Appendix D: LabVIEW™[1] Tutorial

D.1 Introduction

LabVIEW is a programming environment that is presented here as an alternative to MATLAB. Although not necessary, the reader is encouraged to become acquainted with MATLAB before proceeding, since familiarity with MATLAB can enhance the understanding of the relationship between textual (MATLAB) and graphical (LabVIEW) programming languages and extend the functionality of LabVIEW. In this tutorial, we will show how to use LabVIEW to (1) analyze and design control systems, and (2) simulate control systems. This appendix was developed using LabVIEW 2009.

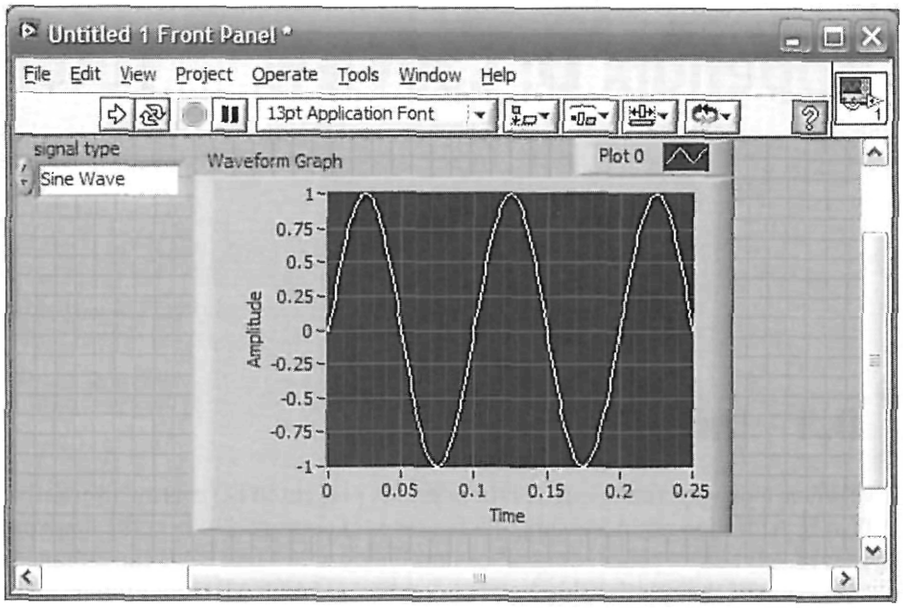
LabVIEW is a graphical programming environment that produces virtual instruments (VI's). A VI is a pictorial reproduction of a hardware instrument on your computer screen, such as an oscilloscope or waveform generator. The VI can consist of various controls and indicators, which become inputs and outputs, respectively, to your program. Underlying each control and indicator is an associated block of code that defines its operation. The LabVIEW model thus consists of two windows: (1) **Front Panel**, which is a replica of the hardware front panel showing the controls and indicators, and (2) **Block Diagram**, which contains the underlying code for the controls and indicators on the **Front Panel**.

Associated with the **Front Panel** window is a **Controls** palette window containing numerous icons representing controls and indicators. The icons can be dragged onto a **Front Panel** window to create that control or indicator. Simultaneously, the associated code block is formed on the **Block Diagram** window.

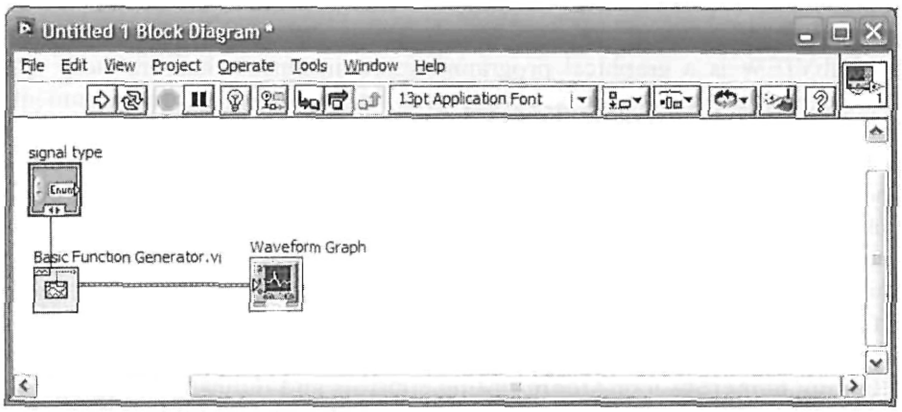
Alternately, the block diagram can be formed first, and then the front panel is created from the block diagram. Associated with the **Block Diagram** window is a **Functions** palette window containing numerous icons representing a wide range of functions. Icons can be dragged onto a **Block Diagram** window to create that code block.

