Design of a Computer-based Control System

Job van Amerongen

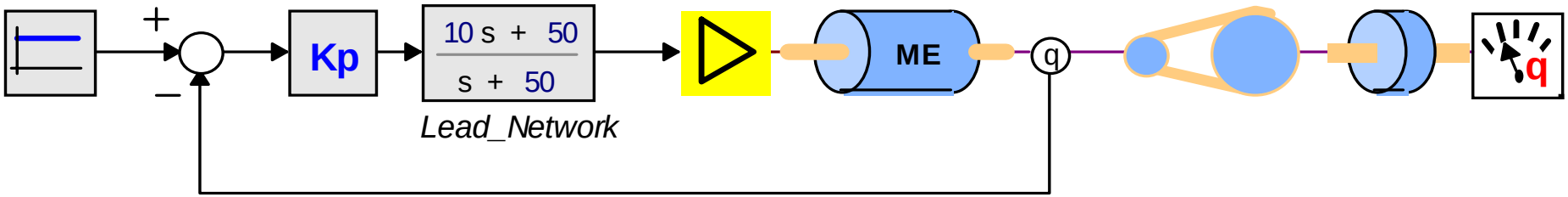
Control Engineering, Department of Electrical Engineering
University of Twente, Netherlands

www.ce.utwente.nl/amn
J.vanAmerongen@utwente.nl

Contents

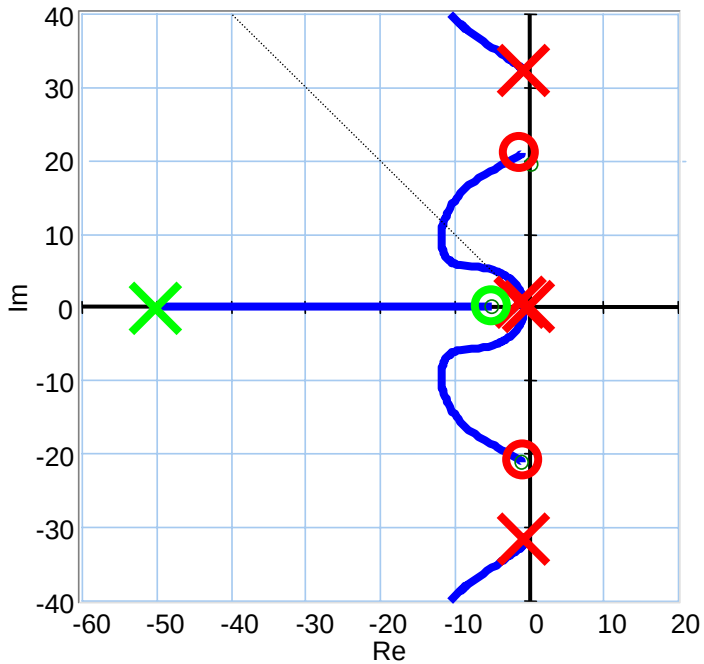
- Translation of continuous-time designs into discrete controllers
- Choosing a proper sampling rate
- Anti aliasing
- From design to realisation
- Automatic C-code generation

Lead network



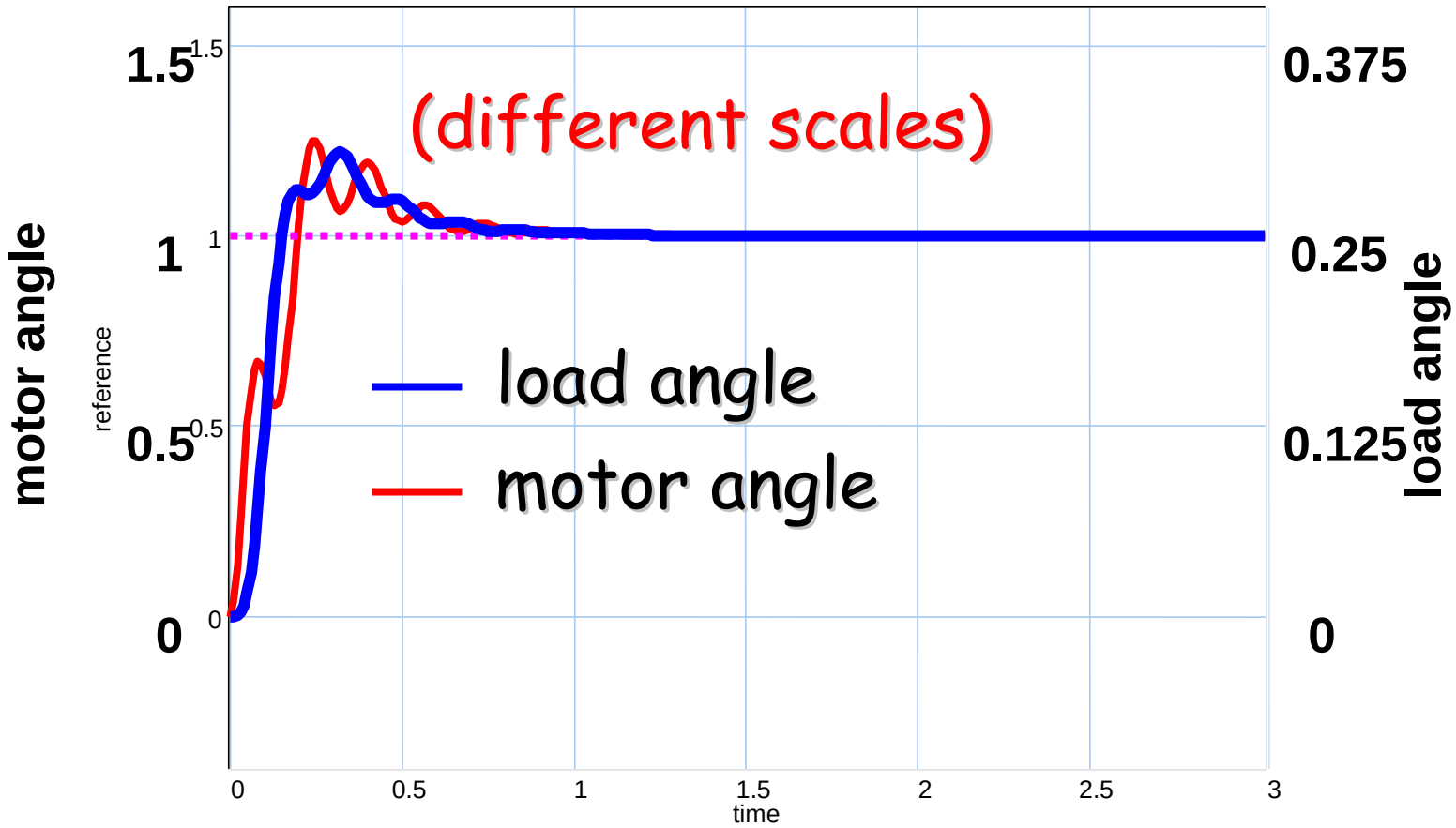
Lead network:

K = 1.9
zero in -5
pole in -50

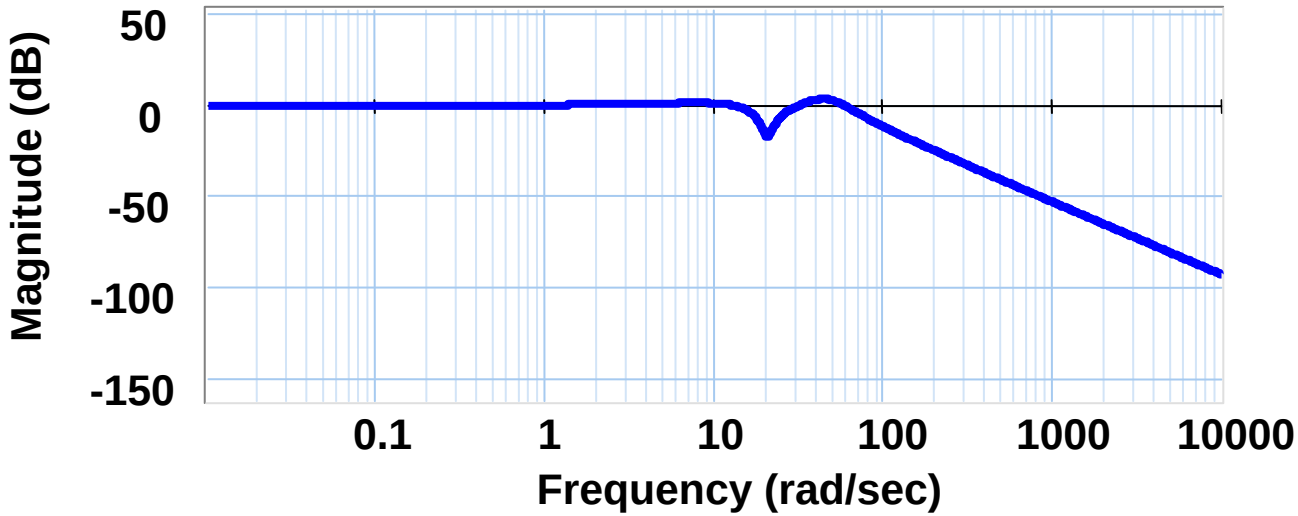


C2_lead_network_motor_only_no prefilter.em

Response



Bode Closed-loop system



Frequencies up to 60 rad/s clearly present in the closed loop system

Requirements

- Design a computer-based controller
- Similar performance as the continuous time system
- Closed-loop bandwidth 60 rad/s

  $\rightarrow \omega_s \geq 10\omega_b$ = 600 rad/s $\rightarrow f_s$ = 100 Hz

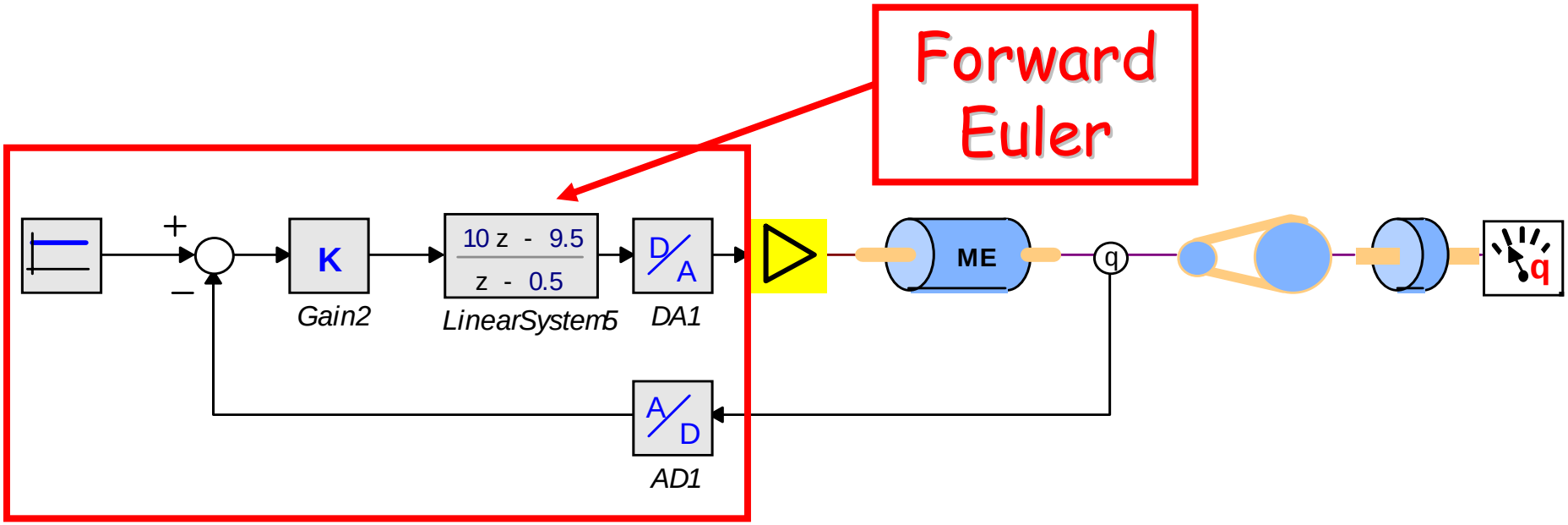
Remember…

- sampling time

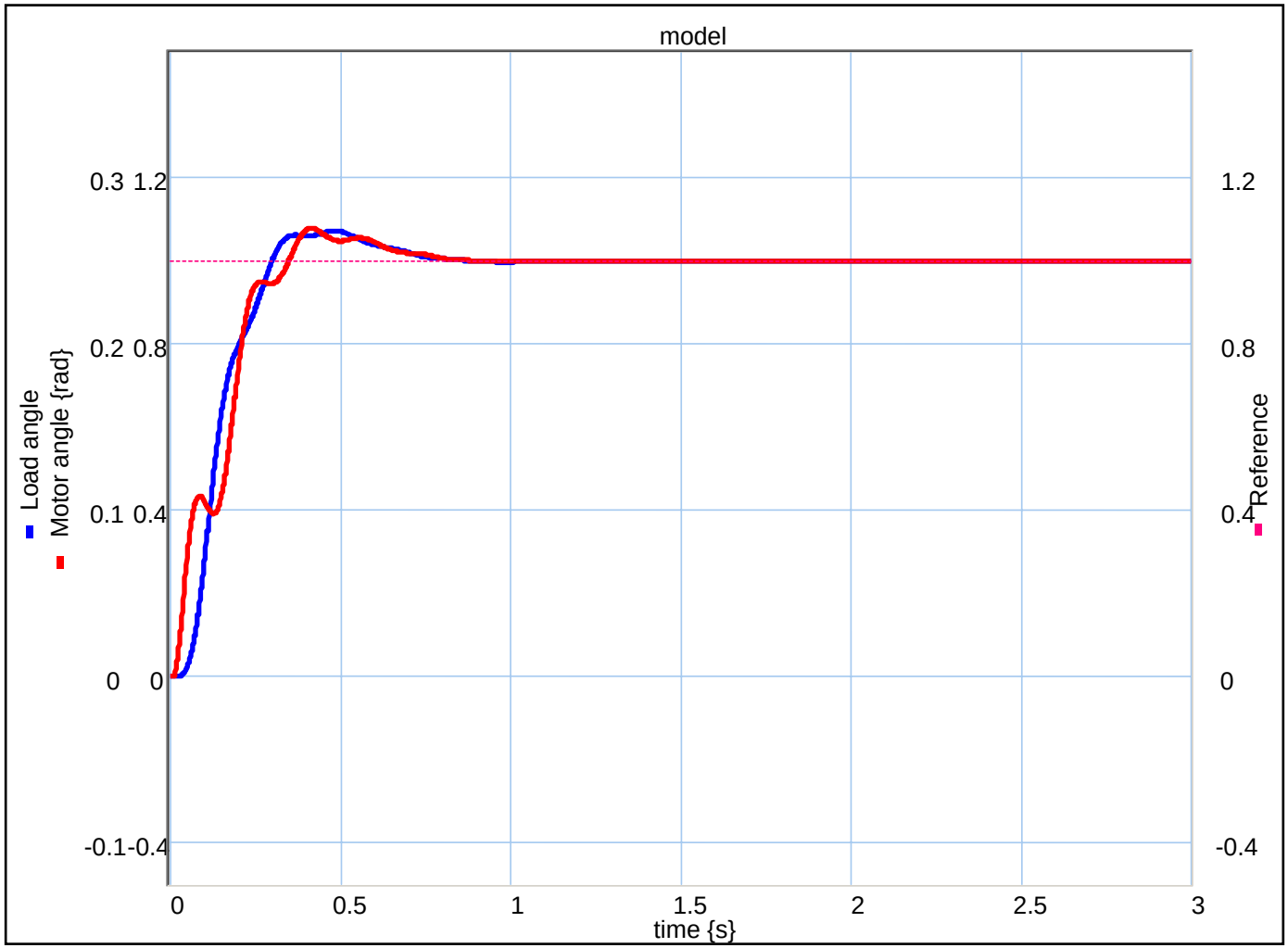
  is something completely different than

  computation step size

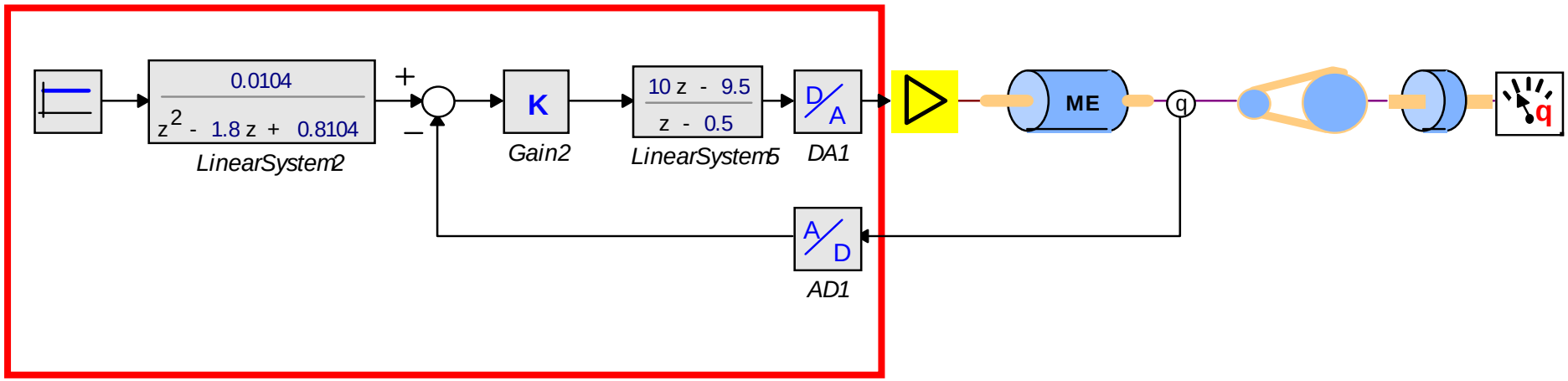
Digital lead network



20-sim demo

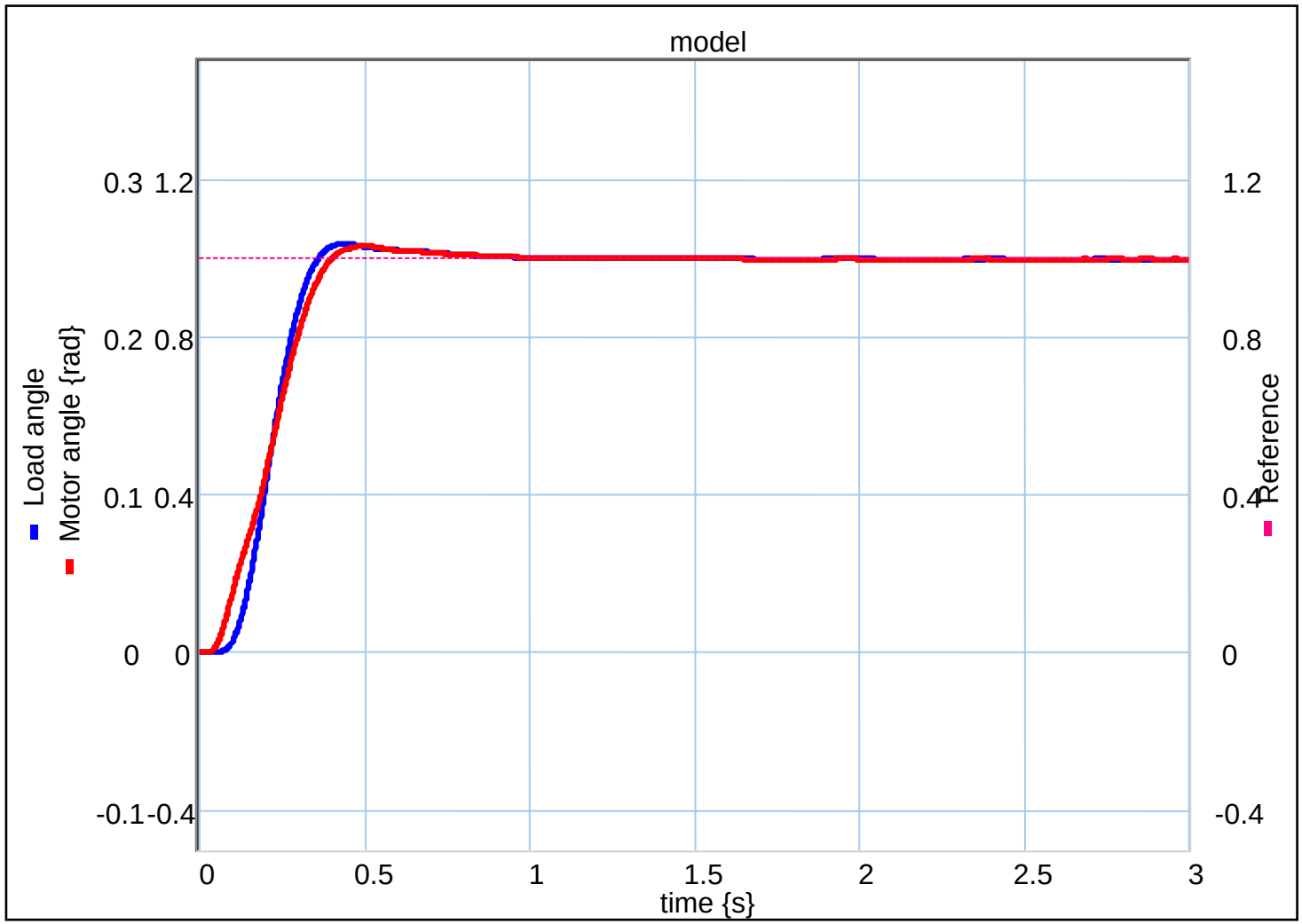


With prefilter



20-sim demo

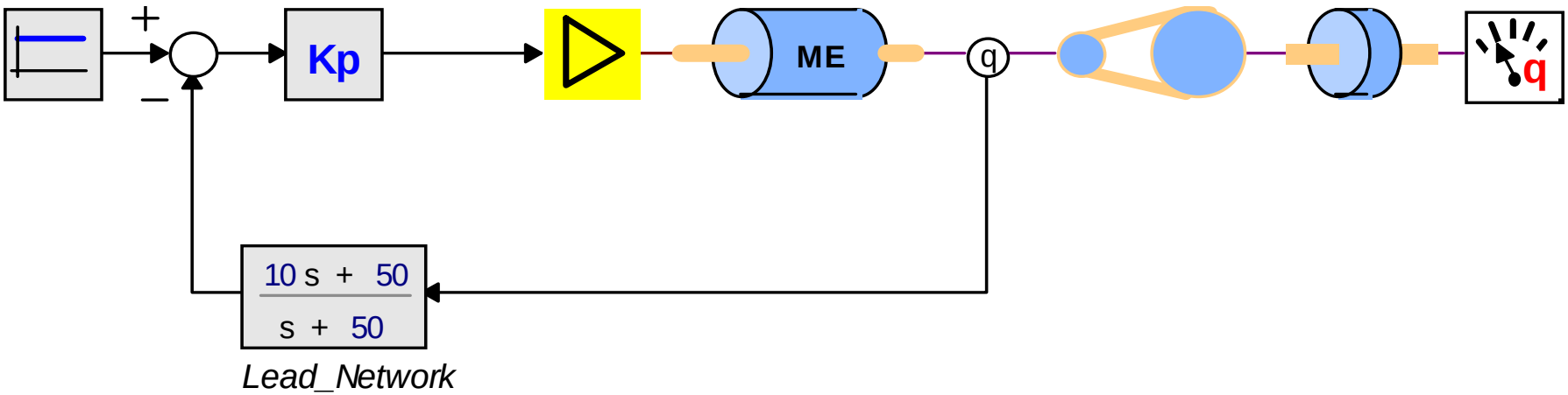
With prefilter



Lead network location

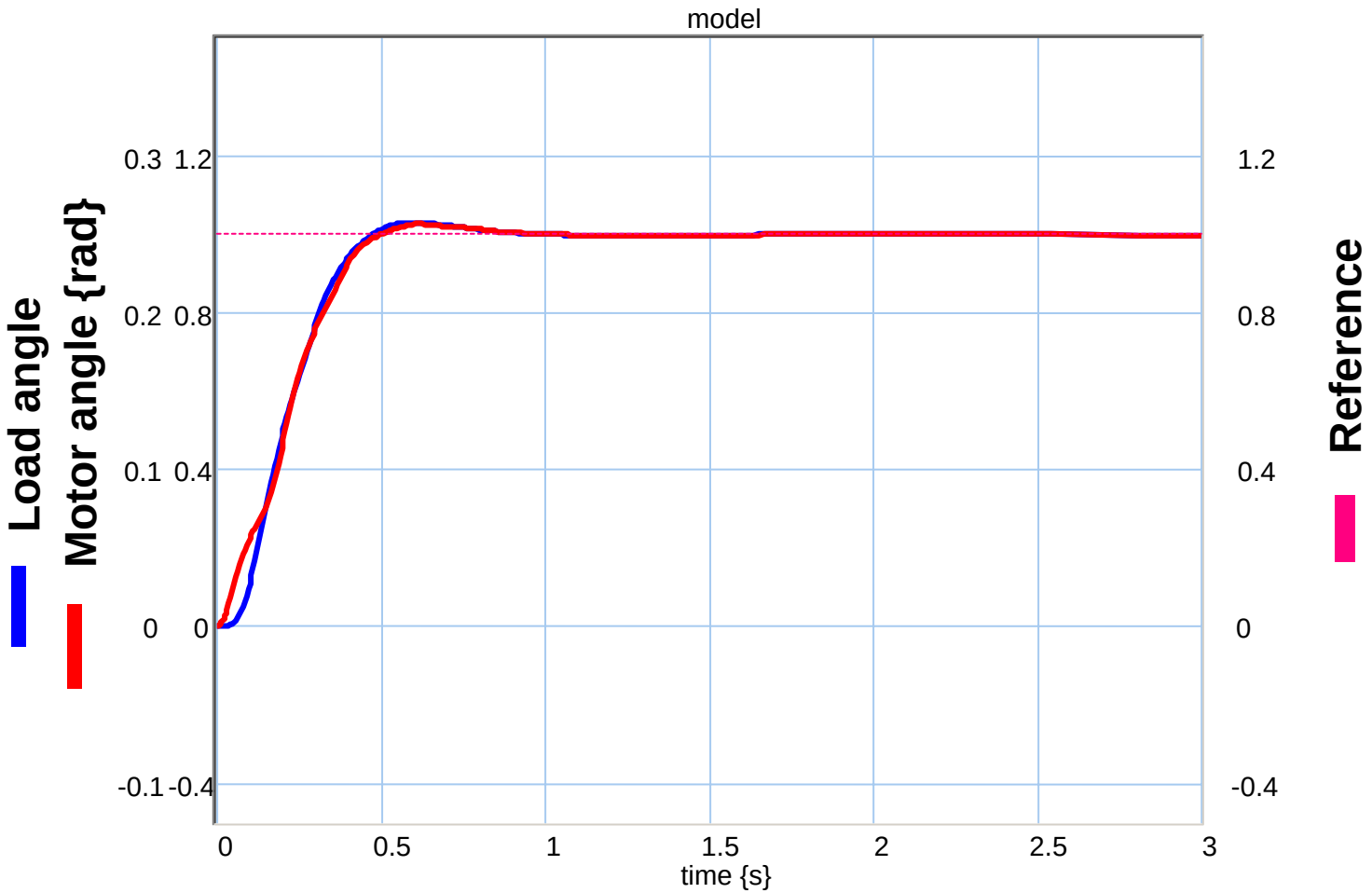
- A lead network in the feedback path gave better results

Lead network in feedback

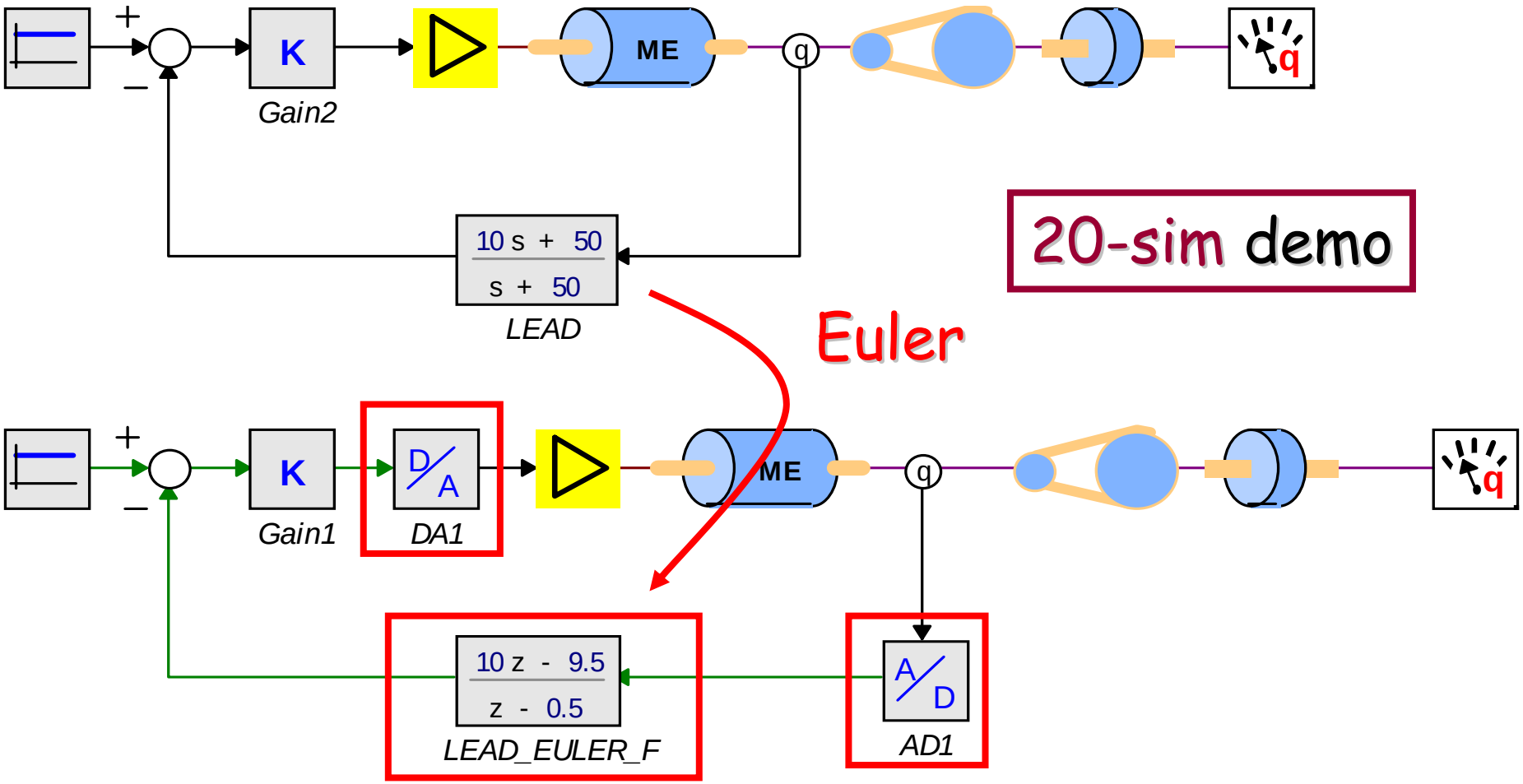


C2_lead_network_in_FB_motor_only.em

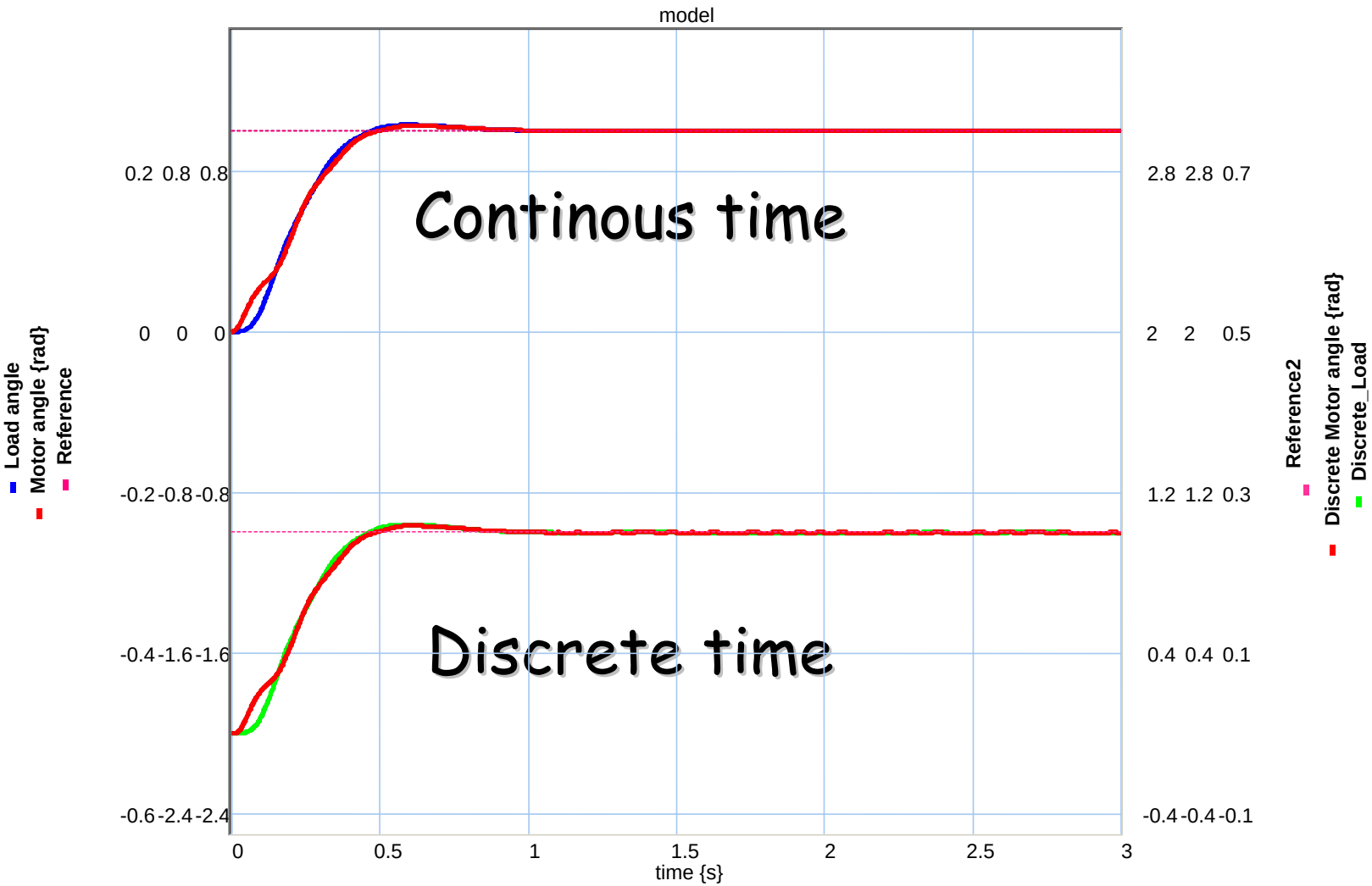
Responses



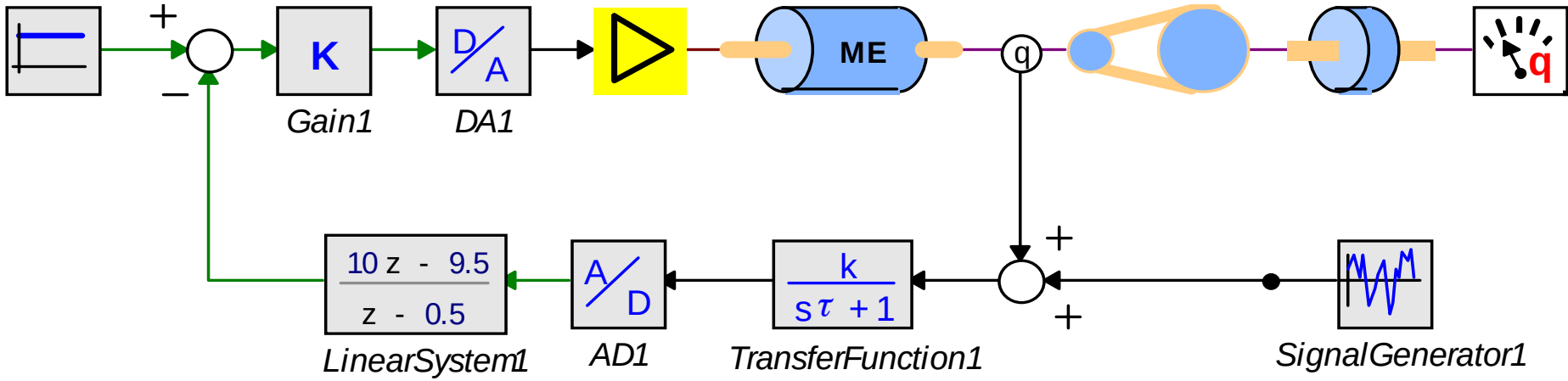
Comparison



Comparison



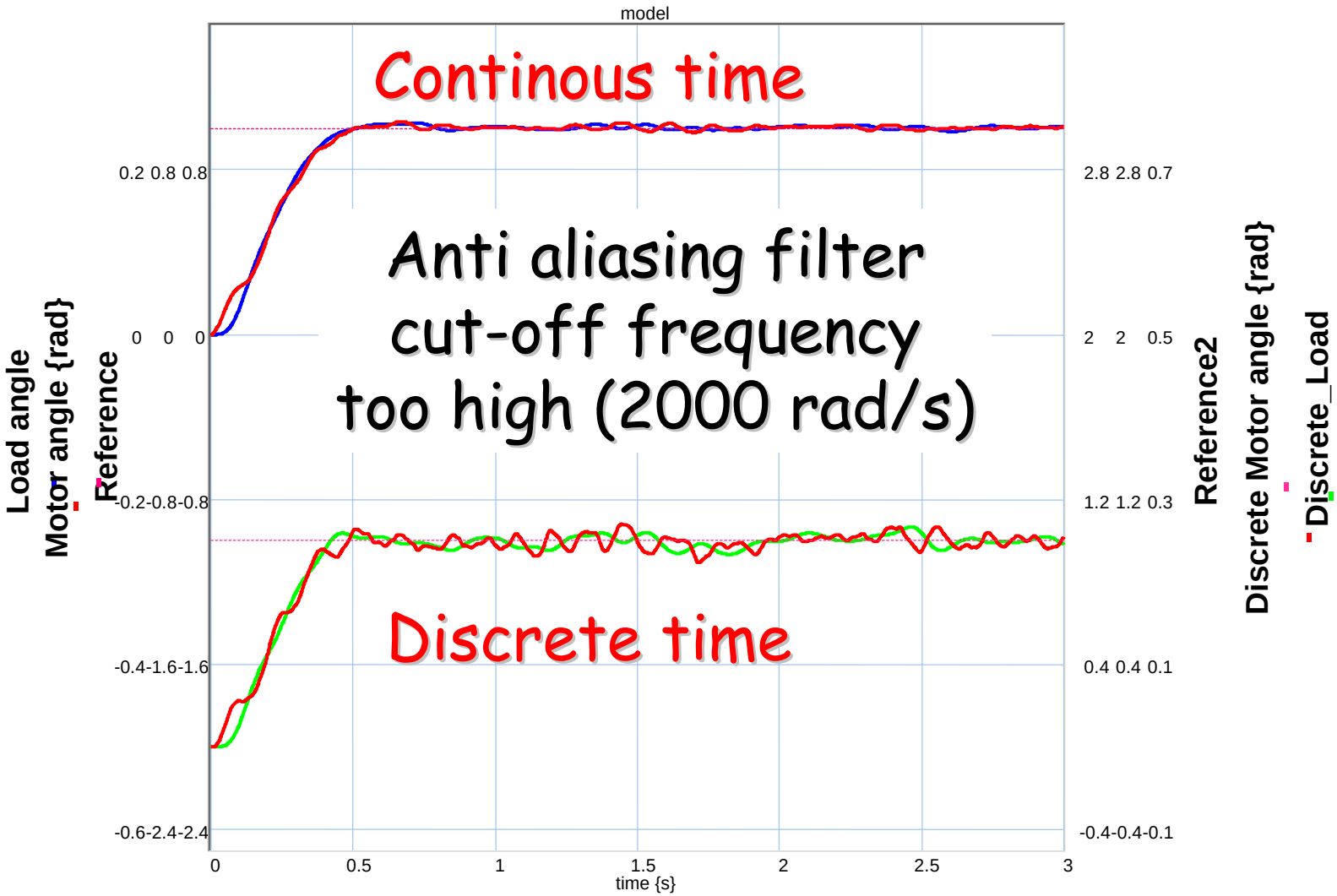
Noise / Aliasing



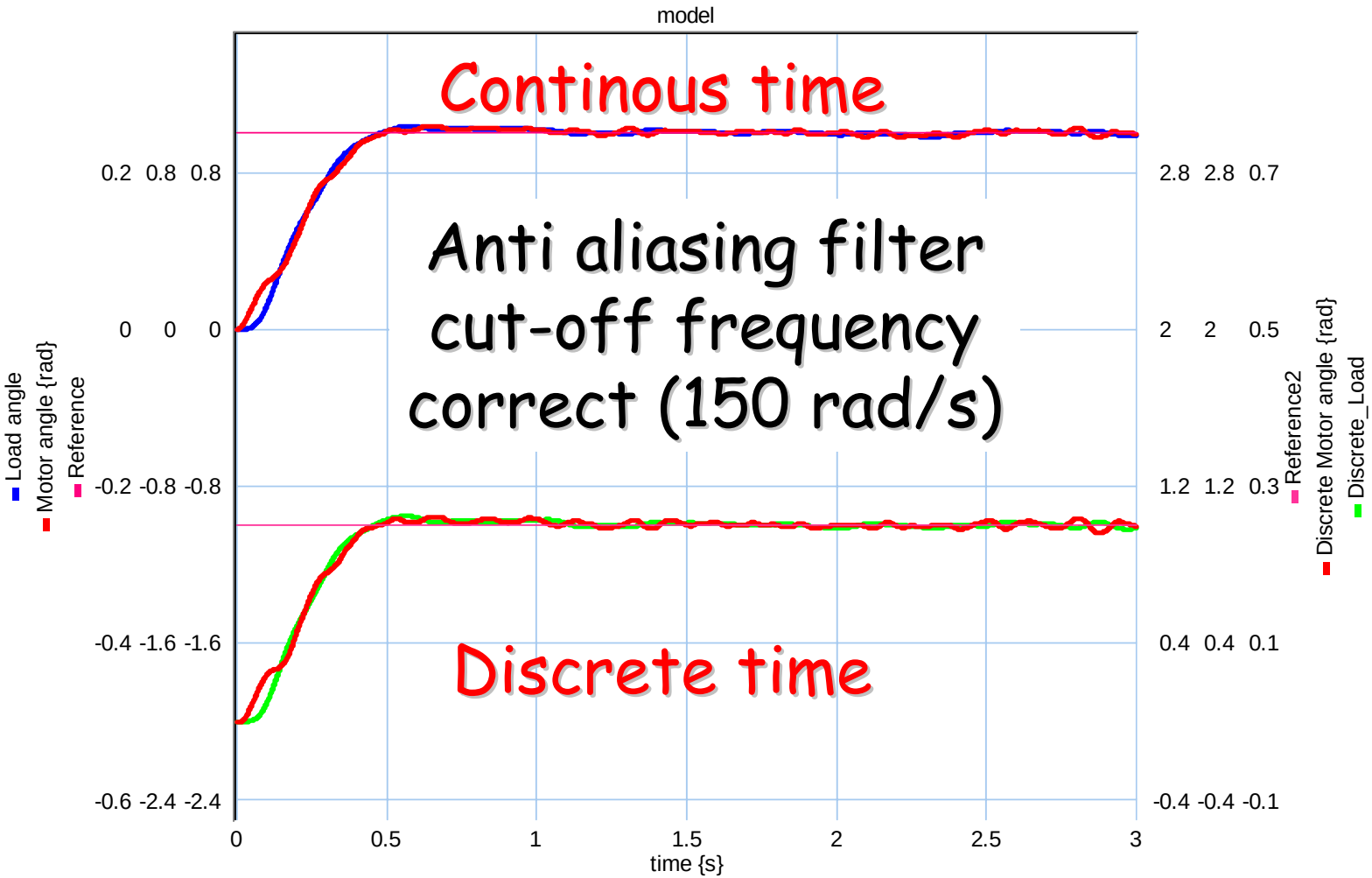
