WESTERN MICHIGAN UNIVERSITY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE-4820 SENIOR DESIGN II

Claims-Investigation Committee (CIC) Multi-Input Testing Device

FINAL REPORT

December 6, 2024

Faculty Advisor:
Dr. Janos Grantner

Team Members:
Dylan-Matthew Garza
Daniel Baker
Rohullah Sah

Sponsor:
ZF
Contact:
Patrick McNally
Patrick.McNally@zf.com

Contents

1 Abstract

This senior design project developed a comprehensive testing platform for ZF Group's automotive sensor validation, focusing on the Brake Signal Transmitter (BST) and related safety components. The system utilizes a dual-core STM32MP157F-DK2 microcontroller, combining an ARM Cortex-M4 for real-time signal processing with an ARM Cortex-A7 running a custom embedded Linux distribution for test management and user interaction.

The platform features a WebAssembly-based frontend interfacing with a Rust web server, enabling intuitive test configuration and real-time result monitoring. OpenAMP facilitates inter-processor communication between the Cortex-A7 and M4 cores, allowing seamless data transfer between the test execution and management layers. The system validates various automotive sensors against manufacturer specifications, providing automated test execution and detailed reporting capabilities through CSV export.

This solution significantly improves testing efficiency compared to existing methods, offering a scalable architecture that supports multiple device types including the BST, Continuous Wear Sensor, and Electronic Stability Control Module. Built with industry-standard technologies and professional engineering practices, the system demonstrates innovative integration of embedded systems, web technologies, and real-time processing for industrial testing applications.

2 Introduction

The Claims Investigation Committee Multi-Testing Input Device automates validation testing for ZF Group's automotive safety sensors. Built on the STM32MP157F-DK2 platform, it combines an ARM Cortex-M4 for real-time signal processing with a Cortex-A7 running custom embedded Linux for test management.

The system features a WebAssembly frontend with a Rust backend, using OpenAMP for inter-processor communication. It processes PWM signals, analog inputs, and CAN communications to validate devices against manufacturer specifications. While primarily focused on Brake Signal Transmitter (BST) testing, the platform supports additional devices including Continuous Wear Sensors and Electronic Stability Control Modules.

This automated solution improves testing efficiency and consistency while maintaining automotive industry standards. Test results are available through CSV export, enabling detailed analysis and documentation of validation procedures.

3 Discussion

3.1 Background

ZF Group, a global Tier 1 automotive supplier specializing in advanced safety systems, identified a need to enhance their Claims Investigation Center's (CIC) testing capabilities. Based at their North American headquarters in Auburn Hills, MI, the CIC analyzes field failure parts and validates warranty claims for commercial vehicles.

Our project addresses this need by developing an automated testing platform that streamlines the validation process for returned components. The system focuses particularly on safety-critical parts like the Brake Signal Transmitter, supporting the CIC's mission of efficient warranty claim processing and product quality improvement.

3.2 Need Statement

ZF Group urgently requires a modernized testing platform to address critical limitations in their current validation system. The existing mBSP tester is incompatible with new components, cannot support prototype testing, and creates significant delays in product validation. With Daimler's new platform implementation and increasing production volumes, ZF needs a flexible, unified testing solution that can efficiently validate multiple safety-critical components while reducing testing time and costs. This system must support both current and future product lines while maintaining rigorous testing standards for warranty claim validation.

3.3 High-Level System Design

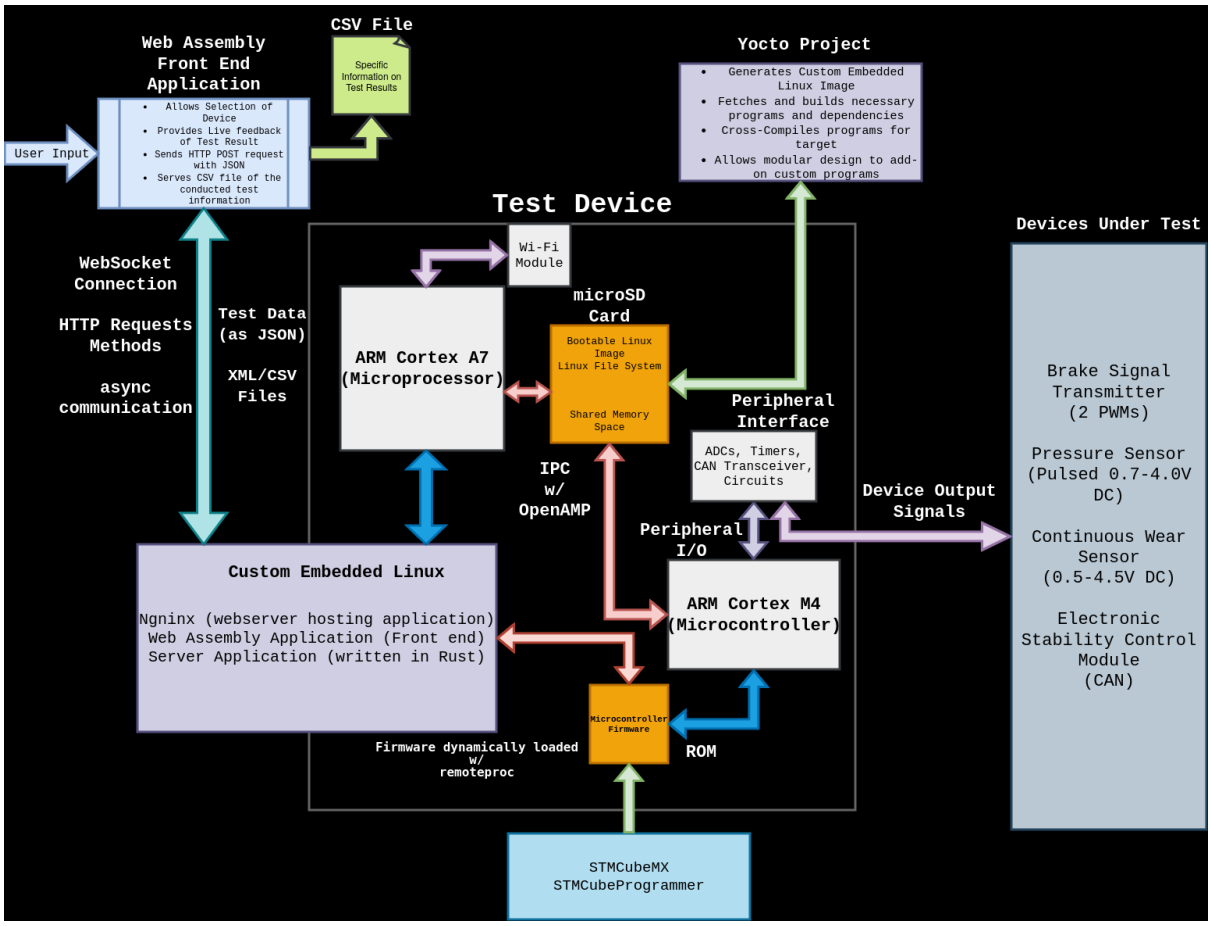

Figure 1: Comprehensive, High-level system block diagram

3.4 Specifications

1. **Device Interfacing**

   1.1 Properly read Device Signals using the ARM Cortex-M4 on the onboard microcontroller on the STM32MP157F-DK2

   - PWM duty cycle
   - Frequency
   - Voltages through an analog-to-digital converter (ADC)
   - CAN frames

2. **Physical Components and Hardware**

2.1 Printed Circuit Board (PCB) for interfacing with DUT

2.2 PCB for scaling and managing power for the DUT and to the microcontroller

2.3 Enclosure for PCBs and STM32MP157F-DK2 board

3. **Software**

3.1 Custom embedded Linux distribution that will run on the onboard ARM Cortex-A7 microprocessor on the STM32MP157F-DK2

3.2 Simple user interface on web-based application

3.3 Custom Webserver to process information from web application to microcontroller

3.4 Communicate collected information from ARM Cortex-M4 to ARM Cortex-A7

3.5 Ability to download measured data, formatted as a CSV, through the web application

3.5 Deliverables

The hardware component centers around custom circuit designs, featuring **two specialized printed circuit boards**: a power management PCB that handles voltage conversion and regulation, and a peripheral interface PCB that manages signal routing and conditioning. Supporting these boards are **protective enclosures** designed to house both the PCBs and the STM32MP157F-DK2 development board, along with a comprehensive **wiring harness system** for reliable device connections.

The software architecture comprises multiple integrated components. At its foundation lies a **custom embedded Linux distribution** built using the Yocto Project, which provides the operating system environment. User interaction is handled through a **WebAssembly-based frontend interface** that enables device selection and test control. A **Rust-based web server** manages device communication and test execution. The system includes specialized **ARM Cortex-M4 firmware** modules for testing various devices: the Brake Signal Transmitter (BST) through PWM signal analysis, the Continuous Wear Sensor (CWS) via voltage measurements, pressure sensor validation, and Electronic Stability Control Module (ESCM) testing through CAN communication.

The testing and documentation deliverables ensure system validation and future maintainability. These include comprehensive **validation test results** demonstrating compliance with manufacturer specifications, a **CSV data export system** for maintaining test records,

detailed **technical documentation** covering system operation procedures, and organized **source code repositories** with thorough documentation to support future maintenance and upgrades. Together, these components form a unified testing platform capable of handling multiple automotive sensor types while maintaining strict compliance with ZF Group's testing requirements.

4 Design and Implementation

4.1 Devices Under Test (DUT)

The Claims Investigation Committee's testing platform is designed to validate multiple critical automotive safety components from ZF's commercial vehicle braking systems. These devices represent key components in the braking system's safety chain, each serving specific functions that together, ensure reliable and safe vehicle operation.

The testing platform currently supports four devices: the Brake Signal Transmitter (BST), Continuous Wear Sensor (CWS), Electronic Stability Control Module (ESCM), and a Pressure Sensor. Each device requires unique testing approaches due to their distinct communication protocols and operational parameters. Understanding these devices' functions and manufacturer specifications is crucial for implementing accurate testing procedures.

4.1.1 Brake Signal Transmitter (BST)

The Brake Signal Transmitter (BST) is a critical component in ZF's commercial vehicle braking systems, providing real-time brake status information to the vehicle's Electronic Control Unit (ECU). The Brake Signal Transmitter generates the electrical and pneumatic signals that represent the braking demand of the driver within the modular Braking System 2.0 (mBSP 2.0) Electronic Brake System (EBS). The device has a single circuit pneumatic design and contains two redundant pedal stroke sensors and a switch that is only used in European design. To reduce the exhaust noise, the BST has a built-in silencer. The BST is actuated by a foot pedal which is provided by the vehicle manufacturer (In this case specifically Daimler Trucks North America). Two Pulse Width Modulation (PWM) signals are generated by the BST, as long as both signals are powered.

The BST's electrical architecture is designed for robust operation in commercial vehicle environments. The BST uses a contact system of HDSCS by TYCO (AMP MCP 1.5/2.8 MIX 7 - pole linear bayonet) for electrical connections. Operating on a nominal 12V DC system, the device maintains functionality across an 8V to 16V DC voltage range with built-in protection

against reverse polarity and overvoltage conditions. Each PWM channel draws no more than 50 mA current, with total load capacitance limited to 6.8nF per channel. The output signals are similarly protected, with maximum current limited to 15 mA per channel, incorporating short-circuit protection and reverse polarity protection to prevent damage to the device or the vehicle's electrical system.

