



Embedded Connectivity SS23

*PROF. DR. ING. ANDREAS GRZEMBA
ING. JOHANN BRETZENDORFER*

G1WS5C1

Lakshmi Manasa Nambaru – 12202542

Vaishnavi Imandi – 22105113

Motivation:

Being from Electrical Engineering with Interest in Embedded Systems is the primary reason for us to opt this course as an Elective, we are willing to learn more and expand our knowledge with practical projects in Embedded Connectivity. Having some knowledge on CAN, Automotive Ethernet and Autosar has helped us to grasp the concepts quickly.

Table of Contents

Introduction

Project Setup

Feature List of MediaGateway

Workstation network structure analysis

Overview of tools

First view

Task – 1 : Andi Tool and Loopback

1.1 Basicloop Python Script

1.2 Further Analysis with wireshark

1.3 Creating sending and receiving scripts

Task – 2: ANDI TOOL AND AUTOMOTIVE ETHERNET

2.1 NETWORK STRUCTURE:

2.2 AUTOMOTIVE ETHERNET CONNECTION WITH MEDIAGATEWAY:

2.3 SETUP – MEDIAGATEWAY:

2.4 MediaGateway exercises:

Task – 3: ANDi Tool and Automotive Ethernet

3.1 BLOCK DIAGRAM:

3.2 MASTER SLAVE CONFIGURATION:

3.3 CONFIGURATION OF MEDIA GATEWAY:

3.4 MEDIA GATEWAY EXERCISES:

Task – 4: CAN over BroadRReach

4.1 Aim of the project

4.2 SETUP

4.3 PROCESS

4.4 UNDERSTANDING OF CAN AND CANOE:

4.5 Configure CAN and CANoe to display the data traffic on the CAN Trace.

4.6 The remaining bus simulation by the CANoe and CANcaseXL IS stopped and replaced by the "ANDi traffic generator".

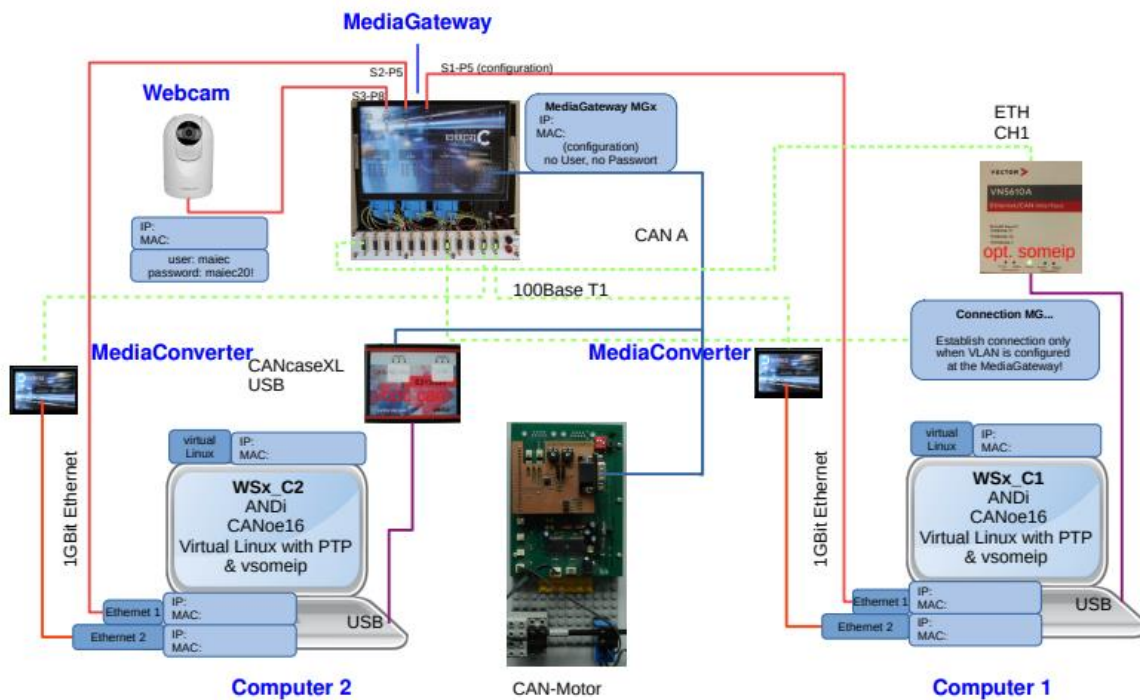
Conclusion

INTRODUCTION:

The primary objective of the embedded connectivity workshop is to provide participants with a fundamental understanding of ethernet communication testing and the various levels of data communication involved. Through this workshop, attendees will gain a basic understanding of MEDIAGATEWAY and ANDi software, which are crucial for creating a test and simulation environment for ethernet electronic control units.

This document presents a comprehensive report on the analysis and implementation of controller area network (can) over BroadReach r. The purpose of this project was to investigate the feasibility of utilizing BroadReach r as a communication medium for can and evaluate its performance compared to traditional wired solutions. The project employed the use of the ANDitool and Wireshark for data analysis and network monitoring. This introduction provides an overview of the project's objectives, methodology, and the significance of exploring this novel approach to can communication.

PROJECT SETUP



In our first lab, we had chance to look over our workstation (WORKSTATION5), consisting two computers each has 3 ethernet port which are used for communication task.

Connection	Function	Port MediaGateway
Ethernet	connection to internet over THD infrastructure Do not use for the Automotive Ethernet and do not change the settings!	—
Ethernet 1	computer 1 - MediaGateway configuration computer 2 - MediaGateway for analyzing with Wireshark	S1-P5 S2-P5
Ethernet 2	computer 1 - MediaGateway Logging/Stimulation computer 2 - MediaGateway Logging/Stimulation	S1-P3 S1-P2

FEATURE LIST OF MEDIAGATEWAY:

The Technica Engineering MediaGateway has the following basic features:

- **12 Ports** Broadcom BroadR-Reach
 - **100 MBit/s** Full duplex on a single unshielded twisted pair
 - **3 Ports** Gigabit Ethernet 10/100/1000 BaseTX Full duplex
 - **1 Port** Gigabit Ethernet SFP module socket
- Broadcom BroadR-Reach Technology
 - Tyco MQS Connectors for BroadR-Reach and Power Supply
 - Webserver for easy configuration:
 - o Master / Slave
 - o Port Mirroring
 - o VLAN Tagging
 - o Port Status Display
 - Import and Export of Configurations
 - WakeUp functionality
 - CAN, LIN and FlexRay interfaces (requires customer specific software)
 - Power output for attached devices: VBAT max. 1,2 Ampere in total (Fused)
 - 19 Status LEDs
 - Possibility to reset to default settings by pushbutton
 - Robust steel case

WORKSTATION NETWORK STRUCTURE ANALYSIS:

By analyzing the network of that you know the IPs and MAC addresses of all connected devices at the end. You should make a note of this data, it is important for the further parts of the workshop. Work only in the subnet of the workstation 192.168.0.0/24!

OVERVIEW OF TOOLS:

i. ANDi Tool:

ANDi (Automotive Network Data and Interpretation) is a powerful tool specifically designed for analyzing and simulating Controller Area Network (CAN) networks. It provides a comprehensive set of features that facilitate the generation, monitoring, and analysis of CAN messages, allowing researchers and engineers to gain insights into the behavior and performance of CAN-based systems.

- We have analyzed the python basic_loopback script using ANDi tool.

ii. **Ping:**

Ping is a diagnostic tool of the Windows or Linux command line that can be used to check whether a particular host on an IP network is reachable.

- Measures the time takes for the packet to travel from the sender to the receiver and back.
- Providing valuable information about the network latency and round-trip time.

iii. **ipconfig/ifconfig:**

ipconfig is a command line command included in Microsoft Windows. It displays the used IP addresses and other network information of a computer. Additional call parameters can be used to trigger certain network actions. The Linux tool "ifconfig" does something similar.

- We determined IP and MAC address of our computer using ipconfig

iv. **nmap:**

nmap is a free port scanner for scanning and evaluating hosts in a computer network. The name stands for Network Mapper. The software is available for Windows and Linux. Zenmap is the corresponding graphical user interface.

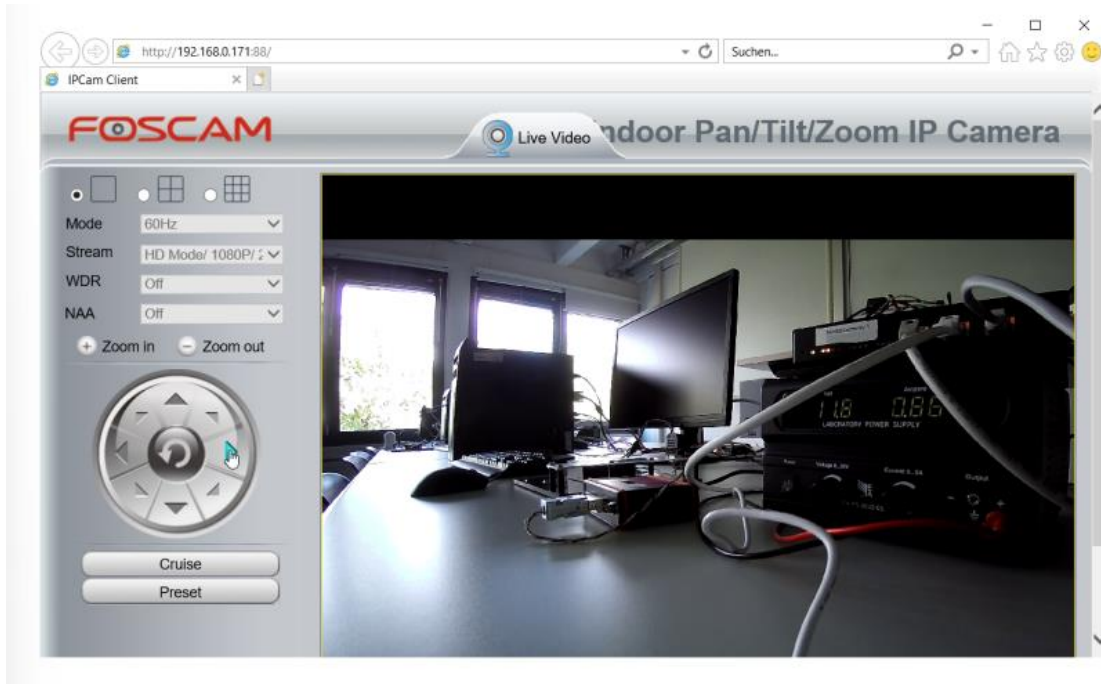
v. **Wireshark:**

Wireshark is software for analyzing and graphically preparing data logs (sniffer). The network analysis tool can help administrators, network experts and security professionals to find network problems, identify botnet connections or manage networks.