20-sim demo

Anti aliasing filter

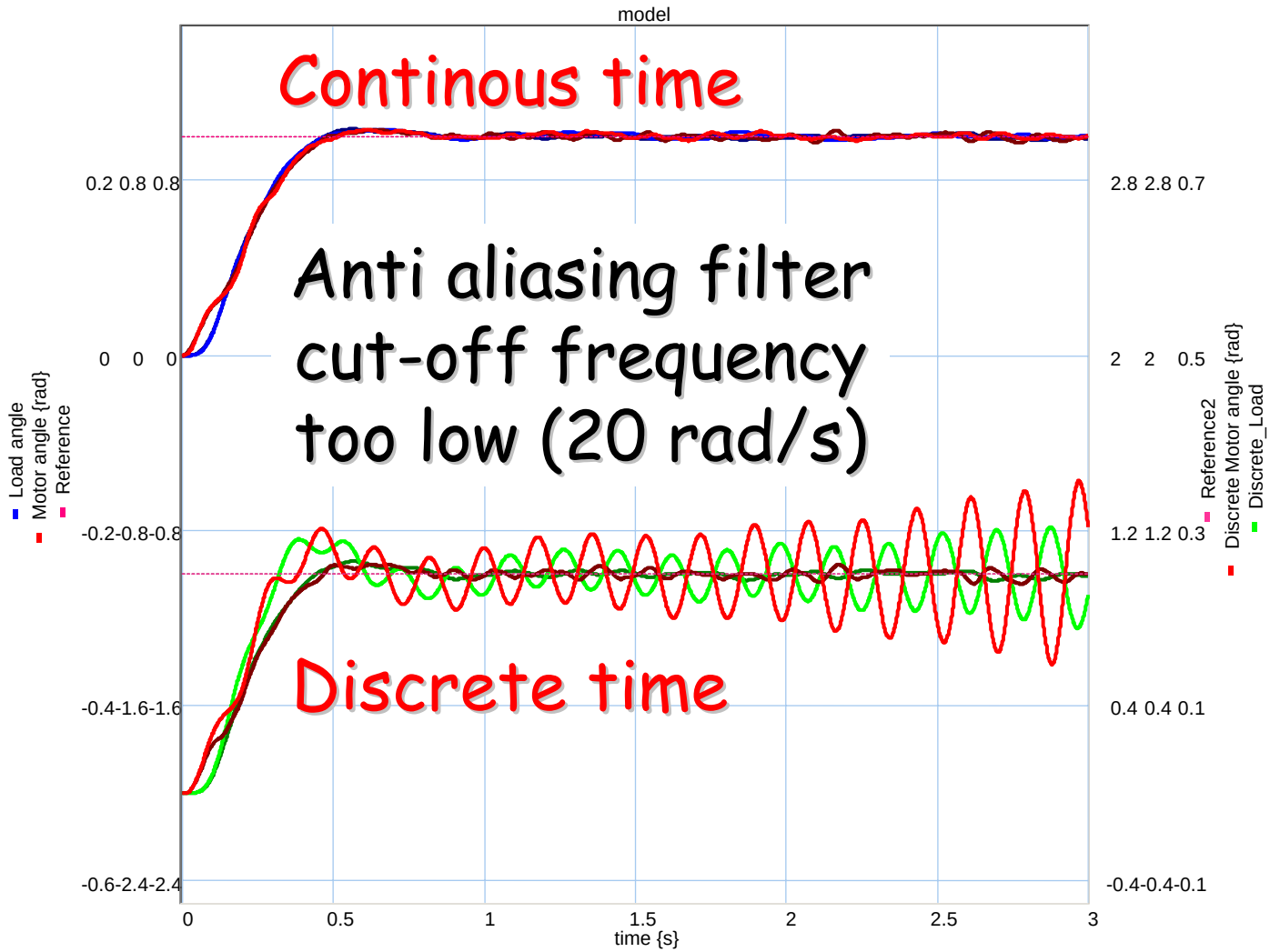
Noise / Aliasing



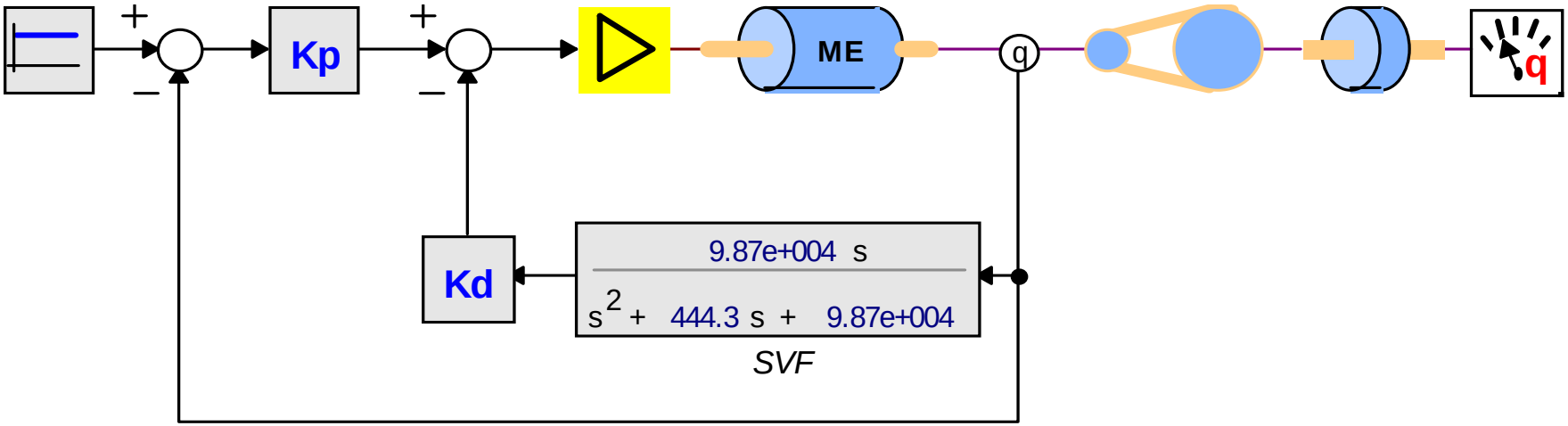
Noise / Aliasing



Noise / Aliasing



Load angle feedback & derivative of motor angle

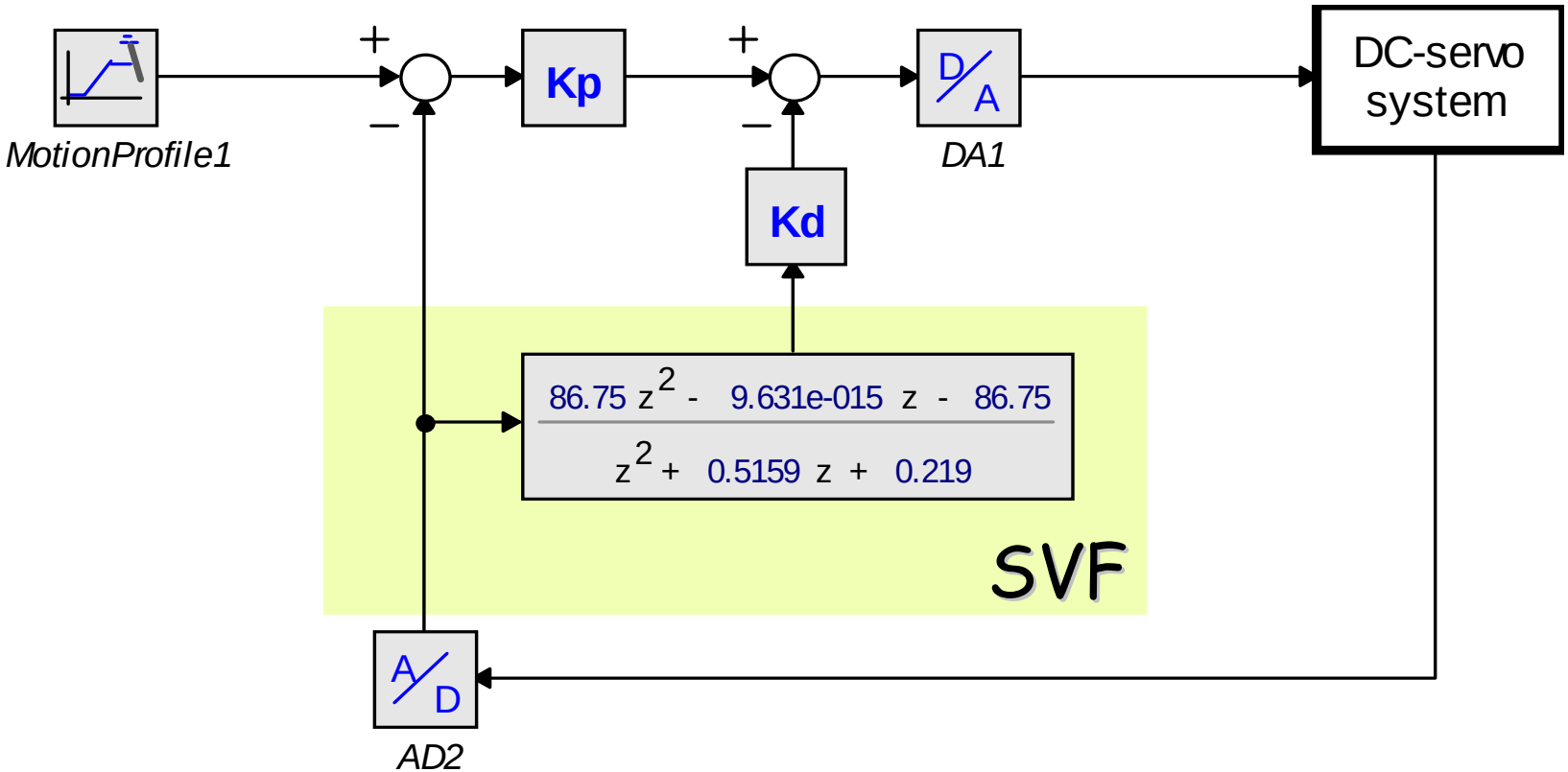


SVF: $$\frac{s\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2}$$

