CHAPTER 1

Test Platform

The test platform consists of the designated hardware, [*MinSeg M2V3 two-wheeled robot, see Section* **??**], and the designated development PC, [*see Section* **??**]. To interface with the hardware, a Simulink model and a hierarchy of Matlab subscripts were created.

The Simulink model is capable of:

- Acting as an algorithm with which to program the hardware, such that it may:
  - · Process
  - · Actuate
  - · Communicate
- Simulate an equivalent model of "the hardware when loaded with the same algorithm".

The Matlab script hierarchy is capable of:

- Initialize model parameters.
- Reconfigure model subsystems.
- Initialize a build or simulate event.
- Initialize a read or write event.
- Post-process raw read data.
- Save processed read data as well as other configuration data.
- Plot processed read data.

1.1 Simulink: minseg_M2V3_2017a.slx

minseg_M2V3_2017a.slx is the label of the Simulink model file. The label includes the label of the hardware which it represents as well as the version of the Mathworks Software Suite with which it was created. [*Using the model in a different version of the Simulink will require conversion; therefore, the two files will not be equivalent.*]

The Simulink model is hierarchical. The sections which follow will describe the model and will be similarly organized, as depicted in the extended-precision List of Contents below.

List of Contents

1.1.1 Root

The top level of the model, also known as the model root, is depicted in Figure 1.1.

The model root is contains the three primary components of the system:

- Plant
- Controller
- Board Inputs and Outputs

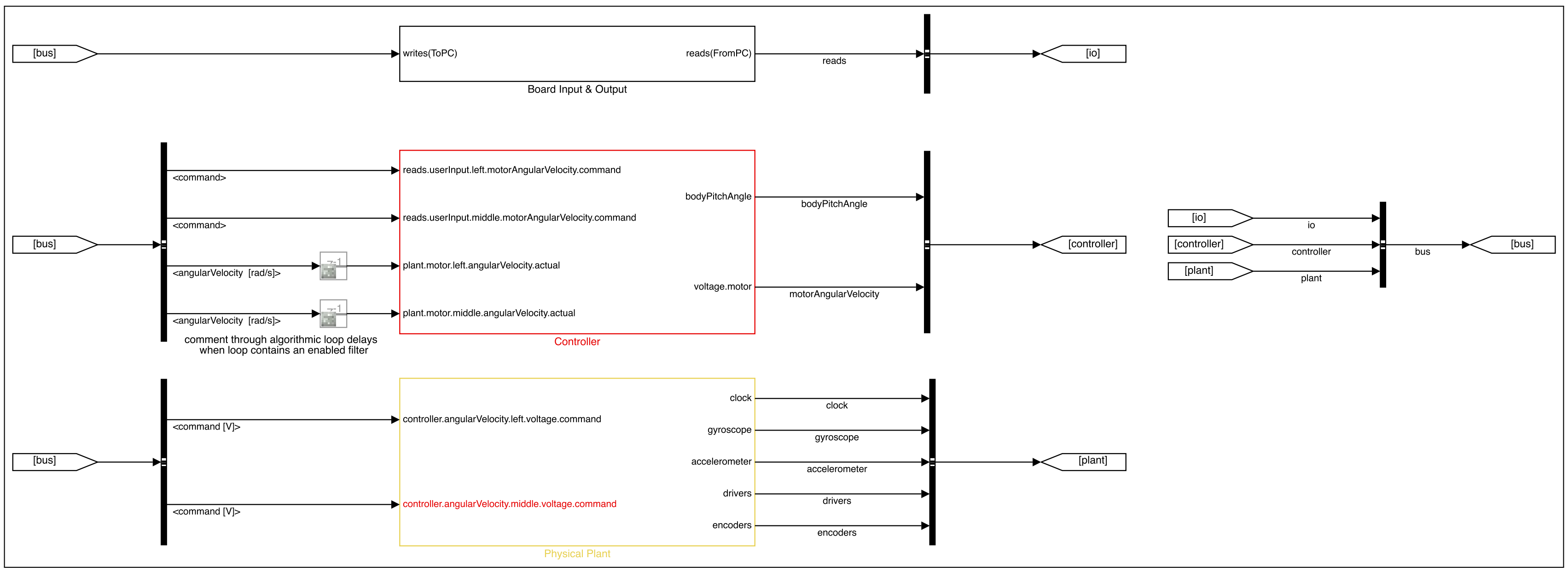

Figure 1.1: [Simulink]: Root

1.1.1.1 Bus Structures

Bus structures are a means of routing large quantities of signals. They are similar to muxed signals; however, it is not necessary to separate all of the signals during the demux process.

It is evident in Figure 1.1 that all of the components are passed into separate bus structures, [*black bars on the right-side of the figure*], and that those bus structures are in turn merged into one global bus structure.

This grants the user the ability to call any significant signal wherever it is needed using bus selectors, [*black bars on the left-side of the figure*]. The user should take care to implement a delay in the path of any signal which is implemented recursively [*as feedback*]. [*This prevents the formation of an algebraic loop*].

1.1.1.2 Variant Subsystems

A variant subsystem is a subsystem containing multiple subsystems, defined as variants. Only one variant can be active at one time. The variant subsystem serves as the switch between them. [*Note that the variant subsystem cannot switch between variants during operation/runtime*].

Several subsystems contained in this model are variant subsystems. These variant subsystems are used to switch system configurations. Examples of these variant configurations include:

- The plant:
  - Actual hardware drivers. [*Hardware implementation only.*]
  - Hardware-equivalent simulation model of nonlinear dynamics. [*Simulation only.*]
  - Hardware-equivalent simulation model of linear dynamics. [*Simulation only.*]
- The controller design:
  - PID. [*Primarily for initial hardware characterization.*]
  - Optimal.
  - Pole-placement.

1.1.2 Plant

The top level of the plant is a variant subsystem which is used to switch between plant configurations. It is depicted in Figure **??**.

The possible plant configurations are:

- Actual hardware drivers. [*Hardware implementation only.*]
- Hardware-equivalent model of nonlinear dynamics. [*Simulation only.*]
- Hardware-equivalent model of linear dynamics. [*Simulation only.*]

