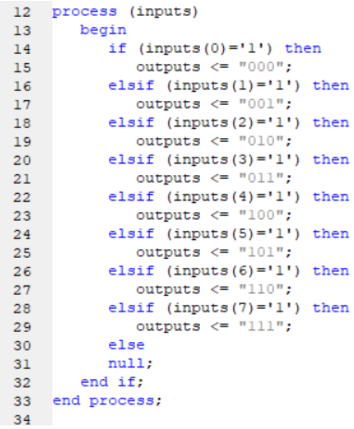
Lab Report 2

- Negin Kheirmand
- Reyhaneh Bagheri

Codes for Priority encoder(8:3)

File: PriorityEncoder.vhd



```
process (inputs)
   begin
      if (inputs(0)='1') then
         outputs <= "000";
      elsif (inputs(1)='1') then
         outputs <= "001";
      elsif (inputs(2)='1') then
         outputs <= "010";
      elsif (inputs(3)='1') then
         outputs <= "011";
      elsif (inputs(4)='1') then
         outputs <= "100";
      elsif (inputs(5)='1') then
         outputs <= "101";
      elsif (inputs(6)='1') then
         outputs <= "110";
      elsif (inputs(7)='1') then
         outputs <= "111";
      else
      null;
   end if;
end process;
```

As you can clearly see, the least the index is, the more priority has in the output.

For example If in the input we have:

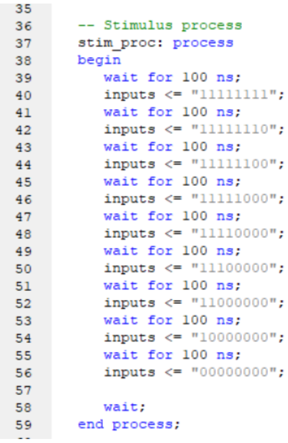
Inputs[0] ='1'

In other words if the "inputs" singal is رلاشیرالشریلس the output would be 000, showing that the encoder does not care about anything other than the house number 0 of our vector and until its value doesn't change from 1 to 0 the output wont change either.

Codes for Pririty encoder(8:3) Test Bench

File: PriorityEncoder_tb.vhd

The stimulus section of the testbench:

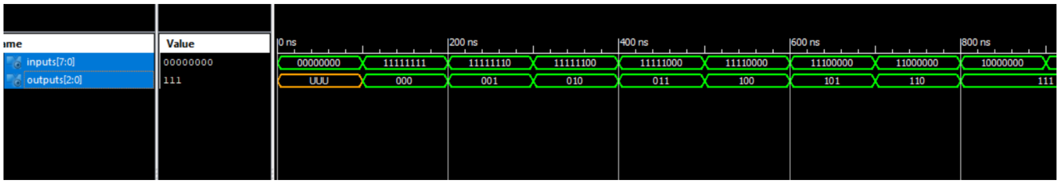


```

   -- Stimulus process
   stim_proc: process
   begin
      wait for 100 ns;
      inputs <= "11111111";
      wait for 100 ns;
      inputs <= "11111110";
      wait for 100 ns;
      inputs <= "11111100";
      wait for 100 ns;
      inputs <= "11111000";
      wait for 100 ns;
      inputs <= "11110000";
      wait for 100 ns;
      inputs <= "11100000";
      wait for 100 ns;
      inputs <= "11000000";
      wait for 100 ns;
      inputs <= "10000000";
      wait for 100 ns;
      inputs <= "00000000";

      wait;
   end process;
```

And the wave form is:



Explanation(in detail):