The core functionality of the BST centers on its stroke sensor characteristics, which generate two complementary PWM signals (S1 and S2) that precisely indicate brake pedal position. These signals operate at 200Hz ±10Hz and follow specific duty cycle relationships:

- Signal S1 initiates at 12.5% duty cycle when the brake pedal is at rest (0mm stroke), increasing linearly at 5.96% per millimeter of pedal stroke.
- Signal S2 initiates at 87.5% duty cycle when the brake pedal is at rest (0mm stroke), decreasing linearly at 5.96% per millimeter of pedal stroke.
- Both signals maintain +- 5% duty cycle tolerance throughout the stroke range.
- The signals intersect at approximately 6.3mm oedal stroke, with both signals at 50% duty cycle.

The PWM signals are defined by several key parameters:

- Amplitude varying between U_{max} (determined by CCU pull-up voltage) and U_{min} (<1V)
- Period T = 1/f, where f is the 200Hz carrier frequency
- Duty cycle calculated as DC = T_{high}/T_{total}

To ensure reliable signal integrity, the BST interface requires specific loading conditions:

- Pull-up resistors: 4kΩ ±1% and 3kΩ ±1% to 12V
- Filtering capacitors: 2.2nF ±10%

The BST's implementation in Daimler's 2025 platform underscores the importance of accurate validation testing. As the highest volume commercial vehicle platform in North America, ensuring consistent and reliable BST operation is crucial for vehicle safety and warranty claim processing. This testing platform's ability to precisely measure PWM characteristics, timing relationships, and stroke correlations provides essential validation capabilities for both production quality assurance and field return analysis.

4.1.2 Continuous Wear Sensor (CWS)

The Continuous Wear Sensor represents a critical monitoring component in ZF's MAXX 2.0 air disc brake generation for commercial vehicles, specifically designed for trucks and buses. The sensor system integrates multiple functions to provide comprehensive brake wear monitoring and maintenance capabilities.

The primary function of the CWS centers on precise wear measurement through contactless sensing technology. Mounted directly on the disc brake caliper, the sensor monitors the adjuster position to determine the collective wear state of both brake pads and the brake disc rotor. This measurement is achieved through a sophisticated distance-to-voltage conversion system that maintains both linear and ratiometric characteristics, ensuring accurate wear detection throughout the brake system's operational life.

A secondary but crucial feature of the CWS is its integrated re-adjustment capability. A dedicated re-adjustment screw or shaft is incorporated into the sensor design, enabling controlled retraction of the brake adjuster unit during maintenance procedures. This integration streamlines service operations while maintaining measurement accuracy.

The sensor's output signal serves multiple critical functions within the Electronic Control Unit (ECU). These functions include continuous wear monitoring, wear limit detection, predictive brake lining service forecasting, wear harmonization across brake components, and comprehensive brake function monitoring. This multi-faceted approach ensures both safety and maintenance optimization.

The electrical architecture of the CWS is designed for robust operation in commercial vehicle environments. The sensor utilizes a 3-pole BVA connector system with specific pin assignments:

- Supply Voltage (Us): 5V DC ±0.5V (Pin A)
- Output Signal (Uout): 0.7V to 4.0V DC (Pin B)
- Ground Reference (Pin C)

The sensor's output characteristics follow a precise voltage-to-wear relationship across four distinct operational ranges:

1. New Brake Pad Condition
    - Output Voltage: 0.7V ±2% (0.630V to 0.770V)
    - Corresponding Position: 18.5mm

2. Normal Wear Progression
    - Output Range: 0.7V to 3.5V
    - Position Range: 18.5mm to 53.5mm
    - Linear voltage increase with wear progression
3. Wear Limit Threshold
    - Output Voltage: 3.5V ±1% (3.465V to 3.535V)
    - Corresponding Position: 53.5mm ±0.5mm
4. Worn Out Condition
    - Output Voltage: 4.0V ±5% (3.8V to 4.2V)
    - Position: >53.5mm

The electrical design incorporates multiple protection features to ensure reliable operation. These include reverse polarity protection up to -5.5V DC, short-circuit protection on the output signal, and current consumption limited to 25mA maximum. When integrated with the ECU, the supply voltage is typically pulsed rather than constant, optimizing power consumption while maintaining measurement accuracy.

The CWS's ratiometric output characteristic ensures measurement stability despite supply voltage fluctuations, as the output signal maintains its proportional relationship to the input voltage. This design feature, combined with the linear response to wear progression, enables precise wear monitoring throughout the brake system's service life. The sensor's output is processed by the ECU to provide real-time wear status information, enabling proactive maintenance scheduling and ensuring optimal brake system performance.

4.1.3 Electronic Stability Control Module (ESCM)

The Electronic Stability Control Module represents a critical safety system in ZF's commercial vehicle brake systems, specifically engineered for trucks and buses. The module integrates sophisticated sensor technology with advanced processing capabilities to enhance vehicle stability and prevent loss of control situations.

The primary function of the ESCM centers on continuous monitoring of vehicle dynamics through integrated sensor technology. At its core, the module contains a yaw rate sensor that precisely measures the vehicle's rotational motion around its vertical axis, coupled with

acceleration sensors that monitor both lateral and longitudinal movements. This multi-sensor approach enables comprehensive monitoring of vehicle behavior in dynamic driving situations.

A key feature of the ESCM is its ability to detect and respond to vehicle instability conditions. The module continuously analyzes sensor data to identify potential skidding about the vertical axis and rollover tendencies. This real-time analysis enables the system to initiate corrective actions through the brake system when instability is detected, representing a crucial advancement in commercial vehicle safety systems.

The mechanical architecture reflects the module's critical safety role. The device's construction centers around a sophisticated printed circuit board that houses the sensor array, microcontroller, and CAN interface components. This PCB is secured within a molded plastic housing, which includes integrated mounting points for secure attachment to the vehicle frame, ensuring stable sensor readings in dynamic driving conditions.

The electrical architecture of the ESCM is designed for robust operation in commercial vehicle environments. The module operates with the following specifications:

- Operating voltage range: 8V to 32V DC
- Normal current consumption: 60mA to 120mA
- CAN bus configuration: 500 KBaud with 120Ω ±5

The communication interface utilizes CAN 2.0A protocol, enabling reliable data exchange with other vehicle systems. The module demonstrates precise timing characteristics in its startup sequence, achieving initial CAN message transmission within 160-200ms of power application. Full sensor signal validation is established within 1000ms, ensuring reliable stability control once the vehicle is operational.

The ESCM's sophisticated processing system integrates the sensor inputs with vehicle dynamics models to provide comprehensive stability control. This integration enables the module to detect complex vehicle behaviors and initiate appropriate responses through the brake system. By monitoring yaw rates and accelerations, the system can identify potentially dangerous vehicle states before they become critical, allowing for preventive intervention.

The module's output through the CAN bus provides critical vehicle dynamics information to other control systems, supporting coordinated vehicle safety responses. This communication capability, combined with the module's sophisticated sensor array and processing capabilities, establishes the ESCM as a fundamental component in ensuring commercial vehicle safety and stability.

4.1.4 Pressure Sensor

The pressure sensor represents a critical measurement component in ZF's commercial vehicle control systems, serving as an actual-value transmitter for continuous pressure monitoring. The sensor system integrates sophisticated measurement technology with comprehensive protection features to ensure reliable operation in demanding automotive environments.

The primary function of the pressure sensor centers on precise pressure measurement through piezo-resistive sensing technology. A silicon diaphragm forms the core sensing element, with measuring resistors precisely positioned to form a Wheatstone bridge circuit. When pressure acts upon the diaphragm, the resulting mechanical deformation induces proportional changes in the resistors' values, creating a measurable electrical signal that accurately represents the applied pressure relative to atmospheric conditions.

The sensor's signal processing capabilities represent a crucial aspect of its design. The bridge output signal undergoes internal amplification and calibration processes, with integrated temperature compensation ensuring measurement stability. This sophisticated processing approach enables accurate pressure monitoring across varying environmental conditions while maintaining signal integrity throughout the measurement range.

The sensor's functionality extends beyond basic pressure measurement through comprehensive protection features. The design incorporates reverse polarity protection, overvoltage protection, and safeguards against high-frequency disturbances. Combined with short-circuit protection, these features ensure reliable operation in electro-pneumatic control systems where electrical noise and power fluctuations are common challenges.

The electrical architecture of the pressure sensor is designed for robust operation in commercial vehicle environments. The sensor utilizes a standardized three-pole connector system with specific pin assignments:

- Supply Voltage (UB): 8V to 32V DC (Pin 1)
- Output Signal (Uout): 0.5V to 4.5V DC (Pin 2)
- Ground Reference (Pin 3)

The sensor's output characteristics follow a precise voltage-to-pressure relationship with clearly defined parameters:

- Offset voltage: 0.5V at atmospheric pressure (0 bar)
- Linear sensitivity: 0.4V per bar across measurement range

- Maximum output: 4.5V at full scale
- Characteristic limit: 4.7V ±100mV

The electrical design incorporates specific requirements for signal integrity and protection. The output circuit requires minimum load impedances of 5.1kΩ to ground and 68kΩ to supply voltage, ensuring reliable signal transmission. Current consumption is limited to 15mA maximum, while operational readiness is achieved within 10 milliseconds of power application, enabling rapid pressure monitoring in dynamic vehicle systems.

The pressure sensor's ratiometric output characteristic ensures measurement stability despite supply voltage fluctuations. This design feature, combined with the sensor's broad operating voltage range and comprehensive protection features, enables reliable pressure measurement in demanding automotive applications. The sensor's output provides critical pressure information to the vehicle's control systems, supporting safe and efficient operation of pneumatic and hydraulic systems in commercial vehicles.

4.2 Custom Printed Circuit Board for Device Interfacing and Power Management

This project involved the development of schematic designs and custom PCBs to manage power supply and interface with various devices under test (DUTs) using KiCad. The designs were implemented to ensure stable power distribution and accurate signal processing for the DUTs, with specific focus on hardware efficiency and durability.

4.2.1 Power Supply Schematic Design

The power supply schematic begins with a 16V DC input, connected via a power jack, which acts as the main source of power for the system. 10uF and 22uF Capacitors are placed before and after each voltage regulator to ensure a stable voltage level and filter out any electrical noise. These capacitors also prevent voltage fluctuations, ensuring a stable supply to the components.

Voltage Regulation

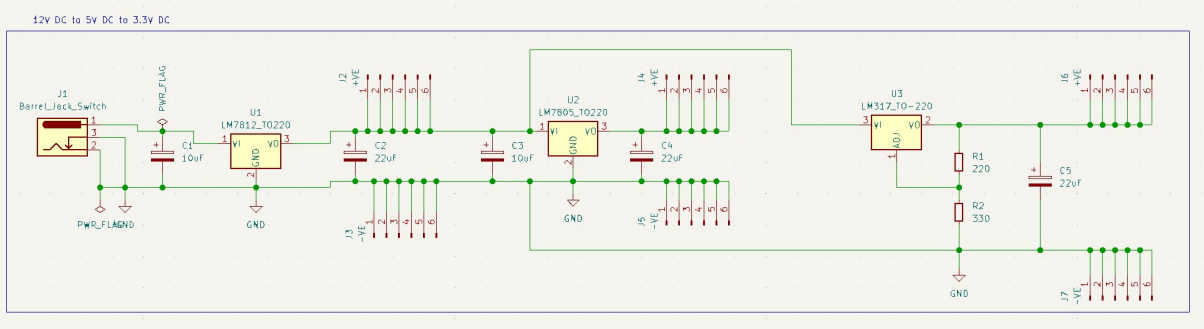

Figure 2: Schematic Design of the Power Supply Management System

- The LM7812 voltage regulator converts the 16V DC input to 12V DC.
- The LM7805 voltage regulator further regulates the 12V DC to a 5V DC, suitable for devices that require low voltage.
- The LM317 adjustable regulator is used to step down the voltage for the microcontroller, providing a stable 3.146V DC. This precise voltage is achieved by using a combination of 220-ohm and 330-ohm resistors, which set the voltage according to the parameters.

