

# Module 2

## Introduction to SIMULINK

Although the standard MATLAB package is useful for linear systems analysis, SIMULINK is far more useful for control system simulation. SIMULINK enables the rapid construction and simulation of control block diagrams. The goal of the tutorial is to introduce the use of SIMULINK for control system simulation. The version available at the time of writing of this textbook is SIMULINK 4, part of Release 12 (including MATLAB 6) from MATHWORKS. The version that you are using can be obtained by entering `ver` in the MATLAB Command Window.

The easiest way to learn how to use SIMULINK is to implement each step of the tutorial, rather than simply reading it. The basic steps to using SIMULINK are independent of the platform (PC, MAC, UNIX, LINUX...).

M2.1 Background

M2.2 Open-loop Simulations

M2.3 Closed-loop Simulations

M2.4 Developing Alternative Controller Icons

## M2.1 Background

The first step is to startup MATLAB on the machine you are using. In the Launch Pad window of the MATLAB desktop, select SIMULINK and then the SIMULINK Library Browser. A number of options are listed, as shown in Figure M2.1. Notice that Continuous has been highlighted; this will provide a list of continuous function blocks available. Selecting Continuous will provide the list of blocks shown in Figure M2.2. The ones that we often use are Transfer Fcn and State-Space.

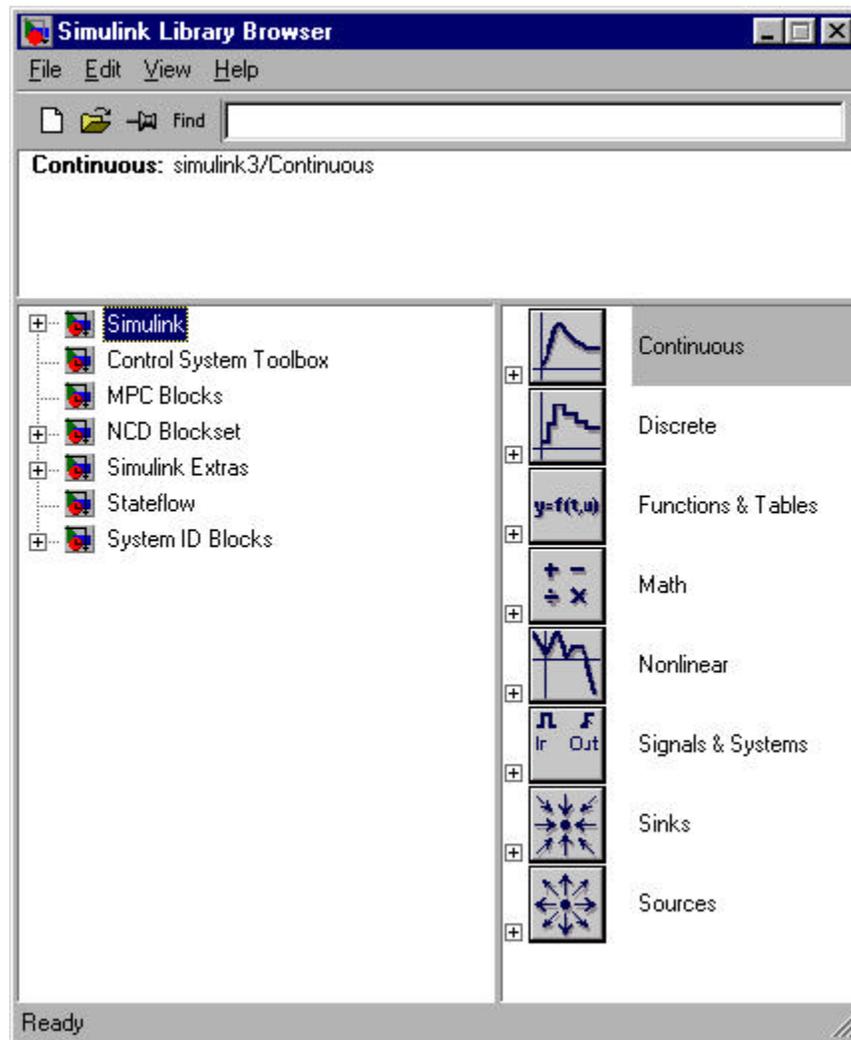
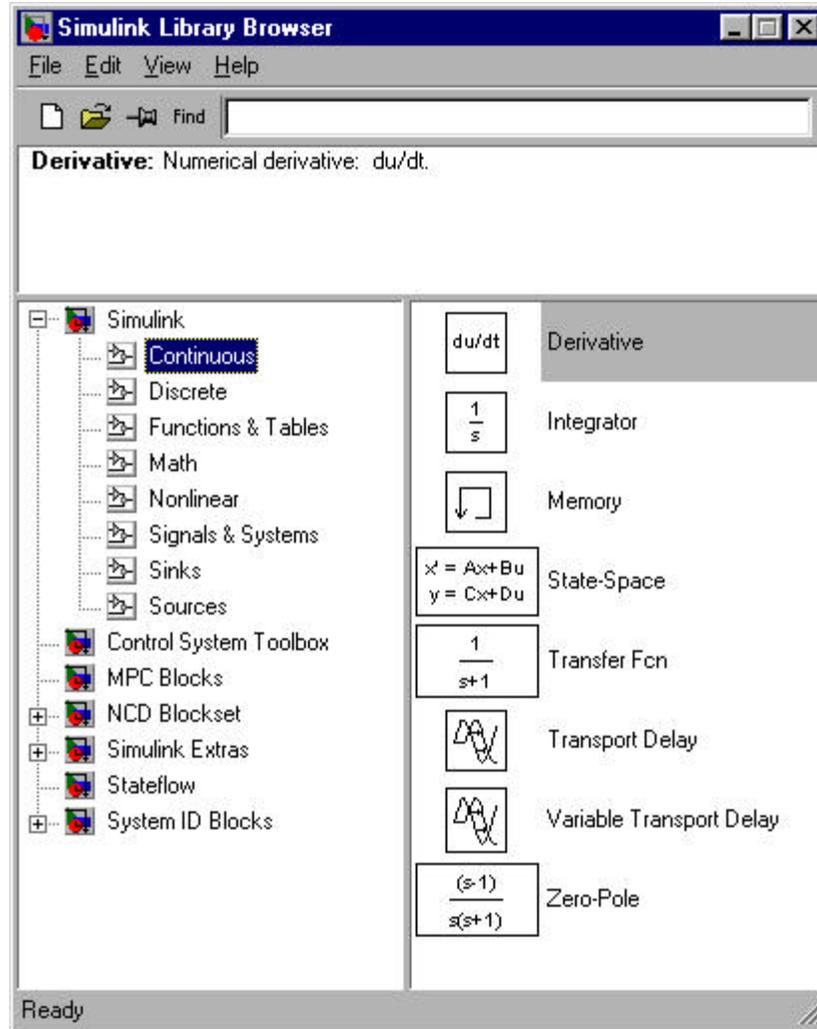
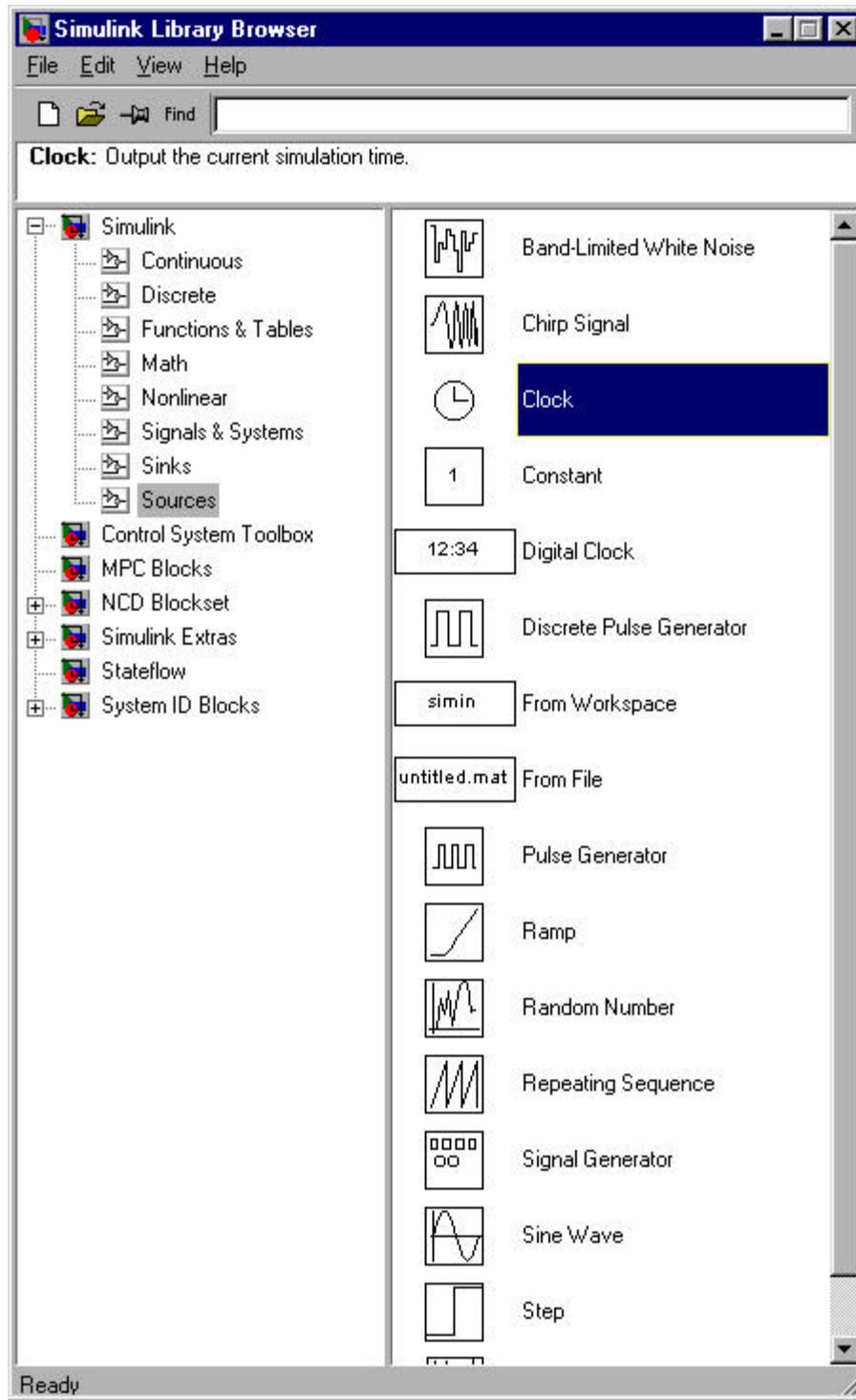


Figure M2.1 SIMULINK Library Browser.



