

---

## Novel set of wireless CAN tools for vehicle testing

Pawel Jaworski  
Tim Edwards

October 24, 2011

- The CAN tools have been developed with use on the MIRA proving round in mind
- The tools are applicable in any situation requiring:
  - Live monitoring of vehicle CAN
  - Recording CAN data from a moving vehicle using stationary equipment
  - Debugging CAN equipped devices
- Tools are being developed in cooperation with:



- CAN Relay
  - a wireless CAN telemetry tool
- CAN Signal Injection (CSI)
  - a tool to inject or modify CAN messages in real time
- Use case example
  - MIRA proving ground application

---

**Tool 1:  
The CAN Relay – Wireless  
CAN telemetry system**

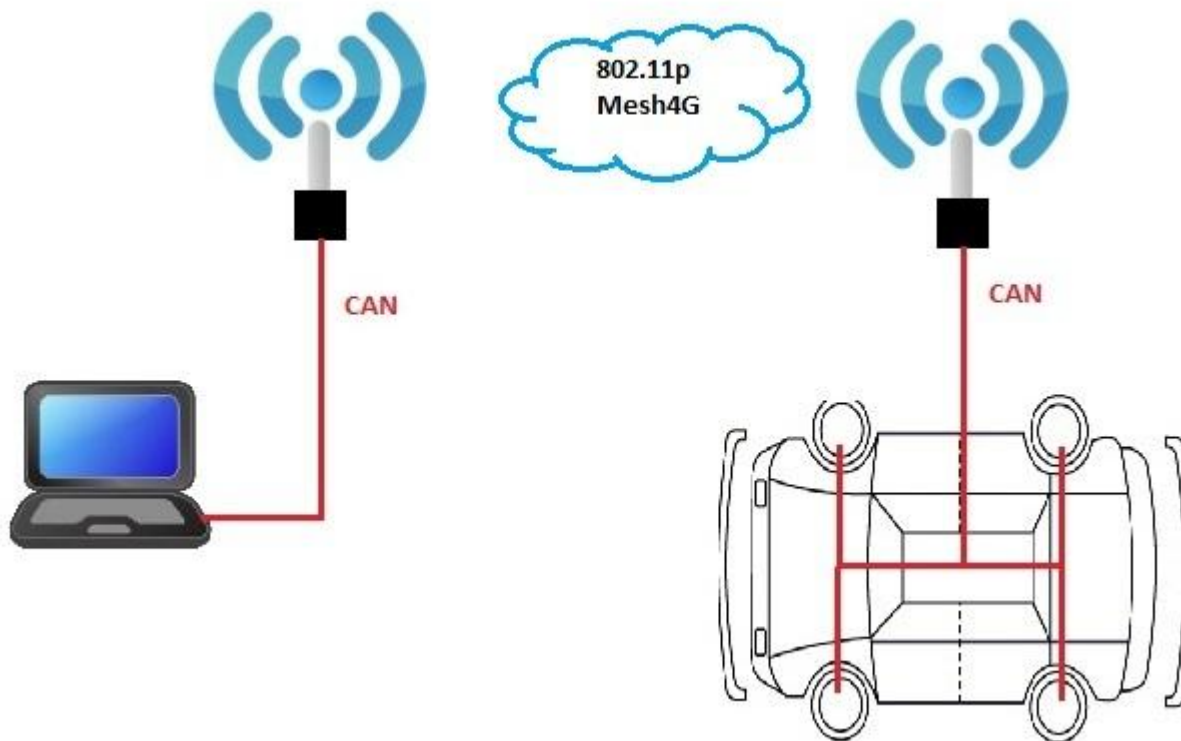


October 24, 2011

# CAN Relay

The purpose of CAN Relay:

- Allow bidirectional CAN communications between a fixed computing station and a moving vehicle



# The CAN Relay System



Base system for CAN Relay comprises of:

- Hardware:
  - Lippert Cool FrontRunner single board (PC104) computer
  - PEAK System – PC104 CAN adapter
- Software:
  - Linux OS with SOCKET-CAN communication stack
  - CAN Relay application