E1_fb_motor_only_SVF.em

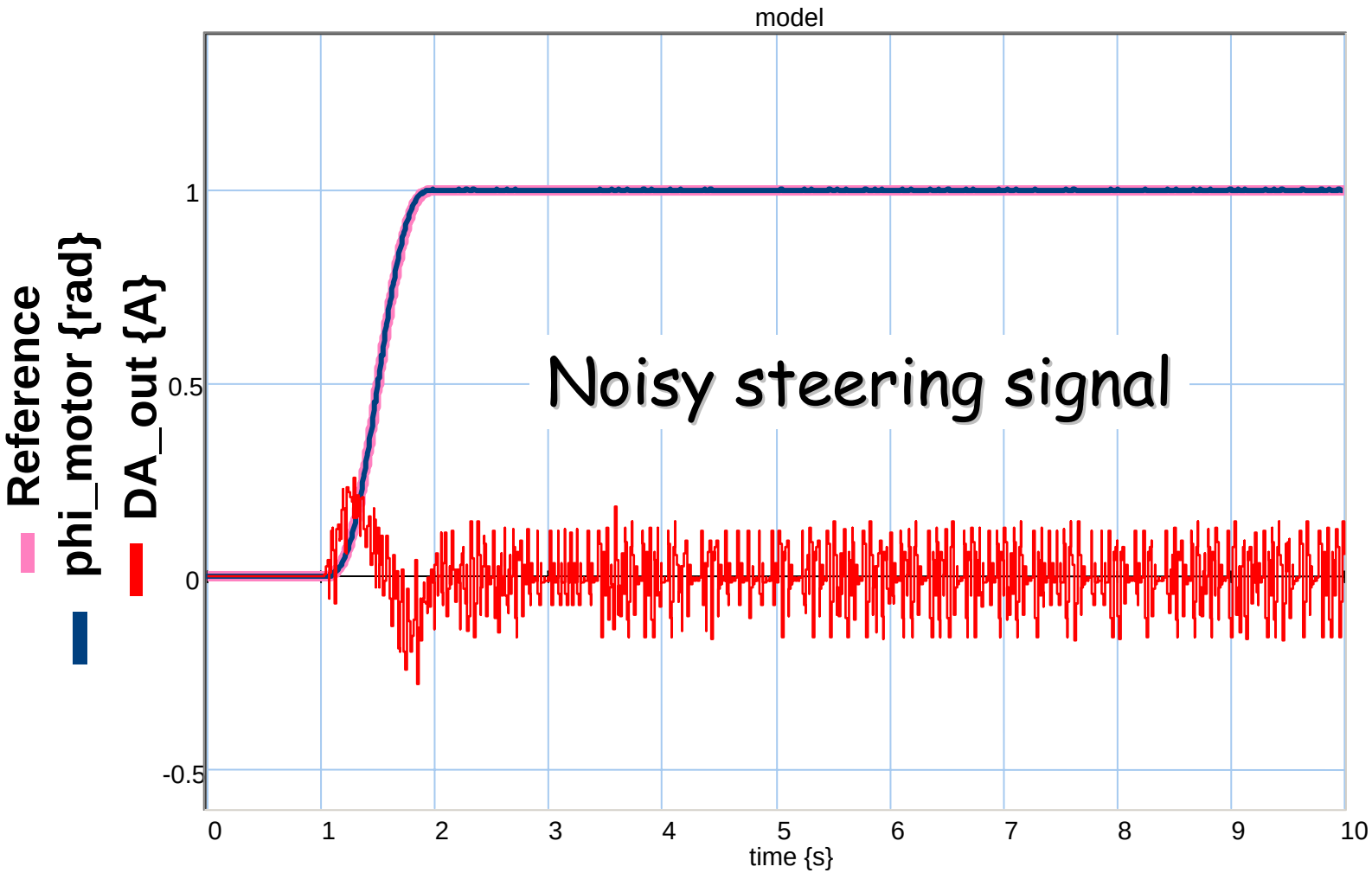
$\omega_n =$ 10 times ω resonant poles

Load angle feedback & derivative of motor angle

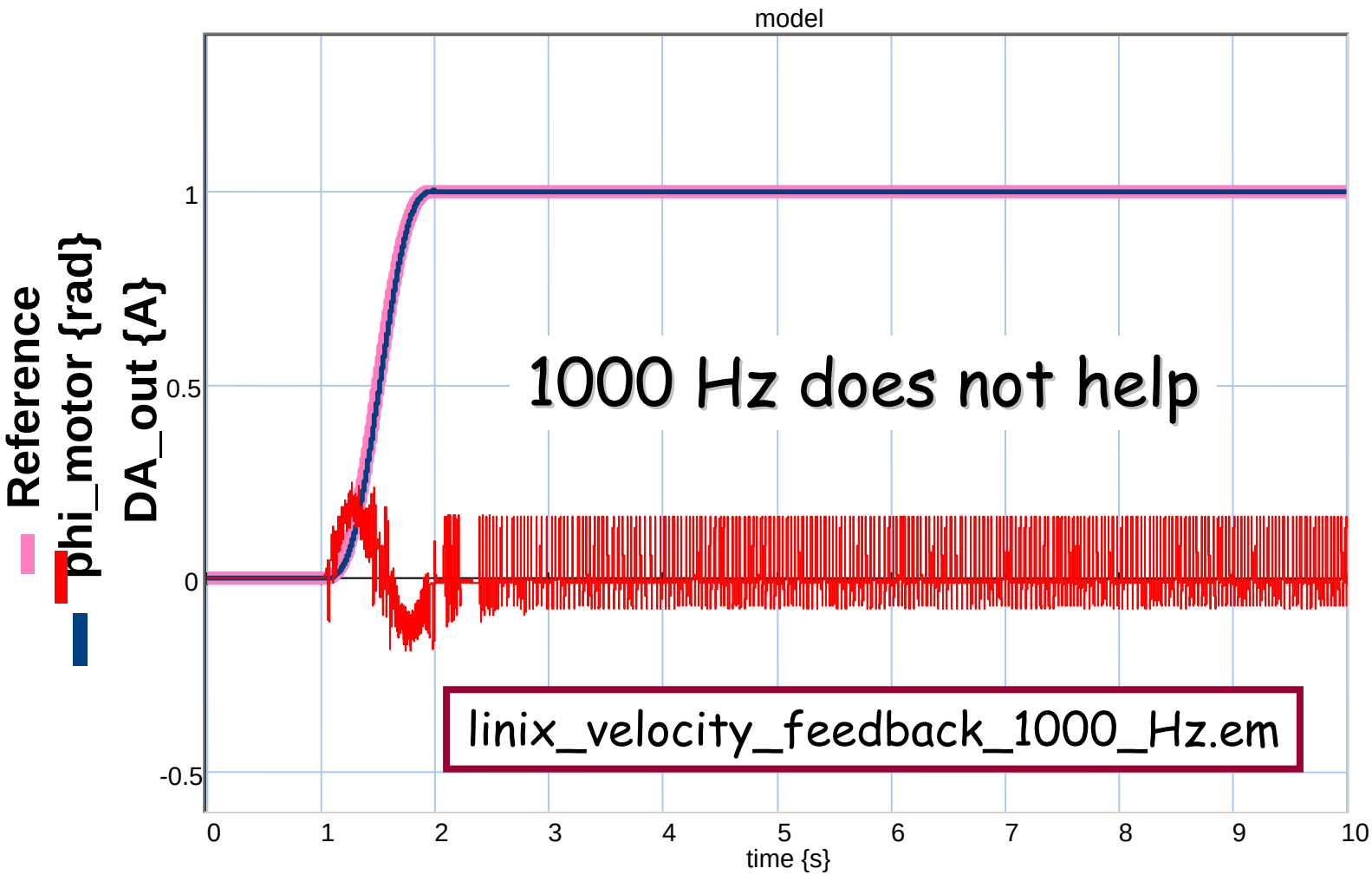


E1_fb_motor_only_SVF_DIGITAL.em

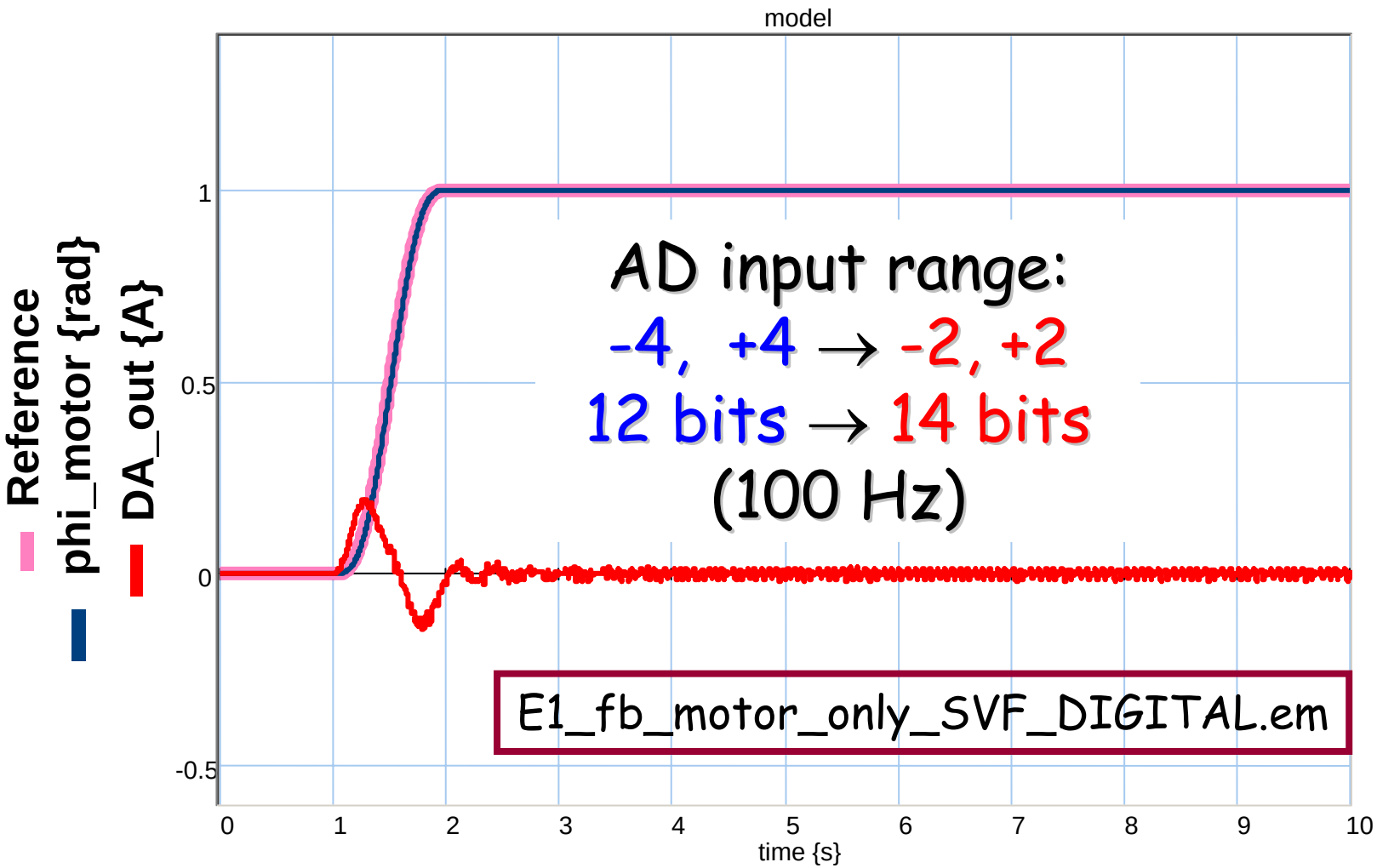
Responses (f_s = 100 Hz)



