Dissertations and Theses

5-2016

Sensing, Actuation, and Embedded Control for a Custom SCR Nitrogen Oxides Emissions Reduction System

Anuj Kalidas Sharma

Scholarly Commons Citation
Sharma, Anuj Kalidas, "Sensing, Actuation, and Embedded Control for a Custom SCR Nitrogen Oxides Emissions Reduction System" (2016). *Dissertations and Theses*. 247.
https://commons.erau.edu/edt/247

SENSING, ACTUATION, AND EMBEDDED CONTROL FOR A CUSTOM SCR NITROGEN OXIDES EMISSIONS REDUCTION SYSTEM

A Thesis

Submitted to the Faculty

of

Embry-Riddle Aeronautical University

by

Anuj Kalidas Sharma

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Electrical and Computer Engineering

May 2016

Embry-Riddle Aeronautical University

Daytona Beach, Florida

SENSING, ACTUATION AND EMBEDDED CONTROL FOR A CUSTOM SCR NO_X EMISSIONS REDUCTION SYSTEM

by

Anuj Kalidas Sharma

A Thesis prepared under the direction of the candidate's committee chairman, Dr. Ilteris Demirkiran, Department of Electrical, Computer, Software & Systems Engineering, and has been approved by the members of the thesis committee. It was submitted to the School of Graduate Studies and Research and was accepted in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Computer Engineering.

THESIS COMMITTEE

[illegible]
Chairman, Dr. Ilteris Demirkiran

[illegible]
Member, Dr. Marc Compere

[illegible]
Member, Dr. Tianyu Yang

[illegible]	4/12/16
Department Chair, Dr. Timothy A. Wilson or Graduate Program Coordinator, Dr. Jianhua Liu	Date
[illegible]	04/14/16
Dean of College of Engineering, Dr. Maj Mirmirani	Date
[illegible]	4/22/16
Vice Chancellor for Academics, Dr. Christopher D. Grant	Date

1. Acknowledgments

First I would like to thank my thesis advisers Dr. Ilteris Demirkiran, Dr. Marc Compere, and Dr. Tianyu Yang at Embry Riddle Aeronautical University. They have all always been available to help me solve all queries in my research and writing work. They steered me in the right direction to ensure successful completion.

I would also like to thank the whole team involved in different aspects of this research. It was with the the teams active participation at every stage of the project that all goals were met.

Finally, I would like to express gratitude to my parents for providing for giving me immense support and continuous encouragement throughout my studies. This accomplishment would not have been possible without them. Thank you.

ABSTRACT

Sharma, Anuj Kalidas MSECE, Embry-Riddle Aeronautical University, May 2016. Sensing, actuation, and embedded control for a custom SCR nitrogen oxides emissions reduction system.

Diesel engines are the main power source for medium and heavy duty on road vehicles. With the rising standards for mileage of vehicles set by the Corporate Average Fuel Economy and the Environmental Protection Agency, diesel engines should soon be considered in lieu of gas engines. Despite advantages of diesel engines, diesel emissions include harmful gasses like carbon monoxide, nitrogen oxides and particulate matter. Selective Catalytic Reduction (SCR) systems have long been used to reduce diesel emissions from medium to heavy-duty diesel engines.
Primary focus of this research effort is the implementation and improvement of a SCR system on light diesel engines. To improve efficiency we implemented a control law to limit the emissions within bounds ensuring low emissions for varied drive cycles. We developed a controller network using user datagram protocol to collect engine data using arduino. The controller network establishes communication link between sensor data collection and a raspberry pi controller to enable full control over the test station.

TABLE OF CONTENTS

LIST OF FIGURES

2. Introduction

2.1 Background

The number of vehicles on the road has increased tremendously over the past decade. While people enjoy the convenience brought by vehicles, it adversely affects the environment people rely on to survive. In today's atmosphere, more than one third of the carbon monoxide and nitrogen oxides come from vehicular emissions (The Union of Concerned Scientists, 2015). Controlling air pollution has become an urgent global problem. Living in a polluted environment causes people to suffer from respiratory diseases and other health complications. In addition to poor health, the uncontrolled emission exacerbates the greenhouse effect. The United States (US) Congress enacted the Corporate Average Fuel Economy (CAFE) in 1975 to regulate the fuel-economy standards. Due to the new policy, the average fuel economy for cars will be increased from 27.5 mpg (miles per gallon) to 37.8 mpg by 2016, which will further be increased to 54.5 mpg by 2025 (Car & Driver, 2015). Based on the fact that diesel engine vehicles have a better mileage and power density, they will have a larger market share in the future (Hsieh & Wang, 2010). Therefore, controlling the diesel emissions from diesel engines has become increasingly important. Selective Catalytic Reduction (SCR) systems have become the standard and a promising technique for reducing nitrous oxides, NO and NO_2 often written as NO_X, emissions from light to heavy duty diesel engines (Chen & Wang, 2013, 2014).

SCR system is one of the most cost effective and fuel efficient technologies to help reduce diesel engine emissions. SCR systems can reduce upto 90% NO_X emissions. SCR systems have been in use for decades to reduce emissions from marine and cargo vessels, tugboats and other heavy-duty vehicles (Forum, 2015).

The term SCR can be used to describe a chemical reaction that reduces NO_X to nitrogen and water. Figure 2.1 (MTU-Report, 2014) shows the same rection. In order to complete this chemical reaction, a reducing agent is required. This is a non toxic odorless agent widely used in the USA since 2010. Often referred to as 'Adblue', it is a 32.5% pure grade solution of urea and de-ionised water. In order to ensure minimal emissions, the reducing agent added is about five to seven percent of the fuel consumption (Sinzenich & Wehler, 2014).

The use of SCR systems to reduce emissions from hybrid-diesel engine vehicles, methodology adopted, test procedures, and challenges faced will be discussed in the coming sections.

2.2 Goals and Objectives

The goal of this thesis is to develop a control system for SCR system which can be implemented on diesel engines suitable for diesel hybrids to reduce harmful emissions. To meet the goal, following objectives were enumerated:

1. Meet standard regulations set by the Environmental Protection Agency and the Corporate Average Fuel Economy for diesel engines.

2. Build a safety compliant test shed that allows complete testing of the system. Include power connections for the electrical and network components. Rewire engine start switch to provide for emergency cut-off system.

3. Install Internet Protocol (IP) cameras in the test shed for continuous monitoring of the shed.

4. Install Diesel Oxidation Catalyst in addition to SCR system to enhance its operation.

5. Develop a dedicated communication network to allow communication between different software modules. Install network components in the test shed.

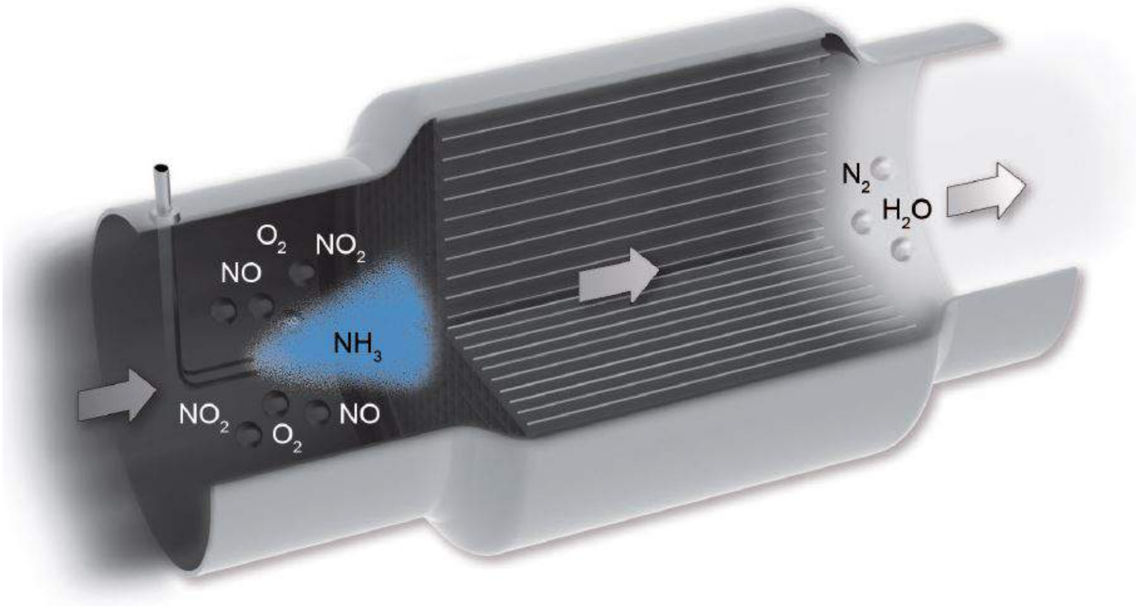


Figure 2.1.: SCR Reaction

6. Design circuit and update system software to provide better control over the SCR system. Update the Arduino program to sense engine NO_X in exhaust and diesel exhaust fluid tank pressure. Integrate Raspberry Pi to actuate the SCR system using Simulink model.

7. Develop an algorithm to control the diesel exhaust fluid injection rate and achieve pressure control for diesel exhaust fluid tank.

8. Develop a new test stand managing system to provide complete control and monitoring of the test parameters using Matlab/Simulink.

2.3 Outline of Thesis

This thesis work addresses the issues concerning the increasing emissions from diesel engines, the main focus being enhancing the SCR system. We use a particulate filter to reduce particulate matter and soot emissions. Diesel Exhaust Fluid injection is provided after the particulate filter to produce a chemical reaction in the exhaust which reduces the nitrogen oxides from the exhaust. We also incorporate a diesel oxidation catalyst in the exhaust pipe which enhances the catalytic reaction to further reduce nitrogen oxides and carbon monoxide. We develop an algorithm to control the amount of diesel exhaust fluid injection. User datagram protocol network is used to communicate sensor data to a controller that actuates the hardware and computes the control algorithm, thus regulating the engine emissions regardless of the type and size of a vehicle. With the increasing standards of the CAFE and EPA, this approach is to be implemented on hybrid diesel engines vehicles.

In following chapters of the report, the proposed SCR system controls are detailed. Chapter two includes the literature review to discuss the different aspects concerned with an SCR system. Chapter three is a detailed project description of the different components and sub-systems of the implemented approach followed but the SCR process in Chapter four. Chapter five will detail the control and sensing network of the system, and the control system as a whole is discussed in chapter six. Conclusions and results obtained are covered in Chapter seven.

3. Literature Review

3.1 Types of Vehicular Emissions

The exhaust gasses from a diesel engine are a complex mixture of byproducts of complete combustion, some amount of oxidation products of sulphur and nitrogen, and compounds derived from fuel and lubricants. Major constituents of the exhaust are carbon monoxide, carbon-di-oxide, oxides of sulphur, oxides of nitrogen, hydrocarbons, and particulate matter (Elliott, Nebel, & Rounds, 1955). As compared to a gasoline engine, a diesel engine contains a higher proportion of nitrogen oxides (NO_X) and diesel particulate matter (DPM). Thus, nowadays the main focus is on the reduction of NO_X and DPM (Prasad & Bella, 2010). DPM is the result of incomplete combustion of hydrocarbons. It mainly consists of carbon with minor components of organic compounds from fuel and lubricating oil and inorganic sulphur compounds (Prasad & Bella, 2010). Nitric Oxide and Nitrogen-di-oxide, together are referred to as nitrogen oxides. Diesel engines and other diesel equipment account for more than 50% nitrogen oxides emissions (The Union of Concerned Scientists, n.d.). Nitrogen oxides lead to the buildup of ground level ozone that lead to serious health issues (The Union of Concerned Scientists, n.d.). Carbon monoxide, hydrocarbons and aldehydes are generated by incomplete combustion of fuel. Hydrocarbons can lead to smog when present in higher proportions in the environment. Vehicles used in enclosed spaces like tunnels or warehouses, these emissions can pose a severe health issue (N. T. Inc., n.d.-a).In this research effort our main focus will be on reducing nitrogen oxide and soot emissions from light duty diesel engine vehicles.

3.2 Emission Control Methods

With the increase in the use of diesel engines, it is essential to keep a check on the harmful exhaust gas emissions. One of the biggest challenges in diesel engines is simultaneous reduction of NO_X and soot (Kitamura T & H, 2002; Heywood, 1988). Numerous approaches have been proposed to reduce emissions from a diesel engine. Particle swarm optimization (PSO) is a technique applied by Bambang Wahono and Harutoshi Ogai to reduce NO_X and soot simultaneously (Wahono, Ogai, Ogawa, Kusaka, & Suzuki, 2012). The used PSO to determine optimal input parameters for engine optimization. Exhaust gas re-circulation is another approach to reduce engine NO_X by reducing in-cylinder temperature and oxygen concentration (Ming Zheng, 2004).Diesel particulate filters (DPF) have now become a standard component to reduce particulate matter or soot emissions. A diesel oxidation catalyst (DOC) is typically installed to reduce carbon monoxide emissions. Selective catalytic reduction (SCR) system is now a leading device to reduce NO_X emissions (Ming-Feng Hsieh, 2012).

Several approaches to implement SCR systems have been presented in the recent years. J. Chi and H. DaCosta presented controller that uses time-varying input information for the desired NO_X reduction rate, inlet/outlet exhaust gas temperatures, and catalyst inlet/outlet NO_X emission rate to determine urea dosage amount (J. Chi, 2005). Authors in (Maruthi Devarakonda & Johnson, 2009) and (Devesh Upadhyay, 2005) present similar controllers based on linearized SCR models to reduce NO_X and ammonia emission for specific operating ranges. The approach suggested by Ming-Feng Hsieh and Junmin Wang suggests controlling the ammonia surface coverage ratio of the SCR system which leads to effective regulation of tail pipe NO_X and ammonia emissions (Hsieh & Wang, 2011). Since most SCR systems are employed for heavy duty diesel engine vehicles, we propose a new design for a SCR system to reduce emissions for light duty diesel engine vehicles. For a heavy diesel engine, SCR system is the most effective on highway drive cycles since a constant injection rate is provided. The main challenge for a light duty vehicle is to develop an algorithm to control the diesel exhaust fluid injection rate for varying drive cycles in a city.

3.3 Control System

One of the major concerns of SCR systems is its inability to efficiently regulate the amount of Adblue (32.5% aqueous urea solution), also known as Diesel Exhaust Fluid (DEF), for varying engine loads. Pingen Chen and Junmin Wang estimated the actual ammonia concentration input and ammonia in exhaust emission, referred to as ammonia slip, for low temperature operations (Chen & Wang, 2014). These estimations were used to prevent ammonia slip by varying the injection rate. Yue-Yun Wang, Hui Zhang and Junmin Wang designed a fuzzy logic control system to correct Adblue injector flow rate (Wang, Zhang, & Wang, 2015). They used a feedback control from the exhaust sensors to the injector in order to correct the Adblue injection rate. M. Karthikeyan, K. Annamalai and V.N. Banugopan designed a flow control system driven by engine rpm (rotations per minute) and temperature. The flow control system ensures a proper injection rate for different driving scenarios (Prabhakar, Karthikeyan, Annamalai, & Banugopan, 2010). A model based feed forward controller is presented by Waseem A Rasheed, Parul Goyal, and Cyril Joseph. The controller analyzes the effects of slowly varying engine parameters on engine NOx emissions (Rasheed, Goyal, & Joseph, 2013). Most SCR control strategies regard Adblue injection rate as the sole control input. Pingen Chen and Junmin Wang presented another control strategy that treats engine exhaust emission of NOX as the virtual control to improve fuel economy while reducing overall NOX emissions (Chen & Wang, 2013). Xu Zhai, Zhonglin Chen, Jie Yao and He Wang conducted extensive research on the influences of reaction temperature, NH3/NOx ratio and space velocity on SCR performance to prepare a new catalyst (Zhai, Chen, Yao, Wang, & Li, 2009). Since most researches focus on heavy diesel engines, their primary goal is reducing ammonia slip. In this research effort, our focus being development of SCR system for light duty diesel engines, we developed an enhanced SCR system that prevents both ammonia slip and also provides better reduction of NO_X emissions using Diesel Oxidation Catalyst (DOC) and a raspberry pi controller to compute the injection rate of DEF fluid.

4. Project Description

As described earlier, the main goal is to develop a control system that is capable of reducing exhaust emissions. This control system can be used for hybrid diesel engines running at any rpm and temperature. A diesel emission test stand was built to collect the data from the engine exhaust. The data is to be analyzed in order to estimate an appropriate DEF injection rate. Ensuring correct implementation of an SCR system up to 98% reduction in NO_X (Aerinox-inc.com, 2015) can be achieved. In order to break down pollutants from the diesel engine in the exhaust stream we integrated a DOC into the SCR system. The SCR system is installed between an upstream sensor which senses the engine NO_X level before the ammonia injection and the downstream sensor which sense the tailpipe NO_X level after the injection of ammonia. One of the drawbacks of SCR systems currently in use for heavy diesel engines is that it does not have an efficient means to control the injection rate of DEF. We coped with this issue by means of an adaptive control system. Having analyzed the data received from the NO_X sensors, the SCR control system estimates the required injection rate. Receiving the data from the sensors and transmitting the injection rate information to the injector pump are accomplished by using raspberry pi as the main controller over User Datagram Protocol (UDP) network. The protocol establishes a communication between the engine and a test managing system.

4.1 Methodology

NO_X is a mixed gas including NO (Nitric Oxide) and NO_2 (Nitrogen Dioxide). Not only does NO_X affect our health, but also damages the ozone in the stratosphere. During the engine combustion process, a large amount of NO_X is produced. The amount of NO_X produced drastically increases when engine is under higher loads. To prevent NO_X going into the air, SCR technology is implemented to reduce NO_X. SCR system uses DEF, which is converted to ammonia by the heat of the exhaust (Upadhyay & Nieuwstadt, 2002), is taken into reaction with the NO_X to produce water (H_2O) and nitrogen (N_2) (EBMPAPST, n.d.). Figure 4.1 shows the SCR reaction.

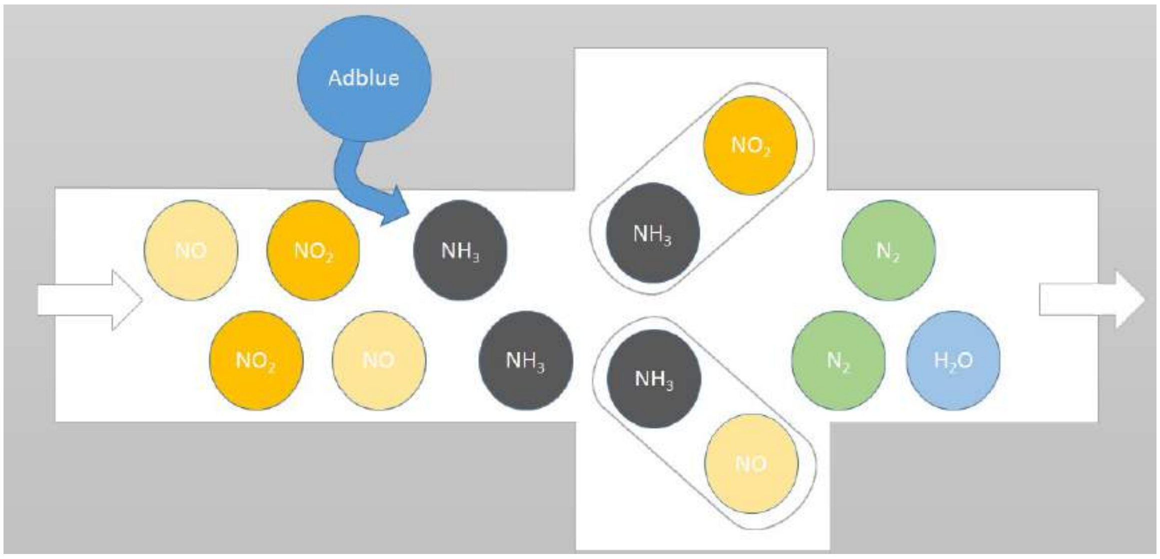


Figure 4.1.: Selective Catalytic Reaction Principle

$$4NO + 4NH_3 + O_2 4N_2 + 6H_2O$$

$$2NO_2 + 4NH_3 + O_2 3N_2 + 6H_2O$$

$$NO + NO_2 + 2NH_3 2N_2 + 3H_2O$$

As shown in the equations above, Ammonia (NH_3) converts the NO_X into Nitrogen, N_2, and water vapor, H_2O. As a result, the air pollution due to NO_X will decrease.

4.2 Test Shed

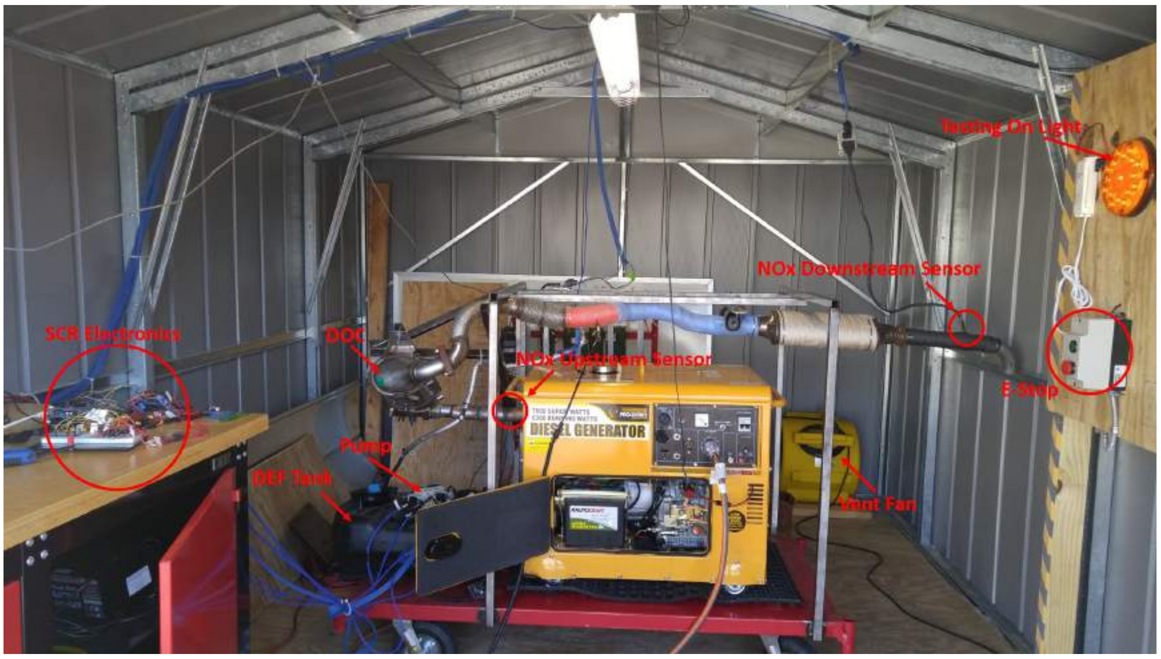


Figure 4.2.: Test Shed

A test shed was built to work with the diesel engine test stand and other necessary testing equipment, and was used to carry out all tests for the SCR system. The shed was built to provide for a better and safer operation of the diesel engine. For the same reason, the engine exhaust was directed outside of the shed. The load bank was placed at a higher level on heat resistant blocks to prevent any fire risks, and a vent fan was placed next to the load bank to help dissipate heat. We also designed an Emergency stop button to cut-off the connection to the engine and testing lights.

The shed also includes electrical connection to power the electrical equipment on the test stand and the networking components. A camera pointed towards the test stand and the electrical panel in the shed was used to monitor the shed while testing.

4.3 Nitrogen Oxide Sensors

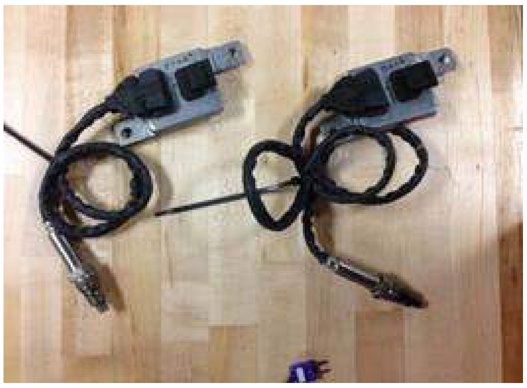


Figure 4.3.: Nitrogen Oxide Sensors

We used two NO_X sensors (shown in figure 4.3, one close to the exhaust inlet (upstream) and a second one at the exhaust outlet (downstream) to gather NO_X data. The upstream reports NO_X values before the SCR reaction and the downstream sensors reports values after the SCR reaction. The downstream sensor is equipped with a 120 ohms resistor, which acts as a controller area network (CAN) terminal, and a second 120 ohms terminal resistor was incorporated in the CAN bus circuit. The sensors report the data to the sensor arduino, different CAN IDs allow us to differentiate between the data from both the sensors. Our test results show a significant reduction of NO_X content in the downstream sensor.

4.4 Diesel Exhaust Fluid

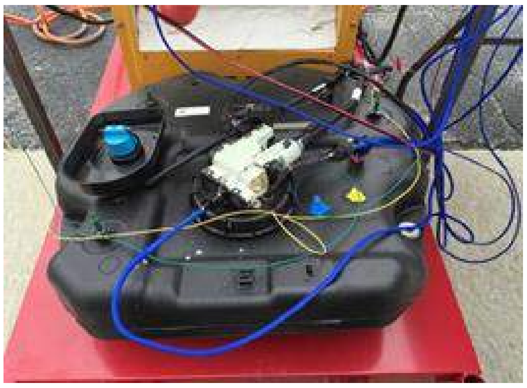


Figure 4.4.: Diesel Exhaust Fluid Tank

Figure 4.4 shows the tank used to store and feed diesel exhaust fluid. Diesel exhaust fluid (DEF) is also known by its trademark Adblue, is a colorless, nonflammable aqueous solution of urea (32.5% urea and 67.5% de-ionized water) and is a required reactant in the SCR process inside the engine exhaust of diesel vehicles. DEF reacts with the nitrogen oxides to produce water vapor and nitrogen which are harmless. The DEF tank is mounted with a pump to run pressurized fluid through the lines upto the injector. The pump uses 100Hz pulse width modulated (PWM) signals to start pumping DEF to the lines. Using the bang bang thermostat control method, we run the pump to maintain a 20 psi pressure in the fluid lines, while the injector releases small amounts of the fluid into the exhaust to react with nitrogen oxides. A pressure of 20 psi is maintained in the tank in order to keep the release small amounts of fluid from the injector.

4.4.1 Pressure Sensor Calibration

The pump is controlled to maintain a steady pressure in the fluid lines using the pressure sensor data. The pump stays on until a pressure of 20 psi is reached, and turns off at a pressure higher that 20 psi. With a small duty cycle for the pump, the pressure can be maintained at 20 psi. The pressure sensor returns digital values ranging from 0 to 1023. In order to make the digital data useful, the pressure sensor was calibrated to report pressure values in psi.

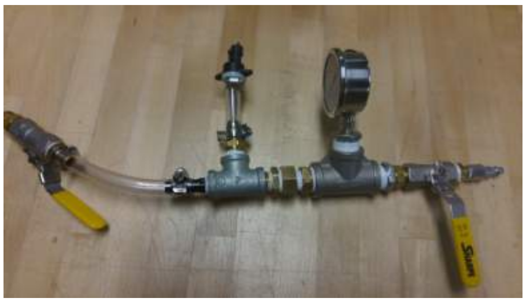


Figure 4.5.: Pressure Sensor Calibration Setup

The figure 4.5 shows the setup for pressure sensor calibration. The setup used clear pressure tubes to connect the pressure sensor to an analog pressure gauge. When air hose is connected to the setup the analog gauge reads the pressure coming in and the pressure sensor returns digital values for the same analog reading. Using a plot for the digital values and their corresponding analog values, we found the following linear relation between digital and analog values

$$AnalogValue = (0.1362 * DigitalValue) - 15.052$$

4.4.2 Injector Calibration

Injector calibration was necessary to provide for a controlled injection on DEF into the exhaust. We measured the mass flow rate of the injector at different duty cycles for the injector using the following relationship:

$$volume = \frac{mass}{density}$$

The injector was mounted to a test stand using a clamp as seen in figure 4.6.

