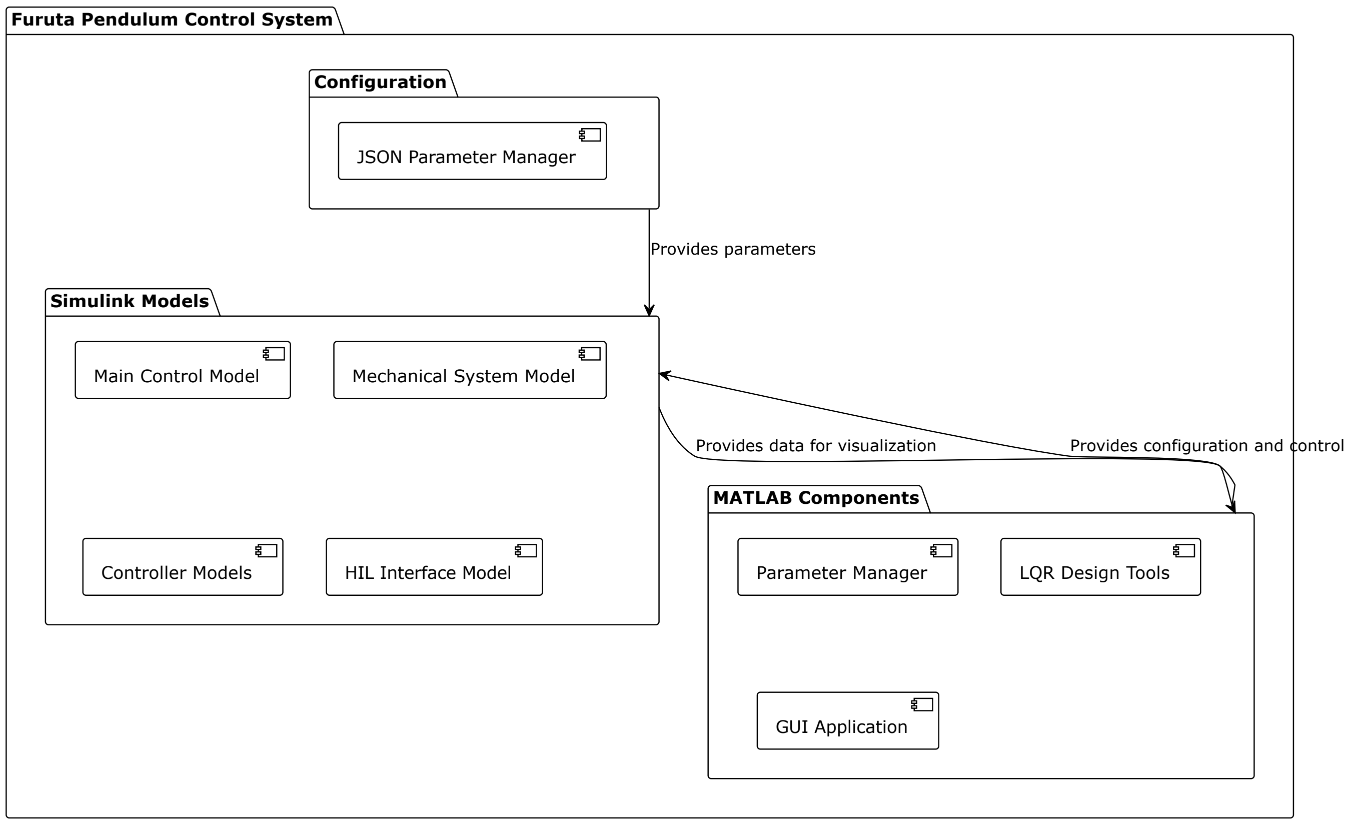
Furuta Pendulum Control System - Architecture Documentation

