1 Introduction

Simulation of dynamic systems has been proven to be immensely useful in system modeling and controller design. Simulink® is a add-on to MATLAB which facilitates the visual programming of a dynamic system. This brief tutorial on Simulink, goes through the steps of a simple design example and explains some basic useful concepts.

2 Simulating a Model

1 Simulink can be invoked from the MATLAB command window by typing *simulink* or by clicking the Simulink icon on the MATLAB Toolbar.

2 When you open the Simulink browser you will see an interface which presents several blocksets on the left and libraries of the current blockset on the right side.

3 Start a new Simulink model using File > New > Model

4 The example which we are going to handle in this tutorial is the Mass/Spring/Damper system. In Figure 1, $z(t)$ denotes the hori-

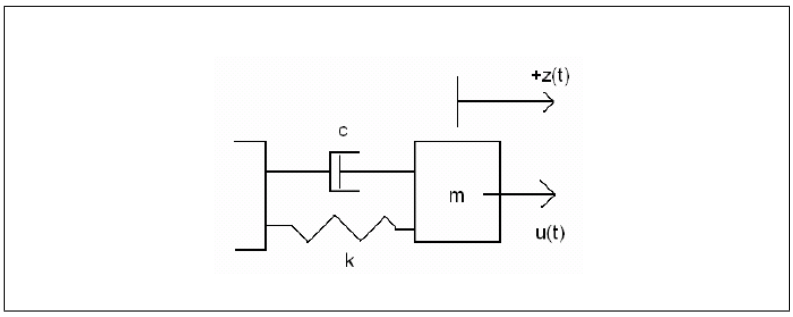

Figure 1: Mass/Spring/Damper System

zontal distance, c and k are the damper and spring constants respectively, m is the mass and $u(t)$ denotes the force exerted on the mass. Choosing the state vector as $\bar{x} = (z\dot{z})^T$ and output as $y = z$, the state space representation can be written as the following form

$$\dot{\bar{x}} = A\bar{x} + Bu$$
$$y = C\bar{x} + Du$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1/m \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}, D = 0.$$

Here let the constants $k = 2$, $m = 1$, $c = 0.5$.

5 Double click on the *Continuous* library under *Simulink* blockset and open the library. Drag and drop the State-space block into your Simulink model window.

6 Double click on the State-space block to define the parameters as in Figure 2.

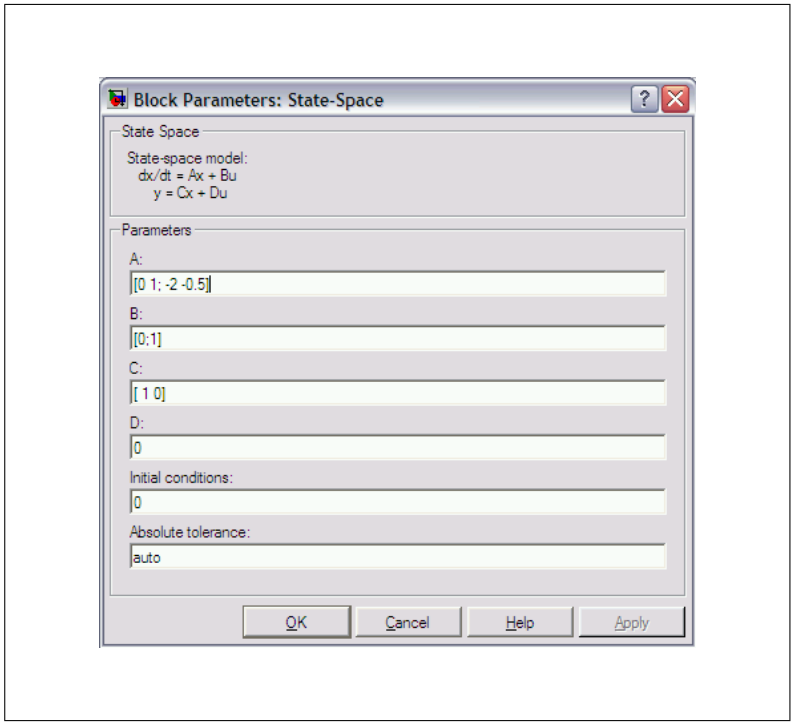

Figure 2: Setting State-space parameters

7 Pick Simulink > Sources > Step and Simulink > Sinks > Scope and connect them into the input and output of the system. The system should look like Figure 3.

