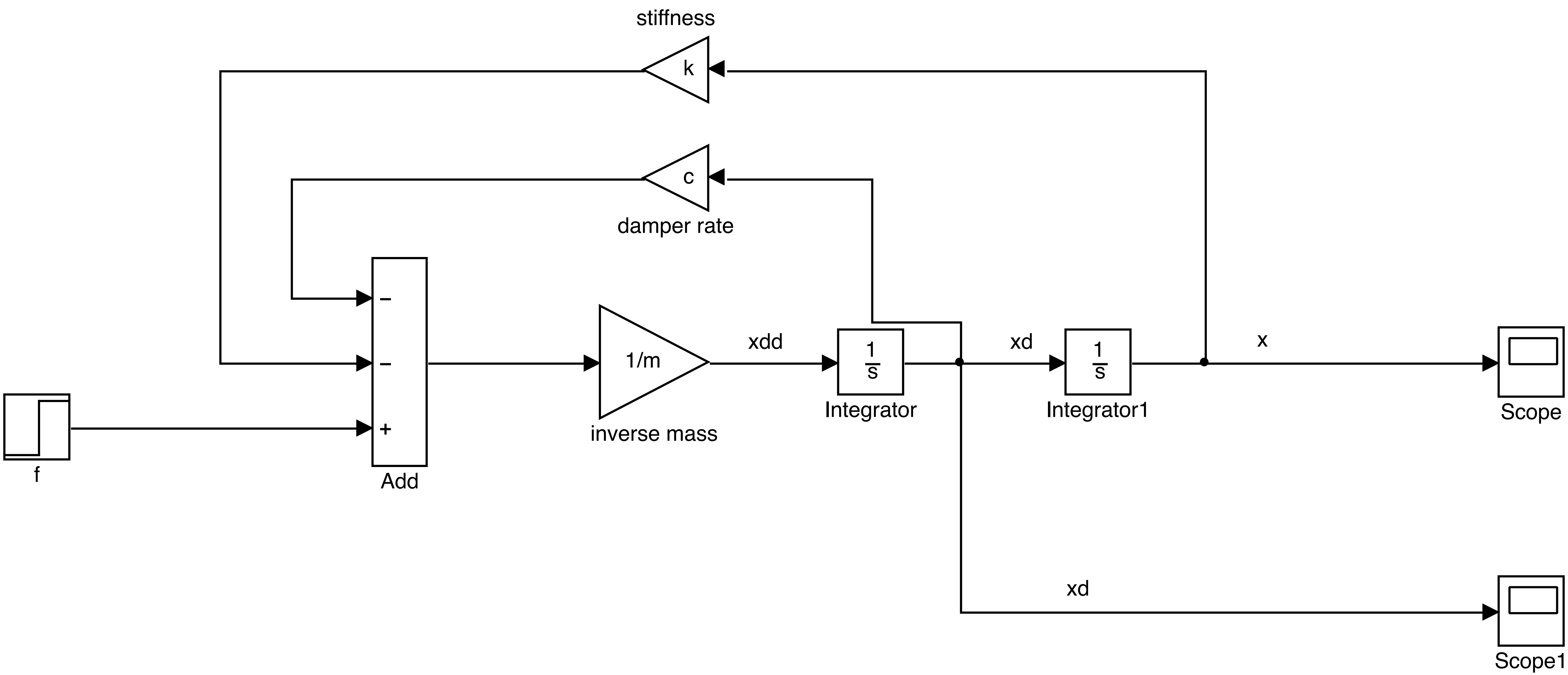
ME40064: System Modelling & Simulation
ME50344: Engineering Systems Simulation
Lecture 18

Dr Andrew Cookson
University of Bath, 2019-20

- Understand how to build a subsystem model in Simulink
- Appreciation of potential complexity and sophistication of Simulink models
- Understand some of Simulink's solvers and relevant settings

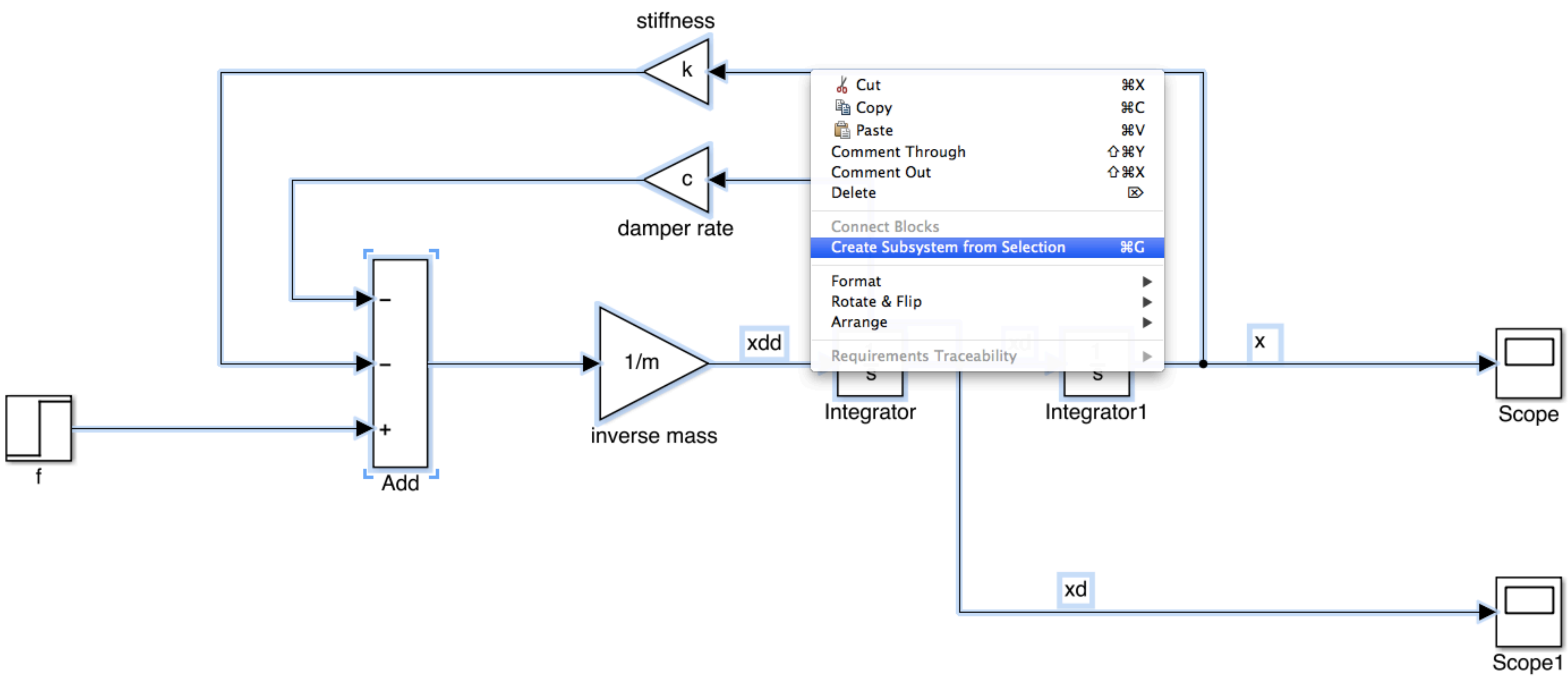
Creating A Subsystem Block Model

Start with a block diagram model



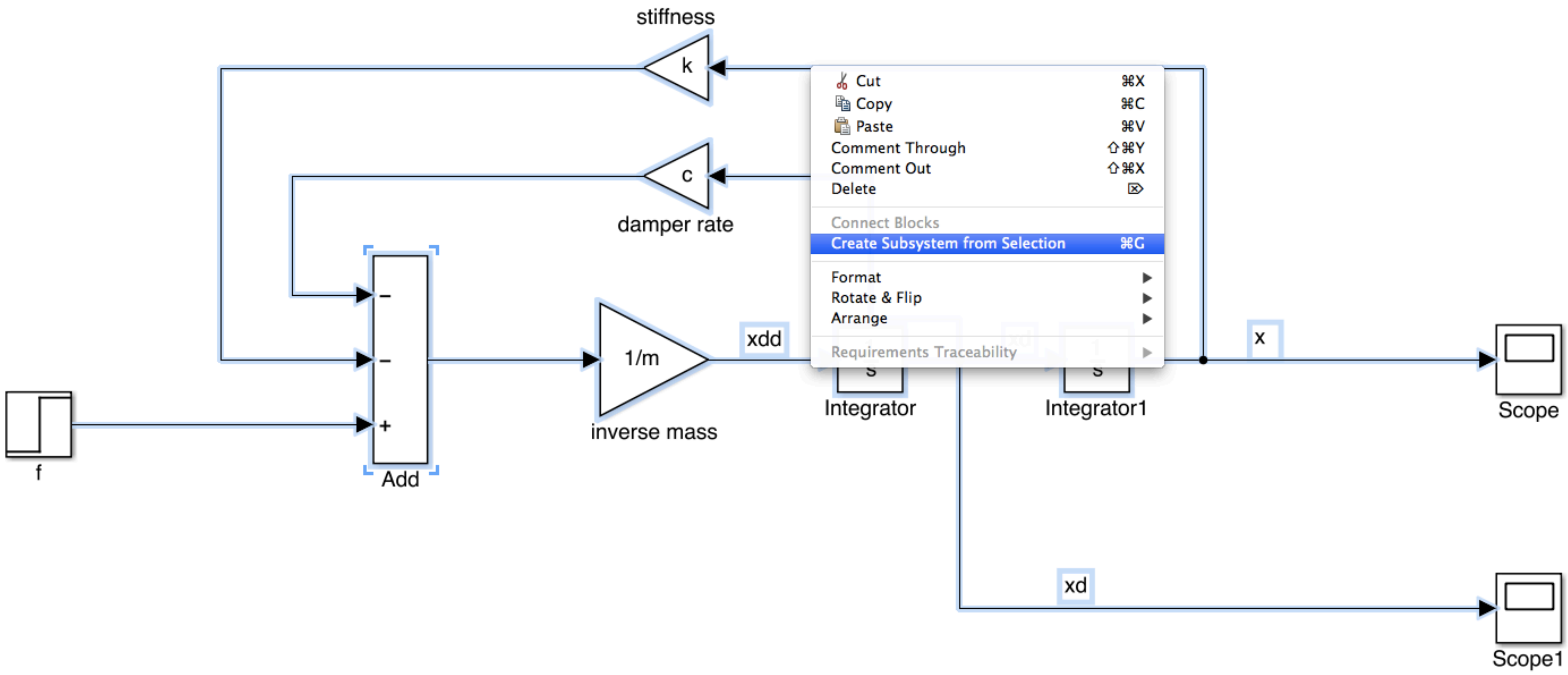
Creating A Subsystem Block Model

Select the model - excluding inputs and outputs, then click the "Build subsystem option"



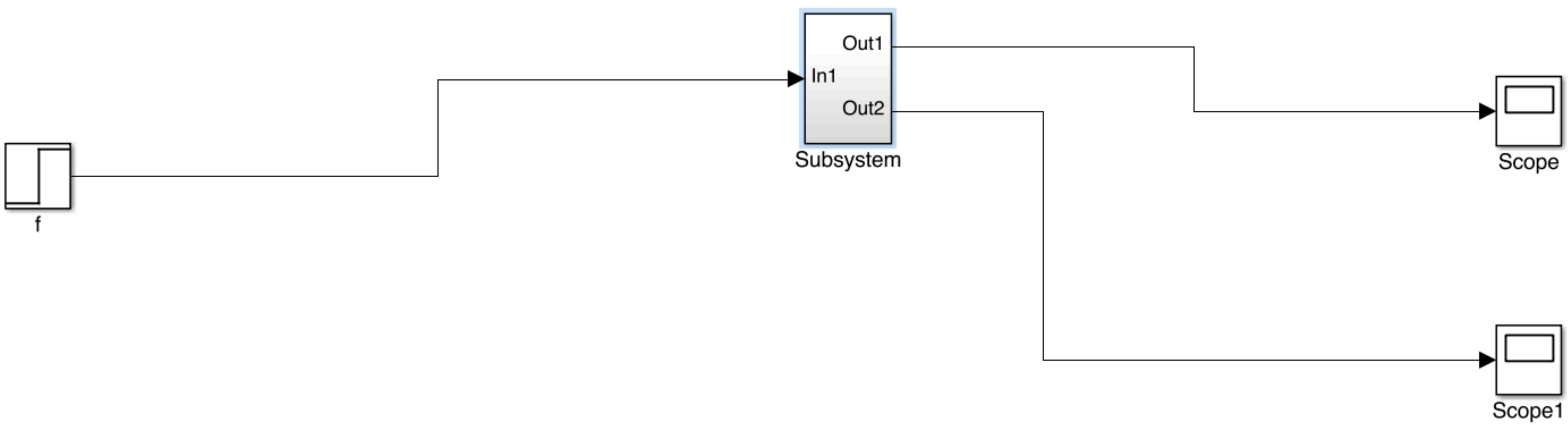
Creating A Subsystem Block Model

Select the model - excluding inputs and outputs, then click the "Build subsystem option"



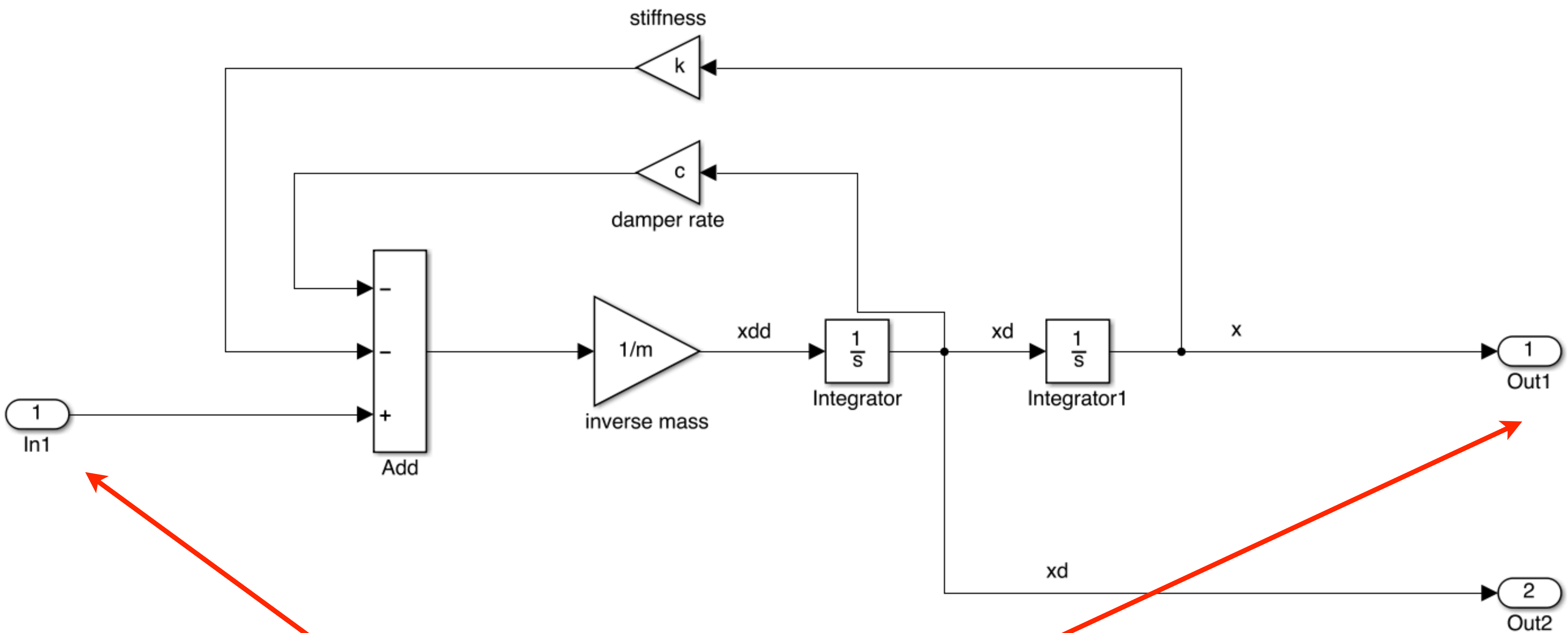
SUBSYSTEM MODELS
Creating A Subsystem Block Model

Produces the following:



Creating A Subsystem Block Model

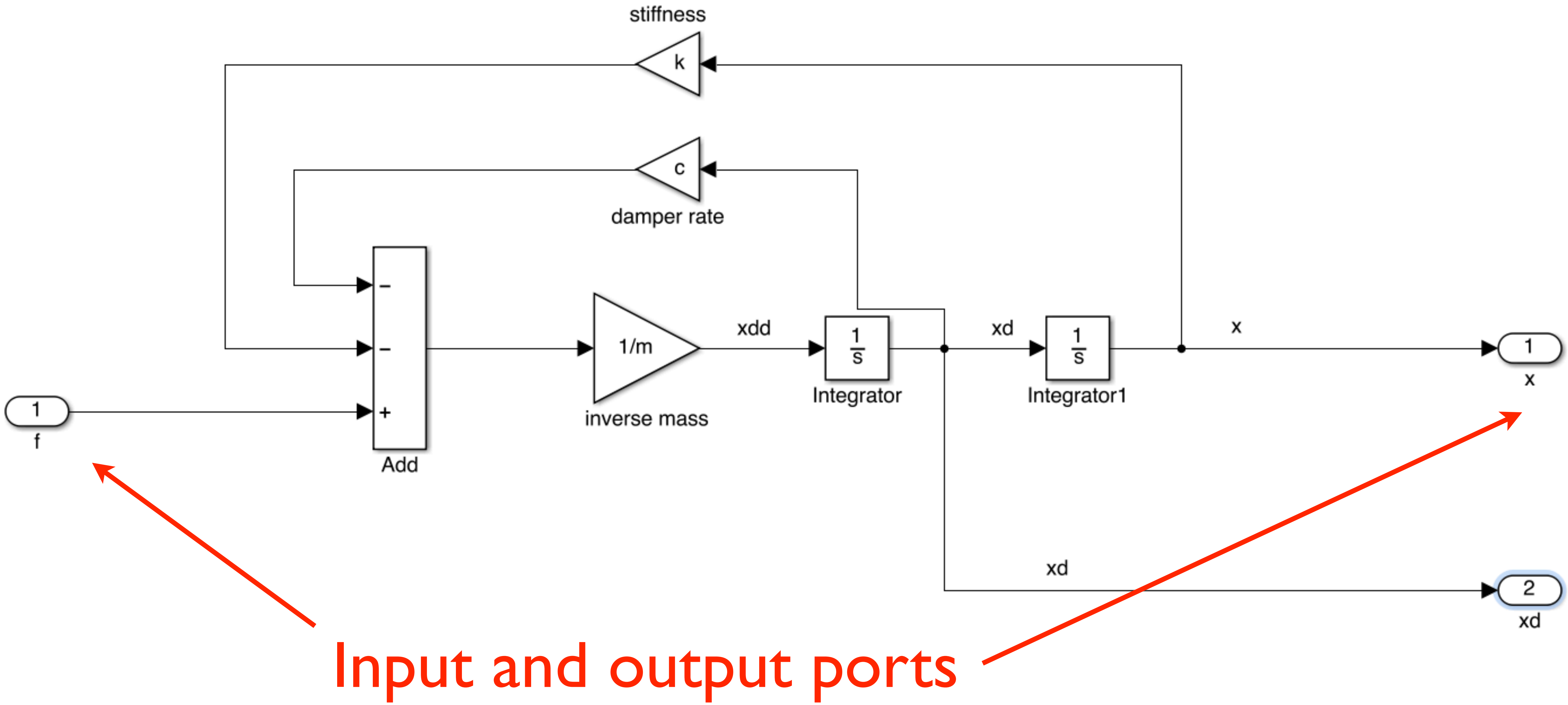
Clicking inside the subsystem block:



Input and output ports

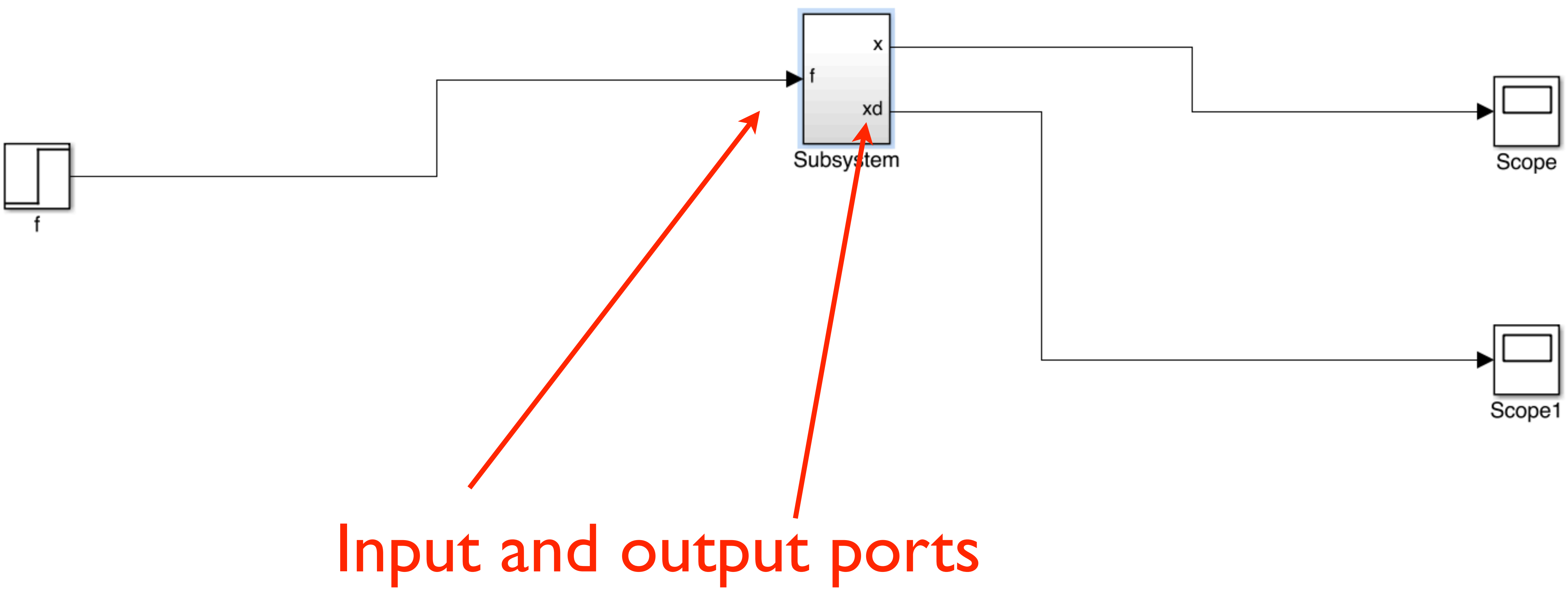
SUBSYSTEM MODELS
Creating A Subsystem Block Model

Relabel these input and output ports to variable names:



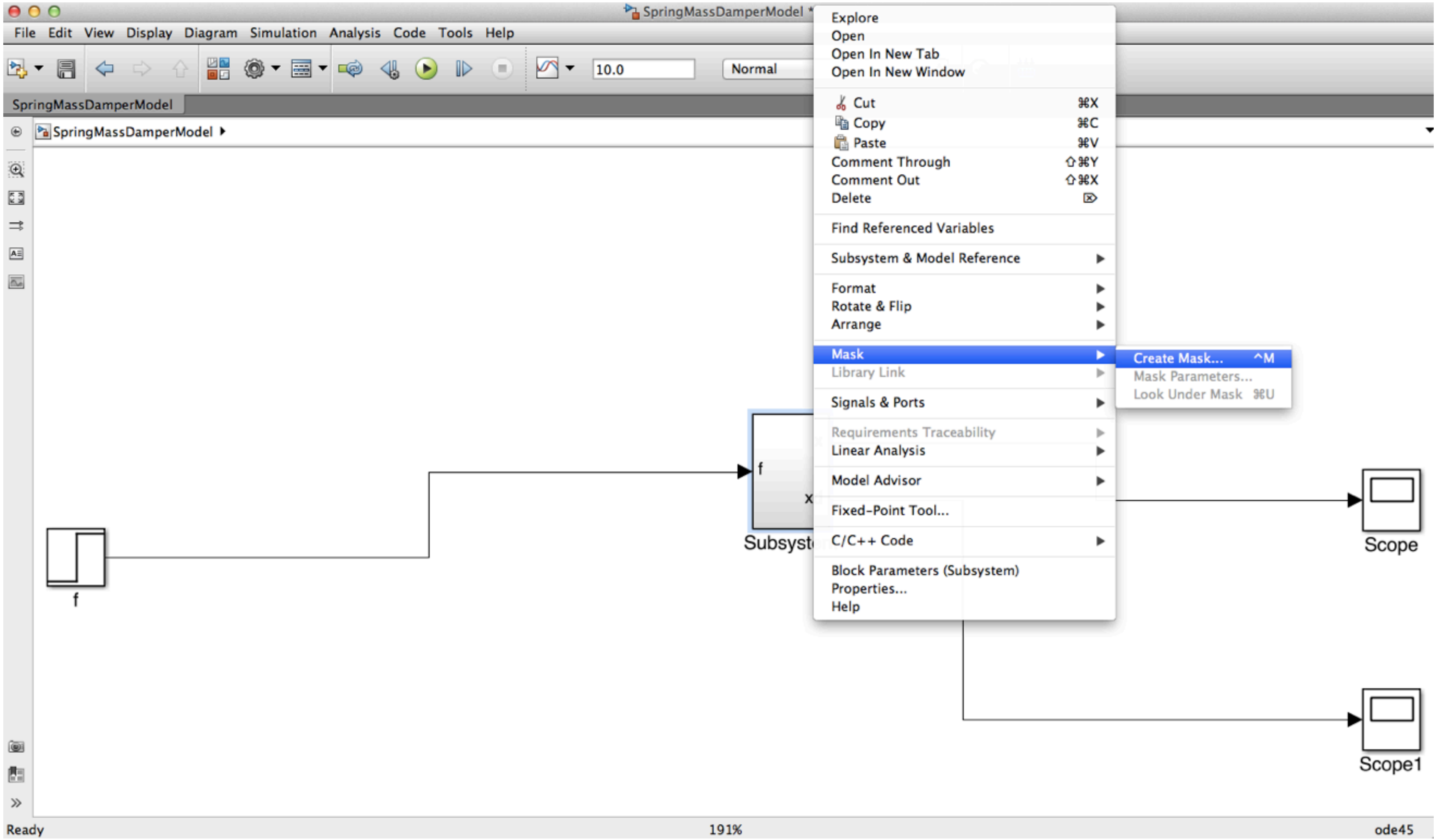
Creating A Subsystem Block Model

Produces the following subsystem block:

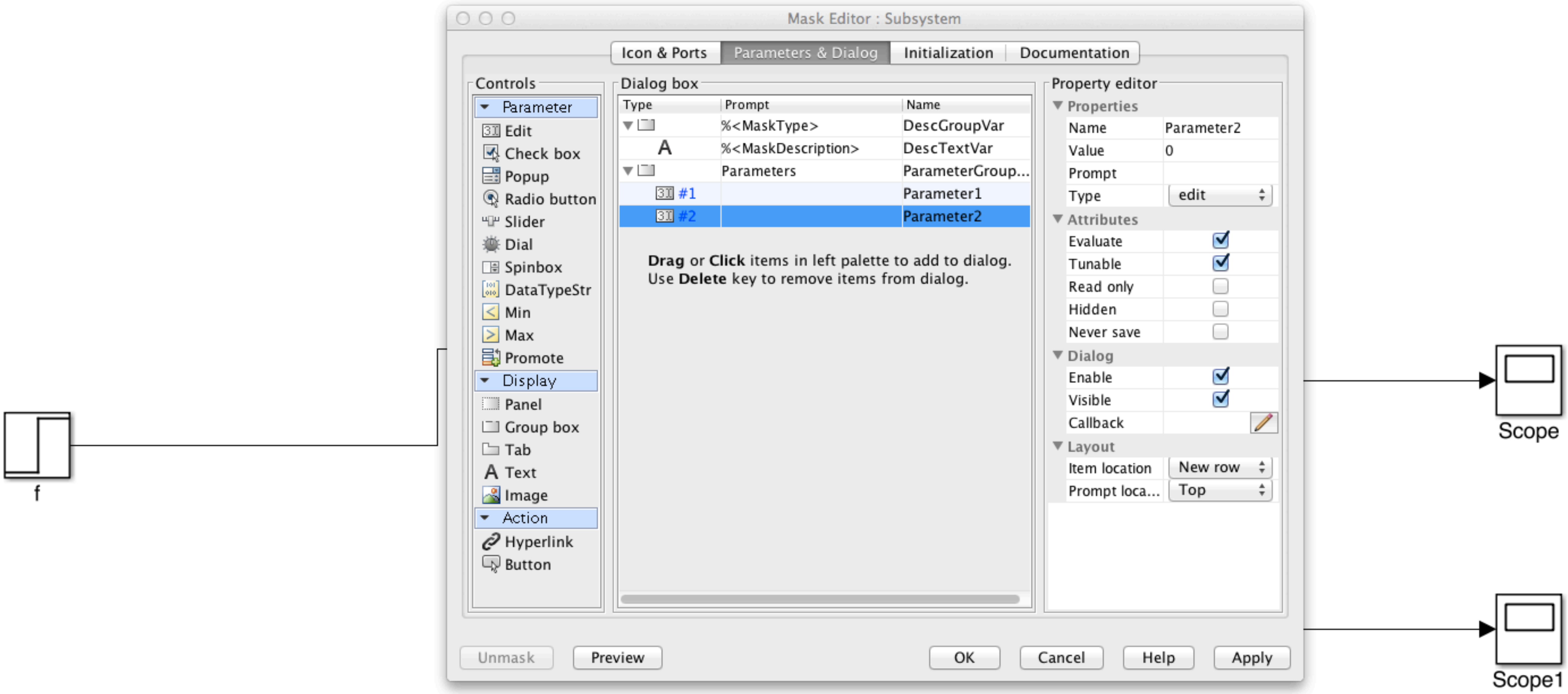


SUBSYSTEM MODELS
Masking A Subsystem Block Model

Now will mask this subsystem:

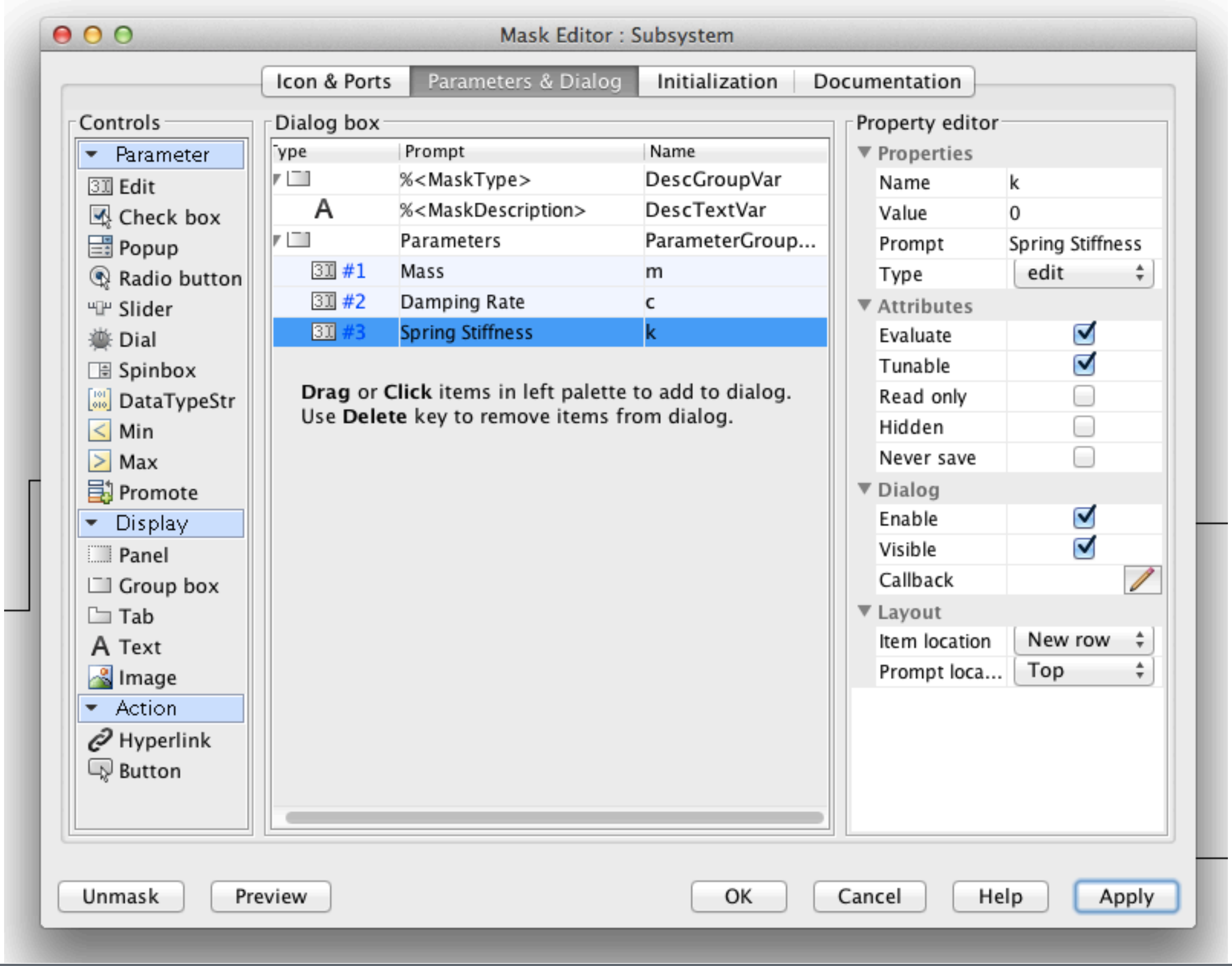


Click “edit” to add the parameters:



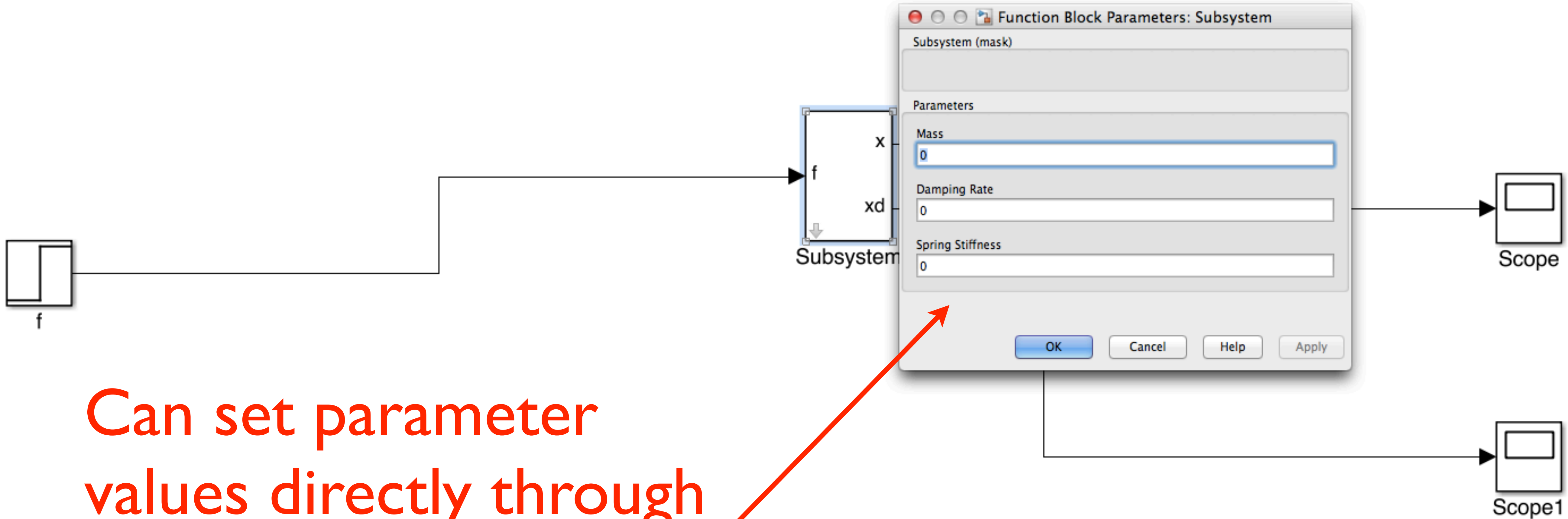
SUBSYSTEM MODELS
Masking A Subsystem Block Model

Now define a label and specify the variable name for each parameter:



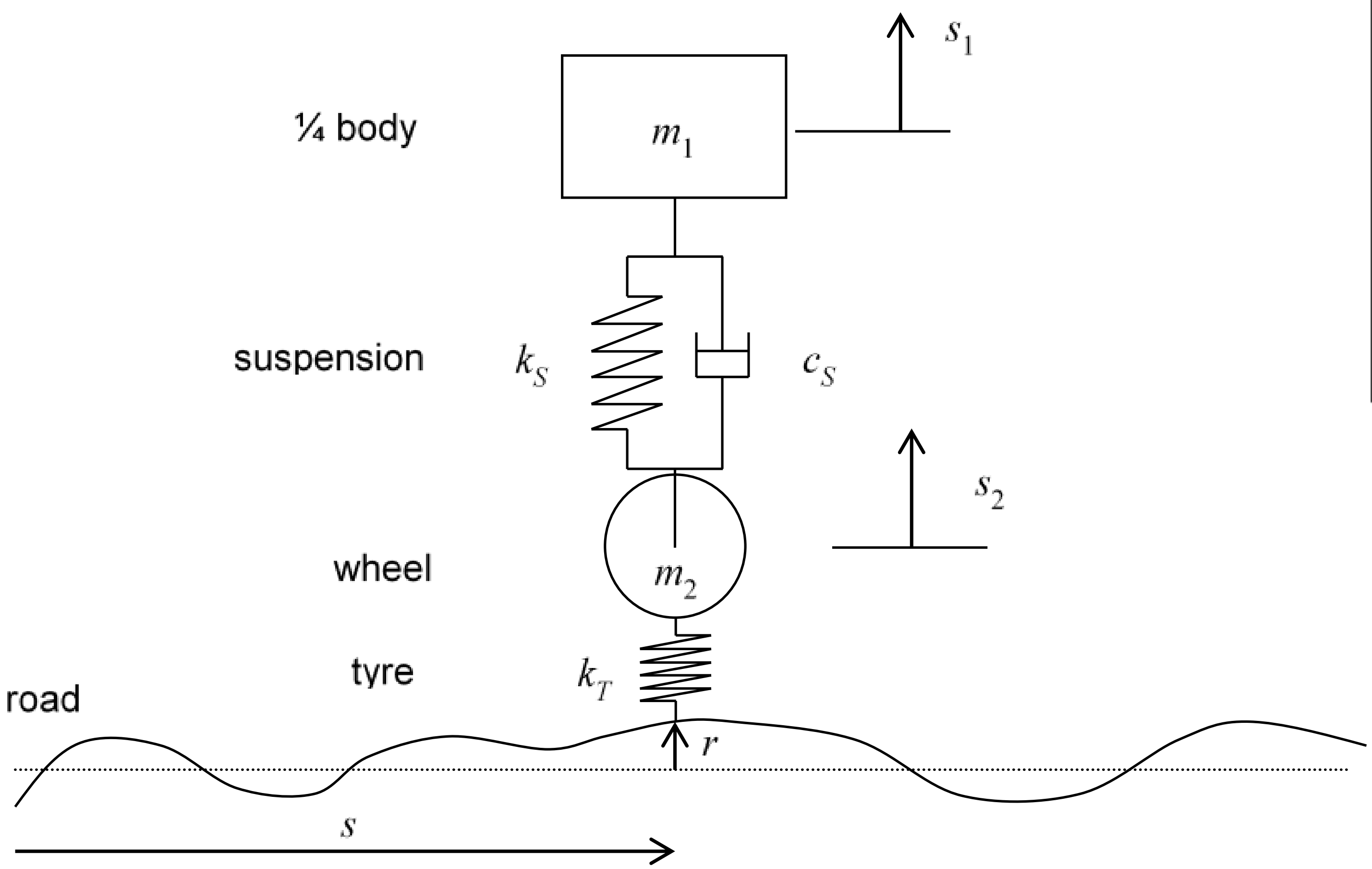
SUBSYSTEM MODELS
Masking A Subsystem Block Model

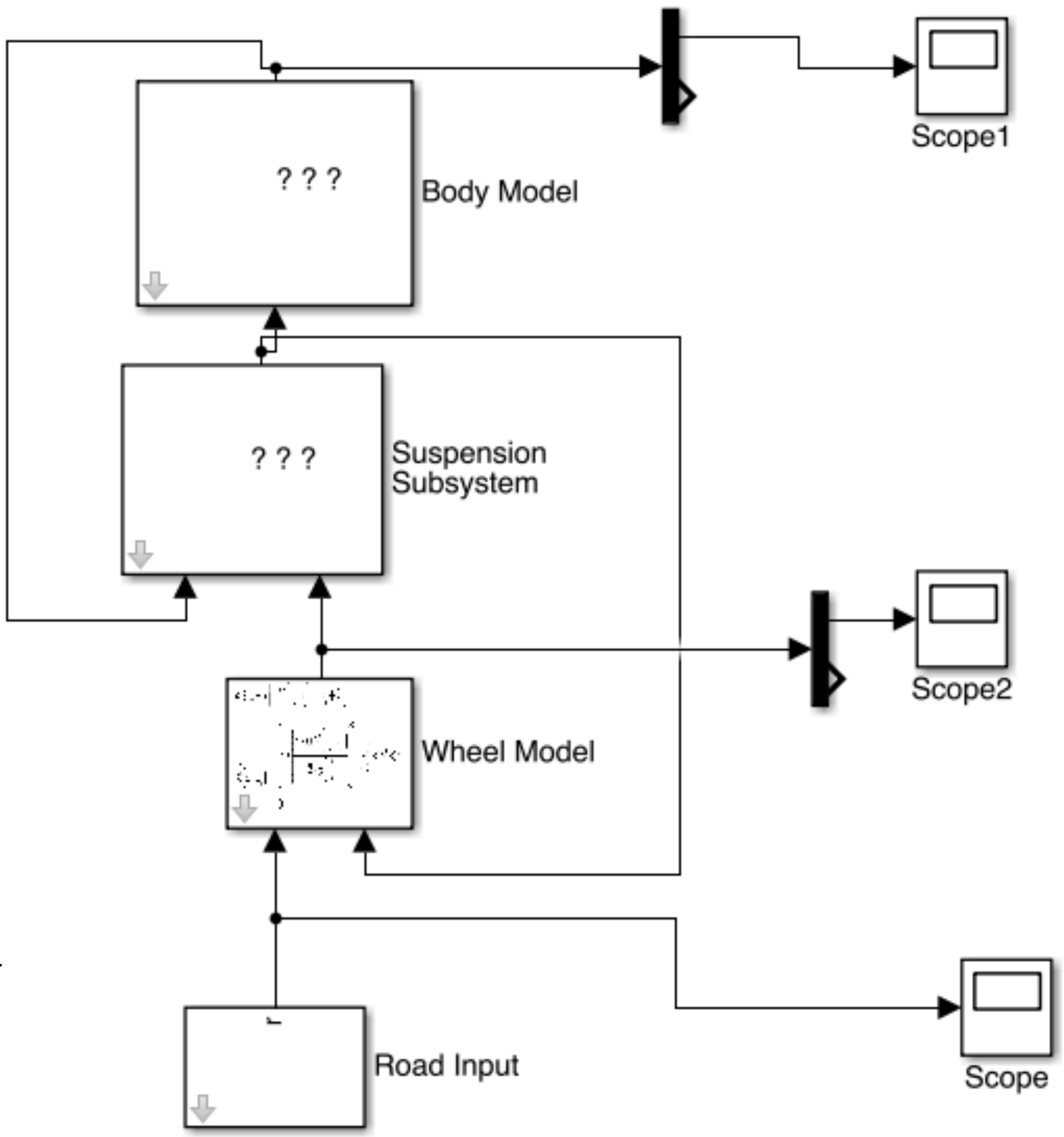
Double clicking on the subsystem block produces following menu:



Can set parameter values directly through this menu, rather than needing to use Matlab

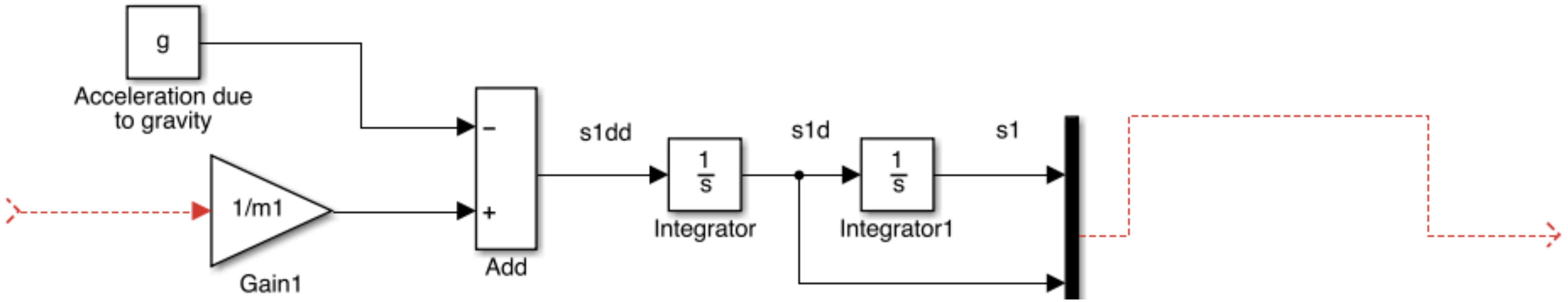
SUBSYSTEM MODELS
Recap Of 1/4 Car Model

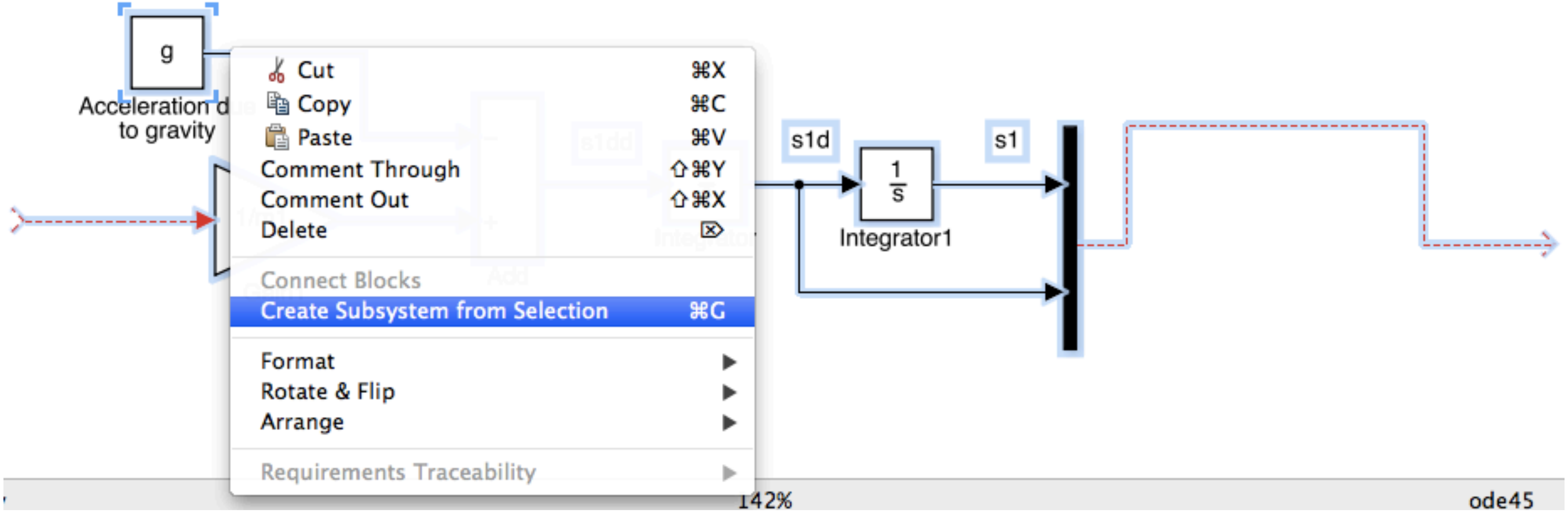




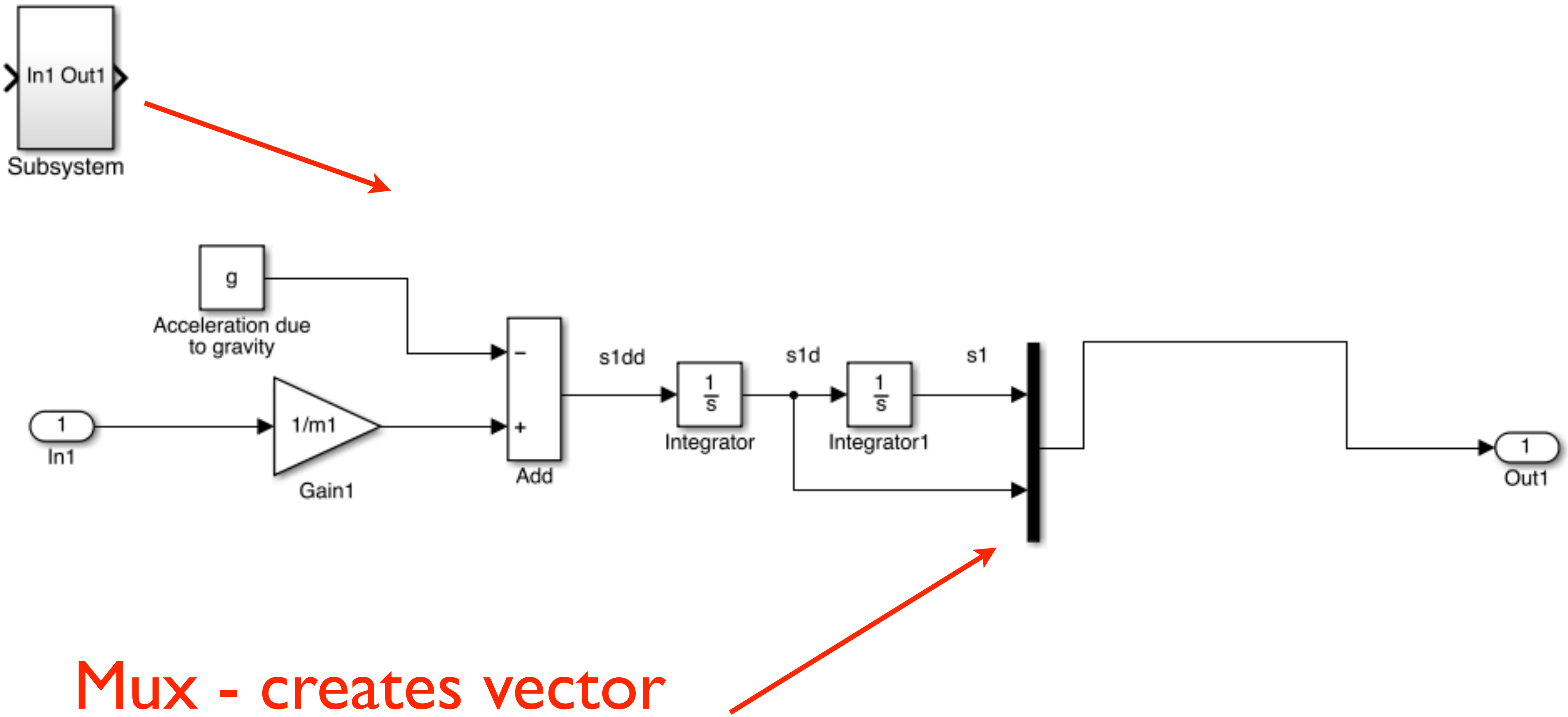
SUBSYSTEM MODELS

Building The Body Model Subsystem



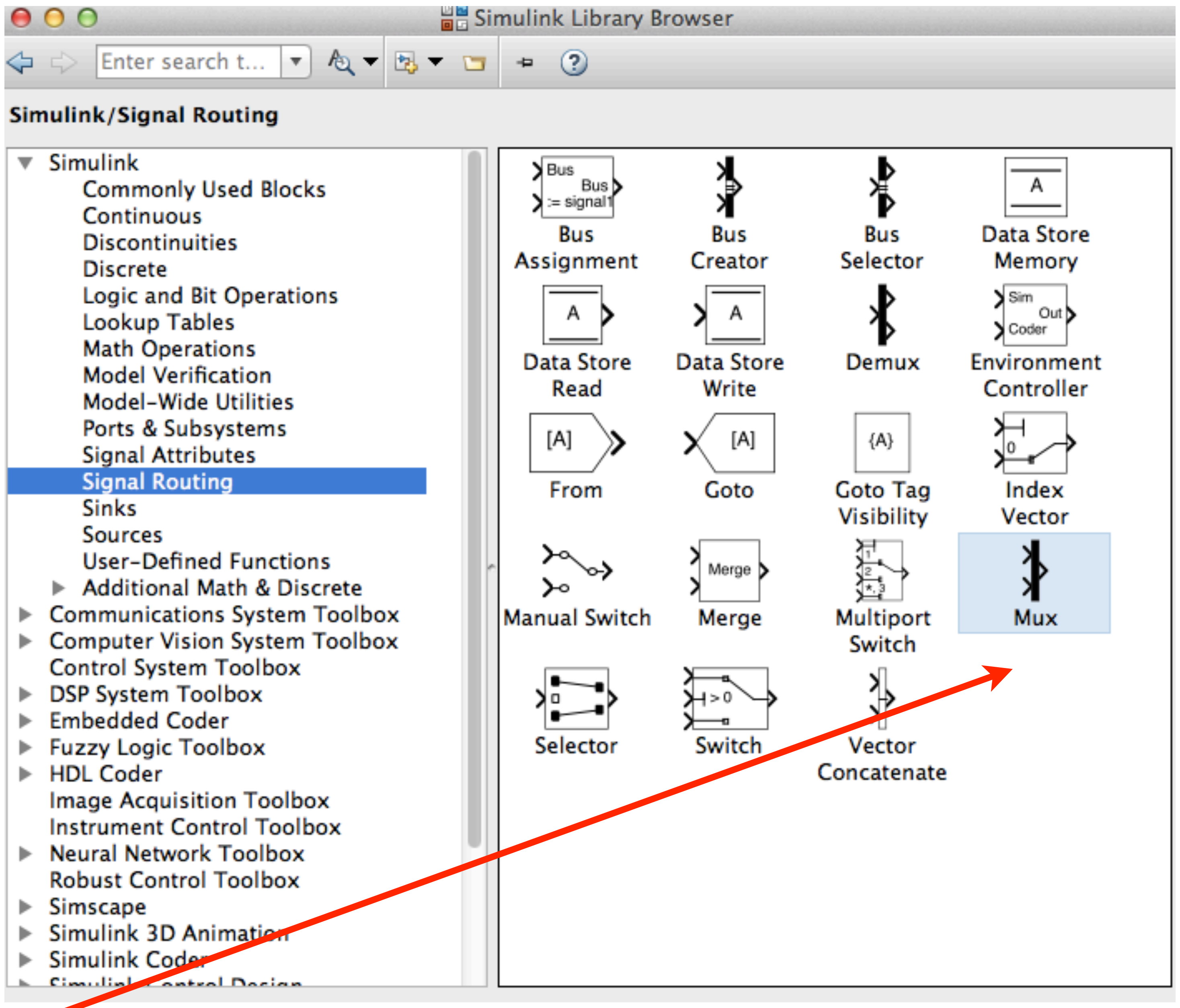


SUBSYSTEM MODELS
Building The Body Model Subsystem



Mux - creates vector
signal from multiple inputs

SUBSYSTEM MODELS
Building The Body Model Subsystem



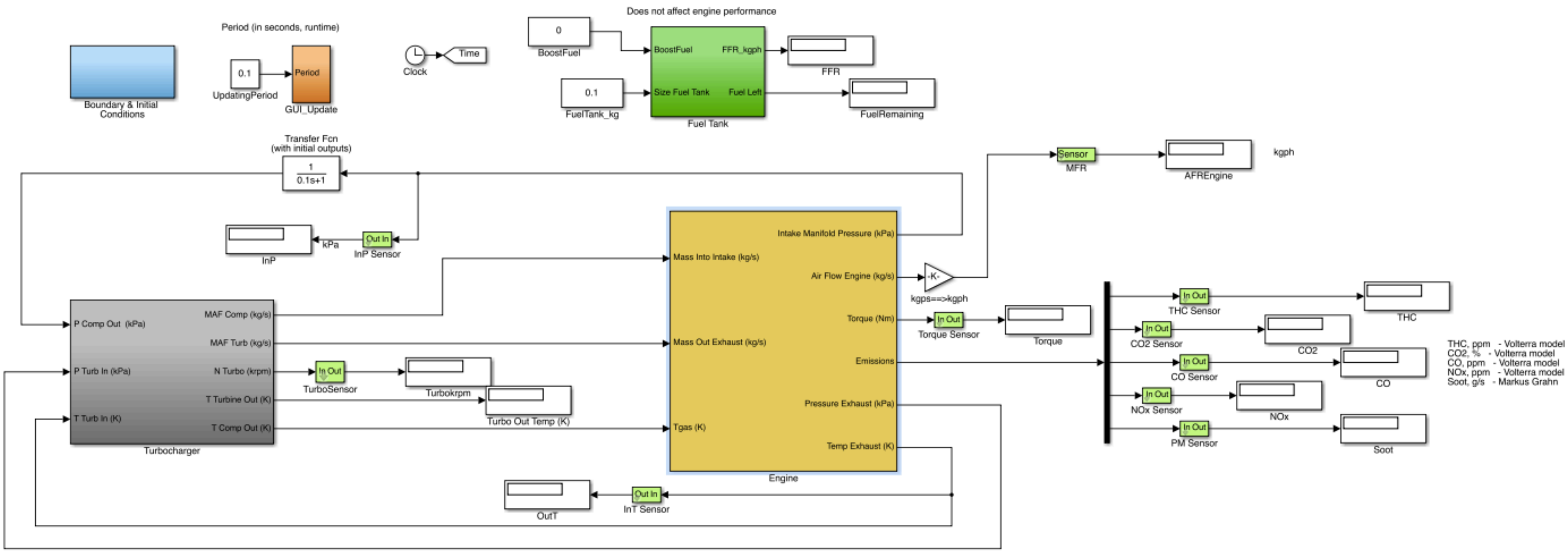
Mux - creates vector signal from multiple inputs

Example Model From Vehicle Engineering

- Following slides show just a few of the blocks and subsystems from an engine model created by Dr. R. Burke

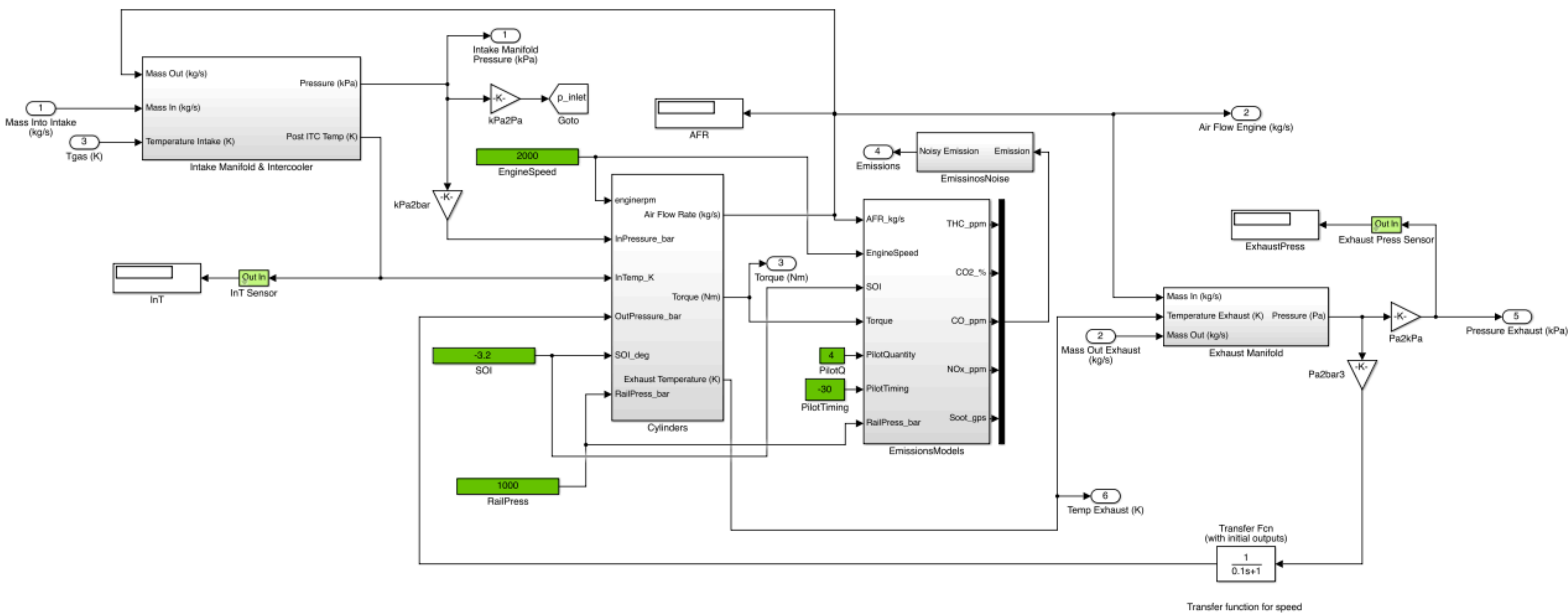
SIMULINK'S ADVANCED FEATURES
Example Model From Vehicle Engineering

Engine control system model (Dr. R. Burke)



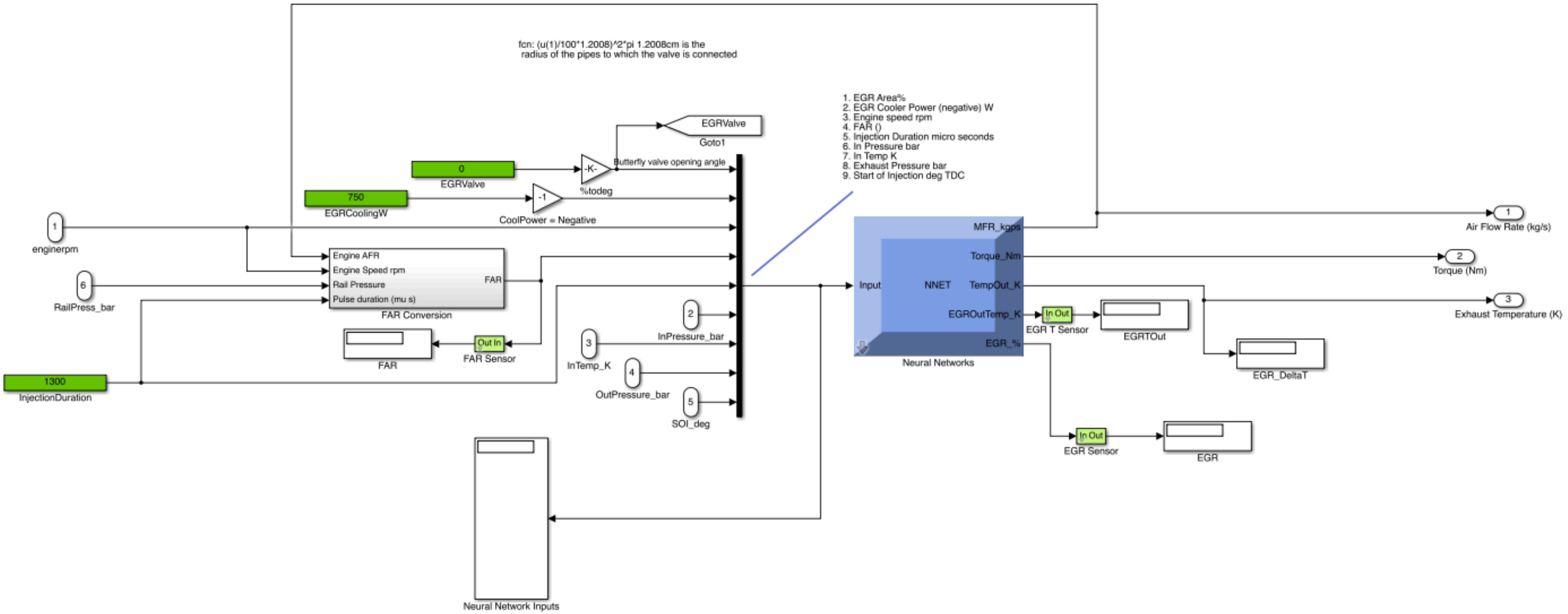
SIMULINK'S ADVANCED FEATURES
Example Model From Vehicle Engineering

Engine subsystem

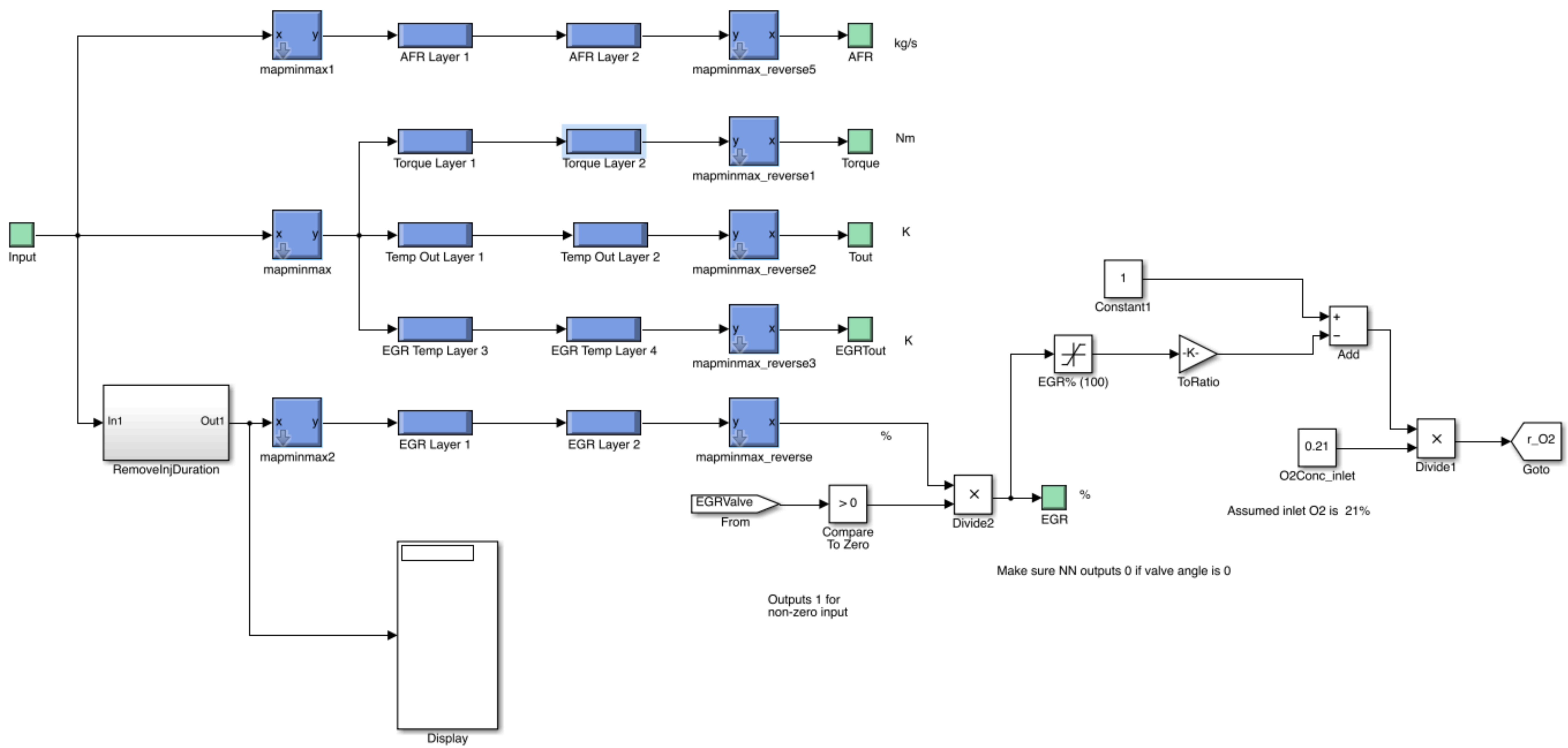


Example Model From Vehicle Engineering

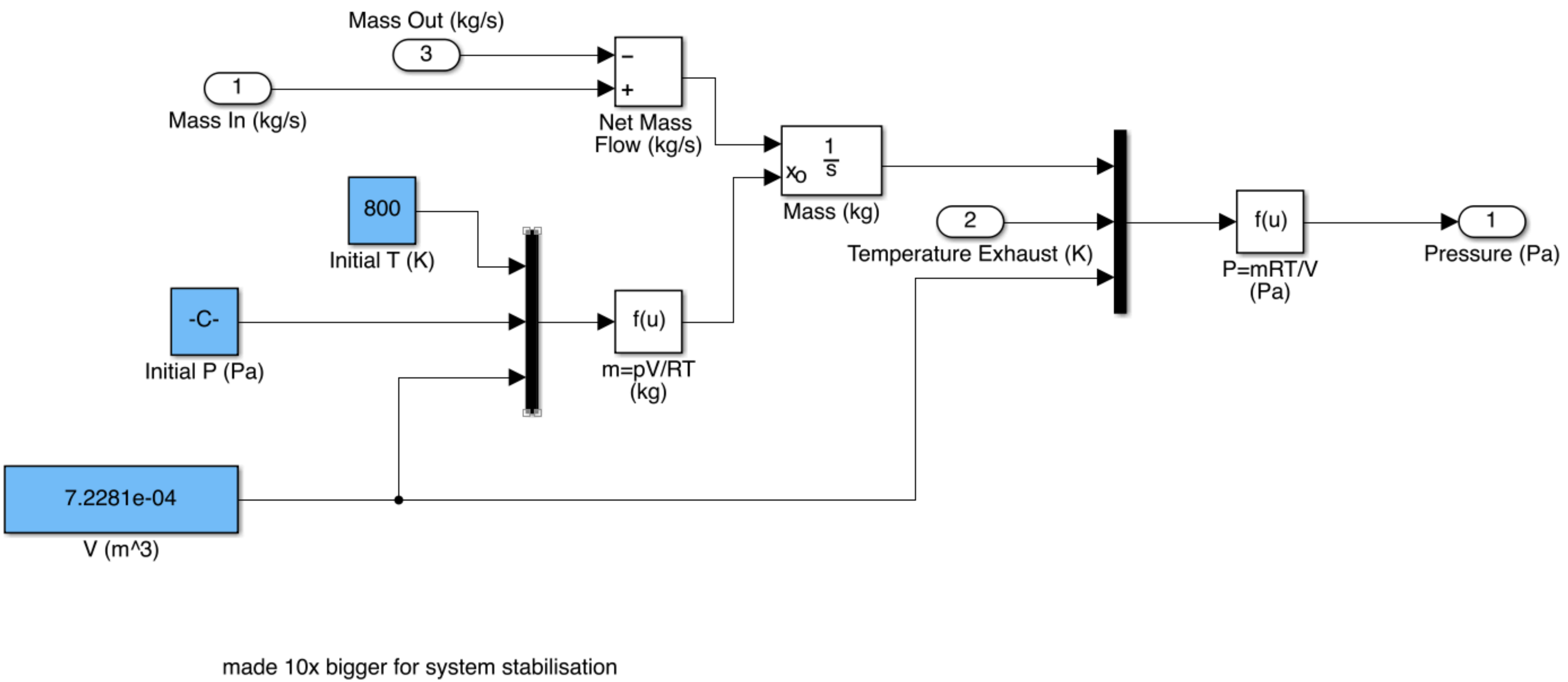
Cylinders subsystem



Neural networks subsystem



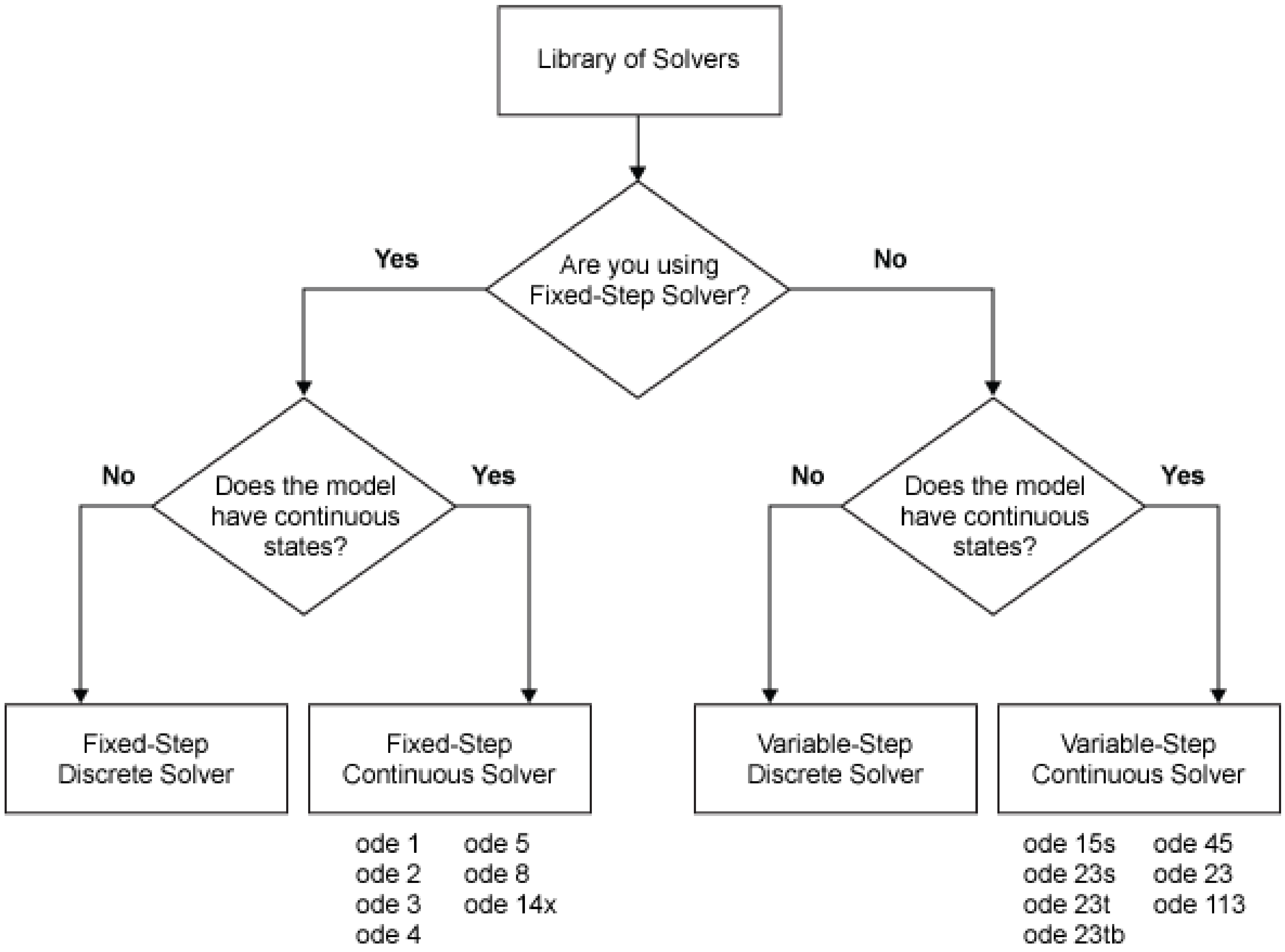
Exhaust manifold



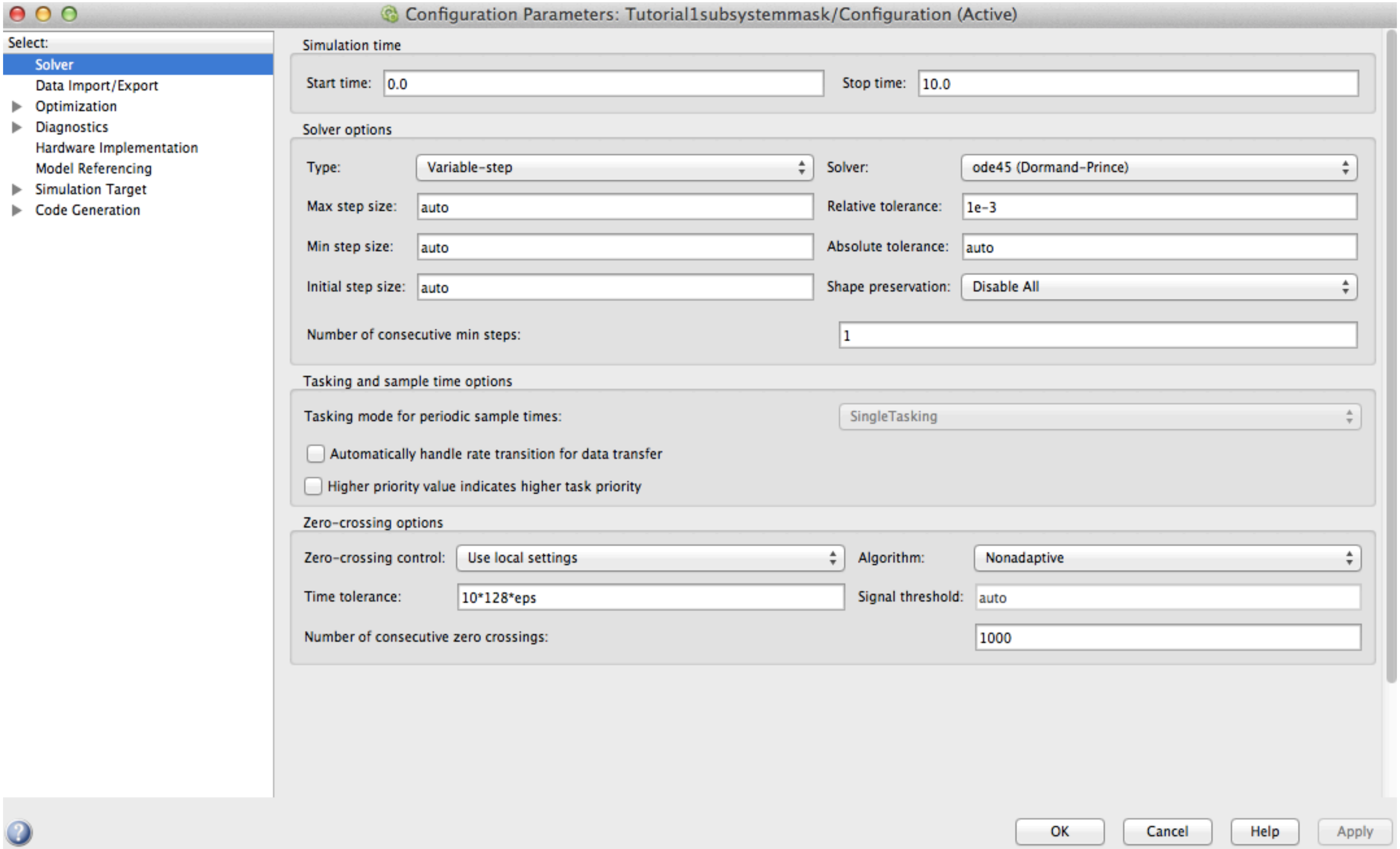
SIMULINK'S SOLVERS

- What are the solvers and their options?

SIMULINK'S SOLVERS
The Options



SIMULINK'S SOLVERS
The Options



SIMULINK'S SOLVERS
Runge-Kutta 4 Scheme

Multi-stage scheme - an explicit method, therefore has stability restrictions

To solve the following types of ordinary differential equations:

$$\frac{dy}{dt} = f(y,t), \quad y(t_0) = y_0$$

For one time step:

$$t_{n+1} = t_n + \Delta t$$

The solution at that next time point is the following weighted sum:

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

These values of k are defined on the following slide:

SIMULINK'S SOLVERS
Runge-Kutta 4 Scheme

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

The solution at that next time point is the following weighted sum:

$$k_1 = f(y_n, t_n)$$
$$k_2 = f(y_n + \Delta t\frac{k_1}{2}, t_n + \frac{\Delta t}{2})$$
$$k_3 = f(y_n + \Delta t\frac{k_2}{2}, t_n + \frac{\Delta t}{2})$$
$$k_4 = f(y_n + \Delta t k_3, t_n + \Delta t)$$

This is a 4th order method

SIMULINK'S SOLVERS
Runge-Kutta 4 Scheme

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

The solution at that next time point is the following weighted sum:

Estimates of the slope

$$k_1 = f(y_n, t_n)$$

Start point of step

$$k_2 = f(y_n + \Delta t\frac{k_1}{2}, t_n + \frac{\Delta t}{2})$$

Mid-point estimated from step 1

$$k_3 = f(y_n + \Delta t\frac{k_2}{2}, t_n + \frac{\Delta t}{2})$$

Updated estimate of mid-point

$$k_4 = f(y_n + \Delta t k_3, t_n + \Delta t)$$

End point of step using updated y estimate

SIMULINK'S SOLVERS
Ode45 (Dormand-Prince)

This is the default solver in Simulink and Matlab - good for many problems

A variant of the Runge-Kutta scheme described previously

This version computes a 4th and 5th order accurate solution - the difference between the solutions indicates the size of the error

The time step can then be adapted to suit this error size for each step of the solution

This kind of adaptive time stepping provides a good balance of accuracy and computational cost

More information available at:

https://uk.mathworks.com/help/matlab/ref/ode45.html

https://uk.mathworks.com/help/simulink/gui/solver.html

SIMULINK'S SOLVERS
Further Reading

Use Auto Solver to Select A Solver

https://uk.mathworks.com/help/simulink/ug/use-auto-solver-to-select-a-solver-1.html

Checking Accuracy Through Multiple Solutions

https://uk.mathworks.com/help/simulink/ug/improving-simulation-accuracy.html