ASCENT

Introduction to UAS Mechatronic Laboratory Tutorial – our way to teach Mechatronics

4^{th} Training in Rio de Janeiro, BRA

6^{th}-9^{th} of May 2019

Dr. Karl Reisinger & Thomas Lechner

Co-funded by the Erasmus+ Programme of the European Union

Agenda

We plan to introduce, how we are teaching mechatronics.

Agenda

- Definition of mechatronics
- Aim of out Mechatronic Tutorial Lab
- Place in curriculum
- Our educational example
- Content and process of the course
- Lessons learned
- Our next steps

Co-funded by the
Erasmus+ Programme
of the European Union

Why do we need mechatronics?

- „Mechatronics is the synergetic integration of mechanical engineering with electronic and intelligent computer control in the design and manufacturing of industrial products and processes“

  Definition in IEEE/ASME Trans. on Mechatronics (1996)

- Synergetic Integration

  Better solutions as each single domain.

- Mechanical Engineering

  … designs the thing itself.

- Electronic

  … to sense and to move.

- Intelligent Computer Control

  Makes the mechanical thing intelligent to perform complex tasks automatically.

- Industrial Products and Processes

  Intelligent products can transact complex processes.

Co-funded by the Erasmus+ Programme of the European Union

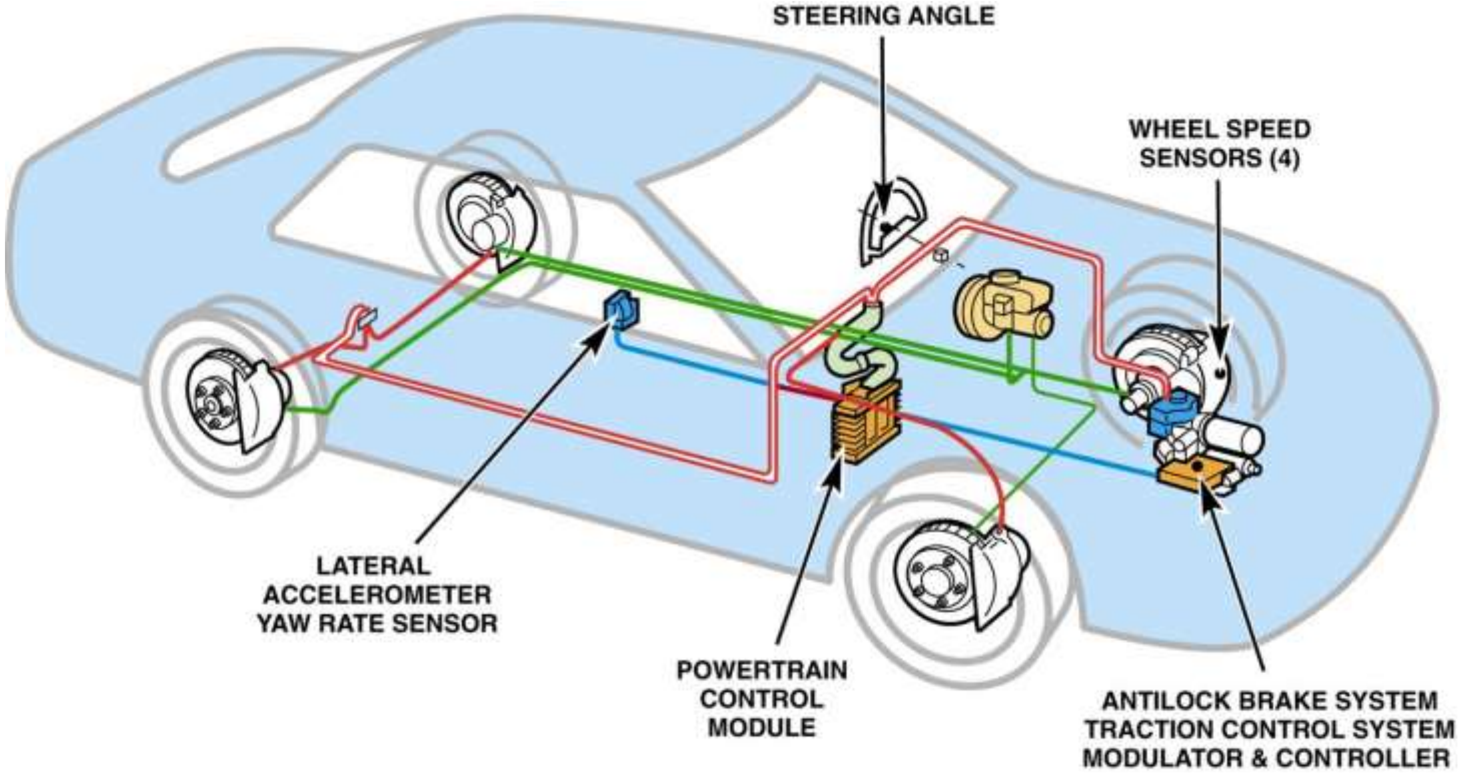
Why do automotive engineers need mechatronics?

- A modern vehicle is a complex mechatronic system.
- It consist of a lot of mechatronic sub-systems. For example:
  - Antilock brake System
  - Electronic Stability Control System
  - Engine Control Unit
  - Etc., etc., etc. …
- The different subsystem must communicate with each other.
  - CAN (Controller Area Network) - Bus
- Institute of Automotive Engineering → We must educate our students in mechatronics!

Co-funded by the
Erasmus+ Programme
of the European Union

Example: Antilock-brake-sytem

- Wheel speed and steering angle → measured
- ECU processes the data → electronic needs software to work
- If necsassary, brake pressure is controlled → electric actuator influences the hydraulic system

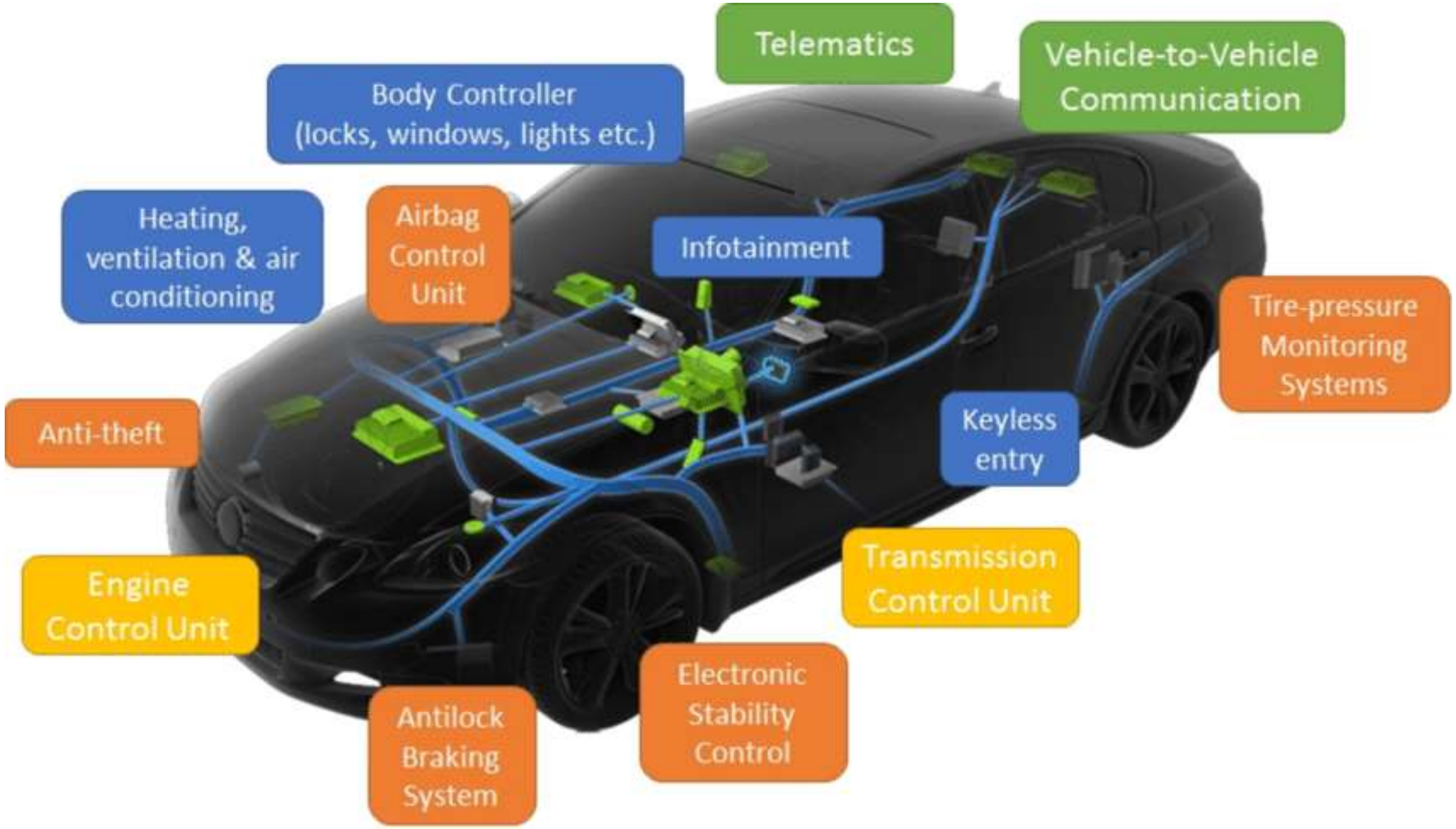


https://www.bwigroup.com/product/antilock-brake-systems/

Co-funded by the Erasmus+ Programme of the European Union

Overview - ECU‘s in a vehicle

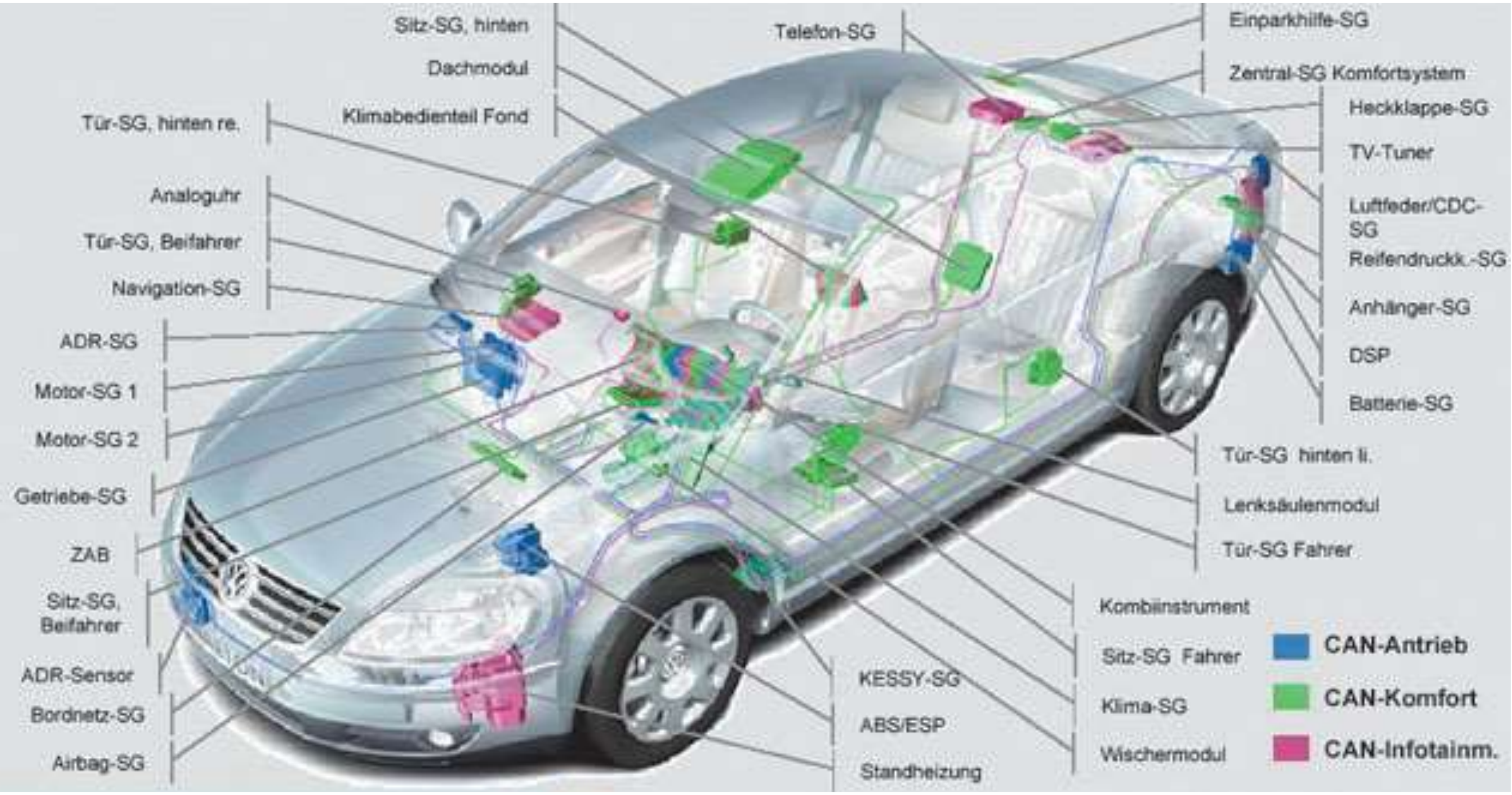
- Modern cars have a lot of ECU’s for different applications.
- They must communicate with each other.
- A network to exchange data is necessary.
- CAN-Bus, FlexRay, LIN



https://www.researchgate.net/publication/320198036_Security_Concerns_in_Co-operative_Intelligent_Transportation_Systems

Co-funded by the Erasmus+ Programme of the European Union

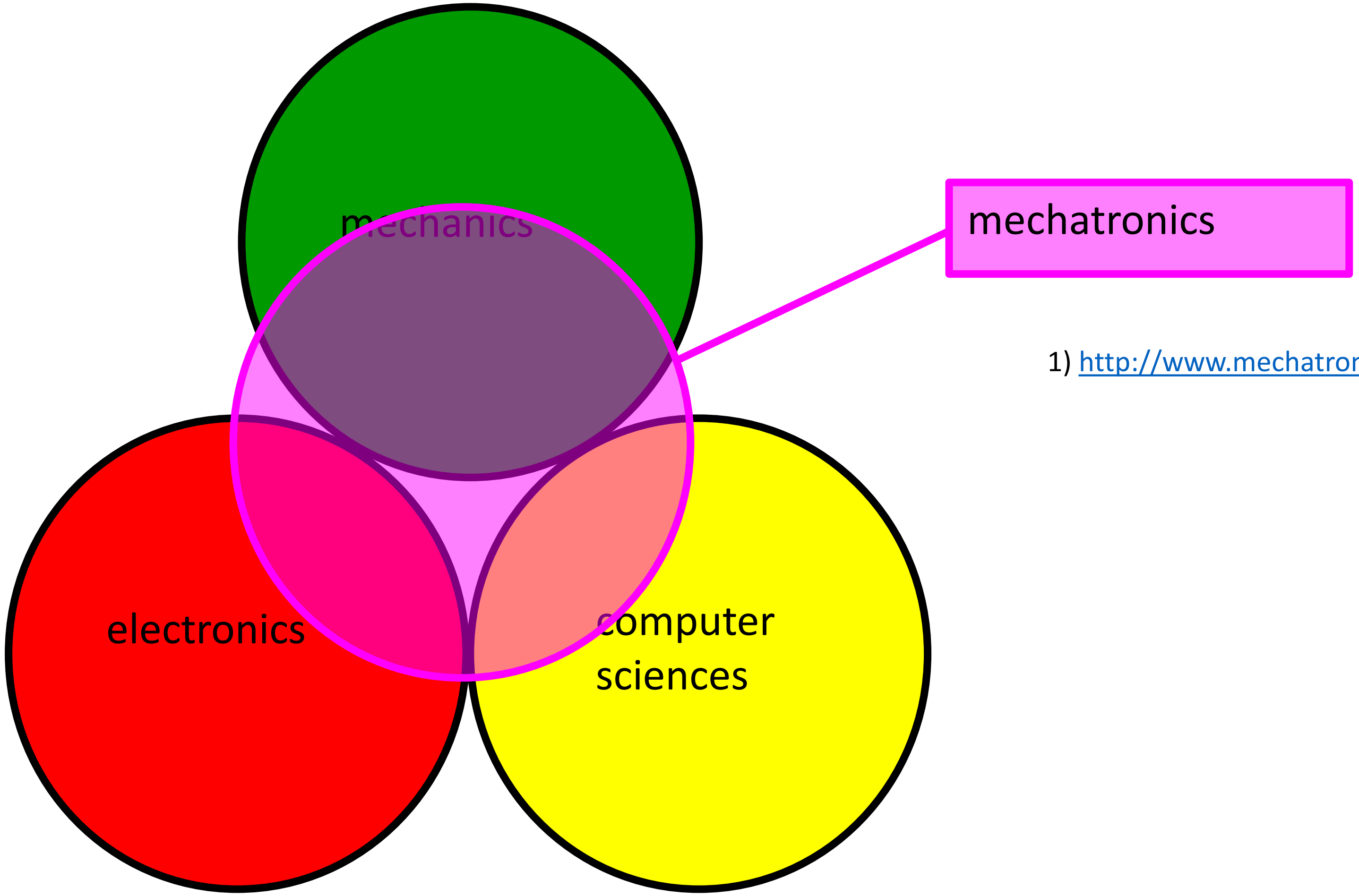
Number of ECU's in a vehicle?



https://www.heise.de/autos/artikel/Daten-unter-der-Haube-1012221.html?view=bildergalerie