For example, Figure D.1(*a*) is the front panel of a signal generator. The generator consists of a control to select the signal type and a waveform graph that shows the output waveform. Figure D.1(*b*) shows the underlying code, which is contained in the code blocks. Here, the signal type selector is a control, while the waveform graph is an indicator. Later we will show how to make connections to other VI's. The palette windows for the front panel and block diagram are shown respectively in Figures D.1(*c*) and (*d*).

[1] LabVIEW is a registered trademark of National Instruments Corporation.



(*a*)



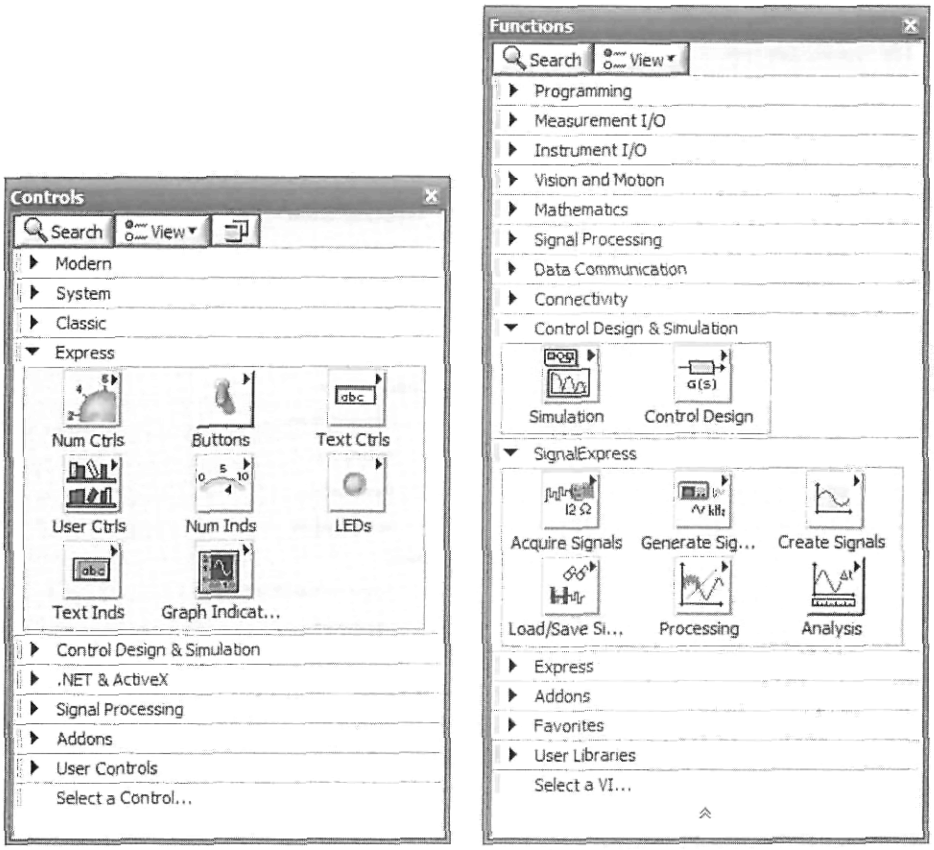
(*b*)

FIGURE D.1 A LabVIEW function generator VI: **a. Front Panel** window; **b. Block Diagram** window; (*figure continues*)

D.2 Control Systems Analysis, Design, and Simulation

LabVIEW can be used as an alternative to or in conjunction with MATLAB to analyze, design, simulate, build, and deploy control systems. In addition to LabVIEW, you will need the LabVIEW Control Design and Simulation Module. Finally, as an option that will be explained later, you may want to install the MathScript RT Module.