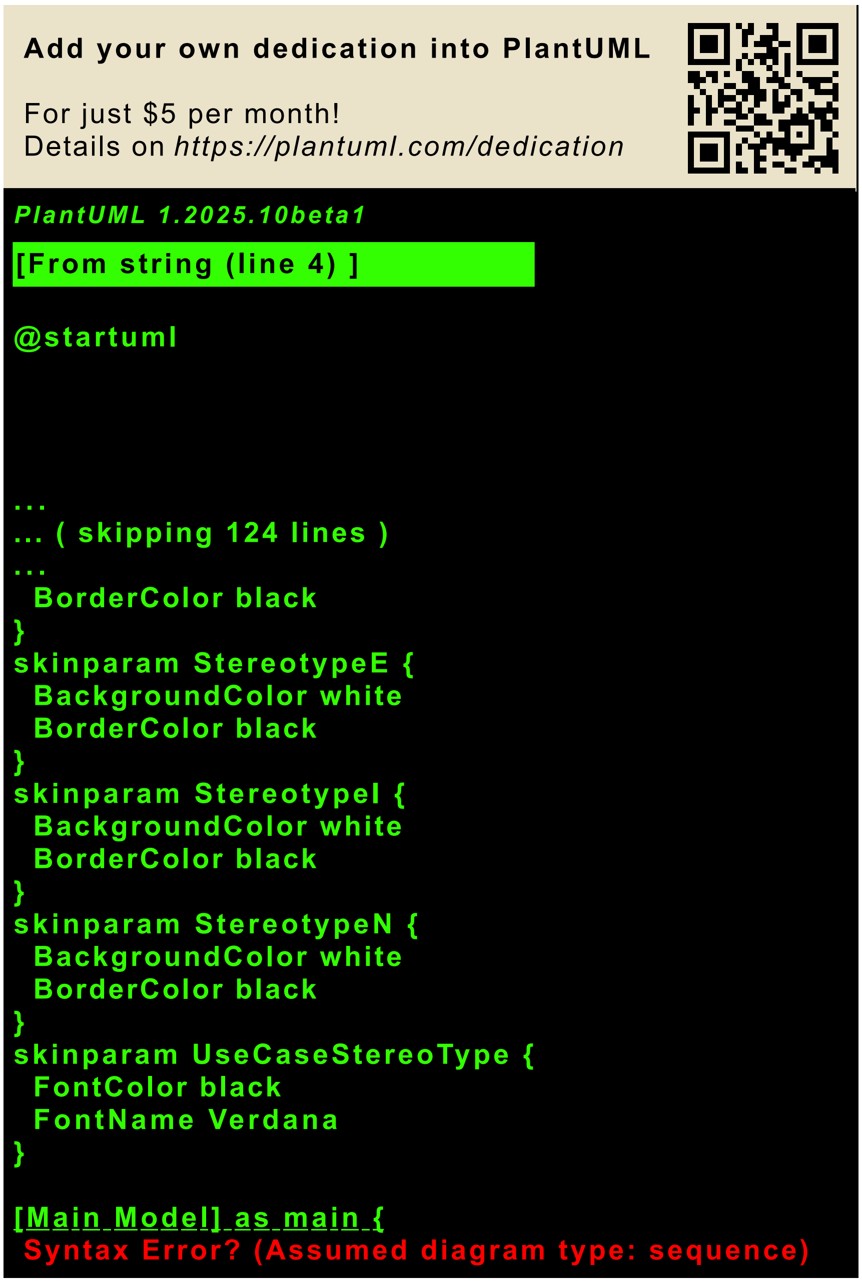
1. System Overview

The Furuta Pendulum Control System is designed with a Simulink-centric architecture to support both simulation and Hardware-in-the-Loop (HIL) operation. The system primarily uses Simulink blocks for control and Simscape for mechanical modeling, with minimal MATLAB code for parameter management and GUI.