- We recorded the network traffic and handled all common ethernet protocols.
- We also used some filters to analyze selected protocols.

FIRST VIEW:

- We can access the webcam with the Internet explorer.
- We opened the Internet explorer and entered the link <http://IP-address:port>.
- And then, the IP address varies from individual workstations, the IP address of our system that we entered is <http://192.168.0.49>.
- After entering the link, we have to type the user id "maiec" and password "maiec20".
- Then, we got access to webcam, and this access works as long as the media gateway has not been programmed.



PART – 1: ANDI TOOL AND LOOPBACK

The main aim is to understand the working of ANDi tool and wireshark by working with ANDi loopback example.

LOADING PROJECT:

the ANDi Tool and open the file “**Demo.atp**” which we downloaded from the ilearn, basic loopback code is displayed on the screen after opening the Demo.atp file.

ADAPTER CONFIGURATION:

Adapters configuration			
Configuration	Add	Delete	Refresh adapters Detect hardware Auto Fix Channels
Channel	Physical network adapter		Default
Stimulation	Microsoft Loopbackadapter für KM-TEST (Ethernet 4)		
Logging	Microsoft Loopbackadapter für KM-TEST (Ethernet 4)		
MediaGateway	Technica Engineering - Automotive Ethernet Switch (70:B3:D5:4C:00:50) channel CAN A		
CANcase	CANCASEXL (65775) Channel 2 (CAN)		

- The important configurations are stimulation and logging which are connected with the loopback interface.
- We used the Stimulation adapter for transmitting the signals, the Logging adapter for receiving the signals.

1.1 BASICLOOP PYTHON SCRIPT:

```

1 from time import sleep
2
3 #providing a custom MAC address
4 custom_mac = "11:22:33:44:55:66"
5
6 #
7 def on_eth_msg_received(msg):
8     if msg.mac_address_source == custom_mac:
9         print("Received following message from {0}: {1}".format(custom_mac, msg))
10
11
12 g_ethernet_msg.mac_address_source = custom_mac
13 g_ethernet_msg.mac_address_destination = Logging.get_mac()
14
15 # attching payload to the packet
16 g_ethernet_msg.payload = System.Array[Byte](bytearray.fromhex("01 02 03 04 09"))
17
18 # received packet will be sent to the custom service
19 g_ethernet_msg.on_message_received += on_eth_msg_received
20 g_ethernet_msg.start_capture()
21
22 sleep(1)
23
24 # packet created is send
25 g_ethernet_msg.send()
26
27 sleep(1)
28
29 g_ethernet_msg.stop_capture()
30 g_ethernet_msg.on_message_received -= on_eth_msg_received

```

- After loading the “Demo.atp” file into ANDi script editor, the code is displayed on the editor.
- We ran the script by clicking the arrow button.
- The primary function of a script is to transmit a packet to the loopback adapter and subsequently receive the same packet from the loopback adapter, ensuring a seamless loopback process.
- To establish synchronization and logging interfaces on a loopback adapter, we can utilize a virtual adapter created on a Windows operating system.
- Here, we are having a sleep function of 1 sec, which means, the messages are transmitted in an interval of 1 sec.
- Since this setup doesn't involve actual two-way communication, there is no need for customization of the MAC address.

1.2 FURTHER ANALYSIS WITH WIRESHARK:

- Wireshark is a packet analyzer that is freely available and open source.
- It is capable of capturing network traffic within the local network and storing it for later analysis.
- In our specific scenario, we observed the source and destination MAC addresses, along with the transmitted and received payload data "01 02 03 04 09" using a loopback adapter.
- We applied a MAC address filter to obtain the desired output.

1.3 CREATING SENDING AND RECEIVING SCRIPTS:

To enhance the functionality of the basic loopback script, we created two new scripts. The first script will be responsible for sending a frame, while the second script will handle receiving a frame. Additionally,

since the initialization of the script is triggered by a designated button, we will need to modify the sleep time values accordingly.

USING TIMER FUNCTION:

- Once the basic loopback script is executed successfully, our send script will incorporate a timer function.
- This function will ease the transmission of an Ethernet frame every second for approximately 20 seconds, ensuring improved verification.
- Additionally, to enhance the verification process, we will increment the first byte of the payload data.

CHARACTERISTICS OF SENDING SCRIPT

- In the script, a timer function of 1 sec is employed to ensure that frames are sent at regular one-second intervals.
- To facilitate easier examination, the first byte of each user data frame is designated as a serial number.
- Here, we used the MAC address of Computer 2 "00:13:3B:B0:0E:A8" as a destination address.
- Finally, the script concludes by transmitting a predetermined message to the destination.

```
from time import sleep

custom_mac = "00:13:3B:B0:10:61"

g_ethernet_msg.mac_address_source = custom_mac
g_ethernet_msg.mac_address_destination = "00:13:3B:B0:0E:A8"
g_ethernet_msg.payload = System.Array[Byte](bytearray.fromhex("01 02 03 04 09"))

g_ethernet_msg.start_capture()
sleep(1)
g_ethernet_msg.send()
sleep(1)
g_ethernet_msg.stop_capture()
```

CHARACTERISTICS OF RECEIVING SCRIPT:

- The receiving script is designed to detect and capture all frames sent by the sending script.
- Based on this mechanism, the receiving script will keep listening and receiving frames until it encounters a payload data of "00".
- Upon receiving this specific data, the script will cease waiting and exit automatically, disregarding any frames received in any timing conditions.
- Here, we used the MAC address of Computer 1 "00:13:3B:B0:10:61" as a source address.

```
from time import sleep

custom_mac = "00:13:3B:B0:10:61"

def on_eth_msg_received(msg):
    if msg.mac_address_source == custom_mac:
        print(("Received following message from {0}: {1}".format(custom_mac, msg)))

g_ethernet_msg.on_message_received += on_eth_msg_received
g_ethernet_msg.start_capture()

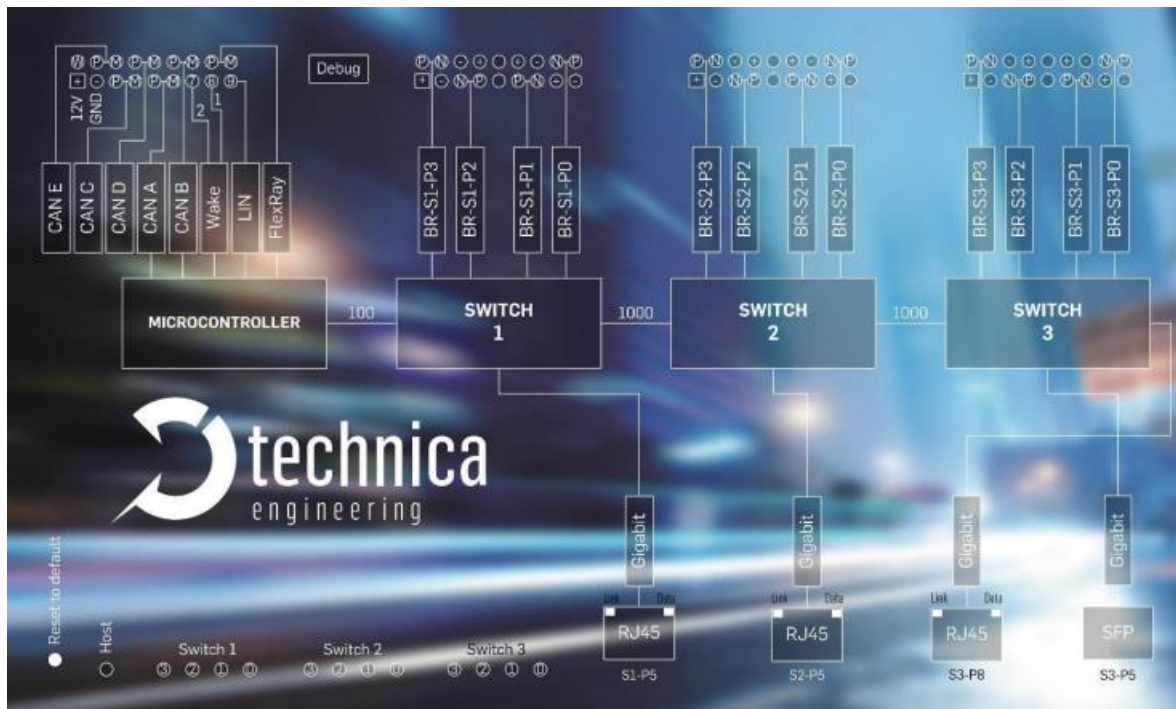
sleep(5)

g_ethernet_msg.stop_capture()
g_ethernet_msg.on_message_received -= on_eth_msg_received
```

PART – 2: ANDI TOOL AND AUTOMOTIVE ETHERNET:

- In the first part, we successfully simulated a point-to-point connection using the loopback adapter.
- Now, we will transition to a real network environment by employing the ANDi tool and the accompanying scripts.
- Our objective in this phase is to establish an actual point-to-point connection, treating the unconfigured media gateway as a Layer 2 switch.

