Example: Implementation of a Digital Low-pass Filter

by Dipl.-Ing. (FH) Marcus Maresch, Dynamic Systems and Control Laboratory

Introduction

The goal of this tutorial is the replacement of an existing analog filter circuit by an appropriate digital filter with nearly identical filter characteristics. With this example a possible procedure is shortly described and applied for a 1st-order low-pass filter.

Z-Transform of CT Transfer Function by the Tustin's formula

For the implementation of an analog filter on digital hardware a transform of the CT transfer function into the DT domain is required. Known from the numerical integration, the so-called z-Transform of the continuous-time transfer function can be easily derived. If we use a trapezoid area a very exact approximation can be achieved. Let's consider a simple integrator with the transfer function

$$G(s)=\frac{Y(s)}{X(s)}=\frac{1}{s}$$ with $X(s)$ as input and $Y(s)$ as output.

For the discrete time description of integrators output y[n] the integral can approximated by a sum of trapezoids, as shown in Figure 1.

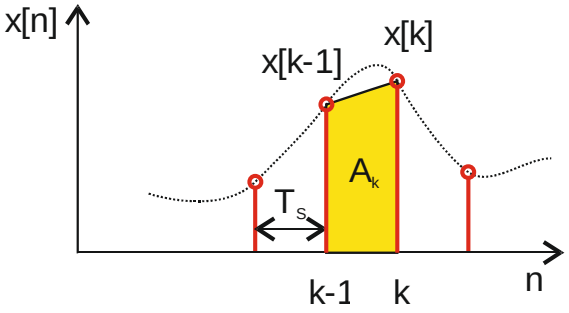

Figure 1: Approximation by a trapeziod area A_k

The increase of area A_k in the interval can be expressed as

$$A_k=\frac{1}{2}T_S\left(x[k-1]+x[k]\right) \qquad \text{with sampling period } T_S=\frac{1}{f_S}$$

Adding the new trapezoid area A_k to the previous sum y[k-1] (last sample) yields to the output signal y[k] and the difference equation of the system:

$$y[k]=y[k-1]+\frac{T_S}{2}\left(x[k]+x[k-1]\right)$$

Taking then the z-Transform of the difference equation above results the z-domain representation of the integrator. The delayed signals y[k-1], x[k-1] can be transformed by the Right Time Shift Property (causal system) of the z-Transform, as stated below.

$$x[n-i] \leftrightarrow X(z) \cdot z^{-i}$$

Appling the Linearity Property gives the z-Transform of a integrator:

$$Y(z) = Y(z) \cdot z^{-1} + \frac{T_S}{2} X(z) + \frac{T_S}{2} X(z) \cdot z^{-1}$$

Sorting the terms with Y(z) on the left-hand side and the others on the right-hand side and comparing the equation with continuous-time transfer function of the integrator yields the following relationship between Laplace- and z-Transform. This is called bilinear transformation or Tustin's formula:

$$s \rightarrow \frac{2\left(1-z^{-1}\right)}{T_S\left(1+z^{-1}\right)}$$

Example:
Now consider the circuit of a 1st-order active low-pass filter in Figure 2:

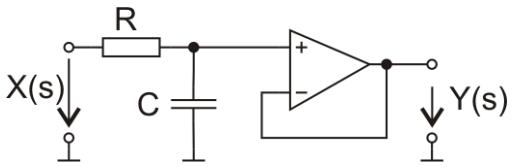

Figure 2: Active 1st-order low-pass filter with operational amplifier

The continuous time transfer function of the system

$$H(s)=\frac{1}{1+\frac{s}{\omega_g}} \quad \text{with } \omega_g=2\pi f_g=\frac{1}{RC}$$

is transformed into the z-Domain with Tustin's method. Replacing the s-operator in the CT transfer function H(s) we get DT transfer function H(z) of the system:

$$H(z)=\frac{1}{1+\frac{s}{\omega_g}}=\frac{1}{1+\frac{2(1-z^{-1})}{\omega_g T_S(1+z^{-1})}}=\ldots=\frac{\frac{\omega_g T_S}{\omega_g T_S+2}\cdot\left(z^{-1}+1\right)}{\frac{\omega_g T_S-2}{\omega_g T_S+2}\cdot z^{-1}+1}$$

