UiA University of Agder Norway

Lecture #10

System modeling and SIL testing

MAS418

Programming for Intelligent Robotics and Industrial systems

Part II: PLC Software Development

Spring 2022

Daniel Hagen, PhD

UiA

Previous Lecture

Object oriented PLC Programming

Lecture	Topic	Week
#7	**Introduction to part II**	9 – Thursday 3/3
#8	**Procedural oriented PLC programming**	10 – Thursday 10/3
#9	**Object oriented PLC Programming**	11 – Thursday 17/3
#10	**System modeling and SIL testing**	12 – Thursday 24/3
#11	**ROS2 interface**	13 – Thursday 31/3
#12	**Machine interface**	14 – Thursday 7/4

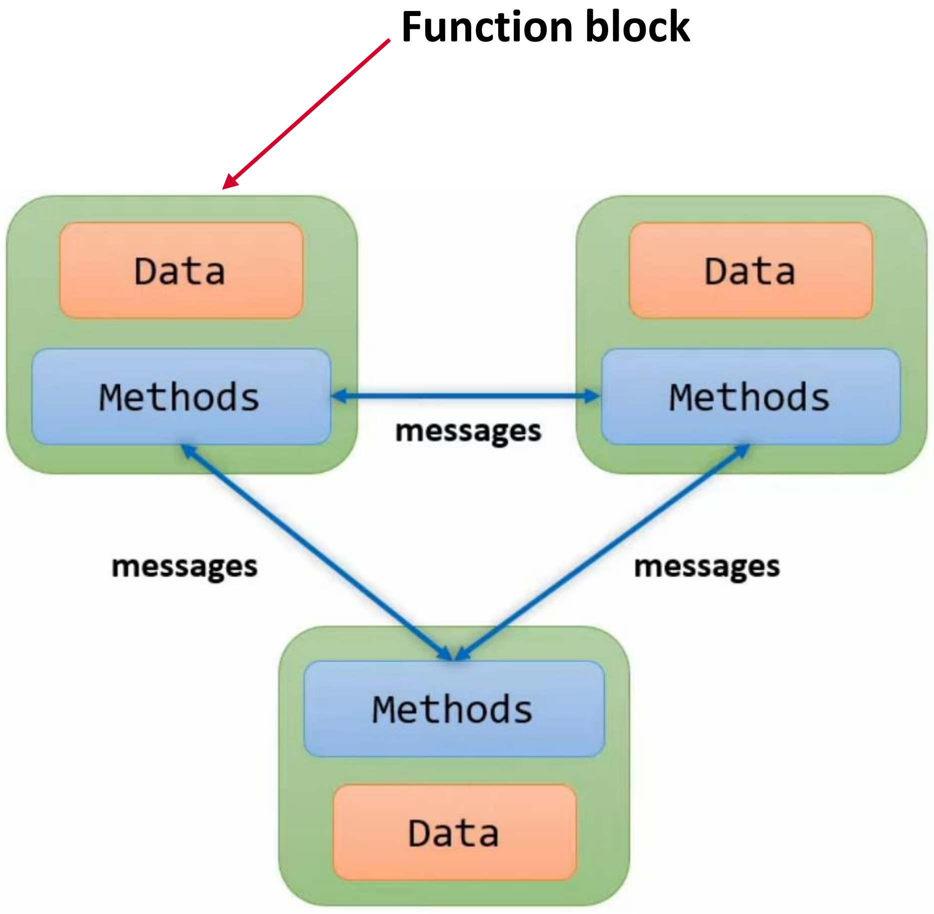
Previous Lecture

Object oriented PLC Programming

I. **Function blocks**
   - Introduction
   - Function blocks
   - Methods
   - Inheritance
   - Interfaces

II. **Interfaces**
   - Conditional statements
   - CASE instruction
   - FOR loops
   - WHILE loops

III. **LAB Lecture (Demo)**

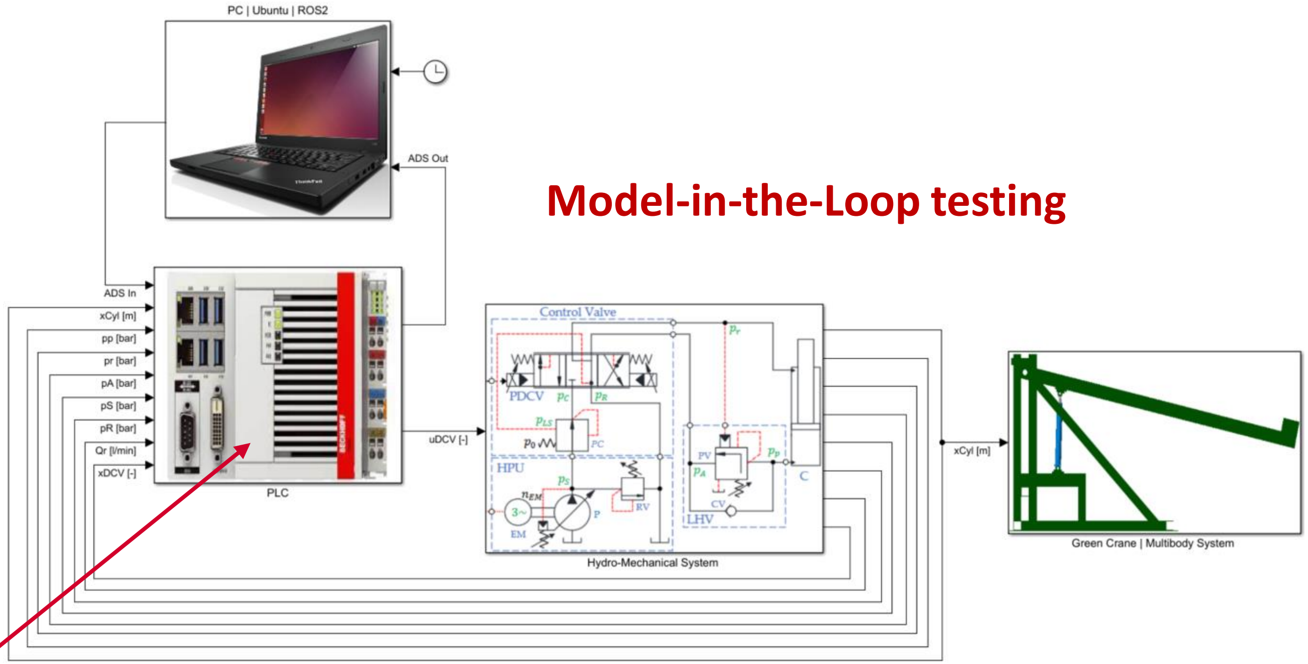


UiA

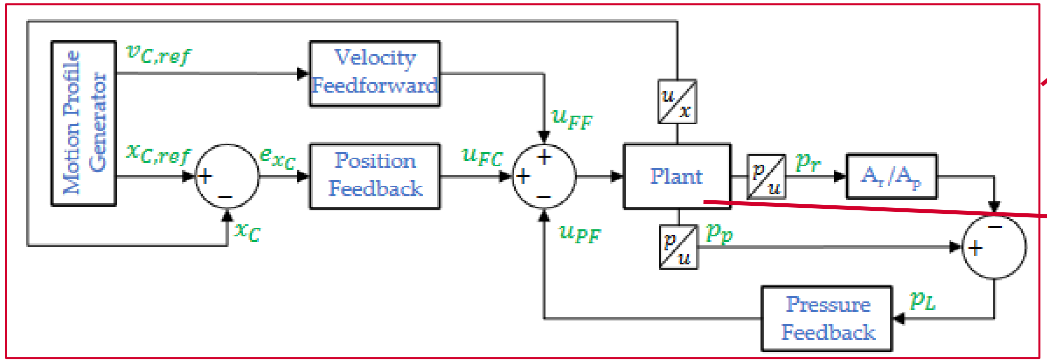
System modeling and SIL testing

Today's lecture is a kick-off for the upcoming, mandatory, LAB exercises

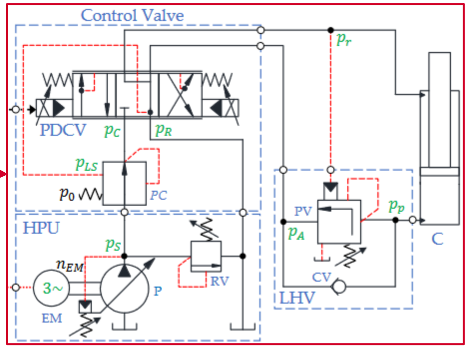
- **System modeling**
  - Safety system
  - Visualization/PLC HMI
  - Open-loop velocity control
  - Closed-loop position control
  - Active damping
- **Software-in-the-Loop testing**



Model-in-the-Loop testing



Simulation Model



Remaining LAB exercises

- **LAB** exercise **#10: Programming of PLC-based control system for a hydraulically actuated single-boom crane**
  - Structured Text programming in TwinCAT 3 based on given system model
  - Mandatory individual – deadline **03. April** by sending link to Git repo (or Archive .zip) by e-mail

- **LAB** exercise **#11: Software-in-the-loop (SIL) testing in TwinCAT and interface with ROS2**
  - **In a group of 3-4 persons do the following:**
    1. Implement the program from LAB #10 and program a simulator representing the hydro-mechanical system of the green crane (simulator program should run on a separate task)
    2. Complete the program and visualization (control box) and test it against the simulator (SIL testing)
    3. Implement and test interface with ROS2 sending reference signal(s) to the motion controller
  - Mandatory group – deadline **18. April** by sending link to Git repo (or Archive .zip) and additional video demonstrating the test and showing all groups members by e-mail (link to cloud sharing)

- **LAB** exercise **#12: Experimental testing of the distributed control system on the single-boom crane**
  - **Test #1:** Operator-in-the loop control using the joystick – visualization in Rviz of crane boom when lifting and lowering the real crane boom
  - **Test #2:** Send motion reference command from **ROS2** to **PLC** resulting in lifting and lowering of the real crane boom
  - Mandatory group – deadline **05. May** by sending link to Git repo (or Archive .zip) and additional video from the two tests and showing all groups members by e-mail (link to cloud sharing)

UiA

Remaining LAB exercises

Group work (LAB #11 and #12)

Gruppe tilgang Grønn Kran

Tidspunkt	Tirsdag 19.04	Torsdag 21.04	Tirsdag 26.04	Torsdag 28.04
08:00-10:00	Gruppe 1	Gruppe 5	Gruppe 4	Gruppe 8
10:00-12:00	Gruppe 2	Gruppe 6	Gruppe 3	Gruppe 7
12:00-14:00	Gruppe 3	Gruppe 7	Gruppe 2	Gruppe 6
14:00-16:00	Gruppe 4	Gruppe 8	Gruppe 1	Gruppe 5

Ikke tildelte studenter (20)

Søk blant brukere

- Alexander Bonner Aksnes
- Rolkana Alo
- Jørgen Dale
- Eskil Gresen Gaustad
- Rasmus Als Hansen
- Benjamin Årøy Ims
- Jon-Erick Kloumann
- Ravi Kumar
- Simon Marheim
- Lars Muggerud
- Endre Myhre
- Magnus Ranestad
- Bjørn Håvard Halvorsen Saghaug
- Kristoffer Sand
- Tarjei Skotterud
- Eirik Magnus Skår
- Alexander Sterk-Hansen
- Juan Nils Thomas Ugland
- Hipolit Edward Wilczek
- Ørjan Øvsthus

Grupper (8)

- Prosjektgruppe 1 — 1 / 3 studenter
  - Tommy Berg Sivertsen
- Prosjektgruppe 2 — 2 / 3 studenter
  - Martin Dahlseng Hermansen
  - Martin Mæland
- Prosjektgruppe 3 — 2 / 3 studenter
  - Gaute Myrland
  - John-Arne Nyheim
- Prosjektgruppe 4 — Full 3 / 3 studenter
  - Ane Sofie Andersen
  - Henrik Borge
  - Pål Kristian Ofstad
- Prosjektgruppe 5 — 0 / 3 studenter
- Prosjektgruppe 6 — 0 / 4 studenter
- Prosjektgruppe 7 — 0 / 4 studenter
- Prosjektgruppe 8 — 0 / 4 studenter

UiA

Overview

Introduction

Part I
System modeling

Part II
Software-in-the-Loop testing

Part III
Demo?

Summary

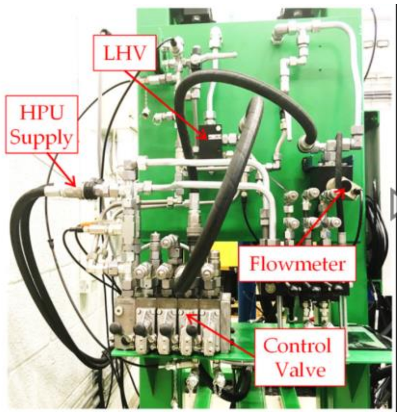
UiA

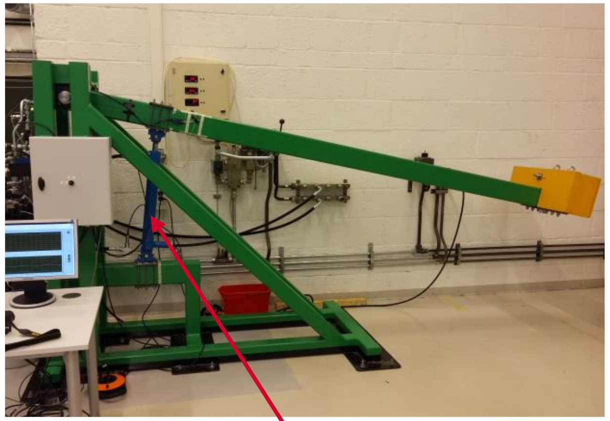
Part I: System modeling

1. System overview
2. Safety functions
3. Visualization/PLC HMI
4. Motion controller
5. Control input

UiA

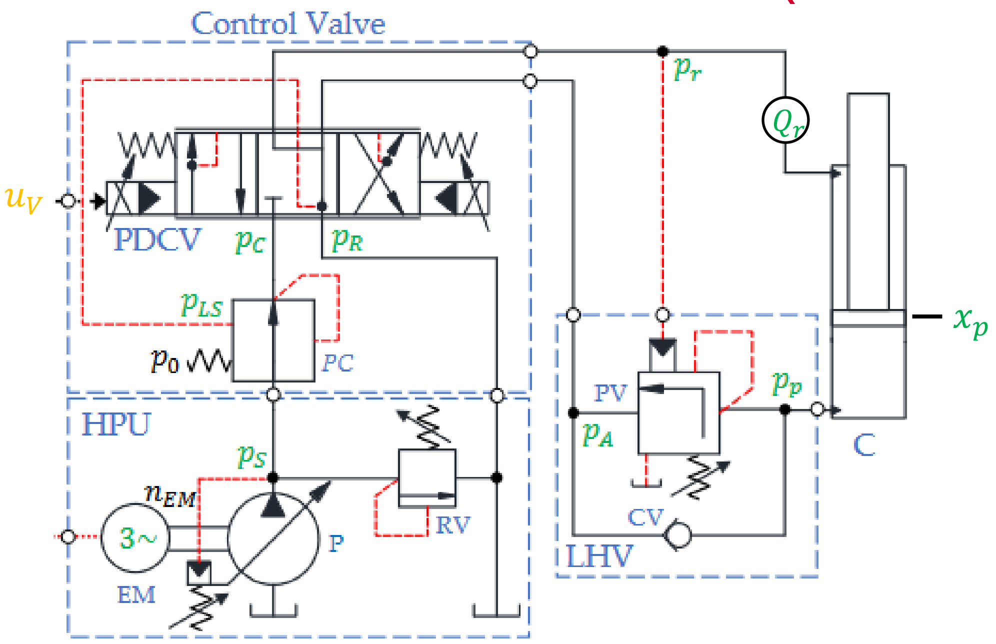
System overview





Relevant IO:

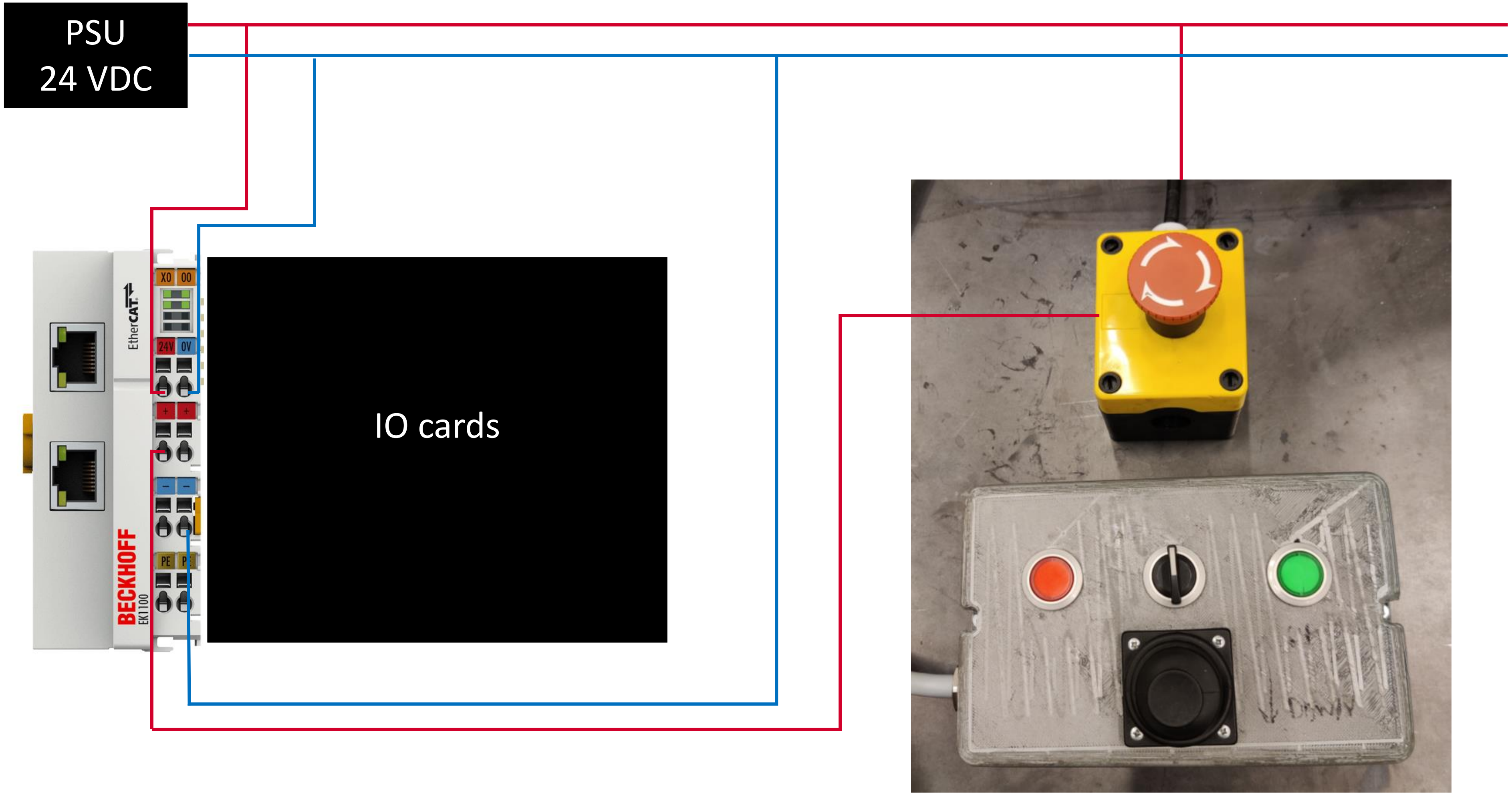
- Cylinder piston position sensor (input: 0…0.5 [m]) - x_p
- Pressure sensors (Input: 0…400 [bar])
  - Supply pressure - p_S
  - Return pressure - p_R
  - Cylinder piston-side pressure - p_p
  - Cylinder rod-side pressure - p_r
  - Pressure between **Control Valve** and **L**oad-**H**olding **V**alve (**LHV**) - p_A
- Flow sensors (Input: 0…30? [l/min])
  - Rod-side cylinder outlet flow - Q_r
- Control Valve (output: -1…1 [-]) - u_V



UiA

Safety functions

Remote IO (RIO)



Safety functions

- **OFF:** bEnable = FALSE

- **MANUAL:** Operator-in-the-loop control with joystick input

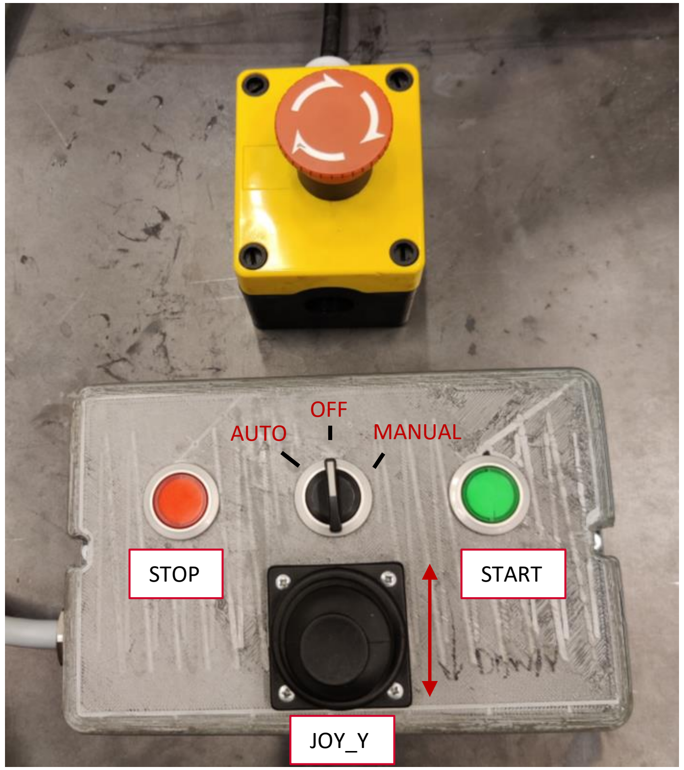
IF bEnable
AND bManualMode
AND bStart
AND NOT(bStop)

→ **FB_Joystick** → **FB_OpenLoop Control** → fValveOutput

- **AUTO:** Automatic motion reference generation and position control

IF bEnable
AND bAutoMode
AND bStart
AND NOT(bStop)

→ **FB_Motion RefGen** → **FB_Position Control** → fValveOutput



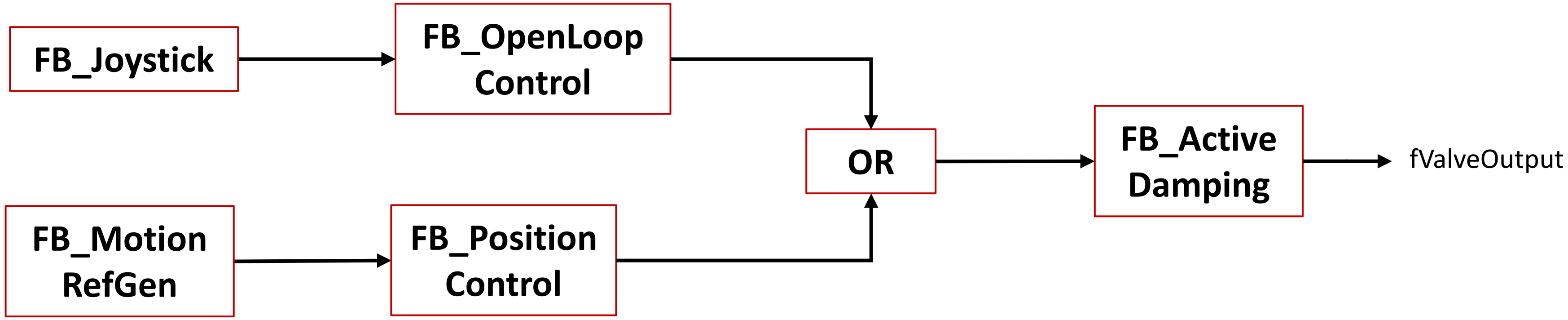
Visualization/PLC HMI

- Create a visualization representing the physical control box and the functions labeled in the figure
  - The Joystick can be programmed as a slider gain with input **-1**...**1** with **0** in zero position
  - Rotary knob gives feedback (**DI**) only when in **AUTO** or **MANUAL**
    - Since the rotary knob in the visualization is either **ON** or **OFF** use two rotary knobs, one for **ON/OFF** signal, and one for **AUTO/MANUAL**
  - The press buttons (**DI**) for START and STOP have light (**DO**). **RUNNING** status → GEEN light and **FAULT** status → RED light.
    - Since the press buttons in the visualizer don't have variable for light use separate lamps
    - In Auto mode, the start button must be programmed to be pressed in all the time to generate motion ref (i.e. clock input). If released clock (motion ref) stops.

UiA

Motion controller

Overview



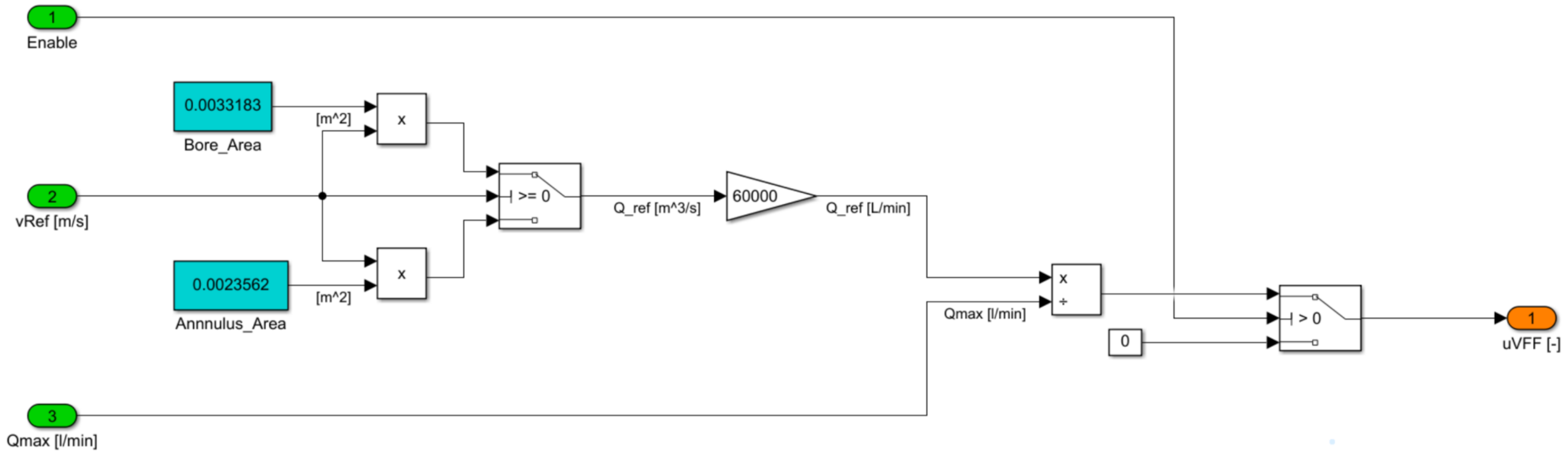
See Simulink model for details: Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)

Motion controller

FB_OpenLoop Control

Open-loop velocity control

- Manual mode: Operator-in-the-loop control with joystick input



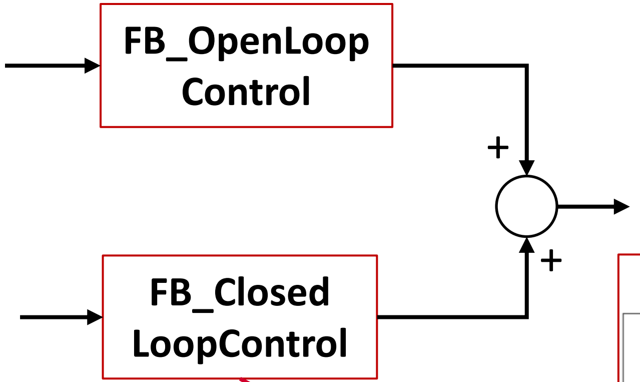
See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

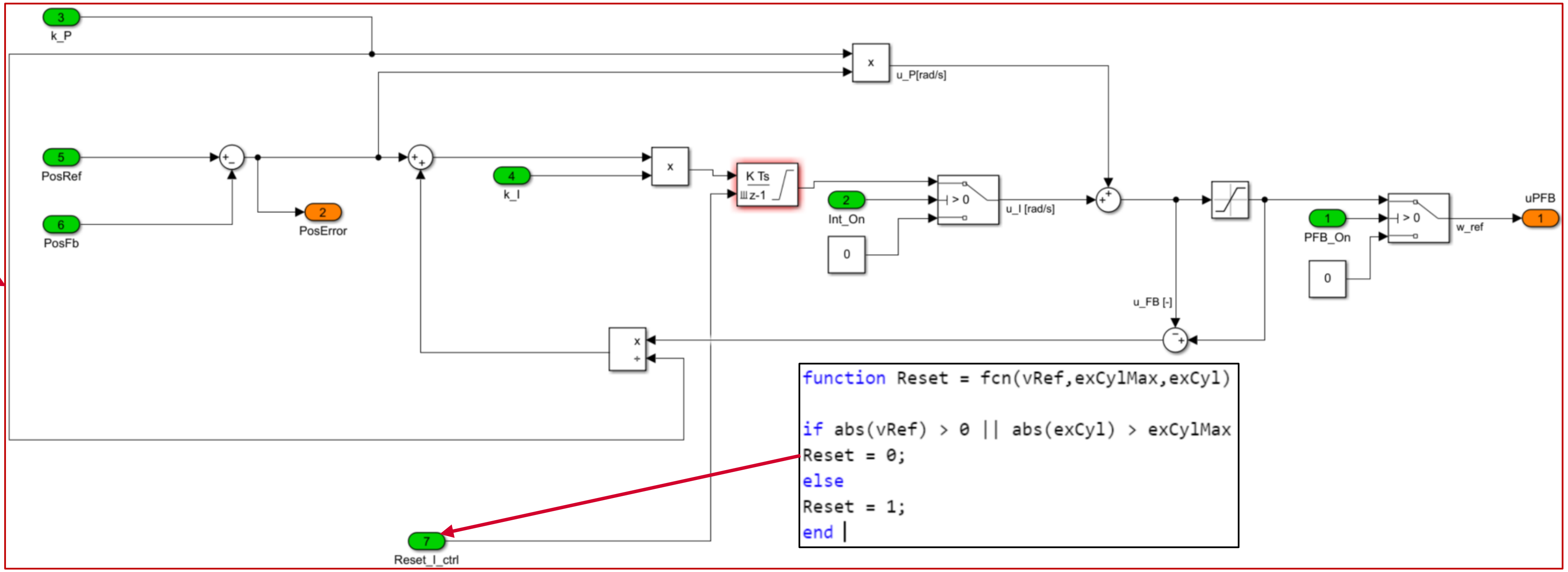
Motion controller

FB_Position Control

Position control

- Auto mode: Automatic motion reference generation and position control





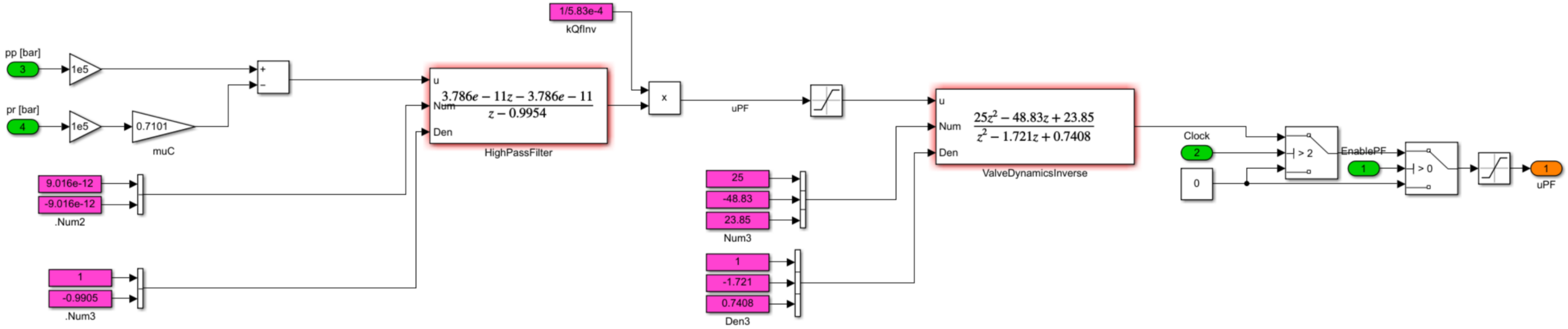
See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

Motion controller

FB_Active Damping

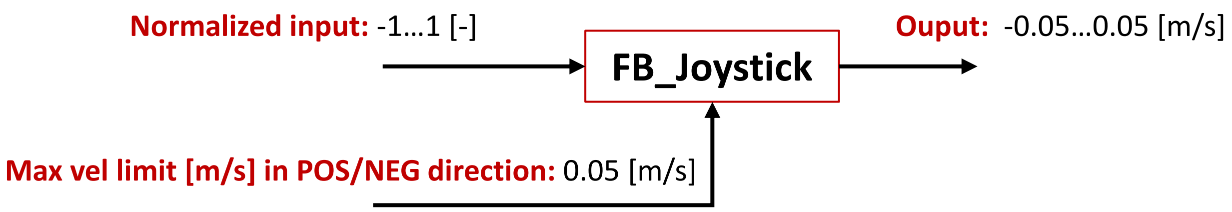
Active damping (Optional in LAB #10)

- ON/OFF function – Pressure feedback



Control input

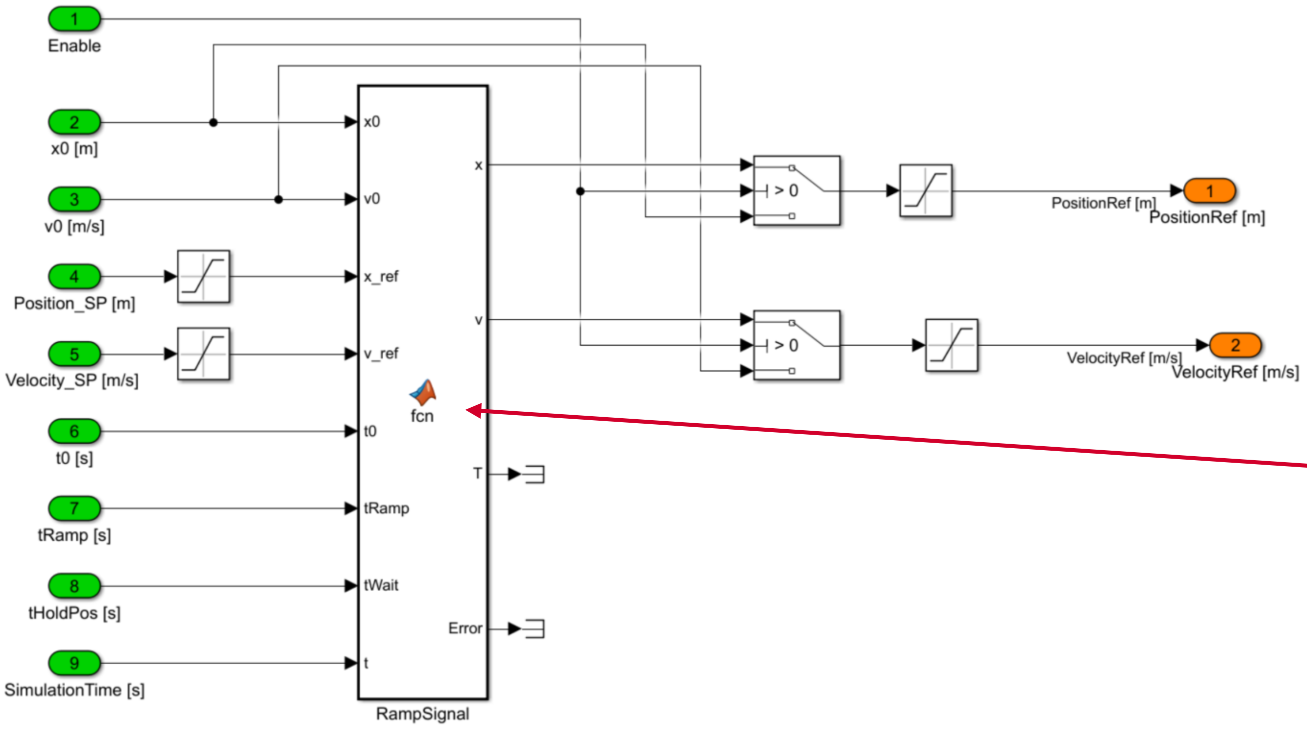
Joystick



See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

Control input

Motion Reference Generator (Ramp function)




```
function [x,v,T, Error] = fcn(x0,v0, x_ref,v_ref,t0,tRamp,tWait,t)
x_SP = x_ref - x0;
vs=v_ref;
slopeExt=v0-vs;
slopeRetr=-vs-v0;

as = vs/tRamp;
s_acc=(vs^2-v0^2)/as;

tHold=(x_SP-s_acc)/vs;

if tHold < 0
Error = 1;
else
Error = 0;
end

t1=tRamp;
t2=tHold;
t3=tRamp;
t4=tWait;
t5=t1;
t6=t2;
t7=t3;

x1 = x0 + v0*((t0+t1)-t0)-(slopeExt/t1)*((t0+t1)-t0)^2/2;
x2 = x1 + vs*((t0+t1+t2)-(t0+t1));
x4 = x_ref - v0*((t0+t1+t2+t3+t4+t5)-(t0+t1+t2+t3+t4))+(slopeRetr/t5)*((t0+t1+t2+t3+t4+t5)-(t0+t1+t2+t3+t4))^2/2;
x5 = x4-vs*((t0+t1+t2+t3+t4+t5+t6)-(t0+t1+t2+t3+t4+t5));

if Error == 1
x = x0;
v = v0;
elseif t>=0 && t<t0
x = x0;
v = v0;
elseif t>=t0 && t<(t0+t1)
x = x0 + v0*(t-t0)-(slopeExt/t1)*(t-t0)^2/2;
v = v0-(slopeExt/t1)*(t-t0);
elseif t>=(t0+t1) && t<(t0+t1+t2)
x = x1 + vs*(t-(t0+t1));
v = vs;
elseif t>=(t0+t1+t2) && t<(t0+t1+t2+t3)
x = x2+vs*(t-(t0+t1+t2))+(slopeExt/t3)*(t-(t0+t1+t2))^2/2;
v = vs+(slopeExt/t3)*(t-(t0+t1+t2));
elseif t>=(t0+t1+t2+t3) && t<(t0+t1+t2+t3+t4)
x = x_ref;
v = v0;
elseif t>=(t0+t1+t2+t3+t4) && t<(t0+t1+t2+t3+t4+t5)
x = x_ref - v0*(t-(t0+t1+t2+t3+t4))+(slopeRetr/t5)*(t-(t0+t1+t2+t3+t4))^2/2;
v = v0+(slopeRetr/t5)*(t-(t0+t1+t2+t3+t4));
elseif t>=(t0+t1+t2+t3+t4+t5) && t<(t0+t1+t2+t3+t4+t5+t6)
x = x4-vs*(t-(t0+t1+t2+t3+t4+t5));
v = -vs;
elseif t>=(t0+t1+t2+t3+t4+t5+t6) && t<(t0+t1+t2+t3+t4+t5+t6+t7)
x = x5-vs*(t-(t0+t1+t2+t3+t4+t5+t6))-(slopeRetr/t3)*(t-(t0+t1+t2+t3+t4+t5+t6))^2/2;
v = -vs-(slopeRetr/t3)*(t-(t0+t1+t2+t3+t4+t5+t6));
else
x = x0;
v = v0;
end

T = t0+t1+t2+t3+t4+t5+t6+t7;
```

UiA

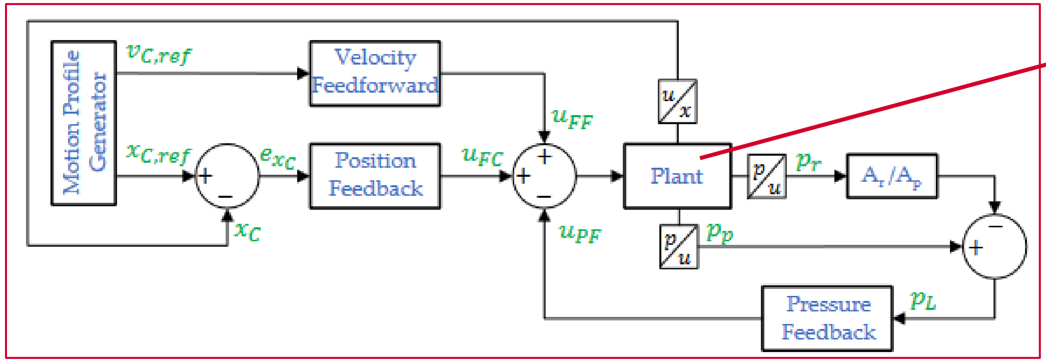
See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

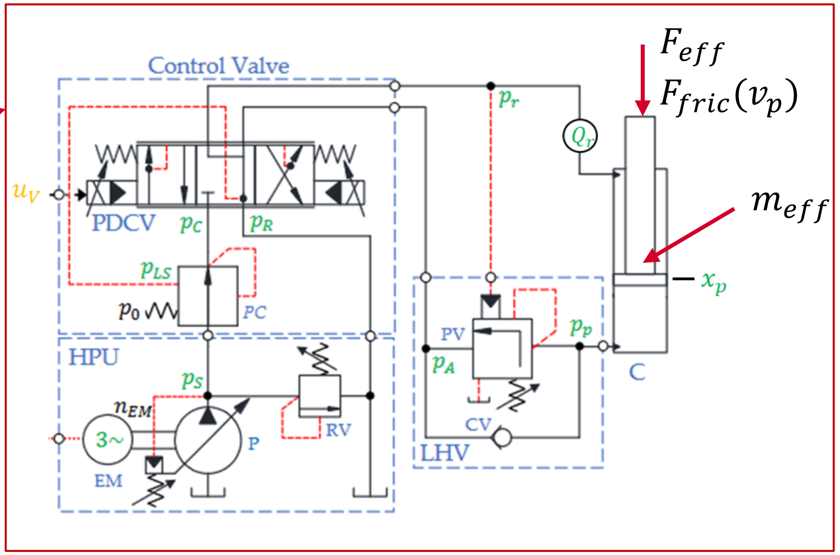
Part II: Software-in-the-Loop testing

1. System overview
2. Simplified hydro-mechanical model
3. Simulink PLC Coder

System overview

Task1 (10ms)		Task2 (0.1ms)
P_Crane ControlSystem	⟷	**P_Crane Simulator**





See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

Simplified hydro-mechanical model

P_Crane Simulator

- **Overview**

FB_HPU → **FB_Control Valve** → **FB_Hydraulic Cylinder** → **FB_Crane**

FB_HPU

Out:
- Ps=180 [bar]
- Pr=1 [bar]

FB_Control Valve

In:
- Ps=180 [bar]
- Pr=1 [bar]
- Uvalve

Out:
- Qa
- Qb

FB_Hydraulic Cylinder

In:
- Qa
- Qb
- xCyl
- vCyl

Out:
- Fhyd

FB_Crane

In:
- Fhyd

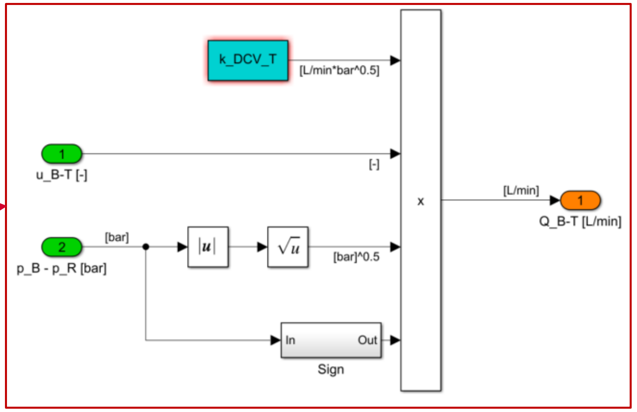
Out:
- xCyl
- vCyl

See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

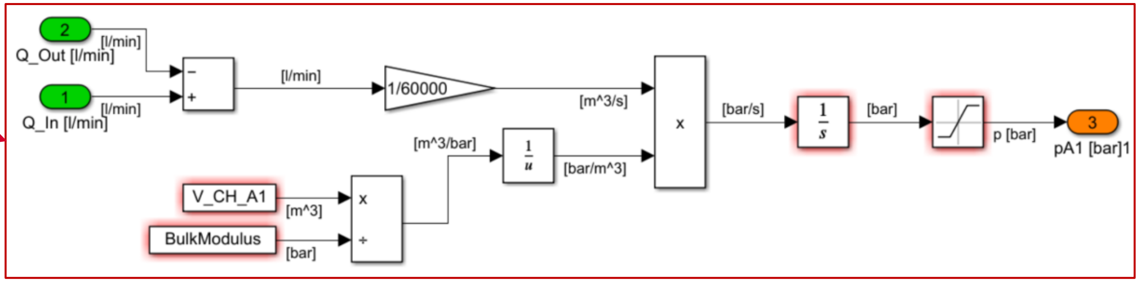
Simplified hydro-mechanical model

- Other **functions / function blocks** you have to make and reuse withing the main FBs

FB_ValveFlow



FB_PressureBuildup

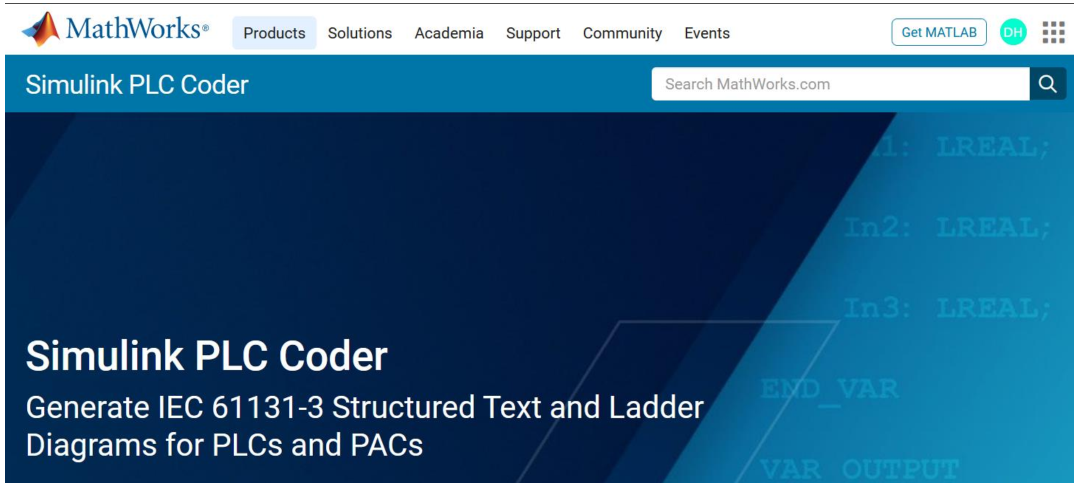


See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

Simulink PLC Coder

Procedure

- Use a separated Simulink project specific settings
- Use supported Simulink blocks (i.e. discrete integrators etc.)
- When building new code, delete previous generated folder
- The blocks need to be in a subsystem

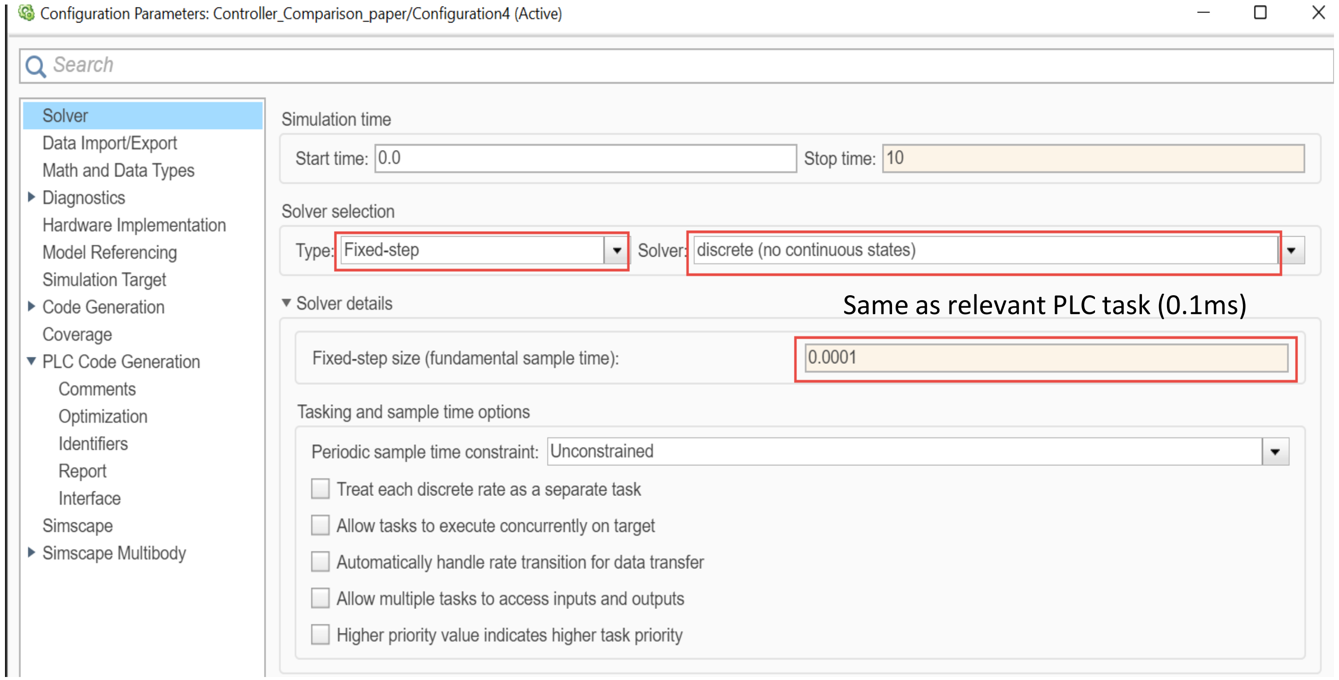


https://se.mathworks.com/products/simulink-plc-coder.html

Simulink PLC Coder

Simulink settings

- Solver

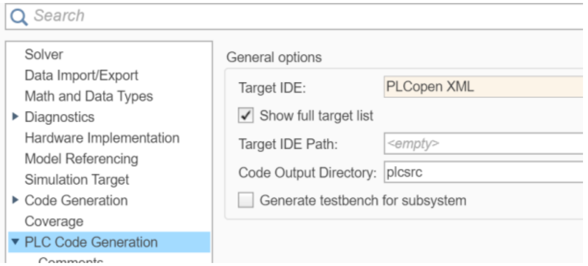


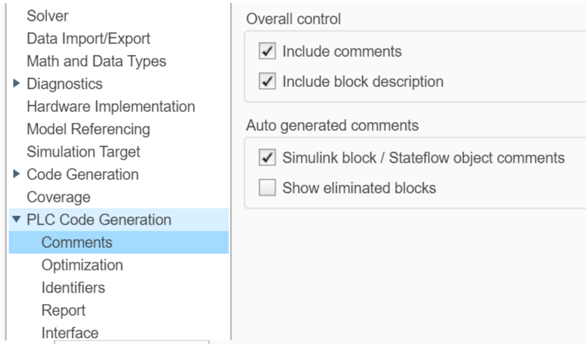
See Simulink model for details: [Simulink Model MAS418 LAB #7 - OneDrive (sharepoint.com)]

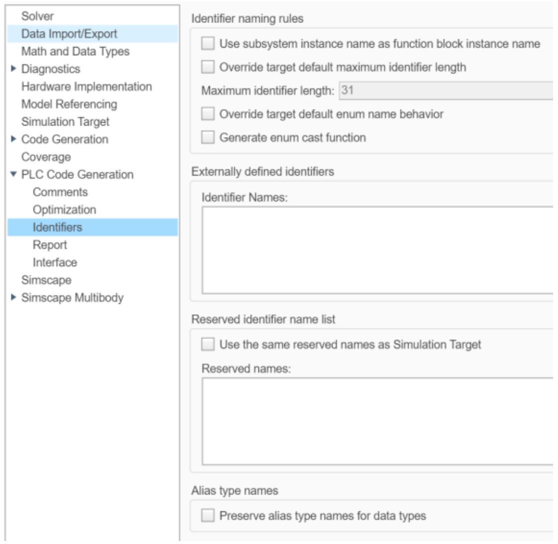
Simulink PLC Coder

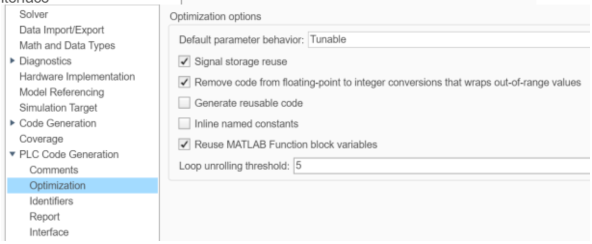
Simulink settings

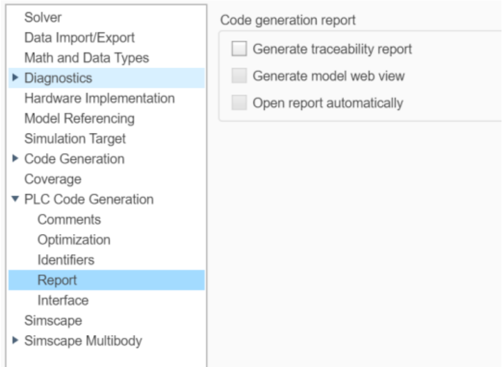
- Solver
- PLC coder options
  - Play with them

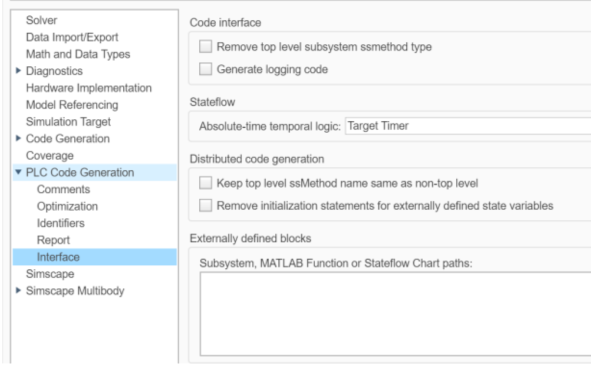


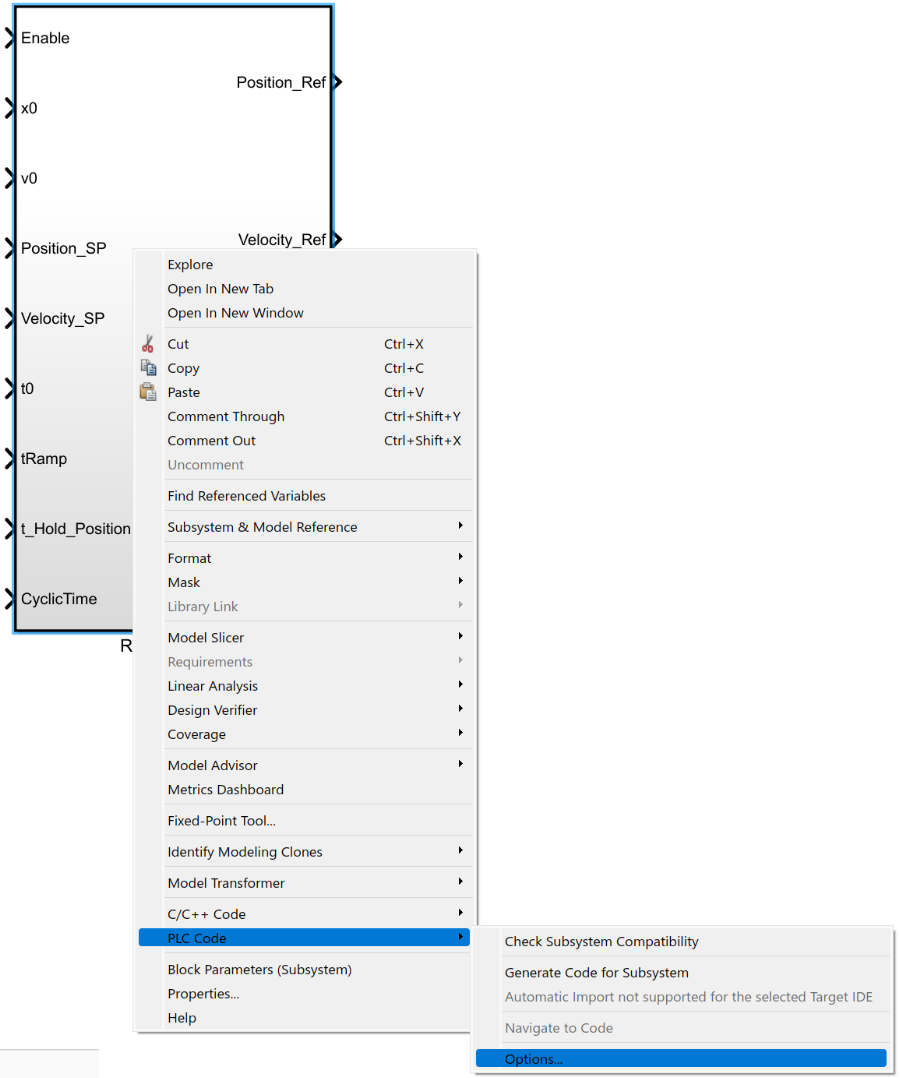










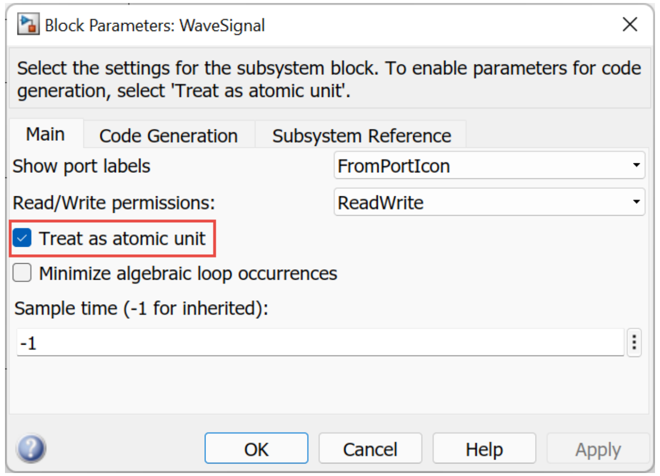


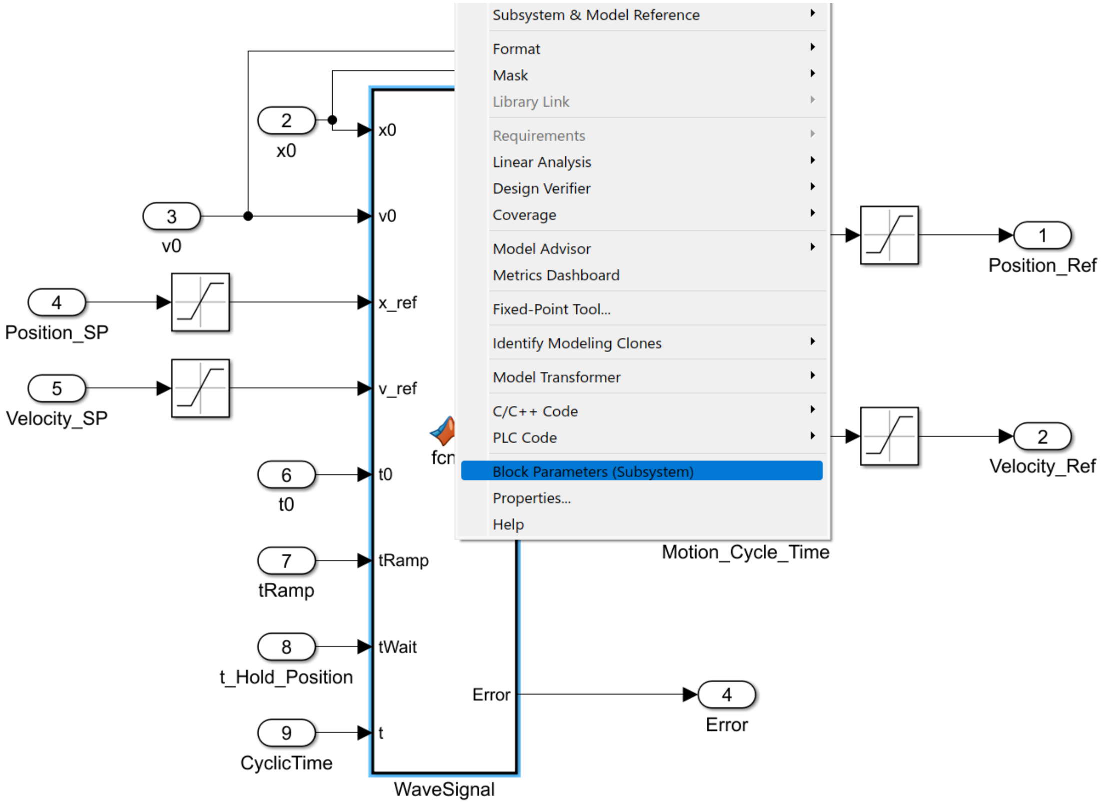
UiA

Simulink PLC Coder

Simulink settings

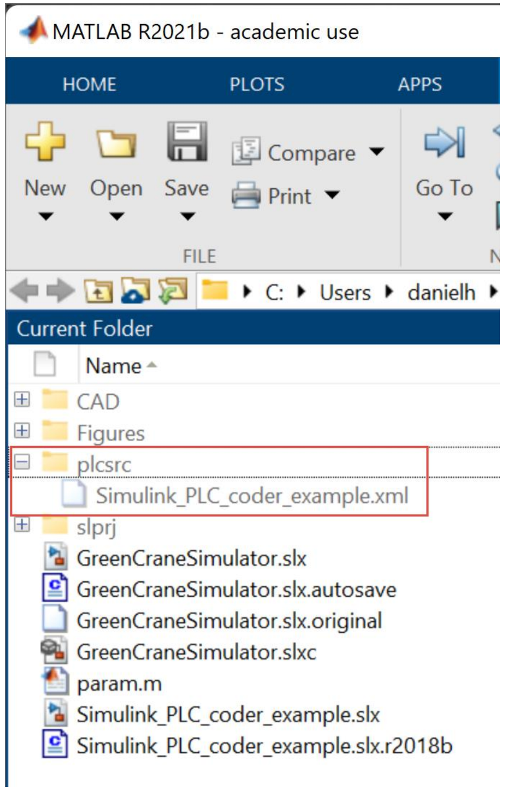
- Solver
- PLC coder options
  - Play with them
- MATLAB function settings

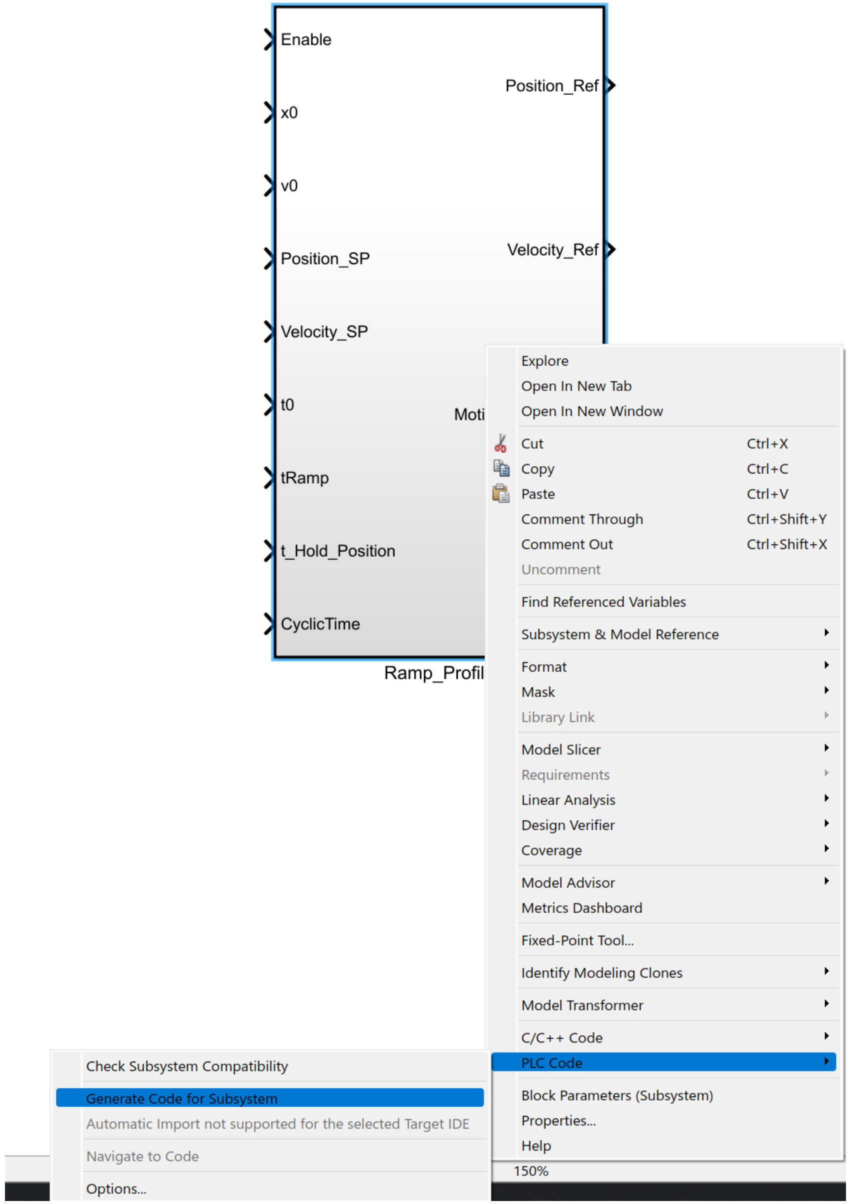




Simulink PLC Coder

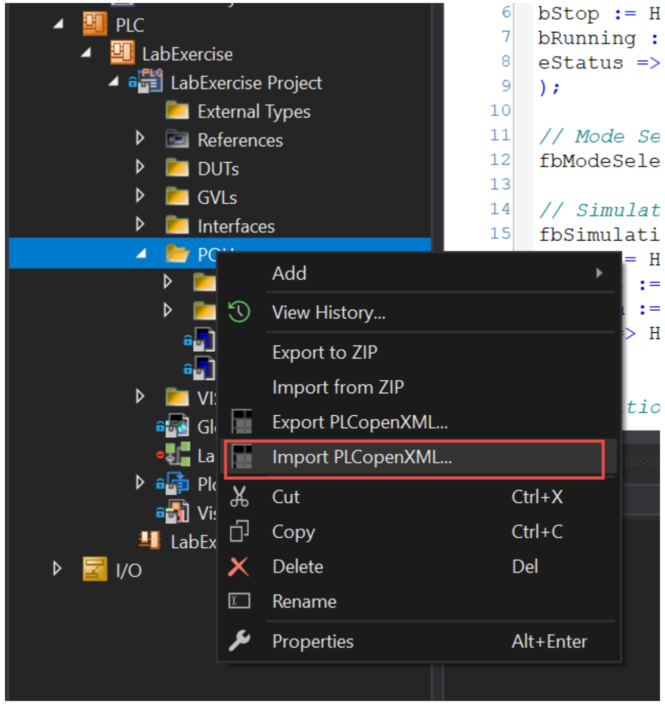
Generate code

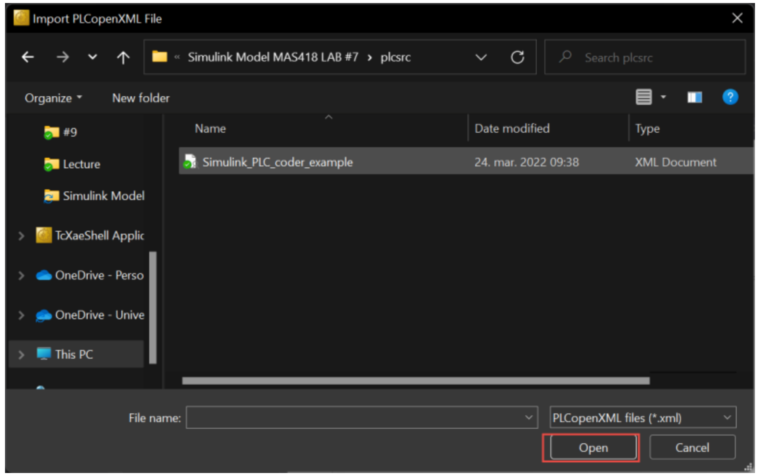


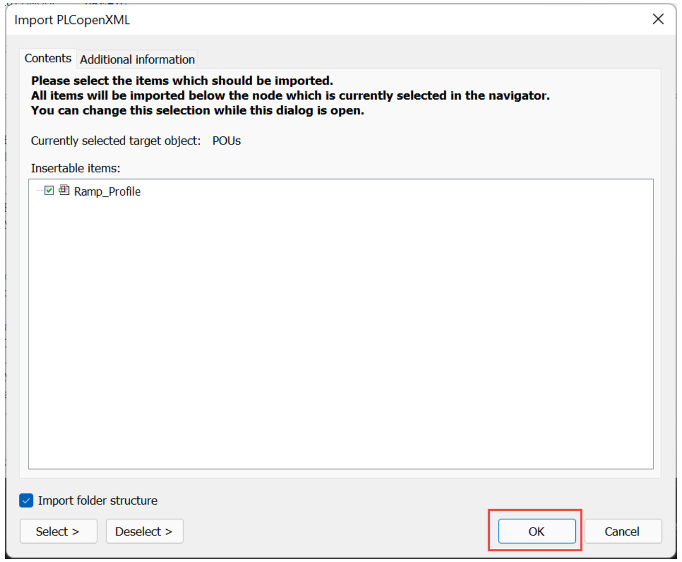


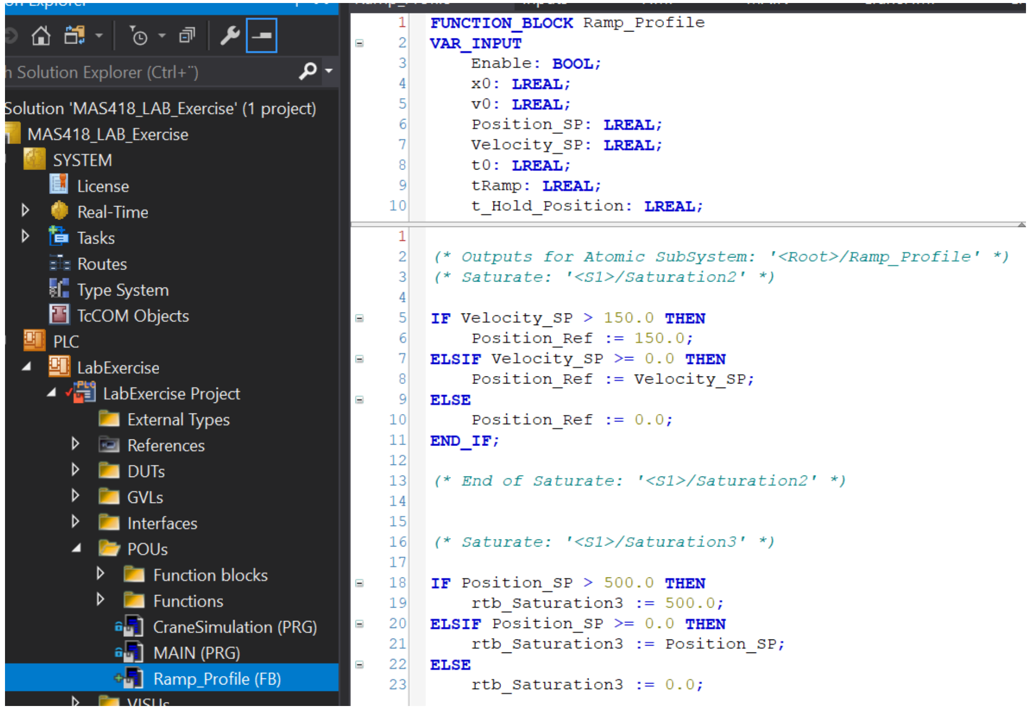
Simulink PLC Coder

Implement code in TwinCAT









UiA

Part III: Demo?

1. How much time left?

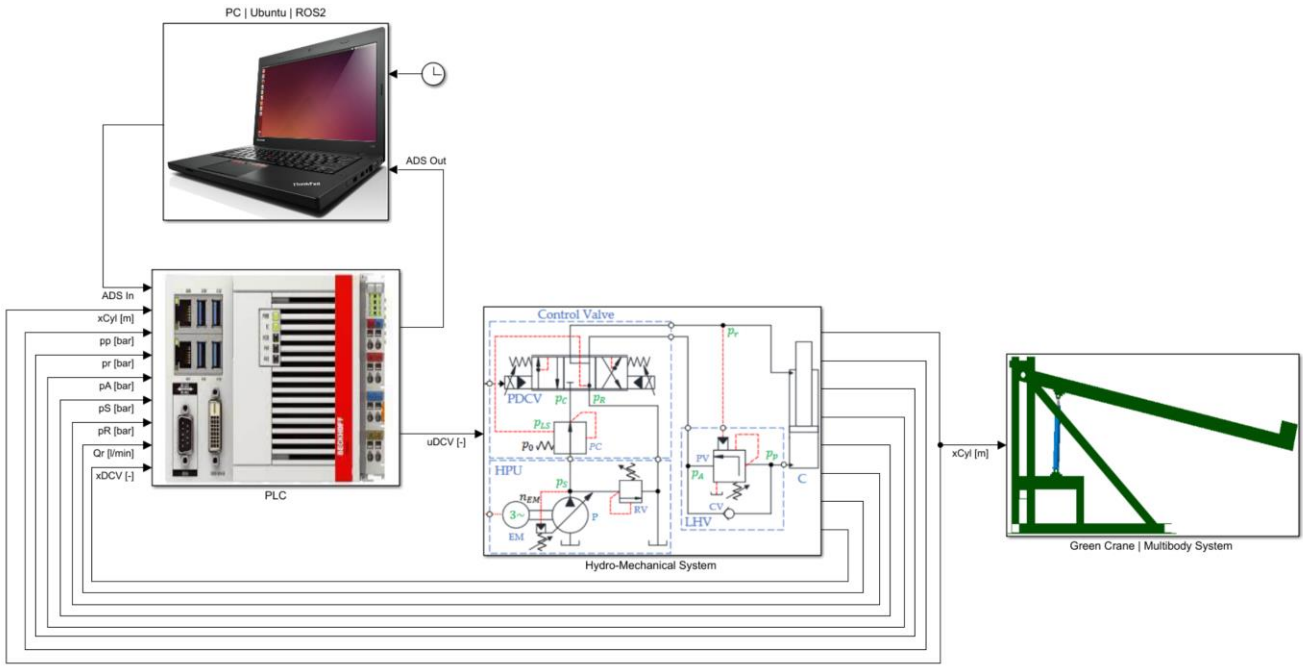
Summary

I. System modeling

- System overview
- Safety functions
- Visualization/PLC HMI
- Motion controller
- Control input

II. Software-in-the-Loop testing

- System overview
- Simplified hydro-mechanical model
- Simulink PLC Coder



Task1 (10ms) | Task2 (0.1ms)

P_Crane ControlSystem ⟷ **P_Crane Simulator**

UiA

Next Lecture

ROS2 interface

Lecture	Topic	Week
#7	**Introduction to part II**	9 – Thursday 3/3
#8	**Procedural oriented PLC programming**	10 – Thursday 10/3
#9	**Object oriented PLC Programming**	11 – Thursday 17/3
#10	**System modeling and SIL testing**	12 – Thursday 24/3
#11	**ROS2 interface**	13 – Thursday 31/3
#12	**Machine interface**	14 – Thursday 7/4