8 Simulate this system by pressing the run button on the Simulink toolbar, clicking Simulation > Start, using *sim* command from the command window or pressing Ctrl+T.

9 Check the response of the system from the Scope. Double click on the scope block to open it up. Press the autoscale button, which appears in te shape of binoculars in the toolbar, to make the graph look better. You should observe the graph in Figure 4.

10 Save this model as *mass_spring_damper_model.mdl*

11 This way of system simulation looks quite straightforward. However if you are making a design and need to try several parameters it could be inconvenient to click on the state-space block and change the parameters every time. That's why we will now explain the use

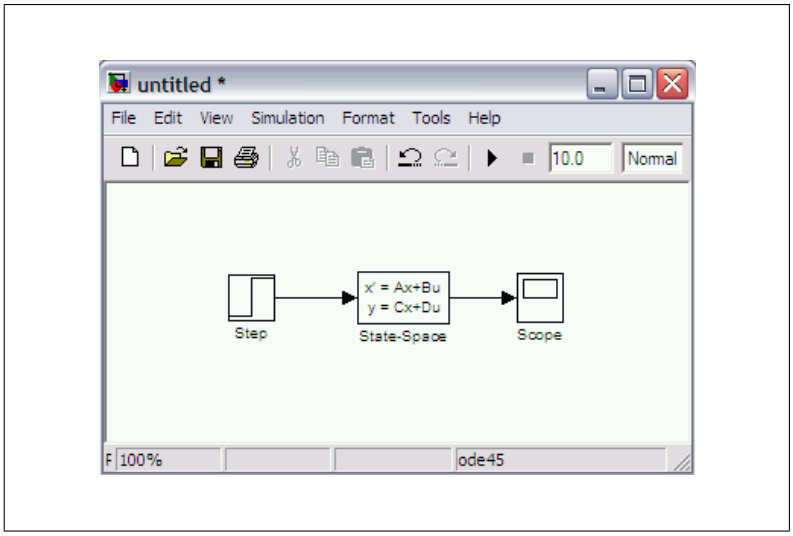

Figure 3: Applying step input

Table 1: Variables

Variable Name	Class	Source
sim_u	struct array	Created by To Workspace block
sim_y	struct array	Created by To Workspace1 block
tout	double array	Created by Simulink automatically

of m-file scripts in Simulink.

Now start a new m-file and define the A, B, C, D matrices in the file as in Figure 5. Now go back to the State-space block and change the parameters as in Figure 6. As you change the parameters in the m-file, the parameters in the state-space block will also change.

12 You may also need to import the input and output of the system to MATLAB. Pick the *ToWorkspace* block under Simulink > Sinks and make the changes shown in Figure 7. Now connect these to the system as shown in Figure 8. When this system is run, three variables shown in Table 1 will be created and exported to the workspace.

13 Going back to the system representation part, we may change the way to represent the system. With the spring and damper constants and mass defined above, we can derive the transfer function of the system to be

$$\frac{Y(s)}{U(s)} = \frac{1}{s^2 + 0.5s + 2}.$$

From this transfer function, it can easily be inferred that the system can be simulated by either a single transfer function block or a combination of several gain and integrator blocks as shown in Figure

