

Claims Investigation Committee Multi-Input Testing Device

ECE-4820: Electrical and Computer Engineering Design II

Dylan-Matthew Garza Daniel Baker Rohullah Sah

Department of Electrical and Computer Engineering
Western Michigan University

ZF Group
Auburn Hills, MI

Fall 2024



Faculty Advisor:
Dr. Janos Grantner

Sponsor Manager:
Patrick McNally

Claims Investigation Committee Multi-Input Testing Device

2024-12-02

1. Hello everyone
2. Today we are excited to present our ZF-sponsored senior design project
3. the Claims Investigation Committee Multi-Input Testing Device
4. This is brought to you by myself, Daniel Baker
5. Dylan Garza
6. and Rohullah Sah

Faculty Advisor:
Dr. Janos Grantner

Sponsor Manager:
Patrick McNally

Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
 - 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
 - 3 Verification
 - 4 Challenges
 - 5 Future Work
 - 6 Closing

Claims Investigation Committee Multi-Input Testing Dev

└ Introduction

└ Table of Contents

Who is ZF?

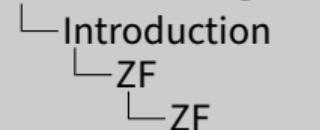
- Global technology company and Tier 1 automotive supplier
 - Provides advanced safety systems and vehicle control solutions
 - Partners with major OEMs: Daimler, Chrysler, Tesla, Waymo(Google), etc.
 - A leading innovator in commercial vehicle technology

North American Headquarters

- Project based at ZF Group's North American headquarters in Auburn Hills, MI
 - Specializes in commercial vehicle solutions
 - This facility is also home to the Claims Investigation Center



Claims Investigation Committee Multi-Input Testing De



1. So what is ZF (or zed-eff)
 2. ZF is a global leader in automotive technology
 3. specialize in advanced safety systems
 4. vehicle control solutions
 5. ZF is a German-based global tech company specializing in the automotive industry.
 6. They're a tier 1 supplier for major automakers like Daimler, Tesla, and Waymo
 7. Located in Auburn Hills, ZF North American HQ focuses on commercial vehicle solutions
 8. This is where our project is based
 9. This is also home to the Claims Investigation Center or CIC.

Claims Investigation Committee

What is the Claims Investigation Center?

CIC's Role in Our Project

```
graph TD; A[Claims Investigation Committee Multi-Input Testing Device] --> B[Introduction]; A --> C[Claims Investigation Committee]; B --> D[Claims Investigation Committee]; D --> E[Claims Investigation Committee]
```

2024-12-02

Motivation for the Project

Challenges with Current Testing Methods

- Testing on current brake system platform (mBSP) was built and industrialized specifically for that platform
 - Long lead time and significant cost to release and document
 - New platform components are not compatible with the current tester
 - Current tester is not capable of testing in prototype phase

Need for Improvement

- The Brake Signal Transmitter's (BST) implementation in Daimler's new platform intensifies urgency
 - High production volumes require efficient testing methods
 - Expanding product line increases testing complexity



Claims Investigation Committee Multi-Input Testing Device

└ Introduction

- Project Motivation
 - Motivation for the Project

1. The CIC currently relies on the mBSP ABS Component Tester to validate parts like
2. Electronic Control Units
3. Valves
4. Pressure sensors
5. Tester has key limitations
6. It's rigid
7. focus only on predefined components
8. unsuitable for new or prototype components
9. It's costly
10. upgrades require significant effort and manufacturer support
11. It's slow
12. manual setups increase testing time and risk human error
13. Upcoming release of ZF's Brake Signal Transmitter
14. Issues become critical
15. Launching in 2025
16. Will be used in Daimler
17. Highest volume commercial vehicle platform in North America

Our Solution

Multi-Input Testing Device

Addressed Challenges

U.S. House Committee on Ethics

-Introduction

└ Our Solution

└ Multi-Input Testing Device

Our solution, a multi-input testing device, directly supports CIC
it provides a unified testing platform
capable of handling multiple devices across various product lines
the device is flexible and adaptable
Designed to accommodate future components
allows us to change acceptance criteria without significant effort
Thirdly, our device automates testing.
Reducing manual set up
minimizing human error
improving accuracy

M-W Input Training Device

10.1002/anie.201907002

- Provides a unified testing platform for multiple devices
 - Flexible and agile to adapt to new product lines
 - Allows for prototype testing and validation
 - Automates data collection and analysis to reduce time and errors
 - Increases testing speed and accuracy
 - Simplifies validation process for warranty claims
 - Enhances capability to analyse field returns efficiently
 - Cost-effective solution

Key Devices Under Test (DUTs)

1. **Brake Signal Transmitter (BST)**
 - Primary focus - critical new component for 2025 production
 - Acts as the brain that reads how hard a driver presses the brake.
 2. **Continuous Wear Sensor (CWS)**
 - Works like a monitor for your brake pads and discs
 - Warns when brakes are wearing down using voltage
 3. **Pressure Sensor**
 - Continuously measures relative pressure in vehicle control systems
 4. **Electronic Stability Control Module (ESCM)**
 - Acts as a safety system that helps prevent skidding and rollovers
 - Monitors the vehicle's movement and intervenes to keep it stable

Our Multi-Input Testing Device supports several key components:

- Brake Signal Transmitter or BST
 - critical new component for 2025 production
 - interprets the driver brake pedal pressure
- Continuous Wear Sensor
 - Apart of new upcoming brake system platform (mBSP2)
 - device monitors brake pad and disc wear
 - provides warnings when brakes need maintenance
- Pressure sensor
 - Apart of previous component tester as well
 - measures pressure in vehicle control systems
 - included as part of BST design
- Electronic Stability Control Module
 - Prevents skidding and rollovers
 - monitors vehicle stability and intervenes if necessary
 - Test platform enhances CIC's abilities

Introduction
oooooo

Design and Implementation
●oooooooooooooooooooooooooooo

Verification
oo

Challenges
oo

Future Work
oo

Closing
ooo

Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification
- 4 Challenges
- 5 Future Work
- 6 Closing

2024-12-02

Claims Investigation Committee Multi-Input Testing Device

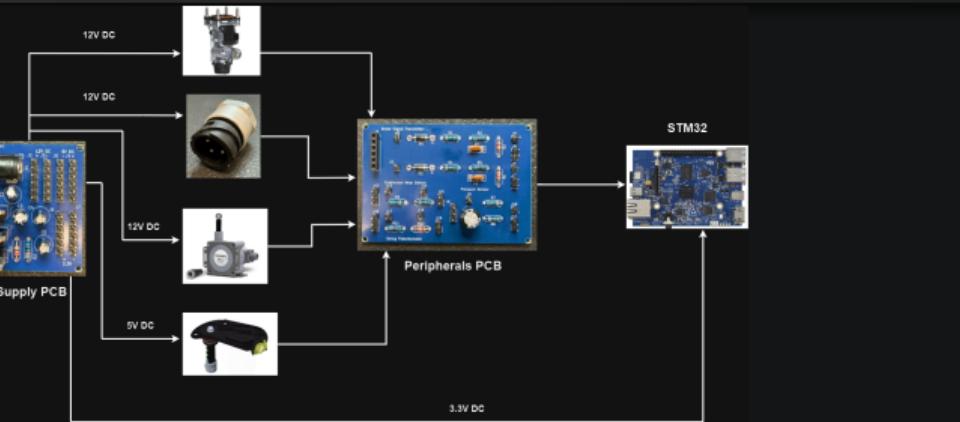
Design and Implementation

Table of Contents

Table of Contents

- Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- Verification
- Challenges
- Future Work
- Closing

Project Solution and Overview



What this project aims to accomplish:

1. Device Interfacing

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

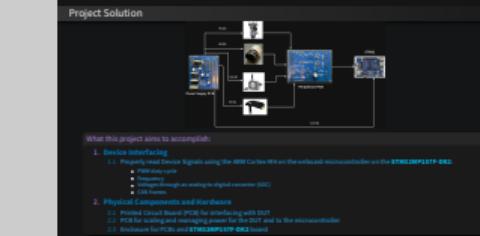
aims Investigation Committee Multi-Input Testing Dev

-Design and Implementation

└ Project Solution and Overview

Project Solution

using stm32mp157f-dk2, has, Arm Cortex-M4 for external signal handling and Arm Cortex-A7 as an application processor
Cortex-M4's primary role is reading signals from the device under test
a pwms duty cycle, its frequency, a voltage level through and analog to digital converter and CAN frames
processing this information is how the devices will be tested
interfacing with the devices under test will also require some hardware for proper signal reading
specifically for scaling and managing power and to ensure clean signals with minimal noise



The diagram illustrates the project timeline across three main phases:

- Introduction:** Represented by 5 circles.
- Design and Implementation:** Represented by 10 circles.
- Project Solution and Overview:** Represented by 5 circles.