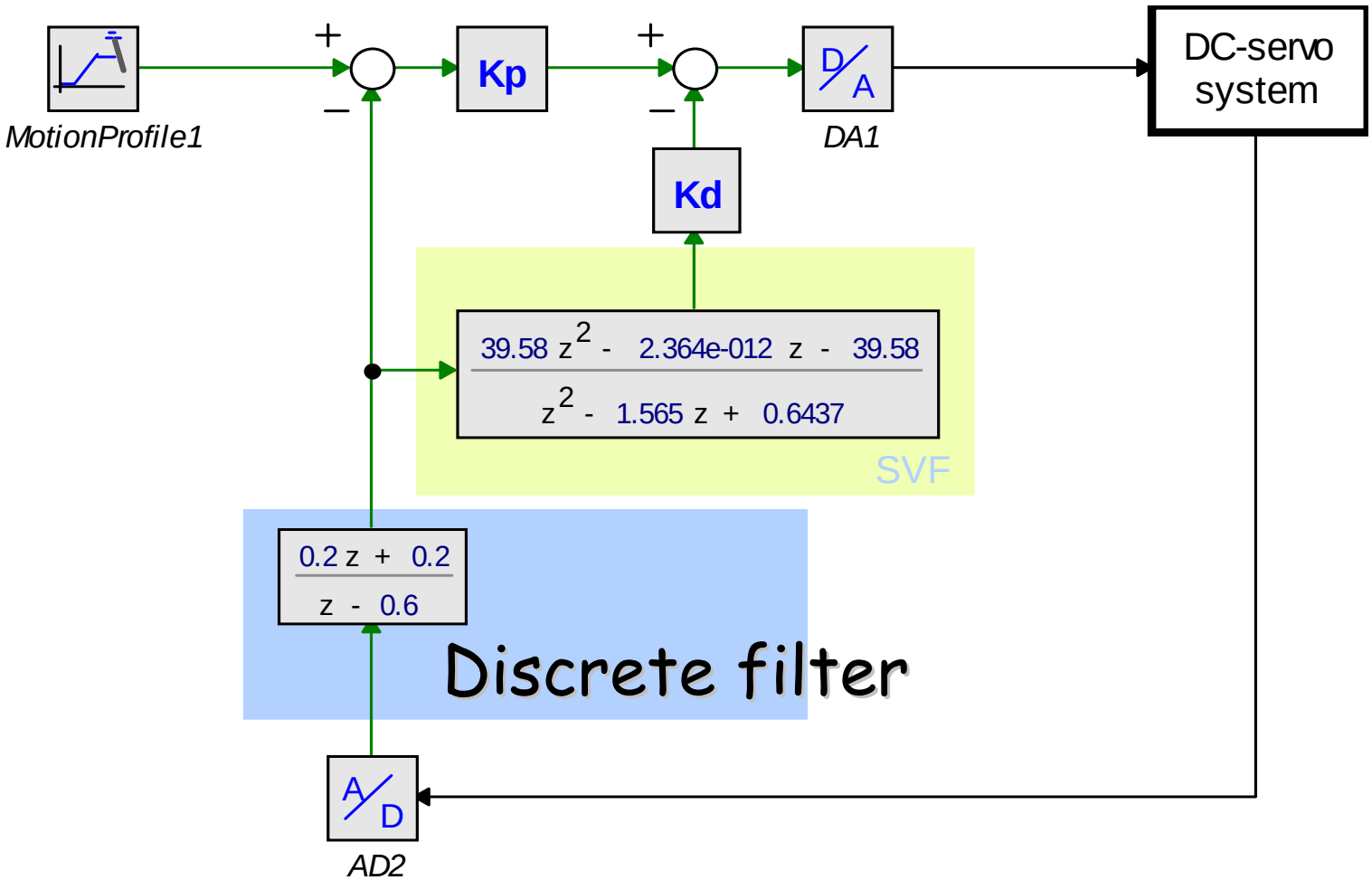
Responses (f_s = 1000 Hz)



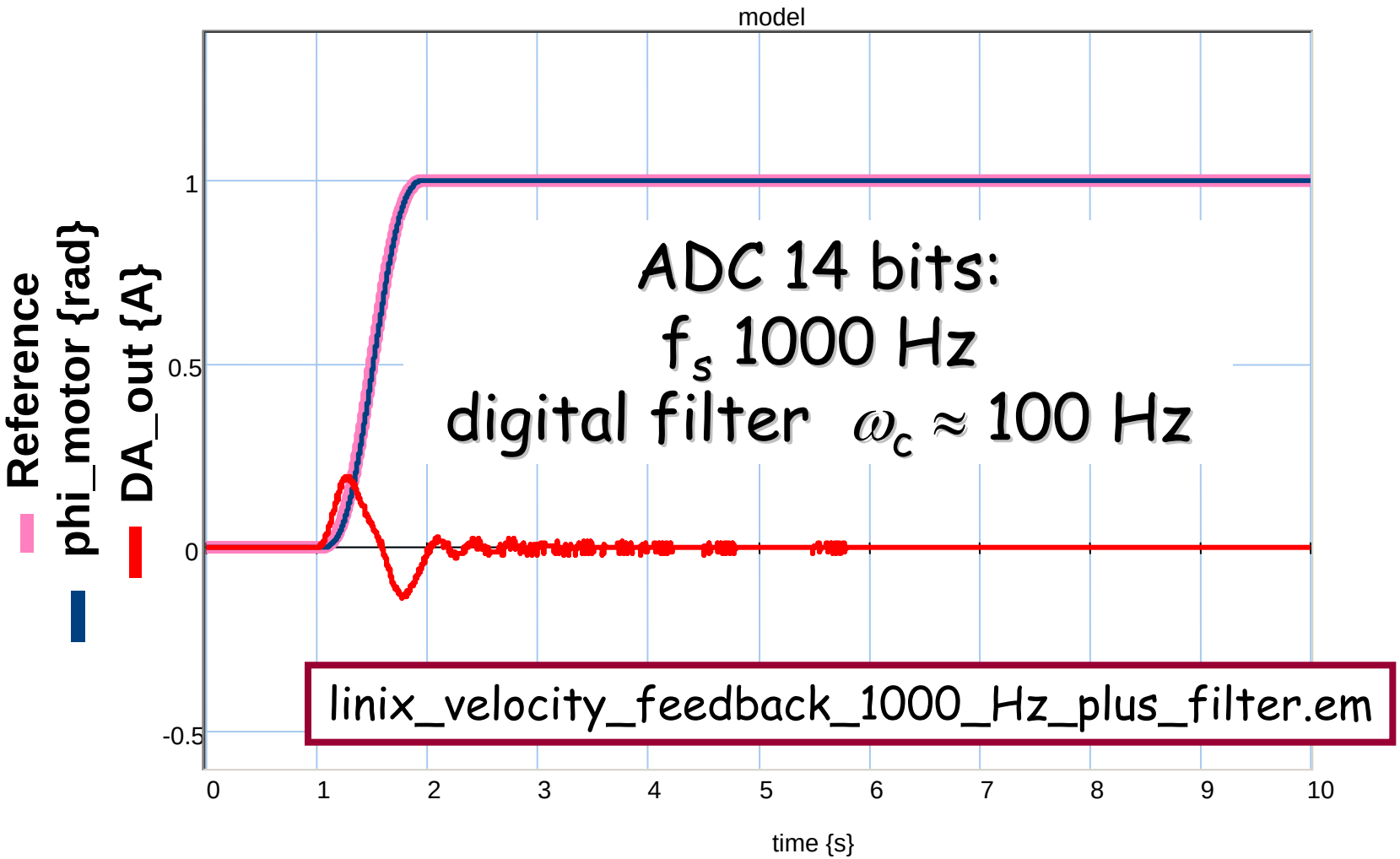
Increase resolution of ADC



Add a digital filter



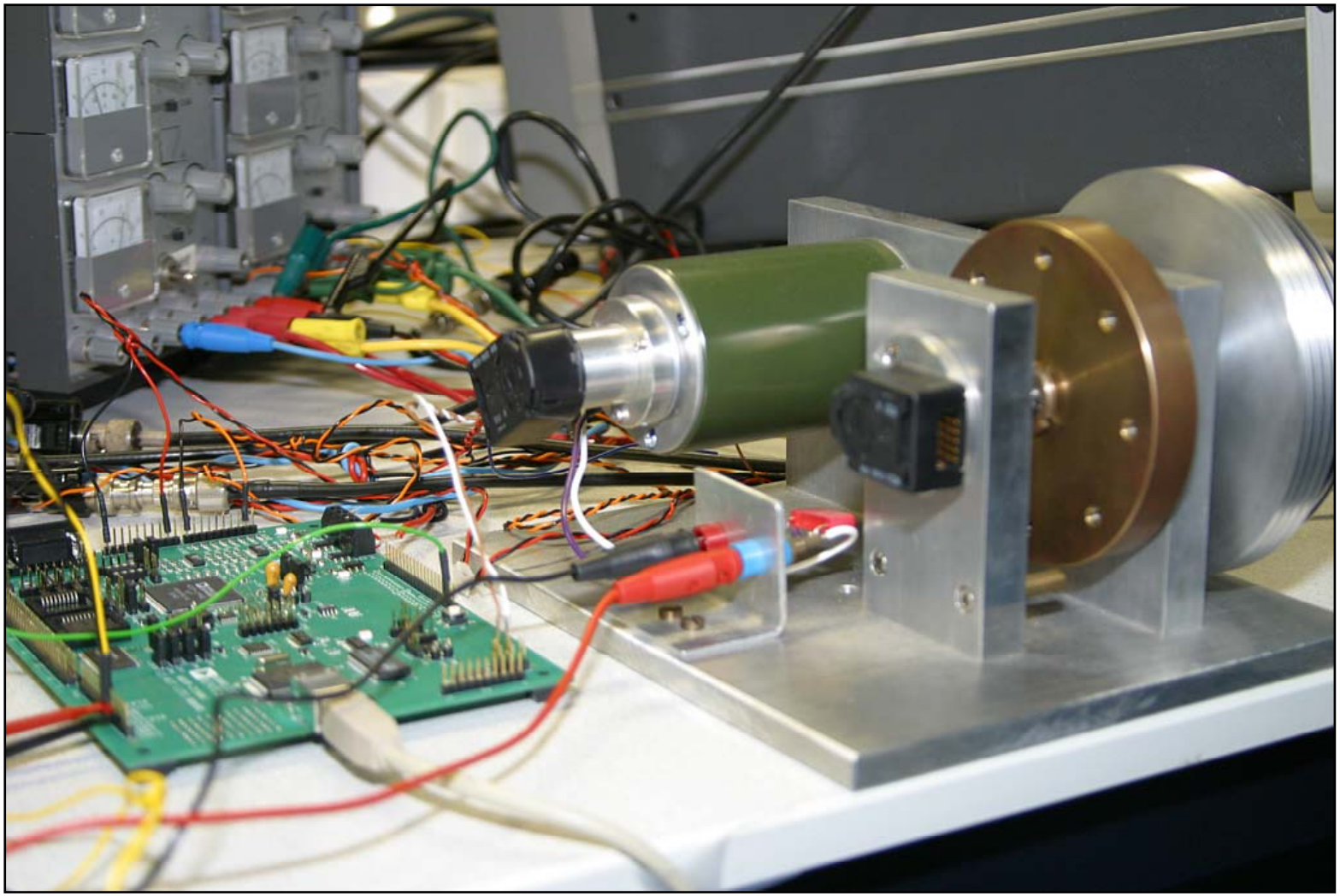
Add a digital filter



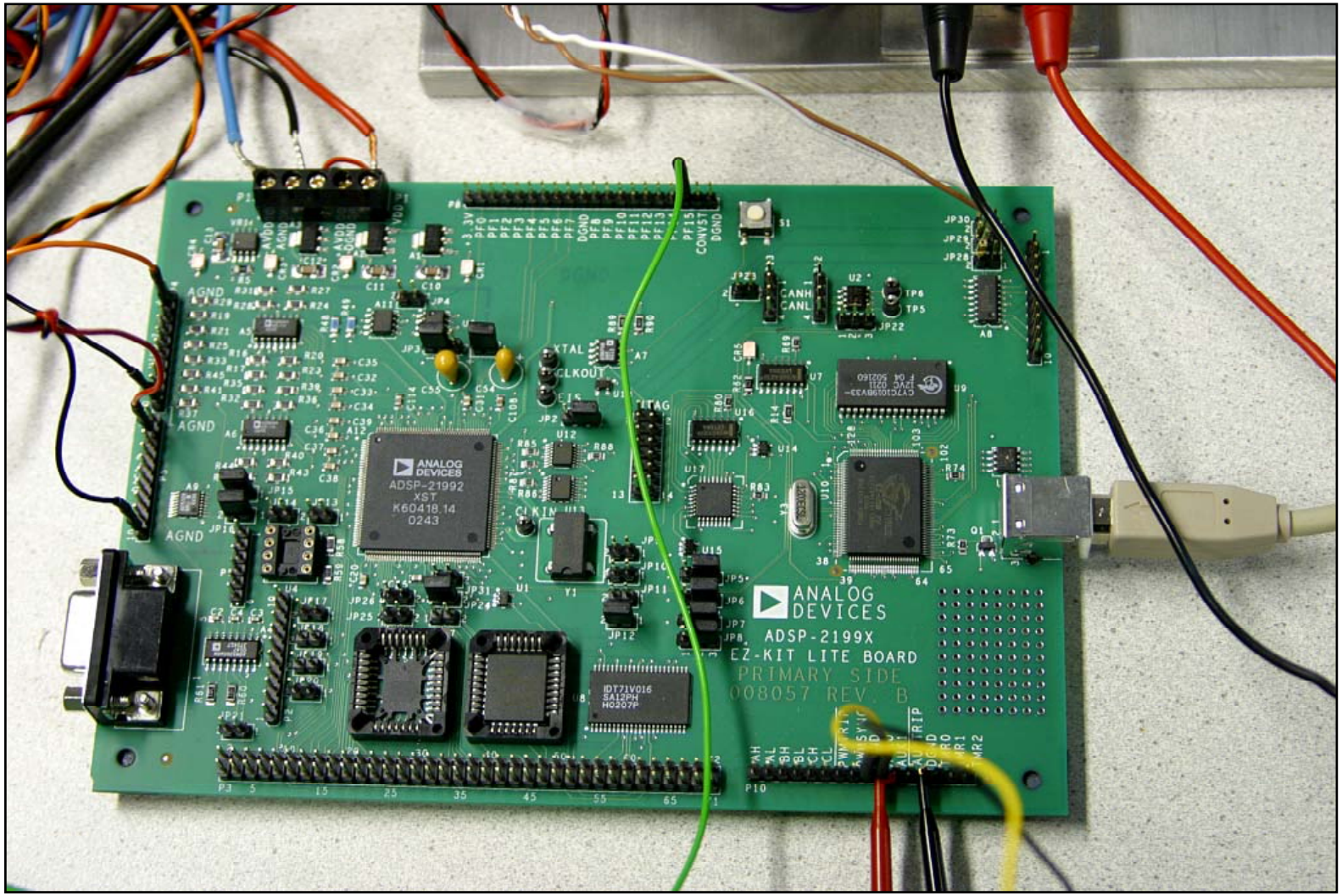
Implementation

Example:
ADSP Board of the
Mechatronics project

ADSP Board



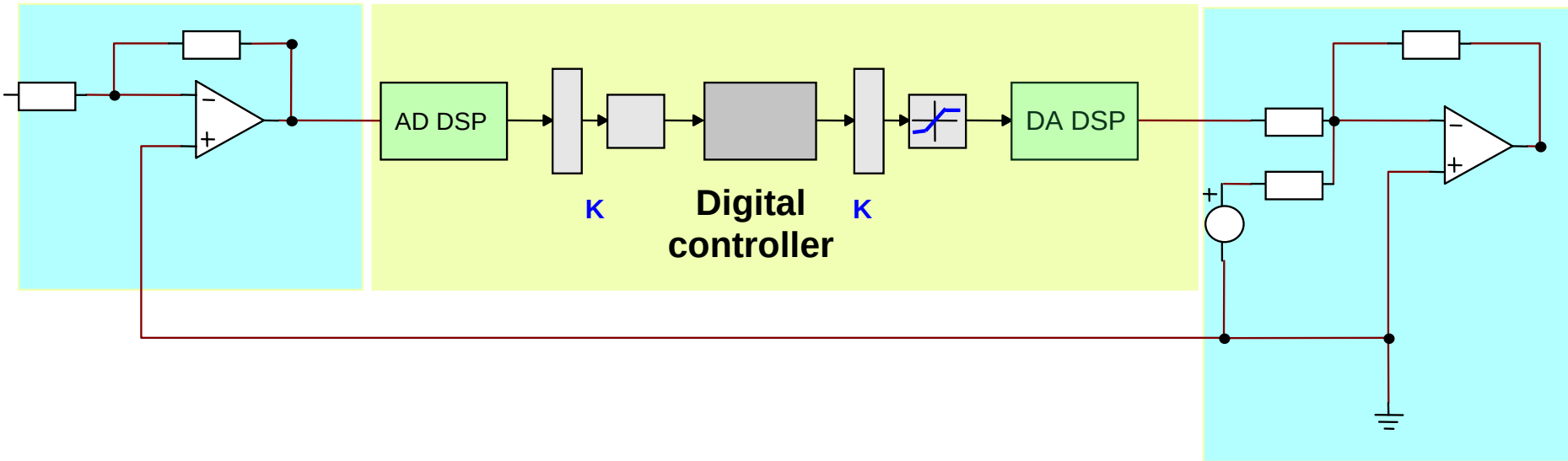
ADSP Board



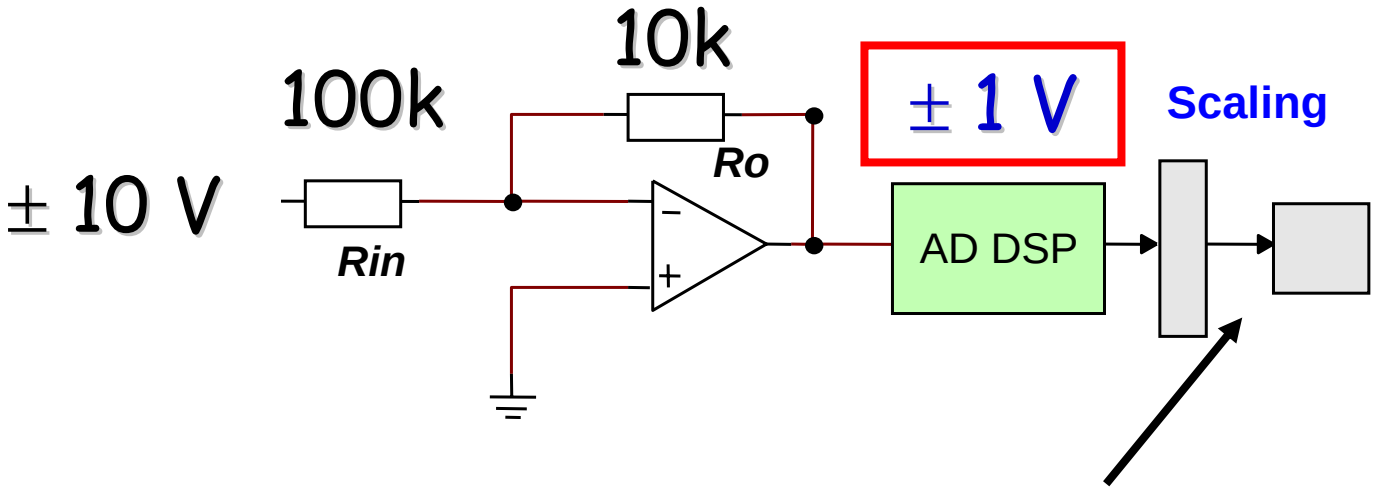
Interfaces

- AD converters 14 bits $\pm$ 1 V
- DA converters 12 bits 0 - 2 V
- Encoder input
- Pulse width modulated output
- Digital I/O

DSP board + interface AD/DA



AD converter



14 bits integer $= 2^{14} - 1 = 16383$

-10V = 0

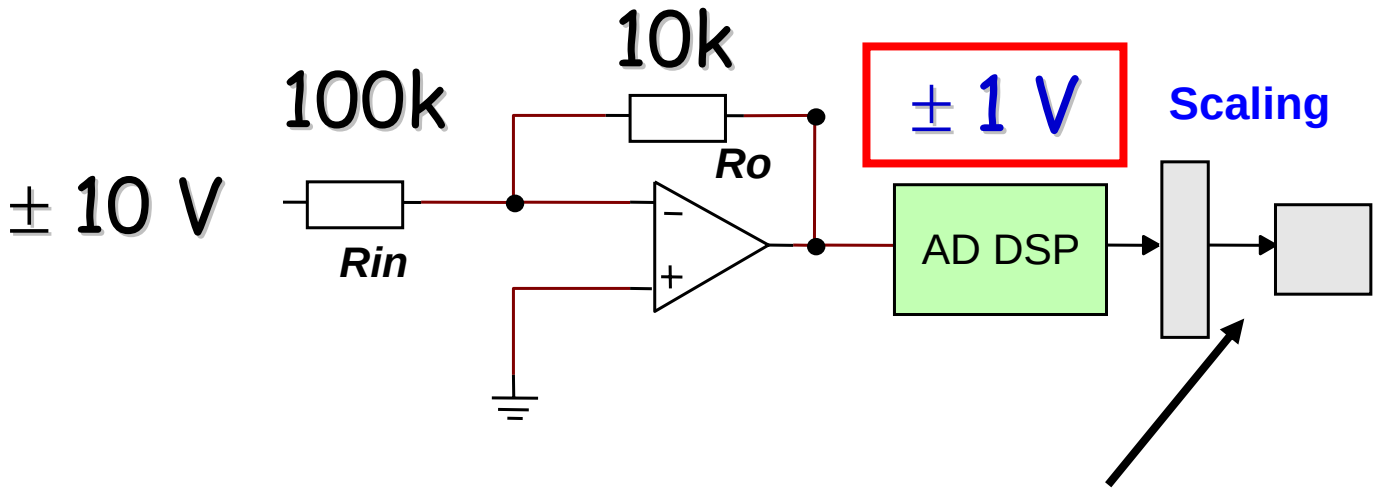
0V = 8191

10V = 16383

internal: $\frac{\#-8191}{8191} * 10$

In driver?

AD converter (alternative)



$$16 \text{ bits integer} = 2^{16} - 1 = 65535$$

-10V = 0

0V = 32767

10V = 65535

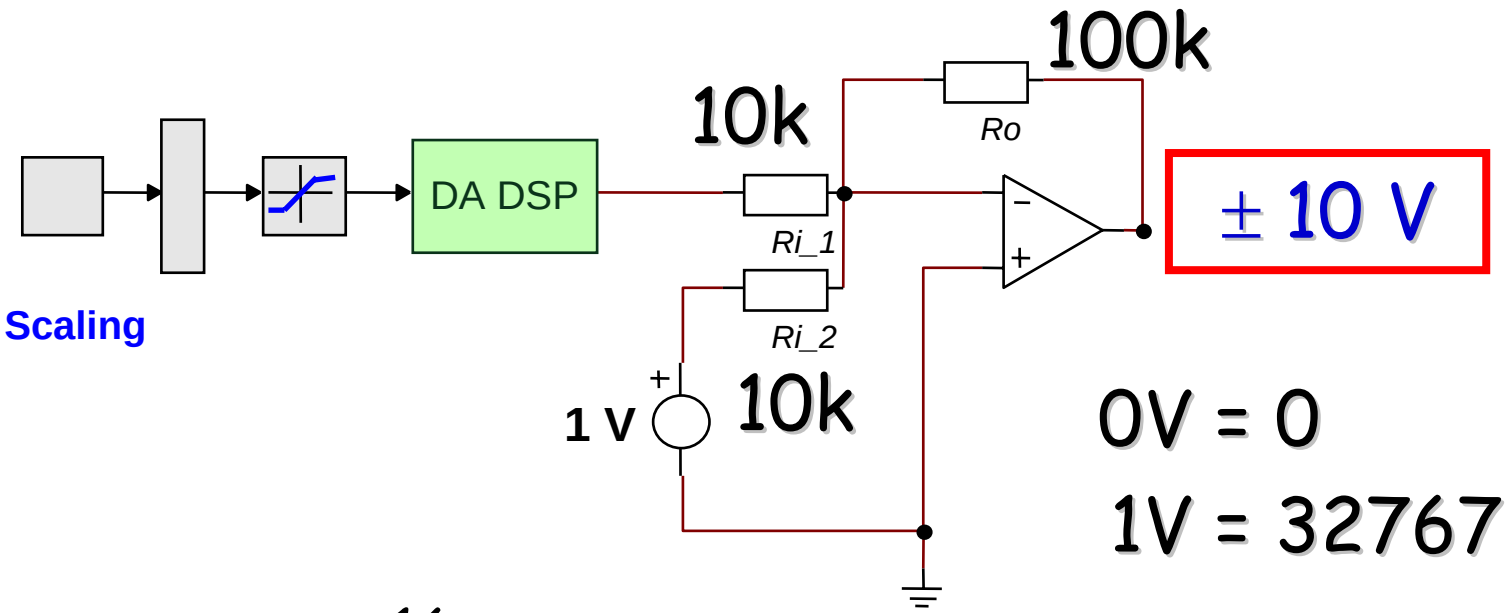
$$\text{internal: } \frac{\#-32767}{32767} * 10$$

In driver?

but 14 bits accuracy !

