Verification of an Automotive Application Using Smart Component Headlight Leveling Circuit and Property Extraction

Jérôme Kirscher
Michael Lenz
Dieter Metzner
Georg Pelz

Infineon Technologies AG

BMAS08 2008-09-25

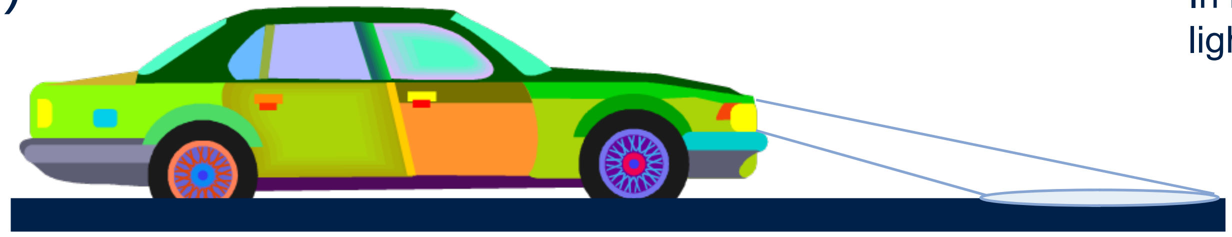
Never stop thinking

Overview

- Introduction to Headlight Leveling
- Application Description
- Application Modeling
- Smart Component Extraction
- Comparing Component measurements with simulation results
- Application Verification (simulation)

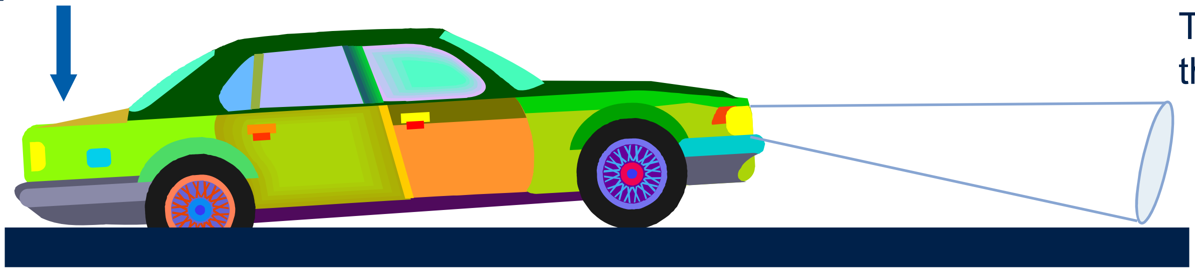
Introduction to Headlight Leveling

1)



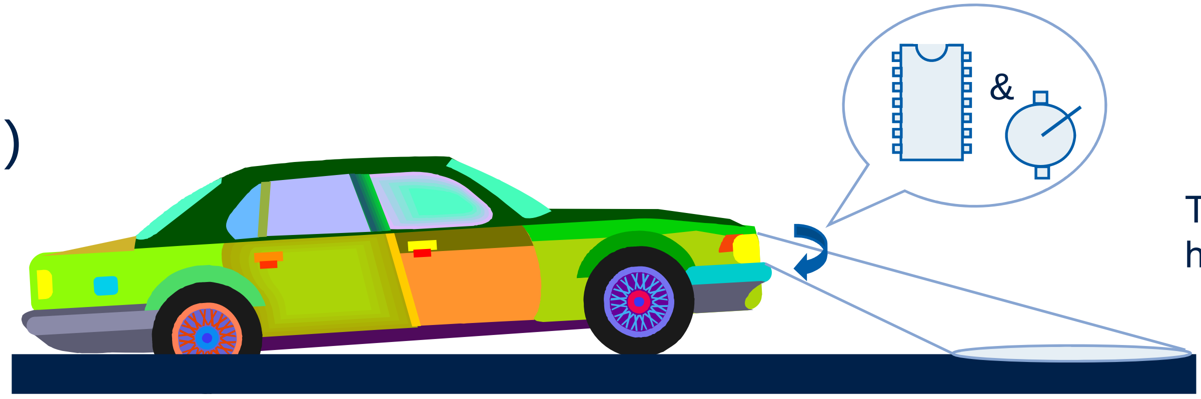
In normal load condition, the beam lights the road ahead

2)



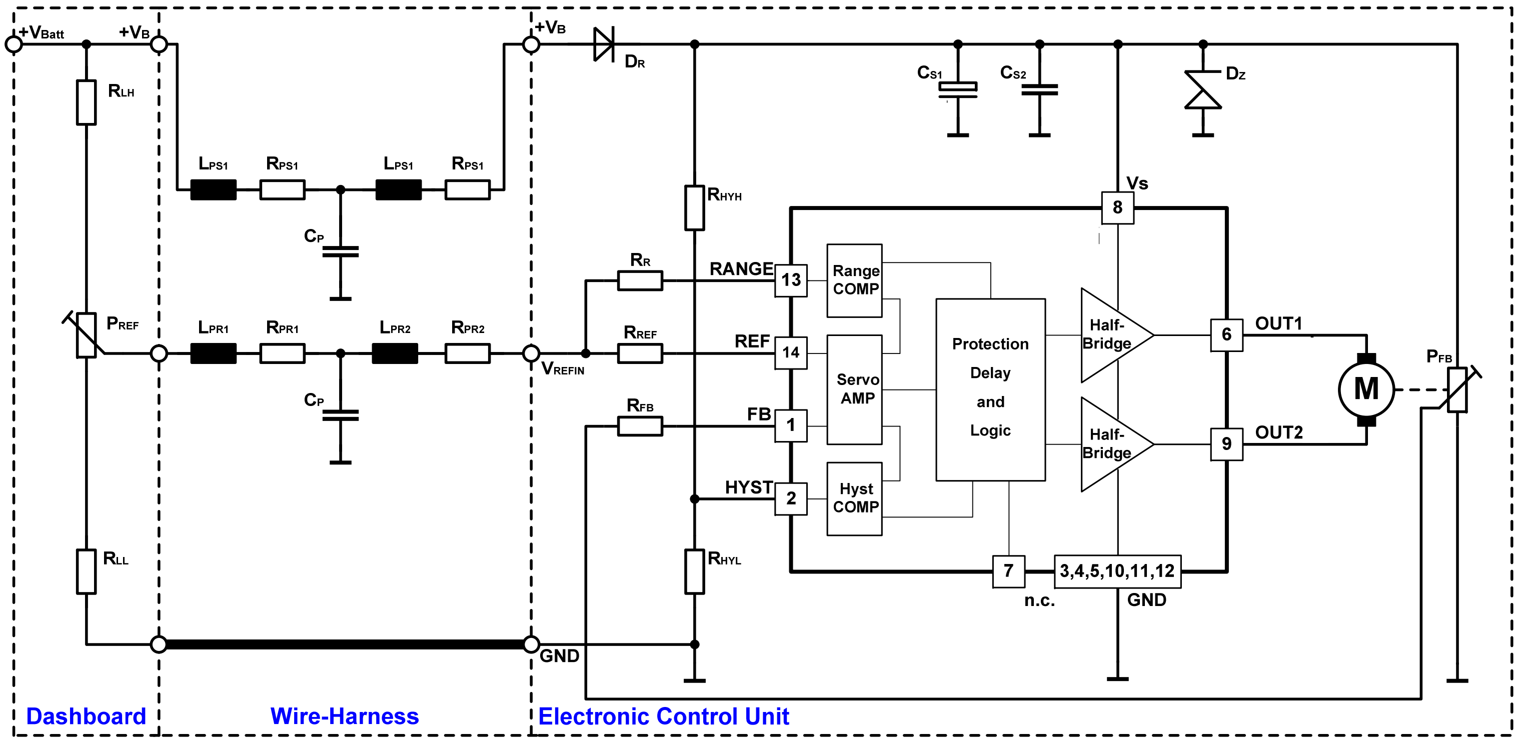
The back of the car is heavily loaded, the beam dazzles oncoming drivers

3)



