Project Overview: - We've demonstrated the working of a Home Alarm System using Basys 3 FPGA board. The system works on the status of 5 switches of which 4 switches represent sensors and 1 switch acts as master switch. The system is functional if the master switch is 'on' else the system is non-functional. We have 3 LEDs and an Alarm to show the working.

System Design: -

Top Module: - It integrates all submodules and coordinates the overall system operation. It accepts switch inputs, processes them, and outputs both the alarm signal and display patterns.

Logic Type	Variable Name	Function
input	sw [4:0]	Five switches representing four sensor inputs and one master switch
output	Alarm	Activates if the system detects a security breach
output	seg	Controls the segments of the 7-segment display
output	an	Controls the anodes of the 7-segment display
output	led1, led2, led3	Display status of the system

Home Alarm System Module: -This module handles the logic for alarm activation.

- The alarm (a) is triggered if any sensor is active (sw[3:0]) and the master switch (sw[4]/m) is on.
- Logic: a = (sw[0] | sw[1] | sw[2] | sw[3]) & m
  led1=a;
  led2=m;
  led3=~m;
- led1(red) activates when master switch and one of the sensors are active. led2(yellow) activates when master switch is active but no sensors are active. led3(green) activates when master switch and sensors are inactive.

Seven Segment Display Module: - This module converts a 4-bit binary input into a pattern displayed on the 7-segment display.

- Displays digits (0-9) and hexadecimal characters (A-F) based on the input.
- Fixed anode control (an = 4'b1110) to light up the first display.

Testbench Module: - A testbench was created to verify the system's functionality.

- **Test Case 1: sw = 5'b00000**(No switches are pressed; the system is inactive)
  - ✓ All sensor switches (sw[3:0]) are 0, indicating no sensors are triggered.
  - ✓ The master switch (sw[4]) is 0, so the system is not enabled.
  - ✓ Alarm = 0 (off) because the master switch is off, and no sensors are active.
  - ✓ led3 is active.

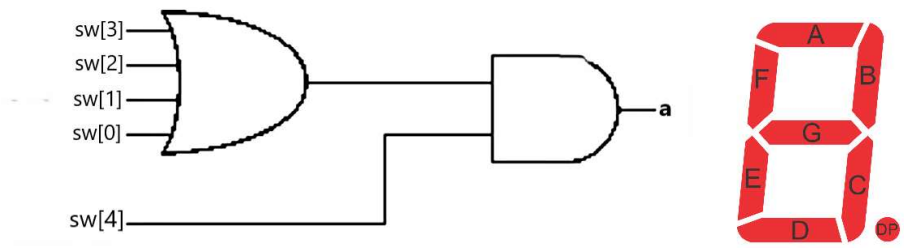
- **Test Case 2: sw = 5'b00001**(Sensor 0 is activated, but the system is still inactive)
  - ✓ Sensor 0 (sw[0]) is 1, indicating it is triggered.
  - ✓ The master switch (sw[4]) is 0, so the system is not enabled.
  - ✓ Alarm = 0 (off) because the master switch is off, even though a sensor is triggered.
  - ✓ led3 is active

- **Test Case 3: sw = 5'b10000**(The master switch is on, but no sensors are active)
  - ✓ All sensor switches (sw[3:0]) are 0, indicating no sensors are triggered.
  - ✓ The master switch (sw[4]) is 1, enabling the system.
  - ✓ Alarm = 0 (off) because no sensors are active, despite the system being enabled.
  - ✓ led2 is active.

- **Test Case 4: sw = 5'b10001**(The master switch is on, and Sensor 0 is activated)
  - ✓ Sensor 0 (sw[0]) is 1, indicating it is triggered.
  - ✓ The master switch (sw[4]) is 1, enabling the system.
  - ✓ Alarm = 1 (on) because the system is enabled, and a sensor is triggered.
  - ✓ led2 and led3 are active.

- **Test Case 5: sw = 5'b11010** (The master switch is on, and a random combination of sensors is activated)
  - ✓ Sensors 3 and 1 (sw[3] and sw[1]) are 1, indicating multiple sensors are triggered.
  - ✓ The master switch (sw[4]) is 1, enabling the system.
  - ✓ Alarm = 1 (on) because the system is enabled, and at least one sensor is triggered.
  - ✓ led2 and led3 are active.

