ICE Lab Simulink Dashboard Manual

This document provides instructions, tips, and general information regarding the usage and development of the real-time data modeling dashboard for the Internal Combustion Engine Laboratory at the University of Oklahoma. Created Spring 2023.

Special thanks to Dr. Bin and TA's for their assistance and guidance this semester.

Usage

This section outlines the general usage of this tool.

Preparation

1. Open MATLAB 2021b on the desktop.
2. **In MATLAB** navigate to the directory containing the dashboard model ('.slx' file).
3. Double-click to open the dashboard model in Simulink.
4. Verify hardware connections.
5. Continue with engine startup and preparations.

Running Dashboard

1. In Simulink, navigate to the `Modeling` tab on the menu bar at the top of the window.
2. On the Toolstrip, verify that the `Mode` section is set to `Connected IO` showing the following image 

3. Verify that the `Stop Time` is set to `inf`.
4. Click on the `Run with IO` button.
   a. Note that this does not begin data collection.

Collecting Data

1. Verify that the model is running, hardware connections are correct, and the desired engine is running.
2. From the dropdown menu entitled `Engine Selection`, select the engine you wish to collect data for.
3. Click on the `Engine Start/Stop` button.
   a. See that the color of the check engine light transitions from red to green.
   b. Additionally, the dials, displays, and plots should begin to show real data.
4. When the round of data collection is complete, click again on the `Engine Start/Stop` button.
   a. See that the color of the check engine light transitions from green to red.
5. Repeat for desired number of data collection rounds.

Shutdown

1. Select the `Stop` button on the Toolstrip.
2. Close out of MATLAB.
   a. If prompted to save changes to the dashboard model, discard and do not save.
3. All output files will be found in the directory containing the dashboard model.
   a. Output files are in comma separated value format and contain headers describing the content of its respective column.

Development

This section contains information regarding the development of the dashboard and some tips, suggestions, and notes.

Project Management

MATLAB Version

It is strongly recommended that the development team's machines, lab machines, and any other
machines used in the process of developing the dashboard are running the same version of MATLAB, major and minor.
While possible to save models as a previous version, this complicates the process of collaboration
and simply downloading and installing the latest version for all machines would solve this headache.

Version Control

For the development of this project, the version control tool used was Git and for the remote repository, Github. This was not done in part to any major benefit or functionality Git and Github provided but rather out of familiarity by the development team.

RECOMMENDATION
It is strongly recommended to follow the guide
https://www.mathworks.com/help/simulink/ug/set-up-git-source-control.html
and specifically the optional section `Configure Git to use MATLAB for Diff and Merge`.
While not confirmed, this appears to outline using MATLAB for examining differences in binary files. Being able to compare the differences in '.slx' files would have been significantly impactful to our ease of development. A cursory look into the Git integration in MATLAB/Simulink showed that there are features integrated into the platform that offer the core functionality of Git such as: Fetch, Push, Pull, Commit as well as viewing branches, logs, and information about the repo.

This step was not followed by the development team and an issue occurred where a model
file was deleted by accident and merged onto the master branch. When working back through
commit history, no information regarding the '.slx' model files was available since they

were binary files and thus a couple of hours of work was lost due to not being able to verify the changes being merged.

MATLAB Project

When developing the dashboard, a single model file was used as the method of housing the tool. It is recommended to instead use a MATLAB Project, which can likely import the existing model. A MATLAB Project appears to offer again more functionality and tools for managing the project and all related materials to the project.

Subsystems Explained

Three subsystems exist in the current model. They are as follows:

Dashboard

This subsystem contains the components that will visually display the data from the post processing subsystem. Note that all components are linked to a signal or value that is found within the `Post-Processing` subsystem. This was a purposeful design choice such that the `Data Input` subsystem could be entirely abstracted from the work done in the `Post-Processing` subsystem so that the implementation of Group 2's data collection work could be done with the least chance of impacting our work.

No major workarounds or tricks were used in this subsystem however there are two features with some consideration required. First, the `Load` slider was implemented with the intention to simulate increasing loads and affects the simulated data. In the future the ability to modulate the actual load on the engines from the dashboard would be very convenient for data collection. This task would likely be significantly more difficult and potentially be worthy of a capstone style project or 4980 project should one be interested in this. Second, the `Units` input is currently tied to a value that has no effect on post-processing however the ability to change the units on the dashboard and in the output file would be nice. The ability to alter the units wihtin post processing is certainly doable by likely adding a filtering function to all the data before it is sent to the `File I/O` subsystem or displayed on the dashboard. The only problem for this is the labels on the plots and displays of the dashboard.

Data Input

This subsystem exists as a stand in for Group 2's data collection work. Currently the subsystem generates simulated data so that the dashboard and output files have something to demonstrate their functionality.

Post Processing

This subsystem houses all the logic and calculations which are applied to the input data in order to determine all the parameters that the dashboard will display. This subsystem takes input arguments of all the measured data and outputs the parameters
that are to be recorded in the output logs. No significant tricks or workarounds were needed for this subsystem as MATLAB Function blocks offered the functionality needed.

A smaller supplemental subsystem exists within the `Post Processing` subsystem.

File I/O

This subsystem exists within the post-processing subsystem. All output log generation, naming, and file creation operations take place in this subsystem. A key function block used for this subsystem was the `Triggered Subsystem` which takes a trigger input and runs once when this input satisfies the condition of the trigger. In our case the trigger was set as a 'Rising' trigger meaning it is activated when the input moves from 0 to 1, but when the input returns from 1 to 0 the trigger condition is not satisfied. This was key because
the triggered subsystem is used to generate a new output file every time the start/stop button is
pressed to begin data collection but not when data collection is stopped.

Another aspect of this subsystem that took considerable workarounds to achieve the desired functionality was the output file name generation. The team was unable to implement a feature that would take a string input and pass that as a filename to the file writing function. The solution to this was to use as standardized file format using date and time as the indicator of uniqueness. MATLAB functions to receivee date and time were used and Simulink string functions were used to interpret these as strings and compose a string containing the whole filename. The filename was then encoded as ASCII and passed into the file writing function. This was done because Simulink does not allow strings to be passed as signals to MATLAB Function blocks because
of some complicated issue about string length. From within the function, the ASCII encoded
string was decoded and the outputs of the post-processing subsystem are written to the file.

GUI Choices

Within MATLAB/Simulink there are different methods of developing a GUI such as this dashboard. The choice of using 'dashboard' components wihtin a Simulink model was not necessarily the best or worst choice but was what the development team determined to be the most effective use of our resources. Other methods certainly exist and should be explored as a method of refining the current dashboard.

Editing This Manual

This manual was written in markdown and any basic text editor such as VSCode, Notepad++, Vim, or Obsidian can be used to edit markdown files ('.md') and have some syntax highlighting or markdown support of some variety. Obsidian is a markdown editor/viewer application and has a handy export to PDF feature which was used to generate this manual. The markdown file of this report should be included in any further development repositories so that future groups can reference or edit the document.

Further Information

Included in this directory is a document entitled 'DashboardReport.doxc' which is a report generated by Simulink and includes an excruciating amount of detail about the design. The level of detail the generated report covers is likely unnecesary and I sincerely hope that you never need to dig into this document however should you need to it exists.