The leveling system adjusts the headlight position

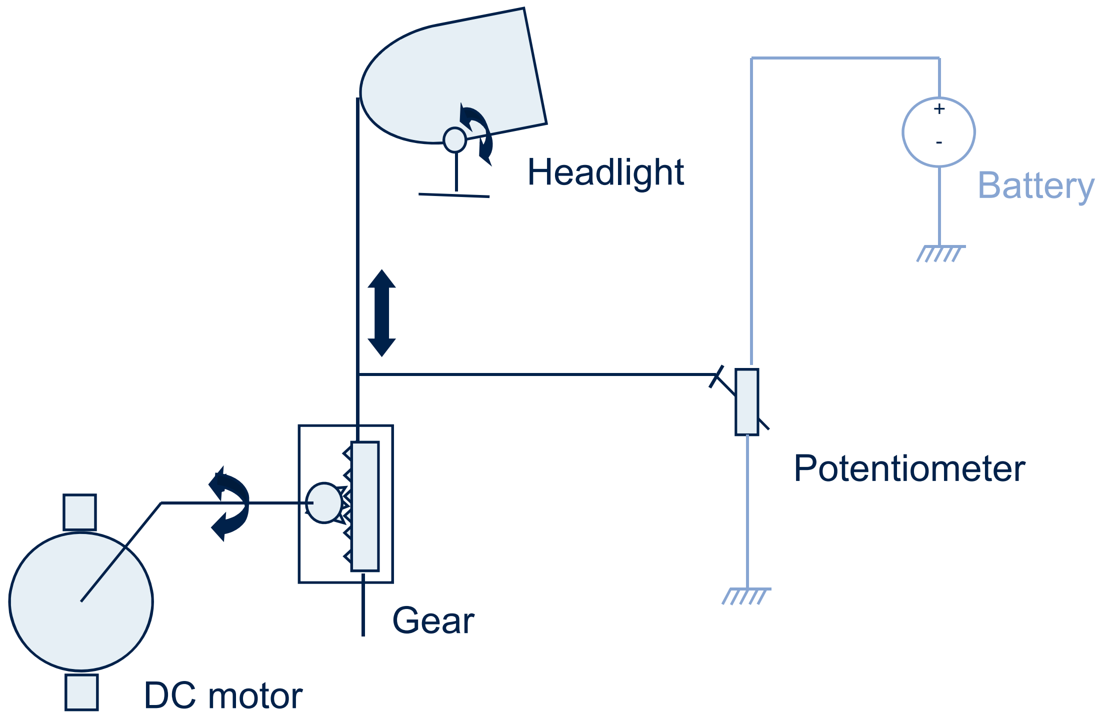
Application Description



- The headlights are moved by a DC motor, which is controlled by a full-bridge circuit (H-Bridge)
- One reference value is set (P_{ref}) and compared to a feedback value (P_{Fb}) by the logic part of the circuit

Application Description (2)

- The Mechanical Part of the system contains:
  - DC Motor
  - Gear
  - Headlight
  - Potentiometer

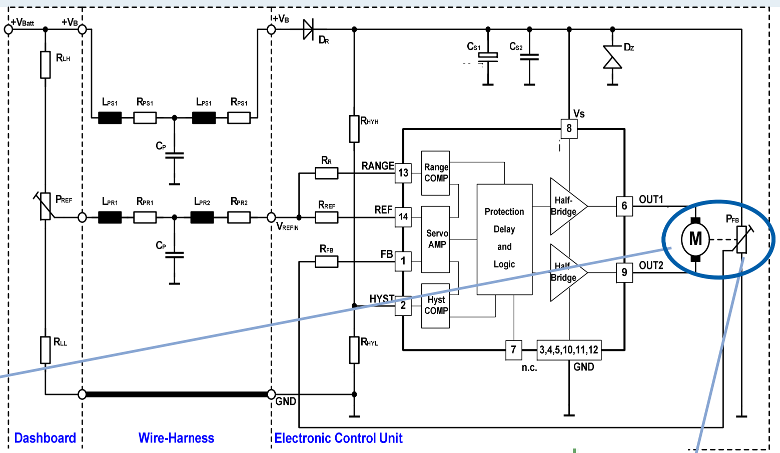


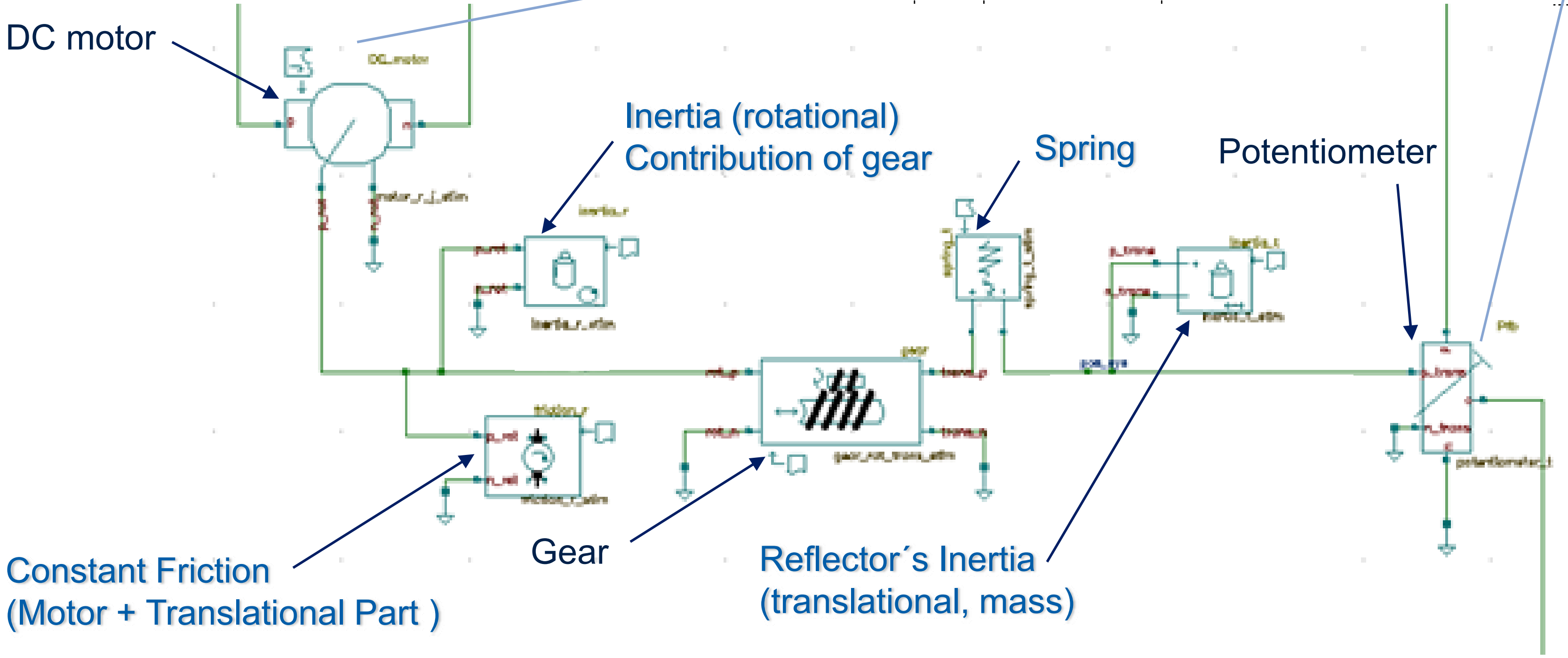
Application Modeling

- The Electronical Components (except the circuit) and the Mechanical Part are modelled using VHDL-AMS
- The circuit is integrated on transistor-level
- The models´ equations are based on electro / mechanical conservation laws
- The models of the Mechanical Part contains following blocks:
  - DC Motor
  - Gear
  - Potentiometer

