https://www.halvorsen.blog

Discrete Control Systems in LabVIEW

Hans-Petter Halvorsen

Contents

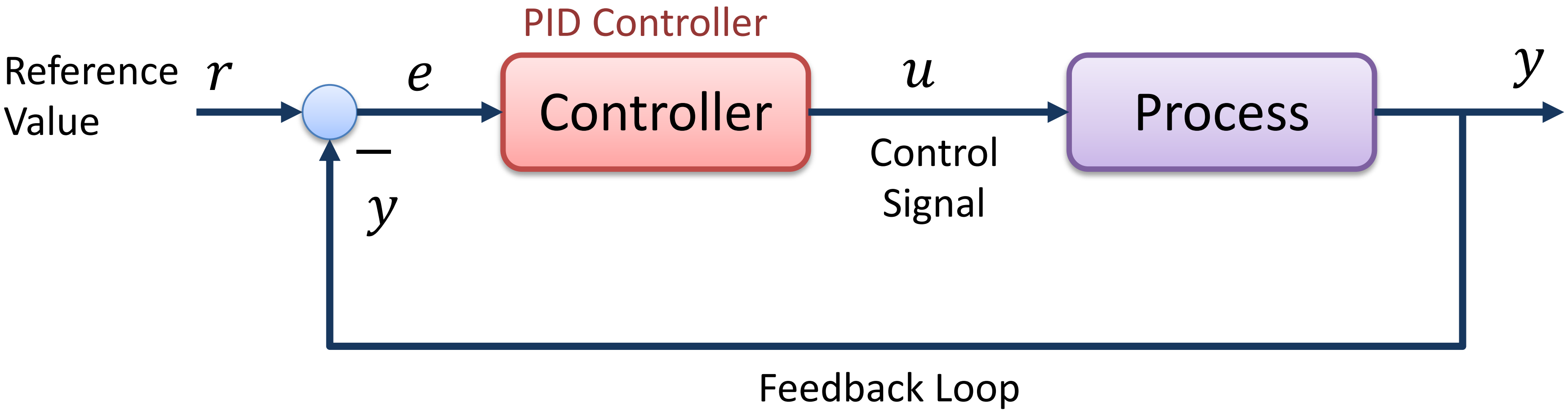
Introduction

Introduction

- We will simulate a 1. Order Process/Differential Equation
  - We will Implement a Discrete version of the Model and perform Simulations
- We will create a basic Control System
  - We will make and Implement a Discrete PI Controller and perform Simulations

Control System

The purpose with a Control System is to Control a Dynamic System, e.g., an industrial process, an airplane, a self-driven car, etc. (a Control System is “everywhere” today)



Mathematical Model

Hans-Petter Halvorsen

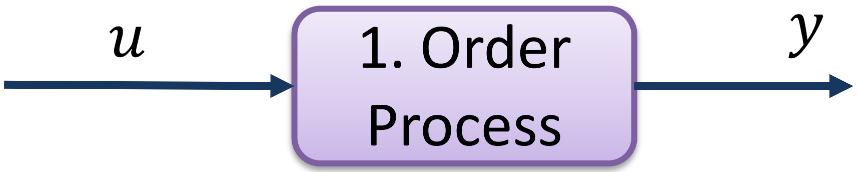
Table of Contents

1. Order System

Differential Equation of a 1. order System:

$$\dot{x} = -ax + bu$$

$$y = x$$

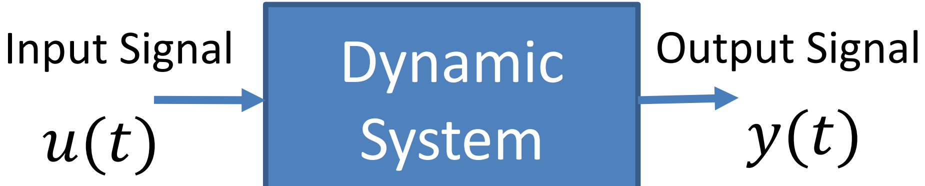


In order to simulate this model in LabVIEW you can make a discrete version of the model, or you can implement it as a “Block Diagram” using the features in LabVIEW Control Design and Simulation Module

1. Order System

Assume the following general Differential Equation:

$$\dot{y} = -ay + bu$$



or:

$$\dot{y} = \frac{1}{T}(-y + Ku)$$

Where $a = \frac{1}{T}$ and $b = \frac{K}{T}$

Where K is the Gain and T is the Time constant

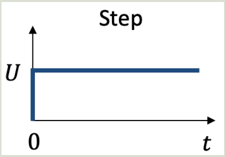
This differential equation represents a 1. order dynamic system

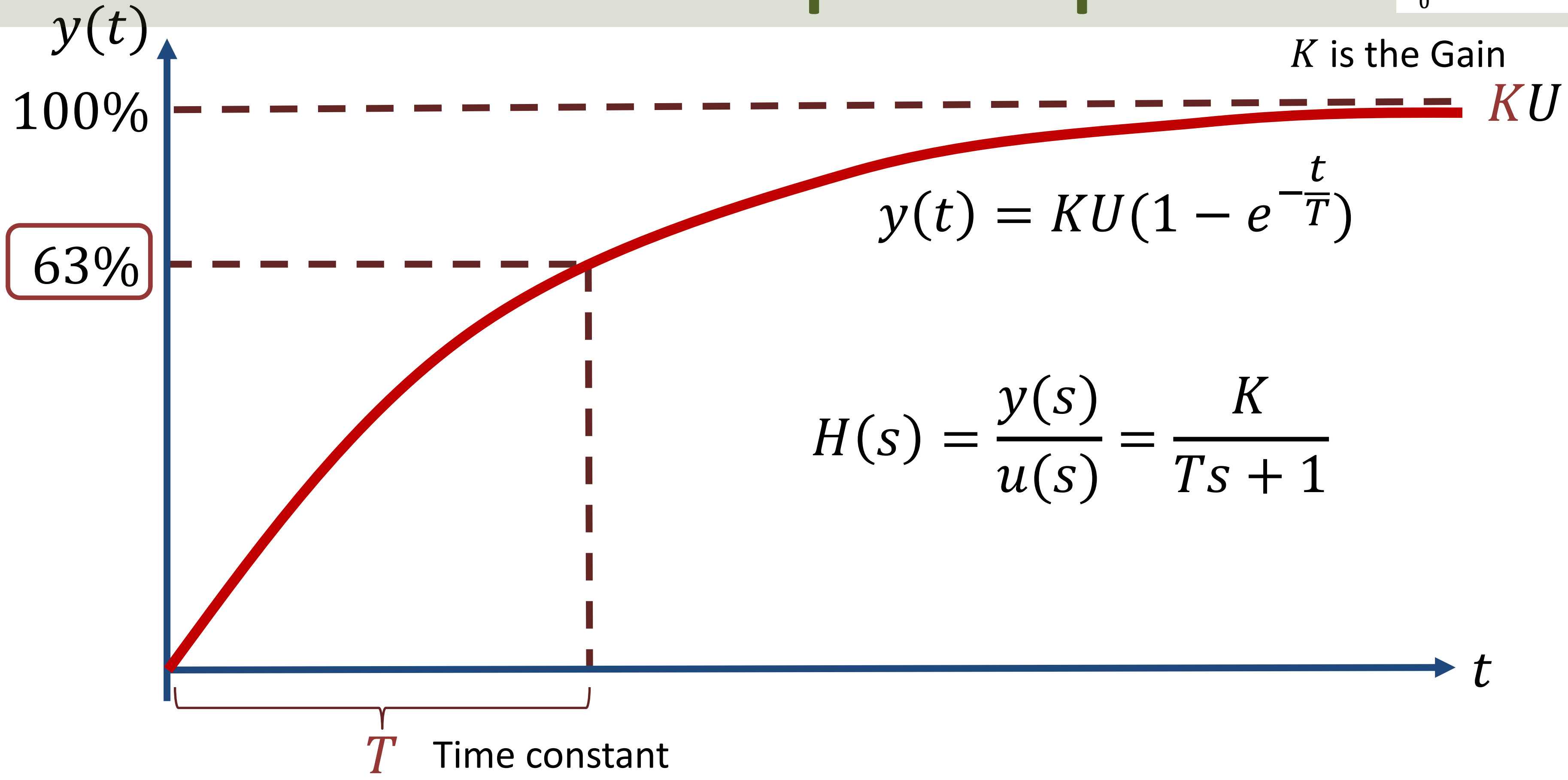
Assume $u(t)$ is a step (U), then we can find that the solution to the differential equation is:

$$y(t) = KU(1 - e^{-\frac{t}{T}})$$

(by using Laplace)

1. Order Step Response





$$y(t) = KU(1 - e^{-\frac{t}{T}})$$

$$H(s) = \frac{y(s)}{u(s)} = \frac{K}{Ts + 1}$$

Discretization

We have the continuous differential equation: $\dot{x} = -ax + bu$

We apply **Euler**: $\dot{x} \approx \frac{x(k+1)-x(k)}{T_s}$

Then we get:

$$\frac{x(k+1) - x(k)}{T_s} = -ax(k) + bu(k)$$

This gives the following discrete differential equation (difference equation):

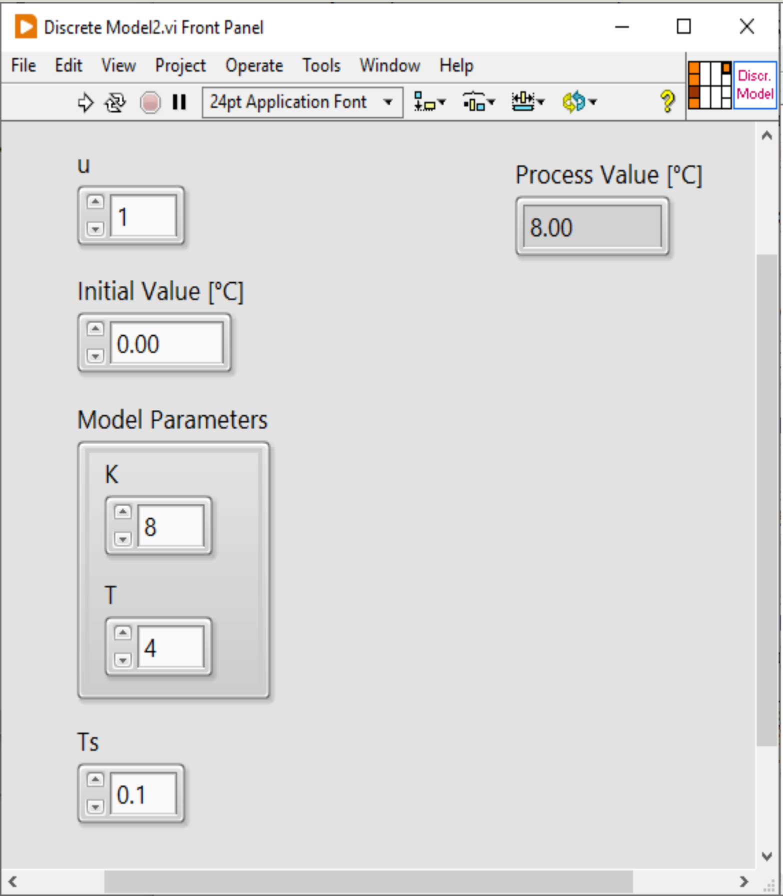
$$x(k+1) = (1 - T_s a)x(k) + T_s bu(k)$$

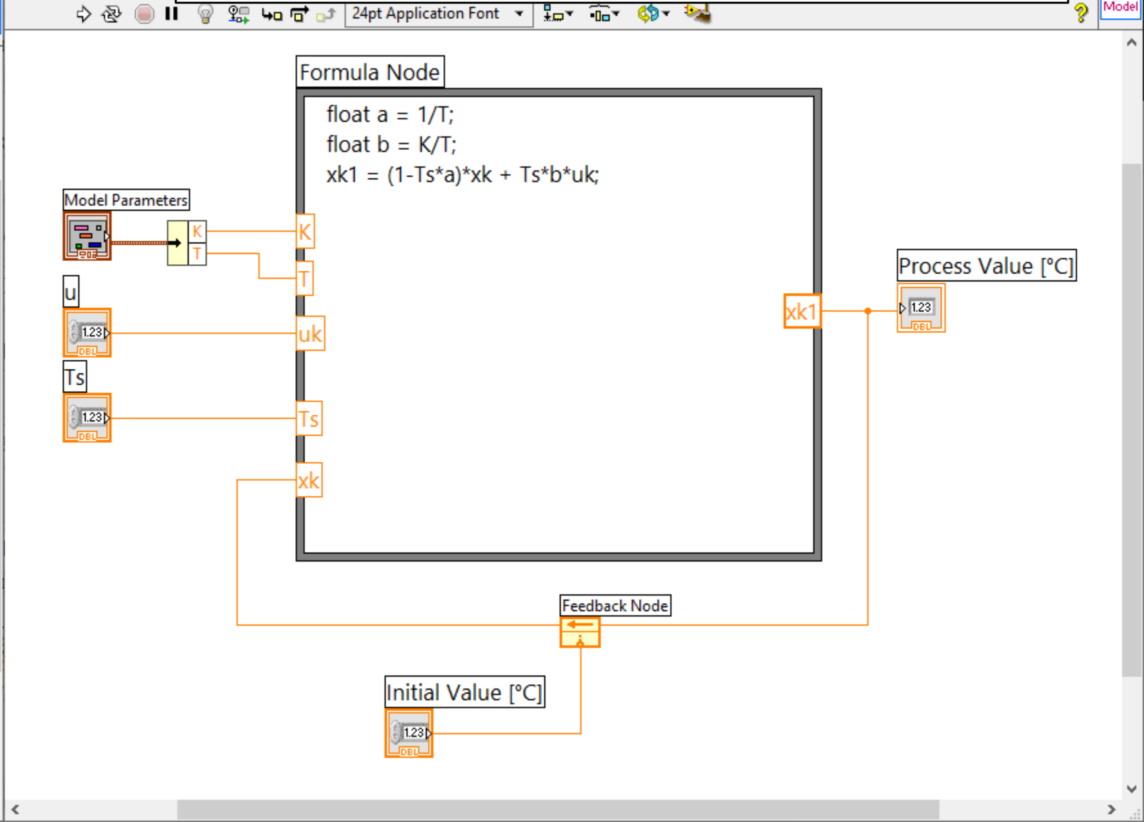
This equation can easily be implemented in any text-based programming language or the Formula Node in LabVIEW

