Project Report: Smart Parking System

Team: #33 (Batch 1)

1. Project Overview

The objective of this project was to design and implement a comprehensive **Smart Parking System** using an Intel MAX 10 FPGA (DE10-Lite). The system manages a garage with a capacity of 8 spots, featuring three distinct modes of operation:

1. **Autonomous Mode:** Uses Infrared (IR) sensors to automatically detect vehicle entry and exit, controlling the gate and updating the available spot count without human intervention.
2. **Manual Mode:** Allows a human operator to control the gate via a toggle button, with strict logic to prevent capacity overflow or underflow.
3. **Hazard Response (Bonus):** An integrated Fire Safety protocol using an external Arduino Mega and Gas Sensor to override the system and force the gate open during emergencies.

2. Hardware Components & Architecture

2.1 Core Processing

- **FPGA Board:** Terasic DE10-Lite (Intel MAX 10 10M50DAF484C7G).
- **Microcontroller:** Arduino Mega 2560 (Used for analog sensor processing).

2.2 Sensors & Actuators

- **IR Obstacle Sensors (x2**
- **MQ-2 Gas/Smoke Sensor**
- **Servo Motor (SG90)**
- **Active Buzzer**
- **Resistors:** 3× 10kΩ (Used for voltage level shifting).

- **Jumper Wires**

2.3 Circuit Integration

To interface the 5V Arduino logic with the 3.3V FPGA logic, a **Voltage Divider circuit** (10kΩ / 20kΩ resistors) was implemented on the communication line to step down the signal voltage, protecting the FPGA GPIO pins.

3. Implementation Details

3.1 VHDL Logic Design (Finite State Machine)

The core logic was implemented in VHDL using a **Finite State Machine (FSM)** to solve common sensor issues such as signal bouncing and object jitter.

- **State** `WAITING`**:** The system waits for sensor input. The gate remains closed.
- **State** `ENTERING`**:** Triggered when the outer sensor detects a car (`01`). The gate opens. The system locks into this state and ignores the exit sensor until the car fully passes the inner sensor (`10`).
- **State** `EXITING`**:** Triggered when the inner sensor detects a car (`10`). The gate opens. The system waits for the car to clear the outer sensor (`01`).
- **State** `WAIT_FOR_CLEAR`**:** A safety state that holds the gate open as long as any sensor is blocked (`01`, `10`, or `00`), preventing the gate from closing on a vehicle.
- **Counter Logic:** The available spots (initialized to 8) are only decremented/incremented once the FSM confirms a completed pass-through sequence.

3.2 Manual Mode Logic

A separate logic block handles manual control. A push-button toggles the gate state.

- **Capacity Lockout:** The system checks `if parking > 0` before allowing the gate to open manually. If the lot is full, the gate remains locked closed.
- **Debouncing:** A 50ms software debouncer filters the button input to prevent a single press from registering as multiple toggles.

3.3 Fire Safety Override

The Arduino continuously monitors air quality. If smoke levels exceed the calibrated threshold (300):

1. Arduino sends a Logic High ('1') to the FPGA.
2. FPGA immediately overrides all Parking FSM states.
3. **Result:** Gate forces Open and Buzzer activates.
4. **Stability:** The Arduino firmware includes a 3-second latch to ensure the gate remains open during brief smoke fluctuations.

2.4 Component Sourcing

- TECH HUB
  - PHONE: 01070739598
  - INFO@TECH-HUBEG.COM
  - LOCATION: THREE TOWERS MALL(TAGAMOAA 1 INFRONT OF REHAB GATE 21)