



DEPARTMENT OF MECHANICAL, AEROSPACE & CIVIL

ENGINEERING

MAJOR GROUP PROJECT – ME5538

DRIVERS CONTROL FOR FORMULA STUDENT CAR



ECU AND MAPPING

ARAVIND SANTHOSH KUMAR

2043039

SUPERVISOR: PROF. THANOS MEGARITIS

CO-SUPERVISOR: LIONEL GANIPPA

Abstract

This report illustrates the Electronic Control Unit (ECU) selection process along with mapping and calibration of the selected sensors and actuators of a Formula Student car and the procedures for designing a datalogger for the car.

The entire electronic components chosen for the car are explained in detail along with their purpose, advantages, and disadvantages. The system is divided into ECU, sensors, actuators, mapping, and data logging. It was really a challenging task to select the appropriate ECU for the new engine. Constant contact with the manufactures made it possible to opt the ECU with detailed specifications and wiring diagram.

Data logger and transmission of real-time data (telemetry data) using Radiofrequency methods are developed for improved performance and for identifying the defects, as well as the real-time status of the running car. Also, the scope for future development and improvement of the data logger is suggested due to the lack of testing of the proposed system

Acknowledgement

For all those who lose their life and livelihood by the epidemic and praying for the fast recovery of the affected humankind around the world.

From the beginning till the end of the project, the weekly meeting with supervisor Prof. Thanos Megaritis helped me to clear my doubts and quires regarding not only the project but also about the course. His fast response even in this tough situation helped me to complete my works within the time frame. Also, my sincere gratitude to co-supervisor Prof. Lionel Ganippa for his support during the project phase.

At the same time, I would like to thank my entire team members along with my colleagues from other teams who stood with me for clearing my doubts and helping me to figure out the possible and best solutions. Constant communications and sharing of data between the collugues made it possible to develop the proposed design within the time frame.

Abbreviations

ECU Electronic Control Unit

FS Formula Student

CAN Controller Area Network

PWM Pulse Width Modulation

TPS Throttle Position Sensor

MAP Manifold Absolute Pressure

MAF Mass Air Flow

ATS Air- Temperature Sensor

OS Oxygen Sensor

UART Universal Asynchronous Receiver-Transmitter

OLED Organic Light-Emitting Diode

Table of Contents

Abstract.....	i
Acknowledgement	ii
Abbreviations	iii
1. Introduction	1
1.1. Formula Student Competition.....	1
1.2. Institution of Mechanical Engineers (IMechE)	1
1.3. Group Project Introduction	2
1.4. The Competition.....	2
1.5. Brunel Masters Motorsport.....	3
1.6. Group Objective	4
1.7. Personal Objective.....	4
2. Literature Review	5
2.1. Electronic Control Unit	5
2.2. ECU Mapping	7
3. Components Selected	7
3.1. Haltech Elite 1000 ECU	7
3.2. Sensors	9
3.2.1. Crankshaft & Camshaft sensor.....	11
3.2.2. Throttle Position Sensor	13
3.2.3. Oxygen Sensor.....	14
3.2.4. Air Temperature Sensor	15
3.2.5. Coolant Temperature Sensor.....	16
3.2.6. Intake Manifold Pressure Sensor.....	17
3.2.7. Wheel Speed Sensor.....	18
3.2.8. Fuel Flow Sensor.....	18
3.2.9 Brake Temperature Sensor	19
3.3. Actuators	20

3.3.1. Ignition Coil.....	20
3.3.2. Spark Plug.....	21
3.3.3. Fuel Injectors.....	22
4. Data Logger	22
4.1. Components Selected.....	23
4.1.1. ESP32 Microcontroller	23
4.1.2. I2C Oled Display	26
4.1.3. LORA SX1278 Module	28
4.2 Software Used	30
4.3 Working	30
4.3.1. Installing the Required Libraries	31
4.3.2. Transmitter:	32
4.3.3. Receiver:	33
5. Results and Discussion	34
5.1. Individual Analysis	34
5.2. Group Analysis	35
6. Conclusion	36
7. Scope for Further Research	37
8. References	38
9. Appendix	39
A1. Project Management	39
A2. Project Methodology.....	40
A3. Haltech Elite 1000 General WD (34 pin)	41
A4. Haltech Elite 1000 General WD (24 Pin).....	42
A5. Haltech Elite 1000 Proposed WD	43
A6. Data Logger and Telemetry System Script	45
A6.1 Transmitter Code	45
A6.2 Receiver Code	49
A6.3 Webpage Code	56

Table of Figures

FIGURE 1: HALTECH ELITE 1000 ECU	8
FIGURE 2: AUTOMOTIVE SENSOR OVERVIEW	10
FIGURE 3: CRANK SENSOR	11
FIGURE 4: HALL EFFECT SENSOR DESIGN	12
FIGURE 5: THROTTLE POSITION SENSOR	13
FIGURE 6: THROTTLE POSITION SENSOR DESIGN	14
FIGURE 7: OXYGEN SENSOR.....	15
FIGURE 8: AIR TEMPERATURE SENSOR	16
FIGURE 9: COOLANT TEMPERATURE SENSOR DESIGN	16
FIGURE 10: COOLANT TEMPERATURE SENSOR.....	17
FIGURE 11: MANIFOLD PRESSURE SENSOR.....	17
FIGURE 12: HALL EFFECT WORKING.....	18
FIGURE 13: FUEL FLOW SENSOR	19
FIGURE 14: IR BRAKE TEMPERATURE SENSOR	20
FIGURE 15: IGNITION COIL.....	21
FIGURE 16: SPARK PLUG	21
FIGURE 17: FUEL INJECTOR	22
FIGURE 18: ESP32 MC	23
FIGURE 19: ESP32 PINOUT DIAGRAM.....	24
FIGURE 20: OLED DISPLAY	27
FIGURE 21: LORA MODULE	28
FIGURE 22: LORA PINOUT DIAGRAM	30
FIGURE 23: BLOCK DIAGRAM OF TRANSMITTER.....	32
FIGURE 24: BLOCK DIAGRAM OF RECEIVER	33

1. Introduction

1.1. Formula Student Competition

Formula Student is an annual competition for student engineering held in the United Kingdom. Student teams from around the world design, build, test, and race small prototype racing cars. The cars are assessed on various criteria based on competition. The competition is under the governance of the Institution of Mechanical Engineers (IMechE).

There are two entry classes in the FS competition.

Class 1: - This is the biggest event in which teams compete with the cars they designed and constructed. Teams in 6 categories must be scrutinised and strictly inspected by judges before they can compete for dynamic events. 100-120 teams in this class are usually present. For the 2021 competition, 92 teams from 20 countries will participate. Out of 92 teams, 54 teams use IC engine, 28 teams use EV, and 1 team with hybrid engine.

Class 2: - This is a concept class for teams that only have a Class 1 project and plan. It can contain parts or work completed so far in the project, but this is not necessary. Teams are assessed on the presentation, cost, and design of companies. Schools are allowed to enter both Class 1 and Class 2 cars and to use Class 2 before the full entry for inexperienced students. 52 teams will compete in class 2 competition in 2021 from 16 countries. [1]

1.2. Institution of Mechanical Engineers (IMechE)

The Institution of Mechanical Engineers (IMechE) is an autonomous professional association and learned society that serves mechanical engineers and the engineering

profession. It is based in London, United Kingdom. The Institution is accredited by the Engineering Council to evaluate applicants for inclusion on its Register of Chartered Engineers, Incorporated Engineers, and Engineering Technicians, with over 120,000 members in 140 countries employed in industries such as railways, automobile, aerospace, manufacturing, oil, biomedical, and construction. [1]

1.3. Group Project Introduction

The project aims to design a race-car to take participate in the concept class category of the Formula Student 2021 competition. A full set of business plans to participate in the competition also have to be framed. In general, teams from more than 100 universities across the globe take part in the competition held at Silverstone. Due to the pandemic, the 2020 competition was cancelled and most probably the 2021 competition will be online.

1.4. The Competition

FSUK's challenge is not only a challenge for team expertise and automotive building, but it also tests the magnitude of the participants' projects and teams.

The following elements of this project will be assessed:

- Engineering Design

The engineering design holds the maximum points among the individual marking criteria with 160 points. The teams should submit an Engineering Design Report and Engineering Design Specification sheet ahead of the competition. The concept of the proposed engineering design of the car has to be demonstrated by drawing and related resources to manifest the understanding of the subject. All the noticeable and important decisions on the design have to be justified based on calculations.

- Lap Time Simulation (LTS) & Driver in the Loop (DIL)

LTS and DIL are introduced for the 2021 competition with the expectation that more teams will switch to the concept class from the FS class. The maximum awarded points will be 40 points. But the points from these individual events will not contribute to the overall concept class score.

➤ Business Plan Presentation (BPP)

At first, a Business Plan Pitch Video (BPPV) has to be submitted which must be less than 30 seconds and the Business Plan Presentation will be held on the same day of competition. The marking criteria within the BPP are divided into 9 categories including BPPV, novelty, content, etc. The maximum weightage for the BPP is 120 points.

➤ Cost and Manufacturing

This section with 150 points weightage assesses participants' understanding of the manufacturing process and component costs. A detailed bill of material, a costed bill of material, and a series of cost report documents should be submitted before the competition. [1]

1.5. Brunel Masters Motorsport

Brunel Masters Motorsport was founded in 2004 is the MSc team for Brunel University that competes in the Formula Student event that is carried out each year. For the 2021 competition, 31 students from MSc Automotive and Motorsport Engineering will participate in the concept class team. Team members have been divided into 5 groups representing powertrain, chassis, suspension, driver controls, and battery management.[2]

1.6. Group Objective

The task for the Driver Control group is to develop and optimise the driver control systems. Each task is divided between the team members to get the maximum level of driver-vehicle interaction. 7 members of the group take the responsibility for 6 individual subsystems in which the pedal box is divided among the two members. Each individual subsystem was selected by the team members based on their strong knowledge and expertise on their selected subsystem. Maximum efficiency can be obtained with the increasing driver-vehicle interface. The task of each individual team member is to develop the chosen subsystem which gives out maximum efficiency, improves and rules out the previous defects of the subsystem designed in past years. The subsystems included in the driver control groups are

- ❖ Steering
- ❖ Pedal Box
- ❖ Brake System
- ❖ Electronic shifter & Clutch
- ❖ ECU & Mapping
- ❖ Wiring Loom

This report mainly concentrates on the design and parameters opted for selecting the appropriate ECU along with mapping procedures and development of Data Logger.

1.7. Personal Objective

The main objective of the writer is to choose an ECU that is accurate and gives out maximum performance from the engine. A new ECU has to be selected and remapped along with appropriate sensors and actuators which adapt the proposed engine.

The second objective is to design a real-time data logging device that assists the team to gather the data and understand possible occurring errors while the vehicle is in

motion. Driver information display which can be mounted on the steering wheel is also developed within the project.

2. Literature Review

In consideration of the human body, the heart is the engine and ECU is the brain. ECU controls most of the subsystems of the FS car. Sensors embedded in various places within the car act as input to the ECU and the output is given out to the actuators such as start motor, injectors, etc. The whole electronic system comprises not only ECU but also sensors, actuators, fuses, battery, wiring loom.