Analysis and design can be thought of as similar to writing MATLAB code, while simulation can be thought of as similar to Simulink. In LabVIEW, analysis and design, as opposed to simulation, are handled from different subpalettes of the **Functions** window's **Control Design and Simulation** palette. See Figure D.1(*d*). Analysis and design, and simulation will typically begin with the **Block Diagram** window, where icons representing code blocks will be interconnected. Parameters



(*c*) (*d*)

FIGURE D.1 (*Continued*) c. **Controls** palette; **d. Functions** palette

used by the code can be conveniently selected, changed, and passed to the code through VI controls on the **Front Panel** window created from the code icons. Any results, such as time response, can be displayed through VI indicators on the **Front Panel** window created from the code icons.

D.3 Using LabVIEW

The following steps start you on your way to using LabVIEW for control systems analysis, design, and simulation. These steps will be illustrated in the examples that follow.

1. **Start LabVIEW** LabVIEW starts with the **Getting Started** window shown in Figure D.2, where you can select a **New** file or **Open** an existing file. You may also select various resources. Selecting **Blank VI** under the **New** label or **New VI** under the **File** menu brings up the **Front Panel** and **Block Diagram** windows shown in Figure D.1. If necessary, a window can be opened from the **Window** tab on the menu bar of the **Front Panel** and **Block Diagram**.

   Right-click the **Block Diagram** window to bring up the **Functions** palette and click the thumb tack in the upper left-hand corner to dock the window. Repeat for the **Front Panel** window to access the **Controls** palette.