**Figure M2.2** SIMULINK Continuous Blocks.

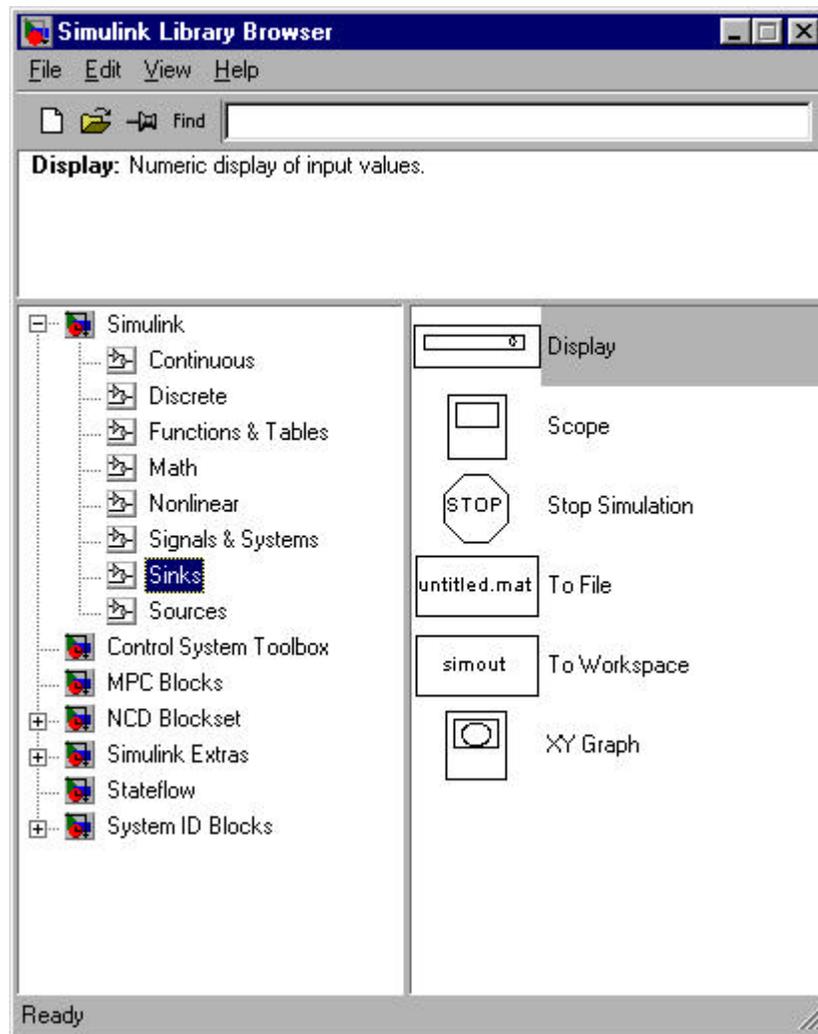
Selecting the Sources icon yields the library shown in Figure M2.3. The most commonly used sources are Clock (which is used to generate a time vector), and Step (which generates a step input).



**Figure M2.3** SIMULINK Sources.

The Sinks icon from Figure M2.1 can be selected to reveal the set of sinks icons shown in Figure M2.4. The one that we use most often is the To Workspace icon. A variable passed to this

icon is written to a vector in the MATLAB workspace. The default data method should be changed from “structure” to “matrix” in order to save data in an appropriate form for plotting.



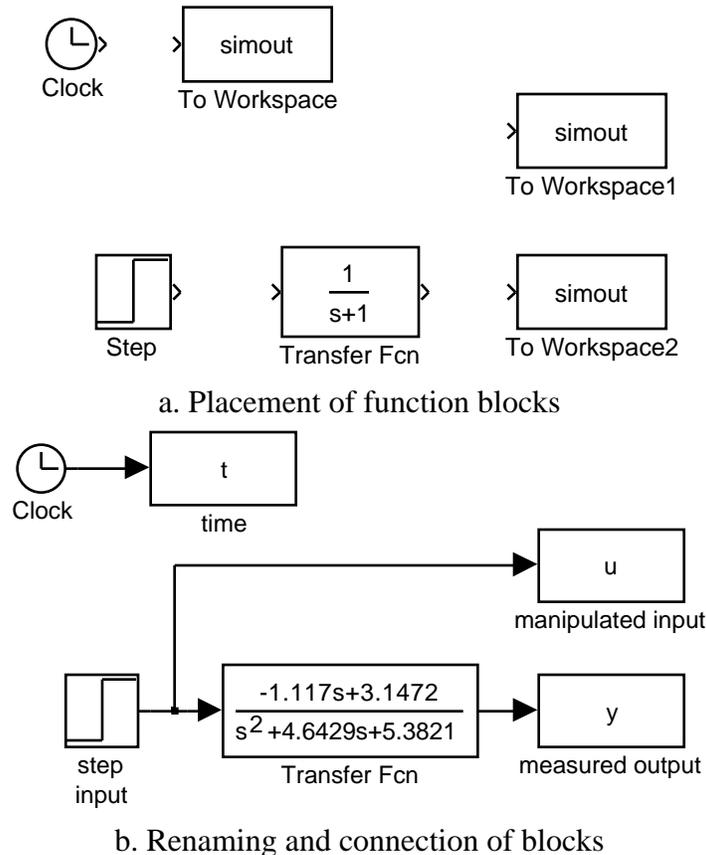
**Figure M2.4** SIMULINK Sinks.

## M2.2 Open-Loop Simulation

You now have enough information to generate an open-loop simulation. The Clock, `simout`, step and Transfer function blocks can be dragged to a model (.mdl) workspace, as shown in Figure M2.5a. Renaming the blocks and variables, and connecting the blocks, results in the model shown in Figure M2.5b. The transfer function studied is the Van de Vusse reactor (Module 5).

The s-polynomials in the process transfer function were entered by double-clicking on the transfer function icon and entering the coefficients for the numerator and denominator

polynomials. Notice also that the default step (used for the step input change) is to step from a value of 0 to a value of 1 at  $t = 1$ . These default values can be changed by double-clicking the step icon. The simulation parameters can be changed by going to the Simulation “pull-down” menu and modifying the stop time (default = 10) or the integration solver method (default = ode45).



**Figure M2.5** Development of an Open-loop Simulation.

The reader should generate simulations and observe the “inverse response” behavior of the output with respect to a step input change. Use the `subplot` command to place the process output ( $y$ ) on the top plot, and the manipulated input ( $u$ ) on the bottom plot. *Perform this now.* If desired, change the default simulation stop time by selecting the parameters “pull down” menu.

