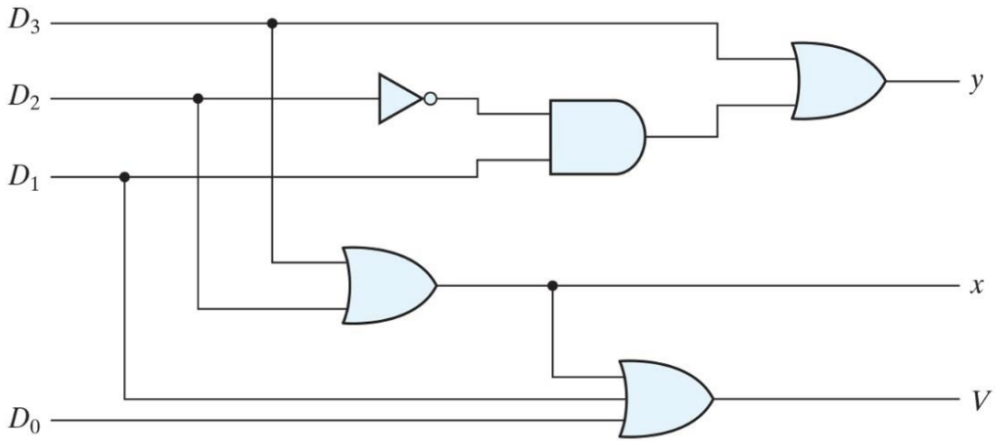
Midterm Lab: Encoder Circuit and FPGA Programming

Alexander Gaskins and Nikola Ciric

Purpose

In this lab we were tasked with using our existing code from the midterm and programming the FPGA board with it. Our exciting code is a representation of the given schematic seen below. Using our knowledge of basic digital circuits such as AND and OR gates we were given a schematic with four inputs. Then the circuit would perform various single and conjunctional operations to transform these input signals into three distinct output signals. Since we have already coded the given schematic and performed a test bench simulation on it this lab was focused on creating the constraint file which would program the FPGA board with our digital circuit logic.



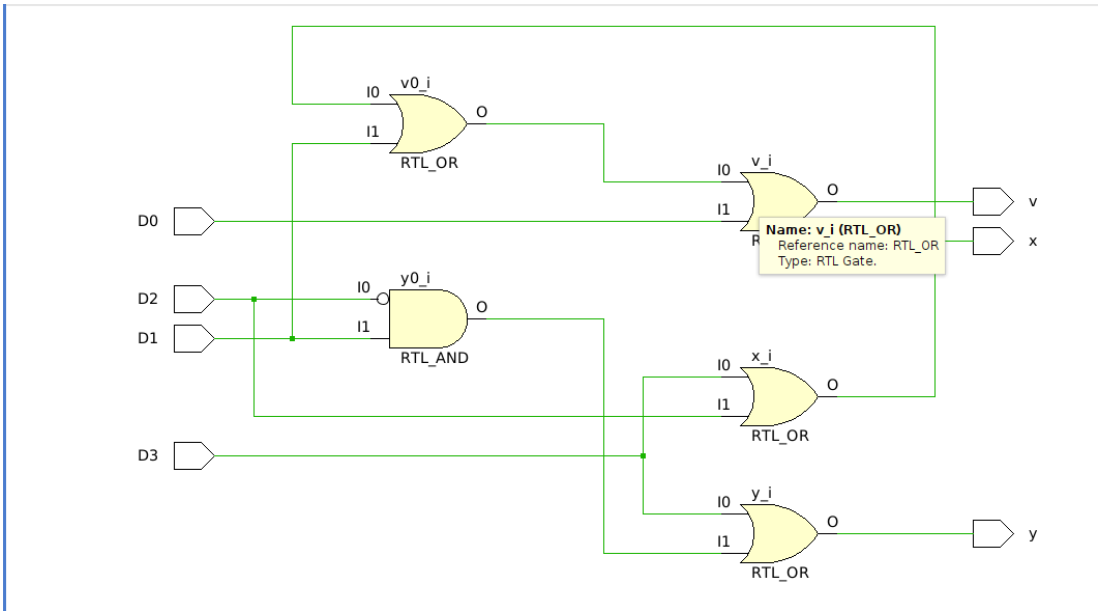
Data Collected

From the given schematic from the midterm assignment as seen in the picture above, first we had to write a description of the encoder using vivado and VHDL. From the picture it can be seen that there are four distinct input signals, being processed by five different logic gates in order to output three distinctive signals. This logic is described in the architecture section of the code snippet below. Once we have run the synthesis we can look at the schematic generated by vivado and although at first it may seem very different, if you follow the outputs to the inputs the logic is the same.

VHDL Code

```
1   library IEEE;
2   use IEEE.std_logic_1164.ALL;
3   entity Encoder is
4   port(D3,D2,D1,D0:in std_logic; x,y,v : out std_logic);
5   end Encoder;
6
7   architecture Behavioral of Encoder is
8   begin
9   y <= D3 or ((not (D2) and D1));
10  x <= D3 or D2;
11  v <= D3 or D2 or D1 or D0;
12  end Behavioral;
13
14
15
```