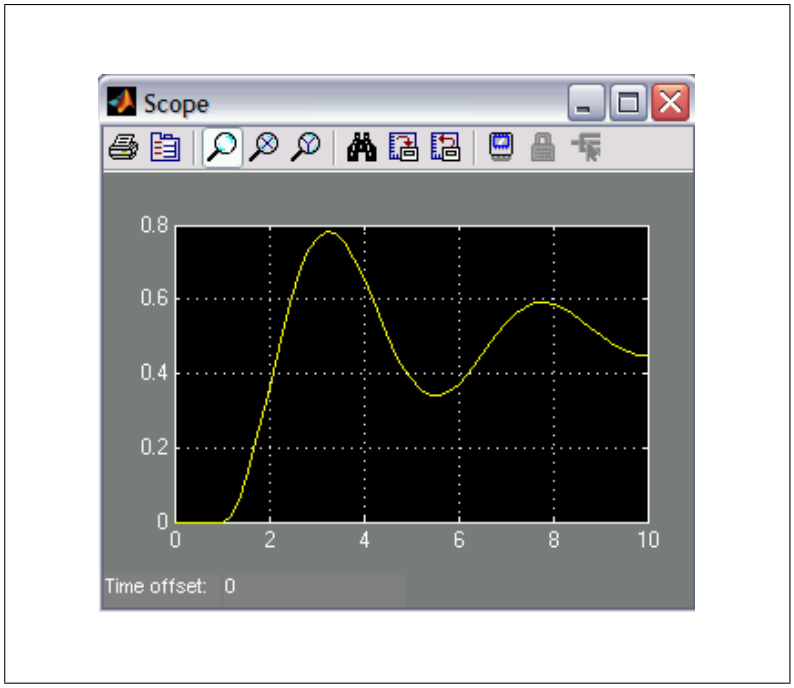

Figure 4: Step response

9. The above model can be constructed by entering the appropriate parameters in the $Transferfunction$ block under Simulink > Continuous. The second model needs the $Integrator$ block under Simulink > Continuous, Sum and $Gain$ blocks under Simulink > Math Operation. The number of signs on the Sum block may be changed by double clicking on it. The value of the is adjusted in the same manner too. After setting up the two models you should observe the same step response as in Figure 9.

14 The use of representing the system by integrators can be seen when one needs to deal with the nonideal characteristics of the systems. Say, the mass/spring/damper system defined above has some nonlinear elements. In this case finding an analytic solution can be very hard and one can use the facility of simulation of nonlinear elements in Simulink.

After replacing the transfer function block with a combination of integrators the block diagram may look a little complicated. In order to make it look more simple one may use the $sub - system$ facility. Select the the element you would like to include into the subsystem and right click on the selected part as in Figure 10. Choose $CreateSubsystemfromthelist$. The result should look like Figure 11.

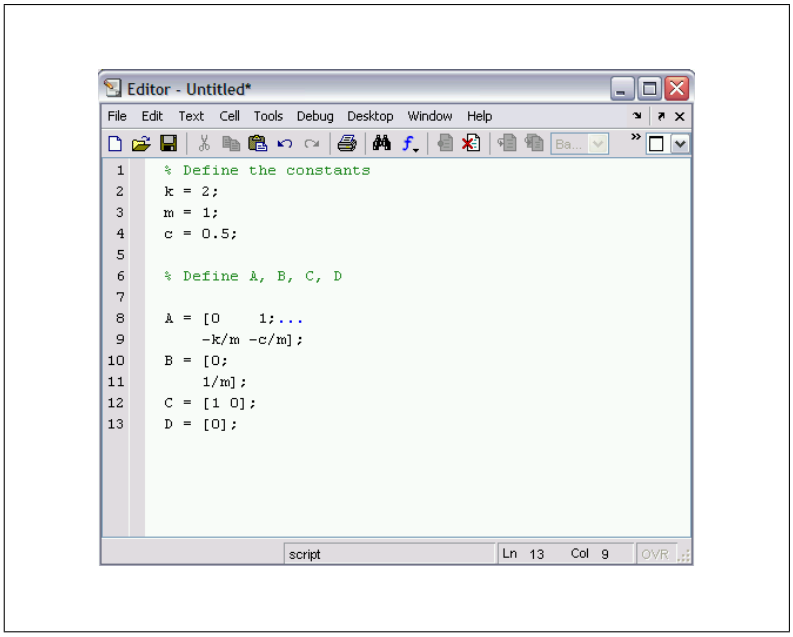

Figure 5: M-file defining the system parameters

3 Design Example

Now lets add a digital controller to this continuous time system and apply feedback in order to get a system with $\zeta = 0.5$ and $t_s = 2$ sec.

15 To construct this system you need the following blocks: Simulink > Discrete > Discrete Transfer Function, Simulink > Discrete > Zero Order Hold and the sublock of the plant that we have previously built. With a unity negative feedback loop the system looks as in Figure 12.

16 From above specifications the system poles are defined by $\zeta = 0.5$ and $\omega_n = 4$. It is obvious that a proportional controller, $C(s) = K$, cannot satisfy these. A wise choice could be to choose the controller in the form

$$C(s) = K\frac{s^2 + 0.5s + 2}{s^2 + as}.$$

The unknowns K and a are solved to be $K = 4$ and $a = 4$.