Co-funded by the Erasmus+ Programme of the European Union

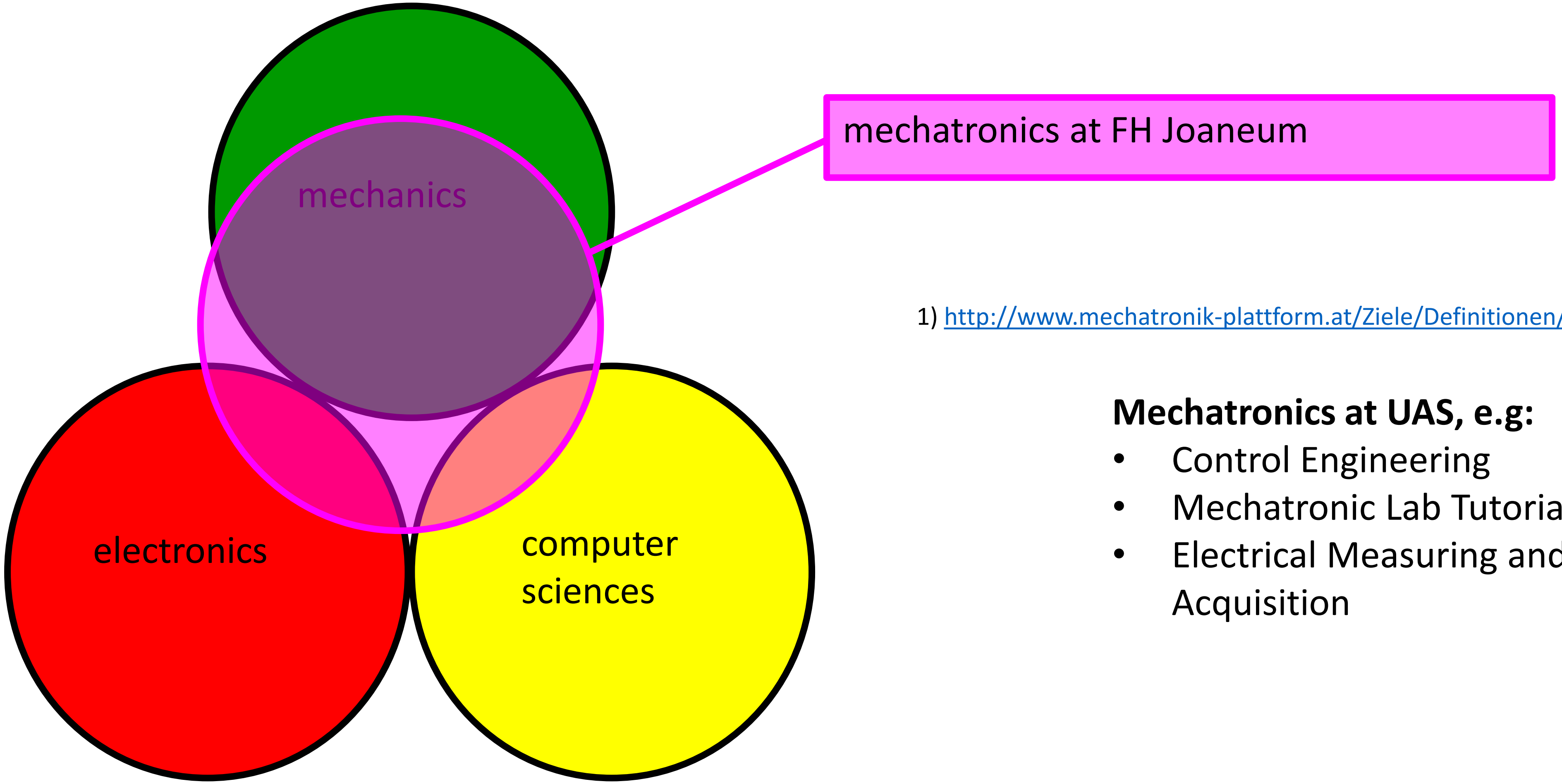
Mechatronics = Mechanics+Elektronics+Computer Sciences+AddedValue[1)]



1) http://www.mechatronik-plattform.at/Ziele/Definitionen/Viktorio-Malisa.pdf

Co-funded by the
Erasmus+ Programme
of the European Union

Mechatronics = Mechanics+Elektronics+Computer Sciences+AddedValue[1)]



1) http://www.mechatronik-plattform.at/Ziele/Definitionen/Viktorio-Malisa.pdf

Mechatronics at UAS, e.g:

- Control Engineering
- Mechatronic Lab Tutorial
- Electrical Measuring and Data Acquisition

Co-funded by the Erasmus+ Programme of the European Union

Aim of the Lab - General

- Understanding how mechatronic systems work
  - work with embedded systems

    linking mechanics, electrics and software
  - Couple mathematical/physical knowledge with software technology
  - Understand imperfections and limits

    A/D-, D/A converter, quantizing effects, cycle time influence
  - Encoding of signals

    Data types, fixed point arithmetic
- Get the real world into the PC
  - How to prepare a virtual prototype

    … something you can play with, to learn the system's behaviour,
    the way, to learn the details of the task and it's components (requirements),
    the tool to proof the feasibility.

Co-funded by the
Erasmus+ Programme
of the European Union

Aim of the Lab – Development Process

- Knowing how to develop automotive software for embedded systems
  - To applicate the V-model to mechatronics.
  - Using requirements to define the product before design phase.
  - To understand Model-based software development methodology using Matlab/Simulink.
  - Ability to develop embedded software: Step by Step Model in the Loop (MIL), Software in the Loop (SIL), Hardware
- Understanding automotive application processes for embedded systems
  - Set up CAN communication
  - Ability to parametrize and meter (=applicate) embedded systems using Can Calibration Protocol (CCP)

Co-funded by the Erasmus+ Programme of the European Union

Place in Curriculum

Prior Lectures

- Bachelor's Program
  - Engineering Mechanics (Statics, Kinetics), Mechanical Components
  - Introduction to Electrical Engineering, Electronic Systems, Electronic Lab Tutorials, Electrical Machines & Inverters,
  - Software Development ‚c#‘, MatLab/Simulink
  - Control Engineering

Lectures following

- Bachelor's Program
  - Measuring electrical and non-electrical Signals
- Master‘s Degree Program
  - Automotive Sensors/Actors,
  - Signal Processing, Digital Control Engineering,
  - Race Car Data Analysis
  - Electric Drive & Propulsion Systems, Energy Management & Storage Systems

Co-funded by the Erasmus+ Programme of the European Union

Our Object to grab the content

<u>Our main goal:</u> Control the torque-distribution between front an rear axel for a 4-WD car.

https://www.audi-mediacenter.com/de/quattro-mit-ultra-technologie-der-permanent-verfuegbare-allradantrieb-5503

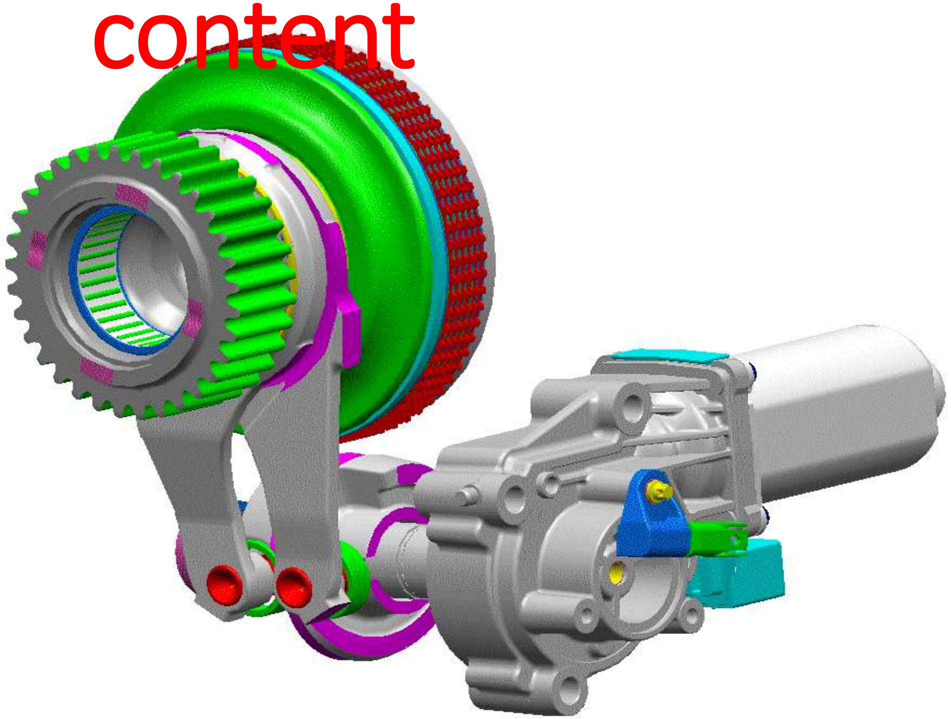
d by the Erasmus+ Programme

Our Object to grab the content

- Electromechanical control of an
  All Wheel Drive System,
  focused on component‘s control.
  - An electric motor can close a multi-plate clutch continuously to control the torque distribution between front and rear axle.
- Input (desired value)
  - Target torque to front.
- Output
  - position angle of the actuator.
- Concept
  - Model based feed forward torque controller using a closed loop position controller.
  - A state machine decides the steps to do (= control process).

Co-funded by the
Erasmus+ Programme
of the European Union

Our Object to grab the content



Quelle: Reisinger, Rühringer, Mathis: Modellgestützte Mechatronik-Systementwicklung für Allradanwendungen; TECHME, Sindelfingen Sept. 2007

Principe of varibale torque distribution
1.) Drivetrain:
http://www.digitalmediatechnik.de/Portfolio_12_ENG.html
2.) Detail ball ramp:
http://www.digitalmediatechnik.de/Portfolio_15_ENG.html

A pair of ball ramps

Ball ramp inner side

Co-funded by the
Erasmus+ Programme
of the European Union

Content 1

- Introduction Lessons
  - Systems concept
  - Modelling mechanics (Clutch, actuator mechanics incl. worm gear)
  - Control concept
    - State Machine to find initial position
    - Feed forward torque controller using mechanical characteristics
    - Position control algorithm using speed cascade
  - CAN
    - CAN principles
    - XCP, CCP protocol
  - Development Process: V-Model
- 5 Lab-Sessions in groups of max. 20 students
  - 1 Lab-Session: 5 times 45 minutes

Co-funded by the
Erasmus+ Programme
of the European Union

V-Modell

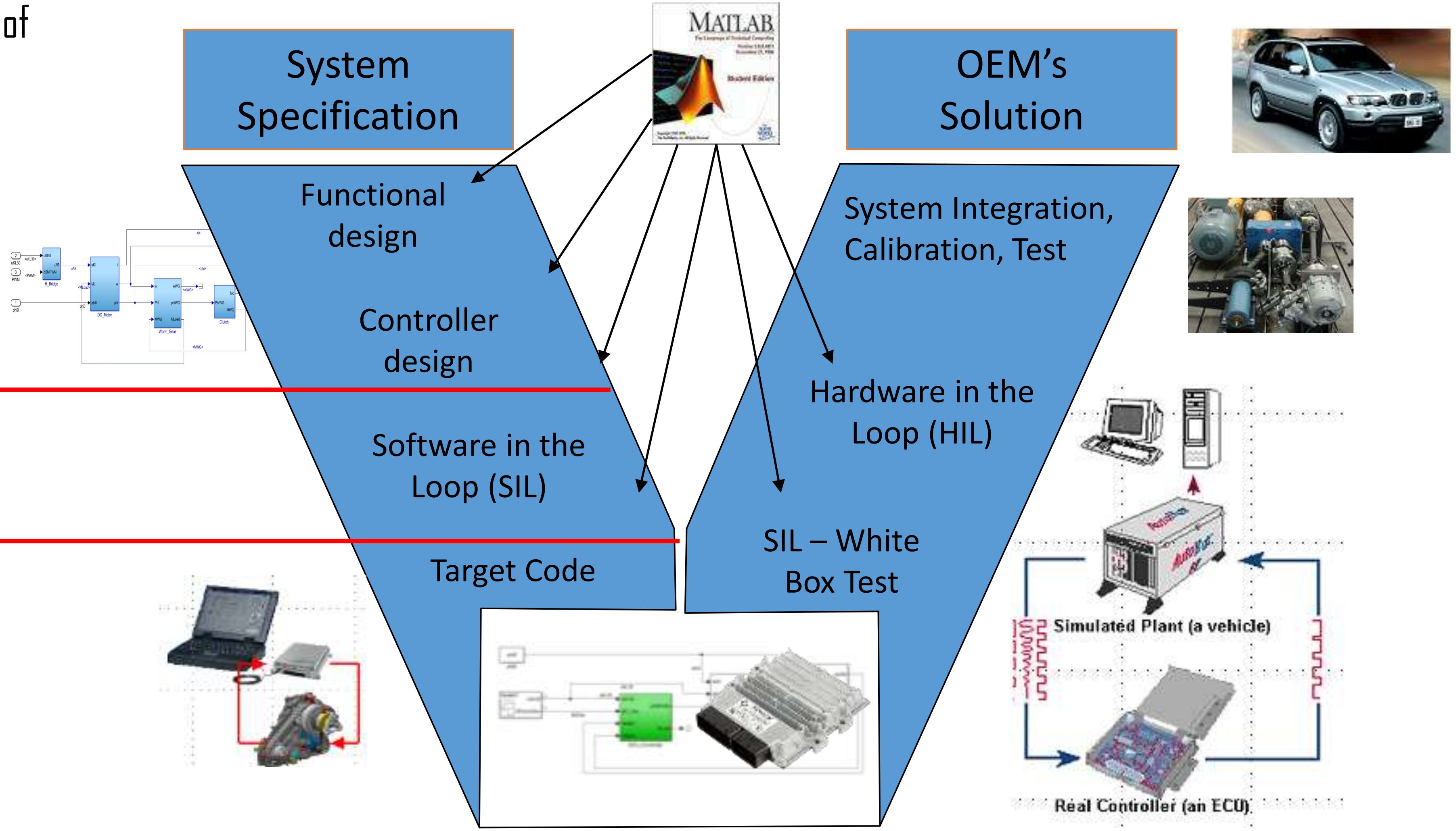
We concentrate on the tasks of an system engineer.

Lab-Session:

1 & 2

(2), 3 & (4)

(4) & 5



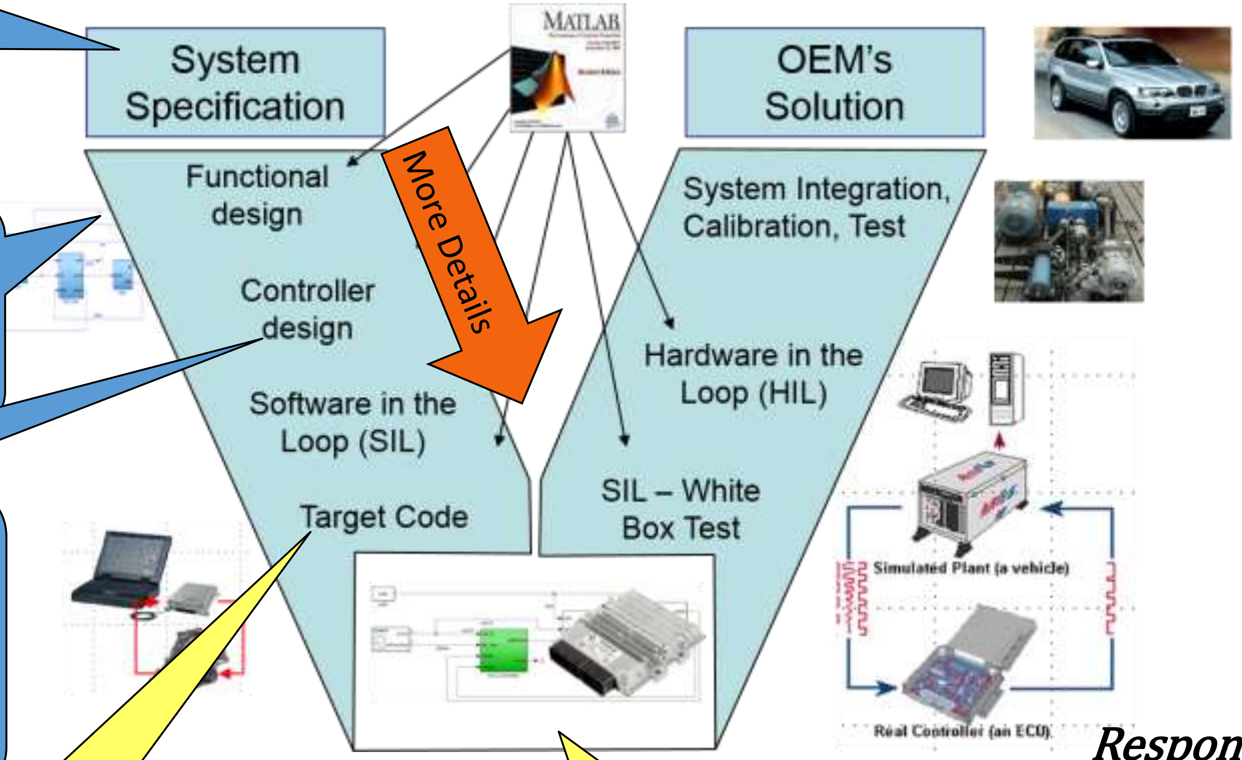
Co-funded by the Erasmus+ Programme of the European Union

Modelbased Design and V-Model

Feasibility Simulink Model of plant & controller + Word-Docu

Software Specification
Simple model of Controller, Requirements definition

Model In the Loop
Model of Simulink Software (ideal) against simulated improved plant model.



Software Design
Module split,
→ re-usable, testable

Programming
Simulink Software → (automatic) C-Code generation.