Project Solution (cont.)

What this project aims to accomplish:

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

1. the software running is the core to the solution, as it aims to simplify the end user, the person running the test on the device
2. The Cortex-A7, the application processor will be running a CUSTOM built linux image
3. it will host a Custom web application in something called web assembly
4. and will interact with a custom webserver that will also handle communication to the Cortex-M4 to gather test data and have it be available for download as a CSV

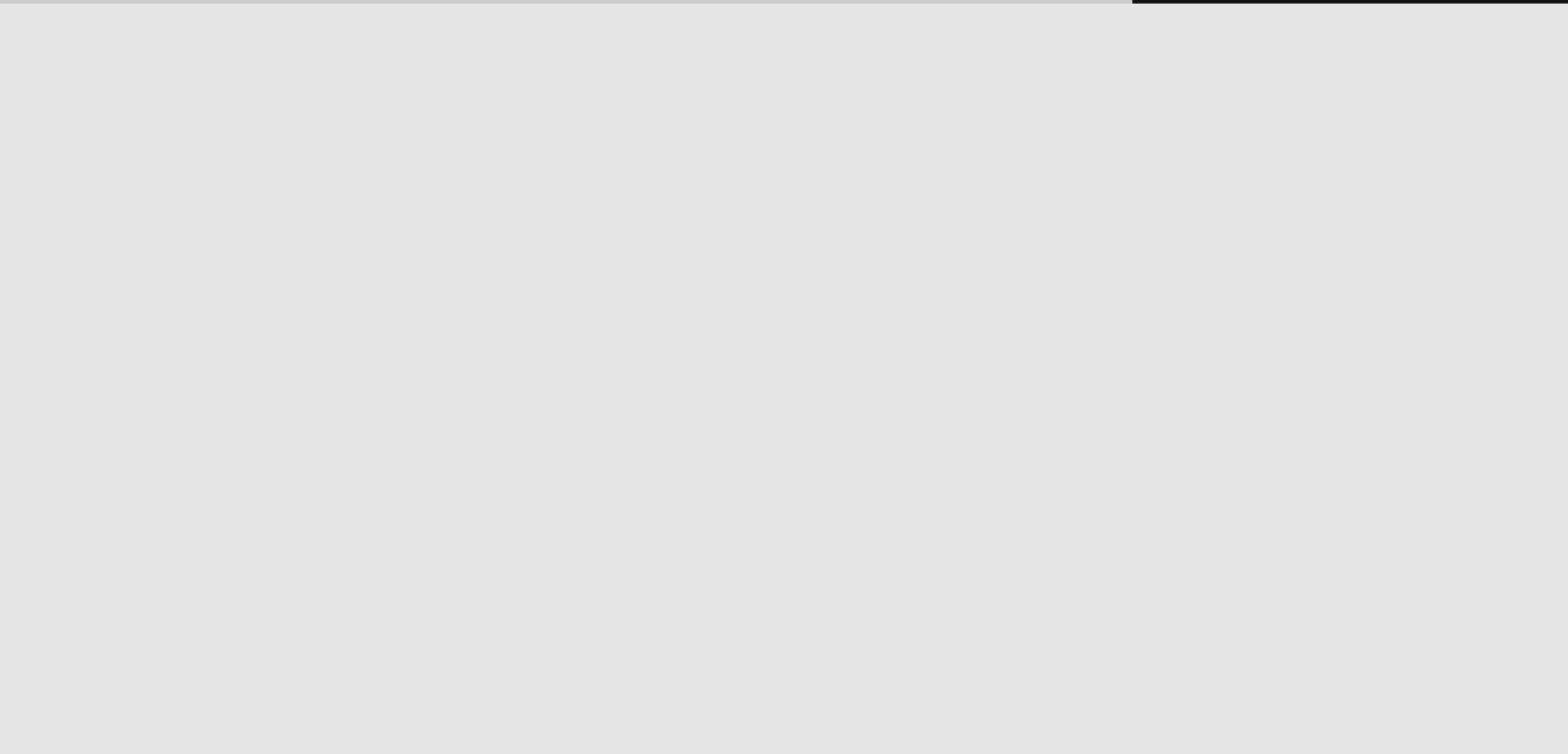
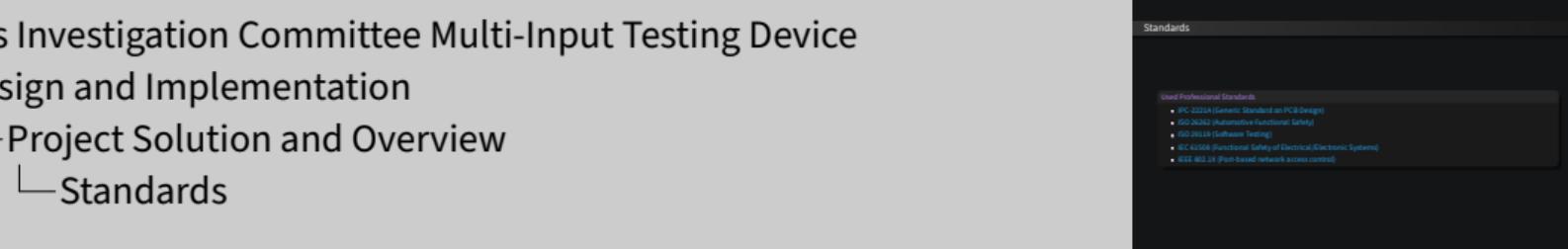
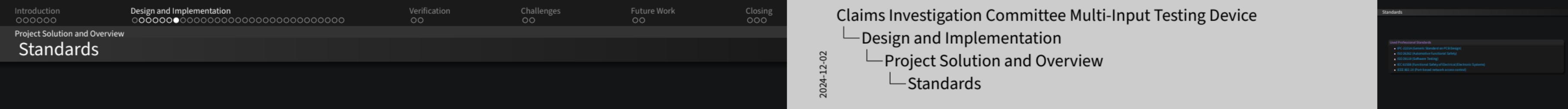
Claims Investigation Committee Multi-Input Testing

└ Design and Implementation

-02 Project Solution and Overview

24-11

The following is our budget to show our total costs. The more expensive items like the STM discovery boards and the continuous wear sensor were provided to us by our sponsor, but the rest was provided by us or was a donation



Introduction
oooooo

Design and Implementation
oooooooooooooooooooo

Verification
oo

Challenges
oo

Future Work
oo

Closing
ooo

Hardware Design
Table of Contents

- 1 Introduction**
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation**
 - Project Solution and Overview
 - Hardware Design**
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification**
- 4 Challenges**
- 5 Future Work**
- 6 Closing**

2024-12-02

Claims Investigation Committee Multi-Input Testing Device

- Design and Implementation**
 - Hardware Design**
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server

Table of Contents

- **Introduction**
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- **Design and Implementation**
 - Hardware Design**
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- **Verification**
- **Challenges**
- **Future Work**
- **Closing**

15 / 47

Hardware Design

Power Supply Schematic Design

The circuit diagram illustrates a multi-stage voltage regulation system. It starts with a 3.3V DC input source. This is fed into a first stage regulator, U1 (LM7812-TO220), which provides a 5V output. This 5V output is then used as the reference voltage for a second stage regulator, U2 (LM7805-TO220), which provides a 12V output. Finally, a third stage regulator, U3 (LM317L-TO-220), is used to provide a second 12V output. The circuit also includes several bypass capacitors (C1 through C5) and a power switch (J1) for the first stage.

- ng LM7805 (1A)
g LM317
ulator

Components

- 5, LM317
tors for
ersion.
oise filtration.
voltages for
/ DC (50µA).