After in-depth investigation and reviews with the team members and powertrain group, the ECU selected for the 2021 FS car is Haltech Elite 1000. There are many parameters and circumstances to be considered while choosing the ECU. For the 2021 competition powertrain group decided to use the Triumph Daytona 675 engine. The default 'Triumph ECU' is not adaptable for race purposes. External sensors that have to be integrated are limited. The digital and analog output ports are also limited number in the case of default ECU. There are also many other disadvantages of using the Triumph ECU. Data logging and data telemetry are also restricted in the case of default ECU. The ECU also cannot handle the vulnerable surroundings within the car as it is not designed for race purposes. Data logger and telemetry systems are being developed as part of the project. Even though the developed system is not mandatory there are several benefits of the system which are detailed within this report.

2.1. Electronic Control Unit

In the automobile industry, the ECU is an embedded electronic device, that reads signals coming from sensors placed at various parts and in different components of the car and depending on this information controls various important units. The ECU uses closed-loop control, a control scheme that constantly monitors the system output

to control the inputs to the system. ECU performs millions of calculations each second by gathering data from dozens of different sensors around the car.

At the heart of the ECU lies the micro-processor which processes the inputs from the engine sensors in real-time. The processor is accompanied by several memories and communication ports. The ECU contains hardware in which pre-written software is stored. All the hardware is integrated on a printed circuit board (PCB).

The basic duty of the ECU is to keep the engine running by controlling the fuel injectors, spark plugs and also to create ambient air/fuel ratio. The ECU has to deal with many other variables to work the engine in an operating window. Engine coolant temperature, air temperature, air pressure, fuel rate, etc have to be considered for the running of the engine. Different type of sensors is used to feedback data to ECU for processing.

Other potential benefits of the modern ECU are that the user can map the device according to the user's needs which may or may not improve the engine performance. The default program stored within the ECU is changed for this purpose. This is referred to as ECU mapping. [3]

The whole ECU can be sub-divided into a number of individual functional blocks:

- Power supply (analog and digital)
- Microprocessor and Memory
- Communication links (CAN bus)
- Discrete Inputs
- Frequency Inputs
- Analog Inputs
- Switch Outputs
- PWM Outputs
- Frequency Outputs

2.2. ECU Mapping

ECU mapping also called ECU tuning is the method or the process which alters the ECU's default functions to improve the vehicle performance. Mapping is done by overwriting the predefined programmes of microcontrollers within the ECU with the assistance of software. The user can configure the ECU by his/her wish based on the level of performance of the engine the user needs. The user can manage the fuel injection rate, firing timings, airflow rates, etc with the help of mapping.

Other than the default ECUs for the engine, the rest of the ECUs have to be tuned or mapped to be compatible with the engine. Most ECU brands had developed software for tuning their ECU. In the case of Haltech ECUs, Haltech ESP is the software used for mapping. The software is inbuilt with various functions which provide the user to configure the ECU in any desired way. [4]

On considering the selected ECU (Haltech Elite 1000) for the FS car, the ECU has to be mapped based on the parameters of the engine used (Triumph Daytona 675). The cylinder firing sequence has to be revised along with the ignition and fuel specifications. The list and the functions of the sensors used also have to be selected while mapping the ECU. The input and the output pins have also been configured with the help of software. The pinout description is included in Appendix A5.

3. Components Selected

3.1. Haltech Elite 1000 ECU

Haltech's Elite series ECUs are the most advanced and powerful set of ECUs which can be easily mapped for race purposes. Advanced tuning options and features make the product unique from other available products in the market. The ECU comes with 32 pin connectors (A) and 26 pin connectors(B) in which 18 output and 19 input pins

are assigned. 4 injection outputs, 4 ignition outputs, 9 user-definable outputs, and 1 dedicated output constitutes the output pin configuration. The input configuration of the ECU constitutes 5 dedicated inputs and 14 user-definable inputs. The dedicated inputs include 1 knock sensor input, 2 crank and cam pulsed inputs, 1 ignition switch along with 1 MAP sensor input. 14 user-definable inputs include 10 analogue voltage inputs along with 4 synchronised pulse inputs.[5]



Figure 1: Haltech Elite 1000 ECU

Overview

- Supports 1 to 12cylinder engines
- Normally aspirated or force induction
- Staged, sequential, semi-sequential, multipoint injection patterns
- Load sensing by throttle position (TPS), Manifold Absolute Pressure (MAP) or Mass Air Flow (MAF).
- 4 Fuel Injection Outputs
- 4 Ignition Outputs
- 10 Digital Outputs
- 10 Analogue Voltage Inputs
- 4 Synchronised Pulsed Inputs
- 3 Dedicated Inputs (knock, ignition switch MAP sensor)

- 2 Dedicated Engine Position Inputs
- Waterproof case
- Input Supply Voltage: 8VDC to 16VDC
- Output Sensor Supply Maximum Current Ratings: 5v & 8V
- Dimensions :185mm (L) X 125mm (W) X 43.5mm(H)
- Weight: 590g

The detailed specification and features are included in Appendix

3.2. Sensors

All sensors are used to gather information related to their particular function. The sensor converts a physical or chemical input quantity into an output electrical quantity. The electrical sensor outputs are not only in the form of current and voltage but also available as current or voltage amplitudes, frequency, phases, pulse duration, etc. Sensors act as a gateway (input) to the control system. Sensors have a wide variety of applications in automobiles. Sensors integrated on various parts of the vehicle collect data and send it to the ECU for processing. The number and type of sensors required to depend upon the factors that affect the operation of the system. [6]

Automotive sensors are designed to work in an extreme environment where the effects of heat, vibration, thermal and mechanical shock. The sensors are also designed to fulfil the requirements of size, power consumption, and compatibility with the electronic circuit and other devices. [7]

Automotive sensors can be classified according to their unique characteristics and application on the vehicle. They can be classified as

I. Based on Application

- Functional sensors (used for open and closed-loop control functions)
- Sensors for safety (airbag, ESP, ABS)
- Sensors for vehicle monitoring (OBD, speed, fuel information)

II. Base on output curves

- Continuous liner curves (used mainly for control applications with a wide measuring range)

- Continuous non-linear curves (used for closed-loop assignments with limited measuring range)
- Step curves (are used for limit-value monitoring in cases where suitable functions have to be processed when the limits are reached)

III. Type of output signal

a. Analog Signal

- Current or voltage amplitudes
- Frequency durations
- Pulse durations

b. Discrete Signal

- Two-step (binary coded)
- Multi-step (irregular steps with analog coded)
- Multi-step (equidistant with analog or digital coded)
- Continuous and discontinuous signals

Different sensors are required along with ECU to process the engine and other corresponding functions of the FS car. These sensors include the sensors which are required by the ECU to operate accurate air/fuel ratio and ignition timing for EFI engines and the supporting sensors to give out valuable information such as temperature, pressure, etc. [8]

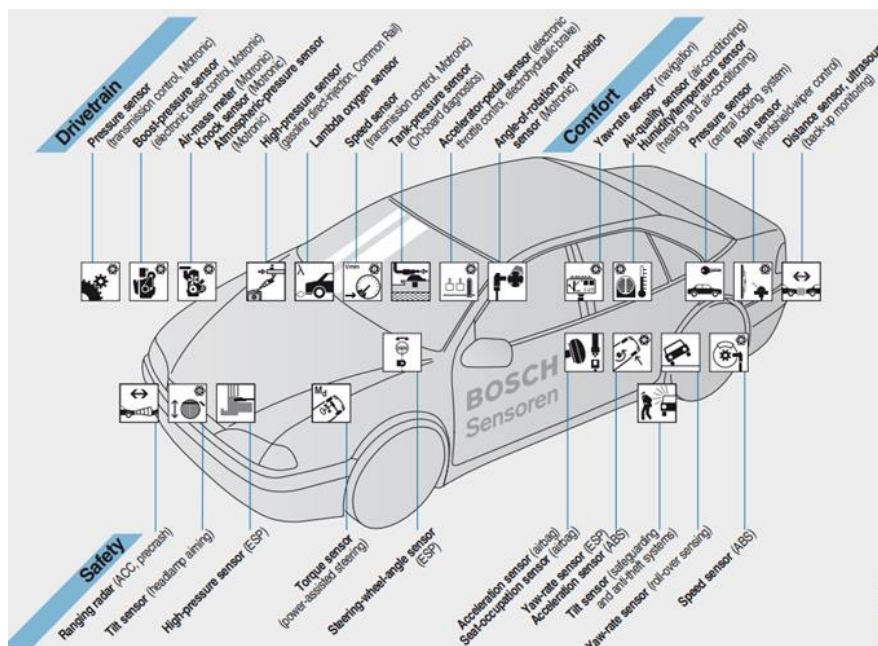


Figure 2: Automotive sensor overview

The different types of sensors used are:

1. Crank and Cam Sensor
2. Throttle Position Sensor
3. Oxygen sensor
4. Air Temperature Sensor
5. Coolant Temperature Sensor
6. Intake Manifold Pressure Sensor
7. Wheel Speed Sensor
8. Fuel Flow Sensor
9. Brake IR Temperature Sensor

The different types of actuators used are:

1. Fuel injectors
2. Ignition coil
3. Spark plug

3.2.1. Crankshaft & Camshaft sensor



Figure 3: Crank sensor

It is one of the important sensors in the entire automotive electronic system. The sensor is required for the ECU to sense the position of the pistons in the engine

in order to configure ignition timing and fuel injection accurately. The ECU needs either a crankshaft or camshaft sensor to find the position of the crankshaft or the camshaft.

The crankshaft is actually a shaft driven by the crank mechanism including cranks and crankpins to which the connecting rods of the engine are connected. Crankshafts convert the reciprocating motion to rotational motion. On other hand, camshafts are used to operate the intake and exhaust valves in IC engines. A camshaft generally is a rotating object which it converts rotating motion to reciprocating motion. Both the cam and crankshaft work in harmony for the proper running of IC engines.

Different types of sensors can be used for the purpose such as the inductive sensor, Hall effect sensor, magneto-resistive sensor, and the optical sensor. Hall effect sensors are more advantageous over other sensors such as the ability to detect static magnetic fields. They can also withstand high temperatures, more accuracy, product life expectancy, and cost.

Hall effect sensor is a passive sensor and uses the principle that a voltage is produced when a thin-film semiconductor carrying an electric current is placed at right angles to the magnetic field. The Hall effect sensor consists of a steel disk having protruding tabs and a magnet for coupling the disk to the sensing element. The steel disk varies the reluctance of the magnetic path. [9]

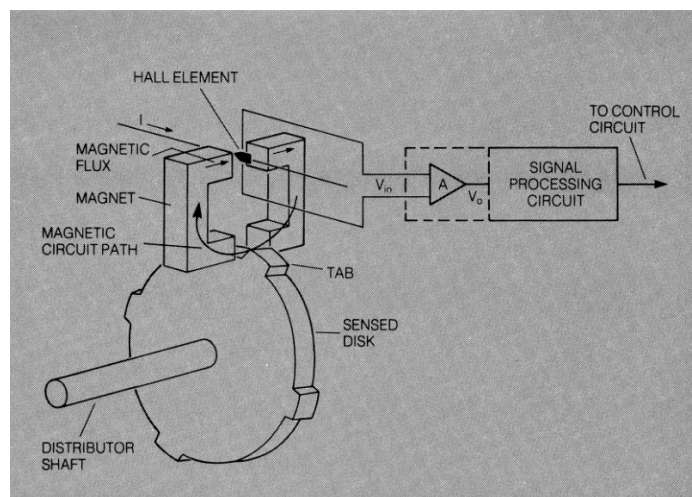


Figure 4: Hall effect sensor design

3.2.2. Throttle Position Sensor

Throttle position determines the air intake of the manifold. The throttle position is one of the key variables in ECU calculations for the amount of fuel needed to create the desired air/fuel ratio. During calculations, the percentage is indicated as the value of the throttle. With 0 percent completely closed and 100 percent totally open, the percentage value is equal to how far the throttle valve is open. To get the exact throttle position of the throttle valve, a throttle position sensor (TPS) is mounted to the throttle valve. Different type of sensors is available in the case of TPS. Potentiometer sensors, magnetically inductive-type sensors, eddy-current sensors can be framed as TPS. The most commonly used is the potentiometer TPS.



