

<https://www.halvorsen.blog>



Hardware-in-the Loop (HIL) Simulation and Testing

Hans-Petter Halvorsen

HIL Simulation and Testing

- Hardware-in-the-Loop (HIL) Simulation is a technique that is used for Testing Control Systems, Embedded Systems, etc.
- Carrying out a HIL Simulation to Test the Control System, Embedded System, etc. is called HIL Testing.

<http://www.hil-simulation.com/home/hil-testing.html>

https://en.wikipedia.org/wiki/Hardware-in-the-loop_simulation

HIL Simulation and Testing

- The machine or physical part of the system (which we call the *plant*) is normally connected with the *control system*, through actuators and sensors.
- With HIL testing the plant is replaced by a simulation of the plant (which we call the *HIL simulator*).
- If the HIL simulator is designed well, it will accurately mimic the plant, and can be used to test the control system.

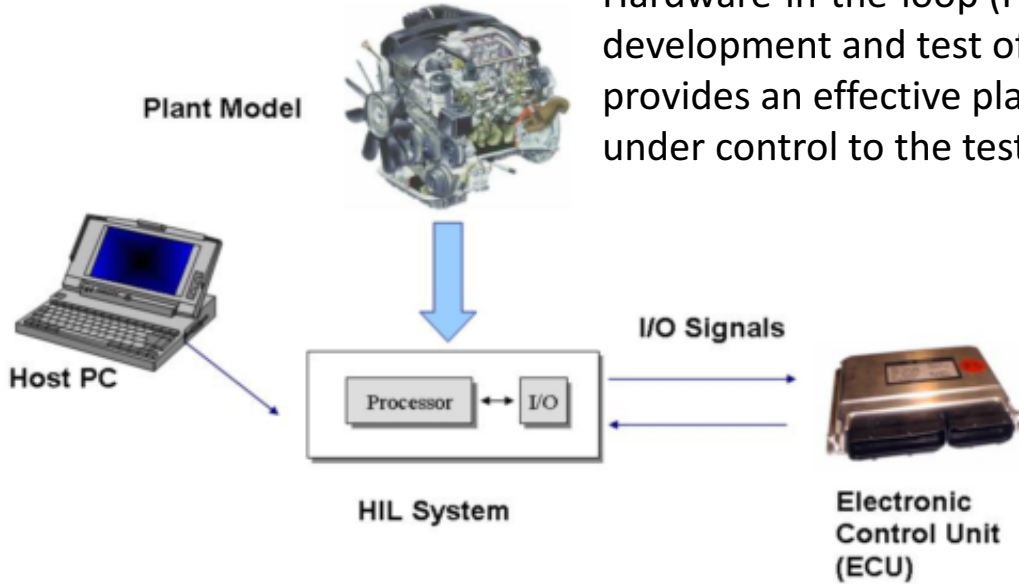
HIL Simulation and Testing

- Hardware-in-the-loop (HIL) simulation is a technique that is used in the development and **test of complex process systems**
- The HIL simulation includes a **mathematical model** of the process and a hardware device/ECU (Electronic Control Unit) you want to test, e.g. an industrial PID controller we will use in our example. The hardware device is normally an **embedded system**
- The main purpose with the HIL Simulation is to **test the hardware device on a simulator** before we implement it on the real process
- It is also very useful for **training** purposes, i.e., the process operator may learn how the system works and operate by using the hardware-in-the-loop simulation
- Another benefit of Hardware-In-the-Loop is that testing can be done **without damaging equipment** or endangering lives.

The main purpose with the HIL Simulation is to test the hardware device on a simulator before we implement it on the real process.

HIL Simulation and Testing

Hardware-in-the-loop (HIL) simulation is a technique that is used in the development and test of complex process systems. HIL simulation provides an effective platform by adding the complexity of the plant under control to the test platform.



The complexity of the plant under control is included in test and development by adding a mathematical representation of all related dynamic systems. These mathematical representations are referred to as the “plant simulation.”

Hardware-In-the-Loop is a form of real-time simulation. Hardware-In-the-Loop differs from real-time simulation by the addition of a real component in the loop. This component may be an “Electronic Control Unit” (ECU).

ECU - Electronic Control Unit

- In automotive electronics, electronic control unit (ECU) is a generic term for any embedded system that controls one or more of the electrical system or subsystems in a motor vehicle.
- ECU could also be an embedded PID Controller, etc. (which we will use in our example)

Examples of Industrial Control Systems (ICS)

Industrial Control Systems are computer controlled systems that monitor and control industrial processes that exist in the physical world



Programmable Automation Controller (PAC) **4**

National Instruments cRIO LabVIEW **1**



PC based Control System/SCADA System (Supervisory Control And Data Acquisition) **5**



I/O Module

Industrial PID Controller



2 Distributed Control Systems (DCS)



Controller I/O Modules

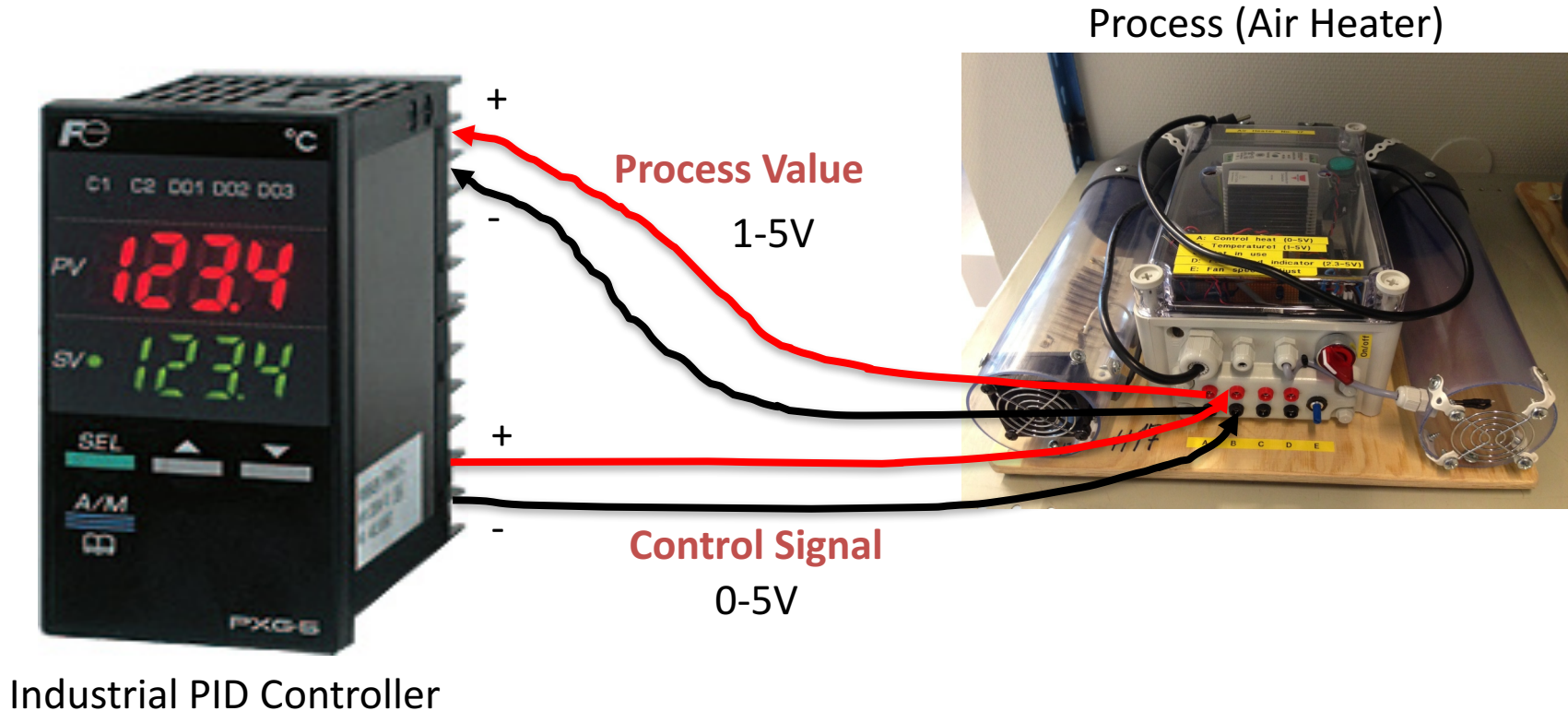
DeltaV from Emerson

PLC (Programmable Logic Controller) **3**

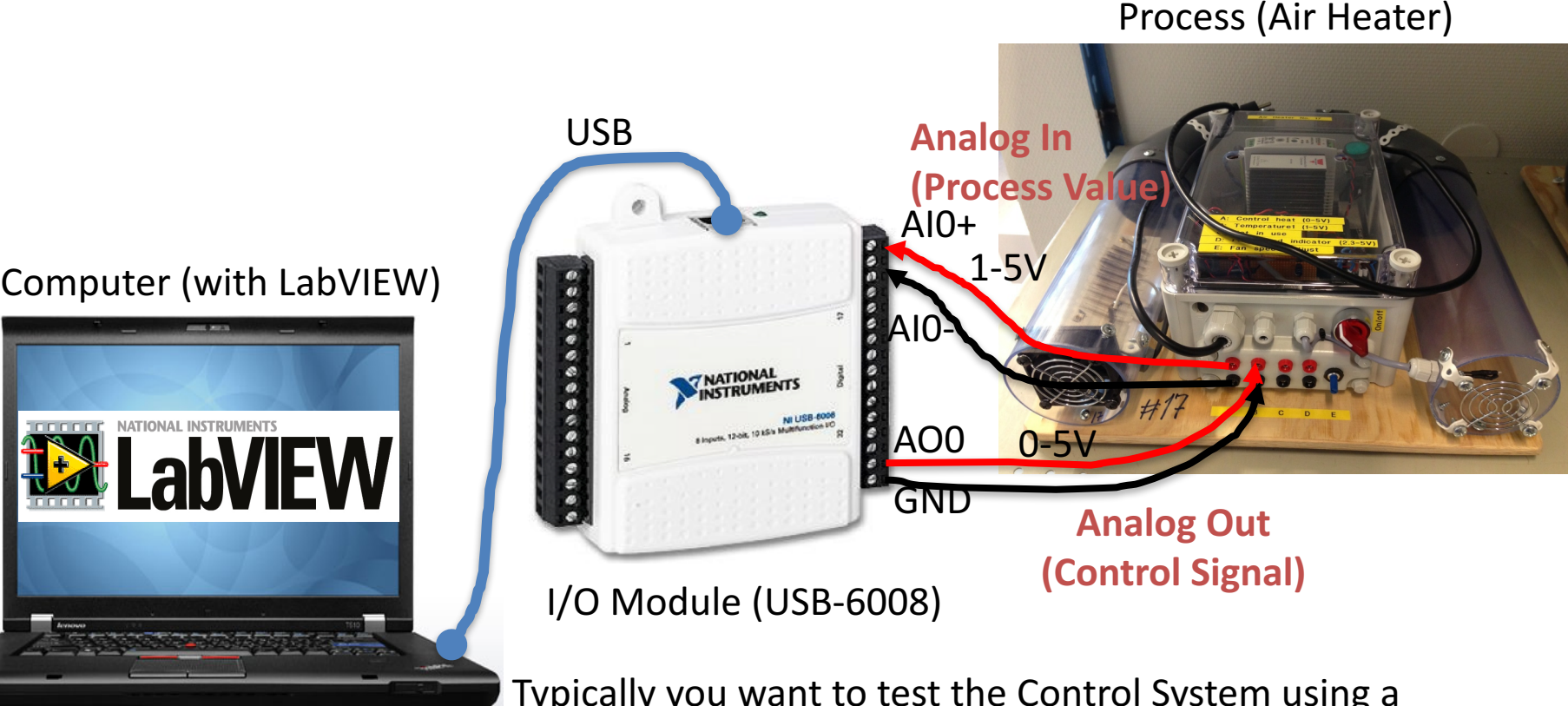


Siemens PLC

Example of Industrial Single Loop PID Control



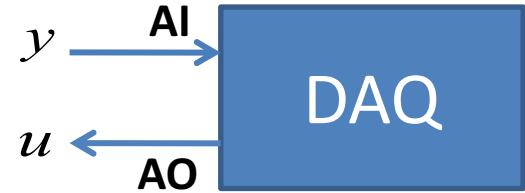
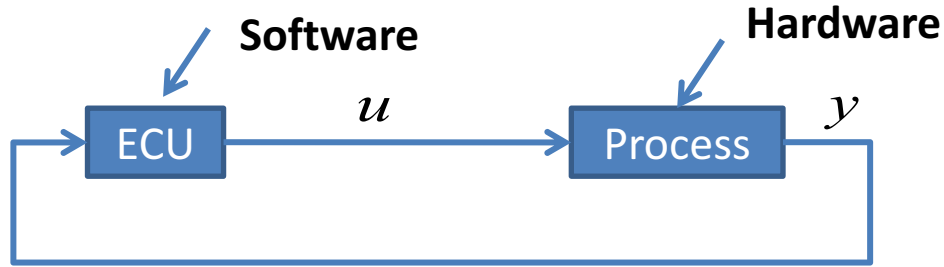
Example of PC-based Control System