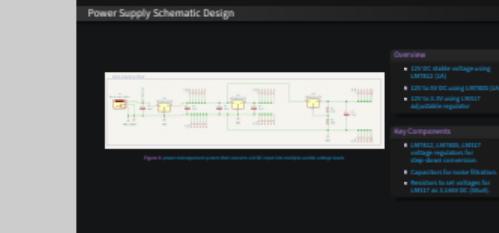
Investigation Committee Multi-Input Testing D

Design and Implementation

Hardware Design

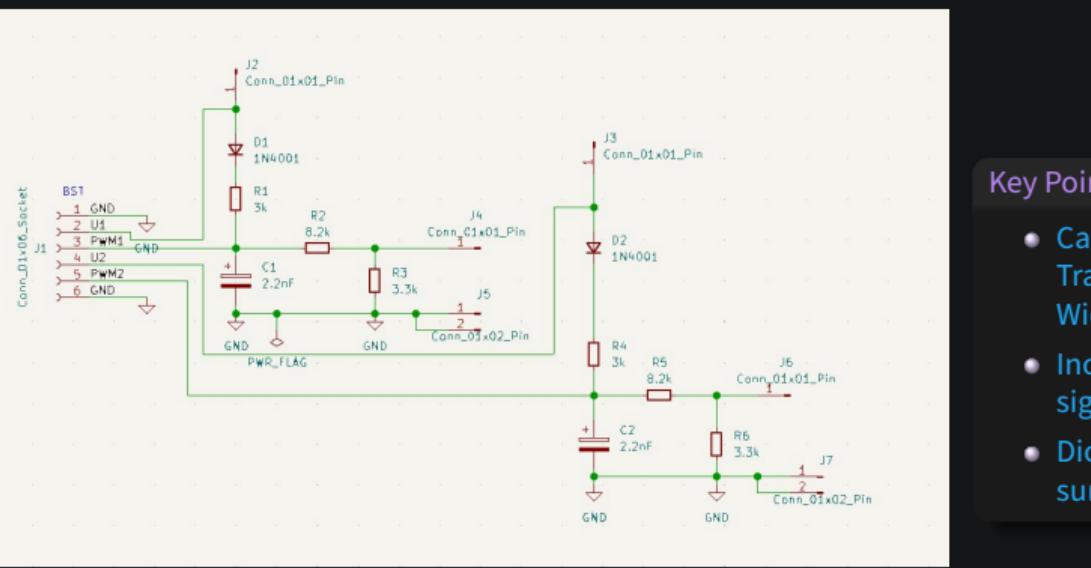
Power Supply Schematic Design

start with the power supply design. Here, the input to our system is through a DC Barrel Jack which supplies 12V DC, but the components like sensors and microcontrollers operate at lower voltages. We use three main voltage regulators: this one ensures the input is smooth and regulated to 12V. Over here, it steps down the 12V to 5V. and the last one is adjustable, steps down further to 3.146V using resistors to act as dividers and capacitors to filter out any electrical noise.



Hardware Design

Schematic Design - Brake Signal Transmitter



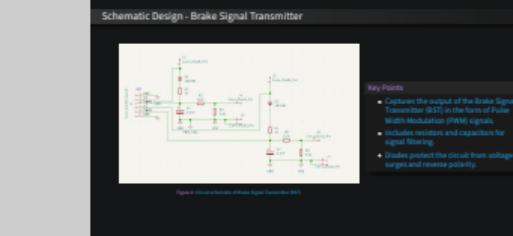
- out of the Brake Signal
in the form of Pulse
Width Modulation (PWM) signals.
and capacitors for
the circuit from voltage
polarity.

Claims Investigation Committee Multi-Input Testing I

└ Design and Implementation

└ Hardware Design

Schematic Design - Brake Signal Trans.



Now, let's move on to the Schematic of the Peripherals. This schematic design shows how the signals of the Brake Signal Transmitter are sent to the microcontroller. The sensor sends PWM signals (.7V – 4.5V) that represent how much force is applied to the brake. Our circuit uses resistors as voltage dividers and capacitors as noise filtering to ensure these signals are stable and within the microcontroller's safe range to read. Diodes act as a safeguard, preventing voltage spikes or reversed connections from damaging the microcontroller.

Peripheral Interface Schematic Diagram

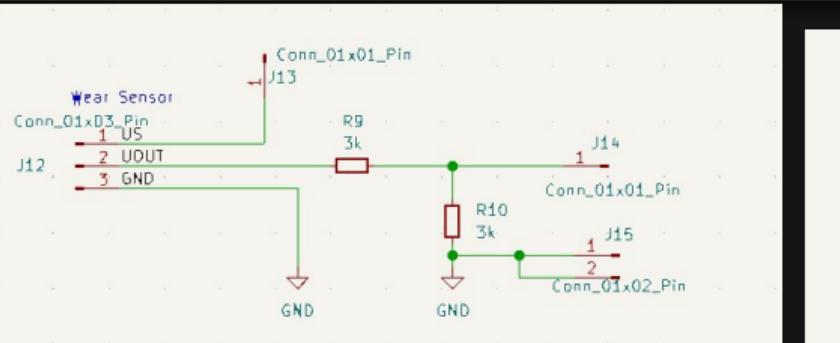


Figure 5: Continuous Wear Sensor Interface Schematic

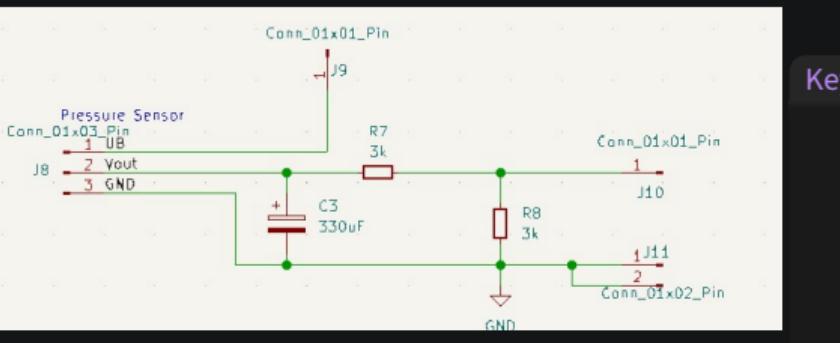
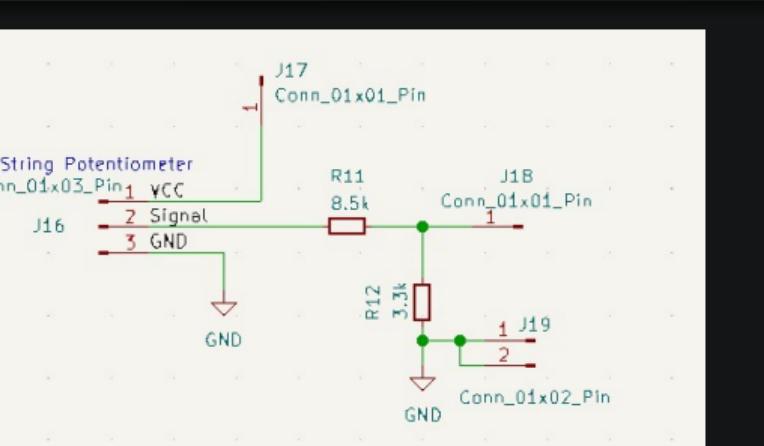


Figure 6: Pressure Sensor Interface Schematic

- Captures analog voltage signals to monitor brake wear and pressure sensor and displacement on the string potentiometer.
 - Uses voltage dividers for safe microcontroller input levels.
 - Uses capacitors to stabilize the output.

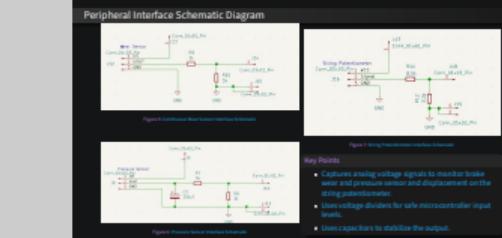
Claims Investigation Committee Multi-Input testing Document

└ Design and Implementation

2-02 └ Hardware Design

0024-1 Peripheral Interface Schematic Diagram

We have the schematic interface of the other three sensors: The Continuous Wear Sensor, which monitors the brake pad wear by outputting a small voltage signal (.7V-4V) | the Pressure Sensor which measures brake system pressure, outputting a voltage (.5V-4.5V) proportional to the pressure level. We have a capacitor to smooth out the noise in the signal | the String Potentiometer which measures the displacement of the brake pedal and converts it to a proportional voltage (0V-4.75V). For all three schematic designs, we have resistors, which condition the signal so the microcontroller can process it correctly.



Hardware Design

Printed Circuit Board Design

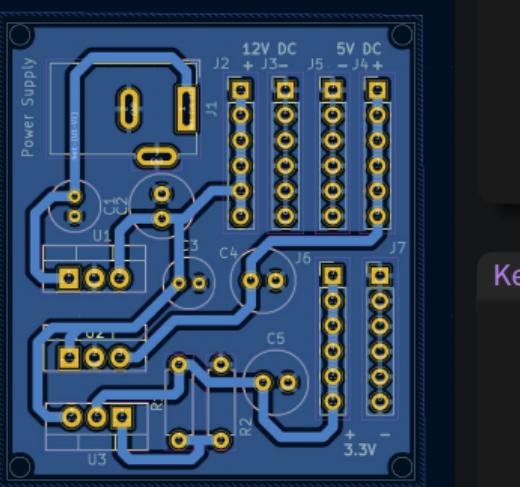


Figure 8: PCB for Power Management

Overview

- 12V DC
 - 5V DC
 - 3.3V DC

is designed based on the schematic with components such as voltage regulators (LM7812, LM7805, LM317), capacitors, and resistors.

