Experiment 3: Prime Detector

Annirudh K P
210070009

August 23, 2022

1 Overview of the experiment

In this experiment, we started working on more designs using structure modelling on VHDL. The problem statement of this experiment is to design a Prime Detector, whose function we derive using the K-Maps. The objective of this experiment was to understand the Quartus Design Flow, work with the Xen10 Board, and give us hands on experience over different technical glitches/problems we may face in this piece of software which has been made unwantedly hard.

2 Experimental Set-up

2.1 Design Requirements

2.1.1 Prime Detector

The Prime Detector takes in every decimal number in its binary form (4 bits). The output is a simple 0 if not prime, and 1 if it is prime.

ABCD	O
0000	0
0001	0
0010	1
0011	1
0100	0
0101	1
0110	0
0111	1
1000	0
1001	0
1010	0
1011	1
1100	0
1101	1
1110	0
1111	0

Figure 1: Prime Detector Truth Table

2.2 Design Schematics

The following design schematics are shown. The min-function was first got from K-Map, and then the design was implemented. Also we have to design 3 input AND gates and 4 input OR gate for the same additionally.

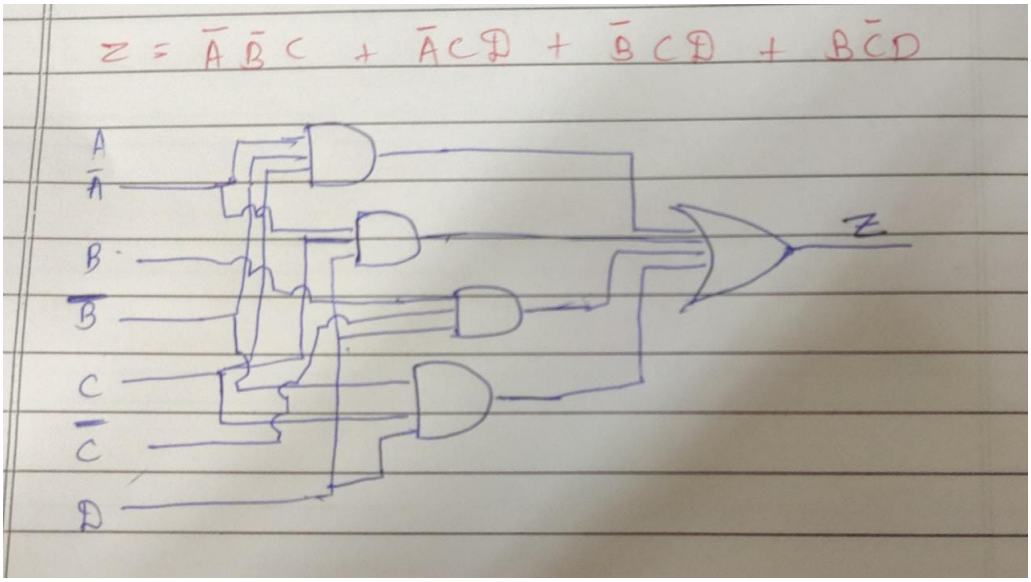

Figure 2: Prime Decider Design

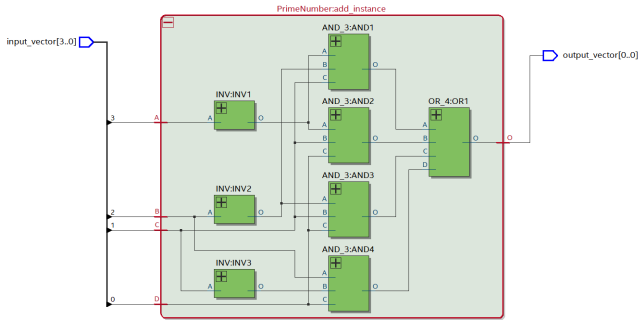

Figure 3: Prime Detector RTL Design

2.3 Description of Components

2.3.1 Prime Detector

```
library ieee;
use ieee.std_logic_1164.all;

entity OR_4  is
  port (A, B, C, D: in std_logic; O: out std_logic);
end entity OR_4;

architecture Struct of OR_4 is
begin
O <= A or B or C or D;
end Struct;

library ieee;
use ieee.std_logic_1164.all;

entity AND_3  is
  port (A, B, C: in std_logic; O: out std_logic);
