TEAM BACKLOG

Aryan Bagoria
SIT 329

Front - End Deployment

User Application Interface
Description : Create a user-friendly interface that provides real-time information to users about parking space availability. This interface could be a graphical display or a simple LED display. The FPGA communicates with this interface, sending data that informs users about available and occupied parking spaces.

Real-Time Data Integration
Description : Integrate real-time parking space availability data from the FPGA into the user application's interface. Update the display dynamically as data changes to provide users with accurate information.

Communication with FPGA Backend
Description : Implement communication between the user application and the FPGA backend. Use the defined communication protocol to send data requests and receive optimised parking space occupancy information from the FPGA.

Hardware Design and Implementation

Communication Protocol Establishment
Description: This functionality involves establishing communication protocols between the selected sensors and the FPGA. We will determine the most suitable protocol (e.g., SPI or I2C) for reliable and efficient data transfer.

Ultrasonic Sensor Integration
Description: Implement the interfacing of ultrasonic sensors with the FPGA. This involves configuring the FPGA's input pins to receive signals from the sensors. The FPGA should be able to measure the time taken for the ultrasonic signals to travel to the car and back, which can be used to calculate the distance to the car and determine occupancy. Also evaluate sensor specifications such as range, accuracy, and interface compatibility. Study the data sheets and protocols of selected sensors for integration.

Distance Calculation and Parking Space Detection
Description: Develop logic within the FPGA to calculate the distance to the detected car based on the time-of-flight of the ultrasonic signals. This information is used to determine if a parking space is occupied or vacant. By comparing the calculated distance to a predefined threshold, the system can make a decision about parking space occupancy.

SOFTWARE AND DEVELOPMENT

Verilog Coding for Sensor Interface
Description: Write Verilog code to interface with the ultrasonic sensors. Define the input and output pins for sensor communication, implement logic to capture and process the sensor signals, and ensure proper synchronisation with the FPGA clock.

Real-Time Data Collection and Processing Algorithm
Description: Implement a mechanism to continuously gather data from all ultrasonic sensors in real-time. This involves designing a pipeline that can handle the incoming data streams from multiple sensors simultaneously. The FPGA should be able to manage and process these data streams efficiently.

Parking Space Status Updates
Description: Develop a mechanism to update the status of parking spaces based on the processed data. This could involve creating a memory structure within the FPGA to keep track of the occupancy status of each parking space. When a change in occupancy is detected, the system updates this memory structure accordingly.

Error Handling and System Integrity
Description: Implement mechanisms to ensure the integrity of the system and handle potential errors. This includes error detection during sensor data collection, communication errors between the FPGA and the user application, and handling unexpected scenarios gracefully to prevent system crashes.

Performance Optimization and Benchmarking
Description: Conduct performance optimization to fine-tune the algorithm and system. Benchmark the system's performance under various conditions, such as different car arrival rates and parking space distributions. The goal is to ensure that the system operates efficiently and meets performance expectations.

TESTING AND INTEGRATION

Test sensor functionality for car detection
Description: Simulate various car presence and absence scenarios to validate the accuracy of the ultrasonic sensors. Collect data on the sensors' response times and reliability. Fine-tune sensor parameters, such as sensitivity thresholds, based on the results of the testing. Ensure that the sensors can consistently detect and report accurate car occupancy status.

Evaluate algorithm's performance under load
Description: Simulate a high load scenario by generating a large volume of incoming data points from the sensors. Measure the algorithm's processing speed, memory utilization, and any potential bottlenecks. Optimize the algorithm's code structure and logic if performance issues are identified. Ensure that the algorithm can handle the expected data load while maintaining real-time processing capabilities.

Testing for Multiple users
Description: Simulate multiple users interacting with the system simultaneously. Test the system's ability to handle concurrent user requests while maintaining performance.

DOCUMENTATION AND PRESENTATION

Document sensor integration and hardware documentation
Description: Prepare comprehensive documentation detailing the sensor integration process. Include circuit diagrams, connection details, and relevant code snippets. Create a comprehensive guide for future reference.