https://www.halvorsen.blog

Read Temperature Data with Lowpass Filter in LabVIEW

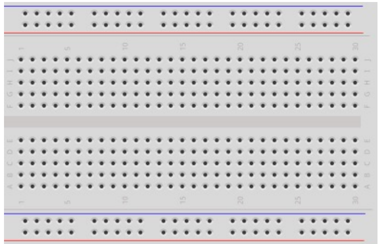
Hans-Petter Halvorsen

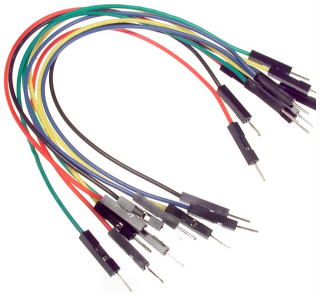
Contents

- We will use LabVIEW to read Temperature data from TMP36 Temperature Sensor
- We will use the USB-6008 DAQ Device or I/O Module
- The Temperature Data will typically include some Noise
- We will create and apply a Lowpass Filter in order to reduce the Noise from the Temperature signal

Hardware

- DAQ Device (e.g., USB-6008)
- Breadboard
- TMP36 Temperature Sensor
- Wires (Jumper Wires)





Software

- LabVIEW
  - Graphical Programing Environment
- DAQmx Driver
  - Driver used for Communication with external Hardware such as USB-6008

Reading Temperature Data

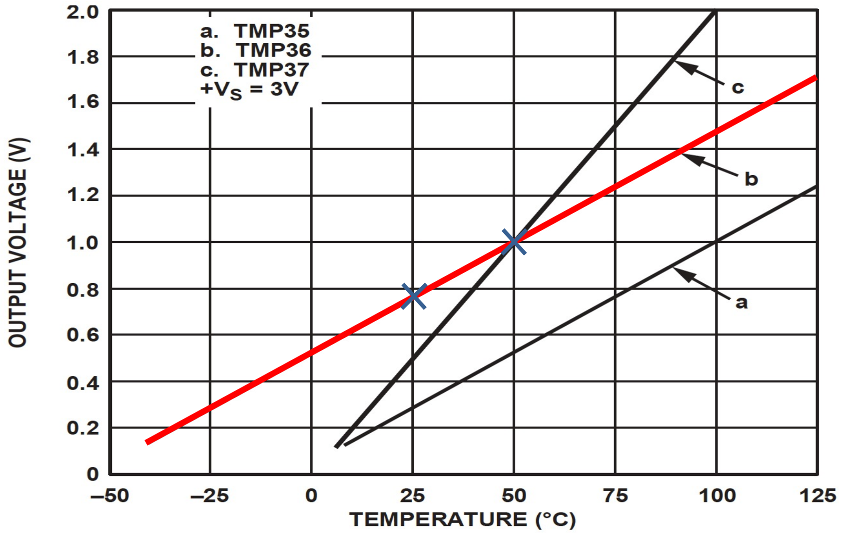
Hans-Petter Halvorsen

USB-6008

- USB-6008 is a DAQ Device from NI
- Can be used within LabVIEW
- NI-DAQmx Driver
- It has Analog and Digital Inputs and Outputs

TMP36 - Linear Scaling

TMP3x Datasheet:



This gives:

$$y - 25 = \frac{50 - 25}{1 - 0.75}(x - 0.75)$$

Then we get the following formula:

$$y = 100x - 50$$

Convert form Voltage (V) to degrees Celsius
From the Datasheet we have:

$$(x_1, y_1) = (0.75V, 25°C)$$
$$(x_2, y_2) = (1V, 50°C)$$

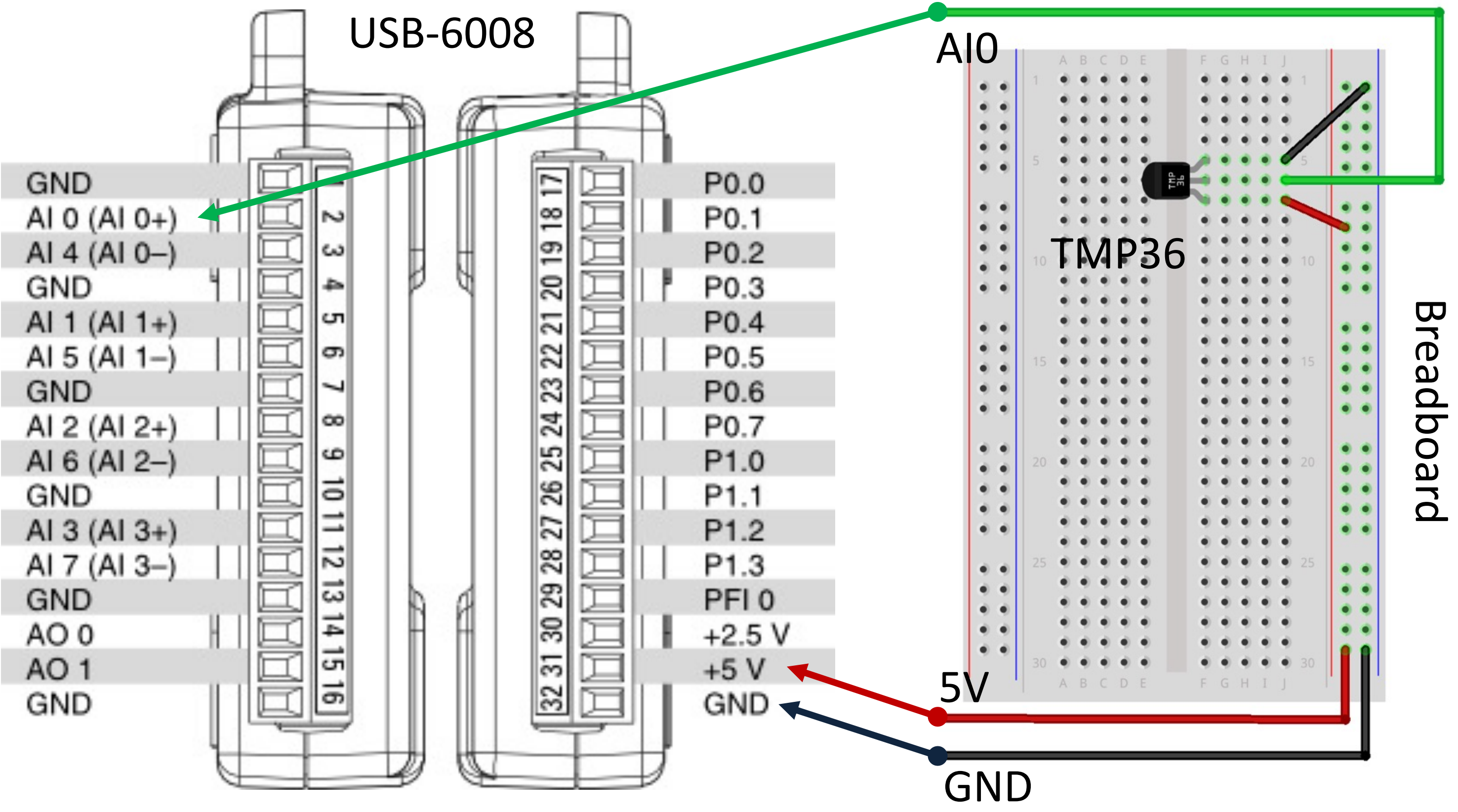
There is a linear relationship between Voltage and degrees Celsius:

$$y = ax + b$$

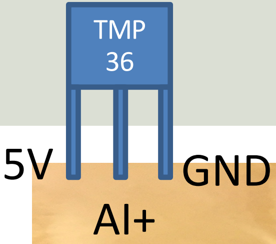
We can find a and b using the following known formula:

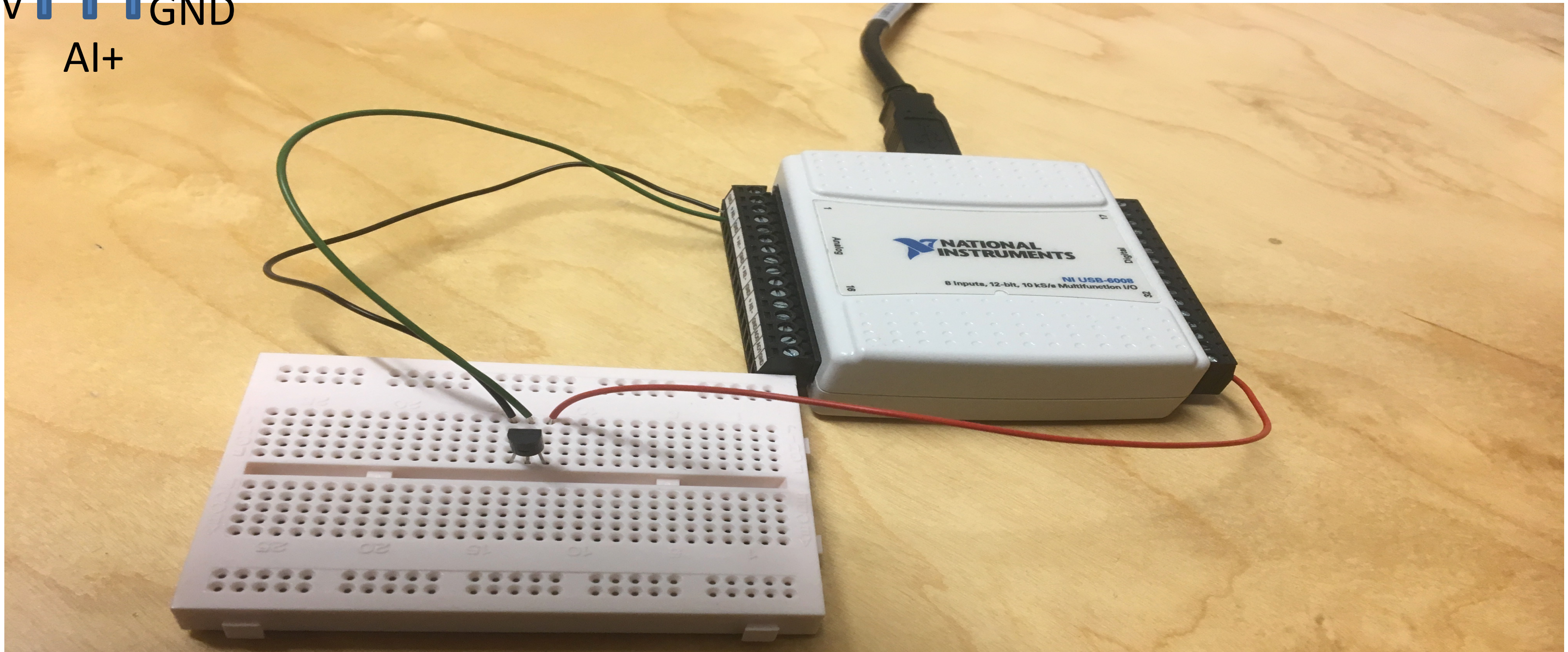
$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

Wiring


USB-6008
AI0
TMP36
Breadboard
5V
GND
GND
AI 0 (AI 0+)
AI 4 (AI 0–)
GND
AI 1 (AI 1+)
AI 5 (AI 1–)
GND
AI 2 (AI 2+)
AI 6 (AI 2–)
GND
AI 3 (AI 3+)
AI 7 (AI 3–)
GND
AO 0
AO 1
GND
P0.0
P0.1
P0.2
P0.3
P0.4
P0.5
P0.6
P0.7
P1.0
P1.1
P1.2
P1.3
PFI 0
+2.5 V
+5 V
GND