Key Components

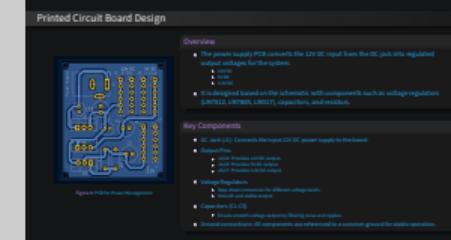
Claims Investigation Committee Multi-Input Testing Device

└ Design and Implementation

-02 └ Hardware Design

024-12

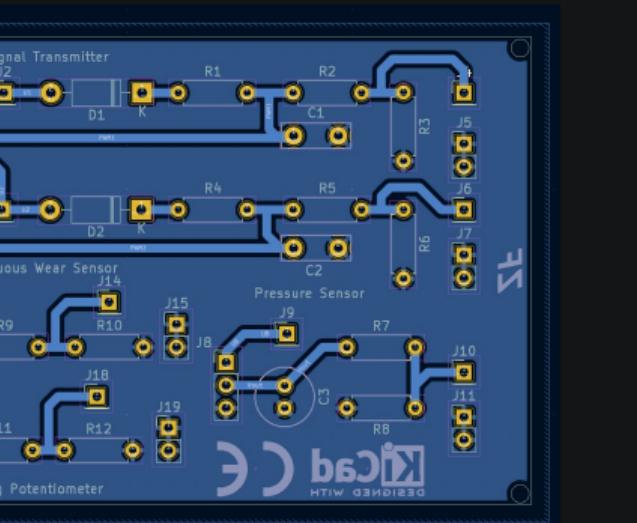
Here is the Printed Circuit Board design for our power supply. This board takes 12V DC input from a DC jack and converts it into the usable outputs. This is where we connect the 12V input DC Jack power supply (J1). We have the capacitors over here, to filter out noise and voltage ripples. These are the voltage regulators, to ensure the outputs are stable and consistent even if the input voltage fluctuates. These are the output pins. These ones provide 12V for devices requiring higher voltage. These provide a regulated 5V and the ones at the bottom provide a 3.146V keeping the resistors as voltage dividers. All components are referenced to a common ground.



Hardware Design
Peripherals Printed Circuit Board

Key Features

- **Input/Power Pins:**
 - Each DUT has a dedicated connector for input signals and a power signal.
 - J1: Inputs for BST (PWM1 and PWM2) | J2/J3: 12V Power Signals for BST (PWM1 and PWM2)
 - J8: Pressure sensor input | J9: 12V DC Power Signal
 - J12: Wear sensor input | J13: 5V DC Power Signal
 - J16: String Potentiometer input | J17: 12V DC Power signal
 - **Output Pins:**
 - Processed signals are sent to the microcontroller through the output pins.
 - J4/J5/J6/J7: BST processed signals
 - J10/J11: Pressure sensor output
 - J14/J15: Wear sensor output
 - J18/J19: String potentiometer output.
 - **Signal Conditioning:**
 - Resistors: Scale signals for safe microcontroller input.
 - Capacitors: Filter noise and stabilize signals.
 - Capacitors: Filter noise and stabilize signals.



- Claims Investigation Committee Multi-Input
- └ Design and Implementation
 - └ Hardware Design
 - └ Peripherals Printed Circuit Board

This is the Peripheral PCB Design. We have kept all the components of each sensor separate. This one over here is for BST, we have the Continuous Wear Sensor over here, the Pressure sensor on the right of the board, and the String Potentiometer at the bottom. Each Device Under Test has its own dedicated connector for both signal input and power supply pin. J1 pins capture the PWM signals from the Brake Signal Transmitter, while these pins (J2 and J3) provide the 12V power needed to operate it. Similarly, J8 reads the Pressure Sensor input, and J9 delivers the 12V power required for its operation. J12 connects the Continuous Wear Sensor input, with J13 supplying it with 5V power. Lastly, the String Potentiometer uses J16 for its signal input and receives its 12V power from J17. We have the corresponding electrical components (resistors, capacitors, diodes). The processed signals are sent through specific output pins, while we have a common ground



Fabricated PCB

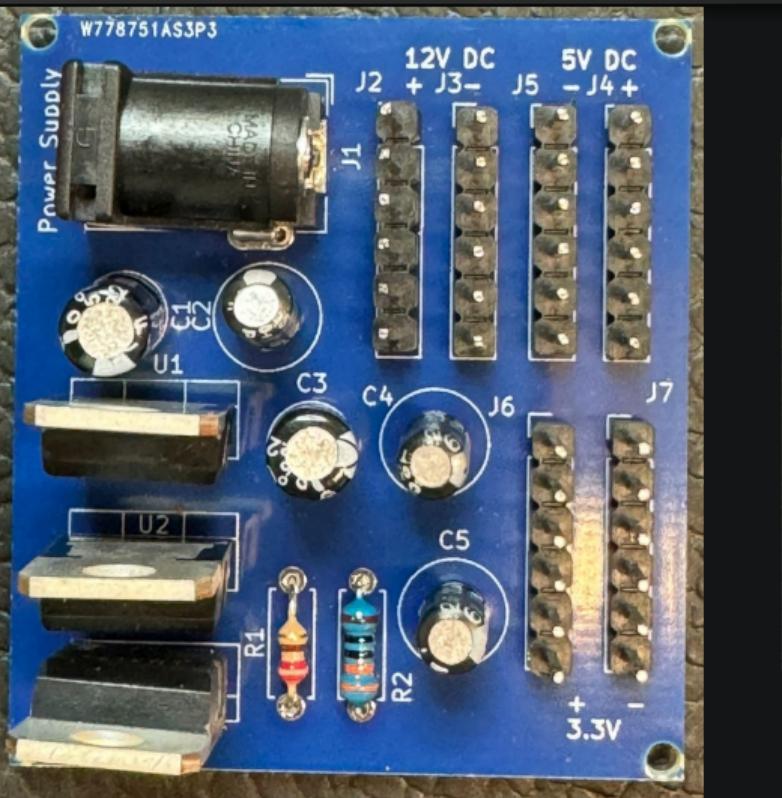


Figure 10: Power Management PCB



Figure 11: Peripheral Interface PCB

Claims Investigation Committee Multi-Input Testing Device

- └ Design and Implementation
- └ Hardware Design
- └ Fabricated PCB

Here is the fabricated and custom assembled PCB's. Power management is on the left, and on the right is the Peripheral PCB.



Enclosure Design

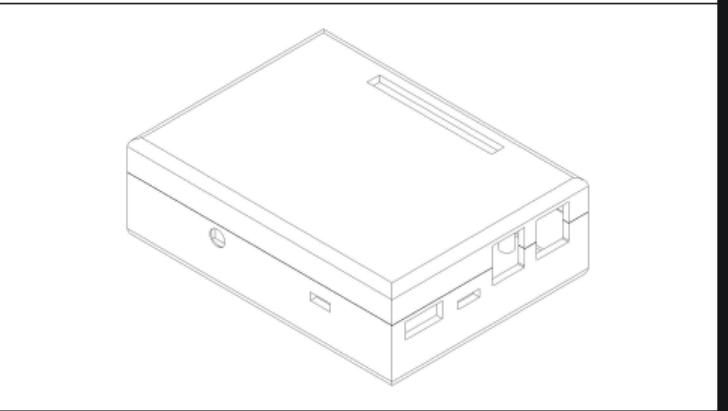
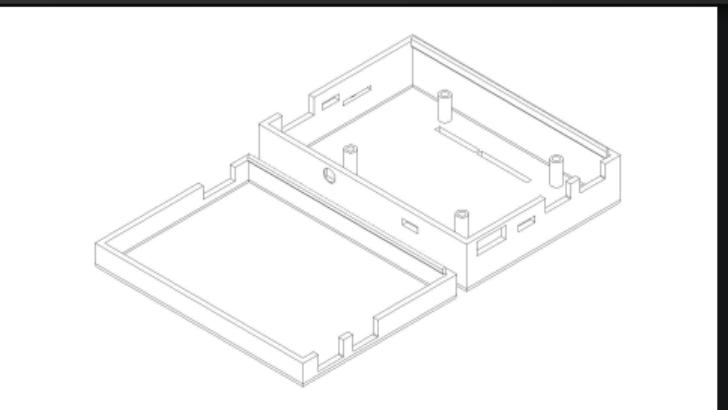