Responsibility

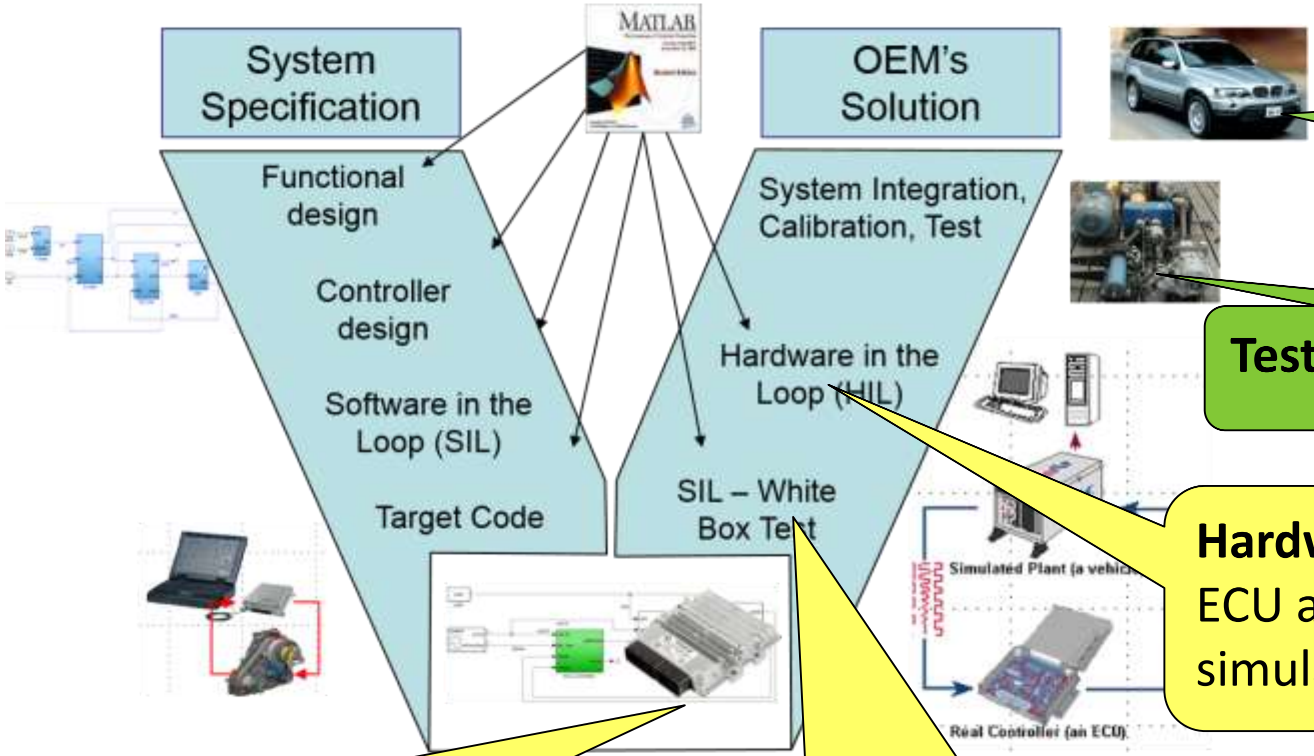
System Engineer

Software

Test Engineer

Co-funded by the Erasmus+ Programme of the European Union

V-Model and Test



Tests in car

Tests on test bench

Hardware-In-the-Loop
ECU against electrically simulated world

Software-Module-Test,
Every single Simulink-WS-model against simple test sequences defined in MATLAB

Software-In-The-Loop
Simulink-WS against simulated Plant

Responsibility

- System Engineer
- Software
- Test Engineer

Co-funded by the Erasmus+ Programme of the European Union

Content 2

- Introduction Lessons
- 5 Lab-Sessions in groups of max. 20 students
    - Identification of plant model parameters (mechanical system + E-motor) on real hardware using vector/CANape.
    - Controller model development and set up using Model In the Loop simulation (MIL).
    - Feasibility and system requirements definition.
    - Simulation of imperfections in hard- and software and 2nd controller set up.
      → Software In the Loop simulation (SIL) using SIMULINK-Model for Software (fixed point arithmetic).
    - Coding ‘C’, flashing, testing.

Co-funded by the Erasmus+ Programme of the European Union

Start with real test bench

Important: It must be safe against missuses!

The sun rises, when the students see the system moving the first time. ☺

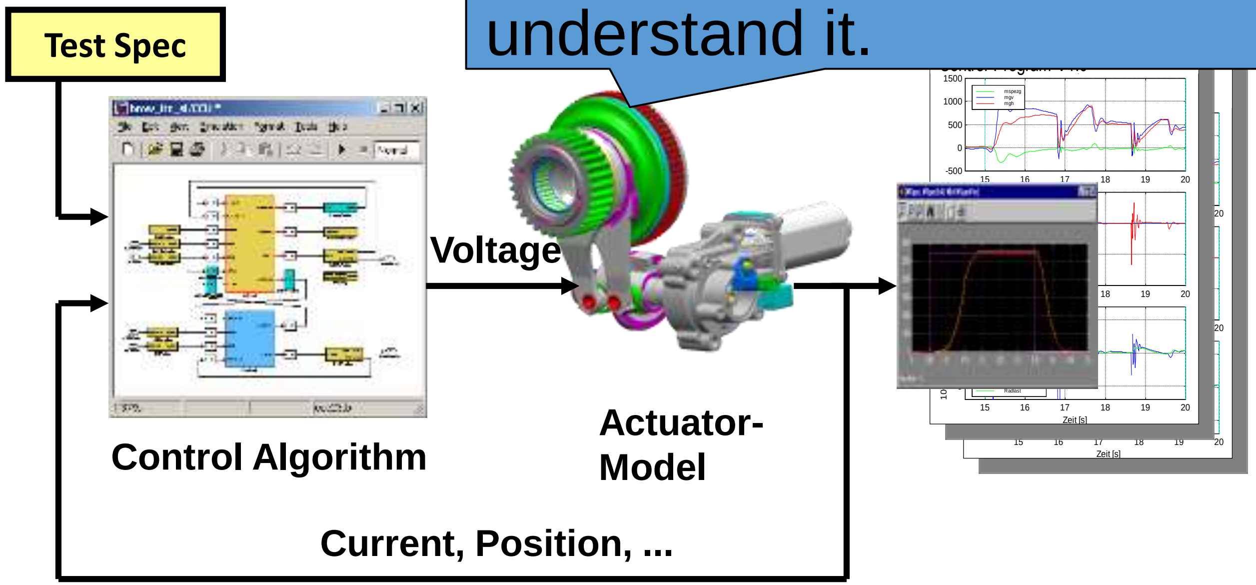
- Generates interest for this topic.
- The students trust, that that functionality is true – not a theoretical one.
- They learn how to applicate a mechatronic system.
- Easy watching of different signals (CCP).
- Comparison between real signals and representation in ECU.
- Base to understand a virtual prototype.

Co-funded by the Erasmus+ Programme of the European Union

Model In The Loop

The virtual prototype on PC we can play with to understan

Students get Simulink-Plant-Model. They should understand it.



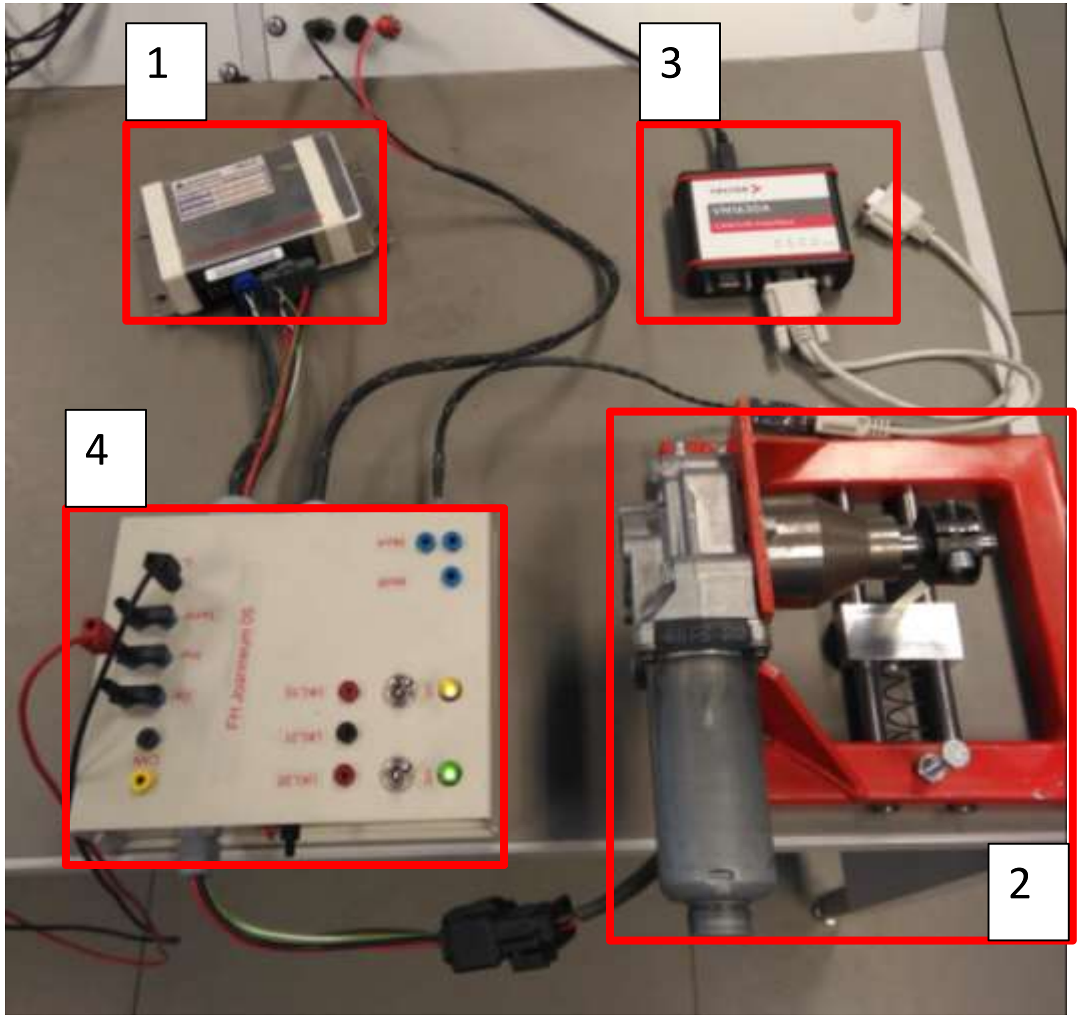
- Visualizes signals needed
- Cause effect analysis
- Easy to check plant parameters influence.
- Playground to develop and understand control algorithms step by step.
- Feasibility study
- Derive requirements for the system and it‘s components.

Any plant model can not
be destroyed by missuses ☺

Quelle: Reisinger, Rühringer, Mathis: Modellgestützte Mechatronik-Systementwicklung für Allradanwendungen; TECHME, Sindelfingen Sept. 2007

Co-funded by the Erasmus+ Programme of the European Union

Torque Control – Modell in the Loop Hardware Overview



1 ECU-Controller

2 Environment (plant model)

3 CAN to USB Interface

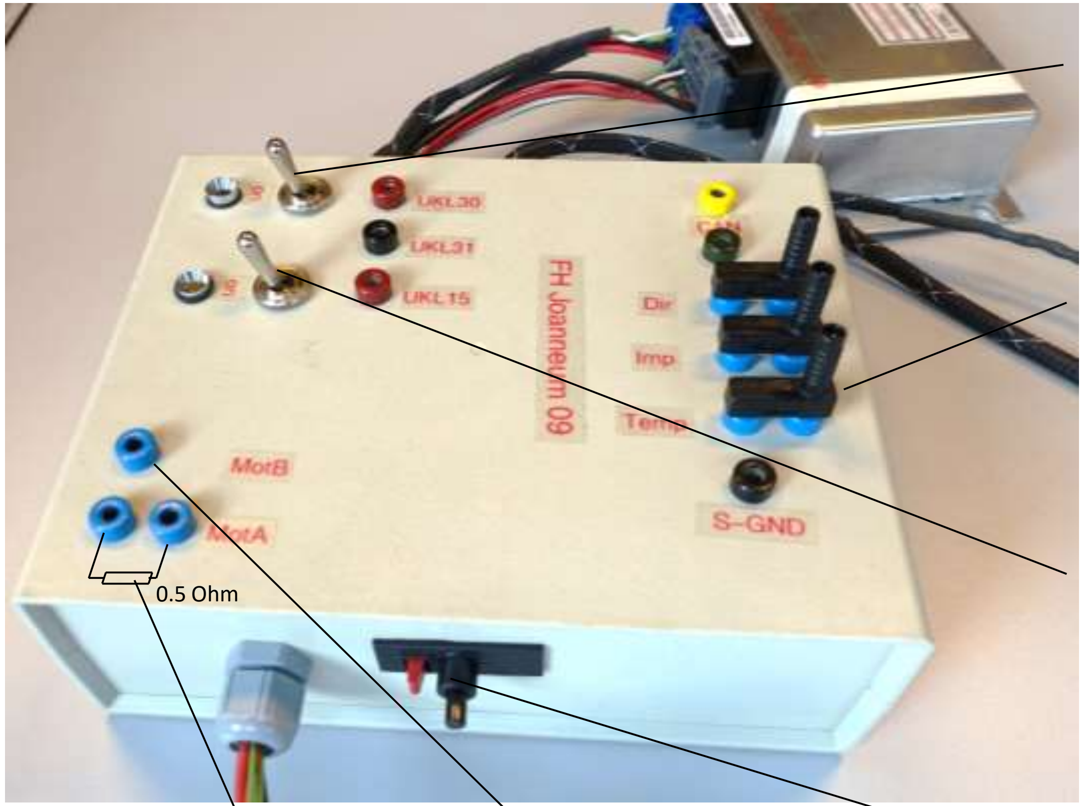
Vector VN 1630

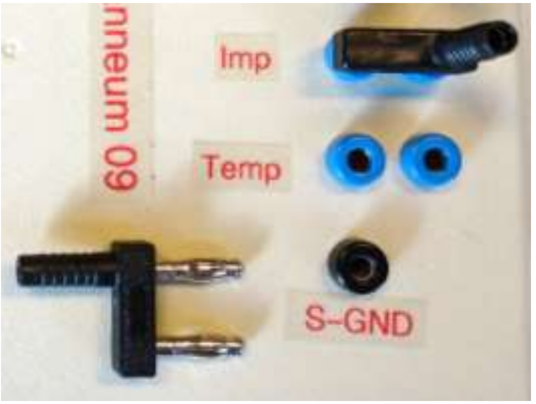
4 Breakbox

Co-funded by the Erasmus+ Programme of the European Union

Breakbox

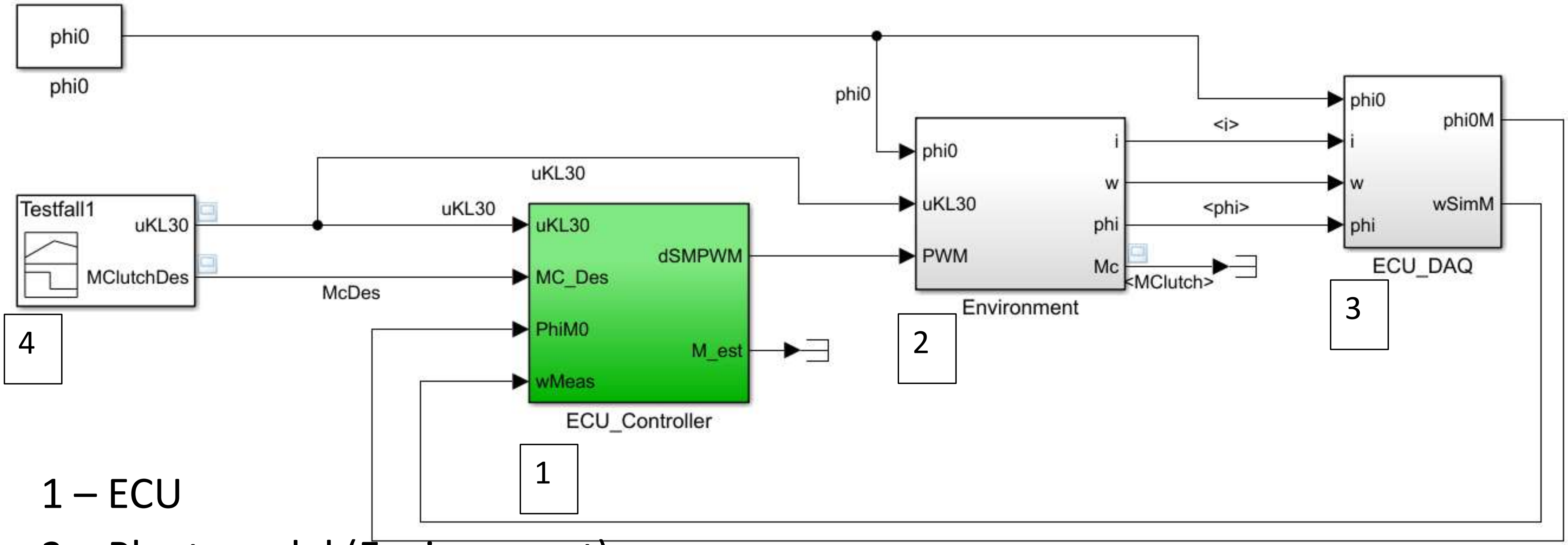
- Replacement for wiring harness
- Connection between motor, sensors, ECU, External CAN-Interface and power supply.
- Switches for car‘s state
- Connectors to measure and test signal failure.