MEDIA GATEWAY:



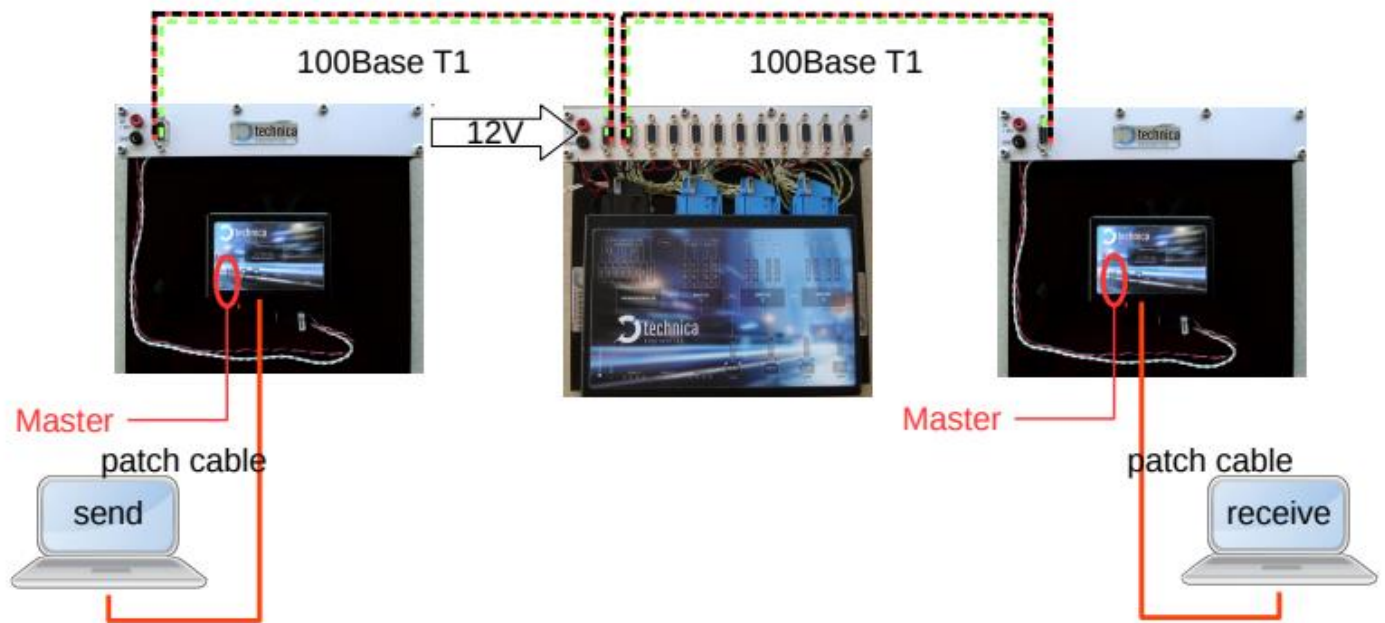
The MediaGateway device serves as a crucial tool for testing Ethernet communication in automotive devices. It acts as an intermediary platform between the testing and controlling devices, simplifying the control and filtering of communication between different systems. Configuration of the MediaGateway can be easily done through a web browser using the provided link. It operates on a 12V input voltage requirement. Technica Engineering's MediaGateway is a widely utilized development tool for testing and analyzing on-board vehicle networks, particularly those utilizing 100BASE-T1. Its built-in automotive switches and 12 x 100BASE-T1 ports allow for capturing traffic between devices while maintaining normal communication, as well as facilitating interaction with these devices within your testing setup. The MediaGateway supports features such as single and double tagging of VLANs, Mirroring, Forwarding, and more. It is highly suitable for subsystem testing stations and can be effortlessly installed in a test vehicle. Configuration is conveniently done via the integrated web server, accessible through a browser.

MEDIA CONVERTER:



The Media Converter serves as a connection bridge between BroadR-Reach and Ethernet-based computers during testing. It facilitates a direct point-to-point conversion between automotive ECU's utilizing 1000BASE-T1 and any standard Gigabit Ethernet device with an RJ-45 connector. The Media Converter does not store or modify packets and supports both 1000 Mbps and 100 Mbps modes through the Marvell 88Q2112 Phy. Overall, it is an efficient solution for seamless integration with the latest 1000BASE-T1 technology, enabling quick and effective operations.

2.1 NETWORK STRUCTURE:



- Firstly, we have to confirm that the network connections of both computers are in the same class c network.
- The connections between the two Media Converters and the Media gateway are made with a twisted pair cable 100Base-T1.
- Then, write down the MAC and IP address of both networks.

Connection	Function	Port MediaGateway
Ethernet	connection to internet over THD infrastructure Do not use for the Automotive Ethernet and do not change the settings!	—
Ethernet 1	computer 1 - MediaGateway configuration computer 2 - MediaGateway for analyzing with Wireshark	S1-P5 S2-P5
Ethernet 2	computer 1 - MediaGateway Logging/Stimulation computer 2 - MediaGateway Logging/Stimulation	S1-P3 S1-P2

- Firstly, we started the ANDi tool and loaded the previous project done in task 1.
- And then, update the ANDi adapter configuration with the appropriate network adapters, then we don't use loopback anymore.
- We used a "Simulation" adapter for transmitting and a "Logging" adapter for receiving.
- And from the scripts we have adjusted the MAC addresses.

SEND SCRIPT WITH DELAY

```

mac_destination = ["00:13:3B:B0:0E:A8"]
mac_source = ["00:13:3B:B0:10:61"]
message_span = 0
# loop for to keep on sending data at an interval of 20s for total 20000s
while (message_span<1000):

    # fill the fields of ethernet message object defined
    g_ethernet_msg.mac_address_source = mac_source[0] #Logging.get_mac()
    g_ethernet_msg.mac_address_destination = mac_destination[0]
    g_ethernet_msg.payload = System.Array[Byte](bytearray("SCRIPT TEST","utf-8"))

    # send the ethernet message object and wait for 20s
    sleep(20)
    g_ethernet_msg.send()
    sleep(20)

    # stop sending message and decrease the message counter by one
    g_ethernet_msg.stop_capture()
    message_span-=1

```

RECEIVE SCRIPT WITH DELAY:

```

def read(msg):\
print ("Received following message from {0} || {1} || {2}".format(msg.mac_address_source,
msg.payload, msg))

# loop for a total of 24000s and wait 10s for receiving each message
i = 0
while (i<1200):

    # on receiving a ethernet frame attach the frame to a ethernet message type object
    g_ethernet_msg.on_message_received += read
    g_ethernet_msg.start_capture()                #start listening to frames

    sleep(20)                                     #wait for a frame

    g_ethernet_msg.stop_capture()                  #stop frame capture
    g_ethernet_msg.on_message_received -= read

```

```
# once message receiving is complete remove function from message object  
i += 1
```

COMMUNICATION:

With the media gateway's default configuration, it is now feasible for one computer within the system to send messages while the other computer can receive them.

2.2 AUTOMOTIVE ETHERNET CONNECTION WITH MEDIAGATEWAY:

VLAN:



VLAN (Virtual Local Area Network) allows the creation of a separate LAN within an existing LAN. It enables the isolation of a LAN by tagging packets with VLAN tags to determine if they are allowed to enter the VLAN. Only VLAN-tagged frames with matching VLAN IDs are forwarded, while those that do not match are dropped. This approach results in a more efficient utilization of bandwidth. Additionally, VLANs enhance security by ensuring that frames can only enter the VLAN if they carry the correct VLAN ID, preventing non-tagged frames from causing trouble in the network. It is important to note that Windows operating systems do not natively support VLANs, so connections to Windows computers should be untagged.

Initially, we establish a direct connection between two systems to facilitate data transfer. The source PC sends data to the target PC (partner system) using the structure `g_ethernet_msg.mac_address_destination` to specify the destination MAC address. The data is received on the target PC through the specified destination MAC address. To establish the connection, we utilize a connector that connects two D-Link adapters. We

employ the `g_ethermsg.send()` function to transmit the payload to the target system. The D-Link Adapter plays a crucial role in extending the transfer speed of previous USB Fast Ethernet adapters to achieve true 10/100 Mbps connectivity. By utilizing the D-Link High-Speed USB 2.0 Fast Ethernet Adapter, the need for ISA, PCI, or PC Card slots is eliminated, making it the simplest way to connect a computer to an Ethernet-based network.

2.3 SETUP – MEDIAGATEWAY:

Before establishing the network connections, it is necessary to configure the media gateway accordingly. It is important to note that making configuration changes during operation can increase the risk of malfunctions.

To begin, launch Firefox and enter the default IP address, 192.168.0.49, in the address field. Next, we need to prepare the internal ports for VLAN by assigning them a unique VLAN number (049) and setting them as untagged for connection with Windows.

Please avoid using this VLAN ID for other ports unless you intend to transfer system data (e.g., CAN port).

Here are the steps to follow:

1. Select "Switch Status."
2. Choose IEEE 802.1q (VLAN) mode (remember not to save the configuration yet).
3. Open P4 (CPU) and enter the VLAN ID (049) as depicted in the accompanying images.
4. Repeat the same process for S1-P5.
5. Finally, check the "Restart after saving" box, and save the configuration.

First, we have to activate the VLAN feature of Media Gateway.

technica engineering

Automotive Ethernet Switch

System Information Control Panel Switch Status Contact

Switch Status

Global configuration

Configuration changed. [Save configuration](#) Restart after saving ☐

IEEE 802.1q (VLAN) mode ☒

Double tagging ☐

Double tagging TPID (hex) 9100

Select a port or switch on the left for details.

To configure the Media Gateway, we utilize S1-P1, which is connected to Ethernet 1 of the PC, and S1-P4, which serves as an internal port of the Media Gateway. Both ports share identical configurations as presented in the table below. Our objective is to assign them as members of VLAN 049, with the default VLAN ID set to 049. Since Windows does not support VLAN, we leave the option unchecked for both ports.

technica engineering

Automotive Ethernet Switch

System Information Control Panel Switch Status Contact

Switch Status

Switch 1 Port 4

Configuration changed. [Save configuration](#) Restart after saving ☐

Port name S1-P4

Default VLAN ID (hex) 049

VLAN membership 049

VLANs to untag 049

Egress VID remarking Inner: As received Outer: As received

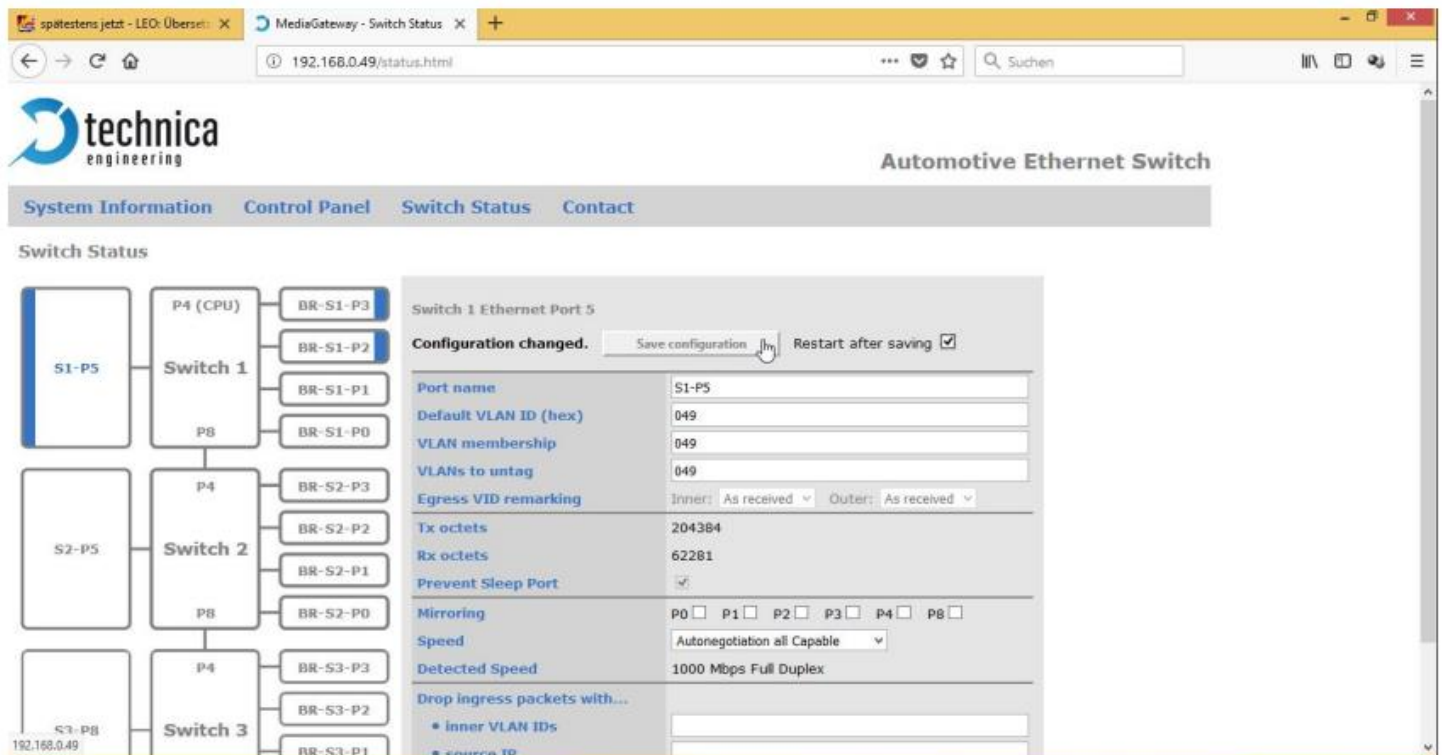
Tx octets 220283

Rx octets 1293296

Mirroring P0 ☐ P1 ☐ P2 ☐ P3 ☐ P5 ☐ P8 ☐

Drop ingress packets with...

- inner VLAN IDs
- source IP
- destination IP



2.4 MediaGateway exercises:

Establish a VLAN connection between the two computers by configuring the media gateway ports accordingly. Validate the functionality by utilizing your transmit and receive scripts. Use Wireshark to verify the traffic between the computers (refer to below table for the connection details). Furthermore, connect one of the computers to a webcam for additional functionality.

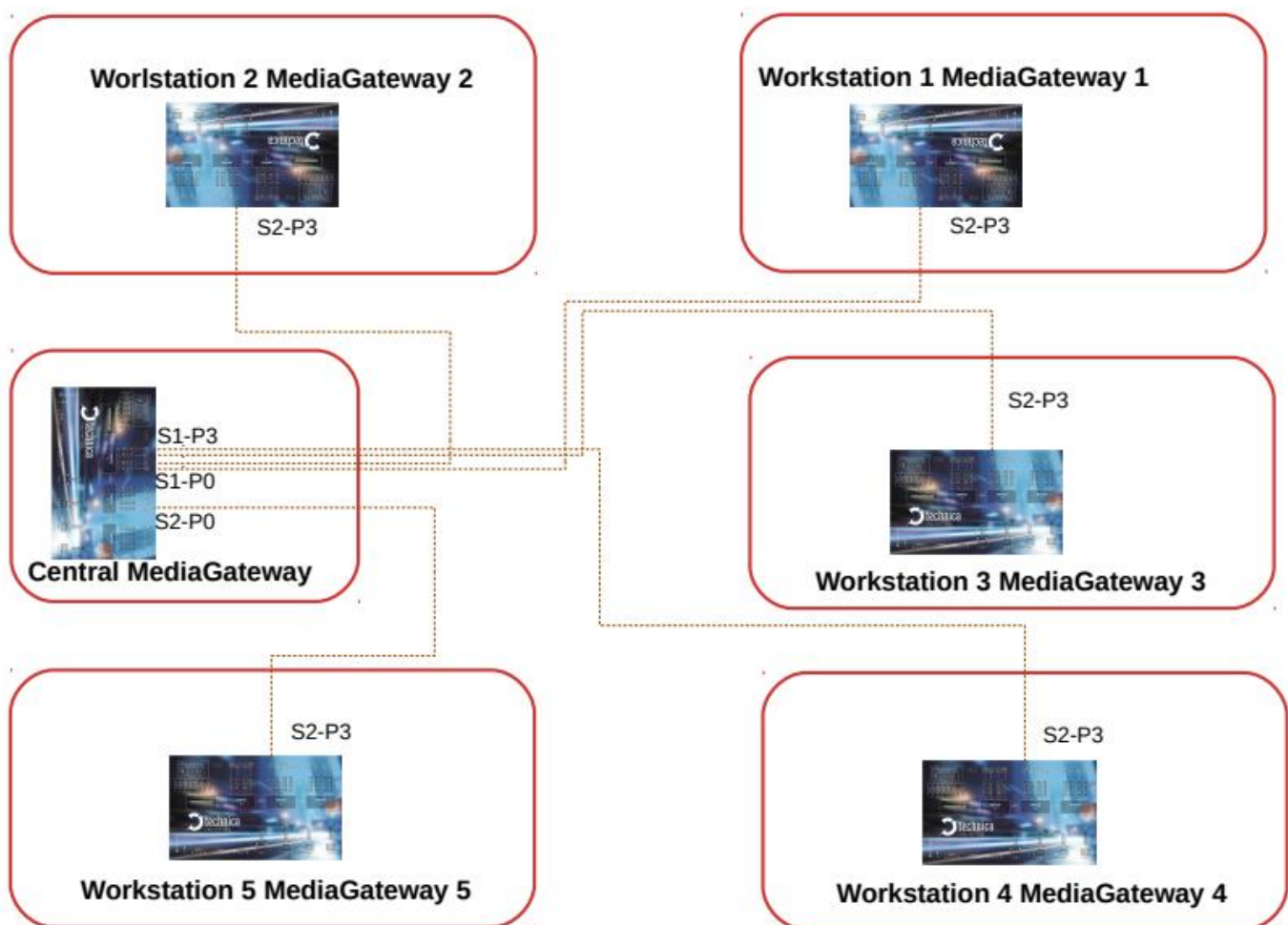
Connection	Function	Port MediaGateway
Ethernet	connection to internet over THD infrastructure Do not use for the Automotive Ethernet and do not change the settings!	—
Ethernet 1	computer 1 - MediaGateway configuration computer 2 - MediaGateway for analyzing with Wireshark	S1-P5 S2-P5
Ethernet 2	computer 1 - MediaGateway Logging/Stimulation computer 2 - MediaGateway Logging/Stimulation	S1-P3 S1-P2

Task – 3: ANDi Tool and Automotive Ethernet

- The star topology, also known as a star network, is a widely used network configuration.
- It involves connecting every node to a central network device such as a hub, switch, or computer.
- The central network device serves as a server, while the peripheral devices function as clients.
- The computers within the star topology are interconnected using either coaxial cables or RJ-45 network cables, depending on the network card utilized in each computer.

3.1 BLOCK DIAGRAM:

Our network is structured in a star topology, where a central MediaGateway acts as the connecting point for all other MediaGateways in different groups.



3.2 MASTER SLAVE CONFIGURATION:

The master/slave configuration is a communication protocol model commonly used in computer networking. In this model, a single device or process, referred to as the master, exercises control over one or multiple other devices or processes known as slaves. Once the master/slave relationship is established, control flows exclusively from the master to the slave(s).

3.3 CONFIGURATION OF MEDIA GATEWAY:

To configure the MediaGateway, the first step is to enable VLAN mode in the Global configuration. The initial configuration remains unchanged from the setup for a single communication path. However, in addition to that, we introduce the concept of mirroring. Mirroring allows the data to pass through a specific port on a switch. To implement mirroring, we select a mirroring port on the switch and assign a VLAN ID to that switch port. The same VLAN membership should be used in all the ports where the data needs to flow through.



Automotive Ethernet Switch

System Information
Control Panel
CAN Functions
Switch Status
Contact

Switch Status

Switch 1 Ethernet Port 5

Port name	S1-P5
Default VLAN ID (hex)	049
VLAN membership	049
VLANs to untag	049
Egress VID remarking	Inner: As received ▼ Outer: As received ▼
Tx octets	208824
Rx octets	63434
Prevent Sleep Port	<input checked="" type="checkbox"/>
Mirroring	P0 <input type="checkbox"/> P1 <input type="checkbox"/> P2 <input type="checkbox"/> P3 <input type="checkbox"/> P4 <input type="checkbox"/> P8 <input type="checkbox"/>
Speed	Autonegotiation all Capable ▼
Detected Speed	1000 Mbps Full Duplex
Drop ingress packets with...	<ul style="list-style-type: none"> inner VLAN IDs source IP destination IP
Drop mirrored packets with...	Mirroring is not active

To facilitate access to another gateway from the workstation, the Central MediaGateway was integrated.

	Port Central MediaGateway
MediaGateway 1 S2-P3	S1-P0
MediaGateway 2 S2-P3	S1-P1
MediaGateway 3 S2-P3	S1-P2
MediaGateway 4 S2-P3	S1-P3
MediaGateway 5 S2-P3	S2-P0
CentralWebcam IP 192.168.0.176	S3-P8

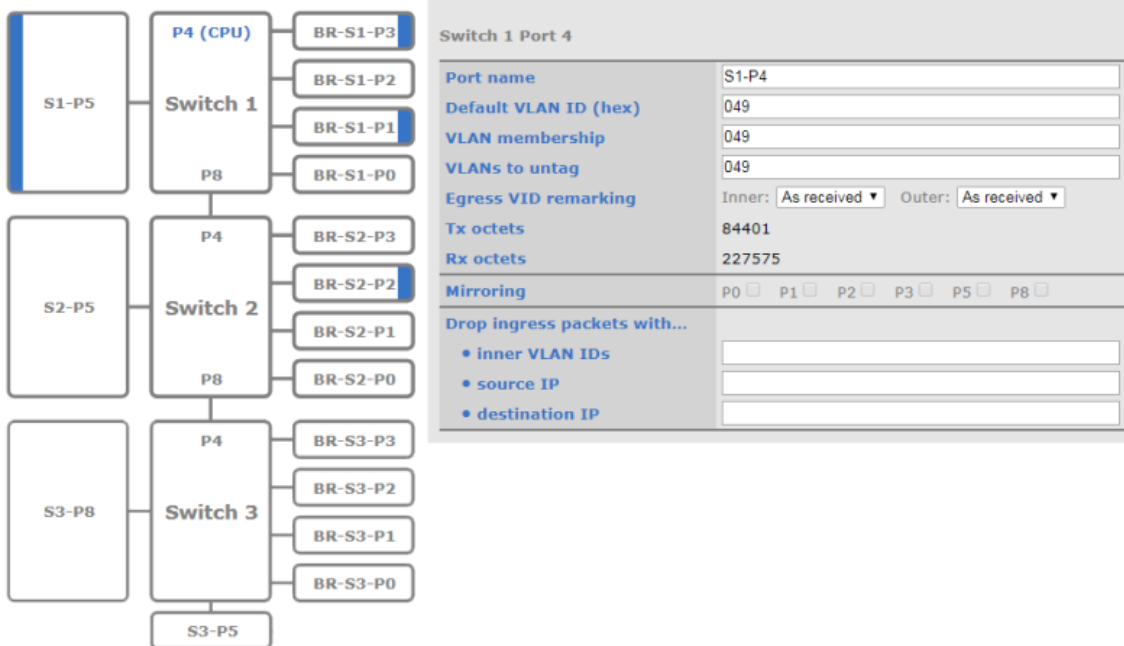
We aimed to establish a communication channel with another team and develop a script for simultaneous transmission and reception. Since it was not possible to connect both cables to the same port, we utilized a second port on the media gateway as a mirroring port. By configuring this port, it mirrors the traffic from another designated port. The diagram below illustrates the mirroring of traffic to port 4.



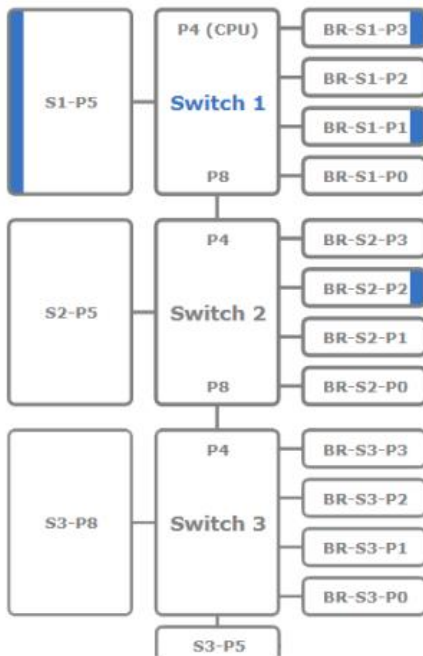
Automotive Ethernet Switch

System Information Control Panel CAN Functions Switch Status Contact

Switch Status



Switch Status



Switch 1

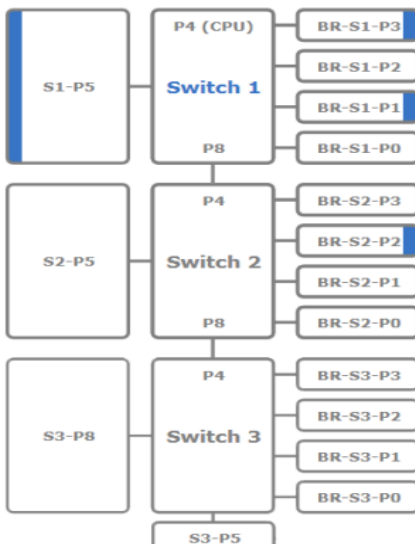
Functional Mode ☒ Dynamic ☐ Static

Address Resolution Table

Export

MAC address	VLAN ID	Fwd port	Age bit	Static bit
70:B3:D5:28:DF:AF	049	4	1	0
28:3B:82:C7:AF:37	049	5	1	0

Switch Status



Switch 1

Functional Mode ☒ Dynamic ☐ Static

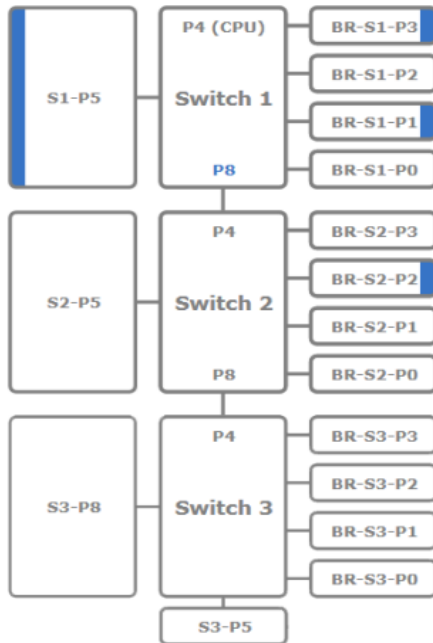
Address Resolution Table

Export

MAC address	VLAN ID	Fwd port	Age bit	Static bit
70:B3:D5:28:DF:AF	049	4	1	0
28:3B:82:C7:AF:37	049	5	1	0

[System Information](#) [Control Panel](#) [CAN Functions](#) [Switch Status](#) [Contact](#)

Switch Status

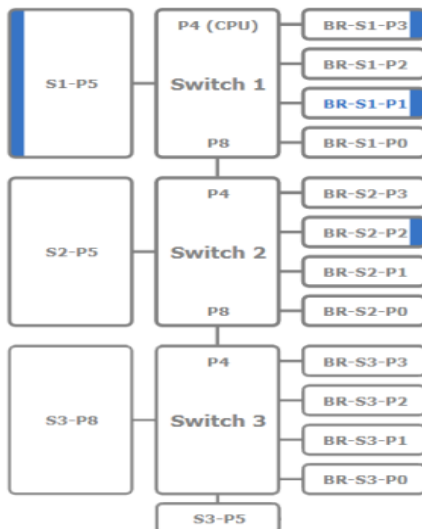


Switch 1 Port 8

Port name	S1-P8
Default VLAN ID (hex)	072
VLAN membership	072
VLANs to untag	072
Egress VID remarking	Inner: <input type="text" value="As received"/> Outer: <input type="text" value="As received"/>
Tx octets	284431
Rx octets	0
Mirroring	P0 <input type="checkbox"/> P1 <input checked="" type="checkbox"/> P2 <input type="checkbox"/> P3 <input checked="" type="checkbox"/> P4 <input checked="" type="checkbox"/> P5 <input type="checkbox"/>
Drop ingress packets with...	<ul style="list-style-type: none"> inner VLAN IDs <input type="text"/> source IP <input type="text"/> destination IP <input type="text"/>

[System Information](#) [Control Panel](#) [CAN Functions](#) [Switch Status](#) [Contact](#)

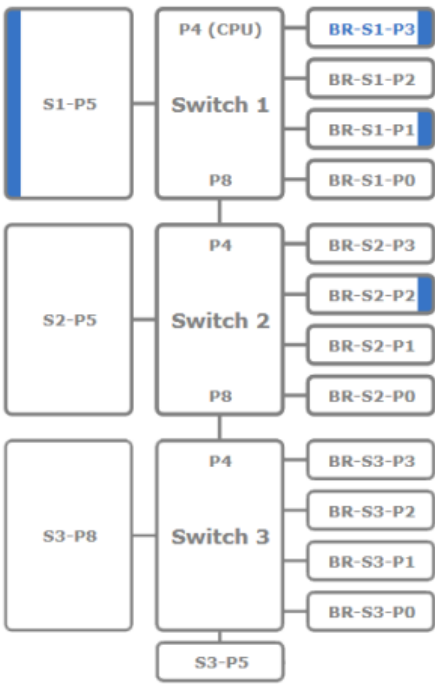
Switch Status



Switch 1 BroadR-Reach® Port 1

Port name	BR-S1-P1
Default VLAN ID (hex)	
VLAN membership	
VLANs to untag	
Egress VID remarking	Inner: <input type="text" value="As received"/> Outer: <input type="text" value="As received"/>
Tx octets	0
Rx octets	1640
Prevent Sleep Port	<input checked="" type="checkbox"/>
Enable port	<input checked="" type="checkbox"/>
BroadR-Reach® mode	Slave
Output level	Fullout
Link quality	Very good (4/5)
802.1AS mode	DISABLE
Test Mode	Normal operation mode
Drop ingress packets with...	<ul style="list-style-type: none"> inner VLAN IDs <input type="text"/> source IP <input type="text"/> destination IP <input type="text"/>
Redirect packets to...	<ul style="list-style-type: none"> Ethernet port <input type="text" value="DISABLE"/> with Destination IP <input type="text"/>

Switch Status

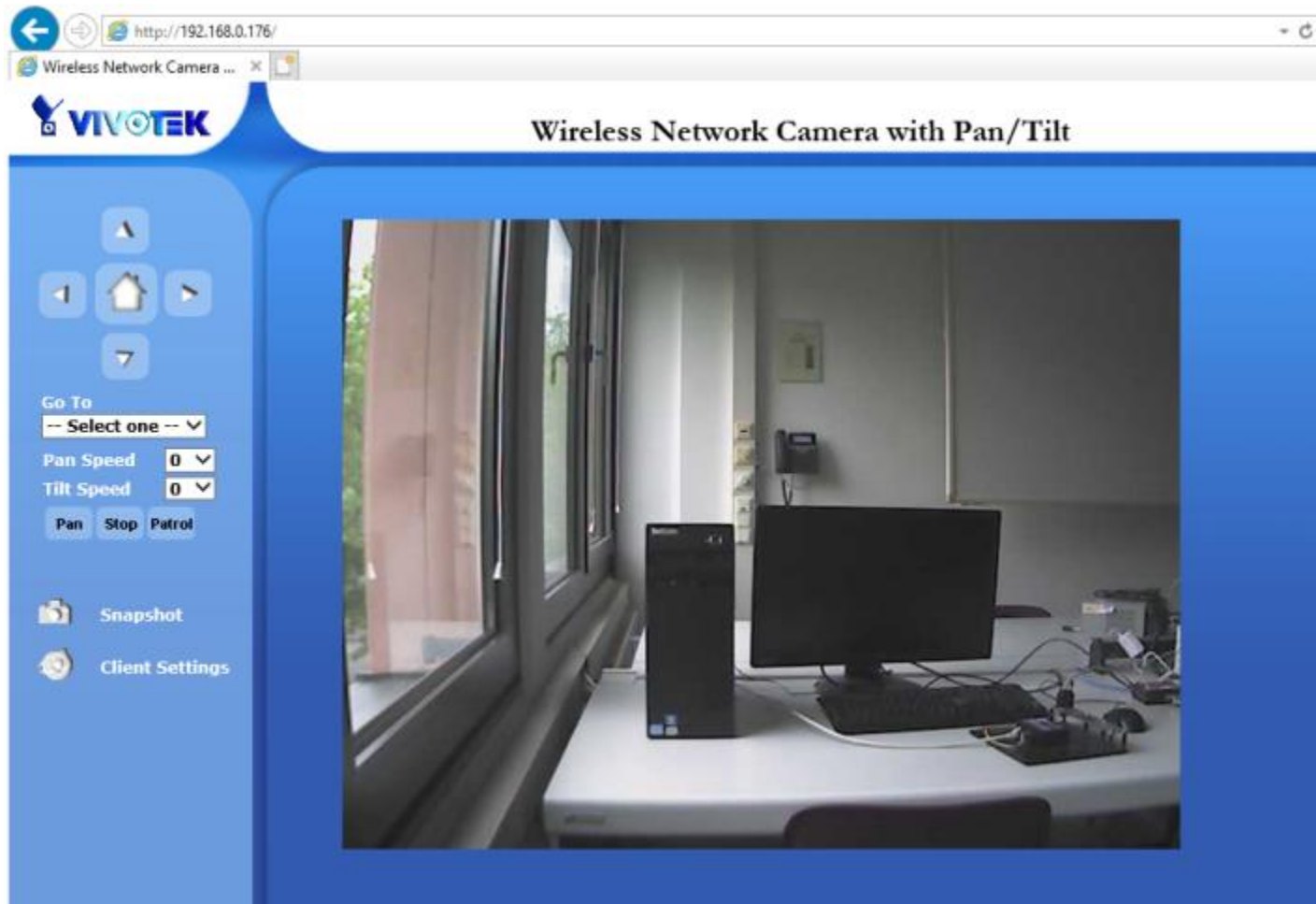


Switch 1 BroadR-Reach® Port 3

Port name	BR-S1-P3
Default VLAN ID (hex)	
VLAN membership	
VLANs to untag	
Egress VID remarking	Inner: As received ▼ Outer: As received ▼
Tx octets	0
Rx octets	32170
Prevent Sleep Port	<input checked="" type="checkbox"/>
Enable port	<input checked="" type="checkbox"/>
BroadR-Reach® mode	Slave ▼
Output level	Fullout ▼
Link quality	Very good (4/5)
802.1AS mode	DISABLE ▼
Test Mode	Normal operation mode ▼
Drop ingress packets with...	<ul style="list-style-type: none"> inner VLAN IDs source IP destination IP
Redirect packets to...	DISABLE ▼ with Destination IP <input type="text"/>

3.4 MEDIA GATEWAY EXERCISES:

- Attempt to establish a connection between your MediaGateway and the central MediaGateway in order to access and display the images from the central webcam.
- Use Wireshark to validate the traffic exchanged between the computers.



- Enter the IP address in the internet explorer, the IP address of our system that we entered is <http://192.168.0.49>.
- After entering the link, we have to type the user id “maiec” and password “maiec20”.
- Then, we got access to webcam, and this access works as long as the media gateway has not been programmed.

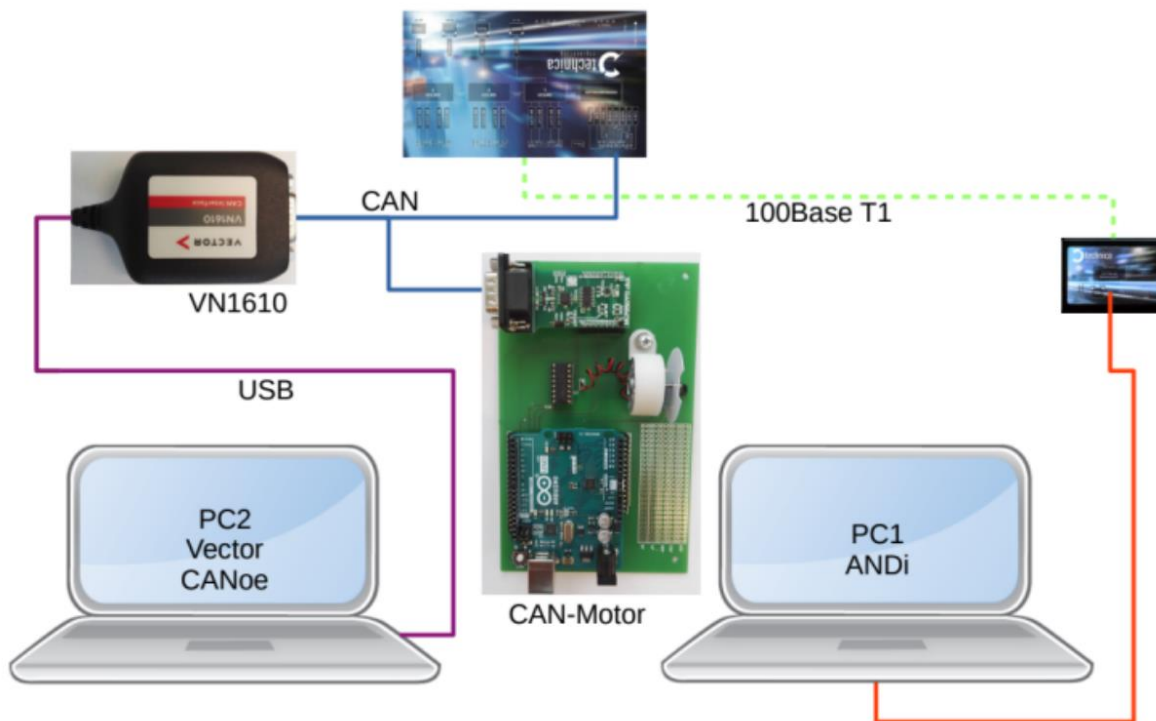
Task 4: CAN over BroadRReach

4.1 AIM OF THE PROJECT

Monitoring and remaining bus simulation with ANDi tool on a can structure via a broadreach line.

4.2 Setup

The structure of the workstation remains unchanged in principle. Only the CAN part is added, the CAN bus is connected to CAN A at the MediaGateway and the CANcaseXL is connected to computer 2 via USB. a Restbus simulation with CANoe is executed on computer 2.



4.3 PROCESS

You should configure the Media Gateway and ANDi to display the data traffic on the can connection line during bus operation in the "ANDi traffic viewer". The remaining bus simulation by the CANoe and CANcaseXL should then be stopped and replaced by the "ANDi traffic generator". This means that ANDi on PC1 transmits can frames encapsulated in UDP or RAW via the Broadreach line to the can bus.

4.4 UNDERSTANDING OF CAN AND CANOE:

CAN bus: It is a message based protocol designed to allow microcontrollers and devices to communicate with each other's applications without a host computer.

ID Allocation: Message IDs must be unique on a single CAN bus. In our project the messages IDs are:

0x101: MotorControl

0x102: MotorStatus

Frames: the standard or base frame format (described in CAN 2.0 A and CAN 2.0 B), and the extended frame format (described only by CAN 2.0 B). The only difference between the two formats is that the *CAN base frame* supports a length of 11 bits for the identifier, and the *CAN extended frame* supports a length of 29 bits for the identifier. In our project, we are using 11 bits for the identifiers.

Bit Arbitration: For example, consider an 11-bit ID CAN network, with two nodes with IDs of 15 (binary representation, 00000001111) and 16 (binary representation, 00000010000). If these two nodes transmit at the same time, each will first transmit the start bit then transmit the first six zeros of their ID with no arbitration decision being made.

	Start bit	ID bits											The rest of the frame
		10	9	8	7	6	5	4	3	2	1	0	
Node 15	0	0	0	0	0	0	0	0	1	1	1	1	
Node 16	0	0	0	0	0	0	0	1	Stopped Transmitting				
CAN data	0	0	0	0	0	0	0	0	1	1	1	1	

CANoe: Trace Log is used to check the status of ECUs whenever a task is initiated, we need and to check the communication between the ECUs. Another major advantage of using CANoe Tool for such purposes is that it saves a lot of time as compared to testing the ECUs in traditional manual approach.

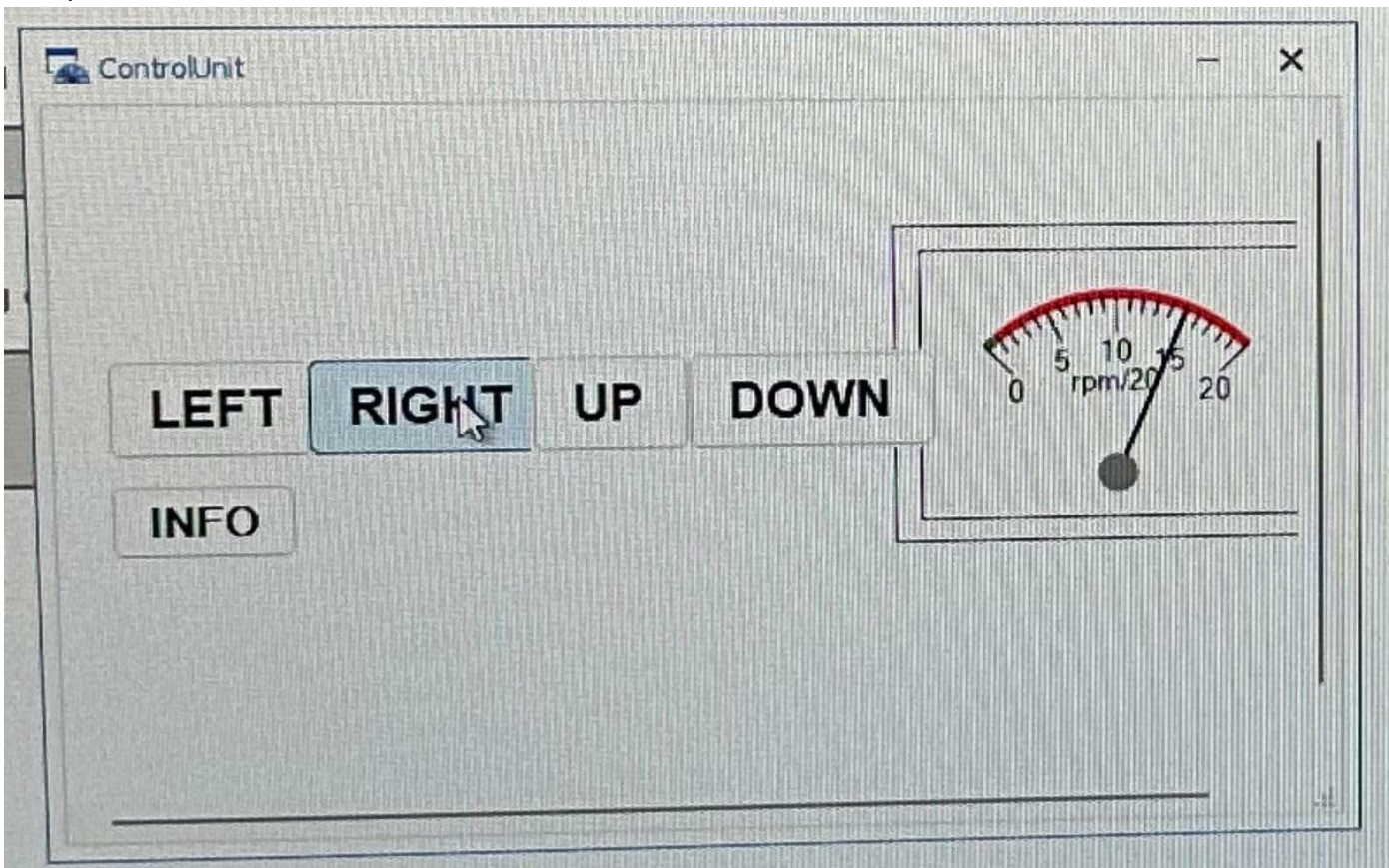
The motor can be controlled by 3 ways:

1. Manually controlling the speed and direction of the motor with the push buttons on the microcontroller.
2. Controlling the motor by CANoe panel.
3. Controlling the motor by media gateway by directly giving the speed and direction in numerical.

4.5 CONFIGURE CAN AND CANOE TO DISPLAY THE DATA TRAFFIC ON THE CAN TRACE

The control unit represents the same 4 pushbuttons on the board,

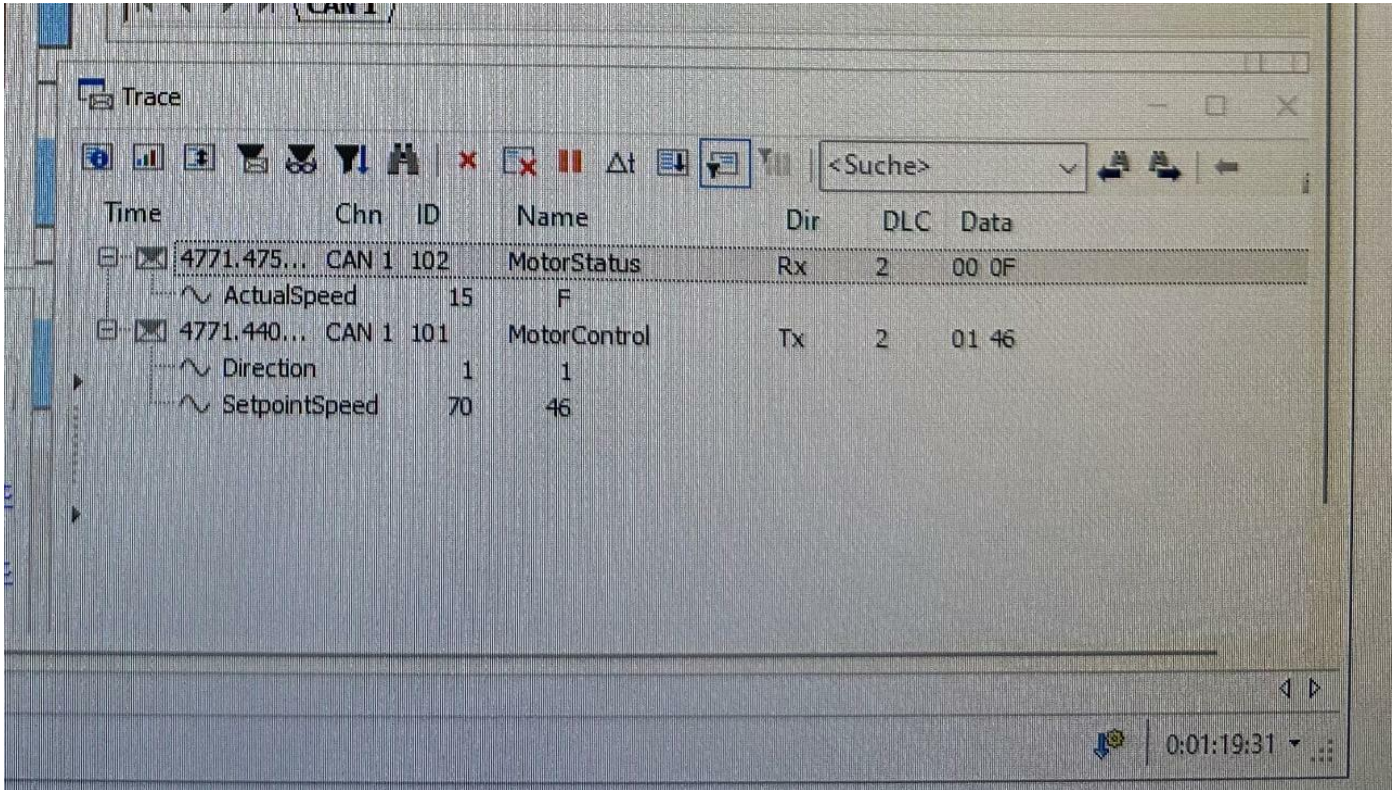
1. Left direction
2. Right direction
3. Speed increase
4. Speed decrease



CAN Trace

Motor control is transmitting the message and motor status is receiving the message. The motor status is showing the status of the current running motor.

The Tx id is 0x101 and the Rx id is 0x102. As we are using CAN bus as a communication protocol. The can bus prioritizes the messages by bit arbitration method. It means the lowest id has the highest priority, so when 0x101 and 0x102 are sent simultaneously on the can bus, 0x101 wins the bit arbitration and is sent first on the CAN bus.



Can frames in detail:

Tx frame: motor control

Tx Id: 0x101

Dlc: 02

Data: 01 46

Here 0x01 is the direction and 0x46 is the speed which is given as an input to the motor.