Where $a = \frac{1}{T}$ and $b = \frac{K}{T}$

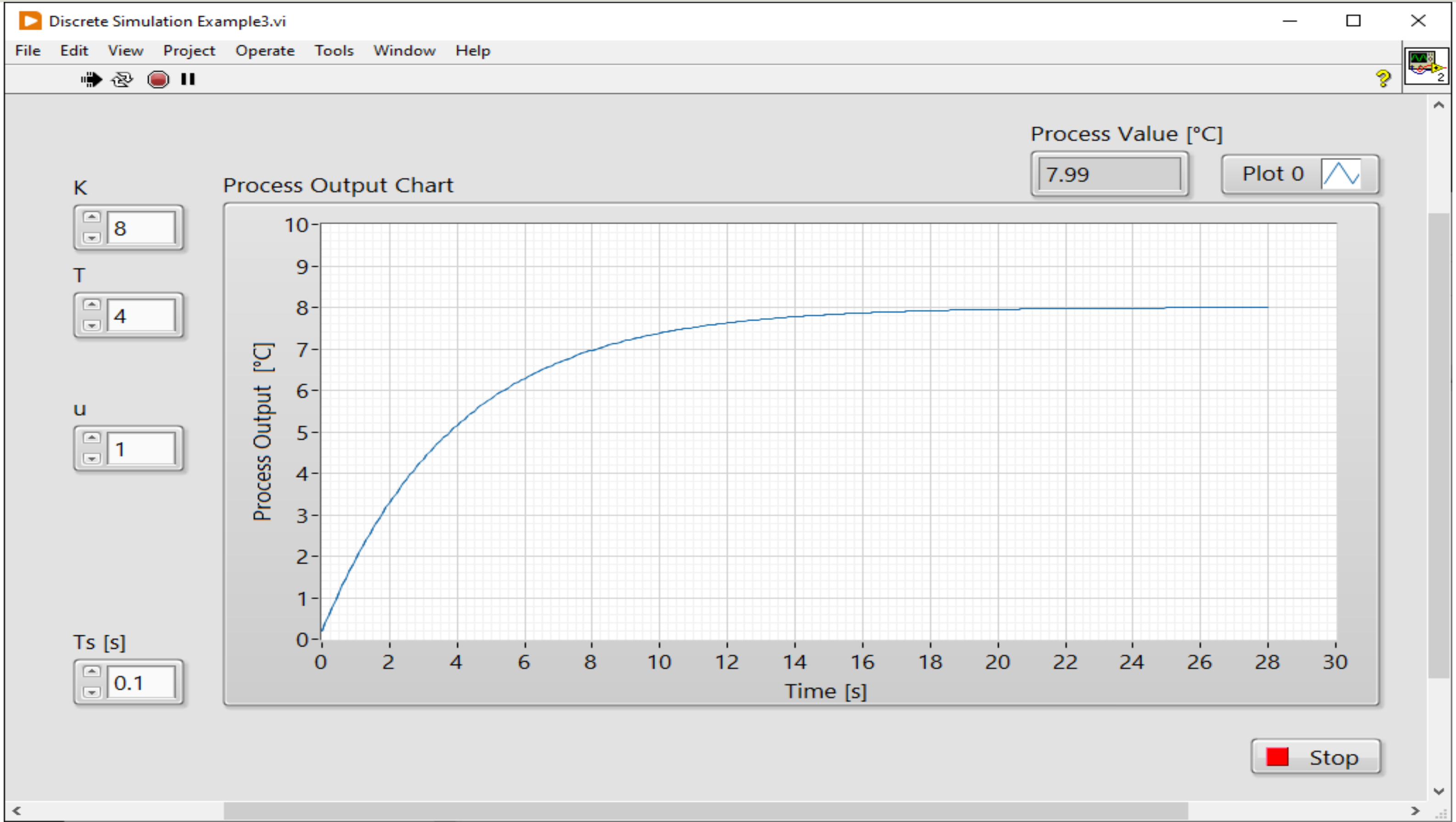
Discrete Model in LabVIEW

$$x(k+1) = (1 - T_s a)x(k) + T_s b u(k)$$

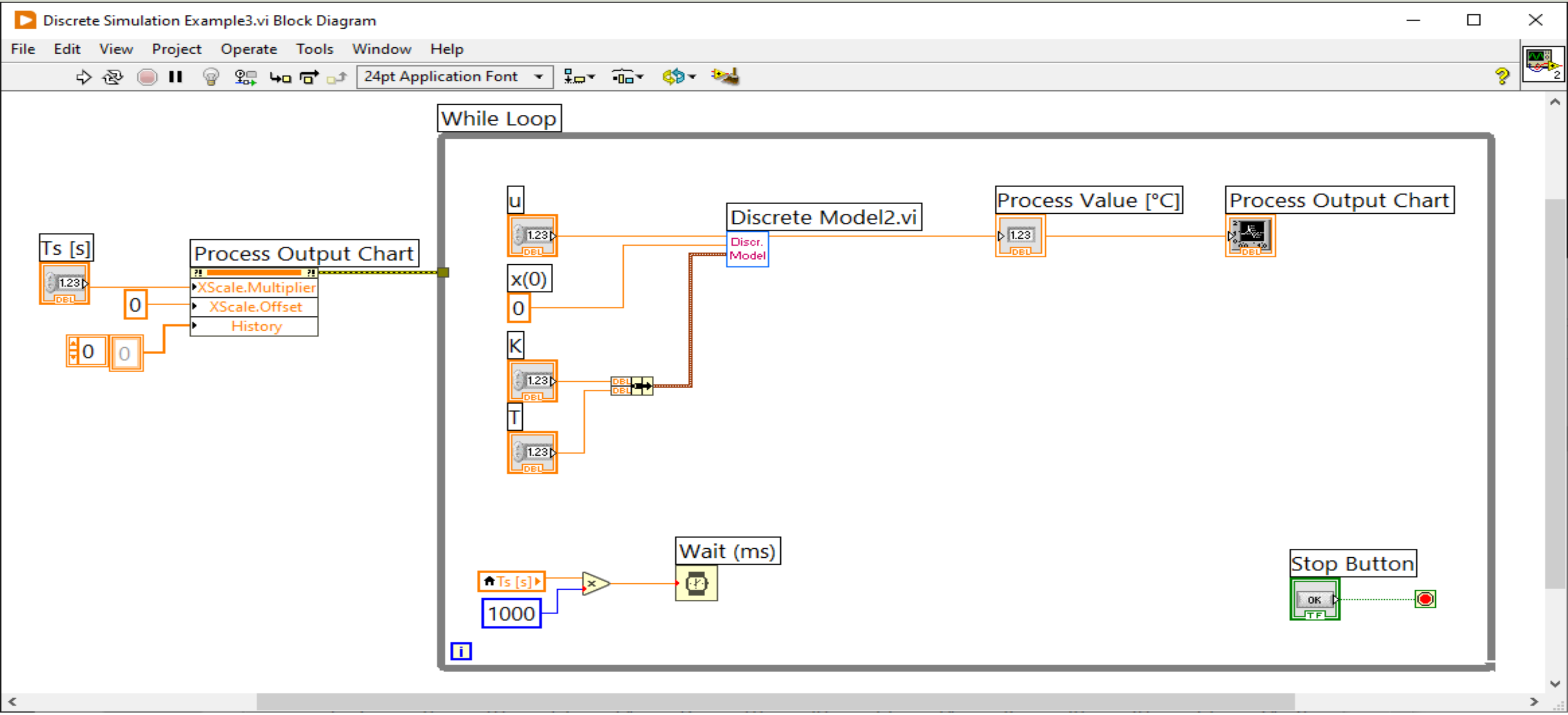




Simulation in LabVIEW



Code



Discrete Simulation Example3.vi Block Diagram
File
Edit
View
Project
Operate
Tools
Window
Help
24pt Application Font
While Loop
Ts [s]
Process Output Chart
XScale.Multiplier
XScale.Offset
History
u
x(0)
K
T
Discrete Model2.vi
Discr. Model
Process Value [°C]
Process Output Chart
Wait (ms)
Ts [s]
1000
Stop Button
OK

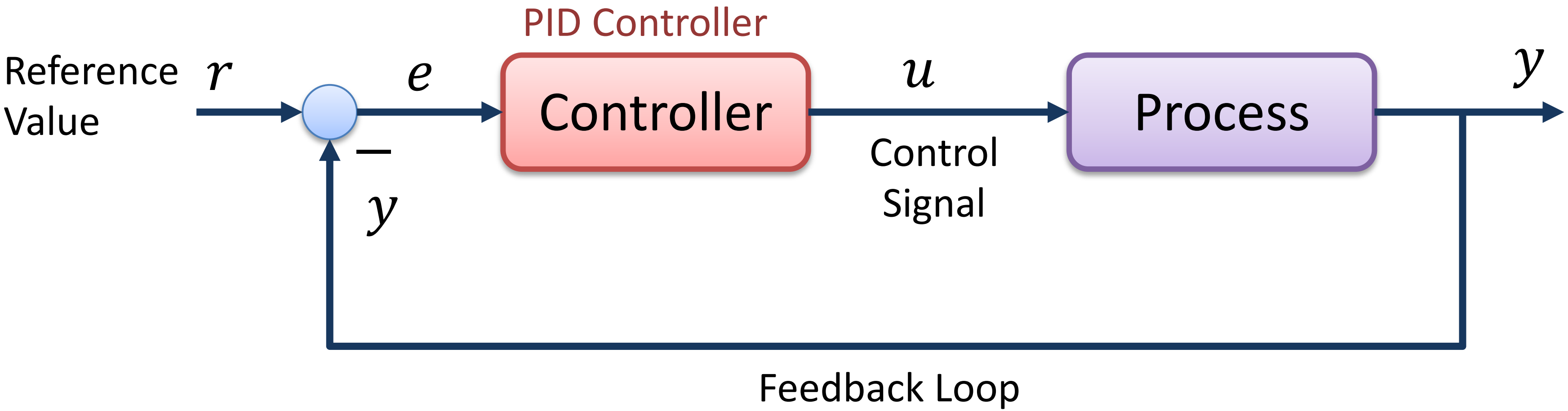
PID Controller

Hans-Petter Halvorsen

Table of Contents

Control System

The purpose with a Control System is to Control a Dynamic System, e.g., an industrial process, an airplane, a self-driven car, etc. (a Control System is “everywhere” today)



PID Controller

$$u(t) = K_p e + \frac{K_p}{T_i}\int_0^t e\,d\tau + K_p T_d \dot{e}$$

	P	I	D
	Proportional Gain	Integral Time	Derivative Time
Tuning Parameters:	K_p	T_i	T_d

PI Controller

Very often we just need a PI Controller:

$$u(t) = K_p e + \frac{K_p}{T_i}\int_0^t e d\tau$$

Discrete PI Controller that we can implement in different programming languages:

$$e_k = r_k - y_k$$

$$u_k = u_{k-1} + K_p(e_k - e_{k-1}) + \frac{K_p}{T_i}T_s e_k$$

Discrete PI Controller

We start with the continuous PI Controller:

$$u(t) = K_p e + \frac{K_p}{T_i}\int_0^t e d\tau$$

We derive both sides in order to remove the Integral:

$$\dot{u} = K_p \dot{e} + \frac{K_p}{T_i} e$$

We can use the Euler Backward Discretization method:

$$\dot{x} \approx \frac{x(k) - x(k-1)}{T_s}$$

Where T_s is the Sampling Time

Then we get:

$$\frac{u_k - u_{k-1}}{T_s} = K_p \frac{e_k - e_{k-1}}{T_s} + \frac{K_p}{T_i} e_k$$