2. High-Level Architecture



3. Simulink Model Architecture

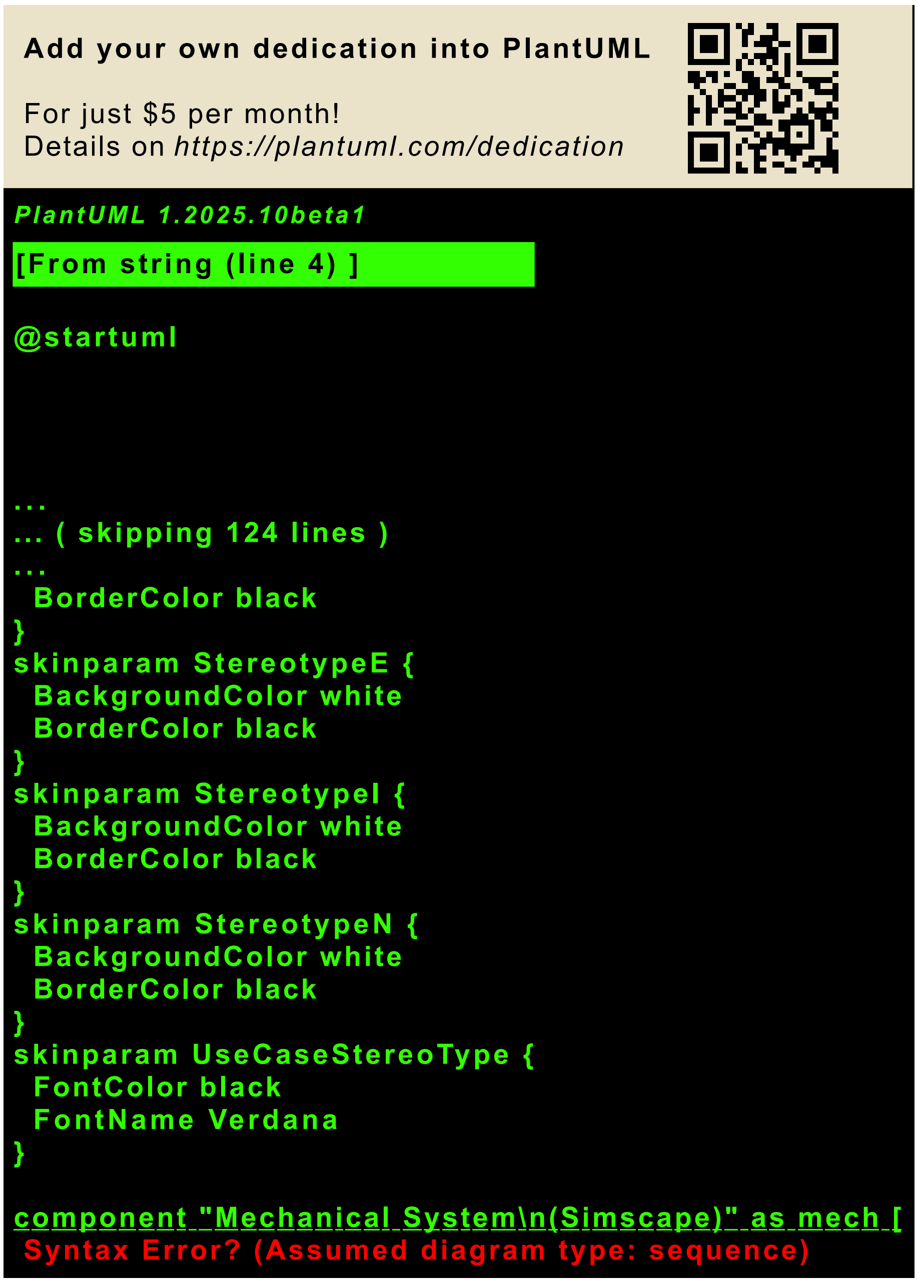


Add your own dedication into PlantUML

For just $5 per month!
Details on *https://plantuml.com/dedication*

```
PlantUML 1.2025.10beta1
[From string (line 4) ]

@startuml




...
... ( skipping 124 lines )
...
  BorderColor black
}
skinparam StereotypeE {
  BackgroundColor white
  BorderColor black
}
skinparam StereotypeI {
  BackgroundColor white
  BorderColor black
}
skinparam StereotypeN {
  BackgroundColor white
  BorderColor black
}
skinparam UseCaseStereoType {
  FontColor black
  FontName Verdana
}

[Main Model] as main {
 Syntax Error? (Assumed diagram type: sequence)
```

