THESIS

SYSTEMS ENGINEERING APPROACH TO ENGINE TEST STAND DEVELOPMENT FOR MICROPATCHING EVALUATIONS

Submitted by

Peter Eliot Lobato

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2022

Master's Committee:

Advisor: Thomas Bradley
Co-Advisor: Jeremy Daily

Bret Windom

Copyright by Peter Eliot Lobato 2022

All Rights Reserved

ABSTRACT

SYSTEMS ENGINEERING APPROACH TO ENGINE TEST STAND DEVELOPMENT FOR MICROPATCHING EVALUATIONS

This project applies systems engineering methodology to develop an engine test stand used to extract, patch and validate the binary file of a diesel engine electronic control module. Electronic control modules operate modern systems ranging from aircraft and spacecraft to automobiles, heavy trucks and industrial equipment. These systems are often used for decades, which may be beyond the period for which manufacturers provide support. The binary code operating these embedded controllers may need to be patched as part of maintenance or compatibility with updated requirements.

The objective of this thesis is to design an evaluation system to test the extraction, patching and deployment of binary code operating an engine control module of a legacy engine platform, a Cummins 6.7L diesel engine with a Cummins CM2350 engine controller, which does not have source code available. However, through binary analysis and micropatching, it is possible to update the binary of the ECM firmware by applying a patch to change specific attributes of the operation of the ECU. To verify the results of the patch, the binary is deployed to the engine controller and the operation of the engine is assessed.

An engine on a dynamometer test stand was reconfigured to be an evaluation platform for assuring non-interference attributes of the ECM binary. Requirements were identified, architecture was established, and validation was tied to corresponding test stand requirements. A method to solve an iterative numerical calculation with convergence criterion set incorrectly was

implemented on the ECM, and that method was then patched with a correct convergence criterion. The evaluation system was documented for other operators to execute the evaluations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

AIS …… Assured Information Security, Inc.
AMP ……..Assured Micropatching
APP ……... Accelerator Pedal Position
CAN ……..Controller Area Network
DDEC ……Detroit Diesel Electronic Control
DEF ……...Diesel Exhaust Fluid
DOC ……..Diesel Oxidation Catalyst
DPF …….. Diesel Particulate Filter
DTW ……. Dynamic Time Warping
ECM …… Electronic Control Module
EEPROM …….. Electrically Erasable Programmable Read-Only Memory
EPA ……...Environmental Protection Agency
HIL ……Hardware-in-the-Loop
HVEDR …….Heavy Vehicle Event Data Recorder
JTAG …….Joint Test Action Group
NASA ……National Aeronautics and Space Administration
NOx ……...Oxides of Nitrogen (NO + NO_2)
PGN …….. Parameter Group Number
SCR ……...Selective Catalytic Reduction

1.0 INTRODUCTION

Embedded controllers are employed in myriad applications ranging from on-road vehicles, heavy equipment and generators to aircraft, industrial assembly lines, ships and spacecraft. It is often the case that these systems continue to be used past their planned life spans and beyond the period for which the manufacturer provides support. These systems are typically major investments, and it is generally not financially attractive, or sometimes not feasible, to replace the entire system only due to an outdated controller or stale source code. In these cases, the code operating in these embedded controllers may need to be patched as part of maintenance or compatibility with newer systems. The primary method for patching these systems is to utilize existing source code and build chains. However, these resources may not be available, so the patch may need to be developed and deployed directly to the executable binary on the microprocessor.

This work focuses specifically on the embedded controllers used to operate diesel engines. Many aspects of heavy-duty vehicles are controlled by electronic control modules (ECMs) such as the engine, exhaust aftertreatment, transmission and brakes. It is possible that source code for older in-use ECMs is no longer available due to suppliers going out of business, platforms being retired, staff turnover, or a variety of other reasons. Micropatching technologies enable some recovery of source code from binary analysis. The objective of this project was to develop evaluation techniques to test the effects of deploying a micropatch in an engine's binary code by operating an engine in a test environment. Utilizing a commercially available engine controller creates a look-a-like scenario to model other proprietary or classified systems that need to be micropatched.

A micropatch is a set of minimally changed binary codes that affect the desired change in the controller. These micropatches are often applied to fix security vulnerabilities, update input parser engines, change convergence criteria, alter parameters, or bypass security checks. The number of bytes changed are on the order of tens of bytes out of a possible megabyte level binary.

1.1 Contributions

In this thesis, the following contributions to advancing the state of the art are described:

1. A technique to perform on-engine testing of binaries modified outside the manufacturer's build chain is developed. This capability has not existed in the public domain to date.
2. Details of realistic and repeatable dynamometer testing to emulate hard braking events in a truck on a test track.
3. An issue with the convergence criterion of an iterative numerical calculation identified on a spacecraft's embedded controller was replicated on an engine's embedded controller and resulting engine behavior was evaluated.

The remainder of chapter 1 describes the background and motivation for the project, and then the general approach to concept development is discussed. Finally, the first chapter finishes by establishing the details of technical requirements.

Chapter 2 discusses the system's architecture using systems modeling language, and then describes hardware setup and configuration. Data acquisition, controls and measurement calibrations are then described, and chapter 2 concludes with verifying requirements.

Chapter 3 describes validating the core requirement of implementing a use case on the system. A patching scenario is identified and discussed and then implemented on the system. The engine test stand is then used to evaluate the effects of the patching scenario.

Chapter 4 discusses systems engineering methodology to tie requirements to verification and validation steps. Chapter 4 concludes by discussing how this work fits into a larger project and the documentation produced for future operators.

Chapters 5 and 6 summarize and conclude this scope of work and detail concepts for future research identified throughout the course of the project.

1.2 Background & Motivation

An ECM is "a specialized process control computer" with "central processing units, memory, storage, and a means of networking or communicating with external devices." [1]. ECMs are part of many components of both light and heavy vehicles including cabin controls, transmission, stability control, infotainment, brake and engine. The ECM serves as a real-time controller which accepts sensor inputs, network communications, and outputs actuator commands. The ECM contains a so-called calibration which determines what the outputs should be based on the current state and sensor input. This calibration is stored as a binary file in the computer's memory along with real-time sensor and model data. These data are often not available on the ECM's networking ports, such as CAN bus, but is stored on data bearing chips like EEPROM, flash memory and internal microprocessor program storage [2]. Techniques for reading, modifying and writing these binary data enable the extraction, modification, deployment and evaluation of the binary images in which issues or vulnerabilities are discovered and patched.

Methods have been developed to extract binary data. Daily et al. [3] showed a method to extract binary data from a Detroit Diesel DDEC V engine control module. The objective of Daily's study was to make sense of binary data stored in a damaged ECM in a forensics context in which the conventional in-vehicle networking connection may not be operable. The study went on to show that binary data extracted directly from the ECM's memory had higher fidelity for certain parameters than the manufacturer's standard report, which is known as a DDEC Report for Detroit Diesel engines. This method was destructive since it involved removing chips from the engine controller circuit board.

Van et al. [2] developed similar techniques for Cummins CM870, CM871 and CM2350 series ECMs containing records from the heavy vehicle event data recorder. In this study, binary data were accessed through the ECM circuit board's Joint Test Action Group (JTAG) in-system programming port, which was non-destructive. The commercially available tools AlienTech's K-TAG and PEmicro's Cyclone were used to extract the binary data. Similar methods are used in the present study to extract and write binary data from an operational engine's ECM for project stakeholders to understand and patch.

As an example of the need to patch the ECMs of legacy equipment, one potential purpose is to fix newly discovered cybersecurity vulnerabilities. The ISO/SAE Surface Vehicle Standard 21434 [4] first issued in 2021 addresses cybersecurity in road vehicles, and sets forth organizational-level best practices for vehicle cybersecurity such as implementing a "defense-in-depth" approach, but the standard provides few specific technical requirements. However, many vehicles on the road were designed and built before such standards or processes existed.

This work partially supports the DARPA funded Assured Micro-Patching (AMP) project under the direction of Dr. Sergey Bratus, which aims "to create the capability for rapid patching of

legacy binaries in mission critical systems, including the cases where the original source code version and/or build process is not available." [5] This project focuses on developing challenge problems and evaluation systems to help drive solutions developed by the different teams in the DARPA AMP program portfolio.

The AMP project timeline is organized into three phases, the latter two of which present performers with challenges based on production automotive hardware in a hardware-in-the-loop (HIL) environment. Rather than providing copies of hardware to work with, a working test stand must be constructed with the ability for performers to utilize for testing. The development of the Phase II & III test stand is the focus of this thesis.

1.3 Scope

This work applies the systems engineering methodology to develop an engine-based test stand used to evaluate binary code patches provided by the collective efforts of performers for the DARPA AMP program. Figure 1 references the systems engineering "Vee" diagram [6][7], which was followed in developing the test stand. The process starts with concept development which reflects the motivation and deliverables of the project. An engine and dynamometer are set up and validated, and the above-mentioned methods are employed to extract and write binary data to the engine's ECM. It ends with transition/operations/support by transitioning the test stand for other project staff to operate and provides written documentation on operating and maintaining the test stand. Each step of the Vee diagram will be described and discussed throughout this document.

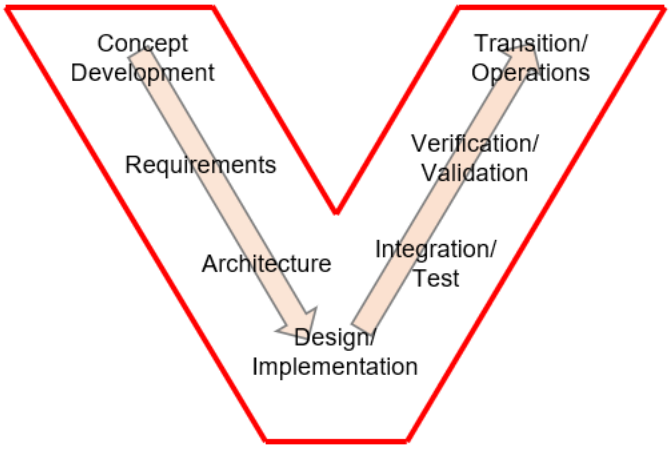


Figure 1. Systems Engineering Vee Diagram [6]

1.4 Concept Development

Outside the scope of this thesis, various challenges will be developed and presented to various performers that create an issue to be patched in an ECM's binary. The performer must patch it, and then that patch is evaluated. After the binary file of an embedded controller on a heavy vehicle has been modified, the functionality of the vehicle must be evaluated: the engine must provide the same functions in its stock configuration as in a patched configuration for the patch to be considered effective in addition to eliminating the vulnerability. Partnering with Cummins, a QSB6.7 installed on an engine dynamometer test stand at the Colorado State University (CSU) Powerhouse and a Kenworth T270 equipped with a PACCAR PX-7 (on-highway version of the Cummins QSB6.7) were used as starting point. The overarching goal of this project was to set up the dynamometer engine test stand such that the engine's ECM binary file could be extracted, patched, re-installed on the ECM, and then the effectiveness of the patch could be evaluated against a set of success criteria based on the ability (or inability) to operate the engine.

1.5 Requirements

The overarching objective of this project is to develop an engine test stand for evaluating patches to the ECM. This section describes the requirements of the test stand. Table 1 summarizes requirements, key performance indicators (KPIs), and remarks. These requirements are described in the subsequent sections.

Table 1. Requirements

	Requirement	**KPI**	**Remarks**
R1	Manage ECM Binaries		
R1.1	Extract ECM Binary	Y/N	
R1.2	Re-Load ECM Binary	Y/N	
R2	Nominal Engine Operation		
R2.1	Highway APP Control	-	
R2.2	Generate rated power	224 kW @ 2500 RPM ± 5%	tolerance for altitude, fuel type, preconditioning
R2.3	Generate rated torque	1044 Nm @ 1500 RPM ± 5%	tolerance for altitude, fuel type, preconditioning
R2.4	Operate at rated power & torque with no fault codes	-	
R3	Protect engine & test equipment in case of faulty ECM patch	-	
R4	Simulate Hard Braking		
R4.1	Engine Ramp Rate	-715 RPM/s: > 1000 RPM -49 RPM/s: < 1000 RPM	
R4.2	Engine Derate	Y/N	
R4.3	Engine Derate message format	-	
R5	Data Logging		
R5.1	Log all CAN traffic	-	Later used for DTW work
R5.2	Log dyno speed and torque	-	From Labview controller
R6	Detect effects of a patch		

R6.1	Show expected behavior of unpatched calibration	-	Certain undesirable behavior is identified
R6.2	Show engine runs nominally after being patched	-	Identified behavior no longer present

1.5.1 Manage ECM Binaries

The system must be able to extract and re-flash the ECM's binary file. Starting with the engine in a nominal and consistent condition, the binary must be extracted from the ECM. It will then be modified by a different group, the "patching" process, and loaded back onto the ECM for evaluation. As described in the Background, methods have been developed to non-destructively extract the binary file from an engine's ECM [3].

1.5.2 Nominal Engine Operation

The engine must operate normally as if it were in a vehicle. Evaluating the ECM binary should not be confounded with other test stand-related problems such as a derate related to a missing vehicle sensor. Much of this activity is similar to engine tethering projects in the automotive industry [8], [9], and techniques were drawn from it. Tethering involves removing an engine from a vehicle, installing it on an engine dynamometer, and sensors, actuators, and data busses are either connected to a vehicle parked outside the test cell or faked to gain full control of the engine without setting any fault codes from the rest of the vehicle not being present. For example, a false wheel speed signal may be generated so the ECM thinks the engine is driving the vehicle normally and prevents de-rating from the vehicle's stability control.

The engine must be able to produce rated power and torque to within 5% without setting any fault codes. The engine's power and torque rating were not particularly important for the objectives of this project, but it is a strong indicator of underlying problems with the test stand that may confound ECM patch evaluations.

Figure 2 shows the engine's nameplate with its power rating. Figure 3 shows a screenshot of the Cummins factory scan tool *INSITE* connected to the engine's stock ECM with its torque rating.

Figure 2. Engine Nameplate with Power Rating

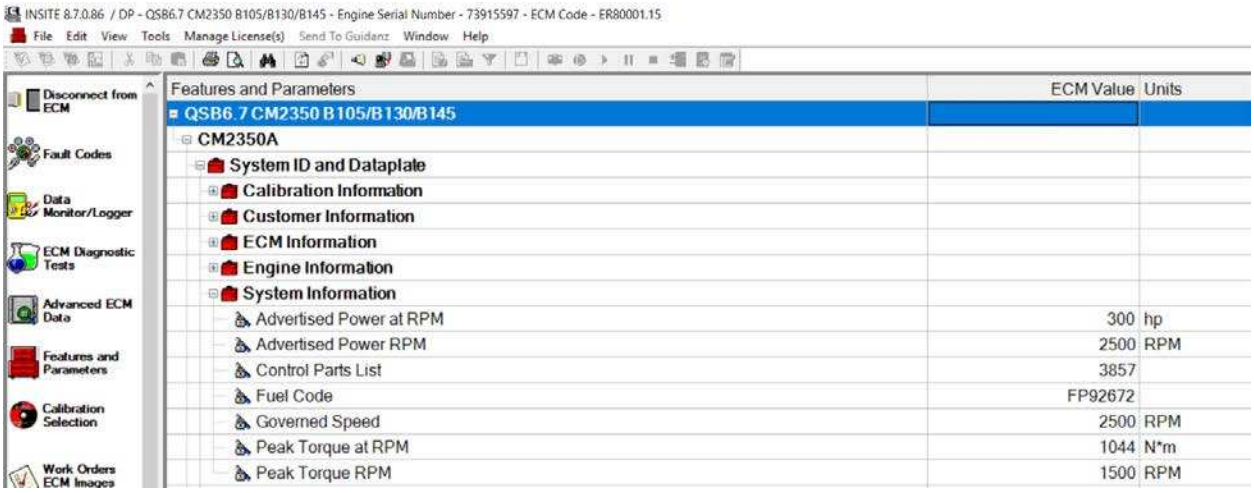


Figure 3. A screenshot from Cummins INSITE maintenance software showing the rated power & torque

1.5.3 Protect Engine and Test Equipment

It is expected that the ECM binary files being evaluated will have errors, either accidental or purposeful. Some of these errors may cause erratic or unknown behavior of the engine. The test stand must be able to protect the engine and test stand hardware from any abnormal or erroneous behavior by the ECM. An example of this is a faulty setting for the engine's peak governed speed, the so-called redline, causing the engine to accelerate past safe engine speed limits of its hardware.

1.5.4 Simulate Hard Brake Event

Some ECM patches will be evaluated by using a drive cycle which simulates a hard brake event in a truck. The specifications of this hard brake event must be defined.

Figure 4 shows a simplified internal block diagram of truck components during a hard braking event. During braking in a heavy truck, the driver presses the brake pedal and air pressure in the brake system is reduced which allows springs to engage the brakes. The truck's stability control system senses air pressure as a measure of brake command. For a high brake command, the stability control system derates the engine to assist with slowing the vehicle. To do this, the stability control system sends a message to the engine controller which overrides APP and sets commanded engine load to zero. This message calls PGN0 as defined in the SAE J1939 communication protocol for torque or speed limiting [10].

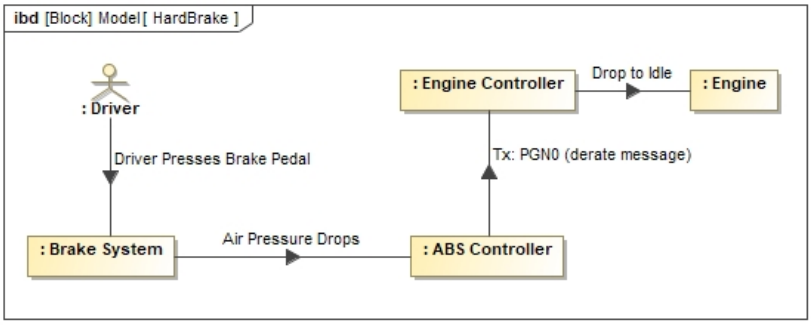


Figure 4. Hard Brake Event Block Diagram

To determine the data required in the message, a hard brake maneuver was done with the Kenworth T270 from roughly 50 mph to a full stop at Christman airfield [11]. The Kenworth T270 under study is shown in Figure 5.

Figure 5. Kenworth Conducting Hard Brake Maneuver at Christman Airfield

During this maneuver, CAN traffic was logged from the vehicle's J1939 diagnostic port [12]. Figure 6 shows recorded vehicle speed and the torque limit set by the brake controller through the PGN0 message. The hard brake maneuver was repeated twice and identified in Figure 6 as T1 and T2. When brakes are engaged, the torque limit ramped down with engine speed until idle.

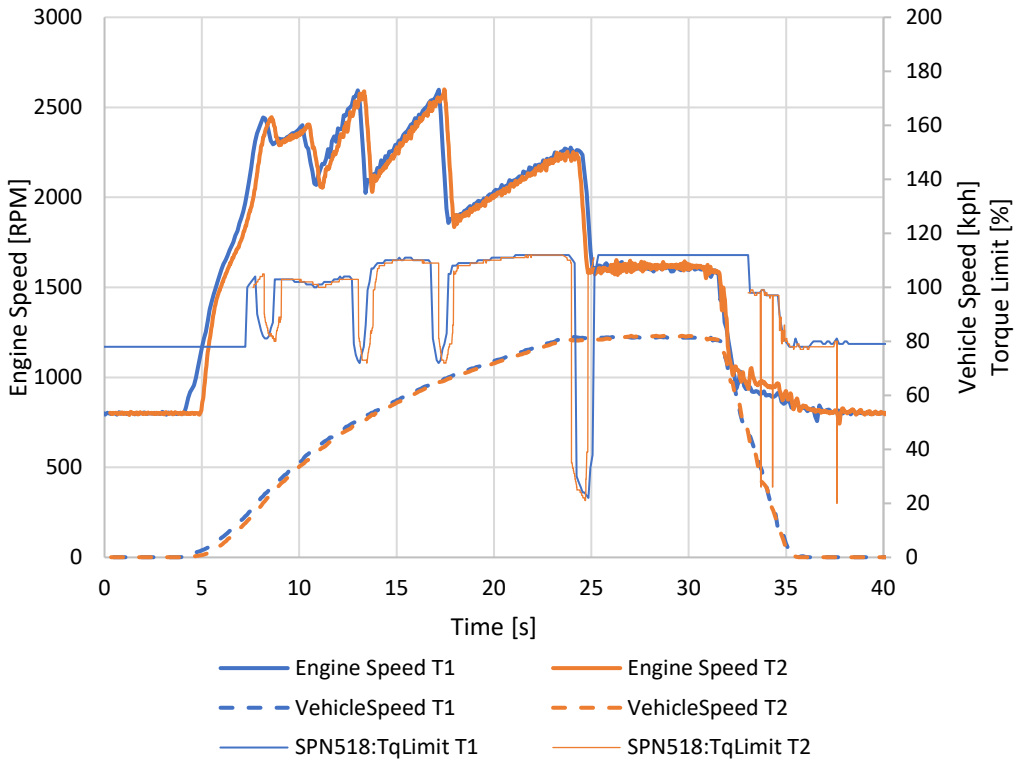


Figure 6. Kenworth T270 Hard Brake

To implement this on an engine, the torque limit was simply set to zero. All other necessary values were recorded from the truck's CAN bus to simulate on an engine test stand. Table 2 shows the message payload as recorded from the Kenworth T270 performing a hard brake event, and Table 3 shows the data as described in J1939. Therefore, the system requirement is for the engine to drop to idle when the message defined in Table 2 is sent to its J1939 diagnostic port.

Table 2. Engine Derate Message Payload

ArbID	B1	B2	B3	B4	B5	B6	B7	B8
0C000003	EB	AB	F9	7D	FF	F5	FF	FF

Table 3. Engine Derate Message Description

Signal	Value	Description
SPN695: Engine Override Control Mode	0x3	Speed/torque limit control
SPN696: Engine Request Control Conditions	0x2	Stability opt driveline engaged lockup 1
SPN897: Override Control Mode Priority	0x2	Medium Priority
SPN898: Engine Request Speed / Speed Limit	0xABF9	5503.1 (off)
SPN518: Engine Request Torque / Torque Limit	0x7D	0%
SPN3349: TSC1 Tx Rate	0x7	Use standard 10ms
SPN3350: TSC1 Control Purpose	0x1F	Temporary powertrain control
SPN4191: Engine Request Torque (fractional)	0x5	0.6%

The initial conditions of the hard brake event were defined as a Class 6 truck at a steady-state 50 mph cruise. While road load coefficients for the CSU-owned Class 6 truck are not available, they are available for a similar truck in a study published by NREL [13] and are listed in Table 4. (Unlike light-duty vehicles in which vehicle emissions are certified and road-load coefficients are determined and published publicly for chassis dynamometer emissions testing, heavy-duty and some medium-duty vehicles undergo engine certification and emissions tested with an engine dynamometer on a g/kW-hr basis, so determining vehicle road load is not necessary for certification and therefore not widely available.) The truck's gear ratios are available from Allison [14], its axle ratio is posted in the truck's cabin, Figure 7, and rear tire size is available from Michelin [15]. Using the data in Table 4, assuming a driveline efficiency of 90 %, and assuming the transmission is in 6^{th} gear at a 50 mph steady-state cruise, engine power will be **53 kW** at **1382 RPM**. This condition was replicated on the engine dynamometer for the hard braking event.

Table 4. Class 6 Truck Road Load and Driveline Specifications

A [N]	577
B [N/kph]	0
C [N/kph^2]	0.242
1st Gear Ratio	3.10
2nd Gear Ratio	1.81
3rd Gear Ratio	1.41
4th Gear Ratio	1.00
5th Gear Ratio	0.71
6th Gear Ratio	0.61
Axle Ratio	5.29
Tire Size [rev/mi]	514

Figure 7. Kenworth Driveline Specifications

Engine ramp rate during the hard braking event was calculated from recorded engine speed. The hard brake from 50 mph to a full stop was repeated twice, and Figure 8 shows its consistency. Gear shifts could be seen, but the data did not have the fidelity to clearly show when and how long gear shifts occurred.

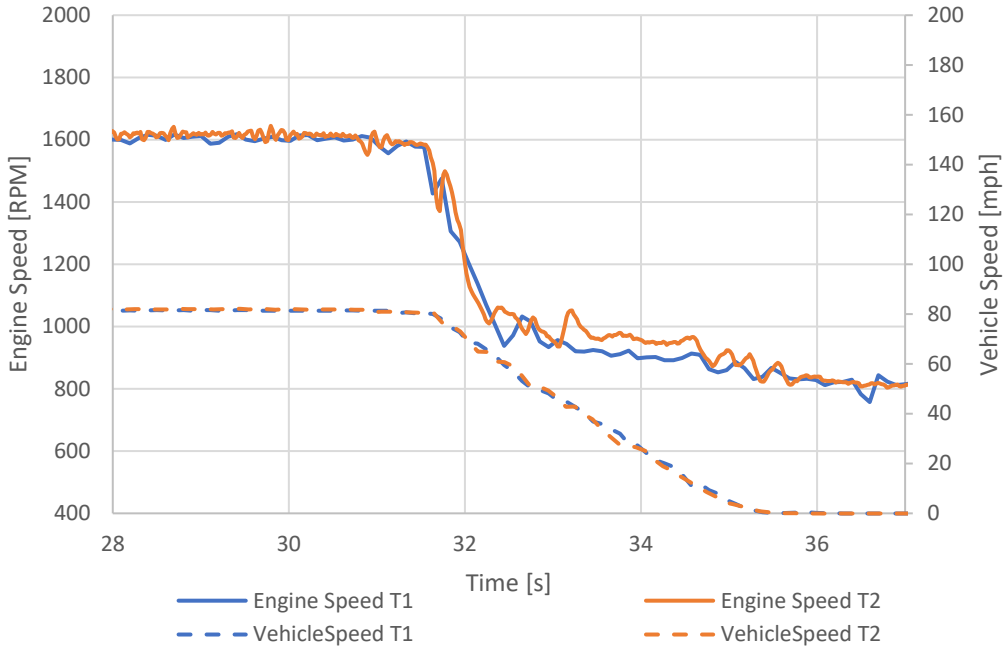


Figure 8. Engine & Vehicle Speed During Hard Brake

An engine ramp rate was modeled as a piecewise linear function. The dynamometer controller is limited in its ability since it can only accept linear engine speed ramps, rather than curve or a time-based lookup table. However, Figure 8 shows that the engine ramp rate from 1600 RPM to 1000 RPM is significantly different than 1000 RPM to idle. Therefore, a piecewise linear function was used to fit the data.

Figure 9 shows where the engine ramp was split and each ramp rate based on the average of the two repeated runs. Engine ramp rate should be -715 RPM/s between 1600 RPM and 1000 RPM and -49 RPM/s between 1000 RPM and idle.

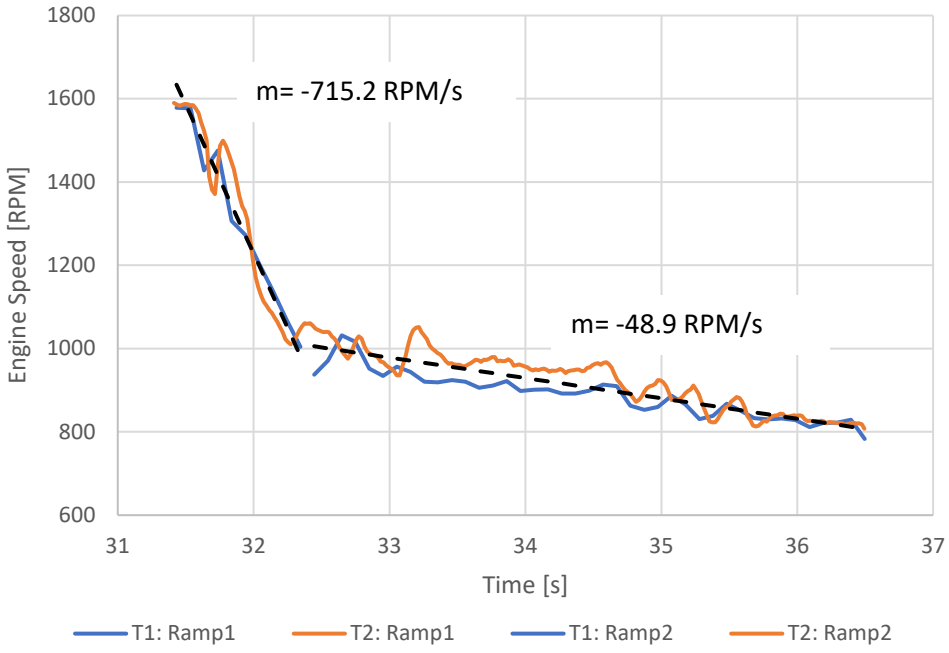


Figure 9. Piecewise Engine Ramp Rate

1.5.5 Data Logging

This system must be able to log several forms of data during operation. The Cummins QSB6.7 engine used in this test stand has two CAN busses: one is for J1939 diagnostics, and one is for exhaust aftertreatment (DOC, SCR, DEF tank, NO_x sensors) and turbocharger wastegate control. Traffic on both busses must be logged reliably without any dropped frames.

Reliable CAN traffic is a deliverable for the test and evaluation of the applied micropatching. As part of the Transition and Support stage of this project, CAN traffic will be used for further analysis to explore dynamic time warping methods as described in the Future Works section.

1.5.6 Detect Effects of a Patch

In addition to loading ECM binaries, the systems must be able to detect the effects of a patch. There are two parts to this requirement:

(1) When a specified behavior is identified, the system must be able to show that specified behavior. This may be a challenge scenario in which an effect is deliberately introduced to the ECM's calibration, and performers must find ways to patch or mitigate that behavior. This effect may be something undesirable in engine operation or another research scenario. When the engine is operated, it must reflect that intended behavior.

(2) The system must be able to show nominal behavior after being "patched." Performers will modify the ECM binary with the deliberately introduced effect. The engine will the operate with that patched ECM binary, and it must not display the previous behavior.

2.0 EXPERIMENTAL SECTION

This section describes the overall architecture and implementation of the test stand. Systems Modeling Language (SysML) diagrams are used to model the system and identify interfaces, hardware assembly is documented, and initial verification tests are described.

2.1 Architecture

Figure 10 and Figure 11 model the overall architecture of the test stand. Figure 10 shows the block definition diagram of the test stand along with critical values of each component. These diagrams help identify critical interfaces and may help with future troubleshooting.

Figure 10 decomposes parts of the test stand system that influence the use cases of the system or those which must interact with end users. The two sub-systems that must be decomposed are the engine and dynamometer (dyno). The engine is decomposed into aftertreatment, because of its ability to set fault codes, and the ECM, with which the test stand's LabVIEW controller and the end user interfaces. The dynamometer is decomposed into its load cell and controller. The components with which performer's ECM patches are evaluated are modeled as the single block *Interface*.

Figure 11 shows the internal block diagram of the test stand. An internal block diagram is applicable to this system because it defines interaction between components and where the test stand interfaces with outside users.