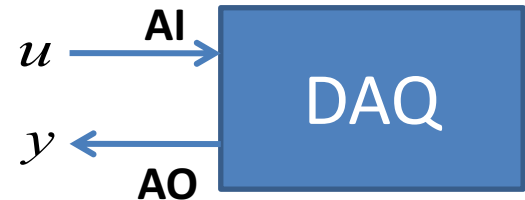
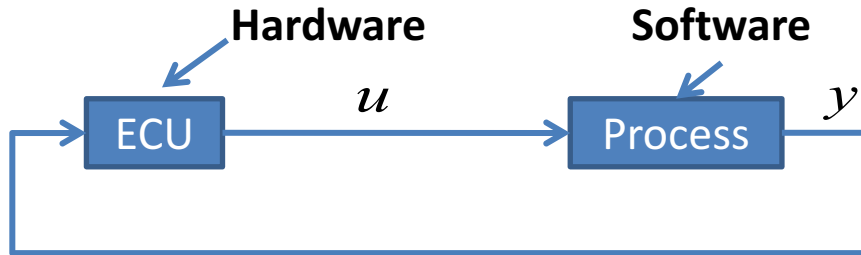
PID Control and Monitoring

Typically you want to test the Control System using a Mathematical Model before you use it on the Real System

Traditional Process Control using Software for Implementing the Control System



HIL Simulation and Testing





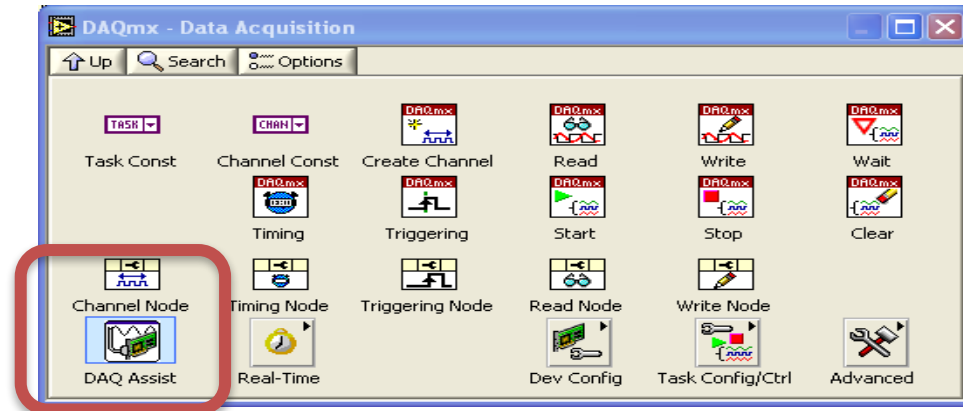
HIL Simulation and Testing Example

Hans-Petter Halvorsen, M.Sc.

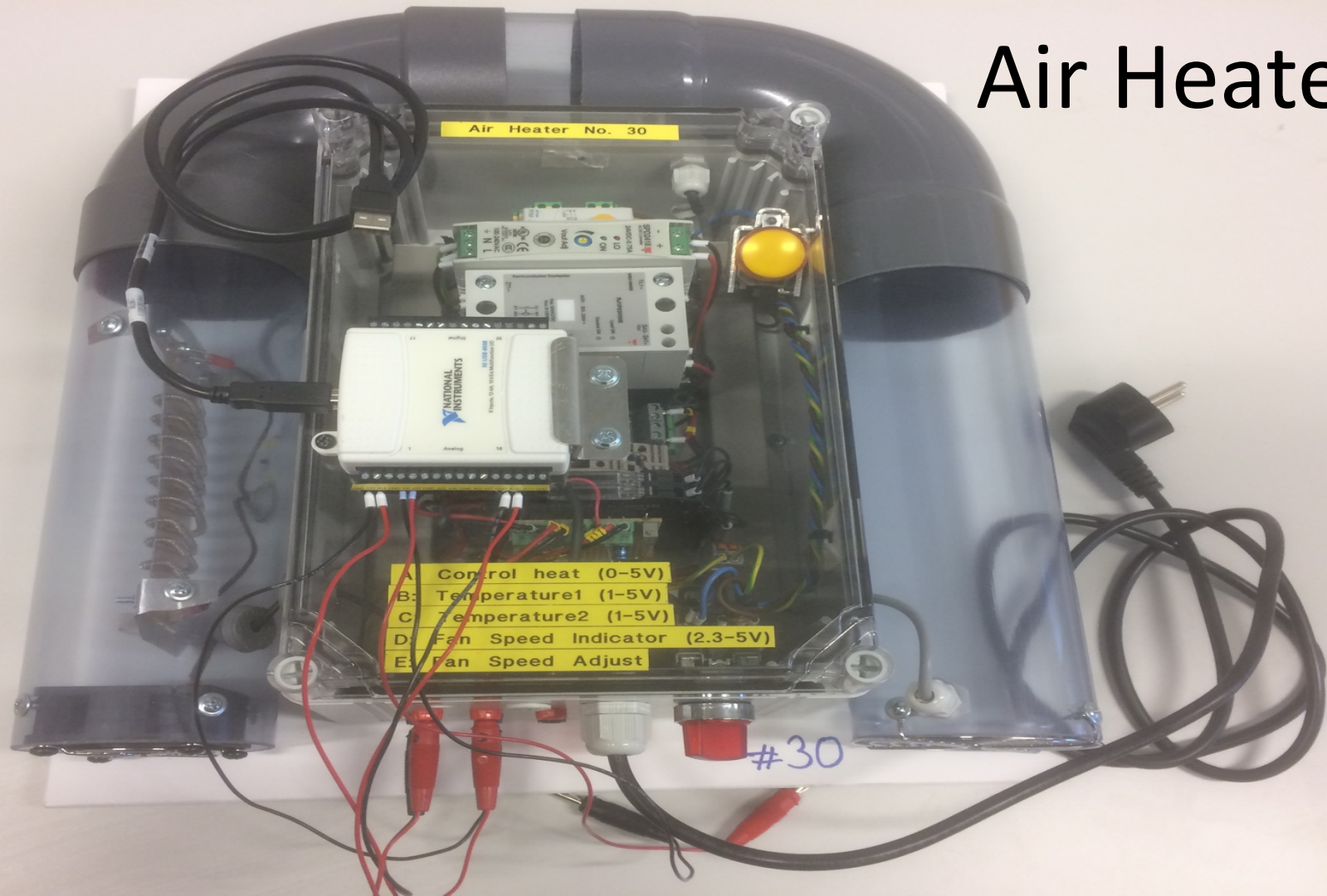
Necessary Software



- LabVIEW
- LabVIEW Control & Simulation Module
- DAQmx Driver Software



Air Heater



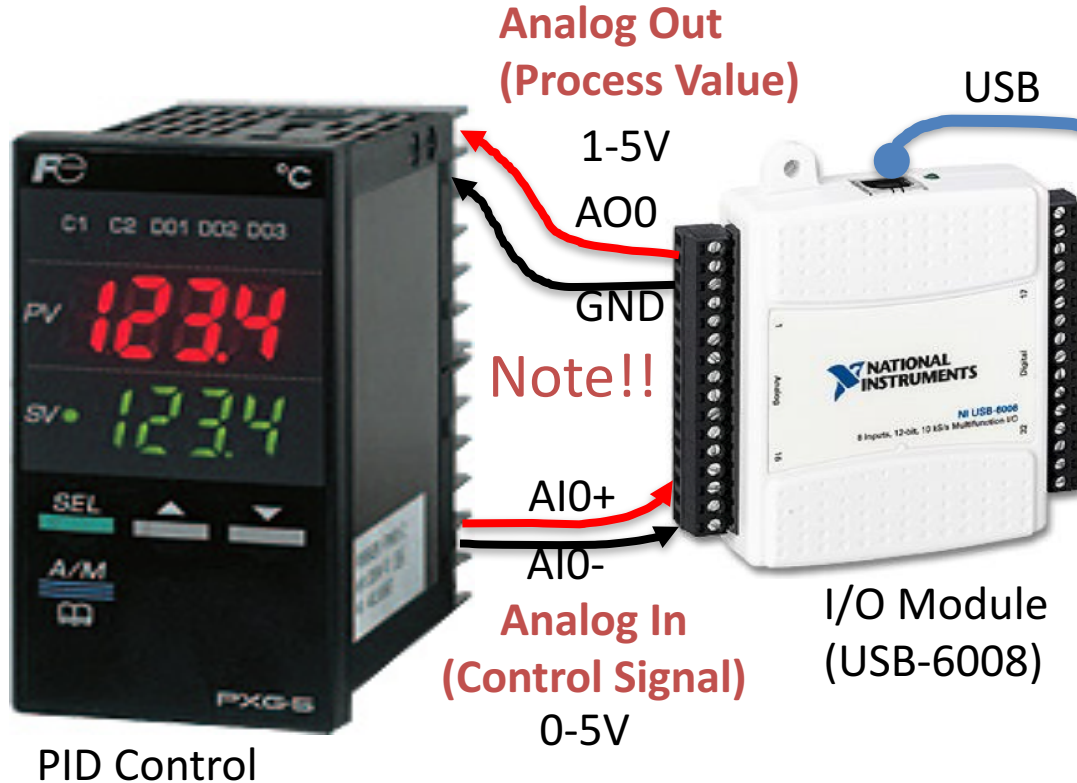
Fuji PXG5 PID Controller



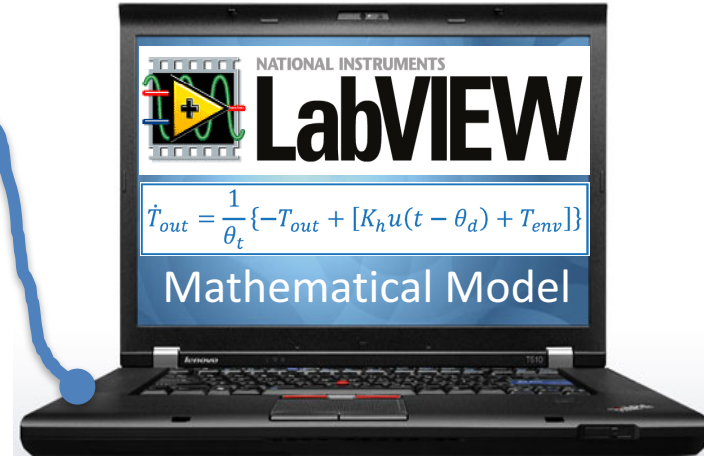
HIL Example

- Typically, a simulator communicates with an “ECU” (“Electronic Control Unit”) via ordinary I/O. Such a system - where the real controller is controlling a simulated process is denoted Hardware-in-the-loop (HIL) simulation.
- The main purpose of this Example is to test the hardware device on a simulator before we implement it on the real process.
- If the mathematical model used in the simulator is an accurate representation of the real process, you may even tune the controller parameters (e.g. the PID parameters) using the simulator.
- We will test the Fuji PGX5 PID controller on a model, and if everything is OK we will implement the controller on the real system.

Example of HIL Simulation and Testing

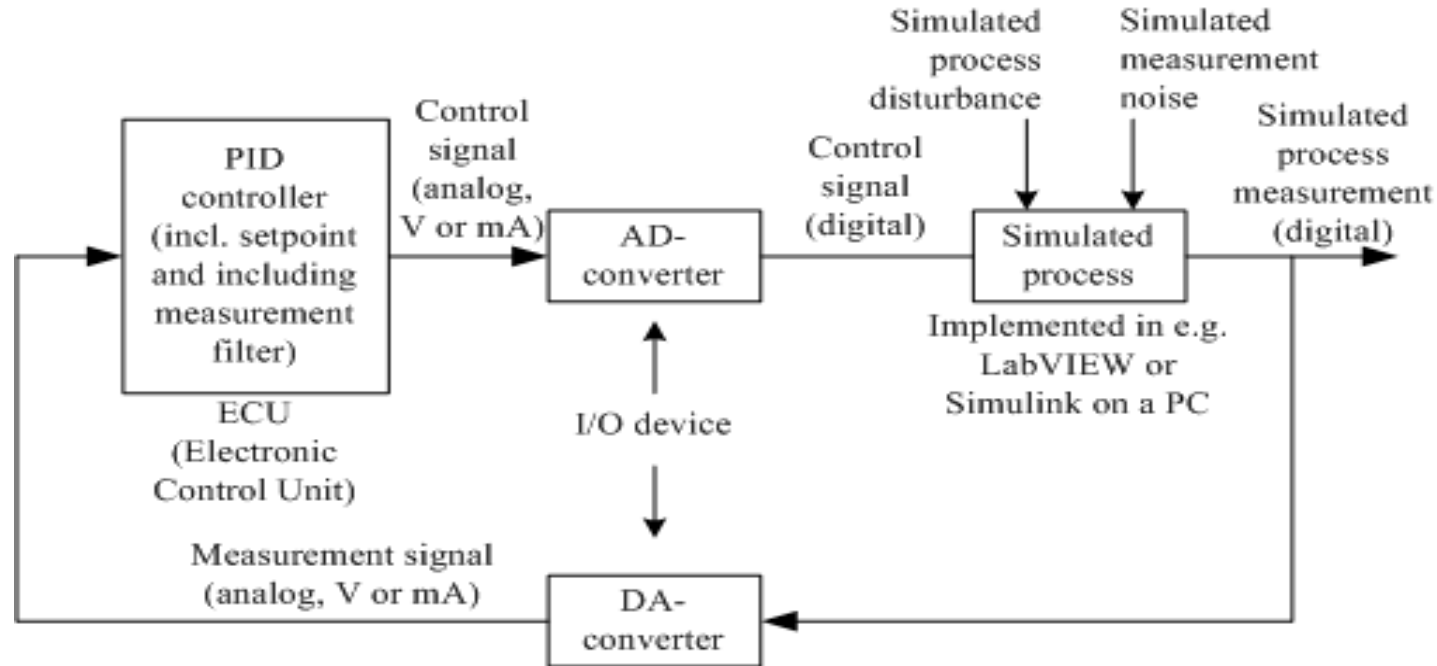


Model of Process (Air Heater)



Computer (with LabVIEW)

HIL Simulation and Testing

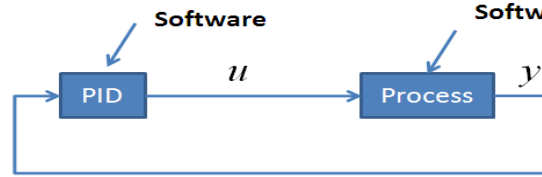


The figure illustrates the principle of testing a control system by replacing the physical system (or process) to be controlled by a simulated system. The controller is assumed to be a PID controller, but the figure applies to any controller function.

HIL Simulation Example - Step-by-step

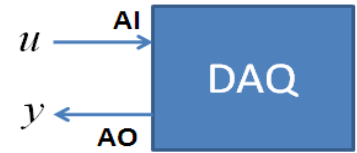
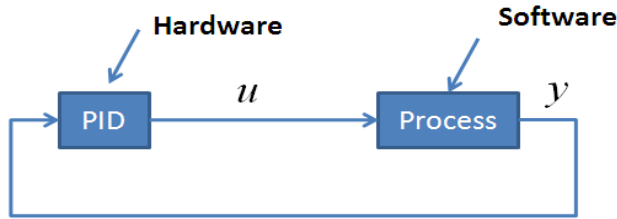
1

Step 1: Ordinary Software Simulation:



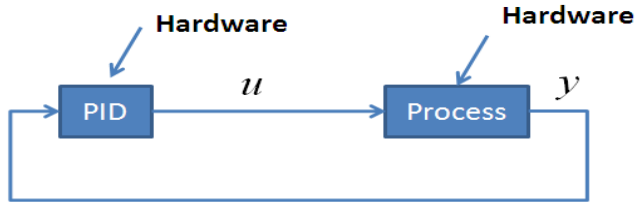
2

Step 2: HIL Simulation:

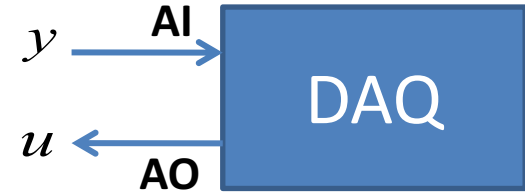
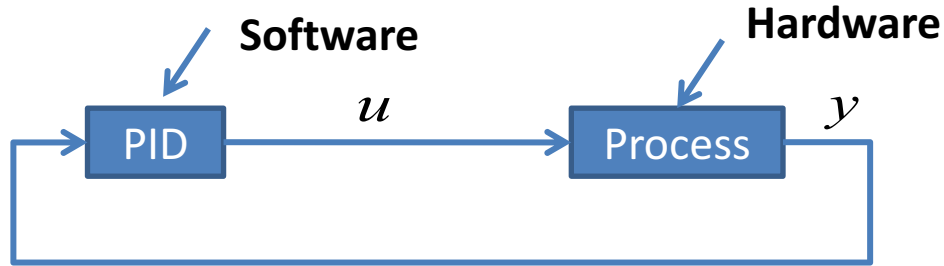


3

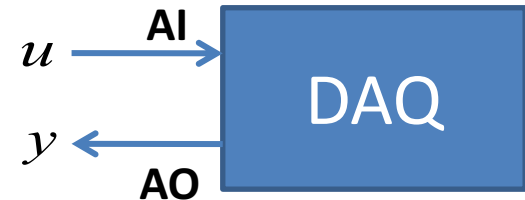
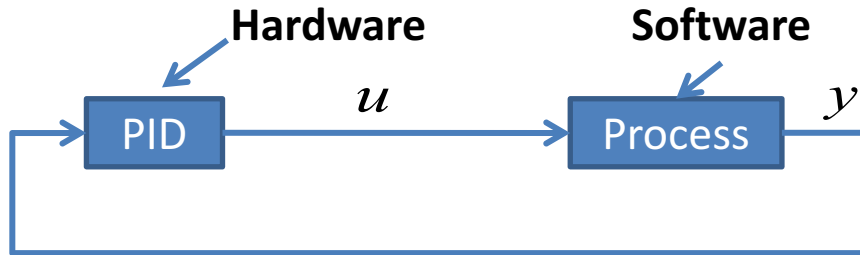
Step 3: Running the Real System:



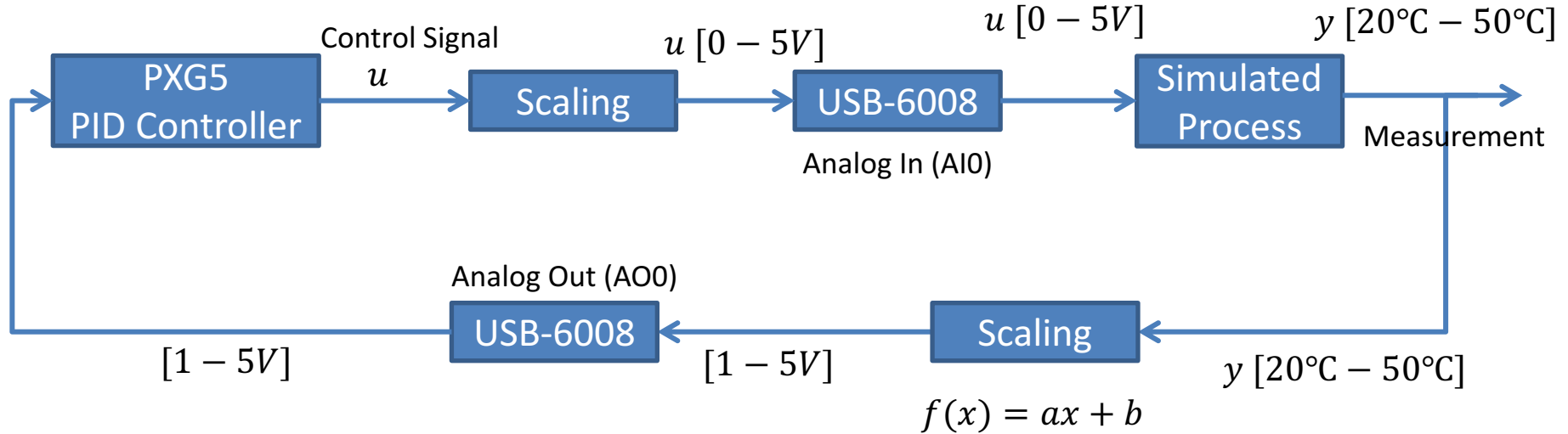
Traditional Process Control using Software for Implementing the Control System



HIL Simulation and Testing



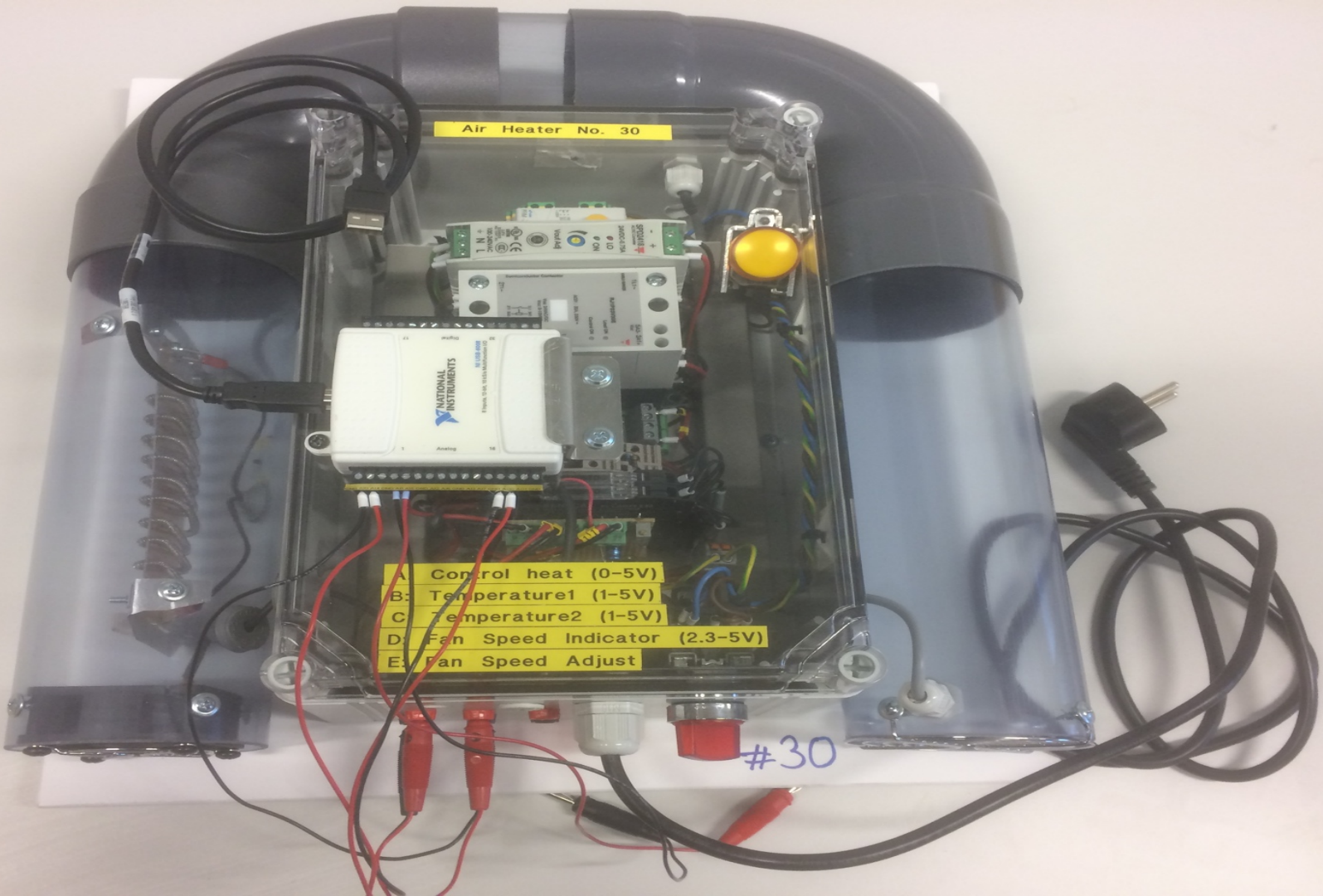
HIL Simulation using PXG5 PID Controller





Mathematical Model

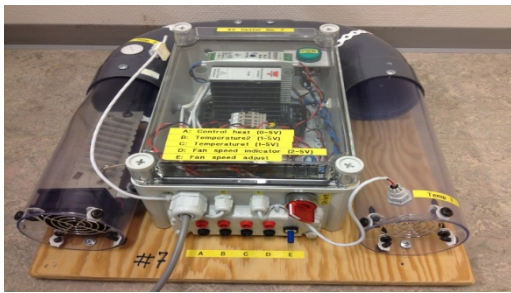
Hans-Petter Halvorsen, M.Sc.



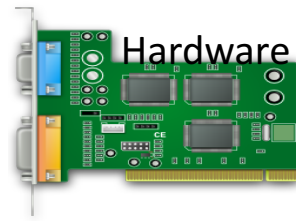
Air Heater No. 30

- A: Control heat (0-5V)
- B: Temperature1 (1-5V)
- C: Temperature2 (1-5V)
- D: Fan Speed Indicator (2.3-5V)
- E: Fan Speed Adjust

#30



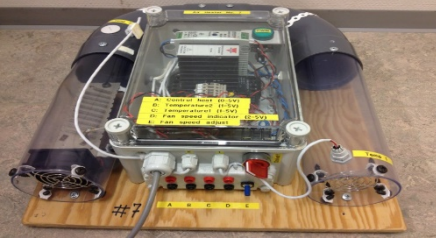
Air Heater



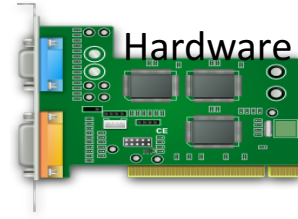
- **Heater:** The air is heated by an electrical heater. The supplied power is controlled by an external voltage signal in the range **0 - 5 V** (min power, max power).
- **Temperature sensors:** A Pt100 temperature sensor. The range is **1 - 5 V**, and this voltage range corresponds to the temperature range **20 - 50°C** (with a linear relation).

Example of Mathematical model:

$$\dot{T}_{out} = \frac{1}{\theta_t} \{-T_{out} + [K_h u(t - \theta_d) + T_{env}]\}$$



Air Heater Mathematical Model



$$\dot{T}_{out} = \frac{1}{\theta_t} \{-T_{out} + [K_h u(t - \theta_d) + T_{env}]\}$$

Where:

- T_{out} is the air temperature at the tube outlet
- $u [V]$ is the control signal to the heater
- $\theta_t [s]$ is the time-constant
- $K_h [deg C / V]$ is the heater gain
- $\theta_d [s]$ is the time-delay representing air transportation and sluggishness in the heater
- T_{env} is the environmental (room) temperature. It is the temperature in the outlet air of the air tube when the control signal to the heater has been set to zero for relatively long time (some minutes)

We will use the following values in the Simulations:

$$\theta_t = 22 \text{ sec}$$

$$\theta_d = 2 \text{ sec}$$

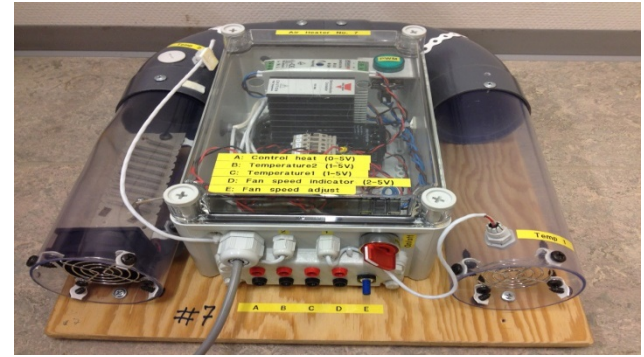
$$K_h = 3.5 \frac{^{\circ}C}{V}$$

$$T_{env} = 21.5 \text{ }^{\circ}C$$

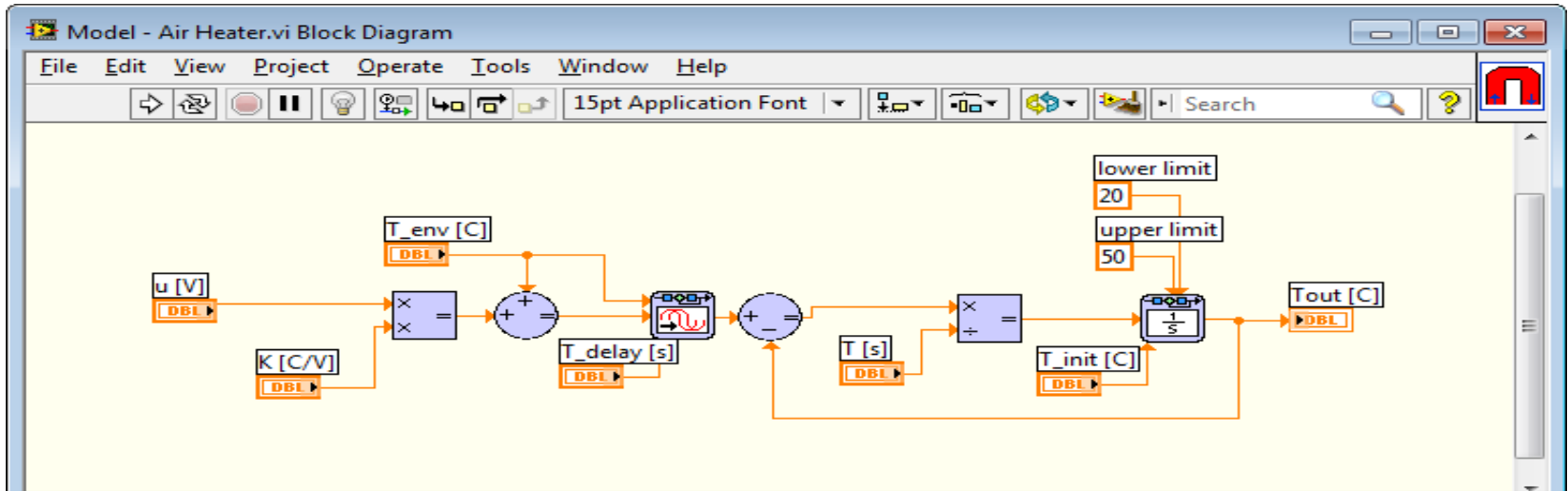
Air Heater in LabVIEW

Heater: The air is heated by an electrical heater. The supplied power is controlled by an external voltage signal in the range 0 - 5 V (min power, max power).

Temperature sensors: Two Pt100 temperature elements are available. The range is 1 - 5 V, and this voltage range corresponds to the temperature range 20 - 50°C (with a linear relation).



Example of Mathematical Model of Air Heater implemented in LabVIEW:



Note! This model is implemented in a so-called “**Simulation Subsystem**” (which is recommended!!!)

Model Simulation Example (Note! No PID Control in this Example)



Step Response - Air Heater.vi Block Diagram

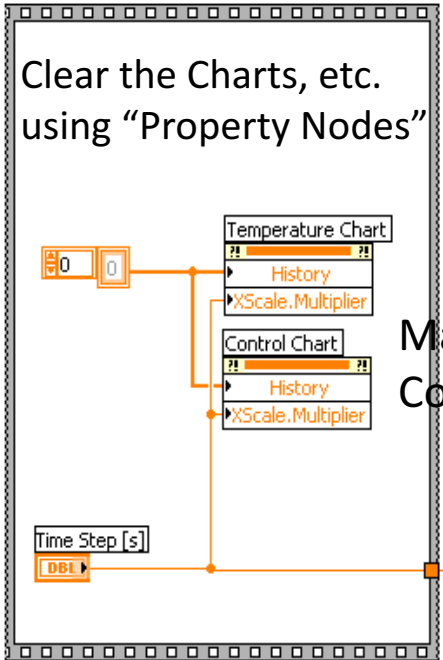
File Edit View Project Operate Tools Window Help

13pt Application Font

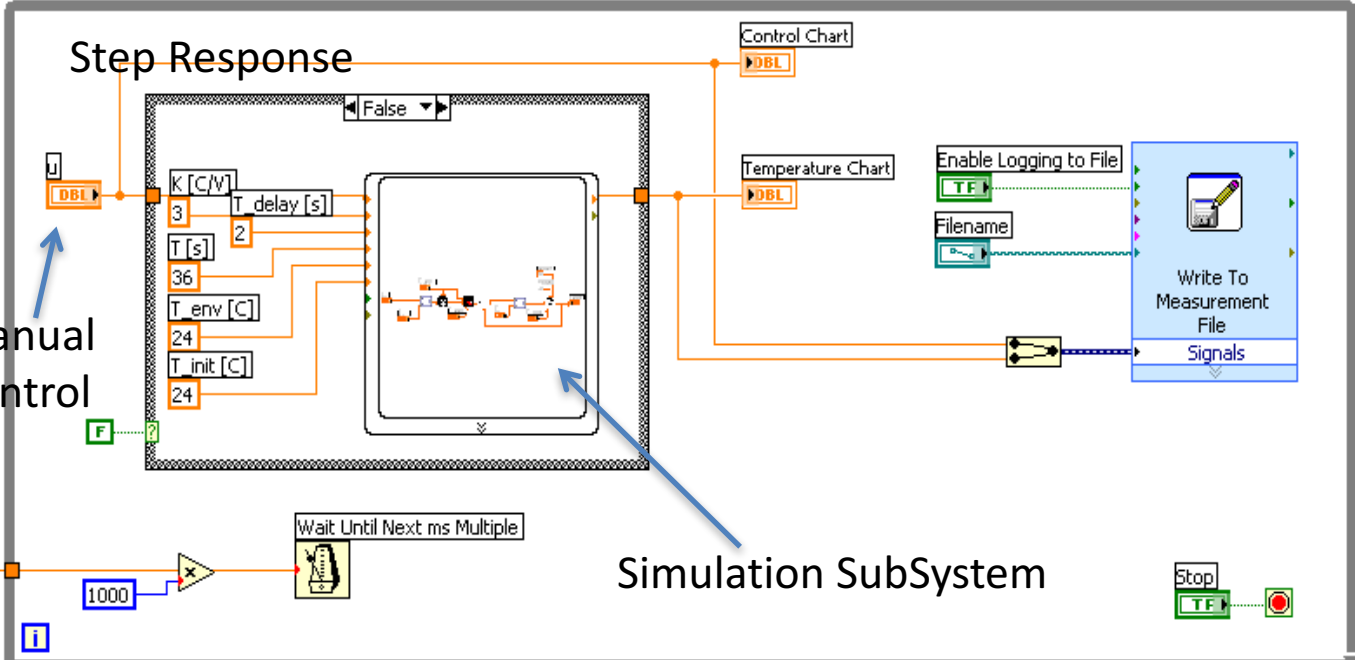
Search

Step Resp.

Note! This is just an Example! – You should create your own personal Application



Manual Control



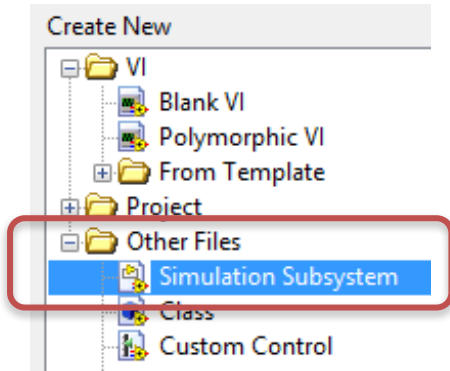
Simulation SubSystem

Initialization

Simulation Subsystem

A Way to structure your code, similar to SubVIs

This is the recommended way to do it! – You can easily reuse your Subsystems in different VIs and your code becomes more structured!



Select File -> New ..., Then choose “Simulation Subsystem”.

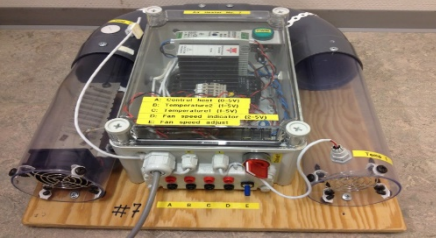
Create your Model within the Simulation Subsystem

DEMO

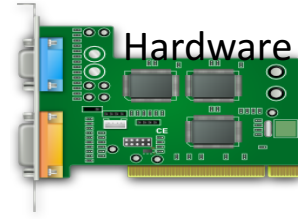


System Identification

Hans-Petter Halvorsen, M.Sc.



Air Heater Mathematical Model



$$\dot{T}_{out} = \frac{1}{\theta_t} \{-T_{out} + [K_h u(t - \theta_d) + T_{env}]\}$$

Where:

- T_{out} is the air temperature at the tube outlet
- $u [V]$ is the control signal to the heater
- $\theta_t [s]$ is the time-constant
- $K_h [deg C / V]$ is the heater gain
- $\theta_d [s]$ is the time-delay representing air transportation and sluggishness in the heater
- T_{env} is the environmental (room) temperature. It is the temperature in the outlet air of the air tube when the control signal to the heater has been set to zero for relatively long time (some minutes)

We will use the following values in the Simulations:

$$\theta_t = 22 \text{ sec}$$

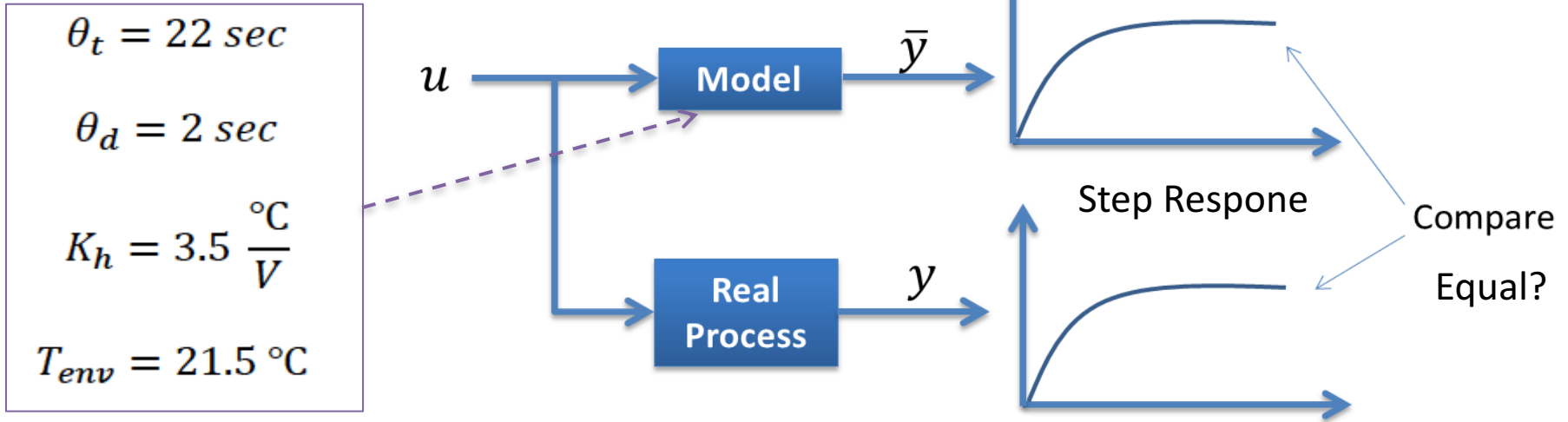
$$\theta_d = 2 \text{ sec}$$

$$K_h = 3.5 \frac{^{\circ}C}{V}$$

$$T_{env} = 21.5 \text{ }^{\circ}C$$

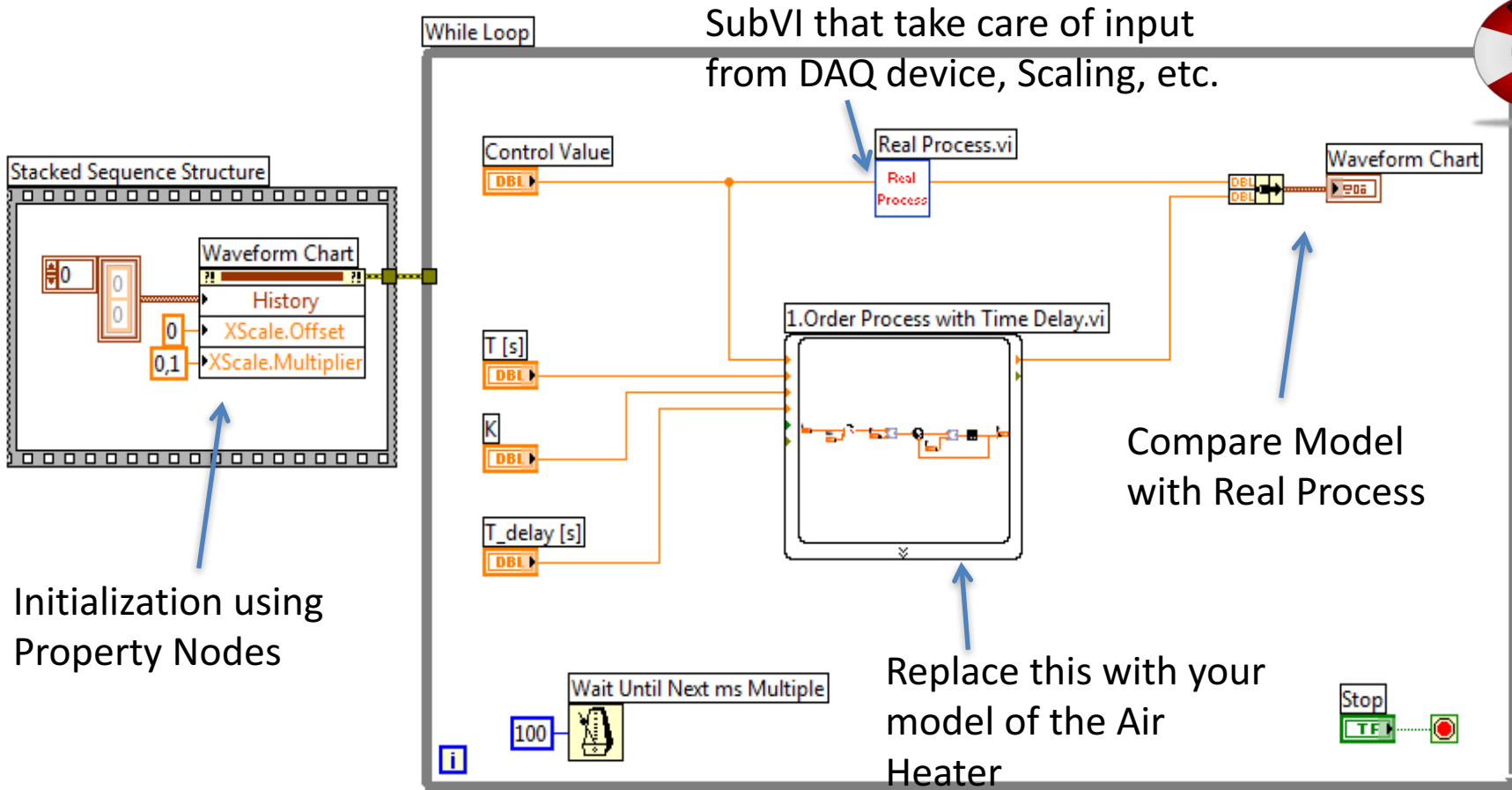
Finding Model Parameters using “Trial and Error”

You may use, e.g., the following Parameters as a starting point, but since every Air Heater is unique, you may want to adjust these parameters. The “Trial and Error Method” may be an easy way to find the Parameters for your Process.

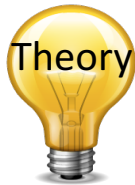


Procedure: You run the Model and the Real Process in Parallel. Adjust the Model Parameters until the output of the Model and the Real Process is “equal”.

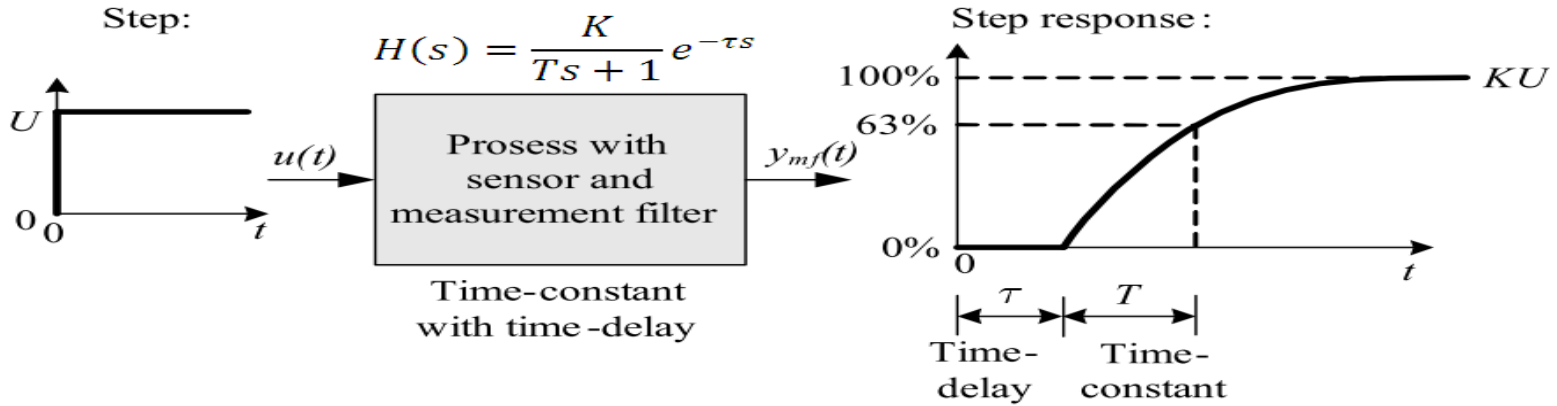
“Trial and Error” Example in LabVIEW



DEMO



Alternative: Finding Model Parameters using the “Step Response Method”



$$\dot{T}_{out} = \frac{1}{\theta_t} \{-T_{out} + [K_h u(t - \theta_d) + T_{env}]\}$$

Laplace

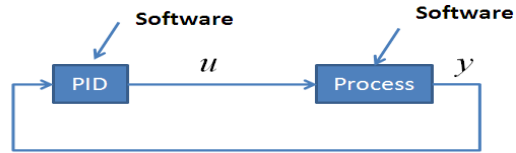
$$H(s) = \frac{T_{out}(s)}{u(s)} = ?$$

Assuming a 1.order model you can easily find the model parameters (Process Gain, Time constant and a Time delay if any) from the step response of the real system

DEMO



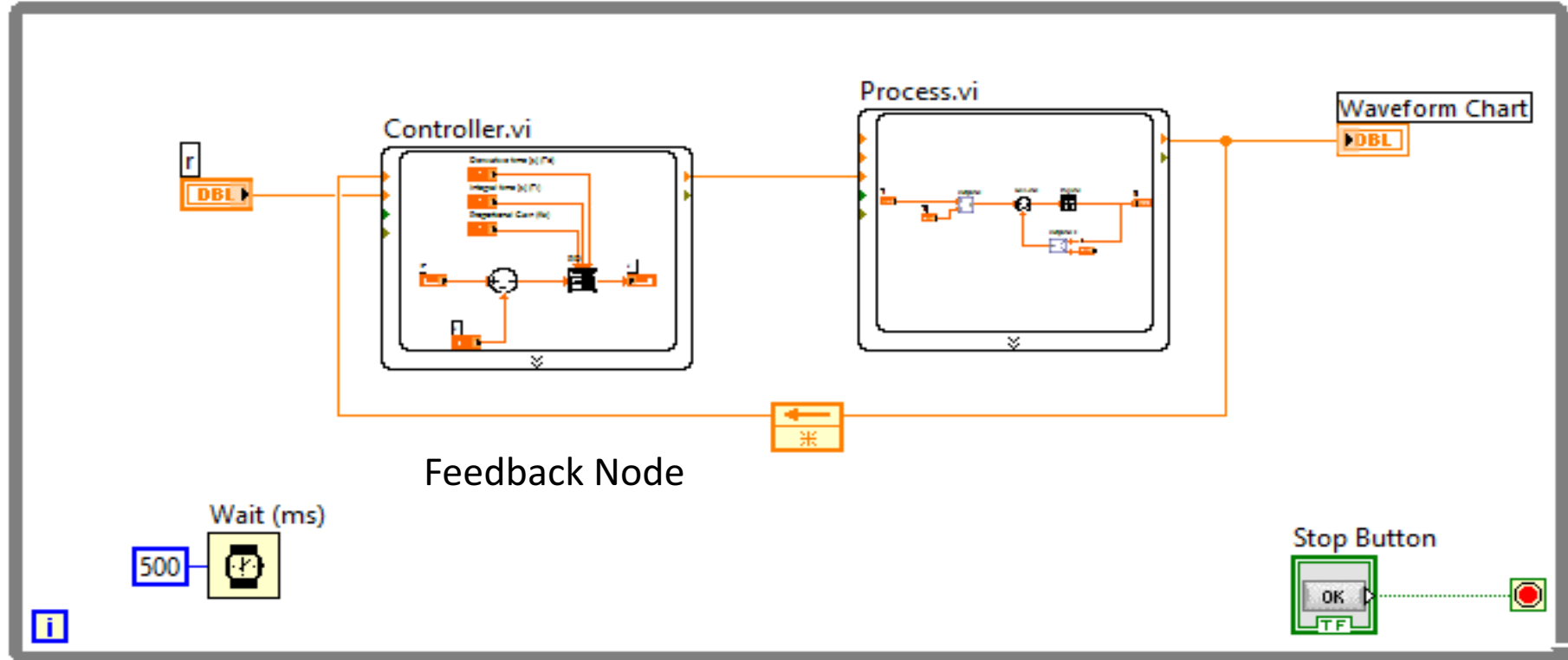
Simulated Control System



PID Control of Model in LabVIEW

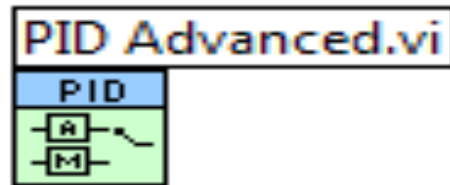


While Loop

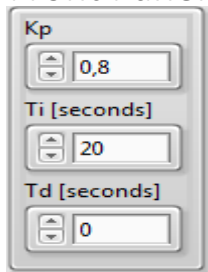




PID in LabVIEW



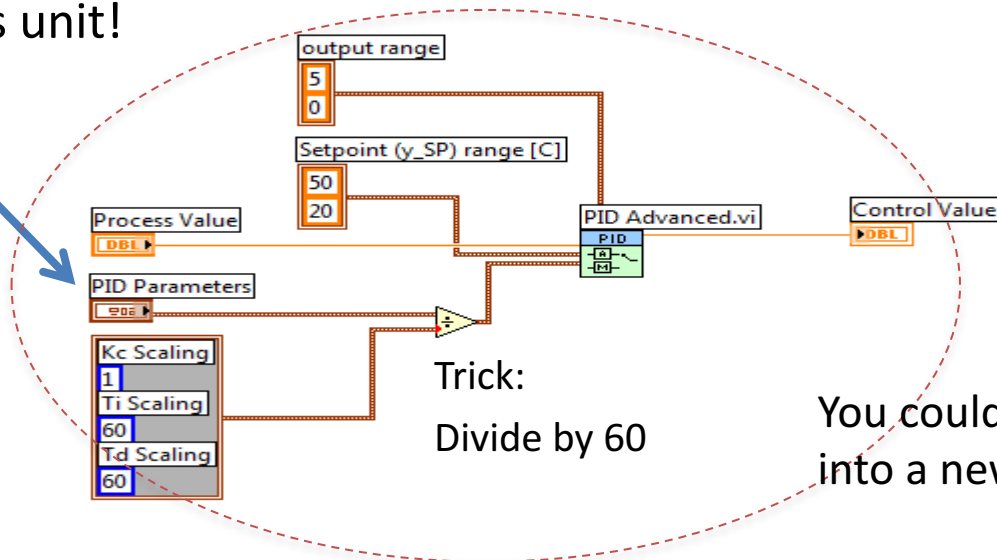
Front Panel



Cluster

Normally we use seconds as unit for T_i and T_d (which is recommended!)
But the built-in PID algorithm in LabVIEW uses minutes as unit!

Block Diagram:



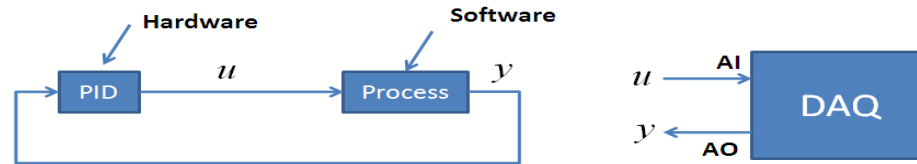
Trick:
Divide by 60

You could also put this code into a new SubVI

DEMO



HIL with Fuju PXG5 PID

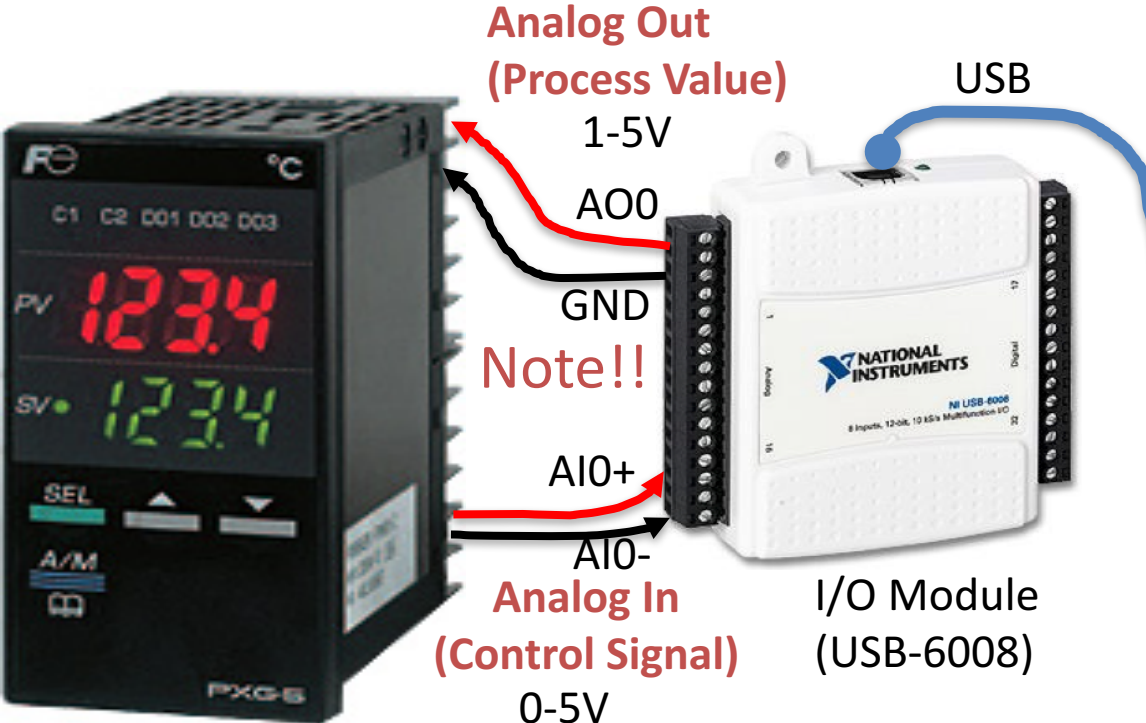


Hans-Petter Halvorsen, M.Sc.

HIL with Fuji PXG5PID

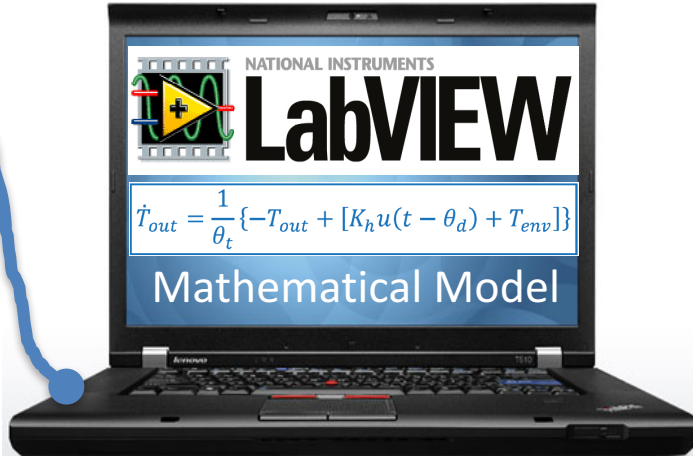
- It may be very useful to test a controller function with a simulated process before the controller is applied to the real (physical) process.
- If the mathematical model used in the simulator is an accurate representation of the real process, you may even tune the controller parameters (e.g. the PID parameters) using the simulator.

HIL Simulation Setup



PID Control

Model of Process (Air Heater)

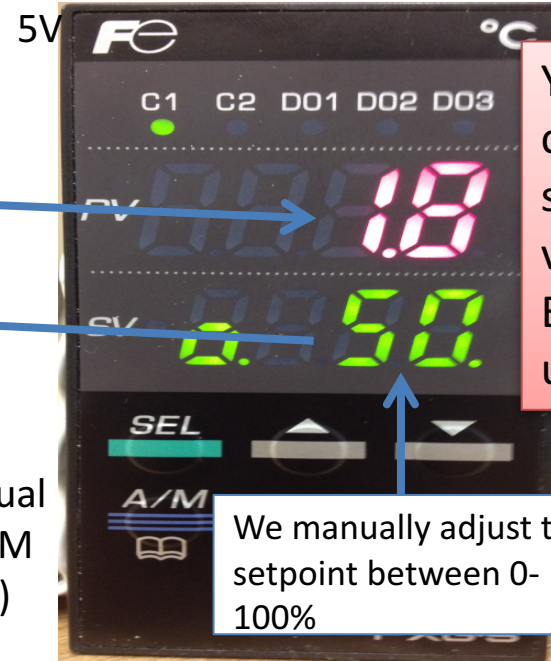
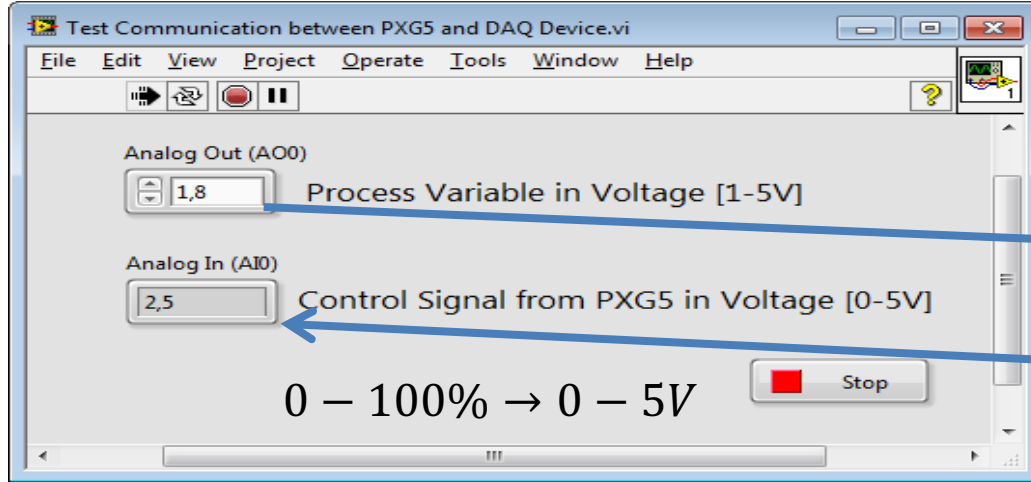


Computer (with LabVIEW)

Test Communication between PXG5 and DAQ Device

You should test the communication before you start working on the task

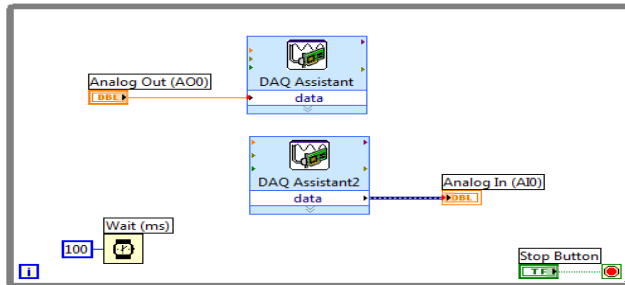
Note! Set Parameter C1r in Ch6: 0-20mA.
A 250 Ω resistor converts the signal to 0-5V



You can choose to show voltage or Engineering units

We manually adjust the setpoint between 0-100%

We run the PXG5 in Manual Mode (hold down the A/M button for a few seconds)

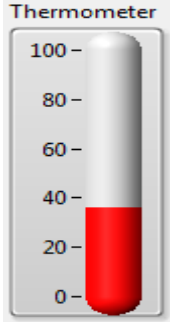
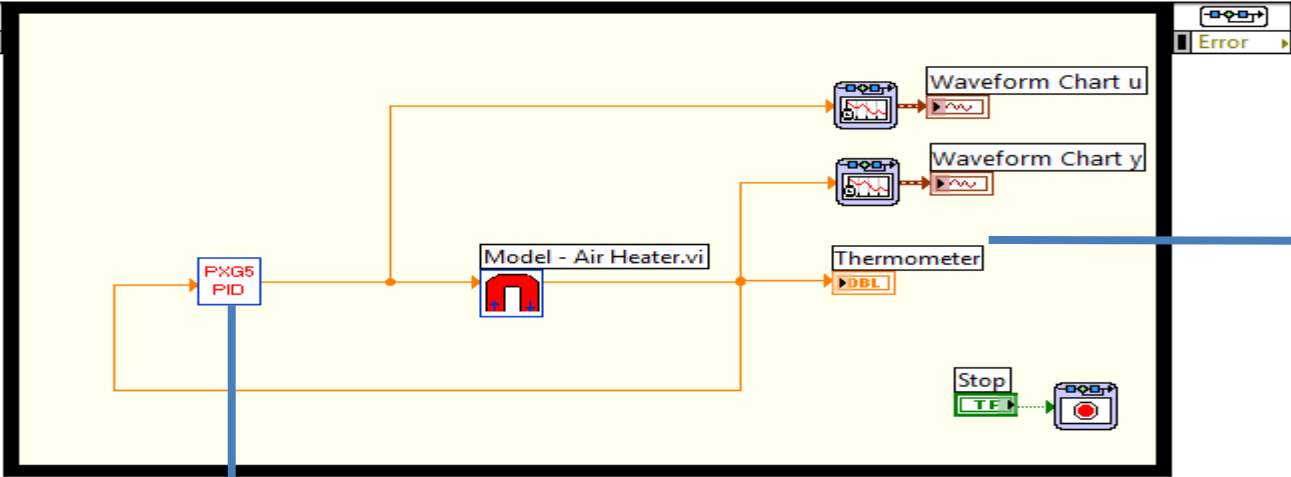


- Changing MV (control output values)
- 1** Switch to manual mode.
 - 2** Change the display to PV/MV display (MAN/AT/SELF lamp is lit). (Pressing the key in manual mode toggles between PV/SV display and PV/MV display.)
 - 3** Change the MV with the keys. (Changes are reflected to the MV as it is changed.)

HIL Simulation in LabVIEW - Example

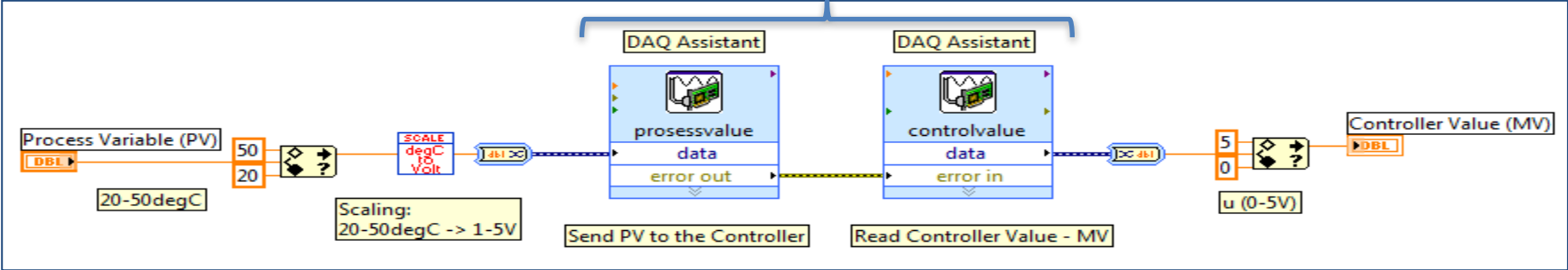


You can use a Simulation Loop or a While Loop



SubVI:

Note! This is opposite of what you normally do



Setting PID Parameters on the PXG5

Sets parameters for controls such as PID.

Parameter display symbol	Parameter name	Function	Setting range	Initial value	Remarks
"P" (P)	Proportional band	Sets the proportional band of the PID parameter. Setting "0.0" will turn it to an ON/OFF control.	0.0 to 999.9%	5.0%	
"I" (i)	Integration time	Sets the integration time of the PID parameter. Setting "0" will turn off integration.	0 to 3200 sec	240 sec	
"d" (d)	Differential time	Sets the differential time of the PID parameter. Setting "0.0" will turn off derivation.	0.0 to 999.9 sec	60.0 sec	

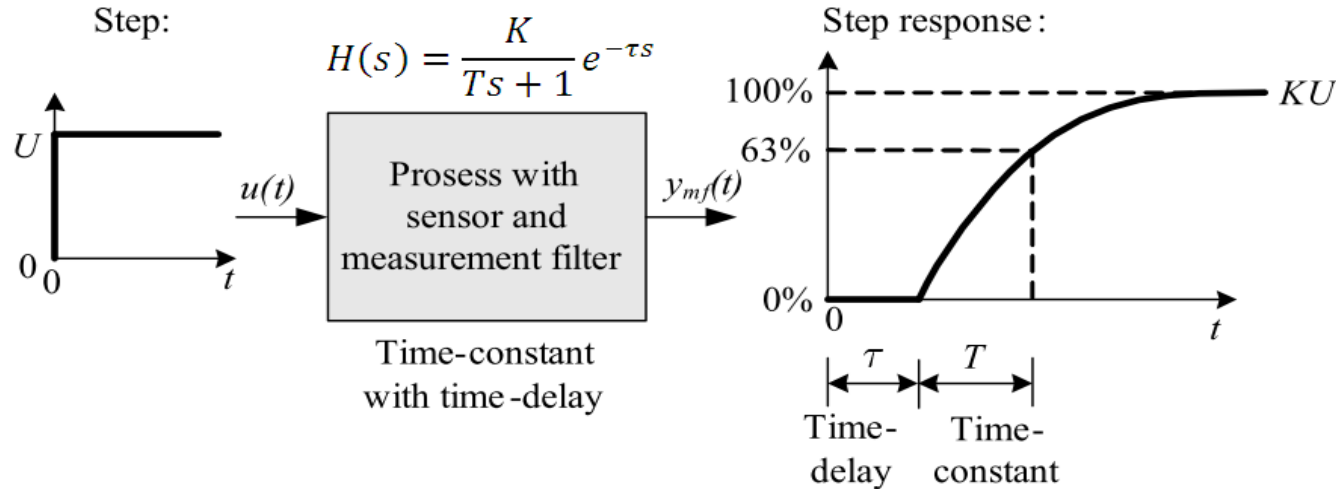
Note! PXG5 uses **Proportional Band**

$$PB = \frac{100\%}{K_p} \Leftrightarrow K_p = \frac{100\%}{PB}$$

Example:

$$K_p = 0.8 \rightarrow PB = \frac{100\%}{K_p} = \frac{100\%}{0.8} = 125\%$$

PID Tuning with Skogestad



We can set, e.g., $T_c=10$ sec. and $c=1.5$. You may use other values if these values give a poor result.

Process type	$H_{psf}(s)$ (process)	K_p	T_i	T_d
Integrator + delay	$\frac{K}{s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	0
Time-constant + delay	$\frac{K}{Ts+1} e^{-\tau s}$	$\frac{T}{K(T_C + \tau)}$	$\min [T, c(T_C + \tau)]$	0
Integr + time-const + del.	$\frac{K}{(Ts+1)s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	T
Two time-const + delay	$\frac{K}{(T_1s+1)(T_2s+1)} e^{-\tau s}$	$\frac{T_1}{K(T_C + \tau)}$	$\min [T_1, c(T_C + \tau)]$	T_2
Double integrator + delay	$\frac{K}{s^2} e^{-\tau s}$	$\frac{1}{4K(T_C + \tau)^2}$	$4(T_C + \tau)$	$4(T_C + \tau)$

Table 1: Skogestad's formulas for PI(D) tuning.

PXG5 – Auto-tuning


1. With the controller in manual mode (open loop control): Adjust the manual control signal so that the process output is approximately 2.5 V.
2. Set the controller in automatic mode (“closed loop control”).
3. Start Auto-tuning



Note! PXG5 uses
Proportional Band

A lamp is blinking when the
auto-tuning is running

- Changing MV (control output values)

1 Switch to manual mode.

2 Change the display to PV/MV display (MAN/AT/SELF lamp is lit).
(Pressing the  key in manual mode toggles between PV/SV display and PV/MV display.)

3 Change the MV with the   keys.
(Changes are reflected to the MV as it is changed.)

6-1 / Operation (Ch1)

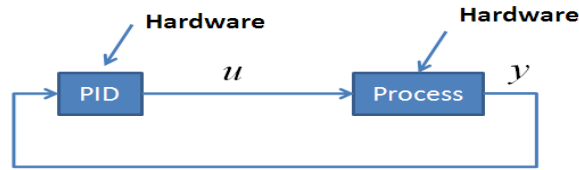
The following is a menu to operate the controller. Switchover between auto and manual control output, switchover between RUN and standby, and other such functions.

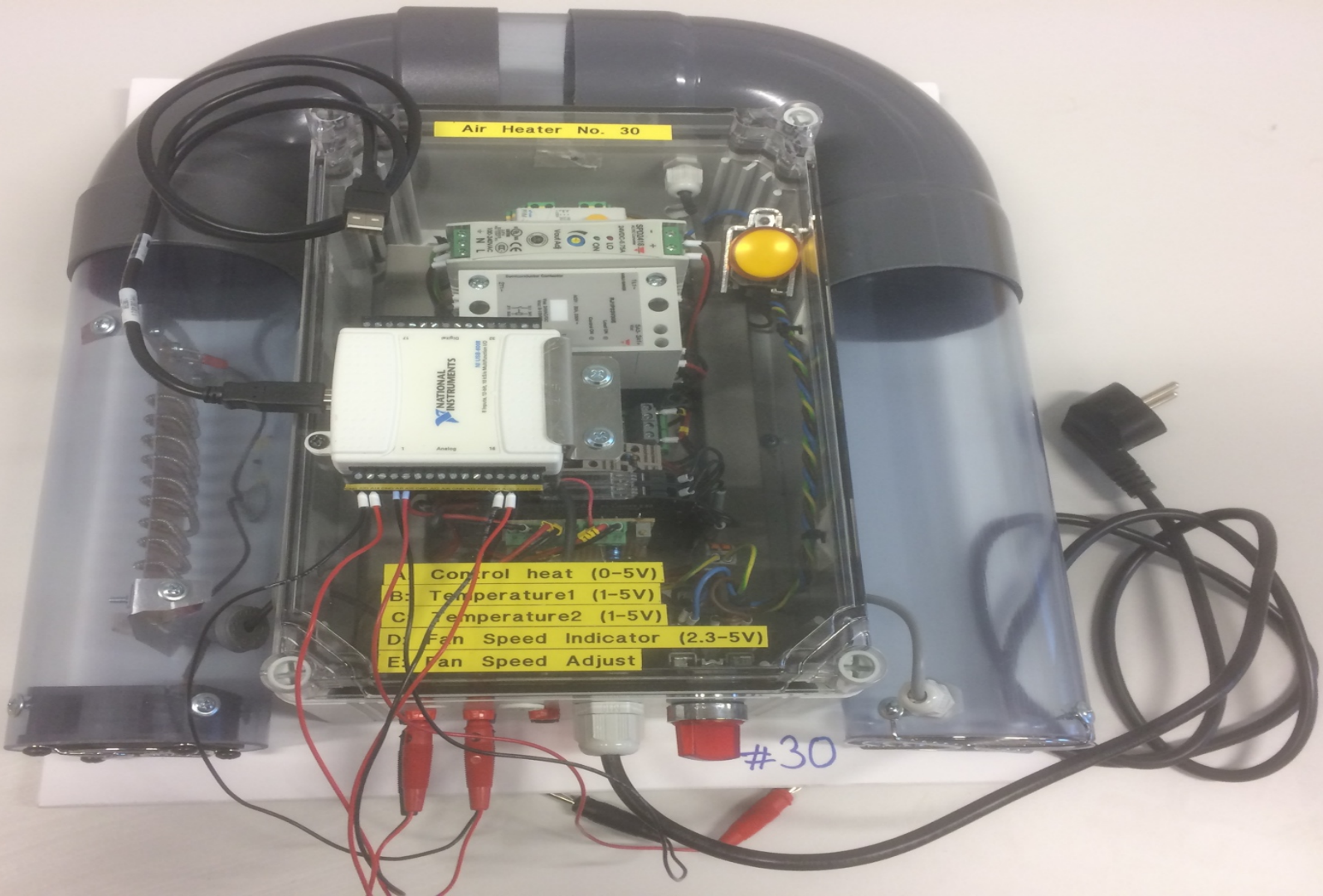
Parameter display symbol	Parameter name	Function	Setting range	Initial value	Remarks
"MAN" (MAN)	Switchover between auto and manual mode	Switchover between auto and manual modes	oFF (auto) / on (manual)	oFF	
"STby" (STby)	Switchover between RUN and standby	Switchover the operation mode between RUN and standby	oFF (RUN) / on (standby)	oFF	
"rEM" (rEM)	Switchover between local and remote SV operation	Switchover between local and remote SV operation	LoCL (local) / rEM (remote)	LoCL	(Note1)
"PrG" (PrG)	Ramp soak control command	Changes ramp soak run states	oFF (stop) rUn (run) hLd (hold)	oFF	Displays End (when ending) or GS (during guaranty soak).
"AT" (AT)	Auto-tuning run command	Runs auto-tuning.	oFF (stop/finish) on (normal type) Lo (low PV type)	oFF	

DEMO



Fuju PXG5 + Real Air Heater





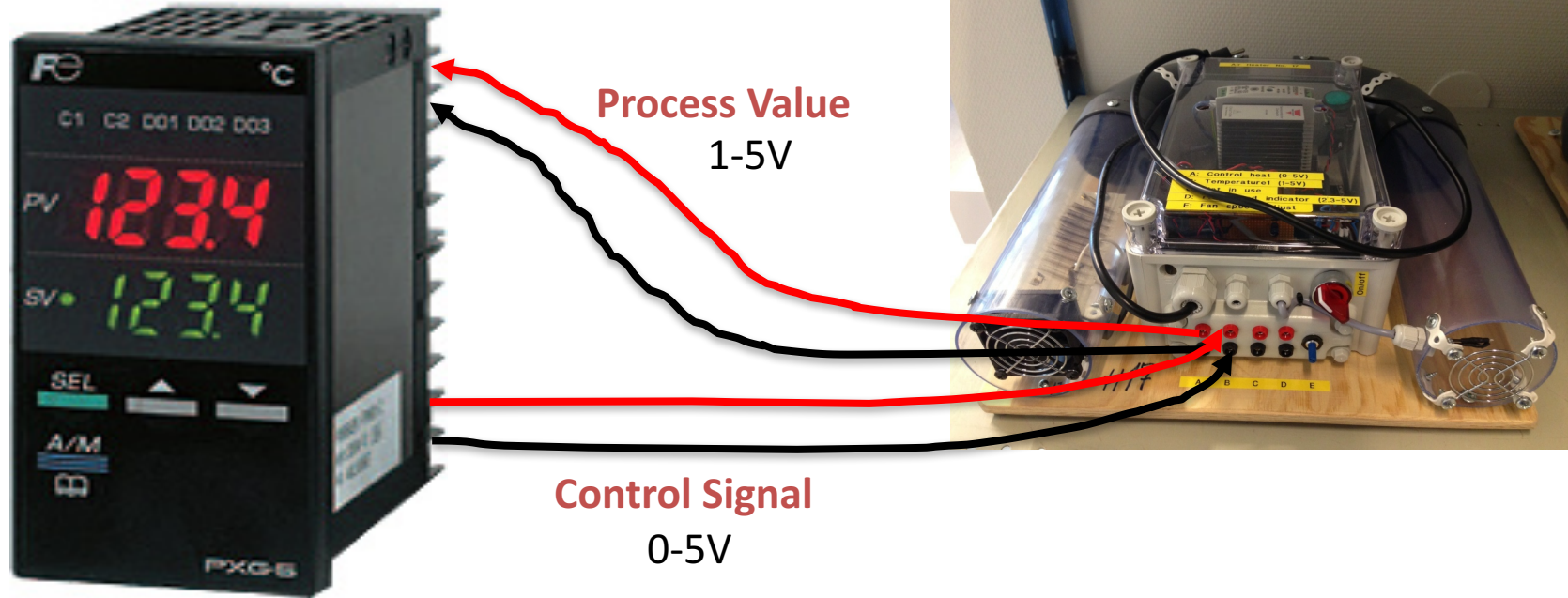
Air Heater No. 30

- A: Control heat (0-5V)
- B: Temperature1 (1-5V)
- C: Temperature2 (1-5V)
- D: Fan Speed Indicator (2.3-5V)
- E: Fan Speed Adjust

#30

Fuju PXG5 + Real Air Heater

Industrial PID Controller



You should Test the PXG5 PID Controller on the real Air Heater. Are you able to use the same PID settings you found using the Model? Test also the Auto-tuning functionality. Do you get the same parameters as using the model?

Fuju PXG5 + Real Air Heater + PC for Monitoring

Industrial PID Controller

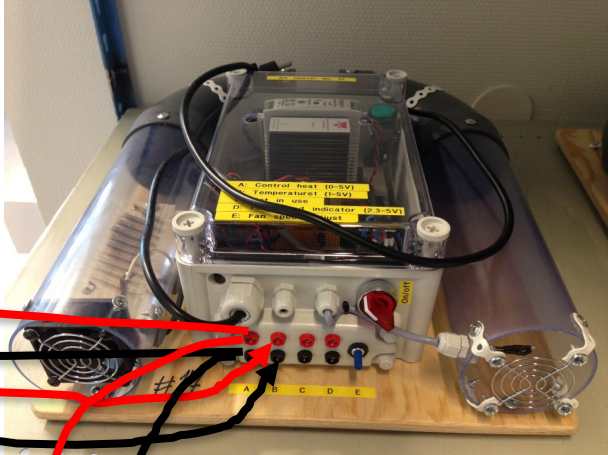


Process Value

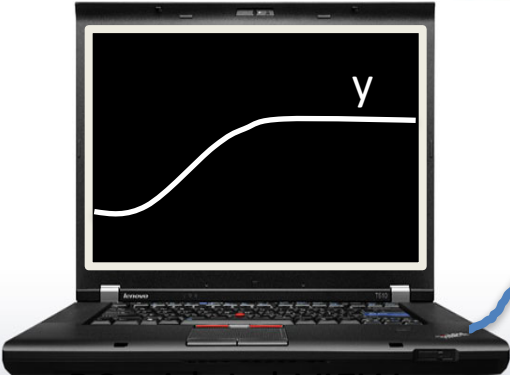
1-5V

0-5V

Control Signal



Trending/Monitoring the Process Value on the PC



PC with LabVIEW

USB



Process Value

1-5V



With this setup you can Monitor the Process Value on your PC

Hans-Petter Halvorsen

University of Southeast Norway

www.usn.no

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

Web: <https://www.halvorsen.blog>