Finally, we get:

$$u_k = u_{k-1} + K_p(e_k - e_{k-1}) + \frac{K_p}{T_i} T_s e_k$$

Where $e_k = r_k - y_k$

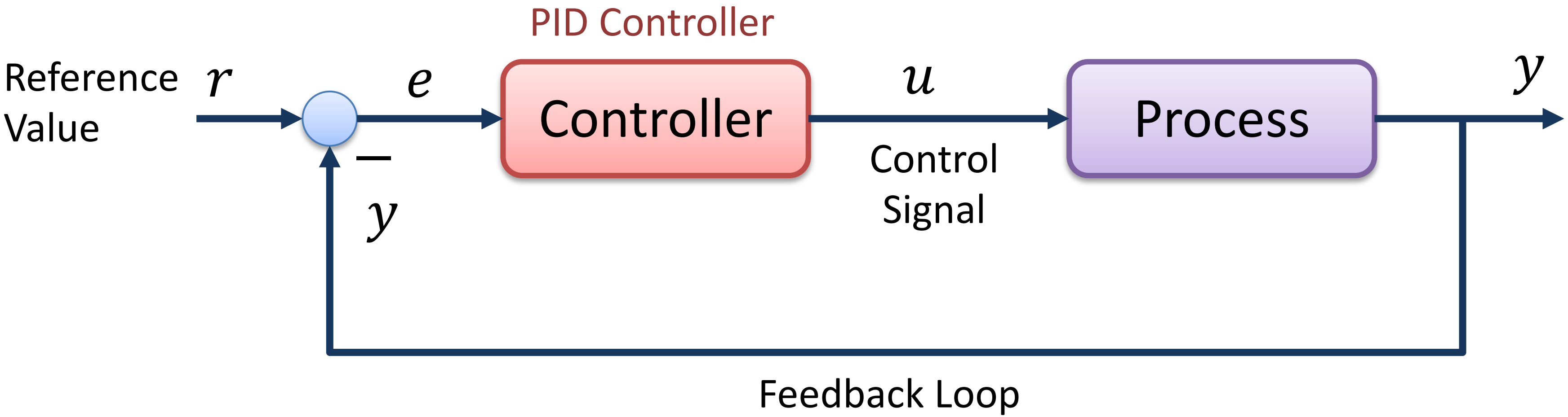
Control System

Hans-Petter Halvorsen

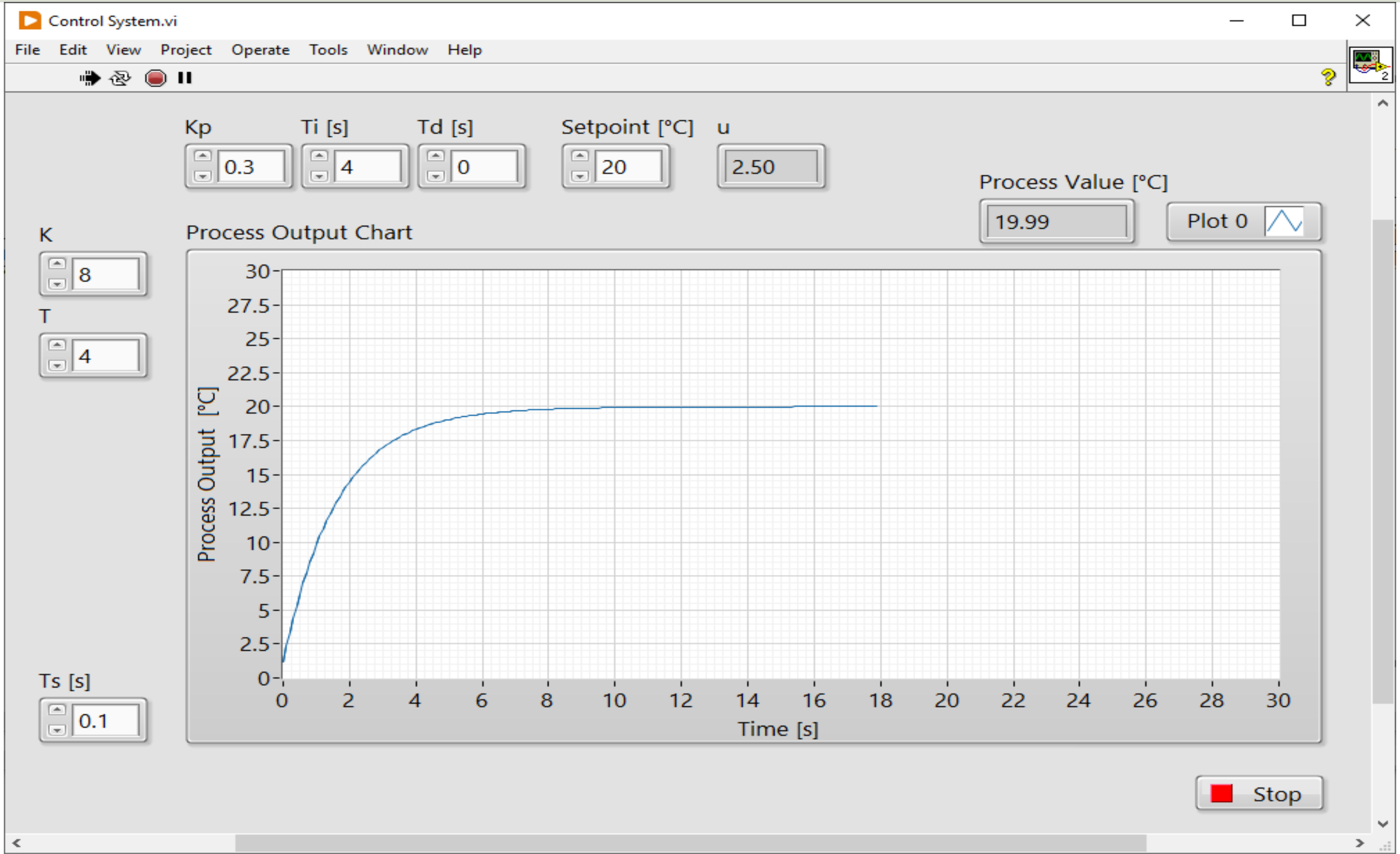
Table of Contents