Figure 12: *Enclosure for STM32MP157F-DK2*

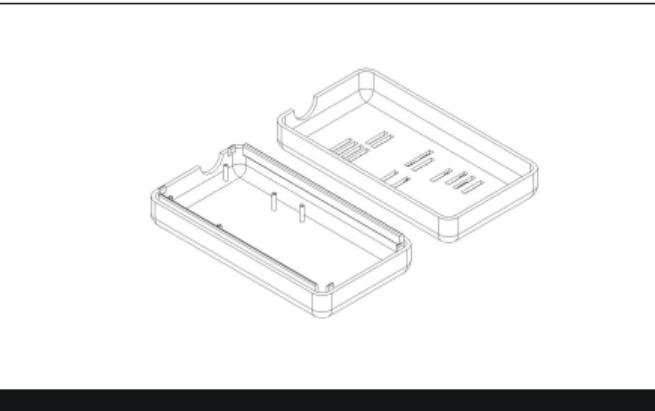
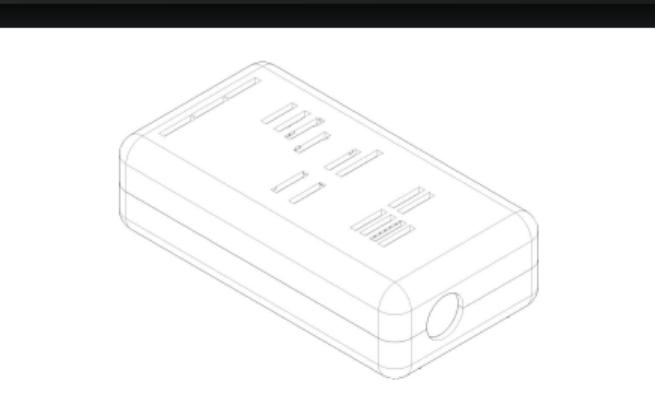
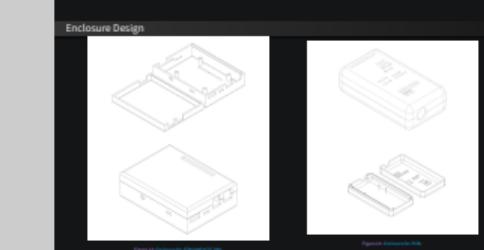


Figure 13: *Enclosure for PCBs*

Claims Investigation Committee Multi-Input Testing Device

- └ Design and Implementation
 - └ Hardware Design
 - └ Enclosure Design

Here, we have the enclosures for the combined PCBs which are the PS and Peri, and on the left is the enclosure for our microcontroller board. These enclosures are custom designed to securely house the PCBs, protect their components, and provide easy access for connections and maintenance. On these enclosures, we have included slots for the DC jack, open ports, and input/output pins ensuring straightforward and well-organized connections. The slots could also be used as ventilation to maintain stable temperature during testing. Inside, we have standoffs to hold the boards securely in place, preventing movement. The two-part assembly makes it easy to open these enclosures for any repairs, upgrades or debugging.



Introduction oooooo	Design and Implementation oooooooooooo●ooooooooooooooo	Verification oo	Challenges oo	Future Work oo	Closing ooo
Device Interfacing and Testing					
<h2>Table of Contents</h2>					
<p>1 Introduction</p> <ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test	<p>2024-12-02</p> <p>2 Design and Implementation</p> <ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server	<p>3 Verification</p>	<p>4 Challenges</p>	<p>5 Future Work</p>	<p>6 Closing</p>
<h2>Claims Investigation Committee Multi-Input Testing Device</h2>					
<ul style="list-style-type: none">└ Design and Implementation<ul style="list-style-type: none">└ Device Interfacing and Testing<ul style="list-style-type: none">└ Table of Contents					
<p>Table of Contents</p> <ul style="list-style-type: none">● Introduction<ul style="list-style-type: none">● Device Interfacing and Testing● Design and Implementation<ul style="list-style-type: none">● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server● Verification● Challenges● Future Work● Closing					

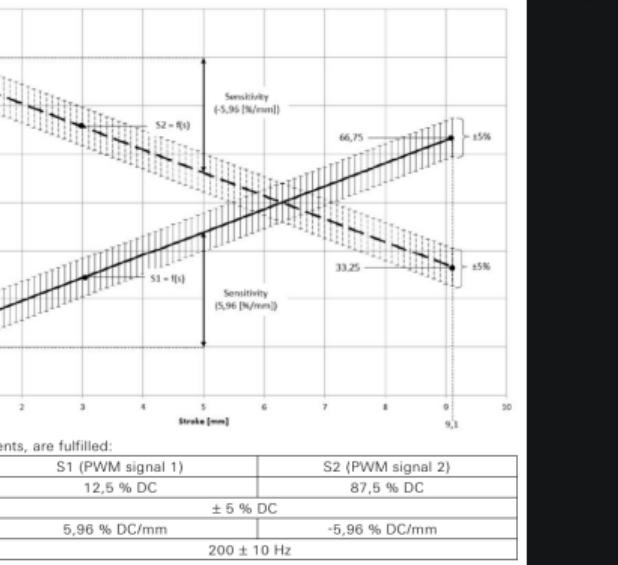
Firmware to Test Brake Signal Transmitter (BST)

Purpose

- Developed firmware on the onboard Cortex-M4 microcontroller to validate BST
 - Ensures brake actuation is accurate to distance moved by brake pedal
 - **Key Specifications:** Output range 1 mm to 9 mm, Sensitivity 5.96% DC/mm, Output signals PWM1 and PWM2 (S1 and S2)

Method

- **Input Capture:** Timers captures read two PWM signals from the BST
 - **ADC Reading:** Optional string potentiometer for direct analog voltage measurements via ADC
 - **Processing:** Calculates duty cycles, frequencies, and estimated stroke via timer interrupts
 - **Validation:** Compare measurements against expected values according to product specifications to verify BST accuracy
 - **Results:** Sends test results to the main processor for logging and user display



ms Investigation Committee Multi-Input Testing Devi

Design and Implementation

└ Device Interfacing and Testing

└ Firmware to Test Brake Signal Transmitter (

- For the BST, the Cortex-M4 receives signals in the form of pulse-width modulation, or PWM
This is a method where signals are turned on and off rapidly
2 key parameters define PWM
the frequency is how fast these signals are read
for BST specifically reads at 200 Hz
the pulse is repeated 200 cycles per second
The duty cycle of the pulse is what percentage the signal is "on"
so a duty cycle of 50
The code measures these signals to calculate the brake stroke
ensures it matches the sensitivity of 5.96
The test also offers option to use a string potentiometer
this provides an analog voltage proportional to the brake stroke
read by ADC on the cortex-M4

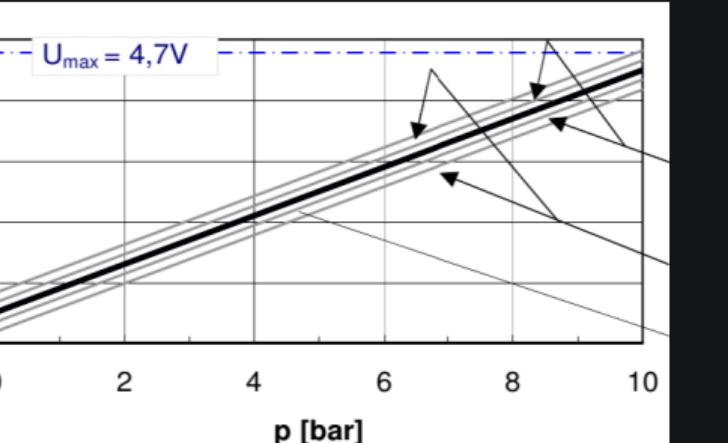
Firmware to Test Pressure Sensor

Purpose

- Developed firmware to validate Pressure Sensor readings on the Cortex-M4 microcontroller
 - Ensures accurate measurement of pressure when given for reliable vehicle control system purposes
 - **Key Specifications:** Output range 0.5V (0 bar) to 4.5 V (10 bar), Sensitivity 0.4 V/Bar, Voltage divider ratio 2:1

Method

1. **ADC Configuration:** Configured the ADC to read analog voltage from the Pressure Sensor using DMA for efficient data transfer and utilized a timer to trigger ADC conversions periodically
 2. **Pressure Calculation:** Mapped the measured voltage to pressure using a linear relationship given in the product specifications with the addition of converted pressure from bar to psi
 3. **Validation:** Compared calculated pressure against expected values based on product specifications with voltage tolerances to determine pass/fail status
 4. **Results:** Sent detailed test outcomes to the main processor for logging and user display



Claims Investigation Committee Multi-Input Testing Device

- └ Design and Implementation
 - └ Device Interfacing and Testing
 - └ Firmware to Test Pressure Sensor

Firmware to Test Pressure Sensor

Purpose

- Developed firmware to validate pressure sensor readings on the Cartesio 900 measurement device.
- Ensure accurate measurement of pressure when given for reliable vehicle control system purposes.
- Key Specifications: Output weight 0.01 g (0.0013 bar), linearity 0.4% F.S., voltage droop ratio 2.5.

Method

- Analog signal input from the pressure sensor, converted to digital using an ADC.
- Pressure calibration: Pressure sensor readings were taken using a known standard pressure source and compared with the calculated value from the sensor's data sheet.
- Linearity verification: A series of pressure measurements were taken across the range of the sensor, and the resulting output was plotted against the expected linear relationship.
- Accuracy validation: The measured output was compared against the expected output for different pressure levels to verify the overall accuracy of the sensor.

Figure 4: Pressure Sensor Calibration Data

1. Similarly, the Pressure Sensor outputs voltages proportional to pressure
 2. 0.5V for 0 bar to 4.5V for 10 bar
 3. The ADC reads these signals
 4. our firmware converts them to meaningful pressure values
 5. Both these tests incorporate thresholds to determine pass or fail outcome
 6. Validation in all test occurs on the Cortex-M4
 7. comparing calculated values with expected specifications.
 8. Results are then sent to the Cortex-A7
 9. logged and displayed for user

Introduction oooooo	Design and Implementation oooooooooooooooooooo●oooooooooooo	Verification oo	Challenges oo	Future Work oo	Closing ooo
Embedded Linux With Yocto Project					
<h2>Table of Contents</h2>					
<ul style="list-style-type: none">1 Introduction<ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test2 Design and Implementation<ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server3 Verification4 Challenges5 Future Work6 Closing					
2024-12-02					
<h2>Claims Investigation Committee Multi-Input Testing Device</h2>					
<ul style="list-style-type: none">└ Design and Implementation<ul style="list-style-type: none">└ Embedded Linux With Yocto Project<ul style="list-style-type: none">└ Table of Contents					
<small>Table of Contents</small>					
<ul style="list-style-type: none">● Introduction<ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test● Design and Implementation<ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server● Verification● Challenges● Future Work● Closing					

Embedded Linux and The Yocto Project

Embedded Linux

- Industry standard for embedded Operating system
- Rich ecosystem of open-source tools and software

The Yocto Project

- Collection of tools to build a **custom** embedded Linux distribution
- Fine-grain control of every aspect of deployed image

Claims Investigation Committee Multi-Input Testing Device

Design and Implementation

Embedded Linux With Yocto Project

Embedded Linux and The Yocto Project

1. due to time constrains i'll go over the contents of the following to slides
2. As i mentioned earlier, the application processor is running embedded linux
3. It gives a lot of benefits, namely its virtually the only option for any serious embedded operating system
4. as it gives a rich ecosystem of open-source tools and software
5. The Yocto Project was used to build a custom linux distribution
6. Since it provides fine grain control of the resulting image that will be used on the device

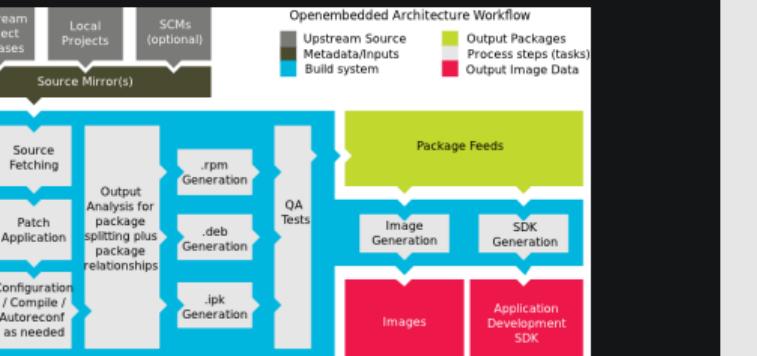
2024-12-02

Embedded Linux With Yocto Project

Using The Yocto Project to Build a Custom Distribution

What is the Yocto Project and why?

- Most popular set of tools for embedded Linux Development
 - Collection of OSS tools to make a custom Linux distribution
 - Independent of target architecture
 - **bitbake** build tool handles **metadata**
 - **MetaData** can be in the form of
 - software build/patch instructions
 - configuration files for software
 - **MetaData** organized in its **Layer Model**



Design and Implementation

Embedded Linux With Yocto Project

Using The Yocto Project to Build a Custom Distribution



Custom Linux Image for the STM32MP157F-DK2

What is used in the deployed image?

- ST's BSP (board support package) layer provides metadata
 - Hardware drivers
 - Kernel Configurations
 - Devicetree
- Custom layer **meta-zf-project**
 - **nginx** (webserver), **wpa_supplicant** (Wi-Fi access client/ IEEE 802.1X supplicant)
 - recipes for custom applications (Web application, Server, Cortex-M4 Firmware)
 - Kernel configurations and custom Devicetree

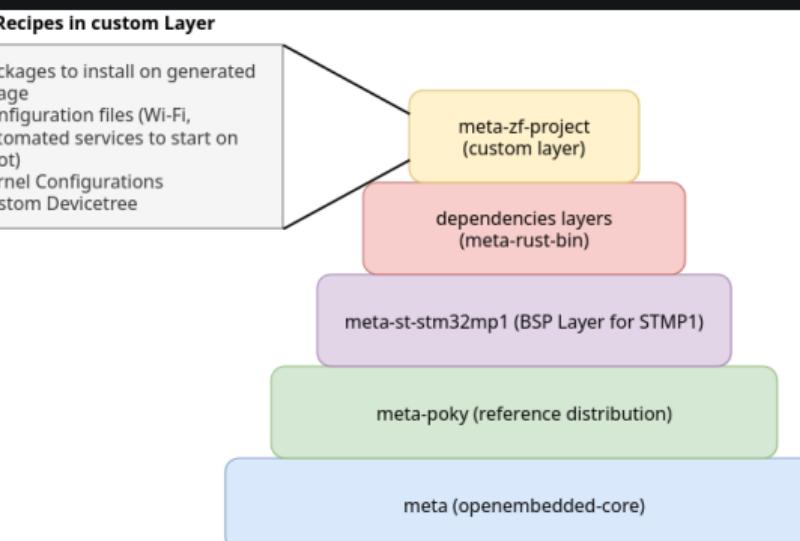


Figure 19: Layer Model representation of this project for deploying onto a STM32MP1-DK2

Claims Investigation Committee Multi-Input Testing Device

Design and Implementation

Embedded Linux With Yocto Project

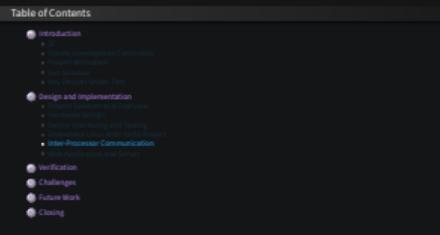
Custom Linux Image for the STM32MP157F-DK2

2024-12-02

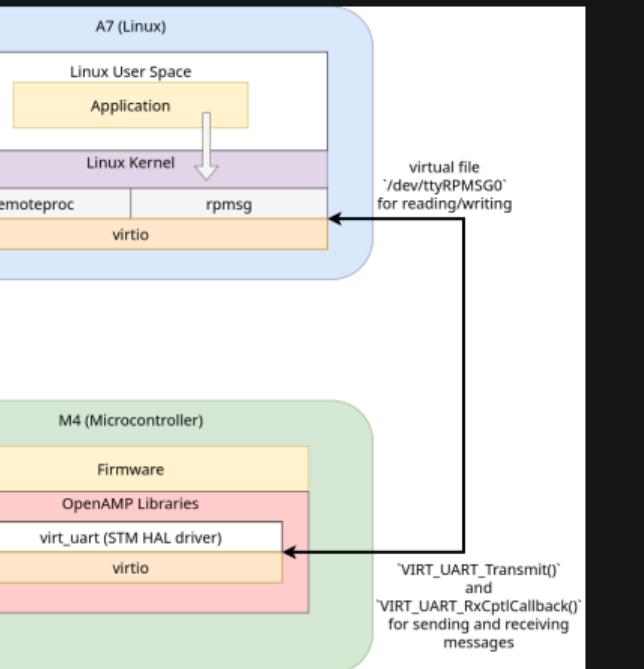


The yocto project was essential for building and configuring

1. hardware drivers
2. Linux Kernel configurations
3. Linux kernel device tree
4. and using the layer structure created by the yocto project a custom layer of meta-data was made to build and configure
5. nginx, a webserver to host the web application
6. wpa_supplicant, a wi-fi access client
7. among the meta-data there were build recipes for
8. written programs like the web-application, web server, and the Cortex-M4 firmware
9. a custom kernel and and a custom devicetree

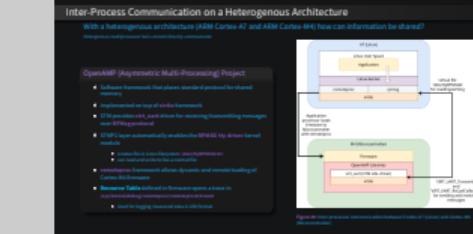
Introduction oooooo	Design and Implementation oooooooooooooooooooo●ooooo	Verification oo	Challenges oo	Future Work oo	Closing ooo
Inter-Processor Communication					
Table of Contents					
1 Introduction <ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test	2 Design and Implementation <ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server	3 Verification	4 Challenges	5 Future Work	6 Closing
2024-12-02	<h1>Claims Investigation Committee Multi-Input Testing Device</h1> <ul style="list-style-type: none">└ Design and Implementation └ Inter-Processor Communication └ Table of Contents				
 A sidebar titled "Table of Contents" on the right side of the slide. It contains a hierarchical tree structure: <ul style="list-style-type: none">● Introduction<ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test● Design and Implementation<ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server● Verification● Challenges● Future Work● Closing					

- OpenAMP (Asymmetric Multi-Processing) Project
- Software framework that places standard protocol for shared memory
- Implemented on top of **virtio** framework
- STM provides **virt_uart** driver for receiving/transmitting messages over **RPMMsg** protocol
- STMPI layer automatically enables the **RPMMSG tty driver** kernel module
 - creates file in Linux filesystem: `/dev/ttyRPMSG<X>`
 - can read and write to like a normal file
- **remoteproc** framework allows dynamic and remote loading of Cortex-M4 firmware
- **Resource Table** defined in firmware opens a trace in `/sys/kernel/debug/remoteproc/remoteproc0/trace0`
 - Used for logging measured data in CSV format



Claims Investigation Committee Multi-Input Testing Device

- └ Design and Implementation
 - └ Inter-Processor Communication
 - └ Inter-Process Communication on a Heterogenous Architec



1. a major problem to solve in this project was having the two processors communicate to eachother
 2. the cortex-m4 and cortex-A7 are on the same system on chip so having them communicate is not for free
 3. OpenAMP is a project that aims to solve this project by standardizing how heterogenous architectures communicate between cores
 4. to briefly summarize how its used in this project:
 5. it allows linux to load the cortex-M4 firmware after a device under test is determined
 6. communication between the two processors is possible, with the cortex-m4 utilizing some abstraction called virtuart and linux having a virtual file to read and write to
 7. and logging from the cortex-m4 is done through writing to a linux kernel trace file, which will be used for logging the CSV data

Introduction oooooo	Design and Implementation oooooooooooooooooooo●oooo	Verification oo	Challenges oo	Future Work oo	Closing ooo
Web Application and Server					
<h2>Table of Contents</h2>					
<ul style="list-style-type: none">1 Introduction<ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test2 Design and Implementation<ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server3 Verification4 Challenges5 Future Work6 Closing					
2024-12-02					
<h2>Claims Investigation Committee Multi-Input Testing Device</h2>					
<ul style="list-style-type: none">└ Design and Implementation └ Web Application and Server └ Table of Contents					
<small>Table of Contents</small>					
<ul style="list-style-type: none">● Introduction<ul style="list-style-type: none">● ZF● Claims Investigation Committee● Project Motivation● Our Solution● Key Devices Under Test● Design and Implementation<ul style="list-style-type: none">● Project Solution and Overview● Hardware Design● Device Interfacing and Testing● Embedded Linux With Yocto Project● Inter-Processor Communication● Web Application and Server● Verification● Challenges● Future Work● Closing					
34 / 47					

The Rust programming language was used to write both major applications (web-based application and web server) for 2 main reasons.



Figure 21: Ferris, universally accepted mascot of the Rust Programming language

Memory Safety and Performance

- A set of rules called **Ownership** enforced by compiler to prevent memory leaks
- **Borrow checker** within the compiler prevents programs unsafe programs from compiling*
- Nearly as or just as performant as C with **Zero Cost Abstractions**
- Advocated by/used by several United States government agencies:
 - **National Security Agency (NSA)** and **Cybersecurity and Infrastructure Security Agency (CISA)**
 - **Defense Advance Research Projects Agency**
 - **The White House**



2024-12-02

Rust

The Rust programming language was used to write both major applications (web-based application and web server) for 2 main reasons:

Memory Safety and Performance

- A set of rules called Ownership enforced by compiler to prevent memory leaks
- Borrow checker within the compiler prevents programs unsafe programs from compiling*
- Nearly as or just as performant as C with Zero Cost Abstractions
- Advocated by/used by several United States government agencies:
 - National Security Agency (NSA) and Cybersecurity and Infrastructure Security Agency (CISA)
 - Defense Advance Research Projects Agency
 - The White House

Figure 21: Ferris, universally accepted mascot of the Rust Programming language



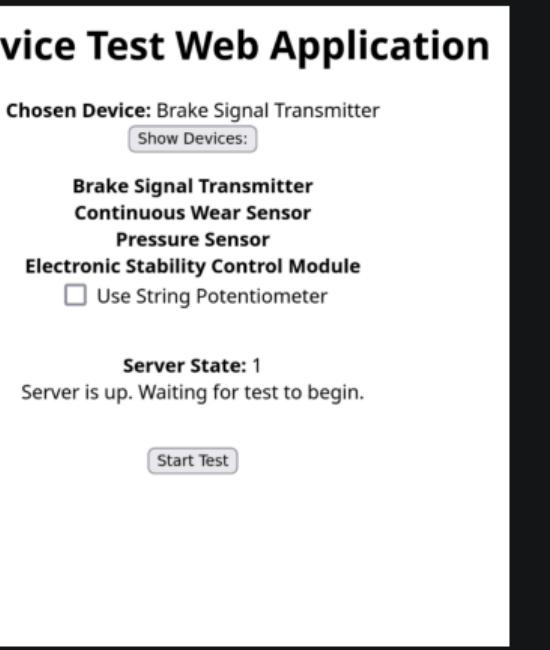
Web-Application for User Interface

Web Application in WebAssembly (WASM)

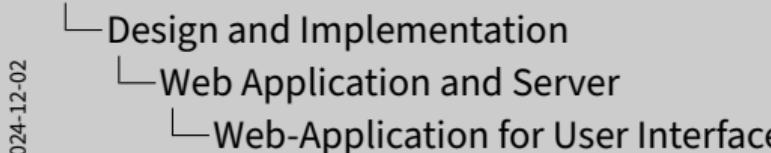
- WASM is a compiled, binary format executable
 - Much faster than traditional Javascript programs
 - Using the Yew framework, written in Rust

Web application Features

- Shows if application is connected to associated server
 - Selection of different devices
 - Shows progress and state of test
 - Allows download to results in a CSV



Claims Investigation Committee Multi-Input Testing De



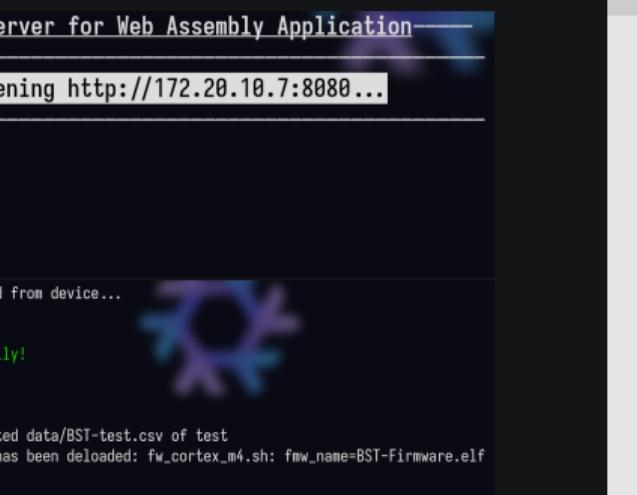
1. The way the tester interacts with our solution is through the web application as a web assembly application ,which is a modern solution to slow web programs which are usually written in javascript.
2. Web assembly is much faster than javascript as it is ran in web browsers as a compiled binary program
3. the web application allows a selection of the device they wish to test, and the starting of the test, which sends serialized JSON data to the web server which we will talk about next