4. Detailed Block Diagram

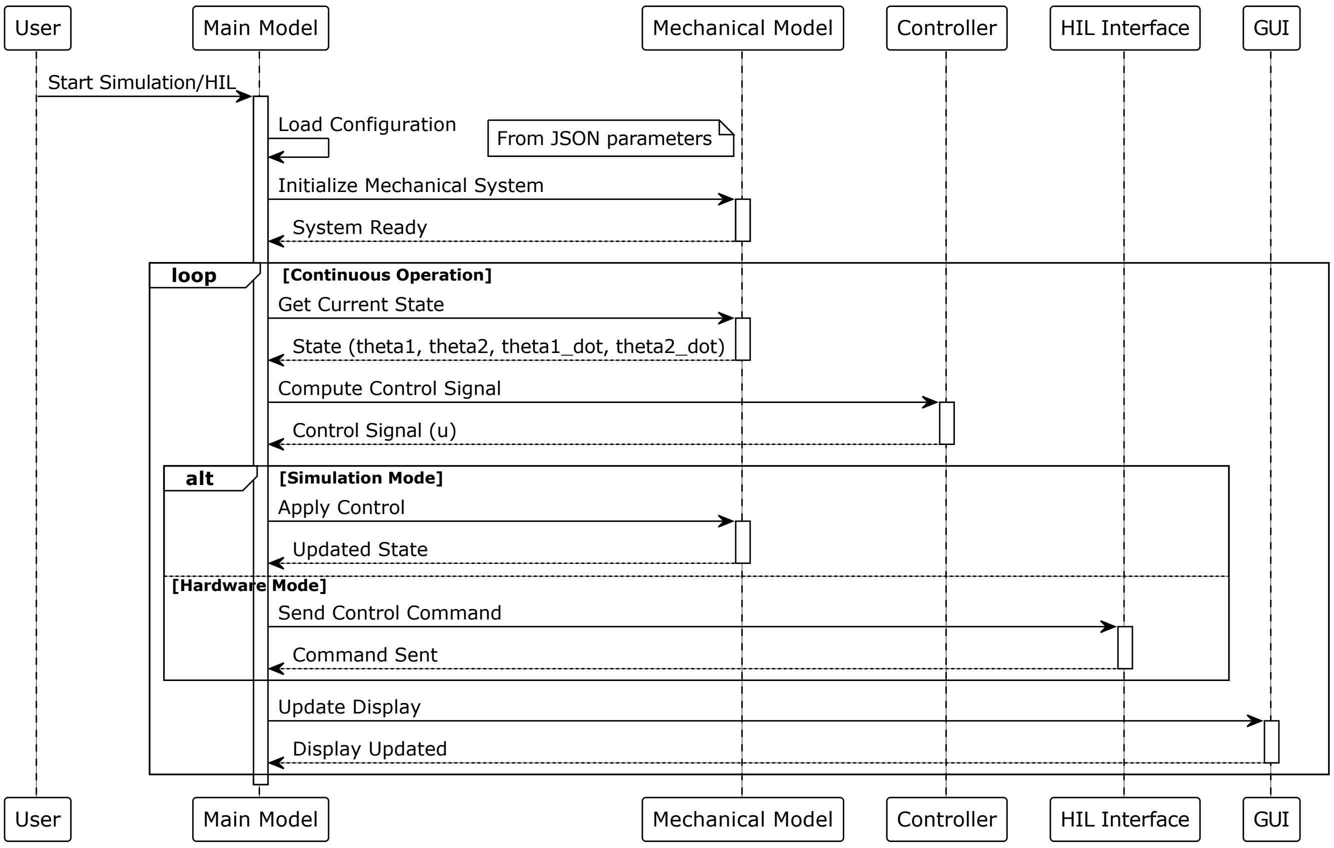


Add your own dedication into PlantUML

For just $5 per month!
Details on *https://plantuml.com/dedication*

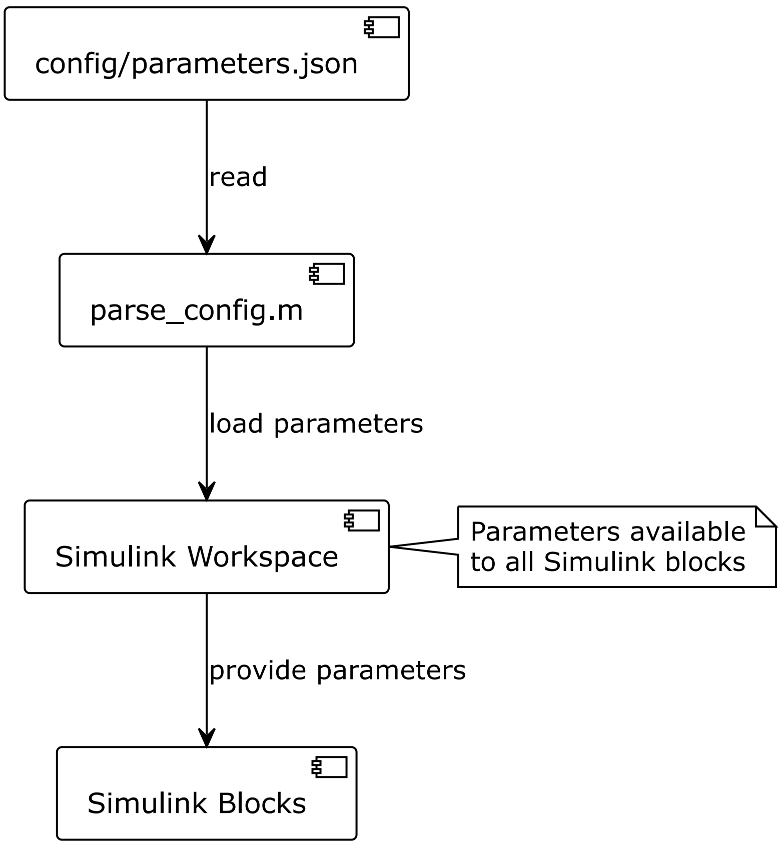
```
PlantUML 1.2025.10beta1
[From string (line 4) ]

@startuml




...
... ( skipping 124 lines )
...
  BorderColor black
}
skinparam StereotypeE {
  BackgroundColor white
  BorderColor black
}
skinparam StereotypeI {
  BackgroundColor white
  BorderColor black
}
skinparam StereotypeN {
  BackgroundColor white
  BorderColor black
}
skinparam UseCaseStereoType {
  FontColor black
  FontName Verdana
}

component "Mechanical System\n(Simscape)" as mech [
 Syntax Error? (Assumed diagram type: sequence)
```

5. Data Flow Architecture



6. Configuration Management

The system uses a JSON-based configuration approach that feeds parameters into Simulink blocks:



7. Component Responsibilities

7.1 Mechanical System (Simscape)

- Models the physical Furuta Pendulum using Simscape Multibody
- Includes pendulum rod, arm rod, rotary joints, and base
- Applies physical properties (mass, length, friction)
- Provides sensor outputs for position and velocity

7.2 LQR Controller (Simulink)

- Implements Linear Quadratic Regulator algorithm
- Stabilizes pendulum in upright position
- Computes optimal control signal based on state feedback
- Uses pre-computed gain matrix from MATLAB

7.3 Swing-up Controller (Simulink)

- Implements energy-based swing-up algorithm
- Controls pendulum motion from hanging to upright position
- Switches control strategy based on pendulum energy
- Coordinates with mode selector for seamless transition

7.4 Mode Selector (Simulink)

- Monitors pendulum state to determine control mode