    + friction and inertia, considered as separated contributors

Application Modeling (2)
Mechanical Part

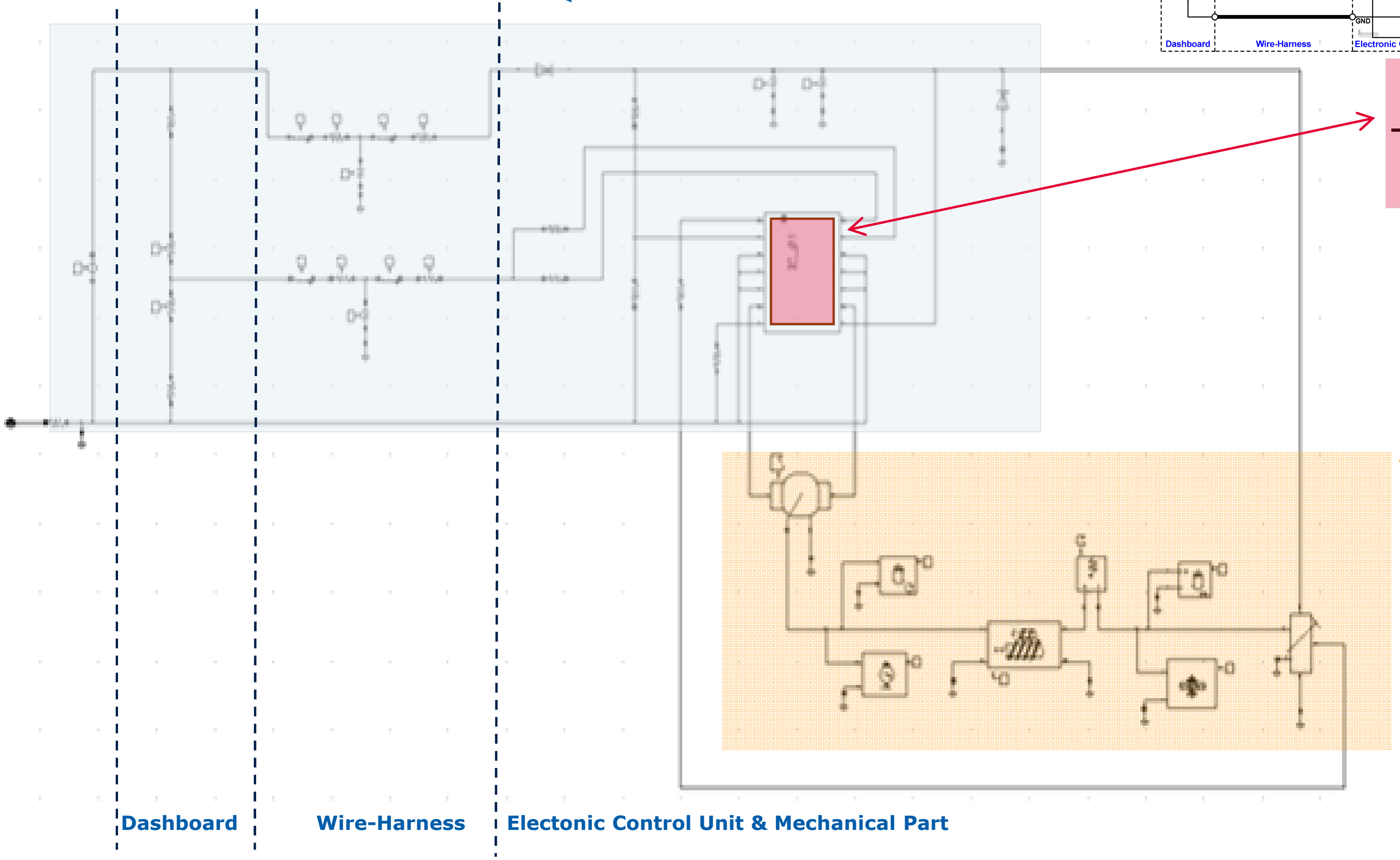


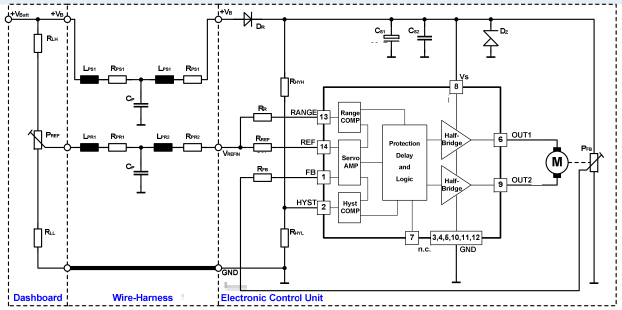


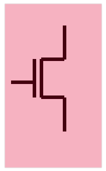
Application Modeling (3)
Whole system

entity resistor is

…







entity dc_motor **is**

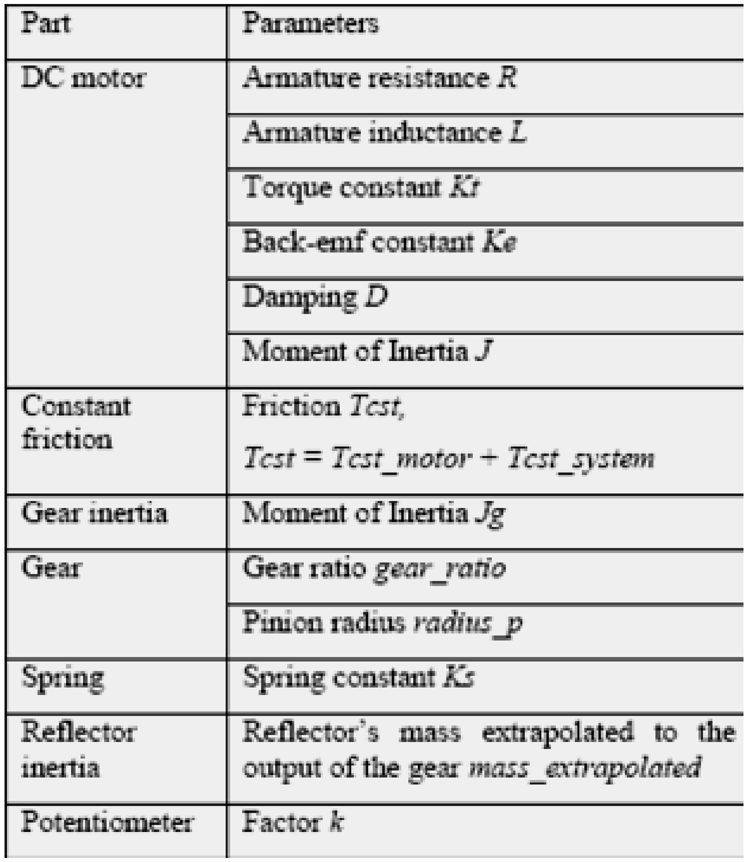
generic (r_wind

…

Parameters extracted from real application

Smart Component Extraction

- All parameters are extracted from simple measurements performed on the real application
- Parameters are listed here:



Part	Parameters
DC motor	Armature resistance *R*
	Armature inductance *L*
	Torque constant *Kt*
	Back-emf constant *Ke*
	Damping *D*
	Moment of Inertia *J*
Constant friction	Friction *Tcst*, *Tcst* = *Tcst_motor* + *Tcst_system*
Gear inertia	Moment of Inertia *Jg*
Gear	Gear ratio *gear_ratio*
	Pinion radius *radius_p*
Spring	Spring constant *Ks*
Reflector inertia	Reflector's mass extrapolated to the output of the gear *mass_extrapolated*
Potentiometer	Factor *k*

- DC motor parameters extraction is hightlighted in what follows

Smart Component Extraction (2)
DC Motor equations

$$T = - Kt * i + D * \omega + J * d(\omega)/dt$$

Smart Component Extraction (3)
DC Motor equations

$$T = \underbrace{- Kt * i}_{\text{- generated torque (Kt : torque coefficient)}} + D * \omega + J * d(\omega)/dt$$

Smart Component Extraction (4)
DC Motor equations

$$T = - Kt * i + \underbrace{D * \omega}_{} + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)

- viscous damping loss (D : damping coefficient)

Smart Component Extraction (5)
DC Motor equations

$$T = - Kt * i + D * \omega + \underbrace{J * d(\omega)/dt}$$

- generated torque (Kt : torque coefficient)

- viscous damping loss (D : damping coefficient)

- **inertial losses (J : moment of inertia)**

Smart Component Extraction (6)
DC Motor equations

$T = - Kt * i + D * \omega + J * d(\omega)/dt$

- generated torque (Kt : torque coefficient)
- viscous damping loss (D : damping coefficient)
- inertial losses (J : moment of inertia)

$U = Ke * \omega + R * i + L * d(i)/dt$

Smart Component Extraction (7)
DC Motor equations

$$T = - Kt * i + D * \omega + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)
- viscous damping loss (D : damping coefficient)
- inertial losses (J : moment of inertia)

$$U = \underbrace{Ke * \omega} + R * i + L * d(i)/dt$$

- back-EMF (Ke : EMF coefficient)

Smart Component Extraction (8)
DC Motor equations

$$T = - Kt * i + D * \omega + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)

- viscous damping loss (D : damping coefficient)

- inertial losses (J : moment of inertia)

$$U = Ke * \omega + \underbrace{R * i} + L * d(i)/dt$$

- back-EMF (Ke : EMF coefficient)

- **winding resistance voltage drop (R : winding resistance)**

Smart Component Extraction (9)
DC Motor equations

$$T = - Kt * i + D * \omega + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)
- viscous damping loss (D : damping coefficient)
- inertial losses (J : moment of inertia)

$$U = Ke * \omega + R * i + \underbrace{L * d(i)/dt}$$

- back-EMF (Ke : EMF coefficient)
- winding resistance voltage drop (R : winding resistance)
- **winding inductance voltage drop (L : winding inductance)**

Smart Component Extraction (10)
DC Motor equations

$$T = -K_t * i + D * \omega + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)
- viscous damping loss (D : damping coefficient)
- inertial losses (J : moment of inertia)

Note:
Kt = Ke when SI units are used

$$U = K_e * \omega + R * i + L * d(i)/dt$$

- back-EMF (Ke : EMF coefficient)
- winding resistance voltage drop (R : winding resistance)
- winding inductance voltage drop (L : winding inductance)

Smart Component Extraction (11)
DC Motor equations

$$T = -Kt * i + D * \omega + J * d(\omega)/dt$$

- generated torque (Kt : torque coefficient)
- viscous damping loss (D : damping coefficient)
- inertial losses (J : moment of inertia)

$$U = Kt * \omega + R * i + L * d(i)/dt$$

- back-EMF (Ke = Kt)
- winding resistance voltage drop (R : winding resistance)
- winding inductance voltage drop (L : winding inductance)

Smart Component Extraction (12)
DC Motor Parameter extraction

- L extraction :

$U = Kt * \omega + R * i + \mathbf{L} * d(i)/dt$

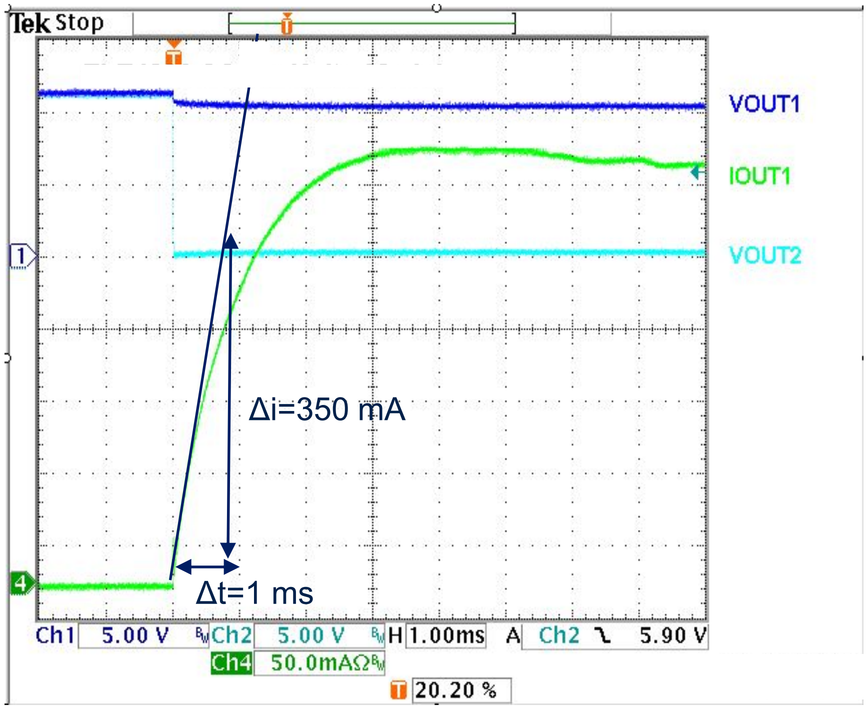
- Motor in inrush

- $U = \underbrace{Kt * \omega}_{=0} + \underbrace{R * i}_{=0} + L * d(i)/dt$

- the slope is measured:

$L = U * \Delta t/\Delta i$

$$\boxed{L = 31.4 \text{ mH}}$$



Smart Component Extraction (13)
DC Motor Parameter extraction

$U = Kt * \omega + \mathbf{R} * i + L * d(i)/dt$

- R extraction :

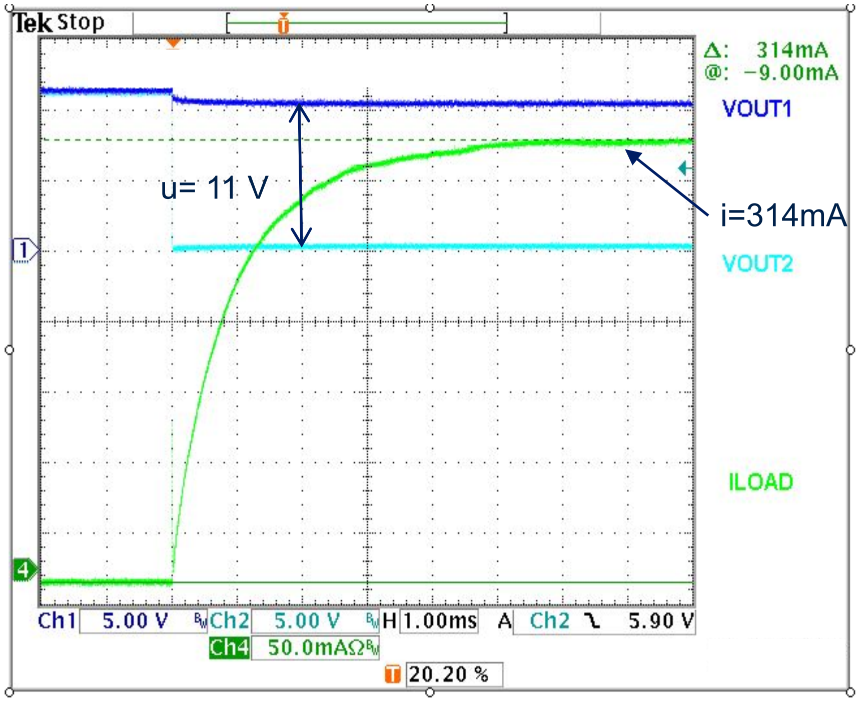
- **The rotor is blocked**

- **When the steady state is reached :**

$$U = \underbrace{Kt * \omega}_{=0} + R * i + \underbrace{L * d(i)/dt}_{=0}$$

- **R = U / i**

$$\boxed{R = 35\ \Omega}$$



Smart Component Extraction (14)
DC Motor Parameter extraction

- Kt extraction :

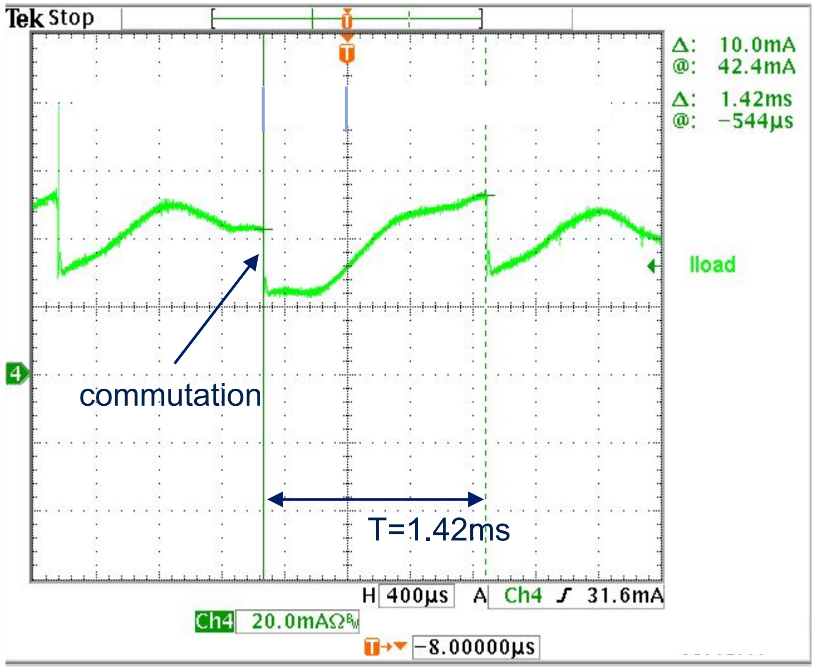
$U = \mathbf{Kt} * \omega + R * i + L * d(i)/dt$

- The rotor rotates
- One rotation induces 4 current commutations in the motor
- The time between two commutations (1/4 rotation) is measured
- The angular velocity (ω_s) is deduced (1 rotation ≡ 2π [rad])

$$\omega_s = 1106 \text{ rad/s}$$

- $Kt = U / \omega_s$

$$\boxed{Kt = 9.95 \text{ mV/(rad/s)}}$$

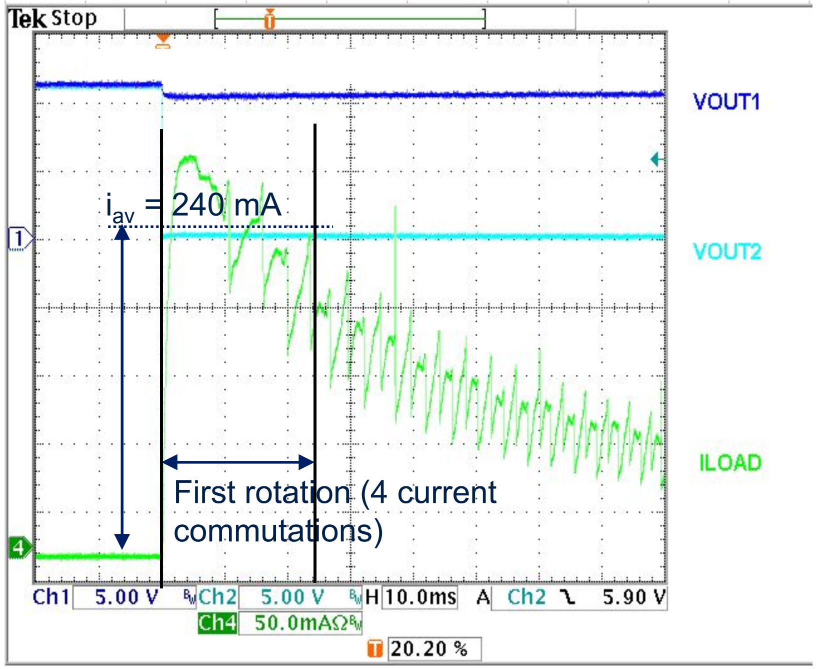


Smart Component Extraction (15)
DC Motor Parameter extraction

- J extraction :

$T = -Kt * i + d * \omega + J * d(\omega)/dt$

- **Motor in inrush without load**
- **The angular velocity reached after 25 ms (1 rotation) is read out**
- **The angular acceleration (d(ω)/dt) is then approximated**
- **The average current during the first rotation (iav) is read out**
- **T is the Torque delivered by the motor (= 0.0 without load)**
- $0.0 = -Kt * i_{av} + \underbrace{d * \omega}_{=0} + J * d(\omega)/dt$



$$\boxed{J = 1.52e\text{-}7\ Kgm^2}$$

Smart Component Extraction (16)
DC Motor Parameter extraction

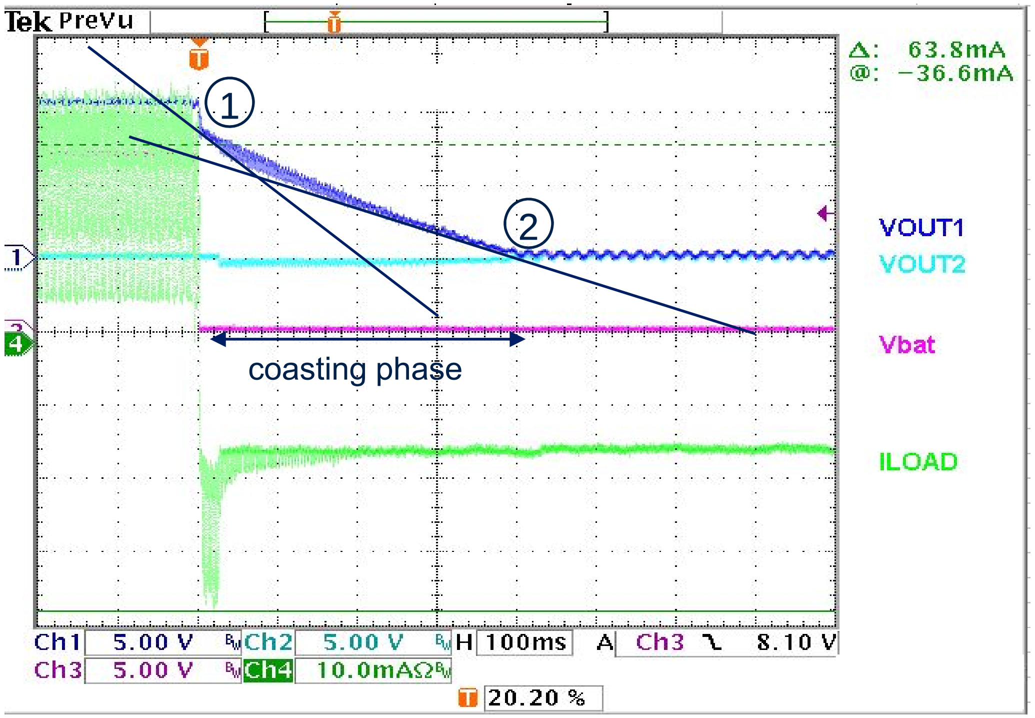
- D and Mr extraction :

$T = -Kt * i + \mathbf{d} * \omega + J * d(\omega)/dt \quad \mathbf{+ Mr}$

(Mr : constant friction)

- The rotor is disconnected from the battery (i = 0.0 A), no load

- The slopes are measured at the beginning ① and at the end ② of the coasting phase to get the angular acceleration (Δω/Δt))



- The equation is evaluated at the beginning and at the end of the coasting phase to isolate D and Mr (with i = 0.0, T = 0.0)

① $J * d(\omega)/dt = -Mr - d * \omega_s$

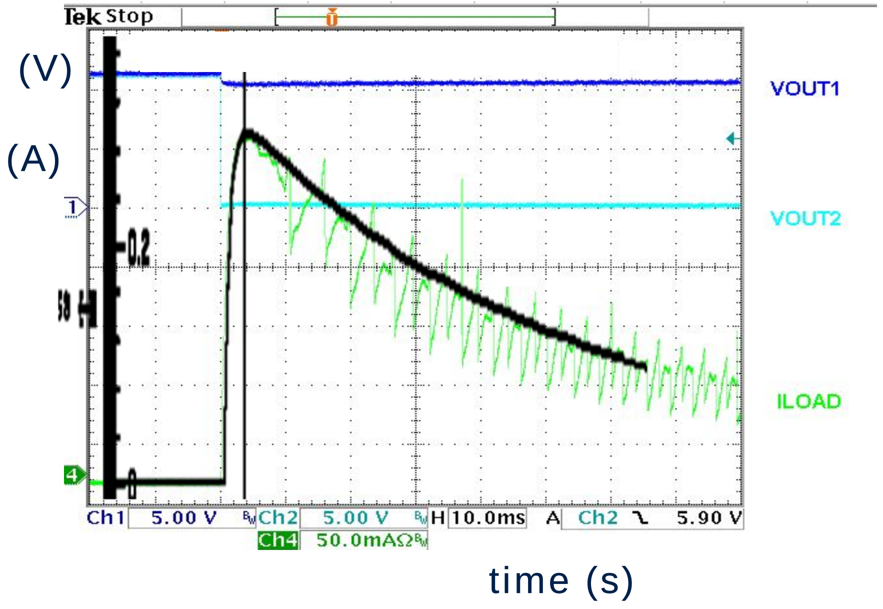
② $J * d(\omega)/dt = -Mr - \underbrace{d * \omega}_{=0}$

Mr = 3.73e-4 Nm

D = 1.69e-7 Nm/(rad/s)

Comparing Component measurements with simulation results

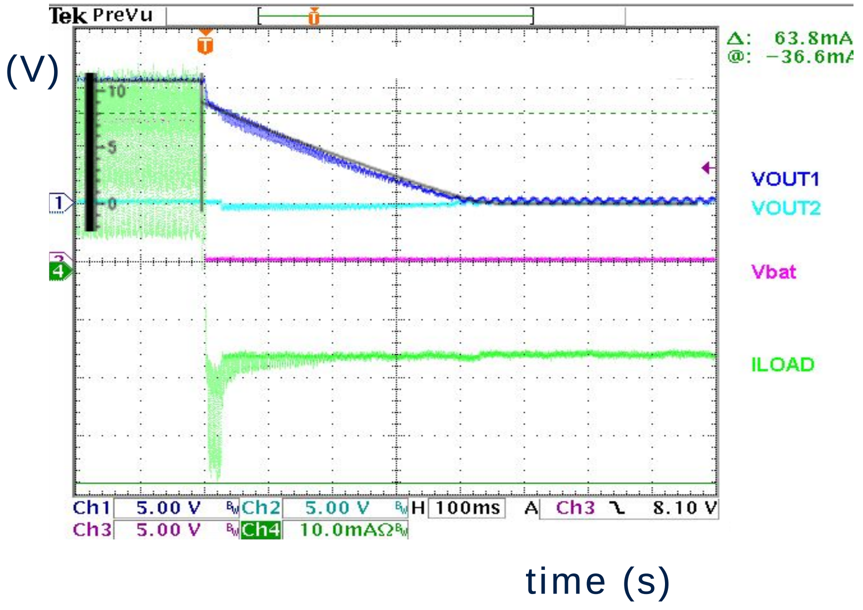
- Motor in inrush



Black : model current (Simulation)

Green : motor current (Measurement)

- Coasting phase

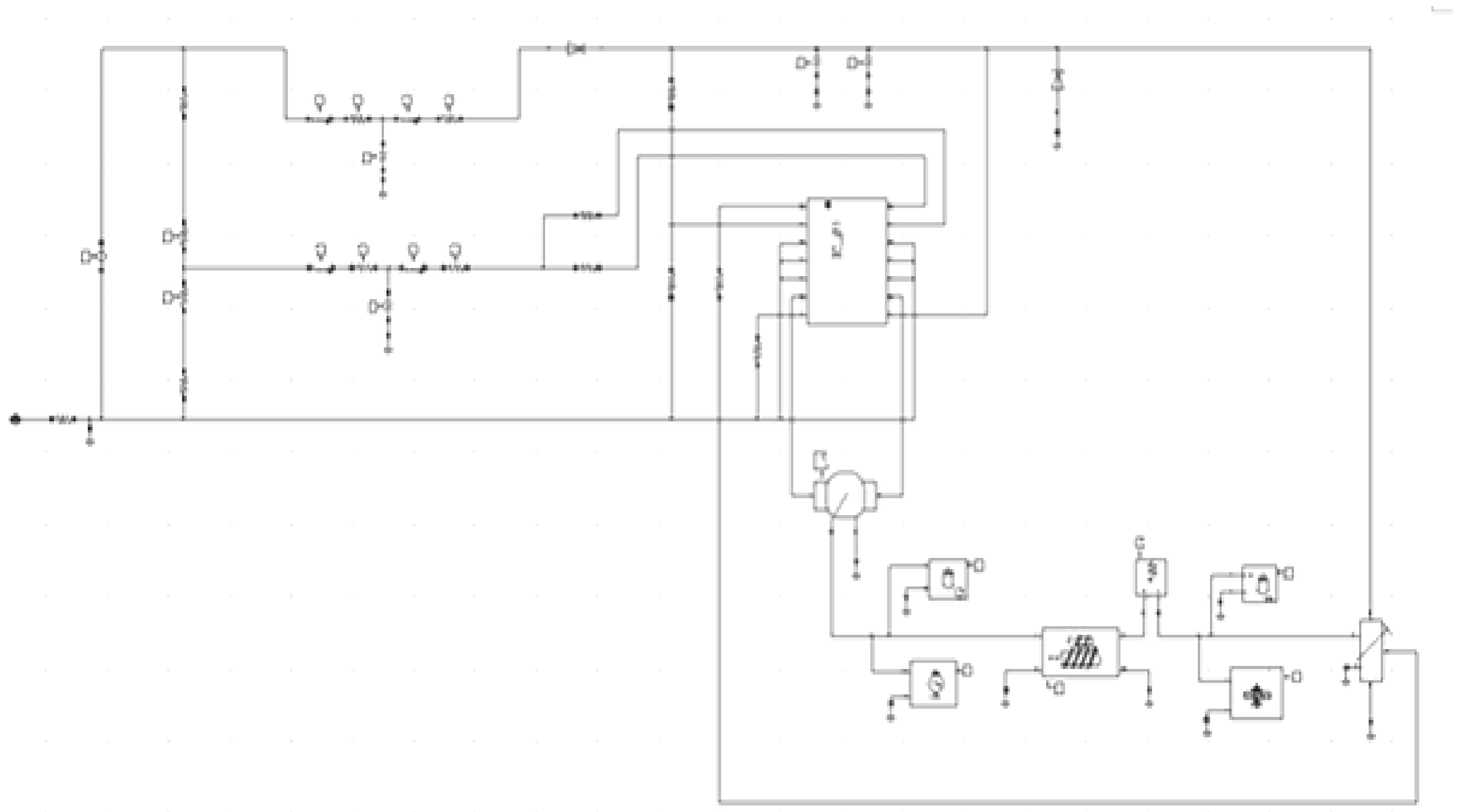


Black : DC motor pin (Simulation)

Dark blue : DC motor connector (Measurement)

Application Verification (simulation)

- The Test Bench is the Application Model defined earlier

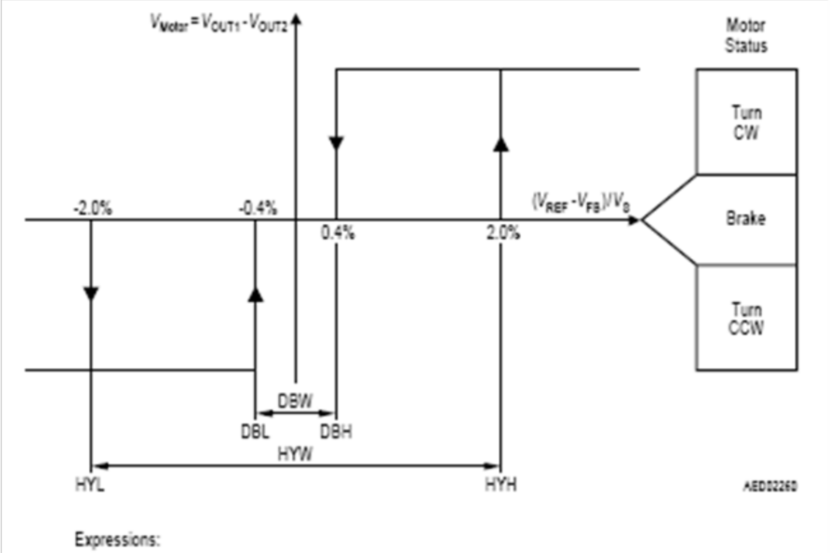


Application Verification (2)
Tracking simulation

- The typical servo behavior is investigated:

  i.e. a reference is settled (input) and the feedback is observed (output)

- Application typical behavior:
  - To prevent oscillations, a degree of *hysteresis* is introduced between the reference and feedback signals
  - To avoid high mechanical stress, the braking is carried gently over a period of time. This region between braking and stopping is referred to as *deadband*