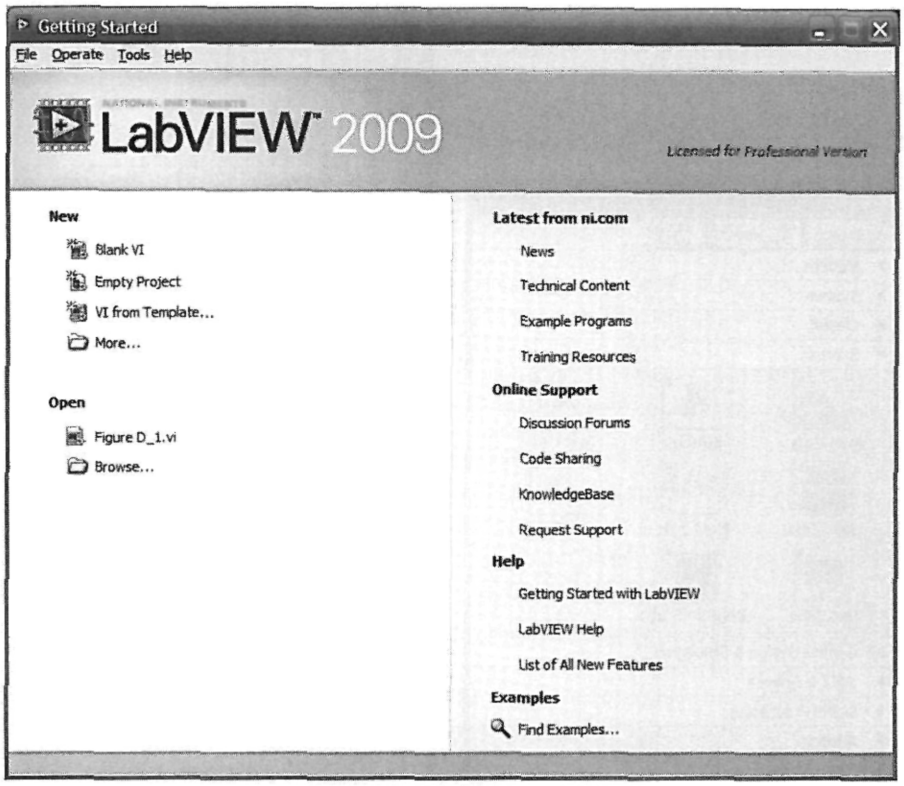


FIGURE D.2 LabVIEW's **Getting Started** window

2. **Select blocks** Make the **Block Diagram** window active, or access it from **Window** on the menu bar. Right-click the **Block Diagram** window or use the **View** menu to bring up the **Functions** palette. Expand the palette window by clicking the double-up arrows at the bottom. At the top of the palette window click **View**, and select **View This Palette As/Category (Icons and Text)** to add a text description below each icon. For control systems analysis, design, and simulation, expand **Control Design & Simulation** in the **Functions** palette by clicking the arrow to the left of this category.

   If you are performing a simulation click the subpalette **Simulation**. If you are performing control system analysis or design, click the subpalette **Control Design**. An arrow in the upper-right corner of a subpalette indicates additional underlying palettes or blocks.

   If the name of the icon is incomplete, resting the mouse over the icon will bring up its complete identification. To obtain detailed help about an icon, right-click the icon and select **Help**.

3. **Move blocks to the block diagram window** To move the icon to the **Block Diagram**, left-click the mouse to attach the icon (some icons take a little time to complete this operation). When the pointer turns into a hand, click the spot on the **Block Diagram** where you want to place the icon.

4. **Obtain information about the block** You will now want to obtain information about how to interconnect the block to other blocks and pass parameters to the block as well as other characteristics about the block. Select the yellow question mark at the right of the **Block Diagram** toolbar to turn on the **Context Help window**. This window will provide help about a particular icon if you rest your

mouse over that icon. Additional help is available under the **Help** menu on the **Block Diagram** menu bar. Finally, right-click the icon to bring up a menu with additional choices, such as **Properties**, if any. In particular, you will use this menu to create the block's front panel's controls and indicators. This front panel will be your interface with the block to choose parameters and see responses.

5. **Interconnect and label blocks** Once blocks are placed on the **Block Diagram** they can be moved about by clicking on them or dragging your mouse across several or them to establish a selection pattern. After the selection pattern has been established, depress the mouse left button and drag to a new location. To delete a block, select the block and press the Delete button.

   The context help for the block includes a description of the block's terminals. Let your mouse rest on a terminal until the mouse pointer turns into a spool of wire. Click the terminal and then move the mouse to the next icon's terminal where you want to make the connection. Click the destination terminal to complete the wiring. Notice that the terminal in the **Context Help** window blinks when your mouse resides above that terminal on the block, ensuring that you are on the correct terminal. If you make an error in wiring, click on the wire and press the Delete button or right-click the wire and select **Delete Wire Branch**.

   Block labels can be displayed or hidden. Right-click on the block to bring up the pop-up menu and check or uncheck **Visible Items/Label** to display or hide, respectively, the label. Double-clicking on the label above some blocks will allow you to select and change the text. One click of the mouse on the label will place a selection pattern around the label and allow you to hold down the left key of the mouse and move the label to a different location.

6. **Create the interface to your block** You will now want to create the interface to your block in order to control or select functions, specify parameters, or view responses. This interface will be accessed via the **Front Panel** window. Right-click a terminal on a block for which you want to create an interface. On the pop-up menu, choose **Create/Control** to be able to interact with the block or **Create/Indicator** to view a response or setting.

7. **Set the controls** Switch to the **Front Panel** window and set your controls. For example, enter parameter values, select functions, etc. If you want to change values and at some future time return to the current values, click on **Edit** on the **Block Diagram** menu bar and select **Make Current Values Default**. To return to the default values in the future, click on **Edit** on the **Block Diagram** menu bar and select **Reinitialize Values to Default**.

8. **Run the program** Click on the arrow at the left of the toolbar on either the **Block diagram** or **Front Panel** window to run the program. The program can be run continuously by clicking the curved arrow button on the toolbar second from the left. Continuously running your program permits changing functions and parameter values during execution.