If we set the cutoff frequency $f_g=100Hz$ and the sampling frequency $f_S=10\ kHz$ we get the following transfer function:

$$H(z)=\frac{Y(z)}{X(z)}=\frac{0,03046\cdot z^{-1}+0,03046}{-0,9391\cdot z^{-1}+1}$$

Difference Equation and Block Diagram

For the implementation of a digital filter on hardware (e. g. the PEC80-Controller in the practical lab training) the difference equation can be determined from DT transfer function by solving it for the output signal y[n]. The resulting recursion formula for y[n] can be used to realize the filter directly by software (e. g. as C-code) or design the associated block diagram:

Rearranging the transfer function H(z) and separating all terms with X(z) and Y(z) at opposite sites yields to

$$\underbrace{Y(z)}_{\rightarrow y[n]} - 0,9391 \cdot \underbrace{Y(z) \cdot z^{-1}}_{\rightarrow y[n-1]} = 0,03046 \cdot \underbrace{X(z)}_{\rightarrow x[n]} + 0,03046 \cdot \underbrace{X(z) \cdot z^{-1}}_{\rightarrow x[n-1]}$$

As mentioned before the multiplication with z^{-n} generally causes a delay of the corresponding signal for n samples.

After the inverse z-Transform is applied the difference equation, solved for the output signal y[n], can be written as follows:

$$y[n] = 0,03046 \cdot x[n] + 0,03046 \cdot x[n-1] + 0,9391 \cdot y[n-1]$$

The corresponding Matlab-Simulink block diagram shown in Figure 3 can be directly drawn from the difference equation. Since the block "Unit Delay" holds up the value until every next clock, contrary it provides every past value one clock before. If delays of more than one clock are required, unit delays can be connected in series. Due to the feedback path from the output, the low-pass filter has an **I**nfinite **I**mpulse **R**esponse, and this type of filter is generally called **IIR**-filter.