Figure 5: Throttle position sensor

The TPS consist of a wiper arm connected mechanically with the throttle-valve shaft and with its brushes slides across the respective potentiometer tracks thus converting the rotation of the throttle valve shaft into a voltage ratio that is proportional to the valve's angle of rotation. The fig shows the TPS design with the throttle-valve shaft, resistance track, wiper arm, and electrical connection.

[8]

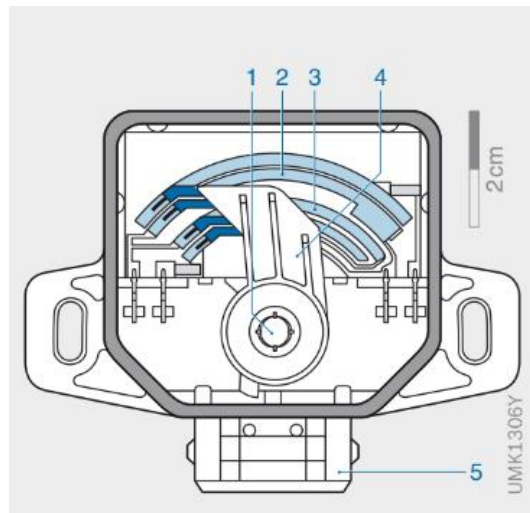


Figure 6: Throttle position sensor design

3.2.3. Oxygen Sensor

The air/fuel system is a closed-loop configuration thus the ECU requires feedback data from different sensors to provide the configuration. An oxygen sensor (OS) is used to measure the percentage of oxygen in the exhaust gas to calculate the air/fuel ratio. OS is also known as ' λ sensor' is mounted in the exhaust pipe. Excess oxygen in the exhaust indicates that the air/fuel ratio is less and the absence of oxygen shows that the mixture is rich. The OS is designed to indicate the effectiveness of the combustion process by measuring the oxygen content in the exhaust gases and provides feedback about the fuel mixture composition. The ECU modifies the data to adjust the amount of fuel injected by the fuel injectors in order to achieve the desired concentration for the fuel being burned. [7]

The conventional OS is the zirconium oxide and titanium oxide types along with wideband sensors in recent years. Wideband OS gives out improved fuel economy, lower emissions, and better engine performance. [9]

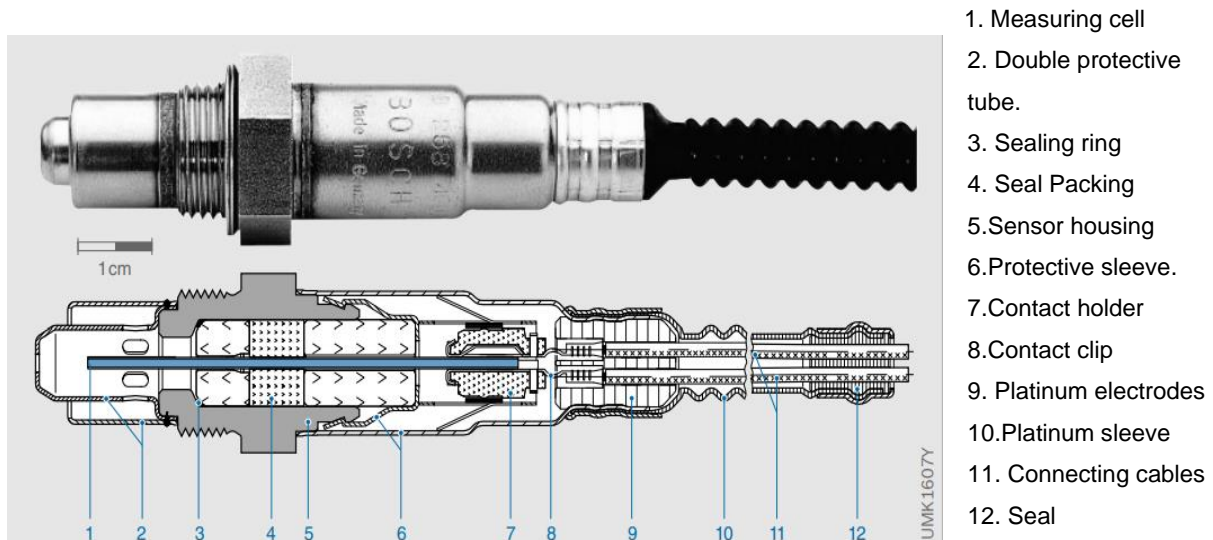


Figure 7: Oxygen sensor

3.2.4. Air Temperature Sensor

The air temperature sensor (ATS) measures the intake air temperature and the sensor is mounted in the air-intake tract. The ECU requires the air density to calculate the air entering the cylinder on each stroke thus able to generate a desired air/fuel ratio. Instead of directly measuring the density of air, incoming air temperature is measured with the help of ATS. Both thermistor and thermocouple sensors are used as temperature sensors. The thermistor can be used up to 200 °C thus these sensors are used for automotive purposes. [8]

A thermistor sensor normally consists of a brass bulb called a thermistor, which is in contact with the substance to be sensed. Thermistors that have a sensing capsule that responds in a way such that the resistance of metal increases with temperature is called positive temperature coefficient (PTC) and vice versa is called as negative temperature coefficient (NTC). [7]



Figure 8: Air temperature sensor

3.2.5. Coolant Temperature Sensor

As the name suggests the function of the coolant temperature sensor (CTS) is to constantly monitor the temperature of the engine coolant. In most cases, the CTS is located in between the radiator and the engine. This is to measure the temperature of the coolant from the engine so as to operate the radiator fan if the coolant temperature exceeds the levels. Coolant temperature can also be used as the hazard indicator and can be used to shut down the engine and protect the components. CTS also uses a thermistor sensor for sensing the coolant.

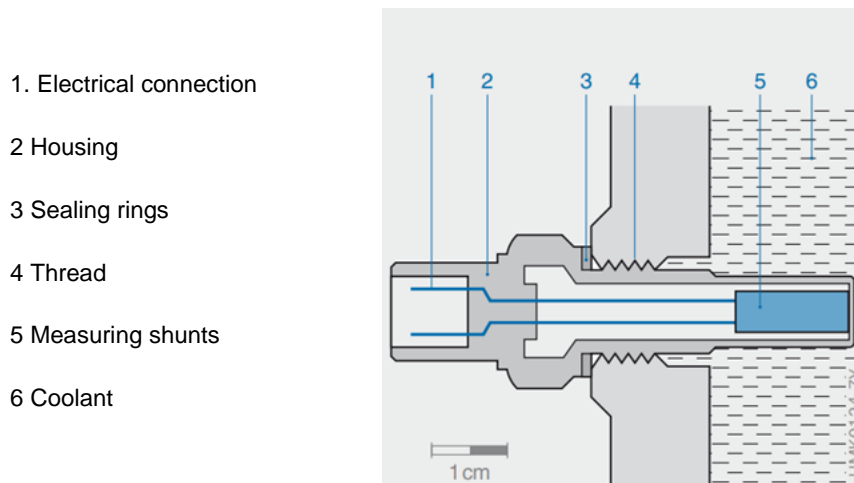


Figure 9: Coolant temperature sensor design

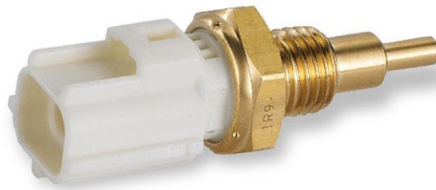


Figure 10: Coolant temperature sensor

3.2.6. Intake Manifold Pressure Sensor

A Manifold pressure sensor (MPS) is used to determine the mass air entering the cylinder on each stroke. Manifold absolute pressure refers to the pressure of the air which is in the intake manifold before the cylinder. The data collected by the ECU are used to calculate the air density and the rate of air mass flow of the engine, which then determines the fuel required to optimise the combustion and influences the advance or delay in the ignition cycle. The MPS is made up of Micro-Electro-Mechanical Systems (MEMS) based pressure transducers and it measures the vacuum created by the intake stroke of the piston.



Figure 11: Manifold pressure sensor

3.2.7. Wheel Speed Sensor

Wheel speed sensors (WSS) are used to measure the speed of the vehicle. The signals for the wheel speed sensor are generated by a steel pulse generator that is fixed to the wheel hub (for passive sensors) or by a multiple magnetic pulse generator (for the active sensor). In most cases, active WSS is used and usually consists of a hermetic, plastic-cast silicon IC that sits in the sensor head.

[10]

- 1. Sensor element
- 2. Multipole rings

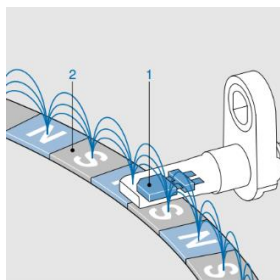


Figure 12: Hall effect working



Figure 14: Wheel speed sensor

3.2.8. Fuel Flow Sensor

A fuel Flow Sensor (FFS) is not mandatory but a useful sensor that can be integrated into the electronics system. FSS as the name suggests measures

the rate of fuel flow from the fuel tank to the engine. The sensor is designed in such a way that in between the fuel line a turbine is placed and when the fuel flows through the designated path the turbine rotates generating the signal to the ECU and gives out the fuel flow rate.

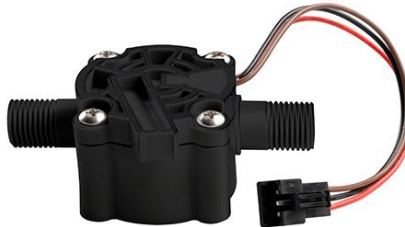


Figure 13: Fuel flow sensor

3.2.9 Brake Temperature Sensor

An infrared temperature sensor is designed to measure the transient surface temperature of brake discs at multiple points, enabling the acquisition of time-based temperature distribution across the surface of a rotor in order to evaluate and optimise pad pressure distribution, cooling efficiency, braking efficiency, and hot spot formation due to thermoelastic instabilities. The sensor is capable of measuring temperature at sampling frequency of up to 100Hz, object temperature between -20 °C to 950 °C. IP66 protection rating prevents the sensor from extreme heat and reverse polarity conditions.



Figure 14: IR Brake Temperature Sensor

3.3. Actuators

Actuators form the link between the electronic signal processor and the actual motion (mechanical motion). They convert low-power signals to operating signals at an energy level that is suitable for process control. There is a different type of actuators such as electromechanical, electromagnetic, electrodynamic, thermal, mechanical actuators. [11]

The actuators used for the proposed FS car are:

- Ignition Coil
- Spark Plug
- Fuel Injector

3.3.1. Ignition Coil

The ignition coil is an induction coil that is used to convert battery voltage to thousands of volts to create an electric spark in the spark plugs to ignite the fuel. An ignition coil is a laminate iron core with two copper wire coils. An ignition coil has an open magnet circuit – there is no closed loop between the windings in the iron core. The basic principle is the same as that of the transformer. The

energy stored in the magnetic field in the nucleus is the energy transmitted to the spark plug.



