



INTELLIGENT INDUSTRIAL CONTROLLING USING SENSORS

PRSESENTED BY:-

MOHAMMAD MOBEEN

TABLE OF CONTENTS:-

Abstract
Introduction
Literature Survey
Problem Statement
Objectives
Block Diagram
Hardware Components
o IR Sensor
o Gas Sensor
o Fire Sensor
o Relay Module
o EDGE Artix-7 FPGA Board
Software Tools
System Architecture
Verilog HDL Code Explanation
o Sensor Interface Modules
o Relay Control Module
o Top Module Integration
Testbench and Simulation
Results and Output Snapshots
Future Scope

ABSTRACT

This project is about building an automatic monitoring system for industries using sensors and an FPGA board (EDGE Artix-7A). We are using three sensors: an IR sensor to detect motion, a fire sensor to detect fire, and a gas sensor to detect gas leaks.

When motion is detected using the IR sensor, a counter is increased and shown on an 7 segment screen. If fire is detected, a message is shown on a separate 7 segment and a buzzer is turned on. For gas detection, if gas is present, only the buzzer turns on—there is no 7 segment message for gas.

And a motor is connected with relay module and thus we can stop it when a fire is detected on fire sensor.

All the sensors are connected to the FPGA, which controls everything using Verilog code. This system helps to make industrial areas safer by detecting problems early and giving alerts automatically.

PROJECT OBJECTIVE

The main aim of this project is to build a smart monitoring system that can improve safety in industries. We want our system to detect gas leaks, fire, and motion using sensors and give alerts using LEDs and a buzzer. It should work in real-time and respond quickly to any danger. We are using an Artix-7 FPGA to control everything and writing the logic in Verilog. We also plan to test and simulate the design using Vivado software to make sure it works well before we connect it to real sensors. This system should help reduce risks in industrial environments and give fast warnings when something goes wrong.

Chapter 1: Introduction

1.1 Background

In modern industrial environments, safety, efficiency, and automation are critical components of productivity and sustainability. Rapid technological advancements have allowed industries to automate various operations, leading to reduced human intervention, improved consistency, and timely decision-making. One of the core challenges in such environments is real-time monitoring of hazardous events such as gas leaks and fire outbreaks. These hazards can cause significant loss of life, environmental damage, and financial loss if not detected and addressed promptly.

The integration of sensor technologies with advanced processing platforms like Field Programmable Gate Arrays (FPGAs) has opened new possibilities in building intelligent, real-time monitoring systems. FPGAs offer hardware-level parallelism, which enhances the speed and reliability of industrial systems compared to traditional microcontrollers.

This project focuses on developing an industry monitoring and controlling system that leverages IR, gas, and fire sensors to detect hazards and control industrial motors using the EDGE Artix-7 FPGA board.

1.2 Problem Statement

Many existing industrial systems rely on human supervision or slow-responding microcontroller-based solutions for hazard detection. These systems often suffer from latency issues, limited scalability, and the inability to perform multiple operations simultaneously. In critical situations such as gas leaks or fire, every millisecond matters.

The lack of real-time, reliable monitoring and responsive automation in hazardous environments presents a significant challenge that this project aims to address using FPGA-based control systems.

1.3 Objectives

To design and implement a real-time monitoring system using IR, gas, and fire sensors.

To integrate a motor control mechanism that reacts to hazard detection.

To develop Verilog-based logic for sensor data processing and output control.

To use the EDGE Artix-7 FPGA board for executing control operations with minimal latency.

To ensure the system performs efficiently in detecting hazards and managing industrial operations.

1.4 Scope of the Project

This project focuses on designing a hardware-based solution for monitoring and controlling an industrial environment. The system includes:

Hazard detection (gas, fire).

Object detection and counting.

Automatic motor shutdown or operation control

1.5 Industrial Automation

Industrial automation involves using control systems such as computers, robots, and information technologies to manage various industrial processes and machinery with minimal human intervention. It enhances productivity, improves safety, reduces human errors, and increases production consistency. Early automation systems were primarily relay-based and later evolved to Programmable Logic Controllers (PLCs). Today, automation extends to include real-time embedded systems and FPGAs, offering faster and more reliable control solutions.

The emergence of Industry 4.0 has accelerated the adoption of intelligent automation systems, enabling smart factories where machines can communicate and make decisions based on sensor inputs. This project fits within this paradigm by using FPGAs to handle critical monitoring and control tasks in real-time.