# CAN Relay Application

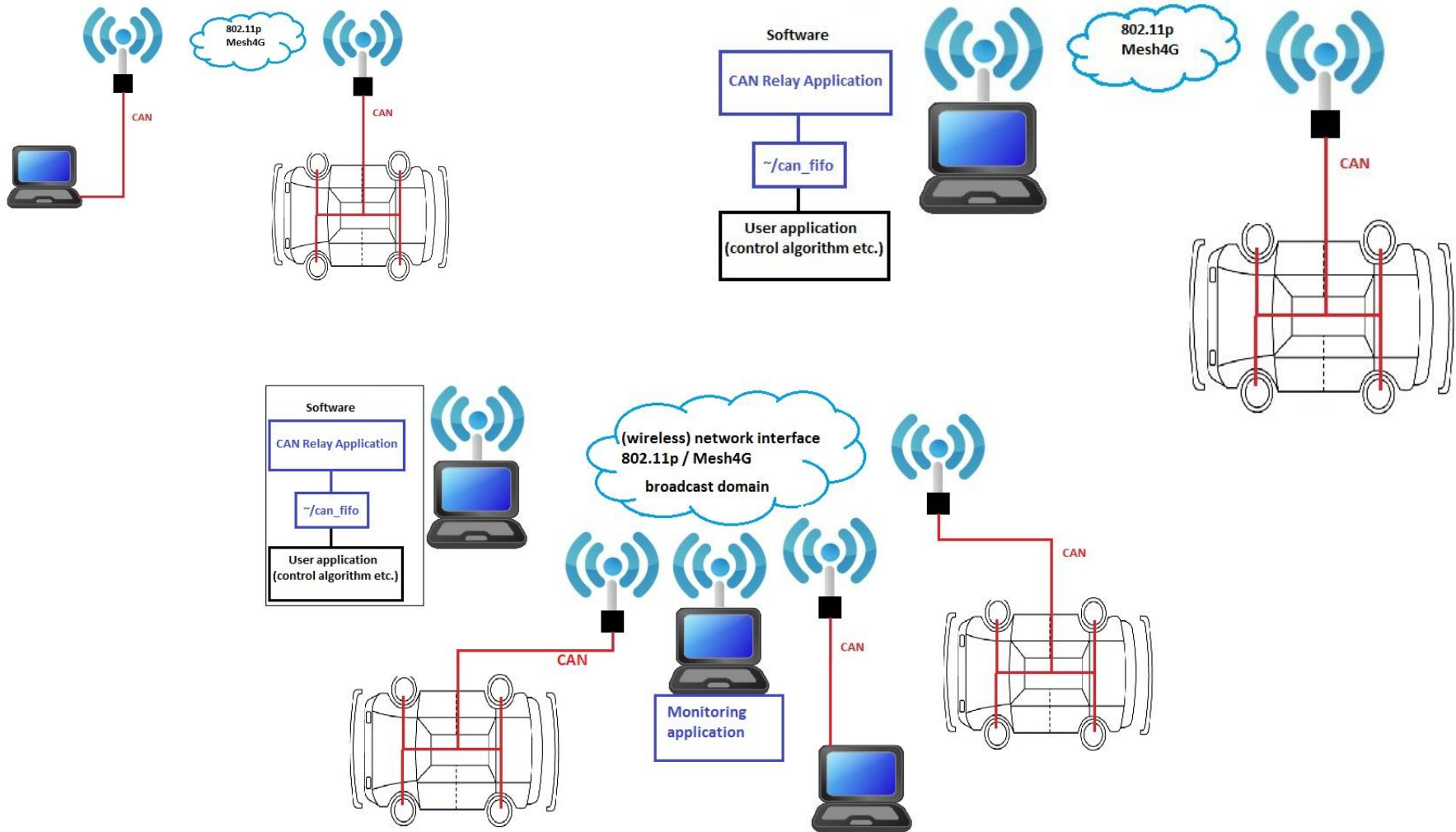
---



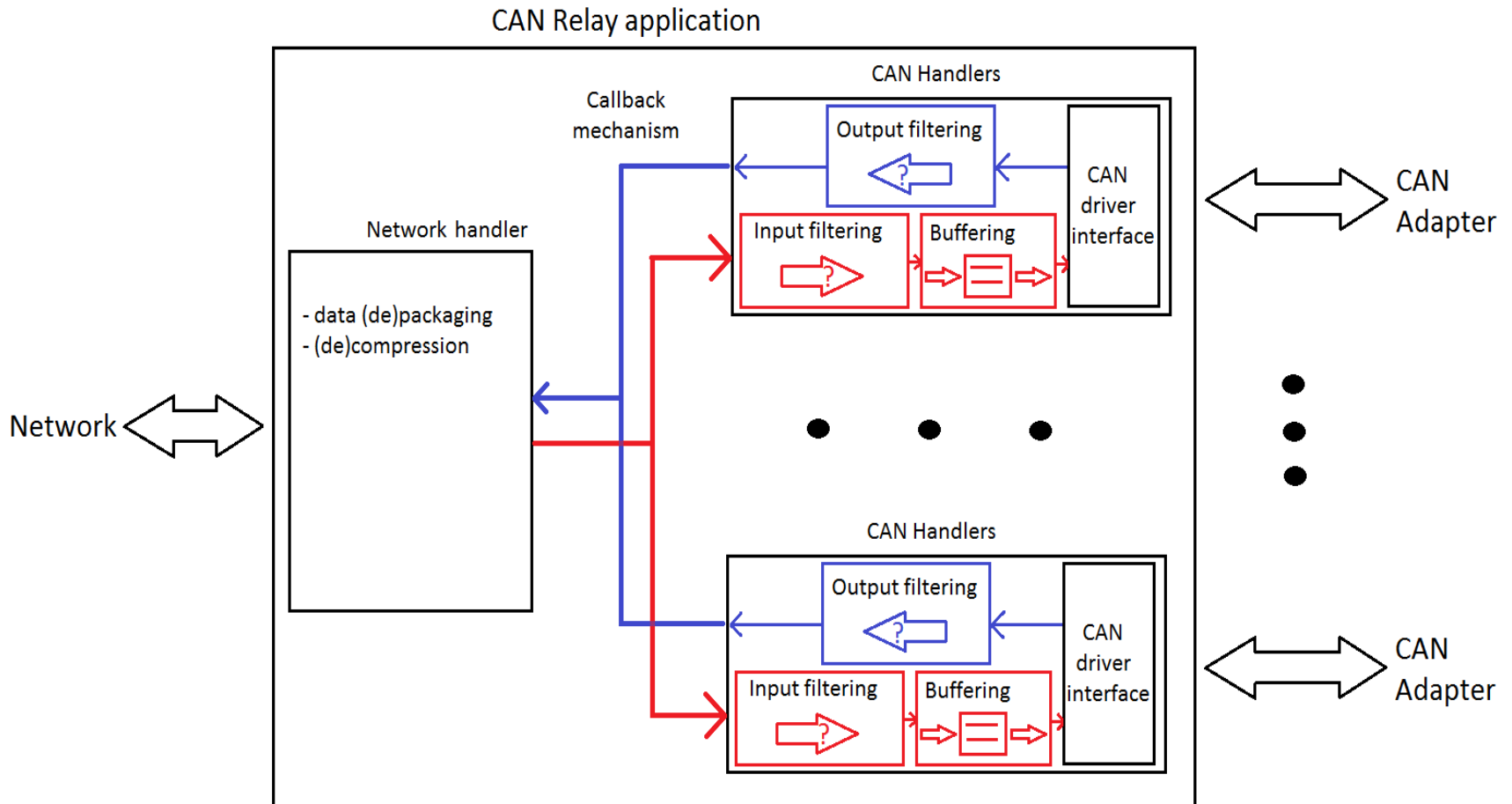
OS specific CAN interface types in use:

- Socket CAN
  - Generic, open source CAN driver interface
- Character device interface
  - “PEAK system” character device CAN driver interface
- Virtual interfaces
  - Linux FIFO queues
  - Text files (output only)

# CAN Relay Configurations



# CAN Relay Software Component



# CAN Relay protocol

---

CAN messages are encapsulated in CAN Relay protocol while being transported

CAN Relay message format:

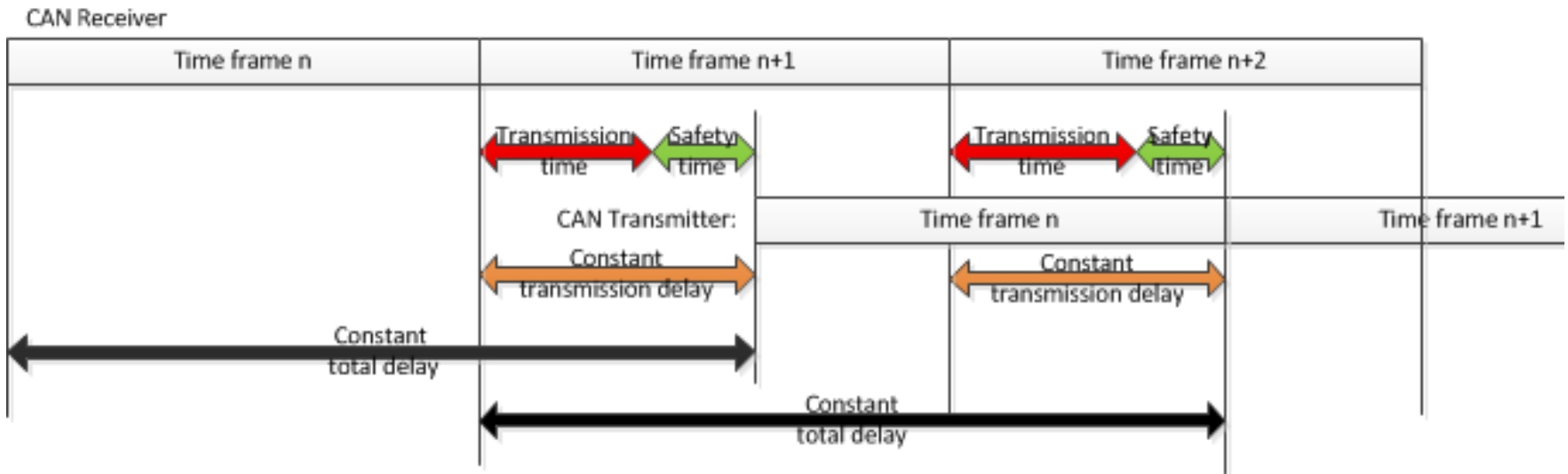


- CAN message:
  - Header:
    - message ID
    - control data
    - time offset (optional)
  - CAN data fields
- Transport protocol:
  - UDP header
  - IP header
- Medium specific header (e.g. Ethernet)

## Operating scenarios:

- It is required to send the CAN data through as fast as possible
  - **Live mode** – CAN messages are forwarded the moment they are received
    - Minimal possible latency
    - Inter-message timings may not be kept
- It is required to preserve inter-message timings
  - **Buffered mode** – CAN messages are buffered
    - Significant but known latency
    - Inter-message timings kept (up to certain degree)

# CAN Relay buffered mode



Assumption:

- The interconnecting network's speed is significantly larger than CAN speed

# CAN Relay Application: Filtering

---



CAN Relay supports message filtering

- Currently by message ID
- Content (byte mask) based filtering planned

Filtering rules:

- Are defined for each CAN interface
- Are defined separately for both transmission directions
- Consist of a default filtering policy and exception list
- Are stored in a text file
- Can be reconfigured in run-time

# CAN Relay Application: Filtering



```
#can relay config sample 1  
[can0]  
in deny  
0x25  
128  
  
out allow  
0x10  
9
```

## Example configuration 1

- One interface with specific filtering rules
  - Accept only specified messages
  - Send all but those specified

```
#can relay config sample 2  
[/mira/fifo_out]  
in deny  
out allow  
  
[/mira/fifo_in]  
in allow  
out deny
```

## Example configuration 2

- Effectively one virtual CAN
- Two FIFO queues
  - One for input and one for output

