TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology

IE70LT

Henri Vennikas 153759IVEM

UPGRADING AUTOMOTIVE ELECTRONICS STUDY EQUIPMENT AT TTK UAS

Master's thesis

Supervisor: Reeno Reeder
Phd
Associate Professor

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

IE70LT

Henri Vennikas 153759IVEM

AUTOELEKTROONIKA ÕPPESEADMETE KAASAJASTMINE TTK-S

Magistritöö

Juhendaja: Reeno Reeder
doktori kraad
Dotsent

Tallinn 2017

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Henri Vennikas

15.05.2017

Abstract

Author develops minimum viable product and conducts feasibility tests for upgrading automotive study equipment for automotive engineering curricula at TKK University of Applied Sciences. In the first part of thesis, overview is given about state of art in automotive electronics and respective education in TTK UAS.

In the second half of thesis existing study equipment is mapped and more complex signals are modelled. New ECU platforms are proposed to fulfil high-level requirements.

A selection of the signals is included in the feasibility tests conducted with widespread development boards. The results indicate feasibility of upgrading existing equipment in proposed manner.

This thesis is written in English and is 69 pages long, including 2 chapters, 43 figures and 19 tables.

Annotatsioon

Autoelektroonika õppeseadmete kaasajastamine TTK-s

Autor uurib olemasolevate autoelektroonika õppeseadmete kaasajastamise võimalusi.

Esimeses osas kajastatakse modernse sõiduki elektroonika osakaalu ja tähtsust ning hetkeolukorda vastava hariduse valdkonnas Tallinna Tehnikakõrgkoolis.

Töö teises pooles seadmed kaardistatakse, keerulisemad signaalid modelleeritakse ning pakutakse välja uute elektrooniliste juhtmoodulite platvormide arhitektuurid.

Valitud signaalide põhjal prototüübitakse signaalide mõõtmine ja edastamine üle võrgu, et uurida väljapakutud lahenduste teostatavust.

Lõputöö on kirjutatud Inglise keeles ning sisaldab teksti 69 leheküljel, 2 peatükki, 43 joonist, 19 tabelit.

List of abbreviations and terms

ADC	Analog to Digital Converter
CAN	Controller Area Network
CRD	Common Rail Diesel injection system
ECTS	European Credit Transfer and Accumulation System
ECU	Electronic Control Unit
EFI	Electronic Fuel Injecton (gasoline engine)
HMI	Human Machine Interface
HW	Hardware
IAV	Instantaneous Angular Velocity
ICE	Internal Combustion Engine
ISO	International Organization of Standardization
LED	Light Emitting Diode
LIN	Local Interconnect Network
LN	Lucas Nülle
MAF	Mass Air Flow sensor
MLF	Malfunction
MVP	Minimum Viable Product
NTC	Negative Temperature Coefficient thermistor
OEM	Original Equipment Manufacturer
PC	Personal Computer
PTC	Positive Temperature Coefficient thermistor
RPM	Revolutions Per Minute
SS	Sum of Squares
SW	Software
TDC	Top Dead Centre
TPS	Throttle Position Sensor
TTK UAS	Tallinn University of Applied Sciences
ZC	Zero Crossing

Table of contents

List of figures

List of tables

1 Introduction

Last significant change of mode of road transport was when humans transcended from using beast of burden as main means of powering road transportation vehicles. For the past century convention of automotive industry has had little change. Passenger car has well kept its concept. There have been incremental increases in safety and efficiency due to advances in electronic systems combining sensors, control modules and actuators and CAD enabled design and simulation techniques.

First ECUs were developed five decades ago. With the addition of networking the number of digital signals during normal vehicle operation has grown over couple of thousands where the transmitting and receiving nodes can have tens up to several hundred neighbors divided in several different buses connected through centralized gateway. [1, p. vii], [2, pp. 1090-1091]

A great fundamental change in terms of road transport systems and vehicles is imminent. First of all, phasing out from use of internal combustion engines. But secondly, and more importantly, the car will drive by its self fully autonomously.

This fundamental change is allowed by complex electronic systems which by now are omnipresent in transportation systems and vehicles.

The impact of rapidly emerging autonomous vehicle technologies also shape the future course of industry. Increased annual revenue is reported by PWC to grow up to 1.56 times by 2030 from 5 to 7.8 billion EUR where about third of the revenue and profits will be gained from mobility services, digital services and new technologies. [3, pp. 12-14]

Moreover, finally environmental impact of the car can be more sustained. Due to the possibilities lying in autonomous ride sharing, need for ownership will be decreasing, resulting in less new cars bought. As the author has stated previously, environmentally friendly car is the one which is never produced. [4]

To face the new reality of automotive engineering it is vital to increase emphasis on state of art education regarding complex electronic systems.

Automotive engineering chair of TTK has purchased study equipment for automotive electronics and electrics 10 years ago for approx. 75,000 €. ECUs and PC software of the equipment have since outdated, are not compatible with newer PC operating systems and are not updateable. Also existing equipment's main purpose is to teach basics of vehicle systems building blocks and operating principles not design aspects of electronic systems.

As a part of developing automotive electronics speciality, as illustrated on Figure 1 in the green field, author proposes to use existing equipment for developing minimum viable product for teaching automotive electronic systems and networking design with the possibility to assess and model existing sensors and actuators and generate basic control algorithms and software and test the results in classroom.

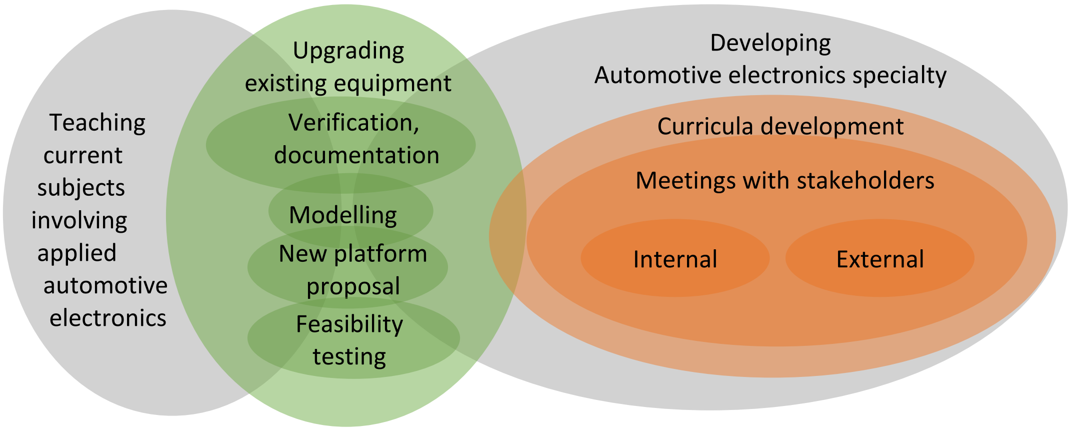

Teaching current subjects involving applied automotive electronics
Upgrading existing equipment
Verification, documentation
Modelling
New platform proposal
Feasibility testing
Developing Automotive electronics specialty
Curricula development
Meetings with stakeholders
Internal
External

Figure 1. Venn-like diagram showing relationships between selected activities relevant to this thesis

The scope of the practical task can be seen on Figure 2. The three domains have each their own I/O and requirements that need to be fulfilled. To be able to teach networking of modules it is a necessity that the modules would have a digital communication channel. High-Speed CAN will be used as it is currently the main backbone network of vehicles. Other communication protocols such as LIN, Flexray and Ethernet can be considered in the future developments when relevant equipment is used.

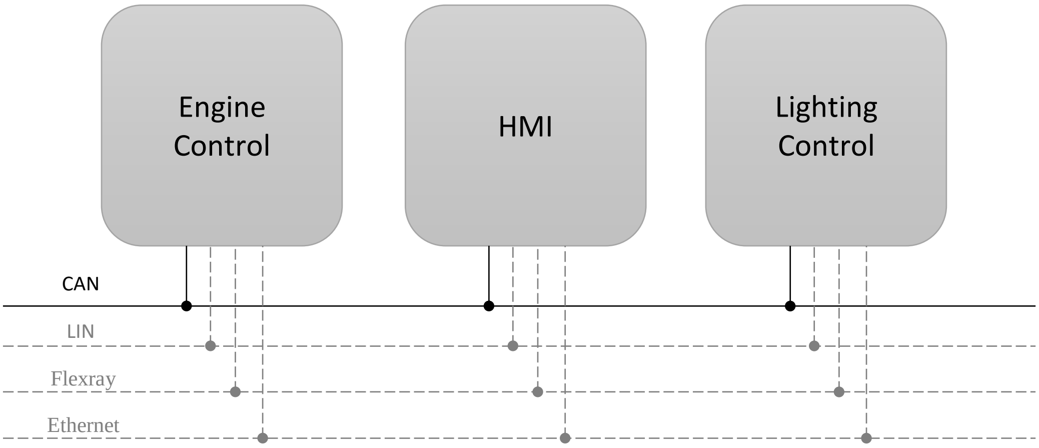


Figure 2. Architecture of planned minimum viable product teaching platform.

Minimum viable product is known from LEAN development processes to implement ideas rapidly and iterate development process over time, it is low-cost but has high impact in general development process of the specialty. Future specialty development efforts will include focus on current and developing trends in the automotive systems (Advanced drivers assistance systems, secure body electronics, hybrid and electric drives, autonomous driving etc.) MVP approach to first develop a small scale case study based on existing equipment will allow to do assumption testing, rapid prototyping and learning launches. [5, pp. 98-99]

1.1 Motivation

The author has taken his mission to offer extended choice of specialization towards automotive electronics for the TTK UAS Automotive Engineering students as soon as possible. Another aspect of developing education involving information and communication technology is to accelerate local industry.

External industry representatives have indicated lack of engineering graduates who have understanding of automotive engineering and electronic systems competences specific to the industry.

Regional employers will have access to specialized engineers with understanding of automotive principles while also equipped with knowledge, skills and competences to start career in the field of working with automotive electronic systems. Also the future graduate having innovative ideas can start their own venture.

It has become general view of internally involved parties that Automotive Engineering curricula needs updates, both from the management level and faculty staff members. [6, p. 63]

1.2 Overview of Automotive Engineering Curricula

Automotive Engineering curricula is based on the Occupational Qualification Standard: Automotive Engineer, level 6 issued by Estonian Qualification Authority. [7] Curricula has been internationally accredited in year 2015. Study program is fulfilled during four years of studies by taking courses and passing practical trainings for 240 ECTS. First half of the studies focus on basics of higher education with little emphasis on speciality. On the second half most of the studies consist of speciality subjects. Graduate is awarded with diploma of professional higher education. As seen from Table 1 the speciality studies make up around 1/5th of the total studies. [8]

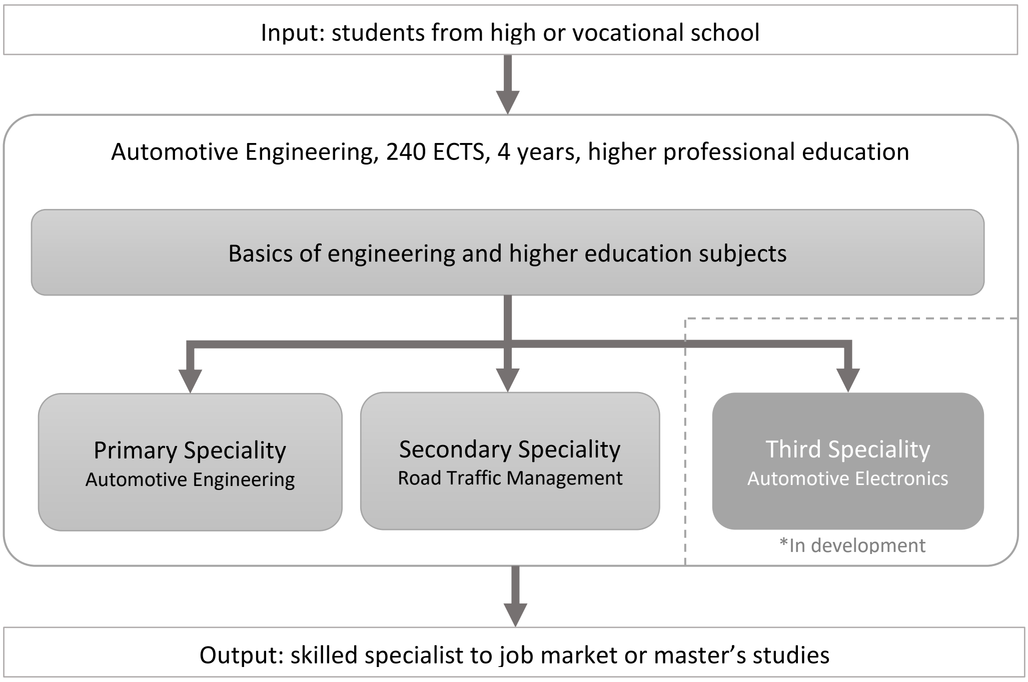

Input: students from high or vocational school
Automotive Engineering, 240 ECTS, 4 years, higher professional education
Basics of engineering and higher education subjects
Primary Speciality
Automotive Engineering
Secondary Speciality
Road Traffic Management
Third Speciality
Automotive Electronics
*In development
Output: skilled specialist to job market or master's studies

Figure 3. Generalization of Automotive Engineering curricula modules.

As currently there are only two options available for specialization: Automotive Engineering and Road Traffic Management, the students get quite broad overview of automotive engineering aspects. After graduating most of the students start their career in local car dealerships and workshops in middle management level. Few of the students continue with master's studies. Structure of the studies can be seen in generalized form on Figure 3. Out of graduates, from year 2015, 98% are working out of whom 73% work in the specialized field. [9]

Table 1. Overview of Automotive Engineering 2016 curricula modules.

Automotive Engineering 240 ECTS		
Module	**Mandatory Subjects ECTS**	**Optional Subjects ECTS**
Humanities	6	51
Mathematics, Sciences and Graphics	42	0
Economics and Entrepeneurship	6	0
Engineering Core Subjects. Speciality Subjects 1	33	0
Automotive Design. Speciality Subjects 2	24	0
Primary Speciality: Automotive Engineering. Speciality Subjects 3	45	0
Secondary Speciality: Road Traffic Management. Speciality Subjects 3	39	12
***Third Speciality**: Automotive Electronics* [1]	*Approx. 45-55*	
Practical Training	36	0
Working Life	15	6
Graduation Thesis	15	0

1.3 State of art of automotive electronics

The term megatechnology is used to describe vehicles due to the fact that many technologies must converge in one unified application with many different functions and requirements. It can be generally applied that automotive electronic systems have developed hand in hand with advances in the electronic industry. From 1950s with the first emergence of semiconductor applications, the number of transistors on integrated chip has grown exponentially. [10, p. 2]

[1] Under development.

1.3.1 Electronic systems architecture

Ribbens describes system design with block diagrams with three major distinctions of intended applications to: control, measure or communicate. [11, p. 3]

Automotive companies and OEMs divide automotive electronic systems between functional domains such as: [1, p. 4] [12, p. 10]

- Engine Management
- Drivetrain control
- Chassis control
- Safety systems (active, passive)
- Comfort convenience
- Multimedia and telematics
- Human-machine interface

Generalized ECU architecture can be seen on Figure 4 which is applied in all above domains for the tens up to a hundred ECUs found on board a vehicle.

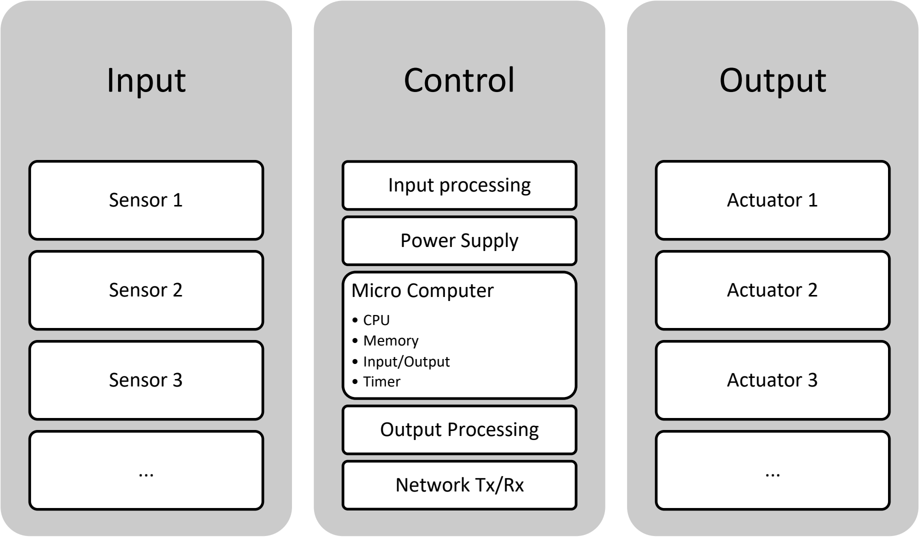


Figure 4. Common ECU architecture.

1.3.2 Requirements

A certain set of common underlying principles are followed to ensure safety and durability of automotive systems. Most of the legislative requirements are evolving over time as does technology, an example of this are progressive exhaust gas emissions restrictions which are only met due to complex electronic control of powertrain.

Requirements can be also seemingly met by using the automotive embedded electronic systems to alter standard test measurements by recognizing test conditions and using separate control algorithm yielding in lower exhaust gas emissions during the test. [13]

Additionally, due to advances in communication aspects of automotive systems there are new challenges to be solved. There are numerous powerful demonstrations of vulnerabilities in networking of embedded systems. Thus meeting the safety and real time operation requirements presents higher demands against external intrusions, higher than most of the existing systems possess today. [14] [15] [16]

To ensure uniform requirements for developing and testing automotive systems, ISO has developed over 800 standards for road vehicles. Which demonstrates the wealth of documentation that may have to be considered in the electronic equipment development process. [17, p. 8] [18]

Failure Mode Effects Analysis (FMEA) and Automotive Safety Integrity Level (ASIL) analyses are said to be essential part of electronic system design.

Another relevant standard is ISO/TS 16949:2009 which for describing quality system requirements among automotive manufacturers is generally adopted throughout the industry to ensure common understanding of quality management and reduce duplication of adopted standards in different regions. It will be revised and launched under the name IATF 16949:2009 in September 2018. [19, p. 3]

Most of the new systems developed on vehicles also adhere to Automotive Open System Architecture (AUTOSAR), which provides global approach towards making it available application development regardless of manufacturer or HW used. [20]

For any automotive system (electronic or not), the amount of documentation to consider is overwhelming, but the project documentation and product documentation still has to be created, meaning a great deal of development effort has to be considered for managing documentation.

1.4 Development process

The V-shaped development model incorporates requirement analysis, system design, component design at first hand. Then it sums up to final product via component testing, system testing and acceptance tests. [12, pp. 166-167]

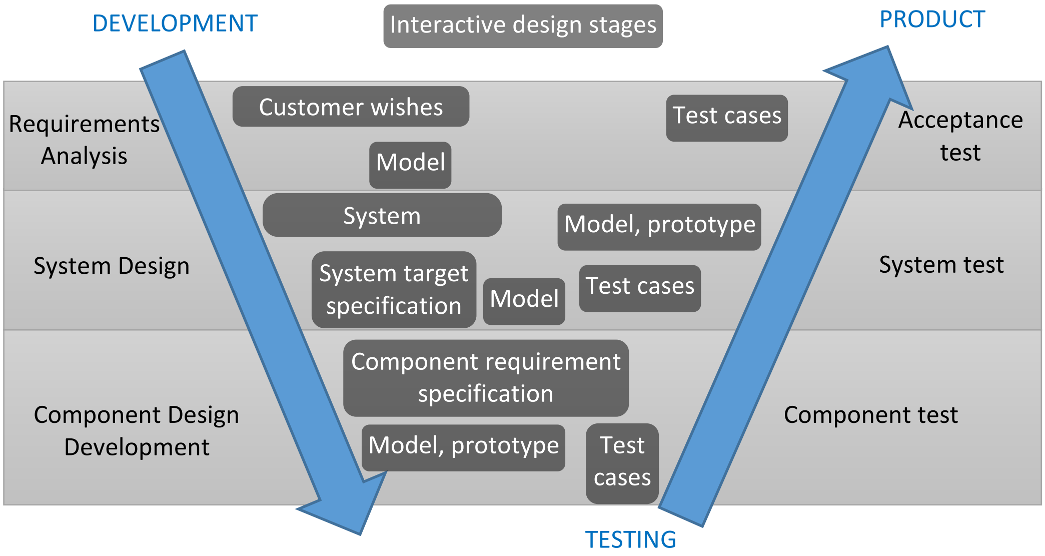


Figure 5. V-shaped development model

Throughout this thesis minimum viable product (MVP) is developed keeping in mind LEAN mindset, but following also conventional V-shaped development process as seen on Figure 5. The requirements are described in paragraph 1.4.1, system design is implemented in paragraph 2.2.

Component design has two sides, firstly the existing equipment mandates components to be used and secondly the same existing components create requirements for additional peripheral devices which are designed from discrete electronic components and devices.