Schematic



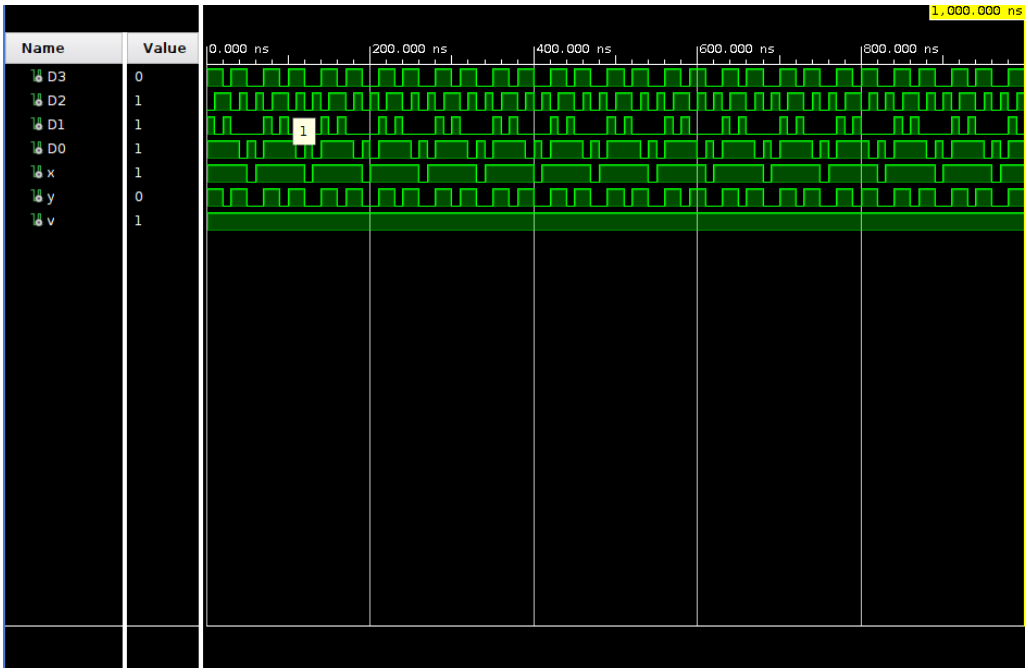
Calculations

VHDL Test Bench Code

Before we uploaded the code to the FPGA board and programmed with it, we had to make sure the encoder logic worked as intended. To do this we created a test bench file that would give some arbitrary signals to D3-D0 as a simulation. This process is seen in the test bench code snippet below. Once we have run the implementation then we are able to run the behavioral simulations which basically simulalates our digital circuit with the values we put into the test bench code. The output in a timestamp style graph is shown below.

```
library IEEE;
use IEEE.std_logic_1164.ALL;
entity tb_encoder is
end tb_encoder;
architecture Behavioral of tb_encoder is
 component Encoder
 PORT(  D3: in STD_LOGIC :='0';
        D2: in STD_LOGIC :='0';
        D1: in STD_LOGIC :='0';
        D0: in STD_LOGIC :='0';
        x:out std_logic ;
        y:out std_logic ;
        v: out std_logic
        );
 end component;
 signal D3,D2,D1,D0,x,y,v : STD_LOGIC;
 begin
 uut: Encoder
    PORT MAP(   D3 => D3,D2 => D2, D1 => D1, D0 => D0, x =>x, y=>y,  v=>v);
 process
 begin
   D3 <= '1';
   D2 <= '0';
   D1 <= '1';
   D0 <= '1';
   wait for 10ns;
   D3 <= '1';
   D2 <= '1';
   D1 <= '0';
   D0 <= '1';
   wait for 10ns;
    D3 <= '0';
   D2 <= '1';
   D1 <= '1';
   D0 <= '1';
   wait for 10ns;
    D3 <= '1';
   D2 <= '0';
   D1 <= '0';
   D0 <= '1';
   wait for 10ns;
    D3 <= '1';
   D2 <= '1';
   D1 <= '0';
   D0 <= '0';
   wait for 10ns;
    D3 <= '0';
```

Encoder Output



Results

By clicking on the link below it will bring you to a video of us testing our encoder logic on the FPGA board itself. As seen in the video, on the bottom of the board there are many switches which control the input signal. Our switches start on the bottom right and are the first four switches from bottom right to bottom left corresponding D3,D2,D1,D0 respectively from the ports from our VHDL logic. The leds that are lighting up are the output signals and since there are only three possible outputs of our encoder, only three leds are programmed to output values based on our x, y, v signals from the VHDL code. From our simulation results when only D3 has a value of 1 all of the outputs x,y, and v have outputs of 1 as well. This result matches with our FPGA leds lighting up because when the bottom right most switch is turned on representing D3 it now has a high value and in the video it can be seen that all three leds light up which means they all have a value of 1 and therefore matches up perfectly to our simulated results.
Midterm FPGA Board

Conclusion

From the results collected, we can see that the provided schematic was accurately simulated by our VHDL code, where the output results provided a closer look at the performance of the bit logic when being downloaded onto the FPGA board. The code in this lab was an extension of the previously performed Full Adder, where three inputs were provided, and two outputs were generated. In this case, the program received four inputs, and had three outputs. There were two essential components of testing the accuracy of our program: creating and comparing the generated schematic to the provided design, and testing the logic performance to ensure the process works as suggested. After setting up our code, we compared the schematic generated in Vivado to the provided circuit to compare how the signals are handled from a strictly visual perspective. The only variation that stood out between the two layouts was the difference in how the inverter was presented, being combined with the AND gate, rather than shown as its own separate component. It was determined to have the same logic operation after careful examination, but we still had yet to know whether or not the design truly did what was previously only discussed from a theoretical standpoint. Thus, we moved to testing the performance of the circuit via a separate testbench encoder. We then exported our testbench code as a bitstream file onto our FPGA board, and compared the LED signals to the switches that were manipulated. After comparing the LED signals to the encoder output in Vivado, we found a successful relation in how the input and output signals were handled, indicating a successful encoder design.