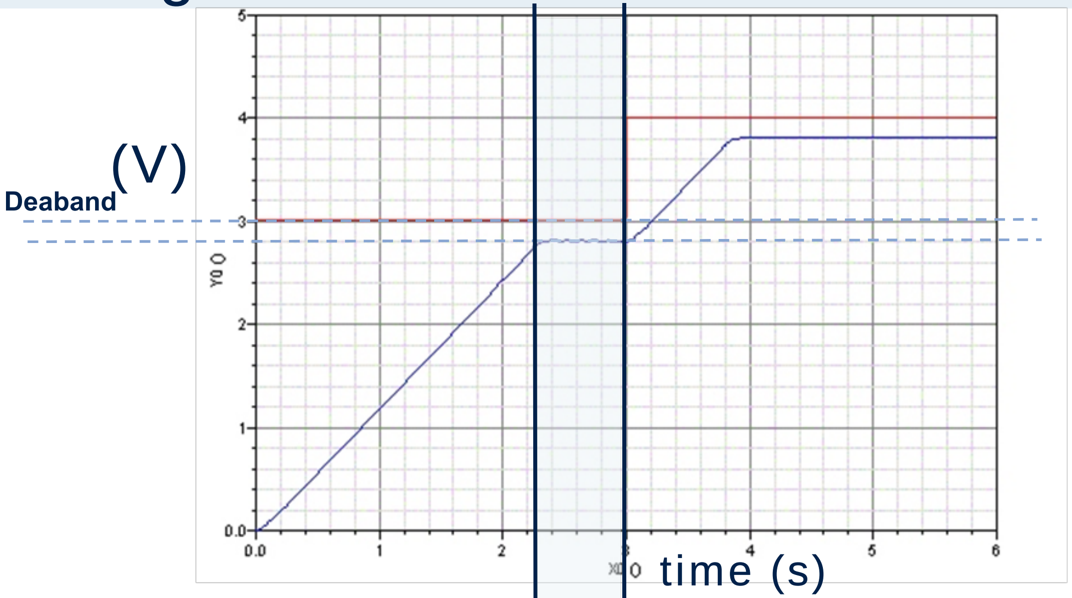


Expressions:
HY = Hysteresis
DB = Deadband
H = High
L = Low
W = Window

CW: clockwise

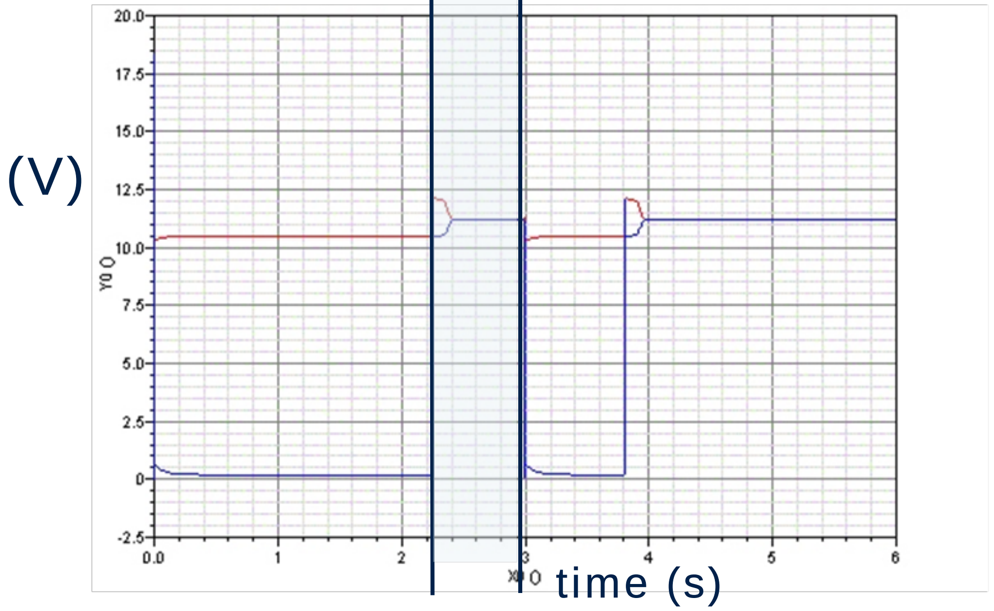
CCW: counterclockwise

Application Verification (3)
Tracking simulation : results



Red: reference voltage

Blue: feedback voltage



Red: pin_1 DC Motor

Blue: pin_2 DC Motor

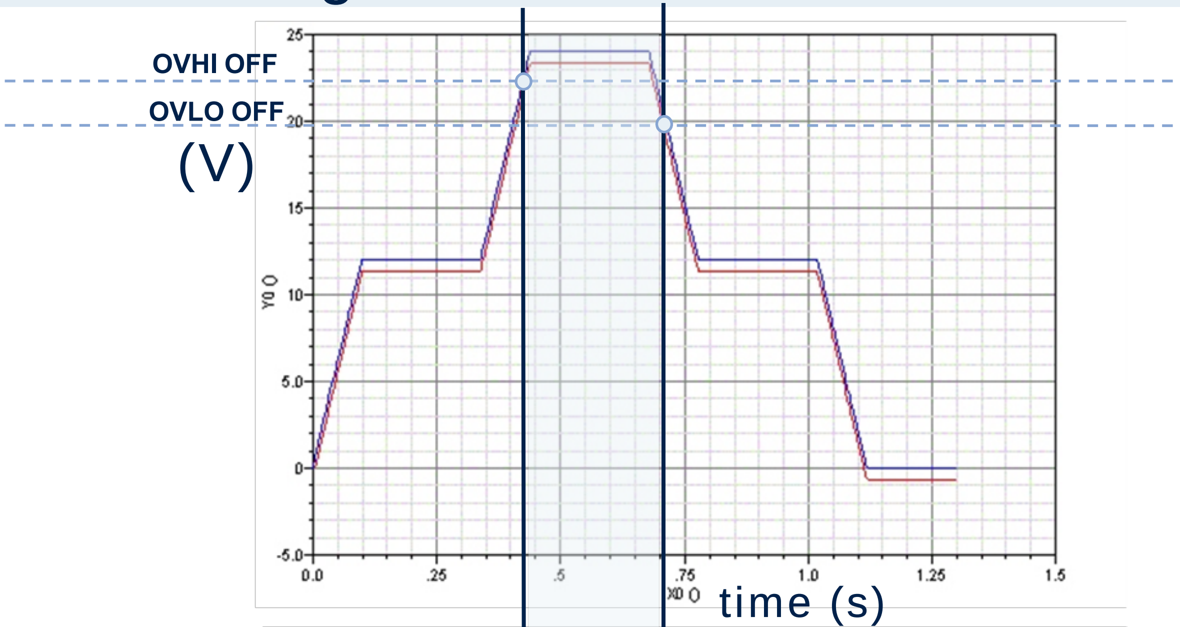
Application Verification (4)
Overvoltage simulation

- The Overvoltage Protection is investigated

- The Overvoltage Protection is implemented as follows:
  - The control circuit switches OFF the output stages to “High Impedance” if the supply voltage of the circuit reaches the overvoltage threshold *OVHI OFF*
  - The device switches on again when the supply voltage decreases to the *OVLO OFF* threshold, which is lower than the previous threshold (hysteresis)

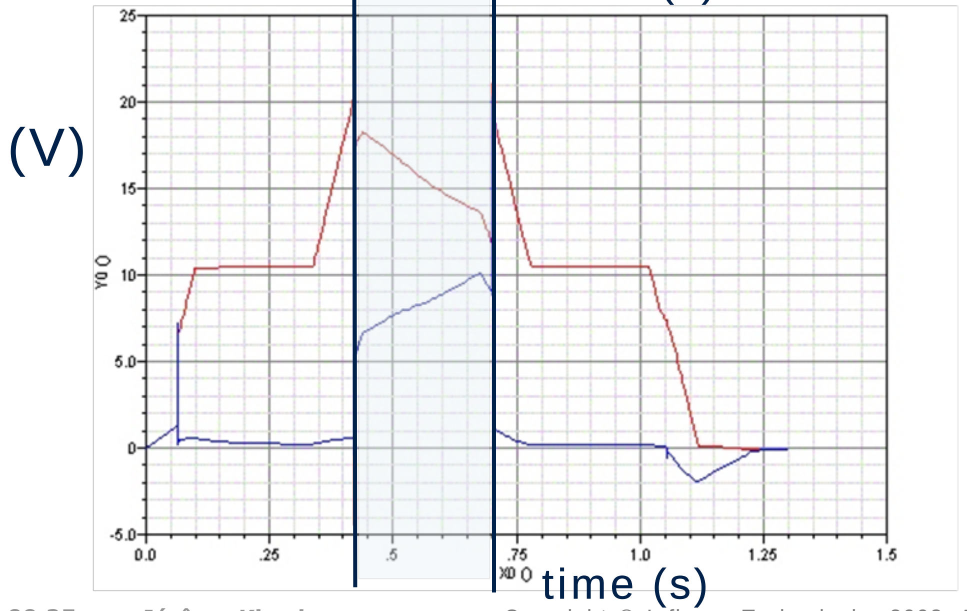
Application Verification (5)