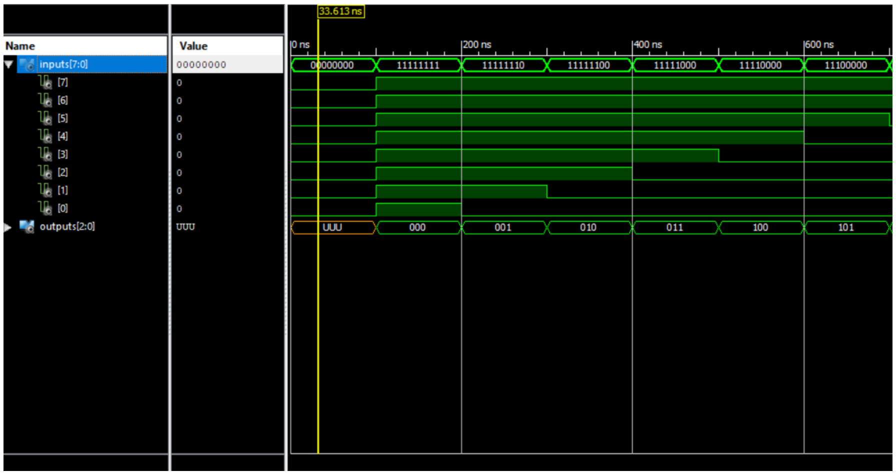
Case 1:

Input: 00000000

Output: UUU

Enters the else part of out if else… cause in this case the input has no 1s so the priority is with none of the pins.

The output will be null (UUU)

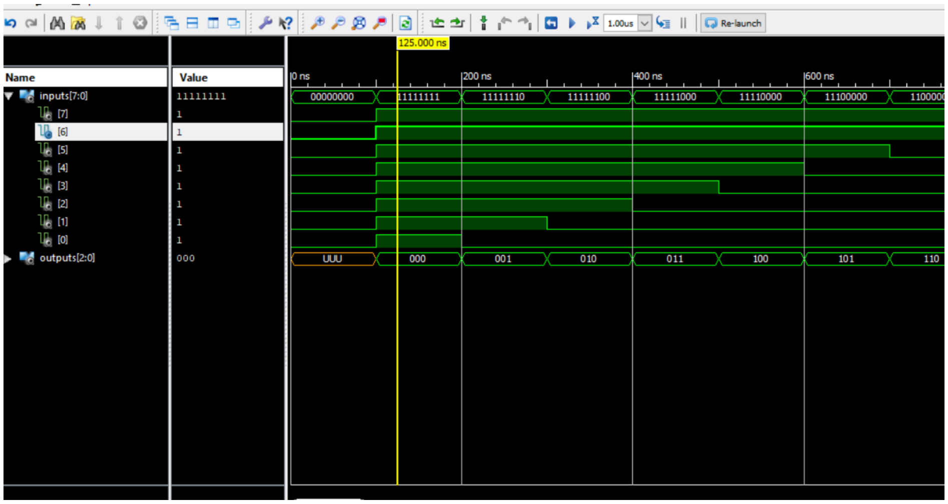


Case 2:

Input: 11111111

Output: 000

Since the inputs[0] is 1 the output will be 0 (= 3’b000)

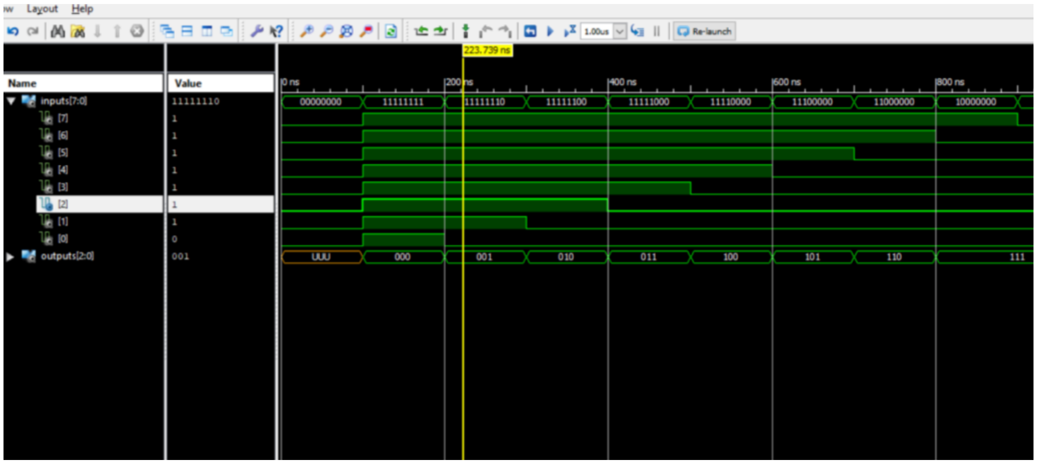


Case 3:

Input: 11111110

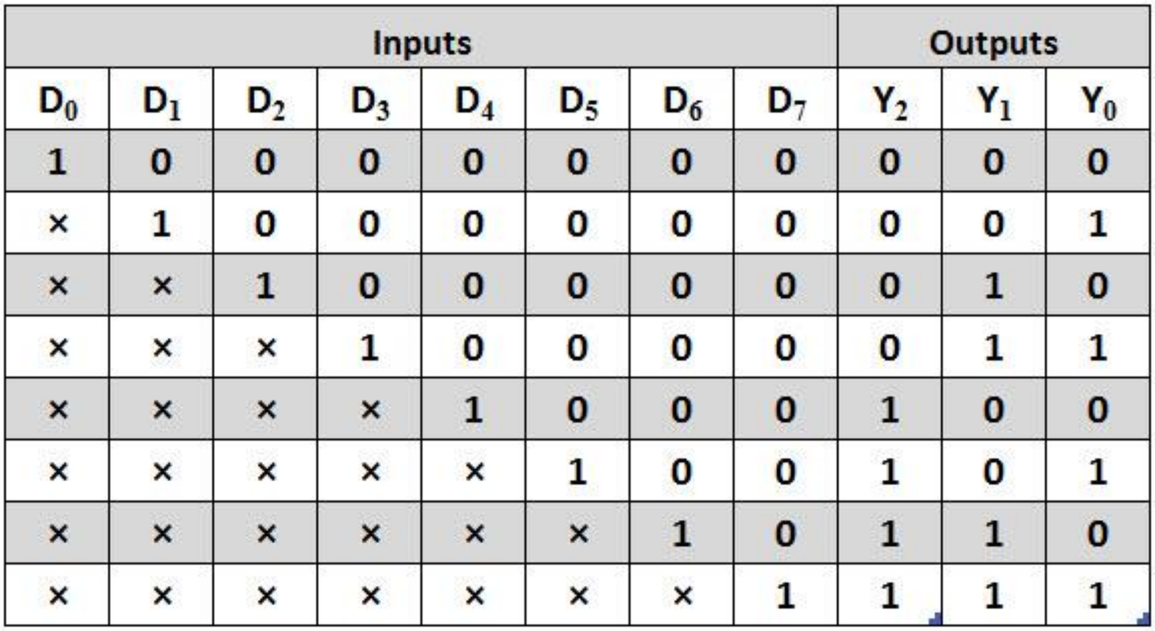
Output: 001

Since the inputs[0] is 0 the program will check the next index with the biggest priority after 0 which is 1: and since inputs[1] 1 then the output will be 1 (= 3’b001)



The rest of the cases follow the same logic.

Priority encoder’s truth table (8->3) :



Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	Y_2	Y_1	Y_0
1	0	0	0	0	0	0	0	0	0	0
×	1	0	0	0	0	0	0	0	0	1
×	×	1	0	0	0	0	0	0	1	0
×	×	×	1	0	0	0	0	0	1	1
×	×	×	×	1	0	0	0	1	0	0
×	×	×	×	×	1	0	0	1	0	1
×	×	×	×	×	×	1	0	1	1	0
×	×	×	×	×	×	×	1	1	1	1

The only difference is in this truth table doesn’t specify what the output will be if the input is all composed by 0s (case : 00000000) and in that case I have chosen the output null (UUU)