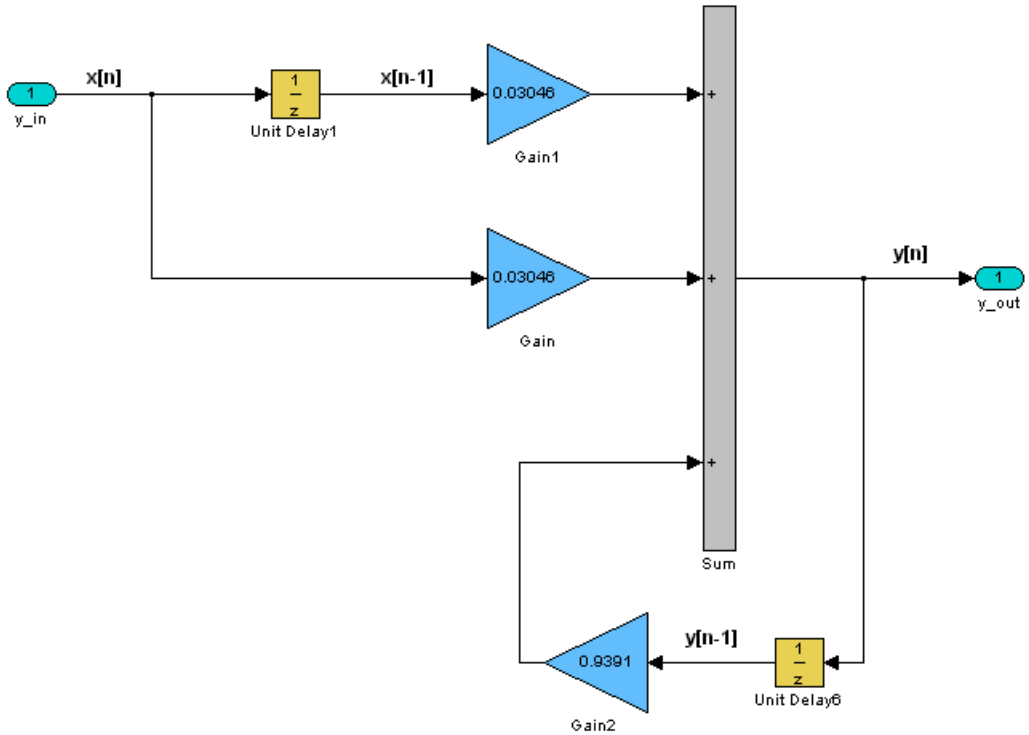

Figure 3: Matlab-Simulink model for the digital low-pass filter

Implementation of the filter

For the implementation of the digital filter a powerful PEC80-Controller (**p**ower **e**lectronics **c**ontroller) from the company ABB Switzerland Ltd. is provided which is especially designed for common applications in Power Electronics, Digital Control and Digital Signal Processing. The PEC80-Controller, shown in Figure 4, offers a lot of various interfaces, such as digital/analog inputs and outputs, optical inputs and outputs, Ethernet, CAN and RS485.

Figure 4: PEC80-Controller

The controller can be graphically programmed with Matlab-Simulink in very convenient way using a special Toolbox (AC 800PEC Toolbox) provided by the company ABB Switzerland Ltd. The integrated tool-chain “Real-Time-Workshop” converts the block diagrams from a Simulink model into real-time C-Code, compiles then into executable binary code and downloads the software over Ethernet to the PEC80-Controller.

As shown below in Figure 5, the Simulink model consists of some particular blocks necessary for the real-time operation system, such as the "PEC80 Interrupt Controller" that triggers periodically three different tasks. Depending on the requirements of the application time-critical parts need to run within tasks with a small cycle time and high priority. In our application the digital filter is implemented in "Task A" (with the highest priority) to ensure a constant sampling period of 100 µs, that equals the cycle time. Less critical program functions, e. g. the digital inputs/outputs, can be processed by the slower tasks "Task B" and "Task C".

Additionally the PEC80-Controller offers an integrated FPGA for extreme time-critical signal processing, whose software is designed in the high level language VHDL. For our project the FPGA is already pre-configured.

The basic structure of the Simulink model for the digital filter is illustrated in Figure 5.

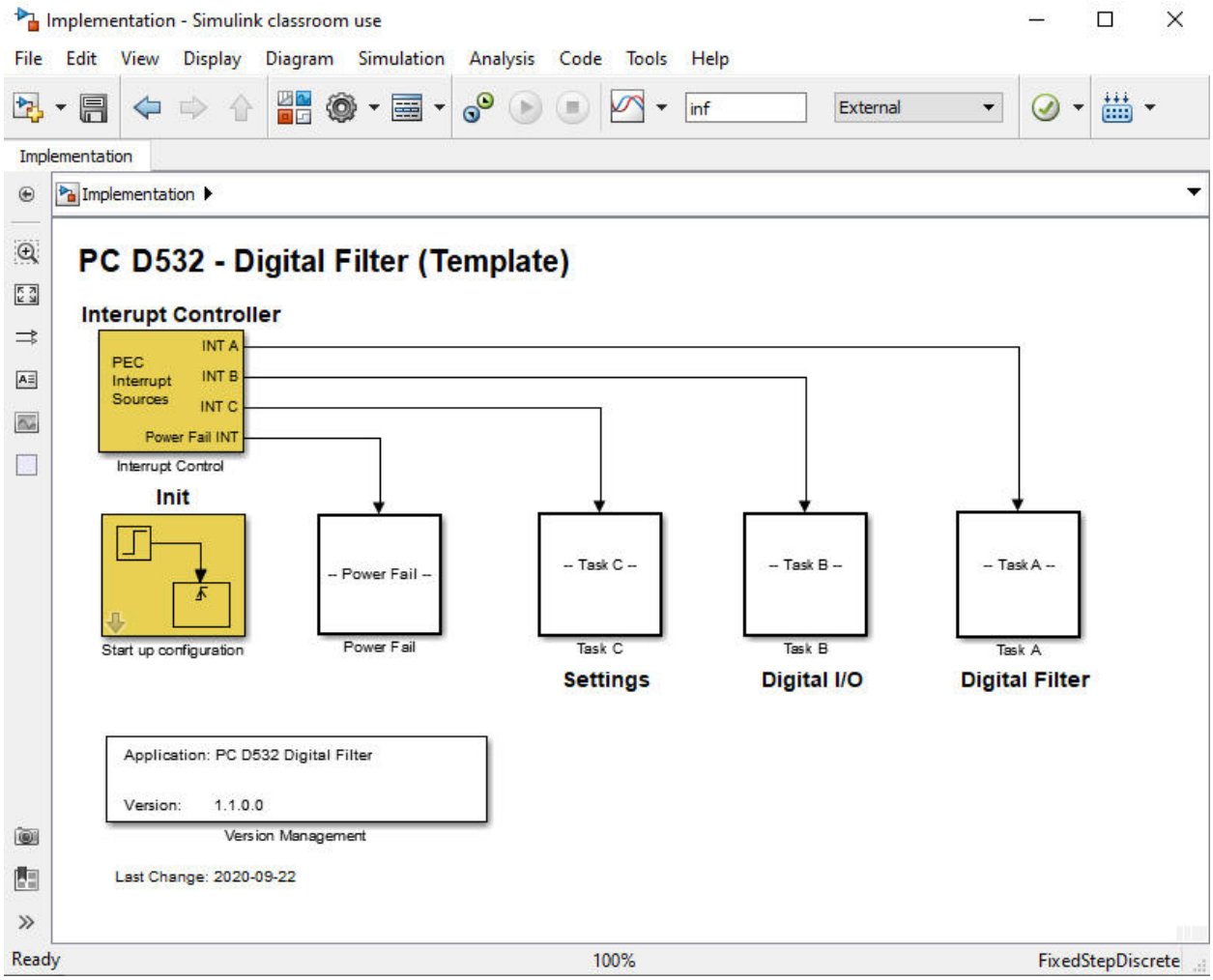

Figure 5: Top level Simulink model for the digital filter

The filter algorithm is periodically executed in "Task A" every 100µs and all other task have usually a larger cycle time and lower priority.

A double-click on the block “Task A” opens the corresponding subsystem (see Figure 6). Whenever the subsystem is called by the task handler, the input signal x[n] (assigned to “Analog In CH1”) is read, the resulting output value y[n] is calculated in the subsystem “Digital Filter” and subsequently y[n] is written to hardware (assigned to “Analog Out CH1).

