few test cases:
0000 0 1
0001 1 1
0010 1 1
0011 0 1
0100 0 1

RTL Simulation:



Gate-level Simulation:



Krypton board*:

N/A

Observations*:

N/A

References:

N/A

* To be submitted after the tutorial on ”Using Krypton.