Co-funded by the Erasmus+ Programme of the European Union

Torque Control – Modell in the Loop
Modell Overview



1 – ECU

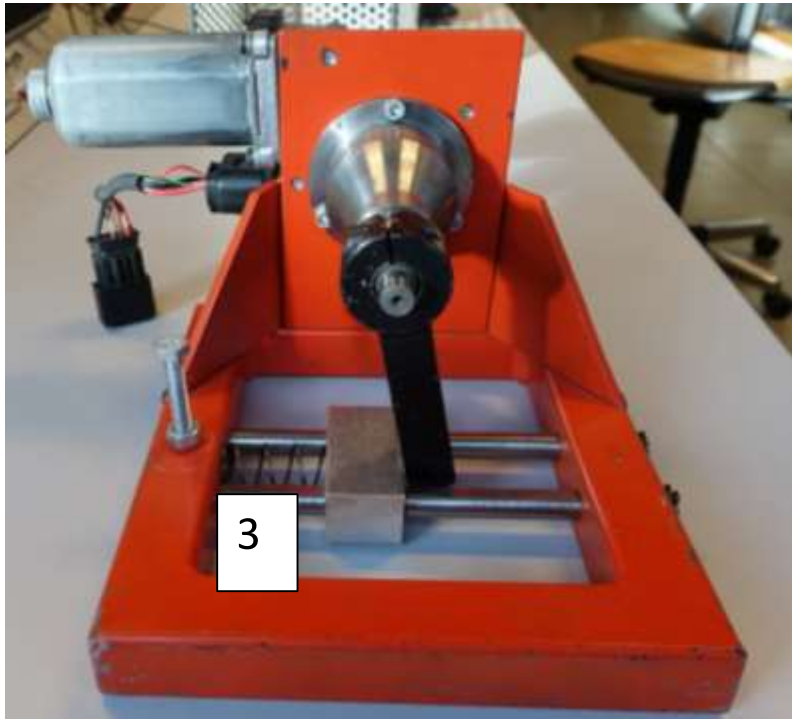
2 – Plant-model (Environment)

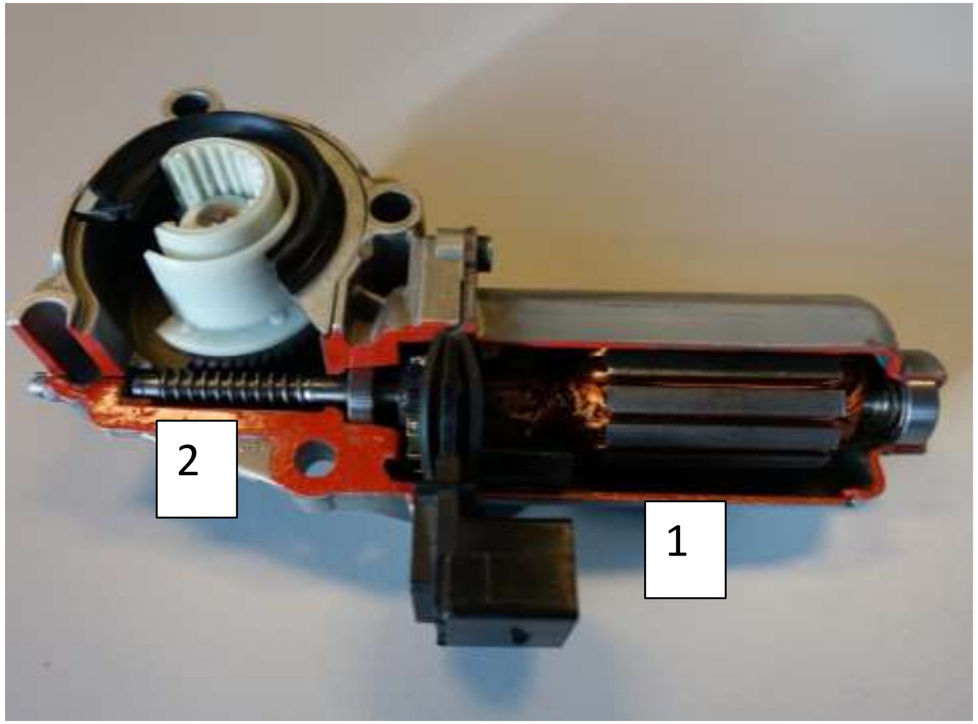
3 – Data acquisition

4 – Stimulus (Simulink: `Signal Generator`)

Co-funded by the Erasmus+ Programme of the European Union

Torque Control - Modell in the Loop Environment → Plant Model





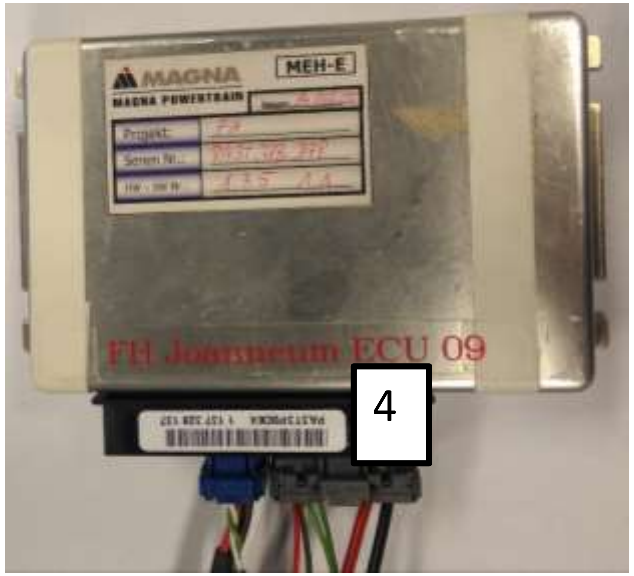
1 – DC-Motor

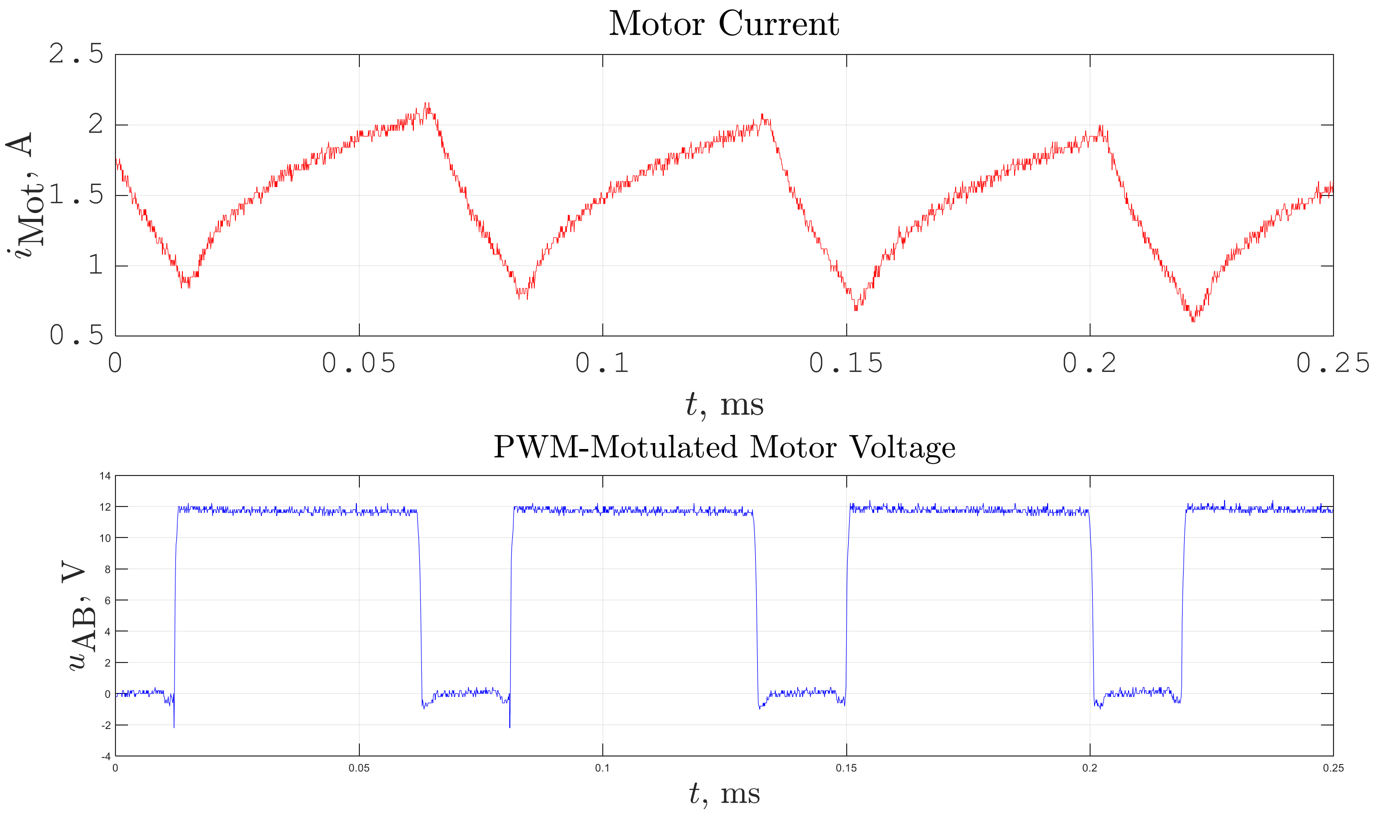
2 – Worm Gear → gear ratio is 56

3 – Spring → simulate the feedback from the clutch via the worm gear

Co-funded by the Erasmus+ Programme of the European Union

Torque Control - Modell in the Loop
Plant Model





4 – The H-Bridge is integrated at the ECU. The output is a PWM-modulated voltage. The mean-value of the voltage is proportional to the motor speed.

Co-funded by the Erasmus+ Programme of the European Union

Torque Control - Modell in the Loop
Plant Model

4 – H-Bridge → Power electronic (included at the ECU)

Input: PWM-Signal from controller. In our model PWM is a numeric values between -1 and +1

Output: PWM-modulated voltage for DC-Motor power supply.

The mean-value influences the motor speed.

Simplification for the model: $u_{\mathrm{AB}} = u_{\mathrm{Kl30}} \cdot \mathrm{PWM}$

u_{AB} DC-Motor input voltage

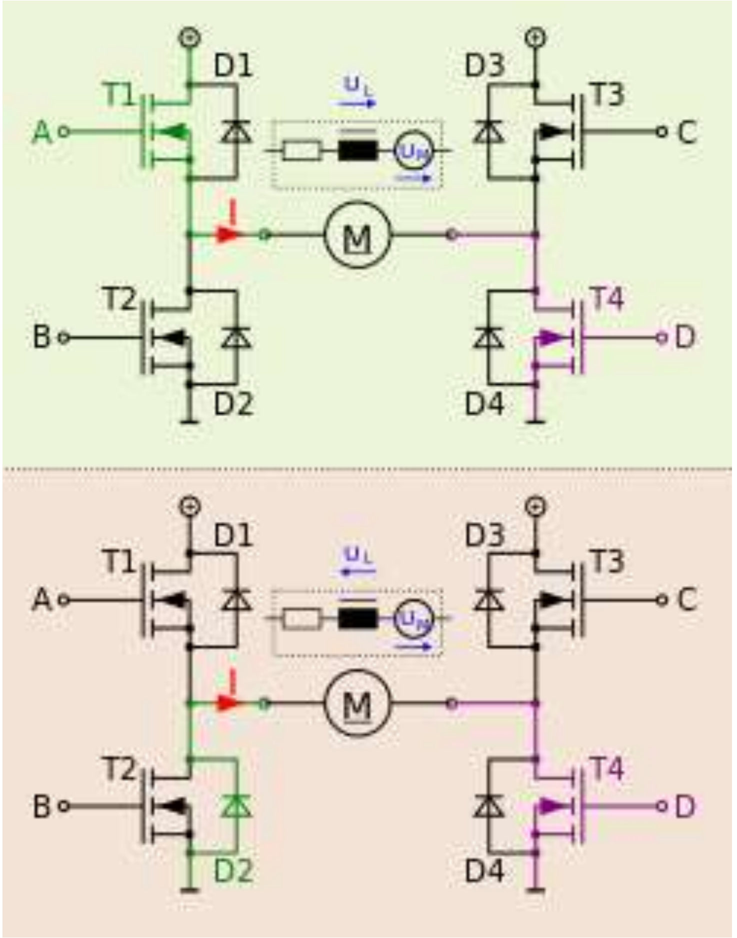
u_{Kl30} Supply voltage

Advantage because of the simplification: higher performance from the model.

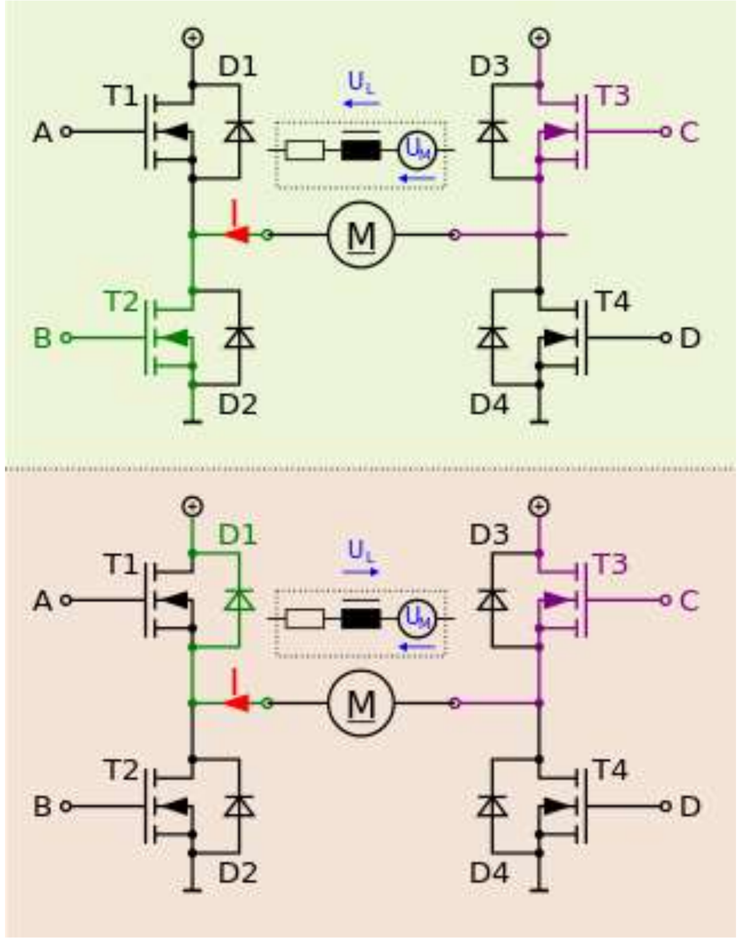
Co-funded by the Erasmus+ Programme of the European Union

Torque Control - Modell in the Loop
Plant Model

Quadrant 1 -
accelerate forward



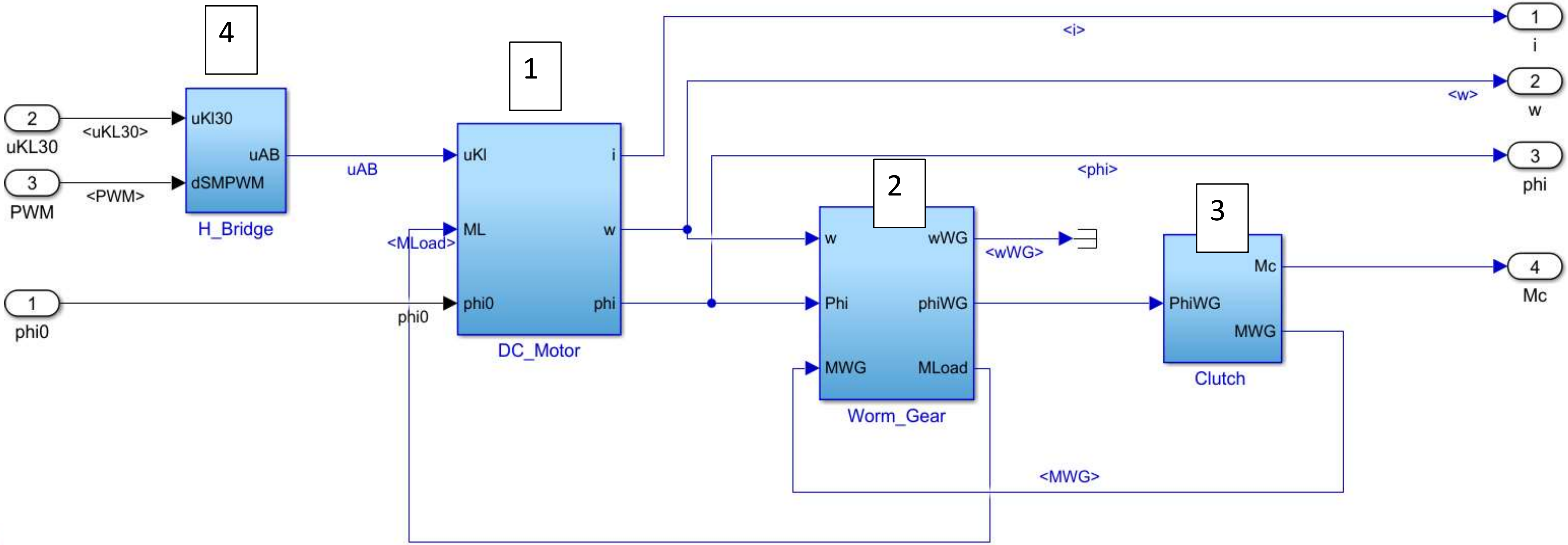
Quadrant 3 -
accelerate backward



https://de.wikipedia.org/wiki/Vierquadrantensteller

Co-funded by the
Erasmus+ Programme
of the European Union

Torque Control - Modell in the Loop
Plant Model

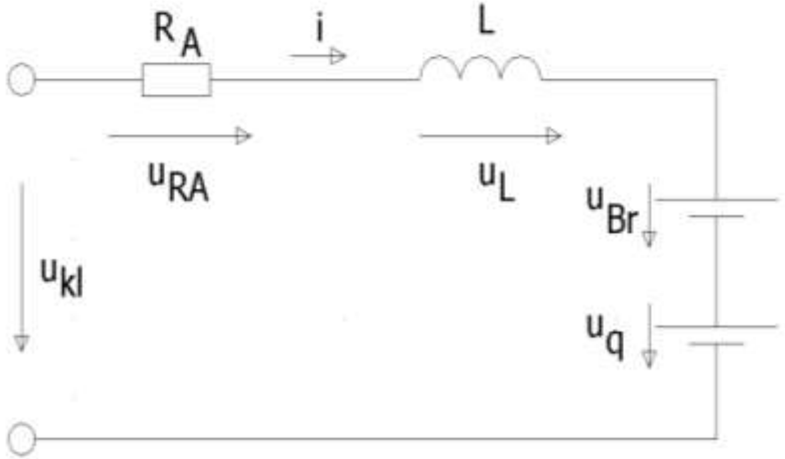


Co-funded by the
Erasmus+ Programme
of the European Union

How to model a device with Simulink?