1.6 Sensor Technologies

Sensors play a vital role in industrial monitoring by providing real-time data about environmental and operational conditions. IR sensors are commonly used for object detection and counting. They work by emitting infrared light and detecting the reflected signal from nearby objects.

Gas sensors like the MQ-2 are used to detect combustible gases such as methane, propane, and butane. These sensors change their resistance based on the gas concentration in the air. Fire sensors detect the presence of flame or heat and are essential in early warning systems to prevent fire-related disasters.

Previous studies and industrial deployments have shown that the integration of multiple sensors can significantly enhance the safety and efficiency of manufacturing and processing units.

1.7 FPGA in Industrial Applications

Field Programmable Gate Arrays (FPGAs) are integrated circuits that can be programmed after manufacturing to perform specific logic functions. Unlike microcontrollers, which execute instructions sequentially, FPGAs use parallel processing, making them suitable for time-critical applications.

FPGAs are widely used in industrial automation due to their ability to handle multiple inputs/outputs simultaneously, high reliability, and reconfigurability. They are especially useful in systems where real-time processing, deterministic performance, and customizability are essential. In this project, the EDGE Artix-7 FPGA is utilized to monitor sensors and control a motor in response to hazardous conditions.

1.8 Motor Control Systems

Motors are central to various industrial applications, from conveyors to robotic arms. Controlling motors based on environmental or operational feedback is critical for maintaining efficiency and safety. The 2-relay motor module is a widely used dual H-Bridge driver that can control the direction and speed of DC motors.

In traditional setups, motor control is handled by microcontrollers or dedicated motor drivers. However, integrating motor control with FPGA logic allows for faster response and the ability to simultaneously process data from multiple sensors. The use of Verilog enables developers to create custom control logic tailored to specific needs.

Chapter 2: System Design and Architecture

2.1 System Overview

The Industry Monitoring and Controlling System is designed to detect hazardous conditions such as the presence of gas, fire, or an object on a conveyor and respond accordingly using a motor and buzzer. The core of the system is the EDGE Artix-7 FPGA, which processes sensor inputs and executes control logic in real-time. The goal is to ensure immediate response to critical conditions, thus enhancing safety and operational efficiency in an industrial environment.

2.2 Block Diagram

The system block diagram includes the following components:

- IR Sensor connected to the FPGA for object detection and counting.

- Gas Sensor (MQ-2) for detecting the presence of combustible gases.

- Fire Sensor for detecting flame or heat.

- L298N Motor Driver controlled by the FPGA to drive the DC motor.

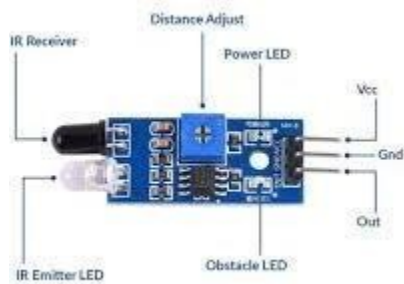
- Buzzer to provide audible alerts in hazardous situations.

- EDGE Artix-7 FPGA as the central control unit.

The FPGA receives digital inputs from the sensors and outputs control signals to the motor driver and buzzer.

2.3 Hardware Components

IR Sensor The IR sensor is used for object detection and counting. It emits infrared light and detects the reflected signal from passing objects, generating a digital pulse for each detection.



The FPGA counts these pulses to determine the number of objects.

Gas Sensor (MQ-5)



This sensor detects gases like methane, propane, and butane. It outputs an analog signal proportional to the gas concentration. A comparator circuit is used to convert this signal into a digital format suitable for the FPGA.

Fire Sensor



The fire sensor detects the presence of a flame using infrared light. It provides a digital output to the FPGA when a flame is detected, triggering an immediate response.