  This layered voltage regulation ensures that all peripherals and the microcontroller receive clean and safe power levels for proper operation.

4.2.2 Peripherals Schematic Design

The peripherals schematic shows the design and signal conditioning circuits for four Devices Under Test (DUT's): the Brake Signal Transmitter (BST), the Continuous Wear Sensor (CWS), the Pressure Sensor, and the String Potentiometer. These circuits ensure accurate data acquisition and compatibility with the STM32 Board by conditioning, scaling and protecting the signals.

Brake Signal Transmitter

The Brake Signal Transmitter is designed to read the Pulse Width Modulation (PWM) signals, which represent the brake's operational state. The circuit includes resistors, such as 3k-ohms and 8.2k-ohms, to scale the PWM signals down to levels safe for the microcontroller. Additionally, a 2.2nF capacitor is used to filter out high-frequency noise, ensuring a clean

and stable signal. 1N4001 diodes are included for overvoltage protection, safeguarding the microcontroller from potential voltage spikes.

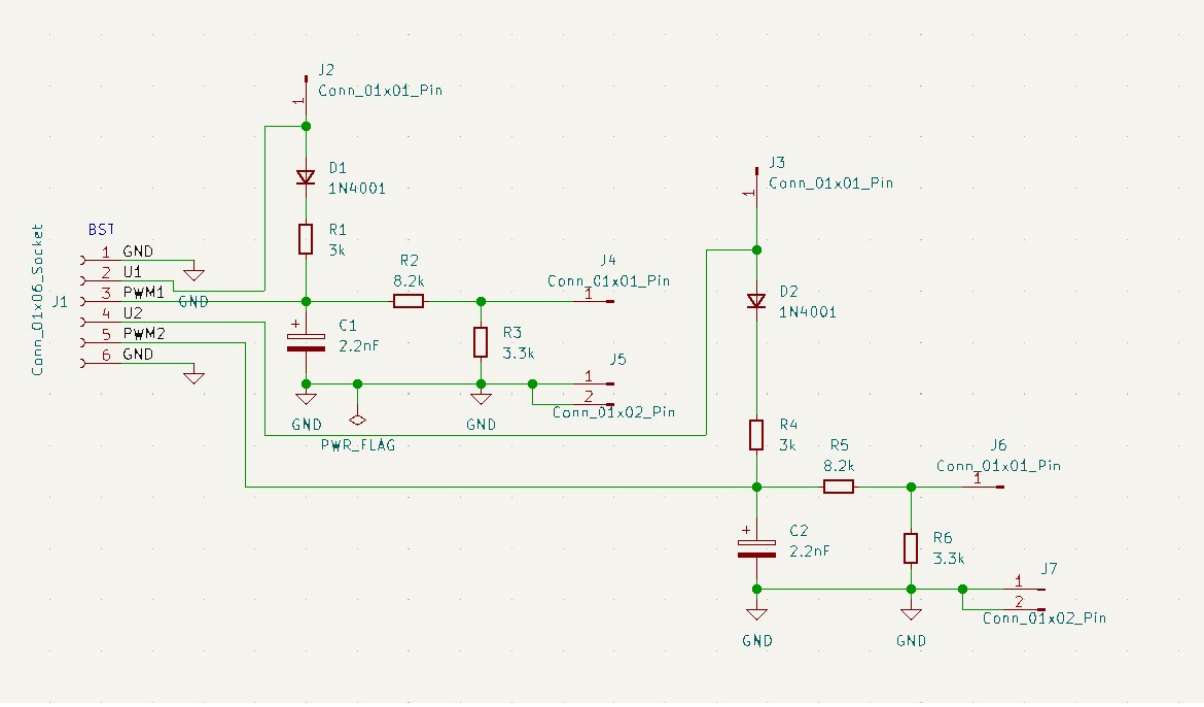

Figure 3: Schematic Design of the Brake Signal Transmitter

Continuous Wear Sensor

The Continuous Wear Sensor (CWS) captures analog voltage signals to monitor brake pad wear. This schematic is relatively simple but effective, using 3k-ohms resistors for signal scaling and stabilization. Proper grounding ensures that the integrity of the signal is minimized and maintain consistent readings.

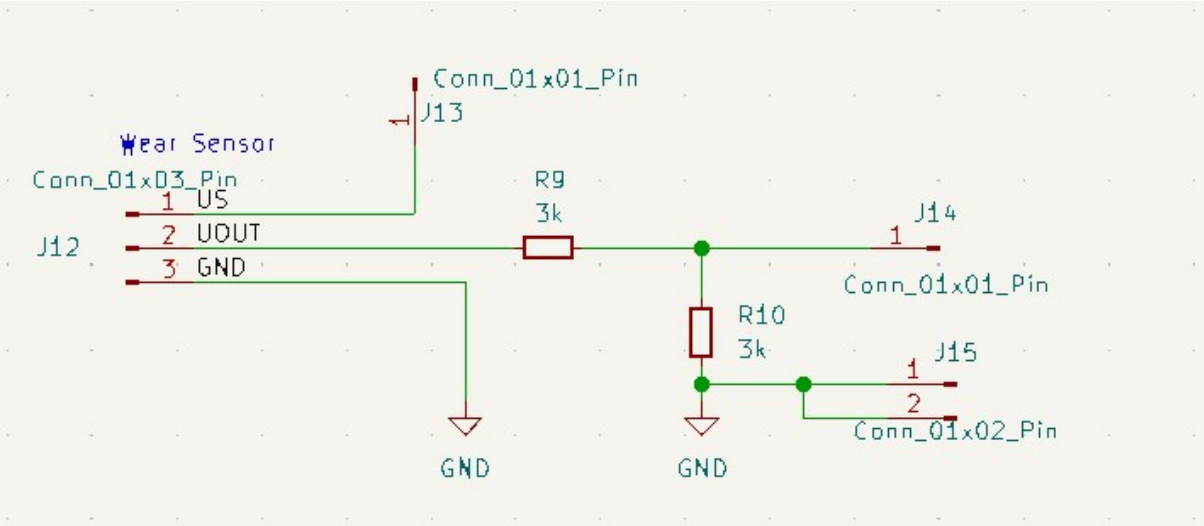

Figure 4: Schematic Design of the Continuous Wear Sensor

Pressure Sensor

The Pressure Sensor measures the pressure within the braking system. It outputs an analog voltage signal, which is conditioned using a 330uF capacitor to stabilize the output and eliminate fluctuations. Additionally, 3k-ohms resistors are used to scale and prepare the signal for processing by the microcontroller.

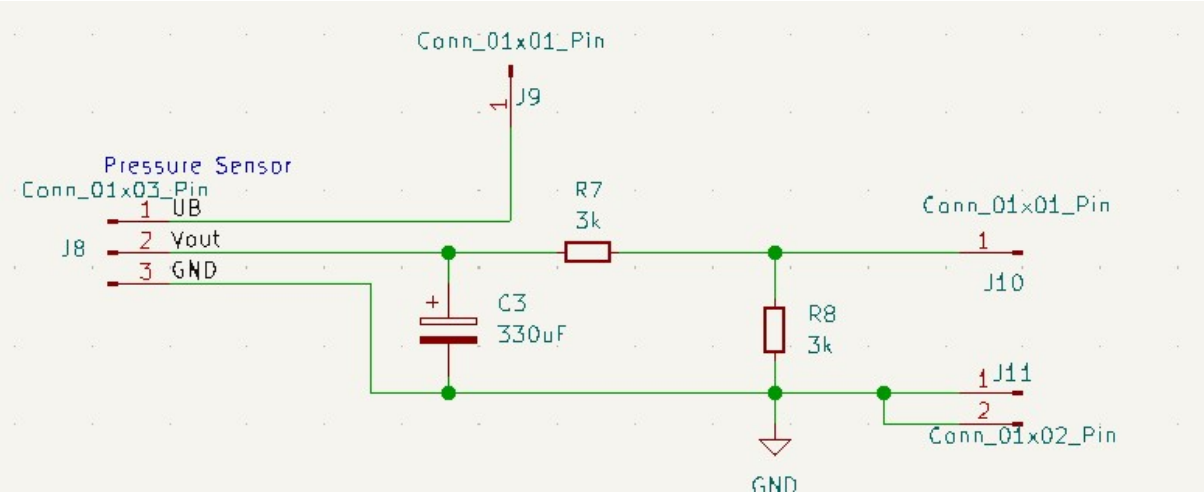

Figure 5: Schematic Design of the Pressure Sensor

String Potentiometer

The String Potentiometer is used to monitor the displacement, such as the movement of the brake pedal, and converts it into proportional voltage signal. This circuit uses resistors, including 8.5k-ohms and 3.3k-ohms, to scale the output signal to levels suitable for the microcontroller. As with the other peripherals, stable grounding and power inputs are implemented to maintain accurate and reliable signal transmission.

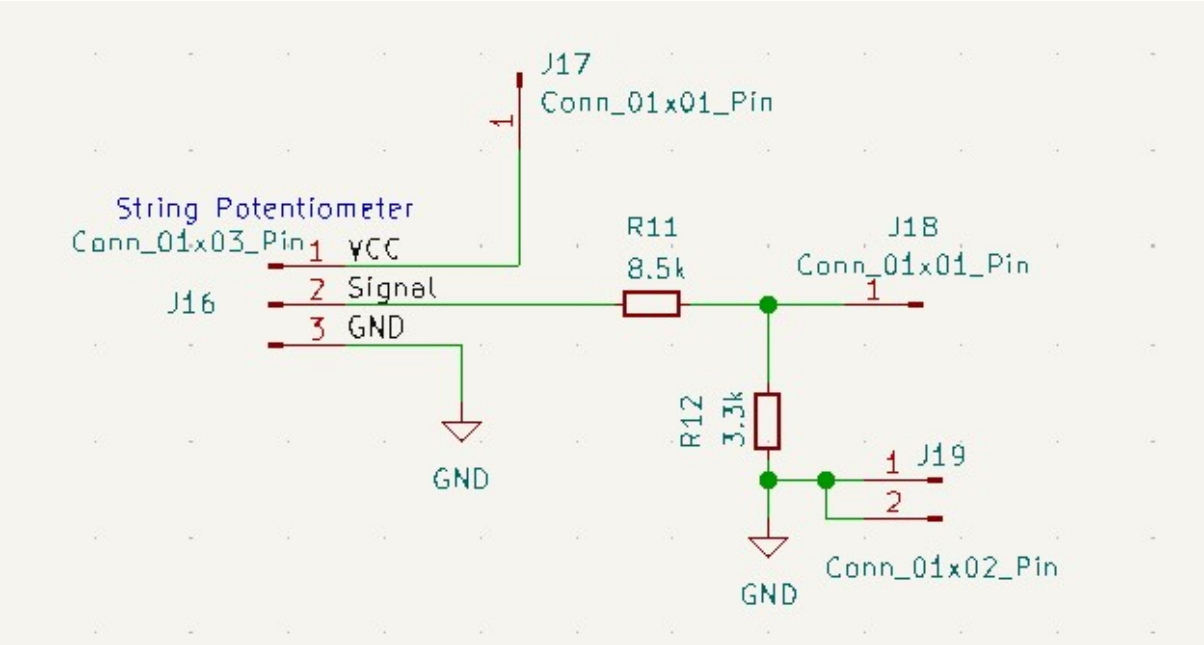

Figure 6: Schematic Design of the String Potentiometer

Overall, the peripherals schematics focuses on providing clean, conditioned, and scaled signals from each DUT's to the STM32 microcontroller while including corresponding components to prevent damage or errors. This ensures the entire system operates efficiently and reliably, with accurate data acquisition from all devices.