17 So far the design is in continuous time. Now we should choose a sampling interval, T, and discretize the controller. A rule of thumb is to choose T such that

$$T \approx \frac{2\pi}{10\omega_n}$$

So lets choose $T = 0.1$ and set it by double clicking on the $ZeroOrderHold$ block in Figure 12.

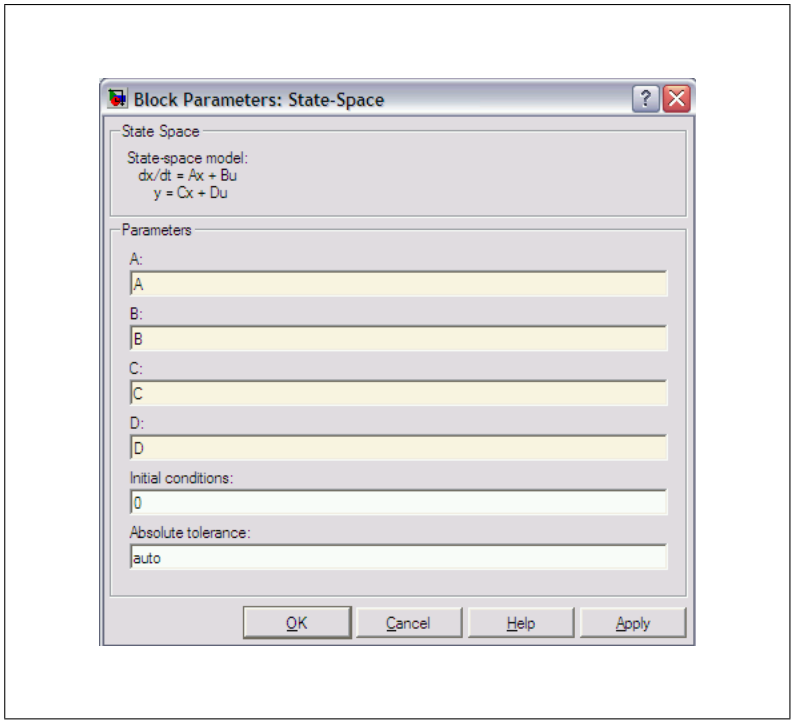

Figure 6: Getting system parameters from the m-file

18 One choice to discretize the controller by Tustin's Trapezoidal rule. In this case the digital controller $D(z)$ will look like