### M2.3 Feedback Control Simulation

The **Math** icon from Figure M2.2 can be selected, resulting in the functions shown in Figure M2.6. Additional icons can be found by selecting the **Simulink Extras** icon in Figure M2.1. Selecting the **Additional Linear** icon from this group yields the set of icons in Figure M2.7. The most useful icon here is the **PID Controller**. Any icon can be “dragged” into the **untitled** model workspace. In Figure M2.8 we show the preliminary stage of the construction of a control block diagram, where icons have been dragged from their respective libraries into the **untitled** model workspace.

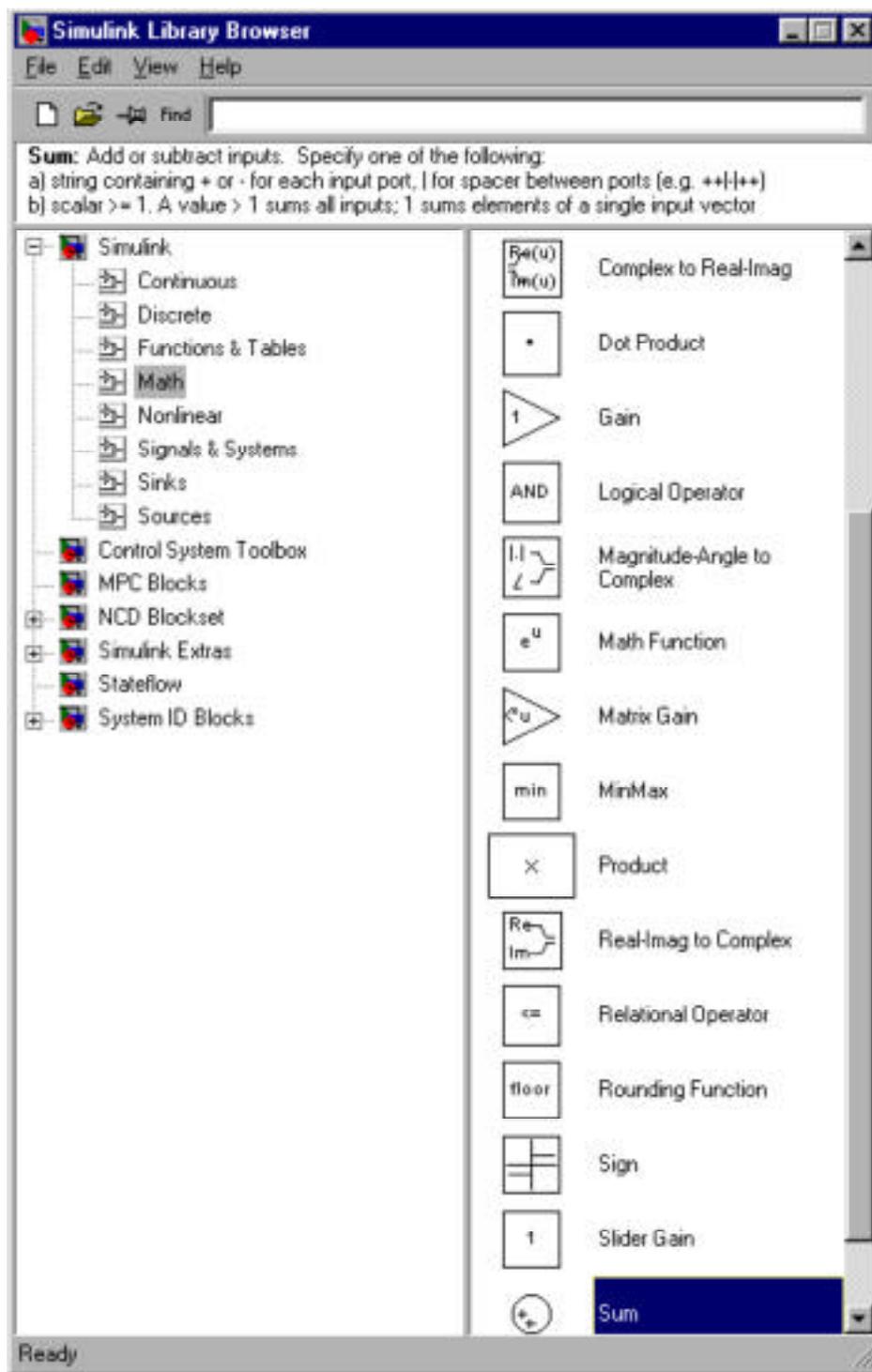
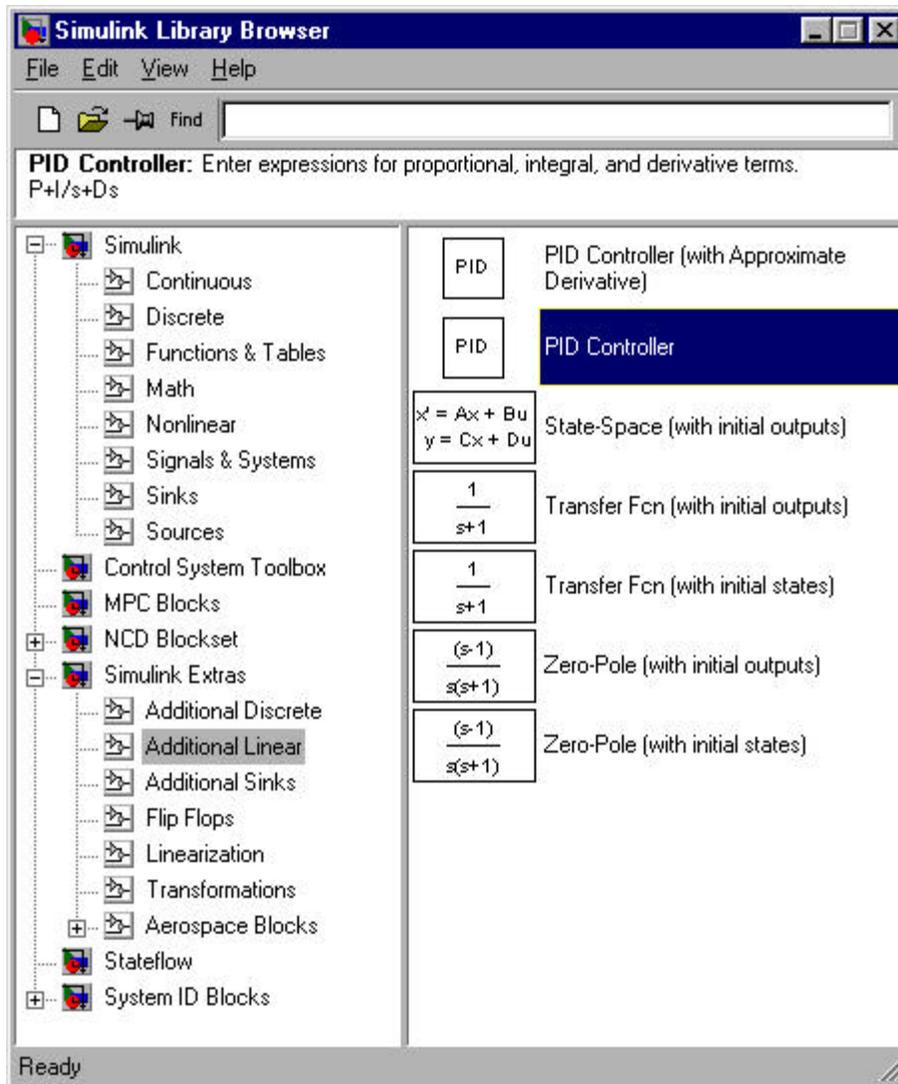


Figure M2.6 SIMULINK Math.



**Figure M2.7** SIMULINK Additional Linear.

The labels (names below each icon) can easily be changed. The default parameters for each icon are changed by double clicking the icon and entering new parameter values. Also, connections can be made between the outputs of one icon and inputs of another. Figure M2.8b shows how the icons from Figure M2.8a have been changed and linked together to form a feedback control block diagram.

It should be noted that the form of the PID control law used by the SIMULINK **PID Controller** icon is not the typical form that we use as process control engineers. The form can be found by double-clicking the icon to reveal the following controller transfer function representation

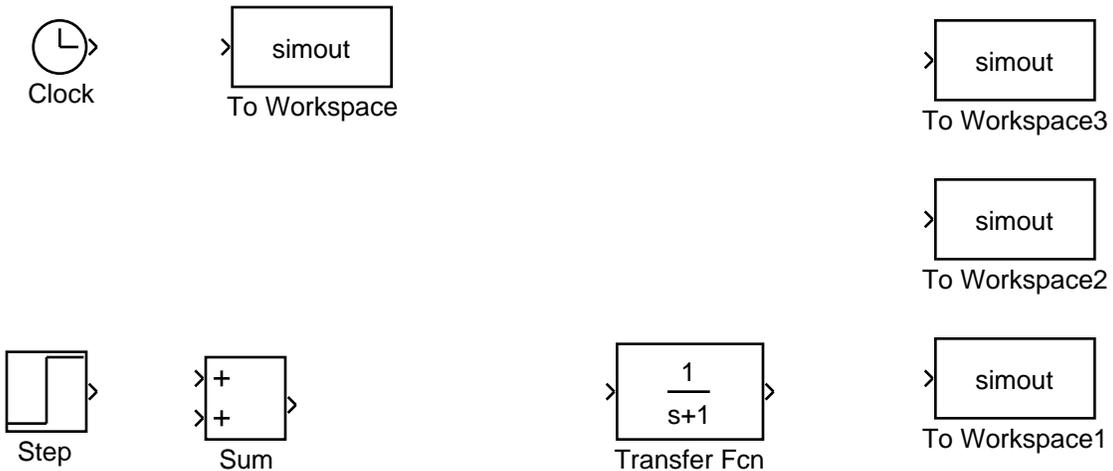
$$g_c(s) = P + \frac{I}{s} + Ds$$

while we normally deal with the following PID structure

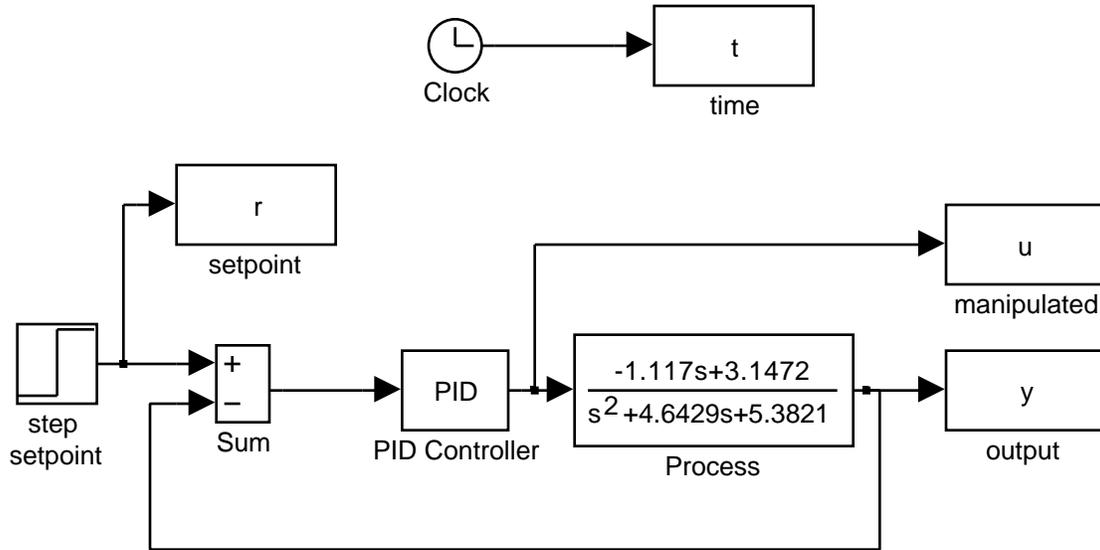
$$g_c(s) = k_c \left( 1 + \frac{1}{\tau_I s} + \tau_D s \right)$$

so the following values must be entered in the SIMULINK PID Controller

$$P = k_c \quad I = k_c / \tau_I \quad D = k_c \tau_D$$



a. Preliminary Block Diagram



b. Completed Block Diagram, with Name and Parameter Changes

**Figure M2.8.** Block Diagram for Feedback Control of the Van de Vusse CSTR

The s-polynomials in the process transfer function were entered by double-clicking on the transfer function and entering the coefficients for the numerator and denominator polynomials. Notice also that the default step (used for the step setpoint change) is to step from a value of 0 to

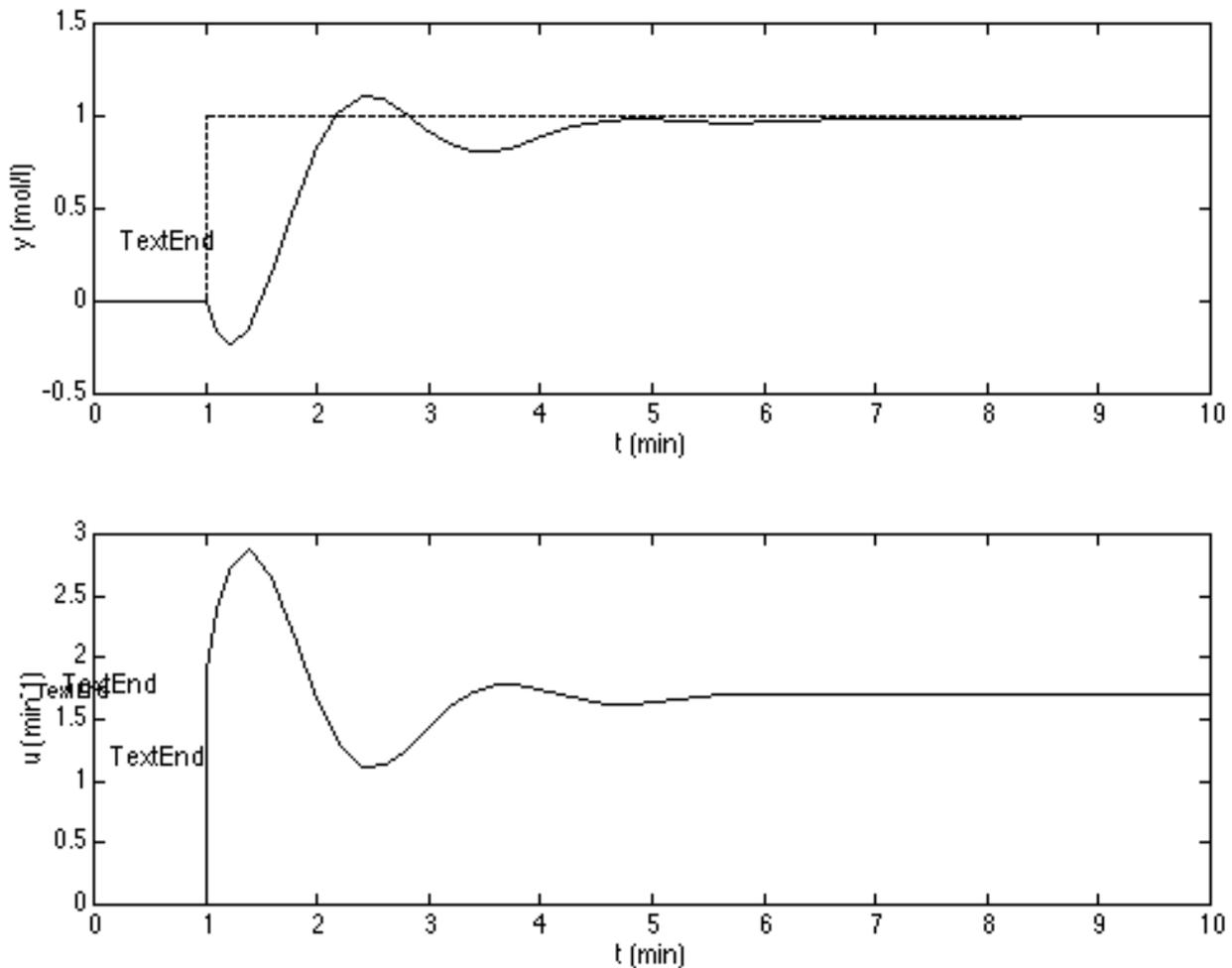
a value of 1 at  $t = 1$ . These default values can be changed by double-clicking the step icon. The simulation parameters can be changed by going to the Simulation “pull-down” menu and modifying the stop time (default = 10) or the integration solver method (default = ode45).

The controller tuning parameters of  $k_c = 1.89$  and  $\tau_I = 1.23$  are used by entering  $P = 1.89$  and  $I = 1.89/1.23$  in the default PID Controller block. The following plot commands were used to generate Figure M2.9.

```

» subplot(2, 1, 1), plot(t, r, '--', t, y)
» xlabel('t (min)')
» ylabel('y (mol/l)')
» subplot(2, 1, 2), plot(t, u)
» xlabel('t (min)')
» ylabel('u (min^-1)')

```



**Figure M2.9.** Measured Output and Manipulated Input Responses to a Unit Step Setpoint Change.

This brief tutorial has gotten you started in the world of SIMULINK-based control block diagram simulation. You may now easily compare the effect of different tuning parameters, or different formulations of a PID controller (“ideal” vs. “real”, for example).

Let’s say you generated responses for a set of tuning parameters that we call case 1 for convenience. You could generate time, input and output vectors for this case by setting:

```
» t1 = t;
» y1 = y;
» u1 = u;
```

after running the case 1 values. You could then enter case 2 values, run another simulation, and create new vectors

```
» t2 = t;
» y2 = y;
» u2 = u;
```

then, compare the case 1 and case 2 results by:

```
» subplot(2, 1, 1), plot(t1, y1, t2, y2, '--')
» xlabel('t (min)')
» ylabel('y (mol/l)')
» subplot(2, 1, 2), plot(t1, u1, t2, u2, '--')
» xlabel('t (min)')
» ylabel('u (min-1)')
```

which automatically plots case 1 as a solid line and case 2 as a dashed line.

Similarly, you could modify the controller type by placing a transfer function block for the controller and using a “real PID” transfer function (this only differs when there is derivative action).

## Other Commonly Used Icons

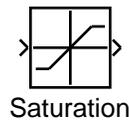
Often you will want to simulate the behavior of systems that have time delays. The Transport Delay icon can be selected from the Continuous library shown in Figure M2.2. The transport delay icon is shown in Figure M2.10. Our experience is that simulations can become somewhat “flaky” if 0 is entered for a transport delay. We recommend that you remove the transport delay block for simulations where no time-delay is involved.



Transport  
Delay

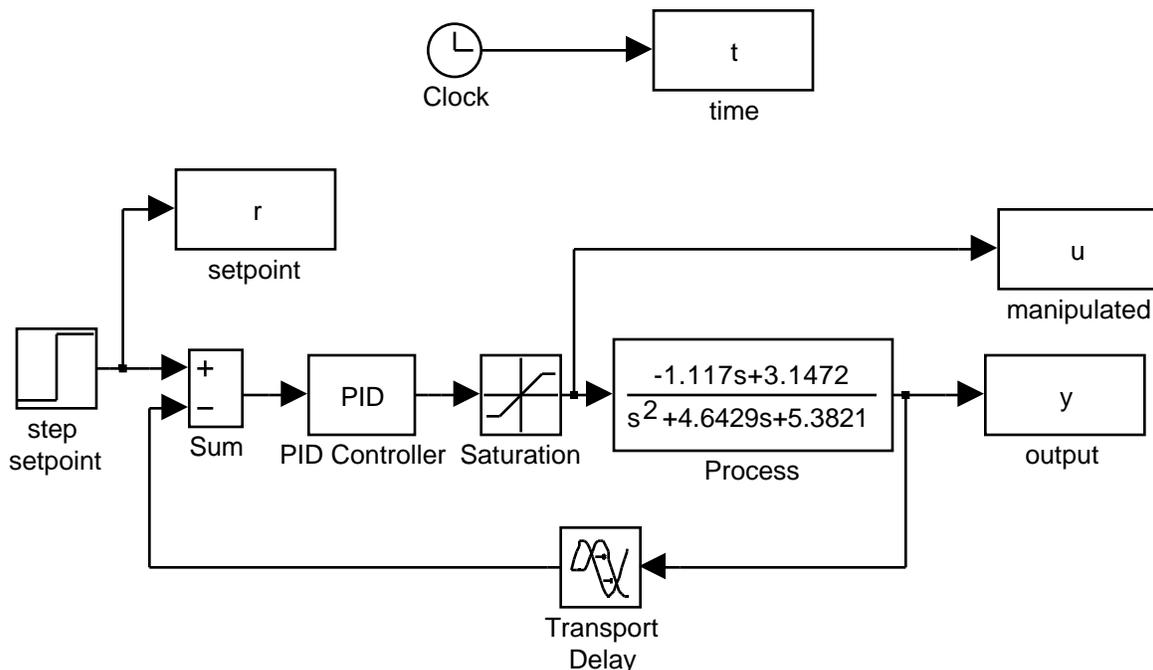
**Figure M2.10.** Transport Delay Icon

Manipulated variables are often constrained to between minimum (0 flow, for example) and maximum (fully open valve) values. A saturation icon from the Nonlinear library can be used to simulate this behavior. The saturation icon is shown in Figure M2.11.

**Figure M2.11.** Saturation Element

Actuators (valves) and sensors (measurement devices) often have additional dynamic lags that can be simulated by transfer functions. These can be placed on the block diagram in the same fashion that a transfer function was used to represent the process earlier.

It should be noted that icons can be “flipped” or “rotated” by selecting the icon and going to the format “pull-down” menu and selecting *Flip Block* or *Rotate Block*. The block diagram of Figure M2.8 has been extended to include the saturation element and transport delay, as shown in Figure M2.12.

**Figure M2.12.** Block Diagram with Saturation and Time-Delay Elements

The default data method for the “to workspace” blocks (r, t, u, y in Figure M2.12) must be changed from “structure” to “matrix” in order to save data in an appropriate form for plotting.

## M2.4 Developing Alternative Controller Icons

It was noted earlier that the default SIMULINK PID controller block uses a different form than that used by most process engineers. It is easy to generate new PID controller blocks as shown below. The default PID controller icon is shown in Figure M2.13a. This is “unmasked” to yield the diagram shown in Figure M2.13b. Again, this has the form

$$g_c(s) = P + \frac{I}{s} + Ds$$

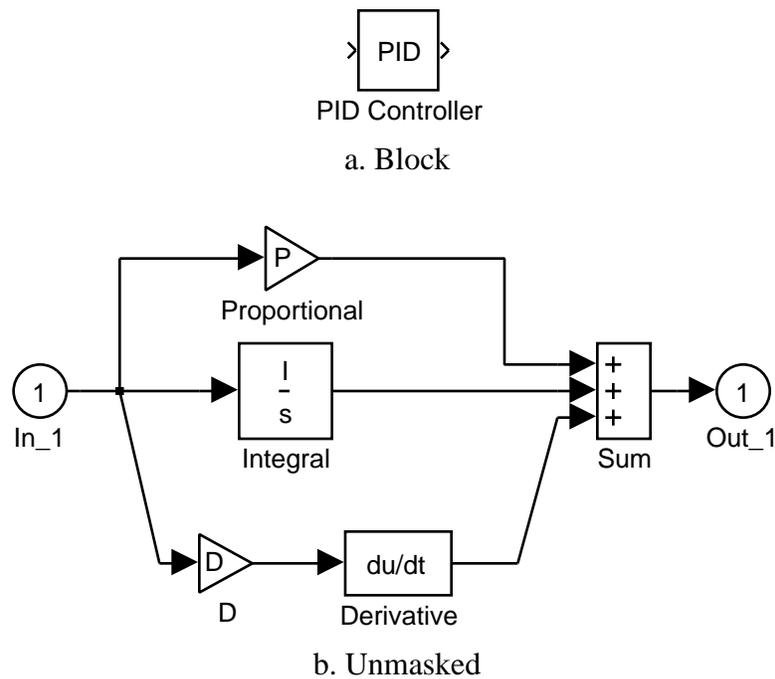
while we normally prefer the following PID structure

$$g_c(s) = k_c \left( 1 + \frac{1}{\tau_I s} + \tau_D s \right)$$

Of course, the two algorithms are related by

$$P = k_c \quad I = k_c / \tau_I \quad D = k_c \tau_D$$

but it would be much less confusing to work with our standard form.



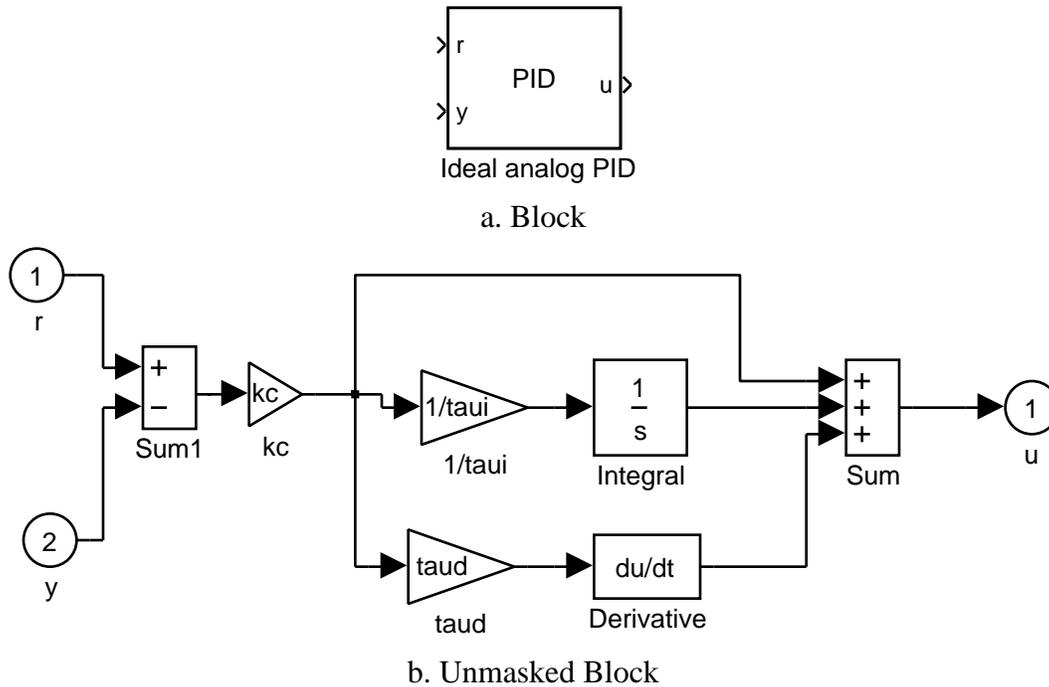
**Figure M2.13** Default Ideal PID Controller

We have generated a new ideal analog PID, as shown in Figure M2.14a. Notice that there are two inputs to the controller, the setpoint and the measured output, rather than just the error signal

that is the input to the default SIMULINK PID controller. Our new implementation is unmasked in Figure M2.14b to reveal that

$$g_c(s) = k_c \left( 1 + \frac{1}{\tau_I s} + \tau_D s \right)$$

is the algorithm used.



**Figure M2.14** Preferred Implementation of Ideal PID Controller

## Summary

SIMULINK is a very powerful block diagram simulation language. Simple simulations, including the majority of those used as examples in this textbook, can be set-up rapidly (in a matter of minutes). The goal of this module was to provide enough of an introduction to get you started on the development of open- and closed-loop simulations. With experience, the development of these simulations will become second-nature. It is recommended that you perform the simulations shown in this module, as well as the practice exercises, to rapidly acquire these simulation skills.

## Practice Exercises

1. Compare step responses of the transfer function and state space models for the Van de Vusse Reactor.

a. Transfer function: Perform the open-loop simulation for a step input change from 0 to 1 at  $t = 1$  in Figure M2.5.

b. State space: Replace the transfer function block of Figure M2.5 with a state space block. Enter the following matrices in the MATLAB command window.

$$A = \begin{bmatrix} -2.4048 & 0 \\ 0.83333 & -2.2381 \end{bmatrix} \quad B = \begin{bmatrix} 7 & 0.5714 \\ -1.117 & 0 \end{bmatrix}$$

$$C = [0 \quad 1] \quad D = [0 \quad 0]$$

Show that the resulting step responses of the transfer function and state space models are identical.

2. Consider the closed-loop SIMULINK diagram for the Van de Vusse reactor (Figure M2.8). Replace the default PID controller with a “Real PID” controller, as shown below in both the masked and unmasked versions. How do your results compare with ones obtained using the default PID controller?