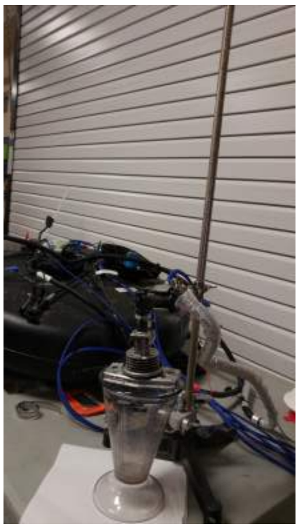


Figure 4.6.: Injector Calibration Setup

The pump was run to maintain a pressure of 20 psi in the line. Since prior results show that DEF injections in small amounts may also saturate the exhaust with ammonia, we concentrated more on lower duty cycles for the injector. Since the injector uses PWM signals to turn on and off, it is easy to control the injector duty cycle by varying the PWM signal duty cycle. Table 4.1 below shows the results from calibration.

Injector Duty Cycle	Mass(grams)	Mass(grams/second)
1%	104.70	0.00375
2%	105.75	0.00813
3%	106.83	0.01263
4%	108.30	0.01875
5%	109.38	0.02325
6%	110.38	0.02742
7%	111.84	0.03350
8%	112.85	0.03771
9%	113.80	0.04167
10%	115.11	0.04713
20%	126.47	0.09446
30%	138.30	0.14375
40%	149.73	0.19138
50%	161.13	0.23888
60%	171.35	0.28146
70%	182.15	0.32646
80%	194.72	0.37883
90%	195.06	0.40338
100%	225.00	0.50500

Table 4.1.: Injector Calibration Results

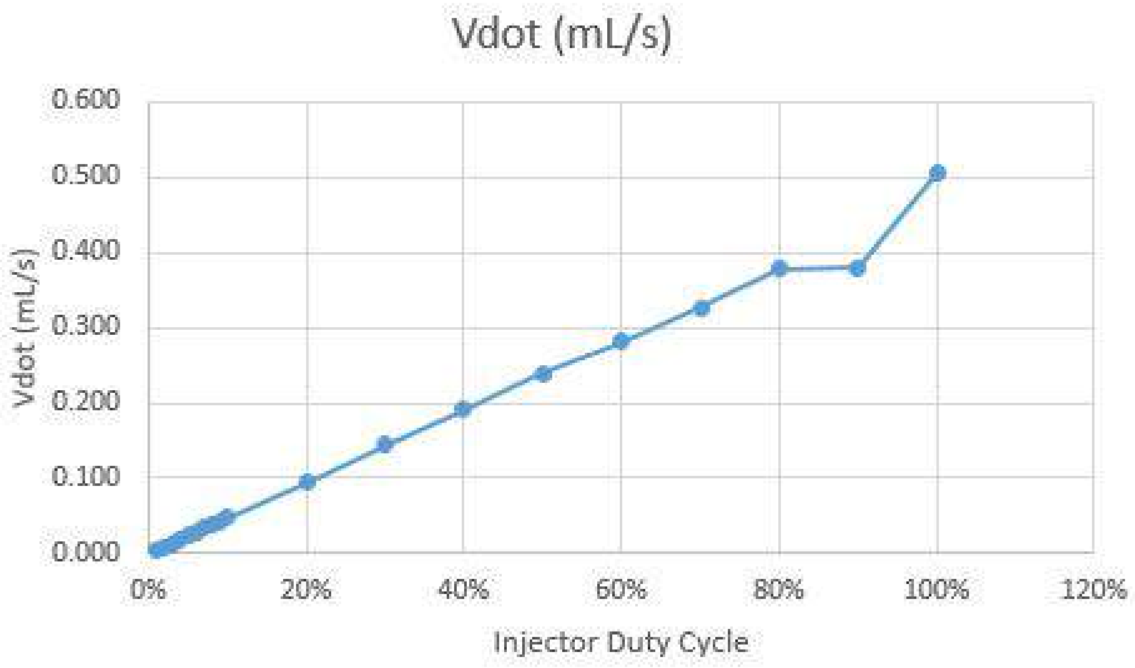


Figure 4.7.: Plot for Injector calibration (Mass grams per second vs Injector Duty Cycle)

We let the pump and injector run for 4 minutes at every duty cycle to collect data. We noticed a few errors caused by the time taken by the fluid to reach the injector.

4.5 Diesel Oxidation Catalyst

Diesel Oxidation Catalyst (DOC) is used to reduce emissions from diesel engines. DOC is used as an after treatment for diesel engines to reduce emissions.

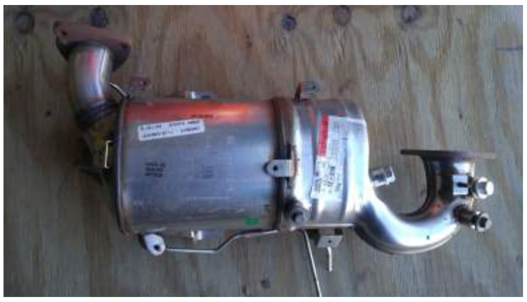


Figure 4.8.: Diesel Oxidation Catalyst

A DOC is a metal coated flow-through honeycomb structure contained in a stainless steel housing. As hot diesel exhaust flows through the honeycomb structure, the metal coating causes a catalytic reaction that breaks down pollutants into less harmful components. (Agengy, 2010) A DOC typically reduces emissions of particulate matter by 20% to 40% or more and gaseous emissions by 50% to 70%. (Inc.", 2015) DOC is designed to oxidize carbon monoxide, gas phase hydrocarbons, and the soluble organic fraction [SOF] of diesel particulate matter to CO_2 and H_2O (N. T. Inc., n.d.-b) as shown in the equations below:

$$CO + \frac{1}{2}O_2 = CO_2$$

$$[Hydrocarbons] + O_2 = CO_2 + H_2O$$

$$[SOF] + O_2 = CO_2 + H_2O$$

After the Diesel Particulate Filter (DPF) and the SCR system, we use DOC as the third step to reduce emissions from the exhaust. Carbon Monoxide (CO) and Hydrocarbon (HC) emissions result from incomplete combustion of fuel (Aerinox-inc.com, 2015). A DOC oxidizes HC and CO to produce water vapor and carbon-di-oxide.

4.6 Load Bank

As shown in figrue 4.9, a programmable load bank was designed and developed to dissipate power from the diesel engine-generator. The load bank is powered by a built-in 120 volts supply on the generator. Under different driving loads, the engine exhaust composition varies.

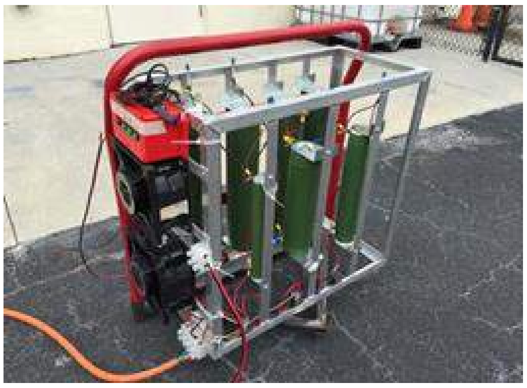


Figure 4.9.: Load Bank

To inject the right amount of DEF to react with the NO_X, it is important to perform the relevant tests. We used the load bank to simulate a range of loads on the engine. Keeping in mind the actual usage of diesel vehicles, the load bank was designed to dissipate the power ranging from 250 watts to 6,000 watts (Sharma, 2016). The load bank uses an arduino to control the power settings required.

4.7 User Datagram Protocol

Transport protocols are source to destination protocols. Users and/or applications can interact directly using the transport protocols. A connection oriented service is provided by the transport control protocol and datagram transport is supported by the User Datagram Protocol. (Rajaboina, Reddy, & Kumar, 2015) User Datagram Protocol (UDP) is a transport layer protocol defined for use with internet protocol (IP) network layer. UDP provides best-effort datagram service to an End System (IP Host). This protocol sends messages with a minimum protocol mechanism. (Postel, 1980) Compared to other transport protocols, UDP is unique, it does not establish end to end connections between communicating end systems. For the same reason UDP can offer very efficient communication for some applications. To transmit a datagram using UDP, a computer completes the UDP header which consists of the following: Source Port: This is the service access point for local client. Destination Port: This is the service access point for the remote server. UDP Length: The number of bytes. UDP Checksum: Used to verify end to end data is not corrupted in the network. (Fairhurst, 2008) UDP does not support congestion control mechanism, and thus can lead to packet loss if not setup correctly. As data rate is increased, number of bytes received also increases to a bottle neck. Once the receive rate crosses the threshold the number of received bytes drops rapidly. (Pakanati, Padmavathamma, & Reddy, 2015) UDP provides multiplexing/de-multiplexing to IP and does not increase end to end delay over IP. ("University of California, n.d.) Here we use UDP to provide a simple communication link to send and receive data packets. UDP link establishes a connection between the sensor arduino and the raspberry pi controller, load bank arduino and rasperry pi controller, and raspberry pi controller and test manager system. A better idea is provided further in figures 6.1 and 6.2.

5. Process

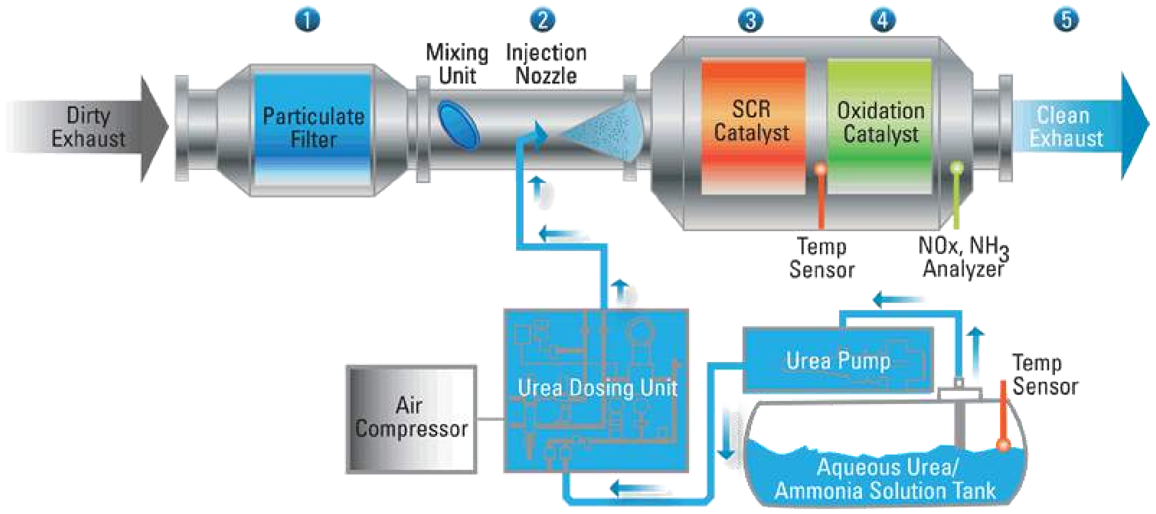


Figure 5.1.: SCR System

A Diesel Particulate Filter (DPF) is the first component to clean the polluted exhaust by capturing the soot and particulatee matter that is emitted from a diesel engine exhaust. A complete SCR system (Aerinox-inc.com, 2015) is shown in Fig.5.1 Nowadays, a DPF has the ability to filter out about 80% of the diesel particulate emissions (Theaa.com, 2015). However, NO_X cannot be filtered by a DPF, so a chemical reaction is required as the second step. The reaction takes place under a catalytic environment. Adblue (DEF) is injected into the exhaust, where it is converted to ammonia by the high exhaust temperatures. Ammonia is used as the reductant to consume NO_X. A tank is filled with Adblue, the pump connected to the tank moves the DEF to the SCR system pipe until it reaches the injector. The injector introduces the right amount of DEF into the exhaust pipe to react with NO_X to produce Nitrogen and water vapor. DOC is implemented into the SCR system in order to improve the chemical reaction and also removing any residual carbon monoxide or hydrocarbons from the exhaust. It also reduces engine emissions after a cold start (Karthikeyan & Krishnan, 2010). The level of harmful gasses like carbon monoxide, hydrocarbons, and organic fraction of diesel particulate (SOF) are reduced by the DOC. These reductions can be described by the following reactions

$$[Hydrocarbons] + O_2 = CO_2 + H_2O$$

$$C_nH_2m + (n + \frac{m}{2})O_2 = nCO_2 + mH_2O$$

$$CO + \frac{1}{2}O_2 = CO_2$$

It can be seen from the above reactions that hydrocarbons, SOF compounds and carbon monoxide are all converted into water and carbon dioxide, which are components of our atmosphere. A DOC also increases the content of NO_2 by oxidizing NO in the exhaust gases;

$$NO + \frac{1}{2}O_2 = NO_2$$

This increase of NO_2 enhances the catalytic reaction of SCR system, and reduces the amount of ammonia slip (Dieselnet.com, 2015). Thus, the addition of DOC not only provides more reduction in the harmful emissions but also improves SCR system performance. Depending on the engine load, the exhaust gas composition varies, which makes it necessary to have a control on the amount of DEF injections. We use an arduino with a controller area network (CAN) bus shield to read engine exhaust data and tank pressure. The addition of an ethernet shield to the same setup gives the arduino the capability to communicate using the user datagram protocol (UDP) network setup for the test. The engine data and tank pressure is communicated to raspberry pi based control system which calculates the amount of DEF injections required at that instant. All data is communicated further to a test stand manager, over the same UDP network, which logs and monitors all test parameters. To achieve this functionality:

- The injector is maintained at a 1% duty cycle of a 3Hz signal.
- The pump is maintained at 10% duty cycle of a 100 Hz signal.
- NO_X sensors are initialized after 500ms if no data is reported.
- Arduinos send data to raspberry at 4Hz.
- Raspberry pi sends to the test manager at 10Hz and receives from the test manager at 20Hz.

6. Sensing and Control Network

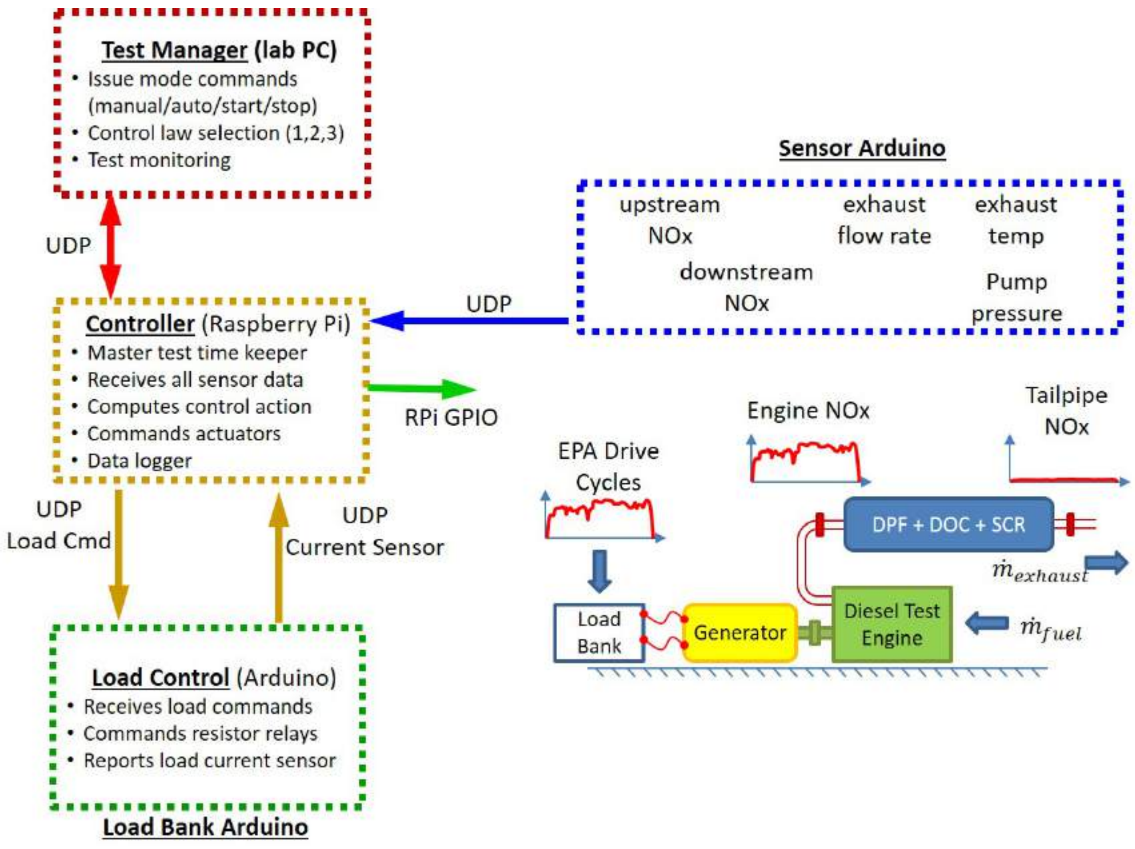


Figure 6.1.: SCR system sensing and control diagram

Sensing and Control network is setup to work over a dedicated UDP network. The network provides for communication link between three sections of the test stand.

- Sensing
- Actuation
- Test Manager

Sensing is done using two Arduinos, one for the sensors from the diesel engine and another for the load bank. Arduino for the diesel engine sensors is programmed to get Controller Area Network (CAN) data from the engine, which includes the NO_X values from the exhaust. The CAN data is sent to the raspberry pi using the UDP network. The Arduino also sends the pressure, temperature and flow rate recorded at the engine.The load bank Arduino commands the load bank for different load settings and sends back the load settings and current value to the raspberry Pi.

Raspberry Pi is the main controller for the SCR system. It is responsible for actuation of all the hardware, computing the DEF injection rate, and logging test data. Sensor arduinos communicate all data to the controller over the UDP network, raspberry pi processes and logs the sensor data ,computes control action and actuates the hardware.

Test Manager acts as the control station allowing all the testing to be conducted. The whole test setup is in the test shed built specifically for all testing carried out with the diesel engine. The Test manager inside the lab allows complete control and monitoring of the test over the same UDP network. The test manager connects to the controller (raspberry pi) network using a wireless bridge router to the test shed router.

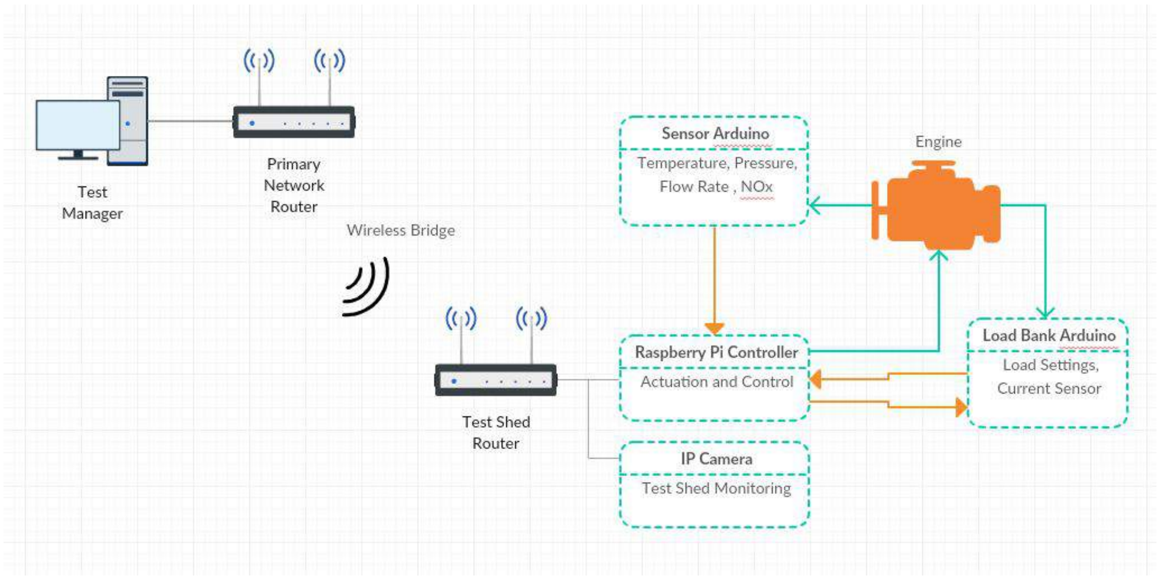


Figure 6.2.: Network Diagram

Figure 6.2 shows the network connections for the whole system. The test manager is the monitoring station, connected using the primary router located inside the lab. A second router, bridged wirelessly to the primary router, is setup in the test shed to connect the raspberry pi controller, sensor arduino, load bank arduino and the IP camera to the network.

7. Control System

The control system includes the following main sub-systems:

- Two Arduinos
- Raspberry Pi
- Test Manager (Computer)

A dedicated UDP network provides a communication link between all the systems.

7.1 Arduino

We have two arduinos in the control system. One getting engine exhaust data and tank pressure readings, and another one attached to the load bank to report resistors in use and current being drawn. both the arduinos are programmed in arduino environment. The arduino codes are available in appendix A.

7.1.1 Sensor Arduino

Sensor arduino collects engine exhaust data and tank pressure readings and send them to the raspeberry pi controller. To collect engine data, we use a controller area network (CAN) bus shield for the arduino. The CAN Bus shield has a Microchip MCP 2515 CAN controller with an MCP 2551 CAN transceiver (M. T. Inc., 2007, 2010), which gives the arduino CAN Bus capability. CAN is a serial bus communications protocol developed by Bosch, and is now widely used in the automotive industry, and other applications with networked embedded control.

The prototype design for the SCR system used only the arduino with CAN capability for the whole system functioning including a control for DEF injection. The DEF injections were manually updated using the arduino serial communication. The CAN network collected all the sensor data, and sent it to test managing system. Test data was controlled and monitored using arduino serial communication.The CAN bus shield requires two 120 ohms terminal resistors to connect CAN high and CAN low wires. The circuit was completed using one 120 ohm resistor from one of the NO_X sensor, and the second one was soldered on a solder board. Also to amplify low current from arduino, six N-channel MOSFETs were used for injector, pump, solenoid, line heater, pump heater and tank heater.

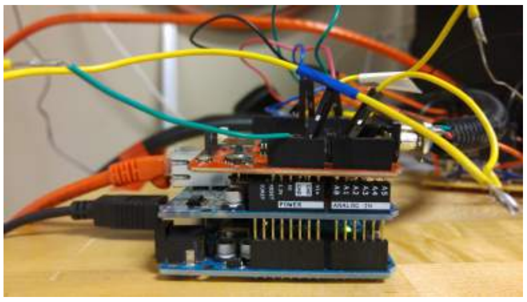


Figure 7.1.: Sensor Arduino with CAN bus and Ethernet Shields

Upon successful testing, we improved the control system by updating serial communication to UDP communication. Updating to UDP communication from serial communication required the addition of an Ethernet shield to the arduino and CAN bus shield combination. Ethernet shield enables the arduino to connect to a network connection using the W5500 chip (WIZnet, 2013). We established a dedicated UDP network for the SCR system to ensure minimum loss of data. The first two systems were entirely dependent of arduinos.

We continued development after successful integration of ethernet shield. The current design uses the arduino only as a sensor. It reads all CAN bus data and tank pressure sensor data. All hardware actuation and control algorithm processing is done by the raspberry pi. Figure 7.1 shows the current arduino setup. At the bottom is the arduino connected to power supply, the ethernet shield with RJ45 ethernet cable sets on top of the arduino, and the CAN bus shield with CAN ternimal connections at the very top. Software development for the arduinos was done using the arduino software. Sensor arduino has three main sections:

- CAN bus functionality
- Tank pressure sensing
- Sending data over UDP

Additional libraries required for the sensor arduino include the mcp_can.h for the CAN bus, Ethernet.h and EthernetUdp.h for UDP communication. Initial setup includes the setting up the buffers, defining the source and destination IP addresses and communication ports for UDP communication. The arduino sends data to the raspberry pi (controller). The outbound buffer from the arduino includes the the calculated tank pressure (psi),upstream and downstream NO_X, exhaust temperature, and arduino time stamp which is sent to the controller.

7.1.2 Load Bank Arduino

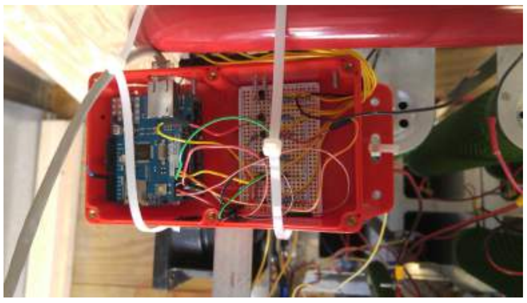


Figure 7.2.: Arduino housing for the load bank

Load bank arduino controls the load bank commands. The test manager sends commands to the controller, which in turn communicates to the load bank arduino to change the load settings for the test. In order to achieve this, even the load bank arduino uses as ethernet shield as can be seen in figure 7.2.The housing is mounted on the load bank itself. The prototype SCR system used serial communication to adjust the load bank settings, which was later updated to UDP communication. The software again developed in arduino software, and uses Ethernet.h and EthernetUdp.h libraries and setup similar to sensor arduino, for UDP communication. It has two sections:

- Load settings
- Sending and receiving data over UDP

The load bank is designed to simulate different driving load conditions. Using the load bank arduinos and a combination of five different resistors, we dissipate power ranging from 0 - 6000 watts. Using switch case, 25 different combinations are used to match the load requirements. UDP communication is used to send the combination in use and the current drawn by the load bank. Load bank arduino is also programmed to receive load setting commands over the same UDP connection.

7.2 Raspberry Pi

In the process of continuous development of the SCR system, we use raspberry pi 2 (Pi) model B as the main controller for testing the current SCR design. Pi has a 900Mhz quadcore CPU and 1GB ram, which gives us faster and reliable processing. Matlab and Simulink have a support package for the raspberry pi 2 model B, using the same package we run simulink and matlab codes on the Pi. Pi acts as the main controller for the whole SCR system, it connects the sensing and actuation together. The arduino reads/senses data form the engine and sends it over to the Pi, which then processes the data, determines and executes the changes required, and sends all information over to the test manager.

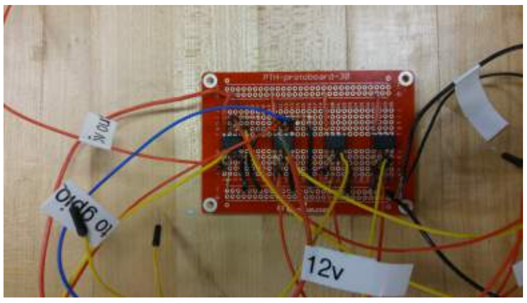
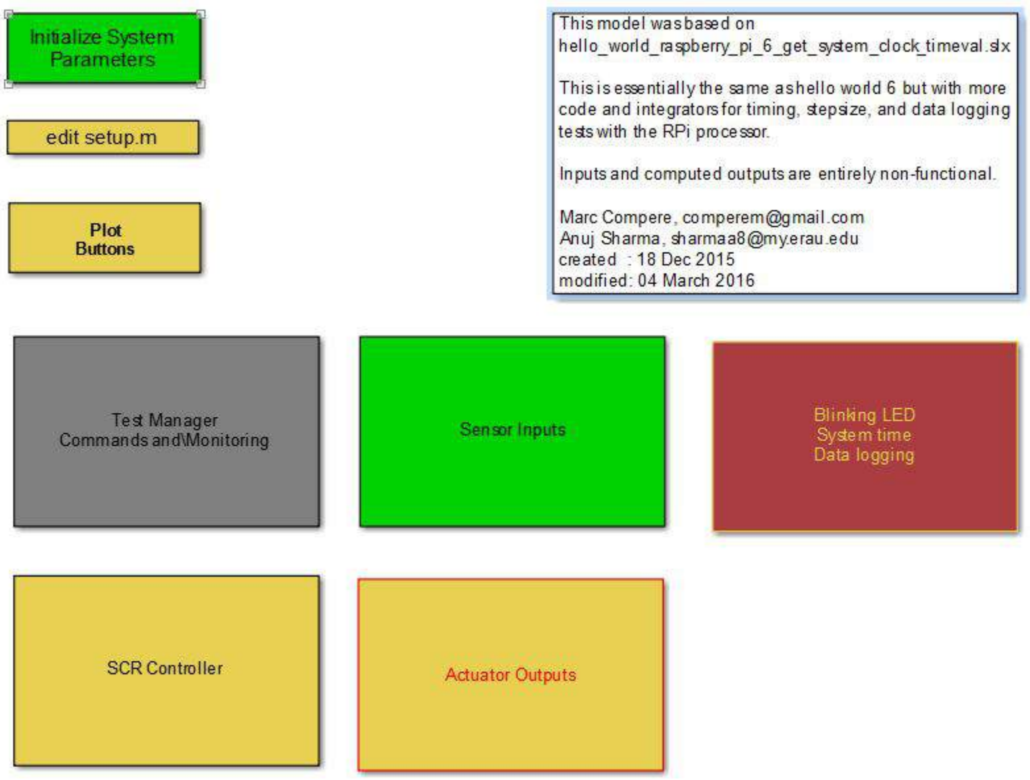


Figure 7.3.: Operational amplifier circuit

Pi has 40 GPIO (general purpose input/output) pins, which can be used to actuate and read data. We use the GPIO pins to actuate the pump, injector, and heaters. All output commands require a 5V signal to actuate while Pi only provides 3.3V output from the GPIO pins. To overcome this issue, we designed and developed a voltage amplifier circuit using LM358 operational amplifiers as seen in figure 7.3. LM358 is a general purpose amplifier that provides two independent channels for amplification. The pinout for the operational amplifier is included in appendix F. The operational amplifier circuit provides amplification for six channels, out of which five are used and one is redundant. We use GPIO pins to control the pump, injector and three heaters.