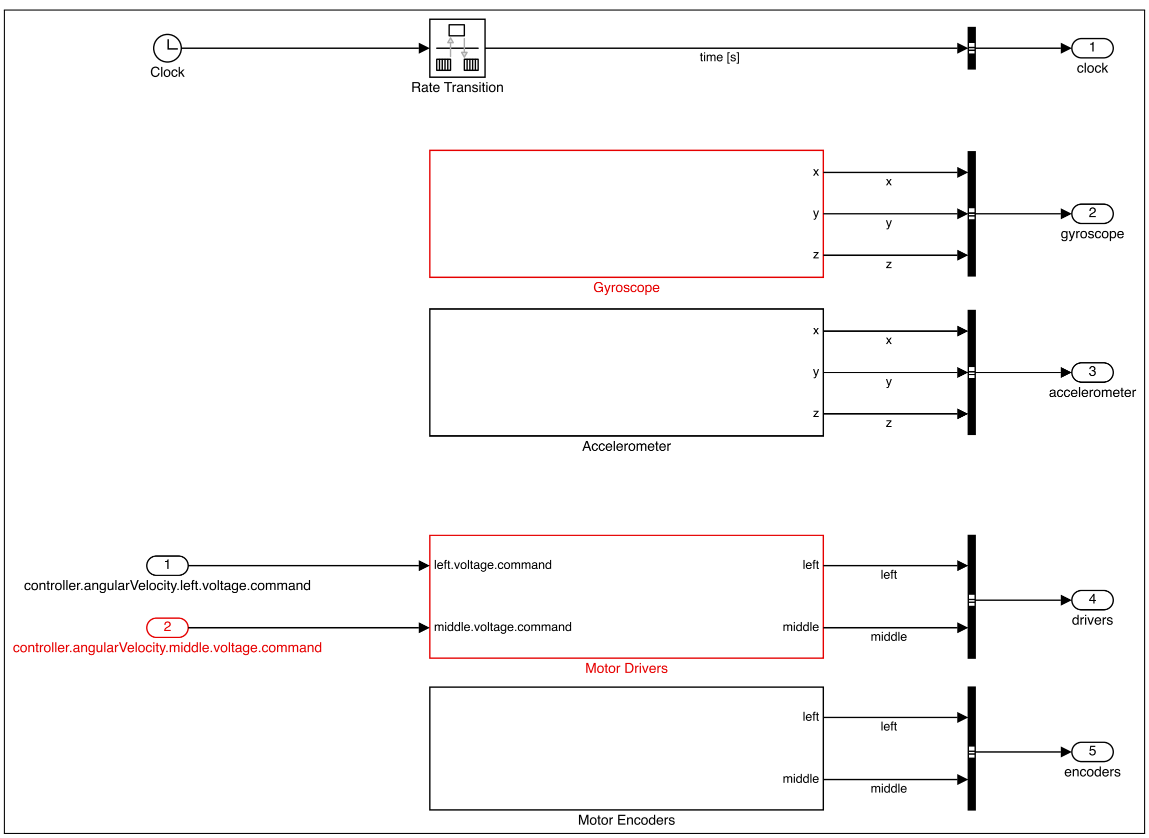

Figure 1.2: [minseg_M2V3_2017a]: Plant

1.1.2.1 Hardware

The *Hardware* variant of the plant contains subsystems for the various hardware components, [*not including the controller*]. It is depicted in Figure **??**.

The hardware components include:

- Clock
- Gyroscope
- Accelerometer
- Motor Drivers
- Motor Encoders

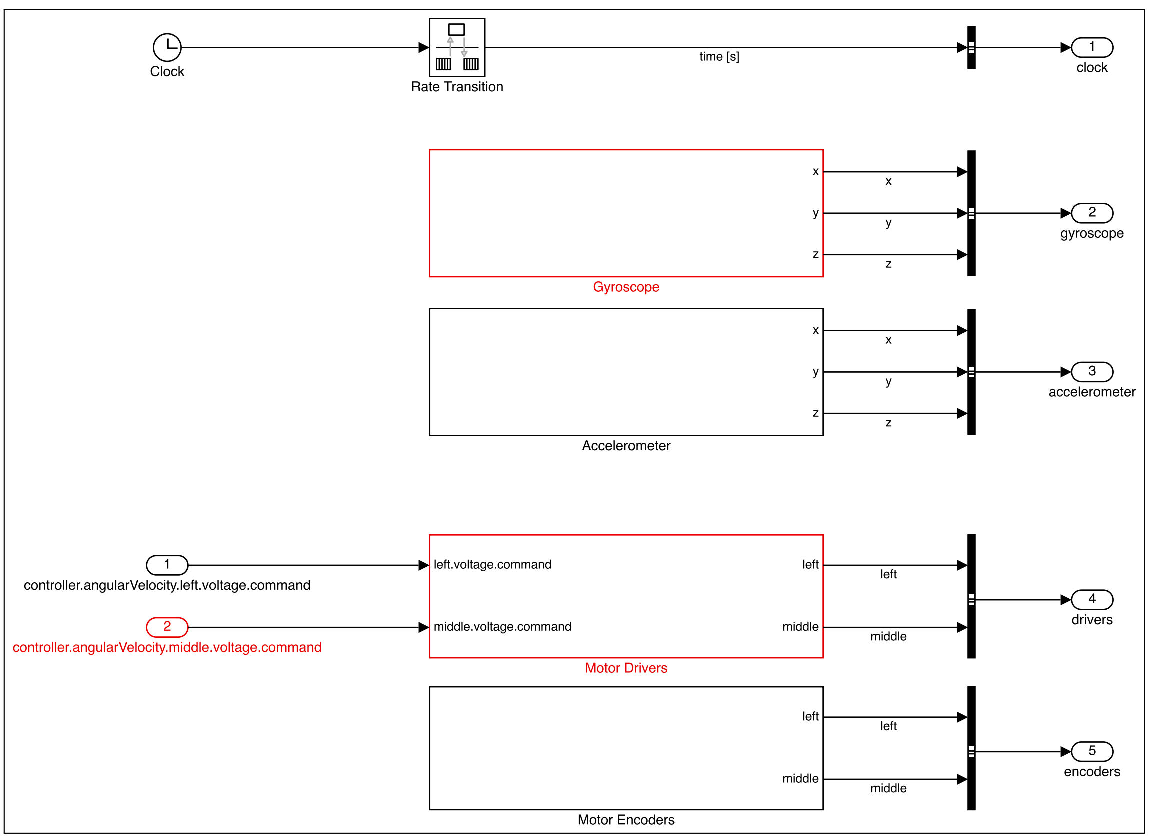

Figure 1.3: [minseg_M2V3_2017a:Plant]: Hardware

1.1.2.1.1 Motor Driver

1.1.2.1.2 Motor Encoder

1.1.2.1.3 Gyroscope

1.1.2.1.4 Accelerometer

The *Accelerometer* subsystem is depicted in Figure **??**.

It reads raw data from the accelerometer driver for each of three dimensional axes [x, y, z]. For each axis that data is reformatted into SI units.

The hardware components include:

- Clock
- Gyroscope
- Accelerometer
- Motor Drivers
- Motor Encoders

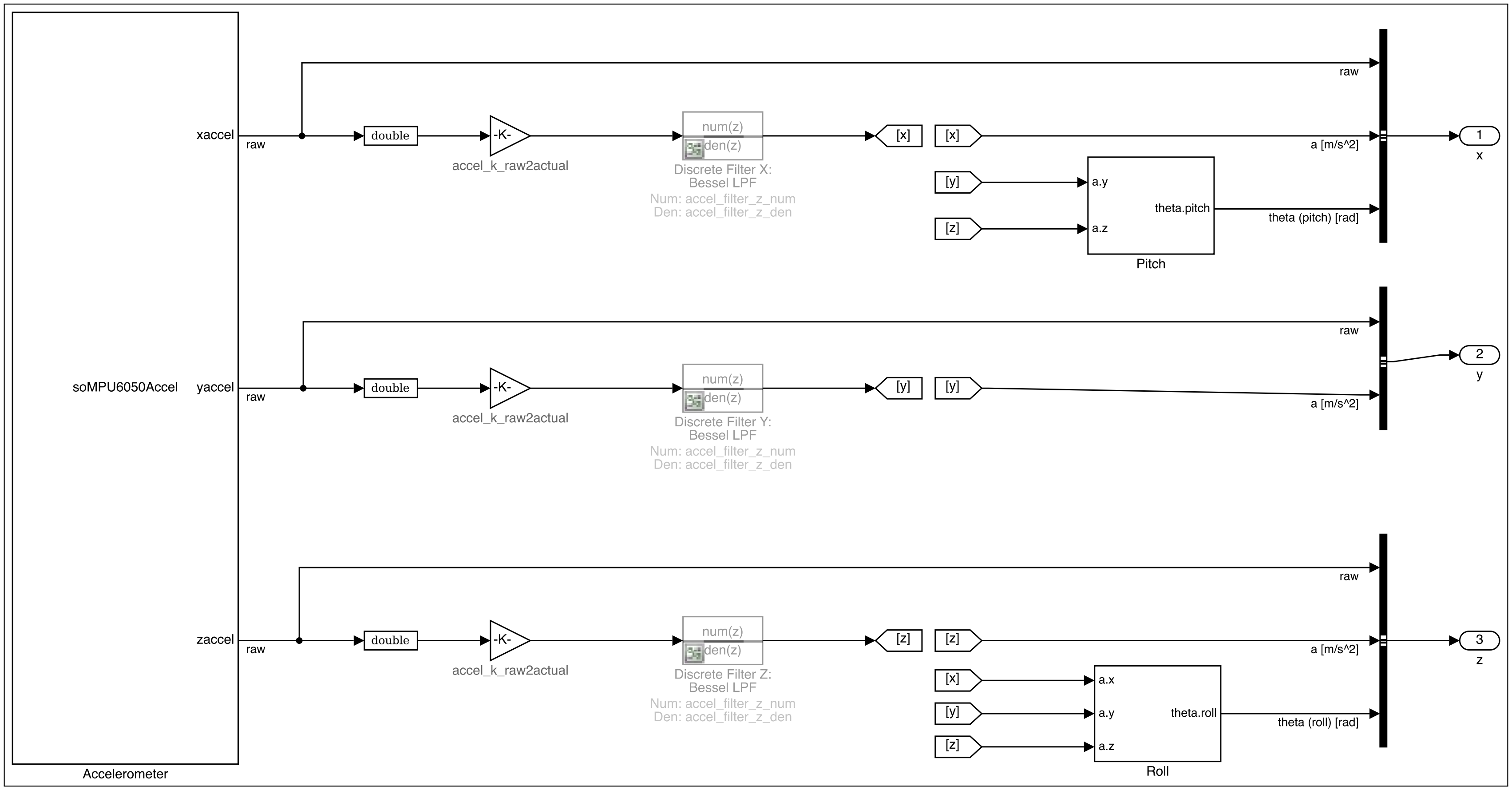

Figure 1.4: [minseg_M2V3_2017a:Plant:Hardware]: Accelerometer

1.1.2.2 Hardware-Equivalent Nonlinear Dynamics

1.1.2.3 Hardware-Equivalent Linear Dynamics

1.1.3 Controller

1.1.3.1 PID

1.1.3.2 Optimal

1.1.3.3 Pole Placement

1.1.4 Board Inputs and Outputs

1.2 Matlab: minseg.m

List of Contents

The *minseg.m* script was developed to control the MinSeg test platform. The script is capable of:

- Reconfiguring the model
- Running a model simulation
- Programming the model hardware
- Communicating with the model hardware
  - Optimizing the communication rate.
  - Reformatting the raw hardware-output data on receipt.
- Saving the initialization parameters and output data.
- Plotting the output data.

The script is hierarchal, and is therefore only the root *or master* file to a series of subfiles. The subfiles are broken up into principal segments of the scripting process:

- Global setup
- User-input
- Initialization
- Processing
- Output
- Global Cleanup

1.2.1 Global

1.2.2 User Inputs

1.2.3 Initialization

1.2.3.1 General

1.2.3.2 Model

1.2.3.2.1 General

1.2.3.2.2 Plant

1.2.3.2.2.1 Hardware

1.2.3.2.2.2 Nonlinear model

1.2.3.2.2.3 Linear model

1.2.3.2.3 Controller

1.2.3.2.4 Board Inputs and Outputs

1.2.3.2.5 Build Parameters

1.2.3.3 Serial

1.2.3.3.1 Write

1.2.3.3.2 Read

1.2.3.3.3 General

1.2.3.3.4 Reads

1.2.3.3.5 Build Parameters

1.2.4 Processing

1.2.4.1 Build

1.2.4.2 Serial Transmission

1.2.4.3 Serial Reads Post-Processing

1.2.5 Output

1.2.5.1 Save

1.2.5.2 Plot

1.2.6 Global Cleanup