4.2.3 Printed Circuit Board (PCB's) Design

Power Supply PCB

The Power Supply PCB is a critical component designed to regulate and provide multiple voltage levels to the system. It starts at 16V DC input through a barrel jack connector. This 16V input is directed through a series of voltage regulators that step down the voltage to meet the requirements of the various components in the circuit.

The board utilizes capacitors of 10μF and 22μF, which are strategically placed before and after each voltage regulator. These capacitors help to stabilize the voltage output by filtering out noise and ensuring smooth operation. The LM7812 regulator steps down the voltage to 12V DC, which is essential for certain peripherals. This 12V is then fed into another voltage regulator circuit comprising an LM317 adjustable voltage regulator along with two resistors, 220Ω and 330Ω, to further step it down to 3.146V DC. This precise voltage level is crucial for the operation of the microcontroller.

The design includes multiple output pins for 12V, 5V, and 3.3V voltages, ensuring compatibility with different devices and peripherals connected to the system. The layout of the PCB is compact yet efficient, allowing for adequate heat dissipation. The traces are kept wide enough to handle current flow effectively, and proper spacing is maintained to prevent any electrical shorts.

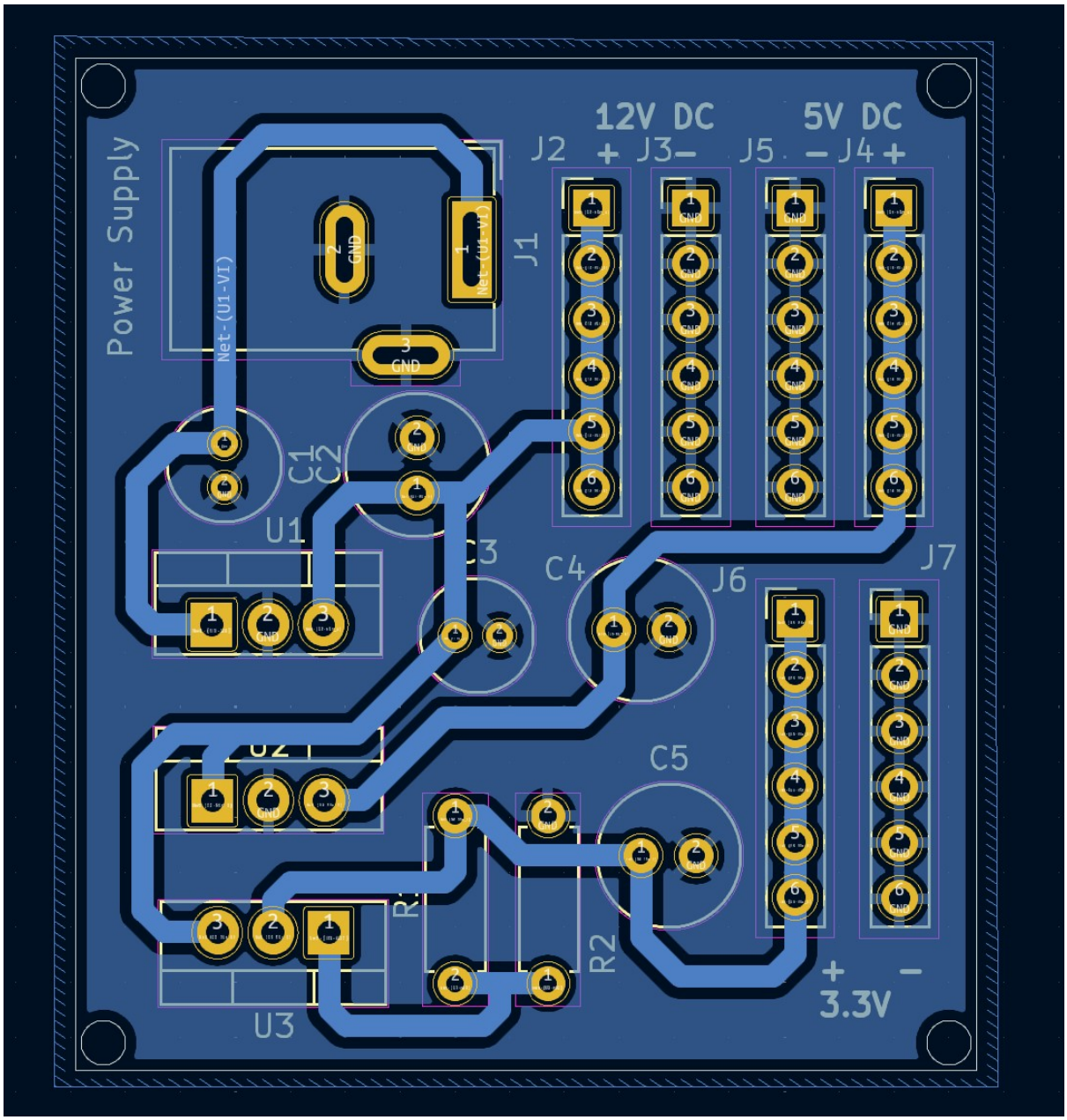

Figure 7: Printed Circuit Board Design of the Power Supply Management

Reference	Value	Datasheet	Footprint	Qty	DNP
C1,C3	10uF	~	Capacitor_THT:C_Radial_D5.0mm_H7.0mm_P2.00mm	2	
C2,C4,C5	22uF	~	Capacitor_THT:C_Radial_D6.3mm_H7.0mm_P2.50mm	3	
J1	Barrel_Jack_Switch	~	Connector_BarrelJack:BarrelJack_Wuerth_6941xx301002	1	
J2,J4,J6	Positive Pin Headers	~	Connector_PinHeader_2.54mm:PinHeader_1x06_P2.54mm_Vertical	3	
J3,J5,J7	Negative Pin Headers	~	Connector_PinHeader_2.54mm:PinHeader_1x06_P2.54mm_Vertical	3	
R1	220	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	1	
R2	330	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	1	
U1	LM7812_TO220	https://www.onsemi.cn/PowerSolutions/document/MC7800-D.PDF	Package_TO_SOT_THT:TO-220-3_Vertical	1	
U2	LM7805_TO220	https://www.onsemi.cn/PowerSolutions/document/MC7800-D.PDF	Package_TO_SOT_THT:TO-220-3_Vertical	1	
U3	LM317_TO-220	http://www.ti.com/lit/ds/symlink/lm317.pdf	Package_TO_SOT_THT:TO-220-3_Vertical	1	

Figure 8: Bills of Material (BOM) for Power Supply PCB

Peripheral PCB Design

The Peripherals PCB is designed to accommodate all the connections and conditioning circuits required for the various devices under test (DUTs), including the Brake Signal Transmitter (BST), the Continuous Wear Sensor (CWS), the Pressure Sensor, and the String Potentiometer. Each DUT is connected through dedicated input and output pins, clearly labeled on the board for ease of use.

The BST section includes diodes, resistors, and capacitors to read PWM signals reliably. The diodes provide protection against voltage spikes, while the resistors and capacitors condition the PWM signals for accurate processing. Similarly, the Pressure Sensor section features a capacitor to stabilize the signal and resistors to scale the output voltage for the microcontroller.

The Continuous Wear Sensor section captures analog voltage signals representing brake wear. This signal is scaled and filtered using resistors and capacitors to ensure accurate readings. The String Potentiometer section, on the other hand, converts displacement into a proportional voltage signal. Resistors in this section ensure the signal remains within the safe operating range for the microcontroller.

The PCB layout emphasizes clean and efficient routing, ensuring that signals are isolated to prevent interference. The components are arranged logically, with each DUT section clearly mentioned, making the board easy to debug and maintain.

Both the Power Supply and Peripherals PCBs were designed using KiCad, an open-source PCB design tool. KiCad's features allowed precise component placement and trace routing, ensuring reliable performance and manufacturability of the boards. These PCBs were custom designed to meet the specific requirements of the project, balancing functionality with simplicity.

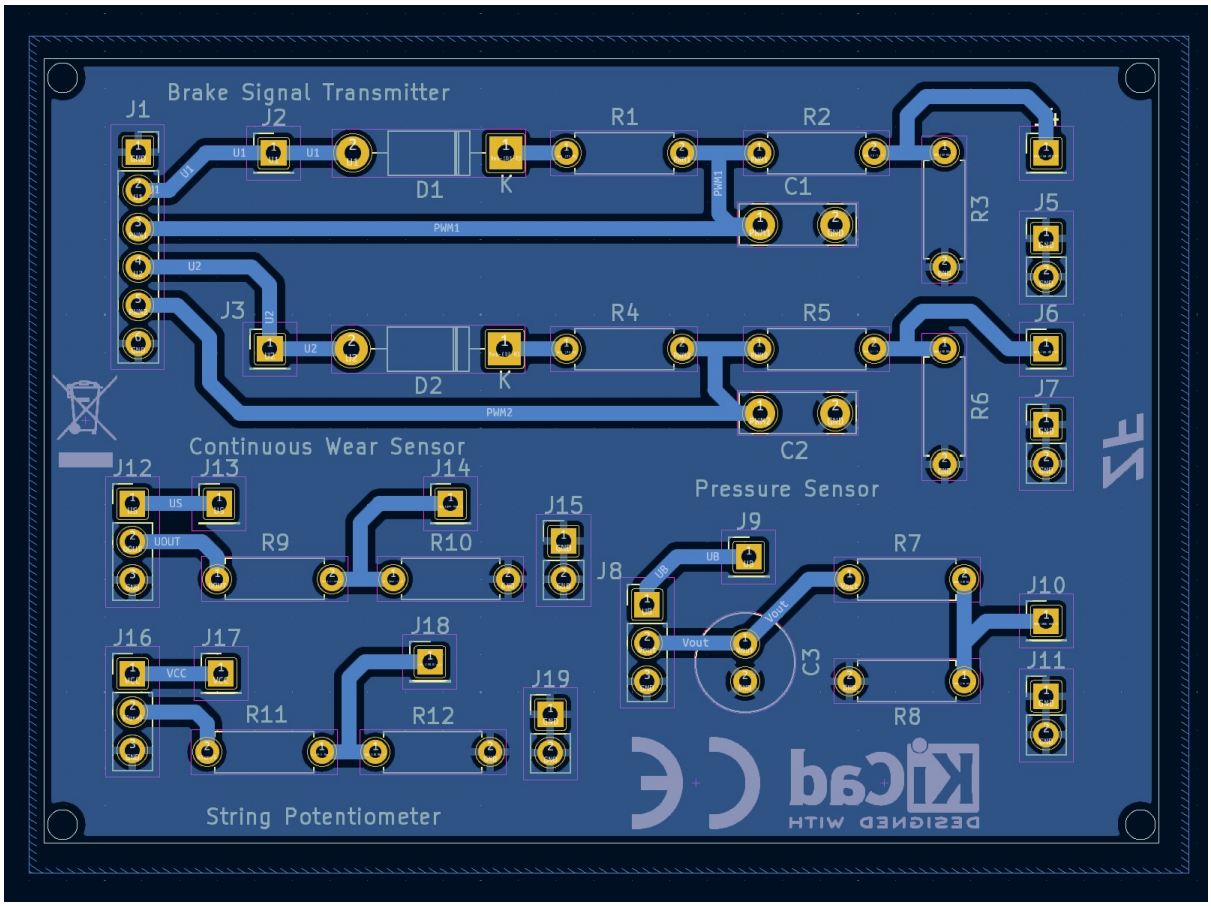

Figure 9: Printed Circuit Board Design of the Peripherals

Reference	Value	Datasheet	Footprint	Qty	DNP
C1,C2	2.2nF	~	Capacitor_THT:C_Disc_D7.5mm_W2.5mm_P5.00mm	2	
C3	330uF	~	Capacitor_THT:C_Radial_D6.3mm_H11.0mm_P2.50mm	1	
D1,D2	1N4001	~	Diode_THT:D_DO-41_SOD81_P10.16mm_Horizontal	2	
J1	Conn_01x06_Socket	~	Connector_PinSocket_2.54mm:PinSocket_1x06_P2.54mm_Vertical	1	
J2,J3,J4,J6,J9,J10,J13,J14,J17,J18	Conn_01x01_Pin	~	Connector_PinHeader_2.54mm:PinHeader_1x01_P2.54mm_Vertical	10	
J5,J7,J11,J15,J19	Conn_01x02_Pin	~	Connector_PinHeader_2.54mm:PinHeader_1x02_P2.54mm_Vertical	5	
J8,J12,J16	Conn_01x03_Pin	~	Connector_PinHeader_2.54mm:PinHeader_1x03_P2.54mm_Vertical	3	
R1,R4,R7,R8,R9,R10	3k	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	6	
R2,R5	8.2k	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	2	
R3,R6,R12	3.3k	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	3	
R11	8.5k	~	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P7.62mm_Horizontal	1	

Figure 10: Bills of Materials for Peripheral PCB Design

4.2.4 Enclosure Design

The enclosures designed for the power supply and peripherals PCBs, as well as the STM32 microcontroller, were custom-built to meet the specific needs of the project. These enclosures

were created using Autodesk's Fusion 360, allowing for precision in their design and ensuring they provided both functional and inventive advantages.