Initialize System Parameters
edit setup.m
Plot Buttons
Test Manager Commands and\Monitoring
Sensor Inputs
Blinking LED System time Data logging
SCR Controller
Actuator Outputs

This model was based on
hello_world_raspberry_pi_6_get_system_clock_timeval.slx

This is essentially the same as hello world 6 but with more code and integrators for timing, stepsize, and data logging tests with the RPi processor.

Inputs and computed outputs are entirely non-functional.

Marc Compere, comperem@gmail.com
Anuj Sharma, sharmaa8@my.erau.edu
created : 18 Dec 2015
modified: 04 March 2016

Figure 7.4.: Top level simulink code for raspberry pi

Figure 7.4 is a screen shot of the simulink code for raspberry pi. The first block of code (code in appendix B) interacts with the test manager, it includes sending data and receiving data to and from the test manager.Test manager commands are detailed in the next section. Second block of code (code in appendix C) for sensor inputs receives UDP packets containing sensor data from the sensor arduino and load bank arduino. The load bank arduino sends relay states, current settings and arduino time stamp to the controller. The sensor arduino sends DEF tank pressure, NO_X values, exhaust temperature and arduino time stamp to the controller. The next code block, Blinking LED, system time and data logging is used for monitoring purposes. It includes a code to blink an LED on the pi, which is used as an indication for a functioning code. The block also includes code for data logging, which logs all recorded data to file on the pi which is sent over to the test manager after testing.The SCR controller block computes the right amount of injection required for the minimum amount of tailpipe NO_X and ammonia. The term used for excess ammonia in exhaust is referred to as ammonia slip. It can also be referred to as emissions of unreacted ammonia resulting from incomplete reaction of NO_X with DEF (Environmental Protection Agency, n.d.). The SCR controller code block prevents ammonia slip and ensures minimal NO_X emissions by estimating right DEF injections from exhaust feedback. The SCR controller block uses exhaust upstream NO_X and downstream NO_X, and exhaust temperature. It also makes use of Enerac 700 emissions analyzers to measure downstream NH_3 content.

The last block, as the name suggests, sends commands to the actuators(code in appendix D). It controls the pump, injector, line heater, pump heater, tank heater and also sends UDP packets with load settings to the load bank arduino.The sensor arduino sends the pressure (in psi), upstream and downstream NO_X values (in ppm), exhaust temperature (in celsius) and arduino time stamp. The pi uses the pressure data to maintain the DEF tank pressure at 20 psi. We used a thermostat type control for the pump to maintain required pressure. If the pressure is below 20 psi the pump stays high, and for pressure over 20 psi the pump state is switched to low. We achieved this functionality using a simple state machine. Pump control also includes a custom defined PWM (pulse width modulation) generation block, which is used to generate a 100Hz signal for the pump. The pump duty cycle is kept constant at 10% to provide for better control over the tank pressure regulation. The current SCR design allows us to set injector duty cycle from the test manager. The set value is sent to the controller and later internally fed to our custom defined PWM generation block to change injector duty cycle as and when changed in the test manager. We use a 3Hz PWM signal for the injector. There are three heaters for the SCR system namely, line heater, pump heater and the tank heater. Each heaters receives only a on/off command from the controller. A simple high or a low state is sent to the heater GPIO pins to control the heaters.

7.3 Test Manager

SCR system testing is monitored and controlled using a windows operating system located away from the test shed, thus providing a safe environment for testing the system. Test manager is like a control station providing full control over the SCR system testing. Test manager communicates with the entire system via the pi controller. It uses a wireless bridge to the system's UDP network as seen in figure 6.2. Another system is used to monitor live video feed from the IP cameras installed in the test shed over the same wireless bridge network.

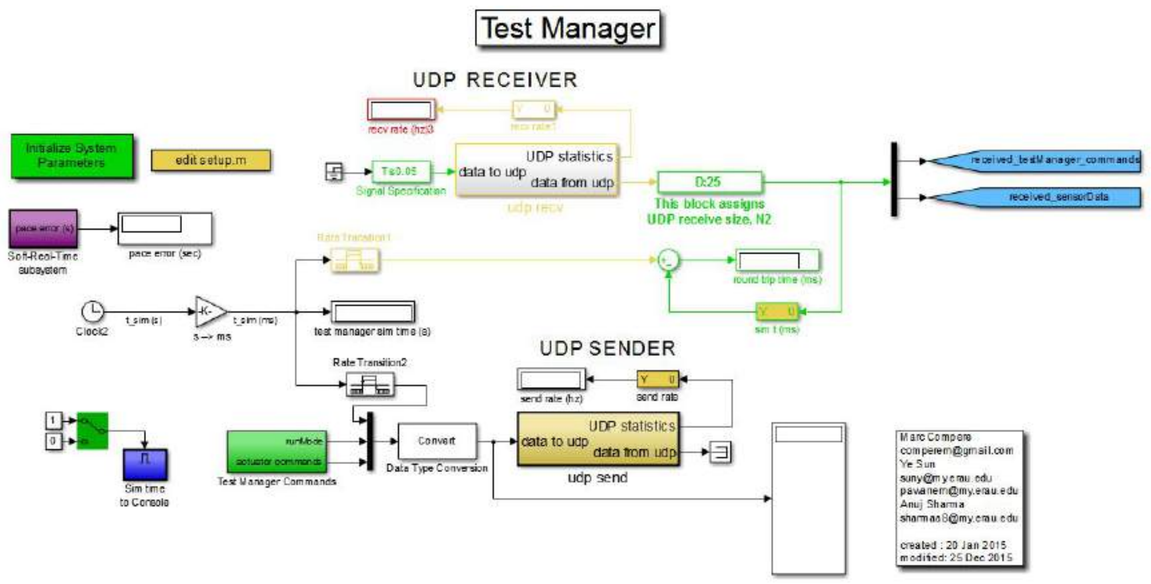


Figure 7.5.: Simulink code for test manager

Figure 7.5 is the top level simulink code for test manager. It allows the user to manually select different controllers for communication also turn on/off data logging. Test manager has three important sections, UDP sender, UDP receiver and the test manager actuator commands. UDP sender and receiver manage the UDP incoming and outgoing messages. UDP sender sends out the test manager actuator commands to the controller and the receiver receives the sensor data, and an echo of test manager commands from the controller.

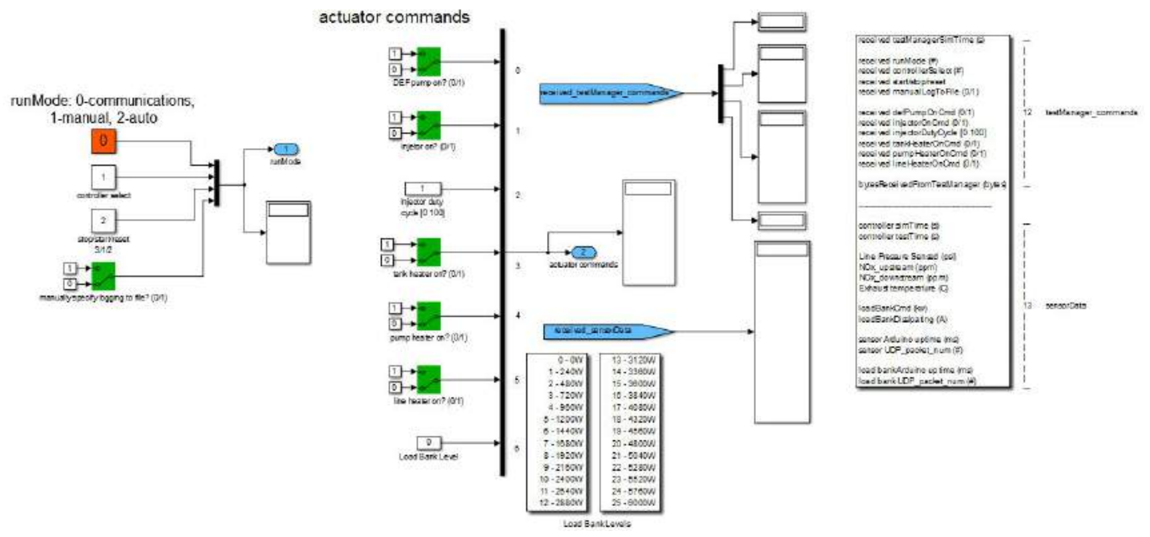


Figure 7.6.: Simulink code for test manager actuator commands

The main function of the test manager is to send out the actuator commands. The test manager can manually turn on/off the pump, injector, tank heater, line heater, pump heater. Also, the test manager allows to change the injector duty cycle and load bank load settings. The injector duty cycle can be varied between 1-99 (percent), and the load settings take inputs between 0-25. Every number for load setting stands for a specific power rating as indicated in figure 7.6. The test manager sends digital on/off commands to the controller which then actuates the hardware as required(code in appendix E). The digital on/off signals in the form of 1s and 0s are sent back to the test manager and displayed on the test manager simulink code as a means to ensure proper communication. All sensor data from the sensor arduino and load bank arduino is also communicated to the test manager to create a test log with all data recorded after a test.

7.4 Testing Procedure

The SCR system test is conducted in the test shed while it is monitored and controlled using the test manager located away from the test shed. Prior to testing:

1. Ensure the test shed doors are latched open.
2. All testing lights are on and fire extinguisher is kept in place.
3. Vent fan is switched on and the load bank placed close to the vent fan.
4. Connect the load bank to the engine, and close the quick disconnect switch for engine power.
5. Check for the correct operation of the network components in the system.
6. Test IP camera network connection.
7. Make sure the required commands are set on the test manager and data logging is switched on.
8. Turn on the engine ensuring no one is present in the shed beyond the caution tapes.

Once the engine is on, the load settings and DEF injection rate can be adjusted using the test manager. If the test is being conducted with the control algorithm for injection rate, the injection rate will be adjusted by the controller to keep the tailpipe NO_X and NH_3 concentration to a minimum. With the engine and the SCR system running, the sensor arduino will sense the DEF tank pressure, engine upstream and downstream NO_X, exhaust temperature and send all data to the controller. The load bank arduino waits for a command from the controller to change the engine load, and sends back the relay settings and current drawn to the controller. The controller, receives all sensor data from both the arduinos, and maintains the tank pressure to 20 psi. All sensor data is processed by the controller to determine the necessary injection rate and is sent over to the test manager. The test manager displays all received data and allows to change the test parameters at any point of time while testing the SCR system.

8. Summary and Conclusions

8.1 Summary

Improving fuel economy has always been a challenge in the automotive industry. With the corporate average fuel economy (CAFE) standards for the average fuel economy for cars being increased from 27.5 mpg (miles per gallon) to 37.8 mpg by 2016, which will further be increased to 54.5 mpg by 2025 (Car & Driver, 2015), automakers will need newer and advanced technological improvements. Diesel engines attend to the mileage standards, but increase problems pertaining to environmental protection. This research work introduces the use of enhanced SCR systems with diesel hybrid engines. SCR systems are currently considered as the best solution to reduce the harmful emissions from diesel engines (Forum, 2015).The SCR system presented in this research work utilizes the following hardware:

- 7000 watt Diesel Generator
- DEF tank with pump and line assemble from Bosch and General Motors
- Custom designed test stand to hold the engine and exhaust
- Two NO_X sensors from general motors installed in the exhaust to read upstream and downstream exhaust
- Enerac Emission Analyzer to provide for correct NO_X and NH_3 concentration

For automation and embedded control the following electrical components were used:

- Raspberry Pi 2 model B and AdaFruit as the main controller to actuate hardware
- LM358 operational amplifiers for GPIO output voltage amplification
- Arduino with a CAN bus shield and and ethernet shield to sense and report engine data
- Arduino with an ethernet shield to control the load bank
- MOSFETs to amplify sensor arduino output current

Additionally more network components were used to establish a network connection for the SCR system. The whole system put together helps meet the CAFE standards with lower exhaust emissions. The approach presented here reduces exhaust emissions from diesel engines using arduino to sense engine data and raspberry pi to actuate SCR system. The control network connects the sensing and actuation to the test manager thus allowing the whole test station to be monitored and controlled over the wireless network.

8.2 Conclusion

The following objectives were achieved in developing the SCR system:

1. A designated testing area that allows continuous monitoring with IP cameras, was built to safely test the SCR system. The test are will be continually used in future for research and development of diesel engine exhaust emissions reduction systems.

2. Diesel oxidation catalyst was integrated to enhance the operation of SCR system.

3. Established a dedicated UDP network connection to enable communication between sensing and actuation.

4. Integrated raspberry pi as the main controller.

5. Developed an algorithm to control the diesel exhaust fluid injection rate and achieve pressure control for diesel exhaust fluid tank.

6. Developed a new test stand managing system to provide complete control and monitoring of the test parameters using Matlab/Simulink.

Figure 8.1 is a plot of downstream NO_X concentration (ppm) against time. The figure clearly indicates the reduction in NO_X concentration as and when manual DEF injections are provided.