Figure 15: Ignition coil

3.3.2. Spark Plug

The spark plug is a device for transferring electric current from an ignition system to the combustion chamber. A spark plug has a threaded metal shell that is electrically isolated by a ceramic insulator from a central electrode. The central electrode that may be resistor is connected to the output terminal of an ignition spiral or magnet by means of a heavily insulated wire. The shell of the spark plug is torn on the cylinder head of the engine and is therefore electrolysed. The central electrode rises into the combustion chamber via the porcelain isolator, creating one or more spark lengths between the inner end of the central electrode and generally one or more protuberances or structures attached to the inner end of the thread-shell and the side, earth, or ground electrode.



Figure 16: Spark plug

3.3.3. Fuel Injectors

The fuel injector is an electronically controlled valve that sprays the fuel onto the air stream to generate a conventional air/fuel ratio inside the combustion chamber. The actuation of the fuel injector is controlled by the ECU. Injectors are capable of opening and closing many times a second. The injector is designed to have an atomising nozzle that distributes the fuel constantly and evenly for optimum combustion and efficiency. Fuel injectors replace traditional carburettor system which helps to get better fuel economy and reduce emissions.



Figure 17: Fuel injector

4. Data Logger

Data logging is the method by which the engine and sensor data are stored and saved for reference and to detect the faults that occurring to the components. Data logging allows viewing all key engine operational parameters. Some of those parameters are rpm, air-coolant, air/fuel, duty injection, throttle, motor speed, etc. In the motor sports industry, the data logger is primarily used for two purposes: firstly, to collect coordinates and speeds in real-time to obtain the lap time to enhance the driver's results. The second data logger collects more parameters in a car like lengths and

sides, temperatures in various parts, etc., so the vehicle's operations can be monitored and the vehicle improved by the engineer. There are data logging devices available in the market which can be adapted with the ECU through the CAN bus which stores the data. These logged data or stats are displayed on the screens mounted on the steering wheels to give information to the driver.

In this report, new novel technology is proposed for data logging and sharing telemetric data. The conventional data loggers available in the market are so expensive and the rules don't permit use in the FS car, thus data logging using cheap and modern electronic components are framed.

4.1. Components Selected

4.1.1. ESP32 Microcontroller

The ESP32-WROOM-32 is a versatile MCU module with a wide variety of uses. The ESP32 is a dual-core processor with built-in Wi-Fi and Bluetooth. It is capable of running 32-bit applications. It has 512 kB RAM and a clock frequency of up to 240 MHz. It supports a wide range of peripherals, including UART, ADCs, DACs, capacitive touch, SPI, I2C, and more. It comes with a built-in temperature sensor and a built-in hall effect sensor. The chip comes with 48 pins with multiple functions but not all pins are exposed in all ESP32 development boards, and there are unused pins.

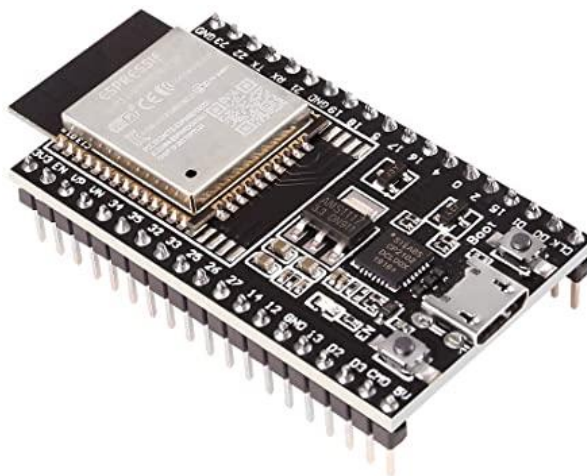


Figure 18: ESP32 MC

The module's combination of Wi-Fi, Bluetooth LE, and Bluetooth means that a wide variety of devices can be targeted and that it is future proof: using Wi-Fi allows for a large geographic range and direct access to the internet with a Wi-Fi router while using Bluetooth allows the user to link to the phone or emit low energy beacons for tracking. The ESP32 chip has a sleep current of less than 5 A, making it ideal for battery-powered and portable electronics. The ESP32 has a data rate of up to 150 Mbps and a 20.5 dBm antenna output capacity for the best physical range. As a result, the chip has industry-leading requirements and the highest performance in terms of electronic integration, range, power consumption, and other factors. As a result, the chip has industry-leading requirements and has the highest performance in terms of electronic integration, range, power usage, and networking. [12]

4.1.1.1. Pinout Description

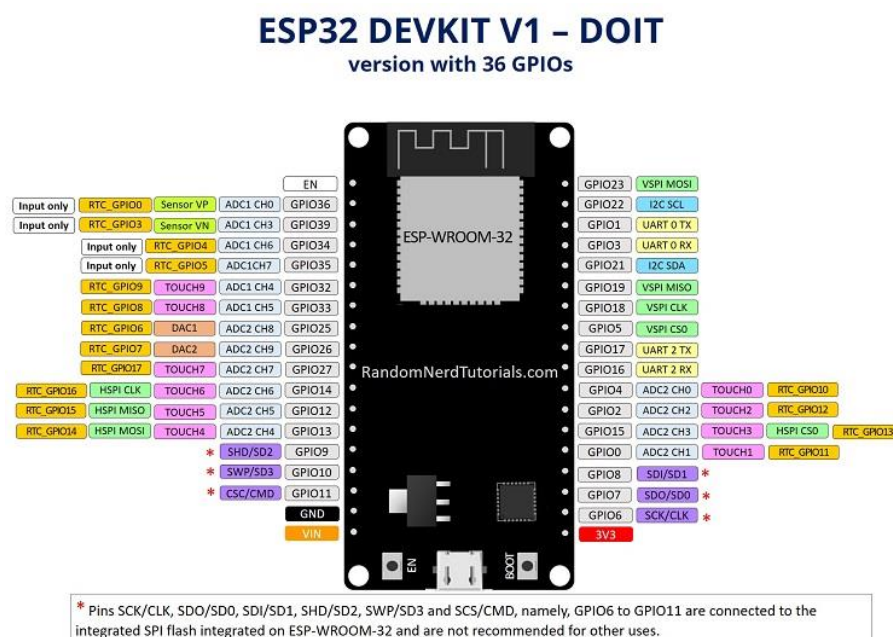


Figure 19: ESP32 pinout diagram

Boot Button: The ESP32 microcontroller is used to import new programs. We must hold down the Boot Button during the upload.

Micro USB: Used to power the board. This port is used to link the board to the server so that programs can be uploaded.

RESET Button: This button is simply used to reset the board.

Touch Sensor: The ESP32 comes with 10 capacitive touch sensors connected to GPIO pins. When these GPIO pins come into contact with anything that conducts electricity, such as human skin, they can detect variations. As a result, when we place our finger on the GPIO pin, it causes a difference that the sensor detects. To get the value from the touch sensor, we can use the Touch-Read (GPIO) feature.

Hall Effect Sensor: Another sensor, a Hall effect sensor, is integrated into the board. The magnetic field in the surrounding is measured using a Hall effect sensor. The ESP 32 board has 38 pins for connecting to external sensors and modules. There are 25 GPIO pins among the 38 pins, which can be used for a variety of functions.

Power Pins: A 5V pin and a 3.3V pin are used on the frame. If you have a controlled 5V voltage source, you can use the 5V pin to directly power the ESP32 and its peripherals. The 3.3V pin, which is the output of a voltage regulator (CP2102), can be used to control external components.

GND: The ground pin of ESP32 is used to complete the circuit.

ADC (Analog to digital) Channels: The board has 18, 12-bit SAR ADCs and supports measurements on 15 channels (analog enabled pins. ADC1_CH0 (GPIO 36)

DAC (Digital to Analog) Channels: Two 8-bit DAC channels are used on the board for converting optical signals to analogue voltage.

UART (*Universal Asynchronous Receiver-Transmitter*) Pins: The UART0, UART1, and UART2 interfaces on the ESP32 dev board provide asynchronous connectivity between UART-enabled devices at a speed of up to 5 Mbps. The dev board's UART interface also has two extra pins – CTS and RTS signals pins – that enable the receiver and sender to communicate their current state.

SPI Pins: The board features three SPIs (SPI, HSPI, and VSPI) in slave and master modes. These pins are used to connect to the external SPI-enabled devices.

PWM Pins: The board comes with 25 PWM enabled pins (Nearly All GPIO pins) The PWM output can be used for driving digital motors and LEDs.

EN or Enable Pin: EN stands for Enable, this pin is a 3.3 V regulator enable pin. When this is pulled LOW it resets the micro-controller.

4.1.2. I2C Oled Display

An organic light-emitting diode (OLED) is a light-emitting diode (LED) with an organic compound film as the emissive electroluminescent layer that emits light in response to an electric current. OLEDs are monolithic solid-state devices made up of a sequence of organic thin films sandwiched between two conductive thin-film electrodes. As electricity is applied to an OLED, charge carriers (holes and electrons) migrate from the electrodes into the organic thin films until they recombine in the emissive zone to form excitons under the influence of an electrical field. These excitons, or excited states, relax to a lower energy level by emitting light (electroluminescence) and/or unwanted heat once they've been created.

Organic carbon-based films are sandwiched between two charged electrodes in OLED displays. The first is a metallic cathode, and the second is a translucent anode, normally made of glass. The addressing schemes for OLED displays are either passive-matrix (PMOLED) or active-matrix (AMOLED). AMOLEDs (active-matrix OLEDs) need a thin film transistor backplane to turn each pixel on or off, but they provide higher resolution and larger display sizes. Since an OLED display does not need a backlight, it can display deep black levels while still being smaller and lighter than a liquid crystal display (LCD). An OLED screen can achieve a higher contrast ratio than an LCD in low ambient light conditions (such as a darkroom), regardless of whether the LCD uses cold cathode fluorescent lamps or an LED backlight. The OLED display does not need backlighting, resulting in excellent contrast in low-light situations. Furthermore, since its pixels only consume energy when they are turned on, the OLED display uses less power than other displays.

The model we're using has just four pins and uses the I2C communication protocol to communicate with the ESP32. Several versions have an additional RESET pin. Other OLED displays that communicate using SPI are also available. This display can support both IIC and SPI communication. When you receive the module from the factory, it will be in 4-wire SPI mode, which is the fastest of all the modes available. You can, however, re-solder the resistors in different positions to make it work with the

3-Wire SPI and IIC protocols as well. Hardware should not be an issue since the IC supports both 3.3V and 5V logic modules.[13]



Figure 20: OLED display

4.1.2.1. Pin Description

Pin No:	Pin Name	Description
1	Ground (Gnd)	Connected to the ground
2	Supply (vdd, vcc, 5V)	Powered by 5V
3	SCK (D0, SCL, CLK)	Clock pin
4	SDA (D1, MOSI)	Data pin
5	RES (RST, REST)	To reset the module
6	DC (A0)	Command pin
7	Chip Select (CS)	Activates when more devices are connected to MCU

4.1.3. LORA SX1278 Module

The LoRa® long-range modem in the SX1278 transceivers offers ultra-long range spread spectrum communication and high interference immunity while consuming minimal current. This is a low-cost RF front-end transceiver module based on the Semtech Corporation's SX1278. It retains the benefits of the RFIC SX1278 while simplifying the circuit design. The module is suitable for low range and low data rate applications due to its high sensitivity (-136dBm) in LoRa modulation and 20dBm high-power performance. The RFIC SX1278, a thin SMD crystal, and an antenna matching circuit make up the module. The antenna port has a 50 Ohm impedance that is well balanced. The module runs at 6V and has a very low standby current, making it ideal for battery-powered applications. Since it is a pure hardware module that uses a 10ppm crystal, the resolution of which is critical in measuring spreading factor, bandwidth, and other parameters.