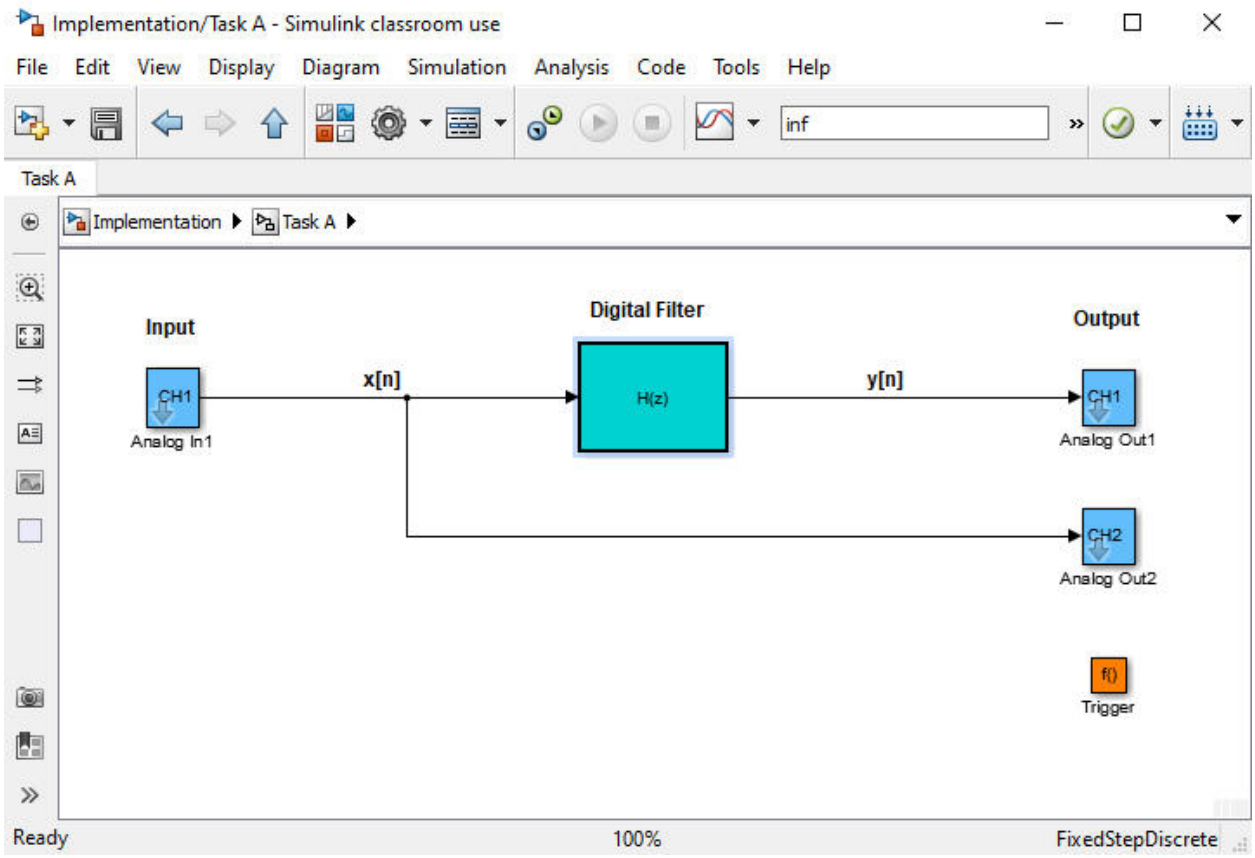

Figure 6: Subsystem „Task A“

Another double-click on the function block “Digital Filter“ shows the block diagram of the digital low-pass filter from Figure 3.

With the command "*Build Model*", located under the menu "*Code > C / C++ Code*" or with by the shortcut "*Ctrl + B*" the toolchain for the compilation and download can be started. After setting the correct IP address of PEC80-Controller we can launch the toolchain with "*Build and Download*". The output messages of the tool chain can be observed with the "*Diagnostic Viewer*" (see menu "*View*").

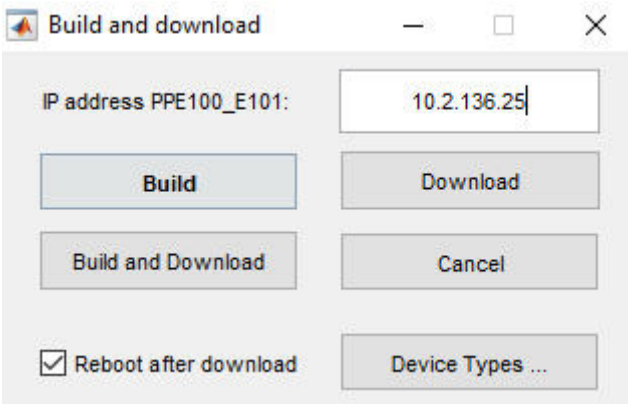

Figure 7: Start of Compilation and Download

After downloading the software, the real-time operating system restarts itself automatically. As soon as the R-LED (Run) is constant on again, a connection between the desktop PC and the controller can be established.