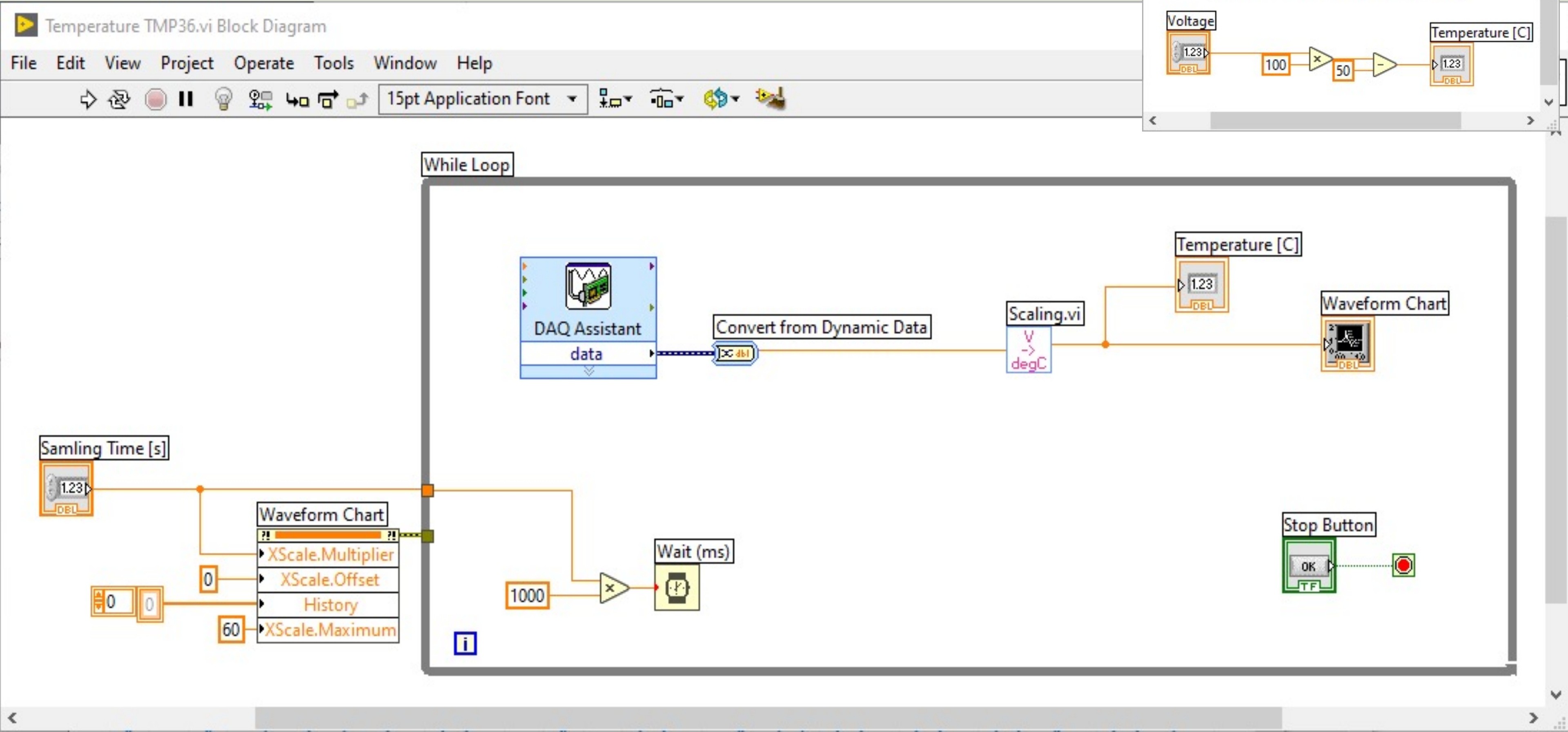
Wiring

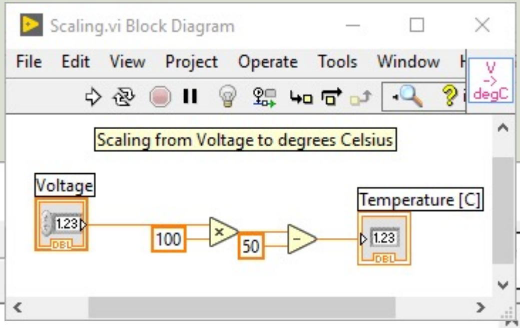




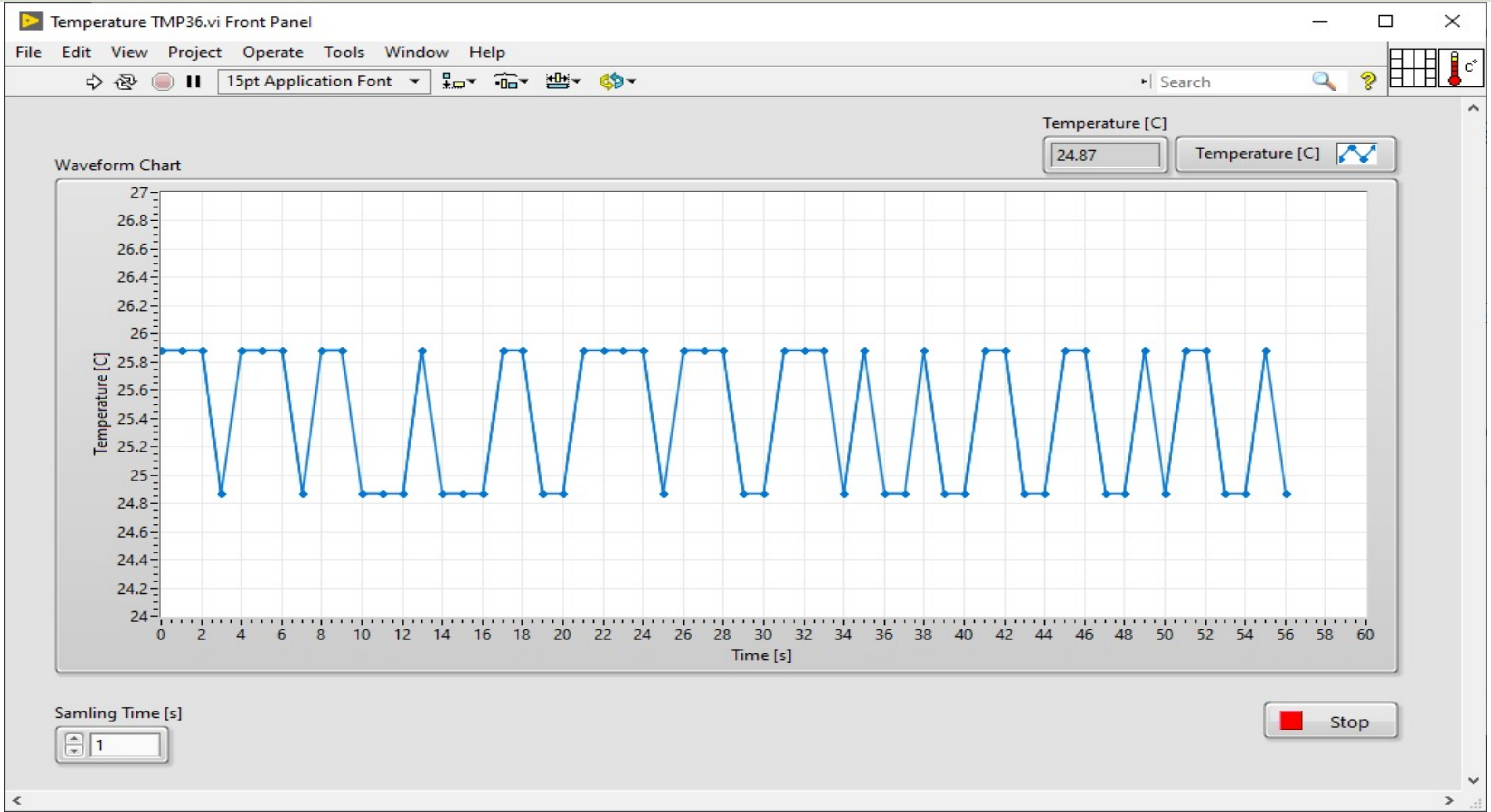
We connect the TMP36 to LabVIEW using a USB DAQ Device from National Instruments, e.g., USB-6001, USB-6008 or similar. I have used a breadboard for the wiring.