Logic Diagram and Seven Segment Display: -



Code Snippets: -

Design Codes: -

```
module Top_Module(input logic [4:0] sw,        // Switch inputs
                  output logic Alarm,          // Alarm signal
                  output logic [3:0] an,       // 7-segment anode control
                  output logic [6:0] seg,      // 7-segment display segments
                  output logic led1,led2,led3); //Indicator LEDs
logic [3:0] act;            // Internal signal for 7-segment input
assign act = {sw[4], 1'b0, sw[4], 1'b0}; // Generate a pattern based on `sw`

// Instantiate submodules
Seven_Segment S1(.in(act),.seg(seg),.an(an));
Home_Alarm_System S2(.sw(sw[3:0]),.m(sw[4]),.a(Alarm),.led1(led1),.led2(led2),.led3(led3));

endmodule
```

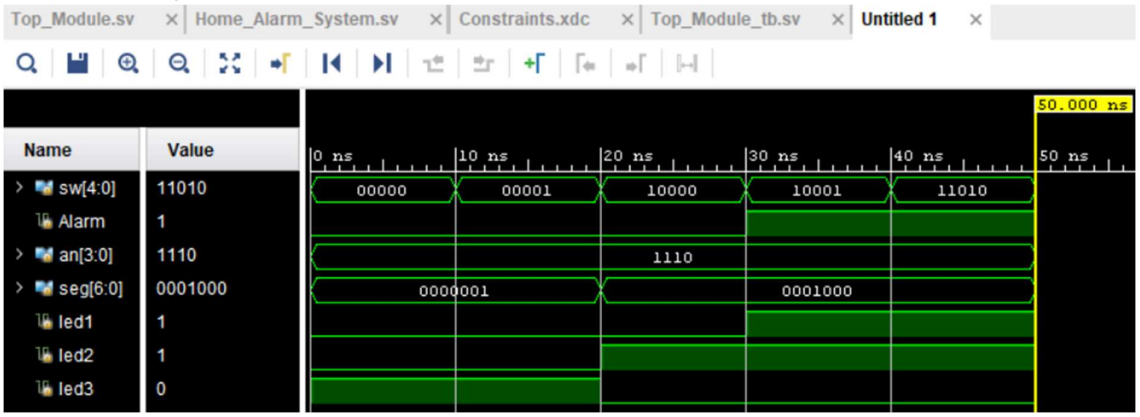
```
module Home_Alarm_System(input logic [3:0] sw,    // 4-bit switch inputs
                         input logic m,           // Master control switch
                         output logic a,          // Alarm signal
                         output logic led1,led2,led3); //Indicator LEDs
assign a = (sw[0] | sw[1] | sw[2] | sw[3]) & m; // Alarm logic
assign led1=a;
assign led2=m;
assign led3=~m;
endmodule
```

```
module Seven_Segment(input logic [3:0] in,      // 4-bit input
                     output logic [6:0] seg,    // 7-segment display output
                     output logic [3:0] an);     // Anode control
always @(in)
begin
    case (in)
      4'b0000: seg = 7'b0000001; // 0
      4'b0001: seg = 7'b1001111; // 1
      4'b0010: seg = 7'b0010010; // 2
      4'b0011: seg = 7'b0000110; // 3
      4'b0100: seg = 7'b1001100; // 4
      4'b0101: seg = 7'b0100100; // 5
      4'b0110: seg = 7'b0100000; // 6
      4'b0111: seg = 7'b0001111; // 7
      4'b1000: seg = 7'b0000000; // 8
      4'b1001: seg = 7'b0000100; // 9
      4'b1010: seg = 7'b0000010; // A
      4'b1011: seg = 7'b1100000; // b
      4'b1100: seg = 7'b0110001; // C
      4'b1101: seg = 7'b1000010; // d
      4'b1110: seg = 7'b0110000; // E
      4'b1111: seg = 7'b0111000; // F
    endcase
end
assign an = 4'b1110; // Fixed default anode activation
endmodule
```

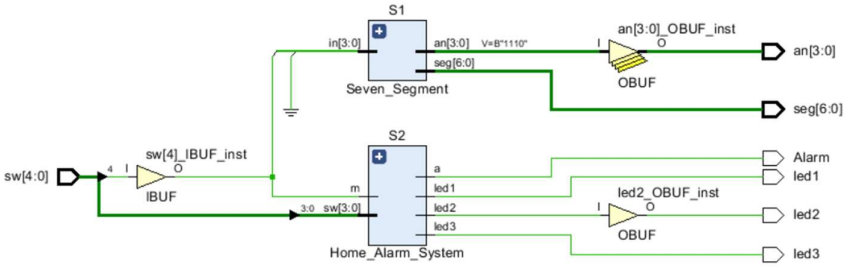
Testbench Code: -

```
module Top_Module_tb();
logic [4:0] sw;        // Switch inputs
logic Alarm;           // Alarm signal
logic [3:0] an;        // 7-segment anode control
logic [6:0] seg;       //7-segment display control
logic led1,led2,led3;//Indicator LEDs

// Instantiate the top_module
Top_Module dut (.sw(sw),.an(an),.seg(seg),.Alarm(Alarm),.led1(led1),.led2(led2),.led3(led3));

initial begin
sw = 5'b00000; #10; //Master Switch and Sensors are inactive
sw = 5'b00001; #10; // Switch 0 pressed but master switch is inactive
sw = 5'b10000; #10; // Master switch on, no sensor active (No alarm)
sw = 5'b10001; #10; // Master switch on, switch 0 pressed (Alarm is active)
sw = 5'b11010; #10; // Master switch on, switches 1 and 3 pressed (Alarm is active)
$stop;
end
endmodule
```

Simulation Graph: -



RTL Schematic: -

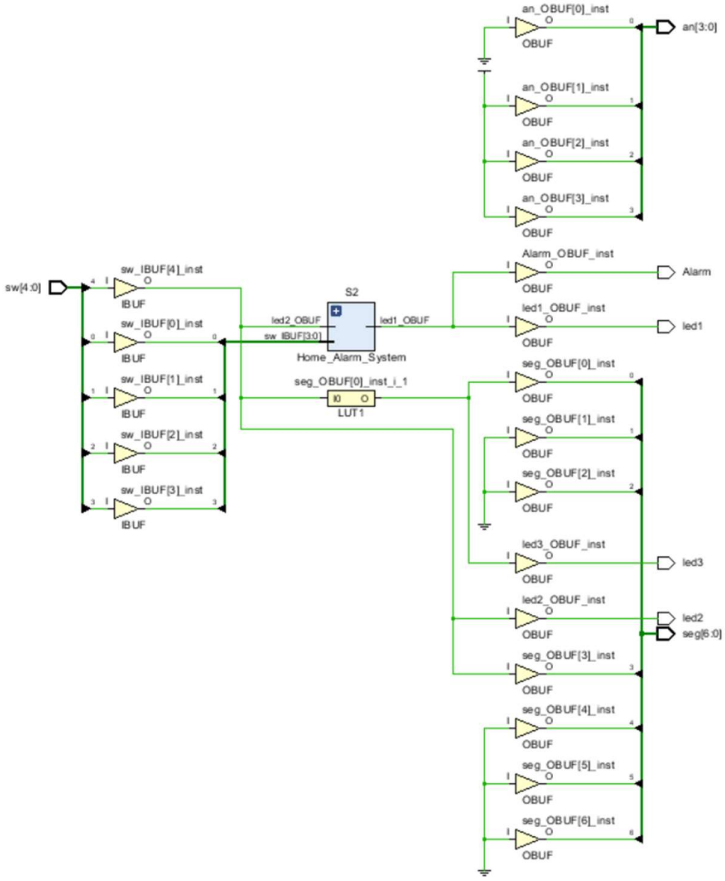


Resource Utilisation: -

Summary

Resource	Utilization	Available	Utilization %
LUT	2	20800	0.01
IO	20	106	18.87

Technology Schematic: -



Constraints: -

C:/Users/Siddharth Bhat/Vivaldo_Projects/Project_HomeAlarm/Project_HomeAlarm.srcs/constrs_1/new/Constraints.xdc

```
set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports Alarm]
set_property PACKAGE_PIN J1 [get_ports Alarm]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
set_property PACKAGE_PIN W7 [get_ports {seg[6]}]
set_property PACKAGE_PIN W6 [get_ports {seg[5]}]
set_property PACKAGE_PIN U8 [get_ports {seg[4]}]
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg[2]}]
set_property PACKAGE_PIN V5 [get_ports {seg[1]}]
set_property PACKAGE_PIN U7 [get_ports {seg[0]}]
set_property PACKAGE_PIN W4 [get_ports {an[3]}]
set_property PACKAGE_PIN V4 [get_ports {an[2]}]
set_property PACKAGE_PIN U4 [get_ports {an[1]}]
set_property PACKAGE_PIN U2 [get_ports {an[0]}]

set_property PACKAGE_PIN V17 [get_ports {sw[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports led1]
set_property IOSTANDARD LVCMOS33 [get_ports led2]
set_property IOSTANDARD LVCMOS33 [get_ports led3]
set_property PACKAGE_PIN H1 [get_ports led1]
set_property PACKAGE_PIN K2 [get_ports led2]
set_property PACKAGE_PIN H2 [get_ports led3]
```