Rx frame: motor status

Rx Id: 0x102

Dlc: 02

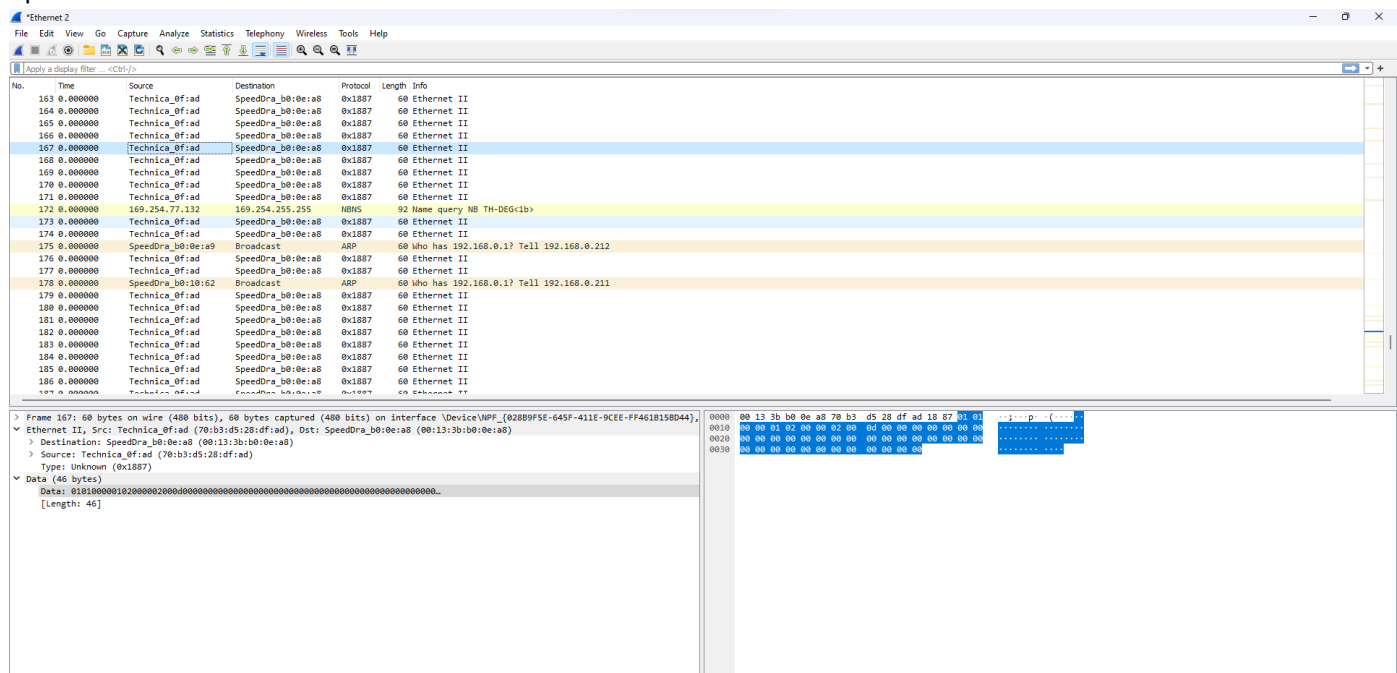
Data: 00 0f

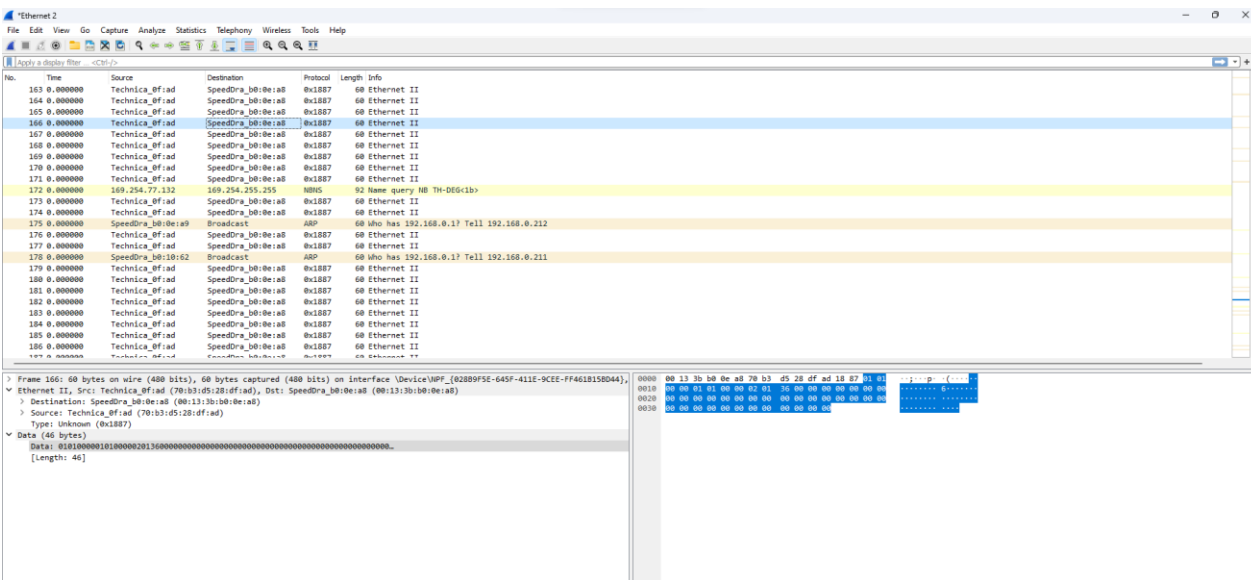
Here 0x0f is the rpm with which the motor is running. This rpm is given as the output from the motor.

Wireshark output:

traffic can be viewed in wireshark also, the same can messages can be seen below, with id 0x102 and speed 0x46.

You should configure the Media Gateway and ANDi to display the data traffic on the can connection line during bus operation in the "ANDi traffic viewer" or "Wireshark".





4.6 THE REMAINING BUS SIMULATION BY THE CANOE AND CANCEXL IS STOPPED AND REPLACED BY THE "ANDI TRAFFIC GENERATOR".

The remaining bus simulation by the CANoe and CANcaseXL should then be stopped and replaced by the "ANDi traffic generator". This means that ANDi on PC1 transmits can frames encapsulated in UDP or RAW via the Broadreach line to the can bus.

Take the messages from the database with message ids

0x100: InfoMotor

0x101: MotorControl

0x102: MotorStatus

0x103: RequestInfo

The data in the messages can be edited from below, it means the direction and speed can be given from the below signal and the motor runs accordingly.

Traffic Generator

Message Table:

Message	Channel	Source	Destination	Id	Length	Active	Cycle	Initial De...	Repetition	Count	Time	Connection	Send Now
RequestInfo	MediaGateway			0x103	1	<input type="checkbox"/>		0	0	0			
MotorControl	MediaGateway			0x101	2	<input checked="" type="checkbox"/>	90	0	0	6823	10:56:32.361		
MotorStatus	MediaGateway			0x102	2	<input type="checkbox"/>	90	0	0	290	10:12:35.993		
InfoMotor	MediaGateway			0x100	1	<input type="checkbox"/>		0	0	0			

Signals

Signal Name	Raw Value	Dec	Phys Value	Generator	Unit	Interpretation
SetpointSpeed	66	66		Value		66
Direction	1	1		Value		1

Payload

01 42

Databases

Enter text to search...

- VanControl
 - VanMotor
 - ControlUnit
 - Transmitted
 - RequestInfo
 - MotorControl
 - Received
 - MotorStatus
 - InfoMotor

Properties

Message: MotorControl

Frame: CAN-FD

CAN-FD: False

Message Type: CAN

CAN Header

Type: CAN Frame

Auto Calculate: All

Name: MotorControl

State: SENDING

Database: VanControl

The messages and speed can be viewed in ANDi Traffic Viewer.

Traffic Viewer : VIRTUAL (100) Channel 1 (CAN)

Adapters Stop Restart Auto Scroll Open File Export Displayed Capture Configuration Normal Fixed Dec Hex Reset Window Layout Window Documentation

Filter : <All Messages>

Counter	No	Time	Delta	Vlan	Direction	Bus	Id	Source	Destination	Protocol	Length	Info	Payload
3126	6673	2367.419555	0.000038		Rx	VECTOR_100/CAN_1	257	ControlUnit	VanMotor	CAN	2		81 42
1	165	13.705912	0.000000		Rx	VECTOR_100/CAN_1	259	ControlUnit	VanMotor	CAN	1		
1	1175	120.434565	0.000000		Rx	VECTOR_100/CAN_1	256	VanMotor	ControlUnit	CAN	1		
3545	4574	42.600158	0.009931		Rx	VECTOR_100/CAN_1	256	VanMotor	ControlUnit	CAN	2		
1	6674	2367.509792	0.000000		Rx	VECTOR_100/CAN_1	257	ControlUnit	VanMotor	CAN	2		81 42

> General Information
> CAN Metadata
> Signals: Database: VanControl; Message: MotorControl

00000000 00 00 01 01 02 00 00 01 42 8

In the below messages, payload is 01 34. Here 01 is direction and 34 is the speed of the motor.

Traffic Viewer : Technica Engineering - Automotive Ethernet Switch (70:83:D5:28:DF:AD) channel CAN A

Adapters Stop Restart Auto Scroll Open File Export Displayed Capture Configuration Normal Fixed Hex Reset Window Layout Documentation

Filter : <All Messages>

Counter	No	Time	Delta	Van	Direction	Bus	Id	Source	Destination	Protocol	Length	Info	Payload
1	57	0.000000	0.000000										
838	1209	0.000000	0.000000	Rx	MediaGateway	259	VanMotor	ControlUnit		CAN	1		01 34
795	1209	0.000000	0.000000	Rx	MediaGateway	257	VanMotor	ControlUnit		CAN	2		01 34
1	58	0.000000	0.000000	Rx	MediaGateway	255	VanMotor	ControlUnit		CAN	1		7E

General Information
 Ethernet: Ethernet II, dst: 70:83:D5:28:DF:AD (Technica_ad), src: 00:13:3E:80:10:61 (SpeedTra_b0:10:61)
 Media Gateway CAN
 Signals: Database: VanControl; Message: RequestInfo

00000000 70 83 D5 28 DF AD 00 13 3E 80 10 61 18 87 01 01
 00000010 00 00 01 03 00 00 01 01

Traffic Generator

Ethernet CAN LIN PDU IEEE1722 Delete Message(s) Adapters Database

Add Message

Message	Channel	Source	Destination	Id	Length	Active	Cycle	Initial De...	Repetition	Count	Time	Connection	Send Now
RequestInfo	MediaGateway			0x103	1	<input checked="" type="checkbox"/>		0	0	0	7 11:18:50.882		<input type="checkbox"/>
MotorControl	MediaGateway			0x101	2	<input checked="" type="checkbox"/>		90	0	0	1009 11:24:01.751		<input type="checkbox"/>
MotorStatus	MediaGateway			0x102	2	<input checked="" type="checkbox"/>		90	0	0	1003 11:22:40.881		<input type="checkbox"/>
InfoMotor	MediaGateway			0x100	1	<input checked="" type="checkbox"/>		0	0	0	1 11:22:38.985		<input type="checkbox"/>

Databases

Enter text to search...

Name

- VanControl
 - VanMotor
 - ControlUnit
 - Transmitted
 - RequestInfo
 - MotorControl
 - Received
 - MotorStatus
 - InfoMotor

Properties

Message

Frame MotorControl

CAN-FO False

Message Type CAN

CAN Header

Type CAN Frame

Auto Calculate All

Name MotorControl

State SENDING

Database

Database Name VanControl

Drag and drop messages from database here

Signals

Signal Name	Raw Value	Dec	Phys Value	Generator	Unit	Interpretation
SetpointSpeed	0	0	Value			0
Direction		0	0	Value		0

Payload

01 34

Conclusion

In this summer semester, In Embedded Connectivity course, we have learnt different topics of Connectivity in Embedded systems like CAN, Automotive Ethernet, InCar Communication, Autosar. Apart from this, in lab, we got more in-depth knowledge by working on different lab tasks. All these tasks were important for the whole subject and lab knowledge. In beginning, we learned about all equipment and system by preparatory setup session using nmap scan and Wireshark.

The first task was about basic communication using sending and receiving scripts in ANDi tool which was interacting with loopback adapter. By this task, we got a basic understanding about use of ANDi tool for communication.

In the next second task, we used these scripts for real network communication using Media Gateway. In the next part, we learned about setting up the VLAN with the Media Gateway to create a virtual network within the network. We learned how to set up all switch and port for media gateway, so that we can communicate with device in the VLAN and analyze the recorded the data as well. Apart from this, we connected one of workstation system with a camera which is connected with the Media gateway.

In the third task, we learned more about workstation's network infrastructure. We connected our system with the Central media gateway as well which can be connected from all workstation of lab. In this, we used the central camera of central Media Gateway also. With the proper configuration, we can connect to the headquarters and can analyse the data as well.

In the final project task, which is the most important topic for us, we have learnt CAN over BroadRReach. We have learnt to configure the Media Gateway and ANDi to display the data traffic on the CAN connection line during bus operation in the "ANDi traffic viewer". The remaining bus simulation by the CANoe and CANcaseXL should then be stopped and replaced by the "ANDi traffic generator". This means that ANDi on PC1 transmits can frames encapsulated in UDP or RAW via the Broadreach line to the can bus.

This project we have got the very good understanding of CANoe tool and its working. The theoretical knowledge on CAN is now a hand-on experience with this project. We have learnt the message formats and CAN ID's and Can Frames.

Overall, the lab tasks were very interesting and a good source of improving knowledge for us.

We would to thank Prof. Dr. Ing. Andreas Grzemba for the teachings and materials were provided through the whole course and Ing. Johann Bretzendorfer for his help during lab sessions and project work. University is providing us a great opportunity to work with expensive tools like vector CAN and CANoe Trace, we are really grateful to that. We firmly believe that this course helped us very much to get in depth knowledge about embedded part of automotive field and will help us as a pioneer understanding to pursue scintillating future in same.