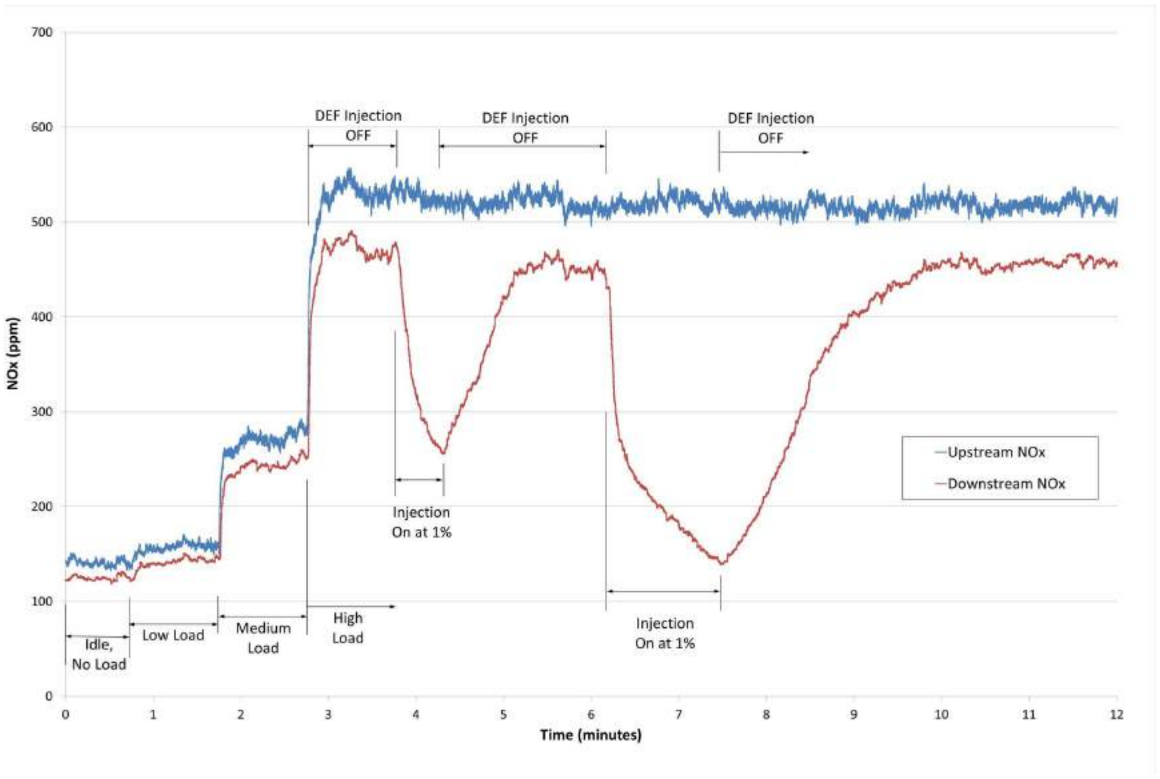


Figure 8.1.: Downstream NO_X concentration (ppm) vs time (minutes)

These results were achieved without the SCR controller being used. The SCR controller calculates DEF injection rate based on the exhaust downstream NO_X concentration.

8.3 Future Recommendations

The project is currently in the initial development stages and requires more refined output. From the current stage, as a recommendation for development:

1. Update simulink code for SCR controller to fine tune the DEF injections rate to minimize NO_X emissions while keeping minimum ammonia at exhaust.

2. Integrate ENERAC 700 emission analyzers to the SCR system.

3. ENERAC 700 emission analyzer results should be compared to the NO_X sensor results for accurate measurements.

4. Improve test manager functionality to include emergency stop for the engine.

REFERENCES

Aerinox-inc.com. (2015). *Aerinox - emission reduction.* Retrieved from http://www.aerinox-inc.com/technology/

Agengy, E. P. (2010, May). *Diesel oxidation catalyst general information.* Retrieved from http://www3.epa.gov/otaq/diesel/documents/420f10031.pdf

Car, & Driver. (2015). *How automakers will meet 2016 cafe standards.* Retrieved from http://www.caranddriver.com/features/how-automakers-will-meet-2016-cafe-standards

Chen, P., & Wang, J. (2013, June). Integrated diesel engine and selective catalytic reduction system active nox control for fuel economy improvement. In *American control conference (acc), 2013* (p. 2196-2201). doi: 10.1109/ACC.2013.6580161

Chen, P., & Wang, J. (2014, June). Estimation of automotive urea-based selective catalytic reduction systems during low temperature operations. In *American control conference (acc), 2014* (p. 1523-1528). doi: 10.1109/ACC.2014.6859044

Devesh Upadhyay, V. N. (2005). Model based analysis and control design of a urea-scr denox aftertreatment system. *Journal of Dynamic Systems, Measurement, and Control.*

Dieselnet.com. (2015). *Diesel oxidation catalyst.* Retrieved from https://www.dieselnet.com/tech/cat_doc.php#form

EBMPAPST. (n.d.). *Clean standard.* Retrieved from http://mag.ebmpapst.com/en/products/pumps/clean-standard_4834/2/

Elliott, M. A., Nebel, G. J., & Rounds, F. G. (1955). "the composition of exhaust gases from diesel, gasoline and propane powered motor coaches". *Journal of the Air Pollution Control Association*, *5*(2), 103-108. Retrieved from http://dx.doi.org/10.1080/00966665.1955.10467686 doi: 10.1080/00966665.1955.10467686

Environmental Protection Agency. (n.d.). *Air pollution control technology fact sheet.* Retrieved from https://www3.epa.gov/ttncatc1/dir1/fscr.pdf

Fairhurst, G. (2008, November). *User datafram protocol (udp).* Retrieved from http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/udp.html

Forum, D. T. (2015). *What is scr?* Retrieved from http://www.dieselforum.org/about-clean-diesel/what-is-scr

Heywood, J. B. (1988). *Internal combiustion engine fundamentals* (J. P. Holman, Ed.). McGraw Hill Inc.

Hsieh, M. F., & Wang, J. (2010, June). Staircase ammonia coverage ratio profile control for diesel engine two-cell selective catalytic reduction systems. In *American control conference (acc), 2010* (p. 3003-3008). doi: 10.1109/ACC.2010.5531260

Hsieh, M. F., & Wang, J. (2011, Nov). A two-cell backstepping-based control strategy for diesel engine selective catalytic reduction systems. *Control Systems Technology, IEEE Transactions on*, *19*(6), 1504-1515. doi: 10.1109/TCST.2010.2098477

Inc.", C. D. T. (2015). *Diesel oxidation ccatalyst (docs).* Retrieved from `http://www.cdti.com/content/americas/products/docs.htm`

Inc., M. T. (2007). Mcp2515 - stand-alone can controller with spi interface [Computer software manual]. Retrieved from `https://www.sparkfun.com/datasheets/DevTools/Arduino/MCP2515.pdf`

Inc., M. T. (2010). Mcp2551 - high-speed can transceiver [Computer software manual]. Retrieved from `https://www.sparkfun.com/datasheets/DevTools/Arduino/MCP2551.pdf`

Inc., N. T. (n.d.-a). *What are diesel emissions?* Retrieved from `http://www.nettinc.com/information/emissions-faq/what-are-diesel-emissions`

Inc., N. T. (n.d.-b). *What is a diesel oxidation catalyst?* Retrieved from `http://www.nettinc.com/information/emissions-faq/what-is-a-diesel-oxidation-catalyst`

J. Chi, H. D. (2005). "modeling and control of a urea-scr aftertreatment system". *SAE International Journal.*

Karthikeyan, S., & Krishnan, S. (2010, Nov). Computational analysis for selection of diesel oxidation catalyst and experimental investigation to meet bs-iii/iv emission norms as low cost solution for lcv applications. In *Frontiers in automobile and mechanical engineering (fame), 2010* (p. 92-96). doi: 10.1109/FAME.2010.5714806

Kitamura T, S. J., Ito T, & H, F. (2002). Mechanism of smokeless diesel combustion with oxygenated ffuel based on the dependence of the equivalence ratio and temperature on soot particle formation. *International Journal of Engine Research*, *3*(4), 223-248.

Maruthi Devarakonda, G. P., & Johnson, J. H. (2009). Model based estimation and control system development in a urea-scr aftertreatment system. *SAE International Journal.*

Ming-Feng Hsieh, J. W. (2012). Adaptive and efficient ammonia storage distribution control for a two-catalyst selective catalytic reduction system. *Journal of Dynamic Systems, Measurement, and Control.*

Ming Zheng, J. G. H., Graham T. Reader. (2004). Diesel engine exhaust gas recirculation-a review on advanced and novel concepts. *Energy Conversion and Management.*

MTU-Report. (2014). *How does exhaust gas recirculation work?* Retrieved from `http://www.mtu-report.com/Technology/Research-Development/How-does-Exhaust-Gas-Recirculation-work`

Pakanati, C., Padmavathamma, M., & Reddy, N. (2015, Feb). Performance comparison of tcp, udp, and tfrc in wired networks. In *Computational intelligence communication technology (cict), 2015 ieee international conference on* (p. 257-263). doi: 10.1109/CICT.2015.37

Postel, J. (1980). User datagram protocol. *ISI.*

Prabhakar, S., Karthikeyan, M., Annamalai, K., & Banugopan, V. (2010, Nov). Control of emission characteristics by using selective catalytic reduction (scr) in d.i. diesel engine. In *Frontiers in automobile and mechanical engineering (fame), 2010* (p. 104-107). doi: 10.1109/FAME.2010.5714808

Prasad, R., & Bella, V. R. (2010). A review on diesel soot emissions, its effect and control. *Bulletin of Chemical Reaction Engineering & Catalysis*, 69-86. Retrieved from `ejournal.undip.ac.id/index.php/bcrec/article/view/794` doi: 10.9767/bcrec.5.2.794.69-86

Rajaboina, R., Reddy, P., & Kumar, R. (2015, Feb). Performance comparison of tcp, udp and tfrc in static wireless environment. In *Electronics and communication systems (icecs), 2015 2nd international conference on* (p. 206-212). doi: 10.1109/ECS.2015.7124893

Rasheed, W., Goyal, P., & Joseph, C. (2013, April). Model based control for a selective catalytic reduction (scr) system in an exhaust gas aftertreatment system for a diesel engine. In *Energy efficient technologies for sustainability (iceets), 2013 international conference on* (p. 744-749). doi: 10.1109/ICEETS.2013.6533477

Sharma, A. (2016, April). A design to improve selective catalytic reduction system. In *Control and automation.* IEEE SouthEast Con.

Sinzenich, H., & Wehler, K. (2014). *Selective catalytic reduction: Exhaust aftertreatment for reducing nitrogen oxide emissions.*

The Union of Concerned Scientists. (n.d.). *Diesel engines and public health.* Retrieved from `http://www.ucsusa.org/clean_vehicles/why-clean-cars/air-pollution-and-health/trucks-buses-and-other-commercial-vehicles/diesel-engines-and-public.html#.Vq-Rt_krK00`

The Union of Concerned Scientists. (2015). *Cars, trucks, and air pollution.* Retrieved from `http://www.ucsusa.org/clean_vehicles/why-clean-cars/air-pollution-and-health/cars-trucks-air-pollution.html`

Theaa.com. (2015). *Diesel particulate filter.* Retrieved from `http://www.theaa.com/motoring_advice/fuels-and-environment/diesel-particulate-filters.html`

"University of California, B. (n.d.). *"lecture 7 transport protocols: Udp, tcp".* Retrieved from `http://robotics.eecs.berkeley.edu/~wlr/12203/transport-slides.pdf`

Upadhyay, D., & Nieuwstadt, M. V. (2002, November). Modeling of a urea scr catalyst with automotive applications. In *Asme 2002 international mechanical engineering congress and exposition* (p. 707-713). doi: 10.1115/IMECE2002-32104

Wahono, B., Ogai, H., Ogawa, M., Kusaka, J., & Suzuki, Y. (2012, Dec). Diesel engine optimization control methods for reduction of exhaust emission and fuel consumption. In *System integration (sii), 2012 ieee/sice international symposium on* (p. 722-727). doi: 10.1109/SII.2012.6427303

Wang, Y., Zhang, H., & Wang, J. (2015). Nox sensor reading correction in diesel engine selective catalytic reduction system applications. *Mechatronics, IEEE/ASME Transactions on*, *PP*(99), 1-1. doi: 10.1109/TMECH.2015.2434846

WIZnet. (2013). W5500 datasheet [Computer software manual]. Retrieved from `https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/W5500_datasheet_v1.0.2_1.pdf`

Zhai, X., Chen, Z., Yao, J., Wang, H., & Li, J. (2009, July). Selective catalytic reduction of no with nh3 over wire-mesh honeycomb supported v2o5 -moo3/tio2/al2o3 catalyst. In *Control, automation and systems engineering, 2009. case 2009. iita international conference on* (p. 462-465). doi: 10.1109/CASE.2009.131

