Project Report: Embedded System State Machine for Car Control

This project involves the design and implementation of a state machine for an embedded system that controls a car's driving behavior. The system utilizes an ultrasonic sensor (US) to detect obstacles, and based on the sensor readings, it either keeps the car in a waiting state or moves it forward using a DC motor. The project is written in C and uses state-driven logic to handle different operational states for the car and motor.

by مصطفى بن عبدالله

1. Introduction

This project involves the design and implementation of a state machine for an embedded system that controls a car's driving behavior. The system utilizes an ultrasonic sensor (US) to detect obstacles, and based on the sensor readings, it either keeps the car in a waiting state or moves it forward using a DC motor. The project is written in C and uses state-driven logic to handle different operational states for the car and motor.

1 **Ultrasonic Sensor (US)**

Measures the distance to an obstacle.

2 **DC Motor**

Controls the movement of the car.

3 **State Machine**

Manages the system’s behavior based on sensor input and other conditions.

2. Objective

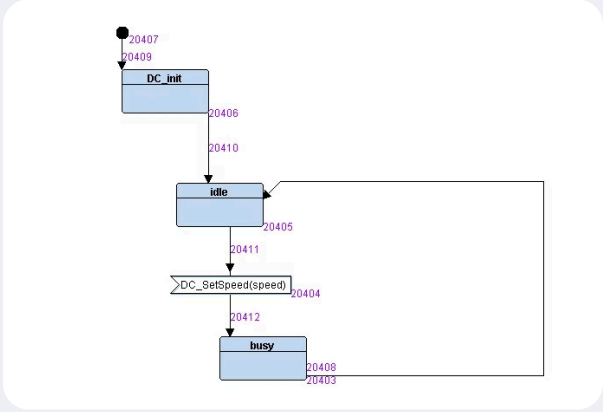
The objective of this project is to develop a simple, efficient state machine to manage the car's behavior:

- The car waits if an obstacle is too close.
- The car drives if the path is clear.
- The system interacts with both the ultrasonic sensor and the DC motor, making decisions based on the sensor’s distance readings.

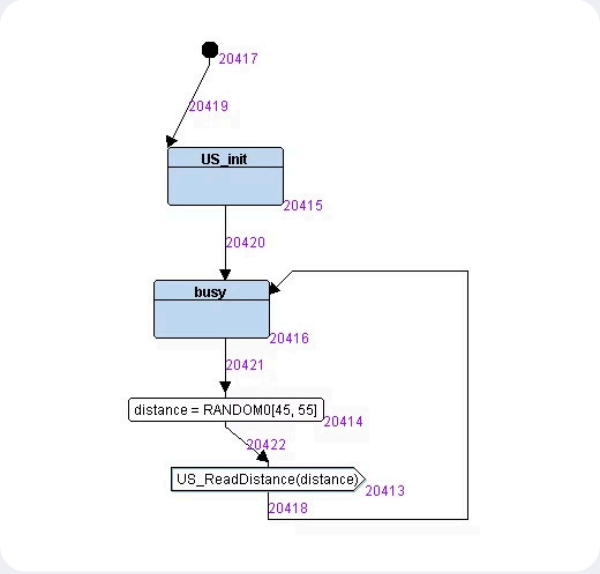
3. State Machines

The system is built on a finite state machine (FSM) that transitions between different states based on sensor input:

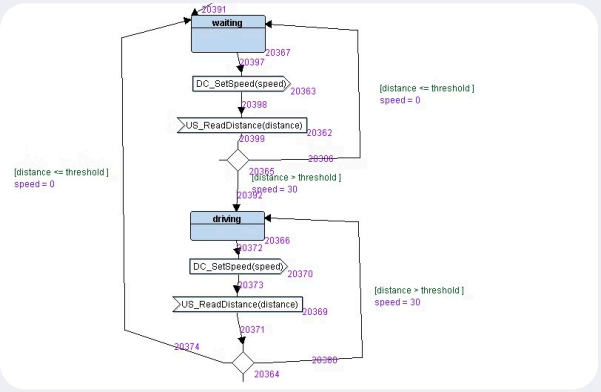
DC State Machine



US State Machine



CA State Machine



4. Execution Flow

1

System Initialization

The ultrasonic sensor and the DC motor are initialized. The car starts in the waiting state (CA_waiting), the motor is idle (DC_MOTOR_idle), and the ultrasonic sensor begins in a busy state (US_busy).

2

Distance Measurement

The ultrasonic sensor continuously reads the distance to obstacles and passes the result to the SetCarState() function.

3

State Transition

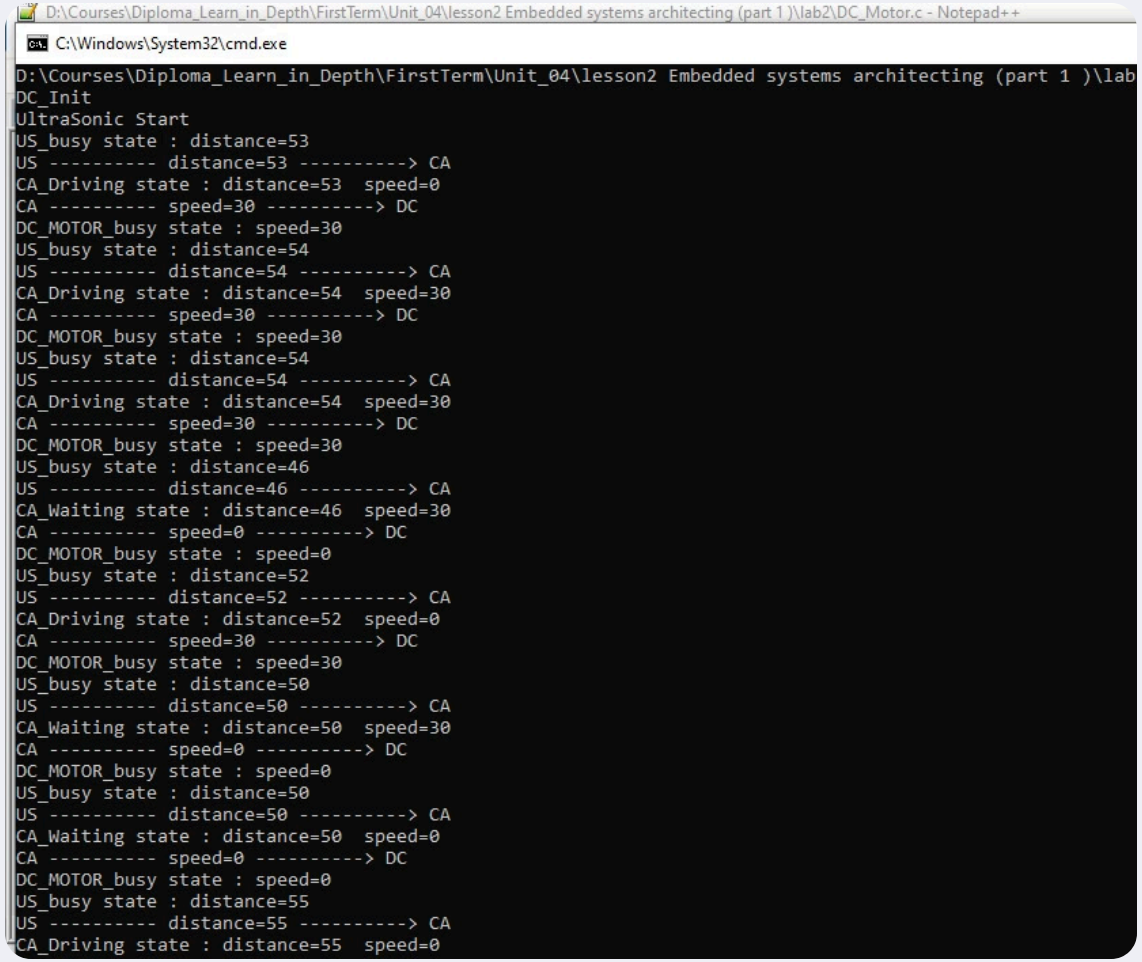
Based on the distance: If the distance is greater than the threshold (50), the car enters the driving state and the motor engages. If the distance is less than the threshold, the car waits, and the motor stops.

4

Motor Control

When the car is in the driving state, the motor is activated, and the car moves at a specified speed (30). The motor transitions between idle and busy states based on whether the car is driving or waiting.

5. System Output



6. Observations and Issues

The system successfully transitions between states based on sensor input. The car drives or waits depending on the distance to an obstacle.

Issue	Description
Infinite Loop in US_busy	The system doesn't leave the US_busy state because the state is hardcoded to remain in that state.
Hardcoded Transition in Motor States	The motor transitions back to idle immediately after setting the speed, making the DC_MOTOR_busy state ineffective.

Modify the US_busy state to allow transitions to other states. Adjust the motor state logic so that the motor remains in the busy state until a specific condition is met (e.g., reaching a target distance or completing a movement).

7. Conclusion

This project demonstrates the implementation of a simple state machine for controlling a car's movement based on sensor input. The system successfully switches between waiting and driving states using an ultrasonic sensor and a DC motor. However, improvements can be made to handle state transitions more effectively and make the system more robust.

8. Future Work

- Implement more complex state transitions based on additional inputs or timing.
- Add functionality for speed adjustment based on varying distances.
- Introduce error handling for sensor failures or motor issues.