Control System

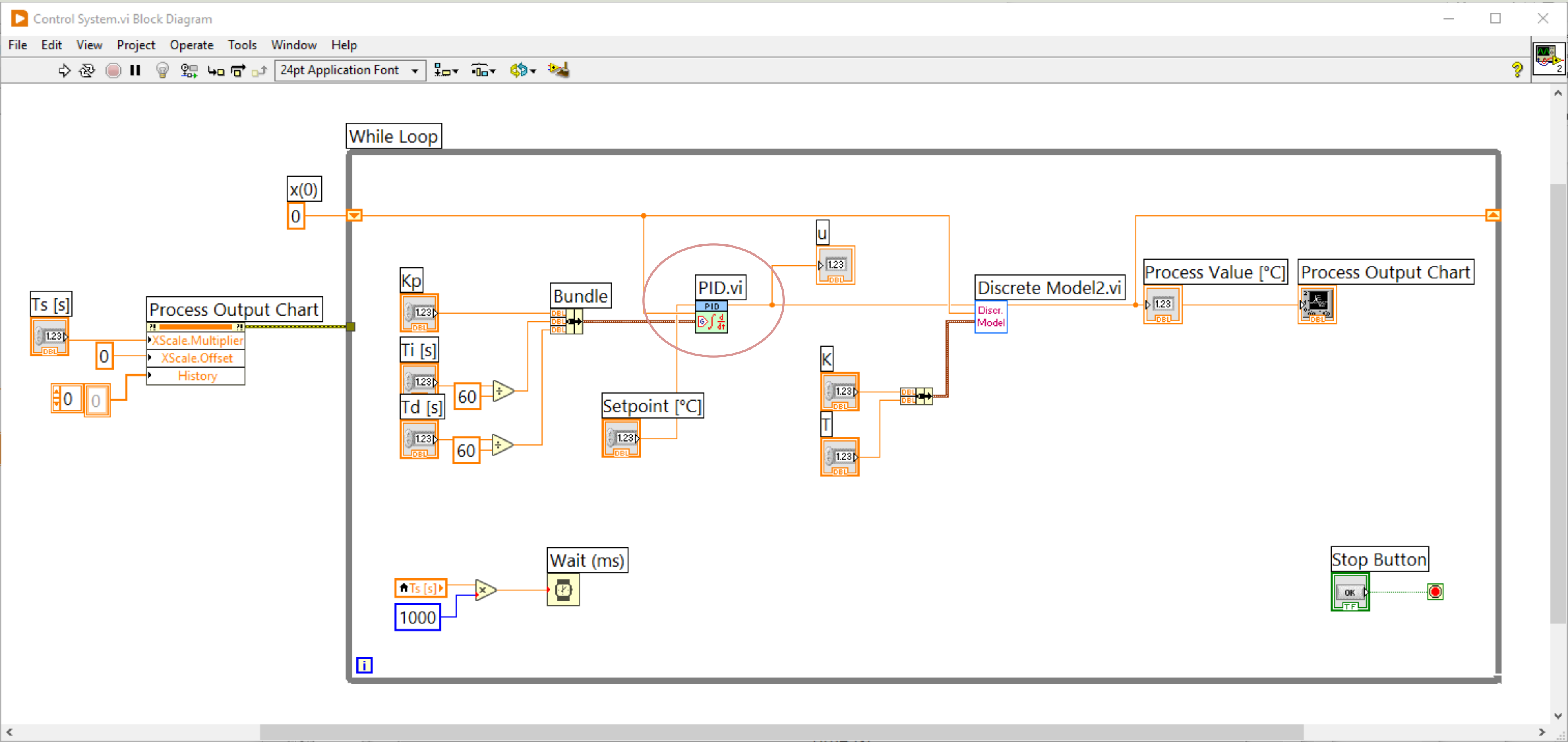
The purpose with a Control System is to Control a Dynamic System, e.g., an industrial process, an airplane, a self-driven car, etc. (a Control System is “everywhere” today)