The engine interfaces with its ECM through sensor signals and actuator commands. These go through ports J1 and J2 on the ECM. The ECM also interacts with the engine's aftertreatment

system through port J1. The exhaust aftertreatment sub-system is decomposed because of its ability to set fault codes and cause unwanted engine behavior.

The engine is connected to the dynamometer with a driveshaft. The eddy-current dynamometer is cooled by plant water through a series of pumps that are turned on and off through the LabVIEW controller. The dynamometer is controlled with a DYN-LOC IV controller which applies current to the dynamometer's coils and controls torque or speed. Torque feedback is provided by a load cell, and speed feedback is provided by a speed pickup.

The LabVIEW interface exchanges feedback and setpoints with the DYN-LOC IV controller over an ethernet interface. The LabVIEW interface also sends dynamometer speed and torque values to the engine's diagnostic bus for datalogging.

Figure 11 also defines where the system interfaces with its user at the system boundary. The user interfaces with the ECM memory through a JTAG port, and the end user's data acquisition can interface with busses CAN1 and CAN2.

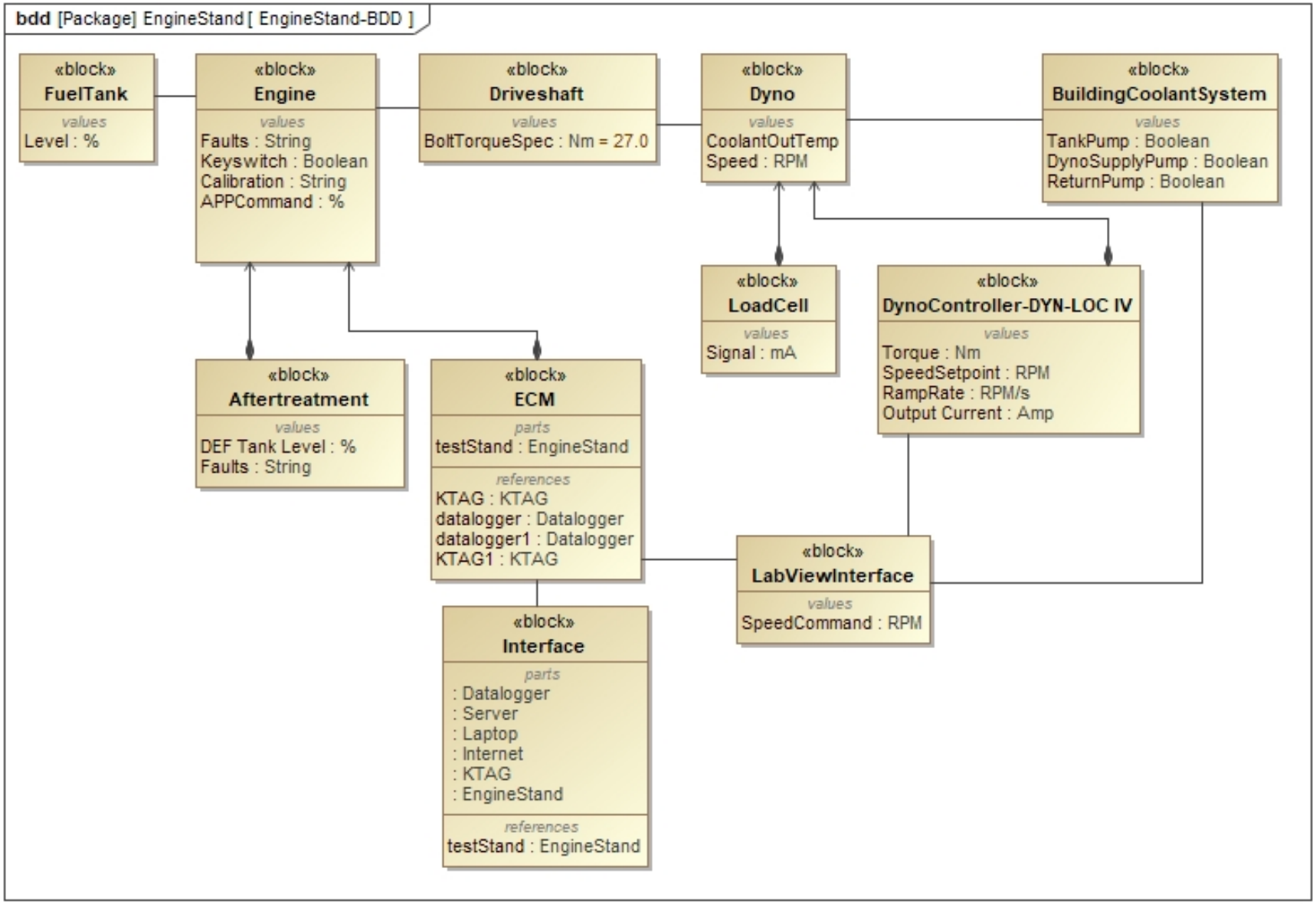


Figure 10. Engine Test Stand Block Definition Diagram

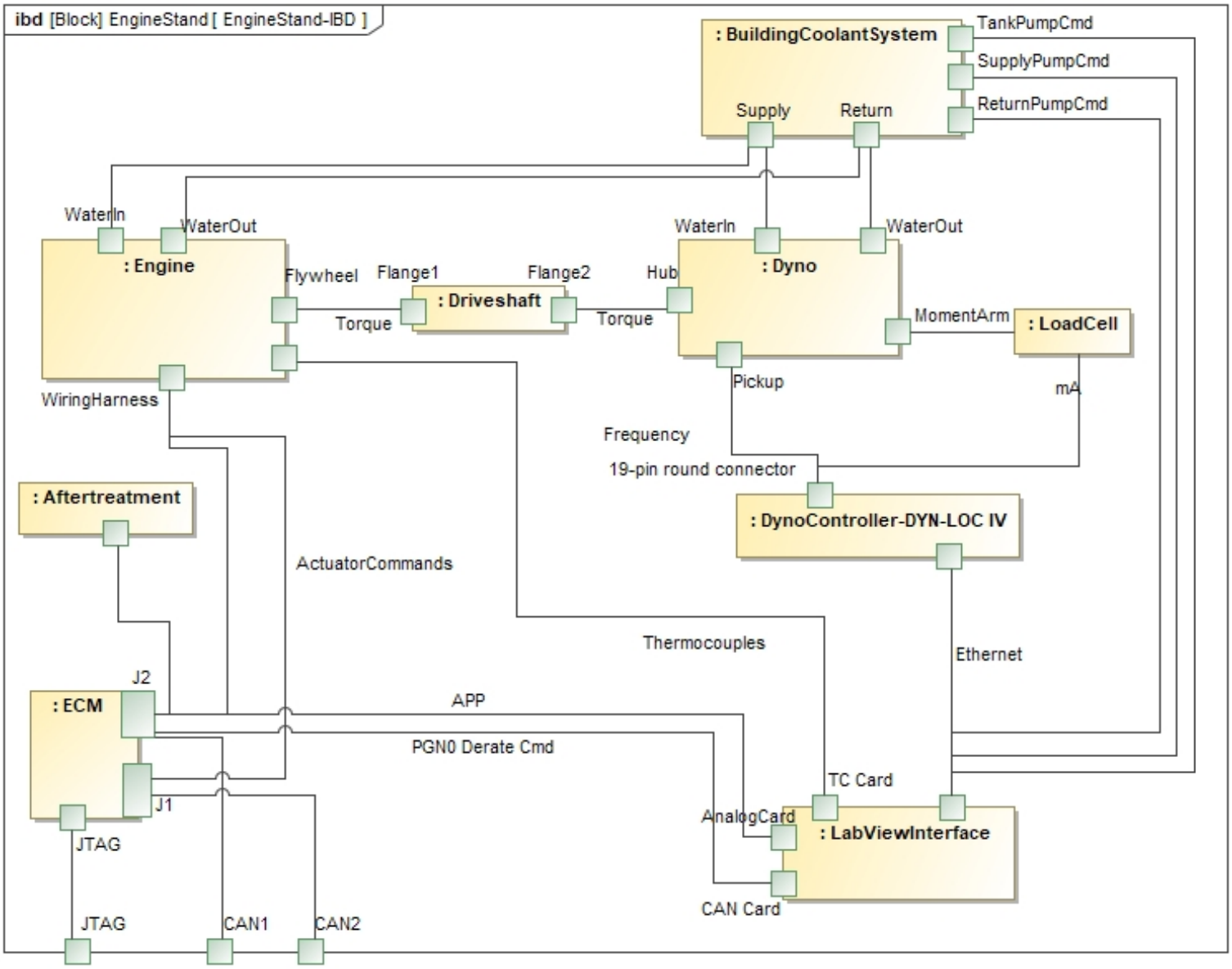


Figure 11. Engine Test Stand Internal Block Diagram

2.2 Design/Implementation

The test stand for this project was implemented with an existing Cummins QSB6.7 engine and a 1000 hp eddy-current dynamometer installed at the CSU Powerhouse shown in Figure 12. Prior to this project, the engine had not been used for several years, had parts missing, and showed numerous fault codes. Table 5 summarizes the specifications of the engine. This section provides details on how this engine was restored and re-configured to meet the requirements of this project.

Figure 12. Engine & Dynamometer Test Stand

Table 5. Engine Specifications

Name	QSB597
Date of Inspection	9/17/2020
Engine Serial Number:	73915597
Rated Power	224 kW @ 2500 rpm
Rated Torque	1044 Nm @ 1500 rpm
Model Year	2015
Engine Make	Cummins
Engine Model	QSB6.7 300
Engine Hours @ start of project	82hr 2 min
Engine Displacement	6.7 L
Number of Cylinders	I-6
Fuel Delivery System	Common Rail DI
Engine Family	FCEXL06.7AAK
Emissions Category	130-560 kW
Engine Oil Viscosity Grade	15W-40 CJ-4
Fuel Type	Dyed ULS diesel
Exhaust Aftertreatment	EPA Tier 4: DOC + SCR w/ urea inject
ECM Part Number	5317106
ECM Code	ER80001.12

2.2.1 Hardware Fabrication

A number of hardware modifications were performed:

- New exhaust pipe was fabricated (Figure 13)
- New intercooler was procured and installed (Figure 14)
- Cylinder pressure transducer ports blocked off (Figure 15)
- Series of thermocouples installed (Figure 16)
- A fuel tank was configured to feed the engine (Figure 17)

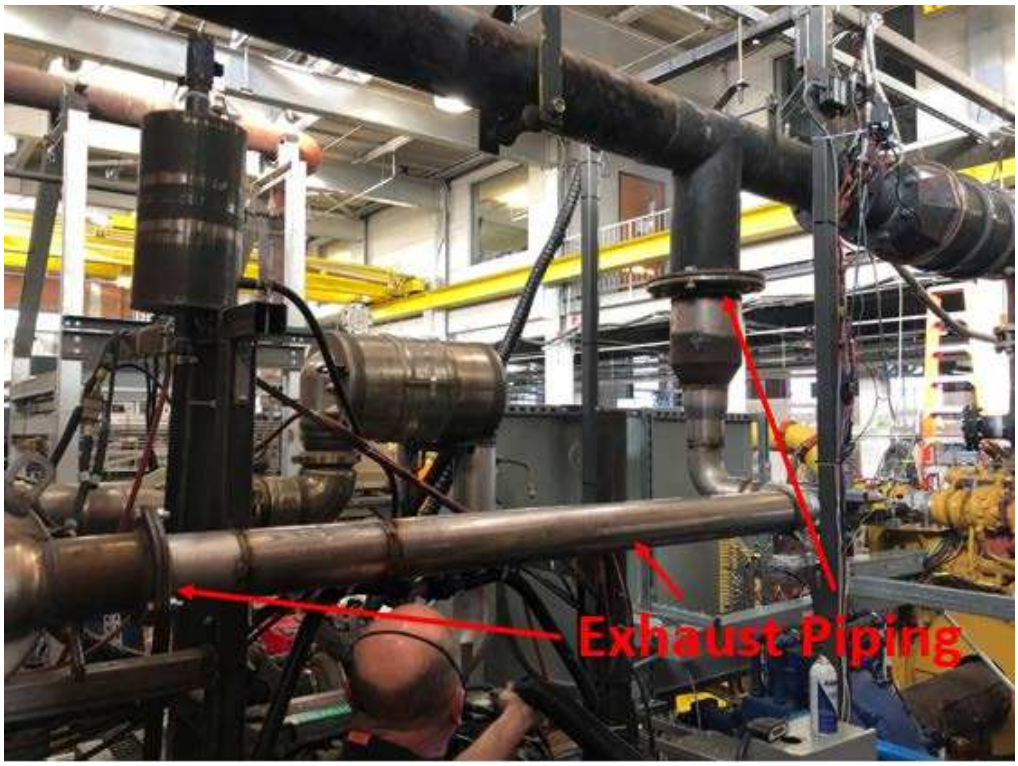


Figure 13. Exhaust Pipe Fabricated

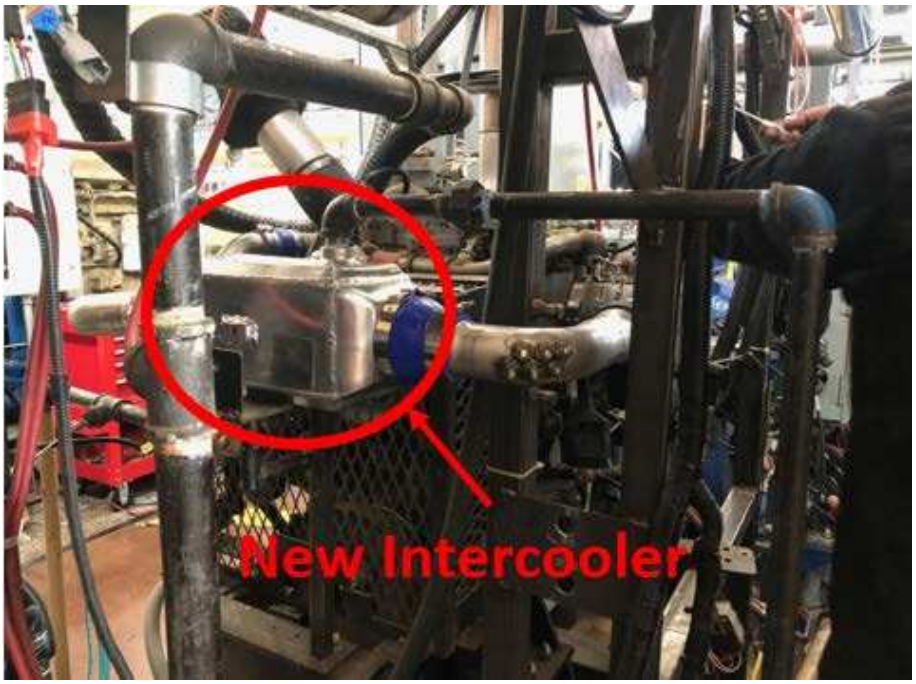


Figure 14. New Intercooler Procured and Installed

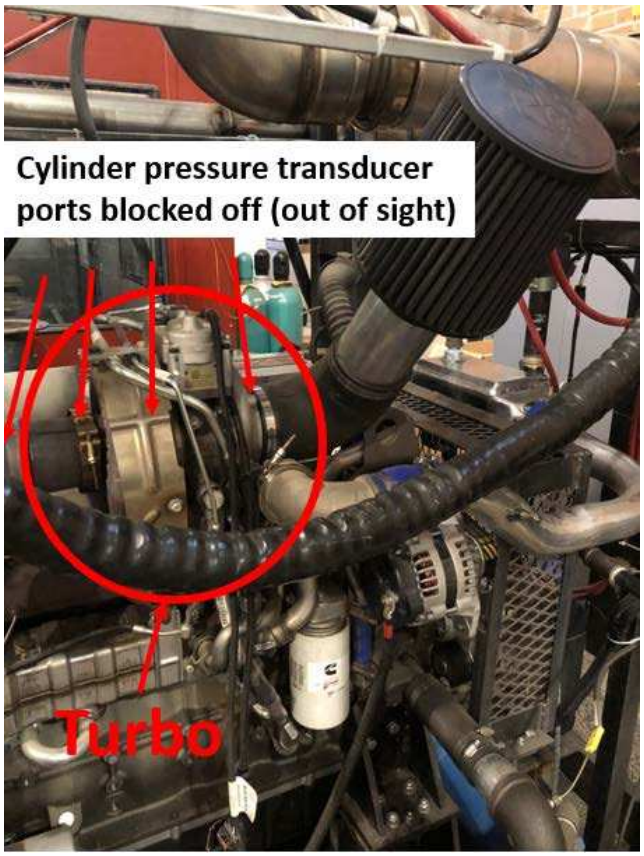


Figure 15. Cylinder Pressure Transducer Ports Blocked Off (behind turbo)

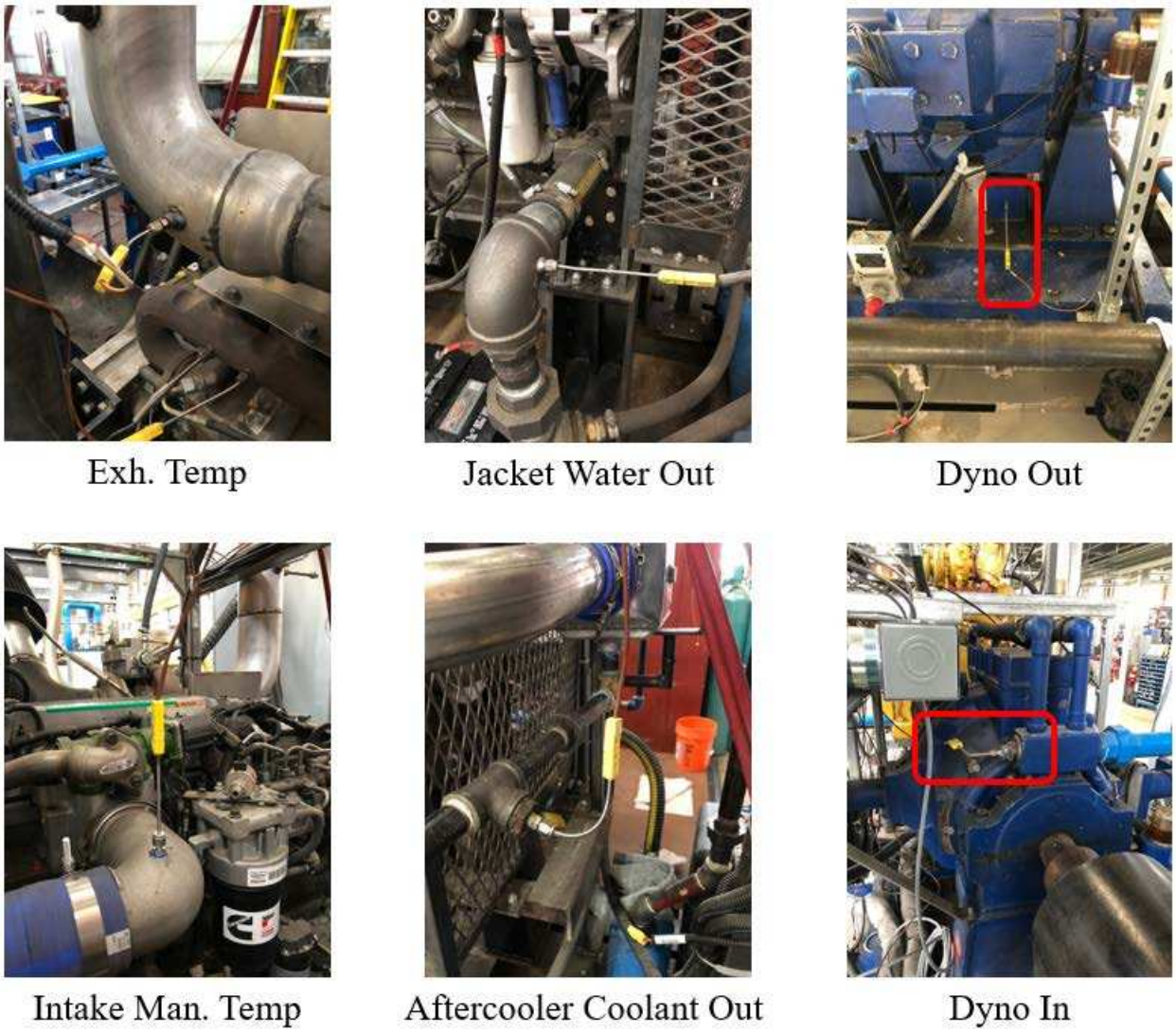


Figure 16. Thermocouples Installed

Figure 17. Overhead view of the fuel tank located outside the east side of the older part of the Powerhouse campus.

2.2.2 Repairing Engine

At project kickoff, the Cummins QSB6.7 engine controller indicated numerous fault codes and was unable to produce rated power. Five main problem areas caused either an engine derate or no-start conditions:

(1) Faulty wiring

(2) Intake Air Temperature & Pressure sensor

(3) Failed DEF dosing unit

(4) Failed DEF tank sending unit

(5) Passive vehicle sensors missing

(1) The majority of the faults were remedied by repairing faulty connections in the wiring harness. Numerous low-quality connections were replaced by either solder or an automotive-

grade Deutsch connectors. Figure 18 illustrates an example of poor aftermarket connections causing power loss to all aftertreatment system controllers.

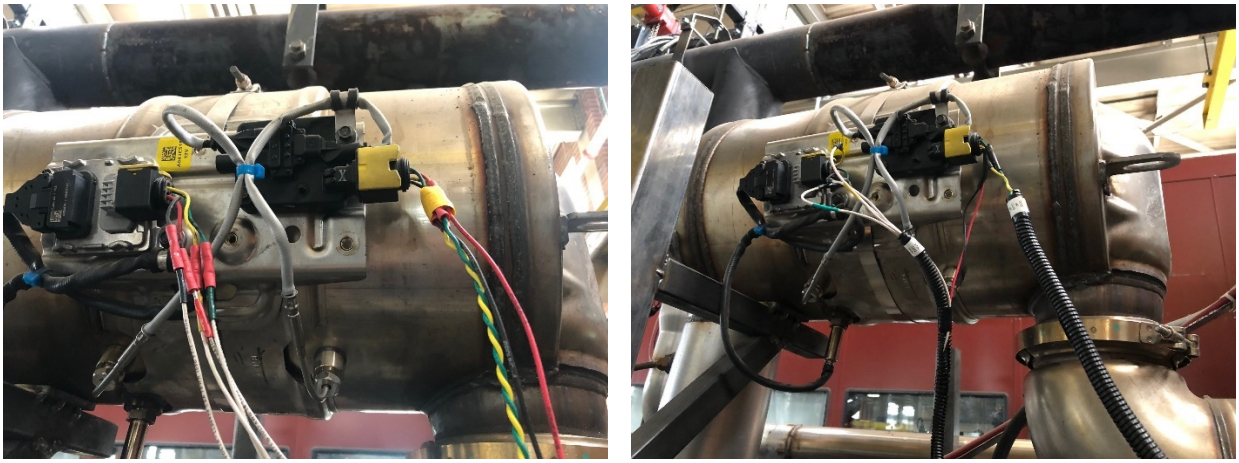


Figure 18. Example of Repaired Wiring

(2) The intake air temperature & pressure sensor was missing from the engine, but the stock wiring harness had a connector for it. This sensor contributes to the barometric pressure reading and the calculated intake manifold air conditions. The sensor was procured and added as shown in Figure 19.

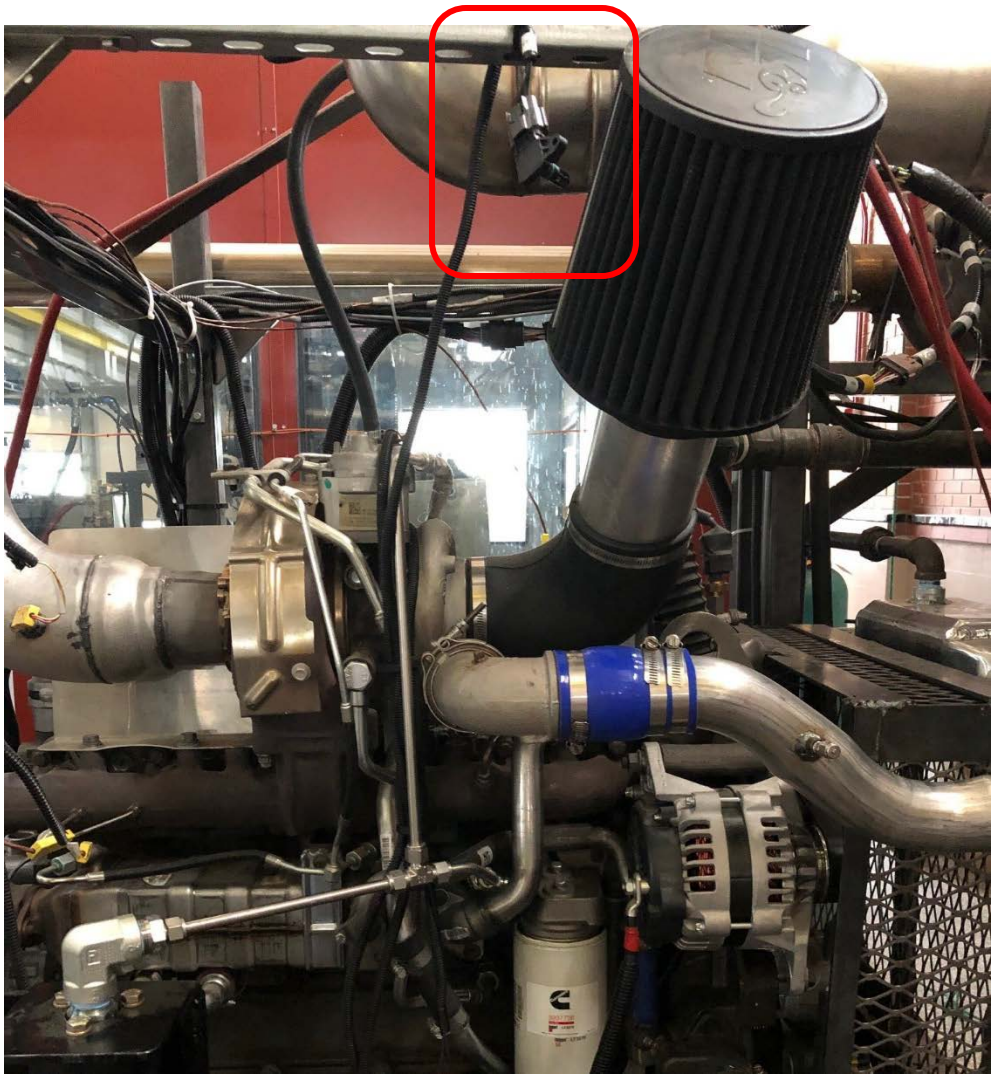


Figure 19. Ambient Air Sensor

(3) The DEF dosing unit had failed and was unable to inject DEF into the exhaust stream. This rendered the SCR catalyst unable to reduce NO_x. Because this makes the engine non-compliant with EPA emissions regulations, the engine controller severely derated the engine, or put it in a "limp-home mode." A new dosing unit was purchased, and the Cummins-recommended installation procedure was followed eliminating this fault. NO_x reduction could also be seen in operation as discussed later in Section 3.1: Verification/Validation

Figure 20. DEF Injector

(4) The DEF tank sending unit (Figure 21) measures DEF quality and temperature. Because DEF freezes at -11°C, the temperature reading is only used to open a valve for engine coolant to flow through the DEF tank and will not cause a derate. However, DEF quality must be measured at 32.5% for the aftertreatment system to function properly. The DEF tank sending unit was determined to have failed with no CAN messages coming from it. A new unit was purchased, and the Cummins-recommended installation procedure was followed. DEF tank-related fault codes were cleared, and CAN messages could be seen from the new sending unit.

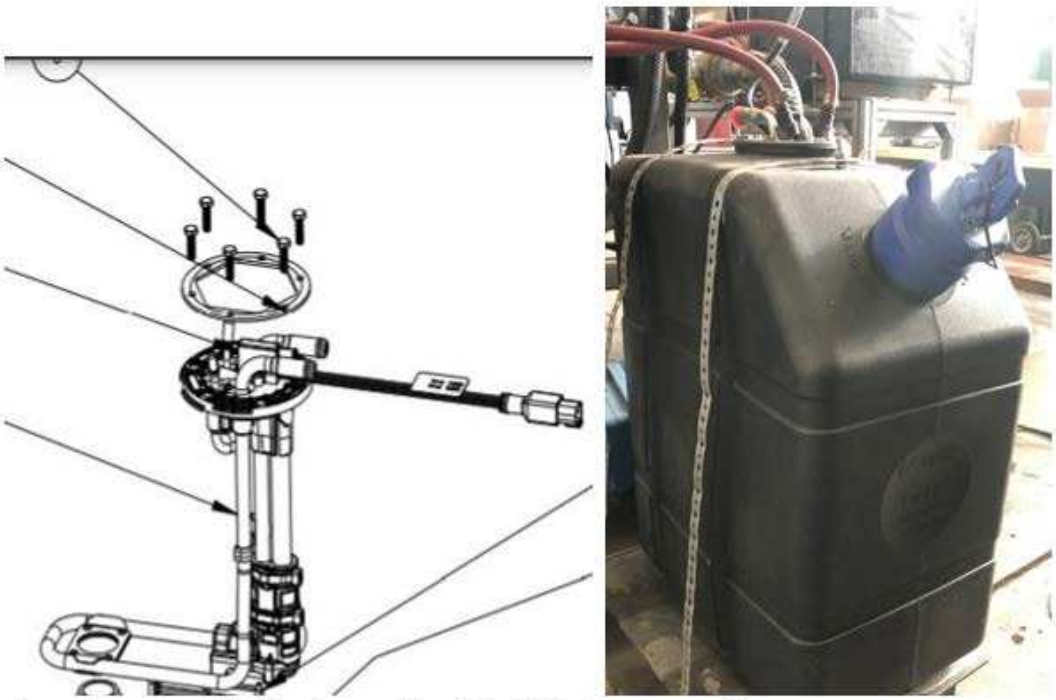
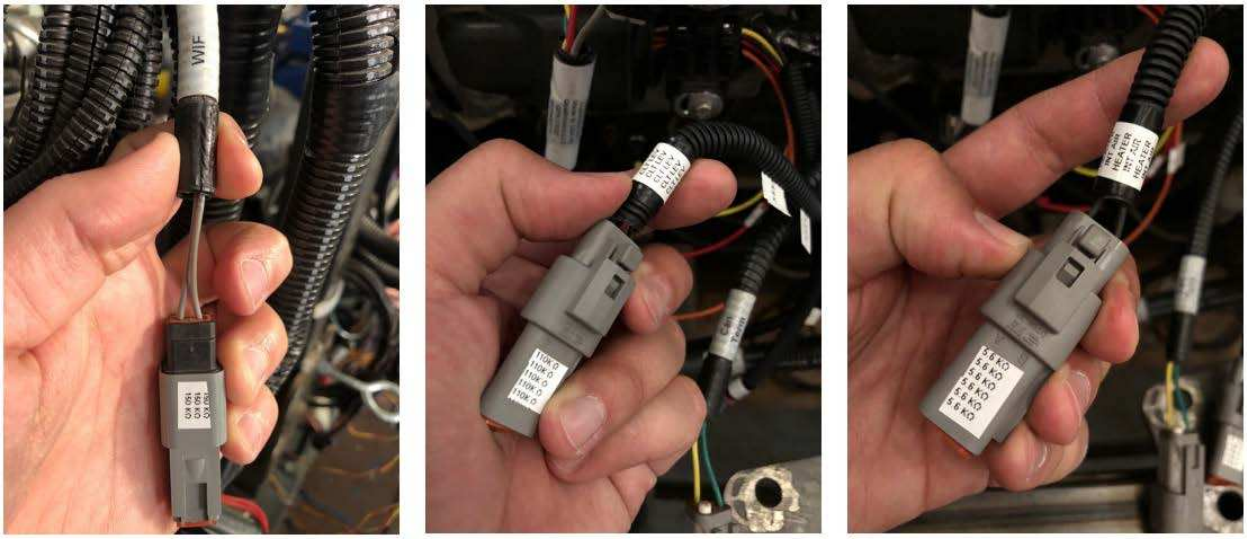

https://www.cumminsfiltration.com/sites/default/files/MB10033.pdf

Figure 21. DEF Tank Header

(5) Several passive sensors normally present on a vehicle were missing from the engine setup. These included the water-in-fuel sensor, coolant level sensor, and the intake air heater. Unlike the ambient air sensor, these sensors simply need to send a constant signal to the ECM indicating a nominal value. Each of these sensors had a connector present on the engine wiring harness, but no sensor present. This was addressed by measuring the resistance of these three sensors on the Kenworth and installing a similar resistor on the engine test stand. Figure 22 shows the Kenworth's coolant level sensor as an example, and Figure 23 summarizes the sensors normally found on an OEM application.

Figure 22. Kenworth Coolant Level Sensor



Water In Fuel Sensor: 150 kΩ | Coolant Level Sensor: 110 kΩ | Intake Air Heater Sensor: 560 Ω

Figure 23. OEM Sensor Resistors

In summary, Table 6 lists all fault codes present at the beginning of the project and their solutions. The INSITE scan tool assigned four "lamps" to each code: None, Maintenance, Amber and Red. "None" had no effect on engine operation. "Maintenance" did not affect engine operation but indicated required normal maintenance. "Amber" caused a derate: the engine would operate but at reduced power. "Red" meant a no-start condition.

Table 6. Engine Faults & Solutions

Fault Code	Lamp	Description	Solution
0141	Amber	Engine Oil Rifle Pressure 1 Sensor Circuit – Voltage Below Normal or Shorted to Low Source	Repaired Faulty 12V power wiring
0212	Amber	Engine Oil Temperature Sensor 1 Circuit Voltage Above Normal or Shorted to High Source	Repaired Faulty 12V power wiring
0222	Amber	Barometric Pressure Sensor Circuit – Voltage Below Normal or Shorted to Low Source	Installed Intake temperature & pressure sensor 
0195	Amber	Coolant Level Sensor 1 Circuit – Voltage Above Normal or Shorted to High Source	Installed 110 kΩ resistor on wiring harness connector 
0285	Amber	SAE J1939 Multiplexing PGN Timeout Error – Abnormal Update Rate	Replaced DEF tank sending unit
1241	Amber	Accelerator Pedal or Lever Position Sensor 2 Circuit – Voltage Below Normal or Shorted to Low Source	Set up APP 0-5V signals from LabVIEW
1928	Amber	Aftertreatment Fuel Pressure Sensor Circuit – Voltage Below Normal or Shorted to Low Source	Repaired Faulty 12V power wiring
1881	Amber	Aftertreatment Diesel Particulate Filter Differential Pressure Sensor Circuit – Voltage Below Normal or Shorted to Low Source	Repaired Faulty 12V power wiring
3134	Amber	Aftertreatment 1 Diesel Particulate Filter Outlet Pressure Sensor Circuit – Voltage Below Normal or Shorted to Low Source	Repaired Faulty 12V power wiring
3137	Amber	Engine Exhaust Gas Recirculation Outlet Pressure Sensor Circuit –	Repaired Faulty 12V power wiring

		Voltage Below Normal or Shorted to Low Source		
0691	Amber	Turbocharger 1 Compressor Intake Temperature Circuit – Voltage Above Normal or Shorted to High Source	Installed Intake temperature & pressure sensor connector	
0249	Amber	Ambient Air Temperature Sensor 1 Circuit Voltage Above Normal or Shorted to High Source	Installed Intake temperature & pressure sensor	
1845	Maintenance	Water in Fuel Indicator Sensor Circuit – Voltage Above Normal or Shorted to High Source	Installed 150 kΩ resistor on wiring harness connector	
3232	Amber	Aftertreatment 1 Intake NO_x Sensor – Abnormal Update Rate	Repaired Faulty 12V power wiring	
2771	Amber	Aftertreatment 1 Outlet NO_x Sensor – Abnormal Update Rate	Repaired Faulty 12V power wiring	
4152	Amber	Aftertreatment Selective Catalytic Reduction Temperature Sensor Module Abnormal Update Rate	Repaired Faulty 12V power wiring	
4151	Amber	Aftertreatment Diesel Particulate Filter Temperature Sensor Module – Abnormal Update Rate	Repaired Faulty 12V power wiring	
2222	Amber	Fuel Level (Main Tank) Sensor Circuit – Voltage Above Normal or Shorted to High Source	Cleared fault codes	
3224	Amber	Aftertreatment Purge Air Actuator Circuit – Voltage Above Normal or Shorted to High Source	Repaired Faulty Wiring	
1923	Amber	Aftertreatment Fuel Shutoff Valve Circuit – Voltage Above Normal or Shorted to High Source	Repaired Faulty Wiring	
1977	Amber	Aftertreatment Doser Circuit – Current Below Normal or Open Circuit	Replaced DEF Dosing Unit	
6418	None	Engine Brake Actuator Driver 1 Circuit – Voltage Above Normal or Shorted to High Source	Cleared fault codes	
4585	Red	Aftertreatment 1 SCR Catalyst System – Special Instructions	Replaced DEF Dosing Unit	

4584	Red	Aftertreatment Diesel Particulate Filter System – Special Instructions	Replaced DEF Dosing Unit & DEF tank sending unit	
1896	Amber	EGR Valve Controller – Out of Calibration	Cleared fault codes	
6656	None	SCR Monitoring System Malfunction – Special Instructions	Replaced DEF Dosing Unit	
6261	None	Engine Starter Motor Relay Circuit – Voltage Above Normal or Shorted to High Source	Cleared fault codes	

2.2.3 Control Room Tether

A tether was installed between the engine test stand and control room for future use. Table 7 lists the cables available in the tether. This tether runs from the engine test stand to the south control room on the 2nd floor in the Powerhouse: a run of roughly 25 meters.

Table 7. Control Room Tether

Description	Gauge	
2x twisted pair 1	18	
2x twisted pair 2	18	
2x twisted pair 3	18	
2x twisted pair 4	18	
CAT6 Network	-	
CAT6 Network	-	
CAT6 Network	-	

It was anticipated that some functionality with the ECM will be moved to the control room in the future. Basic physical elements of a CAN bus are defined in ISO 11898-2 [16] and shown Figure 24. The longest stretch of wire, the trunk, has a 120 Ω termination resistor on each end. Intermediate branches, or stubs, split from the trunk to various controllers such as an engine

control unit, transmission control unit, datalogger, or other CAN devices. Branch lengths should be kept at a minimum and are typically less than 10 m long. This ideal layout gives the best protection from noise. However, the engine test stand's trunk was already built into the engine's wiring harness with both 120 Ω termination resistors. Therefore, the control room tether is a branch. Because this branch was about 25 m, the integrity of CAN bus over this cable bundle was evaluated.

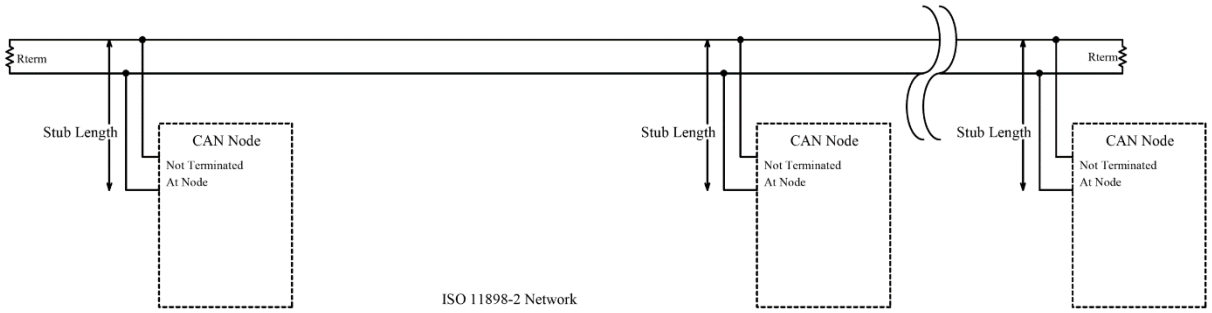


Figure 24. CAN Bus Physical Architecture [17]

To test this, the quality of the CAN bus signal was probed with an oscilloscope and an isolated probe at the engine without the tether connected to determine signal quality under ideal conditions as a baseline. With the engine keyed on to send CAN messages, Figure 25 shows this signal quality.

Next, the tether was plugged in. Very long cable runs can cause signal degradation as can be seen in Figure 26. The falling edge of each pulse was very sharp in the ideal scenario, but gradual decay can be seen with the tether plugged in. This behavior was due to the long length of the cable acting like a capacitor, but because pulses were clearly discernible, tether length was still far too short for this effect to cause meaningful signal degradation.

The tether was then probed at the control room. Figure 27 shows that signal quality was nearly identical to the previous test case. High and low pulses were still clearly discernible, the decay of falling edges did not constitute a problem, and little noise was otherwise observed.

As a final validation check, CAN traffic was logged for five minutes at the control room, and no CAN errors were recorded. Further, the Cummins INSITE scan tool could successfully connect to the ECM through the tether.

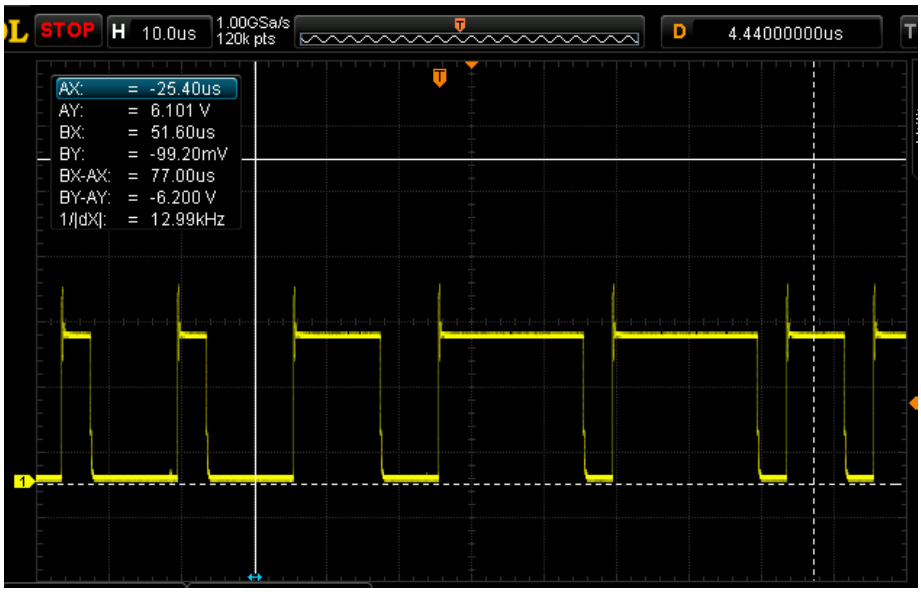


Figure 25. Oscilloscope Image of CAN high relative to CAN low: At Engine with Tether Disconnected

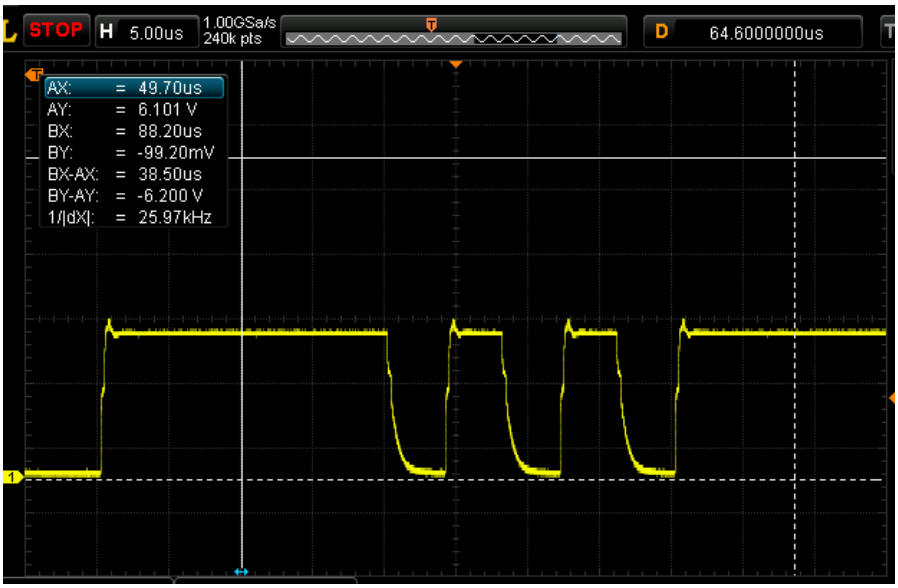


Figure 26. Oscilloscope Image of CAN high relative to CAN low: At Engine

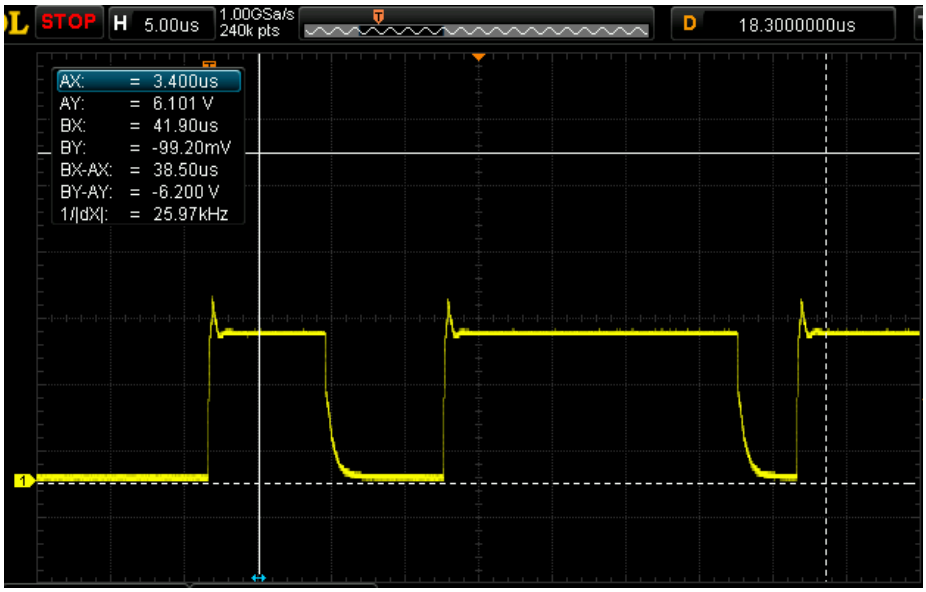


Figure 27. Oscilloscope Image of CAN high relative to CAN low: At Control Room

2.3 Integration/Test

The next stage of developing the test stand was configuring the engine to operate as if it were in an on-highway truck. Unlike its cousin, the Cummins ISB6.7, the QSB6.7 used in the test stand was configured for industrial applications such as tractors or construction equipment. Because the project objective was to evaluate binary patches on a truck engine and simulate a hard braking maneuver, the test stand engine must be configured similar to an on-highway engine.

2.3.1 Accelerator Pedal Setup & Calibration

Requirement: R2.1 - Highway APP Control

The main configuration change to represent an on-highway truck application was the engine speed governor control strategy. The default governor for the off-highway application was speed control in which the accelerator pedal position correlated with an engine speed setpoint. In an on-highway application, engine speed is dictated by vehicle speed and transmission gear, while accelerator pedal position dictates engine load. The engine's speed governor was simply changed to an automotive load control through INSITE in which the accelerator pedal position correlated with percent engine load.

Next, the test cell's LabVIEW controller (described further in Section 4.2) was configured to send accelerator pedal position signals to the ECM. The accelerator pedal position was calibrated by applying seven voltages from 0 to 5 V to the two APP inputs on the ECM's wiring harness. Figure 28 shows the results of this calibration and the slope and intercepts needed for the LabVIEW controller. It is important to note that the ratio between these two voltages (~2) must be consistent, and neither signal should read zero with the engine on. These scenarios are part of

the ECM's fault-finding diagnostics and cause a fault code in which the engine may idle, but the accelerator pedal is ignored.

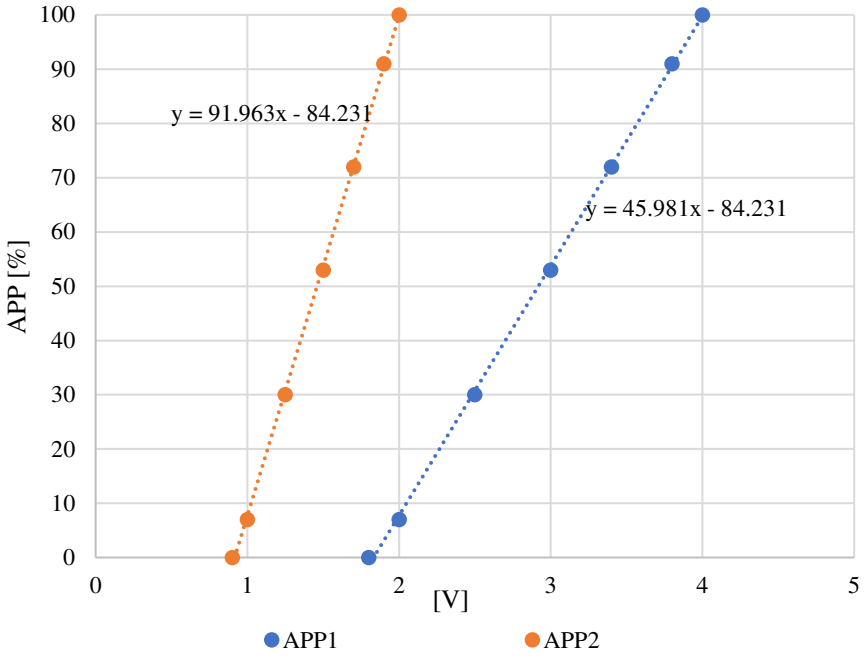


Figure 28. APP Calibration

2.3.2 Speed & Load Calibration

Load

Engine load is applied with an eddy-current dynamometer installed on the test stand. This type of dynamometer can only absorb torque: it is unable to motor, and its rotational inertia is significantly higher than that of the engine, so its torque can never be negative. However, the Cummins engine shares this dynamometer with a different engine which connects to the opposite shaft of the dynamometer causing the shaft to spin the opposite direction. To accommodate the opposing engine and following engineering best practices, the load cell of the dynamometer load cell (Figure 29) was calibrated in both positive and negative directions. Having a record of

acceptable linearity for both directions means load cell polarity can simply be switched for each engine without re-calibrating the load cell.

Figure 29. Load Cell

Using an element of reference architecture, dead-weight calibration was performed following CFR Part 1065 procedures which specifies calibration requirements for engine test cells used for EPA emissions certification [18], [19]. This involved determining an appropriate load range and applying six roughly-spaced weights to the moment arm of the dynamometer.

First, an upper range was selected. The load cell was rated for 8900 N (2000 lb-f), and the highest torque expected from either engine was 1600 Nm. A calibration should not extrapolate, so the load cell was calibrated to 2000 Nm, or 3274 N force at the load cell. Error and linearization calculations used this value as the load cell's full-scale value.

Second, the general calibration procedure in Appendix B: was followed. The load cell was zeroed and spanned based on the 8900 N span point selected. Constants for calibration arm length, calibration arm center of gravity, calibration arm mass, centerline-to-load cell distance were stamped on their respective parts (Figure 30) and used for subsequent calculations. Local

acceleration due to gravity was determined as specified in the CFR, and plate masses were calibrated and labeled on each dead weight. These values are summarized in Table 8.

Table 8. Load Cell Calibration Constants

Cal Arm Total Length [m]	0.9548
Cal Arm C.G. [m]	0.1621
Cal Arm mass [kg]	12.925
Plate mass [kg]	90.72
Local gravity [m/s2]	9.79733
Centerline-to-loadcell [m]	0.6112

Figure 30. Calibration Arm & Load Cell Moment Arm Dimensions

Figure 31 illustrates the final calibration curve and the point spacing. Seven points were used spaced between zero and 2000 Nm based on properly calibrated dead weights available in the lab.

Table 9 shows the complete results of the calibration including adding and removing weights from both the positive (tension) and negative (compression) sides of the dynamometer. The largest error of 4.4 % was observed at the lowest value of 164 N or 100.4 Nm. This is not unexpected: the load cell is rated by the manufacturer at 8900 N, and this point is 1.8 % of full scale. While absolute error at this point is similar to that of other points, percent error becomes large close to zero. It was also observed that increasing and decreasing weights give very similar

outputs and percent errors indicating low hysteresis. The load cell was also very linear and repeatable in both the tension and compression directions.

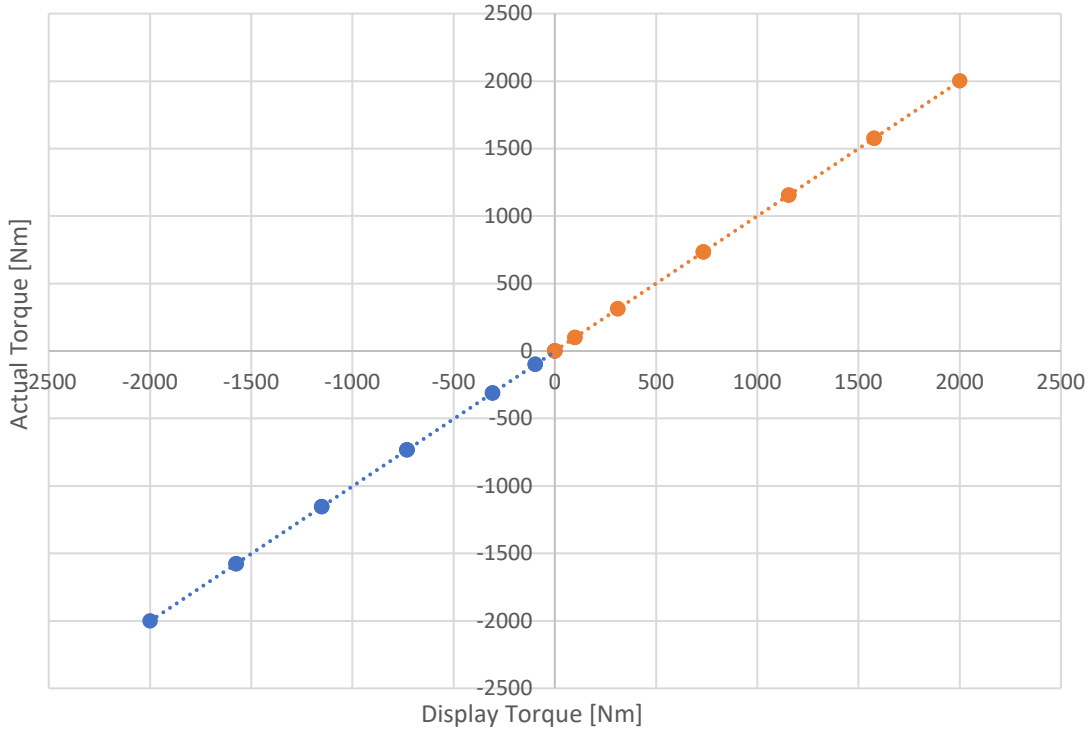


Figure 31. Load Cell Calibration Curve

Table 9. Load Cell Calibration Results

	Dead Weights	Mass (enter value)	Load Cell Force	Actual Torque, y_ref	Display Torque, y_i (enter value)	Absolute Error	% Absolute Error	Linearized	Linearization Error	% Linearization Error	SumSq
		kg	N	Nm	Nm	Nm		Nm	Nm	%	
Compression (north)	Zero	0.0	0	0.0	0.0		n/a	-1.5	-1.5	n/a	2.31
	cal arm	12.9	-164	-100.4	-96.0	4.4	-4.4%	-97.6	2.8	-2.8%	7.79
	hanger	22.6	-511	-312.2	-306.0	6.2	-2.0%	-307.7	4.4	-1.4%	19.49
	#4	45.1	-1201	-734.2	-728.0	6.2	-0.8%	-730.0	4.2	-0.6%	17.45
	#4,3	44.9	-1888	-1153.9	-1150.0	3.9	-0.3%	-1152.3	1.6	-0.1%	2.58
	#4,3,2	45.2	-2580	-1576.6	-1573.0	3.6	-0.2%	-1575.6	1.0	-0.1%	1.01
	#4,3,2,1	45.3	-3274	-2000.8	-2000.0	0.8	0.0%	-2003.0	-2.1	0.1%	4.48
	#4,3,2	45.2	-2580	-1576.6	-1576.0	0.6	0.0%	-1578.6	-2.0	0.1%	4.00
	#4,3	44.9	-1888	-1153.9	-1152.0	1.9	-0.2%	-1154.3	-0.4	0.0%	0.16
	#4	45.1	-1201	-734.2	-732.0	2.2	-0.3%	-734.0	0.2	0.0%	0.03
	hanger	22.6	-511	-312.2	-308.0	4.2	-1.3%	-309.7	2.4	-0.8%	5.82
	cal arm	12.9	-164	-100.4	-97.0	3.4	-3.4%	-98.6	1.8	-1.8%	3.20
	Zero	0.0	0	0.0	0.0	0.0	n/a	-1.5	-1.5	n/a	2.31
Tension (south)	cal arm	12.9	164	100.4	99.0	-1.4	-1.4%	97.5	-2.8	-2.8%	8.01
	hanger	22.6	511	312.2	311.0	-1.2	-0.4%	309.7	-2.5	-0.8%	6.02
	#4	45.1	1201	734.2	733.0	-1.2	-0.2%	732.0	-2.2	-0.3%	4.91
	#4,3	44.9	1888	1153.9	1155.0	1.1	0.1%	1154.3	0.4	0.0%	0.13
	#4,3,2	45.2	2580	1576.6	1577.0	0.4	0.0%	1576.6	0.0	0.0%	0.00
	#4,3,2,1	45.3	3274	2000.8	2001.0	0.2	0.0%	2000.9	0.1	0.0%	0.01
	#4,3,2	45.2	2580	1576.6	1579.0	2.4	0.1%	1578.6	2.0	0.1%	3.84
	#4,3	44.9	1888	1153.9	1156.0	2.1	0.2%	1155.3	1.4	0.1%	1.84
	#4	45.1	1201	734.2	735.0	0.8	0.1%	734.0	-0.2	0.0%	0.05
	hanger	22.6	511	312.2	311.0	-1.2	-0.4%	309.7	-2.5	-0.8%	6.02
	cal arm	12.9	164	100.4	99.0	-1.4	-1.4%	97.5	-2.8	-2.8%	8.01
	Zero	0.0	0	0.0	0.0	0.0	n/a	-1.5	-1.5	n/a	2.31

These data were used to calculate the quality metrics required by the CFR Part 1065 procedure. Table 10 summarizes the results of the calibration and compares to required CFR values. In addition to these metrics, engineering best practices also involve checking the maximum linearization error shown in the last row of Table 10. **In summary, the load cell's calibration fell within CFR Part 1065 specifications and engineering best practices.**

Table 10. Load Cell Calibration Summary

	CFR Spec	Calibration Result		
		All	Tension	Compression
Zero, $x_{min}(a_1-1)+a_0$	≤ 1% T_{max} Or ≤ 20 Nm	1.521 Nm	0.835 Nm	3.257 Nm
Slope, a_1	0.98 : 1.02	1.000	0.999	1.000
SEE	≤ 2% T_{max} Or 40 Nm	2.2 Nm	1.99 Nm	2.53 Nm
r^2	≥ 0.990	1.000	1.000	1.000
Max linearization error	None	2.8 % (2.8 Nm at 100 Nm point)		

The dynamometer's speed measurement was verified by calibrating against the engine's speed measurement. Figure 32 shows the pickup and wheel that the dynamometer controller uses for speed control.

Figure 32. Dynamometer Speed Pickup

The reference measurement used was the engine's production speed measurement reported through INSITE. The engine was operated at six speeds with the dynamometer controlling speed at each condition while speed was recorded from the speed pickup and INSITE.

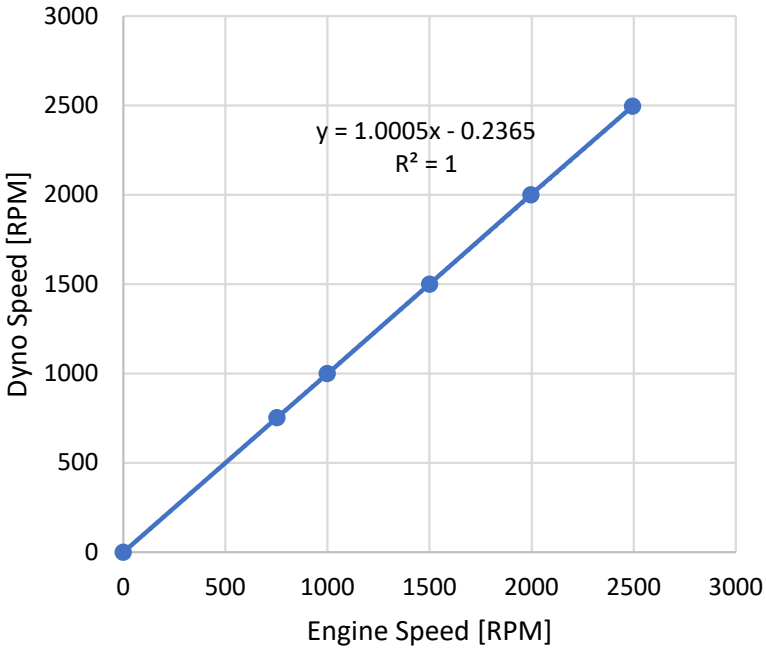


Figure 33. Dynamometer Speed Calibration

The numerical values, absolute error, linearization, and linearized error are summarized in Table 11. The highest error observed was 0.13 % at the 2000 RPM point, and the speed encoder measurement showed strong correlation with the engine's speed measurement.

Table 11. Dynamometer Speed Calibration

Setpoint	Dyno	Engine	Abs. Error	Linearization	Linearized Error	% Lin. Error
0	0.00	0.00	0.00	0.2	0.24	0.00%
750	752.30	752.59	-0.29	752.1	-0.46	-0.06%
1000	1000.00	999.92	0.08	999.7	-0.23	-0.02%
1500	1500.09	1500.23	-0.14	1499.5	-0.72	-0.05%
2000	1999.93	1996.55	3.38	1999.1	2.53	0.13%
2500	2495.69	2495.94	-0.25	2494.6	-1.37	-0.05%
Candidate Slope: 1.000542						
Candidate Intercept: -0.2365						

2.3.3 Data Acquisition

R5.2 Log dynamometer speed and torque

The LabVIEW test cell controller was configured to send speed and torque values, as measured by the dynamometer, onto the engine's diagnostic CAN bus. Dynamometer speed was measured with the speed pickup of the dynamometer. Dynamometer torque was measured with the load cell described in the previous section. Two CAN messages are broadcast continuously at 10 Hz, and their format is shown in Table 12. Table 13 shows the parameters for the two signals. This information can be used to record dynamometer speed and torque by any independent CAN datalogger.

It is worth noting that this torque measurement captured the hydrokinetic losses in the dynamometer and coil excitation torque, but because it was a direct torque measurement, it did not capture the acceleration of the dynamometer rotor or driveshaft's inertia. Therefore, this torque measurement is only meaningful under steady-state conditions.

Table 12. Dynamometer Parameter Message Format

ArbID	B1	B2	B3	B4	B5	B6	B7	B8
18F00210		DynoSpeed						
0CF03810						DynoTorque		

Table 13. Dynamometer Parameter Signal Specifications

ArbID	Signals	Bytes	Parameters
18F00210	Dyno Encoder Speed	2:3	Unsigned Integer Multiplier: 1 Offset: 0 Units: RPM Endian: LSB / Intel
0CF03810	Dyno Torque	6:7	Signed 2's Complement Integer Multiplier: 1 Offset: 0 Units: Nm Endian: LSB / Intel

2.3.4 Binary Data

Requirement: R1 - Manage ECM Binaries

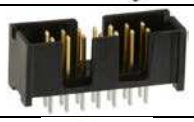
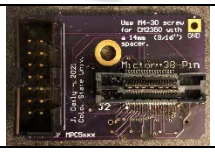
Extracting binary data can be difficult: data can be altered during the extraction process, such as ECM running time parameter, which increments while the ECM is powered on [1]. It is important that meaningful parameters are not changed during the extraction process. Much documentation for extracting binary files already exists in Alientech's documentation and Duy Van's 2020 thesis [2]. This section is intended to illustrate and supplement available information with processes developed in this project. A procedure was developed to extract and re-load binary files onto a CM2350 using the Alientech KTAG.

Procedure:

1. Modify the ECM and build a connector. Most ECMs at CSU already have a JTAG connector soldered to it, and there are a few connecting boards available. If this is the case, skip to step 2.

Table 14 lists the bill of materials needed to connect the KTAG. Soldering to the ECM can be difficult, and shorts between pins are common and difficult to detect. The custom JTAG connector board was designed at CSU and printed by Osh Park.

Table 14. JTAG Setup Bill of Materials

Description	Source	QTY	Image
CM2350 Part number must be 5317106 or P5317106-RX (remanufactured)	Cummins (or salvage)	1	
Samtec 40 Receptacle	Digikey P/N: QSE-020-01-F-D-A-K-TR	1	
Samtec 40 Header	Digikey P/N: QTE-020-03-L-D-A-K-TR	1	
Mictor 38 Receptacle	Digikey P/N: 2-767004-2	1	
Connector Header 14 position	Digikey P/N: 5103308-2	1	
Nylon Spacer	Digikey P/N: 13RSSR0084	1	
Nylon Washer	Digikey P/N: 3123	1	
Custom JTAG Connector Board	https://github.com/SystemsCyber/PowerHouseEngineTesting/	1	

Solder the Samtec 40 receptacle to the JTAG port of the ECM as shown in Figure 34. Solder a bridge across the watchdog timer shown in Figure 35. Solder the Samtec 40 header, Mictor 38 receptacle, and 14-pin header to the custom JTAG Connector board as shown in Figure 36. Using a multimeter, check the connections between pads and connector pins and check for shorts between adjacent pins. Pins 31 and 33 on the Samtec 40 header location are connected on the custom JTAG Connector board, but no other pins should show continuity. The schematic for the custom JTAG connector board is provided in Figure 37.

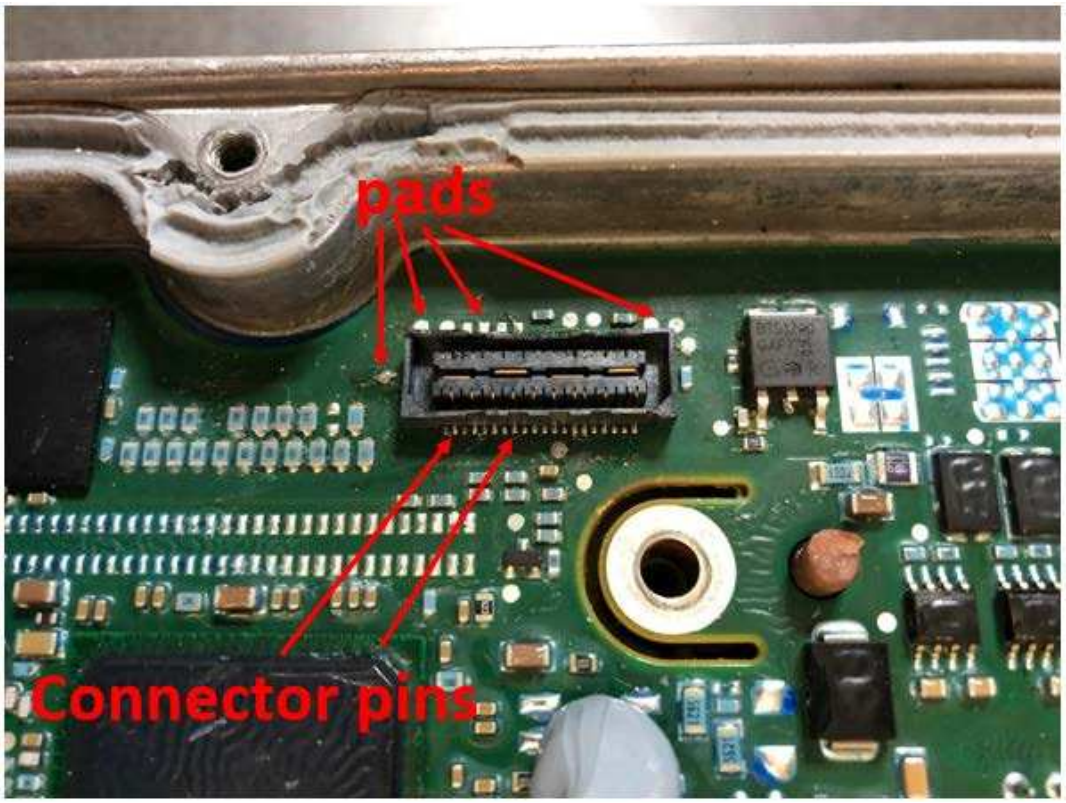


Figure 34. CM2350 Nexus/JTAG Port

Figure 35. Watchdog Timer Jumper (Source: K-Suite documentation)

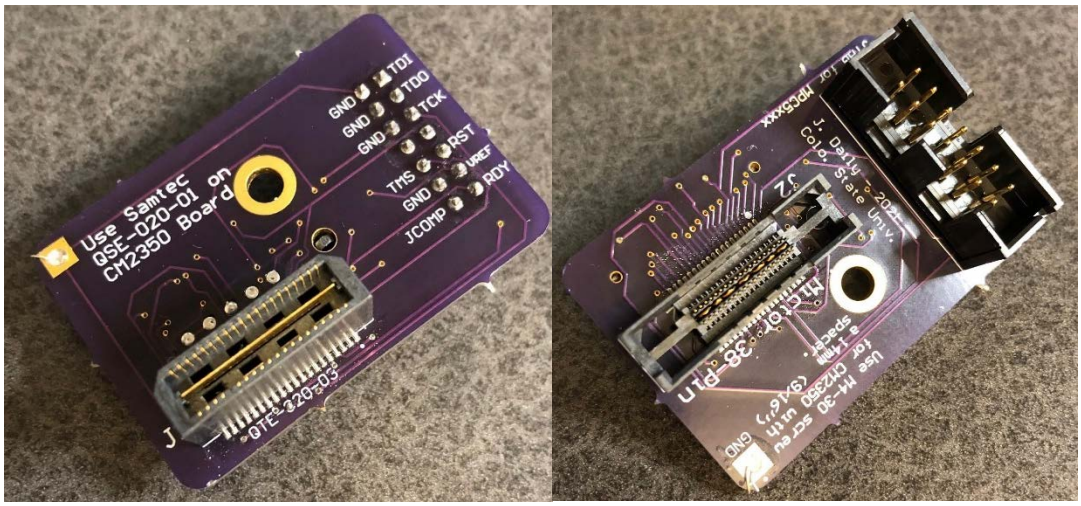


Figure 36. Custom JTAG Connector Board

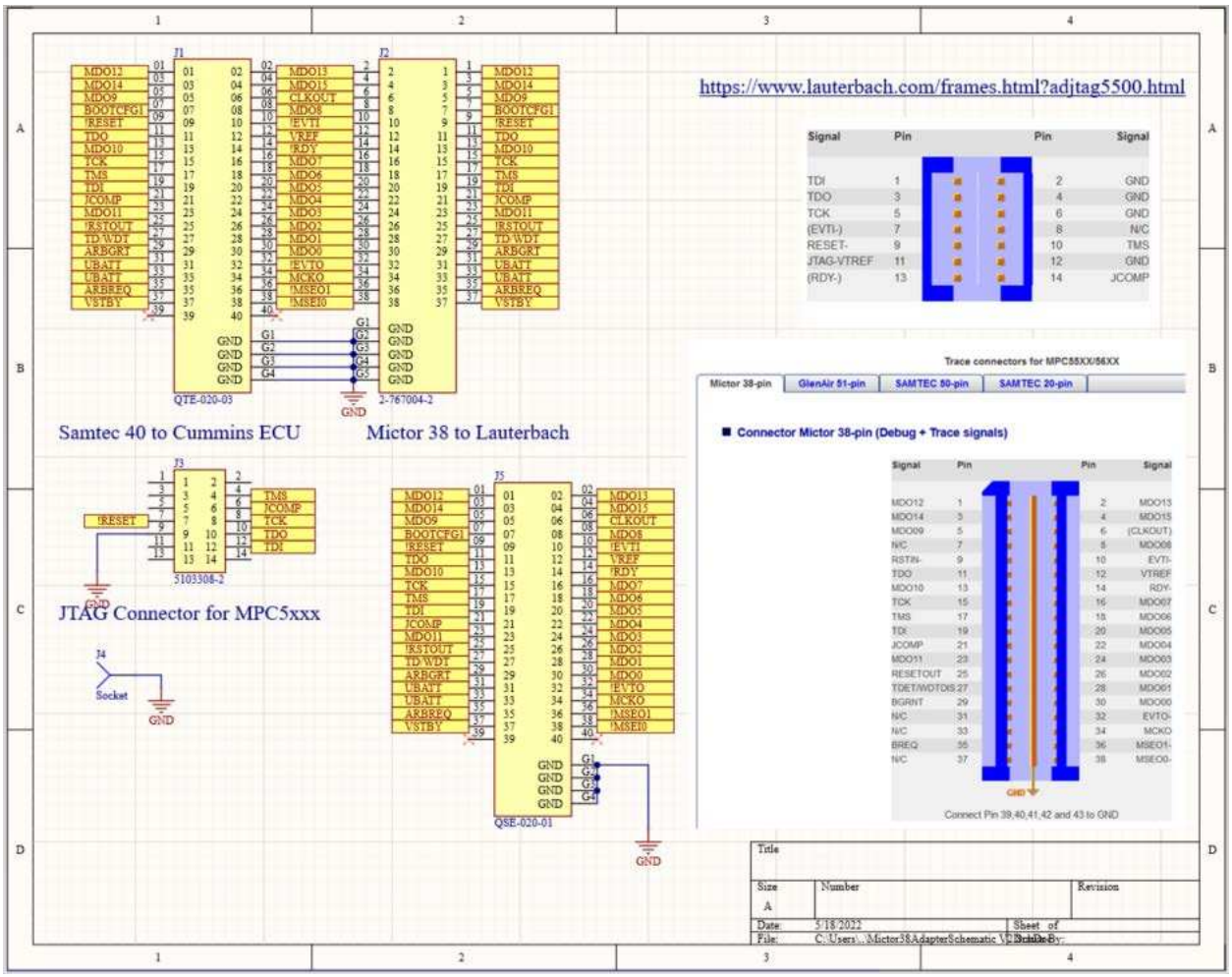


Figure 37. Custom JTAG Connector Board Schematic

Plug the custom JTAG connector board into the Samtec 40 receptacle on the ECM. Check continuity between the pads on the ECM shown in Figure 38 and with the corresponding pins on the 14-pin header shown in Figure 39.

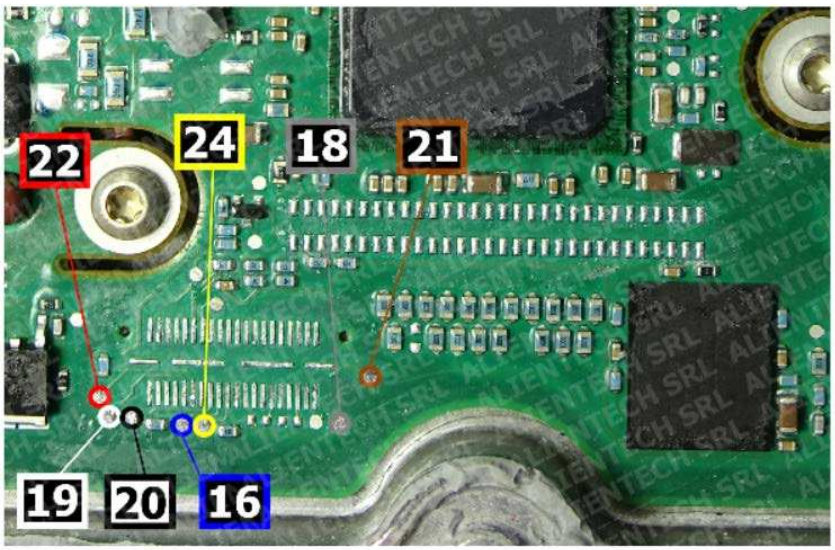


Figure 38. ECM JTAG Port Pads (Source: K-Suite documentation)

13: RDY	11: VREF	9: RST	7	5: TCK	3: TDO	1: TDI
14: JCOMP	12: GND	10: TMS	8	6: GND	4: GND	2: GND

Figure 39. 14-Pin Header Pin-Out

2. Connect the ECM to a power supply with one of the wiring harnesses to the brown ECM connector (J2) shown in Figure 40.

Figure 40. CM2350 with Wiring Harness

3. Connect the KTAG to the ECM and laptop and provide power to the ECM. The complete setup is shown in Figure 41.

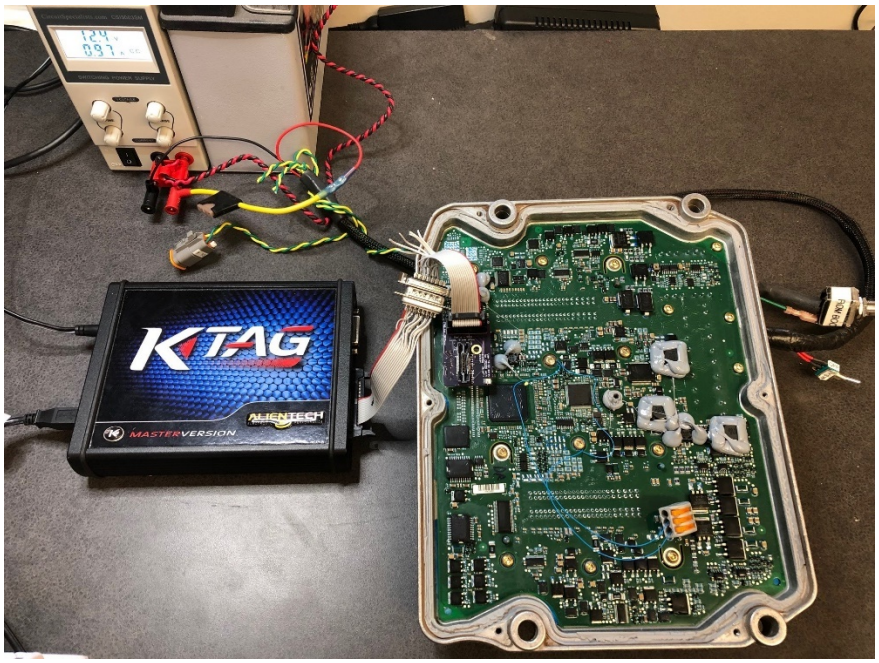


Figure 41. KTAG Reflash Setup

4. Start the K-Suite software on the laptop.  This usually requires at least one update which involve re-installing the software. For the CM2350, select “FREESCALE MPC5xxx.” This should bring up a screen showing Make: Peterbilt and the K-TAG protocol 638. With those selected, hit the green check mark to go to the next screen.

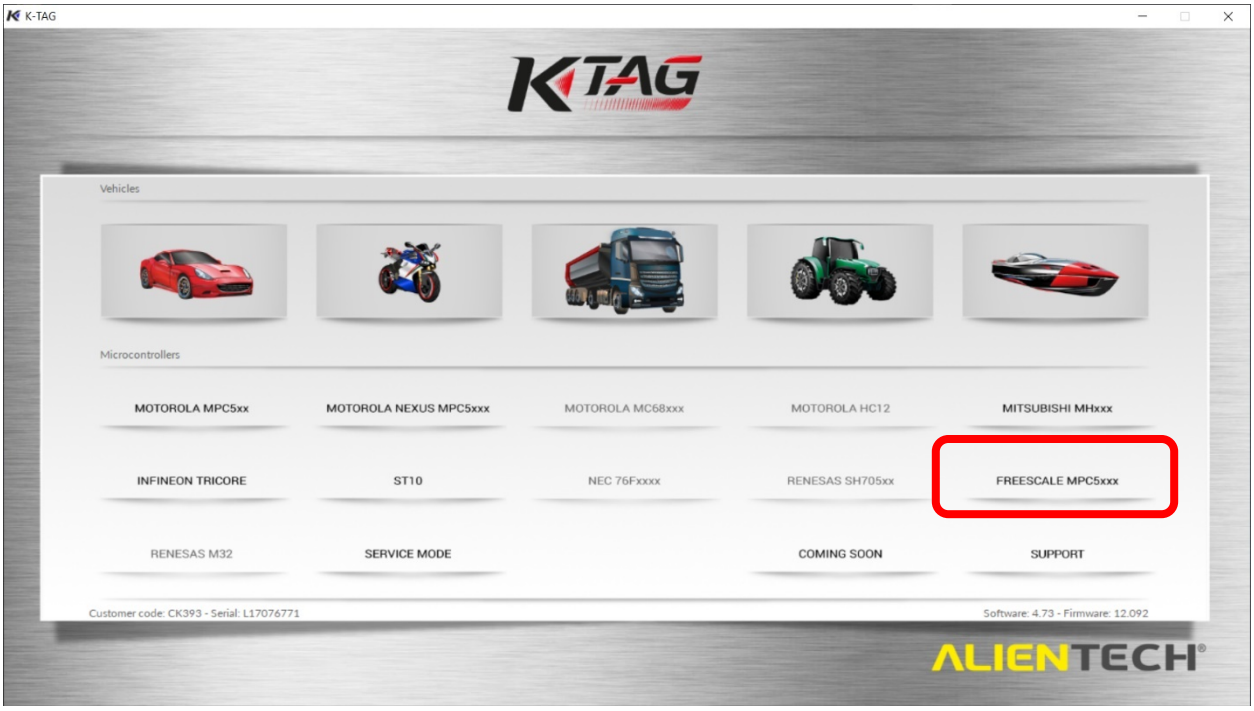


Figure 42. K-Suite Software Processor Type

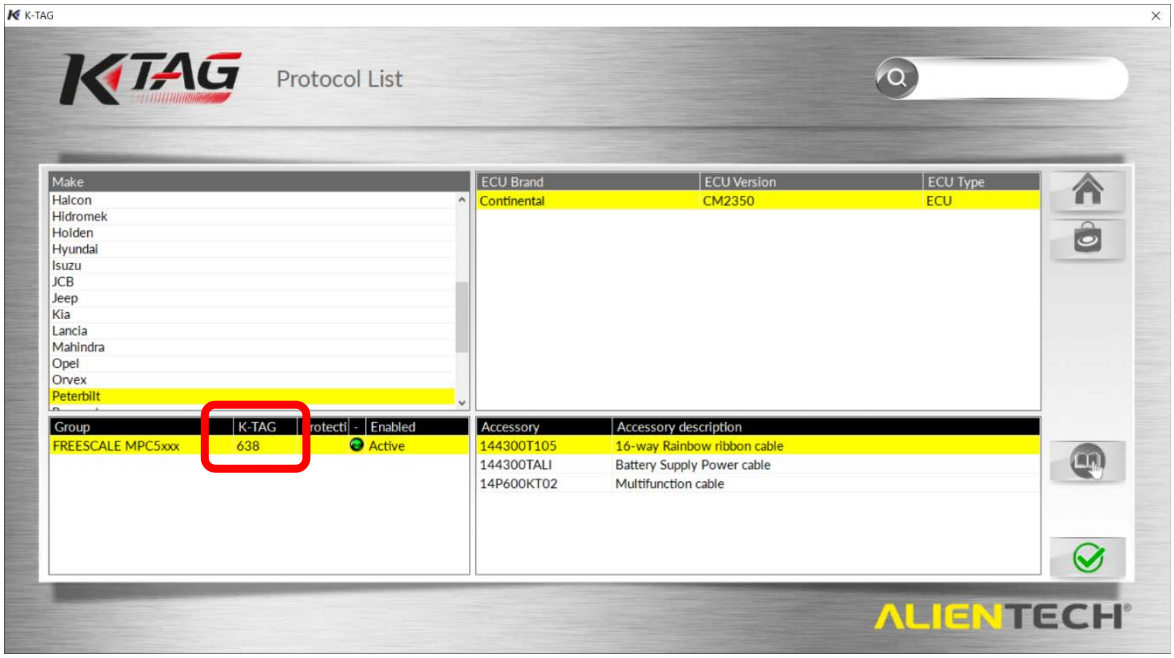


Figure 43. K-Suite Software K-TAG Protocol

5. In the next screen, hit “Identify ECU.” If the ECM is working properly and all connections are good, the field under “ECU Data” should populate (even if no information is given) as shown in Figure 44.

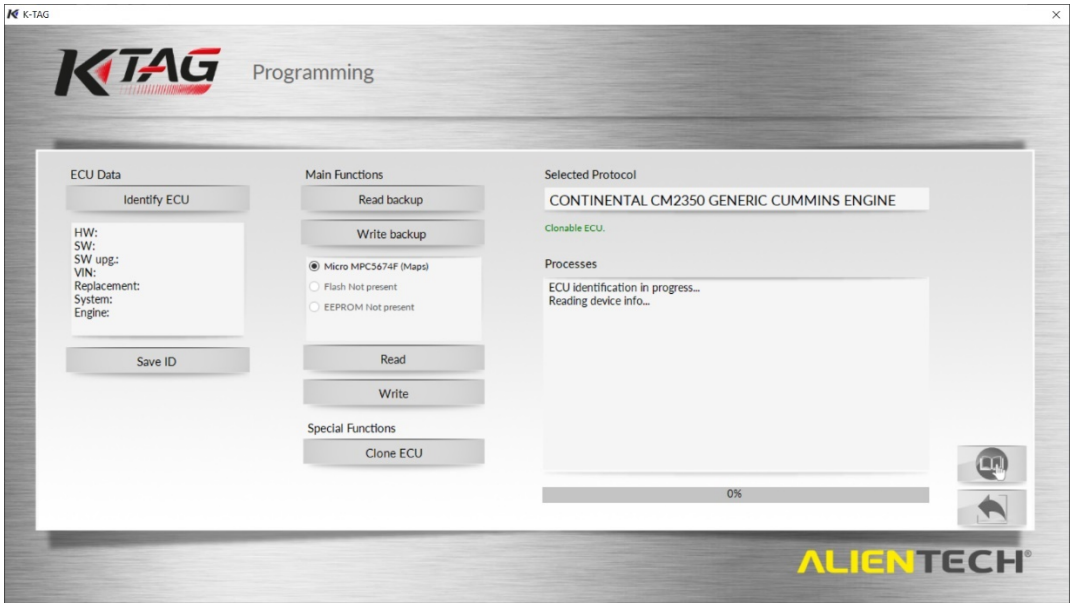


Figure 44. K-Suite Software Functional Panel

6. Any ECM should be backed up before modifying. To do this, use the “Read backup” function. If successful, there should be no errors, and K-Suite will prompt the user for a location to save the backup file.

7. To flash a new calibration, the “Write” function should be used.

8. After the ECM has been re-flashed, fault codes must be cleared using the INSITE scan tool. After the KTAG has finished loading a new calibration, the ECM boots up and operates normally before the watchdog timer short has been disconnected. This causes numerous fault codes that must be erased before operating the engine.

2.3.5 Engine Protection

Requirement: R3 - No damage to engine or test equipment with faulty ECM patch

Because the objective of this project is to evaluate patched ECM binary files, it is possible the ECM may cause unusual behavior during an evaluation. Such behavior may damage the engine or test equipment. Most faulty ECM binaries will likely cause the engine to simply not start. Out of the few that do, two main problems were identified that have the potential to cause engine damage:

1. Cause the engine to exceed its speed governor

To mitigate this risk, the dynamometer controller's (Dyn-Loc IV) overspeed protection was set to 2800 RPM: slightly above the engine's maximum speed of 2750 RPM as shown in Figure 45. If speed exceeds the overspeed trip limit by more than ½ second, the Dyn-Loc will energize the field of the dynamometer at rated current applying its highest possible torque [20].

Figure 45. Dynamometer Overspeed Protection

2. Cause the engine to stall abruptly at high speed and load conditions.

The main cause of damage at this condition is the turbocharger losing oil pressure under very high temperatures. When this happens, residual oil in the turbocharger bearings may coke and cause the turbo to seize. To mitigate this risk, an oil accumulator was installed. This device features a pressurized nitrogen-filled bladder and stores engine oil upon startup. If engine oil pressure drops, the accumulator releases its stored oil into the turbo oil loop cooling the turbo bearings after an abrupt shutdown. The accumulator is shown in Figure 46.

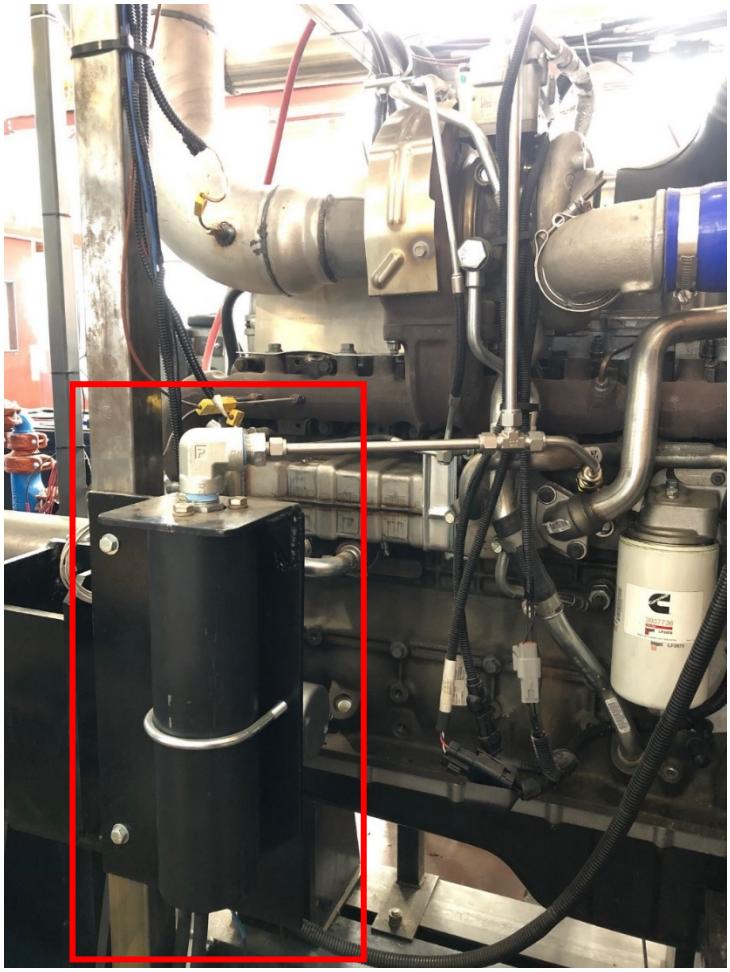


Figure 46. Turbocharger Oil Accumulator

For any other potentially damaging event, the emergency stop (E-stop) switch in Figure 47 was installed at the test stand control computer. It is a locking switch that cuts ignition power to the engine, but not battery power so that data may continue to be recorded after the engine is stopped.

Figure 47. E-Stop Switch

2.3.6 Hard Braking Event

Requirement: R4 - Simulate hard braking

With the engine configured, the hard braking event was tested on the engine test stand. With the engine operating at 1382 RPM and 53 kW, the PGN0 derate message, defined in requirement R4.3 and Section 1.2.4, was sent to the ECM via its stock diagnostic CAN bus simulating the behavior of a truck's brake controller. Figure 48 shows the engine speed trace: after the PGN0

message was sent, the engine ignored the current APP and dropped to idle. This test was repeated four times, and video of the engine is available.

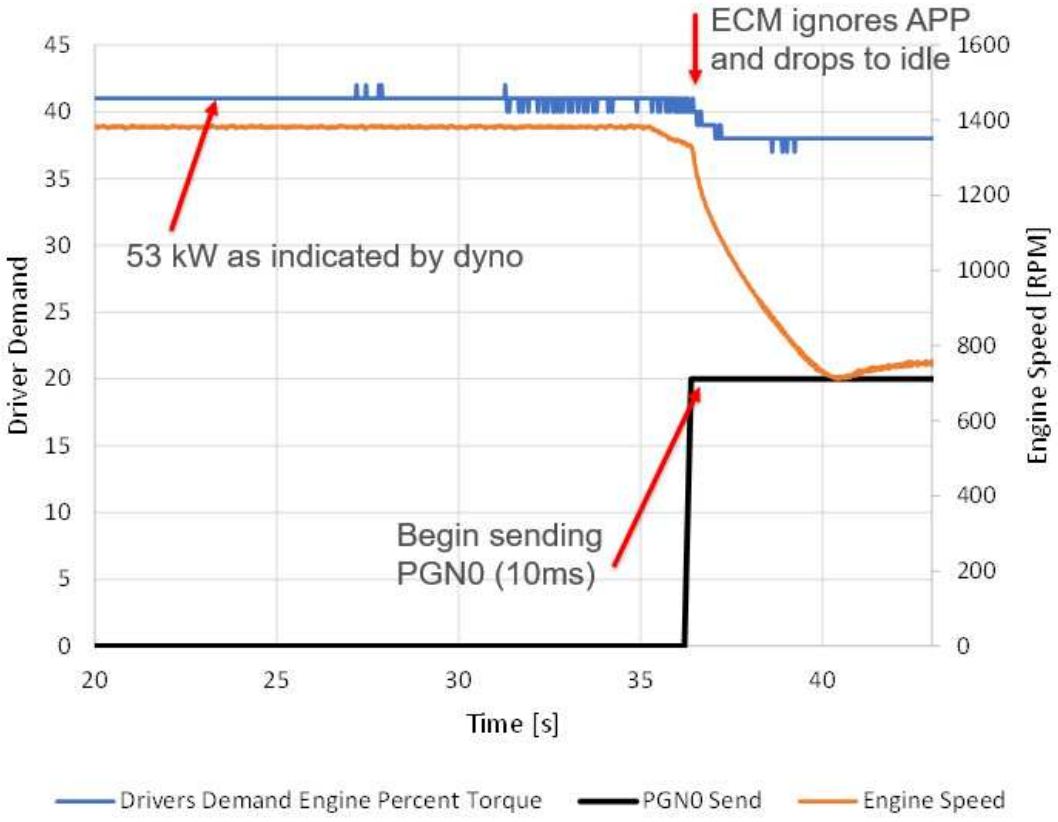


Figure 48. Hard Brake Event Data

To evaluate the reliability of the hard brake event, four repeated tests were conducted. The engine started each test at the same speed and power conditions, and then the PGN0 derate message was sent. Figure 49 shows traces of engine speed and engine torque demand. Engine speed ramped consistently to idle after it received the PGN0 message, also indicating consistent behavior of the dynamometer. Torque demand stayed consistent, but it dropped to zero at different times after the engine had reached idle. Torque demand dropped slightly when PNG0 was sent. This was due to the fact that the parameter "Drivers Demand Engine Percent Torque" as defined in J1939 took into account the engine's torque curve and reports the percent torque

demand as a function of engine speed. Because engine speed dropped to idle, and despite a consistent APP command, this value dropped slightly as it moved to a different engine speed in its map. While this was inconsequential for the purposes of this project, these repeated tests did show that the engine physically behaved in a repeatable manner during the hard brake event.

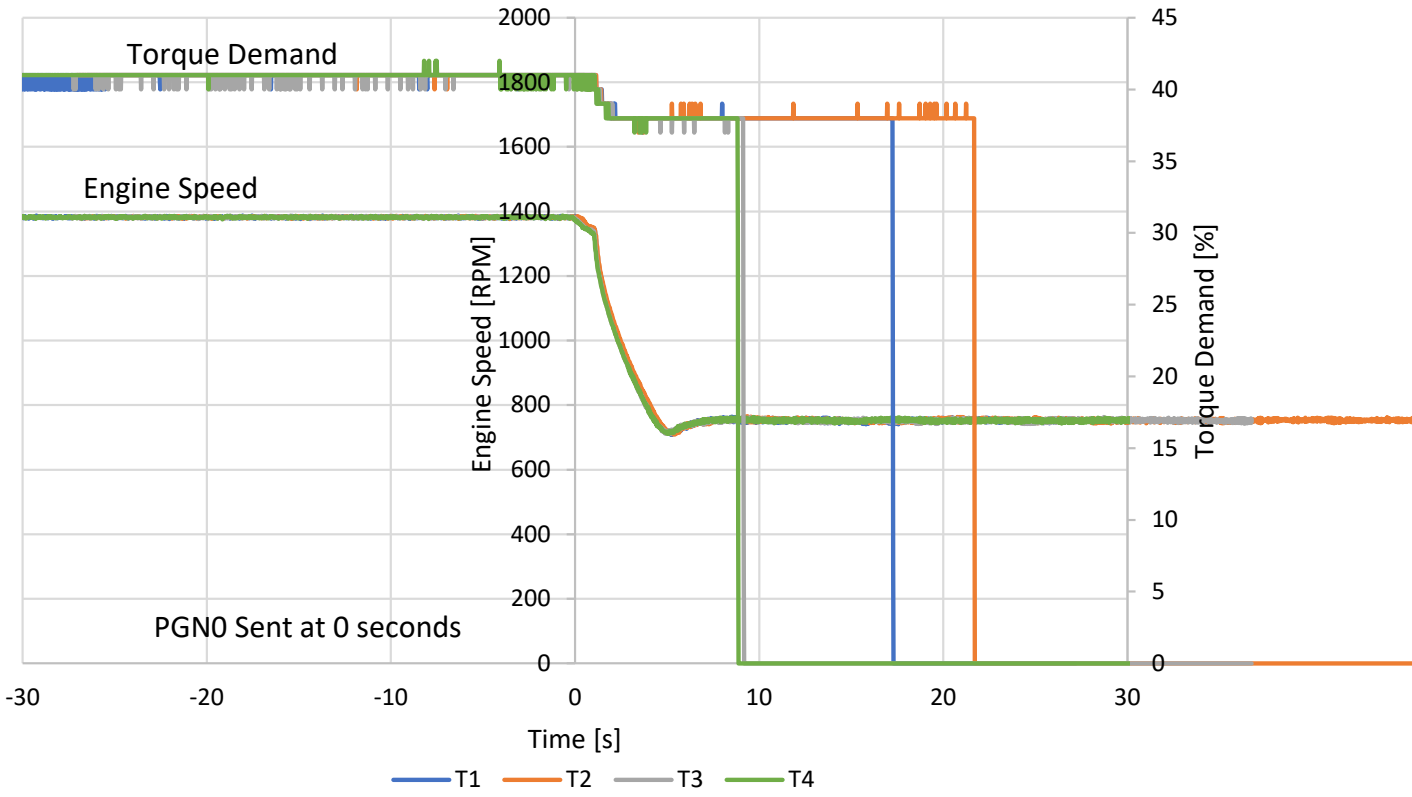


Figure 49. Hard Brake Event Repeatability

2.3.7 Verifying Engine Baseline

Requirement: R2.2 - Generate rated power

Requirement: R2.3 - Generate rated torque

Requirement: R2.4 - Generate rated power & torque with no fault codes

The engine baseline with a stock calibration was validated by running a modified EPA Part 1039 "8-Mode Test Cycle for Variable-Speed Engines" shown in Table 15 [21]. The cycle was modified by operating Mode 4 at 35% torque instead of the recommended 10% because the torque produced by the engine at 10% torque was less than hydrokinetic losses from the eddy-current dynamometer and could not maintain speed. The engine produced slightly more torque than dynamometer hydrokinetic losses at 35% and maintained speed.

Table 15. EPA 8-Mode Test Cycle

Mode Number	Engine Speed	% Torque Scheduled	% Torque Modified	Time in Mode
	RPM	%	%	min
1	2500	100	100	5
2	2500	75	75	5
3	2500	50	50	5
4	2500	10	35	5
5	1500	100	100	5
6	1500	75	75	5
7	1500	50	50	5
8	750	0	5	5

Figure 50 shows the engine power, speed and APP of the 8-mode test as indicated on the INSITE scan tool. Rated power on the engine's nameplate was 224 kW (Figure 51), and peak power in Mode 1 was 223 kW. No fault codes were present during this test cycle.

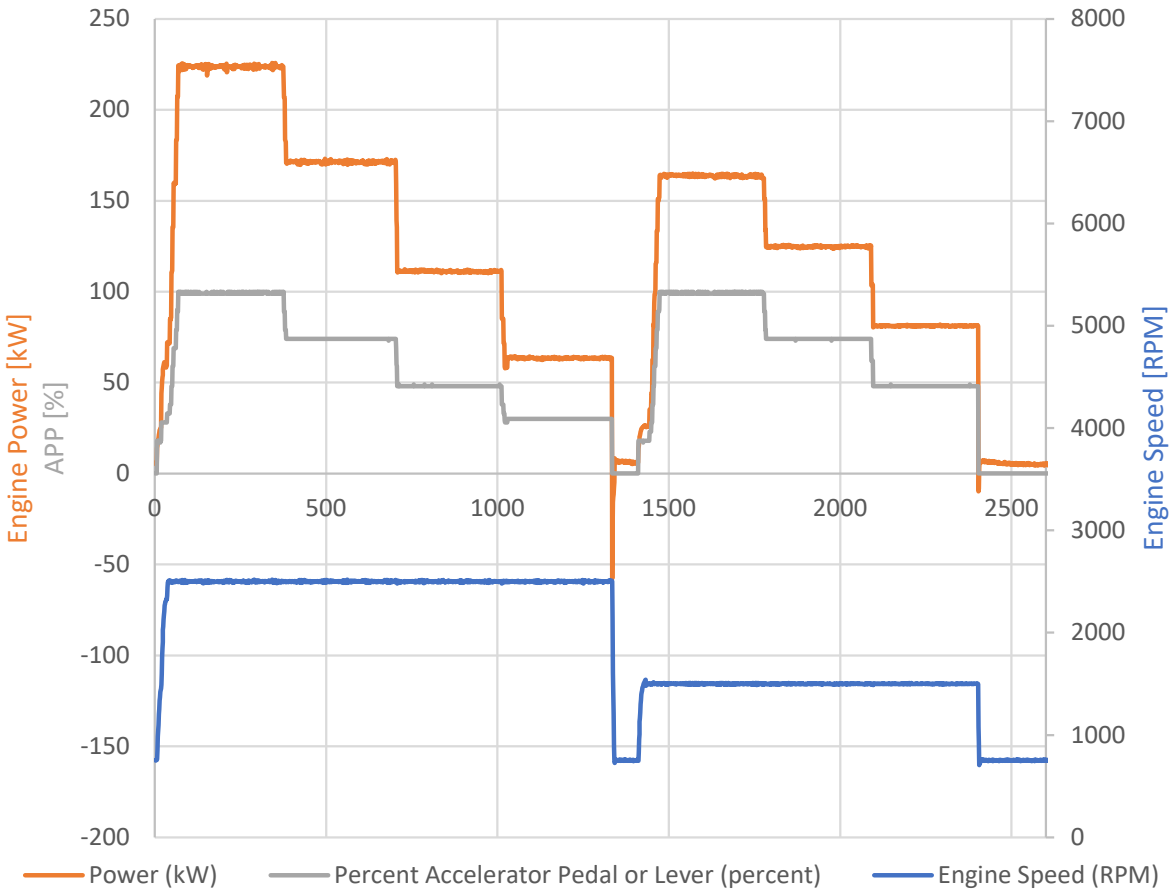


Figure 50. Baseline 8-Mode Power

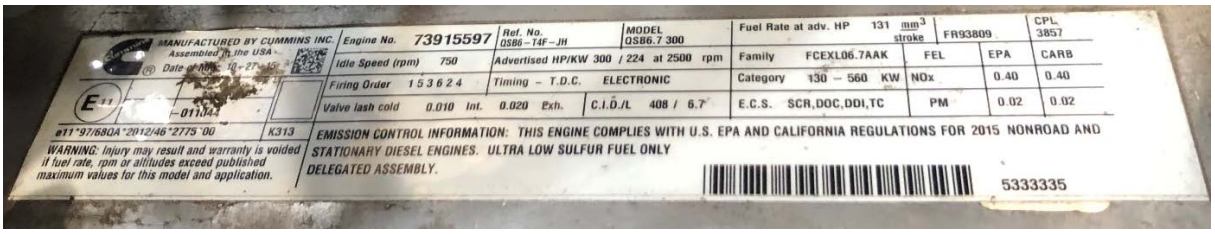


Figure 51. Engine Nameplate – Baseline Configuration

Figure 52 shows the engine torque, speed and APP during the 8-mode test as indicated on the INSITE scan tool. The stock calibration rates engine torque at 1044 Nm; 1044 Nm was indicated on the INSITE scan tool in Mode 5, and the dynamometer controller indicated 1005 Nm at the dynamometer load cell.

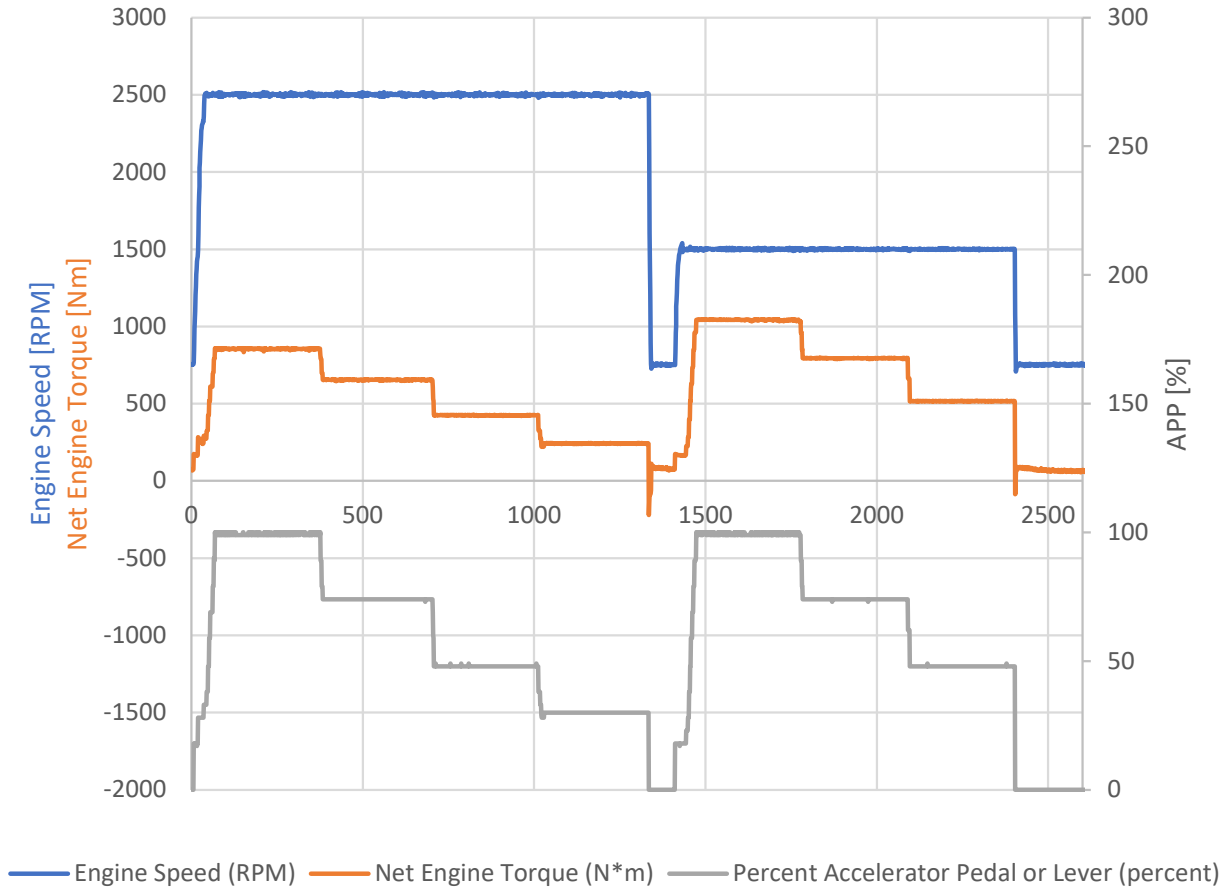


Figure 52. Baseline 8-Mode Torque

Other engine parameters were checked as an indicator of general engine health. Figure 53 shows DOC temperatures during the 8-mode test. As would be expected, modes 1 and 5 showed slightly higher inlet than outlet temperature where temperatures were increasing. An indicator of engine problems would be higher outlet than inlet temperature during non-regeneration conditions which indicates significant unburned fuel in the exhaust.

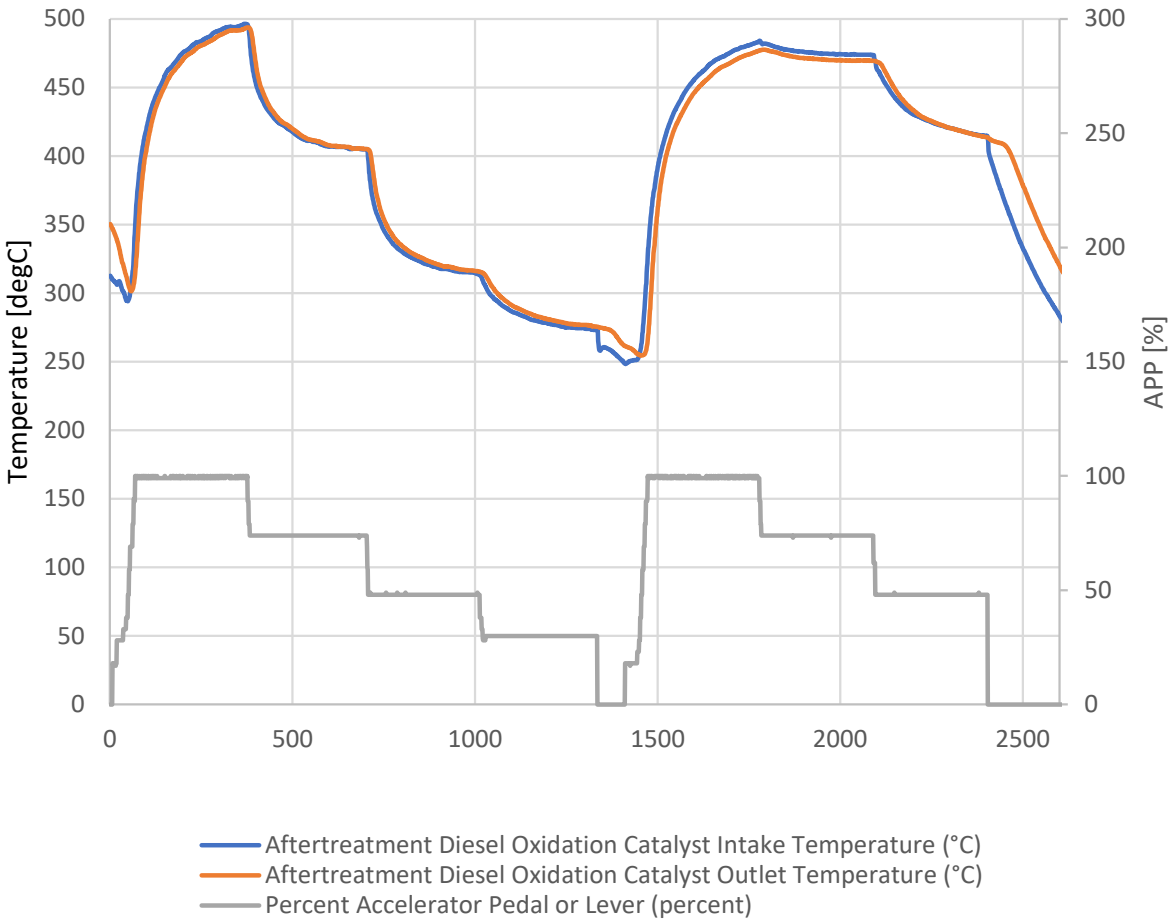


Figure 53. Baseline 8-Mode DOC Temperatures

Figure 54 shows SCR temperatures during the 8-mode test. Similar to DOC temperatures, inlet was higher than outlet temperatures in modes 1 and 5 indicating little exothermic reaction.

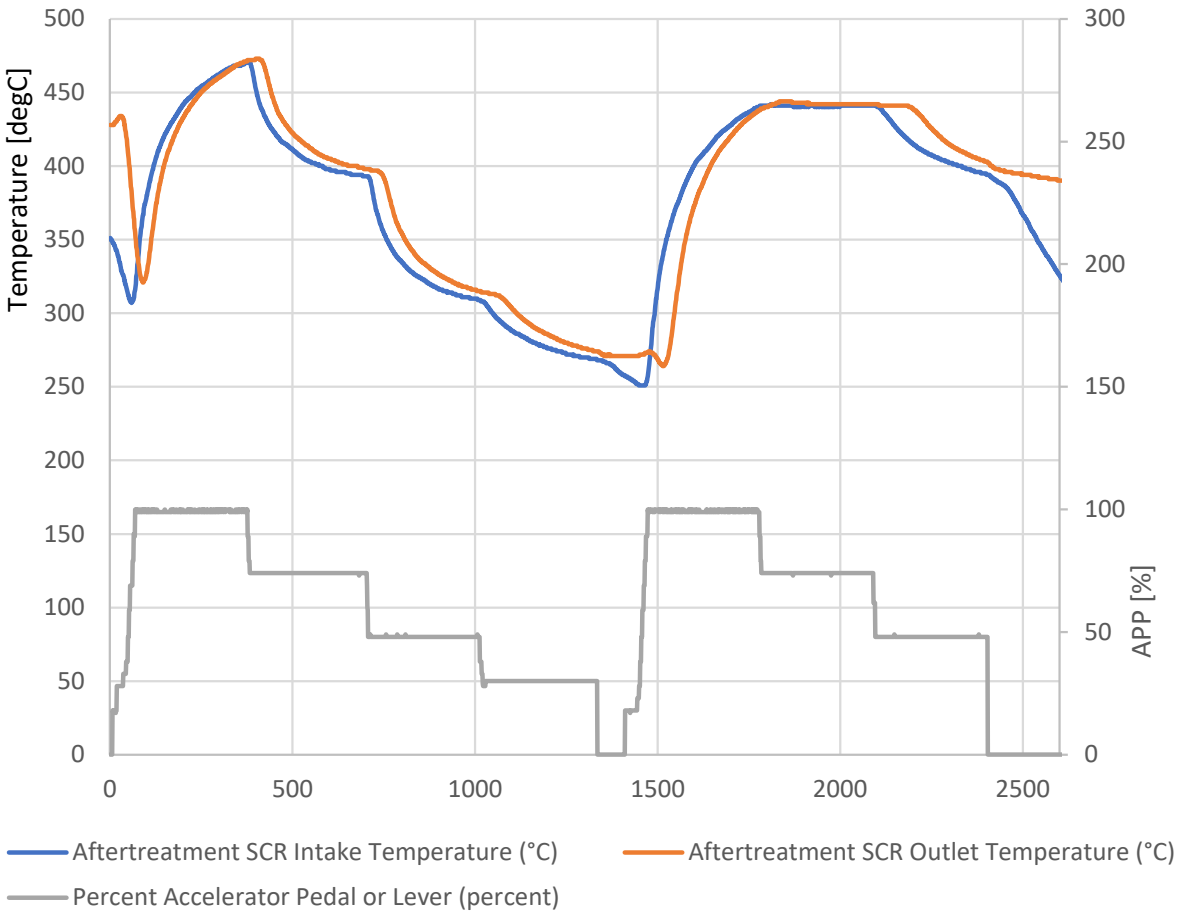


Figure 54. Baseline SCR Temperatures

Figure 55 shows SCR inlet and outlet NO_x during the 8-mode test as indicated by the stock NO_x sensors and INSITE. This was of particular interest due to previously diagnosed problems with the engine's DEF system. Because NO_x is a regulated emission and particularly difficult to control in diesel aftertreatment systems, a non-functioning aftertreatment system causes the engine to derate. Evaluating patched calibrations on a derated engine would not be representative of production heavy-duty vehicles identified as the target industry in the AMP project. Further, patching the ECM's calibration may cause an engine derate. Should this happen, the cause of the derate must be attributed to the patched calibration rather than the test stand as described in Section 2.2.2: Repairing Engine.

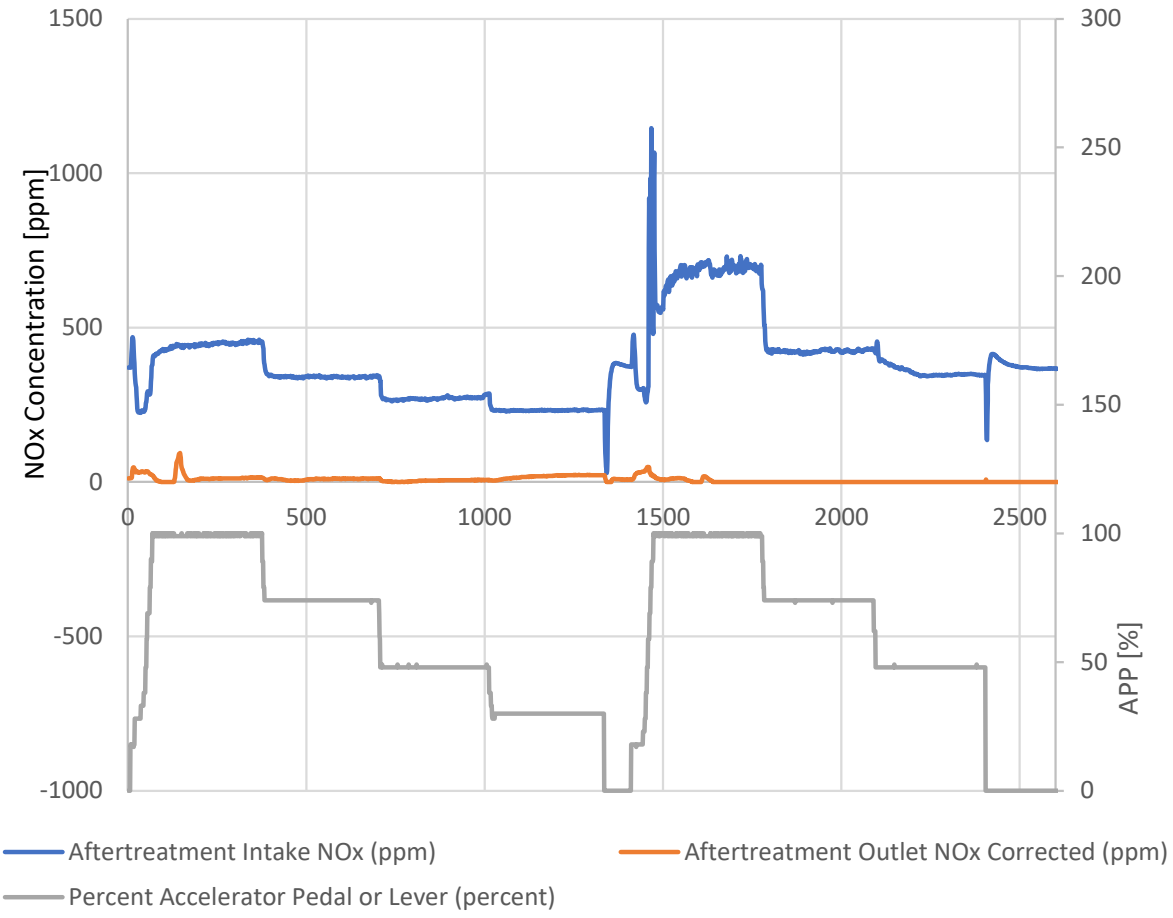


Figure 55. Baseline 8-Mode NO_x

Table 16 shows NO_x conversion efficiency from the data in Figure 55 as a quick reference for SCR operation. Conversion efficiency was calculated by assuming inlet and outlet flow rate was the same, and then averaging the NO_x concentrations during each steady-state mode, excluding transient conditions between modes. NO_x was reduced significantly with SCR conversion efficiencies between 90.5% and 100% (100% really means outlet NO_x concentration was within the uncertainty of the NO_x sensor). This indicated that the aftertreatment system's NO_x reduction was functioning properly.

Table 16. NO_x Conversion Efficiency

Mode	SCR Inlet NO_x [ppm]	SCR Outlet [ppm]	Conversion Efficiency
1	452.1	13.0	0.9712
2	340.1	10.3	0.9698
3	274.1	6.3	0.9769
4	232.5	22.2	0.9047
5	695.4	0.0	1.0000
6	425.6	0.0	1.0000
7	347.9	0.0	1.0000
8	366.2	4.1	0.9887

In addition to checking general engine health, the 8-mode test provided a way to stress test the dynamometer and plant. While the engine was rated at 224 kW (300hp), the dynamometer was rated at 1341 kW (1000hp, Figure 56) so cooling issues were unlikely provided sufficient coolant flow. However, it is engineering best practice to perform a stress test and validate the system before commissioning to uncover any unforeseeable problems.

While dynamometer coolant flow from the plant had to be adjusted after sharp changes in ambient temperature, nothing indicated cooling problems with the dynamometer. Throughout the 8-mode test, the highest dynamometer outlet coolant temperature was 47°C as indicated on the test cell controller during mode 1. The upper limit recommended was 55°C.

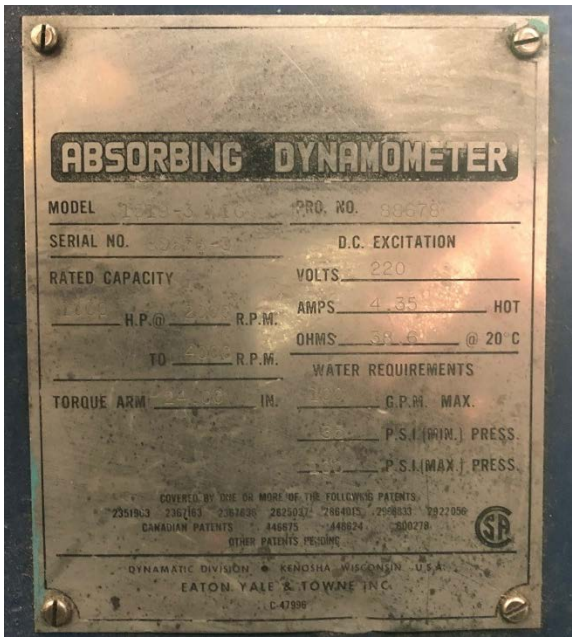


Figure 56. Dynamometer Nameplate

3.0 RESULTS

This section describes system validation. A candidate patching challenge was conducted in which the ECM froze from running a non-convergent iterative calculation, and the system's behavior was described.

3.1 Validation Use Case: Solving Kepler's Law

Requirement: R6 – Detect Effects of a Patch

The test stand was validated by running it through an actual use case scenario of introducing an error into the ECM's binary and evaluating a patch for that error. This is the primary use case expected of this test stand and will be its main purpose when it is transitioned to the AMP project. The general process for this use case is:

Step 1: CSU extracts the target binary from the ECM

Step 2: CSU sends that binary to performers

Step 3: performers decompile, patch, and recompile binary

Step 4: performers return binary, CSU flashes it onto ECM

Step 5: CSU runs an engine cycle to determine if the engine works correctly with the patched binary

The ECM micropatch introduced is an algorithm to solve Kepler's law on the ECM with the engine running.

3.2 Premise of Kepler's Law

The error introduced to the ECM's binary was a numerical methods challenge using Kepler's Law. The NASA Goddard Spaceflight Center, collaborating on the AMP project, discovered a flaw in one of its Magnetospheric Multiscale Mission (MMS) [22] spacecraft in which an "algorithm responsible for producing navigation solutions was trapped in an infinite loop" which solved Kepler's equation [23].

$$M = E - e\ sin(E)$$

Equation (1)

where

M is the mean anomaly

E is the eccentric anomaly

e is eccentricity

This is solved with the Newton-Raphson method where equation 1 is transformed into

$$f(E) = E - e \cdot sin(E) - M$$

Equation (2)

Values of E are tried iteratively until $f = 0$, and e is a constant. The solution should converge until f is less than some tolerance. However, the spacecraft's tolerance was set to the rounding error of floating-point arithmetic, "machine epsilon," and the solution never converged, freezing the computer in an infinite loop.

This scenario was selected as a challenge problem, and Cummins provided an ECM calibration that implemented this calculation and a calibration that implemented this calculation along with a patch to avoid going into an infinite loop. Values for M and e and the instruction to execute the

calculation were provided to the ECM through a CAN message on the ECM's diagnostic port. The specifications of this message are provided in Table 17. *E* and calculation status were returned in another CAN message, Table 18, which was sent continuously by the ECM every 100 ms.

Table 17. Kepler's Law Calculation Request

ArbID	Signals	Bytes	Parameters
15FF14F1	M: Mean Anomaly	1:4	Unsigned Integer Multiplier: 1×10^{-6} Offset: -210 Units: degrees Endian: LSB / Intel
15FF14F1	e: Eccentricity	5:6	Unsigned Integer Multiplier: 0.0015625 Offset: 0 Units: % Endian: LSB / Intel

Table 18. Kepler's Law Calculation Answer

ArbID	Signals	Bytes	Parameters
15FF1500	E: Eccentric Anomaly	1:4	Unsigned Integer Multiplier: 1×10^{-6} Offset: -210 Units: degrees Endian: LSB / Intel
15FF1500	Convergence Status	5	State Encoded 0x0 = Converged within Tolerance 0x1 = Did not converge – exited, exceeded iteration counter 0x2 = Did not converge – exited loop based on time out 0xFE = Error during calculation 0xFF = Convergence status not used
15FF1500	Number of Iterations	6	Unsigned Integer Multiplier
15FF1500	Time Taken to Converge (μs)	7:8	Unsigned Integer Multiplier: 1 Offset: 0 Units: microseconds Endian: LSB / Intel

3.3 Unpatched Calibration Behavior

Requirement - R6.1: Show Expected Behavior of Unpatched Calibration

The first step of implementing this scenario was to show the intended behavior introduced. To do this, a condition was found that causes the ECM to freeze, the ECM was requested to solve Kepler's law with a series of input values of M and e with the engine off, and the ECM response was logged. Eccentricity, e, was chosen as some nominal constant, and mean anomaly, M, was iterated between 0° and 360° in tenths of 360°, or 36° increments. Table 19 summarizes these test cases and whether or not the ECM responded. It was found that a mean anomaly of 324° caused non-convergent behavior which briefly froze the ECM prompting its watchdog timer to reboot the module. The values e = 0.388 and M = 324 were thus used for the next set of evaluations.

Table 19. Kepler's Law Test Cases

	Input											Response
	ArbID	Eccentricity	Angle	B1	B2	B3	B4	B5	B6	B7	B8	
Case #1	15FF14F1	0.388	36	80	A9	A9	0E	D8	00	FF	FF	Y
Case #2	15FF14F1	0.388	72	80	FA	CE	10	D8	00	FF	FF	Y
Case #3	15FF14F1	0.388	108	80	4B	F4	12	D8	00	FF	FF	Y
Case #4	15FF14F1	0.388	144	80	9C	19	15	D8	00	FF	FF	Y
Case #5	15FF14F1	0.388	180	80	ED	3E	17	D8	00	FF	FF	Y
Case #6	15FF14F1	0.388	216	80	3E	64	19	D8	00	FF	FF	Y
Case #7	15FF14F1	0.388	252	80	8F	89	1B	D8	00	FF	FF	Y
Case #8	15FF14F1	0.388	288	80	E0	AE	1D	D8	00	FF	FF	Y
Case #9	15FF14F1	0.388	324	80	31	D4	1F	D8	00	FF	FF	N: ECM resets
Case #10	15FF14F1	0.388	360	80	82	F9	21	D8	00	FF	FF	Y

With the engine idling, the Kepler's Law request CAN message with the selected problem payload was sent to the ECM once using the modified unpatched calibration. The engine

stumbled as the ECM froze briefly and re-set itself. Table 20 shows bus traffic during this event. A 651 ms delay was observed between the time the calculation request was sent (first row) and the time the ECM began sending normal broadcast engine parameters again (last row). During this time, the engine was observed stumbling briefly, and the turbocharger wastegate was audibly observed moving and re-setting as it does during key-on.

Table 20. CAN Traffic After Kepler's Law Calculation

651 ms

Time	Description	ArbID	B1	B2	B3	B4	B5	B6	B7	B8
35.29404	Tx: KeplersLaw	15FF14F1	80	31	D4	1F	D8	0	FF	FF
35.29538	Rx: PGNF004 EngParams	CF00400	70	7D	81	65	17	0	4	82
35.57782	HS CAN $18EAFFFE-boot	18EAFFFE	0	EE	0					
35.65727	HS CAN $18EAFF00	18EAFF00	EB	FE	0					
35.67026	HS CAN $18EEFF00-boot	18EEFF00	D1	51	4B	1	0	0	0	80
35.93447	HS CAN $18FEDF00	18FEDF00	83	E0	2E	7D	FB	FF	FF	F0
35.93507	HS CAN $18F00E00	18F00E00	A0	0F	FF	FF	FF	FF	FF	FF
35.93697	HS CAN $18EAFF00	18EAFF00	DA	FE	0					
35.93981	HS CAN $CF00300	CF00300	D1	0	18	FF	FF	0F	72	7D
35.94461	HS CAN $18FC9600	18FC9600	E2	0	FF	FF	FF	FF	FF	FF
35.94637	Rx: PGNF004 EngParams	CF00400	68	7D	89	8	16	0	4	8A

Engine behavior was recorded from an independent datalogger. Figure 57 illustrates this stumble by showing engine speed, as recorded from the vehicle's diagnostic bus, and the time at which the message was sent with the request to solve Kepler's law. Speed decreased as the ECM briefly shut off and re-set. The 651 ms pause was also seen when fuel injectors stopped firing, and the ECM rebooted.

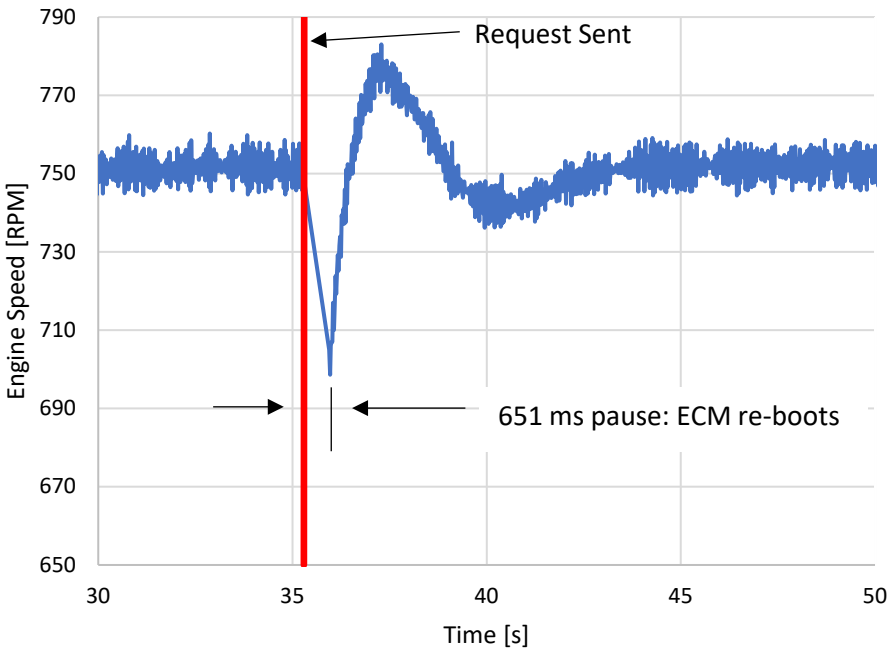


Figure 57. Engine Response to Single Tx: Idle

Based on this 651 ms delay, the engine was stalled by transmitting the Kepler's Law calculation request periodically faster than 651 ms. To investigate engine behavior based on the speed, transmit rate, and calibration patch, the test matrix in Table 21 was performed. This test matrix tested for engine behavior at two different engine speeds and sent the request to solve Kepler's Law as a single transmit or constantly transmit the request at a fixed rate.

Table 21. Kepler's Law Test Matrix

Test	Calibration	Speed	Transmit Rate	Result
1	Unpatched	Idle	Single	Stumble
2	Unpatched	Idle	100ms	Stall
3	Unpatched	1500	Single	Drop to idle, then gradually speed up
4	Unpatched	1500	250ms	Stall

Observations made from this battery of tests were:

- Tests 1 & 3: Transmitting the request to solve Kepler's law faster than 651 ms caused the engine to stall.
- Test 3: At 1500 RPM, transmitting a single request to solve Kepler's law caused an unusual response. The engine stumbled, as it did at idle, but the engine went to an intermediate speed between 1500 RPM and 750 RPM. Despite a constant 20% pedal position sent to the engine throughout the test, driver demand reported by the ECM dropped from 22% to 11%. Figure 58 also shows that engine speed was not consistent. Speed was nominally 800 RPM, and load on the engine was due to normal hydrokinetic losses in the eddy-current dynamometer which are not controlled. This behavior may be exploited for future challenge scenarios but must be explored further as discussed in the Future Work section.

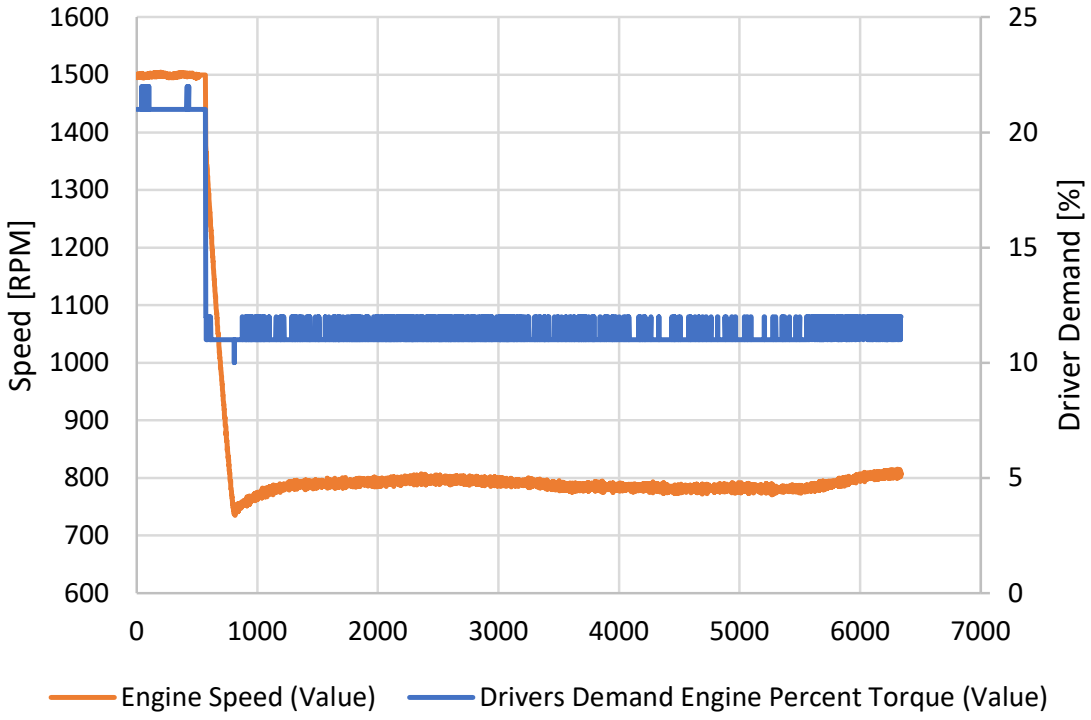


Figure 58. Engine Response to Single Tx: 1500 RPM

From this test, it can be concluded that the engine does behave by either a stumble or stall by being sent requests to solve Kepler's Law due to non-convergence.

3.4 Patched Calibration Behavior

Requirement - R6.2: Show engine runs nominally after being patched

The next step of validation was showing that a patched calibration eliminated the stumble and stall behavior. To do this, the calibration which sets convergence criteria to eliminate non-convergent behavior was flashed onto the engine's ECM. The test matrix in Table 22 was then conducted.

Table 22. Kepler's Law Test Matrix: Patched

Test	Calibration	Speed	Transmit Rate	Result
5	Patched	Idle	Single	No effect
6	Patched	Idle	100ms	No effect
7	Patched	1500	Single	No effect
8	Patched	1500	250ms	No effect

In each of these test cases, no change in engine behavior was observed. Figure 59 shows the worst case test scenario in which the request to solve Kepler's Law was sent every 100 ms with the engine at 1500 RPM. No lag or pauses in CAN traffic were observed, engine speed did not change, and no fault codes were set. It can be concluded that **no change in engine operation occurred while solving Kepler's Law**.

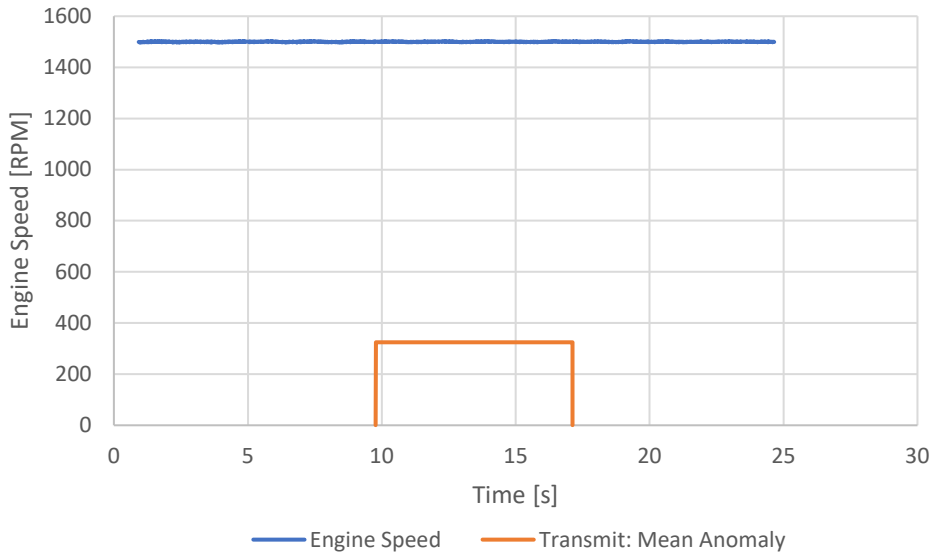


Figure 59. Engine Response to 100ms Tx: 1500 RPM with Patched Calibration

4.0 DISCUSSION

This section discusses how each requirement from Section 1.2 were met. It also provides documentation and instructions as part of the Operations and Transition stage of development.

4.1 Requirements Verification Summary

The test stand will be used to evaluate patched ECM binary files by loading a patched binary and operating the engine to evaluate the effectiveness of the patch. The requirements for this test stand were tabulated in Section 1.2. Table 23 summarizes these requirements and how they were verified.

Table 23. Requirements Verification

	Requirement	**Verification**
R1	Manage ECM Binaries	Section 2.3.4: Extract and write binaries with KTAG
R1.1	Extract ECM Binary	
R1.2	Re-Load ECM Binary	
R2	Nominal Engine Operation	
R2.1	Highway APP Control	Section 2.3.1: APP calibration
R2.2	Generate rated power	Section 3.1.1: Validating Engine Baseline EPA 8-mode test conducted
R2.3	Generate rated torque	
R2.4	Operate at rated power & torque with no fault codes	
R3	No damage to engine or test equipment with faulty ECM patch	Section 2.3.5: Engine Protection Engine overspeed and oil accumulator
R4	Simulate Hard Braking	Section 2.3.6: Hard Braking Event Hard brake simulated on engine dyno
R4.1	Engine Ramp Rate	
R4.2	Engine Derate	
R4.3	Engine Derate message format	
R5	Data Logging	
R5.1	Log all traffic on both engine CAN busses	Partially validated
R5.2	Log dyno speed and torque	Section 2.3.3: Dynamometer Data Acquisition

4.2 Operations & Transition

The last step of the systems engineering Vee diagram is Transition and Operations. This was a particularly important step in development of the engine test stand as it will be handed off to other operators who will use it to evaluate various performers. This section provides operating instructions, maintenance schedule, and documentation for these steps.

4.2.1 Conducting a Test

The process of extracting and re-flashing an ECM binary was described in Section 2.3.4: Binary Data and Appendix C: Procedure for Loading a Calibration onto a CM2350.

Once the ECM binary has been selected and loaded onto a module and a test is ready to run, Table 24 provides a checklist for items to check or turn on before starting the engine.

Table 24. Pre-Test Checklist

	Check building coolant valves are set for the Cummins
	Check oil level
	Check coolant level
	Check for fluid leaks
	Check driveshaft is clear of CAT engine
	Connect INSITE
	Check DEF level
	Check for fault codes | clear
	LabVIEW: Turn on Dyno Return Pump
	LabVIEW: Turn on Dyno Supply Pump
	LabVIEW: Turn on Dyno Cooling Pump
	LabVIEW: Turn on Williams Tower East fan
	LabVIEW: Turn on Williams Tower West fan
	LabVIEW: Check APP is at zero
	LabVIEW: Check thermocouples are reading

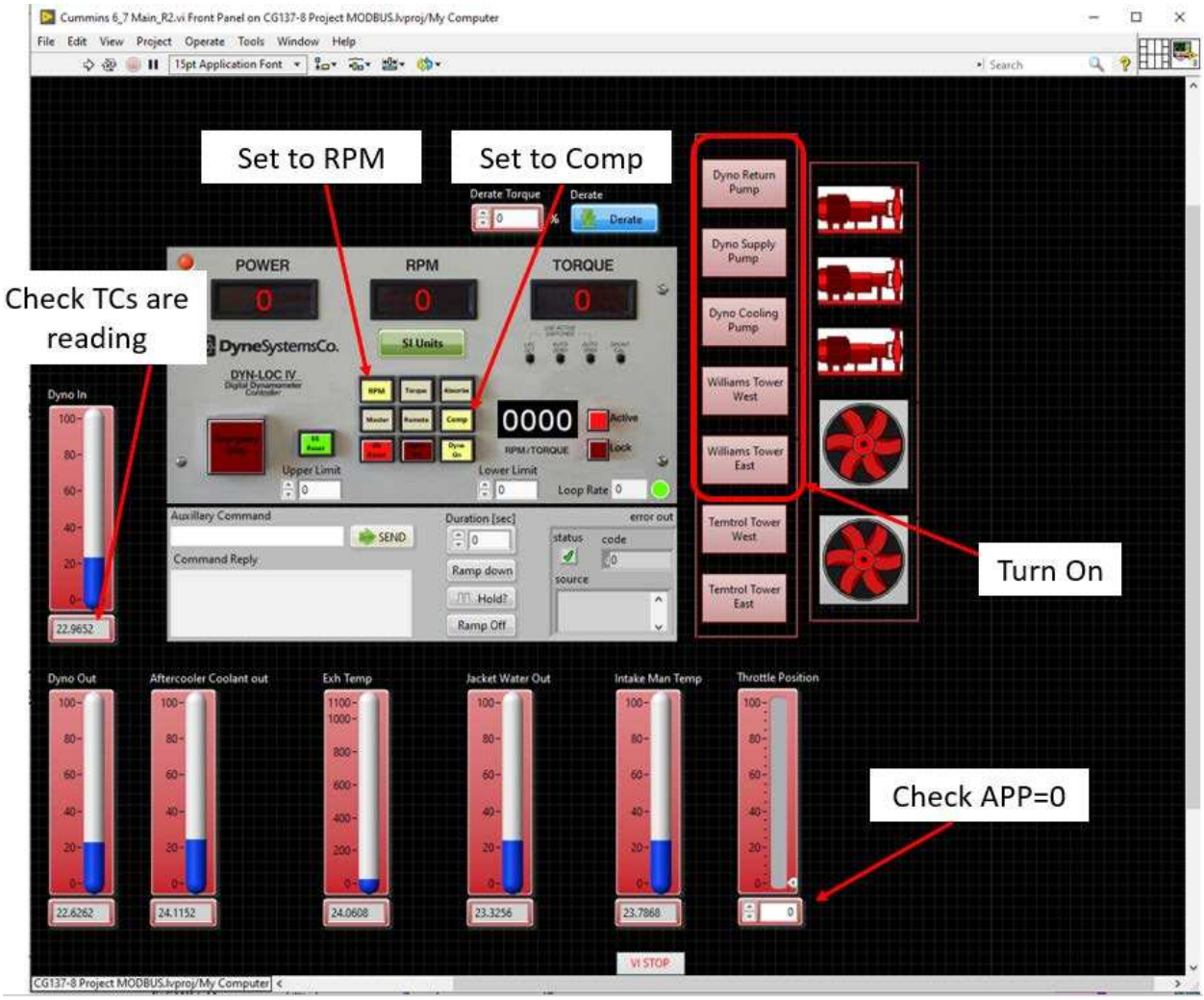


Figure 60. LabVIEW Screen: Pre-Test Checks

To operate the engine, speed is set through the dynamometer controller, and torque is set with the engine's accelerator pedal position. The engine was set up such that the dynamometer controller fixed speed, and the engine's APP fixed torque (the opposite is possible if necessary, but the dynamometer and engine should never both be speed- or torque-controlled). Figure 61 illustrates the LabVIEW control panel. Note that the "dyno off" button should be hit before shutting down the engine; if not, the Dyn-Loc IV controller must be re-set by cycling its main power.

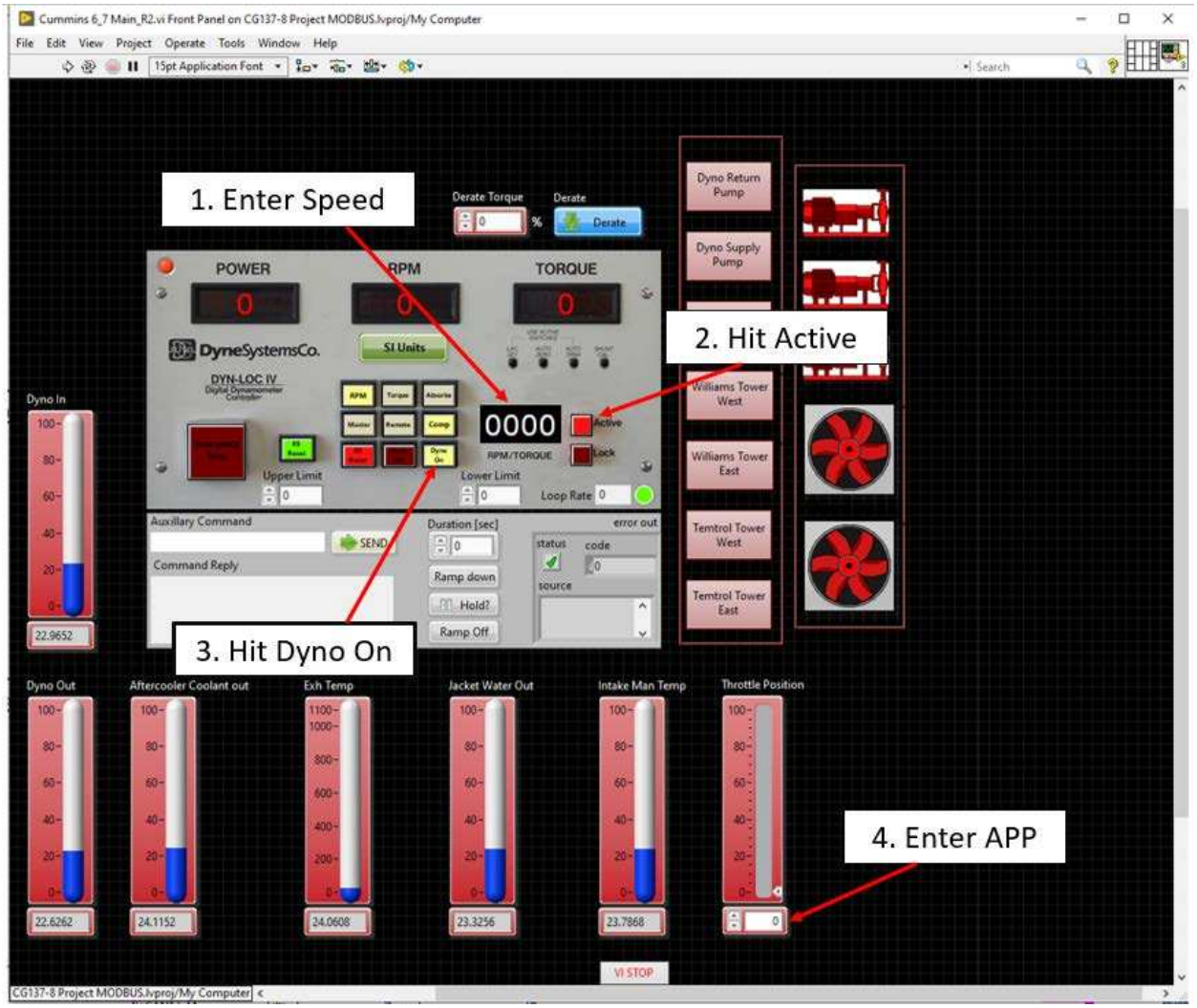


Figure 61. LabVIEW Screen: Operating Engine

4.2.2 Maintenance Schedule

In addition to the maintenance schedule found in the Cummins Operation and Maintenance Manual, the items in Table 25 should be conducted at the listed intervals.

Table 25. Test Cell Maintenance Schedule

In addition to Cummins Maintenance Schedule[1]:	
Yearly	Load cell calibration[2]
Monthly	Check driveshaft bolt torque (20ft-lb)
Daily	Check oil level (dipstick)
	Check coolant level (sight tube)
	Check dyno bearing oil level
	Check fuel level Check DEF level (INSITE)
	Check for fault codes (INSITE)
[1]The complete Cummins maintenance schedule is available for free at https://quickserve.cummins.com/info/index.html using engine serial number 73915597	
[2]The complete load cell calibration procedure is provided in Appendix B.	

4.2.3 Pin-Out Diagrams

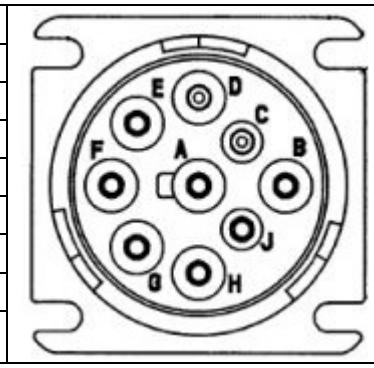
Pin-out diagrams are provided to aid in future troubleshooting or reconfiguring the test stand. Table 26 and Table 27 provide the pin-out information for the two custom connectors installed on the test stand. The complete engine wiring harness diagram is available for free at *https://quickserve.cummins.com/info/index.html* using engine serial number 73915597.

Table 26. 70-pin Crossover Connector Pin-Out

1	15: 12V	29: Gnd	43	57	
2	16: 12V	30: Gnd	44	58	
3	17: 12V	31: Gnd	45	59	
4: 12V +fuse	18: 12V	32: Gnd	46	60	
5	19: 12V	33: Gnd	47	61	
6	20	34: Gnd	48	62	
7	21	35	49	63	
8: CAN 3 high	22: CAN 3 low	36	50	64	
9	23	37: AFT CAN high	51: AFT CAN low	65	
10	24	38	52	66	
11: J1939 high	25: J1939 low	39	53	67	
12	26	40	54	68	
13	27	41	55	69	
14	28: APP1 Gnd	42: APP1 Signal	56: APP2 Gnd	70: APP2 Signal	

Table 27. Round Diagnostic Connector Pin-Out

A	Battery Ground
B	12V
C	J1939 High
D	J1939 Low
E	Shield
F	CAN 3 High
G	CAN 3 Low
H	Aftertreatment CAN High
J	Aftertreatment CAN Low



Power to pin B in on the round diagnostic connector was run through a small rocker switch mounted on top of the test stand's electrical box shown in Figure 62. This is useful for leaving data loggers on for an extended period of time provided external power is supplied to the test stand's batteries.

Figure 62. Diagnostic Port Power Switch

5.0 CONCLUSIONS

The main items addressed in this development project and conclusions are as follows:

1. A systems engineering approach was applied to developing an engine test stand for cybersecurity research
2. An existing engine and dynamometer were repaired, reconfigured and calibrated to meet requirements
3. The test stand was stress tested to uncover potential problems
4. A hard brake event was performed on a real truck and replicated on the test stand
5. A challenge scenario was implemented and evaluated
6. Documentation of the final system was provided for the transition phase

To summarize this project, the original Vee diagram from Section 1 and the process of developing the engine test stand based on the Vee diagram is summarized in Table 28.

Table 28. Project Summary Vee Diagram

Step of Vee Diagram		Step of Vee Diagram	
Concept Development	Develop apparatus to evaluate ECM patches	Design/ Implementation	Hardware fabrication (Section 2.2.1) Engine repairs, exhaust aftertreatment troubleshooting (Section 2.2.2) Control room tether (Section 2.2.3)
Requirements	R5 Data Logging	Integration/Test	Dyno speed & torque, CAN logger installed (Section 2.3.3)
	R1 Extract/Load ECM Binaries		Successfully re-flashed to test Kepler's Law (Section 3)
	R3 Protect engine & test equipment in case of faulty ECM patch		Set dynamometer overspeed Turbo accumulator: conducted hard shut down (Section 2.3.5)
	R4 Simulate hard braking		Tested hard braking event (Section 2.3.6), showed engine ignore APP and derate
	R2 Nominal Engine Operation		Set speed governor & calibrated APP signal (Section 2.3.1) Conducted EPA 8-mode test with configured APP (Section 3.1.1) Faked OEM sensors / eliminated fault codes (Section2.2.2) Measured rated power & torque (Section 3.1.1)
	R6 Detect effects of a patch	Verification/Validation	Tested for engine stumble & stall with ECM solving Kepler's Law Tested for nominal engine operation with patched calibration: engine did not stumble or stall when solving Kepler's Law (Section 3.1.2) Future Work: Verify changes in CPU instructions using Lauterbach
Architecture	Established architecture in SysML diagrams (Section 2.1)	Transition/Operations	Procedures for operating test stand - LabVIEW Operation (Section 4.2.1) - Maintenance Schedule (Section 4.2.2) - Pin-out diagrams (Section 4.2.3) - Load Cell calibration procedure (Appendix B) Ideas for future work (Section 6.0)

6.0 FUTURE WORK

As part of the Transition phase of this system development, the engine test stand developed in this project will be handed off as a tool to be used for different areas of research. This section describes areas of future research with the engine test stand along with potential improvements.

6.1 Validate Effects of a Patch

6.1.1 Instrumentation

The test stand was able to read and write binary calibration files, and its behavior was changed as verified by implementing the Kepler's Law calculation and patch. However, this can be validated further by detecting effects of the patch. To do this, the execution of the binary can be traced with other tools, such as the Lauterbach PowerTrace II. While the KTAG can read and write the static memory of the ECM, the Power Trace II is able to view and record instructions between the static memory, random access memory and processor. This will show if and how the instructions given to the processor have changed as the result of a patch.

The hardware installed on the CM2350 ECMs in Figure 63 includes a Mictor38 connector to allow connection to the Lauterbach. However, operating the Lauterbach requires in-depth knowledge of CPUs and the PowerPC chip installed specifically on the CM2350.

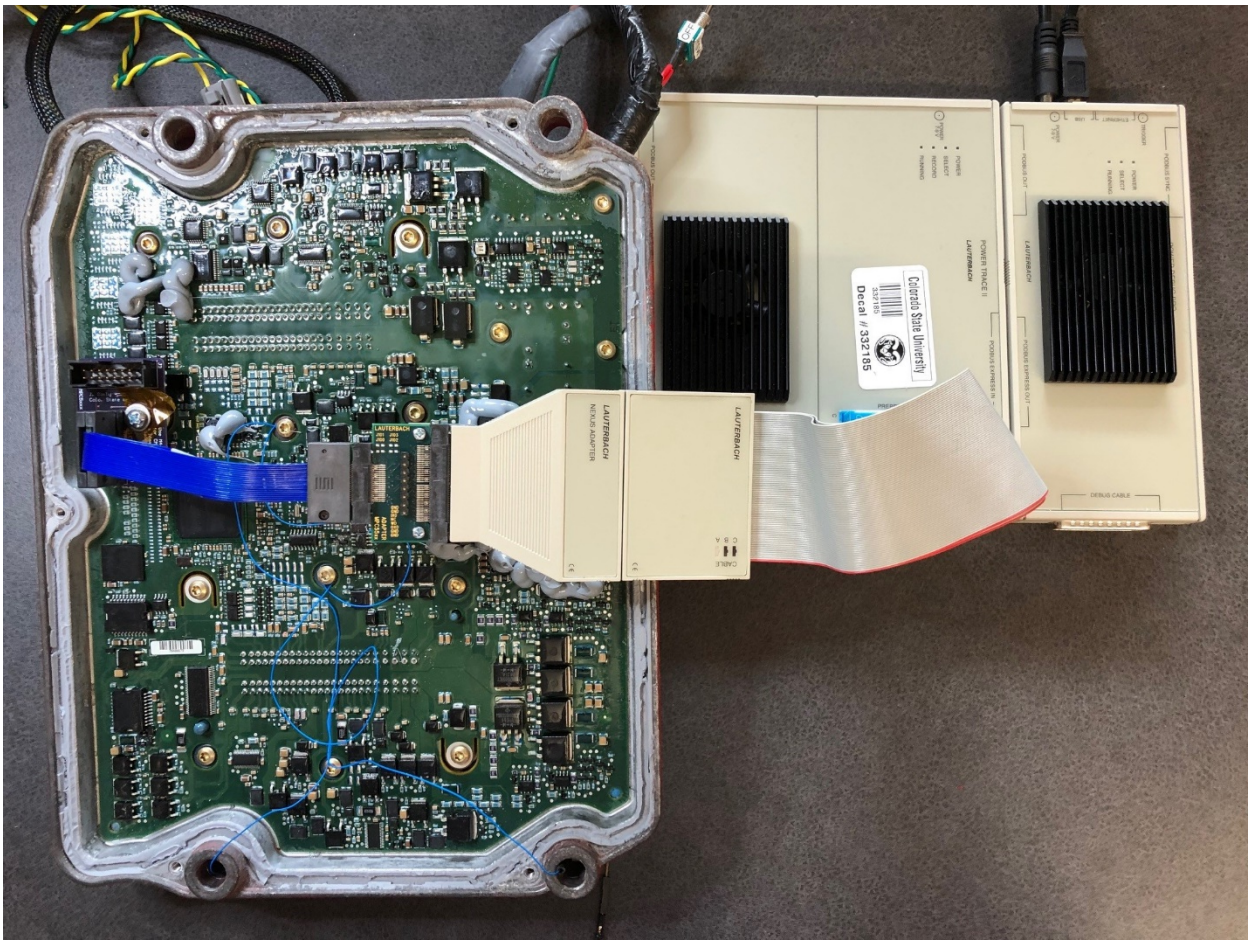


Figure 63. Lauterbach PowerTrace II Connection to CM2350

6.1.2 Validation with Dynamic Time Warping

To validate changes in the ECM binary using the Lauterbach, one area of future work is exploring the stream of CAN data frames coming from both busses of the engine and applying dynamic time warping (DTW) to detect changes between repeated drive cycles. The AMP project involves loading modified binary files onto the ECM and running repeated cycles with the engine as discussed in Section 3.1. Applying dynamic time warping to the data streams

coming from the engine may identify changes in engine behavior as a result of a modified binary file.

Dynamic time warping is an established computational method to determine similarities between two different signals which represent the same underlying phenomenon [24]. It was originally developed for spoken word recognition in the 1970s [25] and has since been applied to genome sequencing, detecting heart disease, and many other fields.

Dynamic time warping may be applied to two repeated data streams from an engine conducting a specified drive cycle using two different ECM binary files. Data streams from an unmodified and modified ECM calibration are two similar signals with a time variance. Applying dynamic time warping, these two data streams would yield a distance score obtained from time variance warping to form confidence bounds about a change in binary data. This would help evaluators identify changes in the data streams due to modifications made to the ECM binary file and understand how and what has changed in the engine calibration.

6.3 CARLA Truck Simulator

Another area of potential future work is to use the test stand as part of a hardware-in-the-loop driving simulator. CSU currently owns a Cascadia truck simulator used for training technicians on the cabin electronics of heavy trucks [26], shown in Figure 64. This simulator includes real production actuators and displays found in a truck such as an accelerator pedal, brake pedal, steering wheel, gear selector, speedometer, tachometer, lights and electrically-driven accessories along with associated electronics. The open-source simulator for autonomous driving research, CARLA, provides a model of vehicle dynamics and a user interface [27]. The Cascadia truck simulator, CARLA, and the engine test stand may be connected to create a hardware-in-the-loop

driving simulator in which a driver can interact with the accelerator and brake pedals of the truck simulator, send pedal commands to the engine test stand and return real-time speed and torque values over an existing dedicated building network. The driver could then see real-time speed of the vehicle as driven by the engine test stand. One potential configuration of this HIL simulator is shown in Figure 65.

Figure 64. Cascadia Truck Simulator

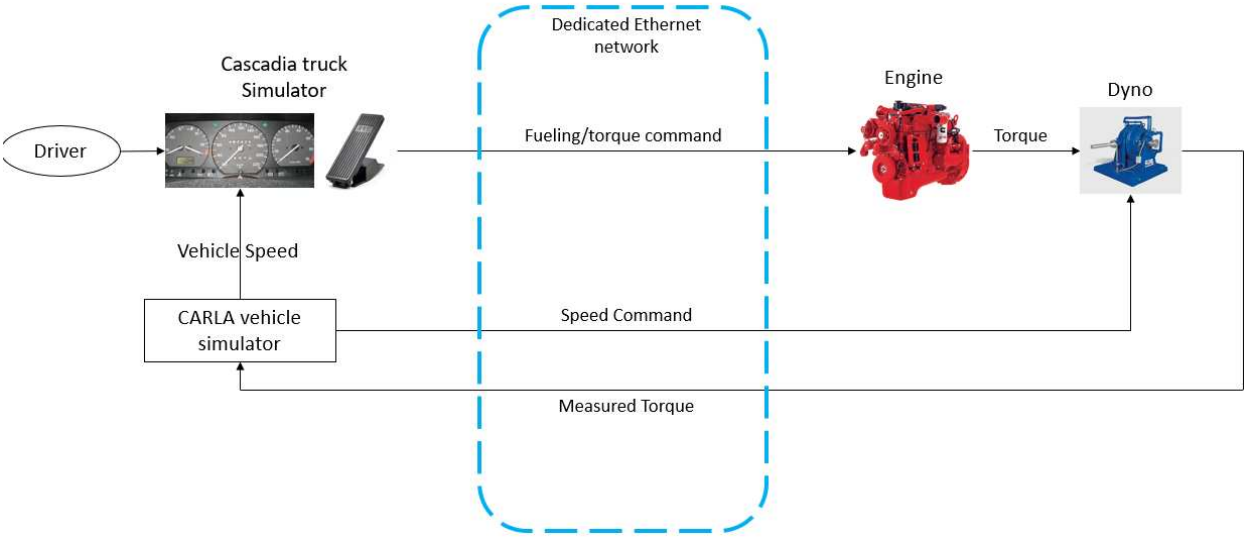


Figure 65. Potential Hardware-In-The-Loop Architecture

6.4 Unusual Behavior Solving Kepler's Law

Section 3.3 described unusual behavior when a single request was sent to the ECM to solve Kepler's Law using a known troublesome input value at 1500 RPM. The engine did not stall or return to its previous operating state. Rather, speed and commanded pedal position went to an intermediate value between idle and 1500 RPM despite a steady commanded APP position.

It was speculated that when the ECM reboots, it maintains the first engine speed it sees when its control functions are running and receives complete crankshaft position data. To test for this, the request for Kepler's law could be sent with and without the water of the dynamometer turned on and no excitation load applied. These will give two different hydrokinetic torques on the engine and cause it to slow down at different rates. Sending the Kepler's law request with dynamometer water turned on should slow the engine more during ECM reboot if this speculation is true.

Another test case involves observing engine behavior with a variable-speed engine governor. Under variable-speed control, the APP commands a set engine speed rather than engine load as described in Section 2.3.1. With this governor type, the engine could be set at a nominal speed, Kepler's law request is sent, and the engine speed response may be observed. Further understanding of these scenarios may be exploited to create new challenge scenarios.

6.5 CAN Logging

An immediate area of future work is to find a solution for logging CAN traffic reliably and automatically storing it in a remote database. A dedicated network with two CAT6 cables was installed between the engine and available servers and may be used to convey data in real-time during engine operation.

6.6 Dynamometer Speed Control for Hard Braking

The ability to control the engine's ramp rate during a simulated hard brake event was a requirement of this project. Future work involves re-configuring the test stand's LabVIEW interface and Dyn-Loc IV controller to create controllable ramp rates. This can be done by sending a continuous speed command to the Dyn-Loc IV controller and verifying that the dynamometer reacts to a new setpoint quickly enough.

REFERENCES

[1] J. Johnson, J. Daily, and A. Kongs, "On the Digital Forensics of Heavy Truck Electronic Control Modules," *SAE Int. J. Commer. Veh.*, vol. 7, no. 1, pp. 72–88, Apr. 2014, doi: 10.4271/2014-01-0495.

[2] D. Van, "SECURE CAN LOGGING AND DATA ANALYSIS," p. 241.

[3] J. Daily, A. Kongs, J. Johnson, and J. Corcega, "Extracting Event Data from Memory Chips within a Detroit Diesel DDEC V," Apr. 2015, pp. 2015-01–1450. doi: 10.4271/2015-01-1450.

[4] ISO and SAE, "Road Vehicles - Cybersecurity Engineering," *SAE Int.*, Sep. 2021.

[5] "Assured Micropatching." https://www.darpa.mil/program/assured-micropatching (accessed Apr. 26, 2022).

[6] J. M. Borky and T. H. Bradley, *Effective Model-Based Systems Engineering*. Cham: Springer International Publishing, 2019. doi: 10.1007/978-3-319-95669-5.

[7] INCOSE, *Systems Engineering Handbook*, Fourth. John Wiley & Sons, 2015.

[8] M. Stuhldreher *et al.*, "Benchmarking a 2016 Honda Civic 1.5-Liter L15B7 Turbocharged Engine and Evaluating the Future Efficiency Potential of Turbocharged Engines," presented at the SAE World Congress Experience, Dec. 2018. doi: https://doi.org/10.4271/2018-01-0319.

[9] J. Kargul *et al.*, "Benchmarking a 2018 Toyota Camry 2.5-Liter Atkinson Cycle Engine with Cooled-EGR," presented at the SAE World Congress Experience, Apr. 2019. doi: https://doi.org/10.4271/2019-01-0249.

[10] "J1939 Digital Annex," presented at the Truck Bus Control and Communications Network Committee.

[11] "Christman Airfield Use," *CSU Drone Center*. https://www.research.colostate.edu/csudronecenter/christman-airfield-use/ (accessed Mar. 24, 2022).

[12] G. V. S. SAE International, "Serial Control and Communications Heavy Duty Network - Top Level Document." SAE International, Aug. 2018. [Online]. Available: https://saemobilus.sae.org/content/J1939_201808/

[13] A. Ragatz and M. Thornton, "Aerodynamic Drag Reduction Technologies Testing of Heavy-Duty Vocational Vehicles and a Dry Van Trailer," NREL/TP--5400-64610, 1330993, Oct. 2016. doi: 10.2172/1330993.

[14] T. Allison, "Allison Transmission 2100/2200 Series." [Online]. Available: https://www.allisontransmission.com/docs/default-source/specification-sheets/int2100_-sa5339(201306).pdf?sfvrsn=817fbf1c_2

[15] “The MICHELIN® X® Multi D Tire | MICHELIN COMMERCIAL TIRES.” https://business.michelinman.com/tires/michelin-x-multi-d (accessed Mar. 17, 2022).

[16] “ISO 11898-2.” https://www.iso.org/obp/ui/#iso:std:iso:11898:-2:ed-2:v1:en (accessed Mar. 22, 2022).

[17] “CAN bus,” *Wikipedia*. Mar. 09, 2022. Accessed: Mar. 22, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=CAN_bus&oldid=1076161431

[18] *40 CFR §1065.310*, vol. 40 CFR §1065.310. 2021. [Online]. Available: https://www.ecfr.gov/current/title-40/chapter-I/subchapter-U/part-1065/subpart-D/subject-group-ECFR1f4576f5ad2e6d5/section-1065.310

[19] *40 CFR §1065.307*, vol. 40 CFR §1065.307. 2021. [Online]. Available: https://www.ecfr.gov/current/title-40/chapter-I/subchapter-U/part-1065/subpart-D/section-1065.307

[20] DyneSystems, Inc., “Dyn-Loc IV User Manual.” Midwest Dynamatic Dynamometers, Jul. 2001. [Online]. Available: www.dynesystems.com

[21] *40 CFR §1039 Appendix II*, vol. 40 CFR §1039 Appendix II. p. (c)(1). [Online]. Available: https://www.ecfr.gov/current/title-40/part-1039/appendix-Appendix%20II%20to%20Part%201039

[22] “Magnetospheric Multiscale About MMS.” https://mms.gsfc.nasa.gov/about_mms.html (accessed Mar. 19, 2022).

[23] G. Cruz-Ortiz and J. Daily, “NASA MMS Patching Story.” https://github.com/SystemsCyber/AMP-Challenge-A-Keplers-Law/blob/main/story.md

[24] “Dynamic time warping,” *Wikipedia*. Feb. 23, 2022. Accessed: Mar. 18, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Dynamic_time_warping&oldid=1073616853

[25] H. Sakoe and S. Chiba, “Dynamic Programming Algorithm Optimization for Spoken Word Recognition,” *IEEE Trans. Acoust. Speed Signal Process. Vol ASSP-26*, Feb. 1978, [Online]. Available: https://ieeexplore-ieee-org.ezproxy2.library.colostate.edu/stamp/stamp.jsp?tp=&arnumber=1163055

[26] “Aftermarket Resource Center.” https://www.dtnaarc.com/daimler/perfCtr/campus/frameset/frameset.jsp (accessed Mar. 18, 2022).

[27] C. Team, “CARLA,” *CARLA Simulator*. http://carla.org// (accessed Mar. 18, 2022).

[28] P. Lobato, “Dyno Load Cell Calibration.xlsx.” https://github.com/SystemsCyber/PowerHouseEngineTesting

APPENDIX A: EXHAUST AFTERTREATMENT CONFIGURATION

At project kick-off, there was some concern about the engine's DPF soot load model confounding the evaluation of performers' calibration patches. Upon initial inspection, the engine's scan tool "INSITE" provided a value for estimated DPF soot load (in grams). On engines equipped with a DPF, this value is based on a model built from (a) binned time at engine speed/load conditions, (b) exhaust temperatures, and (c) fuel consumption which continuously update over time and is typically a reasonable estimate for the actual mass of soot in the DPF. The modeled soot load value is then used to determine when DPF regenerations need to occur to combust and remove the soot.

Because the AMP project involves changing the ECM calibration periodically, there is no guarantee the soot load model in a given calibration accurately reflects the actual level of soot build-up in the DPF. If actual soot load is very little and modeled soot load is very high, the ECM will perform an unnecessary regeneration. If actual soot load is very high and modeled soot load is very low, the ECM will not conduct a regeneration when it is needed, and soot will continue to build. This can lead to high exhaust backpressure or an uncontrolled regeneration, damaging the test stand.

This potential risk had to be addressed. However, it was unclear how this particular engine's aftertreatment system was configured based on available data from Cummins's tech info site. The can where the DPF would normally be located in the aftertreatment system was unusually small to contain a DPF when compared to the same engine installed in the Kenworth T270 and did not have a differential pressure sensor (Figure A-1).

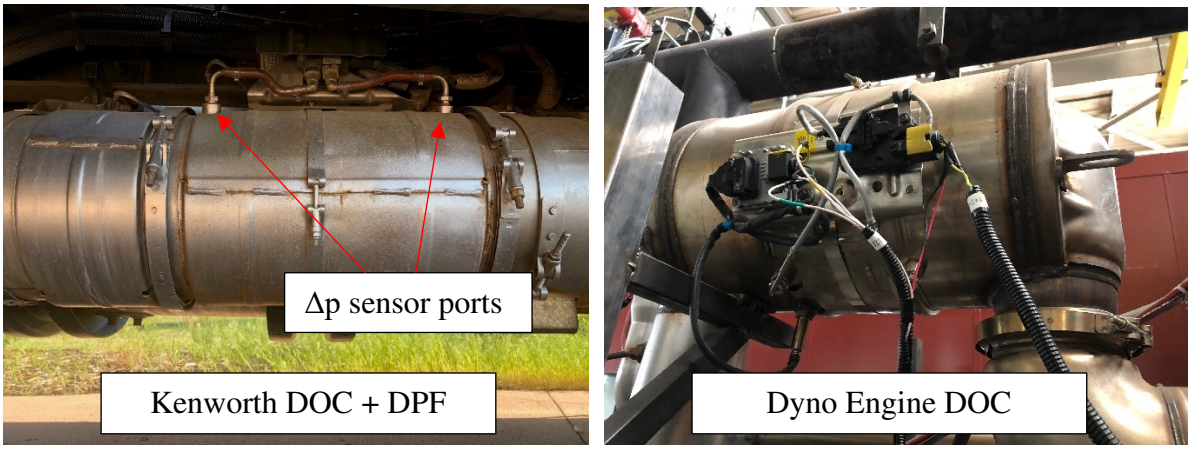


Figure A-1. Upstream Aftertreatment Can on Kenworth & Dynamometer Engines

The cans on the dynamometer engine were viewed with a borescope to verify there was no DPF. To show what a DPF would look like, Figure A-2 shows the basic structure of a typical DPF. Every other channel is plugged on each end of the ceramic brick. This forces exhaust gasses to flow through the porous ceramic. Images from the borescope at two locations in the can showed only a catalyst brick (Figure A-3). The channels were much smaller than those of DPF, and every other channel was not blocked.

Left: https://otomatic.eu/construction-and-principle-of-operation-dpf/
Right: https://www.dpfrestoration.com

Figure A-2. DPF Structure

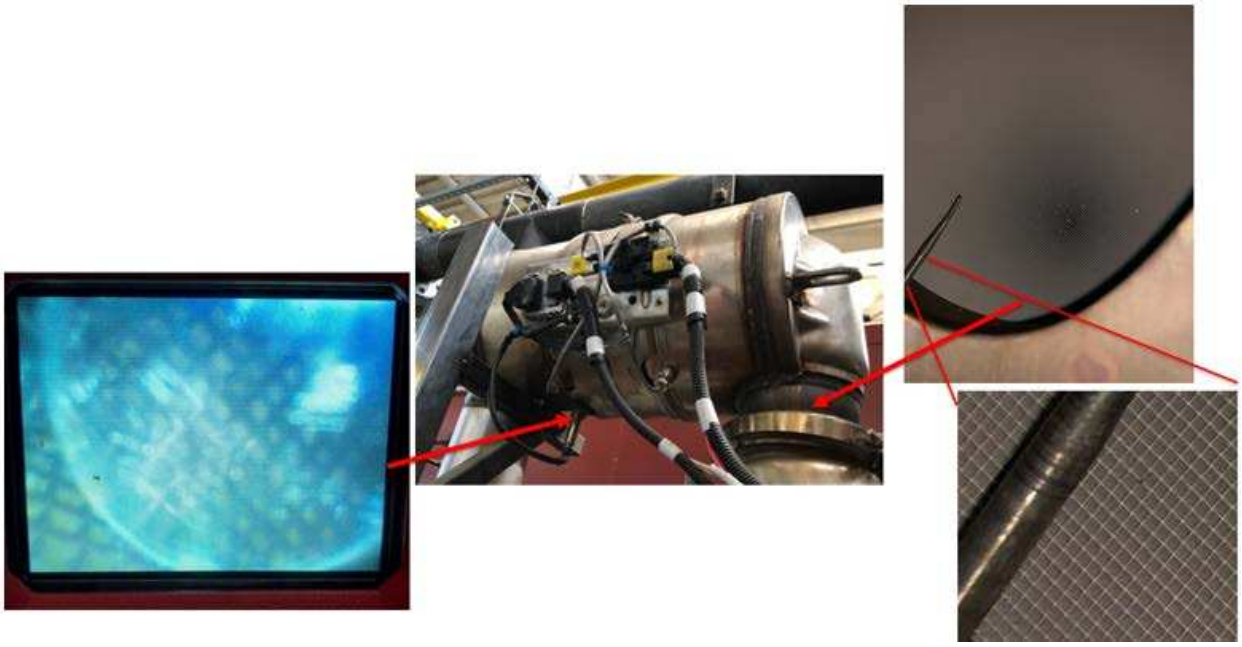


Figure A-3. Dynamometer Engine Aftertreatment Borescope Images

This showed conclusively that the dynamometer engine used for this project did not have a DPF, and the soot load model problem was of no consequence. It was later learned that numerous

parameters, including those for the DPF, in Cummins INSITE are generically populated for many different engine platforms, and components not installed are given a false nominal value.

APPENDIX B: LOAD CELL CALIBRATION PROCEDURE

This procedure instructs an operator how to perform a load cell calibration using dead weights. It is based on CFR §1065.310 Torque Calibration [18] and engineering best practices from a commercial test lab. The spreadsheet used to calculate the 1065 quality metrics is available for future use on Github [28].

First, an appropriate range must be selected for the specific engine, and weights must be located that provide at least six incremental weights which span the selected range roughly evenly. A rule of thumb is the span point should be 150% of the highest torque expected from the engine, and the load cell's range should be taken into account. In the case of the Cummins 6.7L used for this project, a range of 2000 Nm was selected. This equates to 3274 N force at the load cell which has a maximum range of 8896 N (or 2000 lb-f). A combination of the gray 100 lb weights used for the Cooper-Bessemer engine, black 50 lb weights, and the calibration arm of the dynamometer calibration arm and hanger provided the six sufficiently-spaced calibration points shown in Figure B-1.

Figure B-1. Dynamometer Load Cell Dead Weight Configuration

After an appropriate span point and a combination of calibrated dead weights have been selected, the dead weight calibration procedure is as follows:

1. Check there is no force acting on the dynamometer
2. Zero load cell: Hit AUTO ZERO on Dyn-Loc controller (see Figure B-2)
3. Install weight hanger (Figure B-3)
4. Add all weights for full scale
5. At full scale, enter full scale value into Dyn-Loc controller and hit AUTO SPAN (Figure B-2)
6. Remove weights & hanger

7. Record zero point (reading should return to zero; if not, press down and release dynamometer load arm and check for wires or plumbing imparting a torque on the dynamometer casing)
8. Install hanger
9. Sequentially add weights, recording weight and load cell output.
10. Sequentially remove weights, recording weight and load cell output
11. Repeat in opposite direction (compression or tension)

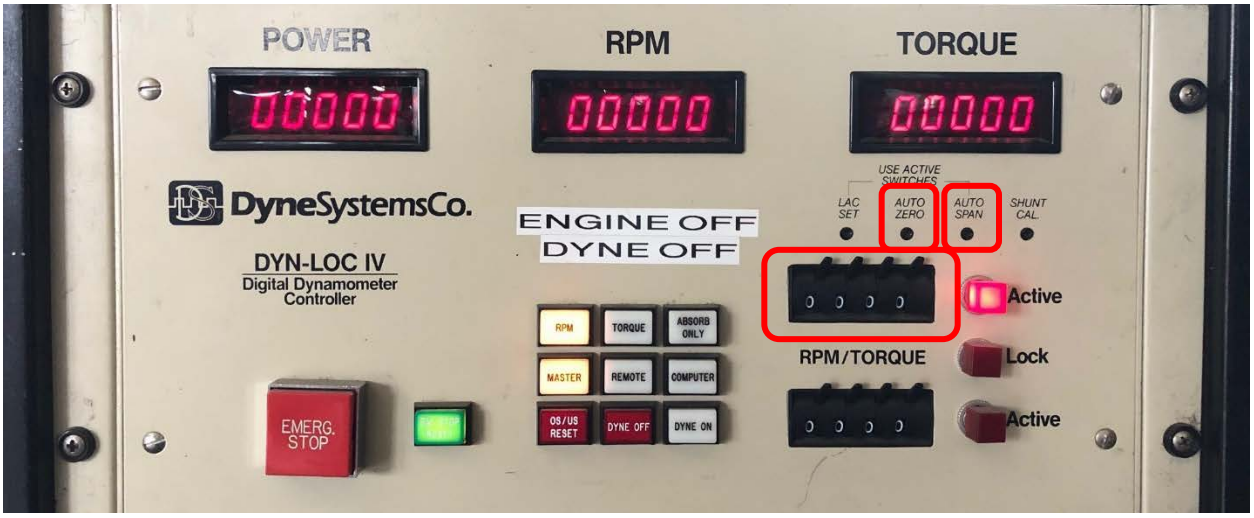


Figure B-2. Dyn-Loc IV Controller

Figure B-3. Dead Weight Hanger and Moment Arm

With the data collected, the values in Table 1 of 40 CFR §1065.307 should be calculated and compared to specifications. Additionally, maximum linearization error and hysteresis should be checked.

Table B-1. Table 1 of §1065.307

TABLE 1 OF §1065.307—MEASUREMENT SYSTEMS THAT REQUIRE LINEARITY VERIFICATION

Measurement system	Quantity	Linearity criteria			
		$\lvert x_{min}(a_1-1) + a_0 \rvert$	a_1	*SEE*	r^2
Speed	f_n	≤0.05% · f_{nmax}	0.98-1.02	≤2% · f_{nmax}	≥0.990
Torque	T	≤1% · T_{max}	0.98-1.02	≤2% · T_{max}	≥0.990
Electrical power	P	≤1% · P_{max}	0.98-1.02	≤2% · P_{max}	≥0.990
Current	I	≤1% · I_{max}	0.98-1.02	≤2% · I_{max}	≥0.990
Voltage	U	≤1% · U_{max}	0.98-1.02	≤2% · U_{max}	≥0.990
Fuel flow rate	$\dot{m}$	≤1% · $\dot{m}_{max}$	0.98-1.02	≤2% · $\dot{m}_{max}$	≥0.990
Intake-air flow rate[1]	$\dot{n}$	≤1% · $\dot{n}_{max}$	0.98-1.02	≤2% · $\dot{n}_{max}$	≥0.990
Dilution air flow rate[1]	$\dot{n}$	≤1% · $\dot{n}_{max}$	0.98-1.02	≤2% · $\dot{n}_{max}$	≥0.990
Diluted exhaust flow rate[1]	$\dot{n}$	≤1% · $\dot{n}_{max}$	0.98-1.02	≤2% · $\dot{n}_{max}$	≥0.990
Raw exhaust flow rate[1]	$\dot{n}$	≤1% · $\dot{n}_{max}$	0.98-1.02	≤2% · $\dot{n}_{max}$	≥0.990
Batch sampler flow rates[1]	$\dot{n}$	≤1% · $\dot{n}_{max}$	0.98-1.02	≤2% · $\dot{n}_{max}$	≥0.990
Gas dividers	x/x_{span}	≤0.5% · x_{max}/x_{span}	0.98-1.02	≤2% · x_{max}/x_{span}	≥0.990
Gas analyzers for laboratory testing	x	≤0.5% · x_{max}	0.99-1.01	≤1% · x_{max}	≥0.998
Gas analyzers for field testing	x	≤1% · x_{max}	0.99-1.01	≤1% · x_{max}	≥0.998
PM balance	m	≤1% · m_{max}	0.99-1.01	≤1% · m_{max}	≥0.998
Pressures	p	≤1% · p_{max}	0.99-1.01	≤1% · p_{max}	≥0.998
Dewpoint for intake air, PM-stabilization and balance environments	T_{dew}	≤0.5% · T_{dewmax}	0.99-1.01	≤0.5% · T_{dewmax}	≥0.998
Other dewpoint measurements	T_{dew}	≤1% · T_{dewmax}-	0.99-1.01	≤1% · T_{dewmax}-	≥0.998
Analog-to-digital conversion of temperature signals	T	≤1% · T_{max}	0.99-1.01	≤1% · T_{max}	≥0.998

APPENDIX C: PROCEDURE FOR LOADING A CALIBRATION ONTO A CM2350

There are two ways to re-flash a calibration onto a CM2350: using Cummins INSITE and using the KTAG. Re-flashing using the KTAG was described in Section 2.3.4, and this Appendix describes re-flashing a calibration using the Cummins INSITE tool.

Cummins INSITE

This procedure is to re-flash the calibration of a CM2350 engine control module. The intent is to use the calibration on the module, make modifications to it, and re-load it onto the ECM.

This can be done either on a bench setup or on the engine. If done on the engine, install a 12V power supply or charger on the battery: the module can be rendered inoperable if voltage falls too low during re-flash.

1. Write down or take a picture of the ECM part number and code printed on the ECM casing.

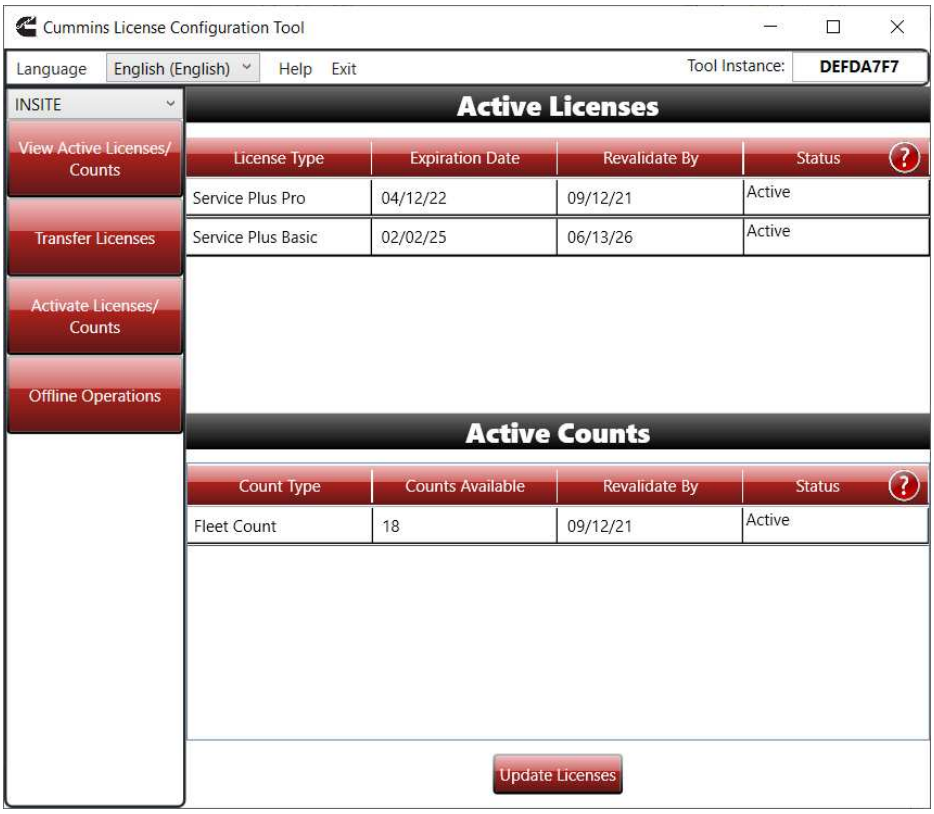
   There are many different part numbers for the CM2350 series of ECMs, and each is only compatible with certain calibrations. These largely differ by engine type: some Cummins X15 models use a different ECM part number than a Cummins QSB6.7, and the calibrations for those two engines cannot be swapped between ECMs despite both being CM2350s. The ECM Part Number for the dynamometer engine used for this project is 5317106.

2. Plug in DPA5 connector to J1939 diagnostic port, key-on engine.

3. On laptop, open Cummins License Configuration Tool 

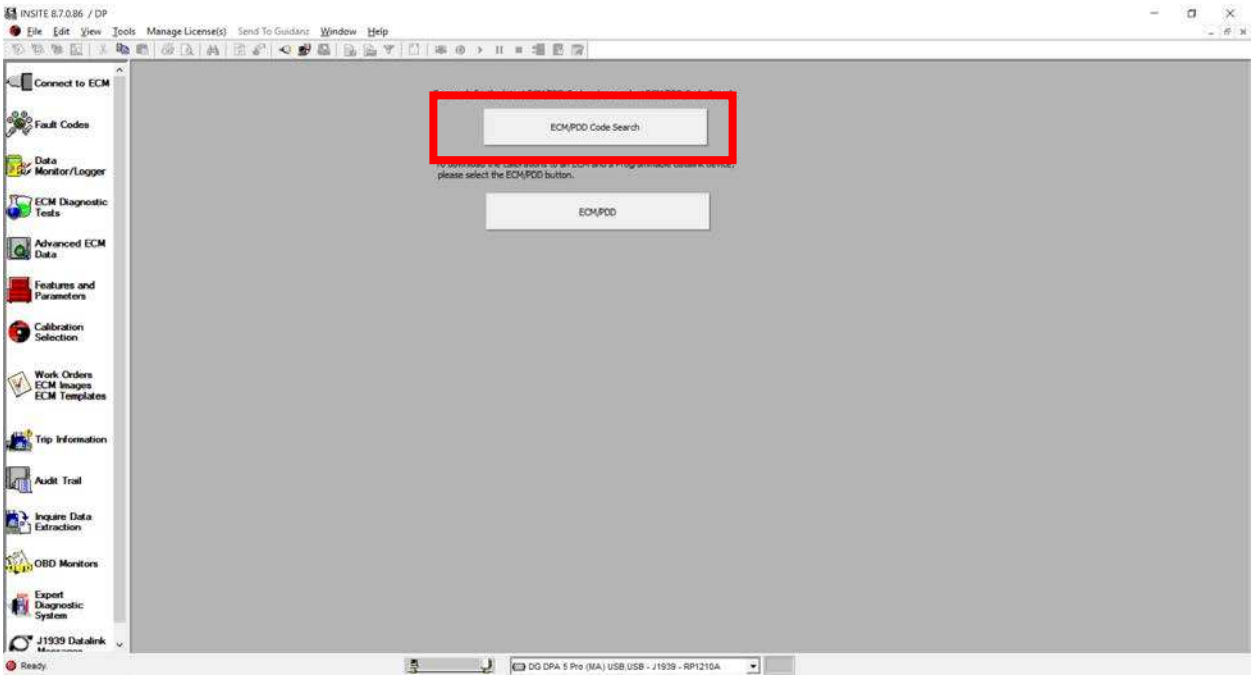
   Hit “Update Licenses”

   This must be done once per day, but not every time INSITE is opened.

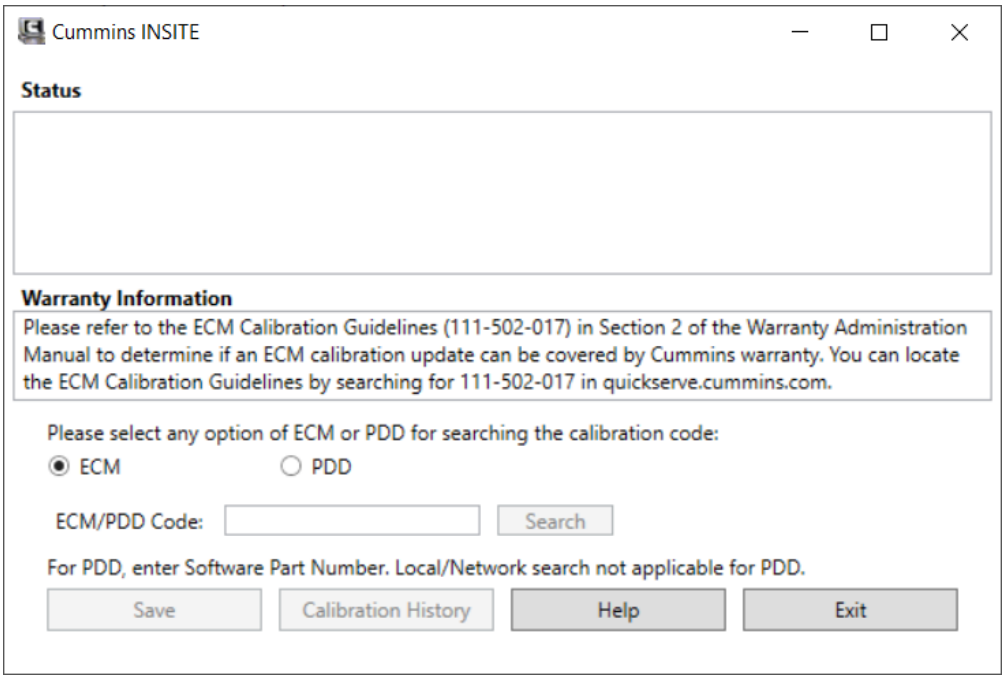


4. Open Cummins INSITE 

   If downloading the most recent calibration from Cummins, do not connect to ECM and hit “Calibration Selection.” The following screen should come up. Hit “ECM/PDD Code Search”



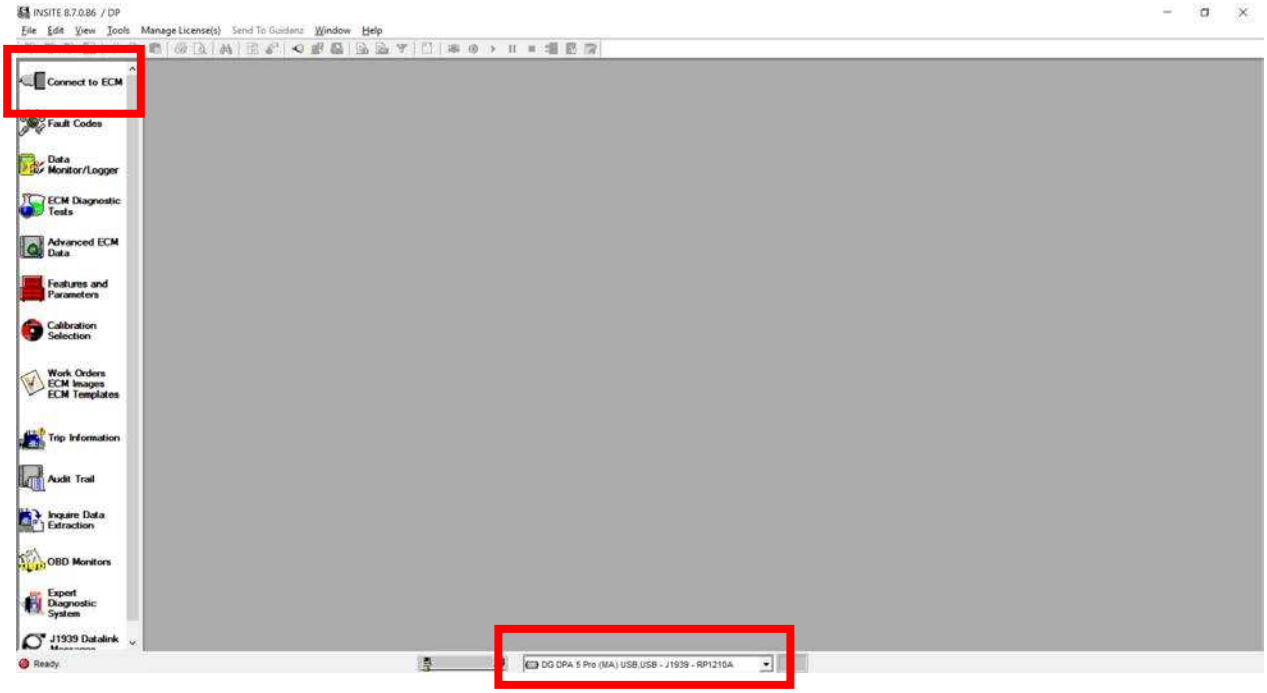
Type in the desired ECM code. (for the dynamometer engine, this is ER80001; for the Kenworth, this is DT90019). Hit “Search”. Save the resulting calibration.



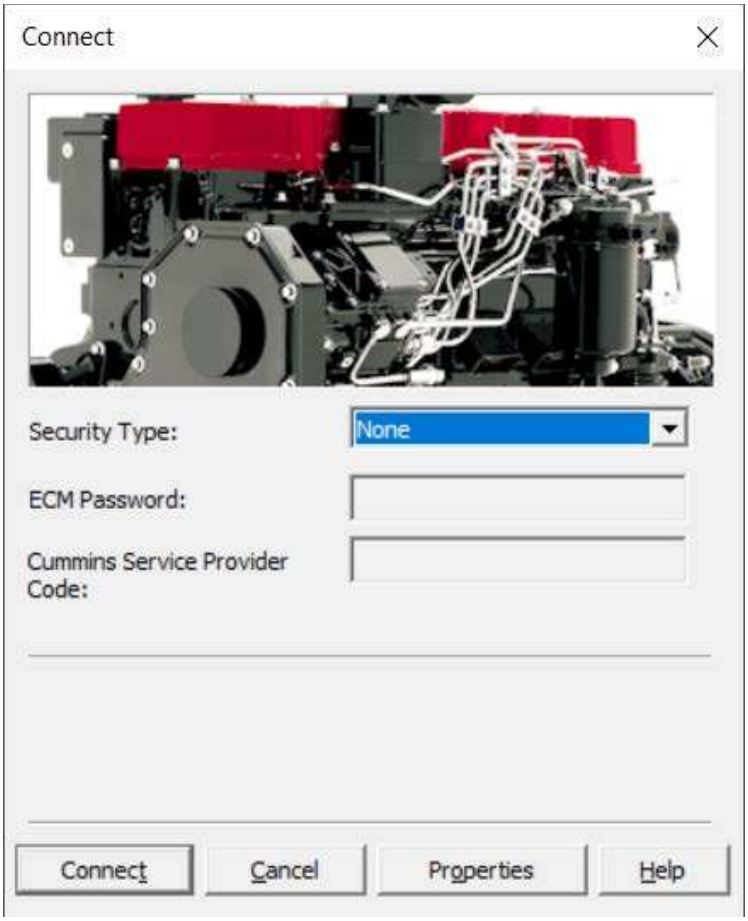
If loading your own calibration, save it to the following directory:

C:\Intelect\INSITE\CalibrationWorkspace

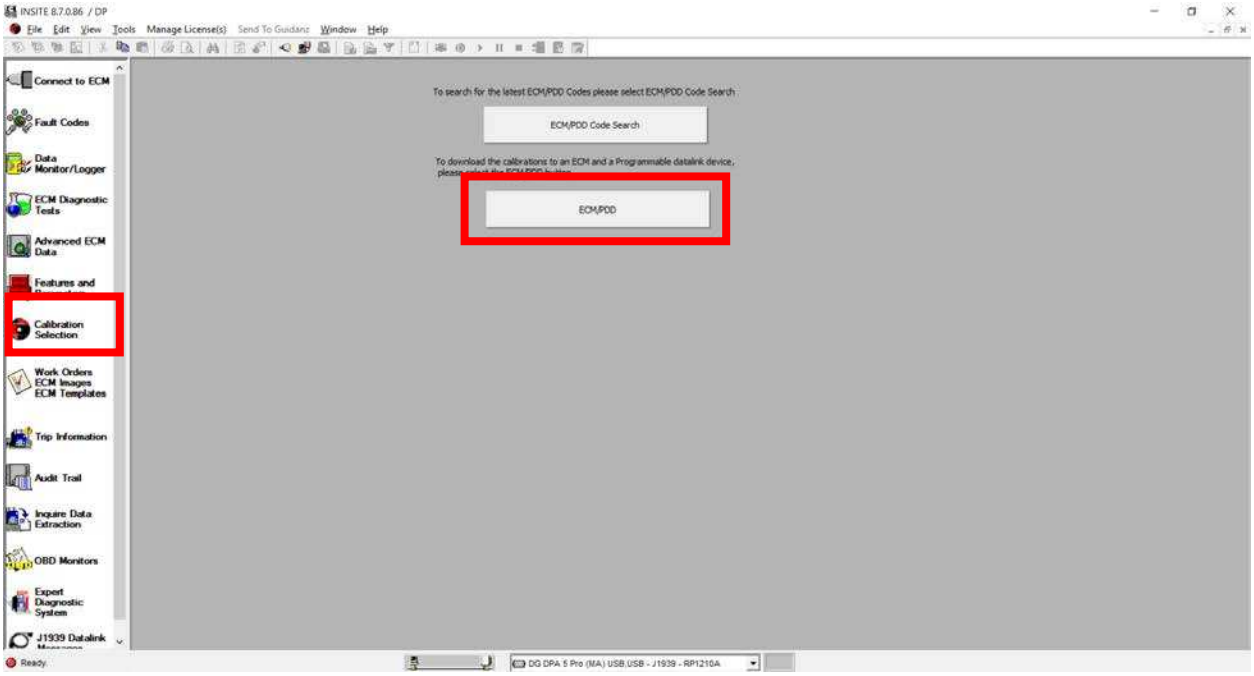
5. Check that “DG DPA 5 Pro” is selected at the bottom of INSITE, then hit “Connect to ECM”



The following window should appear. Use Security Type “None” and hit “Connect.”

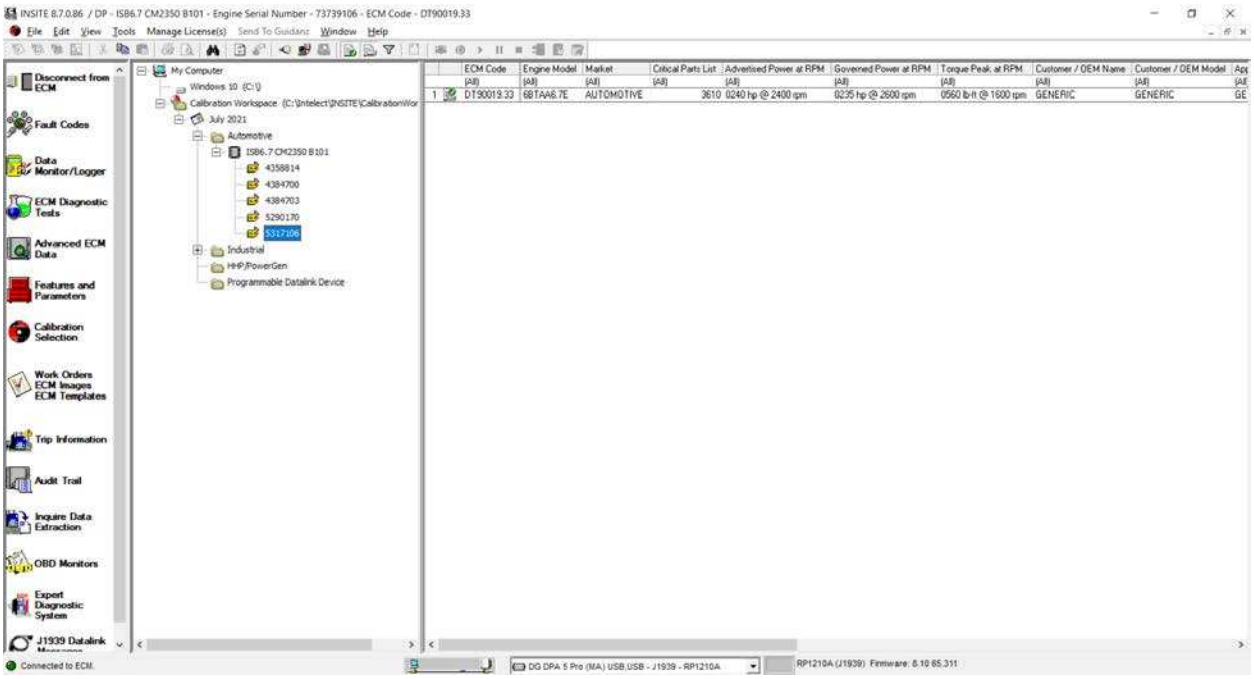


6. Go to “Calibration Selection.” Then hit “ECM/PDD.”



Navigate to the appropriate folder. The screen should look similar to this, but with the calibration type showing.

In the drop-down file directory, select the ECM part number that matches the ECM part number stamped on the physical ECM body. Note: the ECM will not accept a calibration meant for a different part number.



Double-click the appropriate row in the right-hand screen. If the calibration ID matches the ECM part number, the following screen should appear.

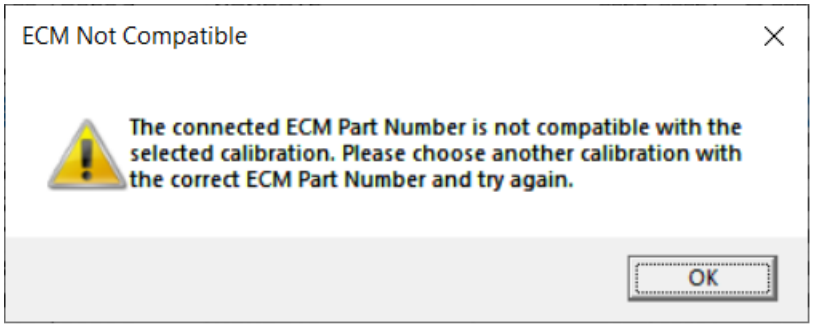
This screen means the re-flash will work. Follow the wizard to perform the re-flash.

Note:

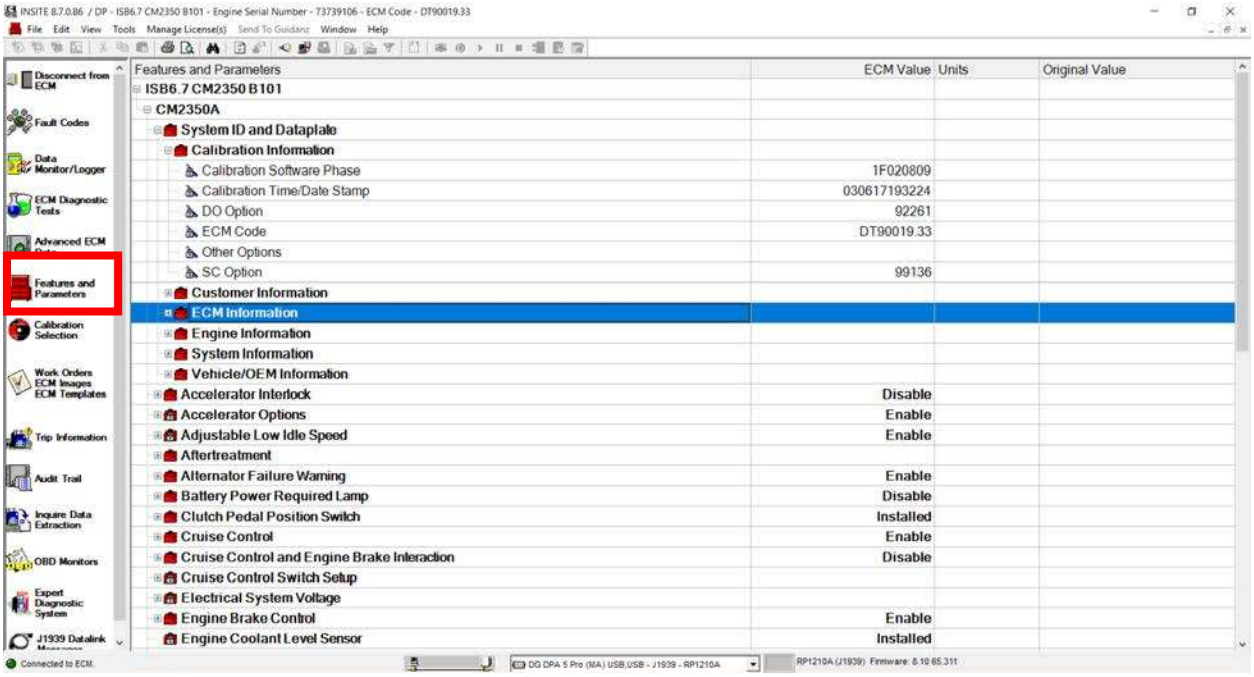
Changing to a different ECM Code (i.e. calibration) costs 1 Fleet Count, which is $200. So, be very sure your operation is correct before proceeding. For example, if the module currently has EF10102.24 and you are flashing DT90019.33, that costs $200 or 1 Fleet Count.

However, updating the version of the existing ECM Code is free. For example, if the module currently has DT90019.15 and you are flashing DT90019.33, that is updating the existing calibration. It does not require a Fleet Count and is free.

If the calibration ID is not compatible with the ECM part number, this error message will appear. This means flashing the selected calibration onto the connected ECM is not possible as there are hardware or firmware differences in the ECM.

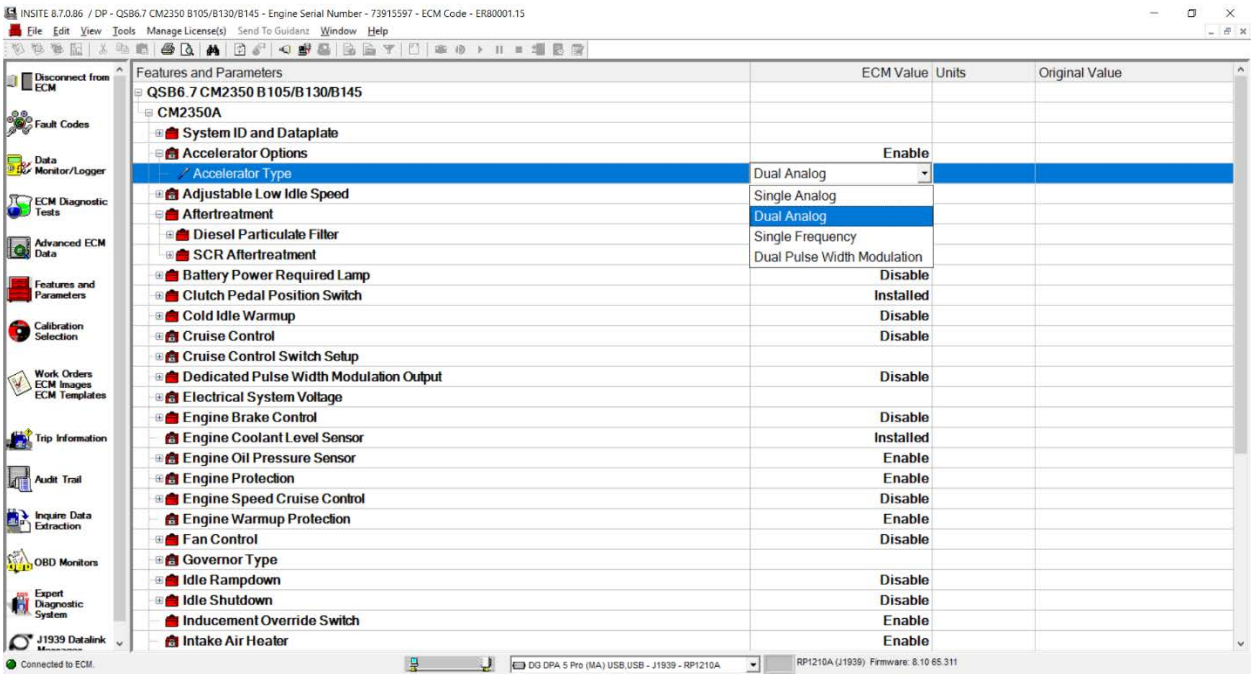


7. To check that the new calibration successfully loaded onto the ECM, go to “Features and Parameters,” and expand “Calibration Information.” The currently loaded calibration ID should appear under “ECM Code.”

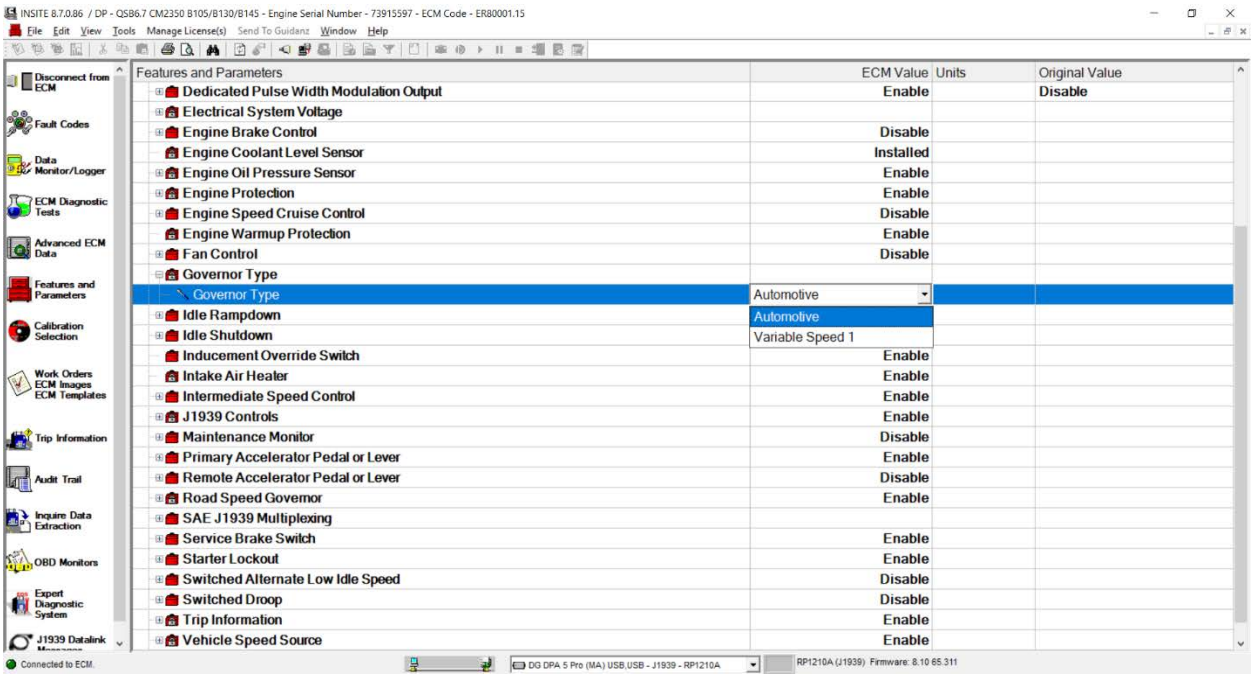


7. There are two parameters that sometimes default to values not appropriate for the dynamometer engine.

a. Accelerator Type: Check this is set to Dual Analog like in the picture below.



b. Governor Type: This should be set to Automotive like in the screenshot below. Automotive makes the accelerator pedal command a fueling or percent load setpoint. This is used for highway vehicle applications. Variable Speed 1 makes the accelerator pedal command a speed setpoint. This is used for tractor or industrial engine applications.


INSITE 8.7.0.86 / DP - QSB6.7 CM2350 B105/B130/B145 - Engine Serial Number - 73915597 - ECM Code - ER80001.15
File Edit View Tools Manage License(s) Send To Guidanz Window Help
Disconnect from ECM
Fault Codes
Data Monitor/Logger
ECM Diagnostic Tests
Advanced ECM Data
Features and Parameters
Calibration Selection
Work Orders ECM Images ECM Templates
Trip Information
Audit Trail
Inquire Data Extraction
OBD Monitors
Expert Diagnostic System
J1939 Datalink
Features and Parameters
ECM Value
Units
Original Value
Dedicated Pulse Width Modulation Output
Enable
Disable
Electrical System Voltage
Engine Brake Control
Disable
Engine Coolant Level Sensor
Installed
Engine Oil Pressure Sensor
Enable
Engine Protection
Enable
Engine Speed Cruise Control
Disable
Engine Warmup Protection
Enable
Fan Control
Disable
Governor Type
Governor Type
Automotive
Automotive
Variable Speed 1
Idle Rampdown
Idle Shutdown
Inducement Override Switch
Enable
Intake Air Heater
Enable
Intermediate Speed Control
Enable
J1939 Controls
Enable
Maintenance Monitor
Disable
Primary Accelerator Pedal or Lever
Enable
Remote Accelerator Pedal or Lever
Disable
Road Speed Governor
Enable
SAE J1939 Multiplexing
Service Brake Switch
Enable
Starter Lockout
Enable
Switched Alternate Low Idle Speed
Disable
Switched Droop
Disable
Trip Information
Enable
Vehicle Speed Source
Enable
Connected to ECM.
DG DPA 5 Pro (MA) USB,USB - J1939 - RP1210A
RP1210A (J1939) Firmware: 8.10 65.311