Read Temperature Data





Read Temperature Data



Discussions

- We see that the signal is quite noisy
- We want to use a Filter in order to remove or reduce the noise from the signal

Lowpass Filter

Hans-Petter Halvorsen

Lowpass Filter

The Transfer Function for a Low-pass filter is given by:

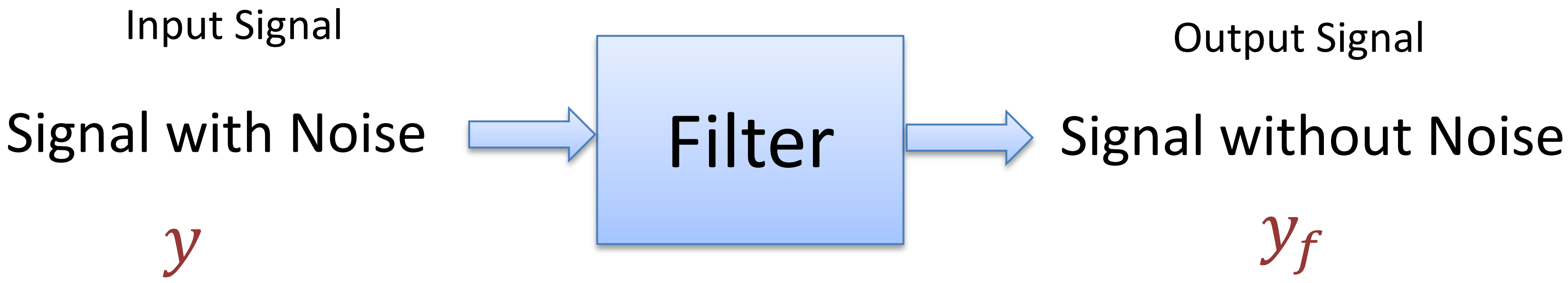
$$H(s) = \frac{y_f(s)}{y(s)} = \frac{1}{T_f s + 1}$$

Where:
y is the Signal from the DAQ device (that contains noise)
y_f is the Filtered Signal
T_f is the Filter Time Constant

Why Lowpass Filter?

- In Measurement systems and Control Systems we typically need to deal with noise
- Noise is something we typically don't want
- Lowpass Filters are used to remove noise from the measured signals
- Noise is high-frequency signals
- A Lowpass Filter make sure the low frequencies pass (the measurements) and removes the high frequencies (the noise)

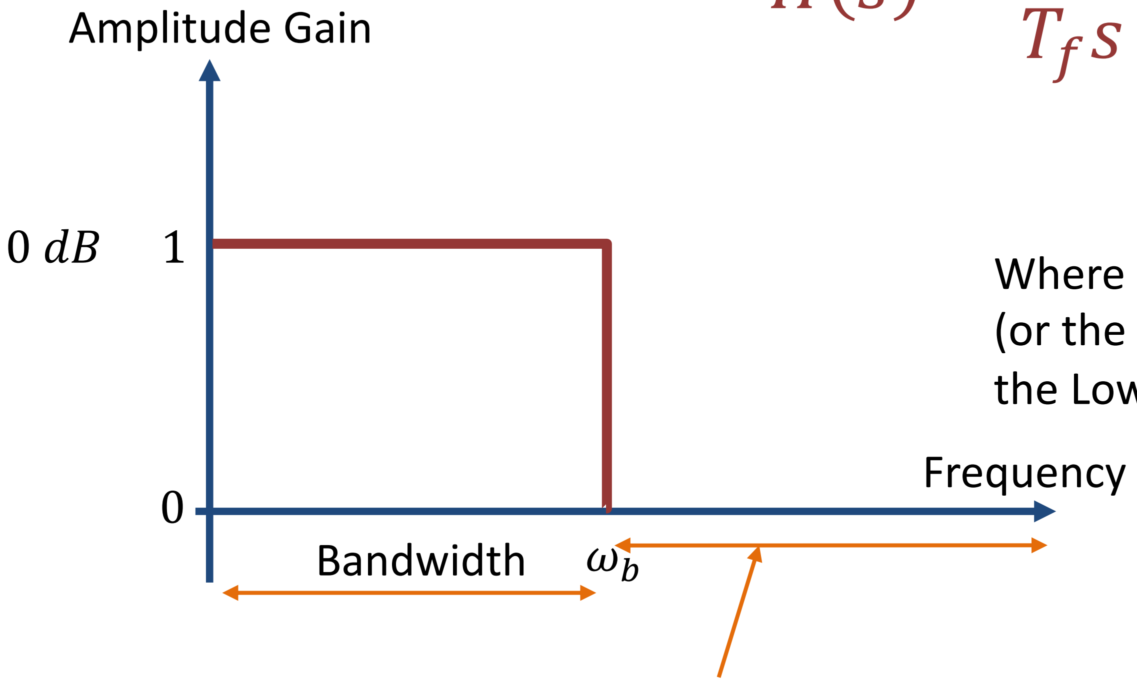
Lowpass Filter



Lowpass Filter

Below we see an Ideal Lowpass Filter:

$$H(s) = \frac{1}{T_f s + 1} = \frac{1}{\frac{1}{\omega_b} s + 1}$$



Where ω_b is the Bandwidth (or the cut-off frequency) of the Lowpass Filter

High frequencies (above ω_b) are removed (or reduced)

From Transfer Function to Differential Equation

A Low-pass Filter has the following Transfer Function:

$$H(s) = \frac{y_f(s)}{y(s)} = \frac{1}{T_f s + 1}$$

We can find the Differential Equation for this filter using Inverse Laplace

We get:

$$y_f(s)[T_f s + 1] = y(s)$$

$$T_f y_f(s)s + y_f = y(s)$$

Finally, we get the following Differential Equation:

$$T_f \dot{y}_f + y_f = y$$

We apply Euler on the Differential Equation in order to find the Discrete Differential equation. See next Page