- Switches between swing-up and regulation controllers
- Implements smooth transition logic
- Manages control signal blending during transitions

7.5 HIL Interface (Simulink)

- Interfaces with Arduino hardware via Simulink Support Package
- Reads encoder values from physical system
- Sends PWM control signals to motor
- Handles communication protocols and timing

7.6 Parameter Manager (MATLAB)

- Loads system parameters from JSON configuration file
- Makes parameters available to Simulink workspace
- Provides interface for parameter tuning
- Manages physical properties, controller gains, and simulation settings

7.7 GUI Application (MATLAB)

- Provides real-time visualization of system states
- Displays pendulum angles, angular velocities, and control signals
- Offers parameter adjustment interface
- Provides data logging and analysis capabilities

8. File Structure

```
Furuta_Pendulum/
├── config/
│   └── parameters.json
├── docs/
│   ├── requirements.md
│   └── architecture.md
├── models/
│   ├── Furuta_Pendulum.slx          # Main Simulink model
│   ├── Mechanical_Model.slx         # Simscape mechanical system
│   ├── Controller_Model.slx         # Control algorithms
│   └── HIL_Interface.slx           # Hardware interface
├── src/
│   ├── parse_config.m
│   ├── furuta_startup.m
│   └── create_furuta_gui.m
└── tests/
    └── test_controllers.slx
```

9. Implementation Approach

The system follows a Model-Based Design approach where:

1. Physical system is modeled in Simscape Multibody
2. Control algorithms are implemented in Simulink blocks
3. Parameters are managed via JSON configuration
4. Hardware interface uses Simulink Arduino Support Package
5. Visualization and high-level operations use MATLAB