end entity AND_3;

architecture Struct1 of AND_3 is
begin
O <= A and B and C;
```

```
end Struct1;

library ieee;
use ieee.std_logic_1164.all;

entity INV  is
  port (A: in std_logic; O: out std_logic);
end entity INV;

architecture Struct3 of INV is
begin
O <= not(A);
end Struct3;

library ieee;
use ieee.std_logic_1164.all;

entity PrimeNumber  is
  port (A, B, C, D: in std_logic; O: out std_logic);
end entity PrimeNumber;

architecture Struct2 of PrimeNumber is
signal S1, S2, S3, S4, A_b, B_b, C_b : std_logic;
component OR_4 is
port (A, B, C, D: in std_logic; O: out std_logic);
end component OR_4;
component AND_3 is
port (A, B, C: in std_logic; O: out std_logic);
end component AND_3;
component INV is
port (A: in std_logic; O: out std_logic);
end component INV;
begin
INV1: INV port map (A => A, O => A_b);
INV2: INV port map (A => B, O => B_b);
INV3: INV port map (A => C, O => C_b);
AND1: AND_3 port map (A => A_b, B => B_b, C => C, O => S1);
AND2: AND_3 port map (A => A_b, B => C, C => D, O => S2);
```

```
AND3: AND_3 port map (A => B_b, B => C, C => D, O => S3);
AND4: AND_3 port map (A => B, B => C_b, C => D, O => S4);
OR1: OR_4 port map (A => S1, B => S2, C => S3, D => S4, O => O);
end Struct2;
```

3 Observations

We get RTL simulation waveforms for corresponding to input and output which is given below and it shows required results.

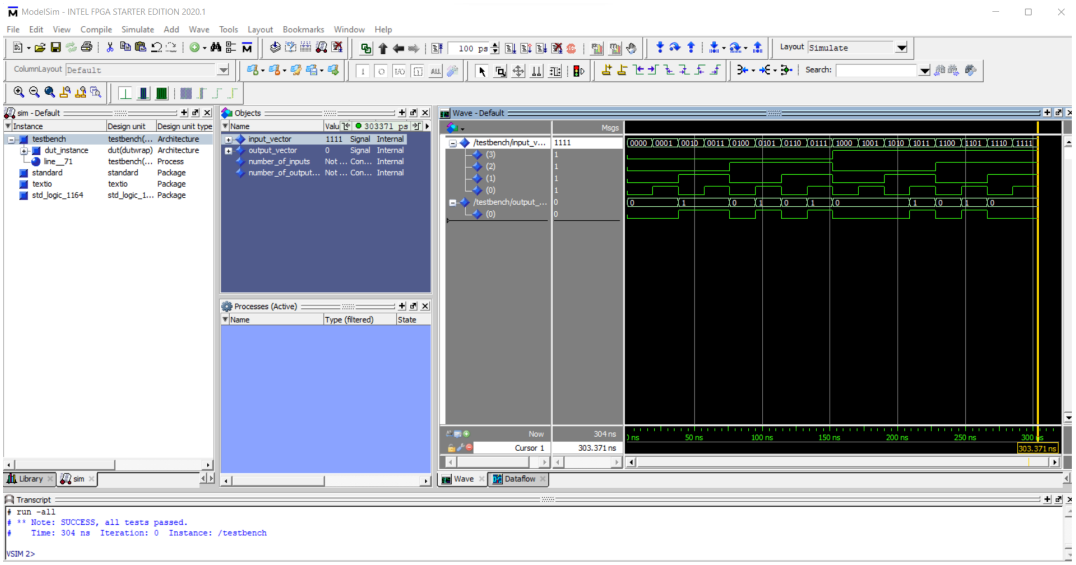

Figure 4: Prime Detector RTL Simulation Waveform

Further the code (in form of .svf file) was flashed onto the Xen10 board, after the pins and LED were mapped accordingly. The output verified the working of the logic, and some example images are shown below.

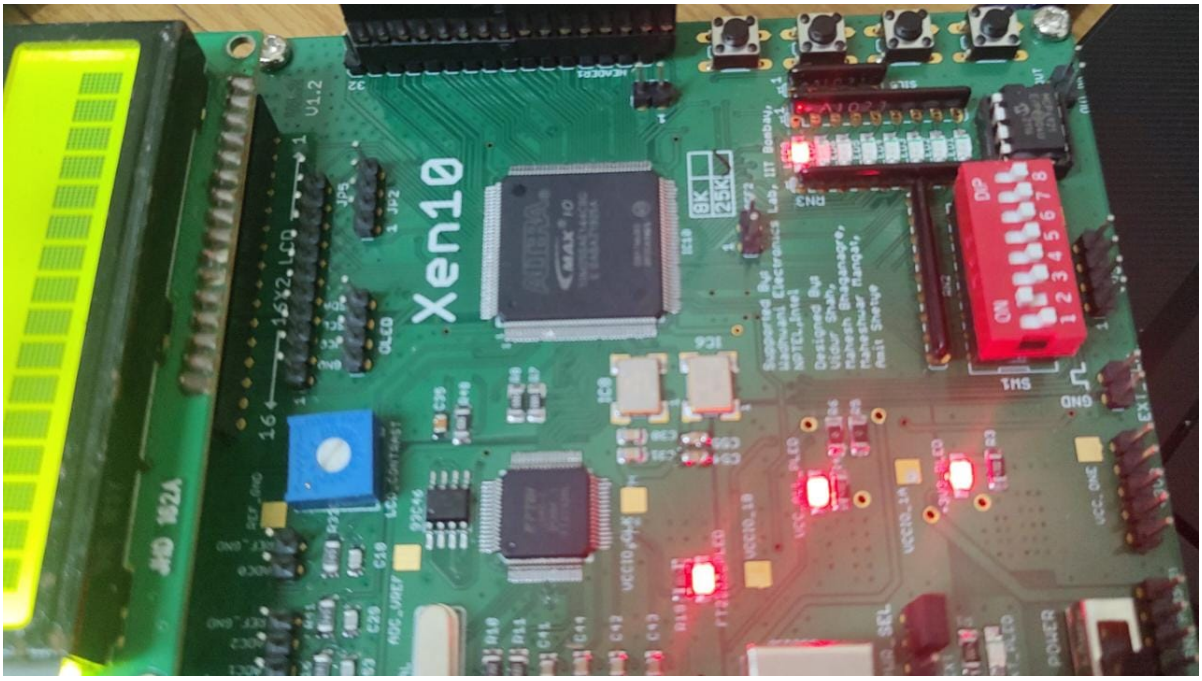

Figure 5: Prime Detector Example 1 - Xen10 Board

Figure 6: Prime Detector Example 2 - Xen10 Board

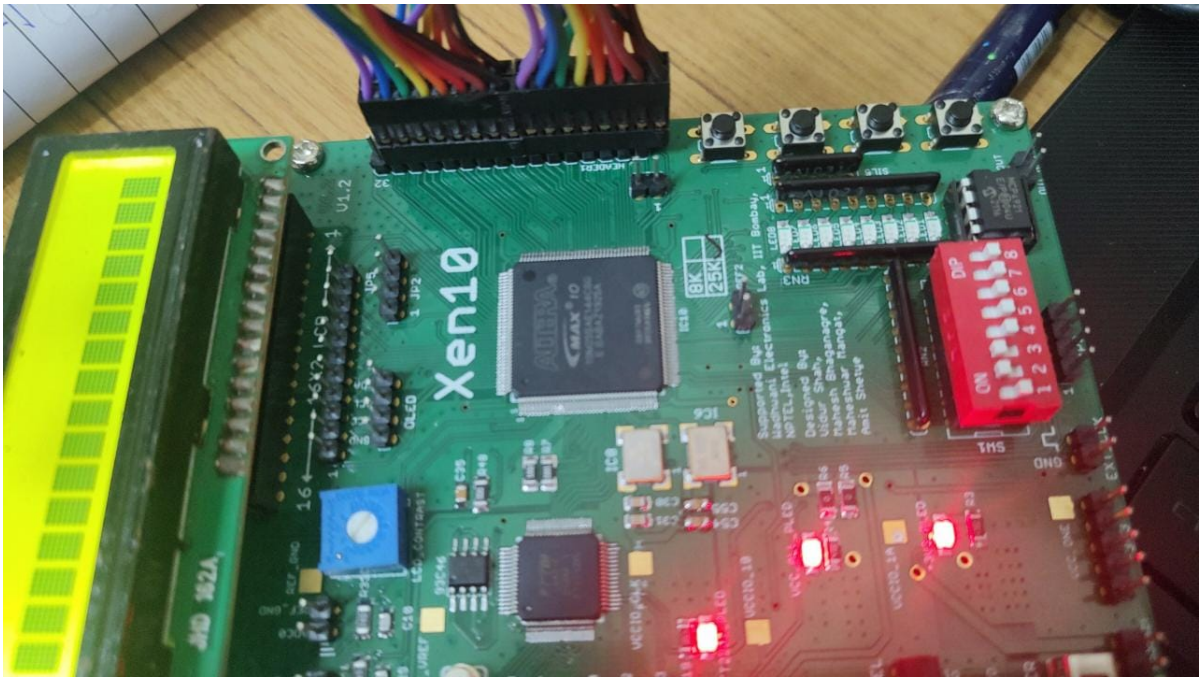

Figure 7: Prime Detector Example 3 - Xen10 Board