   In order to identify the buttons, let your mouse rest on a button to bring up a context menu. Stop your simulation by pressing the red-dot button, third from the left. If you are performing control systems analysis and design, another way to continuously run the program is to place a **While Loop** around your block diagram. The loop is available in the **Functions** palette at **Express/Execution Control/While Loop**. This loop also places a **Stop** button on the **Front Panel**. The program executes until you press the stop button. In lieu of the **Stop** button, any true/false Boolean can be wired to the condition block (red dot) created inside the **While Loop**.

If you are performing simulation, you can use a **Simulation Loop** available in the **Functions** palette at **Control Design and Simulation/Simulation/Simulation Loop**. Place the **Simulation Loop** around your simulation block diagram by dragging the mouse. Right-click on the **Simulation Loop** outline and choose **Configure Simulation Parameters** . . . to determine the parameters for executing the simulation. The **Front Panel** indicators and controls are also configurable. Right-click on the indicator or control and select **Properties**.

D.4 Analysis and Design Examples

In this section, we will present some examples showing the use of LabVIEW for the analysis and design of control systems. In the next section, examples of the use of LabVIEW for simulation will be presented.

Analysis and design examples use icons selected from the **Control Design** subpalette under the **Control Design and Simulation** palette. In the next section showing examples of simulation, we will use icons taken from the **Simulation** subpalette under the **Control Design and Simulation** palette.

Example D.1

Open-Loop Step Response

TryIt D.1

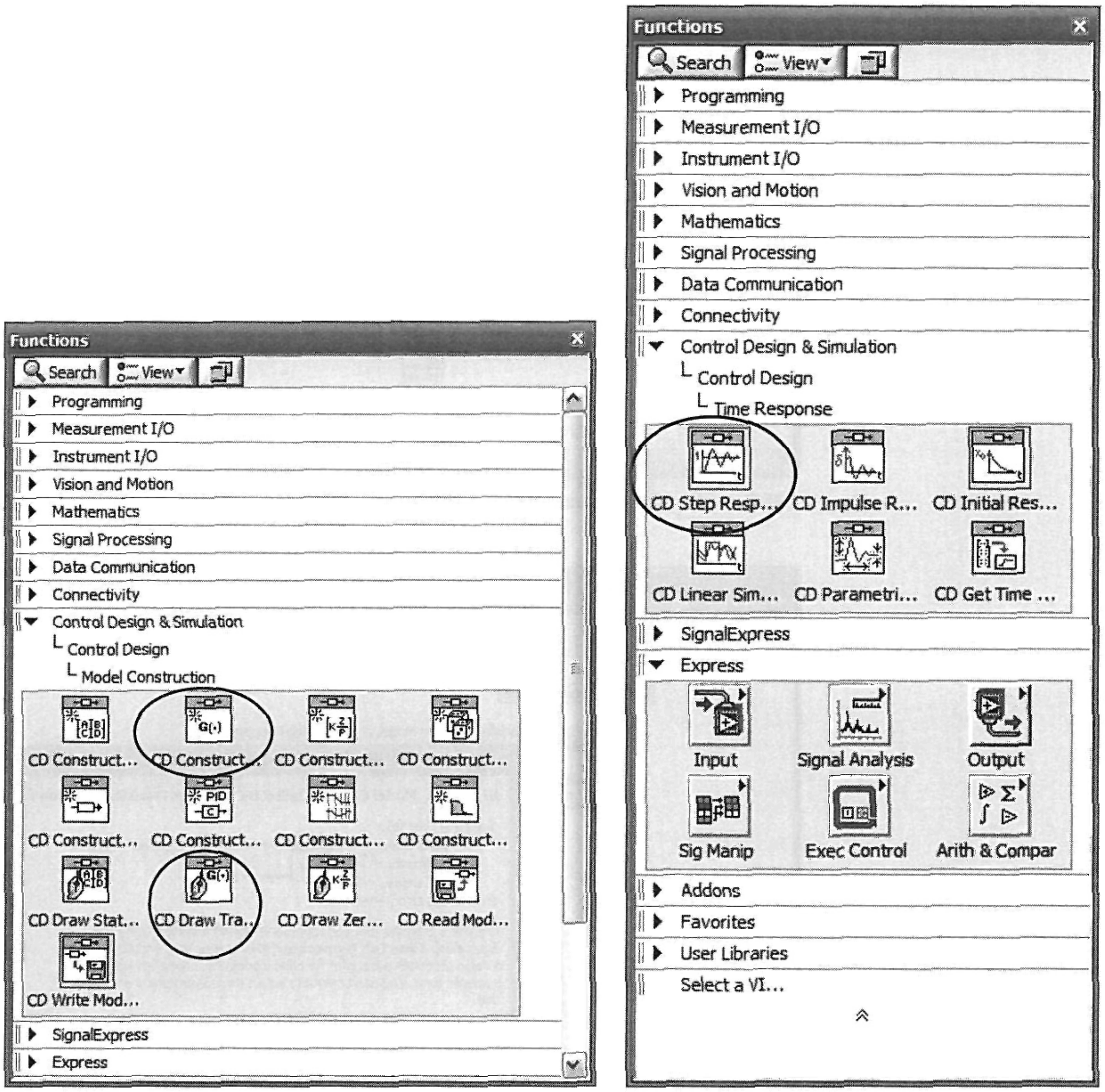
```
numg=100;
deng=[1 2 100];
'G(s)'
G=tf(numg,deng)
step(G);
title('Angular Velocity')
```