Component simulations and test cases are implemented in paragraph 2.4.1 and MVP is tested in 2.4.3

As shown on Figure 6 development process is an iterative process usually demanding returning to the modelling stage of system many times until satisfactory results are obtained.

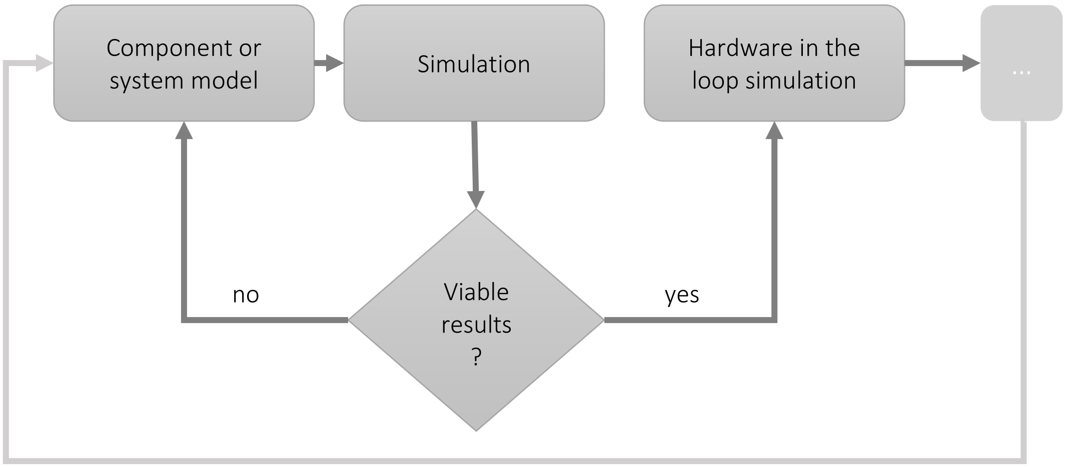


Figure 6. First phases of development process

Most of the simulations conducted in this thesis are result of many iterations in depicted loop of system model not producing sensible simulation results. It is highly likely that the hereby demonstrated models will be subject to change or discard by hardware in the loop and later stages of simulation and testing.

1.4.1 High level system requirements

High level system requirements are demands that the system must fulfil, but have no specifications of by which means these demands must be met. [21, p. 21]

For the thesis at hand author proposes to use high level system requirements for teaching purposes, so that the developed solution would help the process of teaching automotive electronic systems design to students. Such requirements are imposed in Table 2.

Table 2. List of high level requirements.

Requirement index	**Description**
HL_1	Students get to know automotive electronic system design
HL_2	Students can build various system configurations
HL_3	Students can practice digital control through writing their own code
HL_4	System cost is low
HL_5	System is easily maintained and repaired
HL_6	System is highly interactive to induce exhilarating learning experience

1.4.2 Development tools SW

There are countless options for developing automotive electronic systems. Mostly the set of tools used are chosen application specifically and in different stages of development can require different set of software and hardware for developing purposes.

For demonstrating mathematical models and modelling existing equipment throughout this thesis GNU Octave 4.2.0 is extensively used. Most of the code for simulations execute functions which require Signal package to be installed. Functions of Octave and its packages are documented online and available with relative search terms. [22]

Arduino IDE is used for programming the Atmega controller. All of the code snippets are given a descriptive name in the form of *xyz.q* for better referencing. LTSpice is used for conducting circuit analyses. Vehicle Spy 3 is used to receive and plot CAN messages.

1.4.3 Development tools HW

Verification tests on the equipment are conducted using not calibrated handheld digital multimeters Peak Tech 3345. Current consumption measurements for lighting equipment are conducted with Kyoritsu Kew Mate Model 2000 pocket multimeter using current measuring clamp since some of the lighting equipment current exceeds 10A. To measure and record results for signals Picoscope 2206B PC oscilloscope is used. General purpose development board Arduino Uno rev 3 is used to carry out feasibility tests with stacked Elecfreaks CAN shield. ValueCAN3 CAN interface by Intrepid Control Systems, is used as a network node for enabling and verifying bus communication.

1.5 Overview of mathematical models

Electronic system design is subject to system theory where the basis of novel systems or benchmarking existing ones is conducted according to mathematical models. [11, p. 4]

Described mathematical models are used for modelling of the system and its components.

1.5.1 Non-linear curve fitting with polyfit() function

Matlab and GNU Octave have common function `polyfit()` to fit nonlinear data points with a polynomial p(x) which returns coefficients for n-th degree polynomial to match data in least squares sense. [23]

$$p(x) = p_1x^n + p_2x^{n-1} + \ldots + p_nx + p_{n+1} \tag{1}$$

Where the accuracy of the model compared to data can be estimated using unscaled estimator of residual's sum of squares SS[1]. [24, p. 3]

$$SS = \sum_{i=1}^{n} \left(Y_i - \widehat{Y_i}\right)^2 \tag{2}$$

Where:

- Y_i is i-nth observation (data)
- $\hat{Y}_i$ is i-nth prediction using polynomial estimation

1.5.2 Rotating object's angular velocity from zero crossings of sinusoidal signal

In case sinusoidal signal is produced relative to the position of rotating body, the angular velocity of origin can be determined by analyzing time elapsed between signal's zero values.

Instantaneous angular frequency can be determined by one full rotation in radians of the rotating body divided with the number of zero crossings[2] per one rotation multiplied by current sample time subtracted with last sample time of zero crossing. [11, p. 257]

$$\omega = \frac{2 \cdot \pi}{\mathrm{zc}(t_{n+1} - t_n)} \; [rad/s] \tag{3}$$

Equation (3) is empirically proved by using GNU Octave code *'intstant_angular_velocity_w_chirp.m'* for finding calculated instantaneous angular velocity *'mavsignal'* from evaluation elapsed time between consecutive zc. Similarly, functionality of the code is tested over a chirp like signal with starting frequency value equal to *n*av* to have a relative higher number compared to the ideal signal frequency. End frequency of the chirp is close to zero.

[1] n stands for all elements indices in the samples array

[2] Number of lobes is used originally denoted with M. It was found to be unsuitable for continuous sinusoidal signal, as the formula should evaluate number of zero crossings per one revolution to be accurate.

```
% instant_angular_velocity_w_chirp.m

rpm = round(rand()*10000);       % input by rpm
av = rpm*(pi/30);                % calculate angular velocity from input rpm
t = (1/(av/(2*pi)));             % time for 1 revolution
ts = t/800;                      % sampling time

n = 5;                    % number of generated signal periods per 1 revolution
x = 0:ts:t;               % time axis
fi = rand()*6.14;         % phase shift
y = sin(n*av*x+fi);       % signal
y2 = chirp(x,n*av,t,0,'linear',fi);    % chirp, start with higher frequency
xzc = zerocrossing(x,y);               % find zerocrossings
xcc = zerocrossing(x,y2);              % find zc for chirp

for i=2:(length(xzc))     % loop over count of zc starting from second
      mavsignal(i,1)=(2*pi)/(2*n*(xzc(i)-xzc(i-1))); %instant angular velocity
end

for i=2:(length(xcc))
      mavchirp(i,1)=(2*pi)/(2*n*(xcc(i)-xcc(i-1)));
end

dots = interp1(x,y,xzc);  %map zc on x axis
dotc = interp1(x,y2,xcc);

subplot(4,1,1)
plot(x,y,xzc,dots,'rd')
title(sprintf('input rpm = %0.f, angular velocity = %0.f rad/s', rpm, av))
xlim([0 t]) xlabel('time') ylabel('signal')

subplot(4,1,2)
bar(xzc,mavsignal,0.2)
xlim([0 t])
text(xzc,mavsignal/2,num2str(mavsignal,'%0.f'),'color','yellow','rotation',
double(90),  'horizontalalignment','center');
ylabel('rad/s') title('calculated angular velocity at every zc of ideal signal')

subplot(4,1,3)
plot(x,y2,xcc,dotc,'rd')
xlim([0 t]) ylabel('signal') title('chirp signal')

subplot(4,1,4)
bar(xcc,mavchirp)
xlim([0 t])
text(xcc,mavchirp/2,num2str(mavchirp,'%0.f'),
      'rotation',double(90),'horizontalalignment','center','color','yellow')
ylabel('rad/s') title('calculated angular velocity at every zc of chirp signal')
```

Figure 7. Model script for numerical analysis of equation (3)

Above model uses random input between 0 and 10k rpm. Rpm is used to make better sense of the angular velocity in question, as the reader if verifying the code can more relate to rpm from everyday experience. Output of above is depicted on Figure 8. Variable *'n'* denotes number of sinusoidal signal period which are generated per full revolution of rotating body, which is usually linked to the physical constraints of any given system where rotational speeds are measured in similar manner.

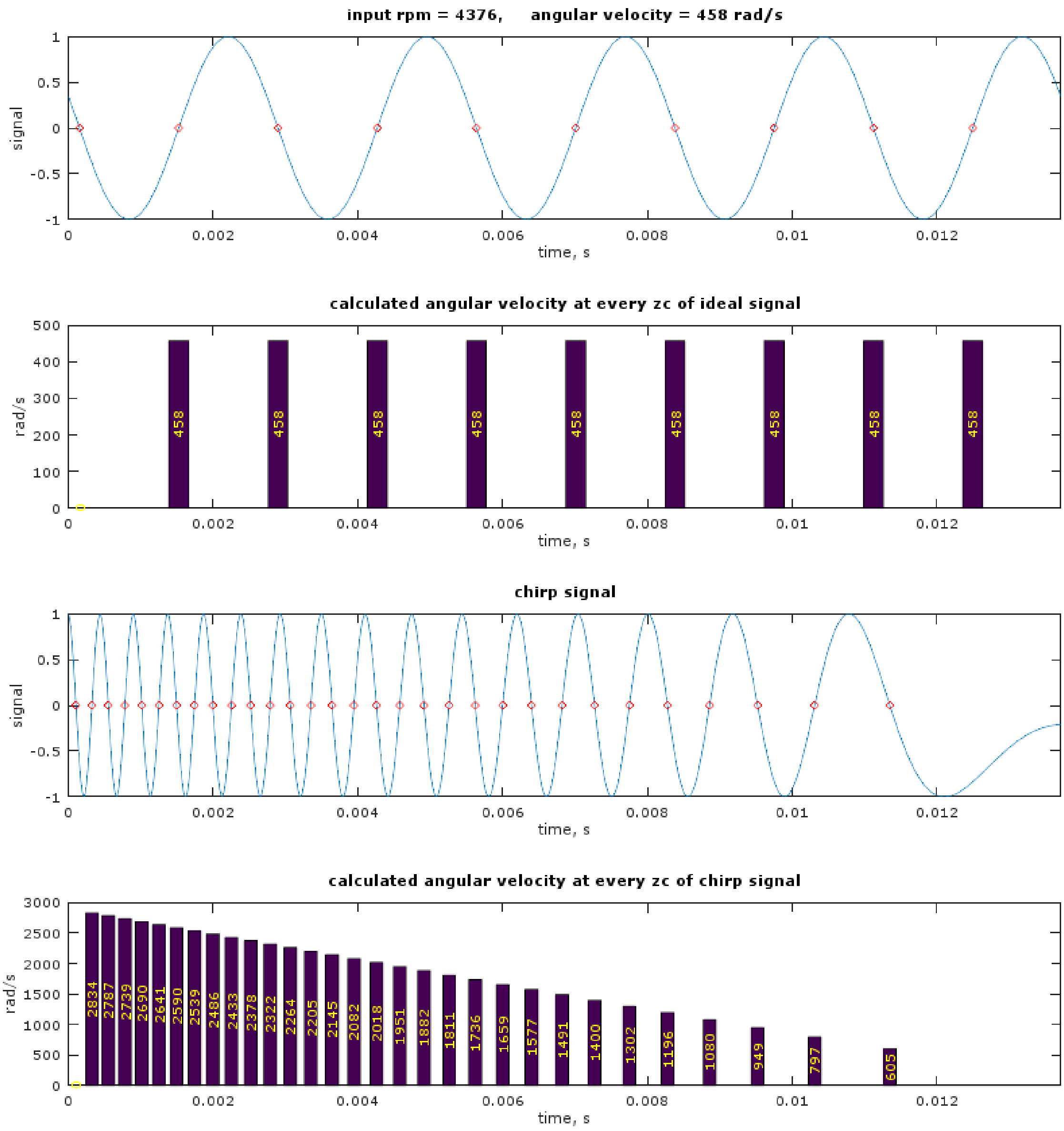


Figure 8. Numerical analysis of equation (3)

Graphs providing signal values include instances of zero crossings with red diamond shaped markers. On above figure it is evident, that the model calculates instant angular velocity on every consecutive ZC of signal, measuring time interval between current and last ZC as formulated in equation (3). It must be noted that first calculated angular velocity in this model produces value of zero since it has no prior ZC instance to evaluate from. These zero valued angular velocities are also visible on both bar graphs where zero is printed across x axes near beginning of the bar graphs.

2 Developing MVP

Existing equipment is mapped to define corresponding domain architecture and requirements which will be basis of defining necessary ECU properties and peripheral devices for I/O processing. The aim of this development process is to come up with justified plan for investment for purchasing necessary ECU's to be used with existing equipment.

2.1 Verification and modelling of existing equipment

Existing LN equipment is mounted on experiment stands which are modular and easily reassembled. Boards are designated with marking SO32YY-XX which is LN catalogue product number. In the following chapters board number markings are used for distinguishing.

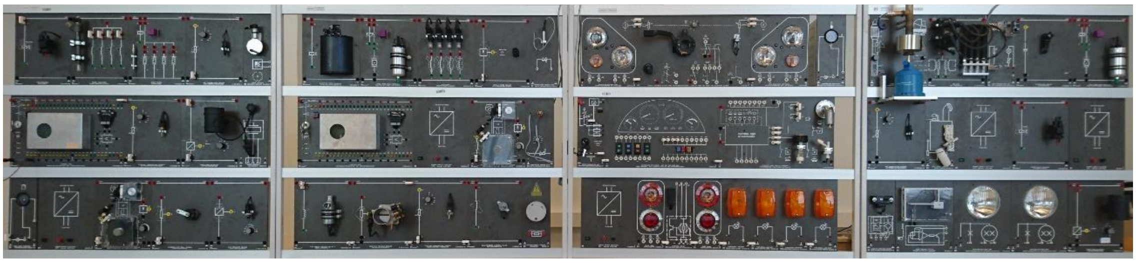


Figure 9. Overview of the equipment on experiment stands

Verification measurements are important for:

- assessing the health of the equipment
- checking if the components work as expected
- independent documentation of the operating parameters

Different types of signals from the sensors are modelled based on the verification measurement to enable future simulations.

2.1.1 AC/DC power supplies

AC to DC power supplies are used to supply automotive operating voltage of approximately 13.5 DC

Table 3. AC to DC power supply technical description.

Attribute	**Description**
Board	SO3216-1C
Pieces	5
Input	110-240 VAC
Output	13.5 VDC
Measured output	13.47-13.54 VDC

2.1.2 Accelerator-pedal sensor

Accelerator pedal sensor is used by the driver to indicate driving torque demand. Linearity of pedal sensor against angle is not mapped.

Table 4. Accelerator-pedal sensor technical description

Attribute	**Description**
Board	SO3220-2K
Pieces	1
Input	13.5 VDC
Measured output	Min: 1.1 VDC
	Max: 8.87 VDC

2.1.3 Air temperature sensor

Air temperature is measured to determine vehicle operating conditions. Existing boards use potentiometer with turn-knob where the temperature “t" is marked on the board’s surface. It must be noted that t values are labels not actual temperatures.

Table 5. Air temperature sensor technical description.

<table>
<tr><th>Attribute</th><th colspan="9">Description</th></tr>
<tr><td>Board</td><td colspan="9">SO3220-1J</td></tr>
<tr><td>Pieces</td><td colspan="9">2 (b1, b2)</td></tr>
<tr><td>Input</td><td colspan="9">None</td></tr>
<tr><td rowspan="3">Measured output</td><td>t</td><td>-20º</td><td>-10º</td><td>0º</td><td>10º</td><td>20º</td><td>40º</td><td>70º</td><td>100º</td></tr>
<tr><td>b1, kΩ</td><td>10.53</td><td>10.41</td><td>7.73</td><td>5.08</td><td>3.67</td><td>2.33</td><td>0.425</td><td>0.001</td></tr>
<tr><td>b2, kΩ</td><td>10.50</td><td>8.14</td><td>5.46</td><td>4.20</td><td>3.00</td><td>1.781</td><td>1.20</td><td>0.50</td></tr>
</table>

By the measurement results it is likely that the board mimics NTC resistor type temperature sensor as seen on Figure 10. Also difference between two boards becomes greater as t increases.

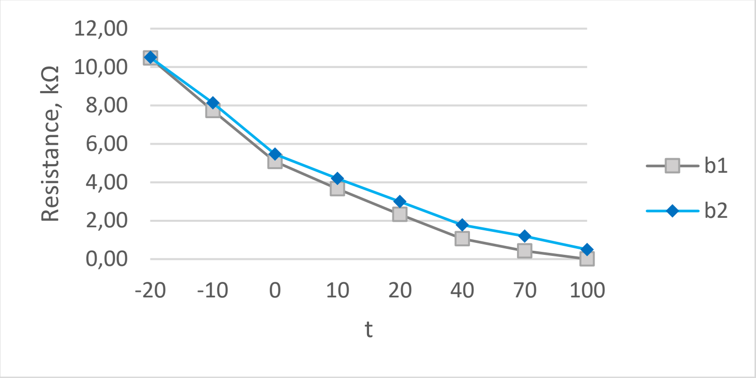


Figure 10. Temperature sensor measurement results graph.

2.1.4 Mass airflow meter

Mass airflow meter is used to determine the amount of air consumed by engine to calculate amount of fuel needed for current operating conditions. Existing boards use potentiometer with turn-knob where the air flow “g/s” is marked on the board’s surface. It must be noted that g/s values are labels not actual measured flow of air.

Table 6. Mass airflow meter technical description.

Attribute	**Description**									
Board	SO3220-2C									
Pieces	2 (b1, b2)									
Input	13,5 VDC									
Measured output	g/s	5	10	20	40	60	80	100	120	150
	b1, V	0.97	1.51	2.04	2.41	2.96	3.39	3.82	4.17	4.79
Notes:	Second board out of order. Potentiometer not working.									

As seen on Figure 11 the measurement results of mass airflow meter output show linear nature compared against linear trend line from 1-5 volts.

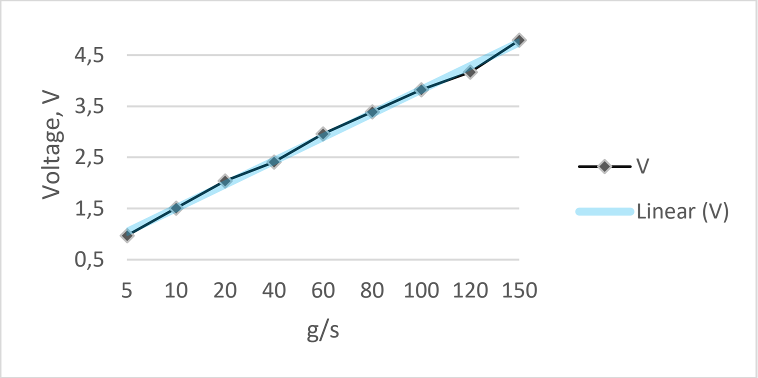


Figure 11. Mass airflow meter measurement results graph.

2.1.5 Throttle position sensor

Throttle position sensor measures the angle of throttle valve inside the engine intake channel. The maximum output voltage value is determined by the design of the study equipment board, where the throttle body leaver is hitting the surface of the board near maximum opening. It can be assumed that unrestricted movement would yield maximum output of the same level as input.

Table 7. Throttle position sensor technical description.

Attribute	**Description**
Board	SO3220-1R
Pieces	2
Input	13.5 VDC
Measured output	Min: 1.1 VDC
	Max: 8.87 VDC

2.1.6 Coolant temperature sensor

Coolant temperature sensor is used to determine coolant liquid temperature in the engine block. Existing boards use potentiometer with turn-knob where the temperature “t" is marked on the board’s surface. It must be noted that t values are labels not actual temperatures.

Table 8. Coolant temperature sensor technical description.

Attribute	**Description**
Board	SO3220-1J
Pieces	2 (b1, b2)
Input	None

Measured output	t	-20º	-10º	0º	10º	20º	40º	70º	100º
	b1, kΩ	6.84	5.15	3.31	2.13	0.90	<0.01	<0.01	<0.01
	b2, kΩ	9.87	7.17	4.44	3.13	1.81	0.624	<0.01	<0.01
Notes	Board b1 potentiometer seems to be out of order.								

By the measurement results it is likely that the board mimics NTC resistor type temperature sensor as seen on Figure 12.

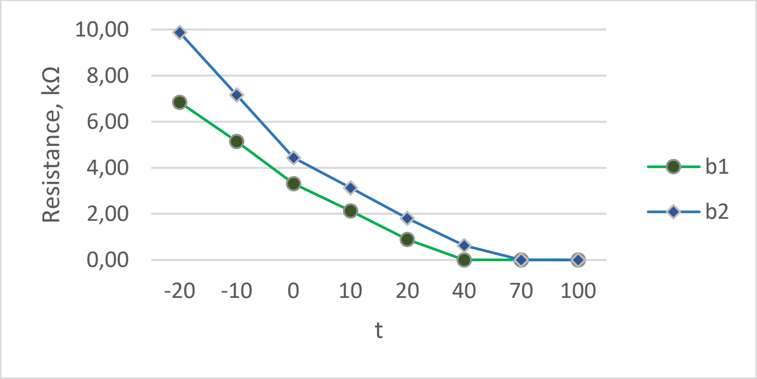


Figure 12. Coolant temperature sensor measurement results graph.

Resistance of position t = 100 on b2 is not distinguishable from t = 70. B1 is not functioning correctly. If subject to future use in lab experiments, the root cause of the malfunction must be investigated and the faulty component replaced. On the other hand, it may be useful to have multiple boards, out of which one is faulty to introduce unexpected results into the learning process with the use of the equipment.

2.1.7 Crankshaft speed sensor

Crankshaft speed sensor is used to acquire engine angular velocity. As seen on Figure 15, crankshaft speed is determined by induction sensor which is triggered by square toothed trigger wheel. Every pass of one tooth from the sensor induces a full period of sinusoidal signal. 1st cylinder TDC can be also determined by the disruption in signal by asymmetry in trigger mechanism but the stroke about to begin cannot be distinguished.

Table 9. Crankshaft speed sensor technical description

Attribute	**Description**
Board	SO3220-2D
Pieces	2

Sensor	BOSCH 0281210
Input	13.5 VDC
Output	AC voltage

Two square teeth are missing, which induces a higher peak to peak voltage. By evaluating signal correlation to hardware it was discovered that the number of periods of the sinusoidal signal does not match the number of teeth on the trigger wheel.

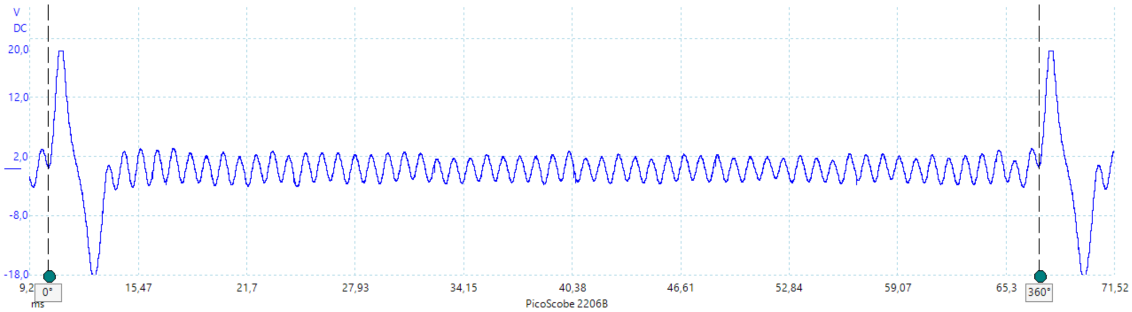


Figure 13. Crankshaft speed sensors output.

It is evident on Figure 13 that the amplitude of the signal is varying in time although it is known that the trigger wheel is driven at constant speed.

One of the hypothesis for the reason of such deviation from uniformity was considered to be coaxial misalignment of the trigger wheel while rotating generating a wobble in the distance between sensor and trigger wheel outer edge.

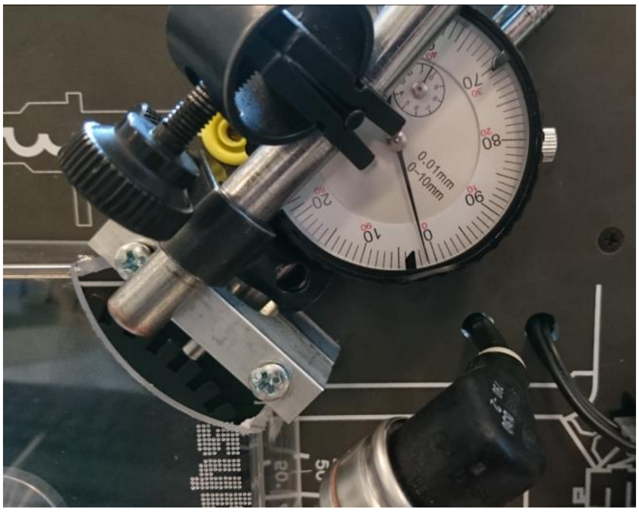


Figure 14. Measuring trigger wheel coaxial alignment

The hypothesis was tested using a distance measuring gauge with one hundredth of millimeter precision. Crank speed sensor is removed from its fixing and gauge is fitted using a side of the metal back panel of the board as anchoring plane. Coaxial misalignment over multiple points of one revolution measured had maximum deviation between min-max of 0.11 mm.

Author assumed that this cannot be the reason for amplitude deviation in the output signal. Actually, the small deviation in the air gap, has exponential impact on the magnetic flux propagation, which in turn has high impact on the output signal. [12, p. 262]

Amplitude deviation is taken account in later simulations starting from *'crank_speed_signal_2pi_radians_ea.m'*.

Trigger wheel geometry was CAD modelled to evaluate geometrical constraints. It was determined that the trigger wheel was divided into 120 equal parts where three degrees length of outer perimeter was either a tooth or a gap.

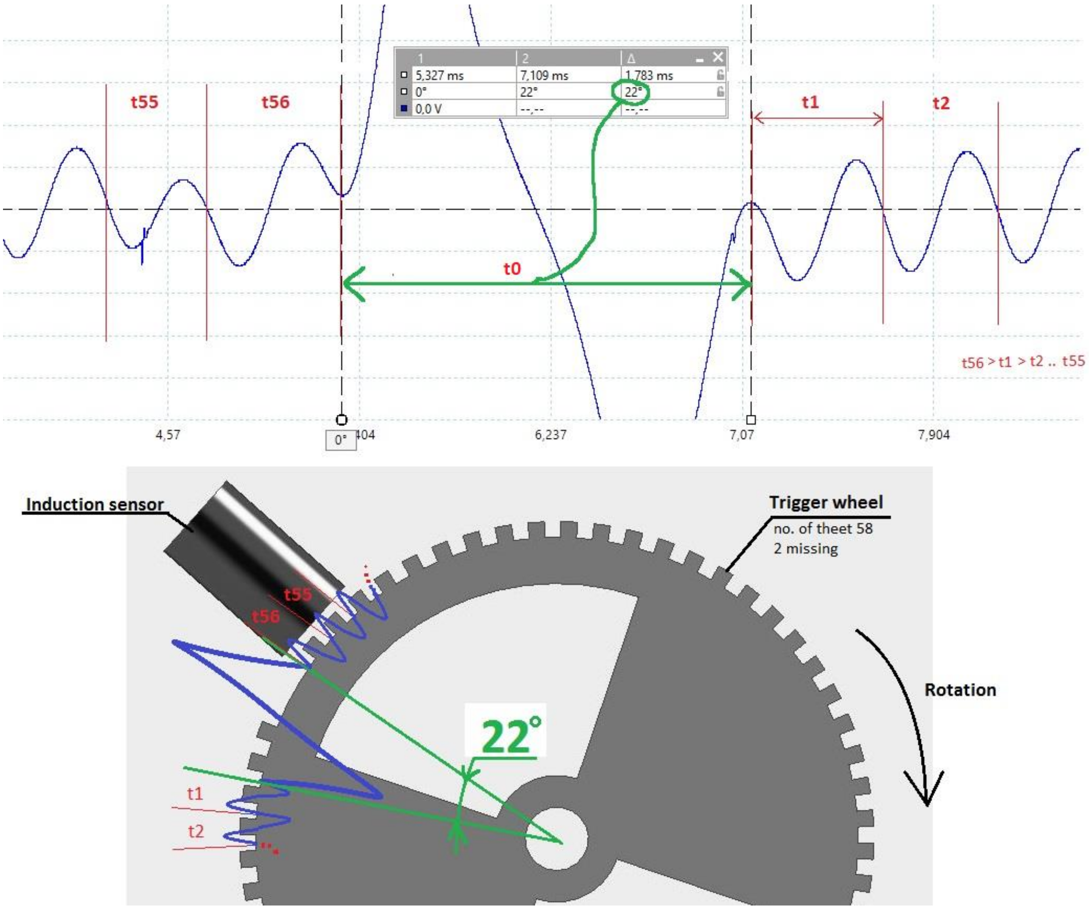


Figure 15. The output signal (top) of the crank speed sensor compared to the crank trigger wheel geometry (bottom).

By comparing the signal to geometry of the trigger wheel, specially the t_0 proportion, it was determined, that the angular duration between t56 - t1 correspond to angular duration of two missing and two present teeth. If the signal is actually inverted and/or shifted to either way in time is not determined.

Low peak to peak voltage part signal periods are denoted with t_n, t56 duration is 7.5 ^{o} being longer than t1 with duration 6.5^o or any t_n in between. In case of ideal signal output from crank sensor, duration of t_n should be exactly 6^o. Which is also the reason why t_0 is 22 degrees not 24 as four teeth and four gaps on the trigger wheel geometry should add up to. The beginning of t_0 of the signal is chosen to be the reference point of 0 ^{o} or 0 rad.

Crankshaft speed sensor output signal is modelled using *'crank_speed_signal_2pi_radians_ea.m'* found in Appendix 1 to be used in further development and testing procedures. Given model is result of 3rd iteration of first phases of development process as described on Figure 6. Different approaches to model architecture were considered, by trying time based and sample based implementations, yielding iteratively more insight towards understanding of how the signal is constructed. Surfaced approach uses angular divisions to estimate signal over trigger wheel rotation measured in radians. Loops in the model are avoided. Measurement units do not have impact on the model, but the division of separate parts and relationships of the divisions set up a path towards viable results as demonstrated below.

The data on which the model is based on originates from the crank speed signal measured in similar way as seen on Figure 13. The oscilloscope outputs roughly about 1M samples where the one rotation of trigger wheel is roughly about 507 000 samples, where as the model with ea = 0.005 rad is using 1258 samples to calculate the model signal sample points. Thus for comparing the data to model, data is down-sampled by every 404 samples and the values are imported as comma separated file which is also available via Appendix 6.

Model of the cranks speed signal takes account that the missing teeth part of 22 degrees has amplitude 8 times higher than rest of the signal. Signal amplitude measured from sensor is relative to angular velocity of trigger wheel, which can be later scaled relative to input rpm of simulations. Phase shift of signals is used to follow measured results where the distinguished parts of the output signal meet.

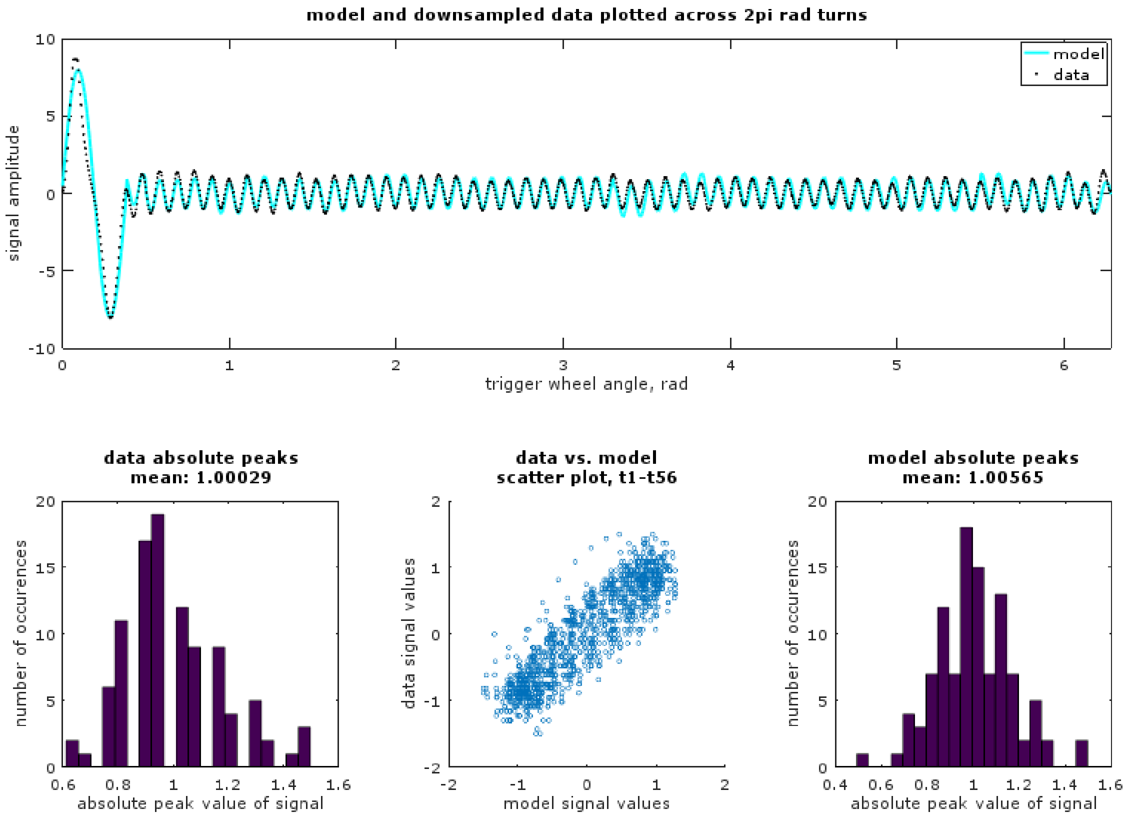


Figure 16. Crankshaft speed sensor signal model compared to data

To follow measured results, but to avoid randomization every signal sample, a random-valued matrix is used to alter signal amplitude during calculation. Interval of randomization is tied to the angle, and independent variable *rm* is used to randomize approx. after every 4-5 signal periods.

A baseline amplitude for modelled signal t1-t56 is chosen to be 1, thus data is scaled so that the mean value of the absolute amplitude value of named signal parts would be close to 1. Using independent variable sf = 0.433, mean of 1.00029 is achieved. The mean of modelled signal absolute amplitude peaks between t1-t56 vary with each run of the script but stay close to one. Estimating histograms on Figure 16 shows that the spread of the signal peak values between data and model is similar with 0.5 – 1.5 range and showing similarities with normal distribution, although modelled signal amplitude peak spread can be more equally distributed as well, depending on the randomization of any execution of the model script.

Scatter plot on Figure 16 shows visual confirmation of similarity of modelled signal compared to data. Ideal model would follow the data on 45 degrees angle across the plot but the small deviations of amplitude and phase create an ellipsoidal with focus points roughly about intersecting the diagonal mentioned before. For improving the model further statistical analyses could be conducted in the next iterations of model development.

Hereby the model is evaluated to be sufficiently accurate for crankshaft speed estimation modelling. Randomization of the modelled signal helps to verify future efforts in determining instantaneous angular velocity estimation algorithms as it cannot be expected that the signal level deviation in demonstrated manner would remain constant with other similar equipment. Every execution of above code results in unique set of zero crossing instances which have to be evaluated, meaning that, benchmarking of IAV evaluation algorithms can be conducted more rapidly.

2.1.8 Camshaft position sensor

Camshaft position sensor is Hall type sensor picking up a section cut out of the trigger wheel. It has 13.5 VDC input which is passed through to the output as the cut out section passes the sensor head. It can be used to control ignition timing as the 1st cylinder TDC can be determined from the output signal.

Table 10. Camshaft position sensor technical description

Attribute	**Description**
Board	SO3220-2D
Pieces	2
Sensor	MB 0031539728
Input	13.5 VDC
Output	DC square impulse

Measured signal waveform can be observed from Figure 18. It must be noted that the board uses an electric motor to drive both crank and cam trigger wheels which are linked together using toothed belt drives.

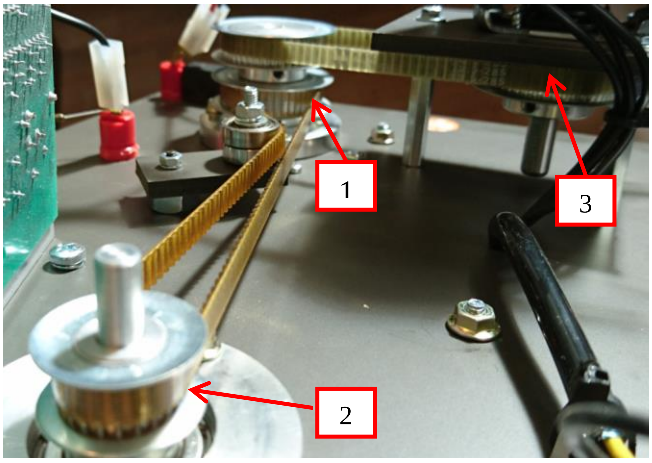
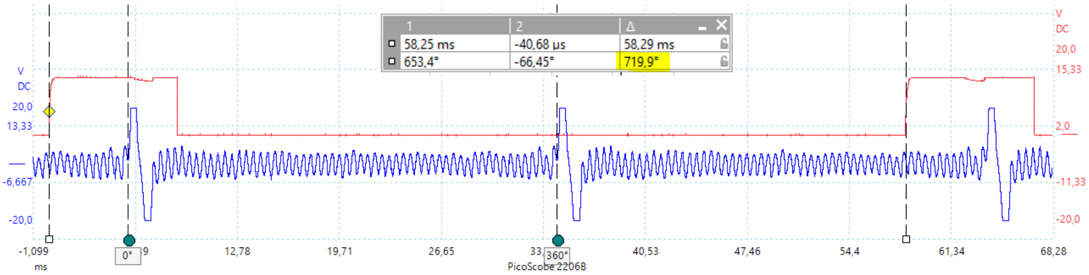


Figure 17. Cam (1) and crank (2) shaft drive gears with electric motor drive (3)

Physical constraints of the drive gear ratios maintain the cam trigger wheel speed of two times lower relative to crank trigger wheel. Thus cyclic signals observed have also 2:1 occurrence ratio as evident below. Offset of the wheels is maintained by the setup of the toothed gears via drive belts. On real ICE’s, a considerable offset is used before TDC to ensure real time computing capability for ECU specially at high rpm.



	1	2	Δ
□	58,25 ms	-40,68 µs	58,29 ms
□	653,4°	-66,45°	719,9°

Figure 18. Camshaft position sensor (red), triggered once every 720 degrees of crankshaft (blue) rotation.

Camshaft position sensor output signal is modelled using GNU Octave code *‘camshaft_position_signal_4pi_radians.m’* to be used in further development and testing procedures. Although y axis signal array could be populated using array indices representing degrees which can be executed with two lines of code:

```
y(1:109) = 13.5;
y(110:720) = 0;
```

Figure 19. Camshaft position sensor output modelled with two lines of code

But to consolidate with crankshaft speed signal modelled previously, again divisions of angular durations are defined which allows unification of x-axes without conversion for crank and cam signals.

```
% camshaft_position_signal_4pi_radians.m

hv = 13.5;                % signal high value
pi4 = pi*4;               % constant
ap5 = (109*pi)/180;       % angular proportion of high value
ea = 0.005;               % evaluation interval in radians

%find angular divisions of x axis and join to one array:
%   |start          |smpl duration      |end
x6 = 0              :ea                 :ap5;
x7 = ap5            :ea                 :4*pi;
x8 = [x6 x7];

% optain array size for populating y
x6len = length(x6);
x7len = length(x7);

y6(1, 1:x6len) = 13.5;    % signal
y7(1, x7len) = 0;         % duration is 2 revolutions = 4pi
y8 = [y6 y7];

plot(x8, y8, 'r'); xlabel('crank shaft rotation, rad');
ylabel('signal apmlitude'); ylim([-2 14]); xlim([0 pi4]);
```

Figure 20. Camshaft position sensor output modelled with angular constraints

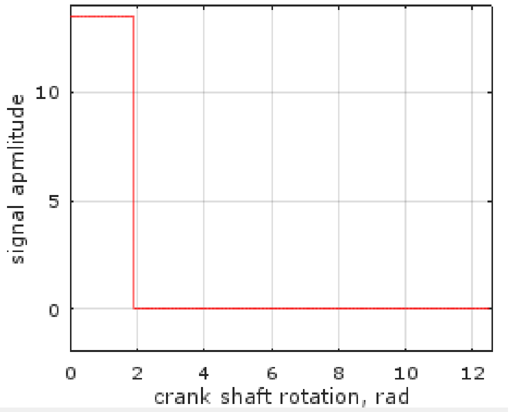


Figure 21. Output of ‘*camshaft_position_signal_4pi_radians.m*’

Given model is relatively simplistic, where the evaluation accuracy does not impact simulated signal given that the ‘*ea < 0.1 rad*’ to match the sloping angles of the measured signal.

2.1.9 Modelling and estimating crankshaft speed and camshaft position

Modelled signals can be used for testing software algorithms to estimate crankshaft speed and angle relative to 1st cylinder TDC. Without this knowledge, the ICE will not be operated due to lack of correct fuel quantization and injection and/or ignition timing. Also other functions of the system such as idle speed control need accurate momentary operating parameters to provide intended services.

Inconsistencies in uniformity of level and amplitude in crankshaft speed sensor output must be taken account. Cam and Crank signals are modeled starting from cam signal rising edge. Matrix operations are used for multiple purposes: tying the signals from end to end, calculating consecutive instances of ZC angles and shifting signals. Full code ‘crank_cam_zc_av.m’ is found in Appendix 2.

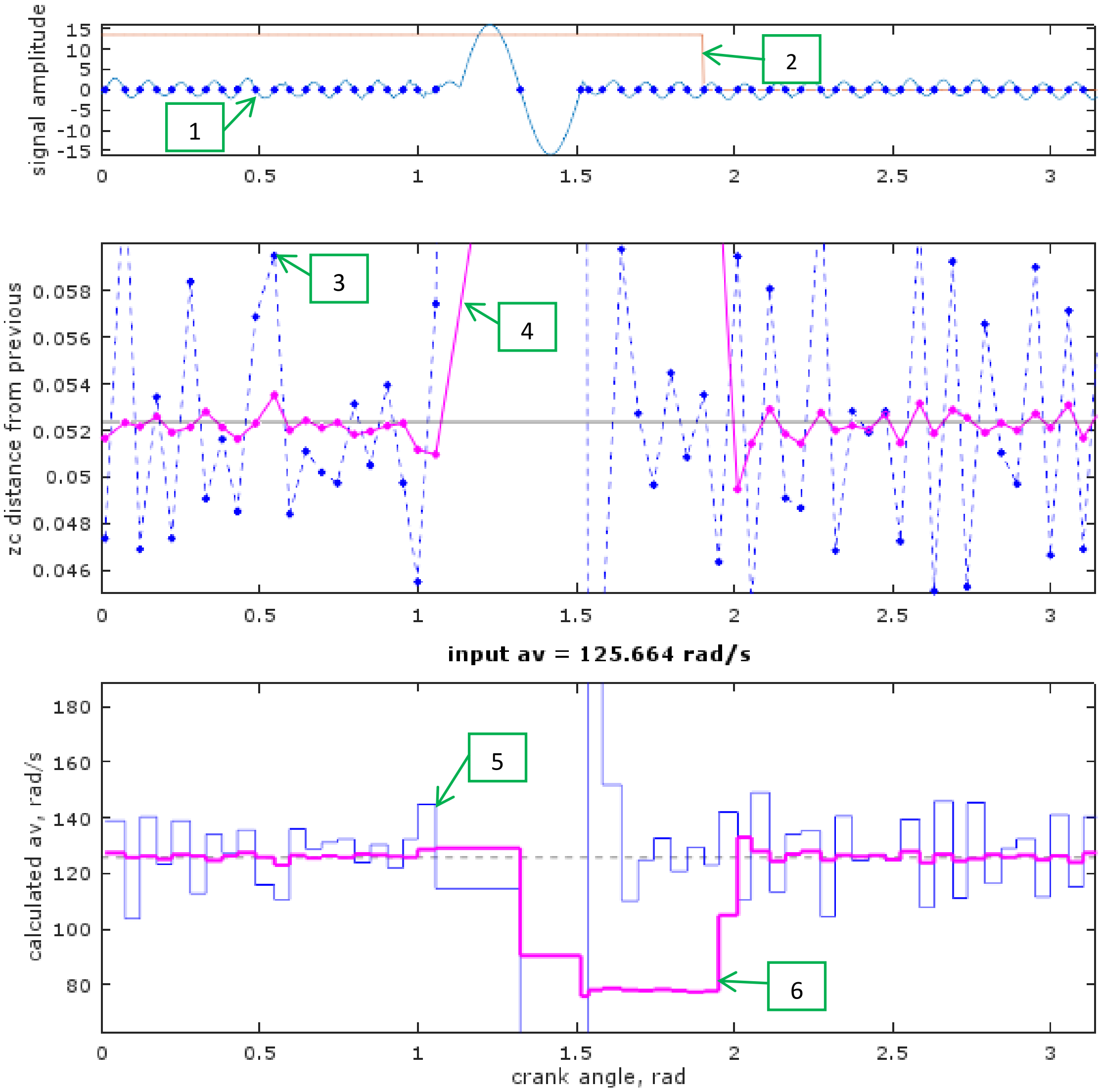


Figure 22. Output of ‘*crank_cam_zc_av.m’ pl = pi; w = 10*

Figure 22 numbered markers explained:

1. Crank speed signal (blue)
2. Cam position signal (red)

3. ZC distance from previous, rad (blue)
4. ZC distance averaged over w, rad (magenta)
5. Angular velocity calculated from ZC distance (blue)
6. Angular velocity calculated from averaged ZC distance (magenta)

The output of the model is shown on Figure 22 with the ‘pl’ variable set to pi. Model also has optional inputs for the plot length up to two revolutions of crank trigger wheel and toggle text to show calculated instantaneous angular velocity values. Axes are limited to better illustrate the relationship between ZC distance and calculated IAV. It must be noted that ZC distance is measured here in radians.

Varying signal amplitude and frequency poses challenges in application of equation (3), as the zero crossing events are not equally distributed given that the rotation of the trigger wheel is at constant speed.

Figure 22 positions 3 and 5 illustrate that instantaneously calculated angular velocity at every ZC differs from the neighboring due to the signal properties inflicting ZC deviations in spreading over time.

Even more so for t_0, where two consecutive ZC-s are further apart yielding approx. six times lower calculated rpm value compared to input. At the end of t_0 lye two consecutive ZC without greater spread meaning that calculated rpm value is many times higher than input rpm which are visible as distinguishable spikes.

To have more accurate calculated results from the randomly distributed ZC averaging over multiple sample’s distance can be considered. In above figure averaging over 10 samples is shown in magenta in middle and bottom plots. With averaging window size w at 10, a tenfold increase in calculation accuracy can be demonstrated as seen on Figure 23.

Further w values were tested between 2 - 100 and the model produces useful results up to w = 49. IAV calculation accuracy above 10 does not give significant absolute median accuracy increase (going closer to zero), where the difference of w = 10 and w = 49 is roughly about 0.5 units of difference in median accuracy compared to input IAV, to compare with no w vs. w = 10 as seen on Figure 23 the median accuracy compared to input IAV is increased roughly about 7 units of rad/s. Median is used to outcast the t_0 induced spikes as mean function would account more for the spikes as well.

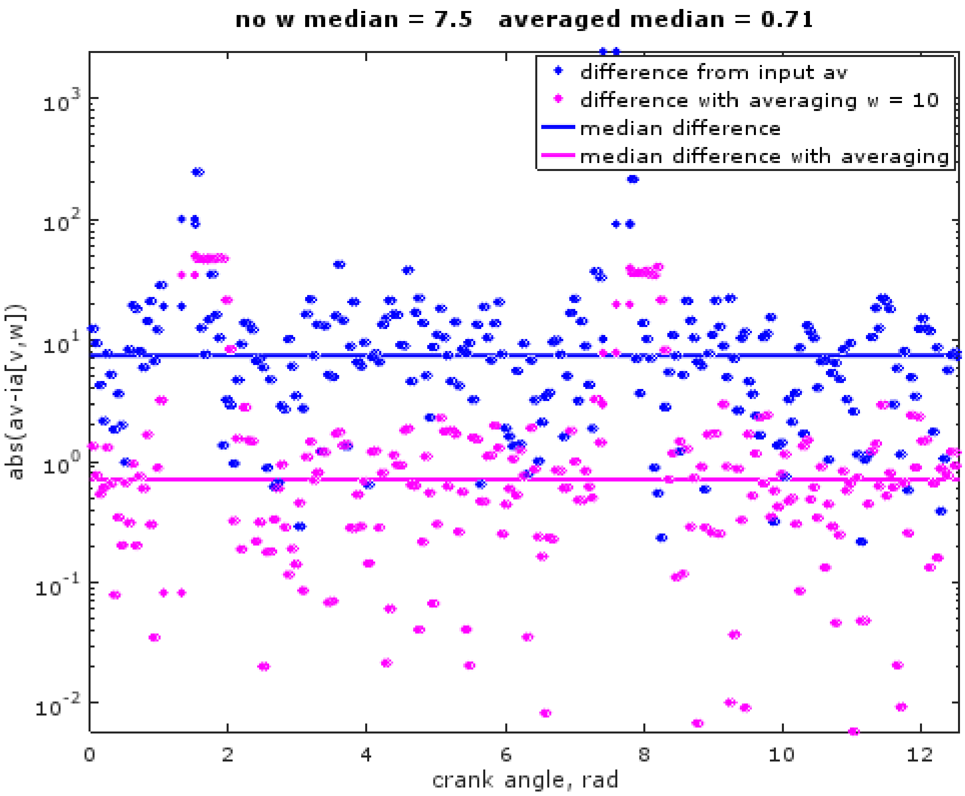


Figure 23. Averaing effect shown on absolute difference from input angular velocity

Looking at the square impulse type slope at angle ~1.3 rad for the averaged ZC calculated IAV, gives reason to consider investigation of slope detection algorithms in future developments to detect TDC.

If engine management system is supposed to control different functions based on any calculated rpm value in previously shown manner, then the obtained values differ from the constant engine operating speed. And as shown also on may differ substantially. Demonstrated model does not account variable instantaneous angular velocity during one crank revolution due to piston engine properties. Acceleration and deceleration of the angular velocity or change of rpm in time is not regarded as well, but must be taken account in the next steps of obtaining information about engine speed from cranks shaft speed sensor.

2.1.10 Temperature sensors NTC and PTC

Radiator temperature sensor is used to determine coolant liquid temperature in the cooling radiator. These are actual temperature sensors which can be heated up using a heating element on the SO3216-7D board. It must be noted that t values are actual measured temperatures using multimeter with temperature measuring probe.

Table 11. Temperature sensor NTC technical description

Attribute	**Description**		
Board		SO3216-7J	SO3216-7J
Pieces	1		
Input	None		
Measured output			
t, °C	NTC, Ω	PTC, Ω	
20	602	108.6	
30	425	110.0	
40	286.6	112.1	
50	199.5	114.5	
60	140.7	117.3	
70	103.7	120.0	
80	77.7	123.0	
90	59.2	126.1	
100	46.6	129.2	
110	37.3	132.4	
120	30.5	135.5	
130	25.5	139.0	
140	21.7	142.2	
150	18.8	145.6	
160	16.5	149.0	

Temperature sensors on automotive equipment usually have a range of -40 to 160 °C, but classroom environment would need special cooling equipment to go below ambient room temperature.

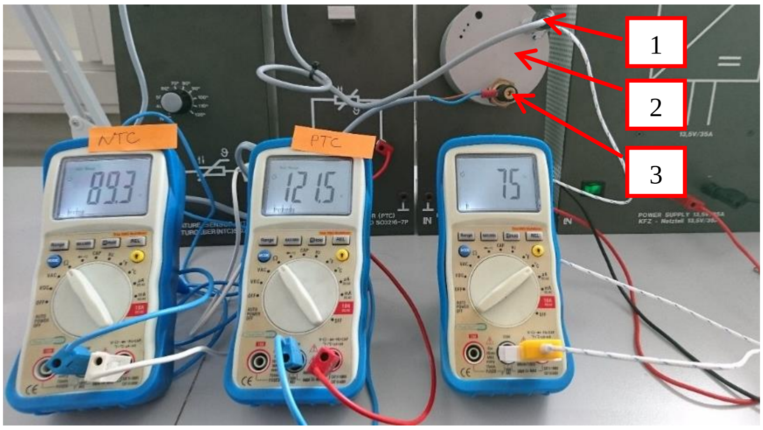


Figure 24. Mapping of temperature sensors. 1- PTC sensor. 2 – Heating element. 3 – NTC sensor.

As seen on Figure 25 PTC type thermistor has linear output characteristic while the NTC type temperature sensor show a non-linear trend. For modelling behavior of the NTC type thermistor's output resistance against temperature data can be modelled with a n-th order polynomial.

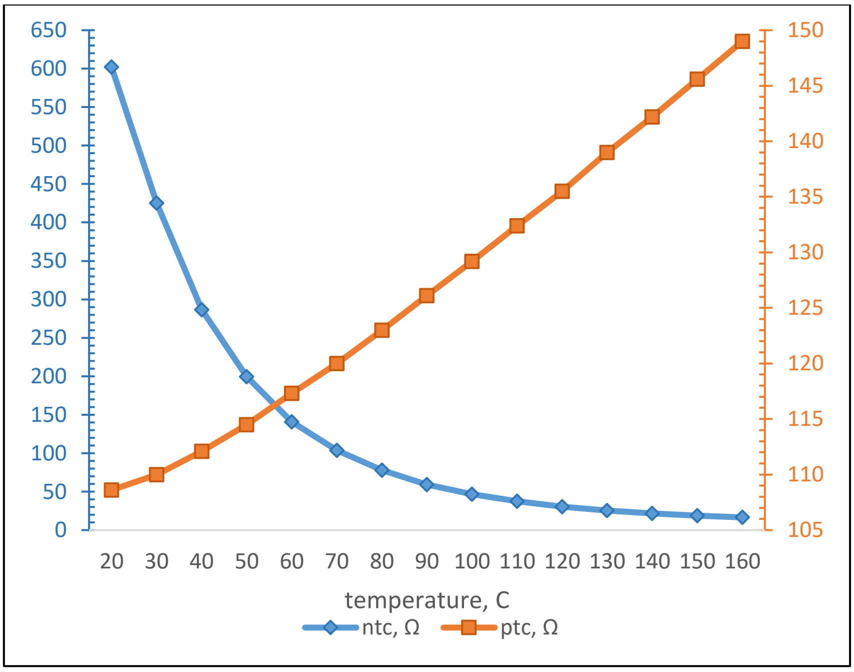


Figure 25. NTC and PTC temperature sensors showing different characteristics.

To assess which order polynomial should be used to model given signal different evaluations of n-th order polynomial are compared to measured data by applying equation (2). NTC thermistor's signal is modelled using *'temp_NTC_plynomial_diff_score.m'* found in Appendix 3.

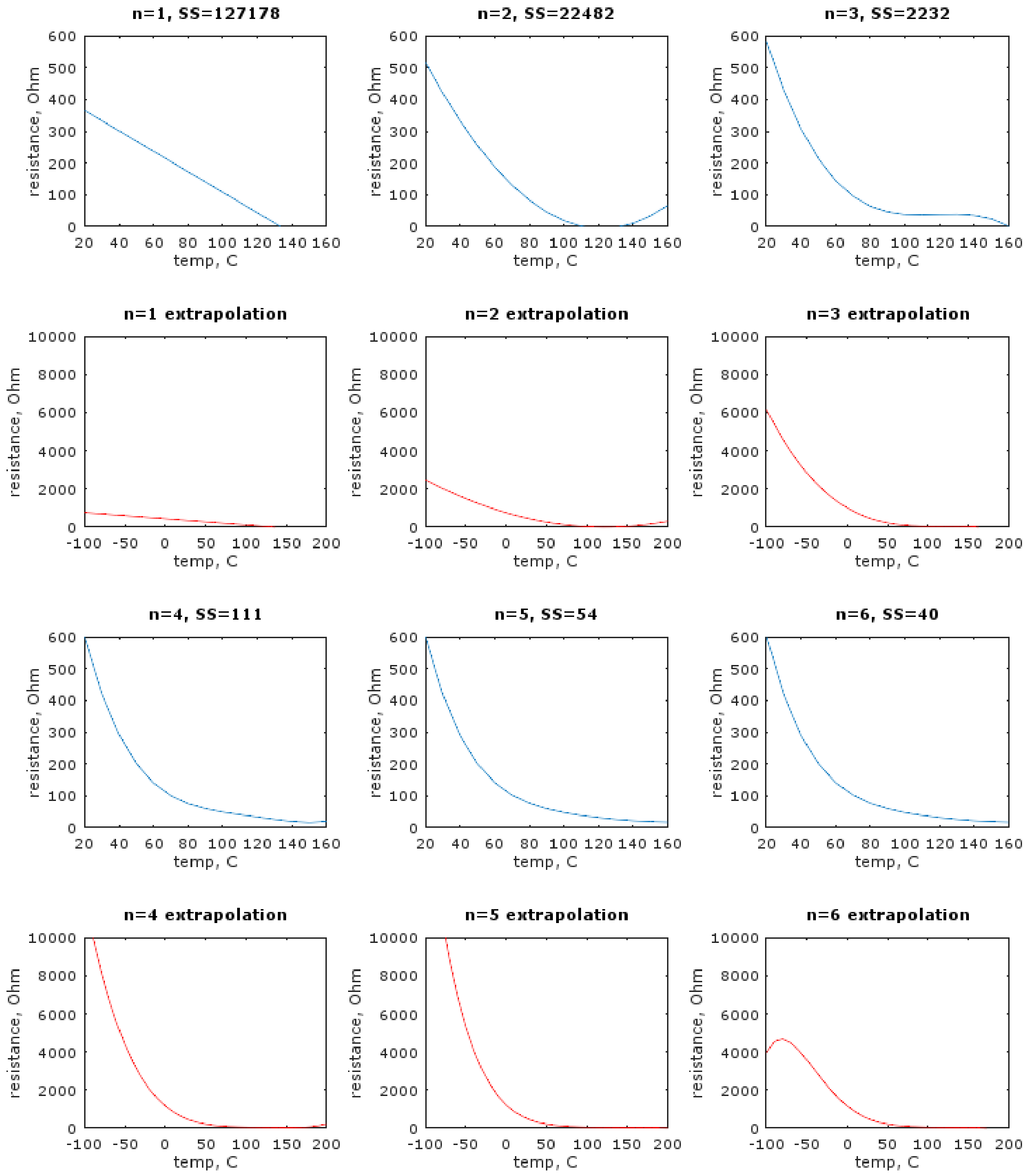


Figure 26. Non-linear curve fitting with n-th order polynomial

From the model output it is evident that 5th order polynomial provides lowest SS with useful extrapolation results.

Between room ambient temperature and 150 degrees Celsius it is possible also to do hardware in the loop testing with the heating element and real temperature sensors as seen on Figure 24. Modelling of the nonlinear curve in this case provides further convenience in future development and testing purposes as the lower temperatures down to -30 and beyond are not achieved easily in classroom environment. But using formulated model of resistance against temperature for the NTC type sensor it is possible to emulate different circumstances.

To increase the accuracy of the model for below ambient room temperatures, further testing should be conducted using freezing equipment. The model can be updated with relatively low burden by updating measurement results array in the code and rerunning GNU Octave script again.

If equation (1) is fitted with 5th order polynomial coefficients based on the actual measurement results then the y axis values representing resistance would be calculated according to equation (4).

$$\begin{aligned} y = \; & -0.000000024363x^5 + 0.000016246x^4 - 0.0043223x^3 + 0.58045x^2 - \\ & - 40.249x + 1208.7 \end{aligned} \tag{4}$$

2.1.11 Lighting equipment

Lighting equipment is used in automotive applications mainly for road illumination and indicating turning or braking action. Other functions such as hazard lights make use of the indicator lights.

Table 12. Lighting equipment technical description

Name	**Board**	**Pieces**	**Current / board, A**
Headlamp with side lamp	SO3216-2A	2	10.6
Indicator flasher	SO3216-2G	2	1.8
	SO3216-2H	2	1.8
Main headlight unit with side lamp, left	SO3216-2M	1	10.8
Main headlight unit with side lamp, right	SO3216-2N	1	10.8
Tail light unit with LED	SO3216-2P	2	3.8
Switch unit for headlights and indicators	SO3216-2T	1	-
Note: Currents measured at input voltage of 13.5 V Total current consumption 57.6 A			

Most of the study boards do not make constantly use of currents as shown in Table 12, but the current is measured as all the elements on the board are turned on simultaneously to take account maximum current consumption to estimate the needs for lighting equipment fuse and relay array dimensioning. It must be taken account that automotive equipment fusible links are usually dimensioned 3 times above constant current consumption for tolerating current peaks while turning the equipment on or off. But also as seen above, summing up total current consumption shows the lack of power from one power supply board which has rating of 35 A. Incase such experiments are conducted where currents exceed power supply rating the addition of second power supply with separate high-side circuit should be considered.

Although using mainly incandescent light bulbs, existing equipment is useful for teaching purposes. Present trend in the automotive sector is that all lights on the vehicle use LED technology. One way to implement LED technology on existing equipment would be to use replacement LED bulbs which fit to current study board’s lightbulbs sockets.

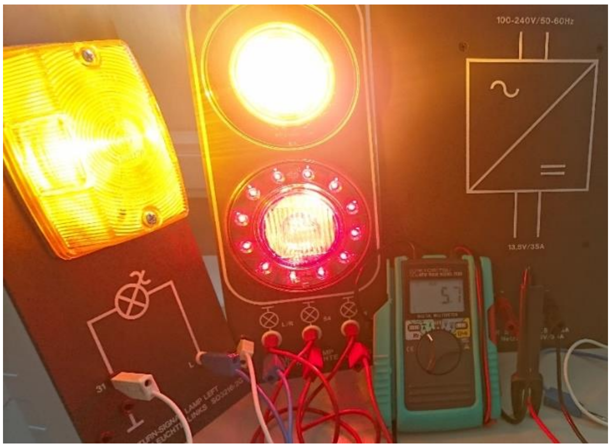


Figure 27. Verifications of lighting equipment operation and current consumption.

The switch unit for headlights and indicators make use of the switches used on actual vehicle dashboard and steering column. These are analogue switches presenting different states vehicle operation and can be used in future development of lighting equipment control as on/off analog signal. But with the future use of electronic lighting control further functions can be developed and implemented such as *‘follow me home’* system with programmable intervals.

2.1.12 ICE actuators

Different actuator devices are used to control functions of ICE such as idle speed, ignition, fuel injection, fuel supply, intake air and fuel pressure regulation etc.

Name	**Board**	**Pieces**	**Current / board, A**
Fuel injection unit with 4 injectors	SO3220-1U	1	3.4
Canister purge solenoid, fuel tank vent	SO3220-1V	1	0.4
High pressure fuel pump	SO3220-1X	2	0.2
Rail pressure regulator	SO3220-2F	1	1.6
Glow-plug simulation	SO3220-2H	1	0.3
Diesel injector simulation	SO3220-2J	1	0.01[1]
Wastegate solenoid	SO3220-2L	1	0.9
EGR solenoid	SO3220-2M	1	0.9
Note: Currents measured at input voltage of 13.5 V Total current consumption is 8 A			

Actuators are driven by ECU through driver circuits depending on the character of the actuator. Most of high current circuits on the study equipment are simulated with visible indicators. Such example is the glow plug circuit which in real application can consume more than 10 amperes per cylinder, but on the study board just a LED is illuminated to indicate activation of glow plugs. Same technique is also used for diesel injectors where each injector is simulated with LED but actual common rail systems use specialized driver circuits which amplify voltage up to couple of hundred volts for rapid actuation of diesel injectors to achieve multiple injection capability per one power cycle.

For upgrading engine control study equipment, power consumption must be considered for the actuators in such way that ECU chip outputs are able to drive actuators in desired manner. ECU chips usually do not interface directly with actuators but have decoupling and conditioning and driver circuits since ECU chip's outputs themselves are low voltage and low current and in need of protection from possible over voltage or over current. Also

[1] Milli-ampere measurement range was used, opposed to all ohter measurements where 10 A measurement range was applied.

different actuators require different matching of the output signal, some require digital to analog conversion, some may require pulse width modulation.

For the above actuators, mostly on/off switching can be implemented with sufficient current driver circuits with either electromechanical or semiconductor relays driven by the ECU’s low voltage low current outputs.

2.2 ECU architecture specifications

ECU architecture will be considered as baseline for acquiring modern development platforms to be used with existing equipment. There are many widespread development boards also available like Arduino and Raspberry PI, but they do not contain automotive application specific features which are readily available from automotive ECU manufacturers. Such features most importantly in this case are the embedded network interfaces for CAN, LIN, Flexray etc. and automotive ECU architecture with various integrated features for input processing, control logic and output processing. Manufacturers also make readily available programming Integrated Development Environments (IDEs), simulation tools with models and software environments for jumpstarting any kind of design (study) processes.

Knowing the possible inputs and outputs gives overview of the minimum requirements I/O wise to which the domain control unit must match and of possible ad hoc circuitry aka peripheral circuitry which must be added.

Schematics provided in this chapter provide structured view of possible inputs and outputs required, though a lot of signals on a modern day vehicle are not presented due to the partial scope of these educational simulation devices.

Without described knowledge it is complicated to find best matching product from available hundreds of possibilities available from different electronic controller and processor chip manufacturers. Included tables in each sub paragraph are demonstration of possible available equipment from known suppliers. Additionally, budgetary price is shown for comparison giving an overview of MVP investment range.

Due to differences in EFI and CRD systems they are separated in the following chapter as opposed to the project scope shown on Figure 2.

2.2.1 EFI ICE ECU architecture based on existing equipment

EFI ICE ECU should consider possible inputs (sensors) and outputs (actuators) from chapter 2.1.

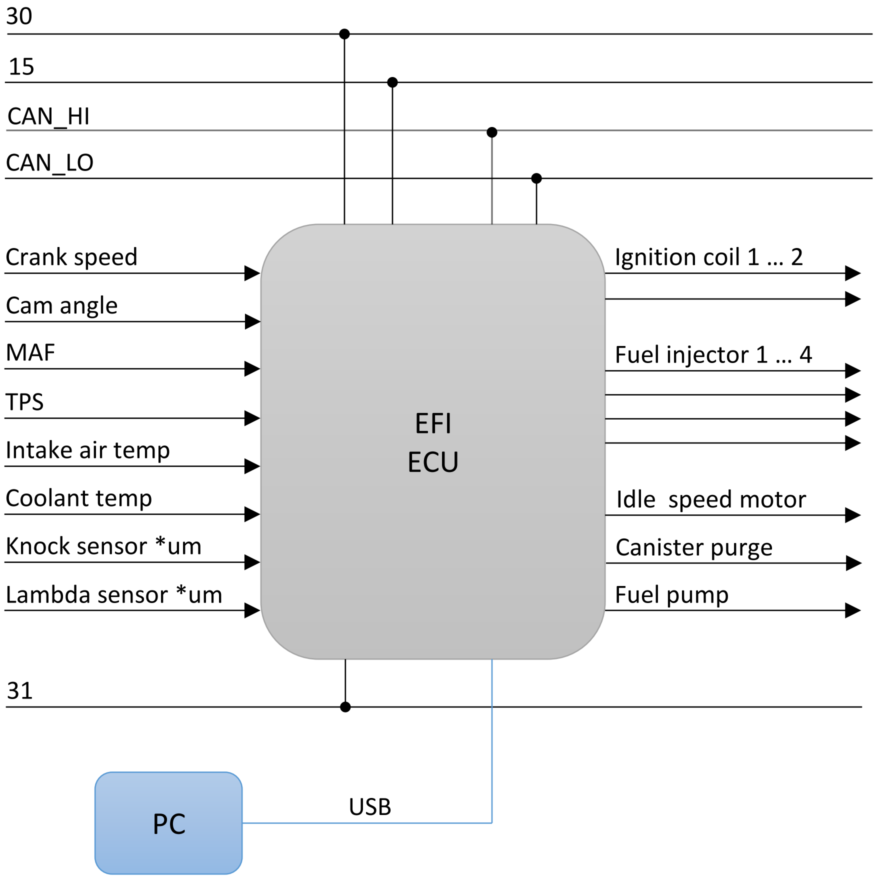


Figure 28. Gasoline Internal Combustion Engine ECU. *um - unmapped

As seen from Figure 28, existing EFI components are originally intended to be used as wasted spark system, where ignition is triggered every TDC in all cylinders where as only one of the cylinders is about to utilize generated spark. Modern EFI engines mostly get ignition energy for the spark plug from separate ignition coil for each cylinder which can all be driven also separately. PC connection is used for interfacing ECU and its peripherals, programming, debugging and executing different tasks.

Table 13. Possible ECU development kit examples

Option	Supplier	Product	Budgetary Cost, €	Comments
1.	Infinieon	Kit_ak_small_engine	120.00	2 cylinder application out of the box [25]

2.	NXP	S32K144EVB	45.00	General purpose development kit [26]
3.	STMicroelectronics	SPC563M64L5	8.00	Processor chip [27]
		EVAL-L9779WD-SPI	130.00	Evaluation board with diver circuit [28]

Depending on the ECU built in capabilities additional peripheral input processing ad hoc circuitry is needed to condition i/o signals to match ECU i/o parameters. The second option is general purpose automotive 32 bit architecture development board.

2.2.2 CRD ICE ECU architecture based on existing equipment

CRD ICE ECU should consider relevant selection of inputs (sensors) and outputs (actuators) from chapter 2.1.

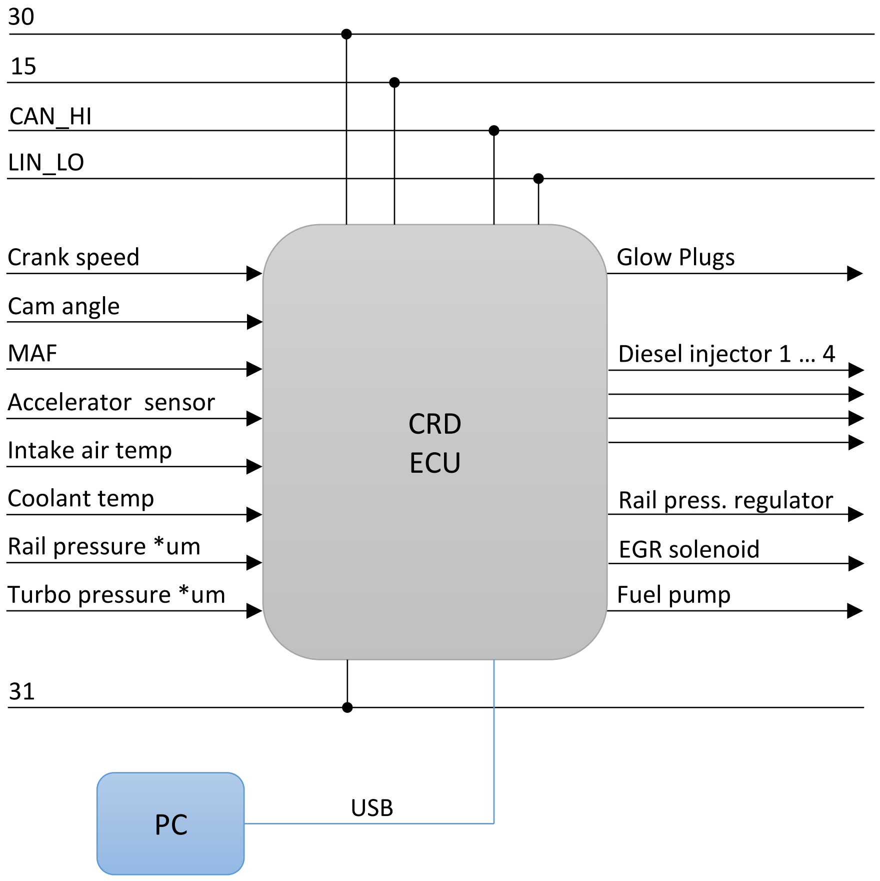


Figure 29. CRD ICE ECU architecture. *um - unmapped

Compared to EFI fuel injector circuit, CRD will not make use of driver circuit for given simulation board since actuation of injectors is simulated with LEDs which can be driven also by ECU. If not directly then by decoupling circuit.

Overview of possible ECU platforms applies with the same remarks as in Table 13 for EFI ECU.

2.2.3 Human Machine Interface ECU + display

Since human machine interface is communicating with the user, then also a modern method of interaction must be chosen. Current trends show fully digital instrumentation cluster displays as standard equipment in modern vehicles.

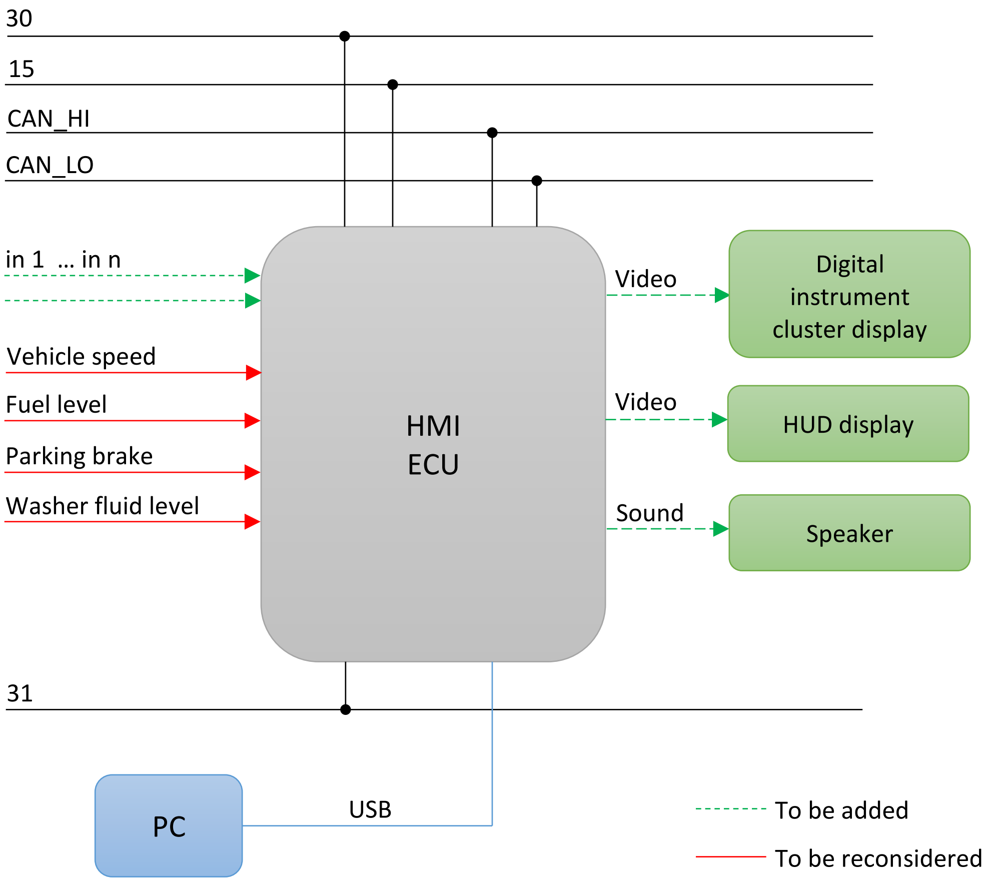


Figure 30. HMI ECU architecture

Inputs ’in’ denote user interaction button signals necessary to interact with the user functions of the provided platform. Such buttons are usually found on the steering wheel or center console of the vehicle. In this case user input buttons can be integrated to ECU development kit platform, which is often already considered by suppliers. Furthermore, motion capture and other novel interaction methods can be considered.

The input signals shown above in red are simulated with devices attached to the existing equipment's HMI control board (which is not updateable), thus they are to be reconsidered, which eventually would lead to new product in the HMI domain, going out of scope of the MVP idea of using existing equipment.

Table 14. Possible ECU platforms for HMI

Option	Manufacturer	Product	Budgetary Cost, €	Comments
1.	Renesas	R-CAR M3SK	300.00	Includes audio-video I/O [29]
2.	NXP	MAC57D5xxEVB	800.00	Needs daughter card [30]
3.	STMicroelectronics	STM32469I-EVAL	375.00	Embedded touchscreen [31]

2.2.4 Lighting ECU

Lighting control in modern vehicle has many computer enabled functions such as corner following lights, follow me home system, programmable interior lights and etc.

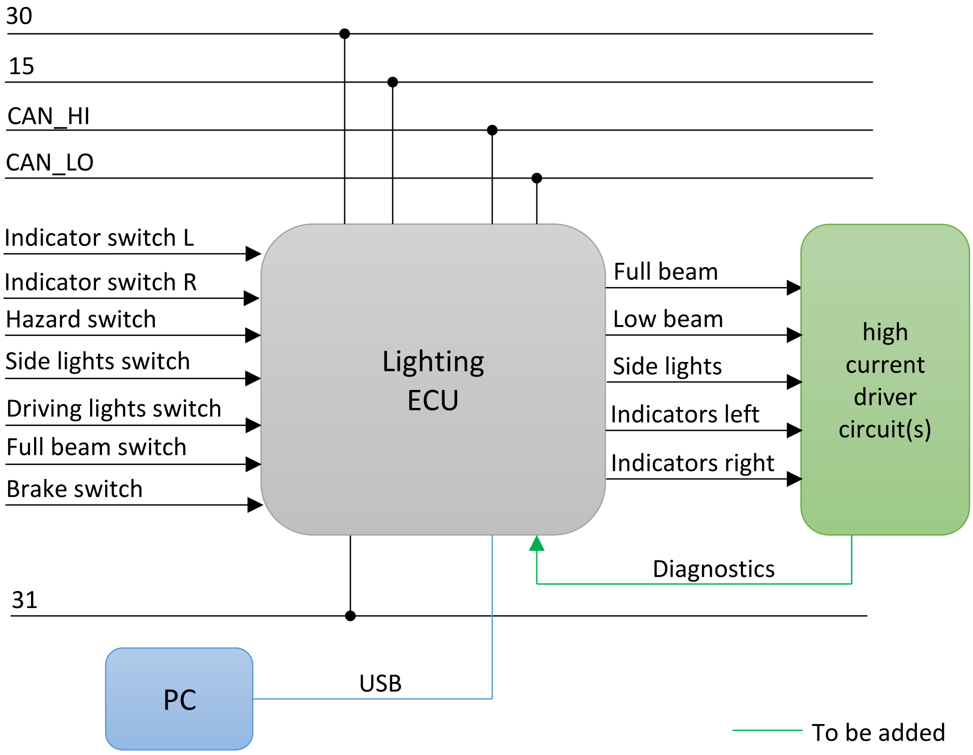


Figure 31. Lighting equipment electronic control unit arhitecture

Lighting control unit should provide feedback if the used lights are operated correctly or a malfunction is occurring. Since lights and their illuminative and indicative properties are safety critical for vehicle operation and traffic

Table 15. Possible lighting equipment ECU platforms

Option	**Manu-facturer**	**Product**	**Budgetary Cost, €**	**Comments**
1.	Renesas	QB-R5F10PPJ-TB	35.00	Suitable for lighting control [32]
2.	Infineon	SPOC+ EVAL KIT1	na	Driver board [33]
		KIT_XC2288H_SK	150.00	PC interface with CAN [34]
3.	NXP	KIT12XS6EVM	130.00	Driver board for lighting [35]

2.3 Networking of automotive ECUs practical example

Based on the ECU architecture definition it can be also specified which signals could be shared over the network between the nodes. In this chapter example specifications are contrived for later concept feasibility testing purposes. Specified signals are mostly to provide HMI visual indication values from ICE and lighting ECUs, but these signal use different multiplexing concepts with different data types to further convey usefulness of digital data bus in modern vehicle.

High-Speed CAN Network is setup with 500 kbps data rate (ISO11898-2). 11-bit identifiers are used with the message length 0—8 bytes.

Ad hoc specification is created here for testing purposes. Message IDs chosen are arbitrary. Engine speed and coolant temperature can be communicated by engine control unit via digital communication channel to the HMI ECU which outputs the results to digital instruments cluster.

Status of lighting system must be communicated to the HMI unit by lighting ECU since it is required to display lights in use to the driver via instruments cluster.

2.3.1 Engine speed and throttle position signals over bus

Engine speed is usually broadcasted to vehicle backbone network with around 10ms interval within the same message frame as well as throttle position. Numerous systems use this information, one example in this thesis context would be HMI which is displaying vehicle operating conditions to the user.

Table 16. CAN message 0x1A2 specification

ID	**DLC**	**D0**	**D1**	**D2**	**D3**
0x1A2	4	Engine rpm MSB	Engine rpm LSB	Throttle MSB	Throttle LSB
Min		0x0000		0x5432	
Max		0xFFFF		0x6432	
Rate	10ms				

Engine speed is measured by ICE ECU and converted to integer values between 0 and 65536 by multiplying measured rpm with 4. Maximum value takes exactly 2 bytes transmit in binary form, thus 2 bytes in the messages are assigned respectively. Rpm readout follows OBD-II convention with min. 0 and max. rpm value at 16383.75 as seen in (5) [36]

$$RPM = \frac{256 * D0 + D1}{4}\ [r/min] \tag{5}$$

Throttle position values are also bound to the needed accuracy, but furthermore, to the sensor accuracy and ECU ADC resolution. Incase ECU has 12bit DAC to convert sensor output voltage to binary value, then possible values are between 0 and 4095 which is more than ten times less than fits into the 2 bytes allocated for transmitting the value over CAN Network. Thus the 4096 values are fitted in the middle range of 16 bits of possible values in the allocated message space. This also improves diagnostics features as when either higher or lower value is detected in the network than is specified here.

$$Throttle = \frac{(256 * D0 + D1) - 21554}{4096} * 100\ [\%] \tag{6}$$

2.3.2 Air and coolant temperature signals over bus

Because of the relatively slow nature of temperature change, there is no need to exchange relevant information over bus as for the previous signals thus 1 s interval can be used.

Table 17. CAN message 0x2B1 specification

ID	DLC	D0	D1
0x2B1	2	Coolant temp	Air temp
Min		0x00	0x00
Max		0xFF	0xFF
Rate	1s		

OBD-II convention is followed where temperatures are specified between -40 and 215 ºC are sent over network.

$$Coolant\ Temp = D0 - 40\ [℃] \quad (7)$$

$$Air\ Temp = D1 - 40\ [℃] \quad (8)$$

2.3.3 Lighting equipment status signals

Status of the lighting equipment is displayed to the user as well as diagnostic information. This information can be multiplexed into a can message by allocating n bits for each input status out of a transmission byte. When any of the inputs are set high then corresponding bit goes high. CAN messages data is usually presented in hex format where the binary data is displayed in compact form. Currently 7 analog input switches are present as seen on Figure 31 thus one byte is sufficient to carry lights input statuses across the network.

Table 18. CAN message 0x3C4 specification

ID	DLC	D0	D1
0x3C4	2	Light switch status	Lightbulb MLF status
Min		0x01	0x07
Max		0xBF	0xFF
Rate	1s or on occurrence		

The unused bits in the byte are set to be always high again for better diagnostics purposes. If any of the static bit values differ then there is reason for investigating the reason.

Table 19. Bit allocation table for D0 and D1

bit	Value	D0 high signal	D1 high signal
b7	128	Indicator Switch L	Indicator bulb MLF
b6	64	Indicator Switch R	Side light bulb MLF
b5	32	Hazard Switch	Low Beam Bulb MLF
b4	16	Side Lights Switch	Full Beam Bulb MLF
b3	8	Low Beam Lights Switch	Brake Light Bulb MLF
b2	4	Full Beam Switch	none (always high)
b1	2	Brake Switch	none (always high)
b0	1	none (always high)	none (always high)

If side lights and low beam are switched on, then corresponding CAN message would be ID:0x3C4, DLC:2, D0:0x19, D1: 0x07 where D0 value is calculated by 1+8+16 = 25 and in hex format it is 0x19. D1 remains 0x07 as long as there is no MLF to output since 1+2+4 = 7 and in hex format it is 0x07.

2.4 MVP feasibility tests with HW in the loop

Feasibility tests of previously documented and described architecture concept are conducted using general purpose development board with CAN shield. Tests carried can be used to evaluate accuracy of models developed previously. As shown the different possible implementations of peripheral devices can benefit HL_6.

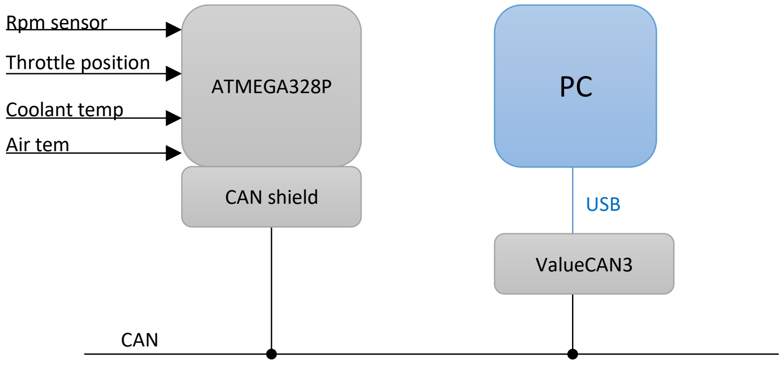


Figure 32. Scope of feasibility testing

Hardware in the loop tests are conducted for measuring IAV from rpm sensor. Voltage dividers were used to feed analogue inputs of ATMEGA328P to provide simulated inputs of:

- Throttle position sensor
- Air temperature
- Coolant temperature

These signal then are multiplexed to CAN messages and transmitted over the network with timing constraints according to specification.

2.4.1 Input conditioning with peripheral devices

As an example of input conditioning necessity, rpm sensor signal is treated with different known methods to provide a way to detect signal frequency which in turn provides rpm count.

One of the possible ways to conduct low frequency signal analysis with controller in use would be to scale the signal between 0-5 volts with 2.5 being the zero level under investigation. The analysis can be conducted programmatically. [37]

Programmatic approach was tested with sinusoidal signal of 2 kHz as is the average range of rpm sensor output signal. With 2 kHz input, the programmatic approach readout yielded results 1923-2024 Hz. As the approach is quite complex going deep into the registries of the controller, it does not serve the purpose which author has in mind for fulfilling HL_6.

Hardware triggering methods were compared as the initial developer of the rpm sensor and engine control unit indicates that Schmitt triggers are used. [12, p. 263]

Different circuit devices can be implemented: [38]

- Optically isolated devices
- Non-inverting op-amp with positive feedback
- Phase-locked loop
- Specially designed circuits (eg. LT1116)

Firstly, optical isolator 4N25 was tested to achieve impulse signal for ECU digital input.

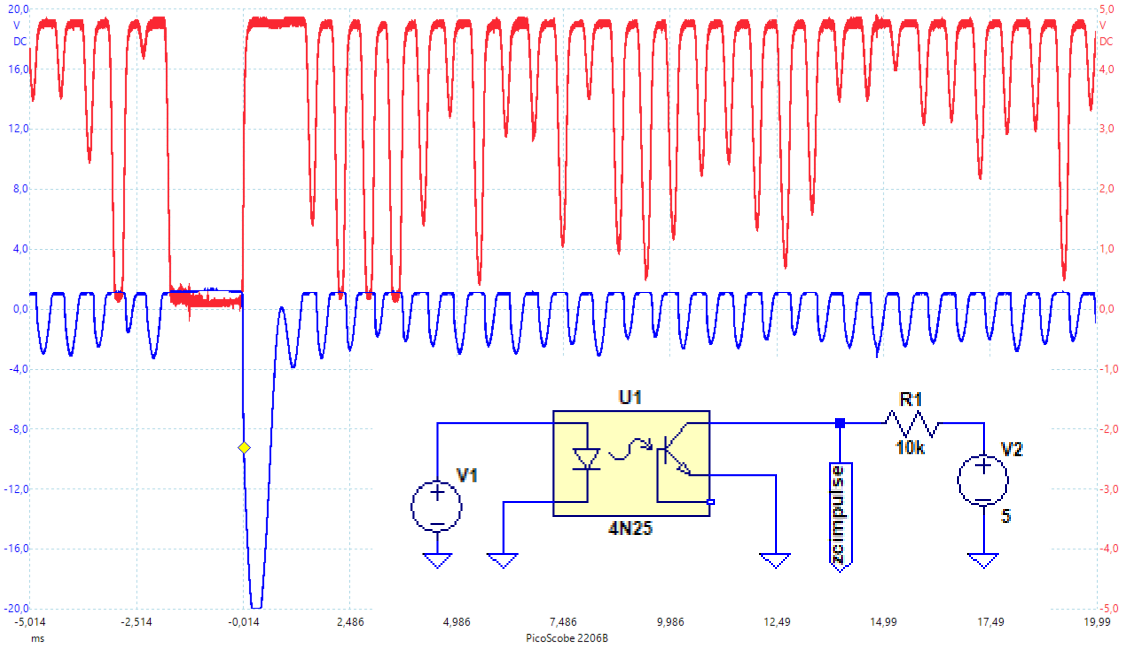


Figure 33. 4N25 as Schmitt trigger hardware testing. Rpm sensor signal (blue) vs. opto-isolator output (red)

With additional circuit design and analysis, it would be possible to use 4N25 type components as triggering devices for ECU input pins to detect ZC events. As seen on Figure 33 output is not pulled down enough and ECU digital input pin would be constantly high.

Further Schmitt trigger implementation simulations were carried out with discrete component design in mind. [39]

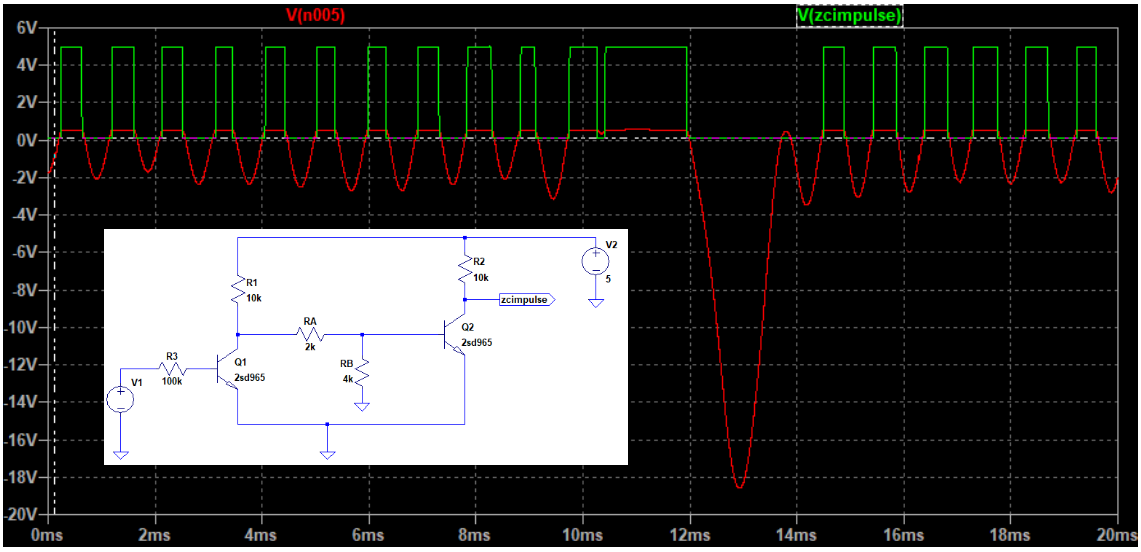
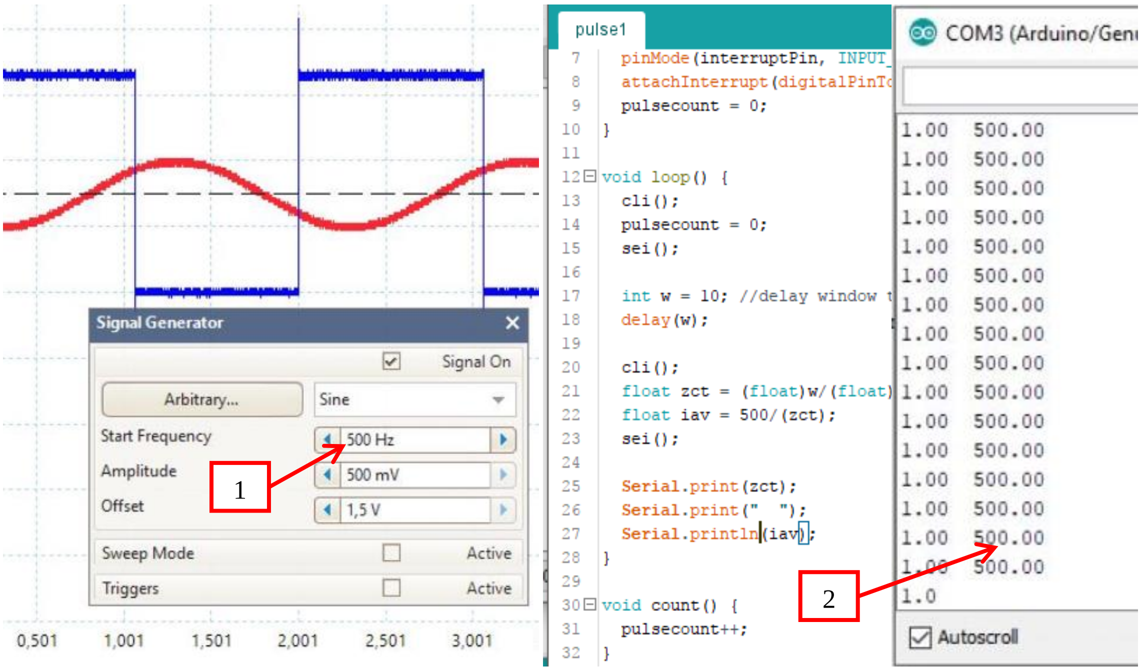


Figure 34. Piece wise linear spice simulation. Rpm sensor signal (red) vs. zcimpulse signal output (green).

With 2SD965 NPN transistors available for prototyping, author added its model to spice program. Simulations yielded promising results as the ZC events are detected with little delay and hysteresis. But prototyping was not carried out due to not available resources for testing NPN transistor’s functioning according to specification. [40]

Further low-cost options where tried and 0.5 € integrated circuit SN74HC14N (SN74 from here on) was obtained for carrying out feasibility tests despite while actually above circuit showed most promising accurate edge detection out of simulated and tested possible implementations.



Signal Generator

☑ Signal On

Sine

Start Frequency 500 Hz

Amplitude 500 mV

Offset 1,5 V

Sweep Mode □ Active

Triggers □ Active

```
  pinMode(interruptPin, INPUT_
  attachInterrupt(digitalPinTo
  pulsecount = 0;
}

void loop() {
  cli();
  pulsecount = 0;
  sei();

  int w = 10; //delay window t
  delay(w);

  cli();
  float zct = (float)w/(float)
  float iav = 500/(zct);
  sei();

  Serial.print(zct);
  Serial.print("  ");
  Serial.println(iav);
}

void count() {
  pulsecount++;
}
```

```
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.00   500.00
1.0
```

☑ Autoscroll

Figure 35. SN74 tested with sinusoidal signal, input (red) vs. output (blue). Input frequency (1) vs. calculated frequency (2) from SN74 output signal.

The output of SN74 was fed into ATMEGA328P controller and using ‘pulse1.ino’ code the frequency was calculated. As seen on Figure 35, averaging period of *w = 10 ms* yielded high accuracy, whilst 500Hz signal crosses zero 10 times. It must be noted that given method uses time based windowing for testing the component.

Later implementations use event count based windowing, as the input frequency is changed over time, the amount of ZC averaged over time will remain a constant based on variable ‘*w*’. With sinusoidal from signal generator and window of 10 ZCs @ 2kHz, accuracy of +6/-1 Hz was witnessed.

2.4.2 Feasibility test setup

Hardware in the loop tests are carried out. It must be noted that shown method testing reliability is as dependant as its weakest link. In the shown setup the weakest link can be considered breadboard connections to components. For example, resistor’s connection wires have small diameter which can provide poor contact with breadboard connectors. This could be defeated by soldering connections of the circuit under test, on the other hand prolonging the testing time and time it takes to reconfigure the setup.

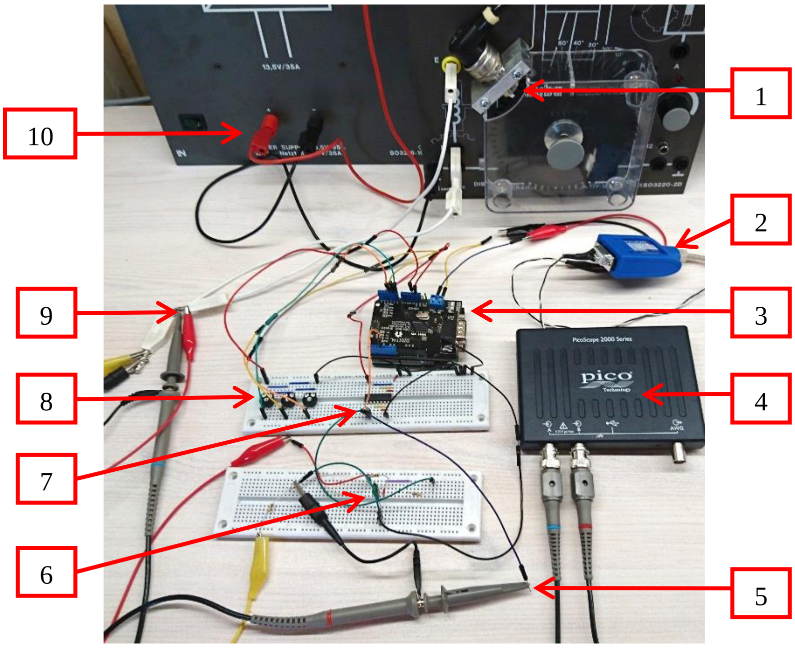


Figure 36. Feasibility test setup

The feasibility test setup as seen on Figure 36 indicates

1. Rpm sensor SO3220-2D
2. ValueCAN 3 CAN-USB interface
3. Arduino Uno rev 3 with Elecfreaks CAN shield
4. Picoscope 2206B
5. SN74 output measurement (CH2)
6. Input filter
7. SN74HC14N (SN74)
8. Voltage dividers for analogue inputs
9. Rpm sensor output measurement (CH1)
10. Power Supply SO3216-1C

2.4.3 Feasibility test results

Prototyping and testing reveals the feasibility of concepts previously imagined by modelling and architecting. A considerable amount of time can be spent with side-line activities for getting the system to work as expected.

Firstly, rpm sensor signal started to resonate in the circuit in a way that none of the previous measurements would have indicated as further described and illustrated below. Secondly, CAN interface with the MCP2515 was firing out 1st message buffer as fast as possible with no timing delay in between the messages.

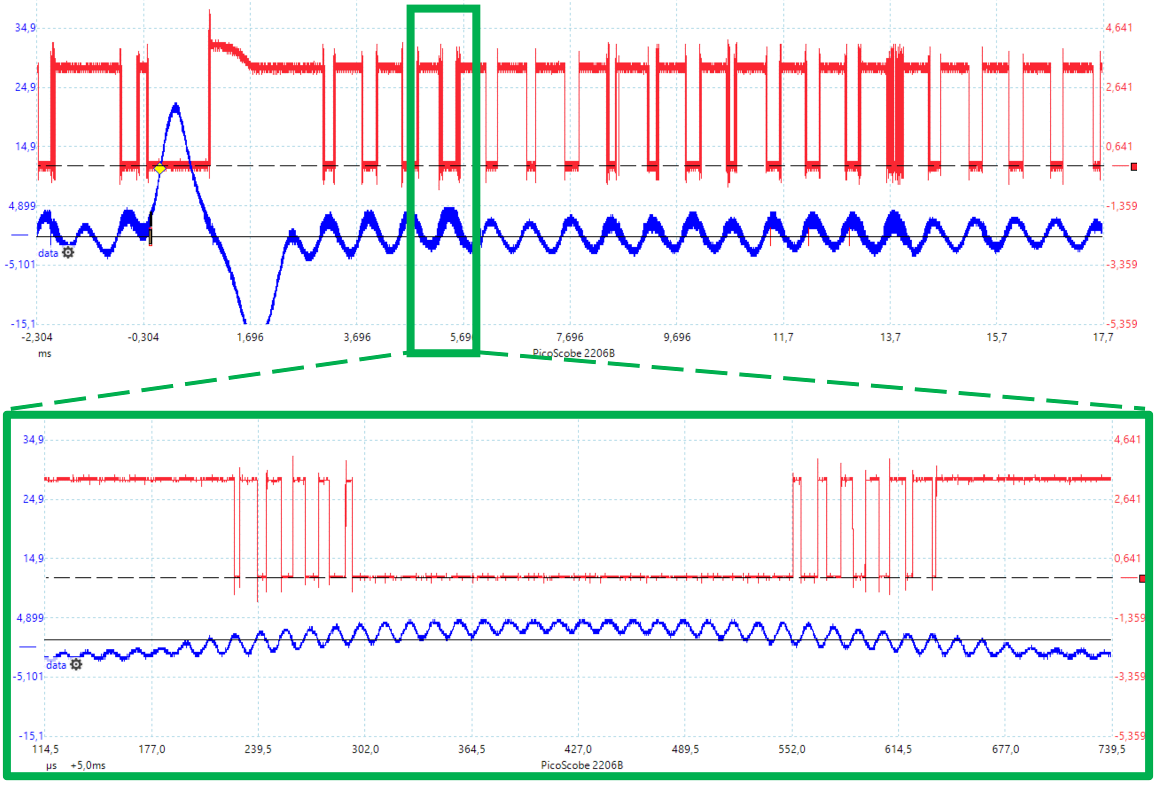
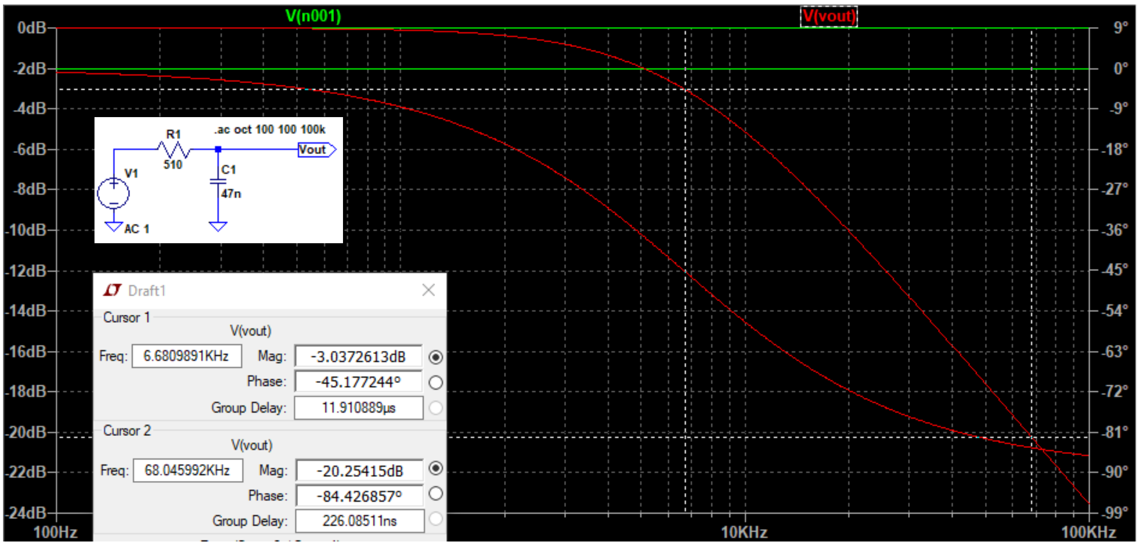


Figure 37. SN74 input (blue) vs. output (red)

First tests with feeding actual rpm sensor signal to SN74 resulted in highly inaccurate results of calculated rpm. Investigating input and output signals revealed parasitic components in the input signals approximately around 68 kHz which had high enough amplitude to cross over SN74 hysteresis threshold values. [41]

Figure 37 reveals that numerous close by impulses where triggered and fed to the controller where the algorithm calculated up to 80,000 rpm.

Passive RC filter is made with available components providing cutoff frequency of 6.7 kHz using 47 µF polyester film capacitor 2A473J and 510 Ω resistor.



Cursor 1

V(vout)

Freq: 6.6809891KHz Mag: -3.0372613dB ●

Phase: -45.177244° ○

Group Delay: 11.910889µs

Cursor 2

V(vout)

Freq: 68.045992KHz Mag: -20.25415dB ●

Phase: -84.426857° ○

Group Delay: 226.08511ns

Figure 38 RC filter characteristics

The RC low pass filter lower 68 kHz signal frequency by -20 dB which is sufficient to suppress the parasitic signal component described previously from triggering ZC counter around the hysteresis threshold levels of SN74.

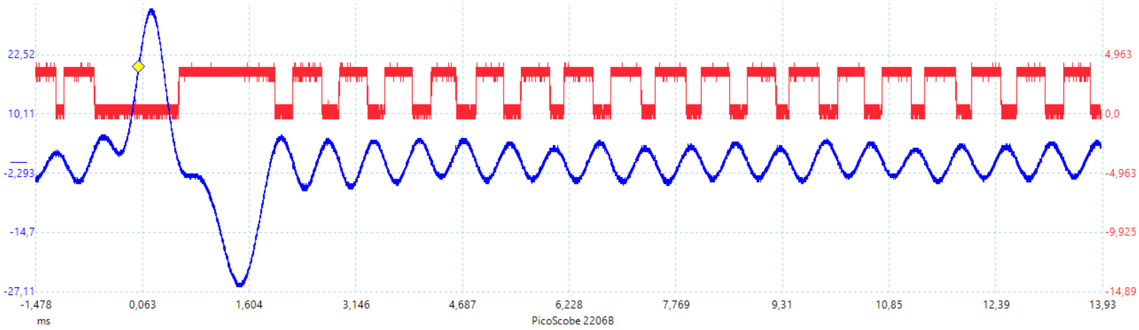


Figure 39. Input signal (blue) to SN74 treated with RC filter vs. output signal of SN74 (red)

As seen from Figure 39 the parasitic triggering of ZC counter is omitted by the use of passive low-pass filter of the rpm sensor signal.

The second side-line activity to getting the system working was related to the configuration of MCP2515 CAN interface as it was firstly tested separately from the other node and investigated with oscilloscope serial decoding tool.

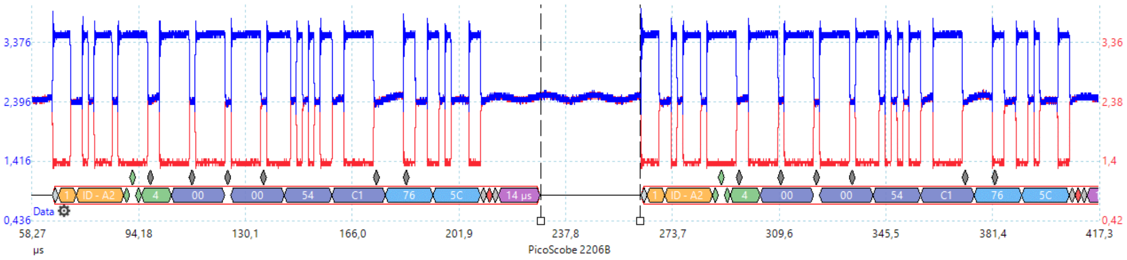


Figure 40. MCP2515 CAN messages transmitted back to back

As seen on Figure 40 the MCP2515 chip outputs the *canMsg1* to the bus with interval of 35 µs, but the *canMsg2* is not output at all, although in *feasibilityTest.ino*, found in appendix 4, it is specified to be output approximately every 10 ms, as also specified in paragraph 2.3.1.

Different possible causes were investigated of which most probable was the CANCTRL.OSM setting in the CAN control register which basically is binary setup register of 8 bits found in the MCP2515 chip. [42, p. 58]

All attempts to write the one-shot mode bit high failed since none of the MCP2515 libraries available had enabled this setting to be modified in embedded functions. Even if they might have been enabled, author did not succeed to verify CANCTRL registry state and any changes to it from behaviour of sending out the messages.

MCP2515 started to transmit messages as intended by trial and error as author wanted to verify if another CAN interface can receive these back to back transmitted messages. Thus the ValueCAN3 was added to the network as receiving node. As of that moment, the MCP2515 started to convey the messages in desired manner with 10ms delay for messages ID=1A2 and 1s delay for ID=2B1.

The reason of constant transmission of canMsg1 is revealed by investigating the MCP2515 transmit message flowchart. [42, p. 17]

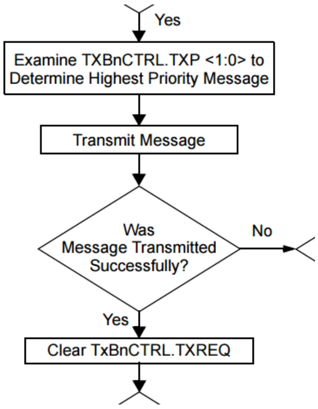

Yes
Examine TXBnCTRL.TXP <1:0> to Determine Highest Priority Message
Transmit Message
Was Message Transmitted Successfully?
No
Yes
Clear TxBnCTRL.TXREQ

Figure 41. MCP2515 transmit message flowchart

After user code with the help of libraries has written ID and Data via SPI to MCP2515 transmit buffers, the chip logic investigates for the message in available buffers.

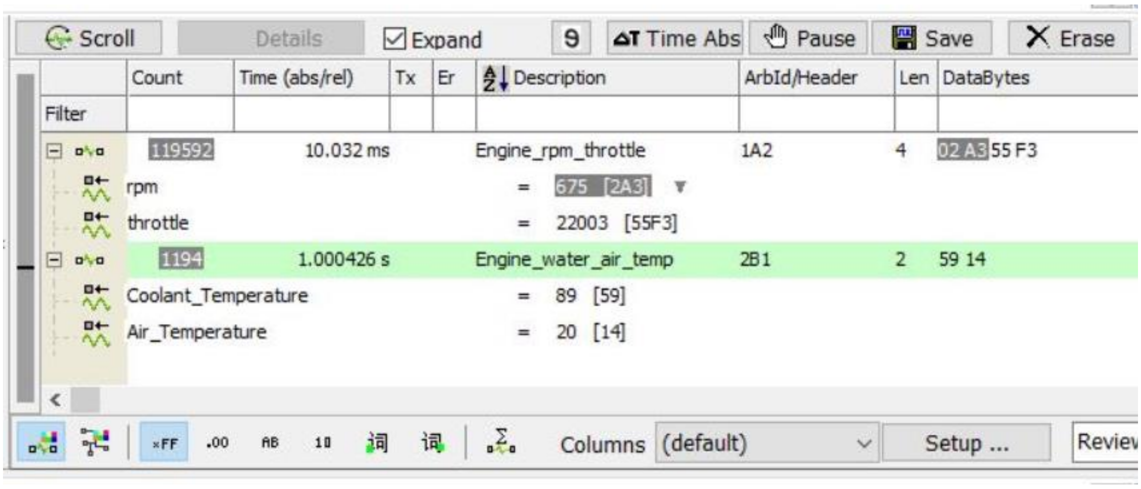
canMsg1 resides in the first buffer and canMsg2 in the second. Out of the two messages generated, message with ID = 1A2 is the one with highest priority. The message is transmitted, but not successfully and T1B3CTRL.TXREQ bit is not written low. Following the flowchart further to the right in the documentation, it is evident that the message error generates an interrupt which instantaneously triggers another cycle of transmitting the message. This cycle then continues as long as power supply to the chip is turned off or another functioning node is introduced to the network.

Above description is also externally visible from the CAN shield indicator LEDs as the interrupt LED is constantly high.

The successful message transmission requires the ACK bit in the can message to be written dominant by other nodes in the network. This is not done by oscilloscope which only decodes the physical level voltage values but doesn’t communicate in the Network.

By adding a second node with the same CAN protocol and parameters communication by MCP2515 took desired form, since now the second node feeds back the embedded network communications diagnostic feature in the form of dominant ACK bit. Of course the confirming ACK bit means receiving and interpreting correctly the entire message with also a matching CRC code.

Finally, the feasibility test setup can be tested to see if the input values are correctly sampled and in desired time sent across network.



	Count	Time (abs/rel)	Tx	Er	Description	ArbId/Header	Len	DataBytes
Filter								
⊟	119592	10.032 ms			Engine_rpm_throttle	1A2	4	02 A3 55 F3
	rpm				= 675 [2A3]			
	throttle				= 22003 [55F3]			
⊟	1194	1.000426 s			Engine_water_air_temp	2B1	2	59 14
	Coolant_Temperature				= 89 [59]			
	Air_Temperature				= 20 [14]			

Columns (default)

Figure 42. Vehicle Spy 3 SW interface

The signals to be received are also specified in the Messages Editor of Vehicle Spy for more convenient investigation of the received values as seen on Figure 42.

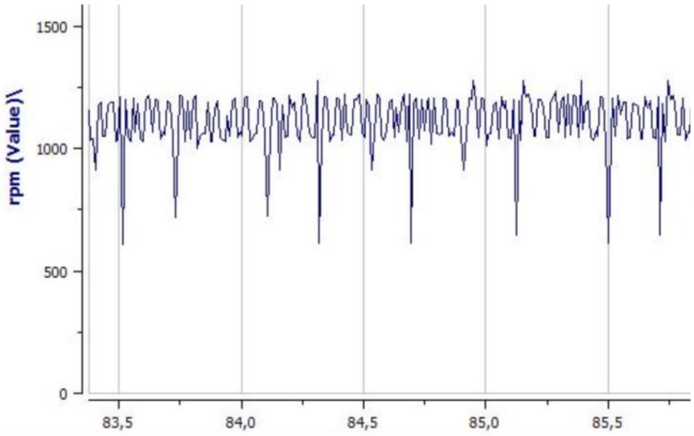


Figure 43. Received data plotted across time

With the averaging window size of w = 10, the rpm readout does not follow the simulation results as shown in paragraph 2.1.9, where around 1% accuracy was achieved during t1-t56. As seen on Figure 43 the rpm readout is as low as 600 and as high as 1200 with high degree of change. The actual input rpm verified with oscilloscope is around 1017 rpm in this test case. The repeating nature of lower rpm value peaks seem to be same as in simulation where the t_0 part of the input signal generates a longer period of ZC yielding in lower rpm count. Slope detection algorithms could be implemented programmatically to eliminate these deviations and also provide TDC information. It could be stated that around 20% accuracy was achieved around 1k rpm. As the rpm increases the accuracy decreases.

Further *w* values were tested and results showed more accurate rpm readout as the averaging window was wider over more amount of ZC events, as expected. An online HW in the loop demo link and description is provided in Appendix 6.

The throttle signal and air and coolant temperature signals behaved as expected.

2.4.4 MVP feasibility testing conclusions

The highest amount of scrutiny was turned on the rpm sensor signal, which proposed most challenges throughout this thesis. Other i/o requirements were mapped and some of them tested as well. Actual implementations demand higher degree of accuracy where the ZC estimation w=2, where the falling edge of the impulse signal is estimated in time, to achieve up to 300 µs precision [12, p. 205] of estimating individual passing of trigger teeth. Which has been proven to be unreachable precision in the scope of MVP feasibility testing.

Accuracy requirements are application specific, and accuracy which can be provided is highly constrained to the hardware. As this development process took existing equipment to be upgraded, the components to be used were dictated by the task, as it would not be the case in normal development process where the requirements and system development finally define the components to be used.

How fast and accurate signal is needed is not specified in the scope of MVP development, but the feasibility tests show that with relatively low cost means, it is possible to use existing equipment to practice basics of automotive electronic control systems design with various types of signals shared over data bus with various timing schemes. In the future practical tasks developments for the students, the input parameters could be acquired from industry contacts to give more meaningful context.

Author demonstrated feasibility with general application development kit. Automotive electronics development kits listed in Chapter 2.3 would provide embedded networking interfaces and various specialized input and outputs and internal control mechanisms for practicing automotive ECU design for fulfilling the high level requirements. In an advanced automotive application there are thousands of signals and I/O. In this feasibility test only two messages and four signals were transmitted, which was manageable with general purpose controller.

3 Summary

The main goals shown as relevant activities presented in Figure 1 green field were fulfilled. Modelling of the signals is a continuous process where many iterations are to be undertaken if higher accuracy is to be achieved with existing equipment. Also the proposed slope recognition algorithms for TDC detection can be started based on the models developed throughout this thesis. Further peripheral circuits design for I/O conditioning can give wider selection of possible implementations on any given system to introduce variety of concepts instead of narrow one configuration option.

Documentation and specifications generated throughout this thesis will be useful in the next steps of MVP development as the V-shaped development model is restarted from system design with documented knowledge of existing equipment.

Theoretical overview of electronic system designed revealed that:

- the number and complexity of automotive electronic systems is ambiguous and ever increasing;
- the variety and scope of requirements and standards is over whelming and should be considered part of the teaching program in future specialty.

Mapping and creating new ECU platforms for existing equipment revealed:

- independent documentation of electrical characteristics and malfunctions;
- that equipment poses different challenges with the types of signals and control functions needed to be implemented;
- modelling of the equipment provides basic properties of its behavior which are useful within the next stages of development;
- the range of available development platforms is wide with hundreds of options with relatively low cost from ECU controller or processor chip manufacturers.

Creating specifications for:

- ECU architectures gave definitive overview of necessary I/O and possible required functionality;
- CAN messages shows an example implementation of networking the domains in the project scope.

Feasibility testing exposed:

- possible pitfalls of which cannot be foreseen as the actual implementation of systems shows the ideal notion of previously conceived models;
- the possibility of using ECU development kits on existing study equipment;
- that imposed high-level requirements can be fulfilled.

Based on the results of documenting state of art of automotive technology, developing MVP based on existing equipment and conducting feasibility tests , author can decisively propose to pursue implementation of specified architectures with relatively low cost to achieve imposed requirements for teaching forward-looking aspects of automotive electronics.

4 Bibliography

[1] N. Navet and F. Simonot-Lion, Automotive Embedded Systems Handbook, CRC Press, 2009.

[2] BOSCH, Automotive Handbook 8th edition, Robert Bosch GmbH, 2011.

[3] R. Viereckl, D. Ahlemann and e. al, "Connected car report 2016: Opportunities, risk, and turmoil on the road to autonomous vehicles," 2016. [Online]. Available: http://www.strategyand.pwc.com/reports/connected-car-2016-study. [Accessed 12 10 2016].

[4] H. Vennikas, *Lõputöö: Auto keskkonnamõju tulevikutrendid,* Tallinn: Tallinna Tehnikakõrgkool, 2012.

[5] M. Poppendieck and T. Poppendieck, The Lean Mindset: ask the right questions, Massachusetts: Poppendieck LLC, 2014.

[6] Tallinna Tehnikakõrgkool, Aastaraamat 2016, Tallinn: Tallinna Tehnikakõrgkool, 2017.

[7] Kutsekoda, "Autoinsener tase 6 kutsestandard," 20 05 2014. [Online]. Available: http://www.kutsekoda.ee/et/kutseregister/kutsestandardid/10518151/pdf/autoinsener-tase-6.4.et.pdf. [Accessed 23 11 2016].

[8] Tallinn University Of Applied Sciences, "Curriculum subjects of curriculum 'Automotive Engineering 2016'," [Online]. Available: https://tktk.ois.ee/en/curriculum-subject?curriculum_id=16&year=2016. [Accessed 24 11 2016].

[9] Tallinn University Of Applied Sciences, "Autotehnika," [Online]. Available: http://www.tktk.ee/sisseastujale/oppekavad/autotehnika. [Accessed 24 11 2016].

[10] M. Kato, "Historical Overview of Electronics and Automobiles: Breakthroughs and Innovation by Electronics and Electrical Technology," in *Encyclopedia Of Automotive Engineering*, Wiley, 2014.

[11] W. Ribbens, Understanding Automotive Electronics, An Engineering Perspective, Seventh Edition, Elsevier, 2012.

[12] BOSCH, Automotive Electrics Automotive Electronics, Robert Bosch GmbH, 2007.

[13] G. Gates, J. Ewing, K. Russell and D. Watkins, "How Volkswagen has grappled with its diesel scandal," The New York Times, 11 01 2017. [Online]. Available: http://nyti.ms/1jQyOC6.

[14] C. Miller and C. Valasek, "Illmatics," 10 08 2015. [Online]. Available: http://illmatics.com/Remote%20Car%20Hacking.pdf. [Accessed 05 01 2017].

[15] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham and S. Savage, "Autosec," 2010. [Online]. Available: http://www.autosec.org/pubs/cars-oakland2010.pdf. [Accessed 05 01 2017].

[16] Mcafee, "Mcafee," 2015. [Online]. Available: http://www.mcafee.com/de/resources/white-papers/wp-automotive-security.pdf. [Accessed 05 01 2017].

[17] International Organization of Standardization, "ISO and road vehicles," [Online]. Available: http://www.iso.org/iso/iso_and_road-vehicles.pdf. [Accessed 02 01 2017].

[18] International Organization of Standardization, "Standards catalogue ISO/TC 22 - Road vehicles," [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_tc_browse.htm?commid=46706. [Accessed 10 01 2017].

[19] International Automotive Task Force, [Online]. Available: http://www.iatfglobaloversight.org/docs/IATF%2016949%20Transition%20Strategy%20and%20Requirements_REV02.pdf. [Accessed 15 01 2017].

[20] Autosar, "Basics," [Online]. [Accessed 15 12 2016].

[21] G. Ulsoi, H. Peng and M. Cakmakci, Automotive Control Systems, Cambridge University Press, 2012.

[22] J. W. Eaton, D. Bateman, S. Hauberg and R. Wehbring, "GNU Octave version 4.2.0 manual: a high-level interactive language for numerical computations," 2016. [Online]. Available: http://www.gnu.org/software/octave/doc/interpreter.

[23] GNU Octave, "Polynomial Interpolation," [Online]. Available: https://www.gnu.org/software/octave/doc/interpreter/Polynomial-Interpolation.html#Polynomial-Interpolation. [Accessed 29 March 2017].

[24] J. O. Rawlings and S. G. D. D. A. Pentula, Applied regression analysis: a research tool. — 2nd ed, New York: Springer-Verlag, 1998.

[25] Infineon, "KIT_AK_SMALL_ENGINE," [Online]. Available: https://www.infineon.com/cms/en/product/automotive-system-ic/engine-management-ic/channel.html?channel=db3a3043394427e401394d609374273b#ispnTab4. [Accessed 08 05 2017].

[26] NXP, "S32K144EVB: S32K144 Evaluation Board," [Online]. Available: http://www.nxp.com/products/automotive-products/microcontrollers-and-processors/arm-mcus-and-mpus/s32-processors-microcontrollers/s32k144-evaluation-board:S32K144EVB. [Accessed 08 05 2017].

[27] STMicroelectronics, "32-bit Power Architecture MCU for Automotive Powertrain Applications," [Online]. Available: http://www.st.com/content/st_com/en/products/automotive-microcontrollers/spc5-32-bit-automotive-mcus/spc5-mcus-for-high-performance-applications/spc56-m-line-mcus/spc563m64l5.html. [Accessed 08 05 2017].

[28] STMicroelectronics, "Evaluation board for L9779WD-SPI," [Online]. Available: http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/automotive-ic-eval-boards/eval-l9779wd-spi.html. [Accessed 08 05 2017].

[29] elinux.org, "R-Car/Boards/M3SK," [Online]. Available: http://elinux.org/R-Car/Boards/M3SK#Introduction. [Accessed 08 05 2017].

[30] NXP, "MAC57D5xxEVB: MAC57D5xx Development Platform," [Online]. Available: http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/mac57dxxx/mac57d5xx-development-platform:MAC57D5xxEVB. [Accessed 08 05 2017].

[31] STMicroelectronics, "Evaluation board with STM32F469NI MCU," [Online]. Available: http://www.st.com/en/evaluation-tools/stm32469i-eval.html. [Accessed 08 05 2017].

[32] Renesas, "QB-R5F10PPJ-TB," [Online]. Available: https://www.renesas.com/en-eu/products/boards-and-kits/boards-and-kits/device/QB-R5F10PPJ-TB.html. [Accessed 08 05 2017].

[33] Infineon, "SPOC+ EVALUATIONKIT1," [Online]. Available: https://www.infineon.com/cms/en/product/evaluation-boards/SPOC%2B+EVALUATIONKIT1/productType.html?productType=db3a3044365308540136638eb5566738#ispnTab1. [Accessed 08 05 2017].

[34] Infineon, "KIT_XC2288H_SK," [Online]. Available: http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XC2288H_SK/productType.html?productType=db3a3044243b532e0124de3132e26ada. [Accessed 08 05 2017].

[35] NXP, "KIT12XS6EVM: Reference design - Gen4 Lighting," [Online]. Available: http://www.nxp.com/products/automotive-products/energy-power-management/can-transceivers/reference-design-gen4-lighting:KIT12XS6EVM?fpsp=1&tab=Overview. [Accessed 08 05 2017].

[36] Wikipedia, "OBD-II PIDs," [Online]. Available: https://en.wikipedia.org/wiki/OBD-II_PIDs. [Accessed 08 05 2017].

[37] A. Ghassaei, "Arduino Frequency Detection," [Online]. Available: http://www.instructables.com/id/Arduino-Frequency-Detection/. [Accessed 04 03 2017].

[38] W. R. W, "Simple Methods for Detecting Zero Crossing," *IECON '03. The 29th Annual Conference of the IEEE,* vol. Vol.3, pp. 2477-2481, 2003.

[39] J. Hearfield, "Schmitt trigger design," [Online]. Available: http://www.johnhearfield.com/Eng/Schmitt.htm. [Accessed 29 04 2017].

[40] Panasonic, "2SD965 Datasheet," [Online]. Available: http://rtellason.com/transdata/2sd965.pdf.

[41] Texas Instruments, "SN74HC14 Datasheet," [Online]. Available: http://www.ti.com/lit/ds/symlink/sn74hc14.pdf. [Accessed 29 04 2017].

[42] Microchip, "MCP2515 Datasheet," [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/21801e.pdf. [Accessed 02 05 2017].

Appendix 1 – crank_speed_signal_2pi_radians_ea.m

```
% crank_speed_signal_2pi_radians.m

a = 1;                  % signal amplitude
n = 32;                 % samples per signal period
pi2 = 2*pi;             % constant for shorter formulas
tf = 60;                % low p-p signal periods during pi2 rad turns
ea = 0.005;             % evaluation precision rad
sf = 0.433;             % scaling factor for downsampled data
bn = 20;                % number of histogram bins
rm = 5;                 % randomization multiplier
rs = a/6;               % randomization scale
xr = randn(1,70)*rs;    % randomizer array for lowpp signal part

% load actual measurements downsampled to 1258 samples, scale to sf
% link to data file in appendix 6
y0 = importdata("measured_signal_n1258_values_row.csv")*sf;

ap1 = (22*pi)/180;      % high p-p angular proportion in radians
ap2 = (6.5*pi)/180;     % angular proportion of t1
ap4 = (7.5*pi)/180;     % angular proportion of t56
ap3 = pi2-(ap1+ap2+ap4);% angular proportion of t2...t55


%find angular divisions of x axis and join to one array:
%      |start           |smpl duration    |end
x1 = 0                  :ea               :ap1;
x2 = ap1                :ea               :ap1+ap2;
x3 = ap1+ap2            :ea               :pi2-ap4;
x4 = pi2-ap4            :ea               :pi2;
x5 = [x1 x2 x3 x4];

%calculate and join different signal parts:
y1 = 8*a*sin((pi2/ap1)*x1+0.06);    % high p-p signal calculation
y2 = a*sin((pi2/ap2)*x2)+xr(round(x2*rm));      % t1
y3 = a*sin(tf*x3+4.3)+xr(round(x3*rm));         % t2-t55
y4 = a*sin((pi2/ap4)*x4+3)+xr(round(x4*rm));    % t56
y5 = [y1 y2 y3 y4];                 % join end to end y arrays

% peak detection
[pks1 xi1] = findpeaks( abs(y0(75:1258)),
                                    'MinPeakHeight',0.3,
                                    'MinPeakDistance',8
                                    );
[pks2 xi2] = findpeaks( abs(y5(75:1258)),
                                    'MinPeakHeight',0.3,
                                    'MinPeakDistance',8
                                    );

pks1m = mean(pks1);           % mean of data peaks t1-t56
pks2m = mean(pks2);           % mean of model peaks t1-t56

subplot(2,3,1:3)
```

```
plot(x5,y5,'c','linewidth',2,x5,y0,'ko','markersize',1)
xlabel('trigger wheel angle, rad')
ylabel('signal amplitude')
legend('model','data')
title('model and downsampled data plotted across 2pi rad turns')
ylim([-10 10])
xlim([0 pi2])

subplot(2,3,4)
hist(pks1,bn)
title({'data absolute peaks'; sprintf('mean: %d', pks1m)})
ylabel('number of occurences')
xlabel('absolute peak value of signal')

subplot(2,3,5)
scatter(y5(75:1258),y0(75:1258),8);
xlim([-2 2])
xlabel('model signal values')
ylim([-2 2])
ylabel('data signal values')
title({'data vs. model';'scatter plot, t1-t56'})

subplot(2,3,6)
hist(pks2,bn)
title({'model absolute peaks';sprintf('mean: %d', pks2m)})
ylabel('number of occurences')
xlabel('absolute peak value of signal')


% - END OF FILE - crank_speed_signal_2pi_radians.m
```

Appendix 2 – crank_cam_zc_av.m

```
% crank_cam_zc_av.m
%---------------------------------------------------------------------
pl = pi*4;        % optional plot length from 0 to 4*pi
rpm = 1200;       % input for visualizing zc time on calculated iav
av = rpm*(pi/30); % angular velocity from rpm
w = 10;           % window size for averaging zc distances
txt = 0;          % options: 1-show iav; 2 - show difference iav
%---------------------------------------------------------------------
% define constants
a = 2;                  % signal amplitude
n = 32;                 % samples per signal period
pi2 = 2*pi;             % constant for shorter formulas
pi4 = 4*pi;             % constant for shorter formulas
tf = 60;                % low p-p signal periods during pi2 rad turns
xr = randn(1,140)*a/6;  % randomizer array for lowpp signal part
ea = 0.005;             % evaluation precision rad
hv = 13.5;              % signal high value

ap1 = (22*pi)/180;          % high p-p angular proportion in radians
ap2 = (6.5*pi)/180;         % angular proportion of t1
ap4 = (7.5*pi)/180;         % angular proportion of t56
ap3 = pi2-(ap1+ap2+ap4);    % angular proportion of t2...t55
ap5 = (109*pi)/180;         % angular proportion of cam high value
ap6 = (65*pi)/180;          % angular proportion of cam signal offset

%---------------------------------------------------------------------
%find angular divisions of x axis and join to one array:
%   |start      |smpl dur   |end
x1 = 0          :ea     :ap1;
x2 = ap1        :ea     :ap1+ap2;
x3 = ap1+ap2    :ea     :pi2-ap4-ea; % -ea for array lengths
x4 = pi2-ap4    :ea     :pi2;
x6 = 0          :ea     :ap5;
x7 = ap5        :ea     :pi4;
x8 = [x6 x7];
x9 = 0          :ea     :ap6;

% optain array size for populating y6 y7 y9
x6len = length(x6);
x7len = length(x7);
x9len = length(x9);

%calculate and join different signal parts:
y1a = 8*a*sin((pi2/ap1)*x1+0.06);           % high p-p signal
calculation
y2a = a*sin((pi2/ap2)*x2)+xr(round(x2*10));     % t1
y3a = a*sin(tf*x3+4.3)+xr(round(x3*10));        % t2-t55
y4a = a*sin((pi2/ap4)*x4+3)+xr(round(x4*10));   % t56
y5a = [y1a y2a y3a y4a];                   % join end to end ya arrays

%randomize again for second turn of crank
y2b = a*sin((pi2/ap2)*x2)+xr(round(x2*20));     % t1
y3b = a*sin(tf*x3+4.3)+xr(round(x3*20));        % t2-t55
y4b = a*sin((pi2/ap4)*x4+3)+xr(round(x4*20));   % t56
y5b = [y1a y2b y3b y4b];       % join end to end yb arrays
y5 = [y5a y5b];                % join ya yb
```

```
%make cam signal
y6(1, 1:x6len) = 13.5;  % cam signal high
y7(1, x7len) = 0;       % cam signal low
y8 = [y6 y7];           % join end to end cam signal

y9 = shift(y5, x9len);  % adjust crank to cam offset

%---------------------------------------------------------------------
% zero crossing and instantaneus angular velocity department is starts
here

zcide = 3*pi/180;           % ideal zc distance if trigger symmetrical
zc = zerocrossing(x8,y9);   % return x coordinates of zc in rad
zclen = length(zc);         % get length of zc array

zcdist = zc-(shift(zc,1));  % find distance of zc from previous zc
zcdist(1) = pi4-zc(zclen)+zc(1);    % find distance of first zc

%window function for averaging over w amount of zc for testing
zcwave = (zc-(shift(zc,w)))/w;      % calculate average distances

% repair first w array elements with overlap from end of pi4 region
for i=1:w
    zcwave(i) = (pi4-zc(zclen-(w-i))+zc(i))/w;
end

%---------------------------------------------------------------------
% instant angular velocity if the trigger wheel was spun in time

iav = zeros(1:1);
iaw = zeros(1:1);

for i=1:rows(zcdist);
    iav(i,1) = pi2/(120*((1/av)*(zcdist(i,1))));
    iaw(i,1) = pi2/(120*((1/av)*(zcwave(i,1))));
end

[ix, iy] = stairs(zc,iav);
[jx, jy] = stairs(zc,iaw);


%---------------------------------------------------------------------
% Plotting department starts here

zcx = interp1(x8,y9,zc);        % interpolate zc on x axis for
plotting

r = 5;      % amount of subplot rows
ms = 4;         % marker size for diamonds on plots

subplot(r,1,1)
plot(   x8,y9,
        zc,zcx,'bd','markersize',ms,'markerfacecolor','auto',
        x8,y8)
ylabel('signal amplitude')
axis('tight')
xlim([0 pl])

subplot(r,1,2:3)
plot(   zc,zcdist,'b--d','markersize',ms,'markerfacecolor','auto',
        zc,zcwave,'m-
d','markersize',ms,'markerfacecolor','auto','linewidth',1,
        [0 pl],[zcide zcide],'k')
ylabel('zc distance from previous')
ylim([0.045 0.06])
xlim([0 pl])

subplot(r,1,4:5)
plot(   [0 pi4],[av av],'k--',
        ix,iy,'b',
        jx,jy,'m','linewidth',2)
title(sprintf('input av = %i rad/s', av))
ylabel('calculated av, rad/s')
% for calculated values text, works good pl < 2 rad
    if (txt==1)
        text(   zc,iaw,num2str(iaw,'%i'),
                'horizontalalignment','right',
                'verticalalignment','top',
                'rotation',double(90),
                'color','magenta')
    elseif (txt==2)
        text(   zc,iaw,num2str(abs(av-iaw),'%i'),
                'horizontalalignment','right',
                'verticalalignment','top',
                'rotation',double(90),
                'color','magenta')
    endif
ylim([(av/2) (av*1.5)])
xlim([0 pl])
xlabel('crank angle, rad')

figure
dv = abs(av-iy);
dvm = median(dv);
dw = abs(av-jy);
dwm = median(dw);
semilogy(   ix, dv,'bd','markersize',ms,'markerfacecolor','auto',
                  jx,
dw,'md','markersize',ms,'markerfacecolor','auto',
                  [0 pi4], [dvm dvm],'b-','linewidth',2,
                  [0 pi4], [dwm dwm],'m-','linewidth',2);
legend(     'difference from input av',
            sprintf('difference with averaging w = %i',w),
            'median difference',
            'median difference with averaging');
axis('tight');
ylabel('abs(av-ia[v,w])')
xlabel('crank angle, rad')
title(sprintf('no w median = %.2d  averaged median = %.2d',dvm,dwm))


% - END OF FILE - crank_cam_zc_av.m
```

Appendix 3 - temp_NTC_plynomial_diff_score.m

```
%temp_NTC_and_PTC_polynomial_sum_squares.m
%define x axes
x = 20:10:160;
xe = -100:10:200;

%number of rows and columns in final plot window
r = 4;
c = 3;

%original measurement results for 10 C' steps from 20-160 C'
y =
[602,425,286.6,199.5,140.7,103.7,77.7,59.2,46.6,37.3,30.5,25.5,21.7,18
.8,16.5];
ym = mean(y);

% curve-fit a n-th order polynomial to match measurement results
p1 = polyfit(x,y,1);
p2 = polyfit(x,y,2);
p3 = polyfit(x,y,3);
p4 = polyfit(x,y,4);
p5 = polyfit(x,y,5);
p6 = polyfit(x,y,6);

% evaluate n-th order polynomial to x values representing degrees in
C'
yp1 = polyval(p1,x);
yp2 = polyval(p2,x);
yp3 = polyval(p3,x);
yp4 = polyval(p4,x);
yp5 = polyval(p5,x);
yp6 = polyval(p6,x);

% find sum of squared residuals
l = length(x);
sc1=0; sc2=0; sc3=0; sc4=0; sc5=0; sc6=0;

for i=1:l
      sc1 = sc1+power(abs(y(i)-yp1(i)),2);
      sc2 = sc2+power(abs(y(i)-yp2(i)),2);
      sc3 = sc3+power(abs(y(i)-yp3(i)),2);
      sc4 = sc4+power(abs(y(i)-yp4(i)),2);
      sc5 = sc5+power(abs(y(i)-yp5(i)),2);
      sc6 = sc6+power(abs(y(i)-yp6(i)),2);
endfor

%extrapolate polynomial evaluation
xe1 = polyval(p1,xe);
xe2 = polyval(p2,xe);
xe3 = polyval(p3,xe);
xe4 = polyval(p4,xe);
xe5 = polyval(p5,xe);
xe6 = polyval(p6,xe);

p = 1;

hold on
subplot(r,c,p)
```

```
plot(x,yp1,'-'); p++;
title(sprintf('n=1, SS=%d', round(sc1)));
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p)
plot(x,yp2,'-'); p++;
title(sprintf('n=2, SS=%d', round(sc2)))
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p)
plot(x,yp3,'-'); p++;
title(sprintf('n=3, SS=%d', round(sc3)))
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe1,'r'); p++;
title('n=1 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe2,'r'); p++;
title('n=2 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe3,'r'); p++;
title('n=3 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p)
plot(x,yp4,'-'); p++;
title(sprintf('n=4, SS=%d', round(sc4)))
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p)
plot(x,yp5,'-'); p++;
title(sprintf('n=5, SS=%d', round(sc5)))
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p)
plot(x,yp5,'-'); p++;
title(sprintf('n=6, SS=%d', round(sc6)))
ylim([0,600]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe4,'r');  p++;
title('n=4 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe5,'r');  p++;
title('n=5 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');

subplot(r,c,p) % extrapolation
plot(xe,xe6,'r');  p++;
title('n=6 extrapolation')
ylim([0,10000]); xlabel('temp, C'); ylabel('resistance, Ohm');
hold off

% - END OF FILE - temp_NTC_and_PTC_polynomial_sum_squares.m
```

Appendix 4 – pulse_1.ino

```
/* pulse_1.ino
 * Based on https://www.arduino.cc/en/Reference/attachInterrupt
 */

const byte interruptPin = 3;
volatile byte state = LOW;
volatile int pulsecount;

void setup() {
  Serial.begin (9600);
  pinMode(interruptPin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(interruptPin), count, CHANGE);
  pulsecount = 0;
}

void loop() {
  cli();
  pulsecount = 0;
  sei();

  int w = 10; //delay window time in millis
  delay(w);

  cli();
  float zct = (float)w/(float)pulsecount; //ZC time averaged over w
  float iav = 500/(zct);
  sei();

  Serial.print(zct);
  Serial.print("  ");
  Serial.println(iav);
}

void count() {
  pulsecount++;
}

// - END OF FILE - pulse_1.ino
```

Appendix 5 – feasibilityTest.ino

```
/* feasibilityTest.ino
 * Uses mcp2515 library: https://github.com/autowp/arduino-mcp2515
 */
#include <SPI.h>
#include <mcp2515.h>

struct can_frame canMsg1;       //struct can_frame canMsg
struct can_frame canMsg2;

MCP2515 mcp2515(10);            // SPI CS signal pin

volatile unsigned long t0;    // timer initial state
volatile unsigned long tw;    // timer after w zc
volatile unsigned long te;    // time evariable for calulating iav
volatile unsigned int ti;     // counter variable for edge events
const int w = 20;            // window size for counting zc
const byte interruptPin = 3;  // digital in pin for edge detection
volatile unsigned long t1, t2, m1r, m2r; //timing variables

void setup() {

  attachInterrupt(digitalPinToInterrupt(interruptPin),zcCount,
CHANGE); // Define interrupt settings
  pinMode(interruptPin, INPUT_PULLUP);

  canMsg1.can_id  = 0x1A2;    // define message ID's
  canMsg1.can_dlc = 4;        // define message DLC's
  m1r = 9700;                 // define message 1 rate in micros

  canMsg2.can_id  = 0x2B1;
  canMsg2.can_dlc = 2;
  m2r = 999970;

  while (!Serial);
  Serial.begin(500000);
  SPI.begin();
  delay(10);

  mcp2515.reset();                  // reset mcp2515
  mcp2515.setBitrate(CAN_500KBPS);  // use HS_CAN speed of 500 kbps
  mcp2515.setNormalMode();
  /* send or receive data, but another working node is needed for
   * writing ACK bit, mcp transmission request bit stays high if no
   * ACK bit is received and CAN message is transmitted constantly
with no timing.
   * With current libraries, writing MCP2515 CANCTRL_OSM high was not
a success*/

  delay(10);

  t0,t1,t2 = micros();
  ti = 1;                     // initiate ti counter

}
```

```
void zcCount(){ // interrupt event triggers this loop
  if(ti<w){
    ti++;
  }
  else if (ti==w){
    tw = micros();
    te = tw - t0;
    t0 = micros(); // write current time as start time
    tw = 0;        // initialize tw
    ti = 1;        // initialize i
  }
}

void loop() {

  if (t1+m1r<micros()){

    double iav = ((w)*1000000)/(4*te); // calculate iav based on w te
    int m1d0 = floor(iav/256);     // calculate rpm MSB
    int m1d1 = iav-(m1d0*256);     // calculate rpm LSB

    int th = analogRead(0)+21554;  // throttle 10bit value
    int m1d2 = floor(th/256);      // calculate throtte MSB
    int m1d3 = th-(m1d2*256);

    canMsg1.data[0] = m1d0;        // assign rpm MSB
    canMsg1.data[1] = m1d1;        // assign rpm LSB
    canMsg1.data[2] = m1d2;        // added 21554
    canMsg1.data[3] = m1d3;        // assign throttle MSB

    mcp2515.sendMessage(&canMsg1); // send 0x1A2 4 D0 D1 D2 D3
    t1 = micros();                 // write t1 current time

    Serial.print(iav);   // Debugging via serial monitor
    Serial.print("       ");
    Serial.print(m1d2);
    Serial.print("  ");
    Serial.print(m1d3);
    Serial.print("  ");
    Serial.println(th);
  }

  if (t2+m2r<micros()){

    int ct = map(analogRead(1), 0, 1023, 0, 255);  // coolant temp
    int at = map(analogRead(2), 0, 1023, 0, 255);  // air temp
    canMsg2.data[0] = ct;           // assign coolant temperature
    canMsg2.data[1] = at;           // assign air temperatuse

    mcp2515.sendMessage(&canMsg2);  // send 0x2B1 2 D0 D1

    Serial.print(ct);
    Serial.print("   ");
    Serial.print(at);
    Serial.println();

    t2 = micros(); // write t2 current time
  }
} // no delay is used to iterate loop

// - END OF FILE - feasibilityTest.ino
```

Appendix 6 – Online Package

1) Data files:

The actual measurement results of crank speed sensor downsampled to n = 1258 and used in *crank_speed_signal_2pi_radians_ea.m*

measured_signal_n1258_values_row.csv

available from

https://goo.gl/mypUv9

2) Video:

Online demonstration video of feasibility testing is accessible at:

https://goo.gl/j4CUiw

where starting from:

00:25 – setup of the oscilloscope according to crank signal

01:57 – calculation of actual rpm

04:06 – change w = 20 to w = 100

05:14 – demonstration of throttle signal response

06:15 – demonstration of temperature signal response

09:50 – as speed increases, accuracy decreases

21:58 – leaving MCP2515 to be the single communicating node in the network triggers its back to back message sending behaviour