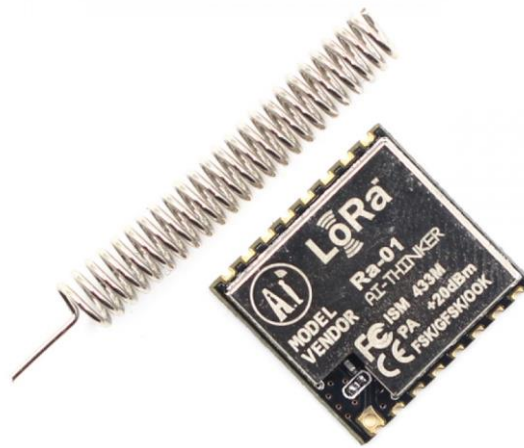


Figure 21: LORA module

Key Features;

- LoRaTM Modem 168dB maximum link budget
- +20dBm - 100mW constant RF output vs. V supply
- +14dBm high efficiency PA

- Programmable bit rate up to 300kbps
- High sensitivity: down to -148dBm
- Bullet-proof front end: IIP3 = -11dBm
- Excellent blocking immunity
- Low RX current of 9.9mA, 200nA register retention
- Fully integrated synthesizer with a resolution of 61Hz
- FSK, GFSK, MSK, GMSK, LoRa and OOK modulation
- Built-in bit synchronizer for clock recovery
- Preamble detection
- 127dB Dynamic Range RSSI
- Automatic RF Sense and CAD with ultra-fast AFC
- Packet engine up to 256 bytes with CRC
- Built-in temperature sensor and low battery indicator
- Using a single battery, the device operates for over a year

Since the LoRa SX1278 communicates using the SPI protocol, it can be used with any microcontroller that supports SPI. It is needed to use an Ariel (antenna) in conjunction with the module; otherwise, the module will be permanently damaged. Just 3.3V can be used to power the module, and the SPI line should be linked to uP/uC.

4.1.3.1. Pinout Description

Semtech created the LoRa wireless protocol, which is a reliable, low-power, long-range wireless technology. It is a proprietary and patented chirp spread-spectrum (CSS) modulation, which is identical to frequency modulation (FM). This CSS modulation is highly noise and Doppler effect-resistant, making it ideal for long-range low-power communication.[14]

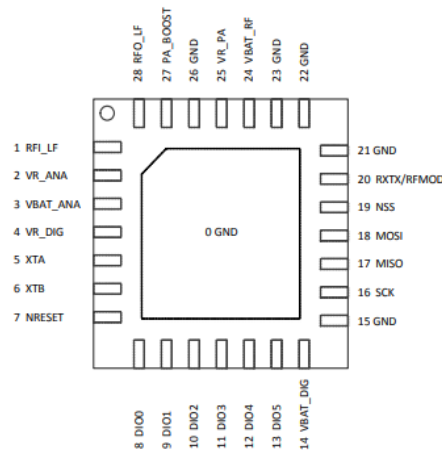


Figure 22: LORA pinout diagram

4.2 Software Used

The Arduino Integrated Development Environment (IDE) is the application used to frame the codes for the ESP32. The functions can be written in either C or C++ in Arduino IDE. The Arduino IDE provides a software library of the Wiring project that includes a number of common processes for input and output which makes the user to develop programs based on his/her needs. Arduino has become an increasingly popular software platform for other vendors that are able to create and upload drawings from custom open-source compilers and tools (core) that aren't supported by Arduino's official line of a microcontroller.

4.3 Working

Using an ESP32 board and the LoRa module sx1278, the sensor output values are continuously monitored. All sensor readings will be sent via LoRa radio to an ESP32 LoRa receiver, along with a webpage. The data will be displayed on a web server and an OLED display inside the vehicle.

The sender will read the various sensor values using respective sensors and it transmits those data using the LoRa Radio. The transmitter component has an OLED

display that shows all of the values. The webpage is saved to the SPIFFS (ESP32 File System), and the LoRa receiver is used as an asynchronous web server. SPIFFS allows the access of flash memory in the same way as a standard filesystem but in a much simpler and restricted way. The most recent reading's time and date are also shown. The Network Time Protocol is used to keep track of time and date. LoRa modules are available in a variety of frequency ranges. This LoRa technology can be used to send bi-directional data over long distances, most likely 15 to 20 kilometers, without consuming a lot of power.

The basic idea is that chirp is used to encrypt information (a gradual increase or decrease in the frequency of the carrier wave over time). The LoRa transmitter will send out a chirp signal before transmitting a message to ensure that the band is free to send the message. The end of the preamble is signaled by the reverse chirp, which informs the LoRa transmitter that it is clear to begin transmission after the LoRa receiver has picked up the preamble chirp from the transmitter.

4.3.1. Installing the Required Libraries

To read the air temperature we need any Air temp thermistor sensor. For Coolant temperature NTC thermistor. Hall sensor is used for wheel speed and for Fuel flow hall effect sensor is used. We need an Arduino library for sending and receiving data using LoRa radios along with two Library for OLED Display. Adafruit_SSD1306 is a library for our Monochrome OLEDs based on SSD1306 drivers. Adafruit GFX Library is the core graphics library for all our displays, providing a common set of graphics primitives (points, lines, circles, etc.). The Network Time Protocol (NTP) is a networking protocol that allows computer systems to synchronize their clocks over packet-switched, variable-latency data networks. When a LoRa receiver receives a new LoRa letter, it will request the date and time from an NTP server in order to obtain the time when the last packet was sent. Every time the LoRa receiver picks up a new a LoRa message, it will request the date and time from an NTP server so that we know when the last packet was received. To build the asynchronous web server we need two libraries, i.e., ESPAsyncWebServer library & Async TCP Library.

4.3.2. Transmitter:

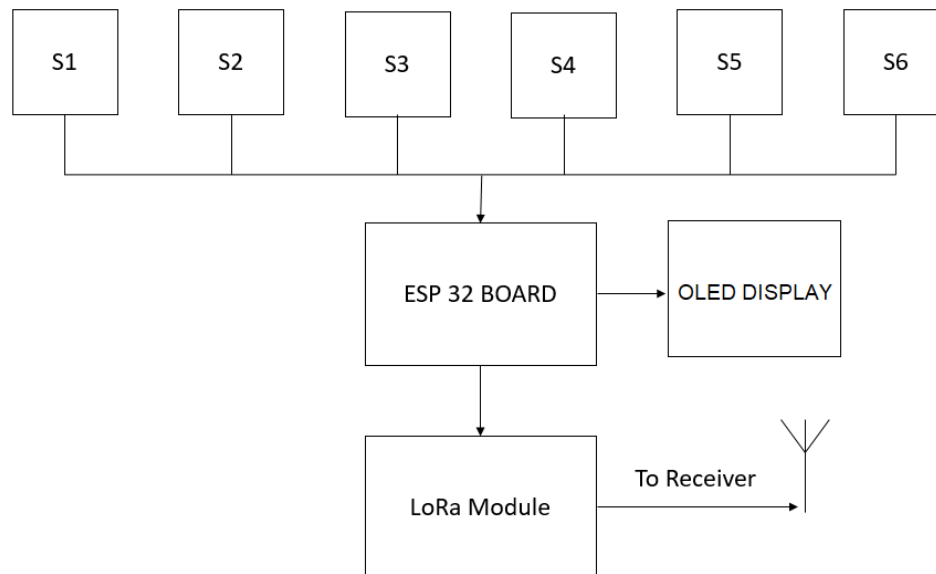


Figure 23: Block diagram of transmitter

The sensors are connected to an ESP32 LoRa Module SX1278 Sender Circuit. This component serves as a transmitter with an OLED display. Air temp thermistor sensor, NTC thermistor, hall effect sensor reads Air temperature, Coolant Temperature, Wheel speed and Fuel flow respectively and transmits via LoRa Radio. The connection of Lora of **SX1278** & **ESP32** as follows.

ESP32 Pins	SX1278
GND	GND
3.3V	VCC
DS	NSS
D23	MOSI
D19	MISO

D18	SCK
D14	RST
D2	DIDo

Similarly connect the sensors digital input pin to GPIO of ESP32. Connect its VCC to 3.3V & GND to GND of ESP32. Connect the I2C Pins of OLED Display, i.e., SDA & SCL pins to GPIO21 & GPIO22 of ESP32 respectively.

4.3.3. Receiver:

The below figure is the block diagram for an ESP32 LoRa Module SX1278 Receiver. This component serves as a receiver. LoRa Radio is used to receive the sensor data.



Figure 24: Block diagram of Receiver

The **ESP32 & LoRa SX1278** connection same as above. Supply it with 3.3V VCC & connect the GND to GND. The LoRa Receiver receives incoming LoRa packets and shows the received readings on a web server that is asynchronous. It also shows the RSSI and the last time the sensor readings were sent, in addition to the sensor readings (received signal strength indicator).

5. Results and Discussion

5.1. Individual Analysis

The selection of a suitable ECU and mapping procedures were the motive of the project. The overall outcome of the project was productive in consideration with the project methodology. All the tasks and development were completed within the time frame. The selected ECU is Haltech Elite 1000 and will give the maximum performance from the engine and can be calibrated using the software to integrate number of sensors. The designated ECU have many advantages over the default ECU clubbed with the engine. These advantages make reason for selection criteria. Haltech Elite series of ECUs are best in market probably suits the race purposes.

The other objective was to design a datalogger and telemetry system. The proposed data logging and telemetry system have both pros and cons. While considering the cost with the data loggers available in the market the suggested system is more advantageous. The entire system costs less than one-third of the data loggers available in the market. With the proposed system, live telemetry data can be viewed by the team while the car is in motion rather than a conventional plugin to the ECU or data loggers to acquire data. This helps the team to know the current status of the car and can keep the records of the data being generated and received. The driver can also view the status of the car through the display screen mounted on the steering wheel, thus helping the driver to stop the car if the temperature levels rise or during other malfunctions. The team can also insist the driver to pull over by analyzing the data they receive from the car.

On other hand, some possible errors and difficulties may occur with the system. As the communication between the transmitter and receiver is accomplished by the RF signals, possible errors due to the interference of other signals can lead to loss of data. Thus, causing difficulty to transmit data over long-range in urban areas with high interference. Another drawback is the damage that may occur to the transmitter components mounted within the car due to the cause of vibration and heat generation from the engine and vehicle motion.

Many difficulties have to be faced during the development of the project. The selection of the ECU was the most difficult process due to the lack of information about ECU available in the market. Constant communication and enquiries with different manufactures made it possible to opt the ECU which suits the proposed engine. The wiring diagram and technical information were provided by the manufacturer. The other major struggle was during the development and testing of the datalogger and telemetry system. Due to the current pandemic situation the purchase of hardware components was limited and was also not able to completely develop the system. The wiring diagram of the ECU with selected sensors and other mandatory sensors is shown in Appendix A5 and the codes to execute the telemetry system is shown in Appendix A6.

5.2. Group Analysis

The result of the entire group is considered to be successful. Each member was able to reach their designs and goals that were proposed during the initial stages. The motive of the driver control group was to maximize the driver-car interface to make the driver comfortable inside the cockpit and also to give out maximum performance from the car. The development and implementation of the Dual Axis Steering (DAS) system was a great innovation from the group which improves the vehicle's handling in corners and in a straight line. The design of steering wheels was other great innovative. The design helped to integrate the paddle shifters and the clutch paddle into the steering wheel without eating up too much space.

The other aim of the group was to reduce the overall weight of the pedal box. This was successfully achieved by omitting the clutch pedal and the materials used were lighter compared to their predecessors. The designed pedal box also has benefits such as the system can be adjusted on the driver height without making changes to the seat position. The motive of the brake system was to optimize the brake calipers, disc, and pads to give high performance, cost-effectiveness, and setting up for maximum efficiency. Each of the individual sections was studied thoroughly to improve the entire

brake system. The selection of materials, brake fluid, and the need for a temperature monitoring system were proved by the member with solid simulation and demos on software.

Considering the importance of the individual project, the other subsystems within the driver-control group and other groups is depended on the ECU and mapping. In view of the corresponding subsystems, the selection of ECU has a vital role. More number of sensors and actuators suggested by the team members were mapped to the ECU. This includes Brake Temperature sensors, pneumatic and brake actuators, etc. The wiring harness was closely related to the ECU and mapping and together all the mandatory circuits within the car along with the color codes, fuses, actuators, etc. were mentioned in their respective reports.

6. Conclusion

The main objective was to select an appropriate ECU for the new engine for FS car. There was a bold move from the powertrain group to suggest and change the engine for this year forced to make changes in the ECU. The project is considered to be successful as a suitable ECU was selected. The schematic wiring diagram of the ECU is provided in the report which helps the team to understand the wiring color codes, input and output pins of the ECU. These input and output pins are used to integrate the sensors and actuators respectively. As the clear wiring diagram is designed and proposed it will be beneficial during the installing phase to avoid confusion and a lot of time can be saved during the process. The default wiring diagram, specification, overview related to the ECU is also provided by the manufacture which benefits the current team and upcoming teams to understand the ECU.

In addition to the prior objective, a data-logger and telemetry system was developed using ESP32 and Lora technology. This was meant to be partially successful as the proposed system and the code written for the components to execute the transfer of telemetry data and data logging was not able to test. Scope for further development is left open for future teams to develop the proposed system.

7. Scope for Further Research

Further studies and research can be carried with the selection of ECU and sensors to adapt to the opted engine. Different range of ECUs and sensors are available in the market making it difficult to choose the appropriate one which gives out the maximum performance and efficiency from the car. Possible errors can also be resolved by running the selected ECU and sensors with the engine. The high demand for sensors and actuators needed for the other subsystems will make tuning more complex, thus accurate calibration software has to be used for the tuning purposes.

Further research can be done on the proposed data logger and telemetry system to reduce the faults and improve the system. Currently, the data from sensors directly feeds to the input of ESP32 board rather than from ECU. If possible further research and studies can be done on the topic to feed the processed data from ECU to the transmitter via CAN communication. The number of sensors was limited to reduce the complexity in the hardware and program of the modules. In the future, the scope of integrating more sensors can be carried out. The transmitter section can be also made free from the interference of other RF communications and disturbances from the engine to eliminate the possible occurring errors.

8. References

1. Imeche.org. [online] Available at < [Formula Student - Institution of Mechanical Engineers \(imeche.org\)](http://www.imeche.org) >
2. Brunel Racing [online] Available at < [» Brunel Masters Motorsport Brunel Racing](http://www.brunelmotorsport.co.uk)>
3. McLaren.com. (2020). Case study: Electronic Control Units. [Online] Available at < [Case study: Electronic Control Units \(mclaren.com\)](http://www.mclaren.com/case-study/electronic-control-units)>
4. Automated tuning of an engine management unit for an automotive engine, University of Canterbury, Christchurch, New Zealand; C Hardie, H Tait, S Craig, J G Chase, B W Smith and G Harris
5. Haltech.com [online] Available at <[Product Overview: Elite 1000 - Haltech](http://www.haltech.com/products/elite-1000)>
6. Understanding Automotive Electronics: An Engineering Perspective, Elsevier Science & Technology, 2012; Ribbens, William
7. Hillier's fundamentals of automotive electronics, 2012; Hillier, V. A. W.
8. Bosch Automotive Electrics and Automotive Electronics, 2007; Robert Bosch GmbH
9. Automotive Mechatronics Automotive Networking, Driving Stability Systems, Electronics, 2015; Dr.-Ing. Konrad Reif.
10. Brake design and safety, 2011; Limpert, Rudolf.
11. Vehicle and engine technology, 1999; Heisler, Heinz
12. ESP32 Datasheet, Espressif systems. [online] Available at <[esp32_datasheet_en.pdf \(espressif.com\)](http://www.espressif.com/en_US/hardware/modules/esp32)>
13. Organic Light Emitting Diode (OLED); Hochschule Bremen City University of Applied Sciences. [online] Available at < [\(PDF\) Organic Light Emitting Diodes \(OLED\) \(researchgate.net\)](http://www.researchgate.net/publication/312111111)>
14. LoRa® and LoRaWAN®: A Technical Overview, Semtech; [online] Available at <[LoRa and LoRaWAN-A Tech Overview-Downloadable.pdf \(semtech.com\)](http://www.semtech.com/LoRaWAN-A_Tech_Overview-Downloadable.pdf)>

9. Appendix

A1. Project Management

The outcome of the individual project was successful under my review. The proposed objectives were obtained but unable to test the clarify errors of the system due to the lack of availability of individual components. On considering the group work all the team members were able to succeed on their respective subsystems. CFD analysis, CAD models as well as Arduino codes were presented to complete the proposed ideas. Project plan and execution was clearly explained within the individual reports of each members. The subsystems within the group were selected by the willingness of members rather than allocations from supervisor, thus benefiting to get maximum outcome of individual sections. Only one of the subsystems was forced to share between two members.

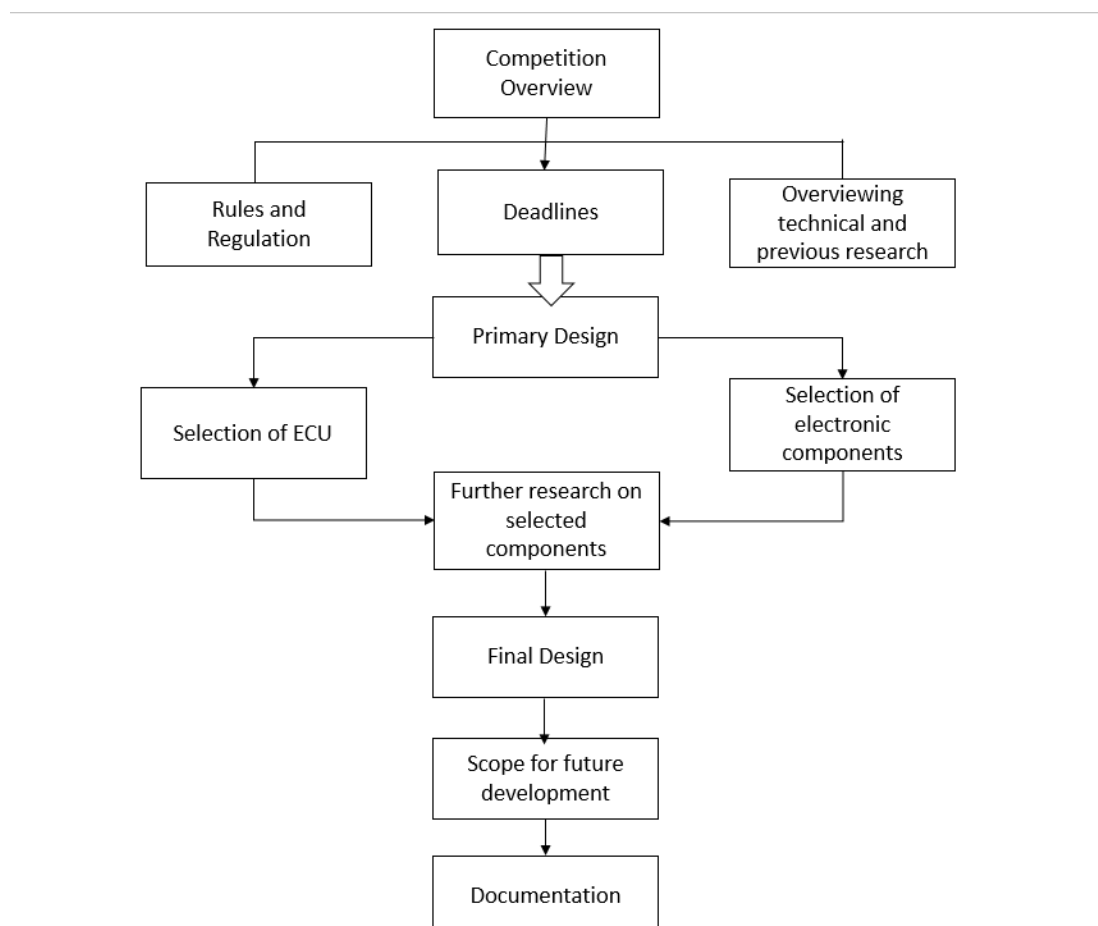
The data logger and telemetry system can be done in latest technologies rather than the proposed systems which may benefit the overall outcome of the system. Possible errors occurring with the proposed system may overcome with trying out other methods. A thorough study and strong information collection was done before the project proposal. All the latest innovations which improve the performance and more advantageous to the FS car than the previous year which admits the rules and regulations were proposed which prevented the team from missing key points and developments during the completion of project. The given role of each member stick through the project phase as no obligations and difficulties were faced by the team members that was big enough to change their roles on half way of the project. No further addition innovations were made other than the mentioned ones in the project proposal.