The Matlab-Simulink implementation on the PEC80-Controller can be accessed by clicking the button "*Connect To Target*" on the toolbar (see Figure 8). Check if the additional preferences "*inf*" and "*External*" are correctly set. When the connection is successfully established, the system time of PEC80-Controller (since the last restart) is displayed on the right bottom of the window

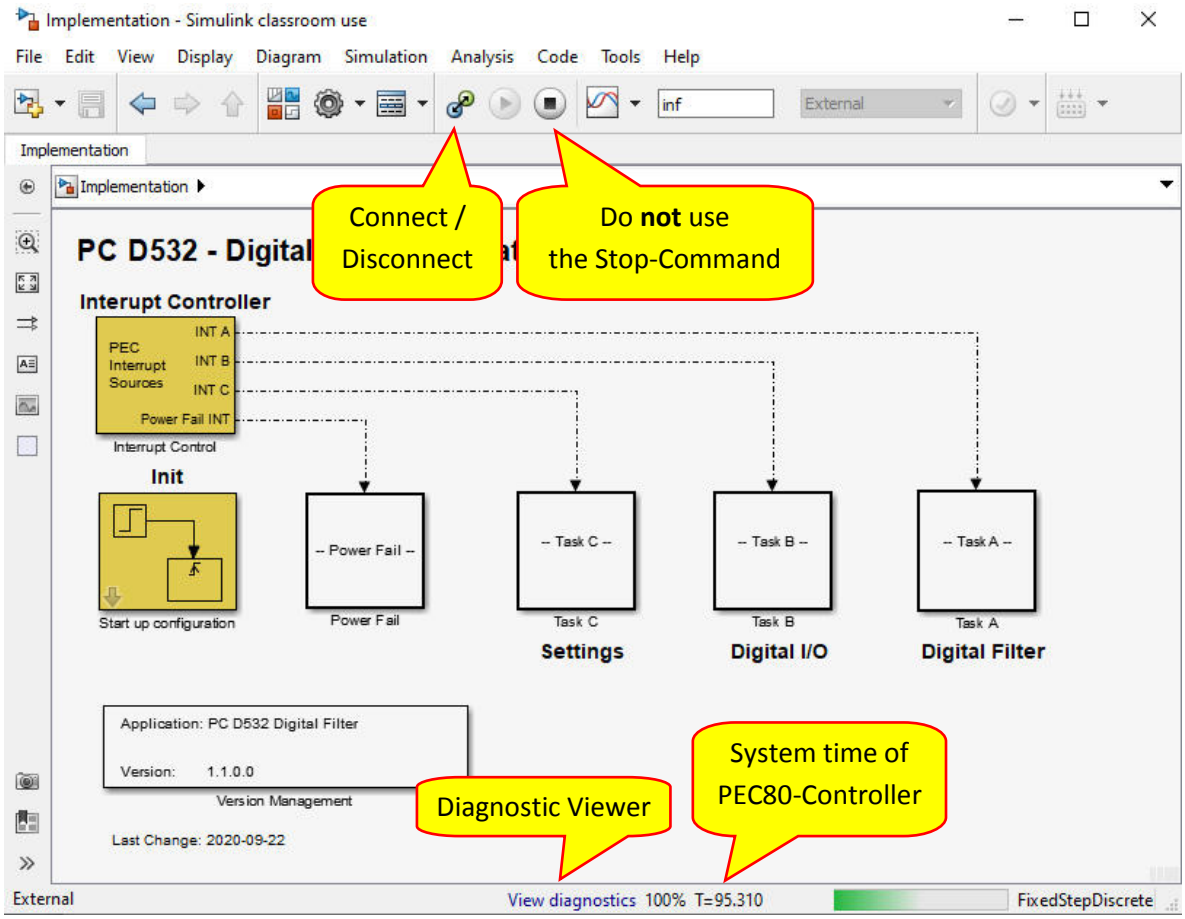

Figure 8: Connection with PEC80-Controller

Before any changes in the Simulink model can performed an existing connection to PEC80 have to be closed before. Note that both connect and disconnect commands are assigned to the same symbol on the toolbar. Do not use the "*Stop-Command*" to avoid connection problems.