Example: Permanent-magnet DC motor

- Describe the motor mathematically
  - 1.) electrical system



Kirchhoffs's law: $u_{\mathrm{Kl}} = u_{\mathrm{RA}} + u_{\mathrm{L}} + u_{\mathrm{Br}} + u_{\mathrm{q}}$ (1)

Voltage drops: $u_{\mathrm{RA}} = i \cdot R_{\mathrm{A}}$ (2)

$$u_{\mathrm{L}} = L\frac{\mathrm{d}i}{\mathrm{d}t} \quad (3)$$

$$u_{\mathrm{q}} = k_{\mathrm{T}} \cdot \omega \quad (4)$$

$u_{\mathrm{Br}} = f(i)$ → Lookup Table

(2), (3) and (4) → (1) $\frac{\mathrm{d}i}{\mathrm{d}t} = \frac{1}{L}(u_{Kl} - i \cdot R_A - u_{Br} - k_T \cdot \omega)$ (6)

Co-funded by the Erasmus+ Programme of the European Union

How to model a device with Simulink?

Example: Permanent-magnet DC motor

- Describe the motor mathematically

  - 2.) coupling between electrical and mechanical system

Torque is proportional to the current

$$M_{\mathrm{el}} = k_{\mathrm{T}} \cdot i \qquad (6)$$

  - 3.) mechanical system

https://de.wikipedia.org/wiki/Anker_(Elektrotechnik)

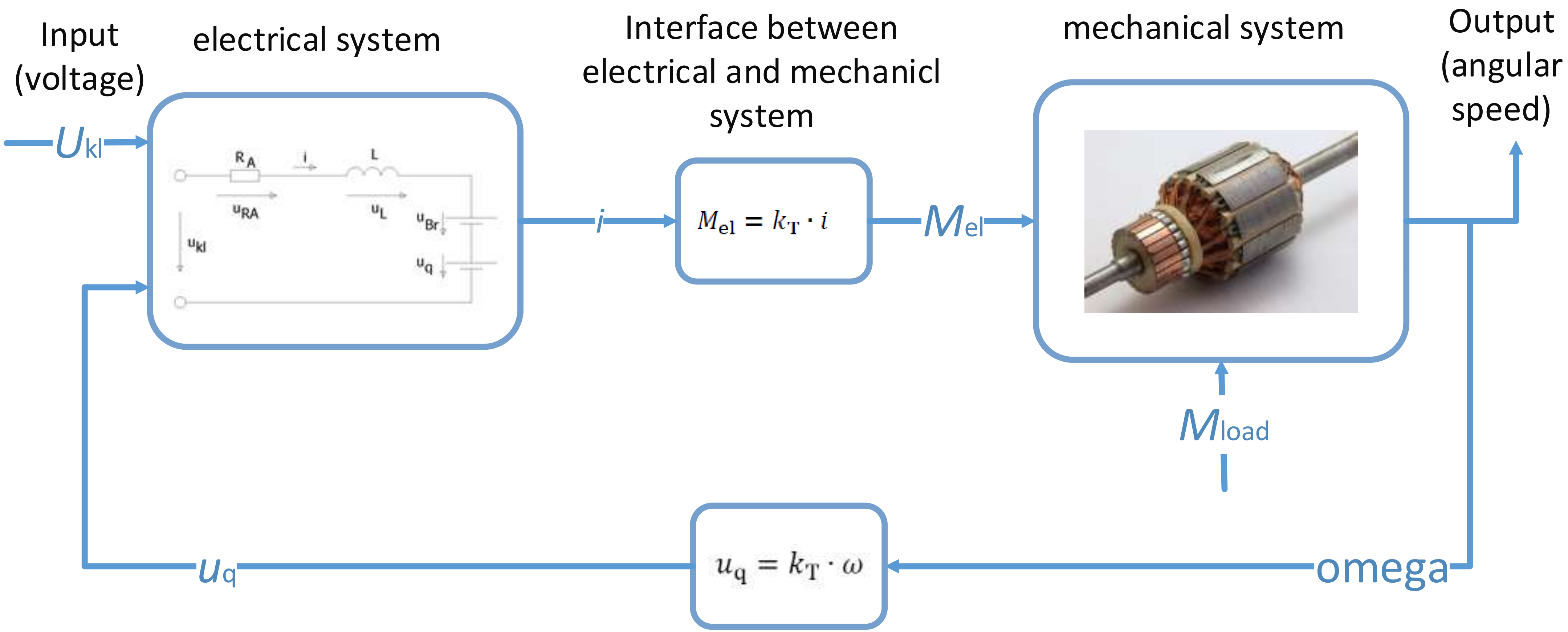
The rotor is a rotatable mounted inertial mass – principle of angular momentum

$$J \cdot \frac{\mathrm{d}\omega}{\mathrm{d}t} = M_{\mathrm{el}} - M_{\mathrm{load}} - M_{\mathrm{fr}} \cdot \mathrm{sign}(\omega) \qquad (7)$$

Co-funded by the Erasmus+ Programme of the European Union

Model of a permanent-magnet DC motor

Scheme of model

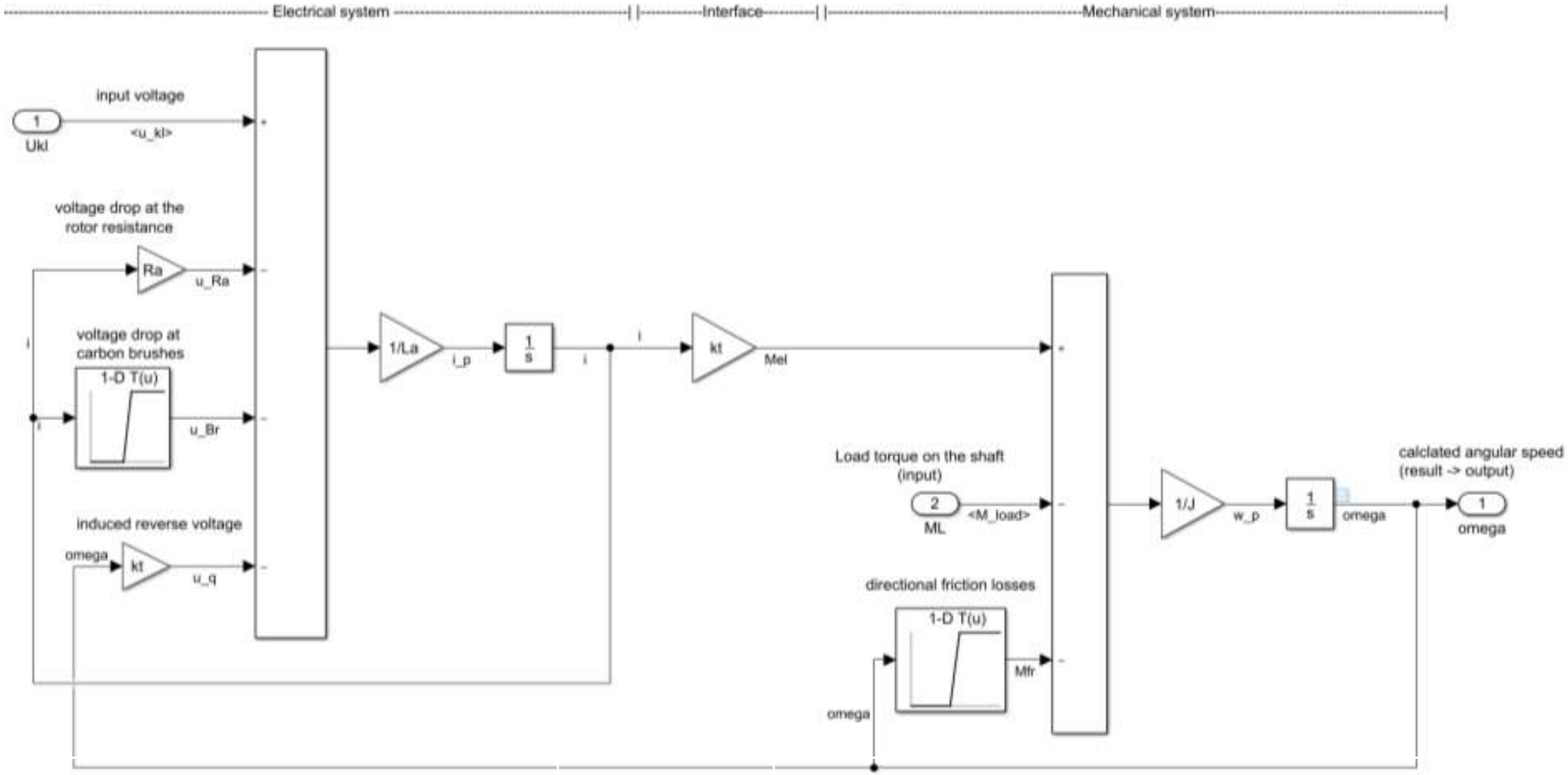


$$\frac{\mathrm{d}i}{\mathrm{d}t} = \frac{1}{L}\left(u_{Kl} - i \cdot R_A - u_{Br} - k_T \cdot \omega\right)$$

$$\frac{\mathrm{d}\omega}{\mathrm{d}t} = \frac{1}{J}\left(M_{el} - M_{load} - M_{fr} \cdot \mathrm{sign}(\omega)\right)$$

Model of a permanent-magnet DC motor

Simulink model



$$\frac{\mathrm{d}i}{\mathrm{d}t} = \frac{1}{L}(u_{\mathrm{Kl}} - i \cdot R_{\mathrm{A}} - u_{\mathrm{Br}} - k_{\mathrm{T}} \cdot \omega)$$

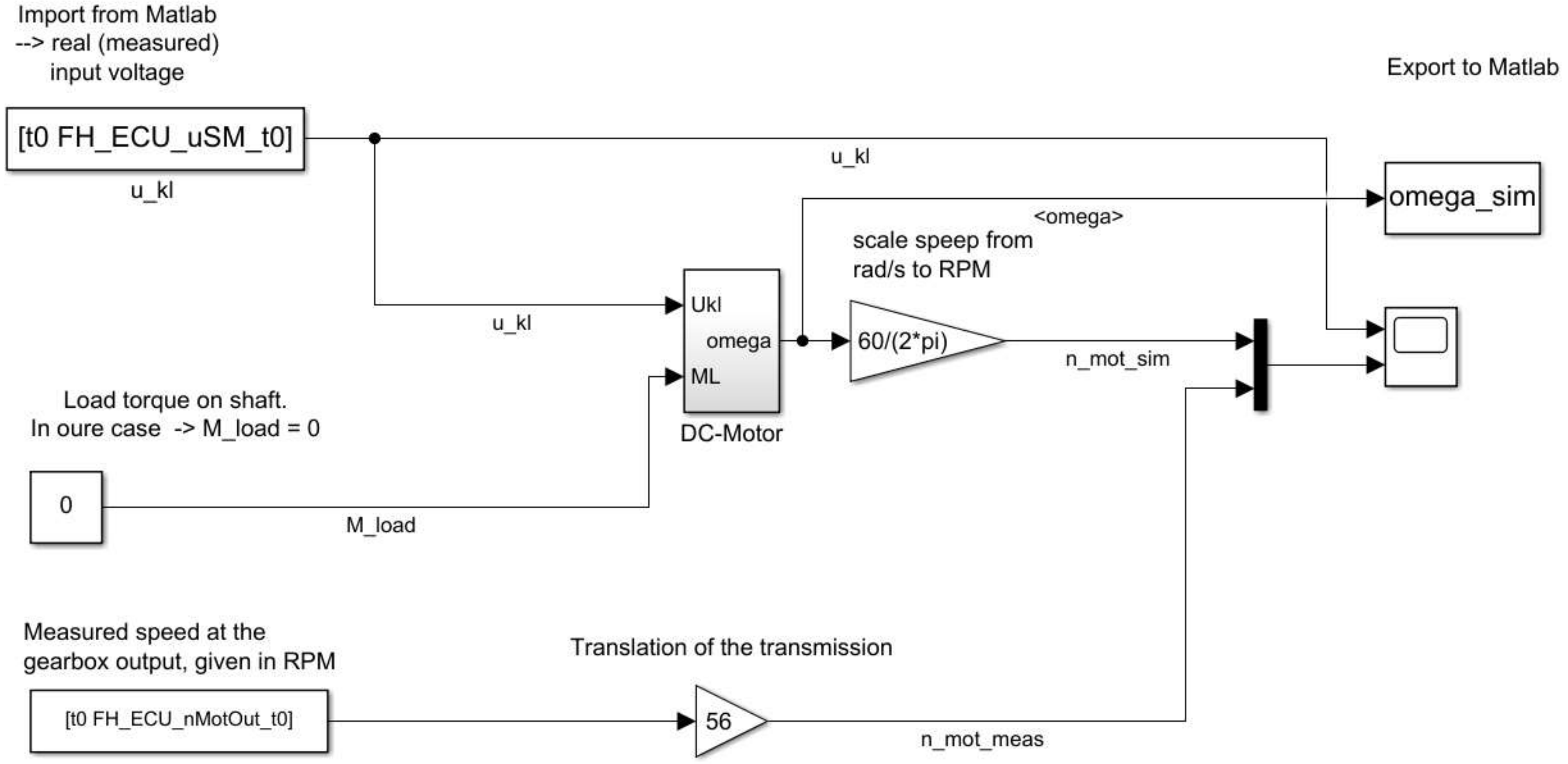
$$\frac{\mathrm{d}\omega}{\mathrm{d}t} = \frac{1}{J}\left(M_{\mathrm{el}} - M_{\mathrm{load}} - M_{\mathrm{fr}} \cdot \mathrm{sign}(\omega)\right)$$

Co-funded by the Erasmus+ Programme of the European Union

Model of a permanent-magnet DC motor

Validate the model → compare measured with simulated values

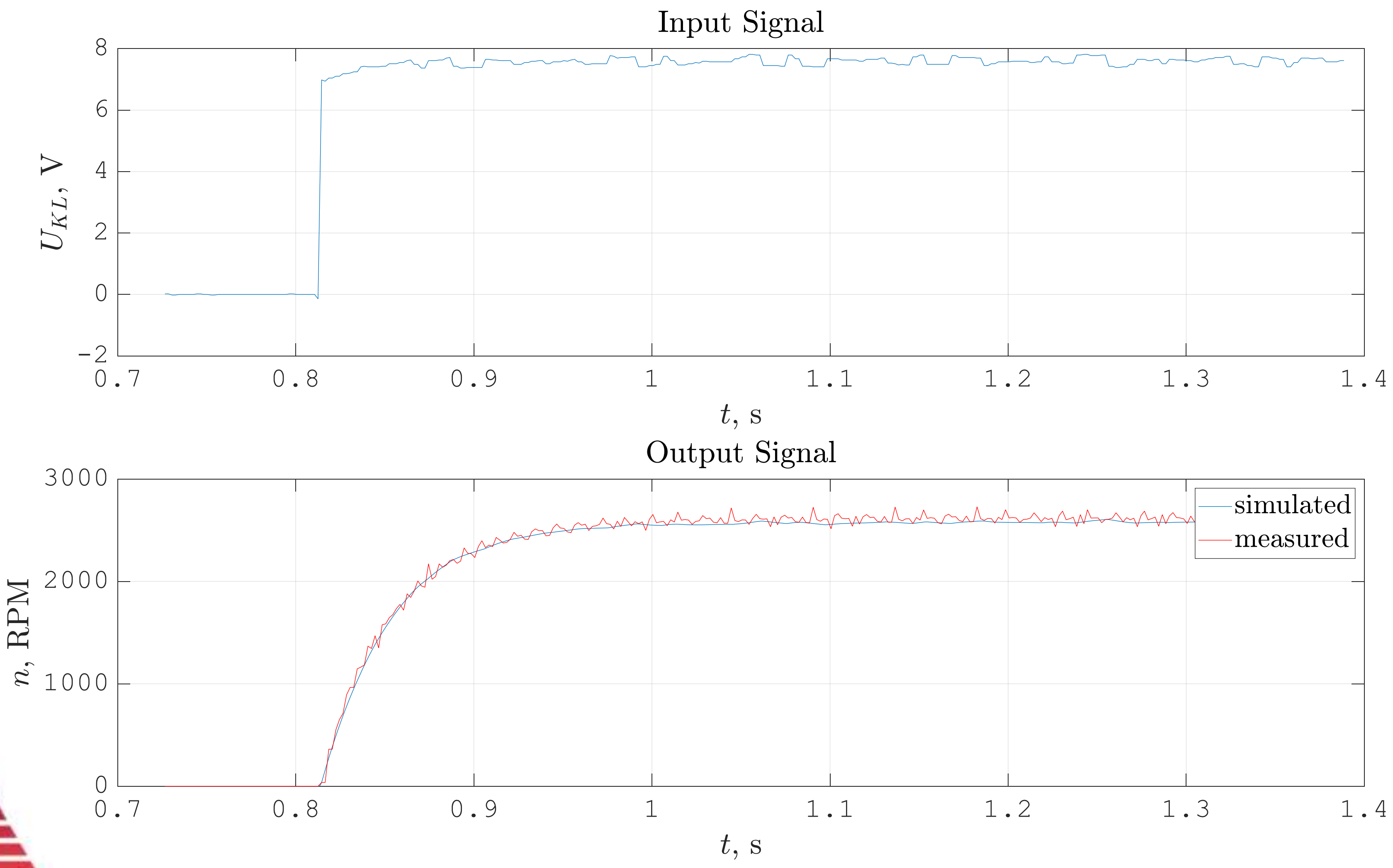
- Use the dc-motor model in a subsystem
- Use measured data for the input signal (`u_kl`)
- Start simulation
- Compare measured output signal `n_mot_meas` with simulation result `n_mot_sim`