9. Appendix

9.1 Appendix A: Arduino Codes

```
/*
mySCR_V11.ino:
 This sketch sends udp out with arrays of uint32_t's
 The UDP recieve is commented out for development
  with Raspberry Pi

 This sketch is designed to work well with
 Simulink's: Test_Manager_v03.slx

 We assume the Arduino Uno R3 with CAN bus
 and an Ethernet shield.

 based on:
 created 21 Aug 2010
 by Michael Margolis

 This code is in the public domain.
```

```
  modified by:
  Marc Compere, comperem@gmail.com
  Anuj Sharma, sharmaa8@my.erau.edu
  first modified: 02 July 2015
  last  modified: 25 March 2016

  */

#include <mcp_can.h>
#include <SPI.h>          // needed for Arduino
 versions later than 0018
#include <Ethernet.h>
#include <EthernetUdp.h>          // UDP library from:
  bjoern@cs.stanford.edu 12/30/2008
#define UDP_TX_PACKET_MAX_SIZE 860 //increase UDP size
(see http://forum.arduino.cc/index.php?topic=103502.0)
#define N_OUT_UINT32 6 // send this many 4-byte
unsigned integers (unsigned long's)
#define␣N_IN_UINT32␣11␣//␣receive␣this␣many␣4-byte
unsigned␣integers␣(unsigned␣long's)
```

```
// Enter a MAC address and IP address for your
controller below.
// The IP address will be dependent on your local network:
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0F, 0xC1, 0x23 };
// mac address of Ethernet shield (Lab's)
IPAddress ipArduino(192, 168, 11, 105);
// ip of arduino/ethernet shield
unsigned int localPort = 8888;
// local port to listen on

// remote machine settings
IPAddress remoteIp(192, 168, 11, 101);
 // IP address for sending
int         remoteport = 8000;
// outbound port for rpi2

int         retVal; // for certainty in the udp send

unsigned int udpARC   = 0;  // UDP Alive Rolling Count

// loop variables
const    long interval = 250; // interval at which
```

```
to loop in milliseconds
unsigned long currentMillis;  // for loop timing
unsigned long previousMillis; // for loop timing
unsigned long inMsgCnt  = 0;   // udp rcvd msg counter
unsigned long outMsgCnt = 0;   // udp sent msg counter

// buffers for receiving and sending data
int  outboundLen= N_OUT_UINT32 * sizeof(uint32_t) ;
int  inboundLen = N_IN_UINT32  * sizeof(uint32_t) ;
int  inboundBufsAvail; // how many bytes just arrived
byte outboundBuf[24]; // outbound message buffer;
send Simulink 30 unsigned long's,␣or␣120␣bytes
byte␣inboundBuf[44];␣//␣inbound␣message␣buffer;
receive␣10␣unsigned␣long's from Simulink, 40 bytes
uint32_t* inMsg;
uint32_t* outMsg;

// Pressure Sensor Setup
int analogPin = 5;       //Pressure sensor connected
to pin 5 on arduino
uint32_t val = 0;        //Variable to store digital
value from pressure sensor
```

```
uint32_t val_psi = 0;
//Variable to store psi value of pressure


const int Temp_Sensor = A2;


//CAN Bus
unsigned char canLen = 0;
unsigned char canBuf[8];
unsigned int  canARC = 0; // A live rolling count



long time_interval_can = 500;


unsigned long previousTime_can = 0;


unsigned long currentTime_can;


// An EthernetUDP instance to let us send
and receive packets over UDP
EthernetUDP Udp;


/* ------------------------------------------------------- */
/* ------------------------ setup() ------------------------ */
/* -------------------------------------------------------- */
void setup() {
  // start the Ethernet and UDP:
  Ethernet.begin(mac,ipArduino);
  Udp.begin(localPort); // init and begin
  listening on this port


  Serial.begin(115200);
  Serial.print("UDP_TX_PACKET_MAX_SIZE=[");
  Serial.print(UDP_TX_PACKET_MAX_SIZE);
  Serial.println("]");


  Serial.print("inboundLen=[");
  Serial.print(inboundLen);
  // note: this number must be less
  than UDP_TX_PACKET_MAX_SIZE
  Serial.println("]");


  Serial.print("outboundLen=[");
  Serial.print(outboundLen);
  // note: this number must be less
  than UDP_TX_PACKET_MAX_SIZE
  Serial.println("]");

  Serial.print("sending␣and␣receiving␣udp
␣␣␣packets␣to␣and␣from:␣[");
  Serial.print(remoteIp);
  Serial.println("]");

   // init can bus, baudrate: 500k
  if(CAN.begin(CAN_500KBPS) ==CAN_OK)
  Serial.print("can␣init␣ok!!\r\n");
  else Serial.print("Can␣init␣fail!!\r\n");

  // zero out the data buffers
  for (int i=0; i<outboundLen ; i++) {
    outboundBuf[i]=0;
  }
  for (int i=0; i<inboundLen ; i++) {
    inboundBuf[i]=0;
  }
  inMsg  = (uint32_t*) &inboundBuf[0];
  // assign uint32_t inMsg pointer  to
  head of inbound message byte array
  outMsg = (uint32_t*) &outboundBuf[0];
  // assign uint32_t outMsg pointer to
  head of outbound message byte array\

}// end setup()

unsigned char msg[8] = {255, 255, 255, 255,
 255, 255, 255, 255};
/* ------------------------------------------ */
/* ----------------- loop() ----------------- */
/* ------------------------------------------ */
void loop() {

  SendCanMessage();
  ReceiveCanMessage();

  // -------------------------------------------
  // ----- read udp when available -------------
  // -------------------------------------------
  // if there's data available, read a packet
//   inboundBufsAvail = Udp.parsePacket();
//   if(inboundBufsAvail > 0)
//   {
//      Serial.print("loop() Received packet of size: [");
//      Serial.print(inboundBufsAvail);
//      Serial.print("] (bytes), inboundLen=[");
//      Serial.print(inboundLen);
//      Serial.println("] (bytes)");
//
//      if (inboundBufsAvail == inboundLen) {
//         Udp.read(inboundBuf,inboundLen);
// inMsg now points to N uint32's from Simulink
//         udpARC++;
//      } else {
//         Serial.print("error: loop(), receiving
                  wrong number of bytes");
//␣␣␣␣}
//
//
//␣␣␣␣uint32_t␣myVar;
//␣␣␣␣Serial.print("inMsg=[");
//␣␣␣␣for␣(int␣i=0;␣i<N_IN_UINT32;␣i++){
////␣␣␣␣␣␣memcpy␣(&myVar,␣&inMsg[i],␣sizeof(uint32_t));
////␣␣␣␣␣␣Serial.print(␣myVar␣);
//␣␣␣␣␣␣Serial.print(␣inMsg[i]␣);
//␣␣␣␣␣␣Serial.print(", ");
//␣␣␣␣}
//␣␣␣␣Serial.println("]");

//␣␣}␣//␣end␣if(inboundBufsAvail␣>␣0)



␣␣/*␣----------------------------------------------------------␣*/
␣␣/*␣--------------------␣send␣udp␣every␣interval␣--------␣*/
␣␣/*␣----------------------------------------------------------␣*/

␣␣currentMillis␣=␣millis();␣//␣get␣current␣timestamp
␣␣if(currentMillis␣-␣previousMillis␣>=␣interval)␣{
␣␣␣␣previousMillis␣=␣currentMillis;␣␣␣//␣update␣previousMillis

␣␣␣␣//␣update␣data␣buffer␣with␣the␣latest␣and␣greatest
␣␣␣␣information␣from␣A/D,␣digital␣pins,␣and␣milli's␣timer
␣␣␣␣updateOutMsg(␣outMsgCnt␣);␣//␣go␣do␣everyting␣necessary
```

```
␣␣␣␣to␣update␣the␣data␣buffer␣before␣sending␣out␣over␣UDP

␣␣␣␣Serial.print("loop(): uptime, t_now = [");
␣␣␣␣Serial.print(currentMillis);
␣␣␣␣Serial.print("] (ms)");

␣␣␣␣//for␣(int␣i=0;␣i<4;␣i++){
␣␣␣␣//␣␣Serial.print("dataBuf[");
␣␣␣␣//␣␣Serial.print(i);
␣␣␣␣//␣␣Serial.print("]=[");
␣␣␣␣//␣␣Serial.print(dataBuf[i],DEC);
␣␣␣␣//␣␣Serial.print("](dec), and dataBuf[");
␣␣␣␣//␣␣Serial.print(i);
␣␣␣␣//␣␣Serial.print("]=[");
␣␣␣␣//␣␣Serial.print(dataBuf[i],HEX);
␣␣␣␣//␣␣Serial.println("](hex)");
␣␣␣␣//}

␣␣␣␣//␣send␣a␣message␣to␣the␣IP␣address␣and␣port
␣␣␣␣␣specified␣above
␣␣␣␣Udp.beginPacket(remoteIp,␣remoteport);
␣␣␣␣Udp.write(&outboundBuf[0],␣outboundLen);
```

```
    // udp deals with the byte arrays
    retVal = Udp.endPacket();
    // performs the udp send
    Serial.print (", Packet Sent Size[");
    Serial.print (outboundLen);
    Serial.println("] (bytes)");
    if (retVal>0) {
      outMsgCnt++; // increment udp send message counter
    }

    uint32_t myVarOutMsg;
    Serial.print("outMSg=[");
    for (int i=0; i<N_OUT_UINT32; i++){
      memcpy (&myVarOutMsg, &outMsg[i], sizeof(uint32_t));
      Serial.print(myVarOutMsg);
      Serial.print(", ");
    }
    Serial.println("]");

  } // end if(currentMillis - previousMillis >= interval)

 /* ——————————— Pressure Sensor ——————————— */
```

```
  val = analogRead(analogPin);    // read the input pin
  float val_psi_float = (0.1362*val)-15.052; //  may be negative
  val_psi = max(0,val_psi_float); // (psi), always > 0
  //Serial.println(val_psi);             // debug value
  //Serial.print(",");
  Serial.println(val_psi);

} // end of loop()

/* ------------------------------------------------------------ */
/* ---------------- message structures, all uint32_t's ---- */
/* ------------------------------------------------------------ */
/*
 * INBOUND MESSAGE
 * ----------------
 * No inbound message for development with rpi2
 *
 * OUTBOUND MESSAGE
 * ----------------
 * oubound message has inbound message in first 10
 uint32_t entries, then additional
 * outMsg[0] = Pressure (psi)
```

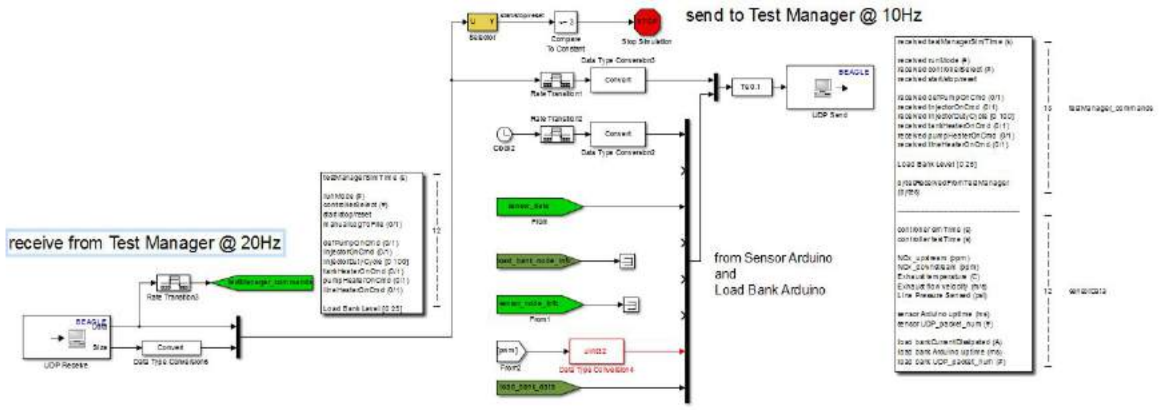
```
␣*␣outMsg[1]␣=␣NOx␣upstream␣(ppm)
␣*␣outMsg[2]␣=␣NOx␣downstream␣(ppm)
␣*␣outMsg[3]␣=␣Exhaust␣Temperature
␣*␣outMsg[4]␣=␣Arduino␣Time␣stamp
␣*␣outMsg[5]␣=␣UDP␣Message␣Count
␣*
␣*/

/*␣————————————————————————————————————————————————␣*/
/*␣————————————————␣update␣outbound␣data␣buffer␣——————␣*/
/*␣————————————————————————————————————————————————␣*/
void␣updateOutMsg(␣unsigned␣long␣outMsgCnt␣)␣{
␣//␣this␣function␣places␣two␣long␣time␣values␣(uint32's)
␣␣followed␣by␣N␣unit16's

␣/*␣————————————␣send␣pressure␣sensor␣reading␣Simulink␣—␣*/
␣memcpy(&outMsg[0],␣&val_psi,␣N_IN_UINT32*sizeof(uint32_t));

␣/*␣————————————————␣place␣Arduino␣timestamp␣————————␣*/
␣//␣insert␣the␣current␣time␣as␣an␣unsigned␣long
␣(unit32,␣4−byte)␣integer,␣arduino␣unsigned
␣long's␣are␣uint32_t's
```

```
 unsigned long now = millis(); // get current uptime
 (in milliseconds) as an unsigned long
 memcpy(&outMsg[4], &now, sizeof(uint32_t));
 // copy contents of 4-byte variable into
  dataBuf, dataBuf[0,1,2,3]

 /* ---------------------- UDP Message Count ------------------ */
 memcpy(&outMsg[5], &outMsgCnt, sizeof(uint32_t));

// Serial.print("updateOutMsg() uptime, t_now = [");
// Serial.print(now);
// Serial.println("] (ms)");

} // end updateData()

//send message to CAN Bus to initialize the NOx sensors
void SendCanMessage()
{
currentTime_can = millis();
if (currentTime_can - previousTime_can > time_interval_can) {
    CAN.sendMsgBuf(0x493, 0, 8, msg);
     // 0x493 is hex for 1171 decimal, CanID=1171
    previousTime_can = currentTime_can;
  }
}



//CAN Bus message AND serial sending
void ReceiveCanMessage()
{
    int temp_voltage = 0; // 2 byte value
    int temp_LSB     = 0; // (LSB) least significant byte
    int temp_MSB     = 0; // (MSB) most significant byte
    int temp         = 0;
    int simplePrint  = 1;
    int exhaust_temp;
    if(CAN_MSGAVAIL == CAN.checkReceive())
    {

      CAN.readMsgBuf(&canLen, canBuf);
      // read data,  len: data length, buf: data buf



      canARC++; // can message alive
      rolling count, uint8's are on [0 254]


      // temp sensor LSB and MSB
      temp_voltage = analogRead(A2);
      // split this into two numbers
      temp_LSB         = temp_voltage % 256;
      // modulo delivers the remainder
      temp_MSB         = temp_voltage / 256;
      // division calculates the primary divisor
      temp = word(temp_MSB,temp_LSB);
      exhaust_temp = temp_LSB + (8*temp_MSB);


      if (simplePrint == 1){


        if(CAN.getCanId()==177){
          Serial.print(millis());
          Serial.print(", ");
          //Serial.print((word(canBuf[0],canBuf[1])*.1)-100);
          // (ppm), word(highByte, lowByte)
          uint32_t NOx_upstream_ppm =
          (uint32_t) (word(canBuf[0],canBuf[1])*.1)-100;
          // 16-bit number cast into a 32-bit var
```

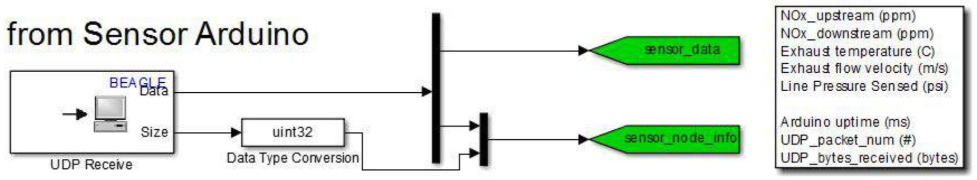
```
␣␣␣␣␣␣␣␣␣␣Serial.print␣(NOx_upstream_ppm);
␣␣␣␣␣␣␣␣␣␣Serial.print(", ");
␣␣␣␣␣␣␣␣␣␣outMsg[1]␣␣=␣(uint32_t)␣NOx_upstream_ppm;
␣␣␣␣␣␣␣␣}
␣␣␣␣␣␣␣␣if(CAN.getCanId()==178){
␣␣␣␣␣␣␣␣␣␣//Serial.print((word(canBuf[0],canBuf[1])*.1)-100);
␣␣␣␣␣␣␣␣␣␣uint32_t␣NOx_downstream_ppm␣=
␣␣␣␣␣␣␣␣␣␣(uint32_t)␣((word(canBuf[0],canBuf[1])*.1)-100);
␣␣␣␣␣␣␣␣␣␣//␣16-bit␣number␣cast␣into␣a␣32-bit␣var
␣␣␣␣␣␣␣␣␣␣Serial.print␣(NOx_downstream_ppm);
␣␣␣␣␣␣␣␣␣␣Serial.print(", ");
␣␣␣␣␣␣␣␣␣␣Serial.print(temp_LSB);
␣␣␣␣␣␣␣␣␣␣Serial.print(", ");
␣␣␣␣␣␣␣␣␣␣Serial.println(temp_MSB);
␣␣␣␣␣␣␣␣␣␣outMsg[2]␣␣=␣(uint32_t)␣NOx_downstream_ppm;
␣␣␣␣␣␣␣␣}
␣␣␣␣␣␣}


␣␣␣␣␣␣//␣safety/debugging␣-␣ensure␣canLen␣no␣larger␣than␣8
␣␣␣␣␣␣if␣(canLen>8)␣canLen=8;


␣␣␣␣␣␣int␣offset=10;
```

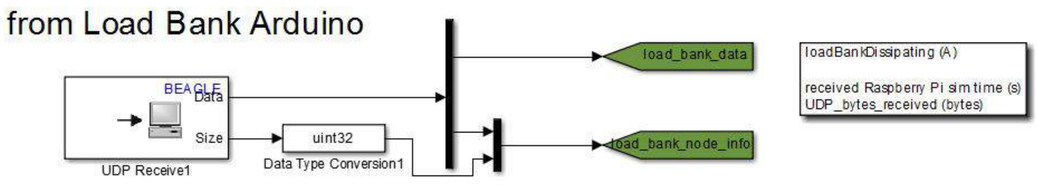
```
␣␣␣␣␣␣if(␣(CAN.getCanId()==177)␣||
␣␣␣␣␣␣((CAN.getCanId()==178))␣){

␣␣␣␣␣␣␣␣outMsg[3]␣␣=␣(uint32_t)␣exhaust_temp;
␣␣␣␣␣␣␣␣//Copy␣exhaust␣temperature␣to␣outMsg[3]

␣␣␣␣␣␣␣␣temp_voltage␣=␣analogRead(A2);
␣␣␣␣␣␣␣␣//␣split␣this␣into␣two␣numbers
␣␣␣␣␣␣␣␣temp_LSB␣␣␣␣␣␣␣␣=␣temp_voltage␣%␣256;
␣␣␣␣␣␣␣␣//␣modulo␣delivers␣the␣remainder
␣␣␣␣␣␣␣␣temp_MSB␣␣␣␣␣␣␣␣=␣temp_voltage␣/␣256;
␣␣␣␣␣␣␣␣//␣division␣calculates␣the␣primary␣divisor

␣␣␣␣␣␣}␣//␣if(␣(CAN.getCanId()==177)␣||
␣␣␣␣␣␣((CAN.getCanId()==178))␣)

␣␣␣␣}␣//␣end␣if(CAN_MSGAVAIL␣==␣CAN.checkReceive())
}
```

9.2 Appendix B: Raspberry pi simulink code for test manager

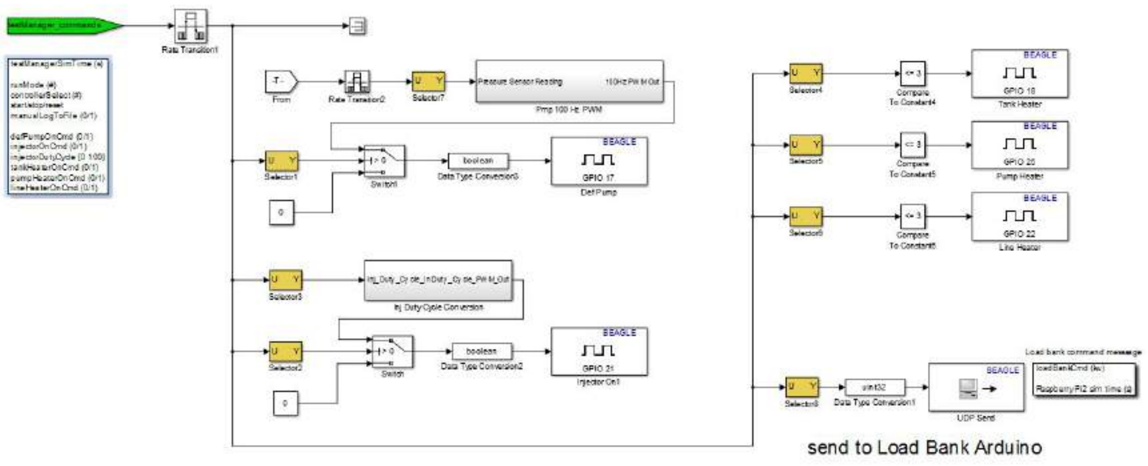


9.3 Appendix C: Raspberry pi simulink code for sensor inputs

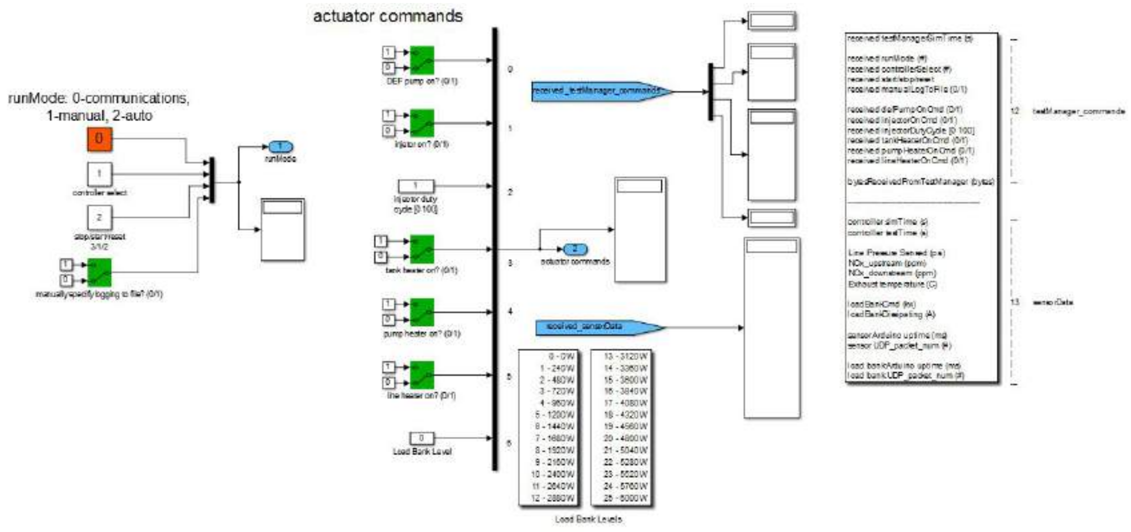




9.4 Appendix D: Raspberry pi simulink code for actuator commands



9.5 Appendix E: Test manager actuator commands



9.6 Appendix F: Datasheets

LM258, LM358, LM358A, LM358E, LM2904, LM2904A, LM2904E, LM2904V, NCV2904

ON Semiconductor®

www.onsemi.com

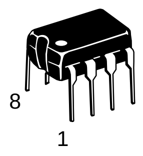
Single Supply Dual Operational Amplifiers

Utilizing the circuit designs perfected for Quad Operational Amplifiers, these dual operational amplifiers feature low power drain, a common mode input voltage range extending to ground/V_{EE}, and single supply or split supply operation. The LM358 series is equivalent to one−half of an LM324.

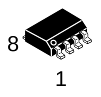
These amplifiers have several distinct advantages over standard operational amplifier types in single supply applications. They can operate at supply voltages as low as 3.0 V or as high as 32 V, with quiescent currents about one−fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

Features

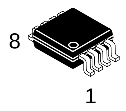
- Short Circuit Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Single and Split Supply Operation
- ESD Clamps on the Inputs Increase Ruggedness of the Device without Affecting Operation
- NCV Prefix for Automotive and Other Applications Requiring Unique Site and Control Change Requirements; AEC−Q100 Qualified and PPAP Capable
- These Devices are Pb−Free, Halogen Free/BFR Free and are RoHS Compliant



PDIP−8
N, AN, VN SUFFIX
CASE 626

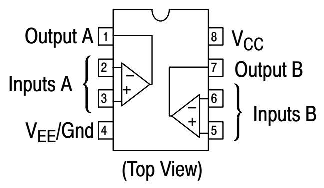


SOIC−8
D, VD SUFFIX
CASE 751



Micro8™
DMR2 SUFFIX
CASE 846A

PIN CONNECTIONS

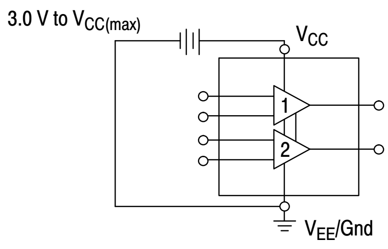


ORDERING INFORMATION

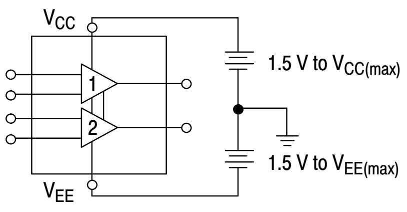
See detailed ordering and shipping information in the package dimensions section on page 10 of this data sheet.

DEVICE MARKING INFORMATION

See general marking information in the device marking section on page 11 of this data sheet.



Single Supply



Split Supplies

Figure 1.

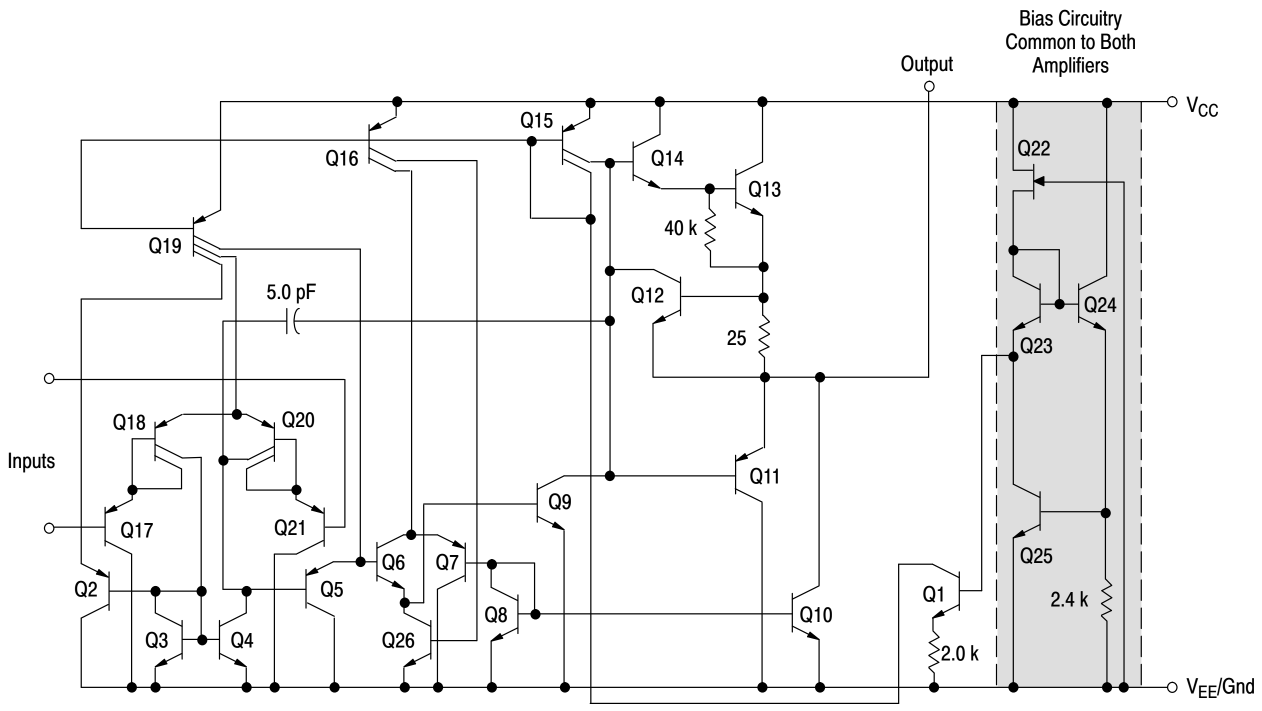


Figure 2. Representative Schematic Diagram
(One−Half of Circuit Shown)

MAXIMUM RATINGS (T_A = +25°C, unless otherwise noted.)

Rating	Symbol	Value	Unit
Power Supply Voltages			Vdc
Single Supply	V_{CC}	32	
Split Supplies	V_{CC}, V_{EE}	±16	
Input Differential Voltage Range (Note 1)	V_{IDR}	±32	Vdc
Input Common Mode Voltage Range	V_{ICR}	−0.3 to 32	Vdc
Output Short Circuit Duration	t_{SC}	Continuous	
Junction Temperature	T_J	150	°C
Thermal Resistance, Junction−to−Air (Note 2) Case 846A	$R_{\theta JA}$	238	°C/W
Case 751		212	
Case 626		161	
Storage Temperature Range	T_{stg}	−65 to +150	°C
Operating Ambient Temperature Range	T_A		°C
LM258		−25 to +85	
LM358, LM358A, LM358E		0 to +70	
LM2904, LM2904A, LM2904E		−40 to +105	
LM2904V, NCV2904 (Note 3)		−40 to +125	

Stresses exceeding those listed in the Maximum Ratings table may damage the device. If any of these limits are exceeded, device functionality should not be assumed, damage may occur and reliability may be affected.
1. Split Power Supplies.
2. All $R_{\theta JA}$ measurements made on evaluation board with 1 oz. copper traces of minimum pad size. All device outputs were active.
3. *NCV2904 is qualified for automotive use.*

ESD RATINGS

Rating	HBM	MM	Unit
ESD Protection at any Pin (Human Body Model − HBM, Machine Model − MM)			
NCV2904 (Note 3)	2000	200	V
LM358E, LM2904E	2000	200	V
LM358DR2G, LM2904DR2G	250	100	V
All Other Devices	2000	200	V