Real-Time CANopen to Spike2 Data Conversion System

Problem Statement

The objective of this project is to create a real-time data conversion system to:

- Read position data from Harmonic Drive FHA motors using the CANopen protocol
- Process this data through a CANUSB adapter into serial USB format
- Integrate the serial data into the Spike2 software environment
- Synchronize and align the motor position data with experimental signals from the 1401 ADC/DAC converter
- Implement this solution using Spike2's C++ SDK

Technical Requirements

- **Multi-threaded Architecture**:
  - Dedicated thread for continuous CANUSB serial monitoring
  - Separate thread for Spike2 data integration and timing synchronization
  - Thread-safe data handling between components
- **Real-time Data Conversion**: Convert and align motor position data with experimental data in real-time
- **Precise Timing Alignment**: Accurate synchronization between converted motor data and 1401 ADC/DAC data
- **Error Handling**: Manage potential issues in data conversion, thread communication, and synchronization
- **Performance Optimization**: Ensure stable real-time operation with minimal latency

Project Requirements and Proposal

Development Phases and Milestones

1. **Design Phase**

   - **Milestones**:
     - Design thread architecture and communication
     - Define data conversion workflow
     - Specify timing synchronization requirements
     - Document performance requirements

2. **Core Development**

   - **Milestones**:
     - Implement CANUSB serial monitoring thread
     - Develop Spike2 data integration thread
     - Create thread synchronization mechanism
     - Integrate with 1401 ADC/DAC data stream

3. **Testing and Optimization**
    - **Milestones**:
        - Verify thread safety and performance
        - Validate timing synchronization
        - Test real-time performance
        - Stress test under various data rates
4. **Documentation**
    - **Milestones**:
        - Prepare technical documentation
        - Create setup guide with examples
        - Document threading architecture

Deliverables

- **Functional Software**:
    - Multi-threaded C++ system for real-time CANopen data conversion
    - Thread-safe data handling between components
    - Configurable timing parameters
- **Documentation**:
    - Technical documentation
    - Setup and usage instructions
    - Performance characteristics
- **Testing Reports**:
    - Thread safety validation
    - Timing synchronization accuracy
    - Real-time performance metrics
- **Example Code**: Sample implementations demonstrating usage