Co-funded by the
Erasmus+ Programme
of the European Union

Permanent-magnet DC motor

Parameter identification



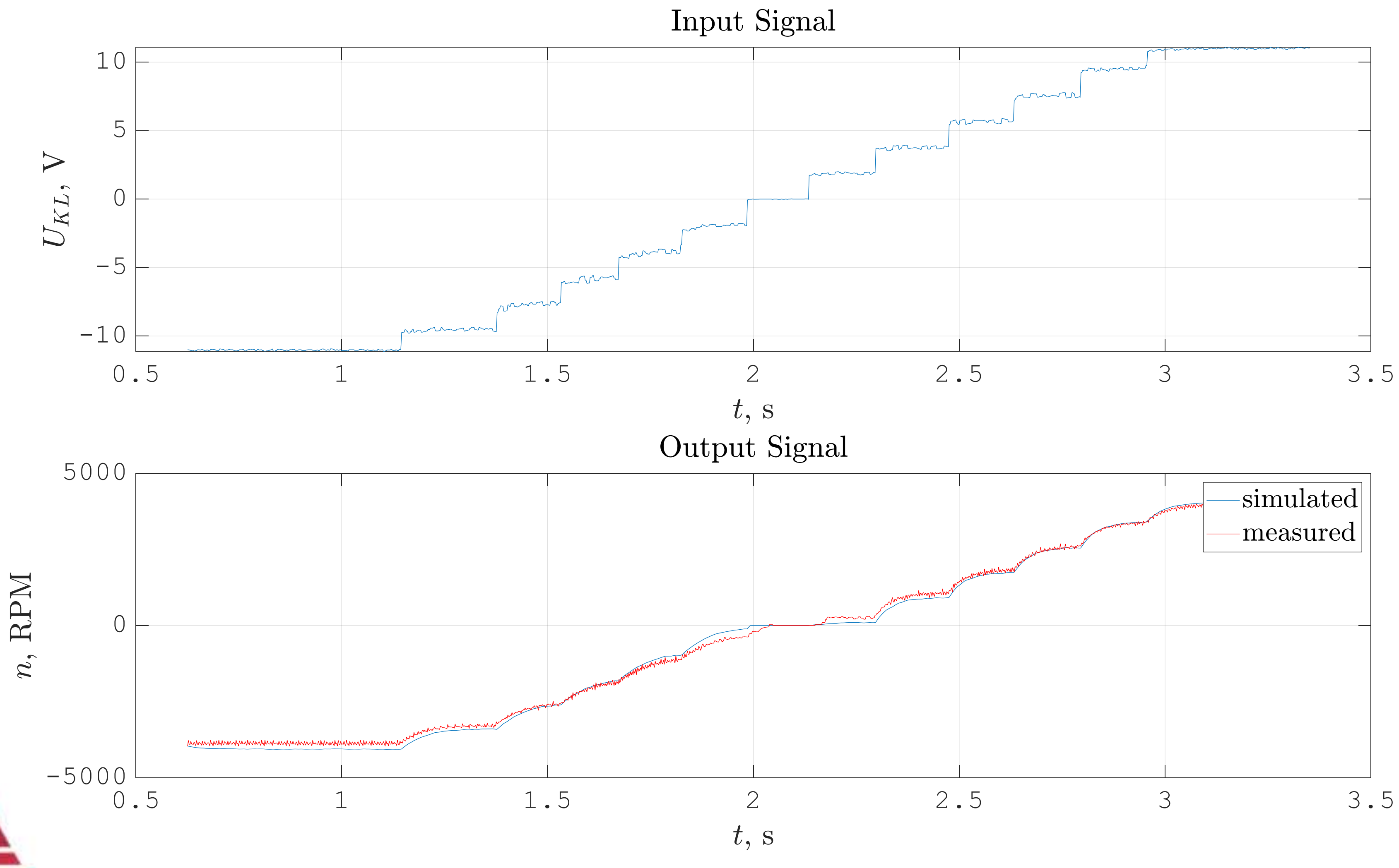
The measured voltage is the stimulus for our model.

Parameter are known from datasheets. Fine adjustment of parameters k_T, R_A for static behaviour, L and J for the dynamic performance.

Co-funded by the Erasmus+ Programme of the European Union

Permanent-magnet DC motor

Parameter validation



The measured voltage is the stimulus for our model.

Measured and simulated results are correlating. Our motor model works and can be used for further tests!

Co-funded by the Erasmus+ Programme of the European Union

Clutch Behaviour

- Clutch Torque M_c ~ Axial Force F_c

$$M_C \cong F_C \cdot \mu \cdot z \cdot r_m$$

- Deformation of
  - Paper-Cover
  - Piston (Steel)
  - Snap Ring ,…
  - Resultant characteristics

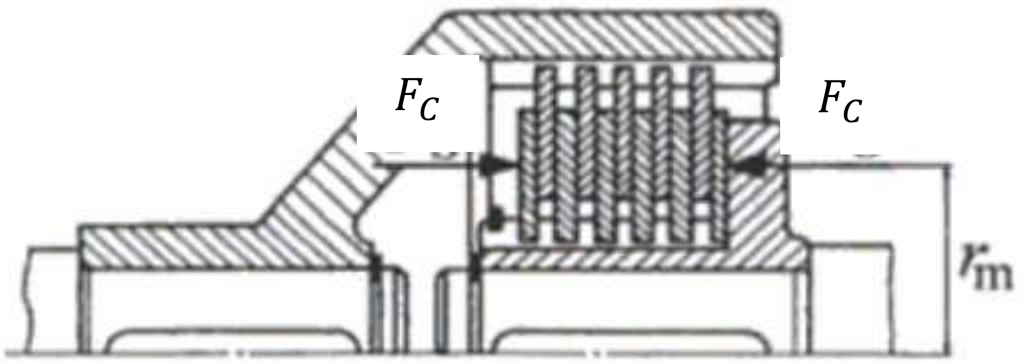
$$F_c = f(s_C)$$

[…]

- Including a gear ratio $s_C/\boldsymbol{\varphi}_{Mot}$ and it‘s efficiency

$$M_c = f(\varphi_{Mot})$$

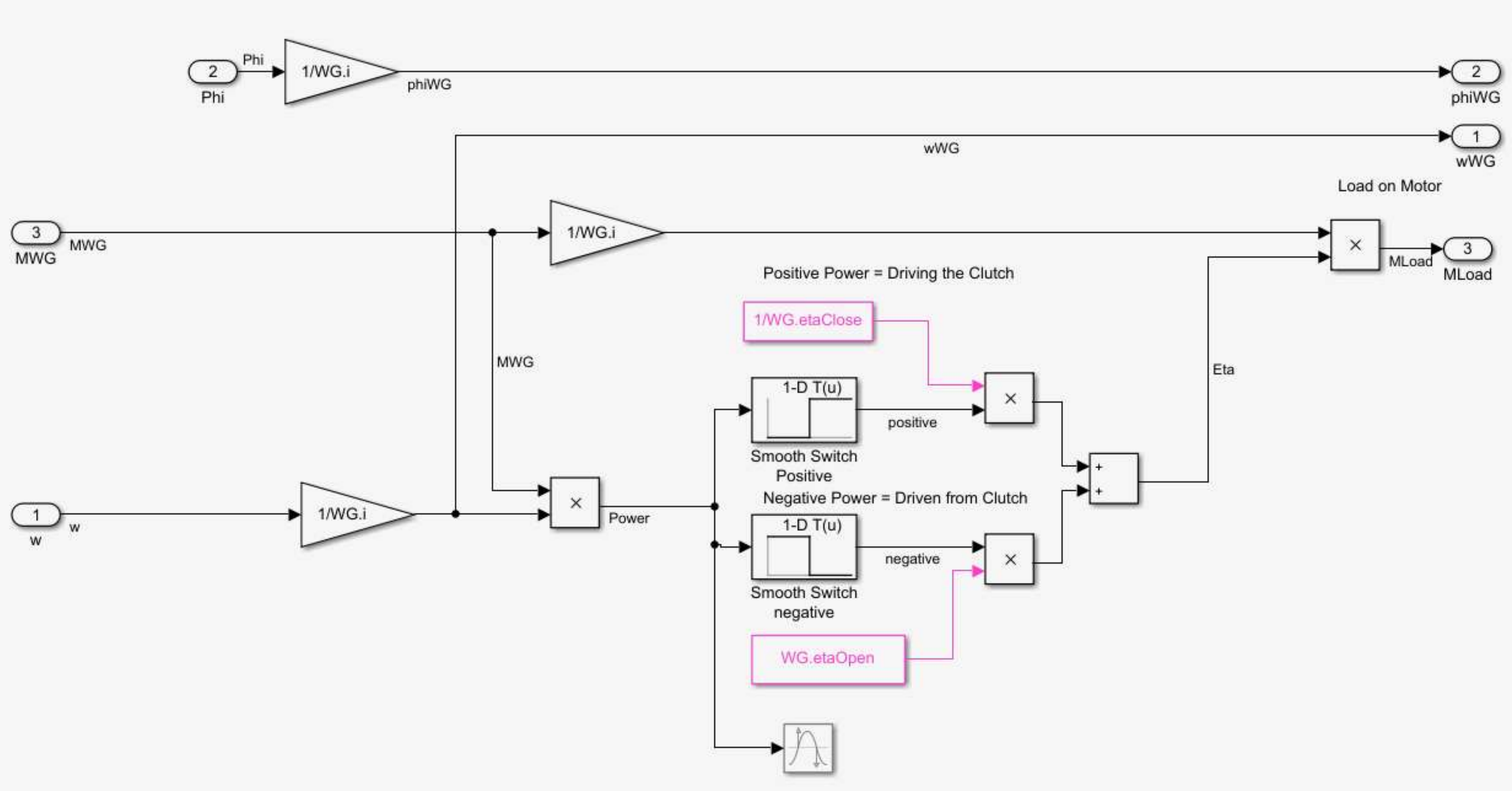
Explanation and derivation of equation on physics level



Künne B.: Einführung in die Maschinenelemente, Teubner

Model of the clutch

Simulink model



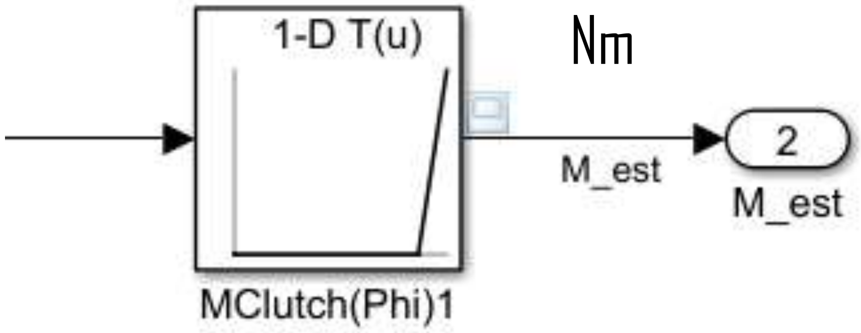
Co-funded by the
Erasmus+ Programme
of the European Union

Torque Control - Programming the ECU

- Our goal is to control torque!
- <u>Problem:</u> We do not measure the torque, because there is no economic measurement device available. But we need a desired value!
- We solve the problem with the following methodology:
  - Measure the torque according to a position (angle in rad) at a test bench.
  - Describe the non-linear connection between torque an position with a characteristic line (Simulink → Lookup Table).
  - Now we can implement a position controller (→ position is measure of torque) .

Input: angle φ in rad

Output: estimated torque in Nm



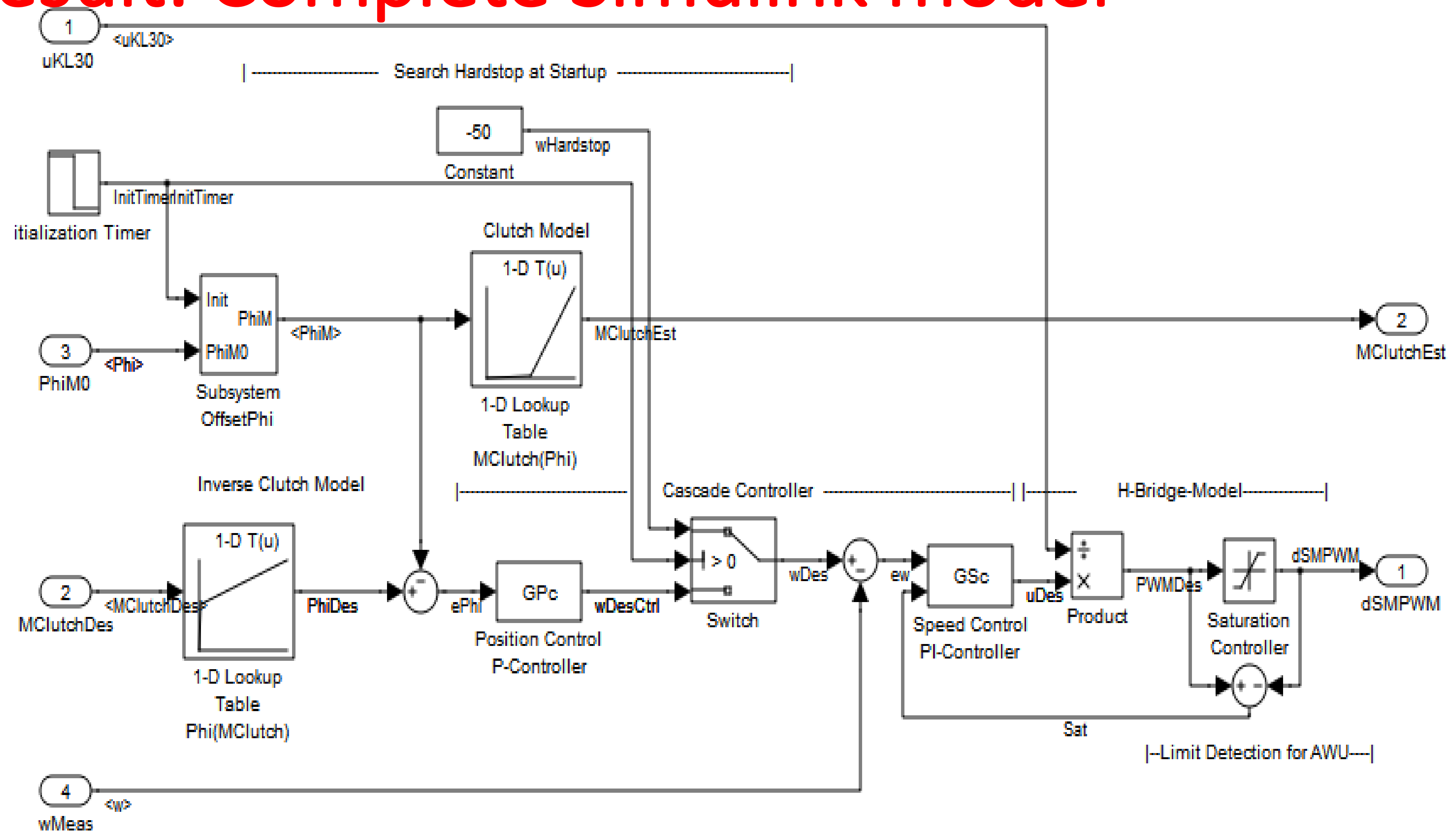
Torque Control - Programming the ECU Controller implementation

- Speed controller with an overlaid position controller
- Position → P-controller
  K_P = …? (40 s)
- Speed → PI-controller with anti wind-up mechanism
  K_P = … ? (0.17 Vs/rad)
  T_n = … ? (0.017 s)

Setting the parameter according to the Ziegler-Nichols method

Co-funded by the Erasmus+ Programme of the European Union

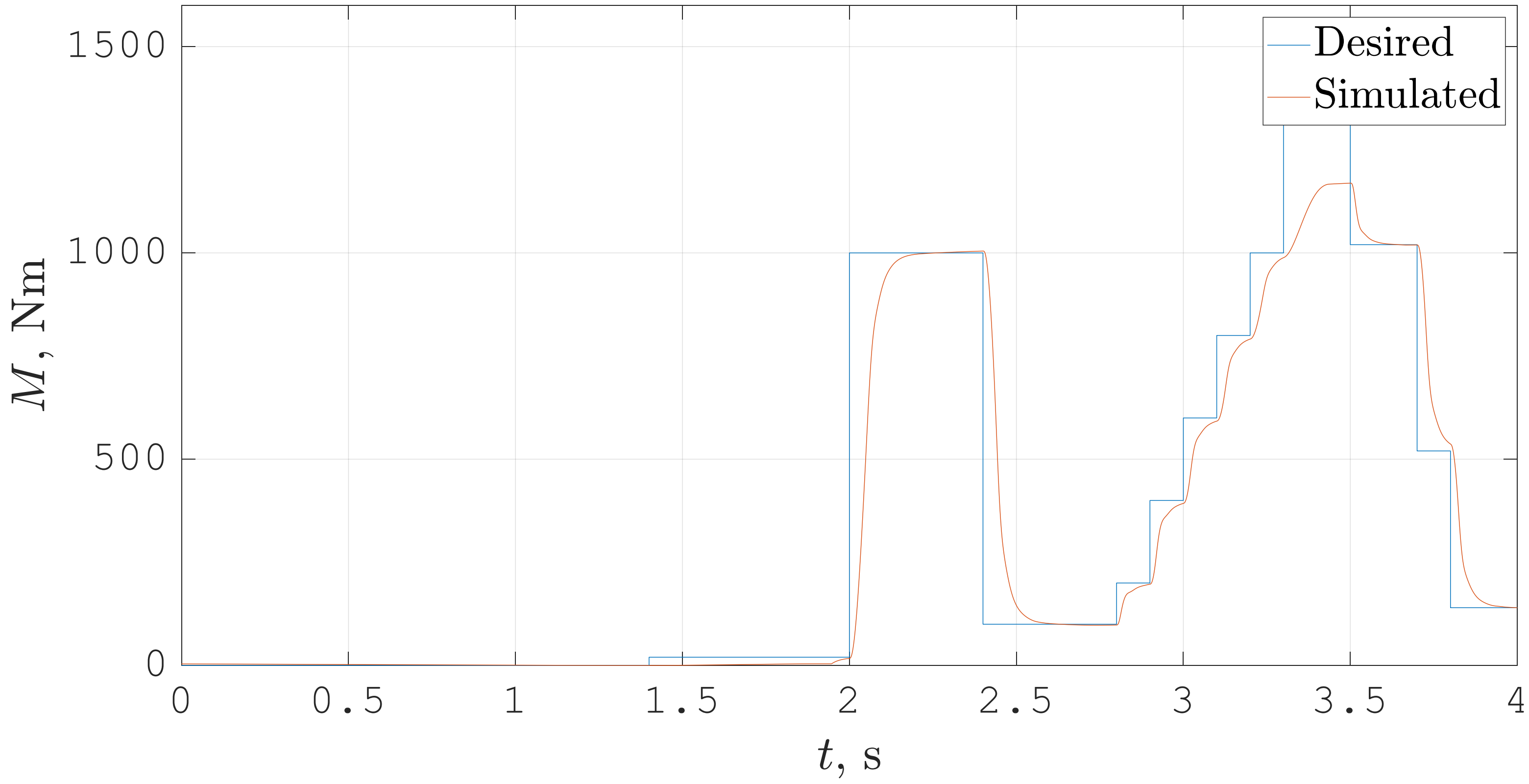
Torque Control – Modell in the Loop
Result: Complete Simulink model



Co-funded by the
Erasmus+ Programme
of the European Union

Torque Control-MIL Result

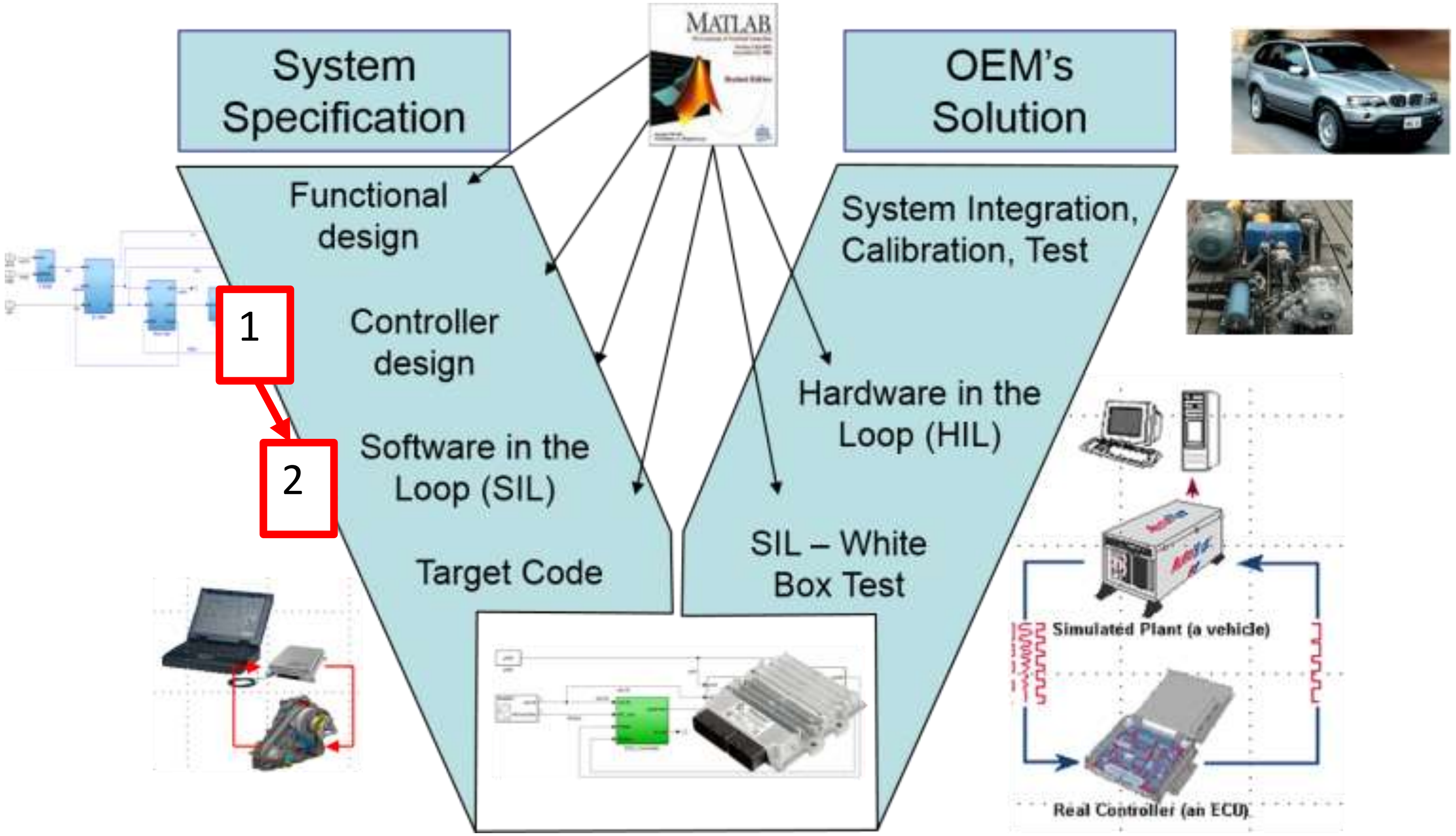
- The validation of the model shows, that the controller works.
- At the moment, we have simulated a „perfect“ environment.
- We must further consider the following technical details:
  - Data Acquisition (DAQ)
  - ECU cycle time
  - Fixed point arithmetic



Co-funded by the Erasmus+ Programme of the European Union

Torque Control – From MIL to SIL

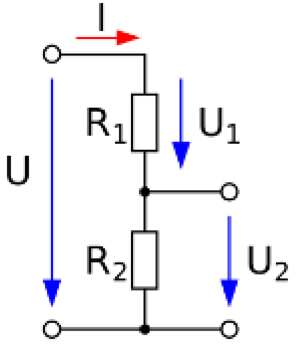
- At the moment, we have simulated a „perfect“ environment (1).
- We must further consider the following technical details (2):
  - Data Acquisition (DAQ)
  - ECU cycle time
  - Fixed point arithmetic

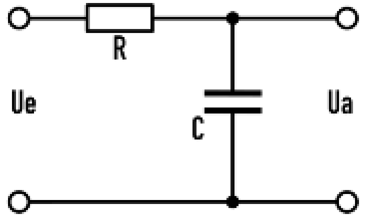


Co-funded by the Erasmus+ Programme of the European Union

How do we get the signal into the µC?

- Adopt the amplitude (voltage divider)
- Supress to high frequencies
  (RC-device for Anti Aliasing)
  - Nyquist-Shannon-Theorem: $f < \frac{f_{sample}}{2}$
  - Otherwise: Aliasing
    = low non low frequencies appear in digital signal!
- Analog-Digital-Converter
  - Assigns at each sample time step a digital number to the signal. → Discrete values + discrete time steps.





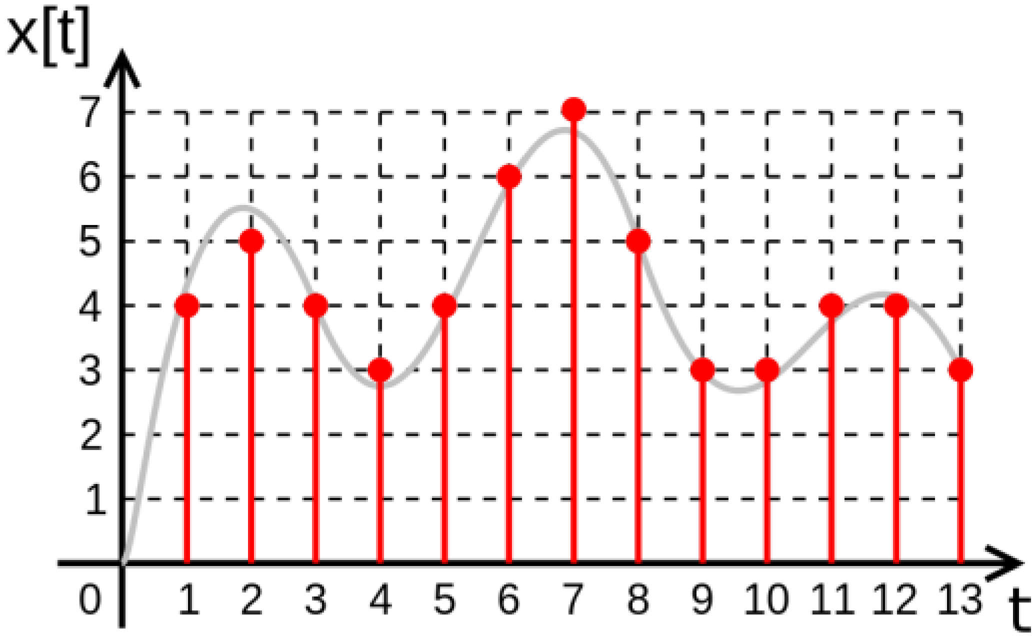
www.wikipedia.org

Co-funded by the Erasmus+ Programme of the European Union

Analog-Digital-Convertion (Sampling)

- Discrete Time → Sample Time
- Discrete Amplitude → Quantizing

- Example:
  2 Bit ADC → 8 steps from 0 to 7
  Samplerate 1s



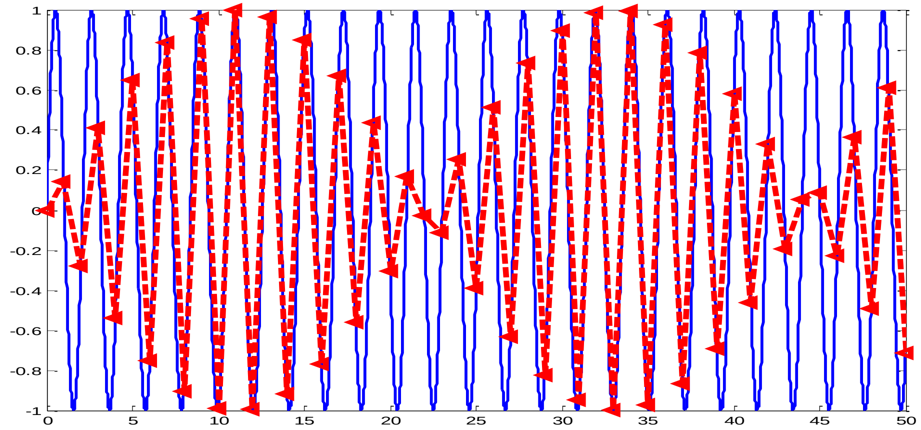
Co-funded by the Erasmus+ Programme of the European Union

Aliasing

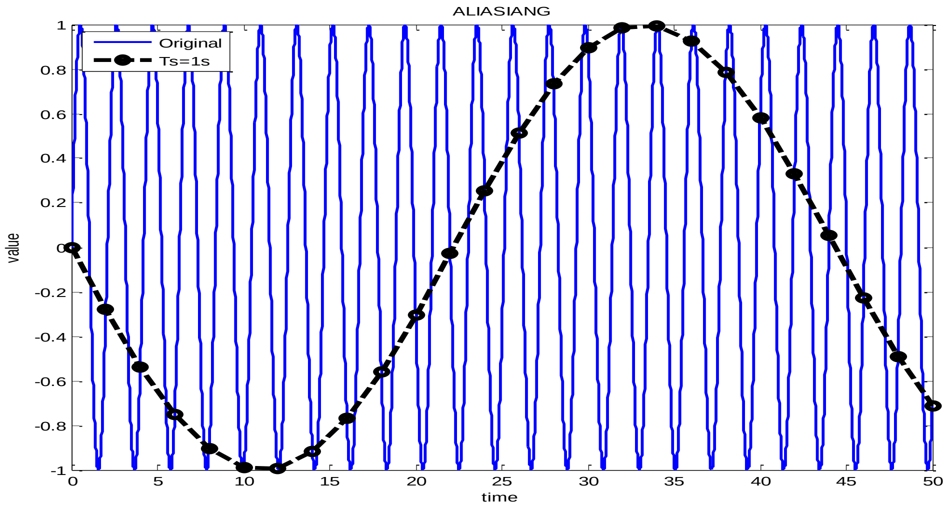
- Nyquist-Shannon-Theorem

$$f_s = \frac{1}{T_s} > 2 \cdot f_{max}$$

- Otherwise aliasing
  - Beat (Schwebung) between sampling frequency and signal
  - Non existing frequencies appear.
- Solution
  - Electrical filter before ADC converts the signal!



$f_s = 2.1\ f$ … no new frequency



$f_s = 1.1\ f$ image frequencies appear

Co-funded by the Erasmus+ Programme of the European Union

CPU number representation

- µP → 16 Bit
- Datatype → Signed Integer

Ukl30 → Maximum value 20 V

Memory map:

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	0	0	0	0	0	0	0	0	0	**1**	0	**1**	0	0	
decimal		0	0	0	0	0	0	0	0	0	0	16	0	4	0	0	**20**

10 Bit unused

5 Bit used

Bit 15 - sign:

Positive → 0

Negative → 1

Co-funded by the Erasmus+ Programme of the European Union

CPU number representation

For a better memory usage → Shift 10 Bits to left (multiplication with 2^{10})

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	
decimal		0	0	0	0	0	0	0	0	0	0	16	0	4	0	0	**20**

$$20 \cdot 2^{10} = 20480$$

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
decimal		16384	0	4096	0	0	0	0	0	0	0	0	0	0	0	0	**20480**

10 free Bits for a higher accuracy

CPU number representation

Least change in value without shifting

Example: Ukl30 = 12 V

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	0	
decimal		0	0	0	0	0	0	0	0	0	0	0	8	4	0	0	**12**

10 Bit unused

5 Bit used

Which value is the next in size?

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	**1**	
decimal		0	0	0	0	0	0	0	0	0	0	0	8	4	0	1	**13**

Least increment:
$1 \cdot 2^0 = 1$

Co-funded by the Erasmus+ Programme of the European Union

CPU number representation

Least change in value with 10-Bit shifting

Example: Ukl30 = 12 V

After shifting:

$$12 \cdot 2^{10} = 12288$$

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
decimal		0	8192	4096	0	0	0	0	0	0	0	0	0	0	0	0	**12288**

Which value is the next in size?

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2^n	sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
binary		0	1	1	0	0	0	0	0	0	0	0	0	0	0	**1**	
decimal		0	8192	4096	0	0	0	0	0	0	0	0	0	0	0	**1**	**12289**

Least increment:

$$1 \cdot 2^{-10} = 0{,}000977 \text{ V}$$

Rescaling: $U_{Kl30} = 12289 \cdot 2^{-10} = 12{,}000977 \text{ V}$

Co-funded by the
Erasmus+ Programme
of the European Union

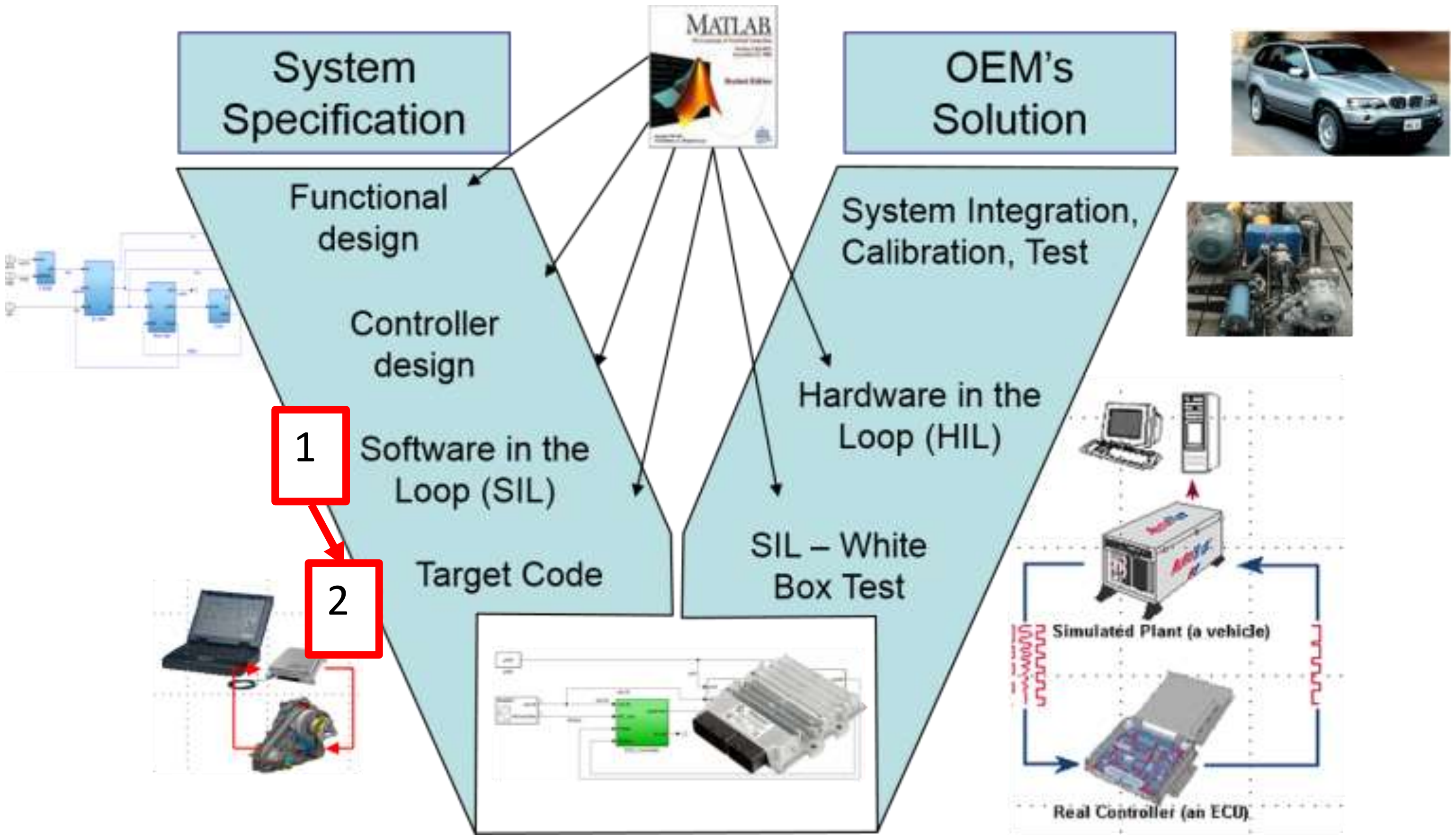
CPU number representation

- Rules for calculation with binary shifted values
  - Addition and subtraction:
    - only possible with same shifted values
  - Multiplication and division, a correction factor must be included:
    - Multiplication → divide with the correction factor
    - Division → Multiply with the correction factor

Co-funded by the
Erasmus+ Programme
of the European Union

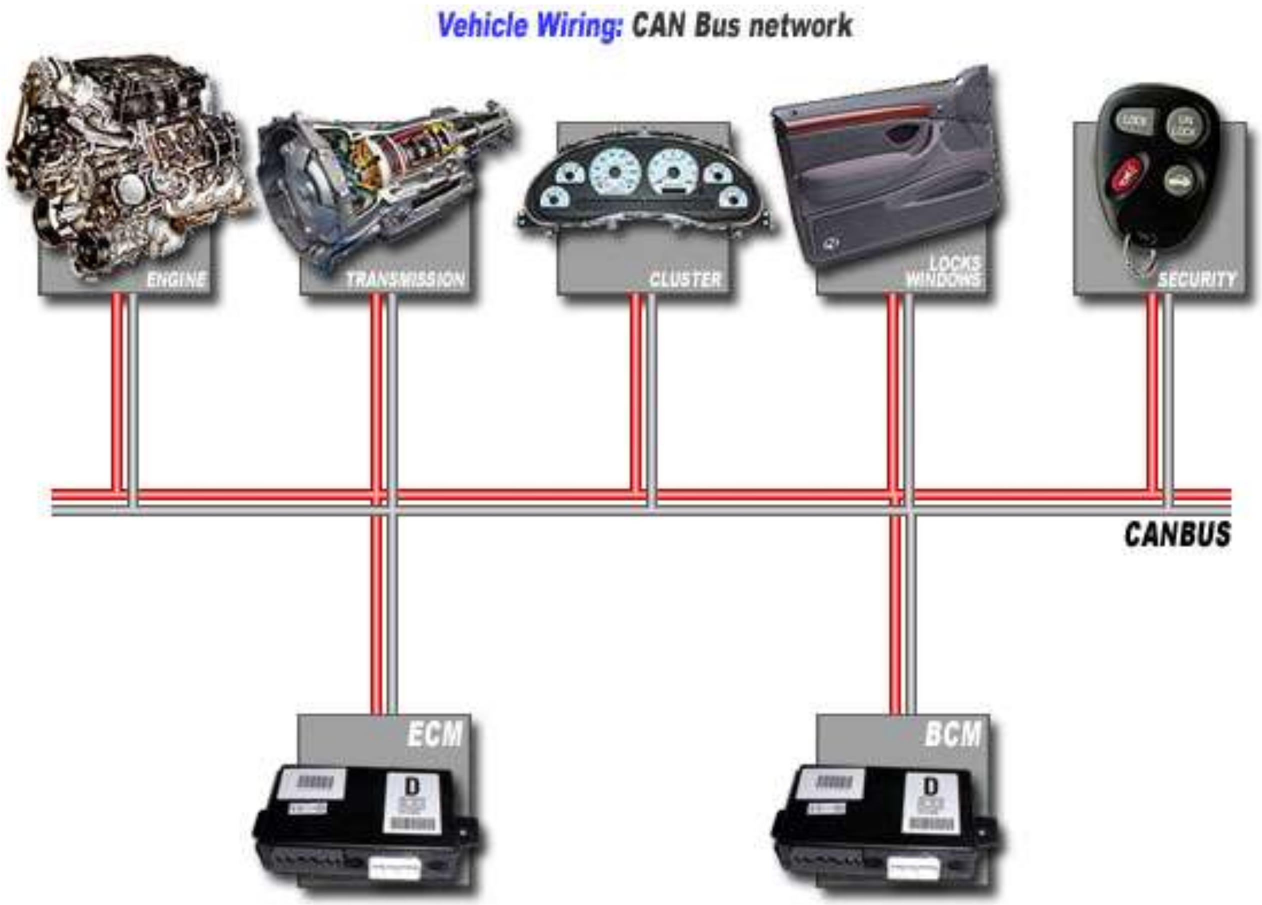
SIL to Target Code

- After a detailed description of the whole system with Simulink (1), we are ready to generate the target-code (2).
- Code generation:
  - Programming language C
  - If possible, directly out of Simulink (best practice)
  - Derive the C-Code from the Simulink Model (in case the automatic code generation does not work).



Co-funded by the Erasmus+ Programme of the European Union

Drivetrain bus system of a passenger car



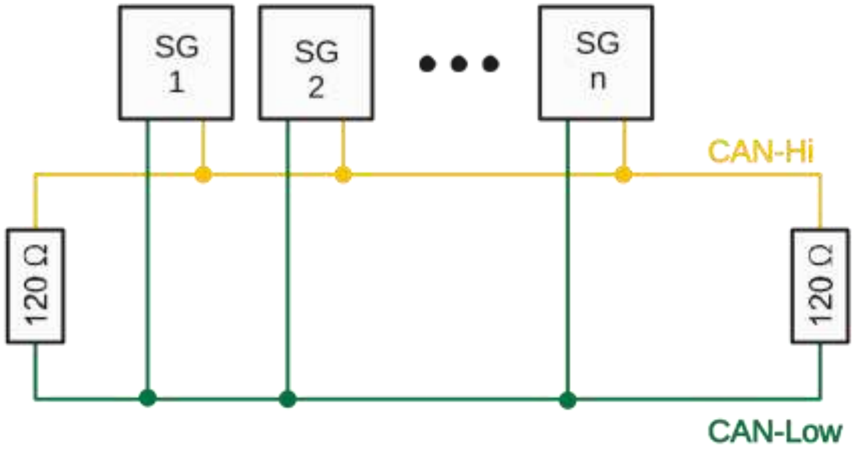
https://canbuskits.com/what.php

- Used for
  - 1 sensor shared for different ECU's
  - Sensor-ECU-connection
  - ECU dashboard connection, …
- Serial bus systems
  - 1 or 2 wires for robust data transfer
- Addidional
  - Low speed CAN for interieur …

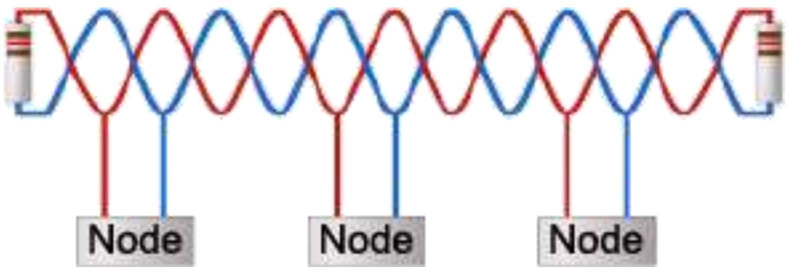
Co-funded by the
Erasmus+ Programme
of the European Union

Drivetrain bus system of a passenger car

- CAN-Bus (Control Area Network)
  - ISO 11898-1:2015
  - High-Speed-CAN , 250kBit/s, 500kBit/s, 1MBit/s
  - Low-Speed-CAN, <= 125kBit/s
  - Serial, members are not synchronized to each other
  - Non deterministic data transfer (no exactly defined transfer rate)
  - Unshielded twisted pair of 2 wires with termination resistors at both ends.



https://en.wikipedia.org/wiki/CAN_bus



https://www.can-cia.org/can-knowledge/

Co-funded by the Erasmus+ Programme of the European Union

CAN-DB Snowbird

Message	DLC	Signal	Startbit	Length	Order	Value Type	Factor	Offset	Min	Max	Unit	Table	Comment
MCU_to_BMS/ID 200	8	Motor speed	0	16	Intel	Unsigned	1	0	0	65000	rpm		
		Main_relay_ON	16	1	Intel	Unsigned	1	0	0	1	-	0 = Relay OFF 1 = Relay ON	BMS has to respect internal safety mechanisms
		not used	17	23	-	-							
		MCU_Temp	40	8	Intel	Unsigned	1	0	0	255	degC		
		MCU_status	48	8	-	-	-	-	-	-	-	Bit 0: driving Bit 1: charging	charger management done by MCU
		not used	56	8	-	-	-	-	-	-	-		
BMS_to_MCU_1/ID 201	8	Pack_Voltage	0	16	Intel	Unsigned	0,1	0	0	5000	V	total battery pack voltage	
		pack_Current	16	16	Intel	Signed	0,1	0	-1000	1000	A	total battery pack current < 0: discharge > 0: charge	
		SOC	32	8	-	Unsigned	1	0	0	100	%		from BMS SOC algorithm
		BMS_status_1	40	8	-	Unsigned	-	-	-	-	-	Bit 0: overvoltage warning Bit 1: undervoltage warning Bit 2: overtemperature warning Bit 3: overcurrent warning Bit 4: overcharge warning Bit 5: overdischarge warning Bit 6: repeated overdischarge Bit 7: isolation fault warning	

Co-funded by the Erasmus+ Programme of the European Union

Demo Regelung.c

Z:\VECTOR\CANAPE\10.0\RTx\Src\Regelung.c

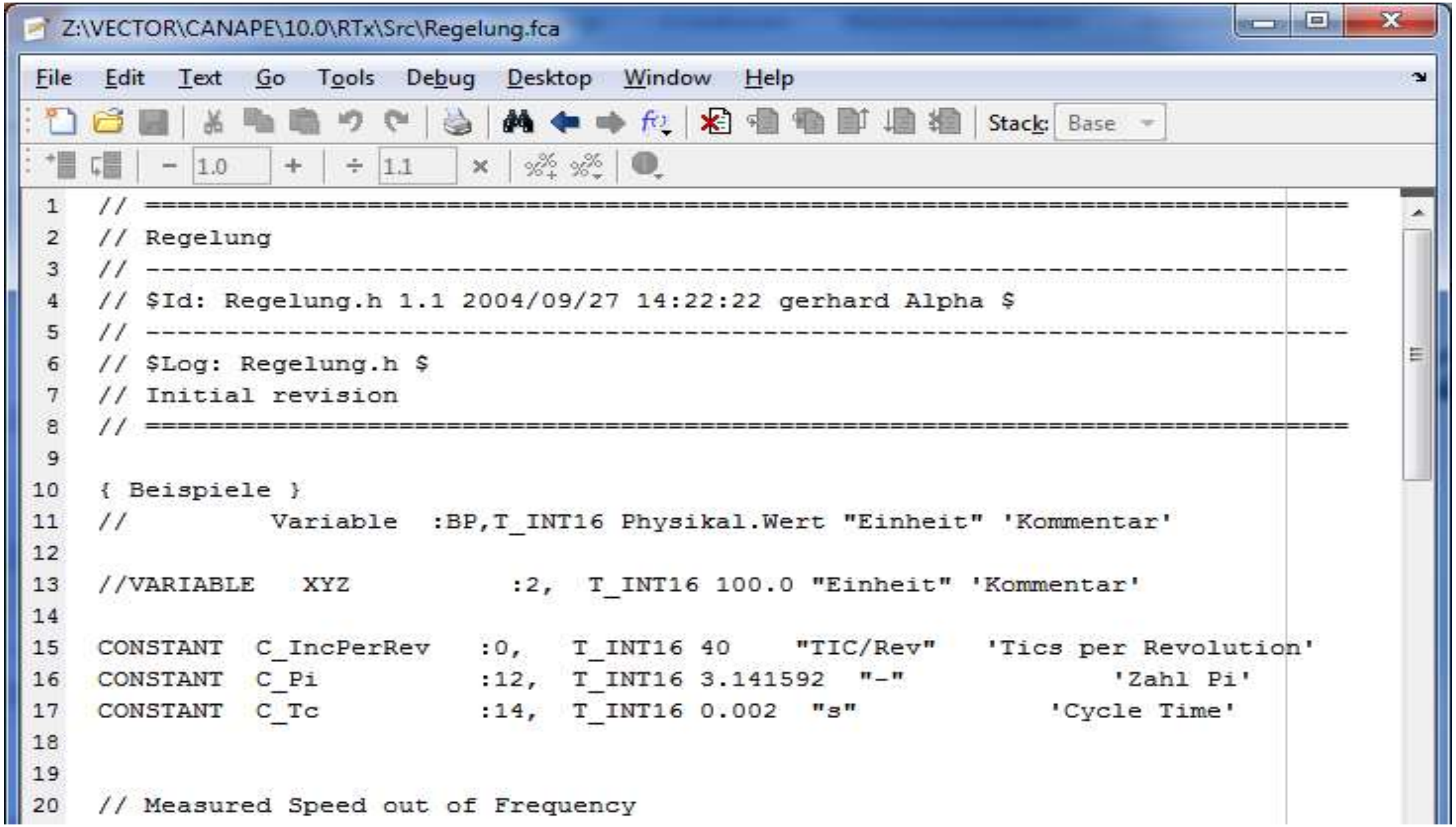
File Edit Text Go Tools Debug Desktop Window Help

```c
// ===========================================================================
// Regelung
// ---------------------------------------------------------------------------
// $Id: Main.c 1.2 2004/10/19 21:44:18 gerhard Alpha $
// ---------------------------------------------------------------------------
// $Log: Regelung.c $
// Initial revision - 12.6.2013 K. Reisinger
// ===========================================================================
#include <Types.h>
#include <Util.h>
#include "Appl.h"
#define NoExternRegelung
#include "Regelung.h"

@far void InitRegelung(void) {
    duSM_AWU = 0; //AWU-Part
    uSM_I   = 0; // Start-Value to Integrate for Controller
} // end InitRegelung()
```

Co-funded by the
Erasmus+ Programme
of the European Union

Definition of ASAM-2-Data



Co-funded by the
Erasmus+ Programme
of the European Union

Lessons Learned

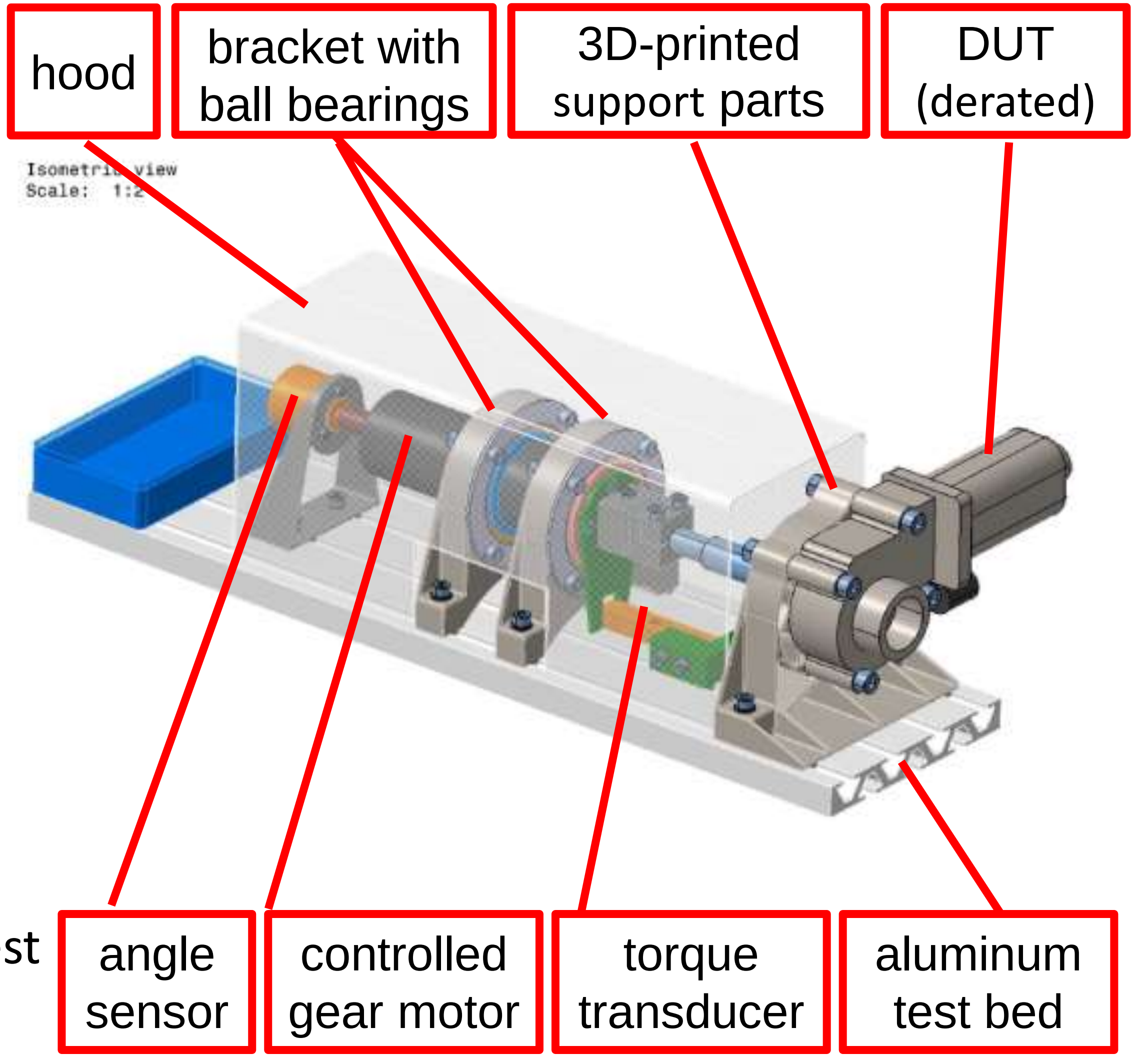
- Wide difference in understanding electrics and µP‘s among the students.
- 2 ECTS is very thought for this content.
- Requirements Management is the most unpopular topic - but necessary.
- Fixed point arithmetic is not that important for the engineer designing the mechatronic system, it‘s a task of the software developer.
- The Simulink-SW-model shall be compiled automatically to be loaded to the ECU – no C-code development for system engineers.
- Stateflow is the real way to model the process automation – but not part of curriculum.
- Simscape is the new way to model the plant – but not part of curriculum.
- Integration of mechatronic systems into test benches shall be added.

Co-funded by the Erasmus+ Programme of the European Union

Our next steps

- Low-Cost Mini-HIL

  Integration of controlled systems into test benches
  - 2 groups of 2 students:
    1. developing control software for current task = Device Under Test (DUT).
    2. Application of a HIL test bench and test automation.
- HIL test bench – low performance, full functionality
  - Controlled DC-motor
  - ECU with Simulink-Interface to develop the plant r
  - Shows all signals to drive a modern test bench.
- CANoe (vector)
  - Test bench automation defines how to drive the test and acquires the resultant signals.



Co-funded by the Erasmus+ Programme of the European Union

Introduction to UAS Mechatronic Laboratory Tutorial – our way to teach Mechatronics

4th Training in Rio de Janeiro, BRA

6th-9th of May 2019

Dr. Karl Reisinger & Thomas Lechner

Co-funded by the Erasmus+ Programme of the European Union