DA converter (alternative)

Control Engineering
University of Twente



$$16 \text{ bits integer} = 2^{16} - 1 = 65535$$

0V = 0

1V = 32767

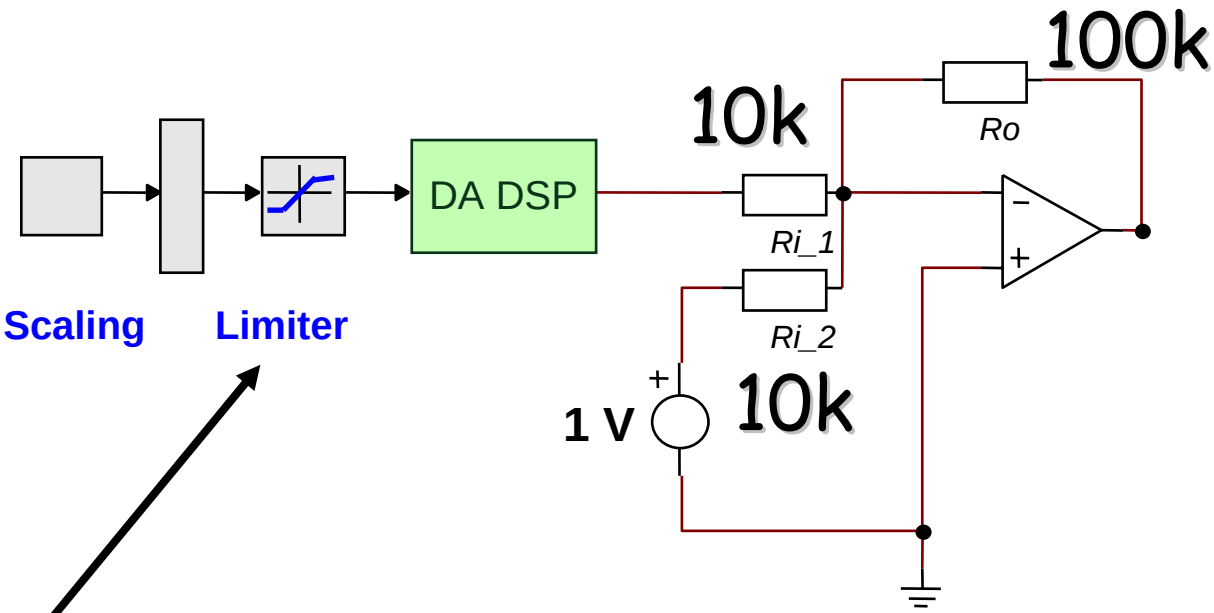
2V = 65535

$$\#: \left(\frac{\text{internal}}{10} 32767 \right) + 32767$$

In driver?

but 12 bits accuracy !

DA converter



needed to prevent overflow

$$= 2^{16} + x = x$$

Encoder interface

- Counts pulses from incremental encoder
- Measures angles
- Scaling !
  - Depends on the hardware used

Pulse width modulation

- Enables that inexpensive and efficient power amplifiers be used

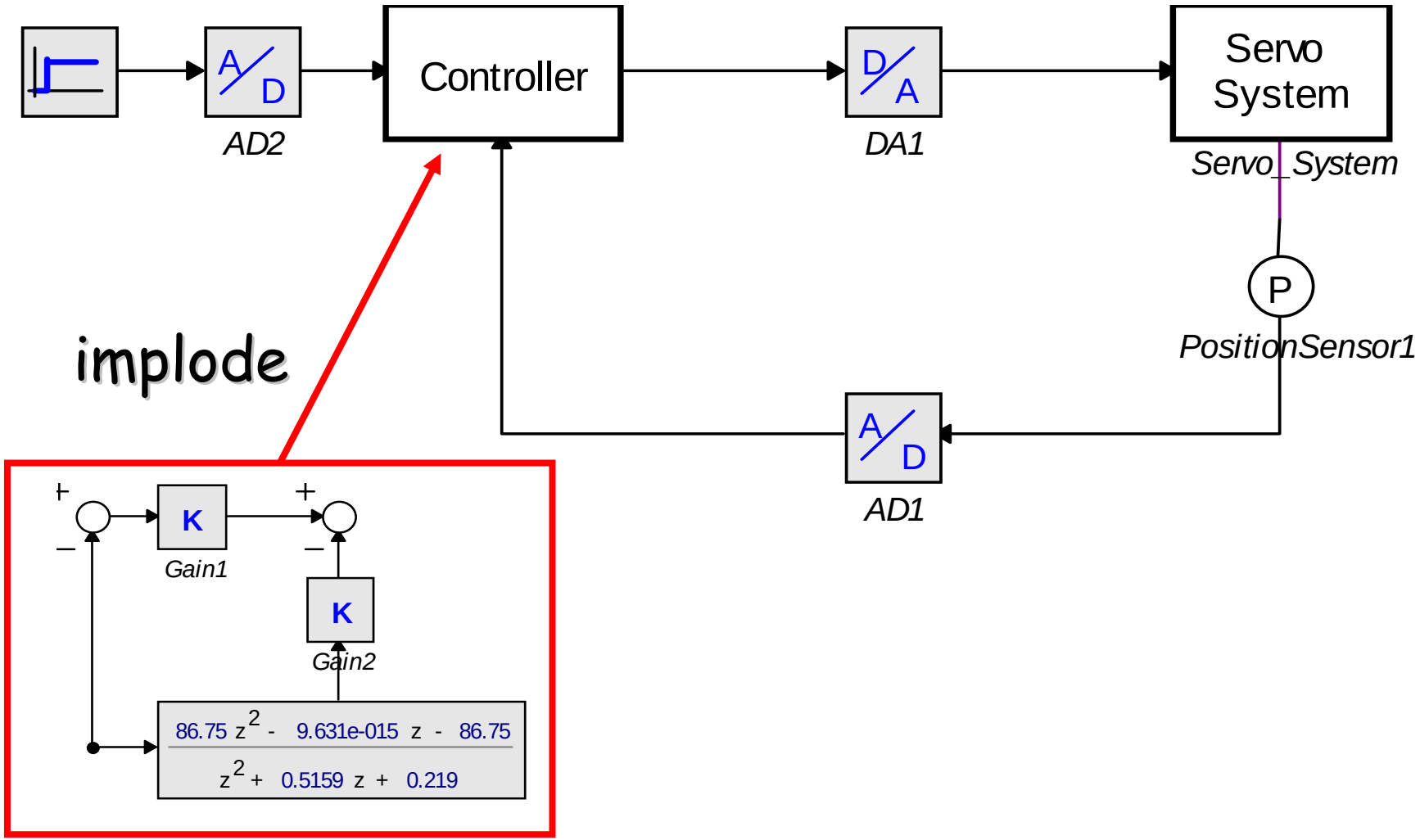
Design cycle

- Design process and controller and test control system in simulation
  - You could start with a relatively simple model and a continuous-time controller
- Make the process model more realistic
- Design a digital controller
- Simulate
- Realise process and controller

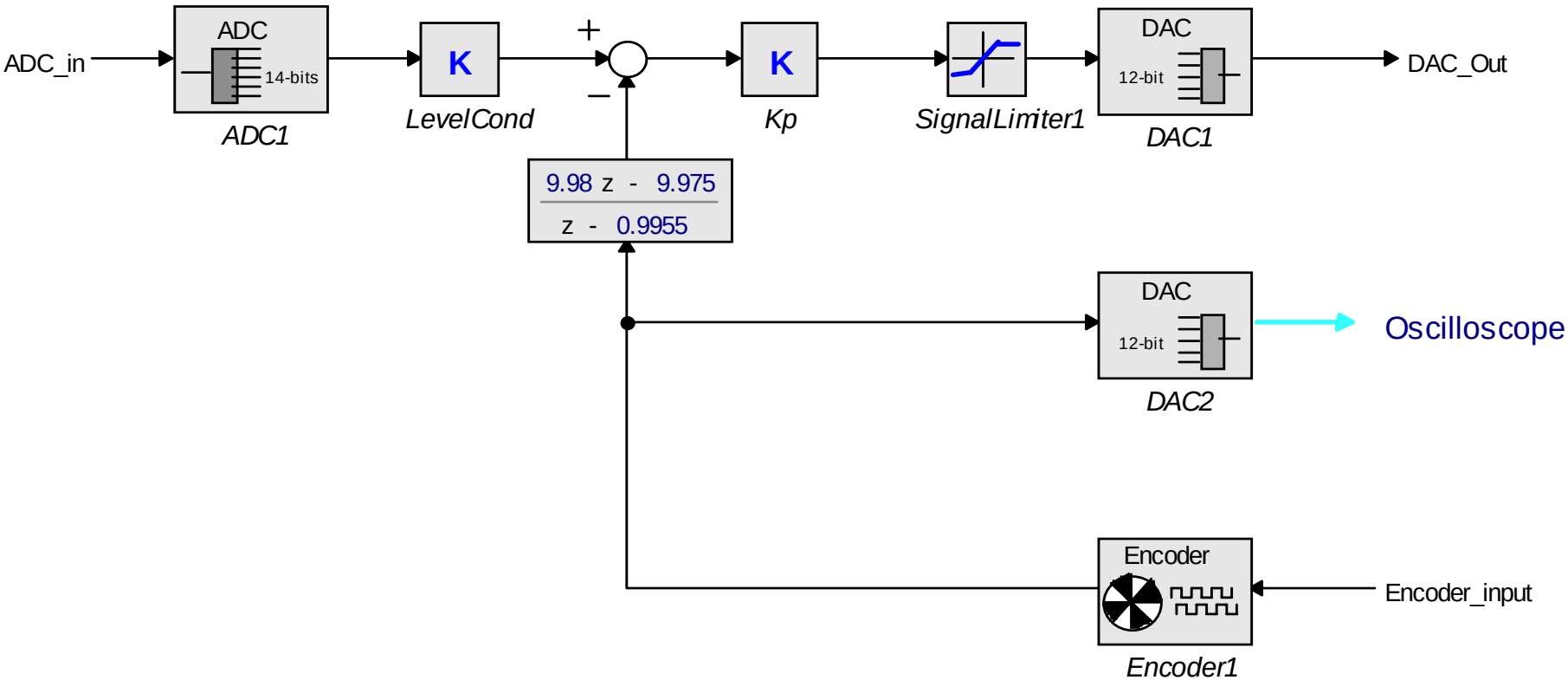
C-code generation

- C-code can be generated automatically
- Convert (implode) the controller into one single submodel
- Include hardware dependent elements
- Generate C-code from the simulator

Controller as a submodel



Inside the submodel



Hint

- Models from the Linear Systems editor can not be directly used in the controller board
- Matrix models
- Use Export to Scalar Equations

Example


```
// 20-sim Linear System Editor
// tf h 0.01
parameters
        real hidden A [2, 2] = [0.0, -0.9951; 1.0, 1.995];
        real hidden B [2, 1] = [-0.0001; 0.0];
        real hidden C [1, 2] = [0.0, -1.0];
        real hidden x0 [2] = [0.0; 0.0];
variables
        real x [2];
equations
        next (x, x0) = A * x + B * u;
        y = C * x;
```

Matrix equations of Linear Systems Editor

Example


```
variables
        real x1;
        real x2;
        real x3;
equations
        next (x1) = x2;
        next (x2) = x3;
        x3 = u + 1.995 * x2 - 0.9951 * x1;
        y = + 0.0001 * x1 + 0 * x2 + 0 * x3;
```

After conversion into Scalar Equations

Conclusions

- Digital implementation has been demonstrated
- More advanced topics such as
  - multivariable systems
  - state estimation and optimisation
  - more on computer controlled systems

follow in course: Digital Control Systems

Finally

- Success with the computer exercises
- Success with the mechatronics project

- Success with the exam
  - And don't under estimate it…