Discretization of Lowpass Filter

We have the following Differential Equation:

$$T_f \dot{y}_f + y_f = y$$

We use Euler Backward method: $\dot{x} \approx \frac{x(k)-x(k-1)}{T_s}$

Then we get:

$$T_f \frac{y_f(k) - y_f(k-1)}{T_s} + y_f(k) = y(k)$$

This gives: $y_f(k) = \frac{T_f}{T_f+T_s} y_f(k-1) + \frac{T_s}{T_f+T_s} y(k)$

We define:

$$\frac{T_s}{T_f + T_s} \equiv a$$

Finally, we get the following discrete version of the Lowpass Filter:

$$y_f(k) = (1-a)y_f(k-1) + ay(k)$$

This equation can easily be implemented in LabVIEW or another programming language

Discrete Lowpass Filter

Discrete Lowpass Filter:

$$y_f(k) = (1 - a)y_f(k - 1) + ay(k)$$

Where:

$$\frac{T_s}{T_f + T_s} \equiv a$$

$y(k)$ is the current Signal from the DAQ device (that contains noise)

$y_f(k)$ is the Filtered Signal

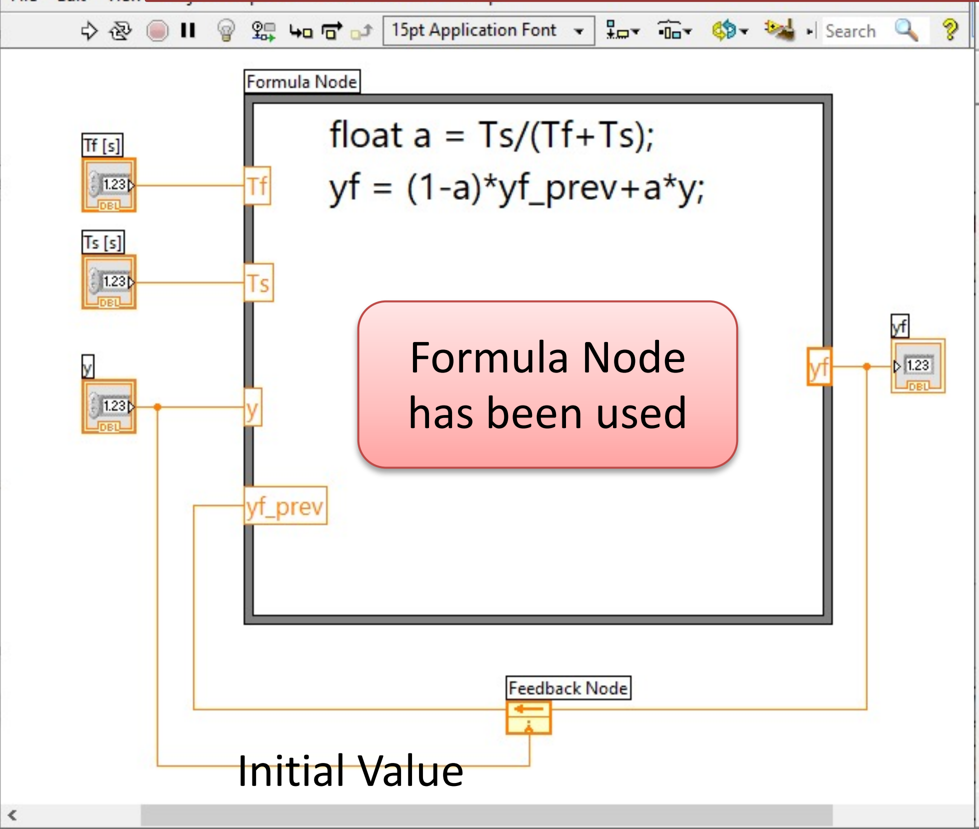
$y_f(k-1)$ is previous filtered signal

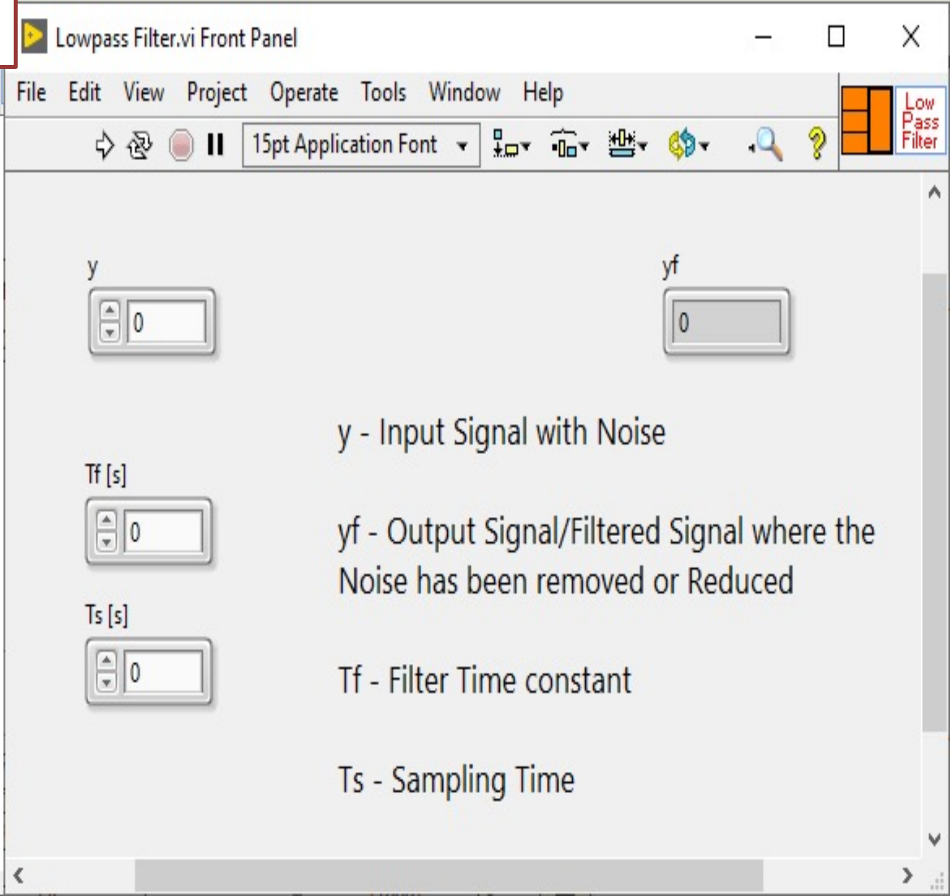
T_f is the Filter Time Constant

T_s is the Sampling Time

