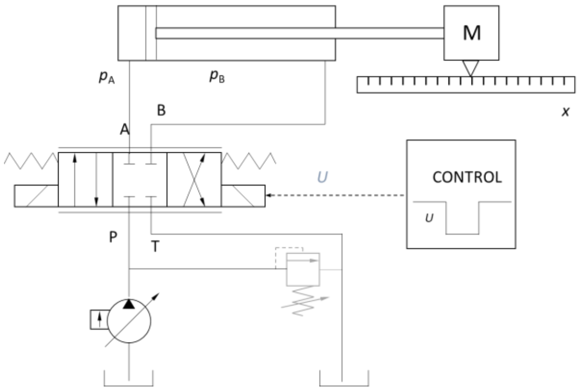
Simscape Fluids exercise

Simscape is an environment for modeling and simulating multidomain physical systems within Matlab/Simulink. More than 10 physical domains, including mechanical, electrical and hydraulic (Simscape Fluids) are covered. In this assignment both the **Simscape Fluids** and **Simulink** domains are mostly utilized.

With Simscape physical component models can be generated based on physical connections by using physical units for both parameters and variables. All unit conversions are handled automatically.



The hydraulic system to be modeled

Cylinder piston's position can be controlled in open and closed loop by using directional proportional control valve. System includes

- Differential cylinder with inertia load (mass)
- Directional proportional control valve
- Pressure compensated pump (constant pressure)
- Control system for position control
- (Presure relief valve)

Hydraulic actuator force needed only for mass acceleration and deceleration since there is no mechanical friction in this system.
In the assignment the cylinder piston is moved
a) into **–** direction and b) into **+** direction
both by using **a) open loop** and **b) closed loop control**.

Open **Matlab.**

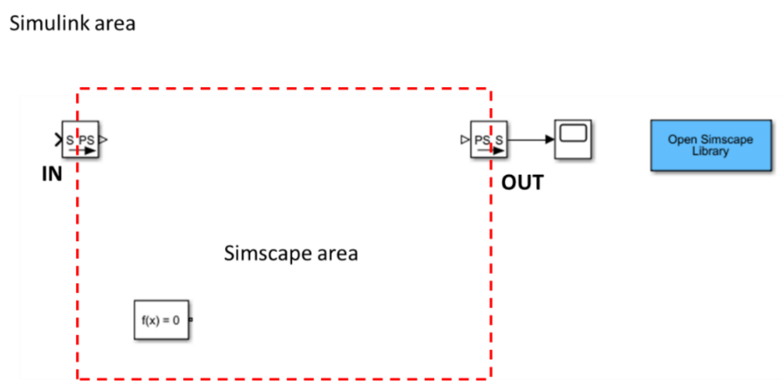
Open the **Simscape** model template for your Simscape Fluids models.

For opening give Matlab command

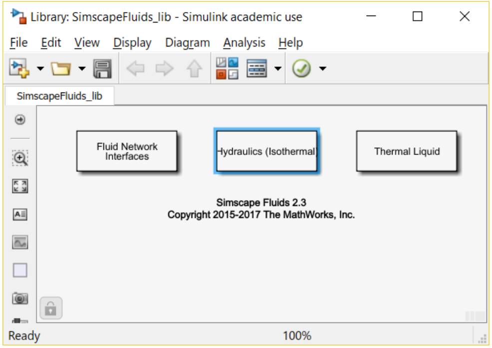
```
ssc_new
```

This will open the

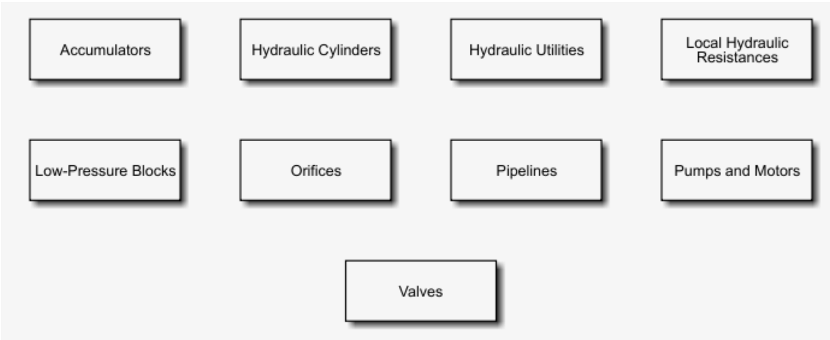
- **Model template** with
  - **Solver Configuration** block to specify the solver parameters for your model
  - **Simulink-PS Converter** and **PS-Simulink Converter** blocks for data transfer between Simscape and Simulink domains
- **Foundation library**



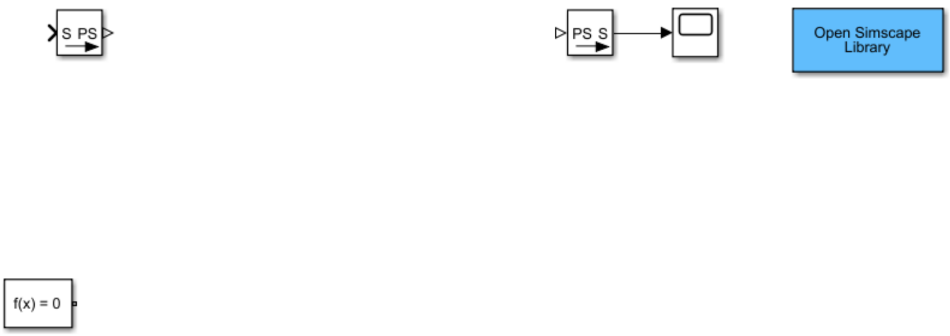
You can enter **Matlab** command `SimscapeFluids_lib` for Simscape Fluids block library.



Library: SimscapeFluids_lib



Library: SimscapeFluids_lib > Hydraulics (Isothermal)



Model canvas

Open loop control – assignment phase 1

Find **Hydraulics (Isothermal)** > **Hydraulic Utilities** library, open it (double clicking) and use the mouse to drag a **Hydraulic Fluid** block to your new model canvas.
With this block you can determine the physical properties of the hydraulic fluid.

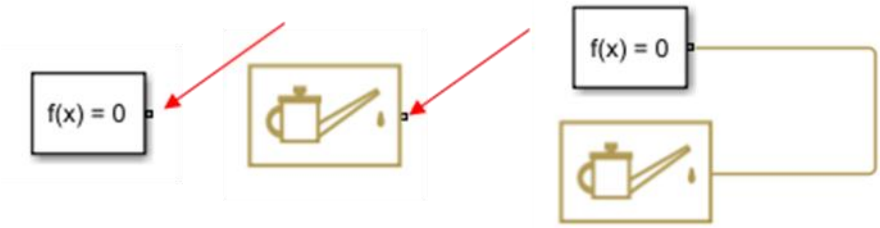
- density,
- viscosity, and
- bulk modulus.

1. Open **Hydraulic Fluid** block by double clicking it.
2. Change the **Hydraulic Fluid** from Skydrol LD-4 to ISO VG 32 (ESSO UNIVIS N 32).
3. Keep the other fluid parameters the same.

Skydrol is fire-resistant aviation hydraulic fluid: http://skydrol-ld4.com/technical_bulletin_skydrol_4.pdf.

- Connect the **Solver Configuration** (f(x)=0) and **Hydraulic Fluid** blocks.

  - With left mouse button draw a wire between blocks' terminals.
  - **OR**
  - Click on the first block with the left button of your mouse
  - Press **Ctrl** button of your keyboard
  - Click on the second block
  - Connection (Wire) will be formed automatically.

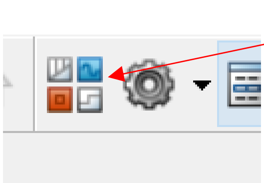


Solver Configuration Hydraulic Fluid

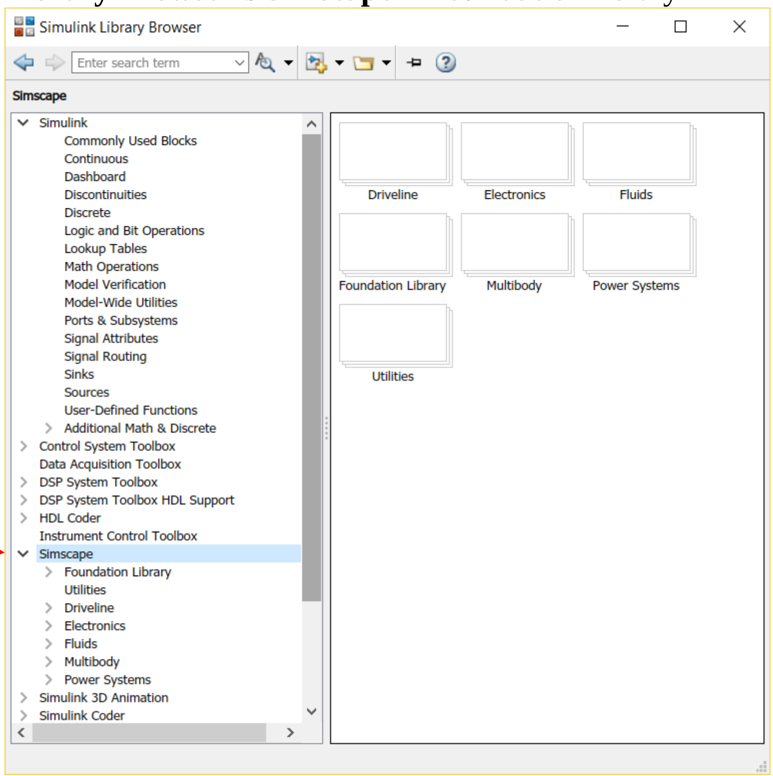
The Solver Configuration block defines solver settings for your model.

Attention! You can rotate blocks by using Crtl-R keyboard command.

In your canvas window click **Library Browser** button to open **Simulink Library Browser**.



From the **Simulink Library Browser's Simscape** > **Foundation** library



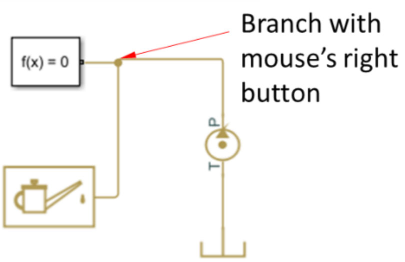
Pick the next two blocks and drag them to the model canvas.

Block	From Sublibrary
Hydraulic Constant Pressure Source	**Foundation Library** > **Hydraulic** > **Hydraulic Sources**
Hydraulic Reference	**Foundation Library** > **Hydraulic** > **Hydraulic Elements**

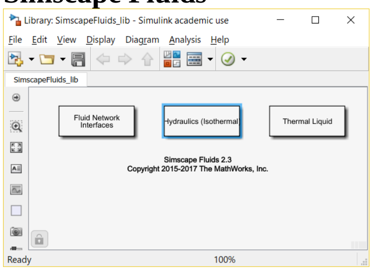
Hydraulic Constant Pressure Source and **Hydraulic Reference**

The Hydraulic Reference block represents a connection to atmospheric pressure.

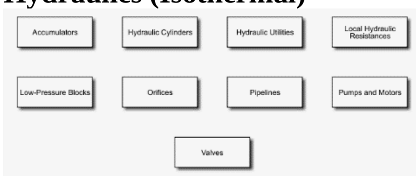
Connect these elements on the canvas. To make the branch use mouse's right button.



Simscape Fluids



Hydraulics (Isothermal)

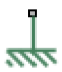


From the **Simscape** > **Fluids** library, pick and drag two blocks to the model canvas.

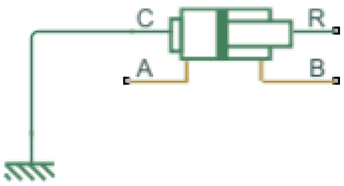
Block	From Sublibrary
Double-Acting Hydraulic Cylinder (Simple)	**Hydraulics (Isothermal)** > **Hydraulic Cylinders**
4-Way Directional Valve	**Hydraulics (Isothermal)** > **Valves** > **Directional Valves**

- The block connections represent the physical connections between the actual components. The cylinder connects to the valve, which connects to the pump, which in turn connects to the fluid reservoir.

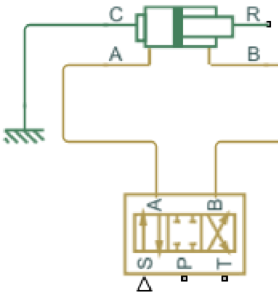
- From the **Simscape** > **Foundation** > **Mechanical** > **Translational** library, add a **Mechanical Translational Reference** block



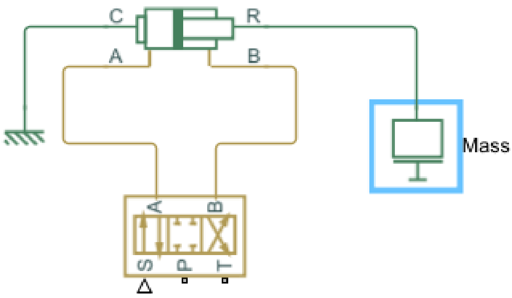
and connect it as shown in the figure below.



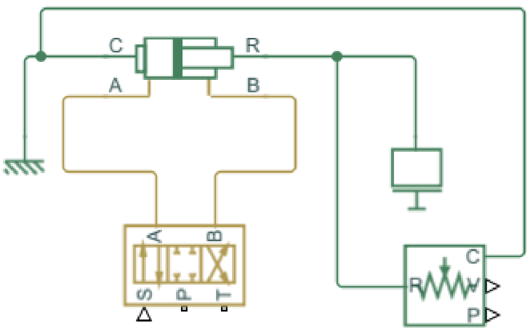
- Connect the **4-way Directional Valve** to the **Double-Acting Hydraulic Cylinder** as follows.



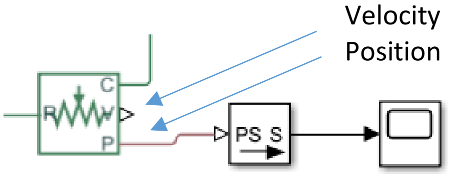
- From the **Simscape** > **Foundation** > **Mechanical** > **Translational Elements** library, add a **Mass** block and connect it as shown in the figure below.



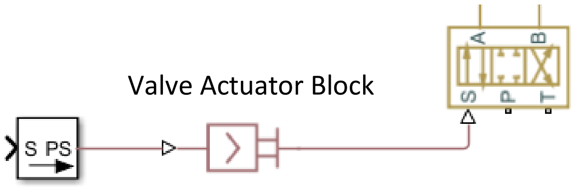
- From the **Simscape** > **Foundation** > **Mechanical** > **Mechanical Sensors** library bring an **Ideal Translational Motion Sensor** block and connect it as in the figure below (both **C** and **R** terminals).



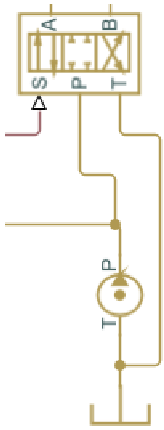
- Connect the **Ideal Translational Motion Sensor** block to **PS-Simulink Converter** and **Scope** as follows.



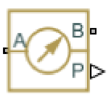
- From the **Hydraulics (Isothermal)** > **Valves** > **Valve Actuators** library bring a **Valve Actuator** block and connect it to **Simulink-PS Converter** and **4-Way Directional Valve** as in the figure below.



- Connect the **4-Way Directional Valve** to **Hydraulic Constant Pressure Source block** and to **the Hydraulic Reference block** as in the figure below.



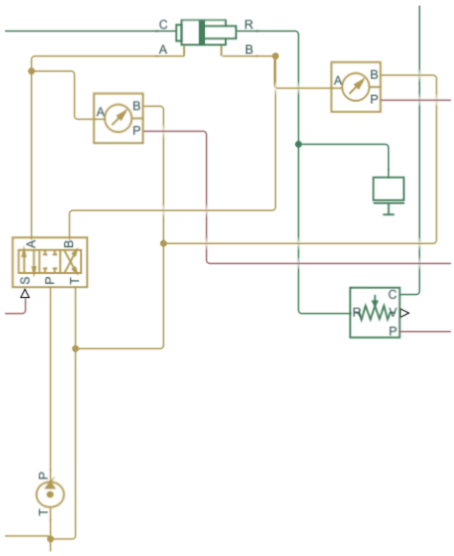
Pressure sensors



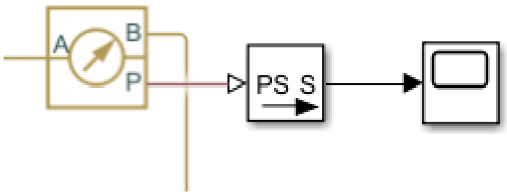
for measuring cylinder chamber pressures.

- From the **Simscape > Foundation** library bring **Hydraulic** > **Hydraulic Sensors** > **Hydraulic Pressure Sensor** block. Make a copy of it (Ctrl-C and Ctrl-V) and connect those

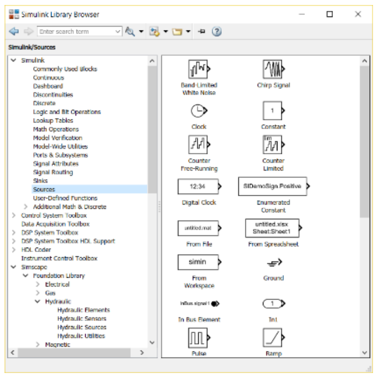
  - between **Hydraulic Cylinder's** A interface and **Hydraulic Reference (B interface)**
  - between **Hydraulic Cylinder's** B interface and **Hydraulic Reference (B interface)**



- Connect both of the **Hydraulic Pressure Sensor(s)** to **Scope(s)** by using a **PS-Simulink Converter** as follows. You can use the existing converter and make a copy of it.



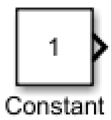
Signal inputs



From the **Simulink Library Browser** > **Sources** bring

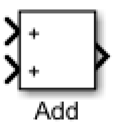
Step block ⇒ clone it (Ctrl-C and Ctrl-V) to get **4 blocks** together.

Constant block

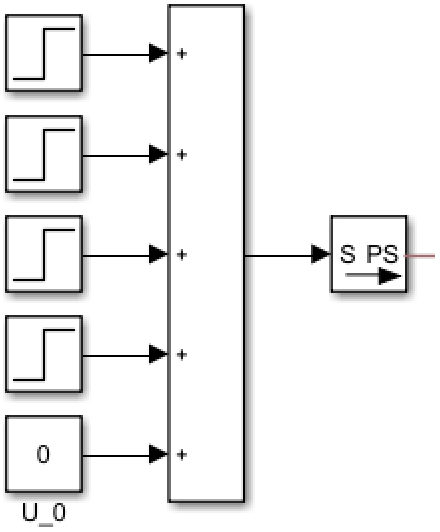


From the **Simulink Library Browser** > **Math Operations** bring

Add block



Connect the blocks with **Simulink-PS Converter** block as follows.



Constant (named U_0) block represents the valve's zero point parameter. At first set the **Constant value** to **0**.

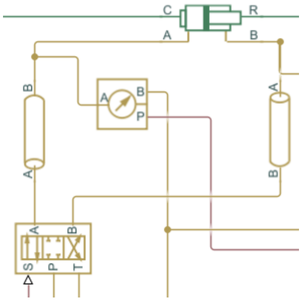
Adjust that parameter later to keep cylinder still during zero input signal.

The **Step** blocks are used for step input commands for the proportional control valve. These parameters values will also be set later.

Pipes

From the Library: **Simscape > Fluids > Hydraulics (Isothermal) > Pipelines** bring **Hydraulic Pipeline** block.

Make a copy of it and connect those two to **Hydraulic Cylinder's** A and B interfaces and corresponding A and B interfaces of **4-Way Directional Valve** as in the figure below.
Attention! Remember that you can rotate blocks by using Crtl-R keyboard command.

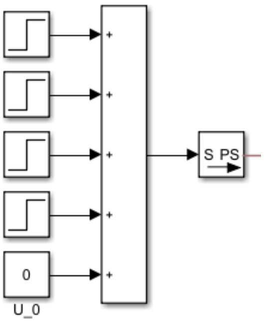


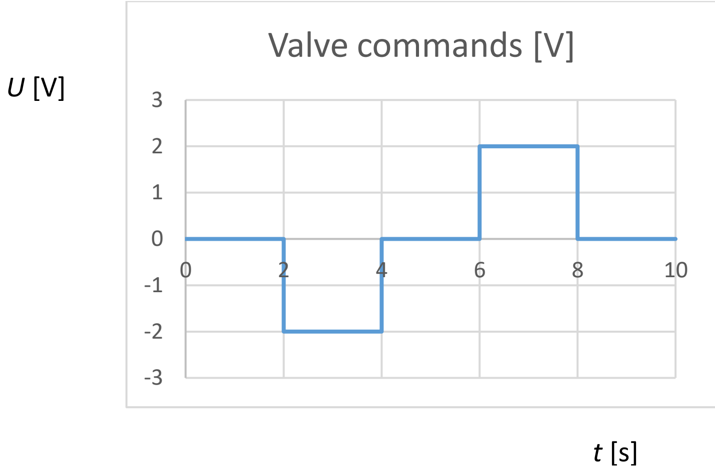
System Parameters

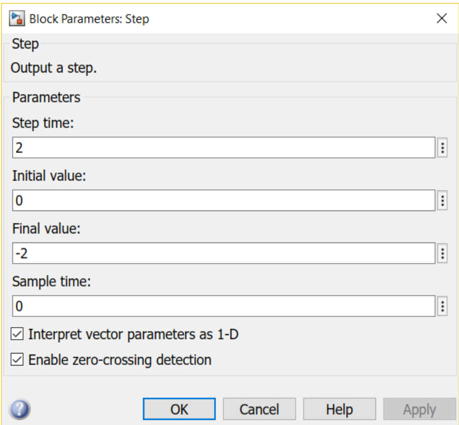
Double click blocks to open

Set system parameters as follows.

Step blocks for timing of the valve commands (usable voltage area: -10 V … 0 … +10 V).

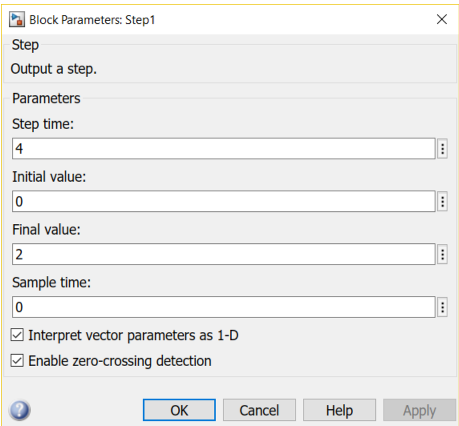






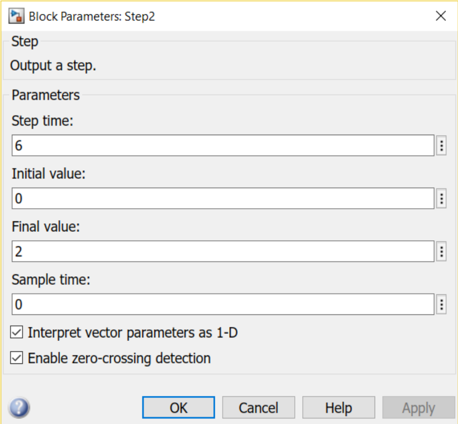
Set **Time** and **Final value**

Step block 1 parameters



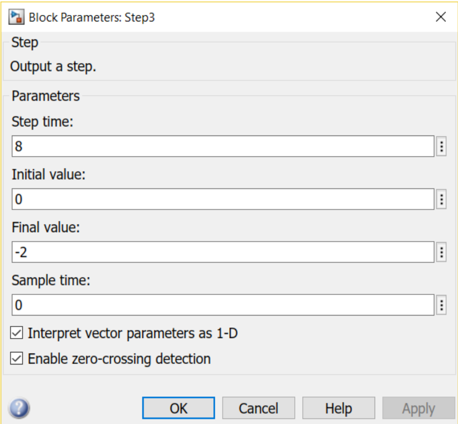
Set **Time** and **Final value**

Step block 2 parameters



Set **Time** and **Final value**

Step block 3 parameters

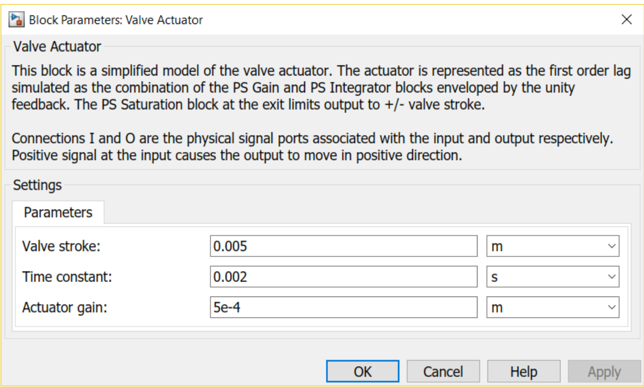


Set **Time** and **Final value**

Step block 4 parameters

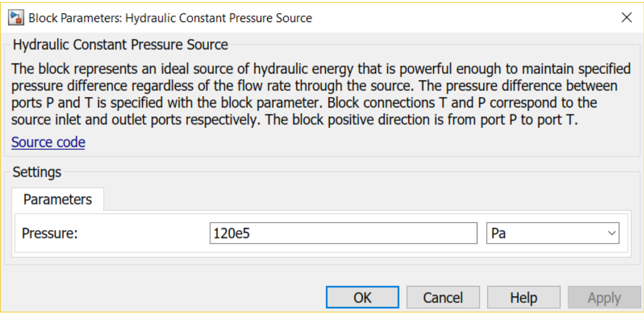
Constant (named U_0) block represents the valve's zero point parameter. Run the simulation (after setting other parameters) and adjust that parameter to keep cylinder still during zero input.

Valve Actuator



- maximum **Valve stroke** 0.005 m (5 mm)
    - by applying 10 V input the Actuator reaches full 5 mm stroke
- therefore the **Actuator gain** is 0.005/10 [m or in theory m/V]
- the value for the **Time constant** can be 0.002 s (2 ms)

Hydraulic Constant Pressure Source (ideal constant pressure pump)



- ideal pump with constant pressure of 120 bar ($120 \cdot 10^5$ [Pa] in Matlab: 120e5 [Pa])

4-Way Directional Valve (proportional valve)

For a narrow (mainly turbulent flow) orifice the flow rate is

$$q_\mathrm{v} = C_\mathrm{q} A \sqrt{\frac{2\Delta p}{\rho}}$$

If we know

- nominal flow rate (q_v),
- nominal pressure drop (Δp),
- fluid density (ρ) and
- flow coefficient (C_q)

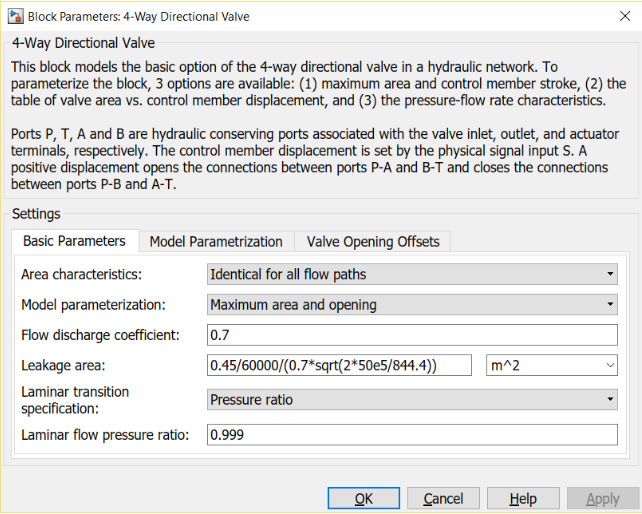
the corresponding flow area can be calculated as follows

$$A = \frac{q_\mathrm{v}}{C_\mathrm{q}\sqrt{\frac{2\Delta p}{\rho}}}$$

For leakage of a certain proportional control valve

q_v	0.45 l/min $\Rightarrow$ 0.45/60000 m^3/s	
Δp	50 bar $\Rightarrow$ $50 \cdot 10^5$ Pa	
ρ	844.4 kg/m^3 (from **Hydraulic fluid** block for ISO VG 32 hydraulic fluid at 60°C)	
C_q	0.7	(for turbulent region)

Open the **4-Way Directional Valve** block dialog box by double clicking.



Set the **Leakage area** parameter as follows.

Leakage area parameter (0.45 l/min @ Δ*p*= 50 bar over each control edge), use Copy+Paste.

0.45/60000/(0.7*sqrt(2*50e5/844.4))

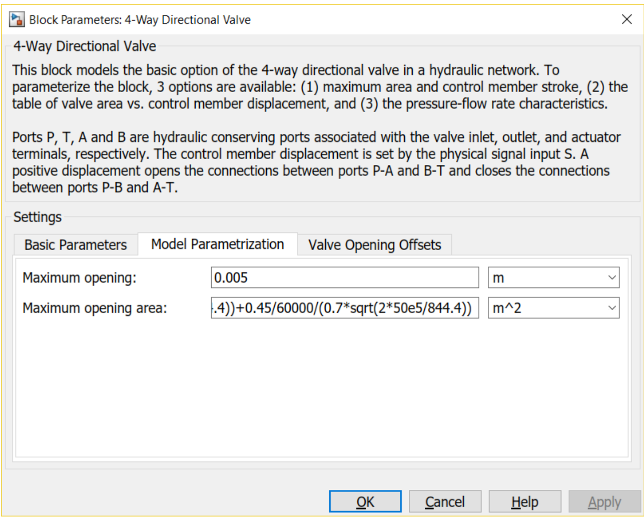
In **Model Parametrization** page

Maximum opening parameter 0.005 m

Maximum opening area parameter, use Copy+Paste

Actual flow area (40 l/min @ 35 bar) + Leakage area +

40/60000/(0.7*sqrt(2*35e5/844.4))+0.45/60000/(0.7*sqrt(2*50e5/844.4))



Simscape's way of modeling orifice area

$$A_{\text{orifice}} = \frac{A_{\text{maximum}}}{h_{\text{maximum}}} h + A_{\text{leakage}}$$

h orifice opening [m]

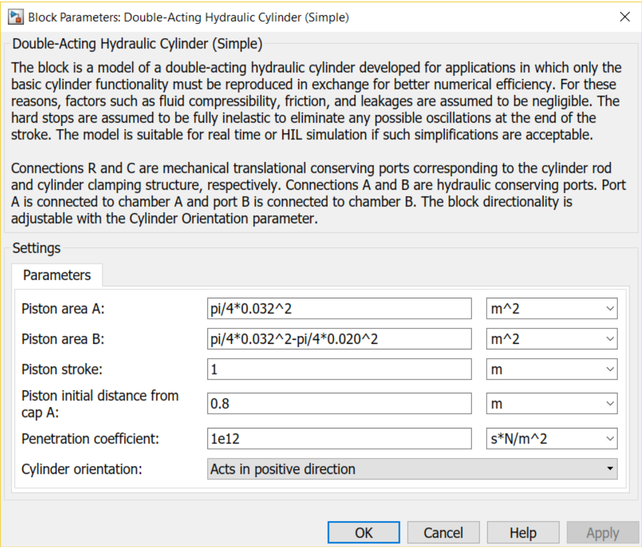
h_{maximum} maximum orifice opening [m]

A_{maximum} maximum orifice area [m^2]

A_{leakage} leakage orifice area [m^2]

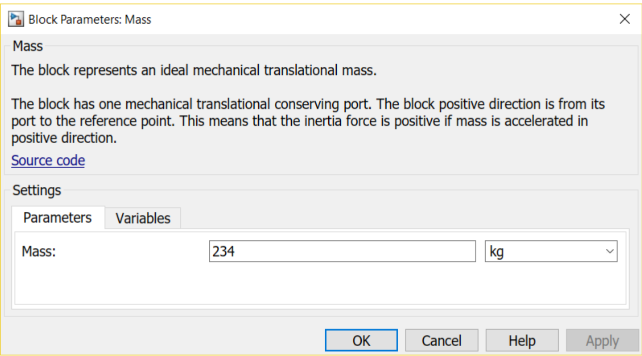
Leakage orifice area is for a "closed orifice".

Double-Acting Hydraulic Cylinder (Simple)



- *D* cylinder diameter 32 mm
- *d* rod diameter 20 mm
- Piston area A: pi/4*0.032^2 ($A_A = \pi/4D^2$)
- Piston area B: pi/4*0.032^2- pi/4*0.020^2 ($A_B = A_A - \pi/4d^2$)
- Piston stroke: 1 m (maximum stroke)
- Piston initial distance from cap A: 0.8 m (initial position of piston)

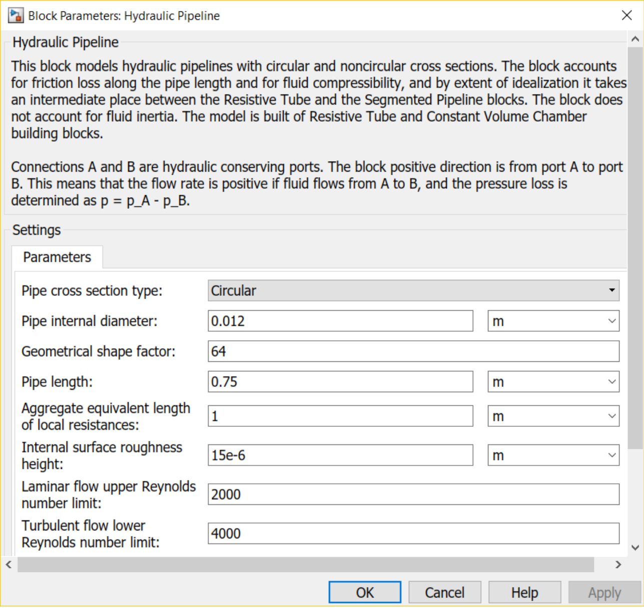
Mass



Pipe parameters

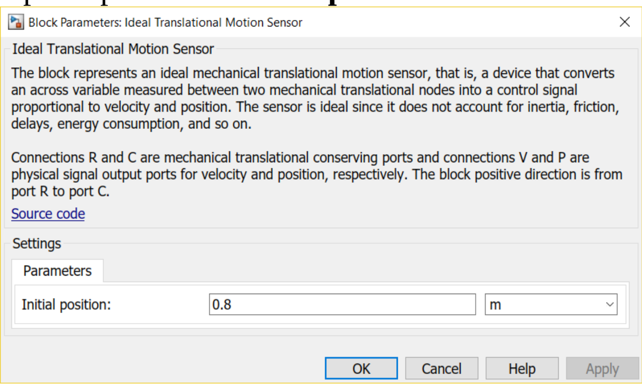
Update parameters

- Pipe internal diameter ⇒ 0.012 m
- Pipe length ⇒ 0.75 m



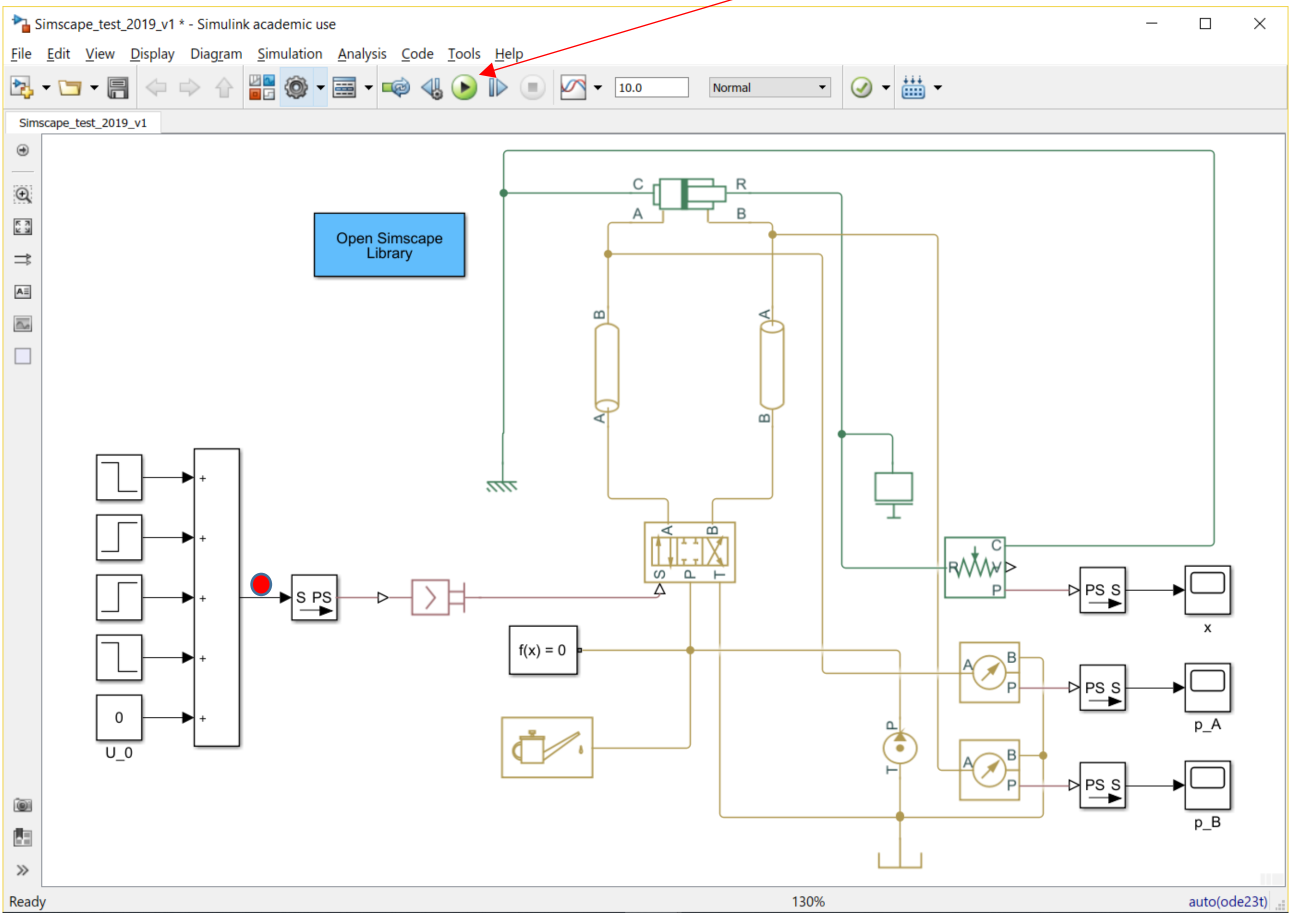
Ideal Translational Motion Sensor

Update parameter **Initial position** ⇒ 0.8 m



Your system should look like this.

Start simulation



Your model should be ready for 10 s simulation.
Use scopes

- x
- p_A
- p_B

for piston position [m], A chamber pressure [Pa] and B chamber pressure [Pa].

You can also test if the input commands are correct by placing a **Scope** between **Add** block and **Simulink-PS Converter.**

Assignment for phase 1 – Open loop control

Make a short document (Word- > pdf)

Documentation Format:

Your name
Assignments

1. Finalize the simulation model
    a. Document **part 1**
        i. Paste a **Figure of the System Model** to your document
        ii. Edit > Copy Current View to Clipboard > Metafile or Bitmap
2. Tune the system with valve's zero point parameter (U_0). Adjust that parameter to keep cylinder still during zero input.
    a. Document **part 2**
        i. Give the proper parameter value for U_0
3. Plot the **Piston Displacement** signal
    a. Document **part 3**
        i. Copy the Scope plot and paste it into your document
        ii. File > Copy to Clipboard (Ctrl-C) OR
        iii. (File > Print to Figure) OR
        iv. Configuration Properties > Logging > Log data to Workspace
            1. Variable name x
            2. Save format: Array
            3. In Matlab workspace
                a. figure
                b. plot(x(:,1),x(:,2));
4. Plot the **Cylinder Pressure A** signal
    a. Document **part 4**
        i. Copy the Scope plot and paste it into your document
        ii. File > Copy to Clipboard (Ctrl-C) OR the options presented above
5. Plot the **Cylinder Pressure B** signal
    a. Document **part 5**
        i. Copy the Scope plot and paste it into your document
        ii. File > Copy to Clipboard (Ctrl-C) OR the options presented above
6. Improvement suggestions to this Tutorial document
    a. Actual errors or misprints (page and location)
    b. Missing information
    c. Actual improvements

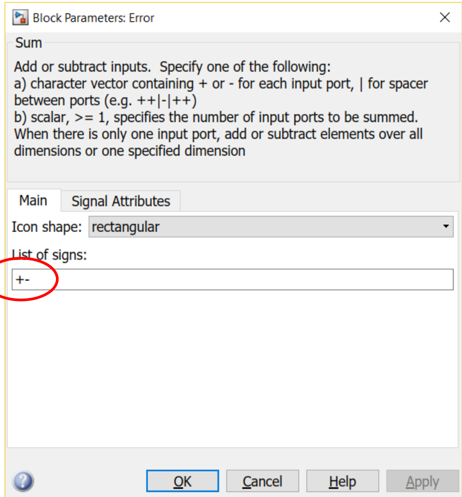
Additional material

Getting started
https://se.mathworks.com/help/physmod/hydro/getting-started-with-simhydraulics.html
Simple actuator model tutorial
https://se.mathworks.com/help/physmod/hydro/ug/creating-a-simple-model.html

Closed loop control – assignment phase 2

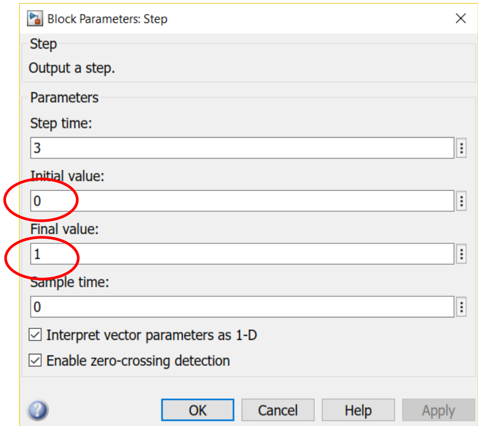
Remove blocks **Step 2** and **U_0**.

Open block **Add** block change the **List of signs** to +-.

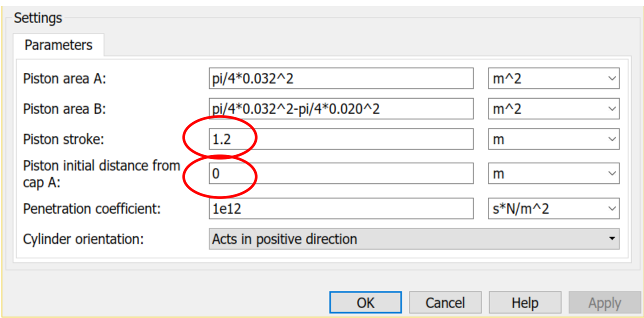


Rename the block to **Error**. Delete **Step** blocks **2, 3, and 4**. Delete **Constant** block (U_0).

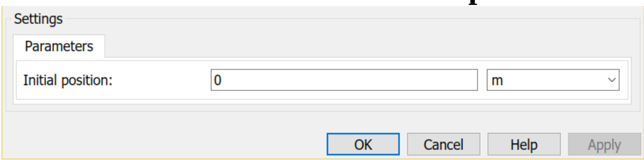
Open **Step 1** block. Rename it as **Step** and update the parameter values as in the figure below.



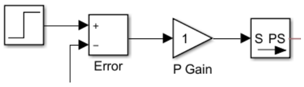
Update **cylinder parameters**
Double click **Double-Acting Hydraulic Cylinder (Simple)** to open it.
Update **Piston stroke** and **Piston distance from cap A** as follows. Click **OK**.



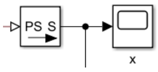
Update also **Ideal Translational Motion Sensor**. Set **Initial position** to **0.**



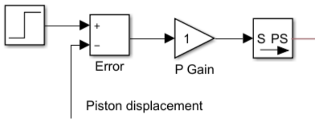
From the **Simulink > Math operations** bring **Gain** block. Place it between **Error** and **Simulink-PS Converter** as in the figure below. Name it as **P gain**. This is the system's P controller (**P**ID).



Branch (mouse right button) and connect **Piston displacement signal** wire from block **PS-Simulink Converter ….**



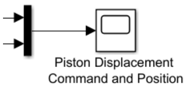
to **Error block's** second interface. The difference between the values tells you how far the actual position is from the target position.



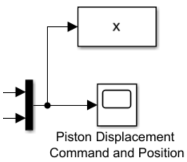
From **Simulink > Signal routing** library bring **Mux** (multiplexer) block. 

Connect **Scope** block to it and name it for example as **Piston Displacement - Command and Position**.
Connect wire from **Step** block to the first interface and **Piston displacement** signal to the second interface.

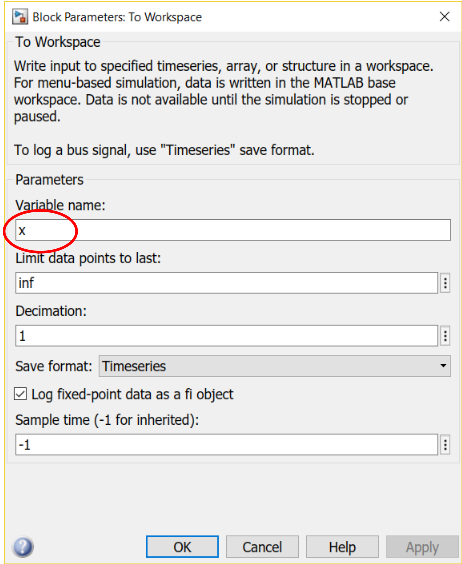
From: Step
From: Piston displacement (x)



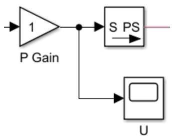
From **Simulink > Sinks** library bring **To Workspace** block.
Connect wire from **Mux** signal(s) to its interface.



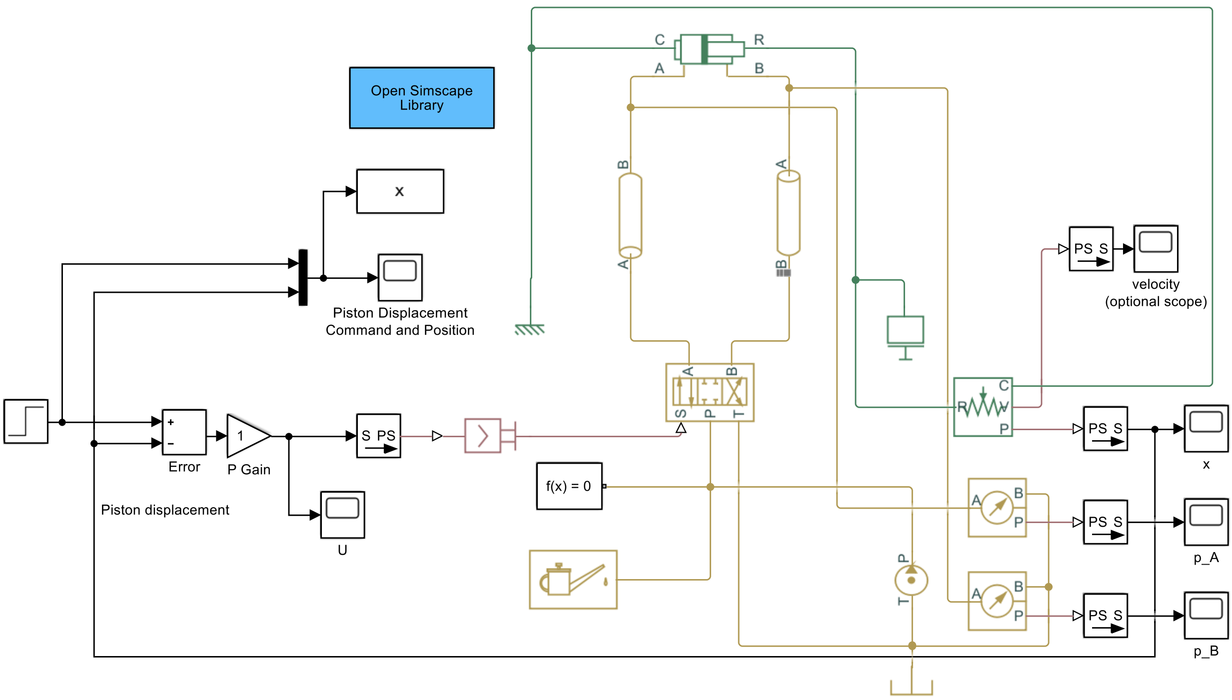
Double click to open **To Workspace** block.
Adjust the parameter(s) as follows > **Variable name** > **x**. Click **OK**.



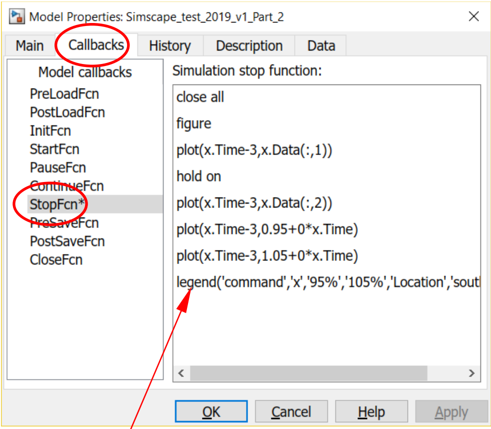
Add also a **Scope** for valve command voltage U between **P Gain** and **Simulink-PS Converter**.



Your system should look (something like) this.



Plotting of variables
File > Model properties > Model properties > Callbacks > StopFcn

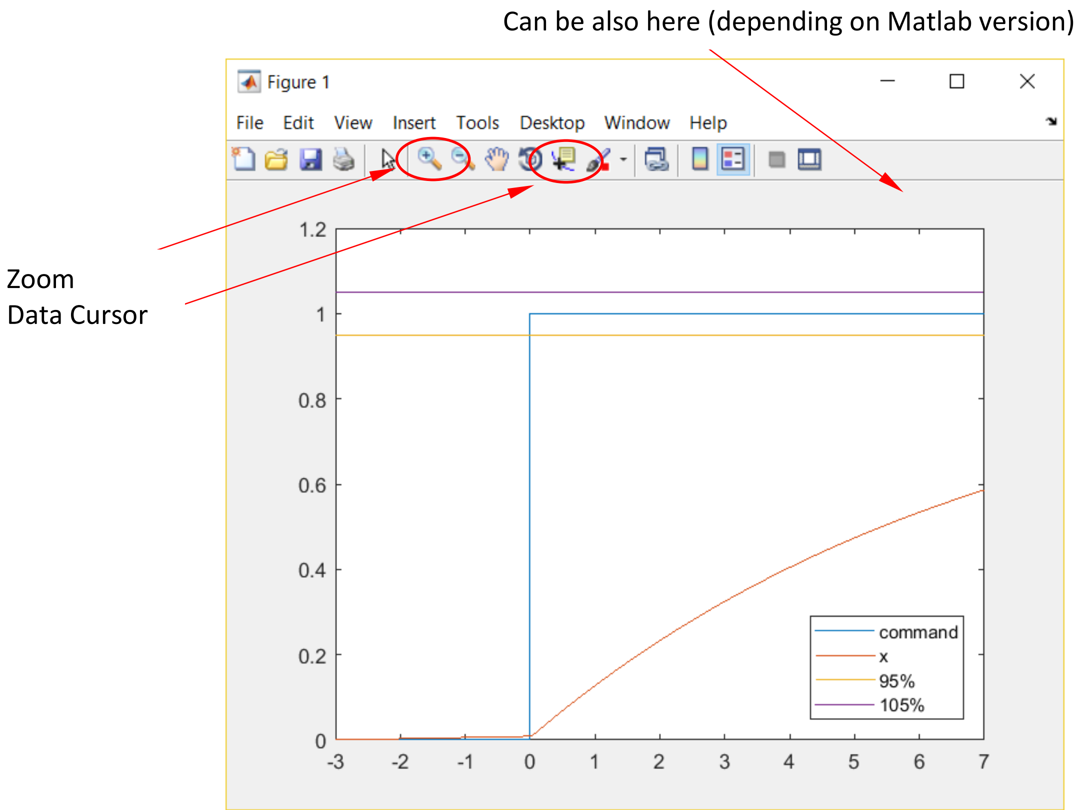


Add the following **code** to **StopFcn**

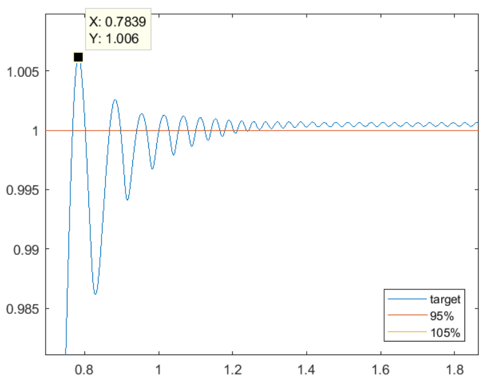
```
close all
figure
plot(x.Time-3,x.Data(:,1))
hold on
plot(x.Time-3,x.Data(:,2))
%plot(x.Time-3,1+0*x.Time)
plot(x.Time-3,0.95+0*x.Time)
plot(x.Time-3,1.05+0*x.Time)
legend('command','x','95%','105%','Location','southeast')
```

Click **OK** to confirm the changes.

Run the model.
You should get also a **Figure** like this.



Use **Zoom** (In or Out) and **Data Cursor** tools for finding detailed information.



Check from the **Figure** or from **Scope Piston Displacement - Command and Position** how well the actuator follows the command.
If the performance is poor increase the **P Gain** value. Raise the value boldly (decades). This is only a simulator!
Notice that the **Piston displacement** signal starts to oscillate if the **P Gain** value is too high.
More specified servo tuning instructions below (**Tuning the P controller according to Ziegler-Nichols**).

Tuning the P controller according to Ziegler-Nichols

- Increase **P Gain** parameter value until the system starts to **oscillate continuously**. Use **zooming**!
- This **minimum** value of **P Gain** (parameter $\mathbf{K_P}$) is so called critical gain $\mathbf{K_{P,\,crit}}$. **Store this value!**
- (To implement controllers as PI or PID you should also estimate the time period of oscillation $\mathbf{T_{crit}}$ corresponding this gain. This can be identified from the response ⇒ time between two successive peaks).
- P controller's gain according to Ziegler-Nichols tuning rules is now simply $\mathbf{0.5 \cdot K_{P,\,crit}}$.

Assignment for phase 2

Continue with your short document (Word -> pdf) for Phase 1

Documentation Format:

Assigments

- Finalize the simulation model
  - Paste a **Figure of the** updated **simulation model** to your document
  - **Edit > Copy Current View to Clipboard > Metafile or Bitmap**
- Test the system with **critical gain** and two different values for the **P gain**
  - $P_{gain,\,0}$= $\mathbf{K_{P,\,crit}}$ (critical gain according to Ziegler-Nichols tuning rule)
  - $P_{gain,\,1}$= $\mathbf{0.5 \cdot K_{P,\,crit}}$ (tuned according to Ziegler-Nichols tuning rule)
  - $P_{gain,\,2}$= $\mathbf{0.25 \cdot K_{P,\,crit}}$ (smaller gain for comparison)

- Plot the **Piston Displacement** signals for
  - $\mathbf{P_{gain,\,0}}$ **(critical)**
    1. overall displacement Figure
    2. zoomed Figure to see the performance near the target position
  - $\mathbf{P_{gain,\,1}}$
    3. overall displacement Figure
    4. zoomed Figure to see the performance near the target position
  - $\mathbf{P_{gain,\,2}}$
    1. overall displacement Figure
    2. zoomed Figure to see the performance near the target position

Analyze the plots and add information to these tables. Check the following page for **Performance** analysis.

P controller parameters

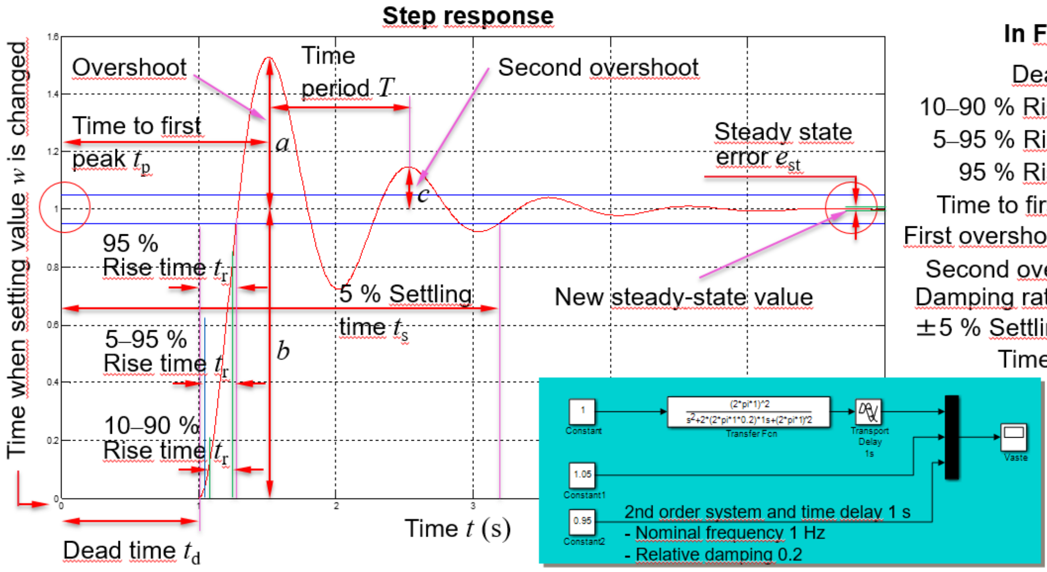
$\mathbf{K_{P,\,crit}}$		V/m
$\mathbf{0.5 \cdot K_{P,\,crit}}$		V/m
$\mathbf{0.25 \cdot K_{P,\,crit}}$		V/m

Performance of P control for $\mathbf{P_{gain,\,1}}$ parameter value ($\mathbf{0.5 \cdot K_{P,\,crit}}$)

Overshoot		%
Rise time 95%		s
Settling time 5%		s
Steady state error		m

Performance of P control $\mathbf{P_{gain,\,2}}$ for parameter value ($\mathbf{0.25 \cdot K_{P,\,crit}}$)

Overshoot		%
Rise time 95%		s
Settling time 5%		s
Steady state error		m



In Figure's case:
Dead time, ca. 1.0 s
10–90 % Rise time, ca. 0.19 s
5–95 % Rise time, ca. 0.23 s
95 % Rise time, ca. 0.28 s
Time to first peak, ca. 1.45 s
First overshoot (a/b), ca. 53 %
Second overshoot, ca. 15 %
Damping ratio (c/a), ca. 0.28
±5 % Settling time, ca. 3.2 s
Time period, ca. 1.0 s

Overshoot
The ratio of difference between output y's first maximum and its new steady-state value to its new steady-state value (= a/b in Figure above). Sometimes this characteristic is marked with M_p, maximum percentual overshoot.

Rise time (95%)
Time it takes for the response to rise from zero to 95% of the steady-state response.

Damping ratio
The ratio of difference between output y's first maximum and its new steady-state value to the difference between output y's second maximum and its new steady-state value (= c/a in Figure above).

Settling time t_s
The time that after a stepwise change in system's setting value w is required for the process output y to reach and remain inside a band whose width is equal to ±5 % of the total change in y (sometimes also other bandwidths are used, e.g., ±1 %, ±2 %).

Time period T
The time between output y's two successive peaks (e.g., first and second maximum) or valleys.

Oscillation frequency f
The frequency that the system oscillated with (= $1/T$).

Steady state error e_{st}
The constant deviation between system's setting value w and actual output value y.