There were difficulties faced in organizing the team meetings because of the pandemic situation. Meetings were restricted to online and even some struggles occurred as the team members were in different time zone. The frequency of meetings reduced during the course of project making it difficult to communicate and execute the individual projects as each one is related to each other. Lack of participation of all team members

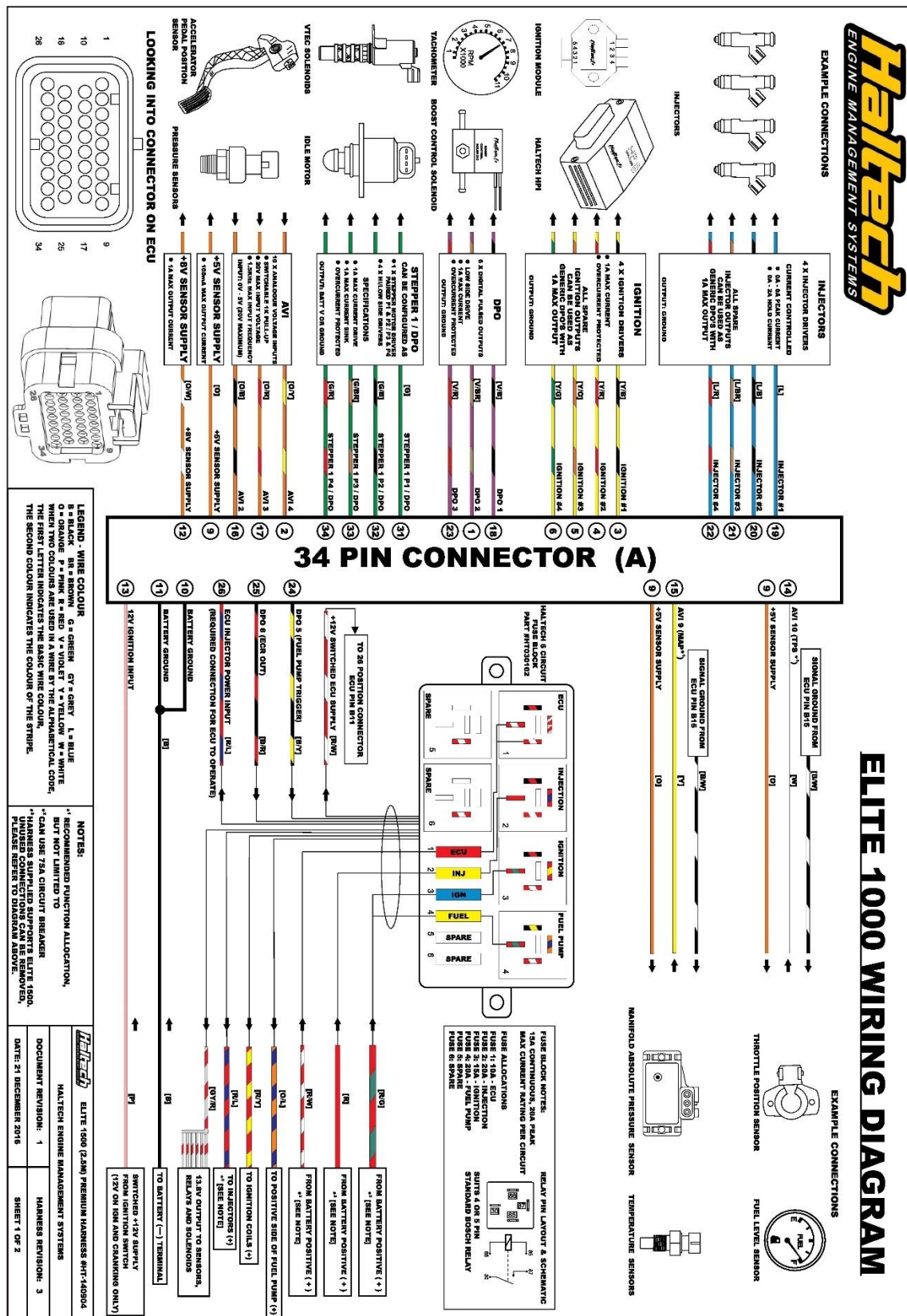
with the project supervisor is also considered to be a drawback. Still positive outcomes can be seen with the help of group chats and individual chats through online platforms.

With the successful completion of the project different aspects of the project phase and how to execute them within time frame were learnt. Constant communication with the team members also helped me to improve communication skills and team work. Constant chats and quires about the topic to team members helped me to gain more valuable information findings which improved the proposed project. Difficulties were also faced during the gathering on previous year proposals. The required data was fewer from previous year resulting in spending more time to gather data. I would like to suggest some improvements on the topic to the person who may or may not carry forward with the project were I left. Testing has to be done on the proposed system as it was not executed due to lack of availability of components and lab facilities.

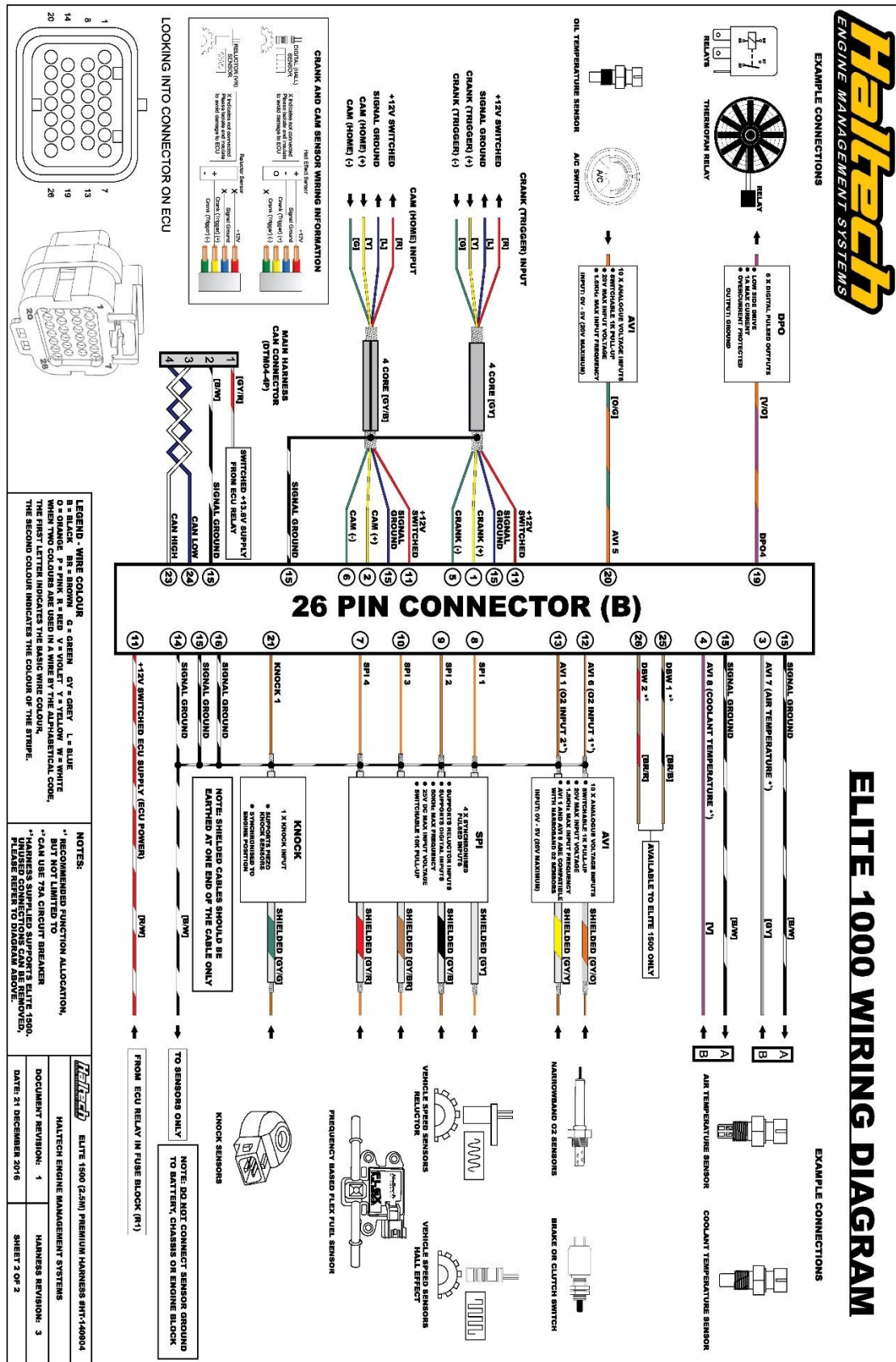
A2. Project Methodology



A3. Haltech Elite 1000 General WD (34 pin)



A4. Haltech Elite 1000 General WD (24 Pin)













A5. Haltech Elite 1000 Proposed WD



















Haltech ESP I/O Report

ECU

Analogue Voltage Inputs

<input type="checkbox"/> B13		GY/Y	AVI1 → Fuel Flow Sensor
<input type="checkbox"/> A16		O/B	AVI2 → Gear Voltage Sensor
<input type="checkbox"/> A17		O/R	AVI3
<input type="checkbox"/> A2		O/Y	AVI4
<input type="checkbox"/> B20		O/G	AVI5 → Selector Position Voltage Sensor
<input type="checkbox"/> B12		GY/O	AVI6
<input type="checkbox"/> B3		GY	AVI7 → Intake Air Temperature Sensor
<input type="checkbox"/> B4		V	AVI8 → Coolant Temperature Sensor
<input type="checkbox"/> A15		Y	AVI9
<input type="checkbox"/> A14		W	AVI10 → Throttle Position Sensor

Digital Pulsed Outputs

<input type="checkbox"/> A18		V/B	DPO1 → Cam Control - Intake Solenoid 1
<input type="checkbox"/> A1		V/BR	DPO2
<input type="checkbox"/> A23		V/R	DPO3
<input type="checkbox"/> B19		V/O	DPO4
<input type="checkbox"/> A24		B/Y	DPO5
<input type="checkbox"/> A25		B/R	DPO6
<input type="checkbox"/> A3		Y/B	IGN1 → Ignition Output 1
<input type="checkbox"/> A4		Y/R	IGN2 → Ignition Output 2
<input type="checkbox"/> A5		Y/O	IGN3 → Ignition Output 3
<input type="checkbox"/> A6		Y/G	IGN4
<input type="checkbox"/> A19		L	INJ1 → Stage 1 - Injection Output 1
<input type="checkbox"/> A20		L/B	INJ2 → Stage 1 - Injection Output 2
<input type="checkbox"/> A21		L/BR	INJ3 → Stage 1 - Injection Output 3
<input type="checkbox"/> A22		L/R	INJ4
<input type="checkbox"/> A31		G	STEP1 P1
<input type="checkbox"/> A32		G/B	STEP1 P2
<input type="checkbox"/> A33		G/BR	STEP1 P3
<input type="checkbox"/> A34		G/R	STEP1 P4

Home Input

<input type="checkbox"/> B2		Y	Home → Cam Control - Intake Angle 1
-----------------------------	-------------------------------------------------------------------------------------	---	--------------------------------------------

Knock Inputs

<input type="checkbox"/> B21		GY/G	Knock Input 1
------------------------------	-------------------------------------------------------------------------------------	------	----------------------

Pressure Inputs

<input type="checkbox"/>			OBPS1 → Manifold Pressure Sensor
--------------------------	-------------------------------------------------------------------------------------	--	-----------------------------------------

Sync Pulsed Inputs

<input type="checkbox"/> B8		GY	SPI1 → Vehicle Speed Drive Train Sensor
-----------------------------	-------------------------------------------------------------------------------------	----	------------------------------------------------

Haltech ESP I/O Report

ECU

Sync Pulsed Inputs

B9		GY/B	SPI2
B10		GY/BR	SPI3
B7		GY/R	SPI4

Stepper Outputs

A31		G] STEP1
A32		G/B	
A33		G/BR	
A34		G/R	

System References

<input type="checkbox"/>	A9		O	+5V DC → 5V Sensors
<input type="checkbox"/>	A12		O/W	+8V Sensor Supply → 8V Sensors
<input type="checkbox"/>	B11		R/W	+12V ECU Power → Fused Power from ECR
<input type="checkbox"/>	A13		P	+12V Ignition Input → Ignition Switch
<input type="checkbox"/>	A26		R/L	+12V Injector Power → 20A Fused Power
<input type="checkbox"/>	A10		B] Battery Ground → Battery (-ve)
<input type="checkbox"/>	A11		B	
<input type="checkbox"/>	AUX2		B	CAN Ground → CAN Signal Ground
<input type="checkbox"/>	AUX3		W	CAN High - Aux connector → CAN High (Haltech CAN System)
<input type="checkbox"/>	B23		W	CAN High - Main Connector → CAN High (Vehicle Bus)
<input type="checkbox"/>	AUX4		L	CAN Low - Aux connector → CAN Low (Haltech CAN System)
<input type="checkbox"/>	B24		L	CAN Low - Main Connector → CAN Low (Vehicle Bus)
<input type="checkbox"/>	AUX1		R	CAN Power → 3 Amp limit
<input type="checkbox"/>	A25		B/R	ECR Out → ECR Output
<input type="checkbox"/>	B2		Y	Home (+ve) → Home System (+)
<input type="checkbox"/>	B6		G	Home (-ve)
<input type="checkbox"/>	B14		B/W] Signal Ground → 0V Sensors
<input type="checkbox"/>	B15		B/W	
<input type="checkbox"/>	B16		B/W	
<input type="checkbox"/>	B1		Y	Trigger (+ve) → Trigger System (+)
<input type="checkbox"/>	B5		G	Trigger (-ve)

A6. Data Logger and Telemetry System Script

A6.1 Transmitter Code

```
#include <SPI.h>
#include <LoRa.h>

//Libraries for LoRa
#include "DHT.h"
#include "DHT.h"

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define DHTPIN 4      //pin where the dht11 is connected
DHT dht(DHTPIN, DHT11);
#define SENSOR 27 //fuelflow sensor
//define the pins used by the LoRa transceiver module
#define ss 5
#define rst 14
#define dio0 2

#define BAND 866E6 for Europe

//OLED pins
#define OLED_SDA 21
#define OLED_SCL 22
#define OLED_RST -1
#define SCREEN_WIDTH 128 // OLED display width, in pixels
```

```
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

//packet counter
int readingID = 0;

int counter = 0;
String LoRaMessage = "";

float temperature = 0;
float wheel_speed = 0;
float fuel_flow = 0;
float coolant_temperature = 0;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RST);

//Initialize OLED display
void startOLED(){
  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C
for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
```

```
display.clearDisplay();
display.setTextColor(WHITE);
display.setTextSize(1);
display.setCursor(0,0);
display.print("LORA SENDER");
}

//Initialize LoRa module
void startLoRA()
{
  LoRa.setPins(ss, rst, dio0); //setup LoRa transceiver module

  while (!LoRa.begin(BAND) && counter < 10) {
    Serial.print(".");
    counter++;
    delay(500);
  }
  if (counter == 10)
  {
    // Increment readingID on every new reading
    readingID++;
    Serial.println("Starting LoRa failed!");
  }
  Serial.println("LoRa Initialization OK!");
  delay(2000);
}

void startDHT()
{
```

```
    if (isnan(wheel_speed) || isnan(temperature) || isnan(fuel_flow) ||  
        isnan(coolane_temperature))  
    {  
        Serial.println("Failed to read from the sensors!");  
        return;  
    }  
}
```

```
void getReadings(){  
    temperature = dht.readTemperature();  
    Serial.print(F("% Temperature: "));  
    Serial.print(temperature);  
    Serial.println(F("°C "));  
}
```

```
// read hall effect sensor value  
Wheel_speed= hallRead(); // print the results to the serial monitor  
Serial.println(Wheel_speed);  
    delay(1000);  
}
```

```
void sendReadings() {  
    LoRaMessage = String(readingID) + "/" + String(temperature) + "&" +  
    String(Wheel_speed) + "&" + String(fuel_flow) + "&" + String(coolant_temperature);  
    //Send LoRa packet to receiver  
    LoRa.beginPacket();  
    LoRa.print(LoRaMessage);  
    LoRa.endPacket();  
  
    Serial.print("Sending packet: ");  
    Serial.println(readingID);  
    readingID++;  
    Serial.println(LoRaMessage);  
}
```

```
}

void setup() {
  //initialize Serial Monitor
  Serial.begin(115200);
  dht.begin();
  startDHT();
  startOLED();
  startLoRA();
}

void loop() {
  getReadings();
  sendReadings();
  delay(5000);
}
```

A6.2 Receiver Code

```
#include <WiFi.h>
#include "ESPAsyncWebServer.h"

#include <SPIFFS.h>

//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

// Libraries to get time from NTP Server
#include <NTPClient.h>
#include <WiFiUdp.h>
```



```
//define the pins used by the LoRa transceiver module
#define ss 5
#define rst 14
#define dio0 2

#define BAND 433E6 //433E6 for Asia, 866E6 for Europe, 915E6 for North America

// Replace with your network credentials
const char* ssid = "myhome";
const char* password = "9496234603";

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP);

// Variables to save date and time
String formattedDate;
String day;
String hour;
String timestamp;

// Initialize variables to get and save LoRa data
int rssi;
String loRaMessage;
String temperature;
String wheel_speed;
String fuel_flow;
```

```
String coolant_temperature;
String readingID;

// Create AsyncWebServer object on port 80
AsyncWebServer server(80);

// Replaces placeholder with DHT values
String processor(const String& var){
  //Serial.println(var);
  if(var == "TEMPERATURE"){
    return temperature;
  }
  else if(var == "WHEEL_SPEED"){
    return wheel_speed;
  }
  else if(var == "FUEL_FLOW"){
    return fuel_flow;
  }
  else if(var == "COOLANT_TEMPERATURE"){
    return coolant_temperature;
  }

  else if(var == "TIMESTAMP"){
    return timestamp;
  }
  else if (var == "RRSI"){
    return String(rssi);
  }
  return String();
}

//Initialize LoRa module
```

```
void startLoRA(){
    int counter;

    //setup LoRa transceiver module
    LoRa.setPins(ss, rst, dio0); //setup LoRa transceiver module

    while (!LoRa.begin(BAND) && counter < 10) {
        Serial.print(".");
        counter++;
        delay(500);
    }
    if (counter == 10) {
        // Increment readingID on every new reading
        Serial.println("Starting LoRa failed!");
    }
    Serial.println("LoRa Initialization OK!");
    display.setCursor(0,10);
    display.clearDisplay();
    display.print("LoRa Initializing OK!");
    display.display();
    delay(2000);
}

void connectWiFi(){
    // Connect to Wi-Fi network with SSID and password
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
}
```

```
}  
  
// Print local IP address and start web server  
Serial.println("");  
Serial.println("WiFi connected.");  
Serial.println("IP address: ");  
Serial.println(WiFi.localIP());  
display.setCursor(0,20);  
display.print("Access web server at: ");  
display.setCursor(0,30);  
display.print(WiFi.localIP());  
display.display();  
}  
  
// Read LoRa packet and get the sensor readings  
void getLoRaData() {  
    Serial.print("Lora packet received: ");  
    // Read packet  
    while (LoRa.available()) {  
        String LoRaData = LoRa.readString();  
        // LoRaData format: readingID/temperature&soilMoisture#batterylevel  
        // String example: 1/27.43&654#95.34  
        Serial.print(LoRaData);  
  
        // Get readingID, temperature and moisture  
        int pos1 = LoRaData.indexOf('/');  
        int pos2 = LoRaData.indexOf('&');  
        readingID = LoRaData.substring(0, pos1);  
        temperature = LoRaData.substring(pos1 +1, pos2);  
        humidity = LoRaData.substring(pos2+1, LoRaData.length());  
    }  
    // Get RSSI
```

```
    rssi = LoRa.packetRssi();
    Serial.print(" with RSSI ");
    Serial.println(rssi);
}

// Function to get date and time from NTPClient
void getTimeStamp() {
    while(!timeClient.update()) {
        timeClient.forceUpdate();
    }
    // The formattedDate comes with the following format:
    // 2018-05-28T16:00:13Z
    // We need to extract date and time
    formattedDate = timeClient.getFormattedDate();
    Serial.println(formattedDate);

    // Extract date
    int splitT = formattedDate.indexOf("T");
    day = formattedDate.substring(0, splitT);
    Serial.println(day);
    // Extract time
    hour = formattedDate.substring(splitT+1, formattedDate.length()-1);
    Serial.println(hour);
    timestamp = day + " " + hour;
}

void setup() {
    // Initialize Serial Monitor
    Serial.begin(115200);
    startLoRA();
    connectWiFi();
}
```

```
if(!SPIFFS.begin()){
    Serial.println("An Error has occurred while mounting SPIFFS");
    return;
}
// Route for root / web page
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send(SPIFFS, "/index.html", String(), false, processor);
});
server.on("/temperature", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/plain", temperature.c_str());
});
server.on("/wheel_speed", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/plain", wheel_speed.c_str());
});
server.on("/fuel_flow", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/plain", fuel_flow.c_str());
});
server.on("/coolant_temperature", HTTP_GET, [](AsyncWebServerRequest
*request){
    request->send_P(200, "text/plain", coolant_temperature.c_str());
});

server.on("/timestamp", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/plain", timestamp.c_str());
});
server.on("/rssi", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/plain", String(rssi).c_str());
});
// Start server
server.begin();
```

```
// Initialize a NTPClient to get time
timeClient.begin();

// Set offset time in seconds to adjust for your timezone, for example:
// GMT +1 = 3600
timeClient.setTimeOffset(0);
}

void loop() {
  // Check if there are LoRa packets available
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    getLoRaData();
    getTimeStamp();
  }
}
```

A6.3 Webpage Code

```
<!DOCTYPE HTML><html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" href="data:,">
  <title>ESP32 (LoRa + Server)</title>
  <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-
fnmOCqbTIWlj8LyTjo7mOUStjsKC4pOpQbqyi7RrhN7udi9RwhKkMHpvLbHG9Sr"
crossorigin="anonymous">
  <style>
    body {
      margin: 0;
      font-family: Arial, Helvetica, sans-serif;
```

```
    text-align: center;
}
header {
    margin: 0;
    padding-top: 5vh;
    padding-bottom: 5vh;
    overflow: hidden;
    background-size: cover;
    color: white;
}
h2 {
    font-size: 2.0rem;
}
p { font-size: 1.2rem; }
.units { font-size: 1.2rem; }
.readings { font-size: 2.0rem; }
</style>
</head>
<body>
    <header>
        <h2>ESP32 (LoRa + Server)</h2>
        <p><strong>Last received packet:<br/><span
id="timestamp">%TIMESTAMP%</span></strong></p>
        <p>LoRa RSSI: <span id="rssi">%RSSI%</span></p>
    </header>
    <main>
        <p>
            <i class="fas fa-thermometer-half" style="color:#059e8a;"></i> Temperature:
            <span id="temperature" class="readings">%TEMPERATURE%</span>
            <sup>&deg;C</sup>
        </p>
```



```

<p>
  <i class="fas fa-tint" style="color:#00add6;"></i> Wheel Speed: <span
id="wheel_speed" class="readings">%WHEEL_SPEED%</span>
  <sup>&#37;</sup>
</p>
<p>
  <i class="fas fa-tint" style="color:#00add6;"></i> Fuel Flow: <span id="fuel_flow"
class="readings">%FUEL_FLOW%</span>
  <sup>&#37;</sup>
</p>
<p>
  <i class="fas fa-tint" style="color:#00add6;"></i> Coolant Temperature: <span id="
coolant_temperature " class="readings">%COOLANT_TEMPERATURE%</span>
  <sup>&#37;</sup>
</p>

</main>
<script>
setInterval(updateValues, 10000, "temperature");
setInterval(updateValues, 10000, "wheel_speed");
setInterval(updateValues, 10000, "fuel_flow");
setInterval(updateValues, 10000, "coolant_temperature");

setInterval(updateValues, 10000, "rssi");
setInterval(updateValues, 10000, "timestamp");

function updateValues(value) {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById(value).innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "http://192.168.1.100:8080/" + value, true);
  xhttp.send();
}

```

```
    }  
};  
xhttp.open("GET", "/" + value, true);  
xhttp.send();  
}  
</script>  
</body>  
</html>
```