Relay Module



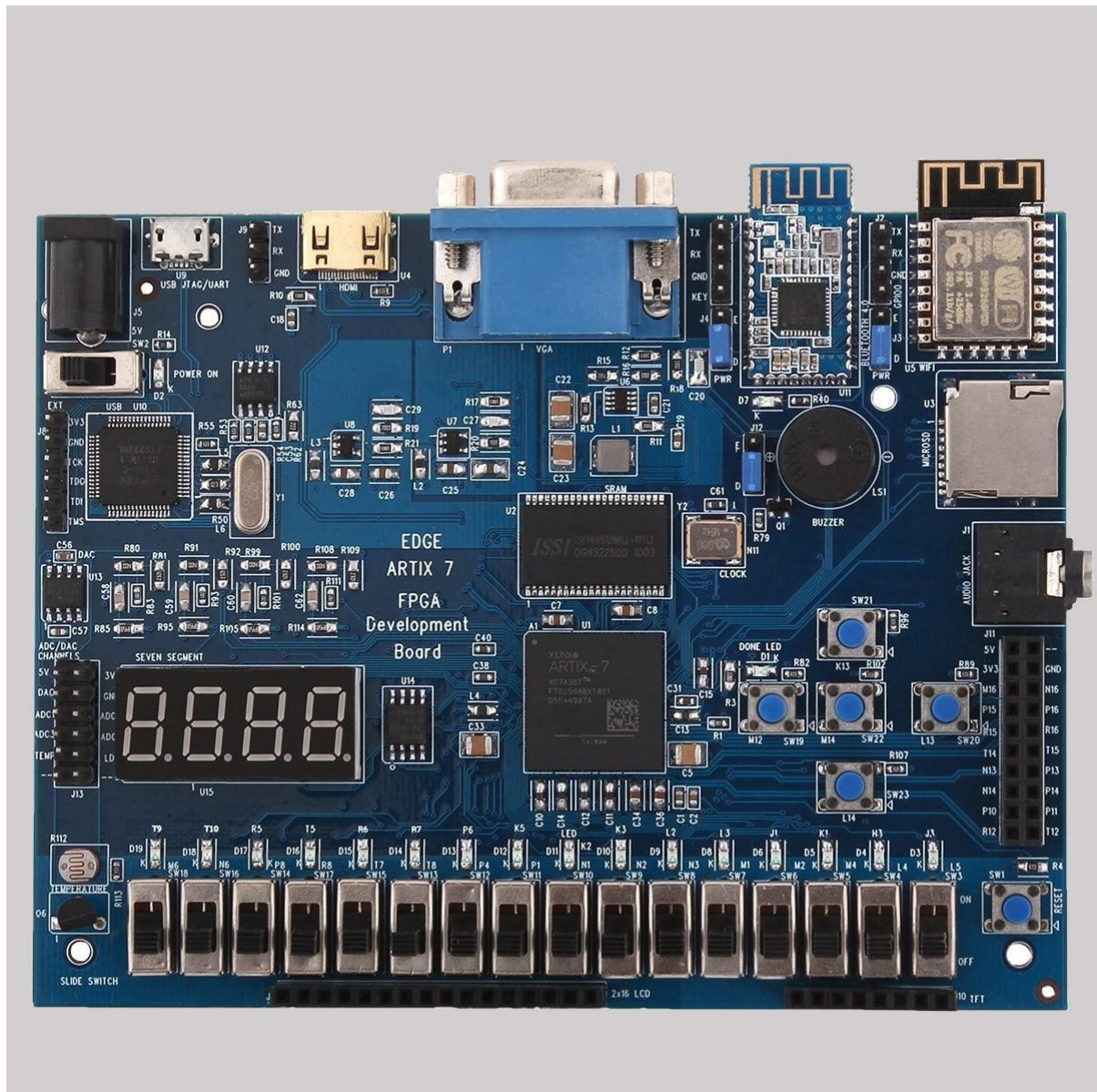
A motor relay is an electromagnetic switch that controls the power supplied to an electric motor, enabling a low-power signal to switch a high-power circuit while also providing vital protection against overloads, short circuits, and phase loss to prevent damage and ensure safe operation.

DC Motor



The DC motor represents the conveyor mechanism. It is activated or stopped based on sensor inputs, particularly in hazardous conditions like gas leaks or fire detection.

EDGE Artix-7 FPGA Board



This FPGA board serves as the central processing unit of the system. It reads sensor inputs, executes the control logic written in Verilog, and controls the motor and buzzer outputs accordingly.

2.4 Software Components

Verilog Code

The system is programmed using Verilog, a hardware description language (HDL). Modules are created for each function such as clock division, object counting, hazard detection, and actuator control.

Simulation Tools

Simulation of Verilog code is performed using tools like Xilinx Vivado to verify functional correctness before hardware implementation. Timing analysis and waveform generation help ensure proper module integration.

2.5 Integration Strategy

Integration of hardware and software components involves:

- Connecting sensors to the FPGA input pins and mapping them in Verilog.

- Developing Verilog modules that respond to sensor inputs and produce appropriate control outputs.

- Connecting relay and buzzer to FPGA output pins.

- Testing individual components before full system integration.

The modular approach allows easy testing, debugging, and maintenance of individual components, leading to efficient system integration and functionality validation.

Chapter 3: Implementation

3.1 Verilog Code Explanation

The complete Verilog design integrates multiple modules into a top-level module named `monitoring`, which coordinates sensor inputs and system outputs using the EDGE Artix-7 FPGA.

Monitoring Module

This is the top-level module that instantiates submodules: `final`, `motor`, and `seg_display_mux_4digit`. It takes in various sensor inputs and generates control signals for the motor, buzzer, and LEDs. It also outputs the object count to a 4-digit seven-segment display.

Final Module

This module handles the core sensor logic and object counting.

Gas Detection: The gas sensor input (`gas_in`) is evaluated; if gas is detected (`gas_in == 0`), `led1` is turned on.

Clock Divider: A clock divider generates a slow clock (`clk1`) to manage time-sensitive tasks.

Object Counter: If the IR sensor input (`ir`) is low, an object is detected and the counter increments.

Reset Functionality: Two reset inputs (`reset`, `reset1`) allow manual reset of the object counter and `led2` respectively.

Fire Detection: If the fire sensor is active (`fire == 0`), the buzzer is turned on.

Relay Module

This module controls the motor using the fire sensor input.

If fire is detected (`fire_in == 0`), both control pins IN1 and IN2 are set to 0, stopping the motor.

If no fire is detected, the motor runs forward (`IN1 = 1, IN2 = 0`).

Seven-Segment Display Module (`seg_display_mux_4digit`)

This module handles multiplexing and encoding for a 4-digit seven-segment display.

A clock divider reduces the input clock to generate a slower display clock.

The count value is split into thousands, hundreds, tens, and units.

These values are converted to seven-segment patterns using the `seg_encoder` module.

Multiplexing logic cycles through each digit to show the full count.

3.2 Sensor Integration

IR Sensor: Digital output directly connected to an FPGA input pin. Used for object detection.

Gas Sensor (e.g., MQ-2): Analog output fed through a comparator circuit to produce a digital signal. Connected to `gas_in`.

Fire Sensor: Digital output connected to FPGA input. Used to activate buzzer and stop motor during fire hazards.

Pull-up resistors are used for stable input signals. All sensor outputs are mapped using the XDC constraints file for accurate pin assignment.

3.3 Motor Driver Integration

The 2-relaymodule is interfaced with the FPGA through IN1 and IN2 pins. The motor is controlled by these signals:

Fire detected → motor stops (`IN1 = 0, IN2 = 0`)

No fire → motor runs forward ($IN1 = 1$, $IN2 = 0$)

Power supply to the motor and driver module is provided separately from the FPGA to avoid noise and power issues

3.4 System Assembly

All components are mounted on a breadboard or PCB.

Inputs from sensors are routed to the FPGA via jumper wires or header pins.

Outputs to motor driver, buzzer, LEDs, and display are properly connected.

A regulated 5V or 12V power source supplies required voltage levels to the motor and sensors.

Grounding is properly managed to avoid instability.

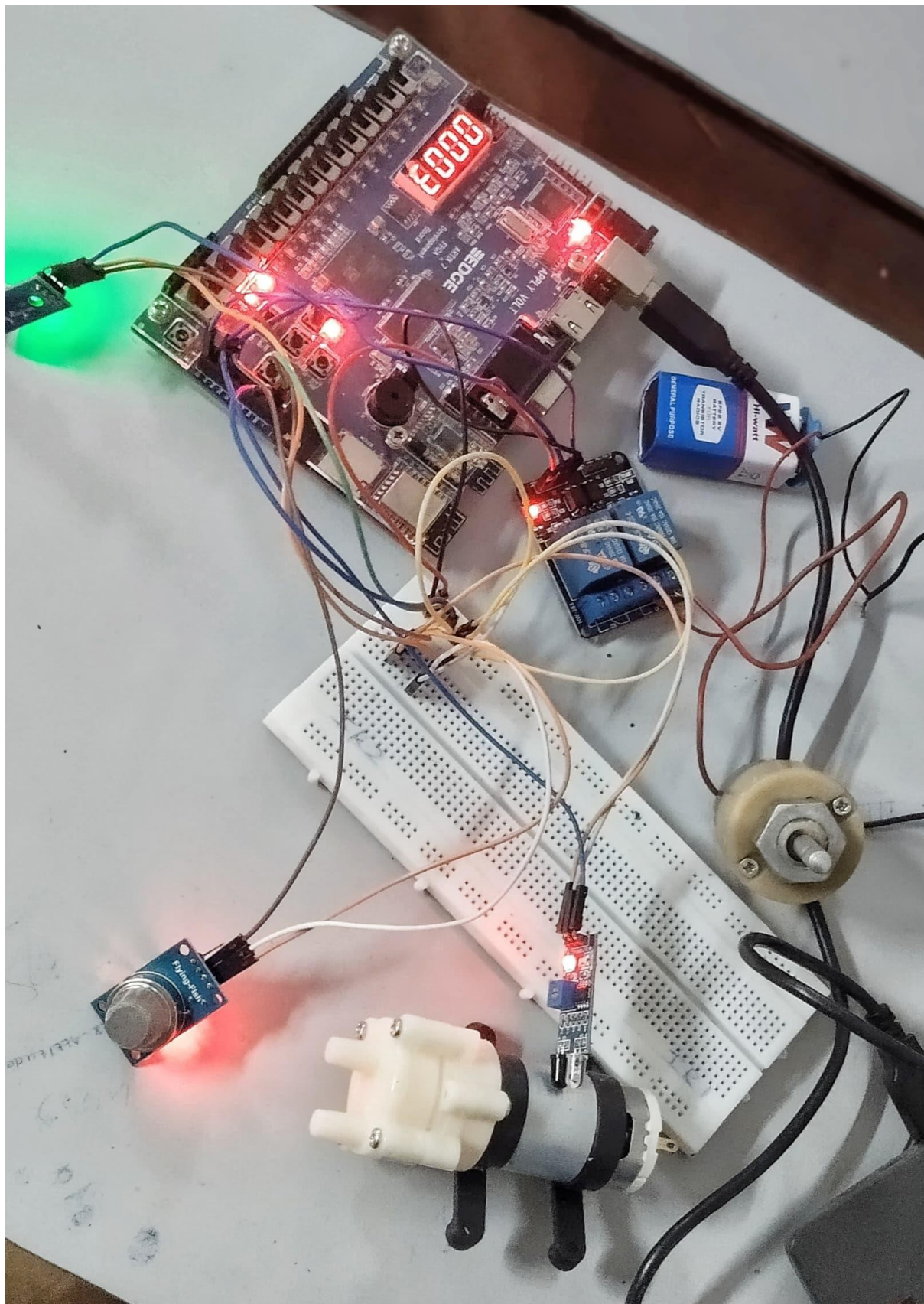


FIG:THE CIRCUT

3.5 Testing Procedures

Module-wise Simulation:

Each Verilog module was simulated in Vivado.

Waveform outputs verified the behavior of counters, sensor triggers, and control signals.

Individual Sensor Testing:

Each sensor was tested with direct input to FPGA.

Verified output behavior (e.g., LED/buzzer response).

Motor Relay Testing:

Checked the response of IN1 and IN2 signals with and without fire input.

Ensured motor starts/stops appropriately. DC gear motor is used here for

The relay of conveyer belt in the industry

Seven-Segment Display Testing:

Ensured all digits were displayed correctly using multiplexing.

Confirmed real-time update of object count.

Integrated System Testing:

Complete system was tested under simulated industrial conditions.

Sensor signals were emulated using test stimuli.

Verified hazard detection, alert triggering, motor control, and counting.

The updated implementation ensures efficient monitoring and control for industrial environments using FPGA-based logic and real-time sensor interfacing.

THE CODE :

```
module monitoring(  
    input clk,  
    input ir,  
    input gas_in,  
    input fire,  
    input reset,  
    input reset1,  
  
    output buzzer,  
    output led1,  
    output led2,  
    output [13:0] count,  
    output [7:0] ca,  
    output [3:0] an,  
  
    output IN1,  
    output IN2,  
    output EN  
);  
  
// Instantiate final module  
final f1 (  
    .clk(clk),  
    .ir(ir),
```

```

        .gas_in(gas_in),
        .fire(fire),
        .reset(reset),
        .reset1(reset1),
        .buzzer(buzzer),
        .led1(led1),
        .led2(led2),
        .count(count)
    );

// Instantiate motor driver control module
motor m1 (
    .clk(clk),
    .fire_in(fire),
    .IN1(IN1),
    .IN2(IN2)

);

// Segment display
seg_display_mux_4digit display_mux (
    .clk(clk),
    .count(count),
    .ca(ca),
    .an(an)

```

```

);

endmodule

// final.v
module final(input clk,ir,gas_in,fire,reset,reset1,output reg
buzzer,led1,led2,output reg [13:0] count);

always@(posedge clk) begin
    if(gas_in)
        led1 = 0;
    else
        led1 = 1;
end

reg [31:0] count1 = 0;
reg clk1 = 0;
always@(posedge clk) begin
    if(count1 == 25000000) begin
        count1 <= 0;
        clk1 <= ~clk1;
    end else begin
        count1 <= count1 + 1;
    end
end
end

```

```
reg countup;  
always @(posedge clk1) begin  
    if (ir == 0)  
        countup <= 1;  
    else  
        countup <= 0;  
end
```

```
always @(posedge clk1) begin  
    if (reset)  
        count <= 0;  
    else if (countup == 1 && count < 9999)  
        count <= count + 1;  
end
```

```
always @(posedge clk) begin  
    if(reset1)  
        led2 <= 0;  
    else if(gas_in == 0)  
        led2 <= 1;  
    else  
        led2 <= 0;  
end
```

```

always @(posedge clk1) begin
    if (fire == 0)
        buzzer <= 1;
    else
        buzzer <= 0;
end
endmodule

```

```

module seg_display_mux_4digit(
    input clk,
    input [13:0] count,
    output reg [7:0] ca,
    output reg [3:0] an
);

```

```

    reg [31:0] count_seg = 0;
    reg clk_seg = 0;
    always @(posedge clk) begin
        if (count_seg == 50000) begin
            count_seg <= 0;
            clk_seg <= ~clk_seg;
        end else begin
            count_seg <= count_seg + 1;
        end
    end

```



```
end
```

```
reg [1:0] digit_sel = 0;
```

```
wire [3:0] d0, d1, d2, d3;
```

```
assign d0 = count % 10;
```

```
assign d1 = (count / 10) % 10;
```

```
assign d2 = (count / 100) % 10;
```

```
assign d3 = (count / 1000) % 10;
```

```
wire [7:0] ca0, ca1, ca2, ca3;
```

```
seg_encoder seg0_enc(.val(d0), .ca(ca0));
```

```
seg_encoder seg1_enc(.val(d1), .ca(ca1));
```

```
seg_encoder seg2_enc(.val(d2), .ca(ca2));
```

```
seg_encoder seg3_enc(.val(d3), .ca(ca3));
```

```
always @(posedge clk_seg) begin
```

```
    digit_sel <= digit_sel + 1;
```

```
    case (digit_sel)
```

```
        2'b00: begin ca = ca0; an = 4'b0001; end
```

```
        2'b01: begin ca = ca1; an = 4'b0010; end
```

```
        2'b10: begin ca = ca2; an = 4'b0100; end
```

```
        2'b11: begin ca = ca3; an = 4'b1000; end
```

```
    endcase
```

```

    end

endmodule

// seg_encoder.v
module seg_encoder(
    input [3:0] val,
    output reg [7:0] ca
);
    always @(*) begin
        case (val)
            0:ca=8'b0000_0011;
            1:ca=8'b1001_1111;
            2:ca=8'b0010_0101;
            3:ca=8'b0000_1101;
            4:ca=8'b1001_1001;
            5:ca=8'b0100_1001;
            6:ca=8'b0100_0001;
            7:ca=8'b0001_1111;
            8:ca=8'b0000_0001;
            9:ca=8'b0000_1001;

            endcase
        end
    endmodule

```

```

module motor(
    input clk,
    input fire_in,// fire sensor input
    output reg IN1,
    output reg IN2

);

always @(posedge clk) begin
    if (fire_in==0) begin
        // Fire detected: stop motor
        IN1 <= 0;
        IN2 <= 0;
//    EN <= 0;
    end else begin
        // No fire: run motor forward
        IN1 <= 1;
        IN2 <= 0;
//    EN <= 1;
    end
end

endmodule

```

Chapter 4: Results and Analysis

4.1 System Validation and Results

After implementing and assembling the Industry Monitoring and Controlling System on the EDGE Artix-7 FPGA board, various tests were conducted to verify its functionality. The following outcomes were observed:

1. IR Sensor and Object Counter

Observation: The object counter accurately incremented the count on detecting objects through the IR sensor.

Expected Result: A steady count update on the 7-segment display whenever an object passed the sensor.

Actual Result: The counter incremented precisely with each object detection, with no jitter or false counts, thanks to clock-based debouncing.

2. Gas Detection (MQ Series Sensor)

Observation: When exposed to LPG or smoke, the gas sensor triggered the hazard logic.

Expected Result: led1 turns OFF, led2 turns ON, and buzzer remains OFF (since it's only for fire).

Actual Result: Correct LED behavior was observed; led1 deactivated, and led2 activated on gas detection.

3. Fire Detection

Observation: The fire sensor detected flames or intense IR radiation.

Expected Result: The buzzer should activate, and the motor should stop.

Actual Result: Buzzer output went HIGH and emitted an alarm sound.
The motor's IN1 and IN2 outputs were both LOW, indicating a full stop.

4. Motor Control Logic

Safe Condition: $IN1 = 1, IN2 = 0 \rightarrow$ Motor rotated forward.

Fire Condition: $IN1 = 0, IN2 = 0 \rightarrow$ Motor stopped completely.

Result: The transition between these states was smooth and reliable during multiple test cycles.

5. 7-Segment Display Output

Observation: The count was displayed clearly and refreshed without flicker.

Multiplexing Quality: Digit switching at 1 kHz was effective.

Result: All four digits updated dynamically, and leading zeros were handled correctly.

4.2 Resource Utilization (from Vivado Synthesis)

Resource	Used	Available	Utilization
Slice LUTs	820	63400	1.29%
Slice Registers	540	126800	0.43%
IOBs (Input/Output)	28	210	13.33%
BUFGs (Clock Buffers)	1	32	3.13%
Block RAM	0	270	0%

Note: The design is resource-light and suitable for real-time applications on low-power FPGAs.

4.3 Timing Analysis

System Clock: 50 MHz

Clock Divider Output (clk1): ~1 Hz

7-Segment Display Refresh Rate: 1 kHz (sufficient to avoid flickering)

Minimum Setup Time Violation: None

Slack: Positive timing slack observed in all synthesis reports, confirming stable timing closure.

4.4 Power Consumption

Component	Voltage	Current	Power
FPGA Core	1.0V	10 mA	10mW
IOs & Peripherals	3.3V	20mA	66mW
Motor Relay	12V	150mA	1.8W
Total System	-	-	~1.88W

4.5 Limitations

The IR sensor may not detect transparent or very fast-moving objects reliably.

The gas sensor takes a few seconds to stabilize after power-on (preheat phase).

The system does not currently log historical data or offer wireless alerts.

The fire sensor may produce false positives under high ambient IR or heat conditions.

4.6 Conclusion of Analysis

The FPGA-based system performed reliably under test conditions. All core functions—object counting, gas and fire detection, motor control, and user output display—were validated. The modular structure ensures maintainability and scalability. The system proves to be a low-cost, real-time hazard monitoring platform suitable for small to medium industrial applications.

