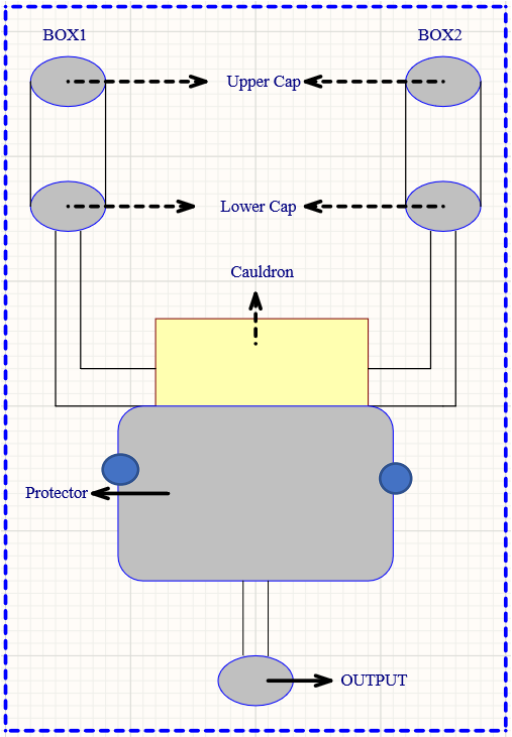
Recycling System Control

In a recycling system, there are containers with 2 lids, upper and lower, used to collect 2 materials, a large boiler providing heat treatment and a protective thin lower boiler. Each container has a sensor that measures its fullness. If the container is full, the sensors will give a value of 1 and open the lower covers and close the upper covers (In this case, the upper covers are opened first and the lower covers closed). Both containers cannot be emptied and opened at the same time until they are full. If both are filled at the same time, the selection will be made with the logic of 2x1 multiplier.



There is a boiler where the containers empty the recycling material. While there is material in the boiler, it starts to heat up and heats the material inside. There is another protective large thin cauldron under the cauldron. This protector has 2 sensors and therefore an output. After the heat treatment, if the temperature in the big boiler is between 7 and 10 (rated as Celsius and written as dual temperature), the LED in our control system lights up and the cold air flap of the protective boiler opens to equalize the temperature inside. and the material inside and outside the boiler cannot be discharged. During the heat treatment, if the temperature is above 10, both the led lights up and the buzzer works, and the cold water cover, which is the other valve of the protective boiler, opens and the substance in the boiler cannot be discharged again. If the temperature is below 7, the protective boiler sensor values do not change and the heated substance is discharged to the relevant place. This control is provided by the FPGA card.

```
module recycling(clk,s,box1,box2,box1_sensor,box2_sensor,protector,temperature,led,buzzer,out);
    input clk; // clock cycle for control the sistem with time
    input s; //control input for multiplexer which is for choicing boxes.
    input box1_sensor, box2_sensor; //  The sensor is used to understand whether the container is full, if the sensor data is 1, the lower cover is opened,
    //if it is equal to 0, the upper cover is opened.
    output reg [1:0] box1,box2;// every outside box(container) has 2 covers as under and upper.
    input [3:0] temperature;   //temperature sensor
    reg  cauldron=0; // where the heat treatment is carried out, not output or input.
    output reg[1:0] protector; // cold water bottom protector used to cool the boiler.
    output reg out;     // output cap(kapak) to send the heated substance to the relevant place.
    output reg led;     // controling the temperature.
    output reg buzzer;  // another controling the high temperature.
    initial begin
    box1=2'b01; // upper cap is open
    box2=2'b01; //upper cap is open
    protector=2'b00;
    out=0;
    led=0;
    buzzer=0;
    end
```

Since this system continues continuously, we keep our previous data and perform non-blocking transactions. All operations take place when the clock signal we provide to the system changes. Each sensor is designated as input. The boiler that we will perform the heat treatment process has no function in the system as neither input nor output, and it is like an intermediate element like wire, but it is defined as a register since the next value will change according to its value.

HAVVA
YILMAZ

150719031

```
always @(posedge clk)
begin
if(((box1_sensor==1))&&((box2_sensor==1)))
begin
    box1[0]=~box1[0];
    box2[0]=~box2[0];

    if(s==0)
    begin
        box1[1]<=1; // bottom cap open
        box2[1]<=0; // bottom cap
        cauldron<=1;
        // withmin 10 minutes
        if((temperature >= 7) & (temperature <= 10))    //temperature >= 200 & temperature <= 400
        begin
            led <=1;
            protector[0] <=1; // air condition is satisfying
            protector[1] <=0;
            out <=0;
        end
        else if(temperature >10)
        begin
            led <=1;
            buzzer <=1;
            protector[0] <= 1;
            protector[1] <=1;       // woter condition is satisfying
            out <=0;
        end
        else if(temperature <7)
        begin
            led=0;
            buzzer=0;
            out=1;      // cycle is completed.
            cauldron <=0;
        end
        box1[1]=~box1[1];
        box1[0]=~box1[0];
    end
```

```
if(s==1)
begin
    box1[1]<=0;  //box1 is close.
    box2[1]<=1; //box2 is open.
    cauldron<=1;
    // withmin 10 minutes
    if((temperature >= 7) & (temperature <= 10))
    begin
        led <=1;
        protector[0] <= 1;  //air condition is satisfying.
        protector[1] <= 0;
        out <=0;
    end
    else if(temperature > 10)
    begin
        led <=1;
        buzzer <=1;
        protector[0] <= 1;      //woter condition is satisfying
        protector[1] <=1;
        out <=0;
    end
    else if(temperature < 7)
    begin
        led=0;
        buzzer=0;
        out=1;
        cauldron <=0;
    end
    box2[1]=~box2[1];
    box2[0]=~box2[0];
end
end
```

```
else if((box1_sensor==1) && (box2_sensor==0))
begin
      box1[1]<=1; // bottom cap open
      box1[0]=~box1[0]; //upper cap close
      box2[1]<=0; // bottom cap
      cauldron<=1;
      // withmin 10 minutes
      if((temperature >= 7) & (temperature <= 10))
      begin
          led <=1;
          protector[0] <=1; // air condition is satisfying
          protector[1] <=0;
          out <=0;
      end
      else if(temperature > 10)
      begin
          led <=1;
          buzzer <=1;
          protector[0] <= 1;
          protector[1] <=1;       // woter condition is satisfying
          out <=0;
      end
      else if(temperature < 7)
      begin
          led=0;
          buzzer=0;
          out=1;      // cycle is completed.
          cauldron <=0;
      end
      box1[1]=~box1[1];
      box1[0]=~box1[0];
end
```

```
else if((box2_sensor==1) && (box1_sensor==0))
begin
      box1[1]<=0;  //box1 is close.
      box2[1]<=1; //box2 is open.
      box2[0]=~box2[0];
      cauldron<=1;
      // withmin 10 minutes
      if((temperature >= 7) & (temperature <= 10))
      begin
          led <=1;
          protector[0] <= 1;  //air condition is satisfying.
          protector[1] <= 0;
          out <=0;
      end
      else if(temperature > 10)
      begin
          led <=1;
          buzzer <=1;
          protector[0] <= 1;      //woter condition is satisfying
          protector[1] <=1;
          out <=0;
      end
      else if(temperature < 7)
      begin
          led=0;
          buzzer=0;
          out=1;
          cauldron <=0;
      end
      box2[1]=~box2[1];
      box2[0]=~box2[0];
end
```

```
  else
  if((box2_sensor==0) && (box1_sensor==0))
  begin
        led<=0;
        buzzer<=0;
        out<=0;
        cauldron <=0;
        protector<=2'b00;
        box1=2'b01;
        box2=2'b01;
  end
end
```

Since we have 2 containers, there are 4 possibilities for the containers to be full, 11, 10, 01.00. First of all, when both containers are full, we make a selection thanks to the 2X1 mux and determine the output according to the relevant conditions. We then consider the case where only one of the containers is full. Containers (box) are 2-bit vectors as box1 and box2. The first bit of the vectors is designated as the front cap and the second bit is designated as the bottom cap. Since the system does not work at first, the conditions are determined in such a way that the top cover of the containers is open and the system is stationary at first.

```
`timescale 1ns / 1ps

module recycling_tb;
    reg clk,s,box1_sensor,box2_sensor;
    reg [3:0] temperature;
    wire [1:0] box1, box2;
    wire [1:0] protector;
    wire led,buzzer,out;

    recycling UUT(clk,s,box1,box2,box1_sensor,box2_sensor,protector,temperature,led,buzzer,out);

    initial begin
    clk=0;
    #10
    box1_sensor=1;
    box2_sensor=0;
    s=1;
   // #600000000000; // after boxes are open wait 10 minutes and then control the heat
   //how much it is but it is too long we cant use it is in testbench
   #100 temperature=4'b1000;

    #10
    box1_sensor=0;
    box2_sensor=1;
    s=0;
    #100 temperature=4'b1011;

    #10
    box1_sensor=1;
    box2_sensor=1;
    s=1;
    #100 temperature=4'b0011;

    #10
    box1_sensor=0;
    box2_sensor=0;
    s=1;
    #100 temperature=4'b1001;
    end

    always
    #10 clk=~clk;
endmodule
```

Thanks to the testbench file, values are given to the sensors of the containers, the selection condition s of the multiplexer and the temperature sensor. Values were obtained by applying all conditions one after the other in a random manner according to the occupancy status of the containers.

Under normal conditions, it takes about 10 minutes for the boiler to heat (liquidize) the substances in it, but since we cannot test it for such a long time in the testbench, this time is set as 100 ns. At the same time, every 10 ns, another state is passed. The time period of the conditions is 110 ns. In addition, the temperature values were tested by dividing them into ranges to be representative as they are normally very powerful sensors and large values.

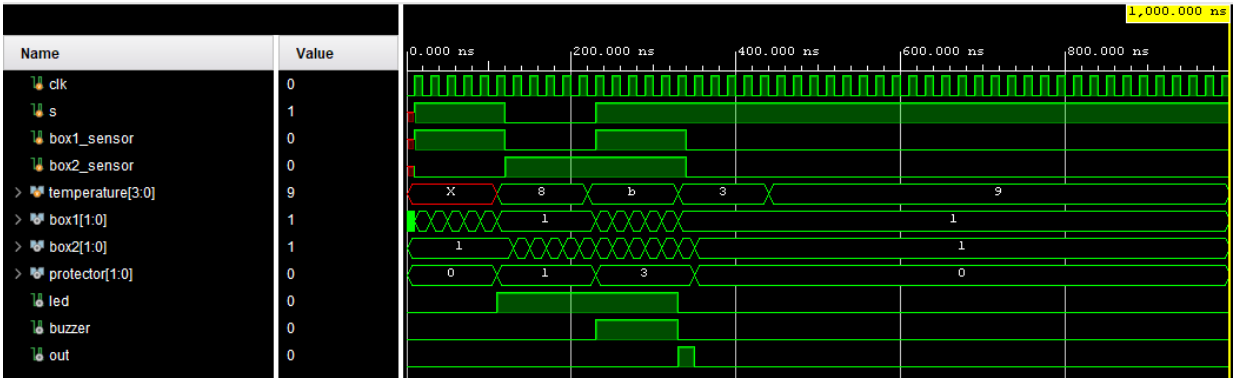


Figure 1 : The results when we run Run Synthesis and Simulation for the above testbench.

```
#10
box1_sensor=0;
box2_sensor=1;
s=0;
#100 temperature=4'b1011;
```

In order to show more specific results, when we tested the code by considering the following condition, the results were as follows. According to the condition, the 1st box should be closed, the 2nd box should be open, when the temperature is 11, both the led and the buzzer should be active and the output should be zero as expected.

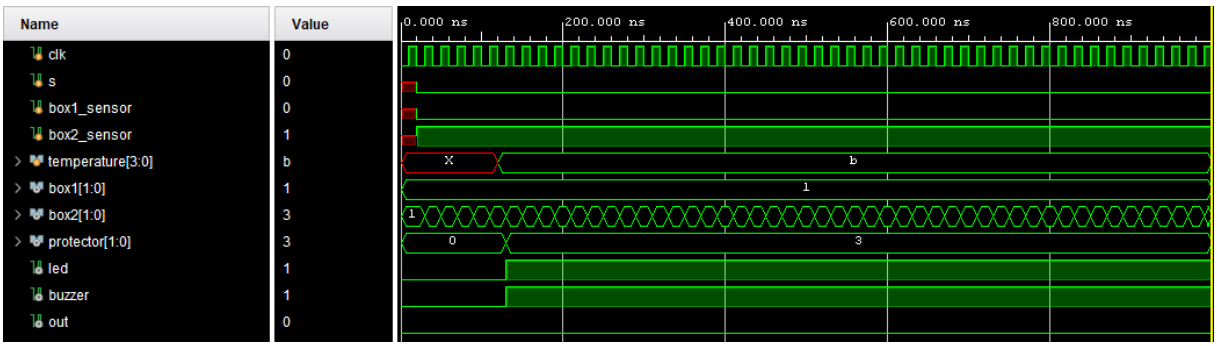


Figure 2 : The Result of Minimize Testbench File.