Overvoltage simulation : results



Red: circuit supply pin

Blue: battery



Red: pin_1 DC Motor

Blue: pin_2 DC Motor

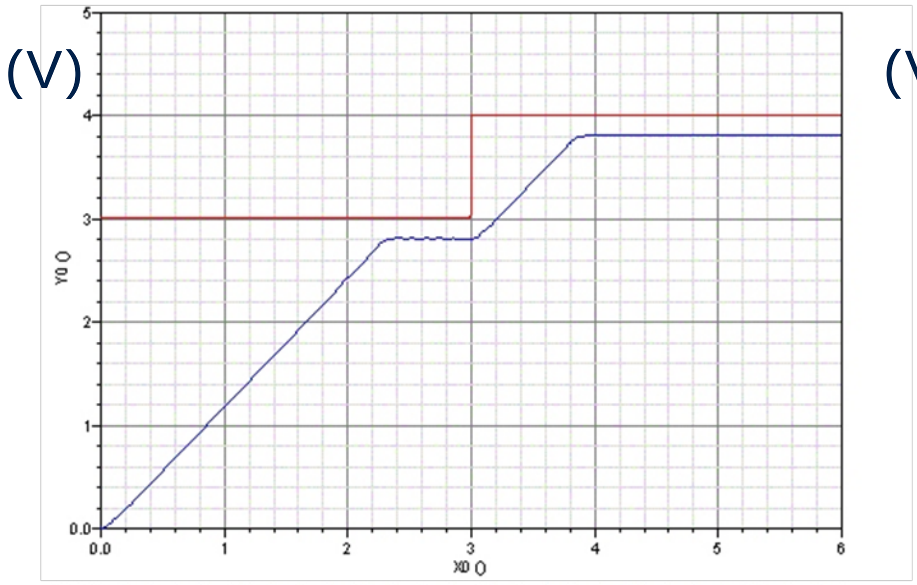
CONCLUSION

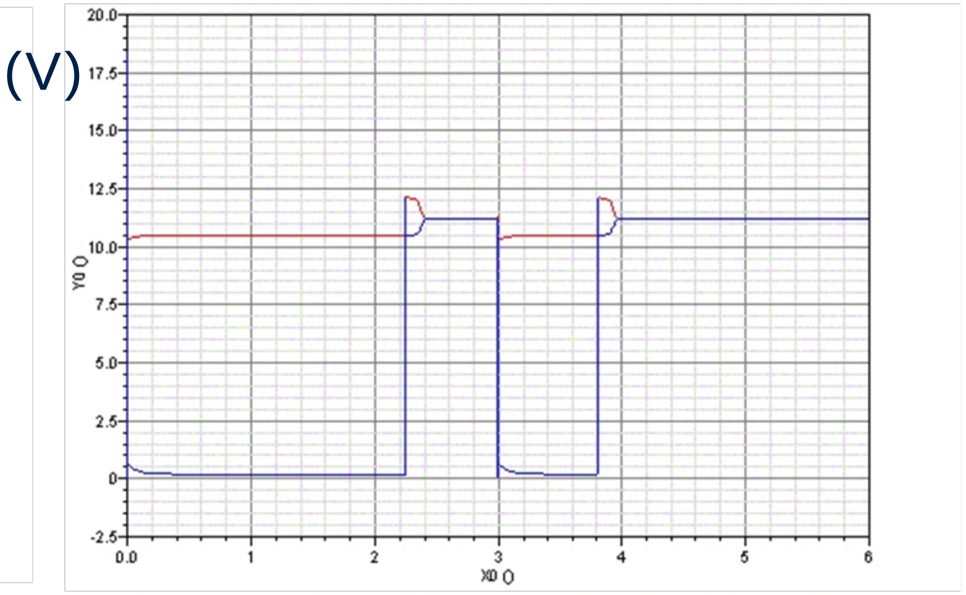
- Smart method to extract properties has been shown
- Verification feasability has been illustrated
- Open wide range of simulative activities
  - Exhaustive Verification
  - Robustness Investigation
  - Worst Case
  - Parametrization
  - Effective sizing
  - ...

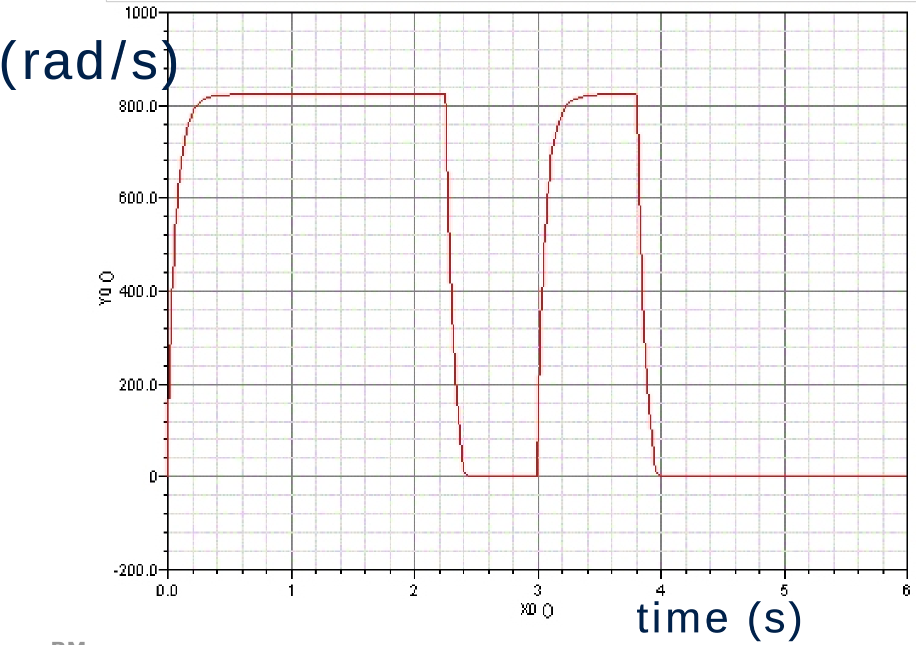
Thank you for your attention,

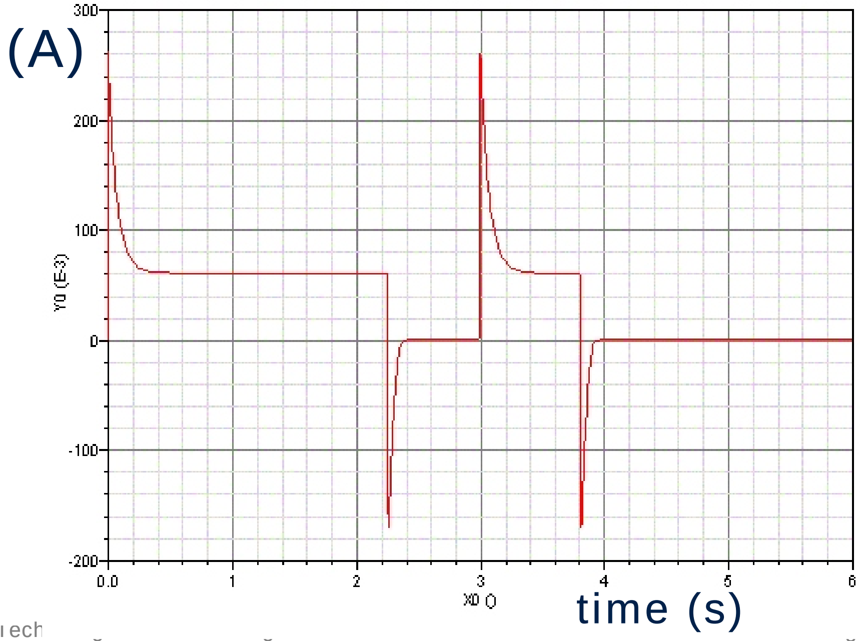
Questions?

Application Verification (4)
Tracking simulation : results


(V)
Y0 ()
X0 ()
0.0
1
2
3
4
5
6


(V)
Y0 ()
X0 ()
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0
-2.5


(rad/s)
Y0 ()
X0 ()
time (s)


(A)
Y0 (E-3)
X0 ()
time (s)