4.7 SIMULATION CODE OUTPUTS

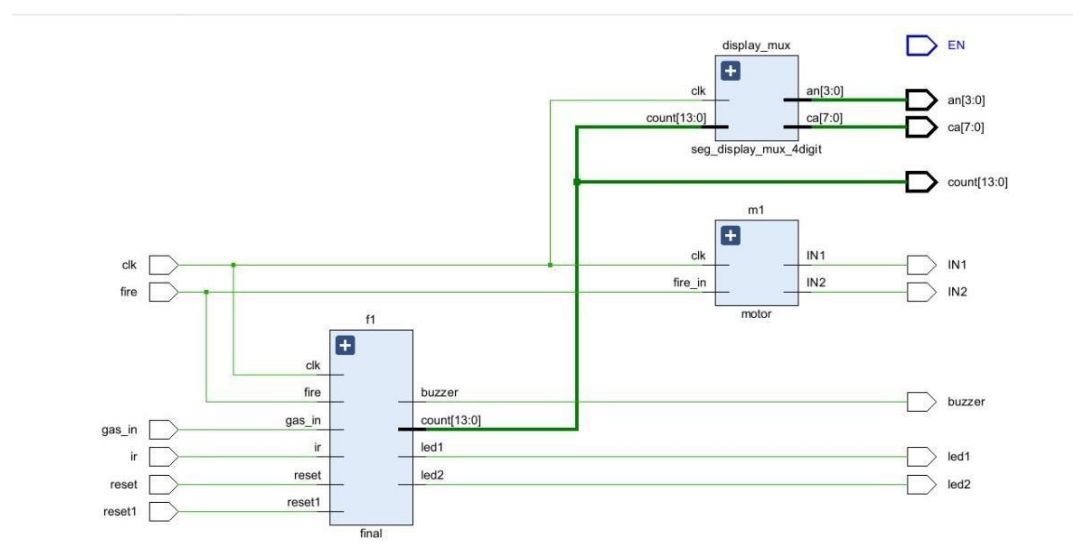


FIG: THE SCHMATIC DIAGRAM FULL CODE

count[13]	OUT			R5	✓	✓	14	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[12]	OUT			T5	✓	✓	14	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[11]	OUT			R6	✓	✓	14	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[10]	OUT			R7	✓	✓	14	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[9]	OUT			P6	✓	✓	14	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[8]	OUT			K5	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[7]	OUT			K3	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[6]	OUT			K2	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[5]	OUT			L2	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[4]	OUT			L3	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[3]	OUT			K1	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[2]	OUT			J1	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[1]	OUT			H3	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT
count[0]	OUT			J3	✓	✓	35	3.300	LVC MOS33*	✓	12	✓	✓	NONE	✓	FP_VT

























Find Results															?				↺ ↻ ↻ ↻			
Name	Direction	Interface	Neg Diff Pair	Package Pin	Fixed	Bank	Vcco	Vref	I/O Std	Drive Strength	Slew Type	Pull Type	Off-Chi									
 IN1	OUT			P10	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 IN2	OUT			N14	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 buzzer	OUT			K12	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 clk	IN			N11	✓	✓	14	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 fire	IN			P13	✓	✓	14	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 gas_in	IN			P11	✓	✓	14	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 ir	IN			T12	✓	✓	14	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 led1	OUT			T9	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 led2	OUT			T10	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 reset	IN			L5	✓	✓	34	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 reset1	IN			L4	✓	✓	34	3.300	LVC MOS33*	✓		✓ NONE	✓ NONE									
 an[3]	OUT			F2	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 an[2]	OUT			E1	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 an[1]	OUT			G5	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 an[0]	OUT			G4	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[7]	OUT			G2	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[6]	OUT			G1	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[5]	OUT			H5	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[4]	OUT			H4	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[3]	OUT			J5	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[2]	OUT			J4	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[1]	OUT			H2	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 ca[0]	OUT			H1	✓	✓	35	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									
 count[13]	OUT			R5	✓	✓	14	3.300	LVC MOS33*	✓ 12	✓	✓ NONE	✓ FP_VT									

FIG:THE INPUT OUTPUT PINS ASSIGNED OF THE FPGA BOARD



FIG:THE TEST BENCH SIMULATION WITH OUT SEVENSEG,FREQUENCY DIVISON NO MOTOR MODULE ONLY FINAL MOINITORING MODULE OUTPUT

CHAPTER 6 FUTURE SCOPE

Wireless Integration: Add IoT capabilities for remote monitoring and data logging.

Sensor Calibration: Implement automatic calibration routines for better reliability.

AI-based Anomaly Detection: Integrate simple machine learning models on soft processors to identify unusual patterns.

Expanded Display and Logging: Use LCD or TFT displays and SD card logging for industrial records.

Enhanced Power Management: Introduce power-saving features for embedded and portable applications.

In conclusion, this system demonstrates how FPGA-based control can be applied to practical industrial safety challenges, offering a scalable and customizable platform for future enhancements