Web Application and Server

Web Server features

- Handles **HTTP requests** from web application
 - Dynamically loads M4 Firmware for selected device with **remoteproc**
 - Polls for results by reading and writing to **/dev/ttyRPMSG0**
 - Saves information from **/sys/kernel/debug/remoteproc/remoteproc0/trace0** as CSV for download



logging of server application

2024-12-02

Claims Investigation Committee Multi-Input Testing De

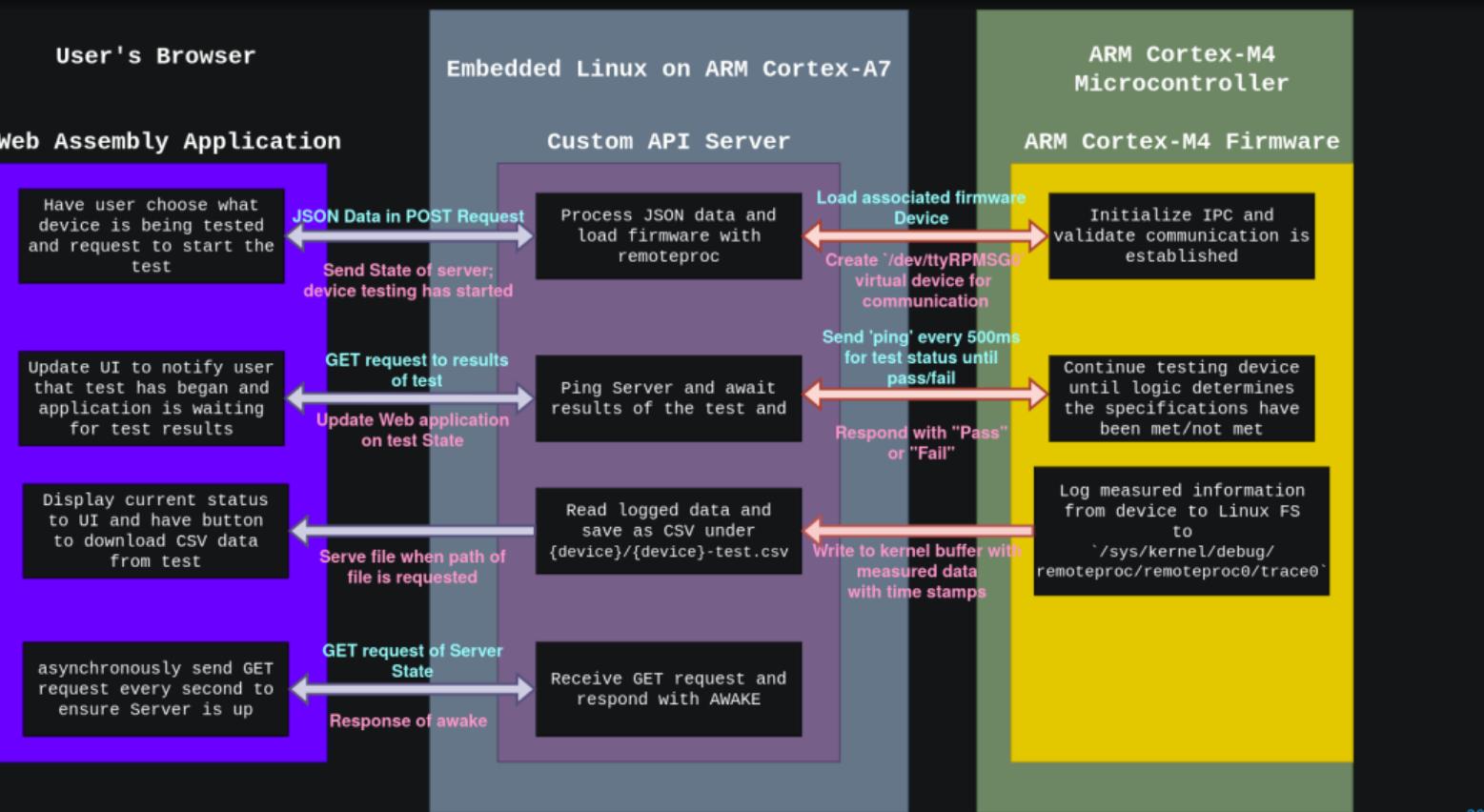
- └ Design and Implementation
 └ Web Application and Services
 └ Custom API Web Servers



the web server, also developed in rust, has multiple responsibilities:

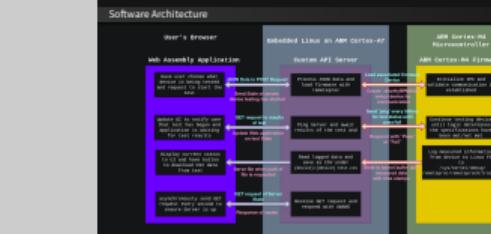
1. it handles http requests from the web application
 2. responsible for the interprocess communication between the two processors
 3. it loads the firmware to the cortex-m4 from the information received by the web-application
 4. It polls the Cortex-M4 for test completion and once it is, it saves the written data in the kernel trace and saves it for download
 5. **image shows the debug logging of the server application

Software Architecture



104

Claims Investigation Committee Multi-Input Testing D



this diagram shows a comprehensive interactions of all 3 moving parts of the project

1. it begins with the user choosing what device to test
 2. it sends a HTTP request to the server and then loads the necessary firmware
 3. once its loaded, a message is sent back to the server and the server updates the state of the web application user interface
 4. the server polls the test for completion and once it is, it notifies the server
 5. the server again updates the user interface on the web application and also shows a button to download the CSV and to restart another test

Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification
- 4 Challenges
- 5 Future Work
- 6 Closing

2024-12-02

Claims Investigation Committee Multi-Input Testing Device

Verification

Table of Contents

Table of Contents

- Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- Verification
- Challenges
- Future Work
- Closing

Introduction
oooooo

Design and Implementation
oooooooooooooooooooooooooooo

Verification
o●

Challenges
oo

Future Work
oo

Closing
ooo

Link to video demonstration

<https://dylxndy.xyz/senior-design-presentation/verification>

Claims Investigation Committee Multi-Input Testing Device

└ Verification

Link to video demonstration
<https://dylxndy.xyz/senior-design-presentation/verification>

Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification
- 4 Challenges
- 5 Future Work
- 6 Closing

Claims Investigation Committee Multi-Input Testing Device

└ Challenges

 └ Table of Contents

2024-12-02

Table of Contents

- Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- Verification
- Challenges
- Future Work
- Closing

Challenges

- System Clock configuration with Devicetree
- Timer configuration for PWM signals
- Mini-360 Buck Converter
- PCB Creation

Claims Investigation Committee Multi-Input Testing Device

Challenges

Challenges

2024-12-02

Challenges

- System Clock configuration with Devicetree
- Timer configuration for PWM signals
- Mini-360 Buck Converter
- PCB Creation

Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification
- 4 Challenges
- 5 Future Work
- 6 Closing

Claims Investigation Committee Multi-Input Testing Device

Future Work

Table of Contents

2024-12-02

Table of Contents

- Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- Verification
- Challenges
- Future Work
- Closing

Future Work

- Finish CAN implementation for ESCM
 - USB to CAN used currently
 - enabled **CAN_GS_USB** module in Linux Kernel
- Improve Web application appearance

Claims Investigation Committee Multi-Input Testing Device

Future Work

Future Work

2024-12-02

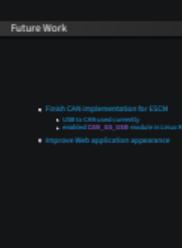


Table of Contents

- 1 Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- 2 Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- 3 Verification
- 4 Challenges
- 5 Future Work
- 6 Closing

Claims Investigation Committee Multi-Input Testing Device

└ Closing

 └ Table of Contents

2024-12-02

Table of Contents

- Introduction
 - ZF
 - Claims Investigation Committee
 - Project Motivation
 - Our Solution
 - Key Devices Under Test
- Design and Implementation
 - Project Solution and Overview
 - Hardware Design
 - Device Interfacing and Testing
 - Embedded Linux With Yocto Project
 - Inter-Processor Communication
 - Web Application and Server
- Verification
- Challenges
- Future Work
- Closing

Special Thanks

- Dr. Grantner (faculty advisor)
- David Florida (lab technician)
- Patrick McNally (Head of Engineering at ZF Group - Auburn Hills, MI)
- Davis Roman (Senior Staff Software Engineer at Rivian - Palo Alto, CA)

└ Closing

└ Special Thanks

2024-12-02

- Dr. Grantner (faculty advisor)
- David Florida (lab technician)
- Patrick McNally (Head of Engineering at ZF Group - Auburn Hills, MI)
- Davis Roman (Senior Staff Software Engineer at Rivian - Palo Alto, CA)

Introduction
oooooo

Design and Implementation
oooooooooooooooooooooooooooo

Verification
oo

Challenges
oo

Future Work
oo

Closing
oo●

Thank you

Any Questions?

Project Sources

- **Custom Yocto Project Layer:**
 - <https://github.com/DMGDy/meta-zf-project>
- **Custom Web Server in Rust**
 - <https://github.com/DMGDy/zf-webserver-app>
- **Web Application in WASM**
 - <https://github.com/DMGDy/zf-yew-app>
- **Microcontroller Firmware**
 - <https://github.com/danb127/Brake-System-Tester>
- **This Presentation**
 - <https://github.com/DMGDy/ECE4820-Presentation>

2024-12-02

Claims Investigation Committee Multi-Input Testing Device

└ Closing

└ Thank you

Thank you

Any Questions?

Project Sources

- Custom Yocto Project Layer:
 - https://github.com/DMGDy/meta-zf-project
- Custom Web Server in Rust
 - https://github.com/DMGDy/zf-webserver-app
- Web Application in WASM
 - https://github.com/DMGDy/zf-yew-app
- Microcontroller Firmware
 - https://github.com/danb127/Brake-System-Tester
- This Presentation
 - https://github.com/DMGDy/ECE4820-Presentation