# CAN Relay Application: Multiplexing and de-multiplexing

---



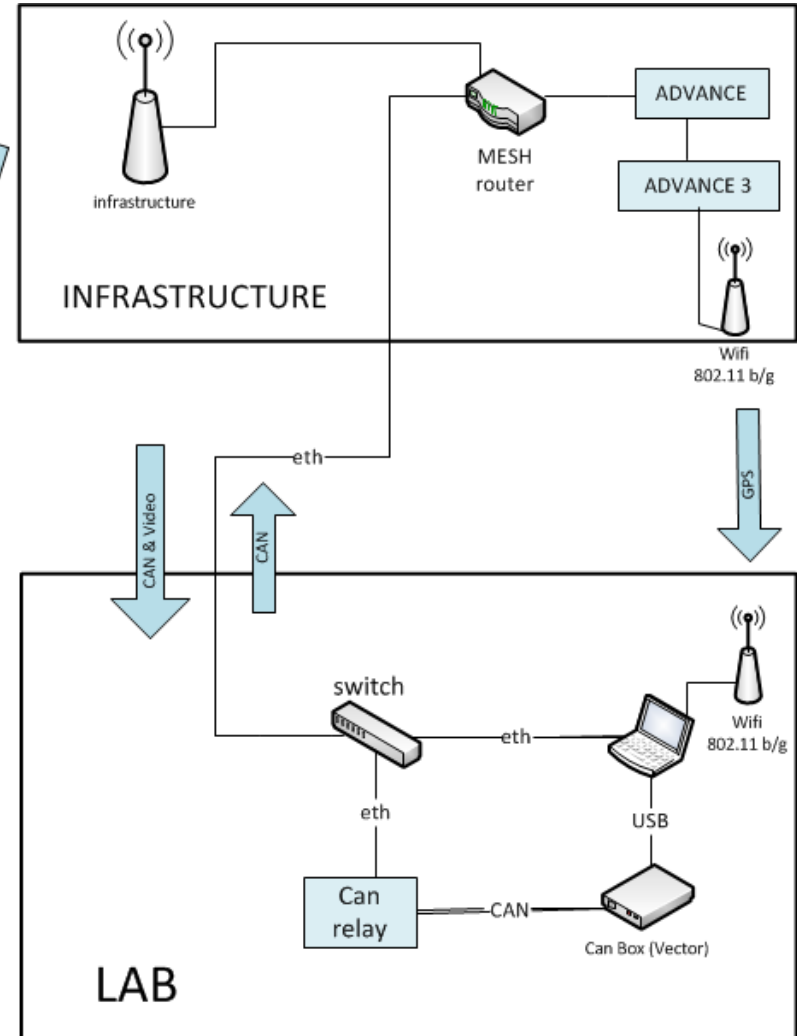
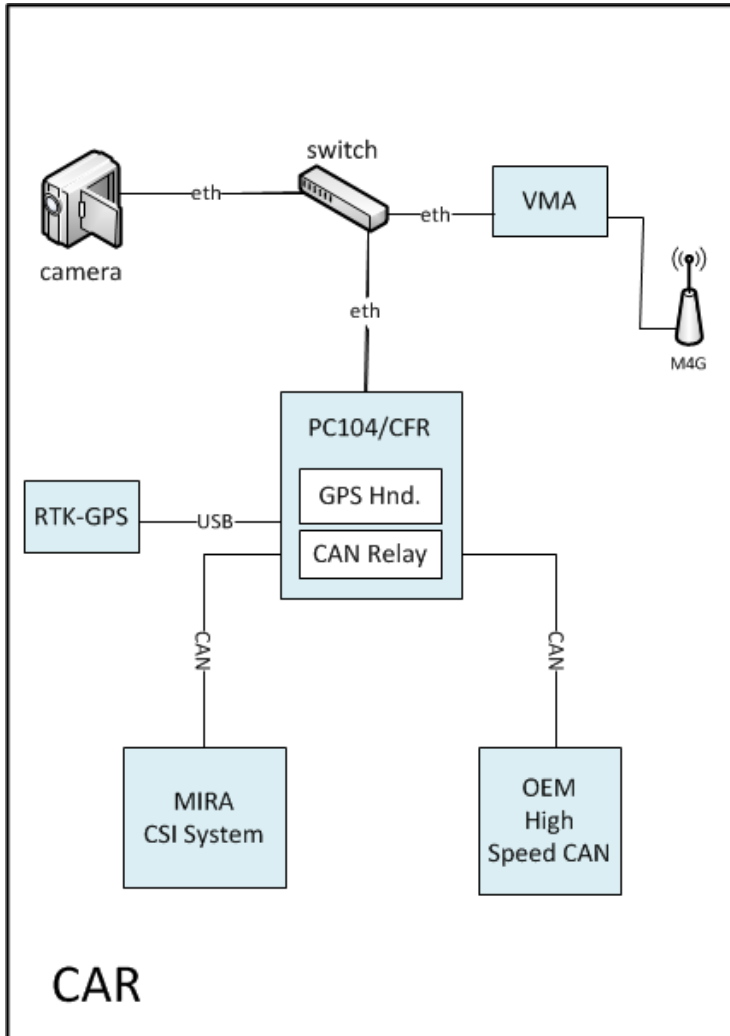
A form of CAN multiplexing can be achieved using the CAN Relay

- Each CAN interface (physical and virtual) is handled by one instance of CAN Relay Application.
- Each instance of CAN Relay Application can handle:
  - Several CAN interfaces
  - One network link to remote instance of CAN Relay Application
- Each system platform can have several instances of CAN Relay Application

Currently limited message routing can be achieved by combining CAN Relay application instances with appropriate filtering rules.

It is planned to expand the CAN Relay protocol by adding message and interface tagging that will allow for more flexible routing.

# CAN Relay use case

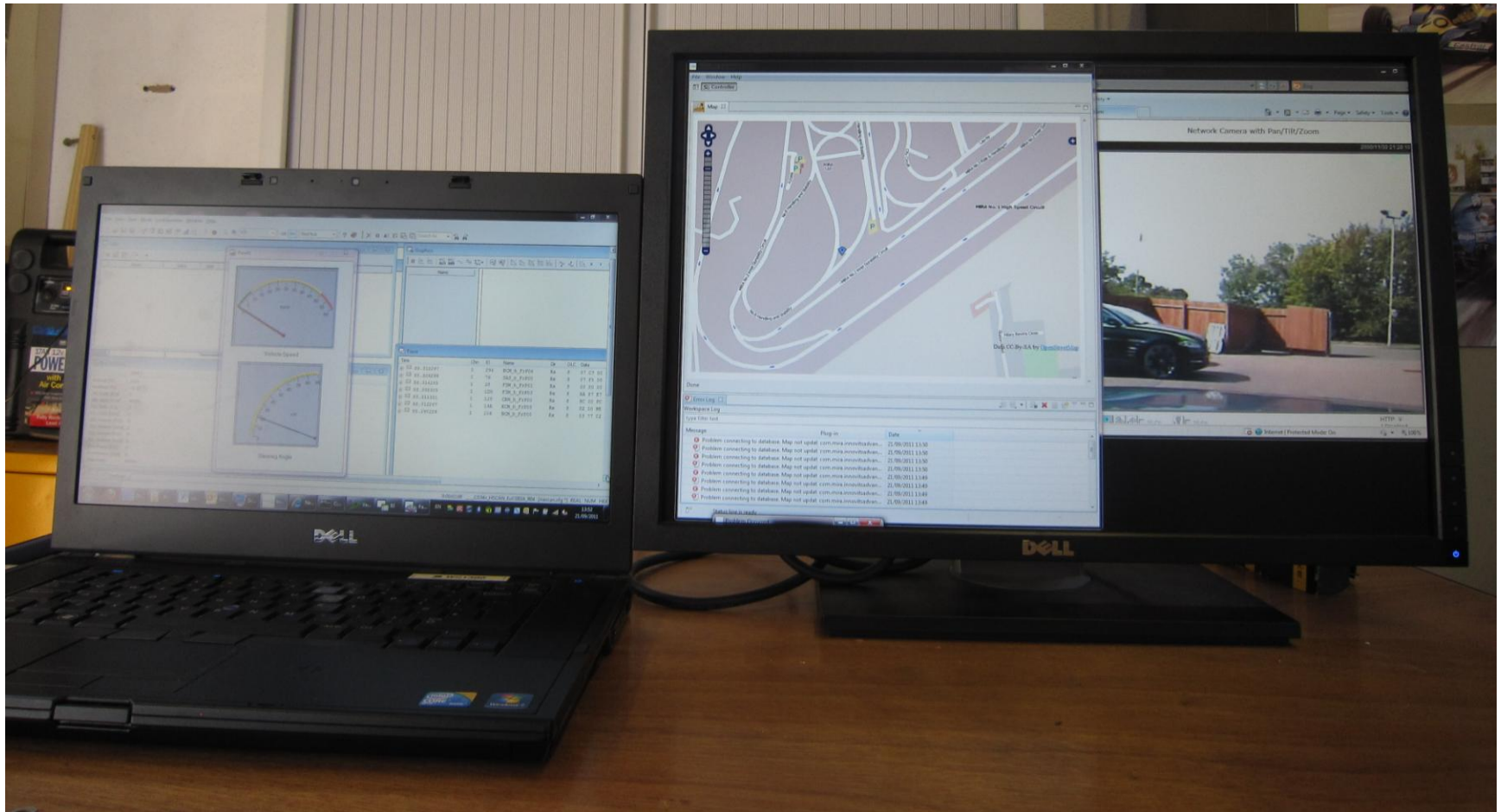




# CAN Relay use case example



## Monitoring station view



# CAN Relay Application:

---

Future development of CAN Relay:

- Implementation of additional features
  - Content based filtering
  - Buffering mechanism
  - More effective routing
    - Message and interface tagging
  - ...
- Migration to a dedicated real time system

---

## CAN Signal Injection System



October 24, 2011

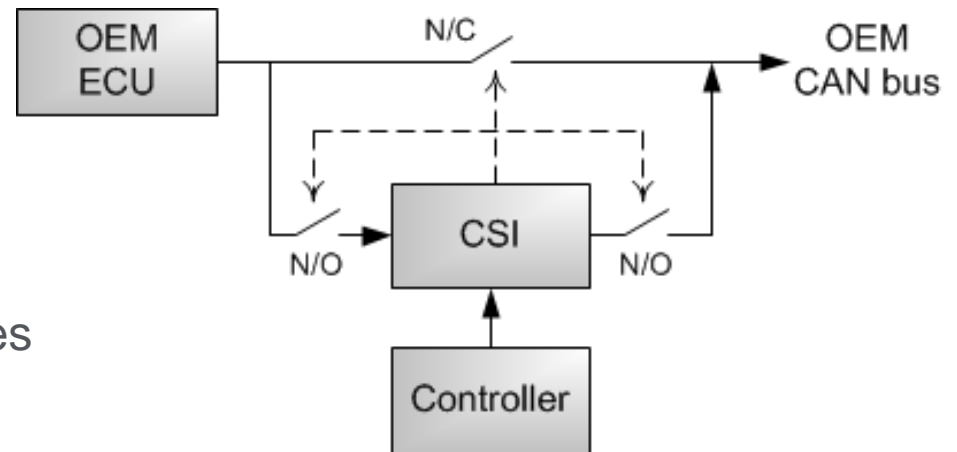
# CAN Signal Injection System

The purpose of CAN Signal Injection System (CSI):

- Inject and modify messages on a live CAN bus

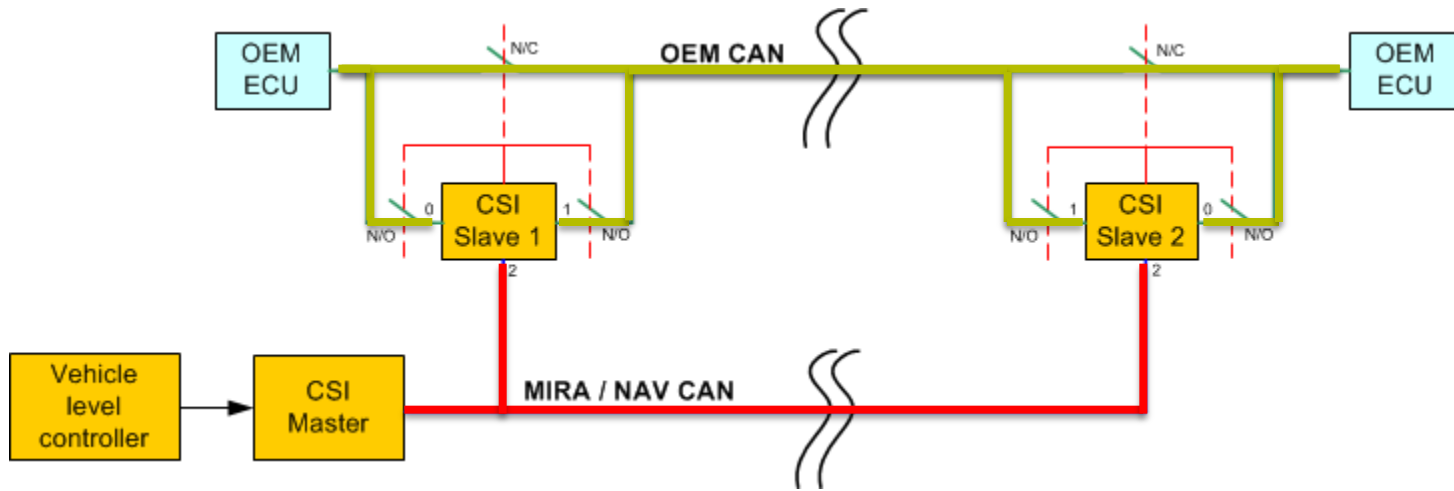
The CSI system is comprised of:

- CSI Master (controller)
  - Configures the slave modules
- CSI Slave(s)
  - Modifies messages on the CAN bus



# The CAN Message Injection System

CSI can have multiple injection points and works in three operating modes:



## Offline:

- OEM CAN is uninterrupted

## Gateway mode:

- CAN messages go through the CSI but are not modified

## Online:

- Selected messages are modified

# The CAN Message Injection System



## Offline mode:

- The target CAN bus is connected directly

## Gateway mode:

- The target CAN bus is connected through the CSI
- There are no message modifications done
- Infineon XE164 has a hardware implementation of gateway mode

## Online mode:

- Outbound messages are modified according to specified demand



## Mode switch:

- The bus is interrupted while the relay switches



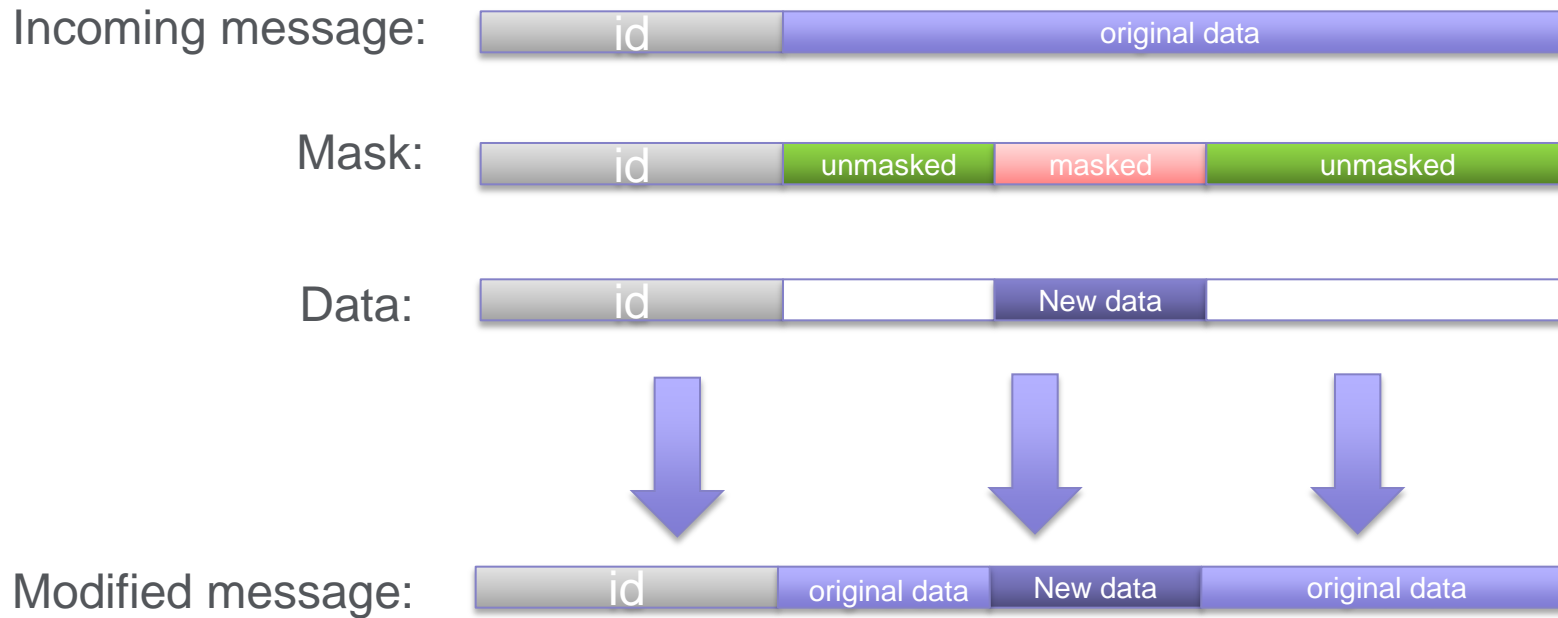
## Mode switch:

- Software only switch, no physical interruption of the CAN bus

# The CAN Message Injection System



Message modification mechanism:



# CAN Signal Injection System

---



CSI Software uses time triggered scheduling

Time triggered scheduling:

- Predictable reaction (transmission) time
- Easier safety case and analysis
- Predictable system behaviour

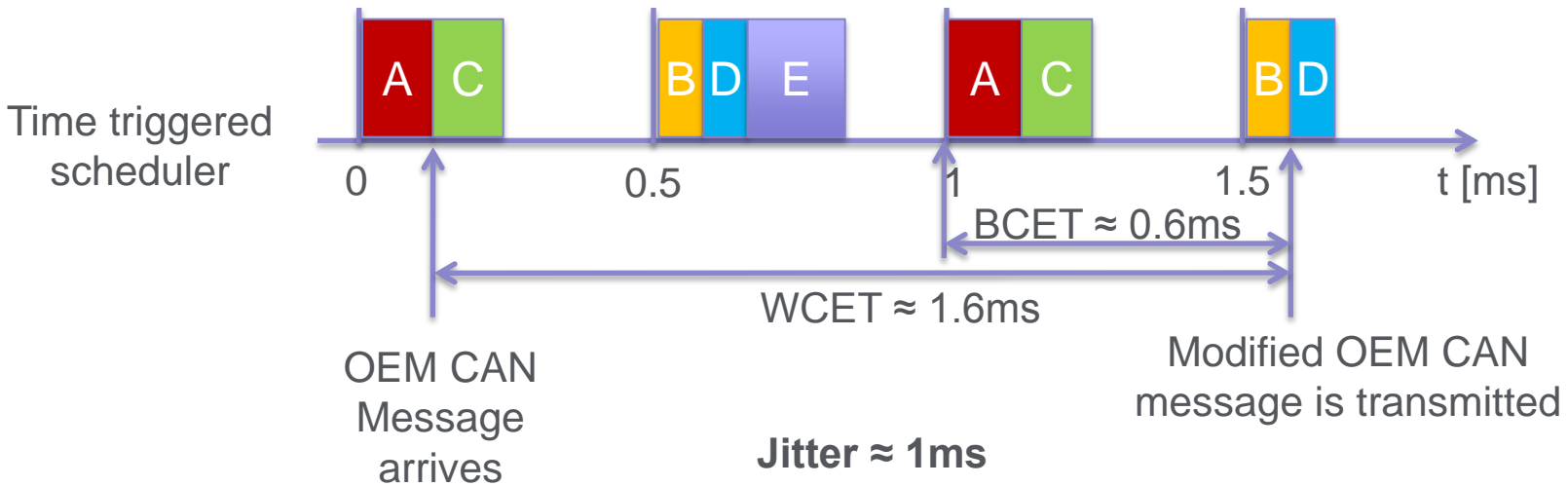
Event triggered scheduling:

- Very fast reaction time
- Potentially unpredictable system behaviour

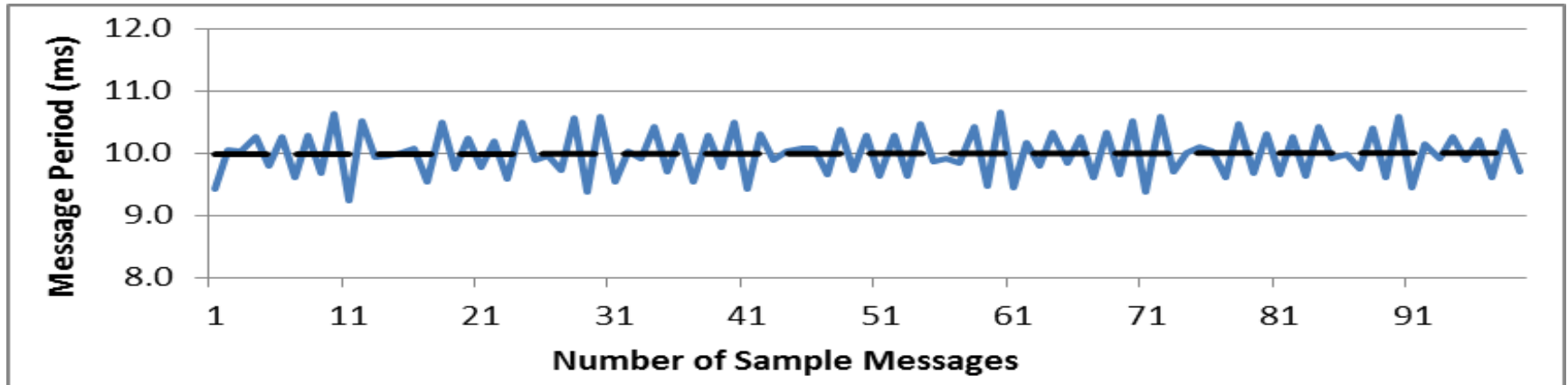
# CAN Signal Injection System: time triggered task scheduling



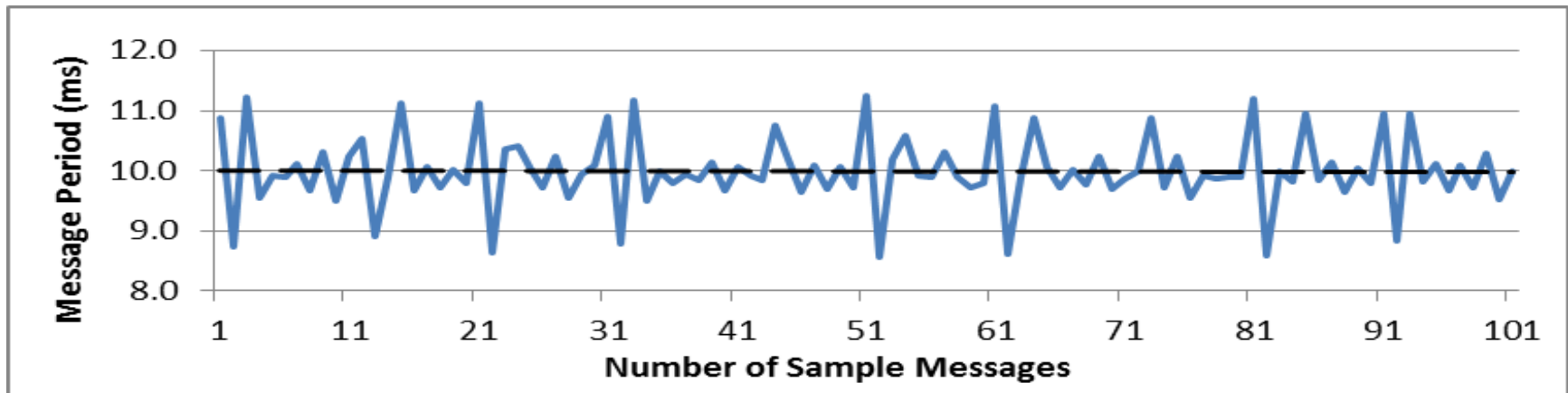
Time triggered tasks:



# CAN Signal Injection System



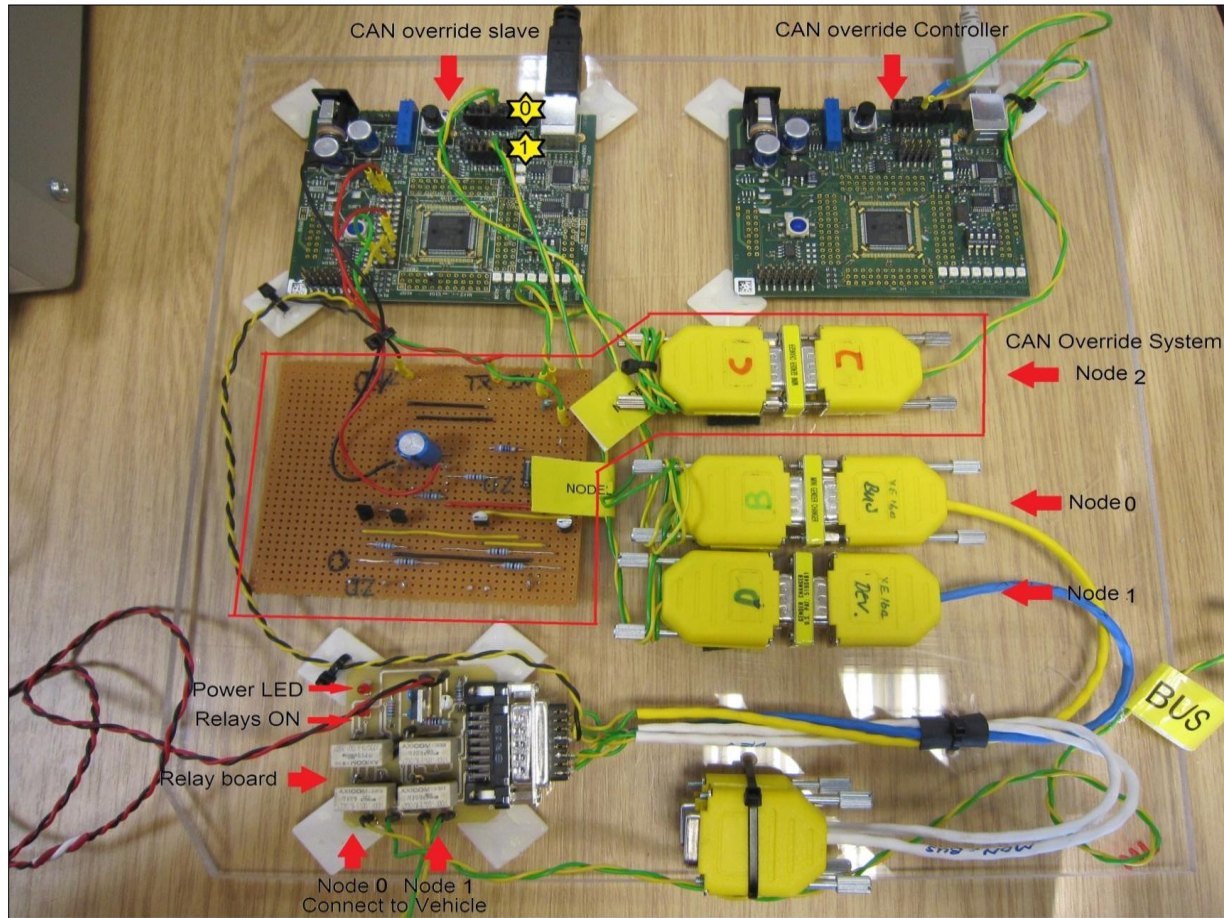
Unmodified data sample



Online mode – Messages being modified

# The CAN Message Injection System:

## The CSI prototype



Future development of CSI:

- Integration with CAN DB
- Development of graphical user interface
  - To make signal changes/injections easier
- Further miniaturisation (single PCB)

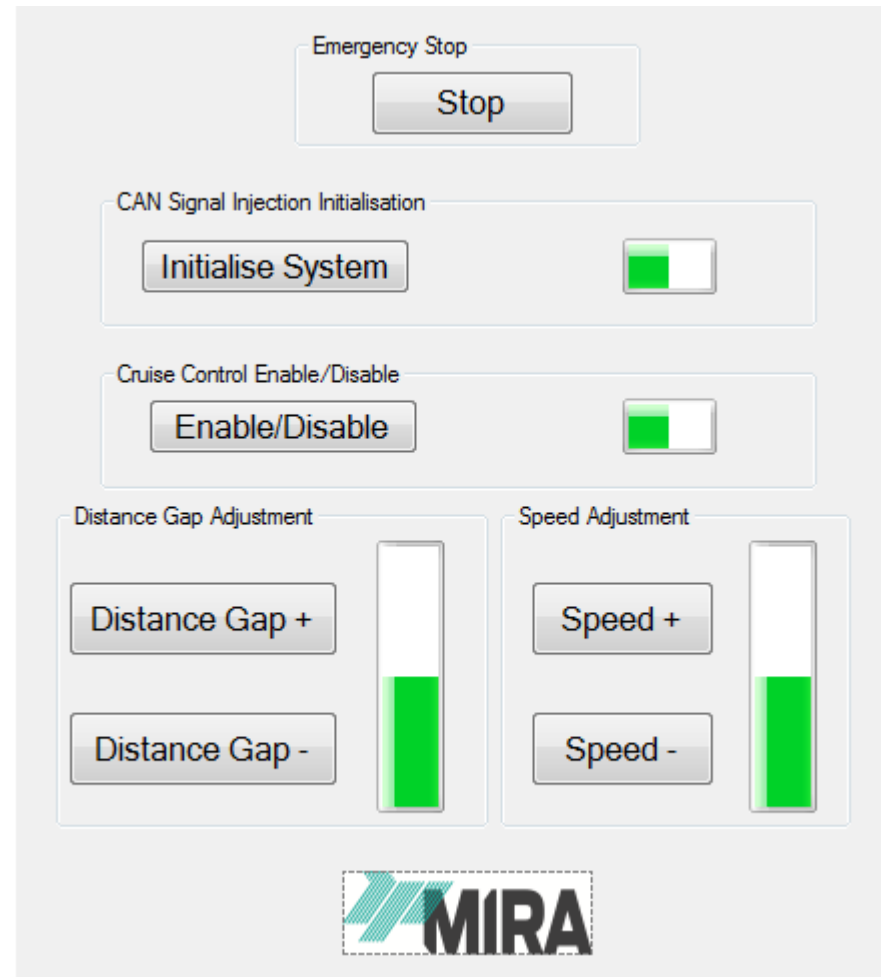
# CAN Signal Injection System

## Use case example



Control of an adaptive cruise control system

- Override appropriate signals on vehicle's CAN bus to enforce requested behaviour
- Remotely using CAN Relay



# Conclusions

---

Both CAN Relay and CAN Signal injection systems:

- Have been designed to work together
  - E.g. remote control of vehicle functions
- Are useful on their own
  - CAN Relay – vehicle in motion telemetry
    - From multiple vehicles simultaneously
    - Using multiple measurement/recording tools
    - Live demonstrations
  - CAN Signal Injection
    - CAN Debugging
    - CAN Testing
    - *Vehicle control override*

# Contact Details



**Pawel Jaworski**  
MSc.  
PhD Student

MIRA Ltd  
Watling Street,  
Nuneaton, Warwickshire,  
CV10 0TU, UK

Direct T: +44 (0)24 7635 5221  
E: pawel.jaworski@mira.co.uk

T: +44 (0)24 7635 5000  
F: +44 (0)24 7635 8000

[www.mira.co.uk](http://www.mira.co.uk)



**Tim Edwards**  
BEng (Hons), Mphil, MIET  
Senior Engineer  
Advanced Technologies

MIRA Ltd  
Watling Street,  
Nuneaton, Warwickshire,  
CV10 0TU, UK

Direct T: +44 (0)24 7635 5484  
E: tim.edwardsi@mira.co.uk

T: +44 (0)24 7635 5000  
F: +44 (0)24 7635 8000

[www.mira.co.uk](http://www.mira.co.uk)