Lowpass Filter in LabVIEW

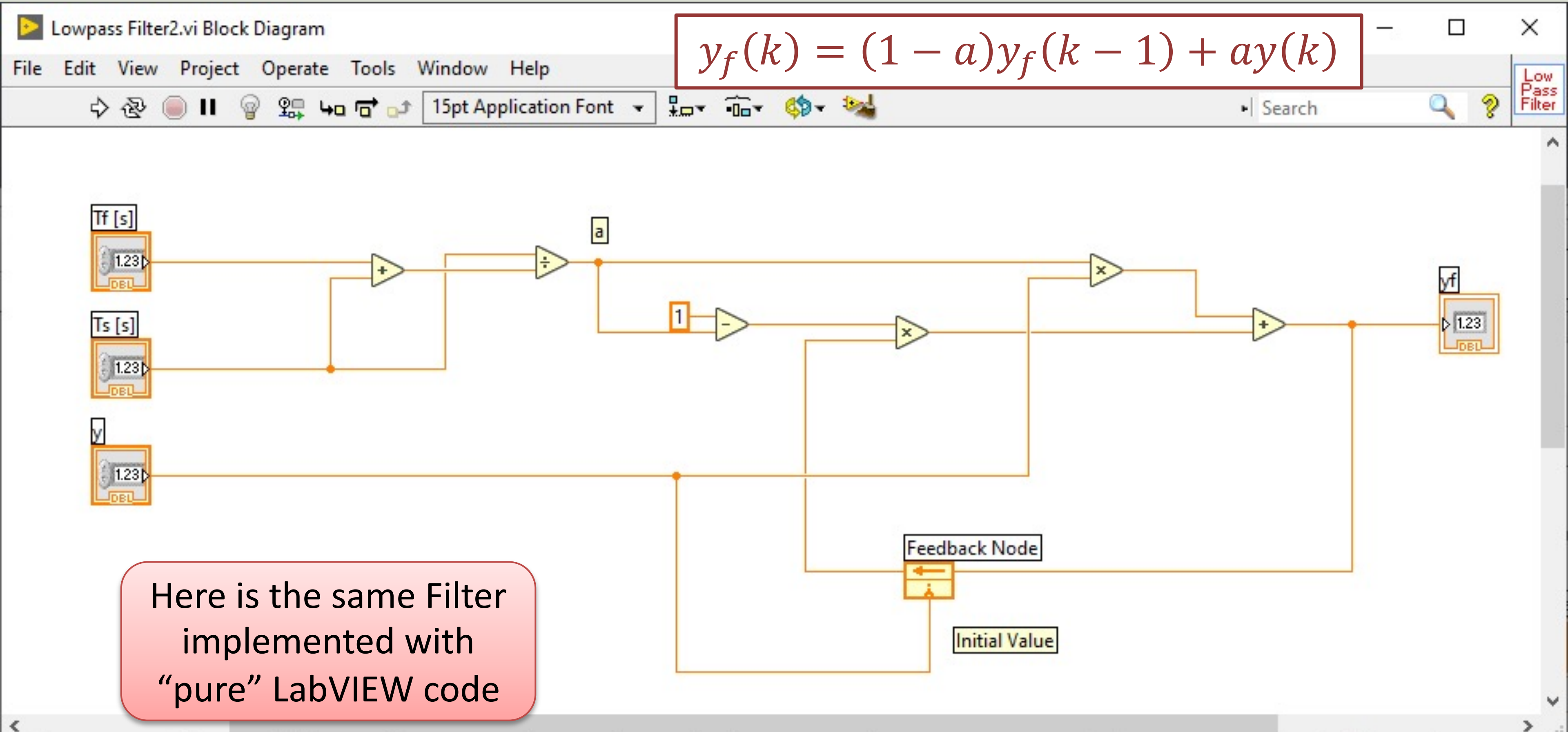
$$y_f(k) = (1-a)y_f(k-1) + ay(k)$$





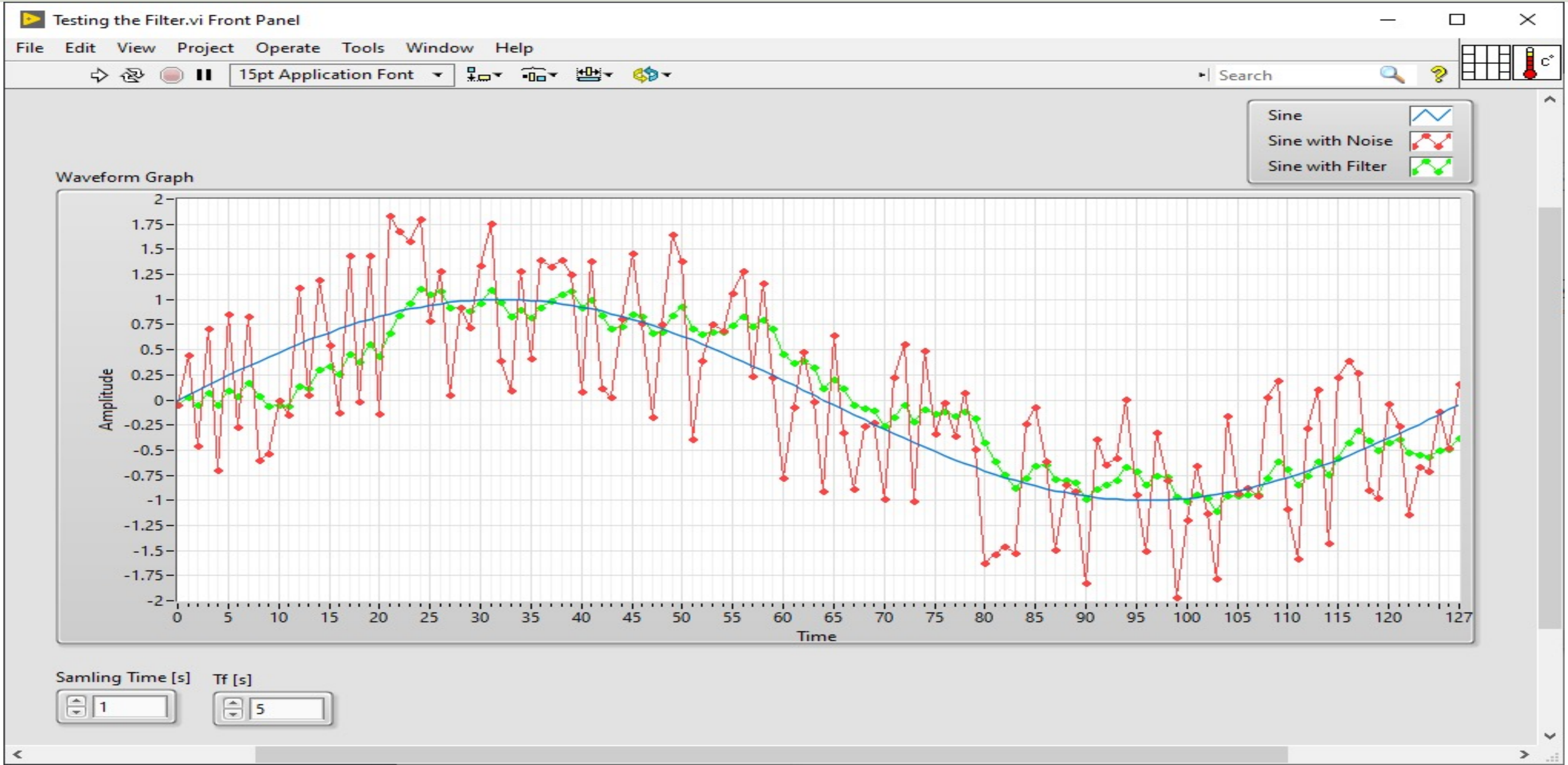
Lowpass Filter in LabVIEW

$$y_f(k) = (1 - a)y_f(k-1) + ay(k)$$

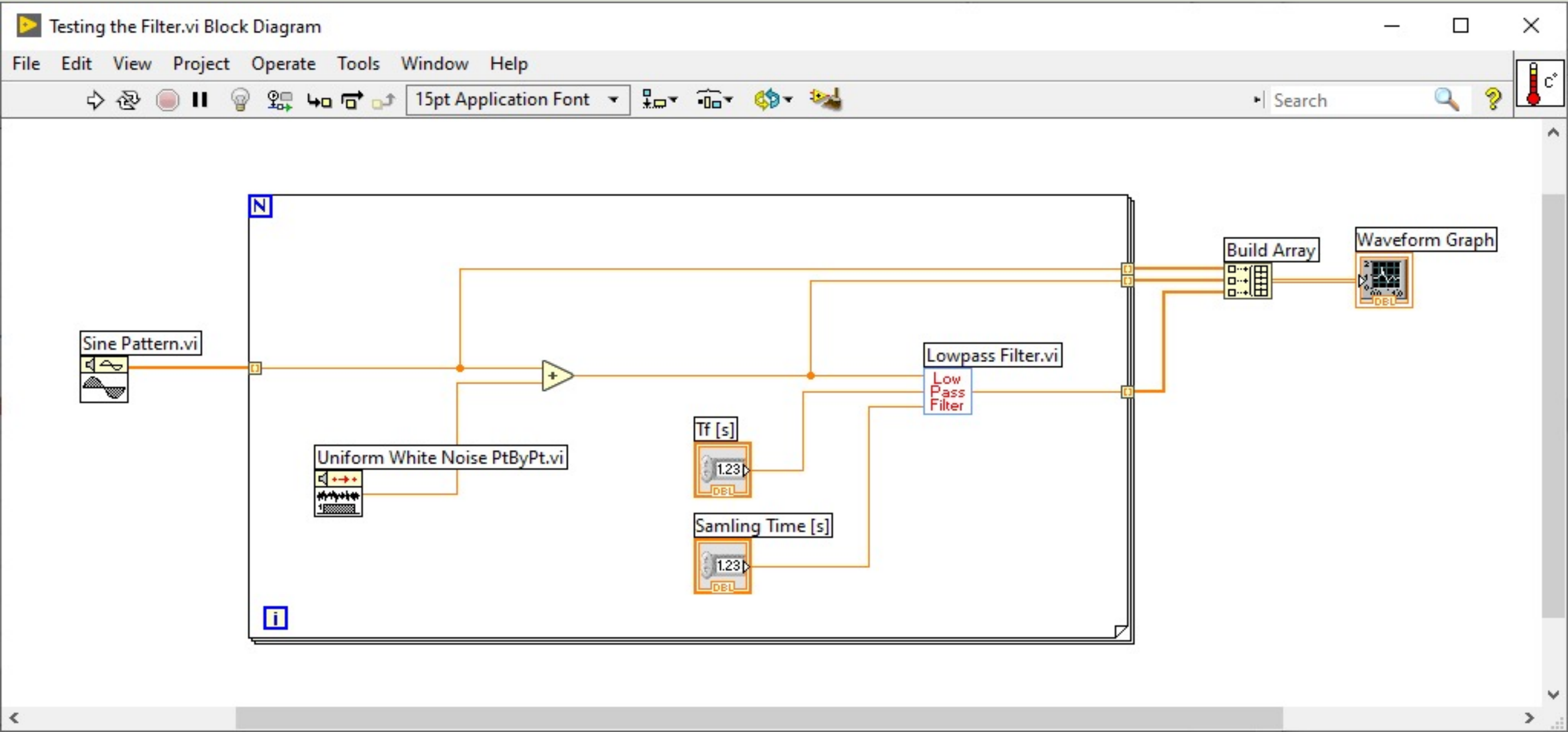


Here is the same Filter implemented with "pure" LabVIEW code

We test the Filter



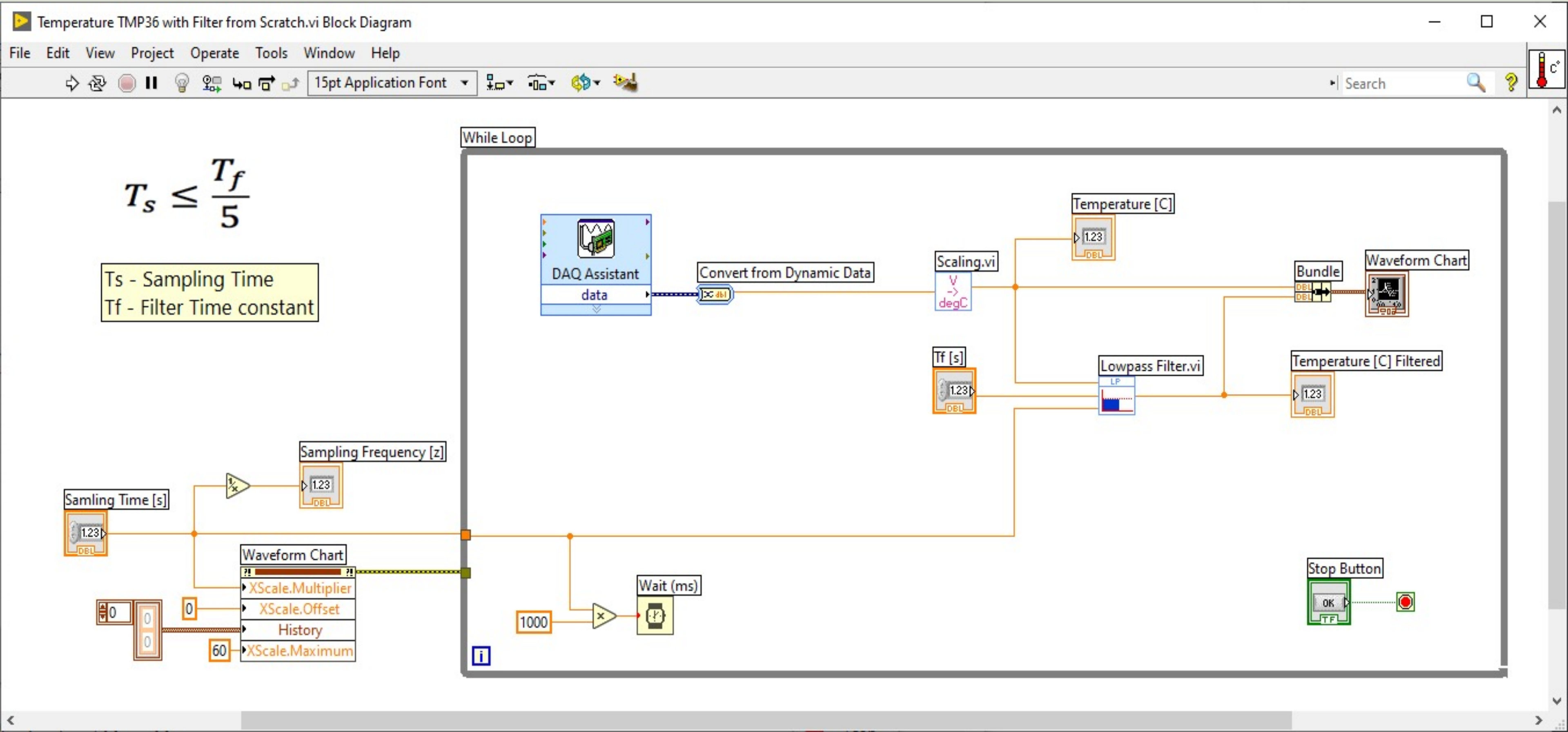
We test the Filter - LabVIEW



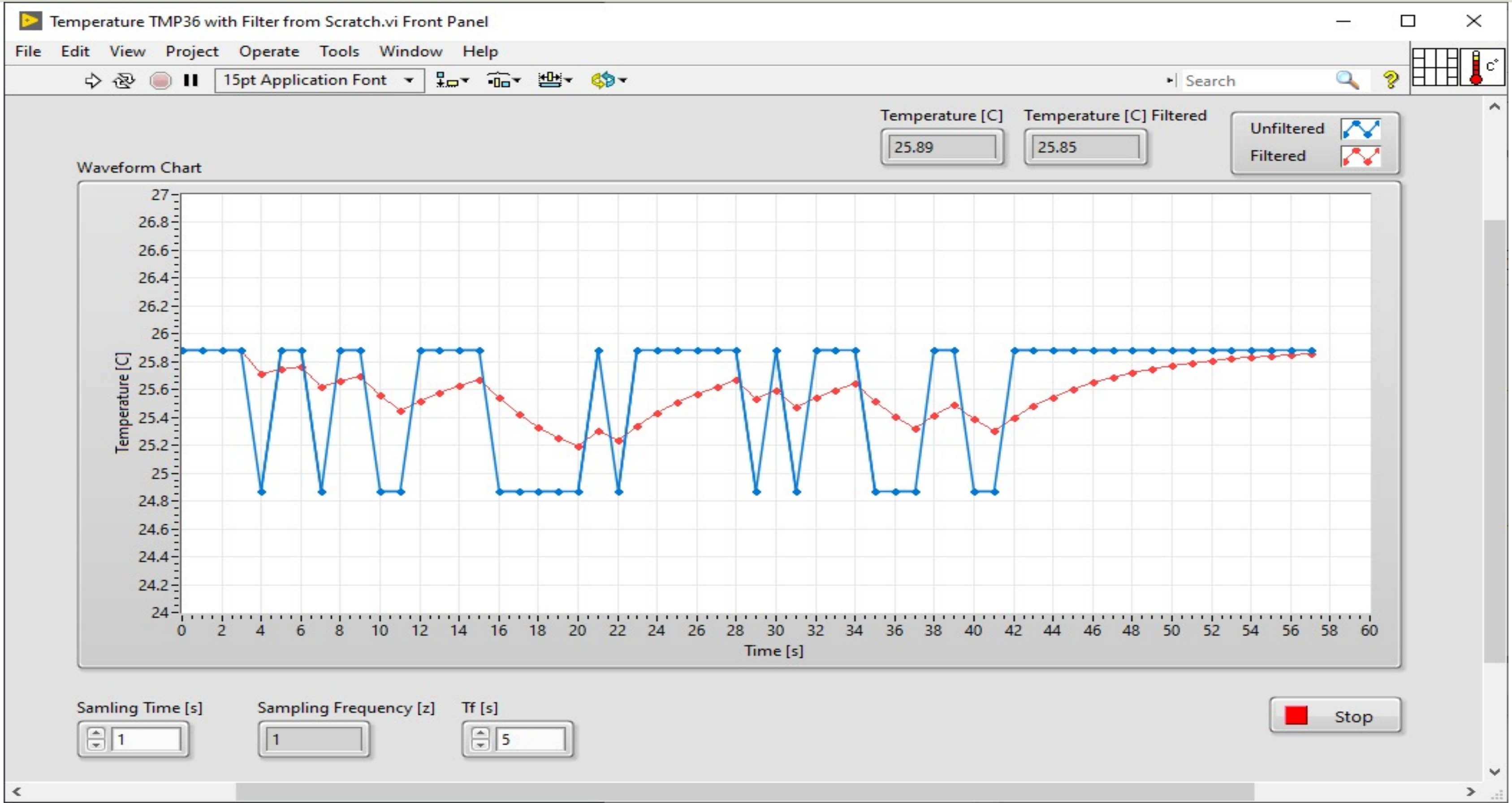
Reading Temperature Data with Lowpass Filter

Hans-Petter Halvorsen

Temperature Data with Filter



Temperature Data with Filter



Summary

- We see that the signal is quite noisy
- We want to use a Filter in order to remove or reduce the noise from the signal
- We see from the results that by implementing and applying a Lowpass Filter we get a much Smoother Signal
- If we use a Noisy Signal as an input to a PID Controller it will affect the stability of the Control System

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog