Project Scope

Project Title: Motor Controller Simulator with UI (ZLAC8015D CANopen Servo Driver)

Project Overview

The objective of this project is to develop a software-based simulator with a graphical user interface (UI) that emulates the behaviour of the ZLAC8015D CANopen motor controller. The simulator will replicate CANopen communication, CiA301/CiA402 state machine behavior, and motor operation modes such as profile position, velocity, and torque control. This tool will enable testing of robotics or vehicle software without requiring physical motor hardware, improving development speed and validation workflows.

Objectives

- Simulate CANopen-based motor controller communication.
- Implement CiA402 state machine and control/status word handling.
- Emulate Profile Position, Velocity, and Torque modes.
- Develop an interactive UI to visualize motor state, commands, and feedback.
- Enable developers to test CAN commands and motion control logic offline.

Scope of Work

1. CANopen Communication Simulation

- Implement CANopen protocol structure including:
  - NMT commands and state transitions.
  - SDO read/write interactions with object dictionary.
  - TPDO and RPDO message handling.
  - Heartbeat message generation.
- Simulate Node-ID behaviour and communication flow based on ZLAC8015D controller architecture.

2. CiA402 Motion Controller Behaviour

- Model the motor controller state machine (Switch On Disabled, Ready to Switch On, Operation Enable, Fault, etc.).
- Implement control word (0x6040) and status word (0x6041) logic.
- Support operation modes:
  - Profile Position Mode
  - Profile Velocity Mode
  - Profile Torque Mode
- Simulate parameters such as target position, target velocity, acceleration, torque, and system status.

3. Graphical User Interface (UI) Development

- Create a dashboard-style UI including:

  - Motor state and operating mode display
  - Control word and status word monitoring
  - Speed, torque, and position visualization
  - CAN message log panel
- Provide user inputs to send commands such as start, stop, enable operation, and target value changes.

4. Integration & Testing

- Validate simulator with CANopen tools or software stack.
- Ensure correct message format and timing behaviour.
- Demonstrate interaction between simulator and external compute or robotics control software.

Deliverables

- Functional ZLAC8015D motor controller simulator software.
- Interactive UI for monitoring and command testing.
- Documentation describing object dictionary usage and simulation architecture.
- Demo showing simulated CANopen communication and motor control behaviour.

Tools & Technologies

- Programming Language: Python / C++ / Qt / Web UI
- CANopen Libraries / SocketCAN / Virtual CAN
- UI Framework (Qt / PyQt / Web-based dashboard)
- Version Control (Git)

Expected Outcomes

By the end of the internship, the intern will deliver a realistic software simulator capable of mimicking a ZLAC8015D CANopen servo driver, including state machine transitions, PDO communication, and motion control modes, enabling reliable software testing without physical motor hardware.