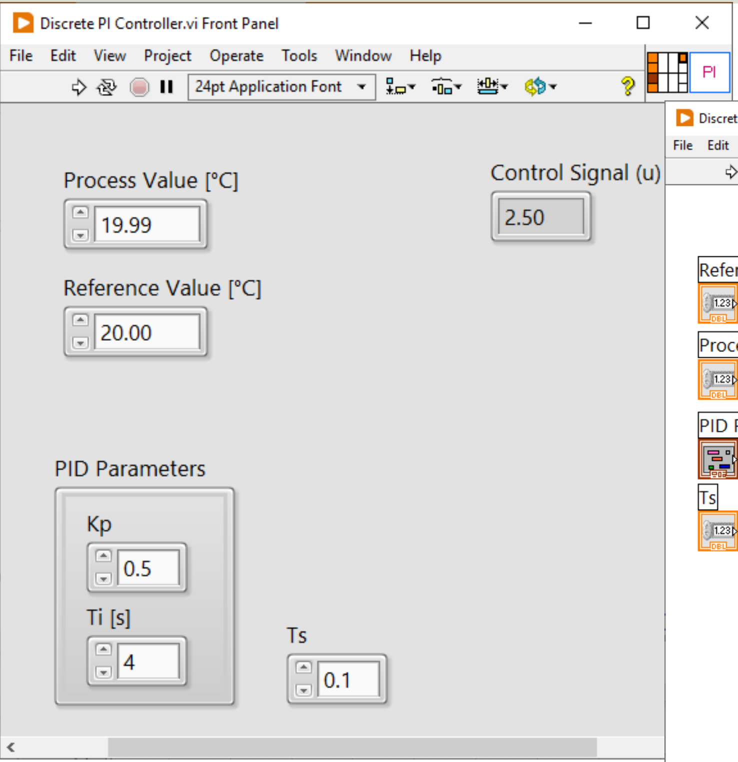
Control System in LabVIEW



Built-in PID Controller



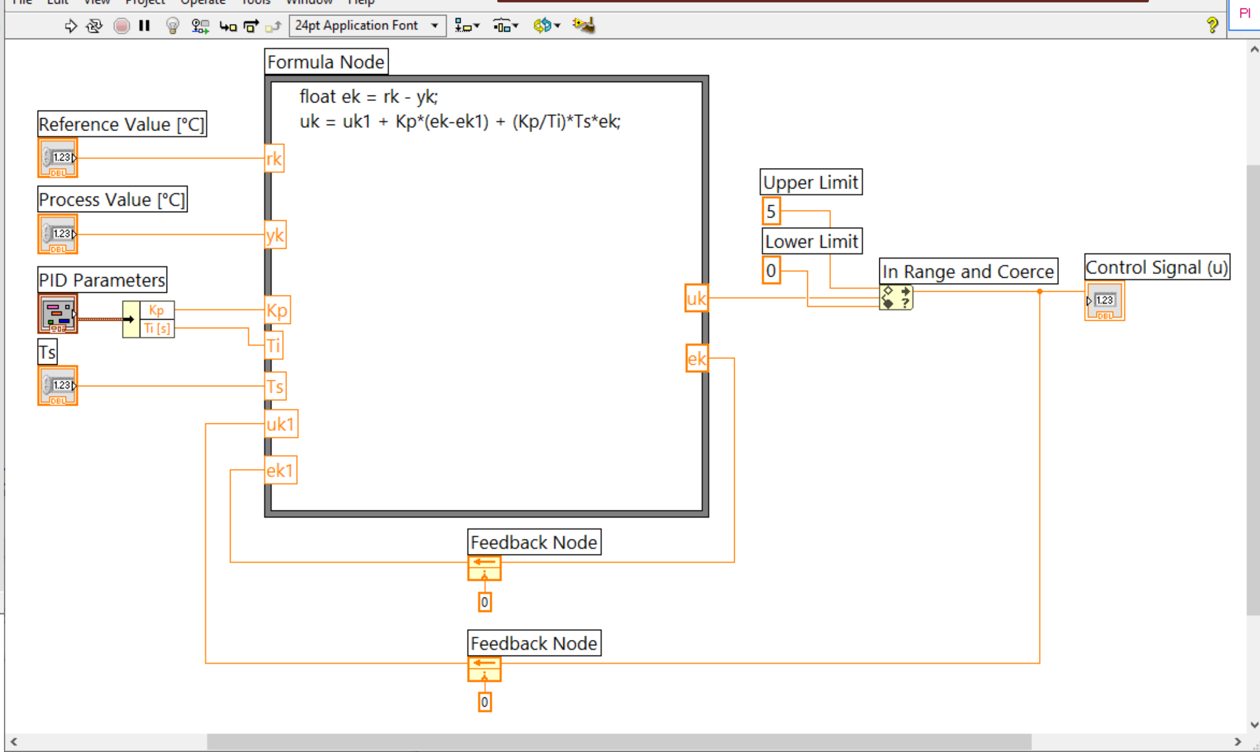
Discrete PI Controller



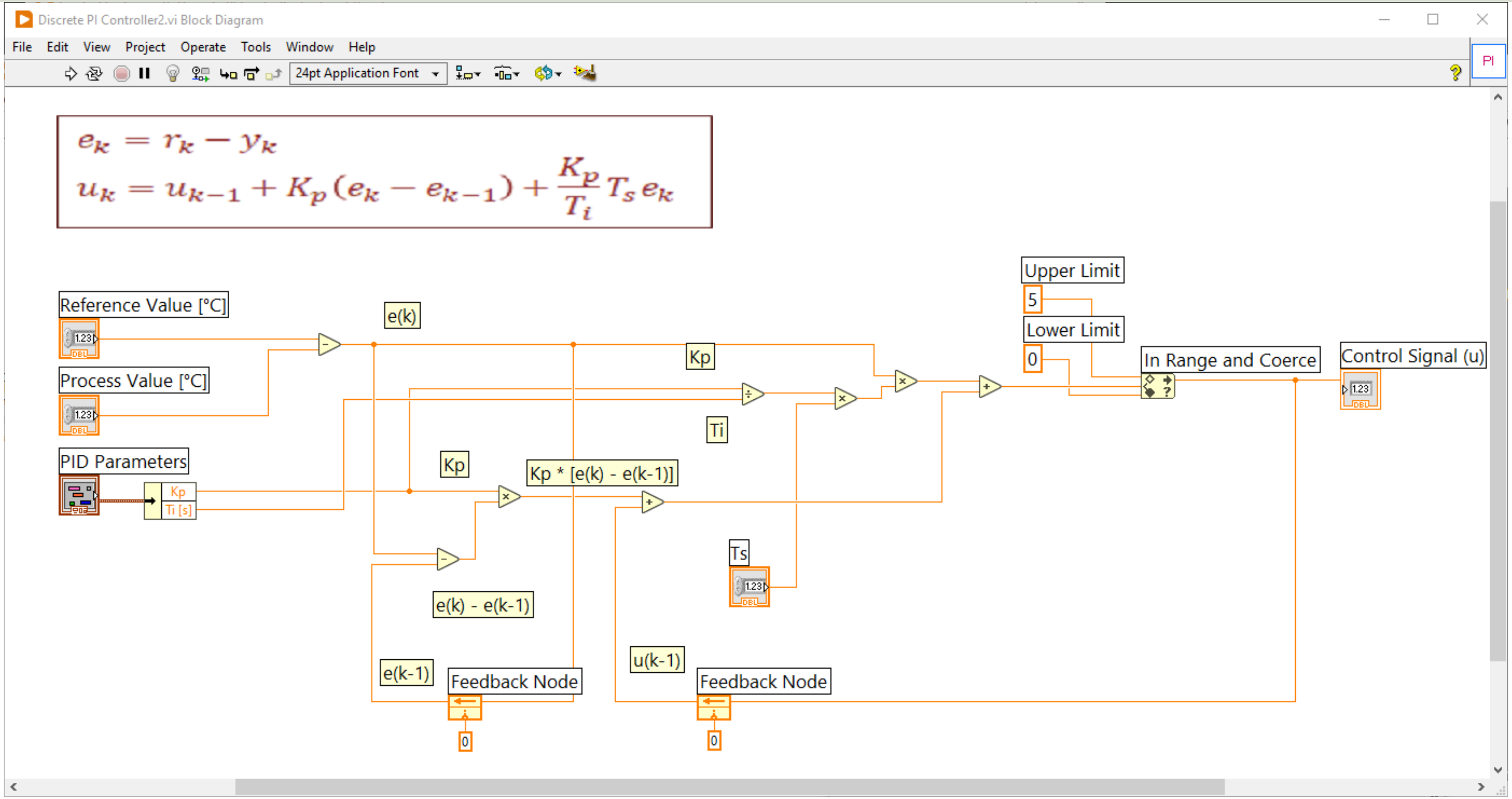
Discrete PI Algorithm:

$$e_k = r_k - y_k$$

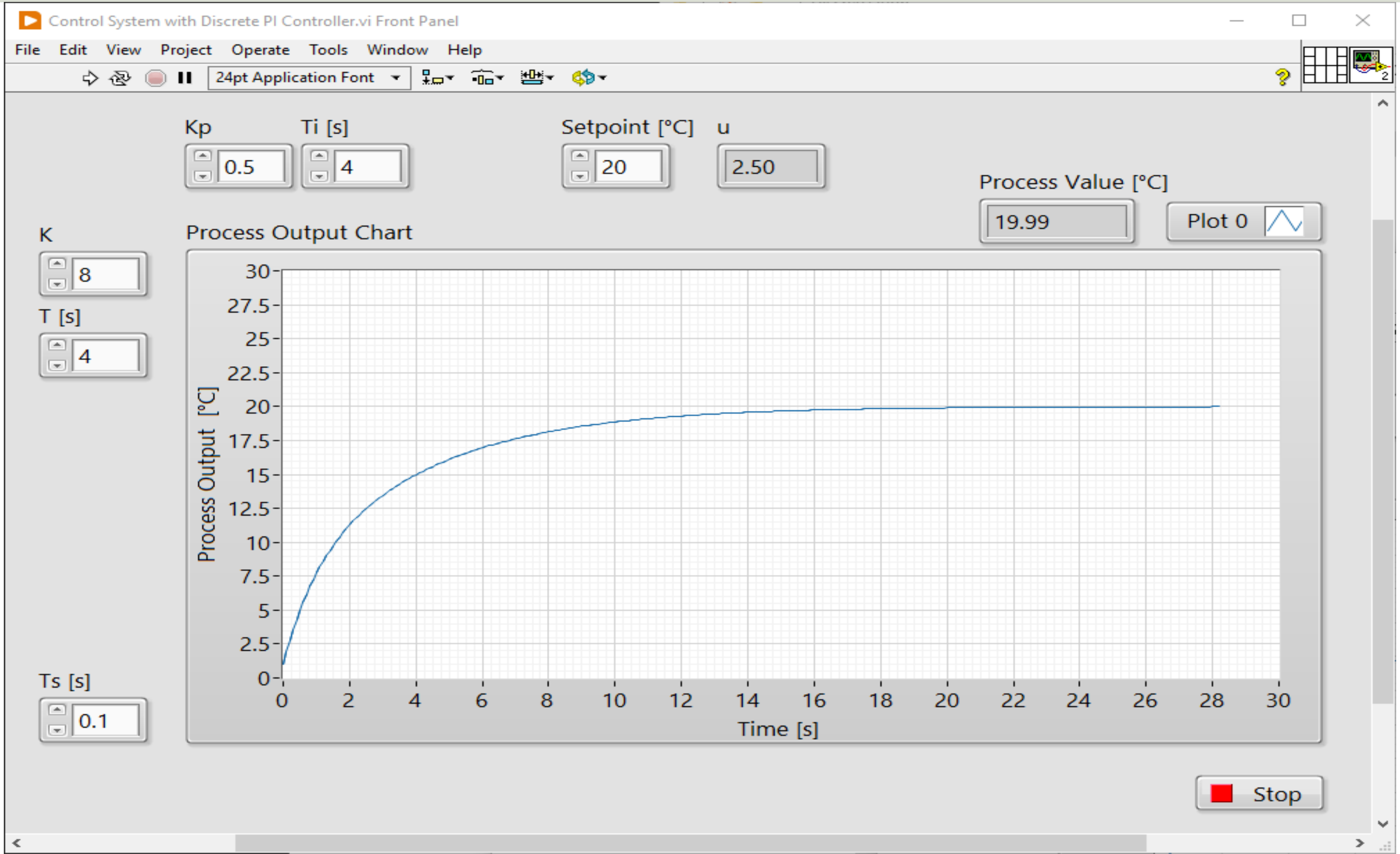
$$u_k = u_{k-1} + K_p(e_k - e_{k-1}) + \frac{K_p}{T_i} T_s e_k$$



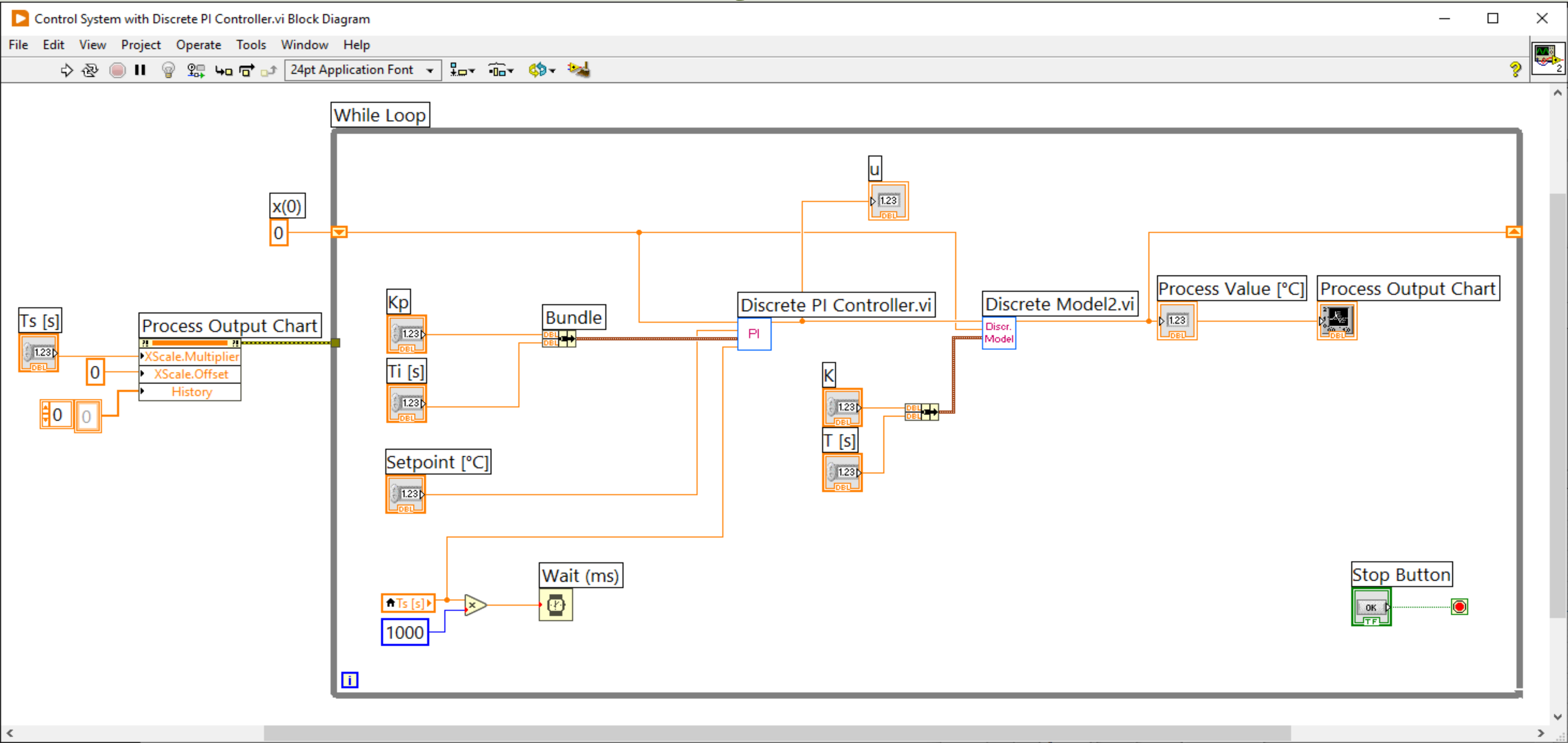
Discrete PI Controller (Alternative Solution)



Control System in LabVIEW



Control System Code



Summary

- A Basic Control System has been made using LabVIEW
- Lots of Improvements can be made, e.g.,:
  - Improve GUI
    - More Features/Functionality, More Intuitive and more user-friendly
  - Improve Code Structure, e.g., use a State Machine principle
  - Make a more robust PI(D) Controller
  - Use and Test with a more complicated Process/Model
  - Find better PI(D) Parameters using different Tuning methods, e.g., Ziegler-Nichols, Skogestad, etc.
  - Connect and Control a Real Process using a DAQ Device
  - Etc.

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog