



# ELEC 341: Systems and Control

## Lecture 21

### Frequency response shaping with Matlab (Simulink simulation)

# Course roadmap

## Modeling

- ✓ Laplace transform
- ✓ Transfer function
- Models for systems
  - ✓ • Electrical
  - ✓ • Electromechanical
  - ✓ • Mechanical
- ✓ Linearization, delay

## Analysis

- ✓ Stability
  - ✓ • Routh-Hurwitz
  - ✓ • Nyquist
- ⇨ ✓ Time response
  - ✓ • Transient
  - ✓ • Steady state
- ✓ Frequency response
  - ✓ • Bode plot

## Design

- ✓ Design specs
- ✓ Root locus
- ⇨ ✓ Frequency domain
- ✓ PID & Lead-lag
- ✓ Design examples



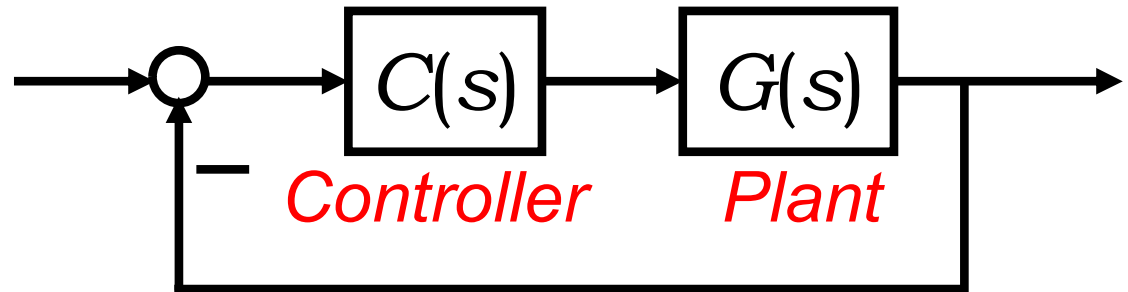
*Matlab simulations*



## Example 1 (SISO Design Tool in Matlab)

- Consider a system:

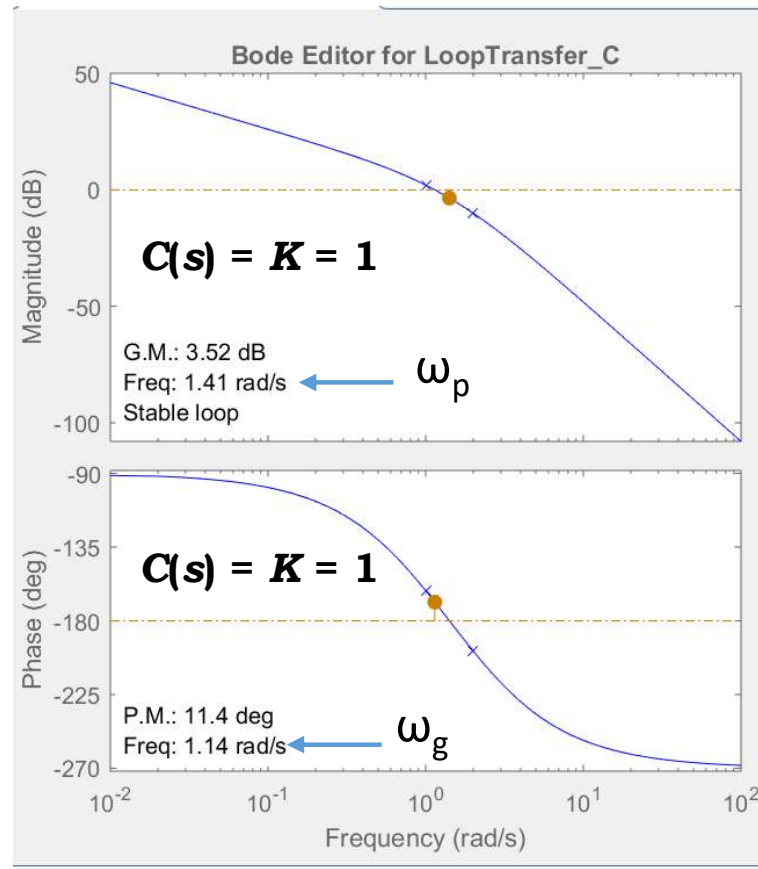
$$G(s) = \frac{4}{s(s+1)(s+2)}$$



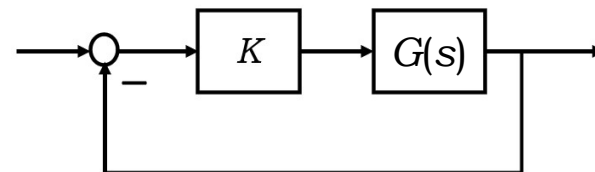
- Design specs:
  - Closed-loop system is stable
  - PM at least 50 deg
  - 2% Settling time < 4 s
  - Steady-state error
    - For unit step input:  $e_{ss} = 0$

# Example 1 (cont'd)

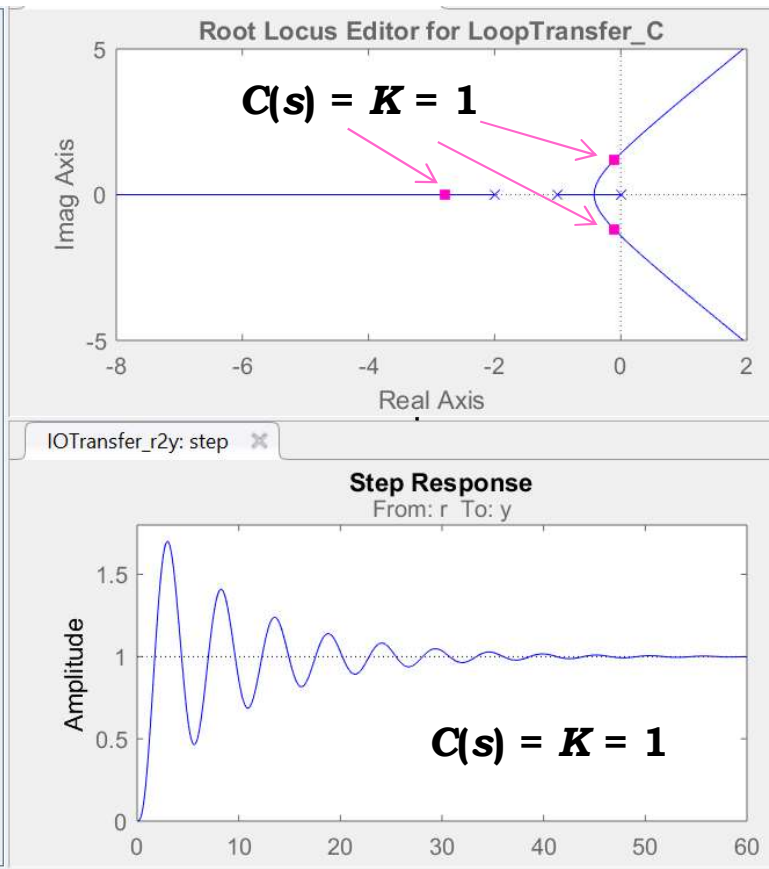
## OL Bode plot



$$G(s) = \frac{4}{s(s+1)(s+2)}$$



## Root locus

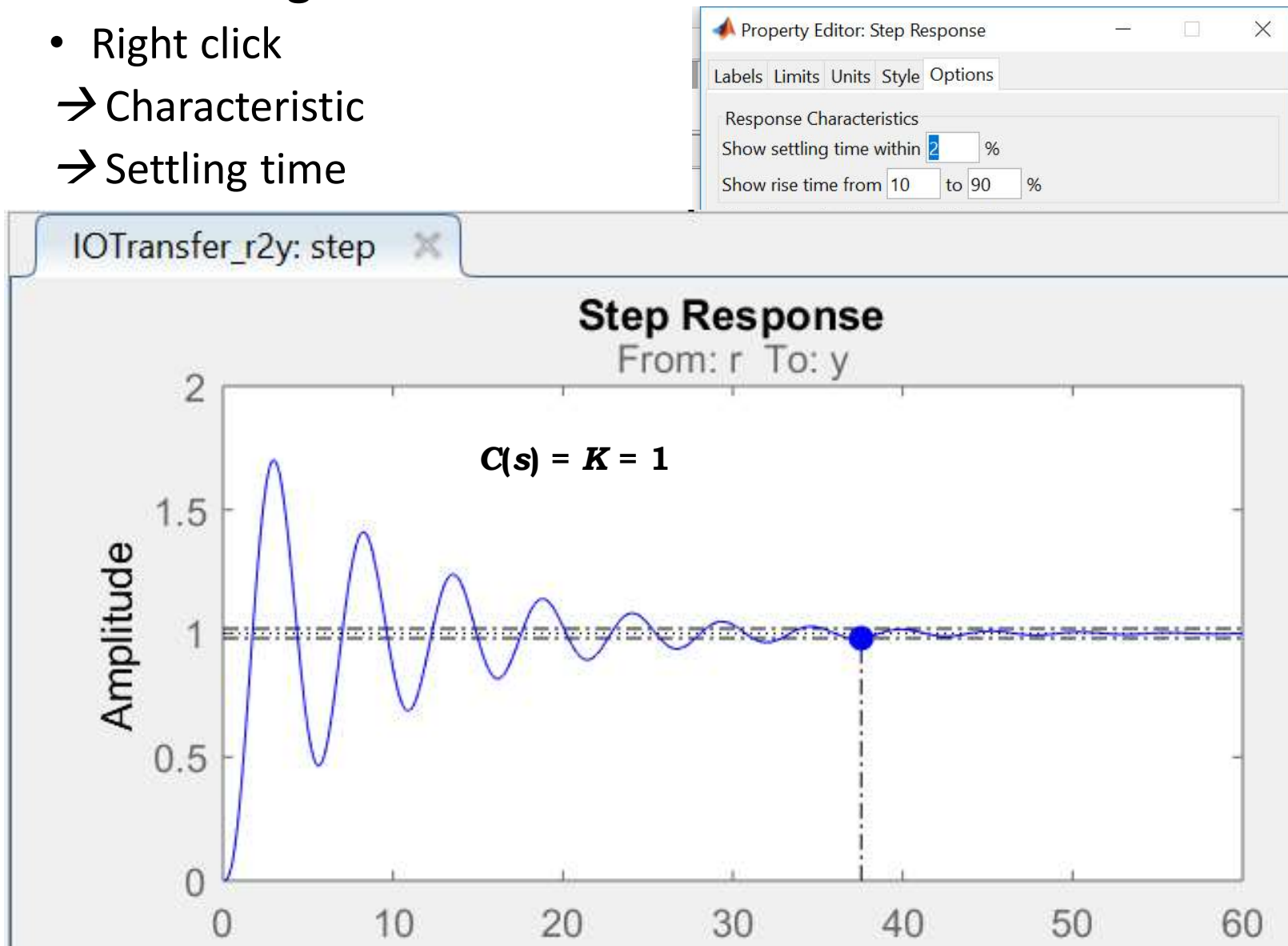


```
>> s = tf('s')
Transfer function:
s
>> sysG = 4/s/(s+1)/(s+2)
Transfer function:
4
-----
s^3 + 3 s^2 + 2 s
>> sisotool(sysG)
```

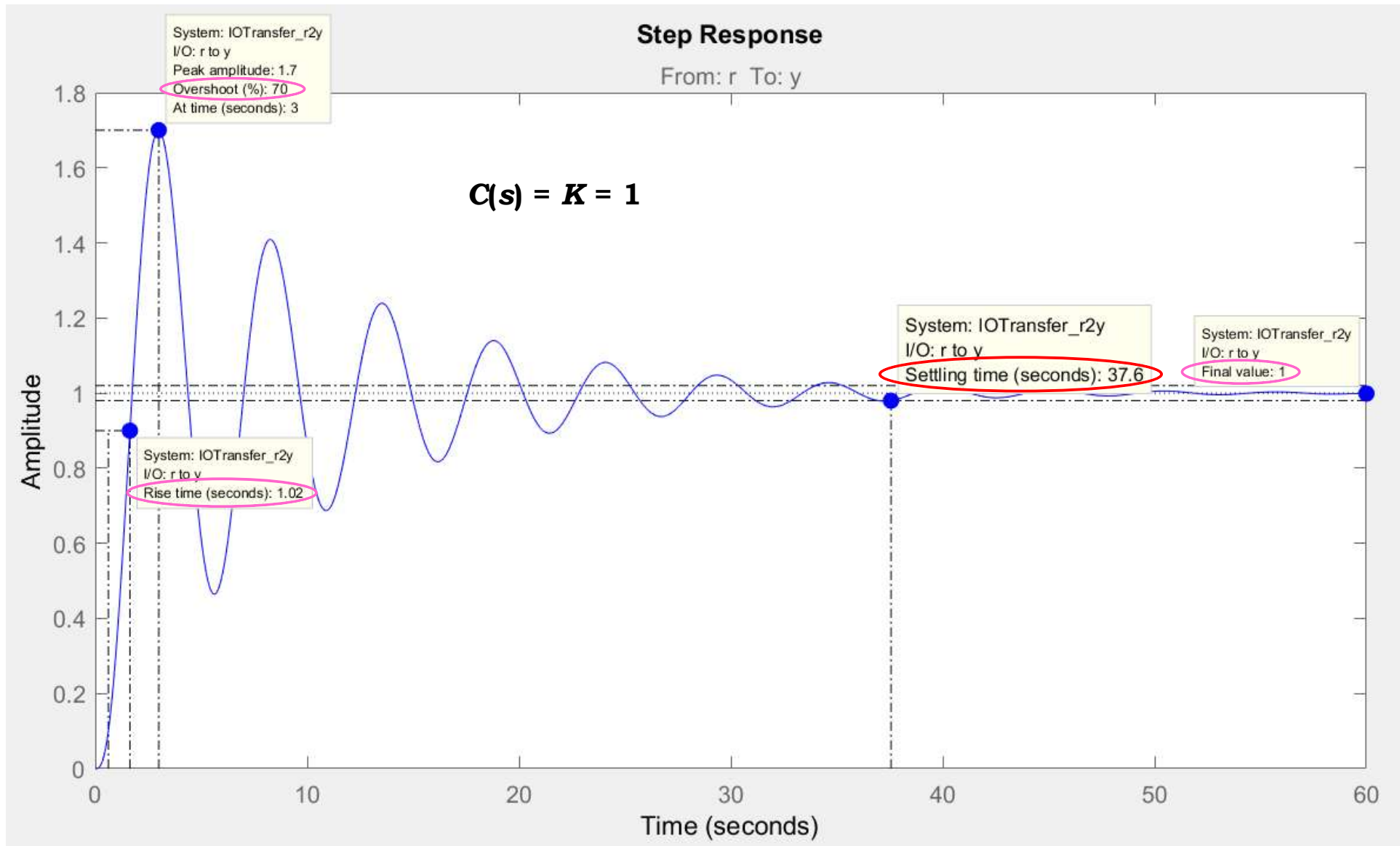
Default setting:  
 $C(s) = K = 1$

# Example 1 (cont'd)

- Show settling time
  - Right click
    - Characteristic
    - Settling time



# Example 1 (cont'd)



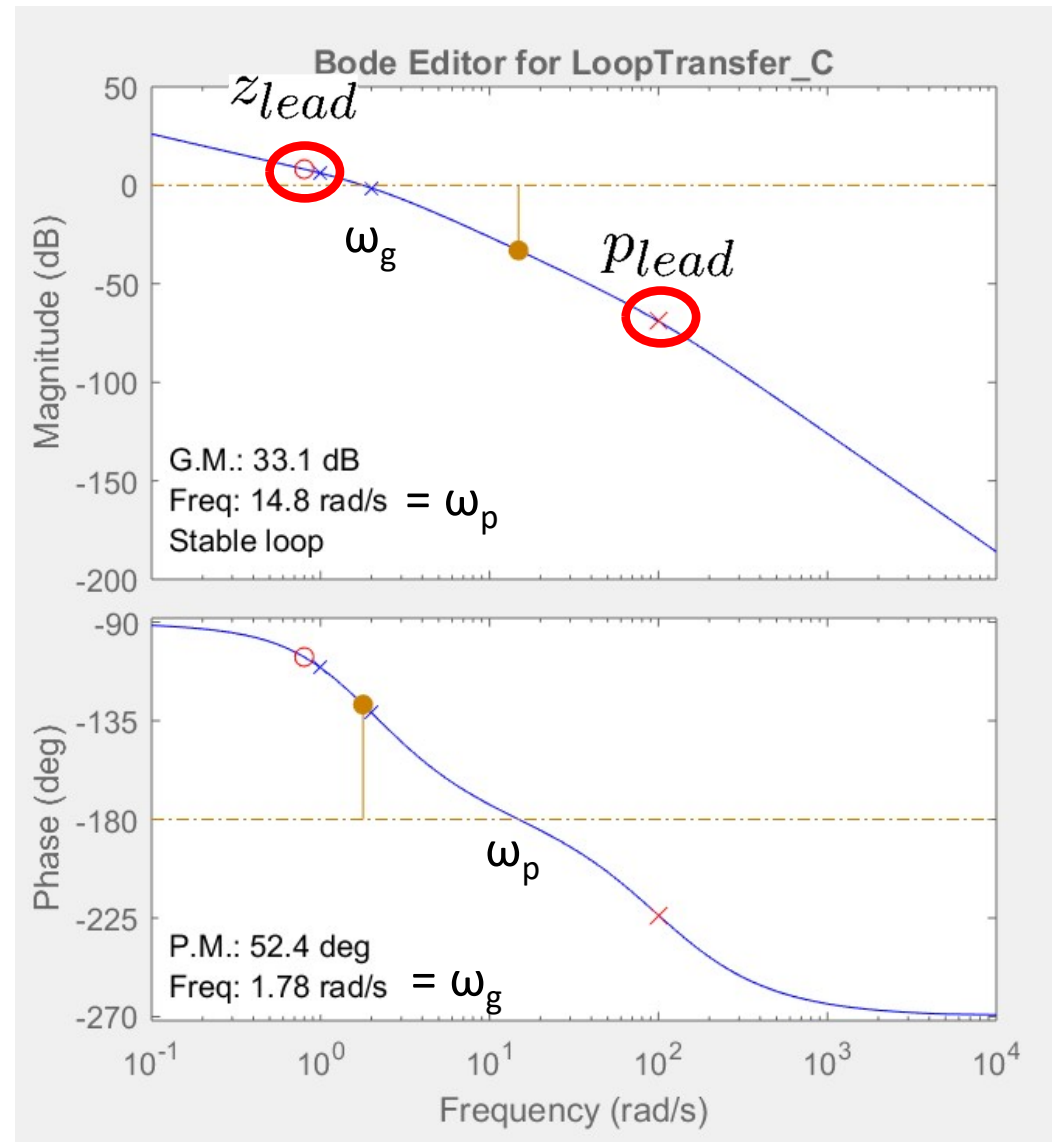
# Example 1 (cont'd)

- Add a pole & a zero of a compensator:

$$C_{Lead}(s) = K \frac{s + z_{Lead}}{s + p_{Lead}}$$

- If necessary, move the pole/zero/gain
  - by click-and-drag, or
  - *Design* → *Edit Compensator...*

```
Tunable Block
Name: C
Sample Time: 0
Value:
  125 (s+0.8)
  -----
        (s+100)
```



**PM (= 52.4) > 50 degree OK!**



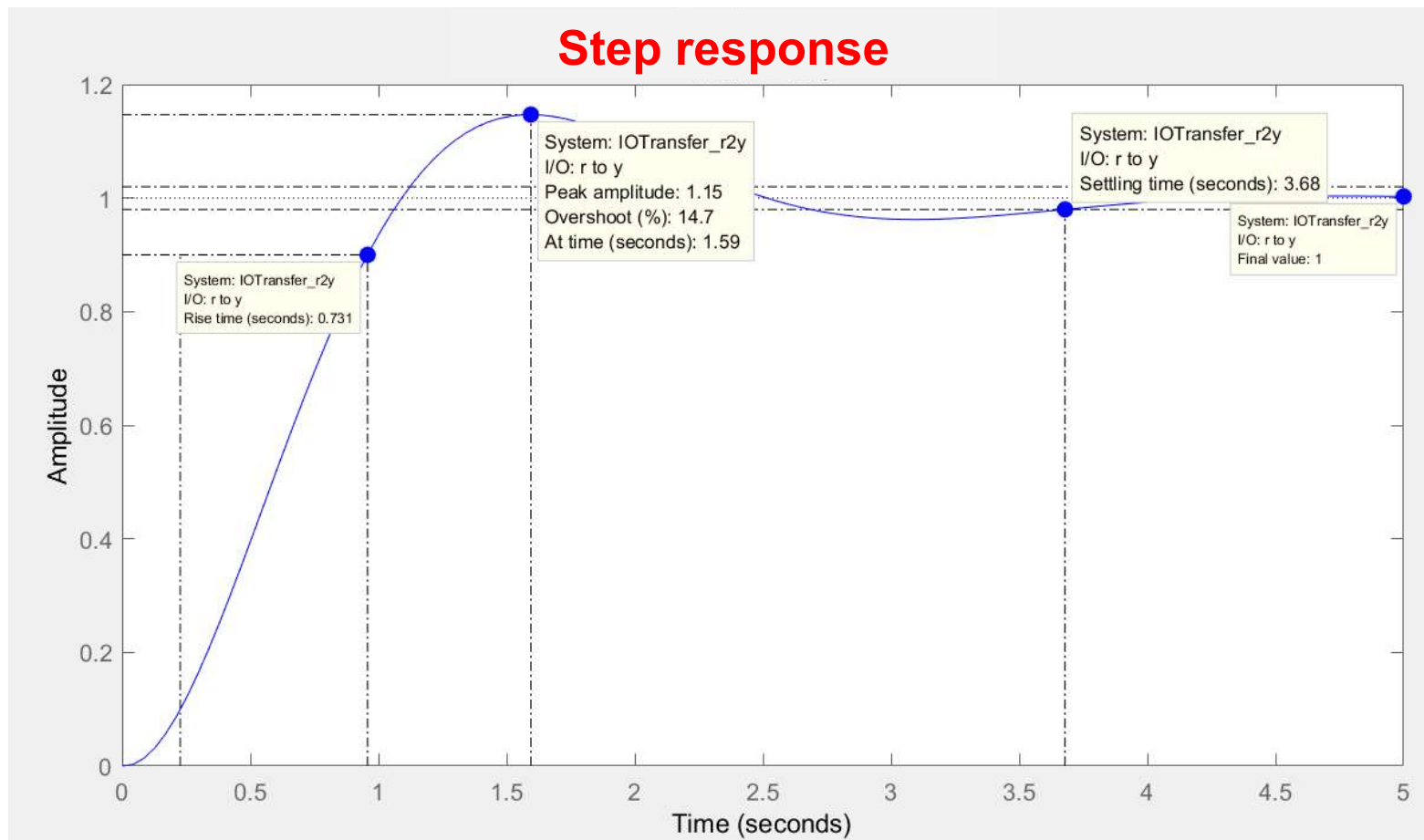
# Example 1 (cont'd)

The following response satisfies all the design specifications:

- (a) Closed-loop system is stable
- (b) PM at least  $50^\circ$ . It is  $52.4^\circ$ .
- (c) 2% Settling time  $< 4$  s. It is 3.68 s.
- (d) Steady-state error is zero for a unit step input

$$G(s) = \frac{4}{s(s+1)(s+2)}$$

$$C_{Lead}(s) = K \frac{s + z_{Lead}}{s + p_{Lead}} = 125 \frac{s + 0.8}{s + 100}$$



$$C_{Lead}(s) \cdot G(s) = 125 \frac{s + 0.8}{s + 100} \cdot \frac{4}{s(s+1)(s+2)}$$



# Simulink

- **Simulink**, developed by MathWorks, is a graphical programming environment for modeling, simulating, and analyzing dynamic systems.
  - Its primary interface is a graphical block diagramming tool and a customizable set of block libraries.
  - It offers tight integration with the rest of the MATLAB environment.
- Simulink is basically a piece of software for modeling and simulating a system, as well as programming and designing controllers.
- Engineers use Simulink to solve engineering problems in many industries, such as:
  - Automotive
  - Biomedical
  - Aerospace
  - Chemical processes
  - Communications
  - Industrial automation
  - Electronics
  - etc.

# Simulink



- Simulink in MATLAB can be directly used with **Arduino** to design, simulate, and deploy control systems and embedded applications without writing traditional code.
- With Simulink support packages for Arduino, you can build a block diagram (***instead of writing code***), simulate how your system behaves, and then download the model directly onto the Arduino hardware.
- This allows you to test real-time control algorithms, read sensors, and drive actuators using a **visual programming** approach, making it ideal for rapid prototyping and education.

# Simulink



## What is Visual Programming?

- A **visual programming approach** means you create programs by **connecting blocks or components in a graphical interface**, rather than writing lines of text-based code. You "program" by dragging, dropping, and linking blocks that represent operations (like reading a sensor, doing math, or turning on an LED).

## How Simulink Uses It:

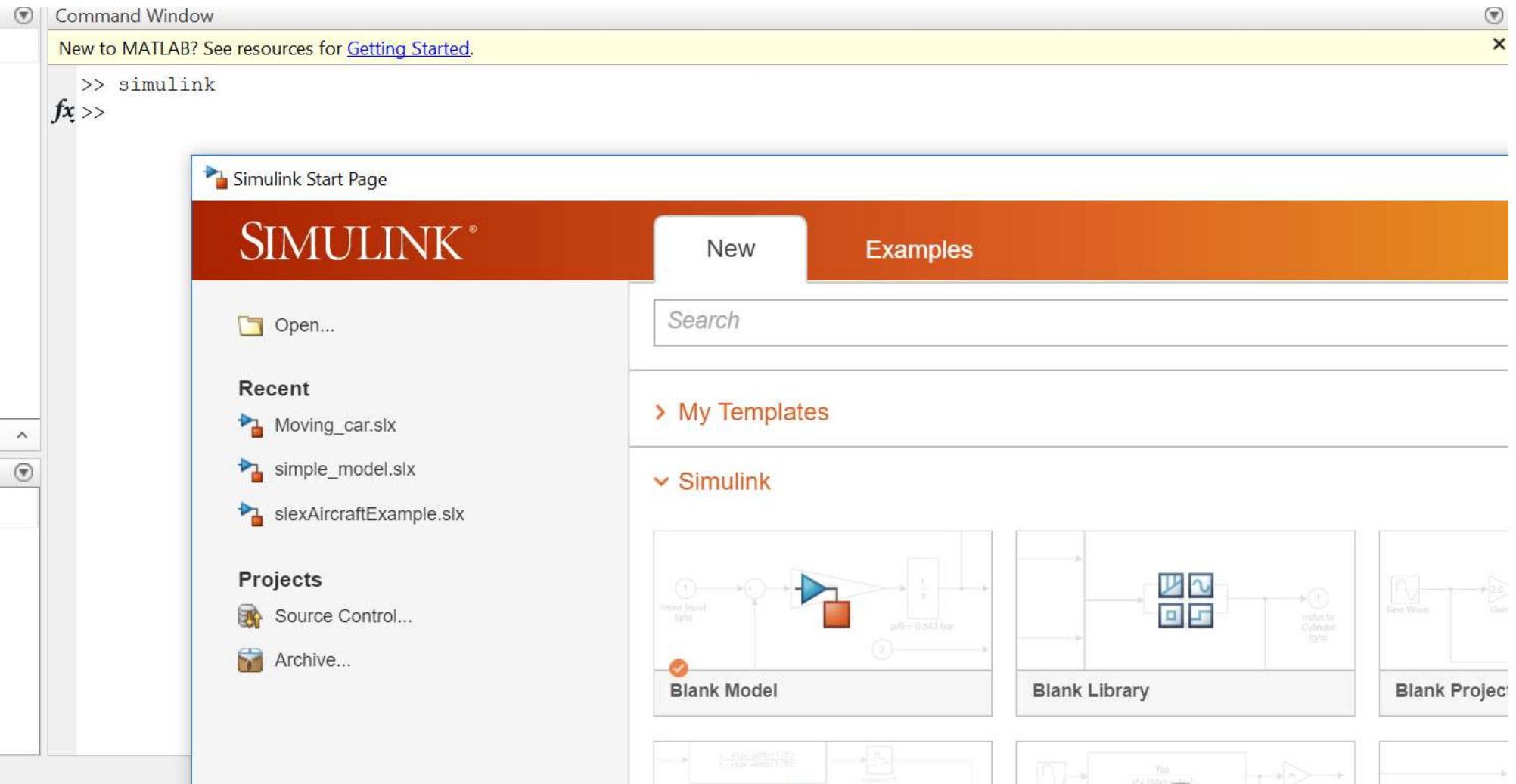
- In **Simulink**, you build your system using block diagrams — each block performs a specific function. For example:
  - A block for reading analog input
  - A block for multiplying a signal
  - A block for outputting a value to a motor
- Once your model is complete, Simulink converts it into C++ code behind the scenes and uploads it to the Arduino.

## Summary:

- **C++** is what Arduino typically runs.
- **Simulink** lets you skip writing C++ by **visually designing** your system.
- This is helpful for students, engineers, and educators who want to focus on logic and control, not on low-level code.

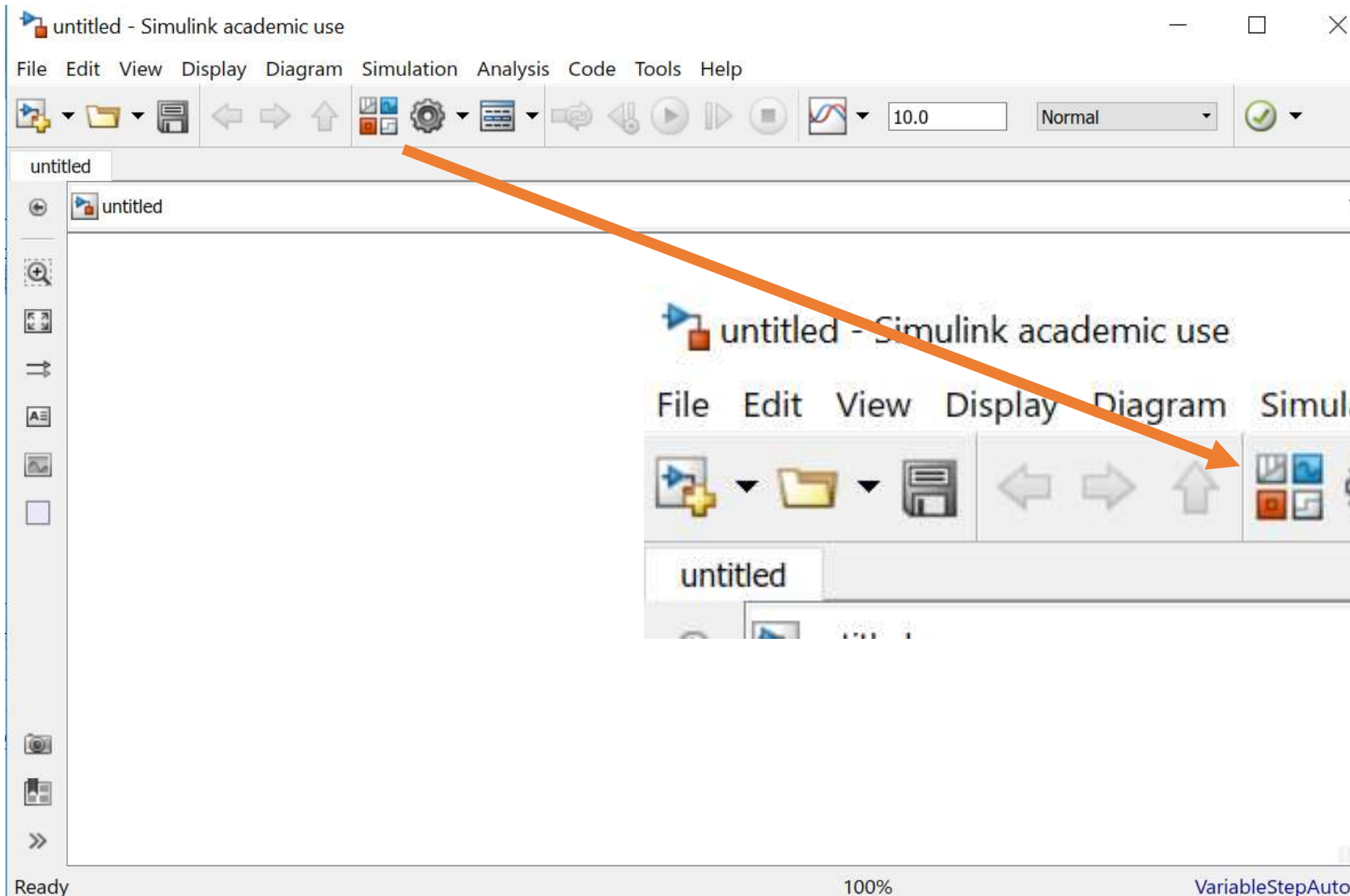
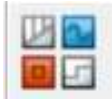
## Example 2 (cont'd)

- In MATLAB prompt, type “simulink”.
- Click on Blank Model.



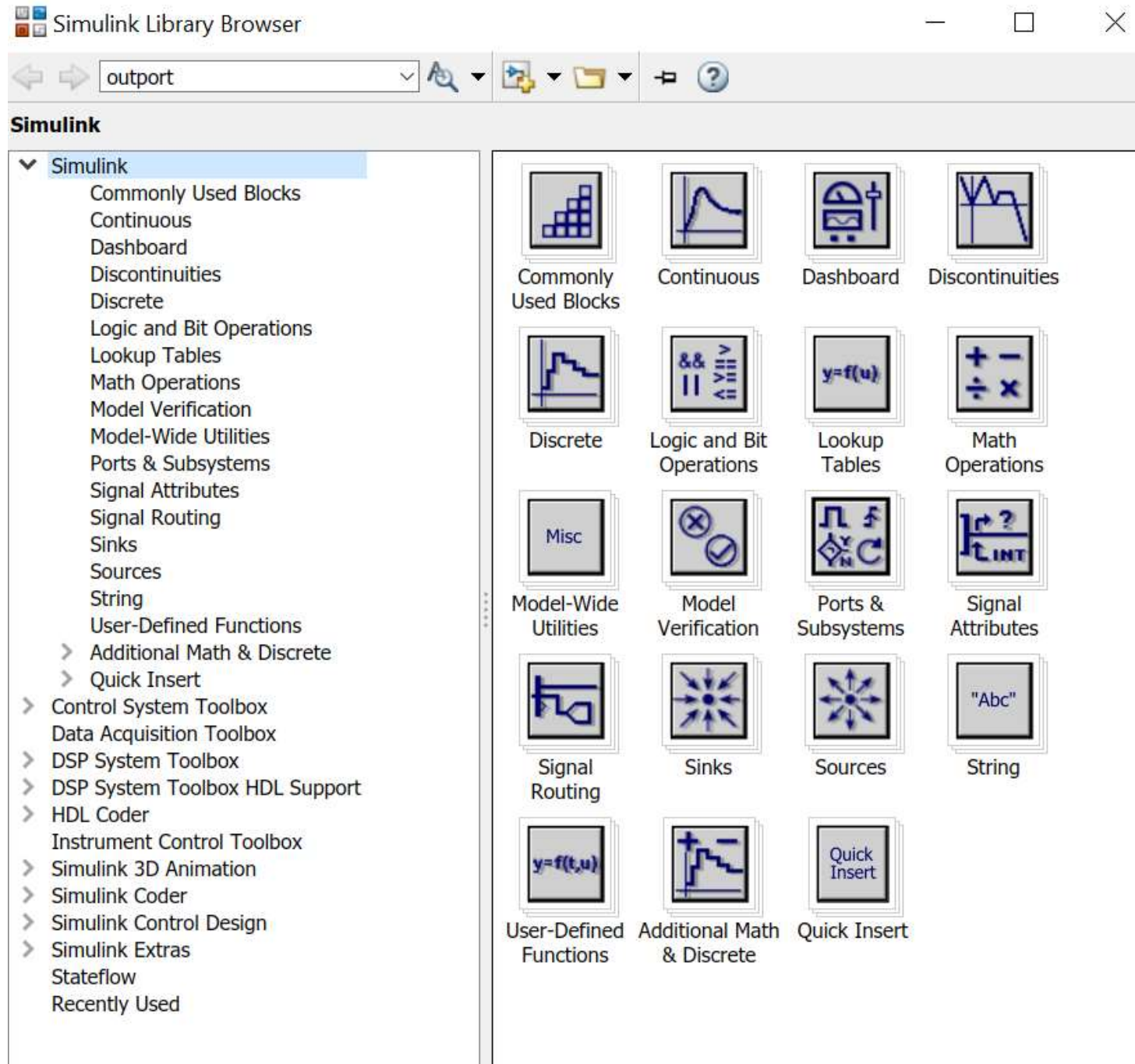
## Example 2 (cont'd)

- Click on



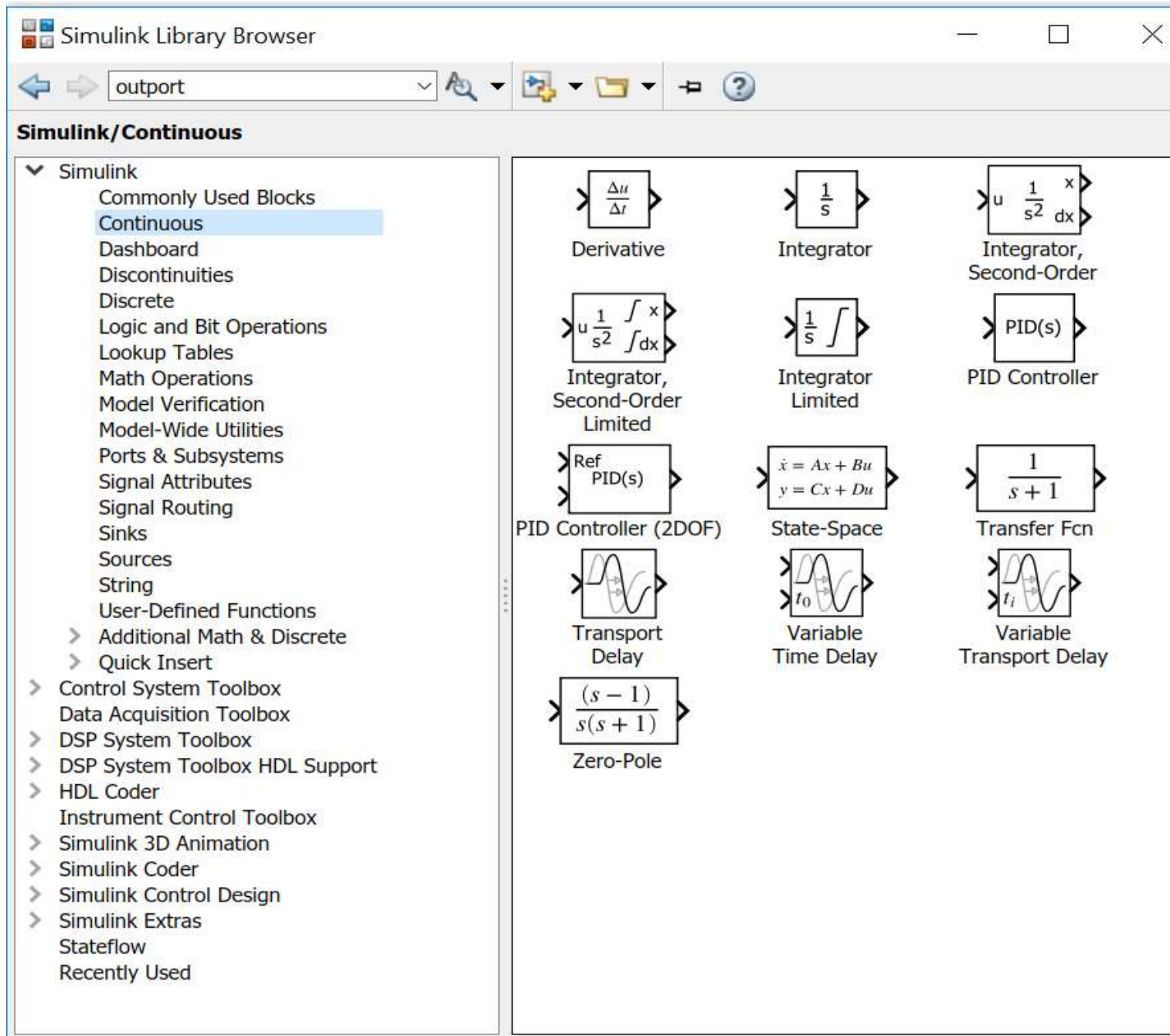
# Example 2 (cont'd)

Then, Simulink Library Browser pops up:





## Example 2 (cont'd)



Simulink Library Browser

Search: output

**Simulink/Continuous**

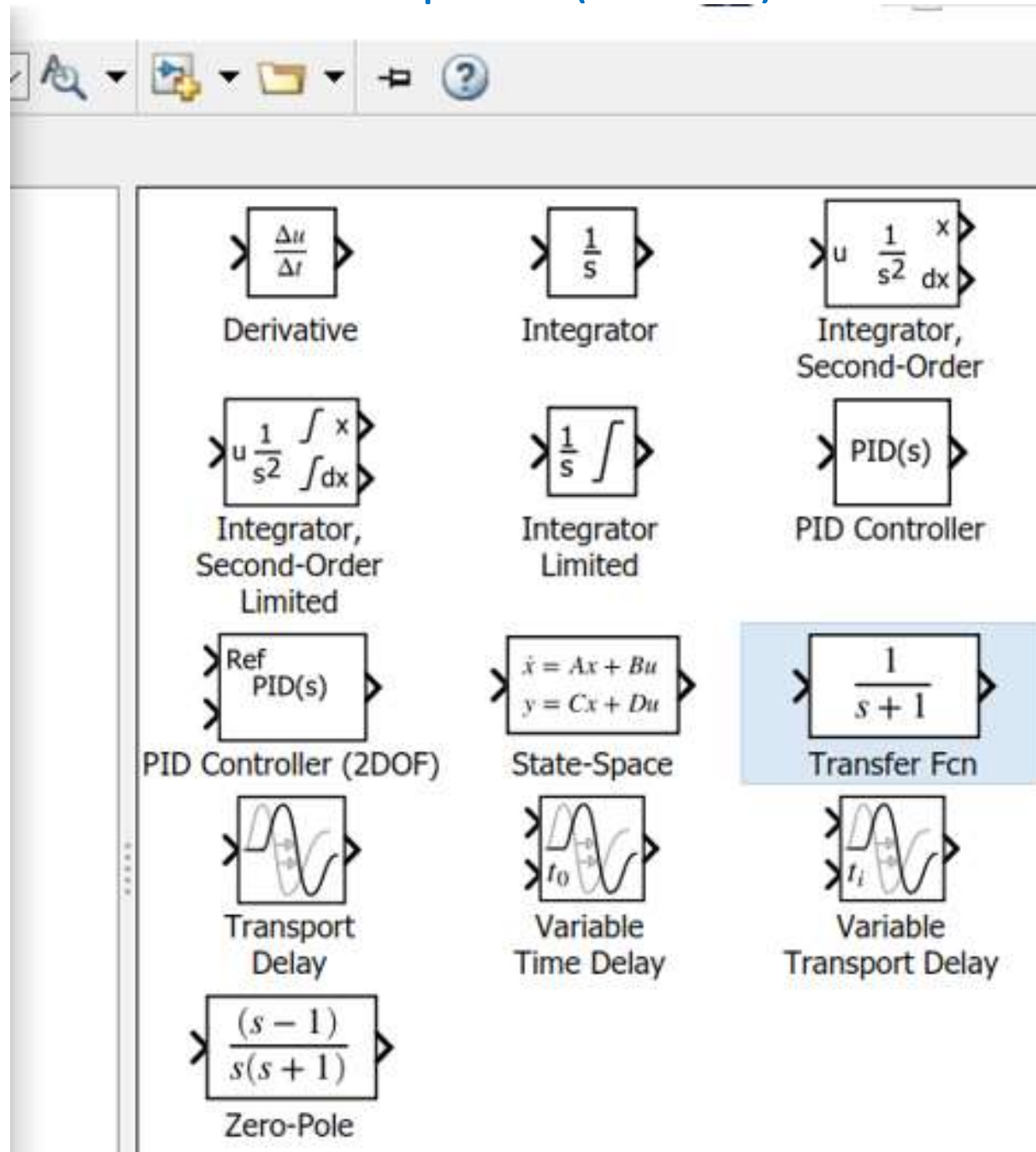
- Simulink
  - Commonly Used Blocks
  - Continuous**
  - Dashboard
  - Discontinuities
  - Discrete
  - Logic and Bit Operations
  - Lookup Tables
  - Math Operations
  - Model Verification
  - Model-Wide Utilities
  - Ports & Subsystems
  - Signal Attributes
  - Signal Routing
  - Sinks
  - Sources
  - String
  - User-Defined Functions
  - > Additional Math & Discrete
  - > Quick Insert
  - > Control System Toolbox
  - > Data Acquisition Toolbox
  - > DSP System Toolbox
  - > DSP System Toolbox HDL Support
  - > HDL Coder
  - > Instrument Control Toolbox
  - > Simulink 3D Animation
  - > Simulink Coder
  - > Simulink Control Design
  - > Simulink Extras
  - Stateflow
  - Recently Used

Available Blocks:

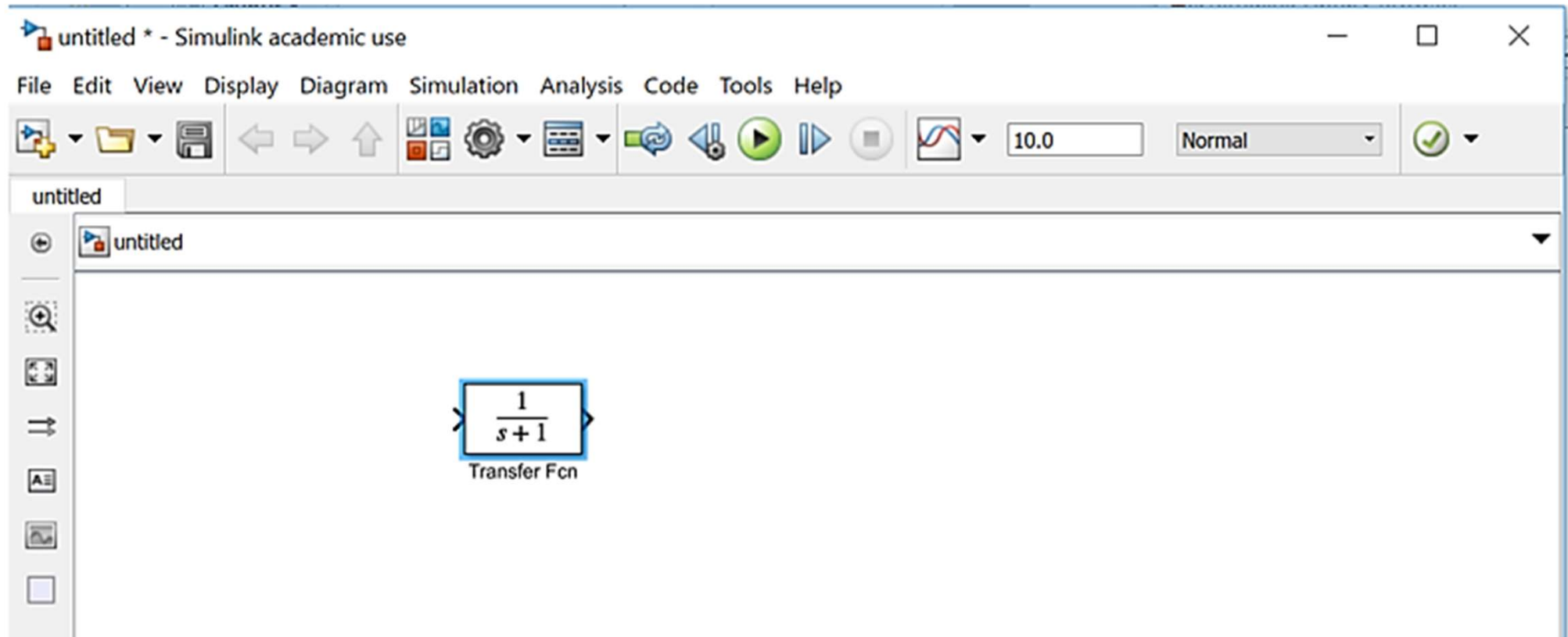
- Derivative:  $\frac{\Delta u}{\Delta t}$
- Integrator:  $\frac{1}{s}$
- Integrator, Second-Order:  $u \frac{1}{s^2} x \frac{dx}{dx}$
- Integrator, Second-Order Limited:  $u \frac{1}{s^2} \int \frac{dx}{dx}$
- Integrator Limited:  $\frac{1}{s} \int$
- PID Controller: PID(s)
- PID Controller (2DOF): Ref PID(s)
- State-Space:  $\dot{x} = Ax + Bu$ ,  $y = Cx + Du$
- Transfer Fcn:  $\frac{1}{s+1}$
- Transport Delay:  $t_0$
- Variable Time Delay:  $t_0$
- Variable Transport Delay:  $t_i$
- Zero-Pole:  $\frac{(s-1)}{s(s+1)}$



## Example 2 (cont'd)



## Example 2 (cont'd)



## Example 2 (cont'd)

Double-click on the block to enter new numerator and denominator.

Block Parameters: Transfer Fcn

Transfer Fcn

The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.

Parameters

Numerator coefficients:

[1 6 10]

Denominator coefficients:

[1 7 1]

Absolute tolerance:

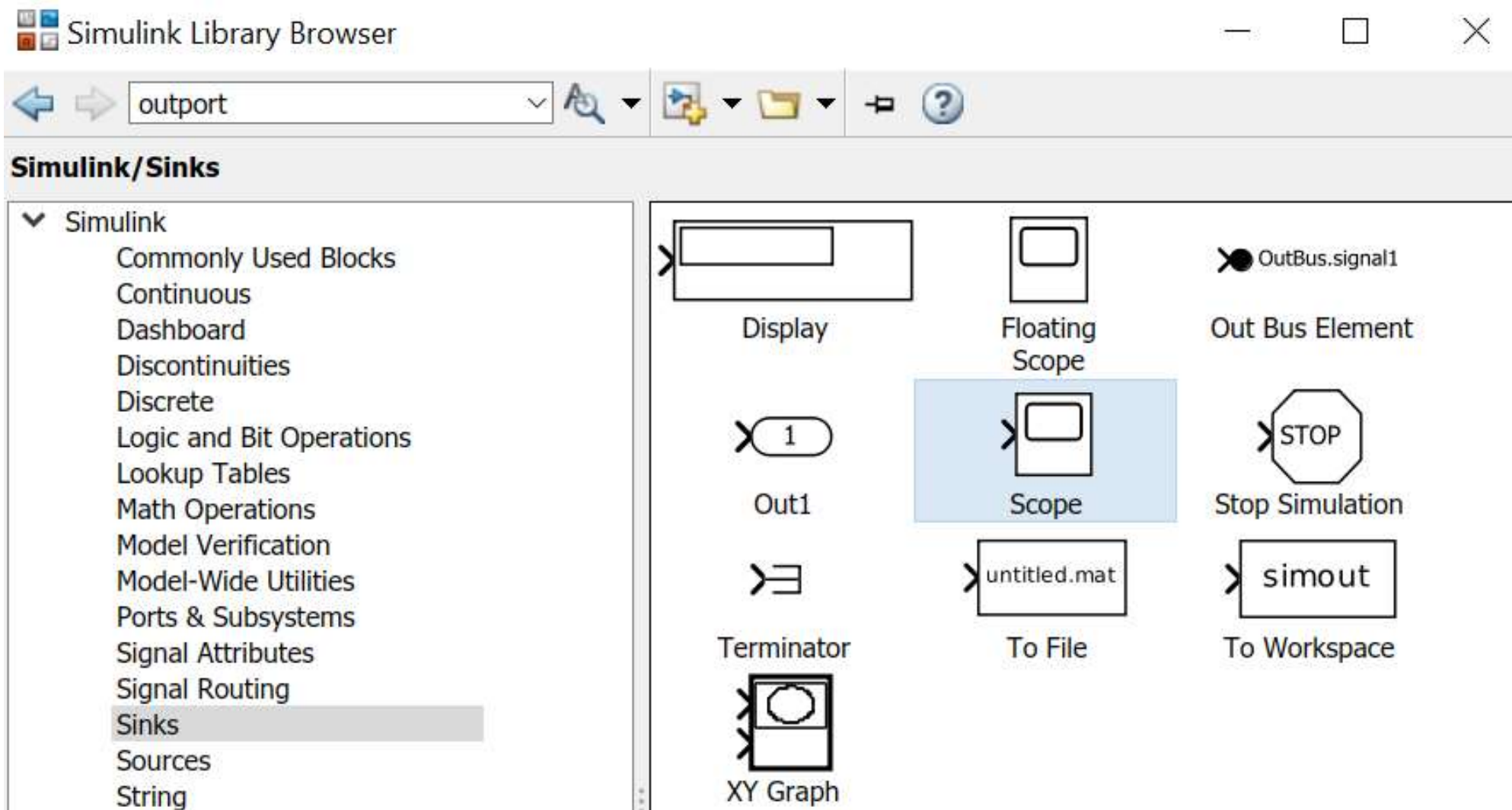
auto

State Name: (e.g., 'position')

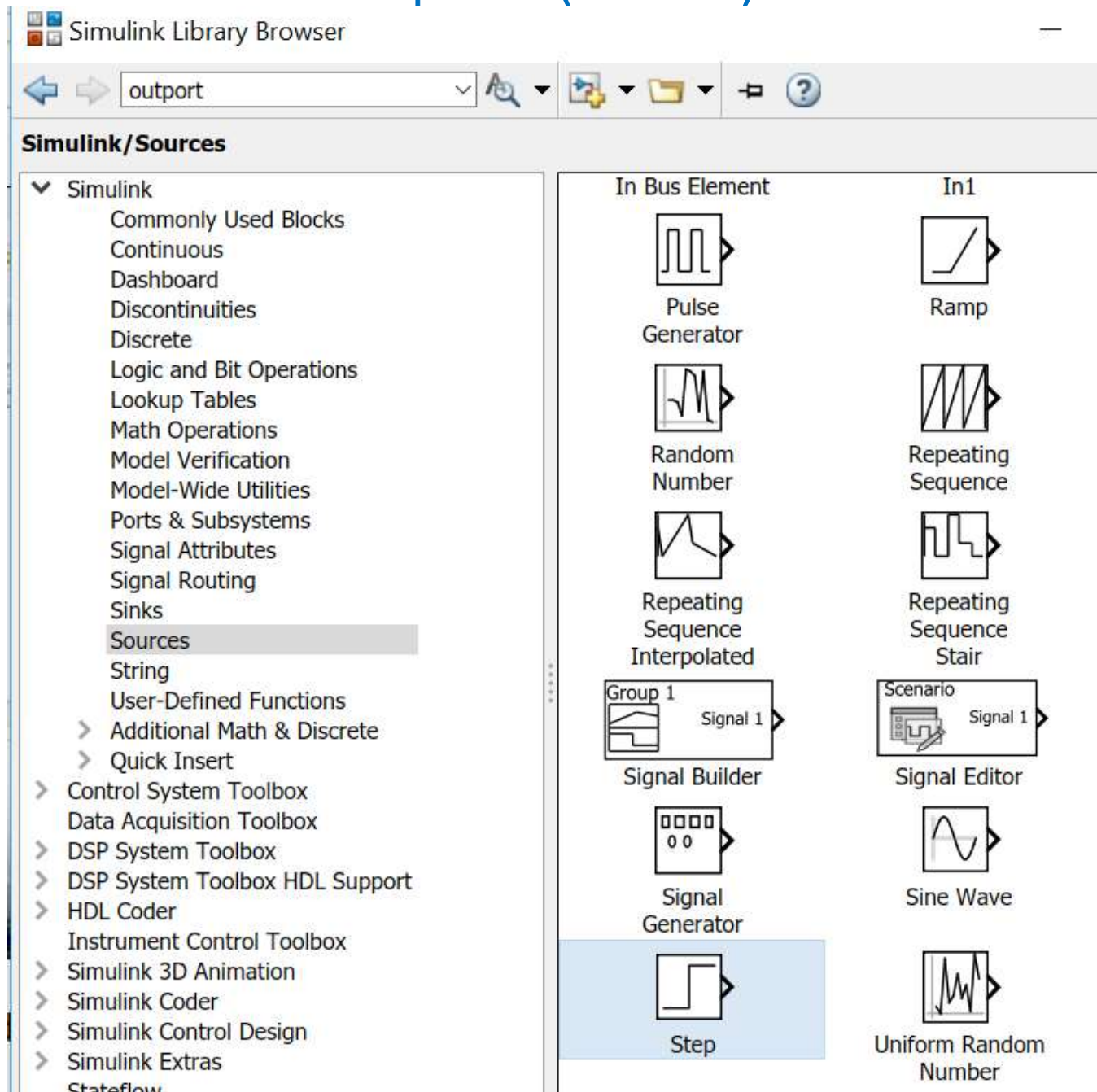
"

OK Cancel Help Apply

## Example 2 (cont'd)



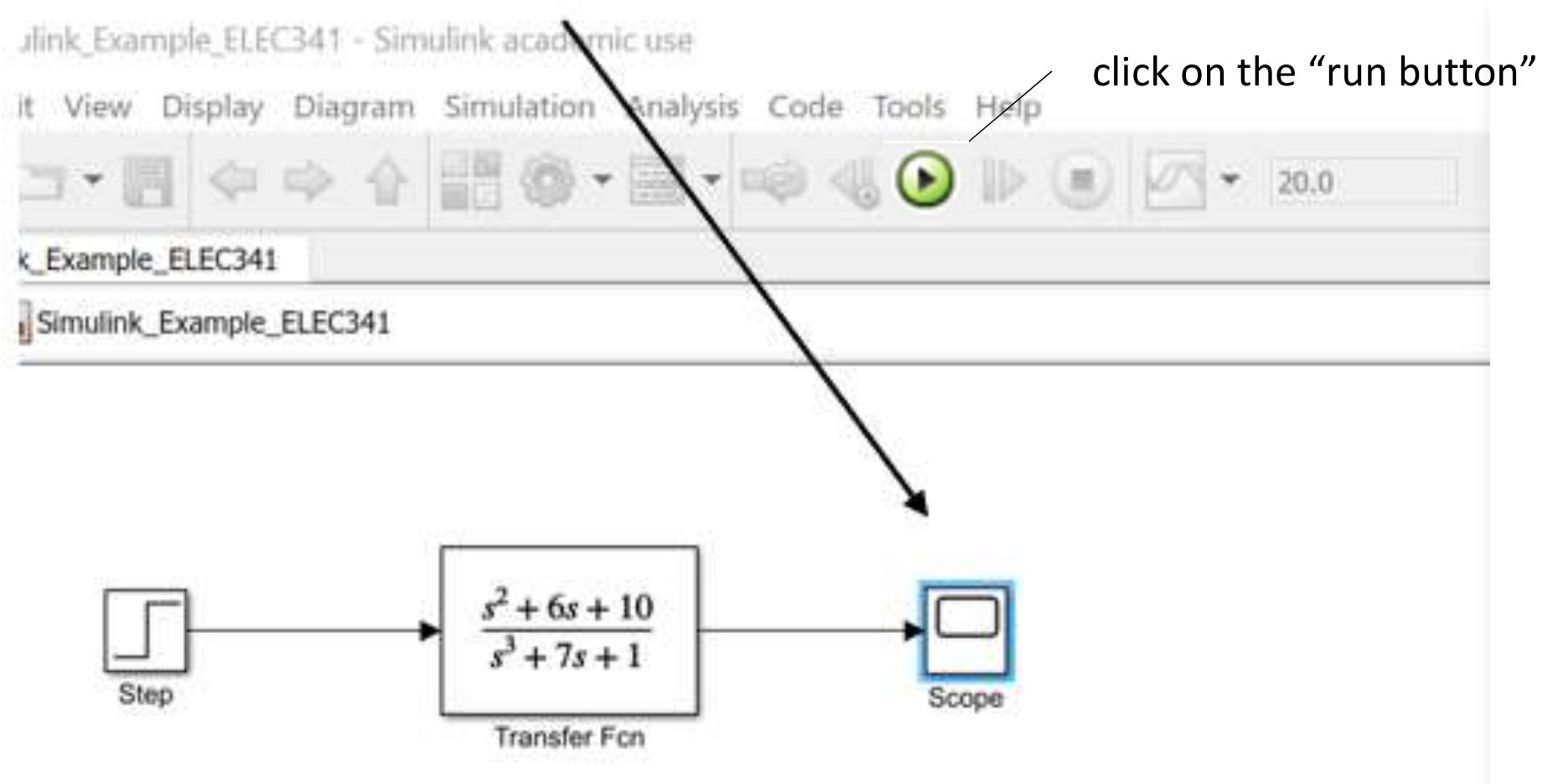
## Example 2 (cont'd)





## Example 2 (cont'd)

Double-click on scope.



Simulink Example\_ELEC341 - Simulink academic use

File View Display Diagram Simulation Analysis Code Tools Help

click on the "run button"

Simulink\_Example\_ELEC341

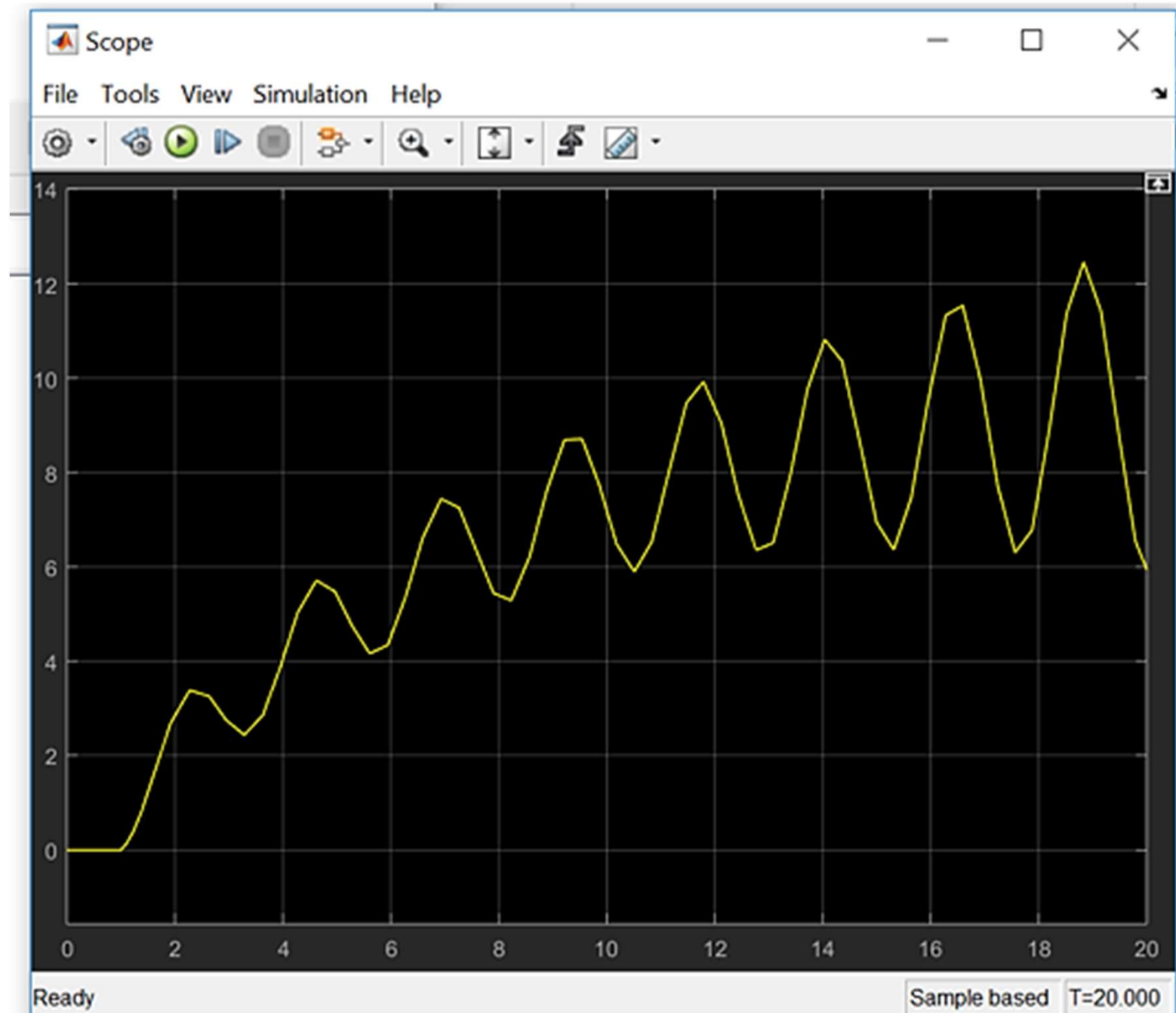
Step

$$\frac{s^2 + 6s + 10}{s^3 + 7s + 1}$$

Transfer Fcn

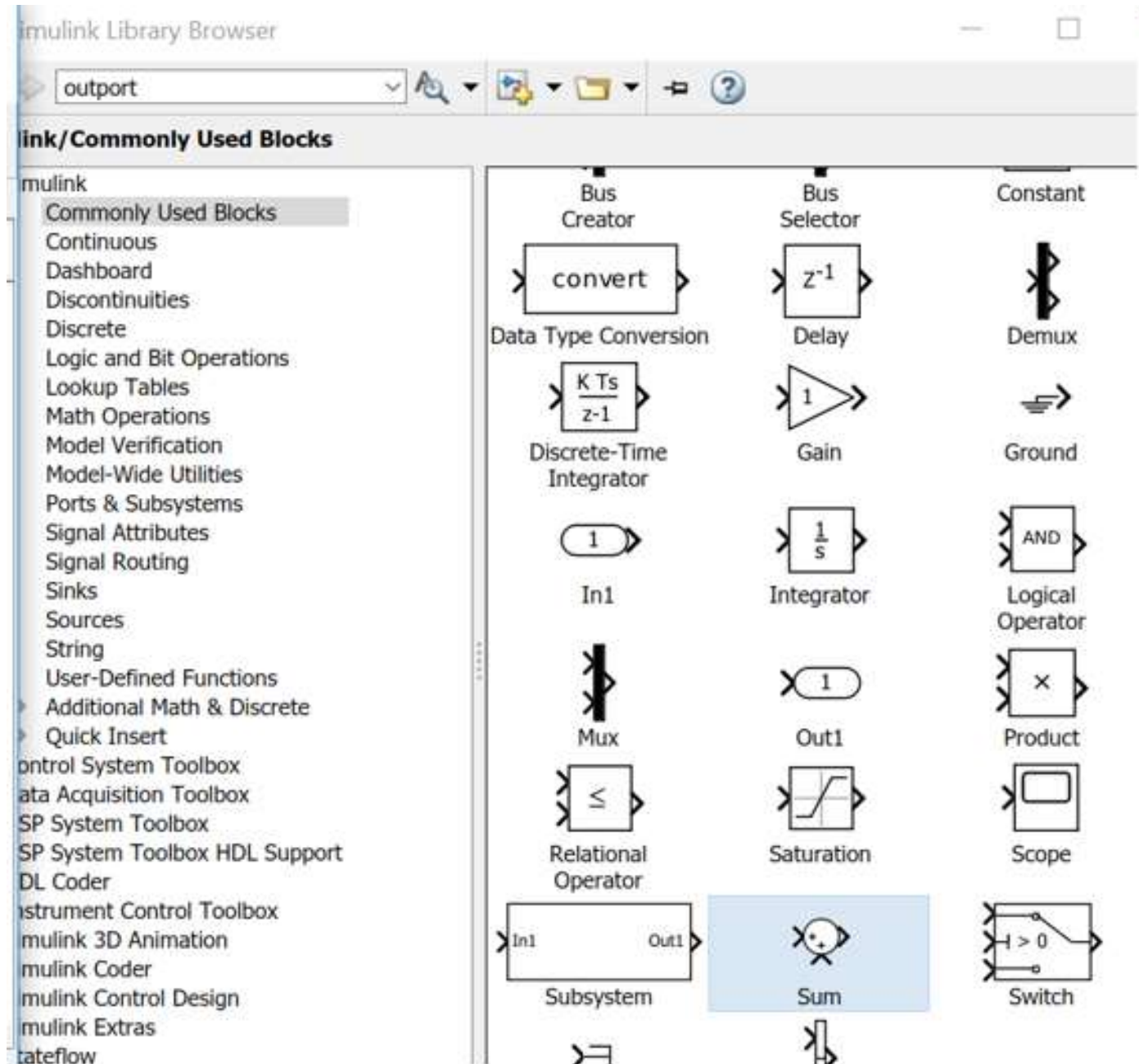
Scope

## Example 2 (cont'd)





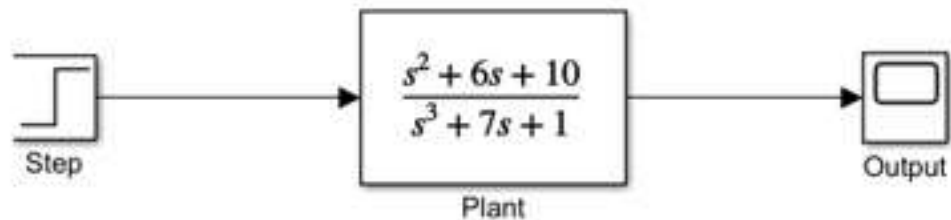
## Example 2 (cont'd)



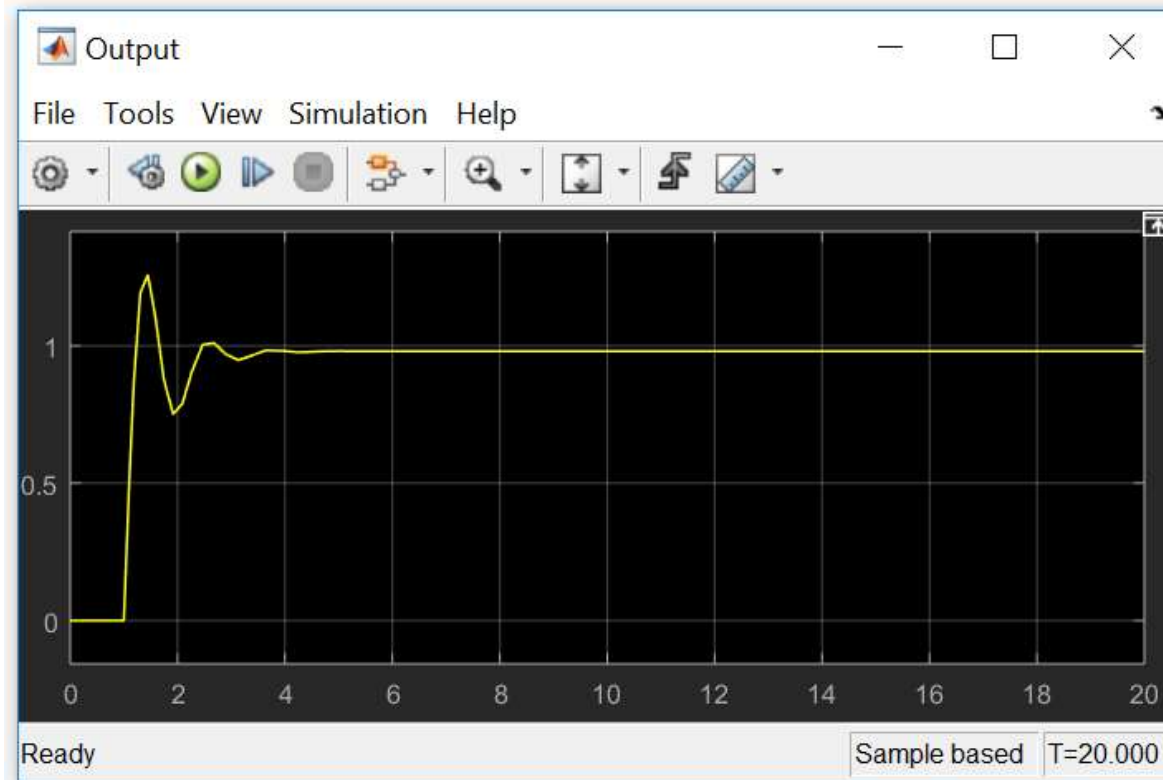
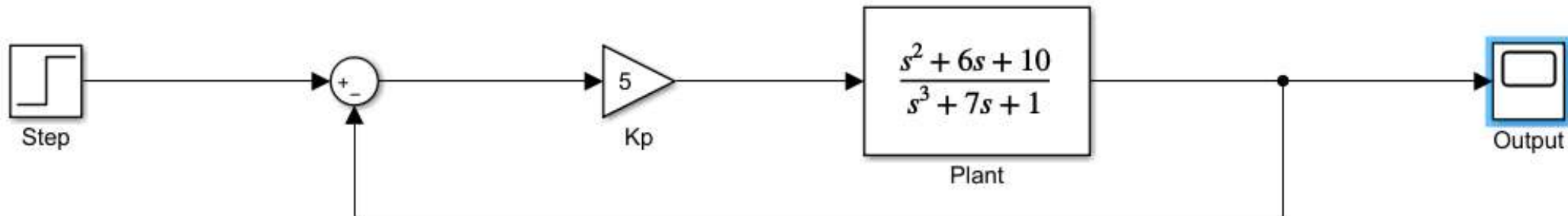
## Example 2 (cont'd)

Example\_ELEC341 \* - Simulink academic use

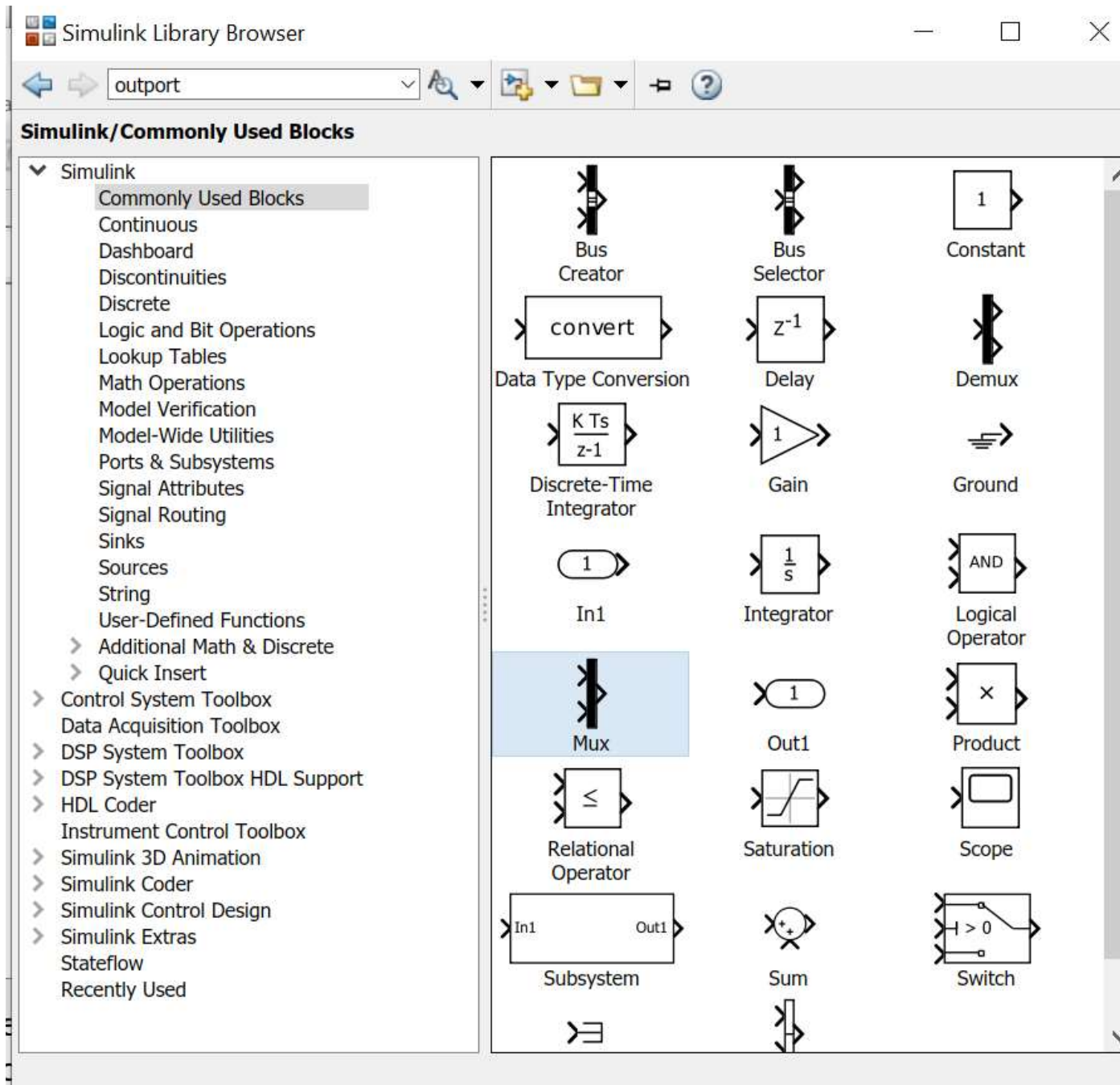
File Edit View Display Diagram Simulation Analysis Code Tools Help



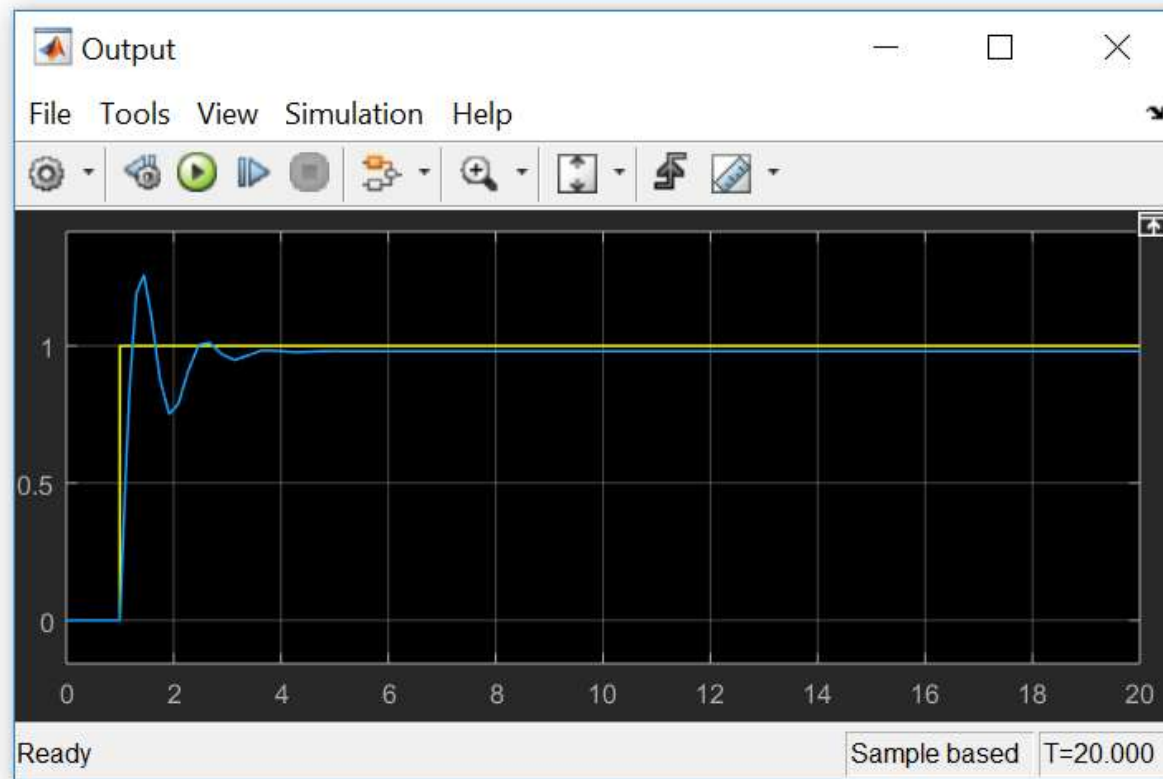
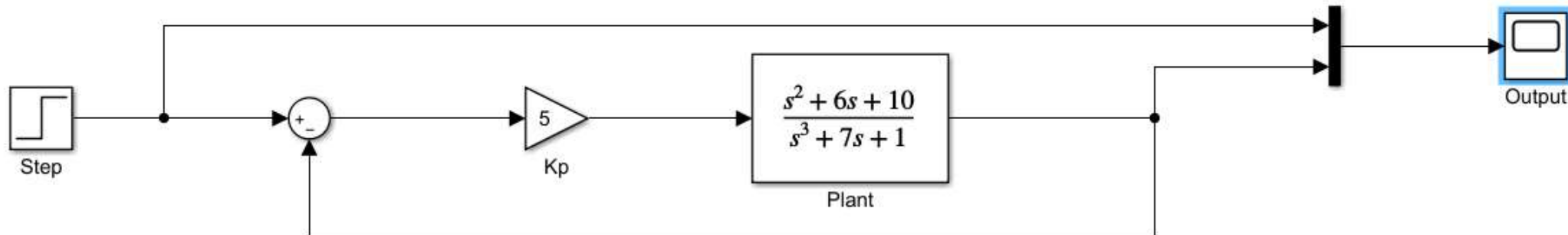
## Example 2 (cont'd)



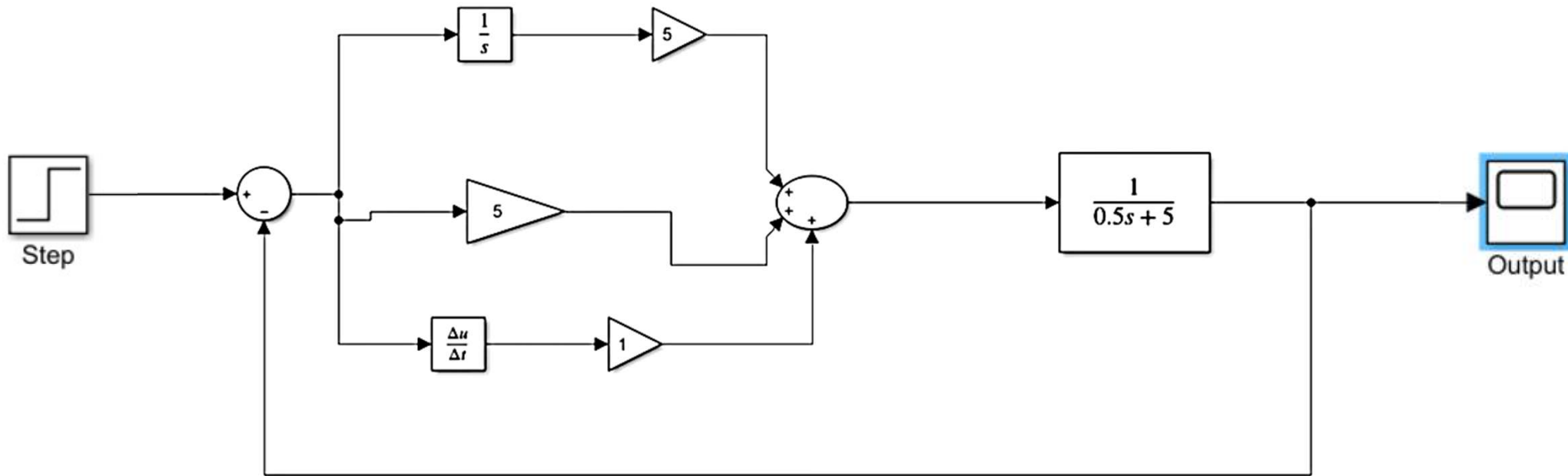
## Example 2 (cont'd)



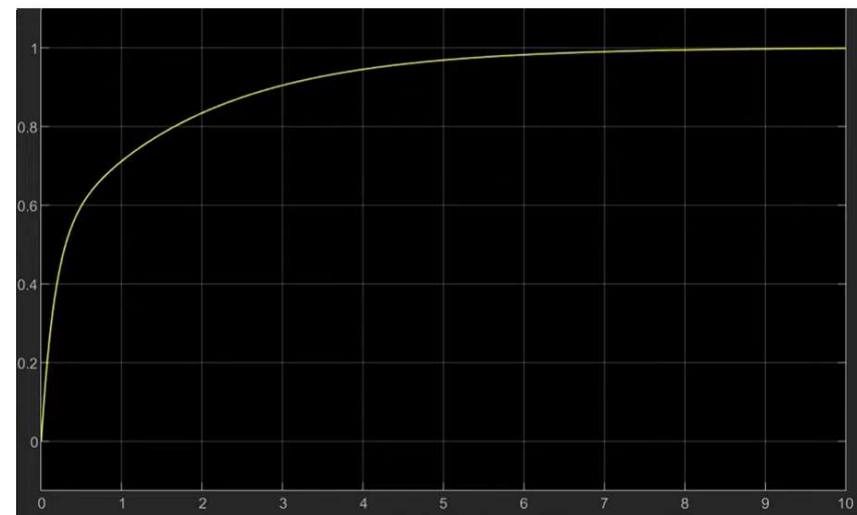
## Example 2 (cont'd)



## Example 3 (PID Controller Design)



**Note:** We can also use a block with “ $s$ ” inside it as the derivative component of the PID controller design (instead of  $\frac{\Delta u}{\Delta t}$ ).





# Course roadmap

## Modeling

- ✓ Laplace transform
- ✓ Transfer function
- Models for systems
  - ✓ • Electrical
  - ✓ • Electromechanical
  - ✓ • Mechanical
- ✓ Linearization, delay

## Analysis

- ✓ Stability
  - ✓ • Routh-Hurwitz
  - ✓ • Nyquist
- ⇨ ✓ Time response
  - ✓ • Transient
  - ✓ • Steady state
- ✓ Frequency response
  - ✓ • Bode plot

## Design

- ✓ Design specs
- ✓ Root locus
- ⇨ ✓ Frequency domain
- ✓ PID & Lead-lag
- ✓ Design examples

✓ *Matlab simulations*

*Thank You!*



# The End