Power Supply and Peripherals Enclosure

The combined enclosure for the power supply and peripherals PCBs was designed with functionality as the primary focus. It ensures the protection of the internal PCBs while also providing easy access for connections and maintenance. Slots were strategically placed for the DC jack and the input/output pins, which allowed efficient cable management and straightforward connectivity during operation. Additionally, ventilation slots were included to maintain stable temperatures and prevent overheating during extended usage.

The interior design included standoffs to securely mount the PCBs, preventing any physical movement that could damage connections. The two-part assembly of the enclosure made it user-friendly for repairs, upgrades, or debugging tasks. This modular design simplified the assembly and disassembly process while ensuring durability.

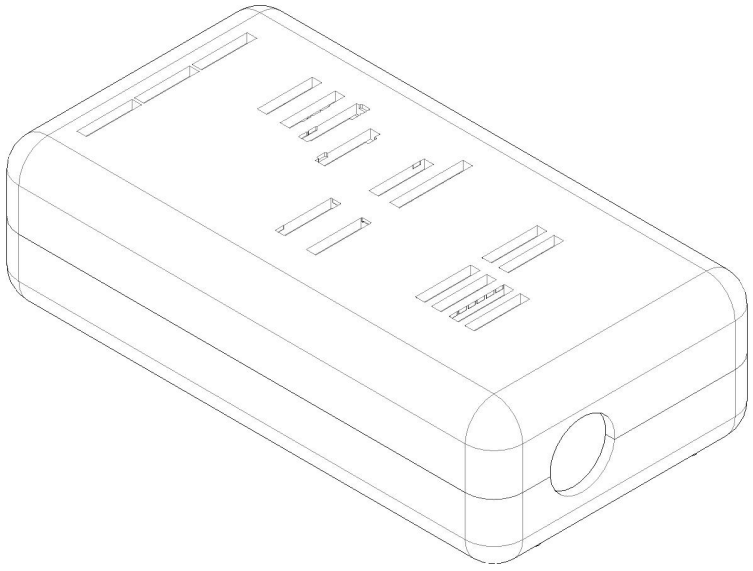

Figure 11: Enclosed Enclosure for Power Supply PCB and Peripherals PCB

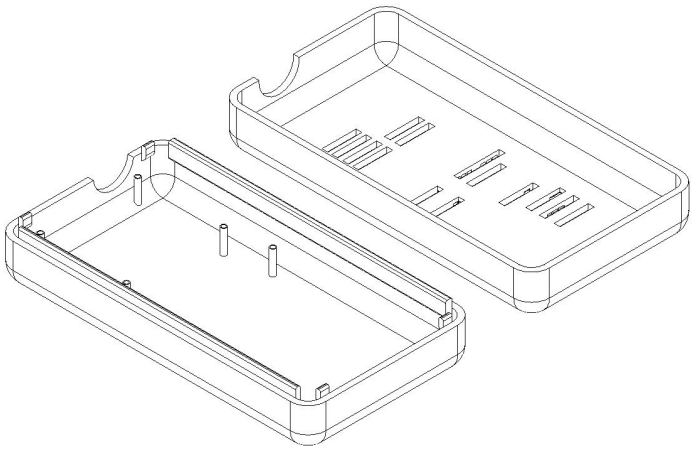

Figure 12: Open Enclosure for Power Supply PCB and Peripherals PCB

STM32 Microcontroller Enclosure

The STM32 enclosure was designed with the same principles of protection and accessibility but tailored specifically to house the microcontroller and its associated components. It featured openings for all essential ports, including USB, power, and communication ports, ensuring seamless integration with other devices and systems.

The enclosure's robust design protected the STM32 board from physical damage and environmental factors while maintaining airflow through ventilation slots to ensure thermal stability. Just like the power supply and peripherals enclosure, it was created as a two-part assembly, which facilitated quick access to the microcontroller during testing or maintenance.

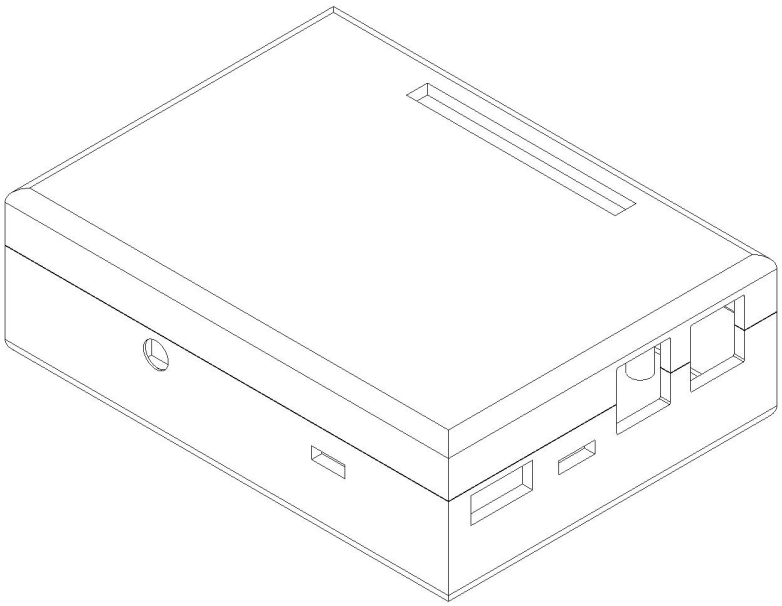

Figure 13: Enclosed Enclosure for the STM32 Board

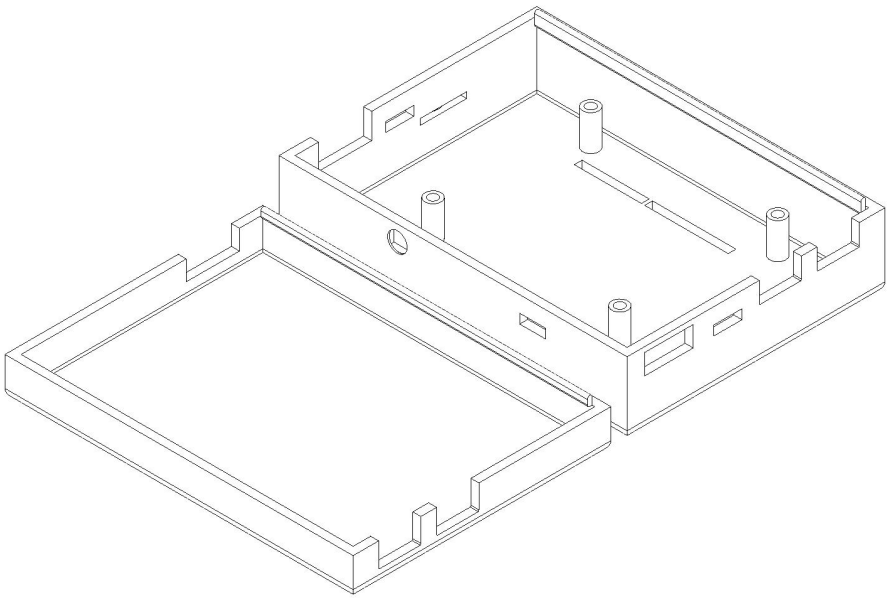

Figure 14: Open Enclosure for the STM32 Board

4.3 Arm Cortex-M4 Firmware

The STM32MP157F-DK2's heterogeneous architecture combines an ARM Cortex-A7 application processor with an ARM Cortex-M4 coprocessor on a single chip. Our testing platform leverages this architecture by implementing dedicated signal processing firmware on the Cortex-M4 core, allowing real-time signal acquisition and analysis while maintaining responsive system operation through the Linux-based interface on the Cortex-A7.

The firmware implementation centers on three key functions: signal acquisition, data processing, and inter-processor communication. For signal acquisition, the firmware configures multiple hardware peripherals of the STM32MP157F-DK2. Timer modules operate in input capture mode to measure PWM signals with microsecond precision. The Analog-to-Digital Converter (ADC), configured with Direct Memory Access (DMA), enables efficient sampling of analog voltage signals without processor intervention. Digital inputs are monitored through GPIO ports with interrupt capabilities for immediate response to state changes.

Data processing occurs in real-time as signals are acquired. The firmware implements specific validation algorithms for each device under test, comparing measured values against manufacturer specifications. These algorithms account for different signal types - from simple analog voltages to complex PWM timing relationships. The firmware maintains separate processing routines for each device type while sharing common utilities for timing, averaging, and error detection.

Inter-processor communication forms the bridge between signal processing and the user interface. The firmware utilizes OpenAMP (Open Asymmetric Multi-Processing) framework to establish reliable communication with the Cortex-A7 core. Through OpenAMP's RPMsg protocol, the firmware creates a virtual UART interface, allowing bidirectional communication. The Cortex-M4 receives test commands from the Cortex-A7 and responds with real-time measurement data, formatted as CSV records for straightforward analysis and storage.

The firmware operation begins when a user selects a device for testing through the web application. This selection triggers the Rust web server on the Cortex-A7 to dynamically load the appropriate firmware (.elf file) onto the Cortex-M4 using the remoteproc framework. Each device under test has its own dedicated firmware implementation that configures the necessary peripherals and implements device-specific signal processing routines. When the firmware successfully loads, it establishes communication with the Cortex-A7 through OpenAMP's RPMsg protocol, creating a virtual UART interface. This interface allows the firmware to receive test parameters and transmit measurement data back to the Linux environment. The firmware continuously validates signals against manufacturer specifications while logging data in CSV format to the Linux kernel trace buffer. This dynamic firmware

loading approach provides clear separation between different device tests while maintaining a consistent interface for the web application.

This implementation offers several advantages over a state-machine approach. Each firmware build contains only the code necessary for its specific device, reducing complexity and potential errors. The dynamic loading system also allows for easier updates and modifications to individual device test routines without affecting the rest of the system. When testing completes or a new device is selected, the current firmware can be safely unloaded and replaced with another test firmware through the remoteproc framework.

This firmware architecture enables comprehensive testing of multiple automotive components through a unified interface. Each device under test has dedicated signal processing routines tailored to its specific requirements, while sharing common infrastructure for timing, communication, and data management. This modular approach allows straightforward addition of new device types while maintaining consistent operation and reliable results.

4.3.1 Brake Signal Transmitter (BST) Testing

he BST firmware implements precise PWM signal analysis to validate brake actuation signals against manufacturer specifications. This testing is critical as the BST represents a primary safety component in the brake system, providing brake demand signals to the Electronic Control Unit (ECU). The firmware must accurately measure and validate two complementary PWM signals that represent brake pedal position. The signal acquisition centers on two key measurements: the PWM signals (S1 and S2) that represent brake pedal position. These signals operate at 200Hz ±10Hz with complementary duty cycles. Signal S1 begins at 12.5For accurate signal measurement, the firmware implements timer-based input capture using interrupt handlers. These handlers precisely measure both the period and pulse width of each PWM signal:

```
#define STROKE_MIN 1
#define STROKE_MAX 9
#define DUTY_CYCLE_TOLERANCE 5.0f     // ±5% DC per specs
#define SENSITIVITY 5.96f             // 5.96% DC/mm per specs
```

The implementation includes sophisticated signal validation that considers several critical parameters. The firmware first captures initial offset values from both PWM channels when testing begins, establishing the baseline for subsequent measurements. During operation, it continuously monitors signal frequency stability, ensuring both channels maintain the required 200Hz ±10Hz timing. Each PWM measurement undergoes multiple validation steps:

1. Frequency Validation: Both signals must maintain 200Hz ±10Hz
2. Duty Cycle Range: Values must stay within physical limits (approximately 10-90
3. Complementary Relationship: S1 and S2 must maintain their inverse relationship
4. Sensitivity Validation: Changes in duty cycle must correspond to the specified 5.96

The firmware offers two testing modes based on configuration from the web interface. In standard mode, the firmware calculates estimated pedal stroke from the PWM signals themselves using the linear relationship defined by the sensitivity specification. When enabled, an optional string potentiometer provides direct stroke measurement through an ADC channel configured with DMA for enhanced validation accuracy. This dual-mode capability allows for both production testing and detailed analysis scenarios. For data logging, the firmware writes timestamped measurements to the Linux kernel trace buffer in CSV format. Each record includes:

```
time(s),duty_cycle1(%),duty_cycle2(%),stroke(mm),freq1(Hz),freq2(Hz)
```

The BST firmware implements comprehensive error detection, identifying issues such as:

- Signal frequency deviation outside ±10Hz tolerance
- Excessive duty cycle changes between measurements (>50
- Loss of complementary relationship between S1 and S2
- Stroke position outside valid range (1mm to 9mm)

Test results are compiled by evaluating measurements against specification tolerances at multiple stroke positions. The firmware transmits the final pass/fail status through the RPMsg interface after completing its validation sequence. This detailed analysis, combined with the comprehensive measurement log, provides thorough validation of the BST's operational characteristics and compliance with manufacturer specifications.

4.3.2 Continuous Wear Sensor (CWS) Testing

The CWS firmware provides precise analog voltage measurement and analysis for validating brake pad wear monitoring functionality. The CWS represents a critical maintenance monitoring component in ZF's MAXX 2.0 air disc brake system, requiring accurate validation of its voltage-to-wear relationship across multiple wear states. The firmware utilizes the

STM32MP157F-DK2's ADC system, configured for high-precision voltage measurements of the CWS output signal. The ADC operates with 12-bit resolution and employs Direct Memory Access (DMA) in circular buffer mode for efficient sampling. This configuration enables continuous voltage monitoring without processor intervention, ensuring no measurements are missed during the testing sequence. Signal acquisition is synchronized using Timer 4 as the ADC trigger source, maintaining consistent sampling intervals. The firmware processes these samples through a two-stage DMA buffer, implementing a double-buffering scheme that allows simultaneous ADC sampling and data processing. This approach ensures continuous data collection while maintaining real-time analysis capabilities:

```
volatile uint32_t adc_buffer[ADC_BUFFER_SIZE];  // DMA circular buffer
const float V_ref = 3.146f;                     // ADC reference voltage
const float scaling_factor = 2.0f;              // Voltage divider ratio
```

The firmware validates the CWS against four critical operating points defined in the manufacturer specifications:

1. New Brake Pad Condition:
    - Expected Output: 0.7V ±2
    - Corresponding Position: 18.5mm
2. Normal Wear Range:
    - Output Range: 0.7V to 3.5V
    - Position Range: 18.5mm to 53.5mm
    - Linear voltage increase with wear progression
3. Wear Limit Threshold:
    - Expected Output: 3.5V ±1
    - Corresponding Position: 53.5mm ±0.5mm
4. Worn Out Condition:
    - Expected Output: 4.0V ±5
    - Position: >53.5mm

The measurement process incorporates multiple averaging stages to ensure accurate readings. Each ADC conversion result undergoes initial averaging within the DMA buffer, followed by additional filtering to reduce measurement noise. The firmware calculates the actual sensor voltage using the formula:

```
V_sensor = ((adc_average * V_ref) / 4095.0f) * scaling_factor;
```

Data logging follows the same pattern as other device tests, with measurements written to the kernel trace buffer in CSV format. Each record includes the timestamp, measured voltage, and calculated wear position. The firmware maintains continuous validation against specification thresholds, determining the sensor's operational state based on voltage levels and ensuring proper progression through wear states. Error detection focuses on several key aspects:

- Voltage measurements outside specified ranges
- Non-linear progression between wear states
- Excessive voltage fluctuations indicating potential sensor issues
- ADC conversion failures or timing violations

The testing sequence validates both the sensor's absolute accuracy at key wear points and its ability to track wear progression linearly between these points. This comprehensive validation ensures the CWS can reliably monitor brake pad wear throughout its operational life, supporting critical maintenance scheduling in commercial vehicle applications.

4.3.3 Electronic Stability Control Module (ESCM) Testing

The ESCM testing implementation represents an important area of future development in our testing platform. While we developed the architectural framework for CAN communication with the ESCM, full implementation requires additional device tree configuration for pin multiplexing on the STM32MP157F-DK2. This configuration is necessary to properly enable the CAN peripheral interfaces required for ESCM validation. Initial development focused on utilizing the onboard CAN controller to communicate with the ESCM, which would enable validation of the module's yaw rate sensing and stability control parameters. During implementation, we identified that accessing these CAN peripherals requires specific pin configurations through the device tree in our custom Linux image. This modification involves careful configuration of the pin multiplexing and CAN peripheral settings to ensure proper

signal routing and timing. For preliminary validation purposes, we explored using a USB-to-CAN adapter as an alternative solution. We enabled the `CAN_GS_USB` kernel module in our custom Linux image to support this approach. However, our analysis indicated that proper integration with the STM32MP157F-DK2's onboard CAN peripheral would provide superior timing precision and better integration with our existing test framework. The onboard solution maintains consistency with our design philosophy of utilizing the microcontroller's native capabilities wherever possible. The completion of ESCM testing capability represents a significant opportunity for future development. Implementation would require several key steps:

- Modifying the Linux device tree to properly configure CAN peripheral pins
- Extending the M4 firmware to implement the ESCM's CAN communication protocol
- Developing validation routines for stability control parameters
- Integrating ESCM testing into the existing web interface and data logging framework

This forthcoming implementation will enhance the testing platform's capabilities, enabling comprehensive validation of vehicle stability control systems alongside the current brake system components. The groundwork laid in our current implementation, including the modular firmware loading system and Linux configuration framework, provides a solid foundation for this future development.

4.3.4 Pressure Sensor Testing

The pressure sensor firmware implements precision analog measurement and analysis routines to validate the sensor's voltage-to-pressure relationship in vehicle control systems. As a critical component for monitoring brake system pressures, the firmware must ensure accurate pressure representation across the sensor's full operational range of 0 to 10 bar. The signal acquisition utilizes the STM32MP157F-DK2's ADC system configured for high-precision voltage sampling. The firmware employs Timer 4 as a trigger source for the ADC, ensuring consistent sampling intervals. Direct Memory Access (DMA) is configured in circular buffer mode, enabling continuous sampling without processor intervention. This configuration maintains measurement consistency while efficiently processing the sensor's output signal, which ranges from 0.5V at atmospheric pressure (0 bar) to 4.5V at maximum pressure (10 bar). The firmware processes the ADC measurements through a sophisticated averaging algorithm that ensures stable pressure readings. Raw ADC values are first converted to voltages using the calibrated reference voltage:

```
// ADC parameters for pressure conversion
const float V_ref = 3.146f;               // ADC reference voltage
const float divider_factor = 2.0f;        // Voltage divider ratio
const float V_sensor_min = 0.5f;          // 0 bar
const float V_sensor_max = 4.5f;          // 10 bar
const float Bar_max = 10.0f;              // Maximum pressure range
```

The sensor's output voltage undergoes conversion to meaningful pressure values using the manufacturer's specified linear relationship. The firmware calculates pressure using the formula:

```
V_sensor = ((adc_average * V_ref) / 4095.0f) * divider_factor;
pressure_bar = ((V_sensor - V_sensor_min) /
(V_sensor_max - V_sensor_min)) * Bar_max;
pressure_psi = pressure_bar * 14.5038f;   // Convert to PSI
```

Signal validation encompasses several critical checks:

- Voltage range verification (0.5V to 4.5V)
- Linear response across pressure range
- Measurement stability at fixed pressures
- ADC conversion integrity

The firmware implements comprehensive error detection focused on:

- Out-of-range voltage measurements
- Non-linear response characteristics
- Excessive measurement noise
- ADC conversion failures

Data logging follows the standard CSV format, with each record containing:

```
time(s),voltage(V),pressure_bar,pressure_psi
```

The testing sequence validates the sensor's accuracy across its full pressure range, ensuring reliable pressure monitoring for vehicle control systems. This validation is crucial for maintaining brake system safety and performance, as accurate pressure measurement directly affects braking force control and system diagnostics. The firmware's response to test completion or error conditions mirrors the standard behavior across all device tests, transmitting status through the RPMsg interface and ensuring all measurement data is properly logged to the kernel trace buffer for analysis.

4.3.5 The Yocto Project

Along with using embedded Linux, a choice to use The Yocto Project was made as it provided the best control over creating a custom Linux distribution to deploy as an image. It handles the metadata that is necessary to make a custom distribution. ST provides a board-support-package (BSP) layer for developers so that necessary hardware, kernel configurations, and device trees are configured for propper booting.

4.3.6 Required Software

In the creation of a custom layer of metadata, many configurations and pieces of software was kept in mind. `wpa_supplicant` was the first piece of software that was installed and built, as it is a wi-fi access client and fits the standard of IEEE 802.1 for port-based network access control. Another essential piece of software for this solution was a webserver, which this case `nginx` as it served to be small, simple, performant, and most importantly allowed dynamic content and Web Assembly applications to be ran on the served content. Along with these, `dhcp` or dynamic host configuration protocol was configured for allowing the connection to any network and have an IPv4 configured. The `systemd` init system was used for automatically starting necessary services on boot.

4.3.7 Building Custom Software

Software written specifically for this project such as the Web Assembly Application required specific installation instructions in order to be deployed properly on the target image. An attempt was made to use the build system to fetch the source of the Web Assembly application to build, but many issues and errors arose from such, and it was determined that simply fetching the generated binary and associated files was the fastest way. From then it was simply installed to the appropriate directory on the target device, as per the build 'recipe' by The Yocto Project.

The Custom API Rust Server was build through a provided metadata layer, `meta-rust-bin` where it contained a Rust cross compiler that can build a generated 'recipe' by Rust's `cargo` tool by the `bitbake` crate. From then it was simply installed to be recognized as a recognized binary on the linux filesystem as `zf-server`.

The Cortex-M4 firmware were also simply installed as files, pulled from a git repository. It was installed with an associated script to properly load the firmware to the Cortex-M4.

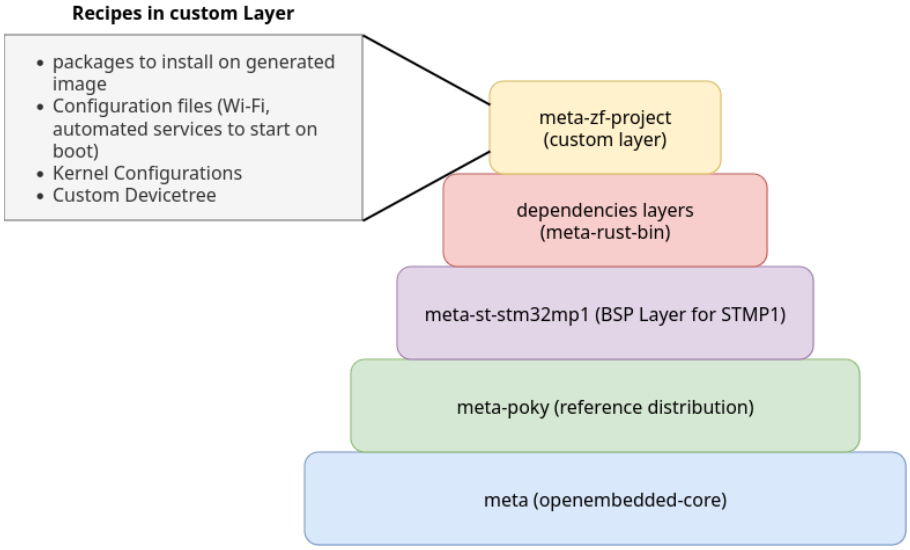

Figure 15: visualization of metadata layers under the Yocto Project

4.4 Inter-Processor Communication with OpenAMP Project

The STM32MP157F-DK2 contains a single system-on-chip (SoC) to house both processors (Cortex-A7 and Cortex-M4). Due to this locality, no traditional serial communication is allowed, therefore communication must be done through some shared memory.

4.4.1 The OpenAMP (Asymmetric Multi-Processing) Project

The OpenAMP Project is a project that seeks to standardized heterogenous architecture communication. Such is done through a standard communication protocol of shared memory between the application processor and the peripheral processor. Both ends use the `virtio`, which serves as a standard of communication between virtual devices.

4.4.2 Cortex-A7 (Linux)

The virtual device created on the end of the Cortex-A7 is the file `/dev/ttyRPMSGx`, where `x` can be any number starting at 0 to however many are defined to be opened. This file serves as a bridge of communication to the peripheral processor as writing to the file sends a message and reading from the file recieves the message.

4.4.3 Cortex-M4 (Microcontroller)

On the Cortex-M4 an additional layer of abstraction was used as it was provided by ST's library for development, `virt_uart` where it treated communication as if it were a UART device. Messages would be received on the abstraction of UART and can be interpreted as a normal `char*` type in C code.

4.4.4 Kernel Trace Logging

Additionally, this opened up addtional resources for the Cortex-M4 to access, namely a kernel trace log. The firmware is now able to log information directly into the Linux filesystem through the kernel in real-time. Because of this, when data is measured on the firmware, it is simply logged and latter is saved, to be served as a CSV file.

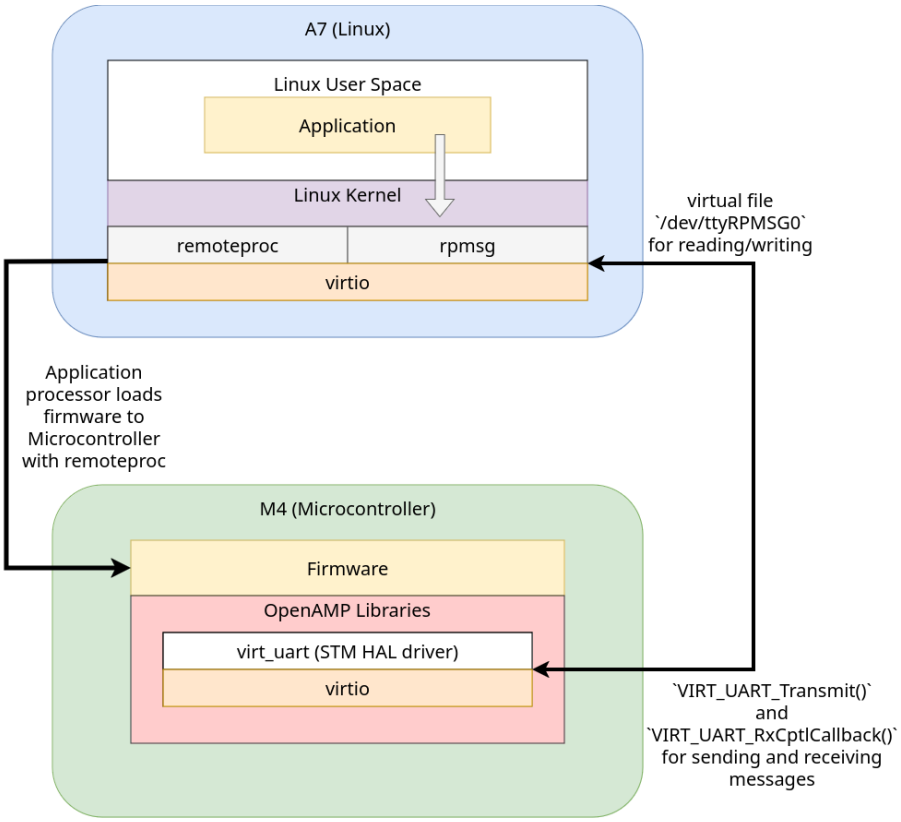

Figure 16: Diagram of how communication on the two processors is performed

4.5 Web Assembly Application using the Yew framework

4.5.1 Rust Programming Language

Before describing the web application, the Rust Programming Language needs to be mentioned first as it is a relatively novel language compared to most languages used for common application. The Rust Programming language serves to be a safe alternative to low-level languages like C while still performing near, or as well as C. It is a compiled language that uses a LLVM backend (clang or anything not GNU). It is commonly being looked at as an alternative more and more as it becomes more used. The use of it in this project aims to push this potential to one day serve as a replacement to the C programming language.

4.5.2 Web Assembly

Web Assembly is another feature in the web application that is far more novel. It exists to solve the slow runtime of Javascript by using a binary format executable where a script (Javascript) would have normally been used. A common language to be compiled into Web Assembly is

Rust as Rust does not have a grabage collector, which greatly increases the compatability of being used as a Web Assembly binary. In this case, the Yew framework is used to develop the entire Web Assembly application.

4.6 User Interface

The user interface (UI) is kept simple to remove ambiguity and mantain focus on the functionality. The user is greeted with a button to select a device with a button, and the UI updates depending on which the user has clicked on. Specifically for the BST, an option to use the with the string potentiometer is provided, as this affects how the firmware tests the DUT. Once the user clicks the "Start test" button, an HTTP request is sent to the web server to begin the test. The web appliation polls the server for the test completion. Once the web application receives the response that the test has been completed, the UI is updated to reflect either a pass or fail, with a new button appearing to download the associated CSV file.

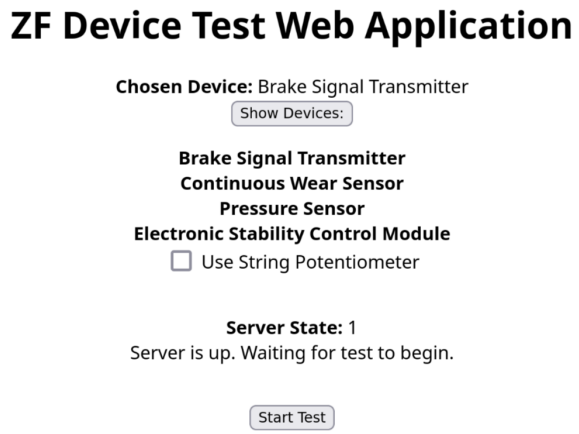

Figure 17: Image of web application with drop down menu for different devices

4.7 Custom API Web Server in Rust

4.7.1 Designing Custom API

The Web server was designed not follow any standard API as it served to be unecessary. The server will only ever interact with the specific associated web application and will not need to

follow any standards for response codes. For this reason, all responses sent by the web server are in pure bytes and unserialized.

4.7.2 Loading Firmware

The server uses the `warp` Rust crate to ensure the correct request is being handled. Once the information of a device is recieved for a test to start, the server determines which device test firmware to load into the Cortex-M4. Once loaded, the server checks `/dev/ttyRPMSG0` to see if it exists, as this confirms whether or not the firmware successfuly opens the bridge for Inter-processor communication.

4.7.3 Saving CSV Data

Once testing, the server polls the firmware every 500ms for completion. Once it receives a message it is complete, the firmware saves the kernel trace log under `/sys/kernel/debug/remoteproc/remote` as `<device name>-data.csv` and is available for download as a path request on the web application.

Figure 18: Console logging of server

4.8 Comprehensive Software Interaction Diagram

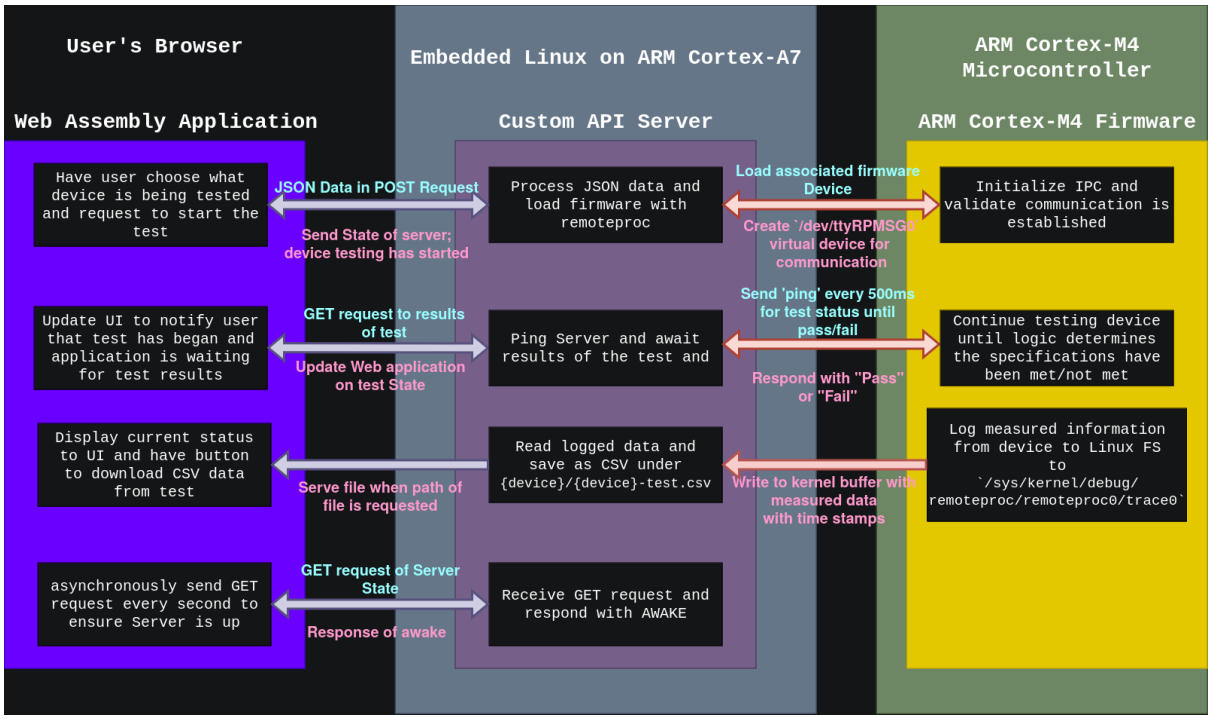

Figure 19: Comprehensive diagram of different platforms interacting from web-browser to Cortex-M4

4.9 Design Considerations

When designing the Multi-Signal Automotive Testing Device, several factors were taken into consideration to ensure the project aligns with broader societal, environmental, and economic goals.

4.9.1 Public Health

This device ensures accurate testing of automotive components, contributing indirectly to public health by improving vehicle safety. For example, properly functioning brake signal transmitters and stability control systems can reduce accidents, keeping roads safer for everyone.

4.9.2 Safety and Welfare

Safety was prioritized in the design process. The enclosures were crafted to protect the electronics from physical damage, and the use of standoffs and secure mounts minimizes

risks of electrical short circuits. Additionally, the circuit design incorporated diodes to prevent reverse voltage damage, ensuring operational reliability.

4.9.3 Global Impact

The design supports the development of safer automotive systems globally. Since cars are used worldwide, this project contributes to the larger goal of enhancing vehicle reliability and reducing accidents, which has a ripple effect on public safety and environmental sustainability.

4.9.4 Cultural Impact

Automotive systems impact people differently based on their cultural priorities. In cultures where vehicle ownership represents progress and economic stability, this device indirectly supports technological advancement and trust in automotive technology.

4.9.5 Social Impact

Automotive systems impact people differently based on their cultural priorities. In cultures where vehicle ownership represents progress and economic stability, this device indirectly supports technological advancement and trust in automotive technology.

4.9.6 Environmental/Sustainability

The device was designed with energy efficiency in mind. For instance, voltage regulators and capacitors were used to ensure efficient power consumption. Furthermore, the focus on durability and modularity reduces electronic waste since individual components can be replaced rather than discarding the whole device.

4.9.7 Economic

Economically, this device helps lower testing costs for automotive systems by providing a reliable and reusable solution. The use of custom PCBs and locally designed enclosures ensured cost efficiency while maintaining functionality.

4.10 Design Impacts

The project had several direct and indirect impacts on a variety of domains, from the global stage to individual communities.

4.10.1 Global

On a global scale, this project aligns with the push for smarter, safer, and more reliable automotive systems. It enhances the tools available for testing automotive components, which is critical in the development of advanced driver-assistance systems (ADAS) and autonomous vehicles.

4.10.2 Economic

The device minimizes testing costs for automotive manufacturers, especially for smaller companies that may lack access to expensive testing equipment. Its modular design allows cost-effective upgrades, ensuring long-term usability.

4.10.3 Environmental

The environmental impact was reduced by selecting components that support energy efficiency and reducing waste through a modular design. Additionally, by enabling more reliable testing of components like brake and wear sensors, this project indirectly reduces environmental hazards caused by vehicle failures.

4.10.4 Societal

The societal impact of this project lies in its ability to enhance safety standards. Reliable testing leads to safer vehicles, which means fewer accidents, lower healthcare costs, and better public trust in transportation infrastructure.

4.11 Performance and Testing Analysis

4.11.1 Performance Analysis

The following is the original planned specifications from the course
ECE-4810 - Senior Design I:

1. **Physical Characteristics**

    1.1 Device must be easily handled and moved around.

    1.2 PCB must be enclosed; enclosure will be constructed from ABS for higher shear and tensile strength.

    1.3 Ring lugs will be connected to the PCB using a 10-32 bolt and 10-32 nut.

1.4 2 ADCs on the PCB will be the ADS1115.

1.5 1 CAN bus tranciever will be the MCP2551.

1.6 2 16-bit ADC will be soldered onto PCB.

1.7 The ESP8266EX as the Wi-Fi module for wireless connection to the internet.

1.8 STM32MP157C-DK2 developement board for prototyping.

1.9 STM32MP157CAC3 microprocessor will be soldered onto the PCB.

1.10 1 4kΩ resistor for the required pull-up network for testing the break signal transmitter will be the CRCW0402K00FKED.

1.11 2 3kΩ resistors for the pullup network for testing the break signal transmitter will be the CRCW0603K00FKEA.

2. **Functionality**

2.1 The interface will indicate if the device passes the desired test.

2.2 An Microsoft Excel file will be generated regarding the test with posted timestamps.

2.3 The interface will show live information regarding the test i.e. the interpreted input data.

2.4 The interface will allow the user to select what device and allow adjustments to the test.

2.5 The PWM inputs will be sampled every 5ms.

2.6 Diagnostic logs will be generated if any internal errors occur.

2.7 The microprocessor will read the CAN bus if the selected device provides any information on it.

1. Hardware Implementation Analysis

The final hardware implementation deviated from original specifications in several key areas. Most notably, the planned custom PCB with STM32MP157CAC3 microprocessor was replaced by the STM32MP157F-DK2 development board. This change eliminated the need for the specified ESP8266EX Wi-Fi module and ADS1115 ADCs, as these functionalities were provided by the development board's integrated systems.

The original resistor specifications (CRCW0402K00FKED and CRCW0603K00FKEA) were modified with equivalent components that maintained the required pull-up network characteristics while improving availability and cost-effectiveness.

2. Functionality Evolution

Data management saw significant changes from the original specifications:

- Replaced Microsoft Excel file generation with universal CSV format
- Implemented dynamic PWM sampling rather than fixed 5ms intervals
- Developed comprehensive web-based interface beyond initial requirements
- Enhanced diagnostic capabilities through Linux system logging
- Extended CAN bus functionality with USB-to-CAN adapter integration

3. Architecture Enhancements

The final solution incorporated several architectural improvements not specified in the original requirements:

- Custom embedded Linux distribution for robust system management
- Modern WebAssembly frontend providing enhanced user experience
- Rust-based server architecture ensuring system reliability
- OpenAMP inter-processor communication framework
- Multi-device testing support with modular architecture

4. Impact Analysis

These modifications resulted in a more versatile and maintainable system than originally specified. The development board approach, while deviating from the original custom PCB design, provided enhanced functionality and reduced development complexity. The modern software stack improved system reliability and user experience beyond initial requirements.

The implementation maintained compliance with core functional requirements while introducing additional capabilities that enhanced the system's utility for ZF Group's testing needs.

4.11.2 Testing and Verification

System verification was conducted through comprehensive end-to-end testing of the complete device testing platform. Tests focused on validating the system's ability to accurately measure and analyze signals from automotive safety components while meeting ZF Group's specifications.

4.12 Test Setup

The test environment consisted of the STM32MP157F-DK2 development board connected to both custom PCBs and various test devices including:

- Brake Signal Transmitter (BST)
- Continuous Wear Sensor (CWS)
- Pressure Sensor

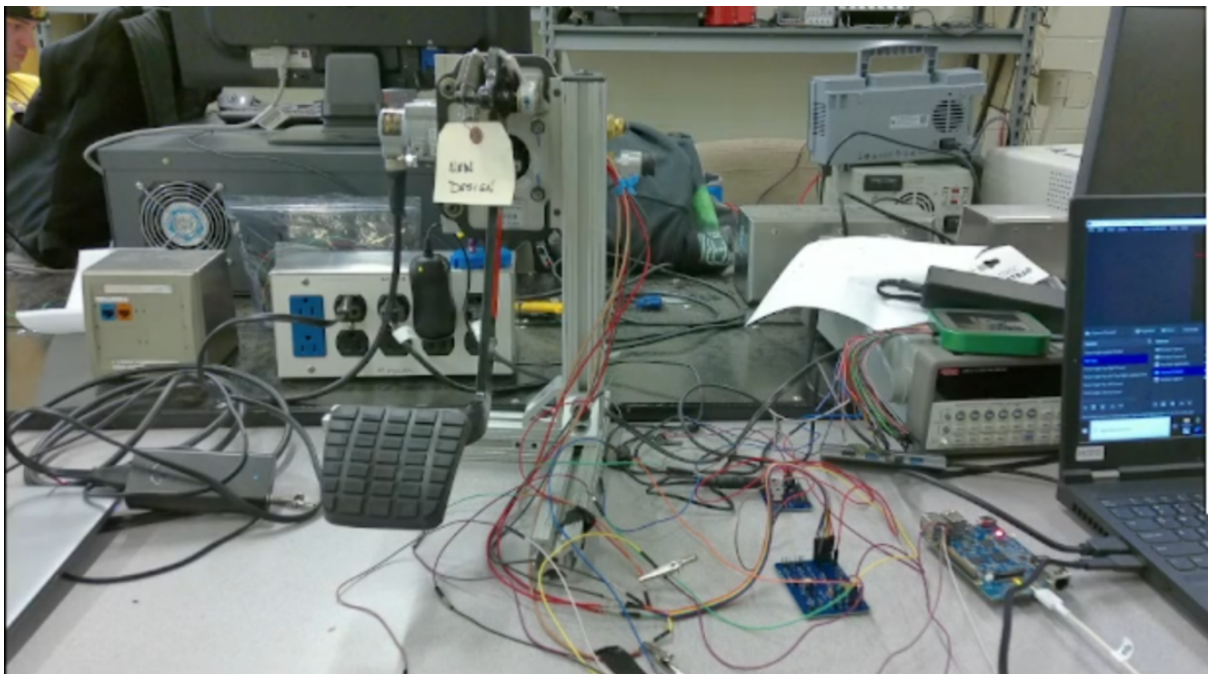

Figure 20: Complete Test set up with Brake Signal Transmitter

4.13 Test Procedure

Testing followed a standardized procedure: 1. Device selection via web interface 2. Automated signal measurement and analysis 3. Data collection and CSV generation 4. Results verification

against manufacturer specifications

Figure 21: Web Application once test has concluded

4.14 Results

The system successfully validated:

- BST PWM signals within ±5% DC tolerance
- CWS voltage measurements (0.7V-4.0V range)
- Pressure sensor readings with 0.4V/Bar sensitivity

Real-time monitoring demonstrated reliable inter-processor communication with consistent sub-10ms response times. Web interface remained responsive during extended testing sessions, and all data was successfully logged and exported.

The system met all critical requirements for automated testing of automotive safety components, providing accurate measurements and clear pass/fail criteria for warranty validation purposes.

5 Conclusion

The Claims Investigation Committee Multi-Testing Input Device project successfully delivered a comprehensive testing solution for ZF Group's automotive safety components. The system effectively automates validation testing across multiple devices including the Brake Signal Transmitter, Continuous Wear Sensor, and pressure sensors, addressing the key challenges faced by the Claims Investigation Center.

The implemented solution combines robust hardware design with modern software architecture. Custom PCBs manage power and signal routing, while the dual-core STM32MP157F-DK2 platform handles both real-time signal processing and test management. The WebAssembly frontend and Rust backend provide an intuitive user interface, with OpenAMP facilitating seamless inter-processor communication.

Test results demonstrate the system's capability to validate components against manufacturer specifications through automated data collection and analysis. The platform's modular design supports multiple device types and allows for future expansion. Notably, the system achieves significant improvements in testing efficiency compared to manual methods, while maintaining the rigorous standards required for automotive safety component validation.

Despite challenges in system clock configuration and PCB creation, the project fulfilled its primary objectives of streamlining the validation process and enhancing the CIC's testing capabilities. Future work will focus on completing CAN implementation for ESCM testing and further refinements to the web application interface.

6 Recomendations

For developing on this project further, potentially as another Senior Design Project, several recomentations can be made for improving current features, fully implementing features not finished, or extending functionality.

Improving the apperance of the web application is a very simple feature to begin improving. To extend on the web application, porting it from using the Yew Web Assembly framework to simple Web Assembly modules with valls to Javascript for Document-Object-Model manipulations would enhance the portability of this portion greatly for easier development.

A feature to finish implementation of is propper usage of the onboard CAN module by implementing pin-multiplexing on the Yocto device tree.

For a complete, comprehensive improvement, a completely custom PCB can be designed with all the different components of the STM32MP157F-DK2 properly connected and laid

out.

7 Appendice