Simulation in LabVIEW

Hans-Petter Halvorsen, M.Sc.

Software

- LabVIEW
- LabVIEW Control Design and Simulation Module
  - This module is used for creating Control and Simulation applications with LabVIEW.
  - Here you will find PID controllers, etc. The module is available as a palette on your block diagram.

All LabVIEW Software can be downloaded here: www.ni.com/download

Contents

- Block Diagram Simulation based on differential Equations
  - Using the Simulation Loop
- PID Control with built-in PID blocks/functions
- Creating and using Simulation Subsystems
- Simulations using a While Loop with Subsystems inside

What is LabVIEW?

Hans-Petter Halvorsen, M.Sc.

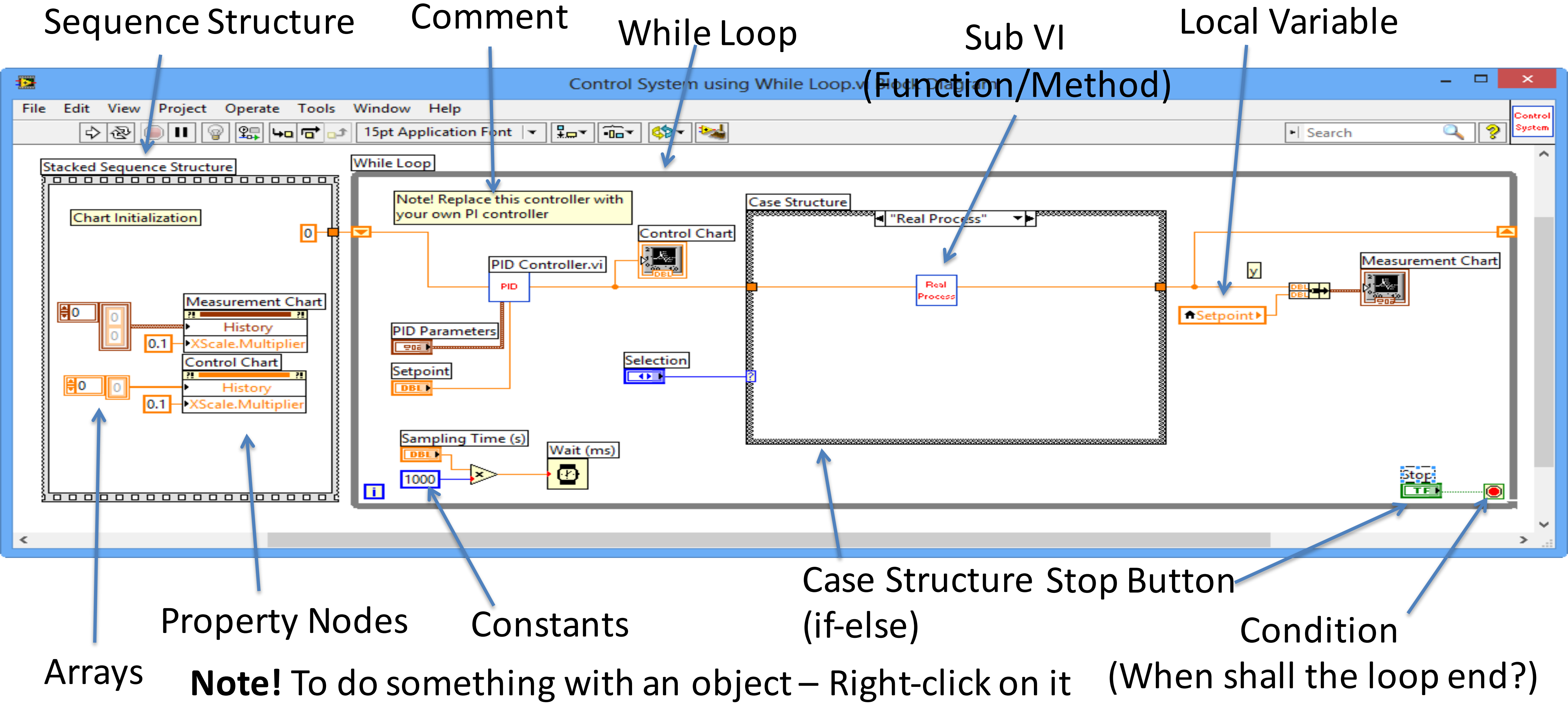
LabVIEW = Fun!

Graphical Programming:

- Very different from traditional programming like VB, C#, Maple, MATLAB, MathScript, etc.
- It is more like a “drawing program” than a Programming Language
- This makes it easy to use for those who are not programmers (or dont like programming ☺)
- Excellent tool when using Hardware, when you need to take Measurements (DAQ), etc.
- It is fun and makes you very creative!

LabVIEW Example

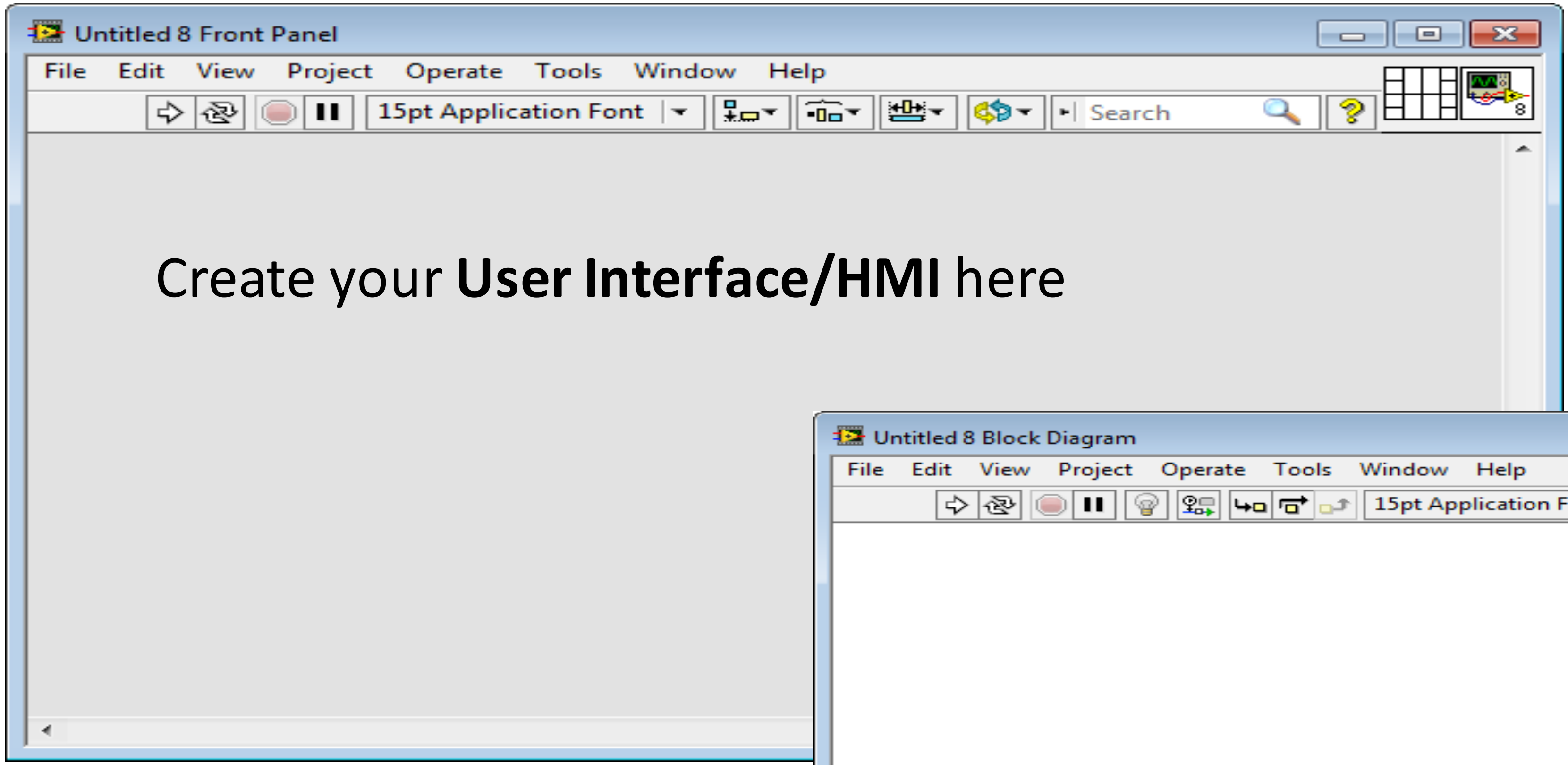
LabVIEW has the same things as other programming languages, but in a graphical way!



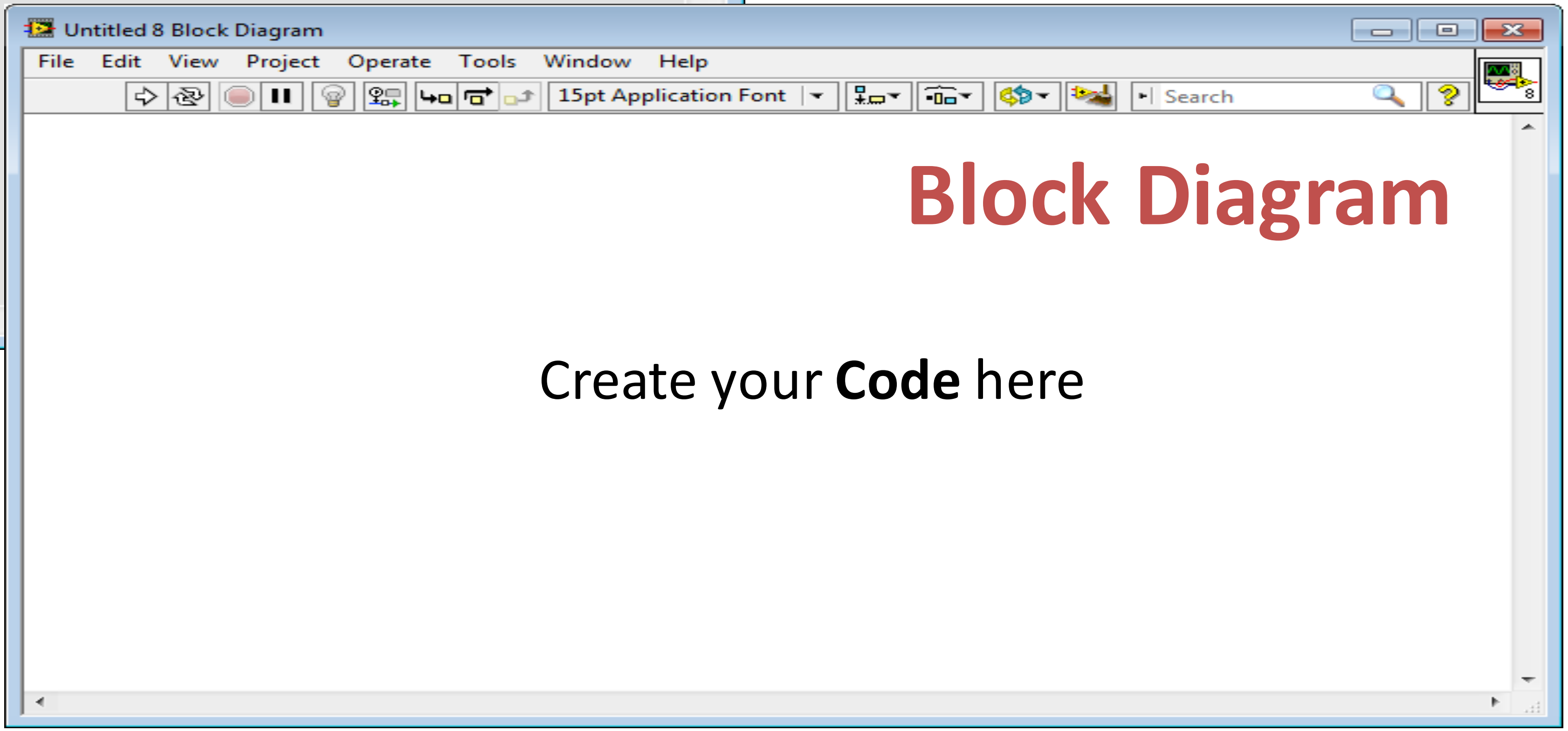
Note! To do something with an object – Right-click on it

LabVIEW Environment

Front Panel



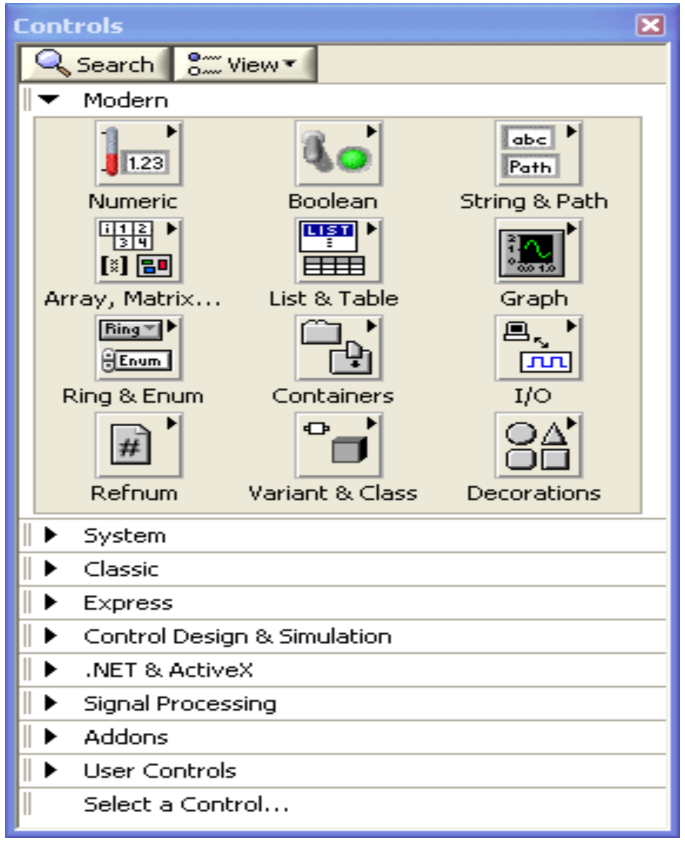
Note! Both the Front Panel and the Block Diagram are stored in one single file. These files are called **VIs** (because the file extension is ".vi"). VI = Virtual Instruments



Switch between them: **Ctrl + E**

Controls and **Functions** Palette

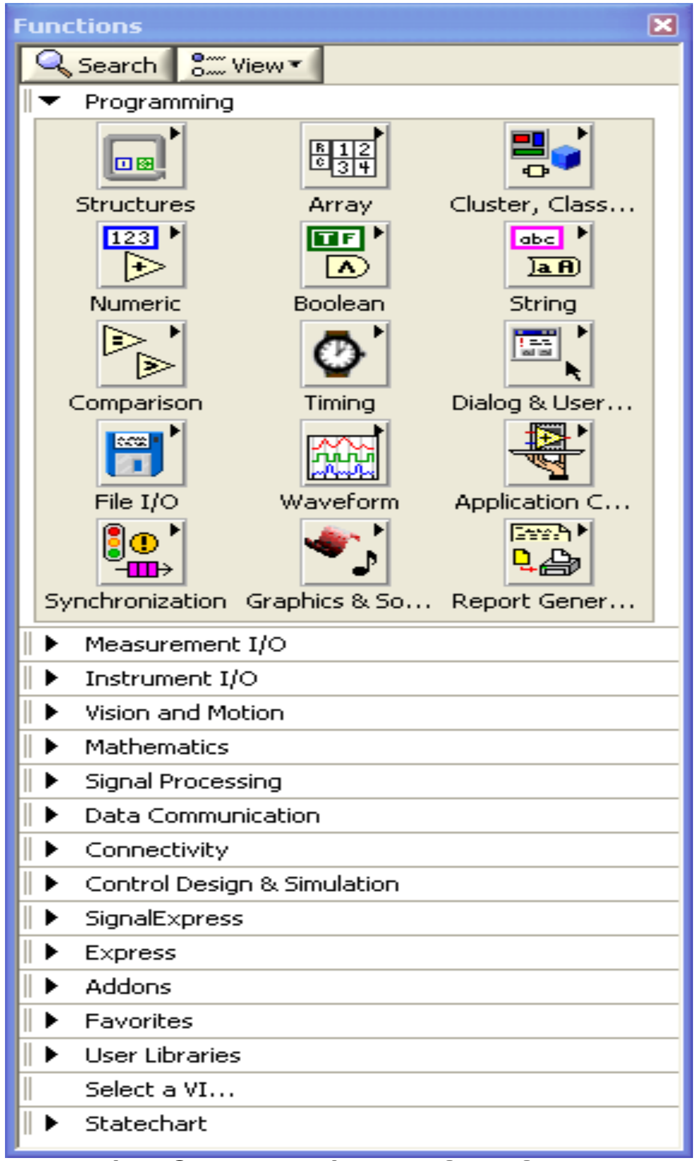
You can "pin" them!



Available only from the **Front Panel**

You create your <u>User Interface</u> with help of these Controls

Right-click on the Front Panel

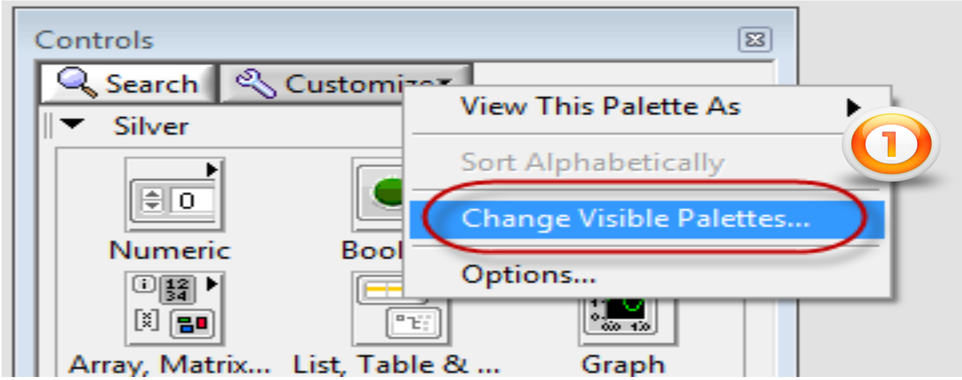


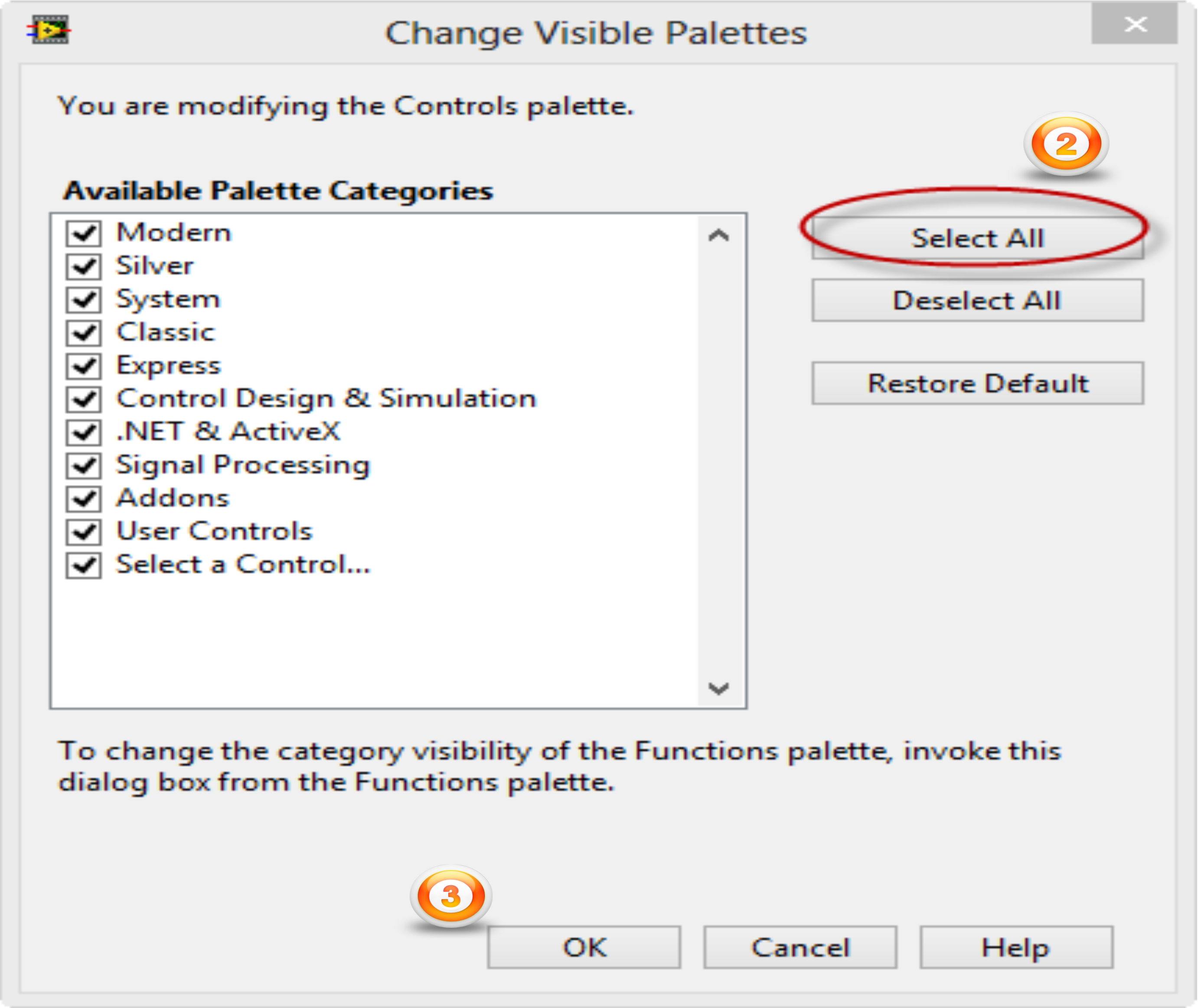
You create your <u>Code</u> with help of these Functions

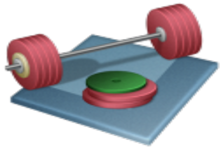
Right-click on the Block Diagram

Available only from the **Block Diagram**

Customizing Controls and Functions Palettes



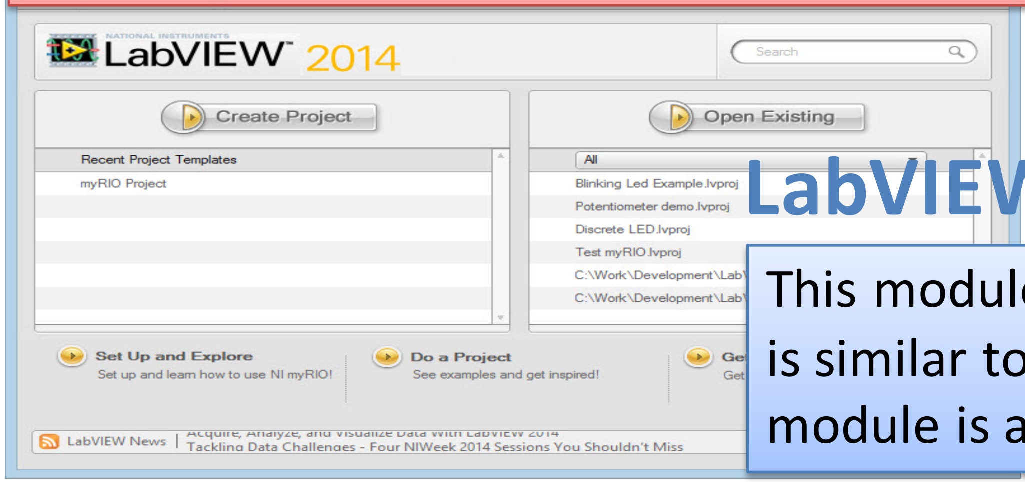


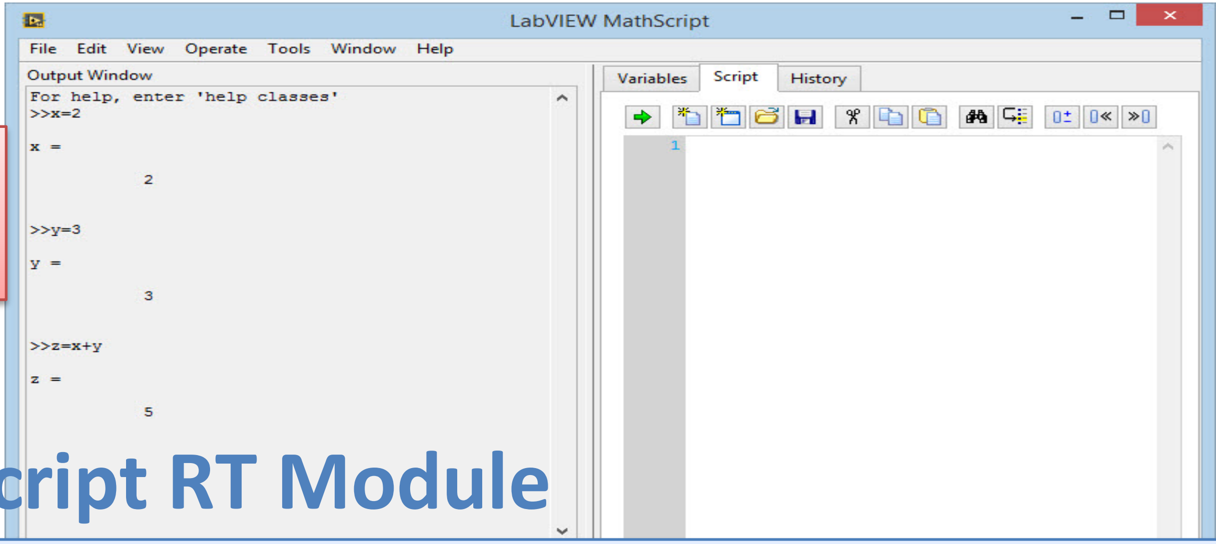


Do this for <u>both</u> the **Controls Palette** and the **Functions Palette**

LabVIEW

This is the core LabVIEW installation that installs the LabVIEW Programming Environment.





LabVIEW MathScript RT Module

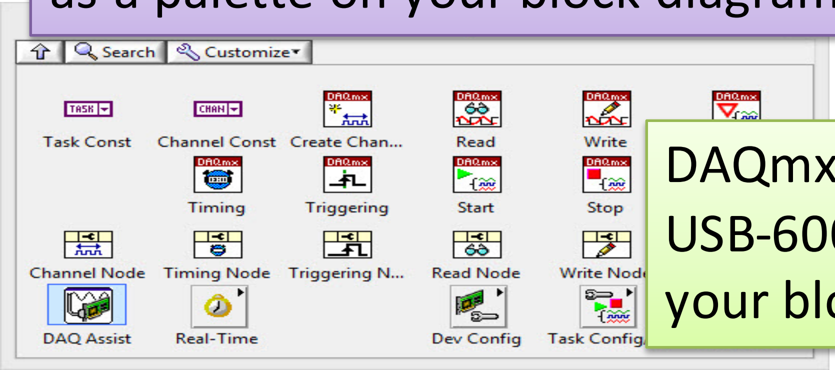
This module is a text-based tool that is very similar to MATLAB. The syntax is similar to MATLAB, you can create and run so-called m files, etc. The module is available from the Tools menu inside LabVIEW.

LabVIEW Control Design and Simulation Module

This module is used for creating Control and Simulation applications with LabVIEW. Here you will find PID controllers, etc. The module is available as a palette on your block diagram.

NI-DAQmx

DAQmx is the Hardware Driver needed in order to use hardware devices like NI USB-6008, NI TC-01, etc. inside LabVIEW. The module is available as a palette on your block diagram.



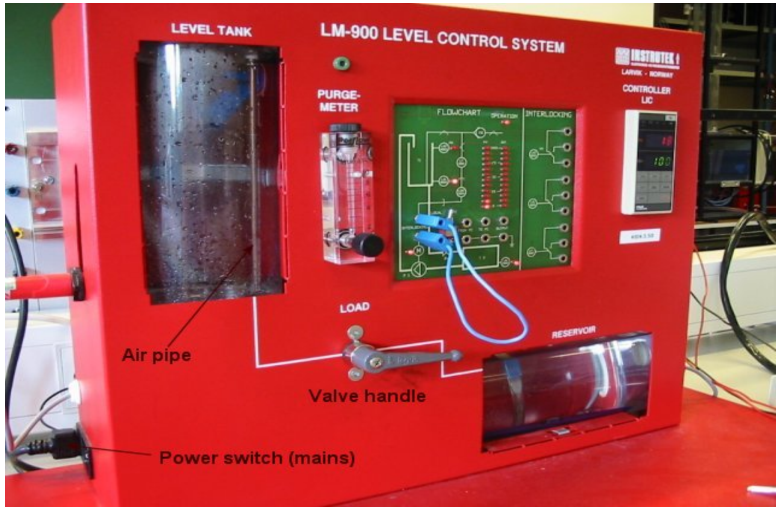
Modelling of Dynamic Systems

Hans-Petter Halvorsen, M.Sc.

These are small-scale processes we have at HiT

Dynamic Systems Examples

Water Tank:



h – Level in the tank

Mathematical Models (differential equations):

Alt 1 (Integrator):

$$\dot{h} = \frac{1}{A}\left[K_p u - F_{out}\right]$$

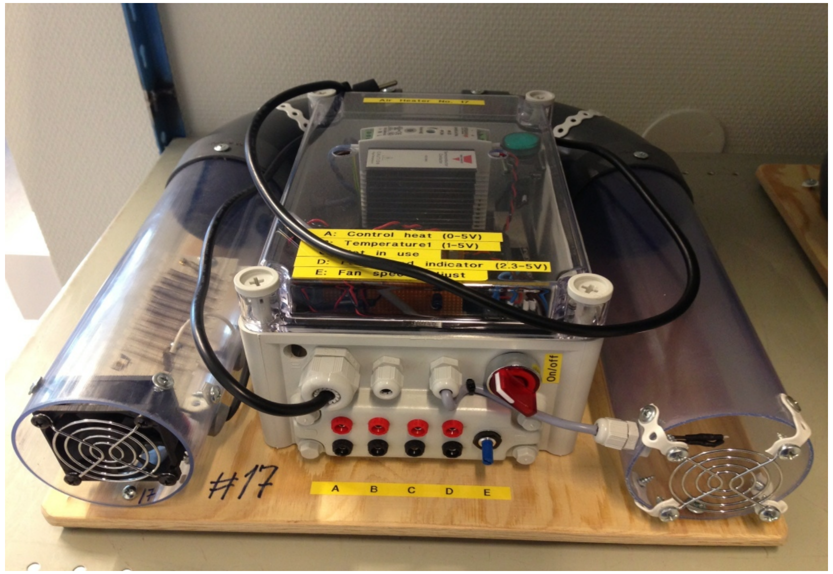
Alt 2 (Time constant/1.order):

$$\dot{h} = \frac{1}{A}\left[K_p u - K_v h\right]$$

Alt 3 (Nonlinear):

$$\dot{h} = \frac{1}{A}\left[K_p u - K_v \sqrt{\frac{\rho g h}{G}}\right]$$

Air Heater:



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

T – Temperature in the tube

Dynamic Systems

Dynamic system represented as a differential equation (1.order system):

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

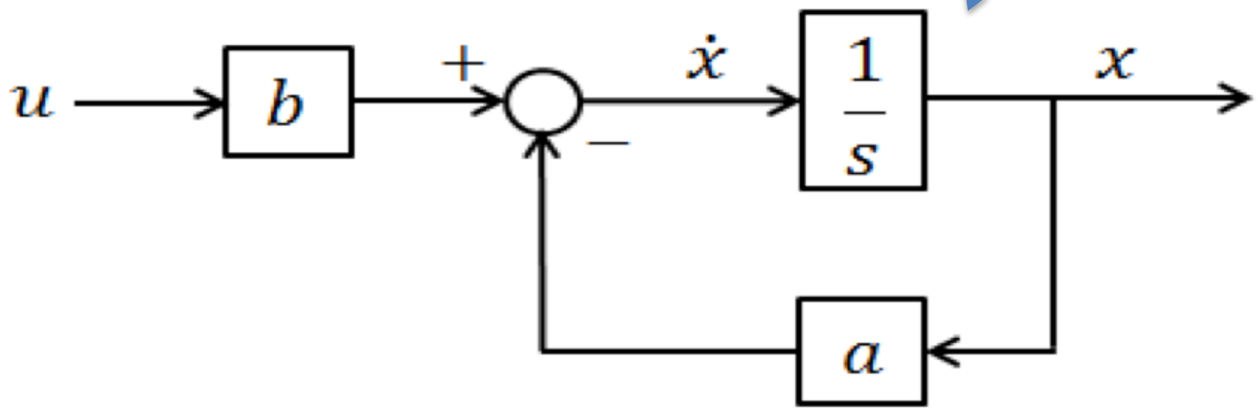
$$\frac{dx}{dt}$$

Note

We can "easily" create a block diagram from the differential equation(s)

Dynamic system represented as a block diagram

Integrator symbol



When we have the block diagram for the system, we can easily implement it in LabVIEW

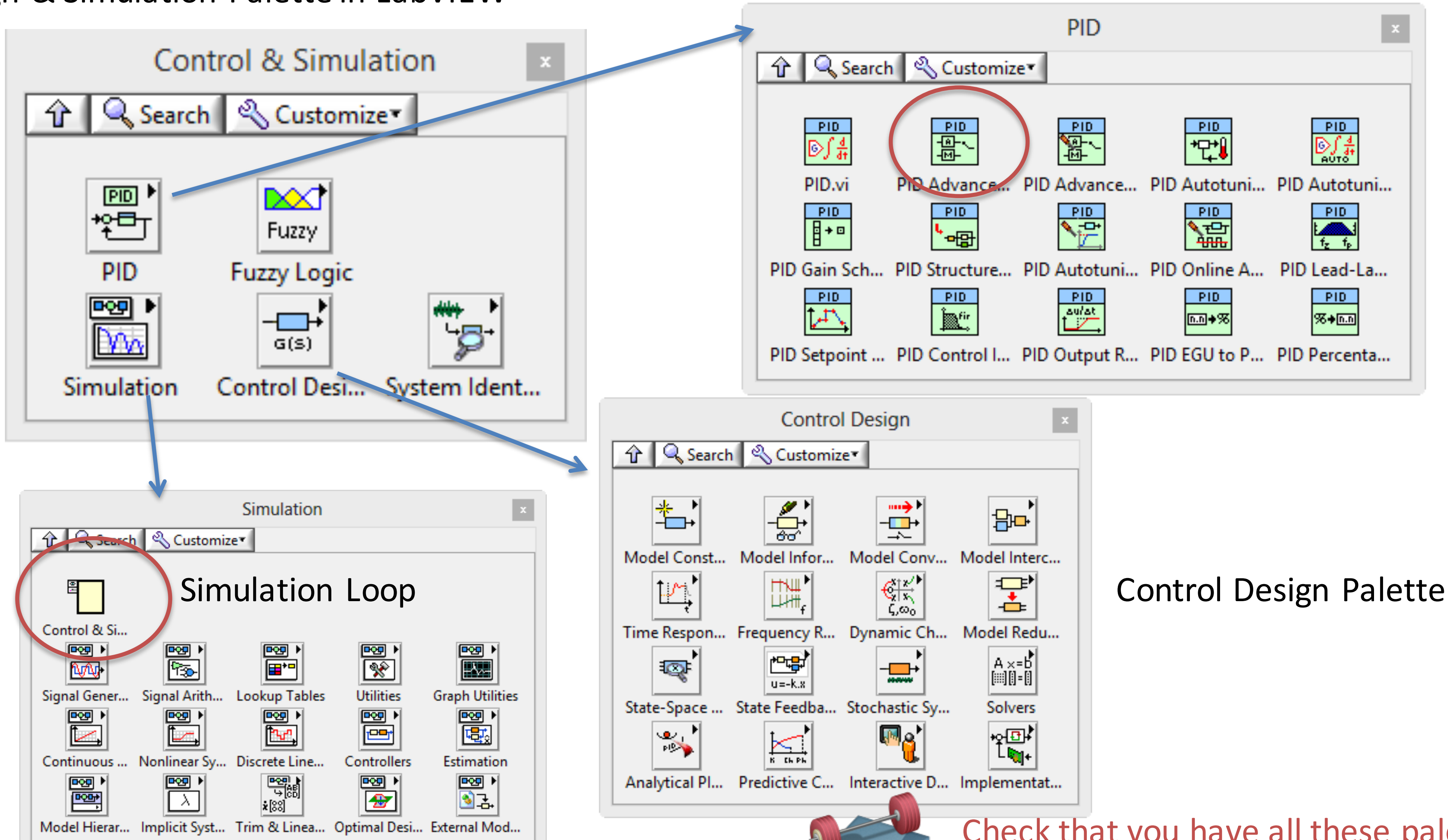
Simulation in LabVIEW

Hans-Petter Halvorsen, M.Sc.

Control and Simulation in LabVIEW

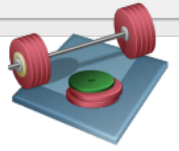
Control Design & Simulation Palette in LabVIEW

PID Palette in LabVIEW



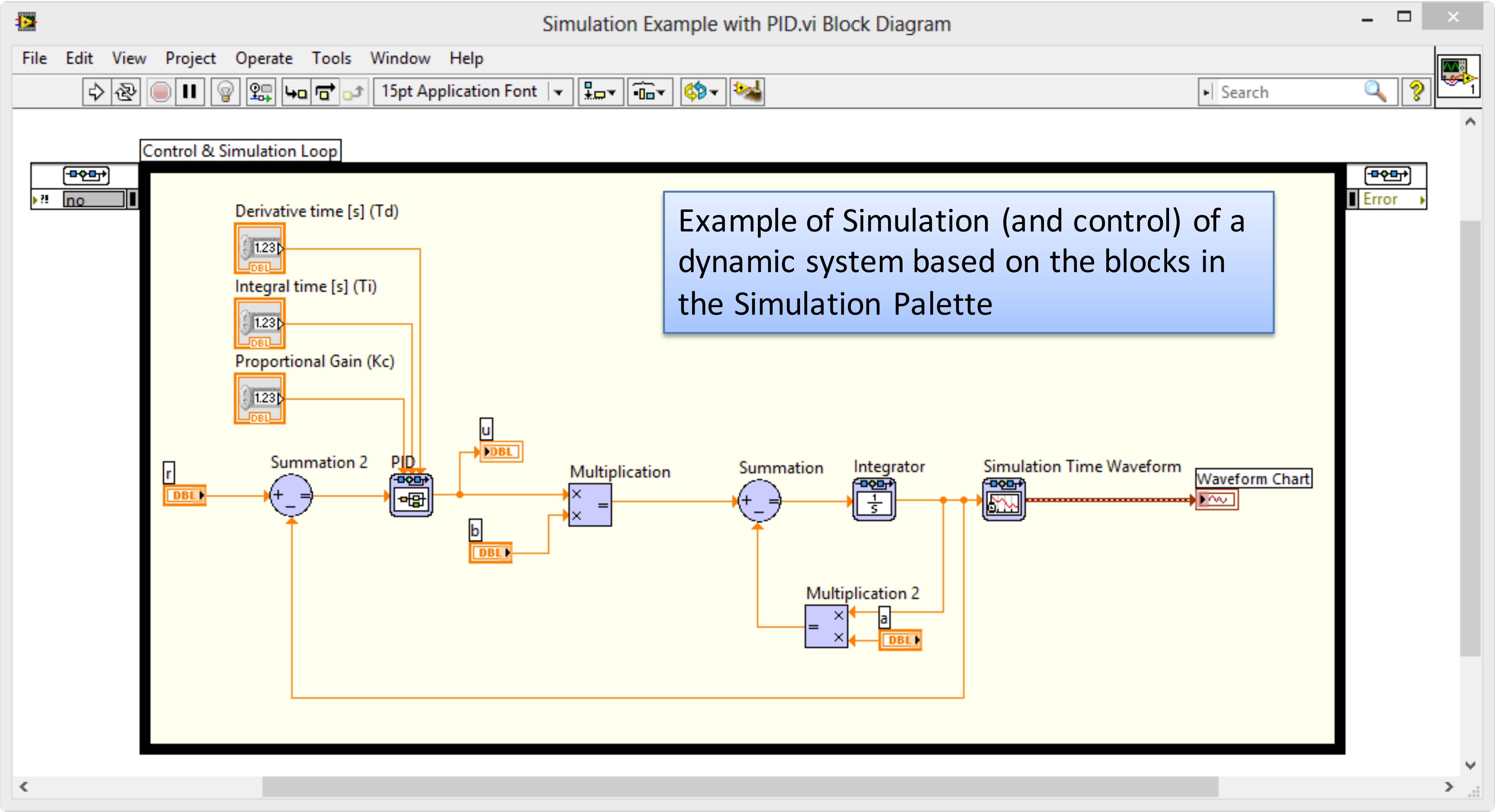
Control Design Palette in LabVIEW

Simulation Palette in LabVIEW



Check that you have all these palettes. Open the different subpalettes, etc.

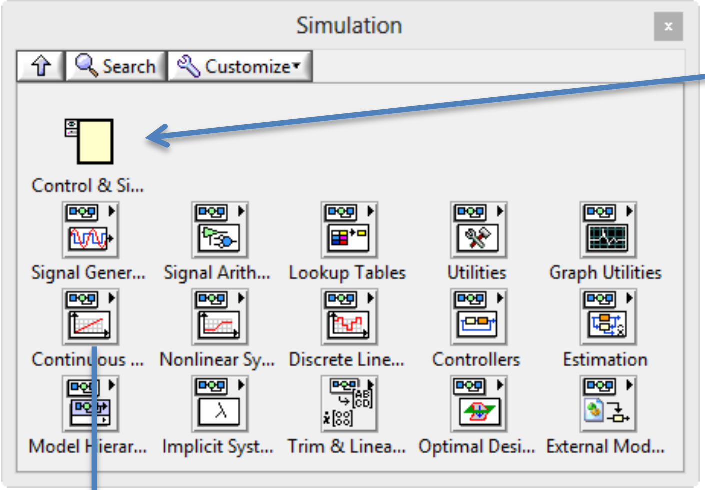
LabVIEW Control and Simulation Example



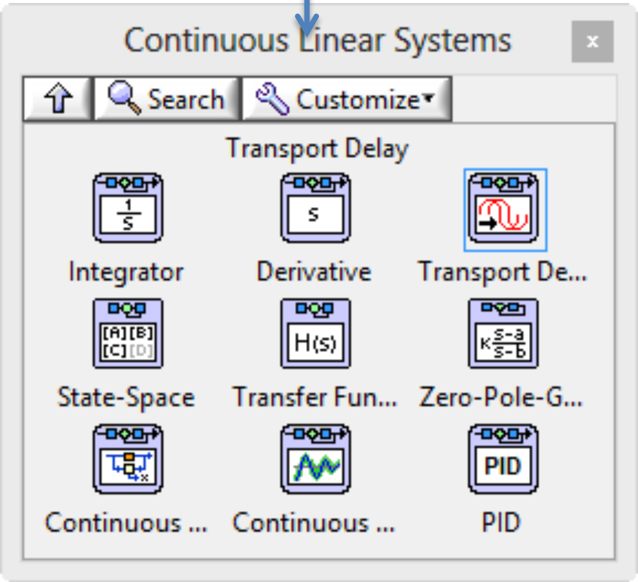
We are going to learn to create such a system (and much more)!

The Simulation palette in LabVIEW

Simulation Palette in LabVIEW



Simulation Loop: Similar to a While Loop, but customized for used together with the Simulation Blocks available in LabVIEW



Different Simulation Blocks by Category

- Continuous Systems
- Discrete Systems
- Nonlinear Systems
- etc.

LabVIEW Example

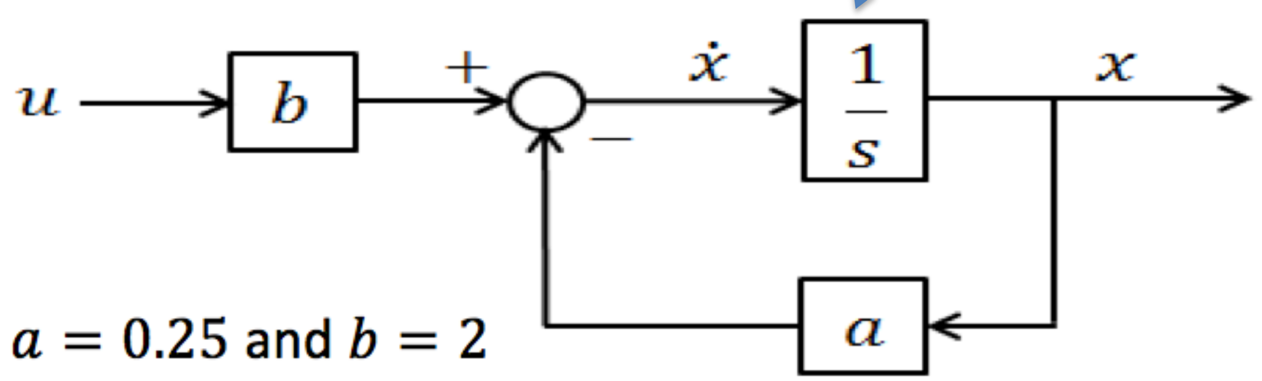
Hans-Petter Halvorsen, M.Sc.

Simulation Example

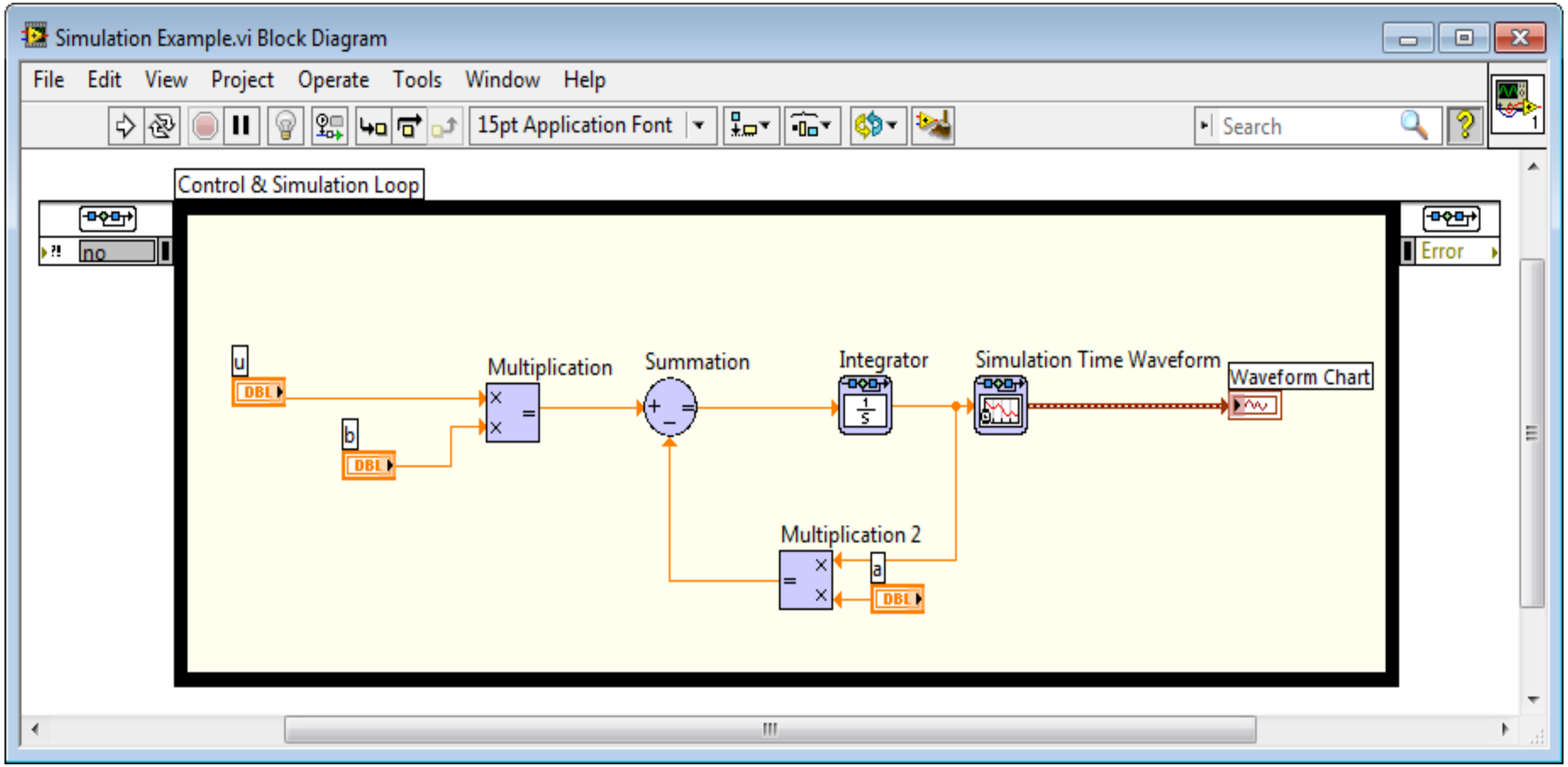
Integrator symbol

Dynamic system represented as a differential equation

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

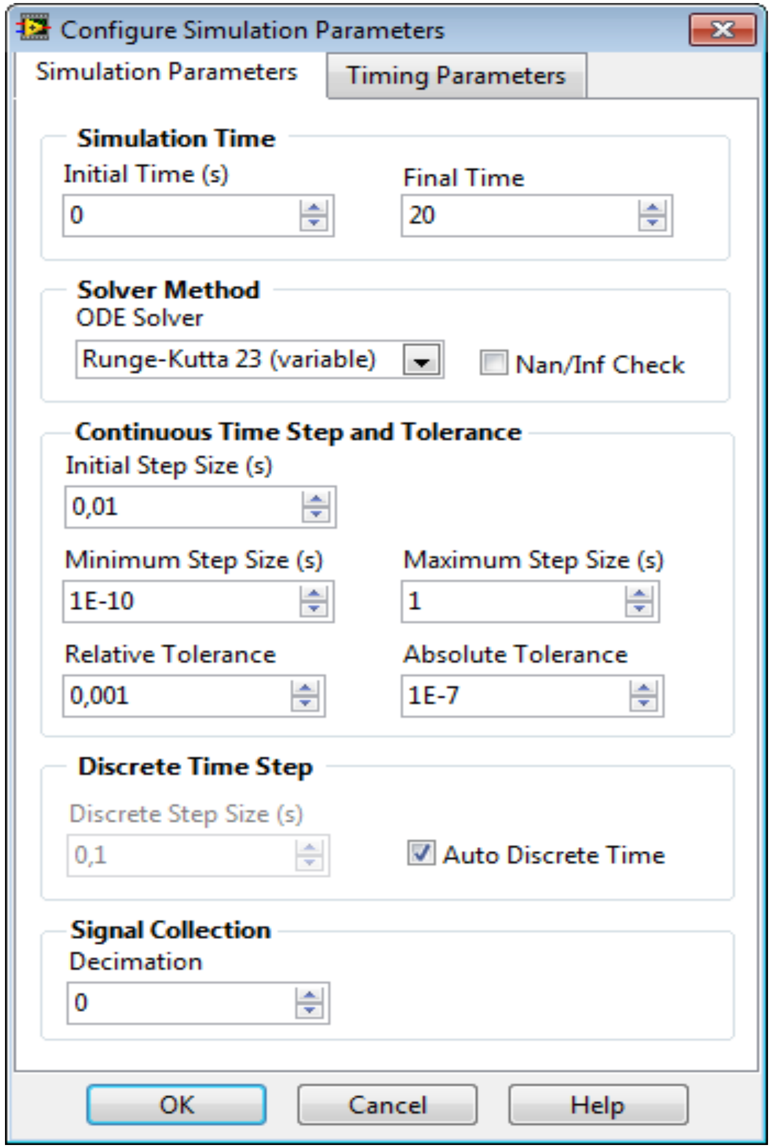


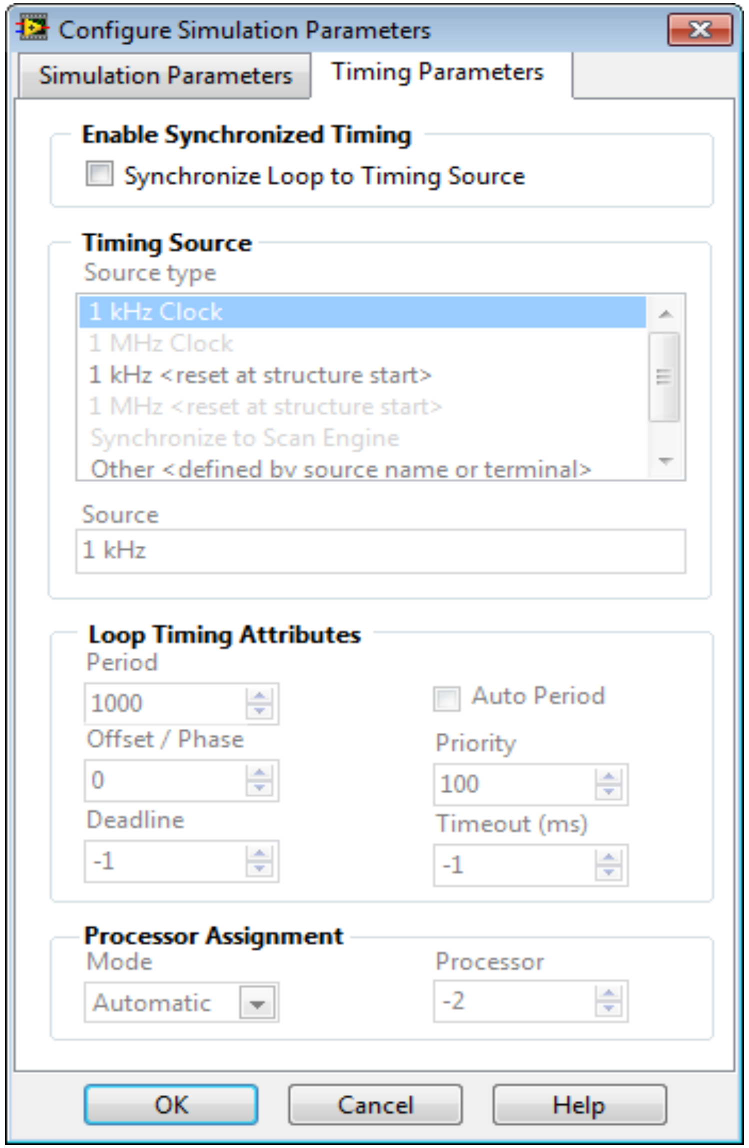
set $a = 0.25$ and $b = 2$



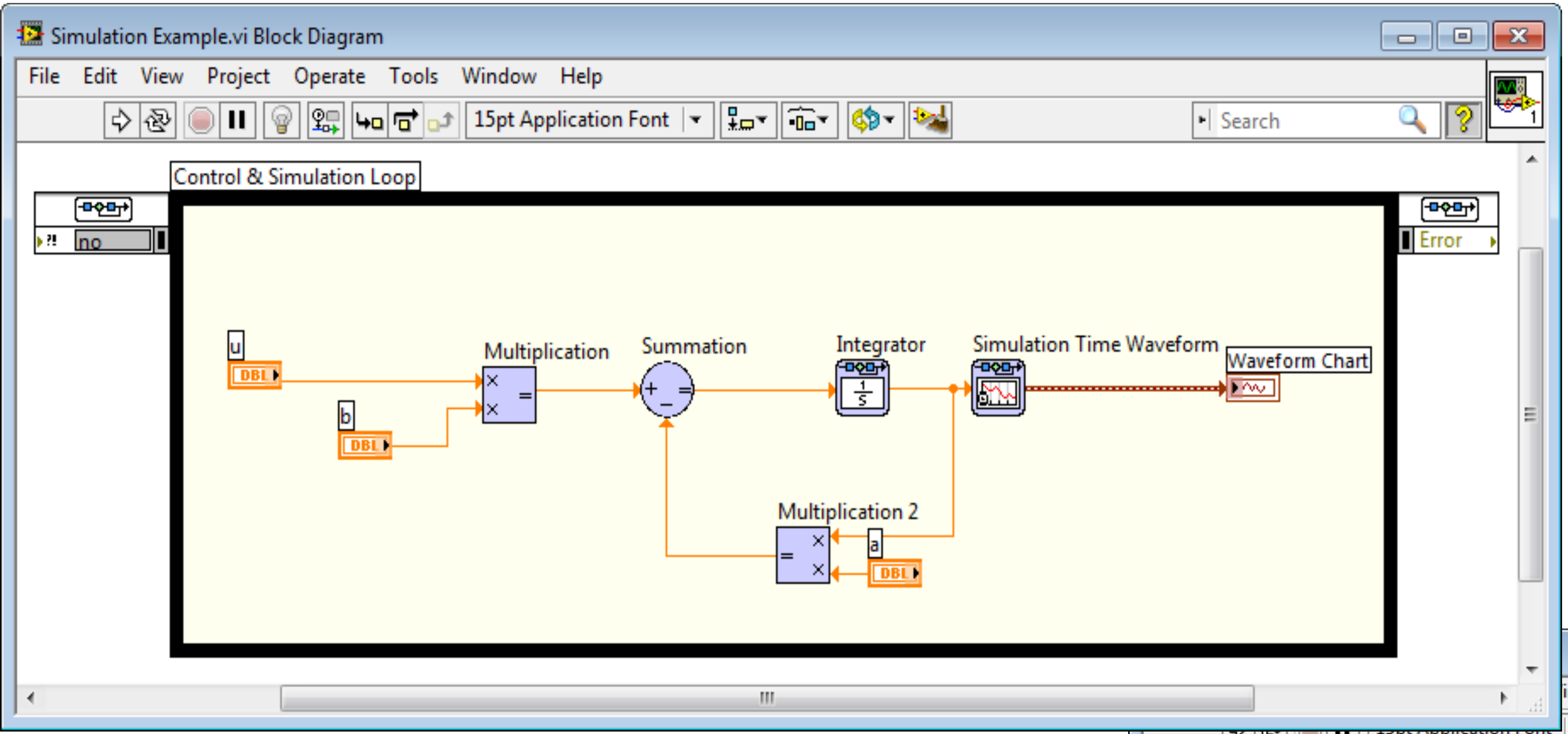
Simulation Example - Configuration

In the example the following simulation parameters could be used (right-click on the Simulation Loop border and select “Configure Simulation parameters…”):





Simulation Example - Solutions

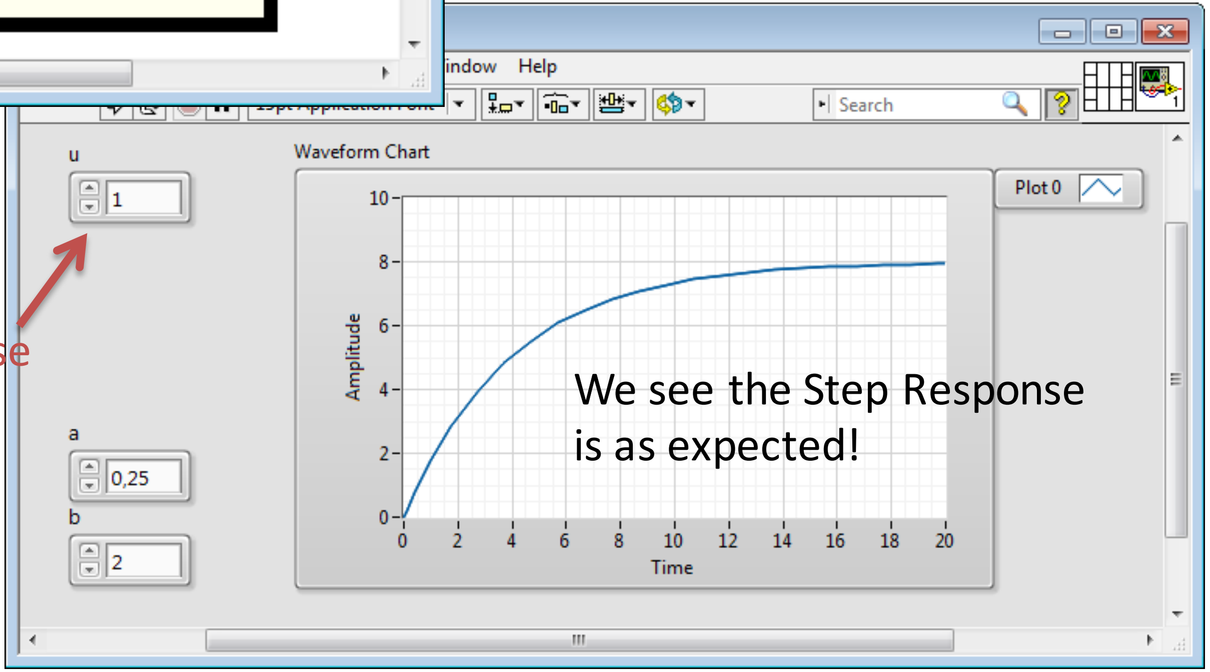


$$\dot{x} = -ax + bu$$

set $a = 0.25$ and $b = 2$

Try with different values for u

Step Response



We see the Step Response is as expected!

Correct results? – Check static response:

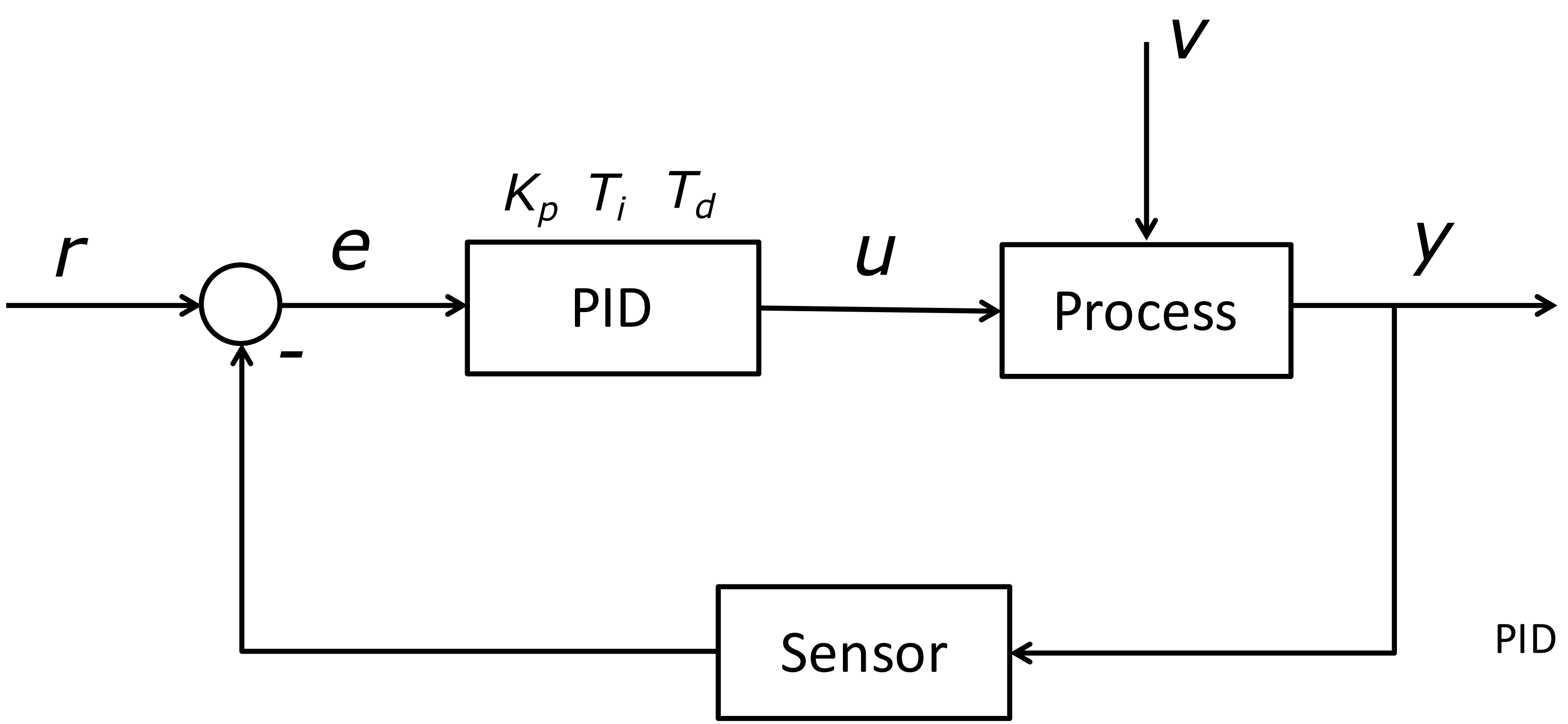
$$0 = -ax_s + bu_s$$

$$x_s = \frac{b}{a} u_s = \frac{2}{0.25} 1 = 8$$

PID Control in LabVIEW

Hans-Petter Halvorsen, M.Sc.

Control System



PID Algorithm:

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

r – Reference Value, SP (Setpoint), SV (Set Value)

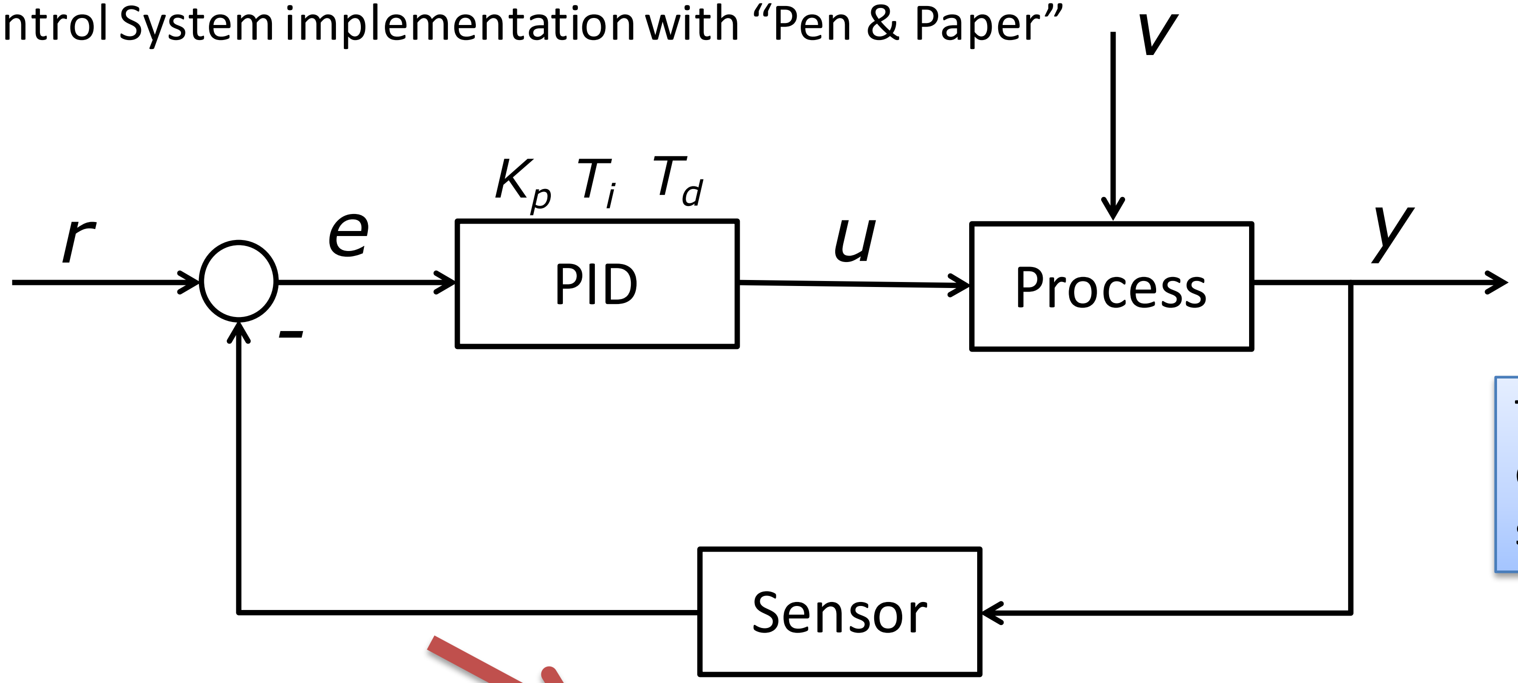
y – Measurement Value (MV), Process Value (PV)

e – Error between the reference value and the measurement value (e = r – y)

v – Disturbance, makes it more complicated to control the process

Kp, Ti, Td – PID parameters

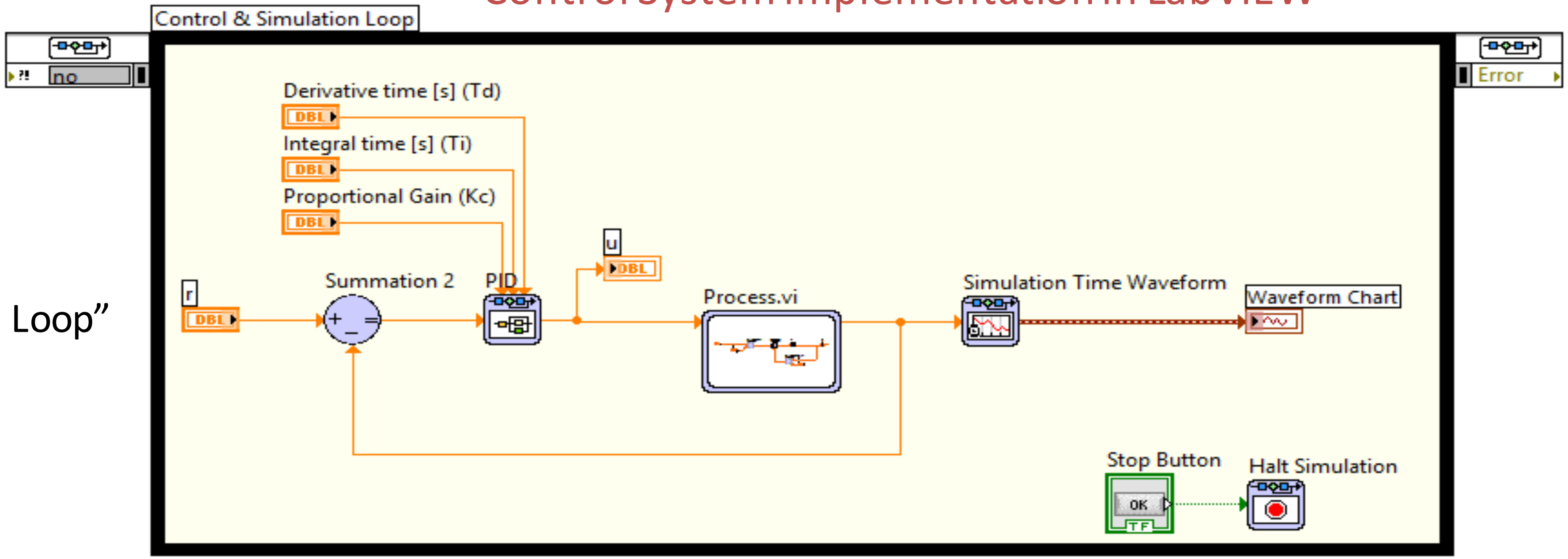
Control System implementation with “Pen & Paper”



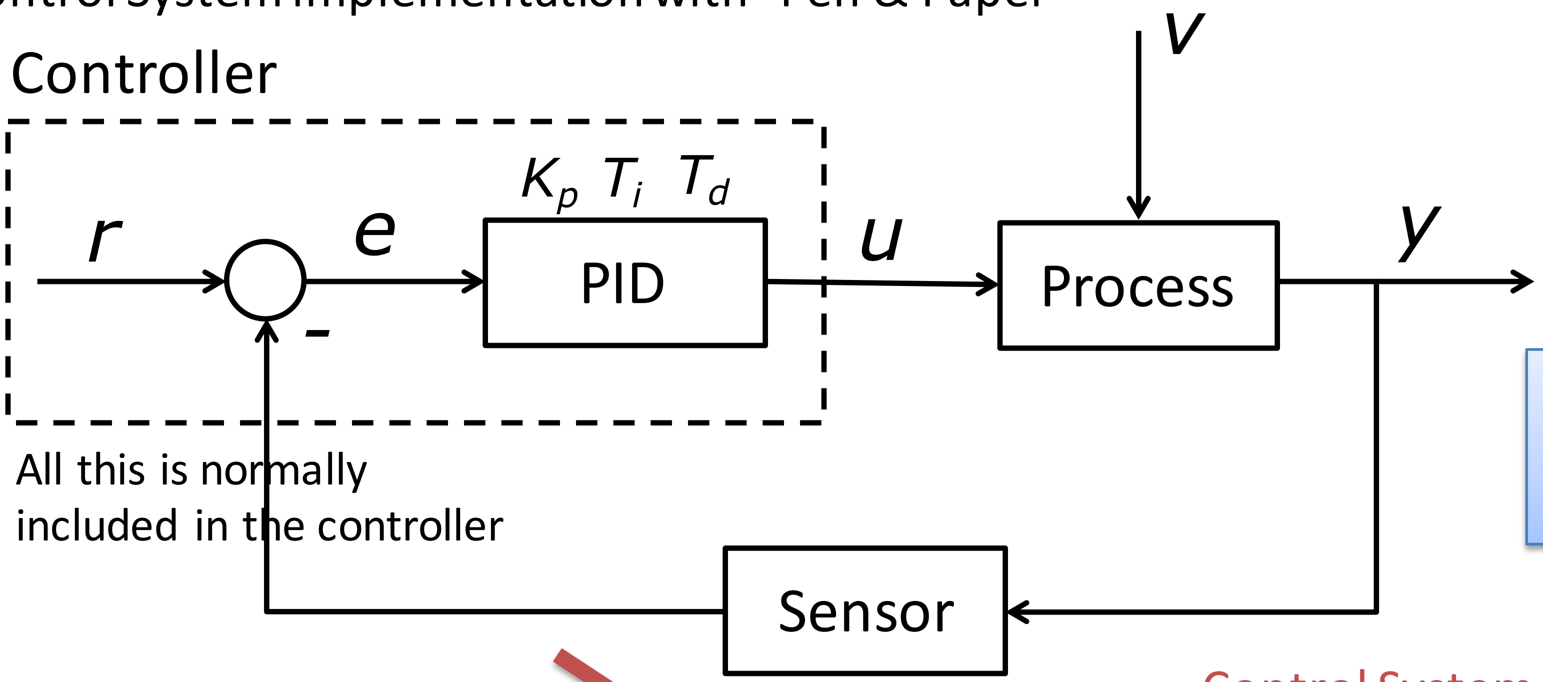
The transition from “paper” to LabVIEW is easy, because the implementation is very similar to the “paper” version

Control System implementation in LabVIEW

Here we have used the “Simulation Loop”



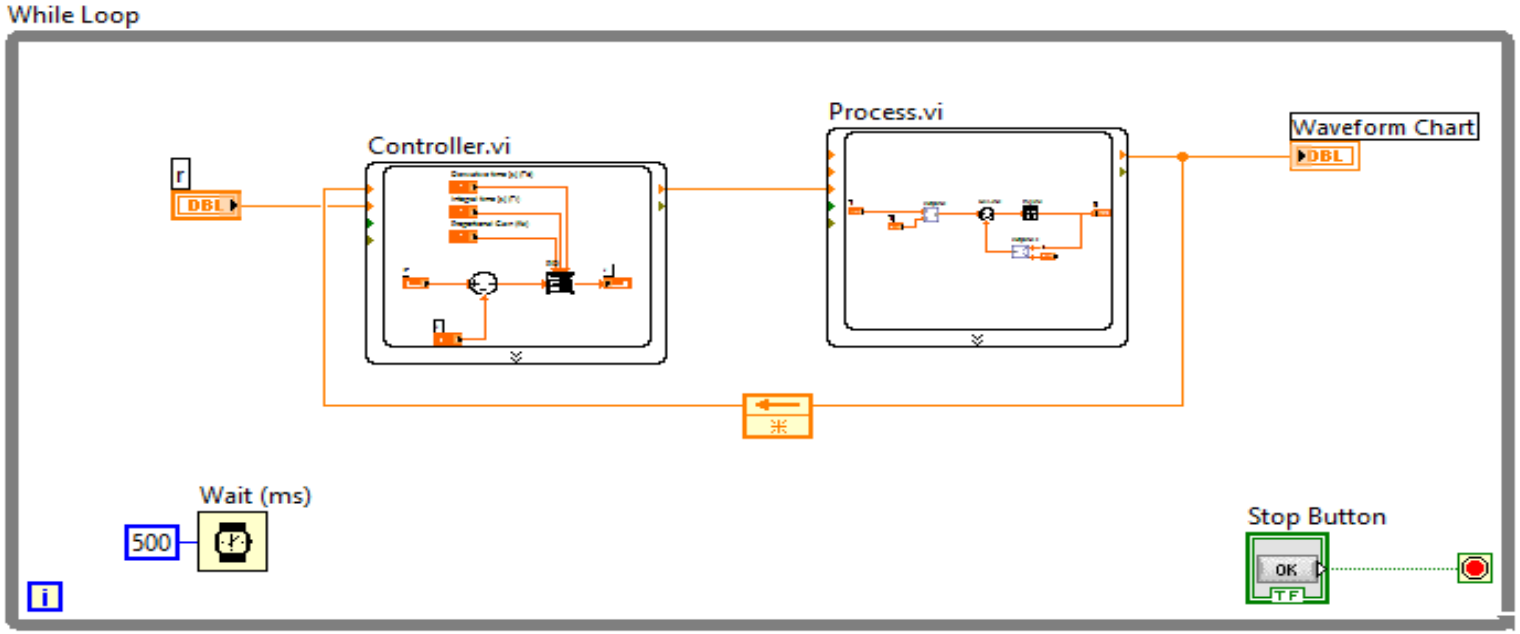
Control System implementation with "Pen & Paper"



All this is normally included in the controller

The transition from "paper" to LabVIEW is easy, because the implementation is very similar to the "paper" version

Control System implementation in LabVIEW

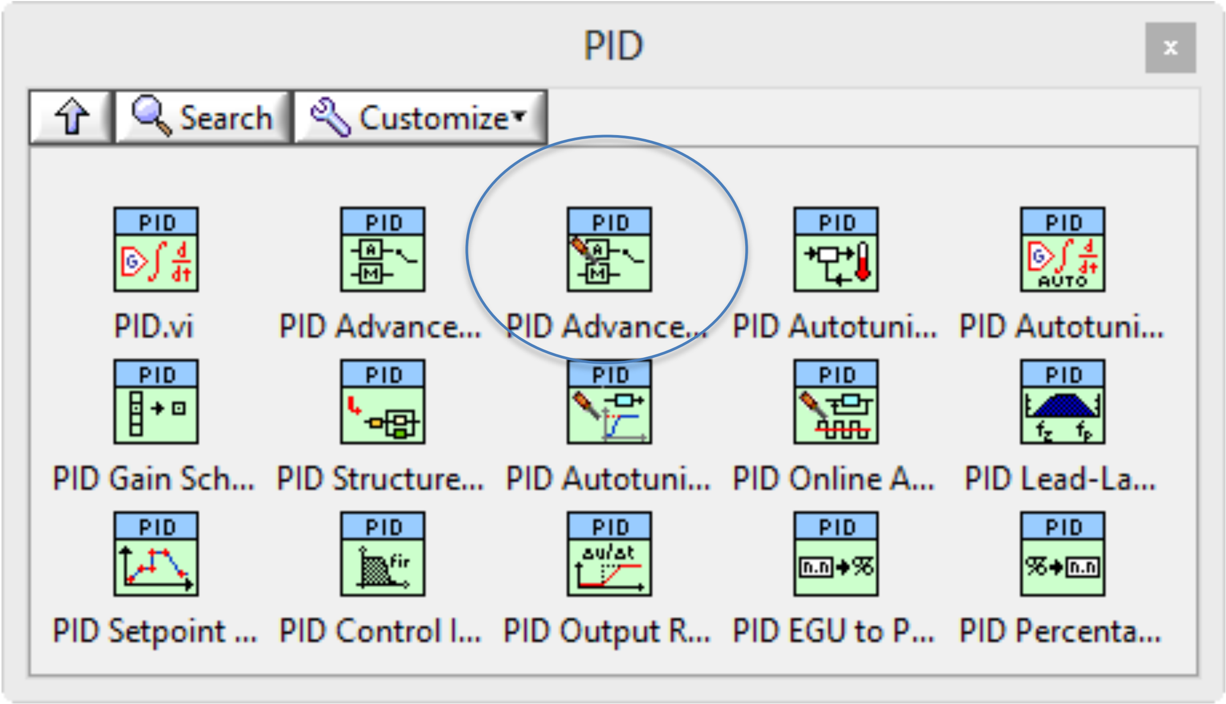


Here we have used an ordinary While Loop (which is recommended!)

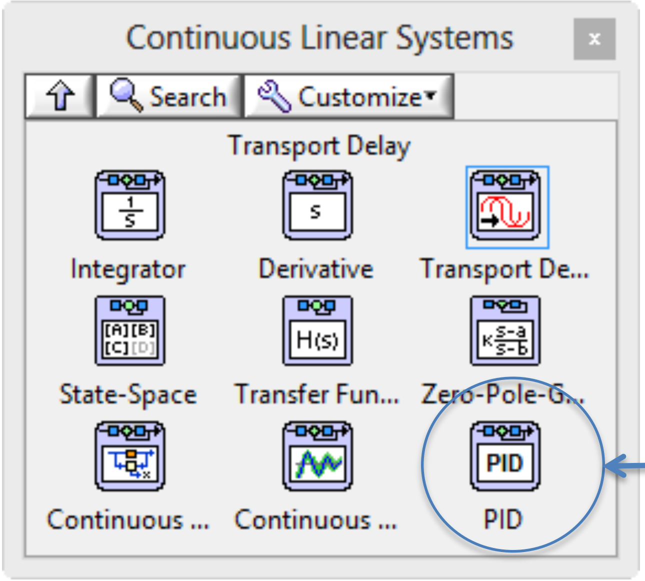
PID Control in LabVIEW

Alternative 1:

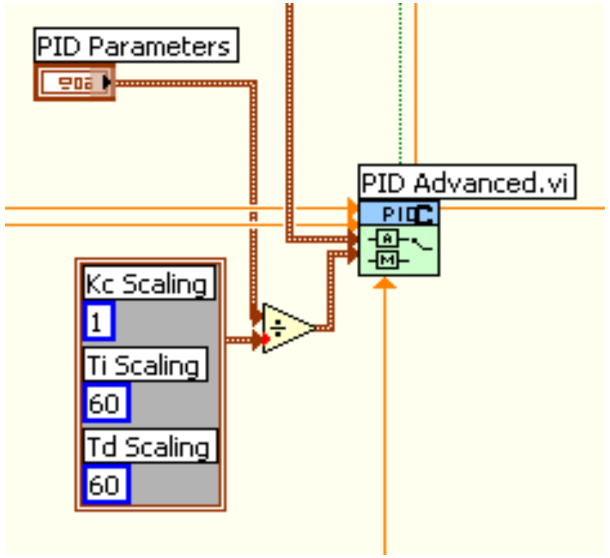
PID Palette in LabVIEW (PID Toolkit)



Alternative 2:



This alternative uses seconds!



Note! The functions "PID.vi" and "PID Advanced.vi" requires that Ti and Td are in minutes, while it's normal to use seconds as the unit for these parameters. You can use the following piece of code in order to transform them:

This means we enter values for Ti and Td in secons on the Front Panel and the values are converted to minutes in the code.

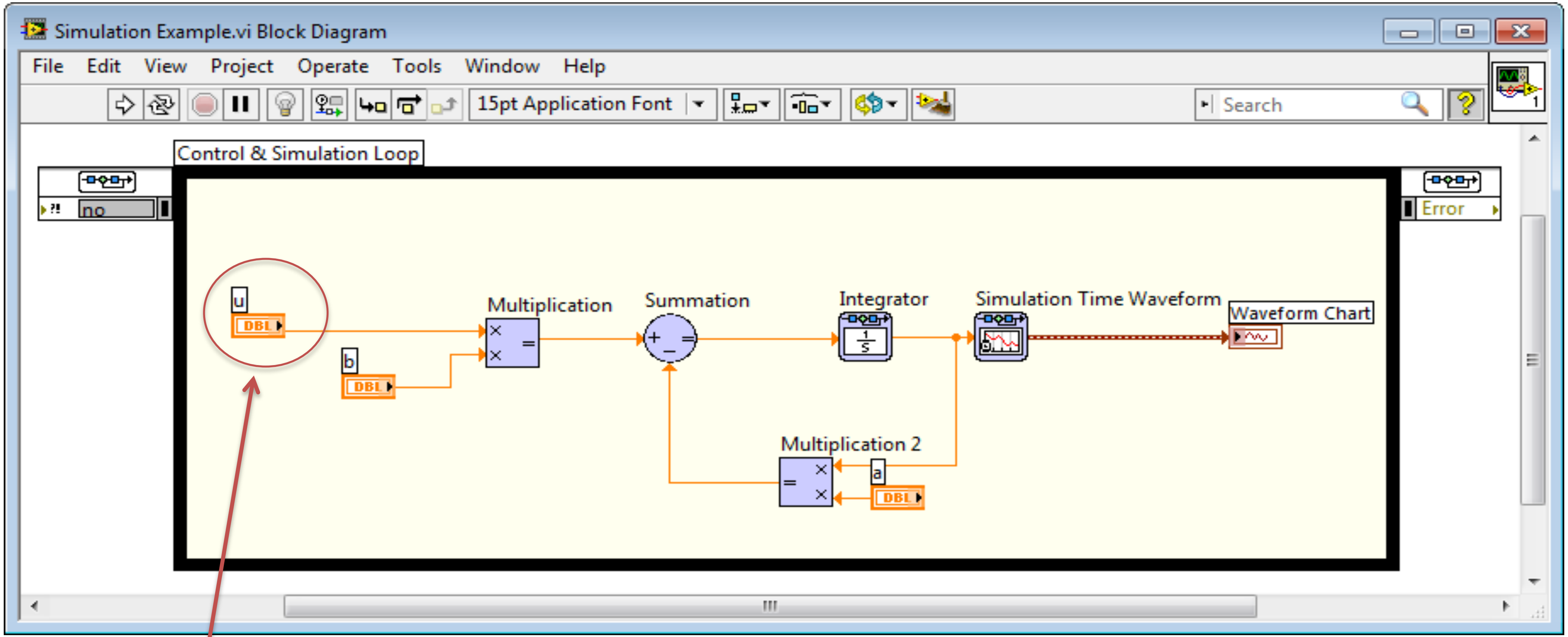
LabVIEW Example

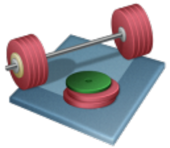
Hans-Petter Halvorsen, M.Sc.

LabVIEW PID Example

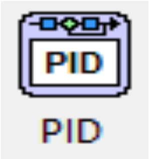
$$\dot{x} = -ax + bu$$

set $a = 0.25$ and $b = 2$





We will replace u in the previous example with the built-in PID Controller (use Alternative 2)

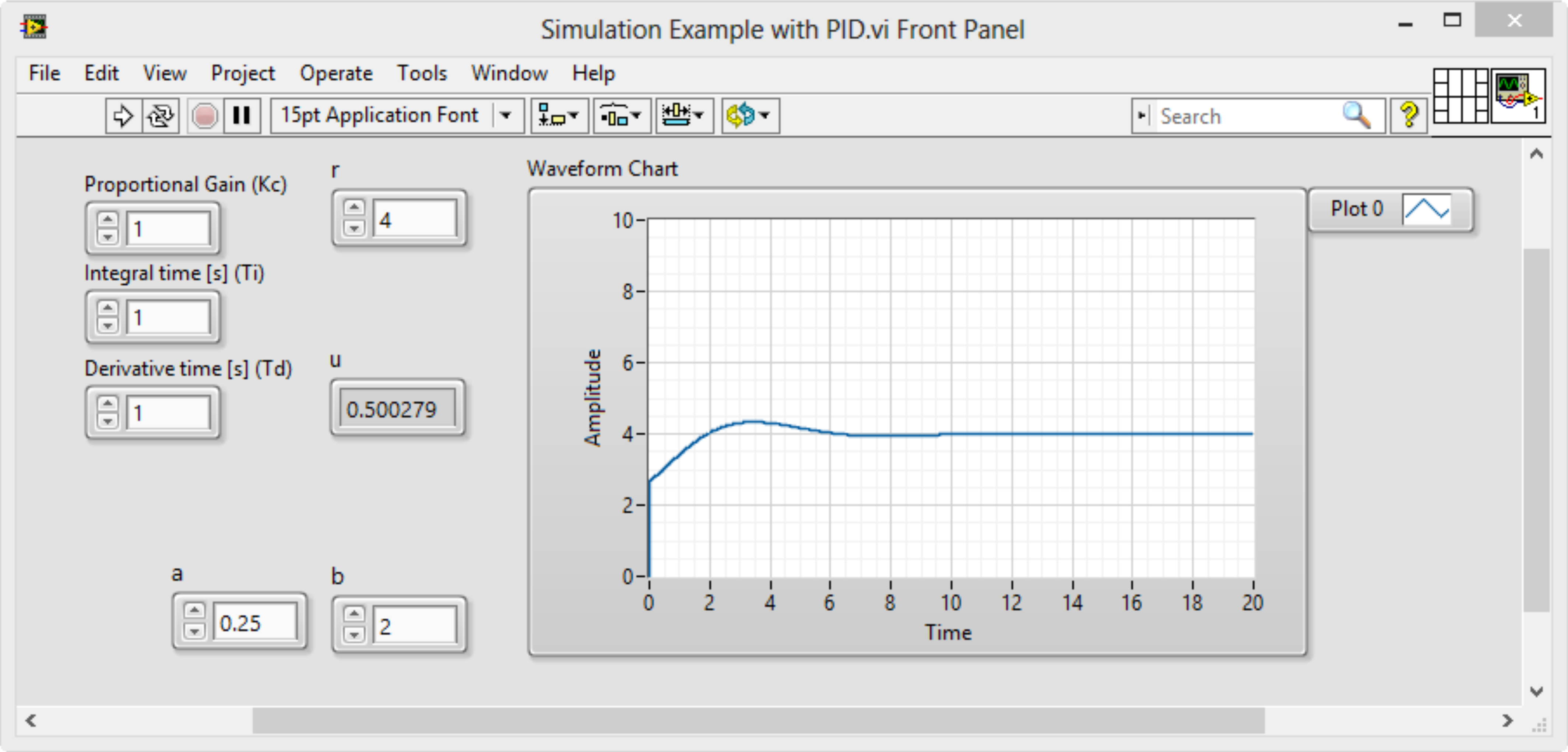


PID Example - Solutions

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

set $a = 0.25$ and $b = 2$

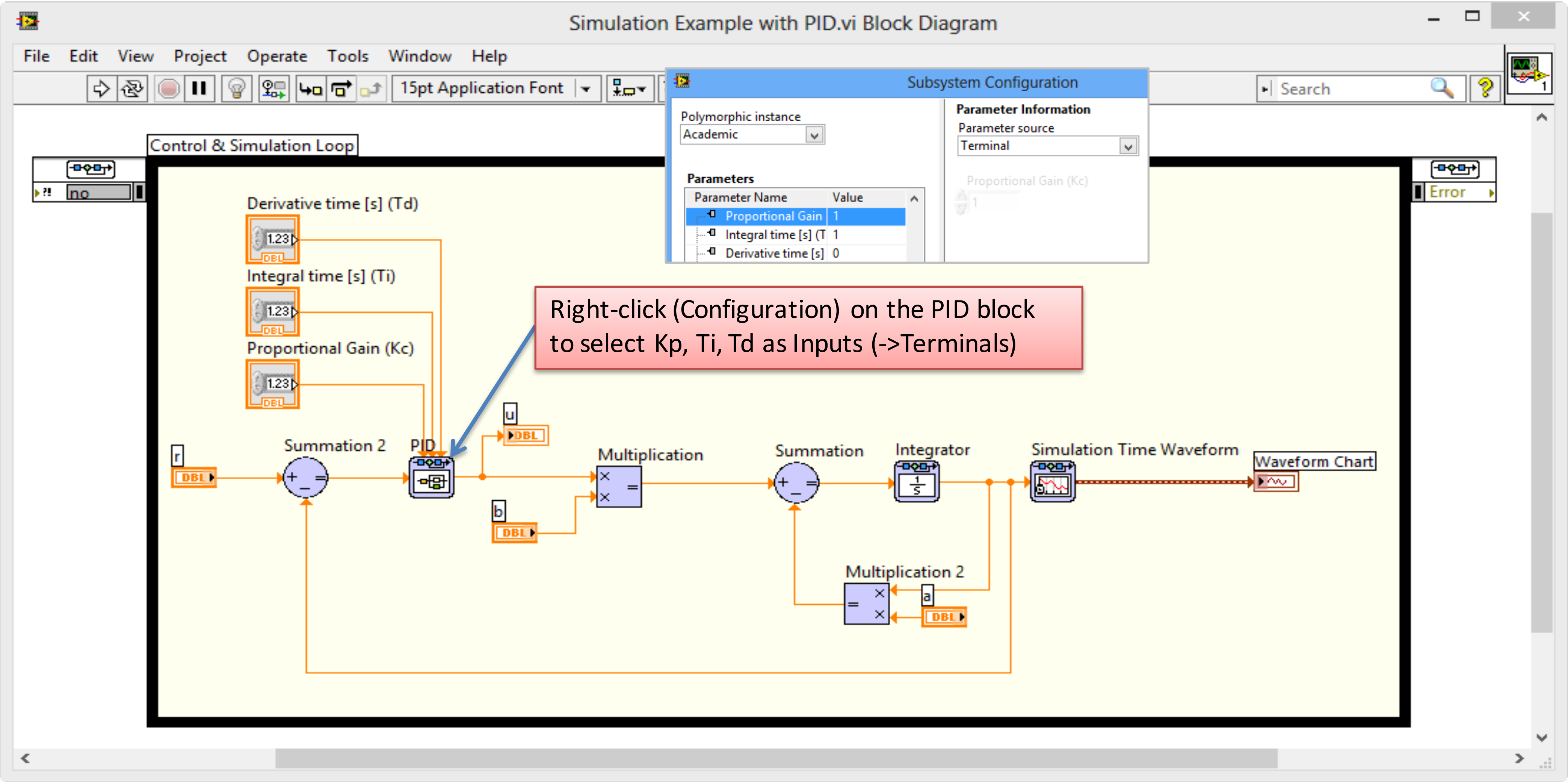
Front Panel:



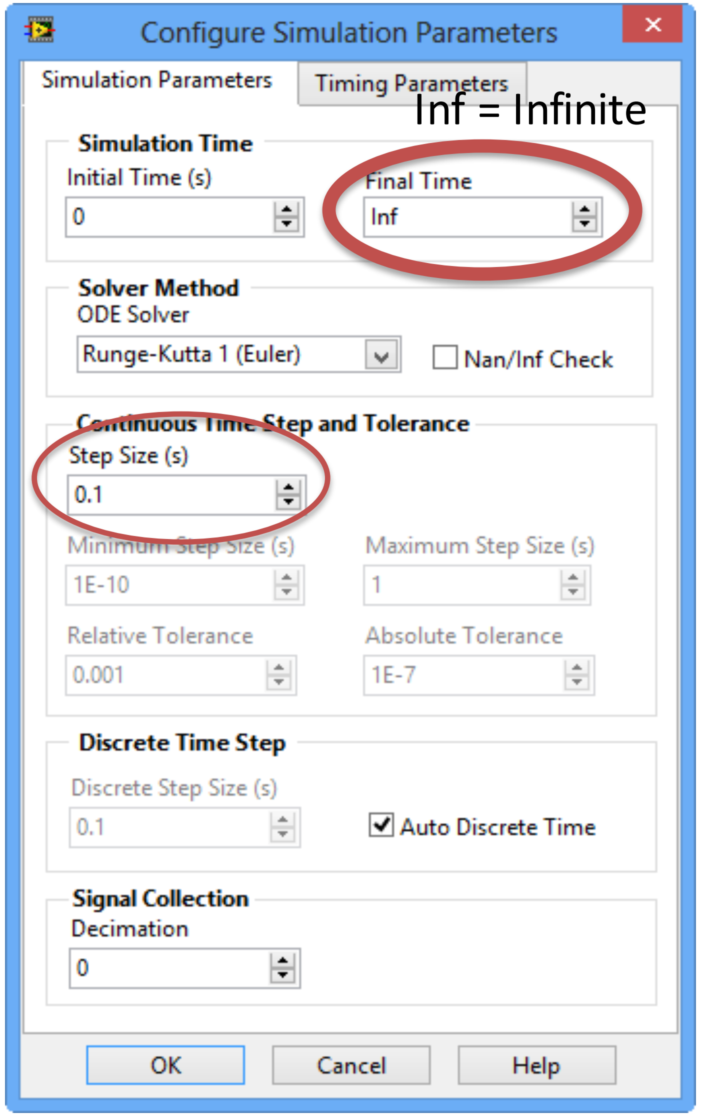
PID Example - Solutions

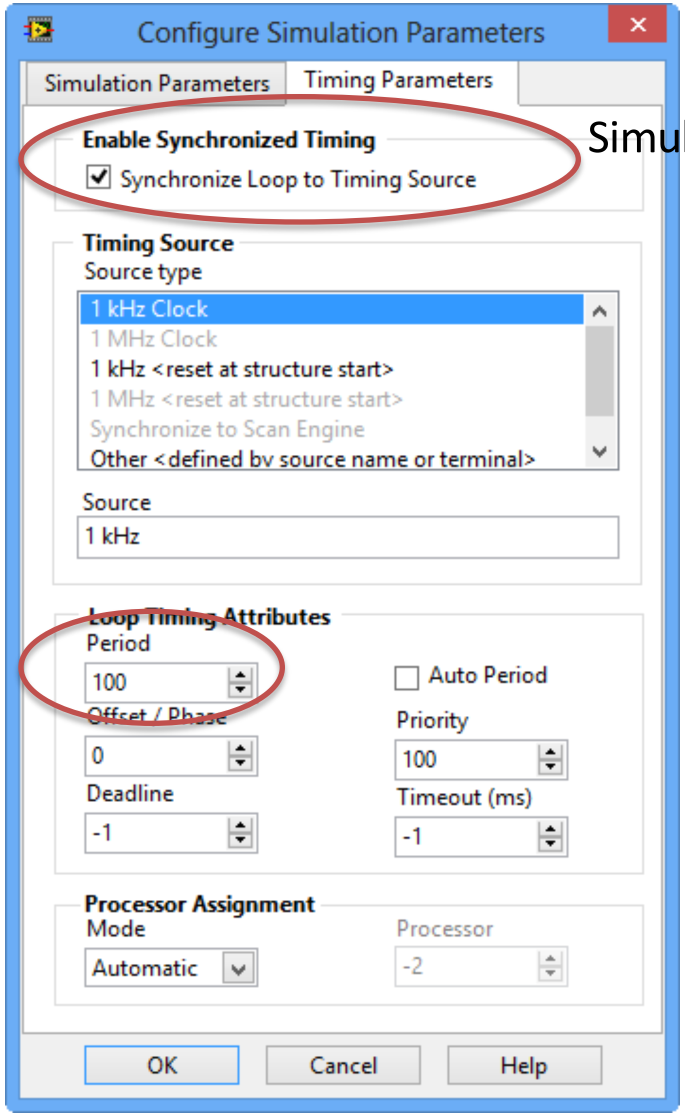
$$\dot{x} = -ax + bu$$ set $a = 0.25$ and $b = 2$

Block Diagram:



Next Step: Continuous Simulation

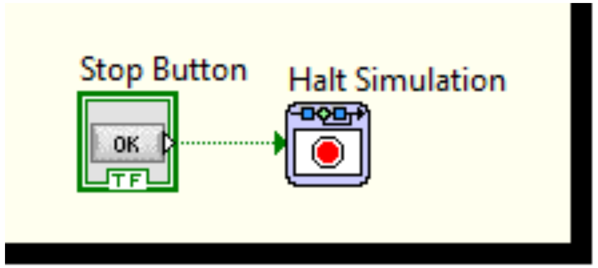




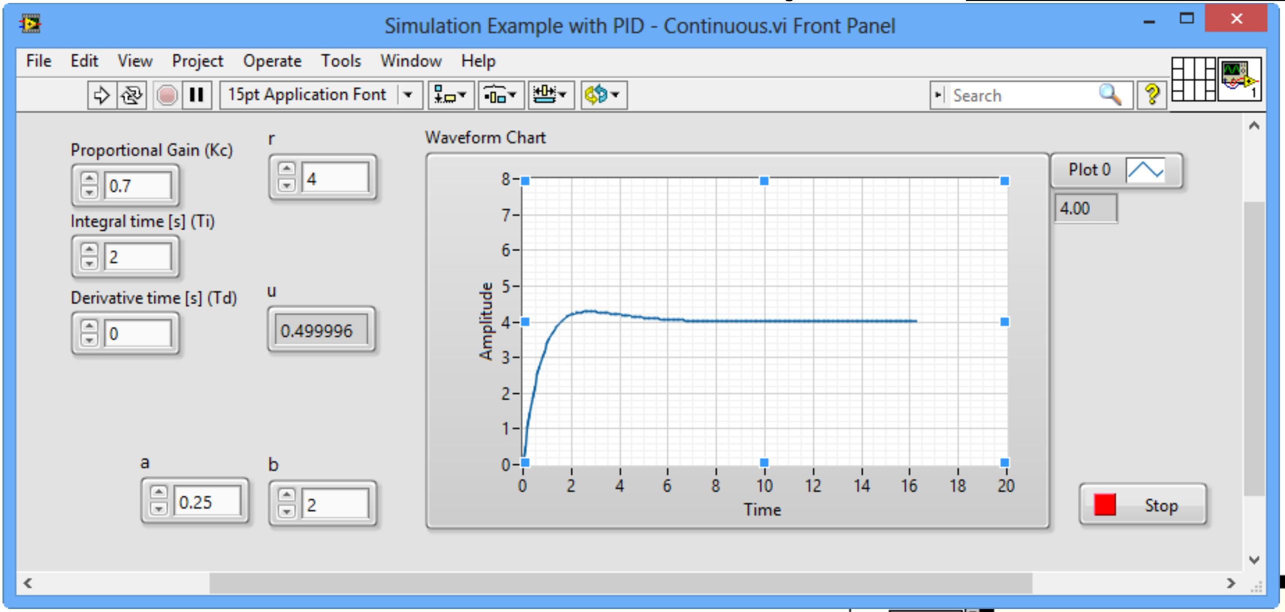
Simulation in “Real Time”

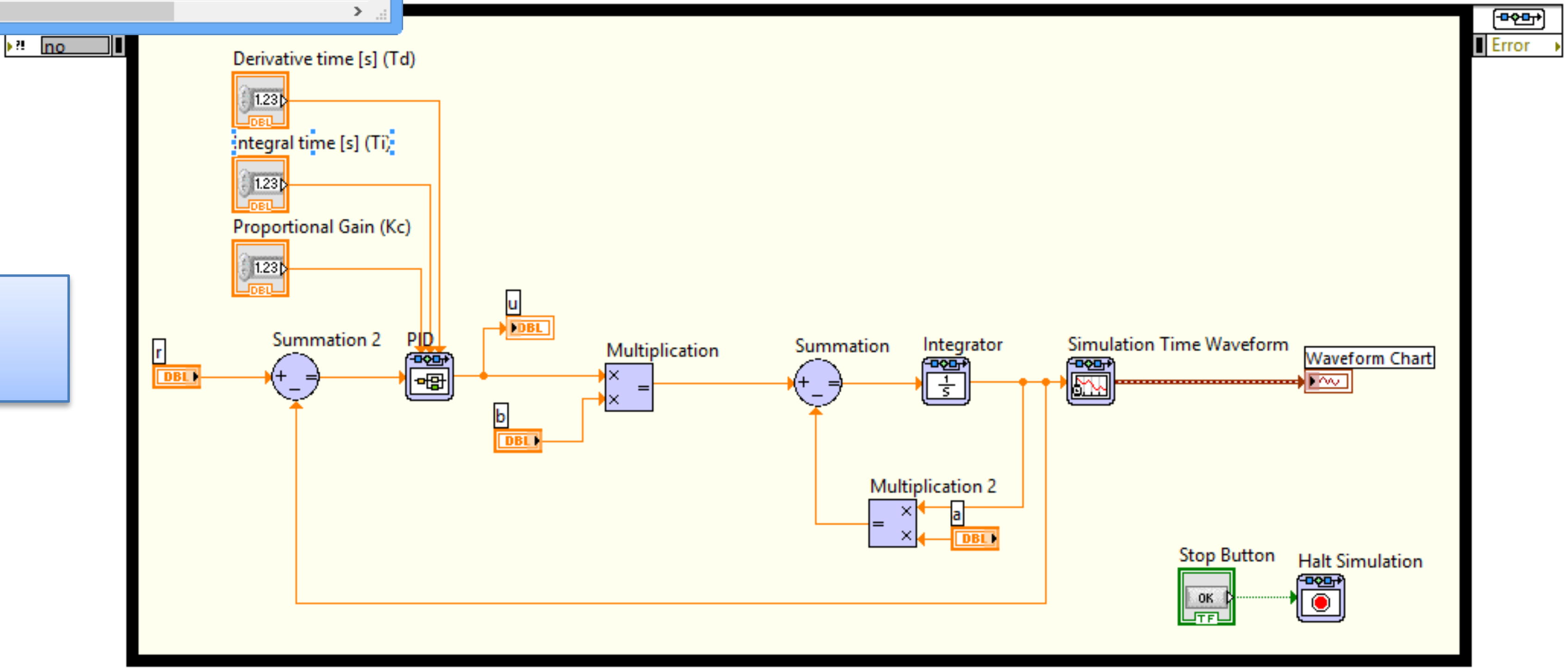
Right-click on the Simulation Loop border and select “Configure Simulation Parameters…”

Add a Stop Button and a “Halt Simulation” block



PID Example – Continuous Simulation - Solution





The Simulation now runs until you press the Stop button

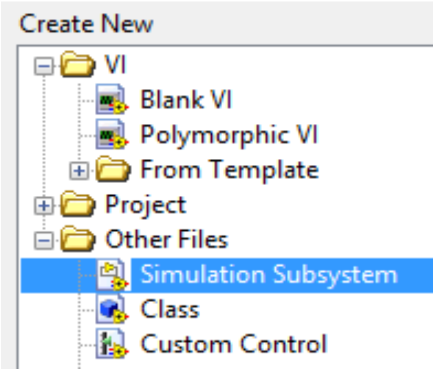
Simulation Subsystems

Hans-Petter Halvorsen, M.Sc.

LabVIEW Example

Hans-Petter Halvorsen, M.Sc.

Simulation Subsystem



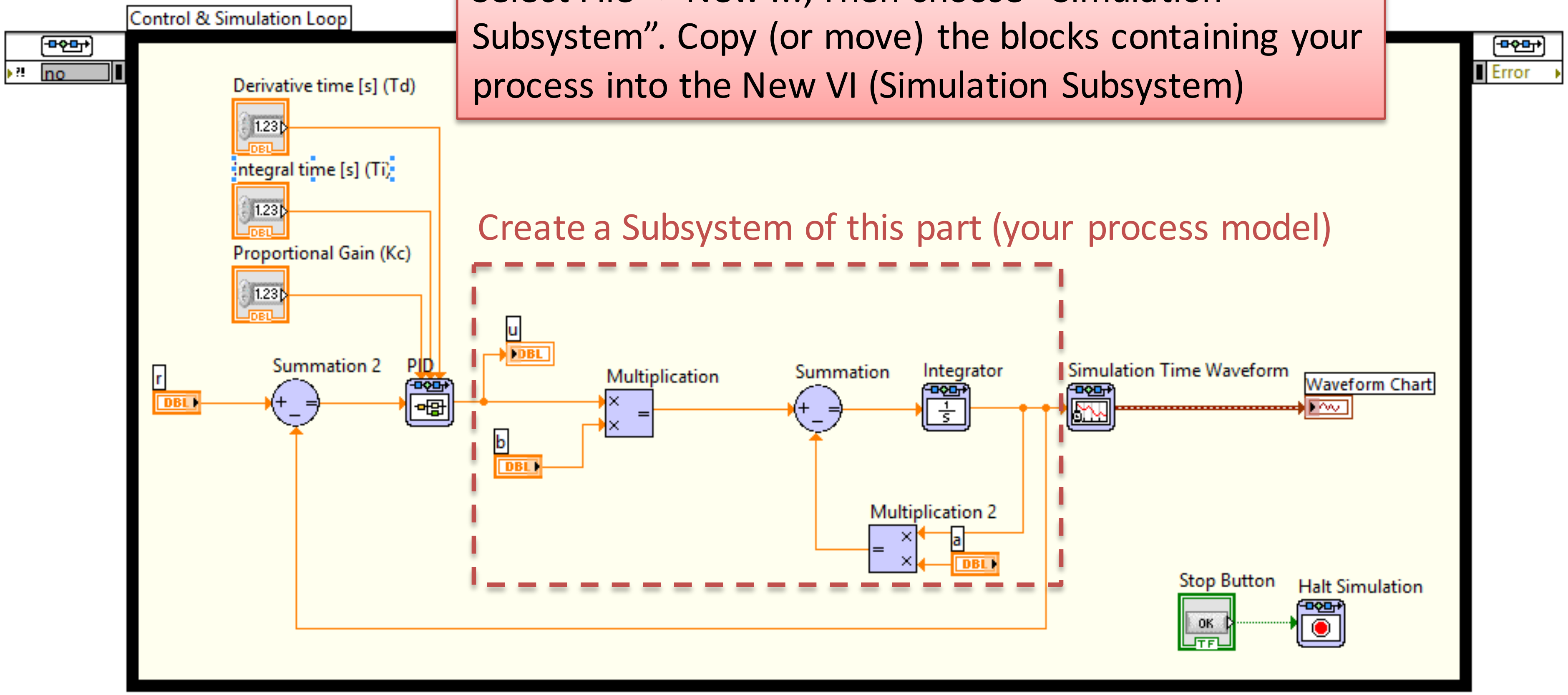
File -> New...

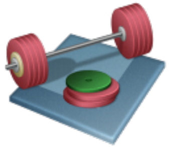
A Way to structure your code, similar to SubVIs

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

Select File -> New ..., Then choose “Simulation Subsystem”. Copy (or move) the blocks containing your process into the New VI (Simulation Subsystem)

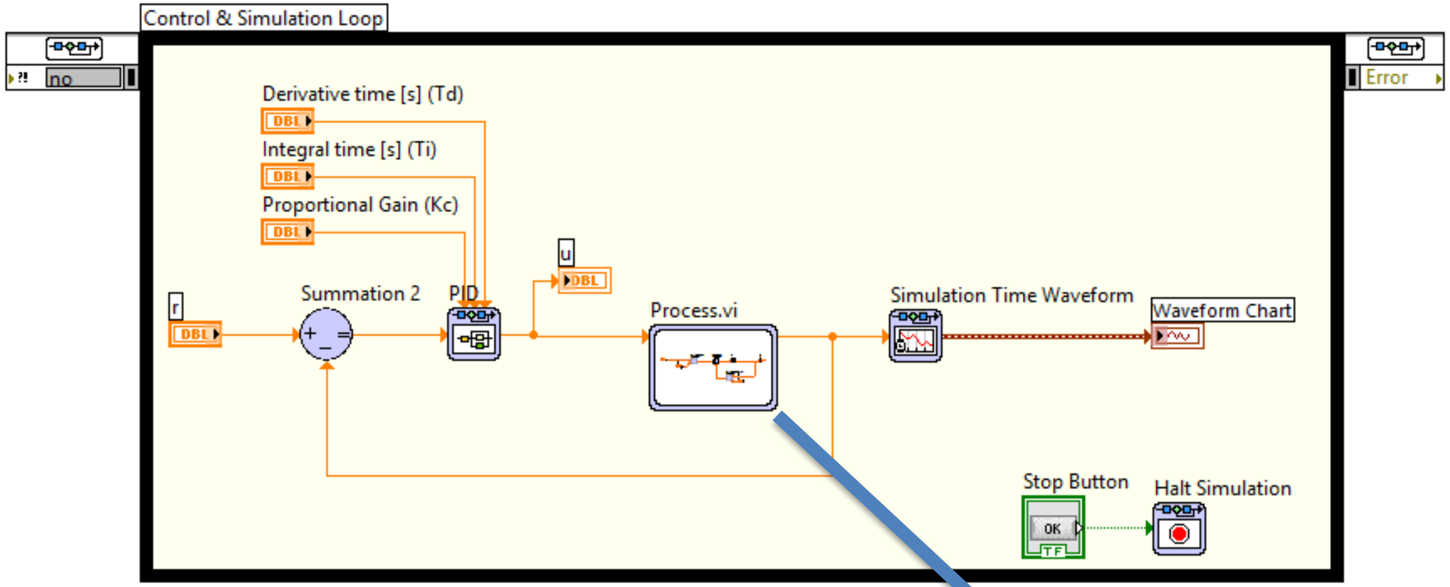
Create a Subsystem of this part (your process model)



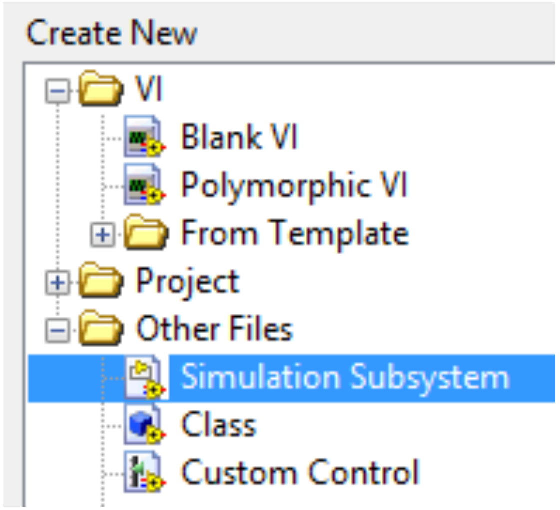


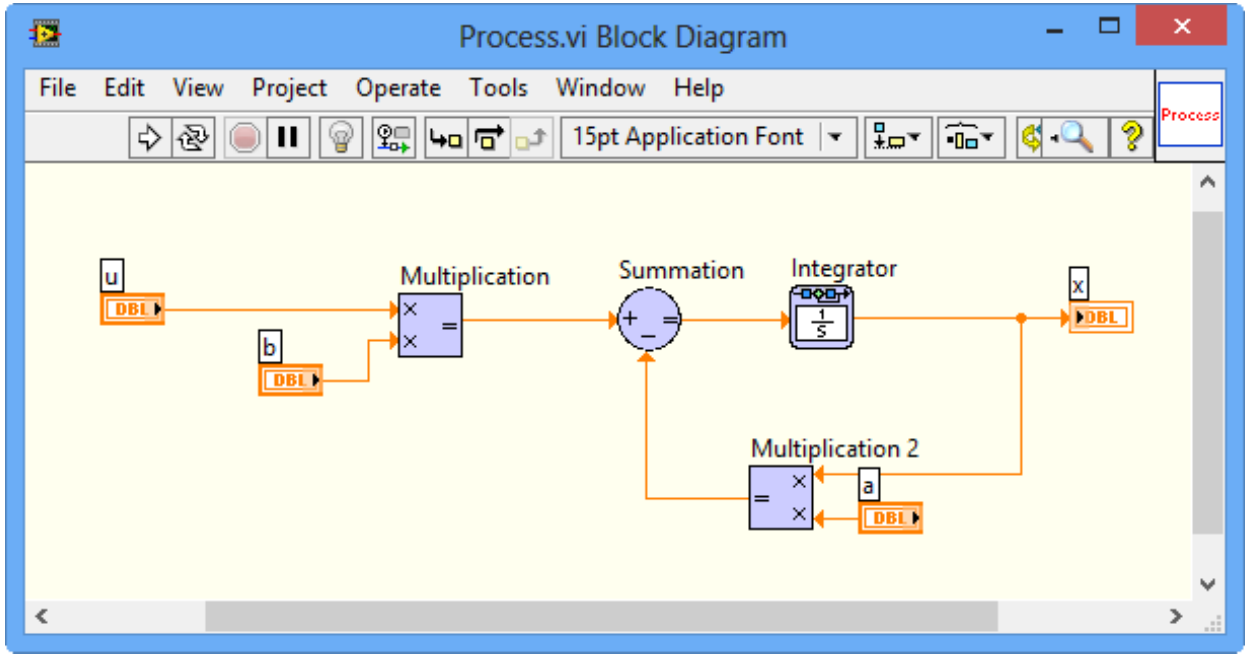
We will change your code above where you create a Simulation Sub System for your Process

Simulation Subsystem - Solutions

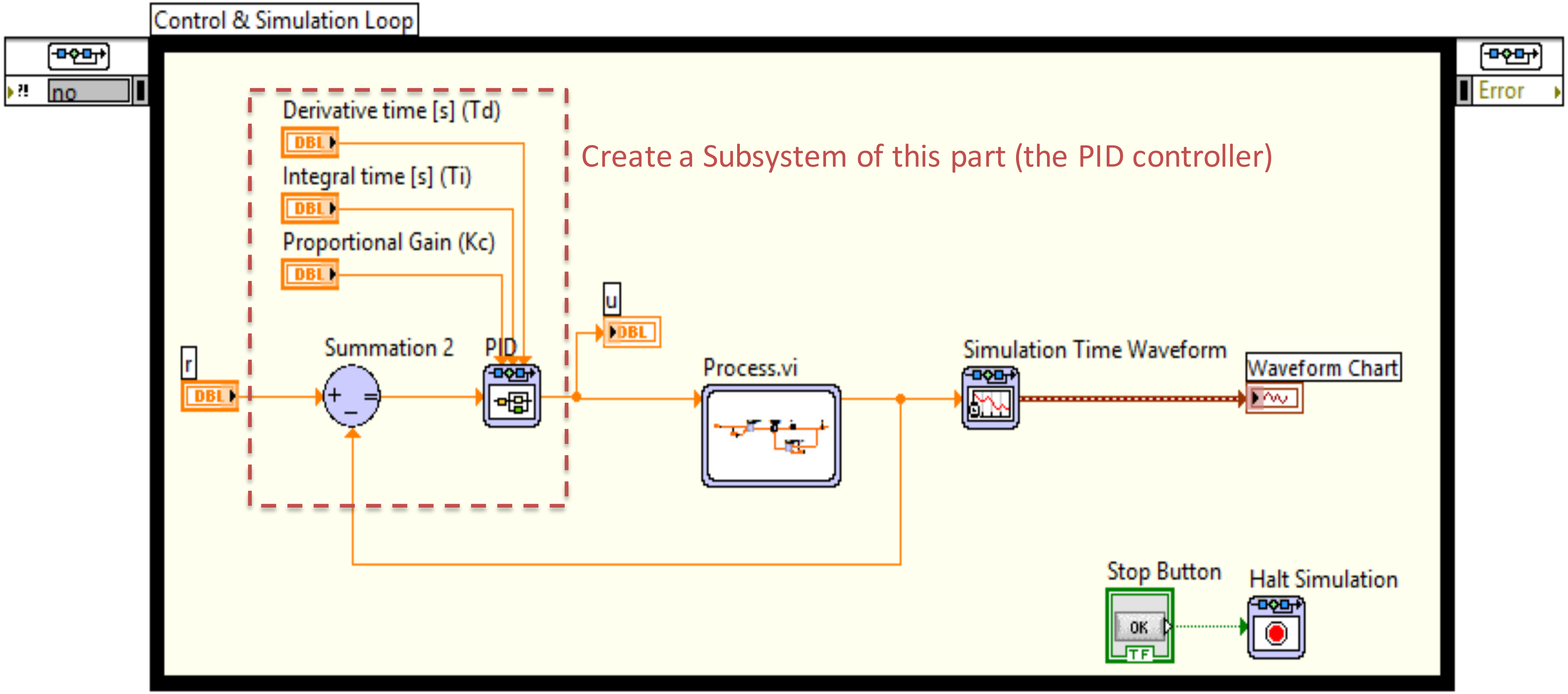


File -> New...



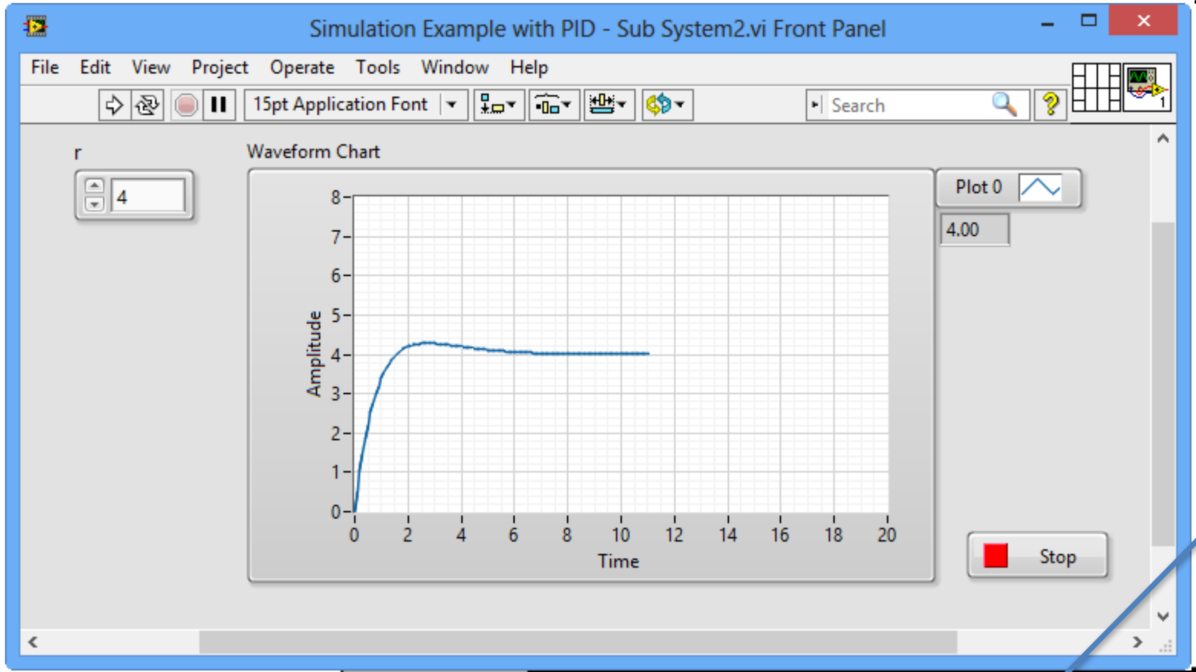


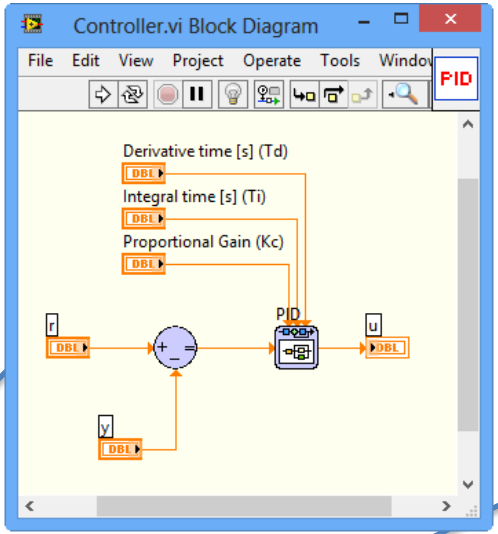
Simulation Subsystem 2 (PID Controller)

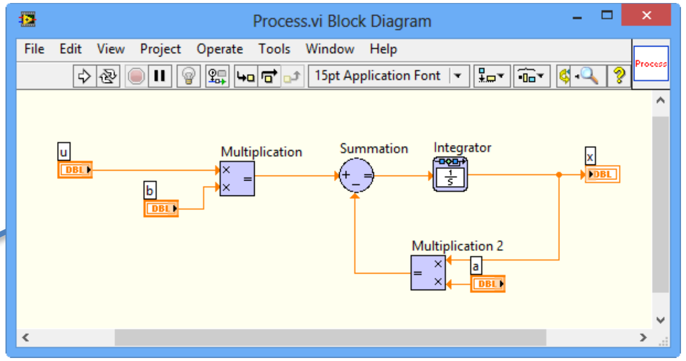


We will change your code above where you create a Simulation Sub System for the PID Controller as well.

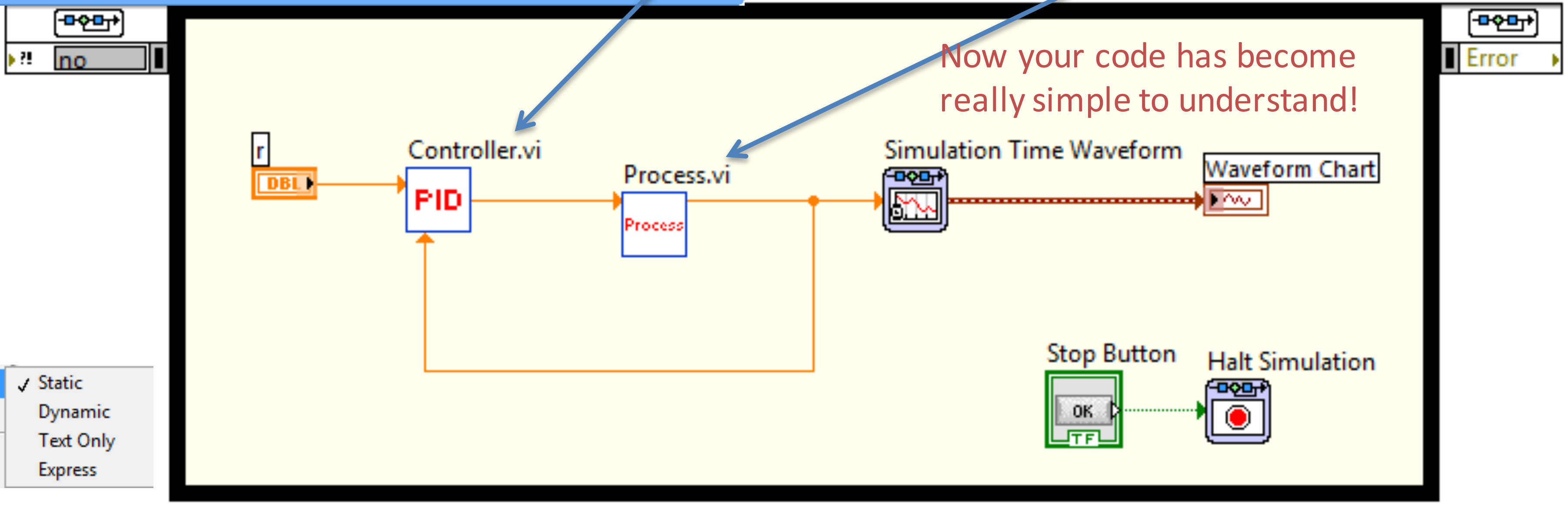
Simulation Subsystem – Solutions2



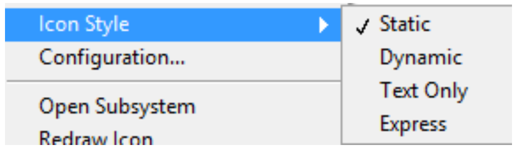




Simulation Sub Systems



Note! You may select different icon style

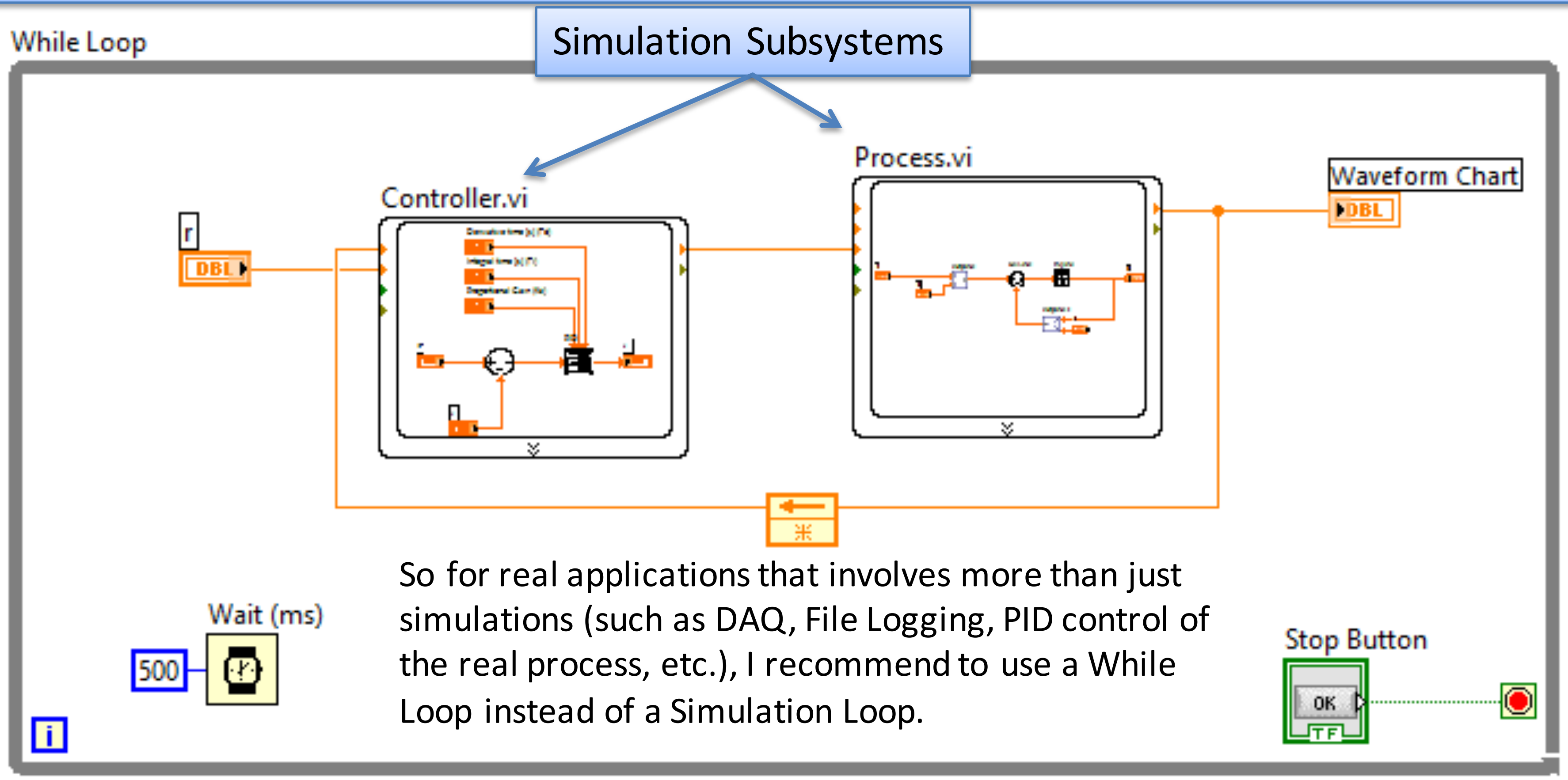


LabVIEW Example With While Loop

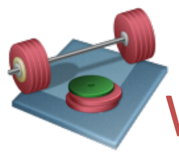
Hans-Petter Halvorsen, M.Sc.

Simulations using a While Loop

Note! The Simulation Loop has some drawbacks/is more complicated to use than an ordinary While Loop. If we use Simulation Subsystems, we can use them inside a While Loop instead! - which becomes very handy!



So for real applications that involves more than just simulations (such as DAQ, File Logging, PID control of the real process, etc.), I recommend to use a While Loop instead of a Simulation Loop.



We will add the Controller and Process Subsystems inside a While loop as shown above

Hans-Petter Halvorsen, M.Sc.

University College of Southeast Norway

www.usn.no

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

Blog: http://home.hit.no/~hansha/