$$D(z) = 16\frac{412z^2 - 796z + 392}{480z^2 - 800z + 320}.$$

19 Set the parameters of the discrete time transfer function to these and change the sampling time.

20 By double clicking on the *Scope*, observe the output.

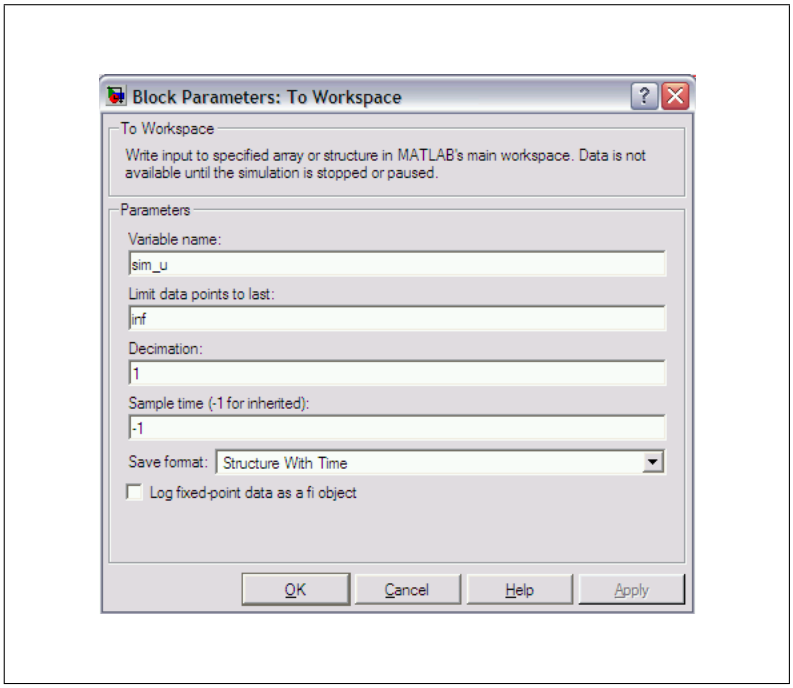

Figure 7: Parameters for the *ToWorkspace* block

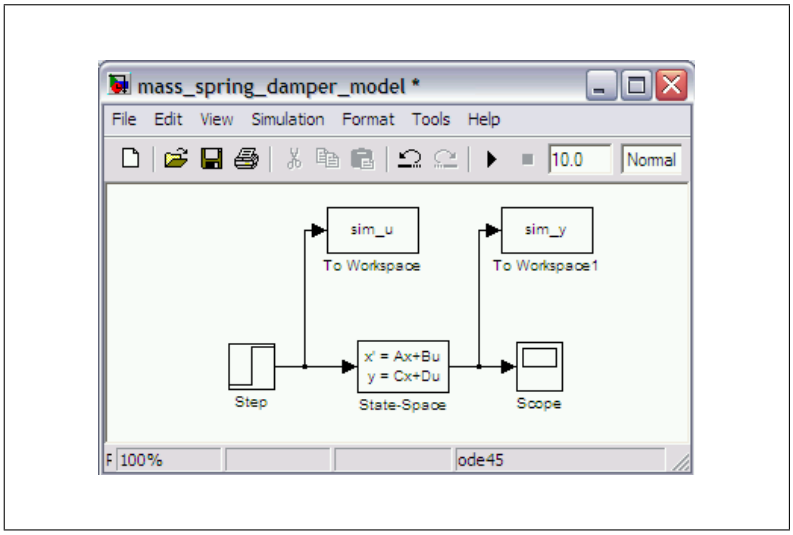

Figure 8: Simulink model with *ToWorkspace* blocks

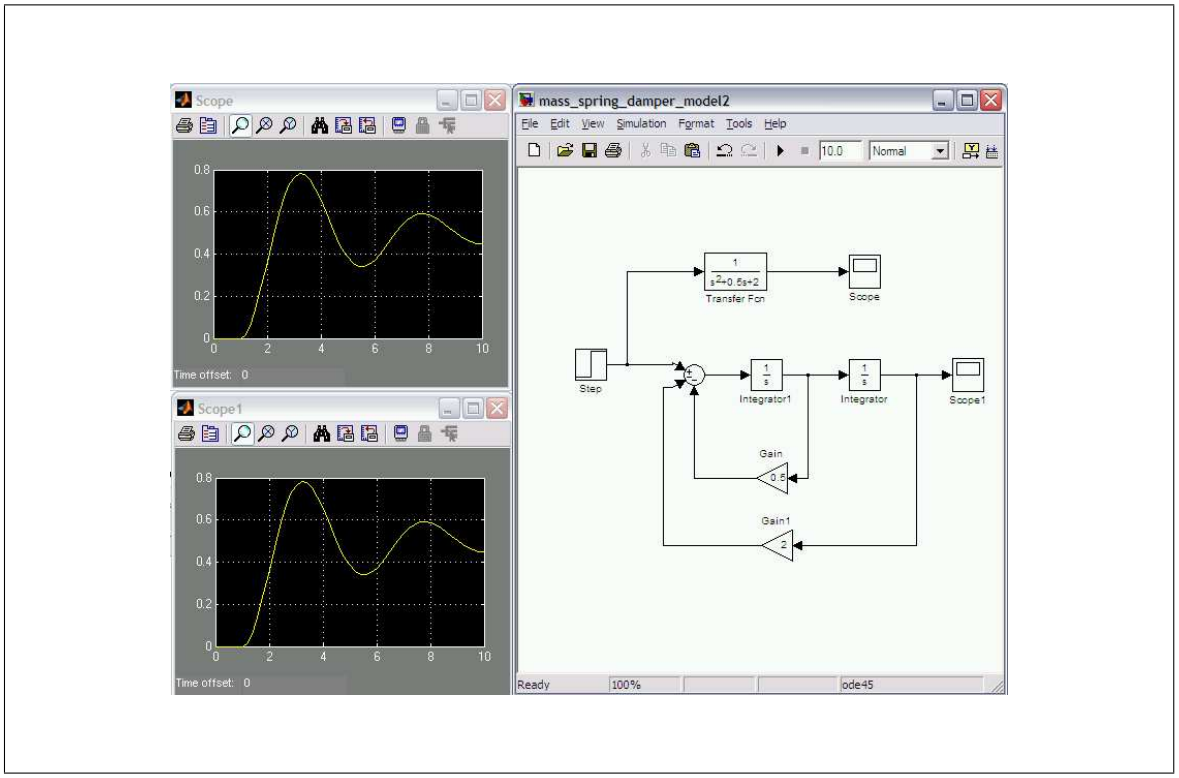

Figure 9: Different representations and corresponding step responses

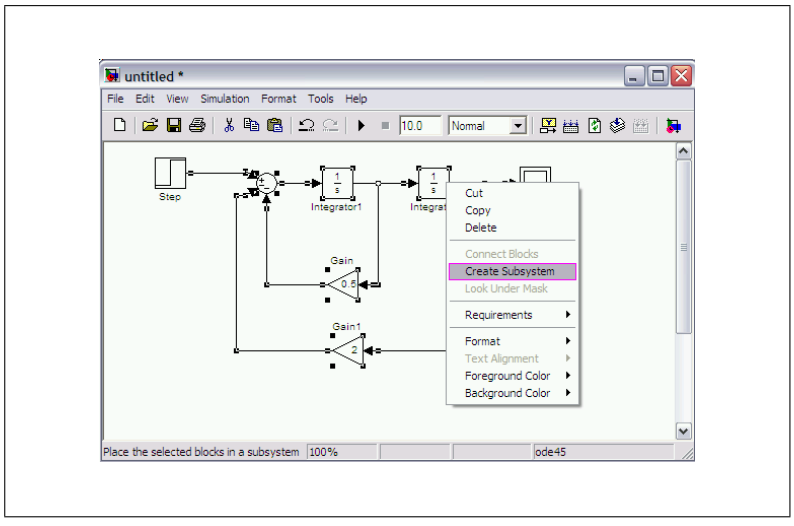

Figure 10: Creating Subsystem

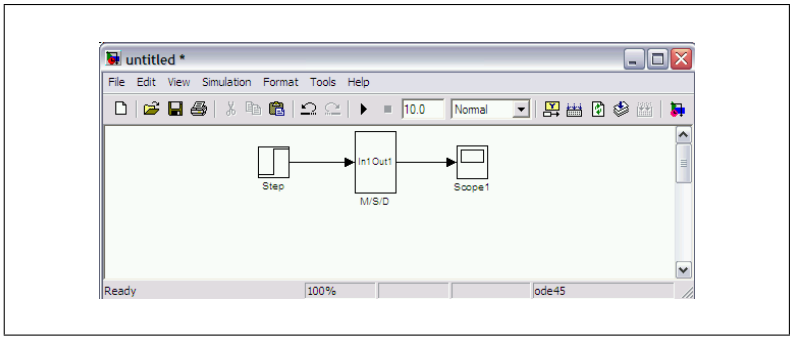

Figure 11: The Subsystem

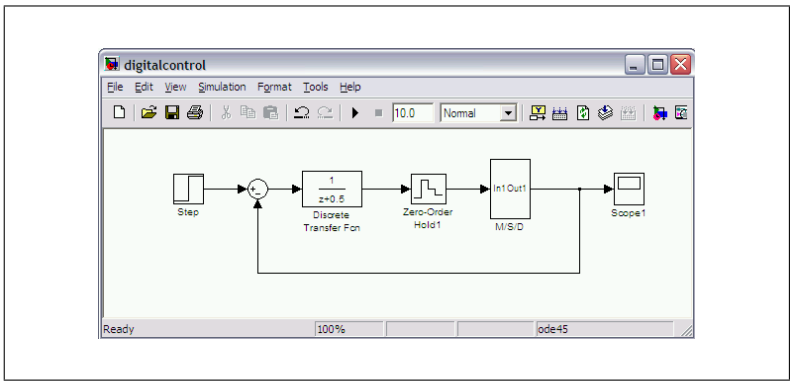

Figure 12: The Subsystem