Analysis and design usually begins by selecting icons from the **Control Design** subpalette and dragging them to the **Block Diagram** window. The icons represent blocks of code and the cascading of code blocks can be thought of as a sequence of lines of code. Thus, an advantage of LabVIEW over MATLAB is that the programmer does not need to memorize coding language. For example, consider the MATLAB code shown in TryIt D.1 that produces the step response of $G(s) = 100/(s^2 + 2s + 100)$:

This step response can be produced in LabVIEW without knowing any coding language. We now demonstrate by following each step of Section D.3:

1. ***Start LabVIEW*** *Start LabVIEW and select* **Blank VI** *from the window shown* in Figure D.2.
2. **Select blocks** From the **Functions** palette, select the blocks shown in Figure D.3(a) and (b).
3. **Move/blocks to the Block Diagram window** Drag your icons one at a time to the **Block Diagram** window, Figure D.4.
4. ***Obtain information about the block*** *Right-click each of the blocks and be sure* the first two items under **Visible Items** are checked. Look at the **CD Construct Transfer Function Model.vi.** A **Polymorphic VI Selector** is shown at the bottom. Click the selector to bring up the menu. Select **SISO.** This block effectively creates the transfer function shown in the first four steps of the MATLAB code in TryIt D.1.

   Repeat for the **CD Draw Transfer Function Equation.vi** and select **TF** from the **Polymorphic VI Selector.** This block will write the transfer function symbolically in the display. Your selection from the polymorphic vi selector should match the format of the transfer function created by the **CD Construct Transfer Function Model.vi.**



(*a*) (*b*)

FIGURE D.3 Selecting **a. CD Construct** . . . and **CD Draw** . . . ; **b. CD Step Response** . . .

Repeat for the **CD Step Response.vi.** and select **TF** from the **Polymorphic VI Selector**. This block will collect the data for the step response and permit plotting the data. This block effectively creates the last two commands of the MATLAB code shown in TryIt D.1.

5. **Interconnect and label blocks** You should now have the **Block Diagram** window shown in Figure D.4. Interconnect the code blocks. Click on the question mark on the right side of the toolbar to bring up the context menu. As your mouse passes above an icon, its context menu appears, showing the terminals. See Figure D.5. Interconnect the terminals by letting the mouse rest on a terminal until it becomes a spool of wire.

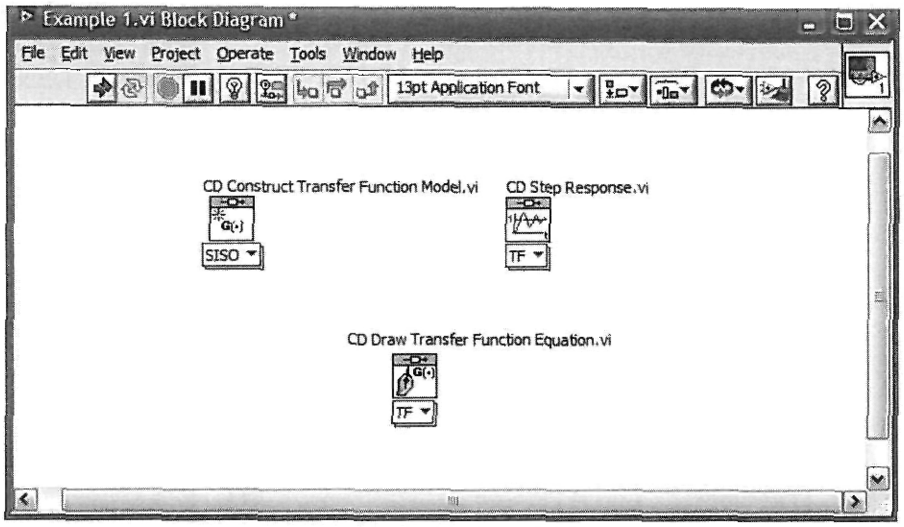


FIGURE D.4 **Block Diagram** window

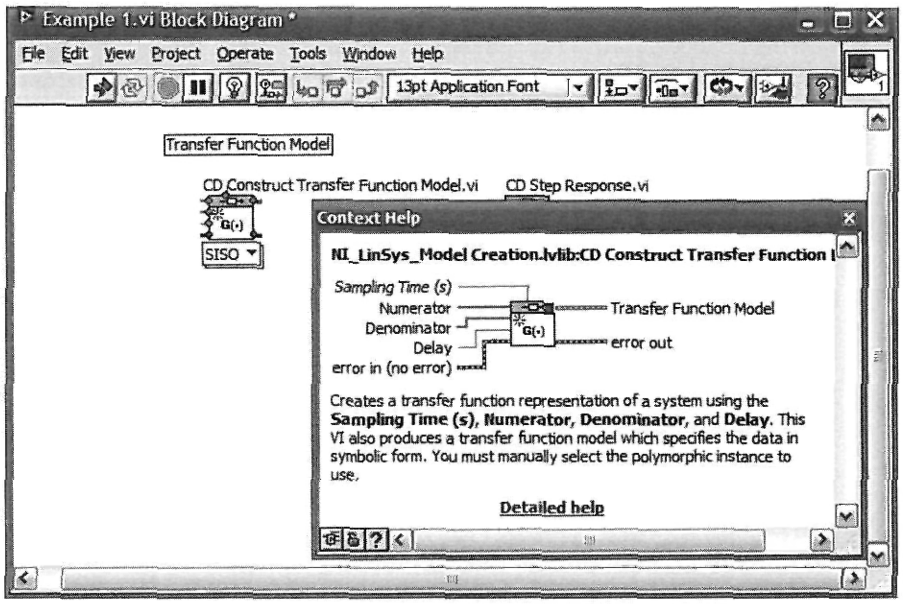


FIGURE D.5 **Context Help** for **CD Construct Transfer Function Model.vi**

Click on the terminal and then click on the destination terminal. The two terminals will appear as wired together. Continue wiring terminals until you have the **Block Diagram** window shown in Figure D.6. Mid-wire connections as shown can be made by letting your mouse rest at the connection point until it becomes a spool of wire.

6. **Create the interface to your block** You will now want to create the interface to specify parameters and view responses. This step will create the interface that will be accessed on the **Front Panel** window. The interfaces we will create are:

   - ***CD Construct Transfer Function Model.vi*** *input parameter controls.* Right-click on the numerator terminal shown in Figure D.5 and select **Create/Control**. Repeat for the denominator.

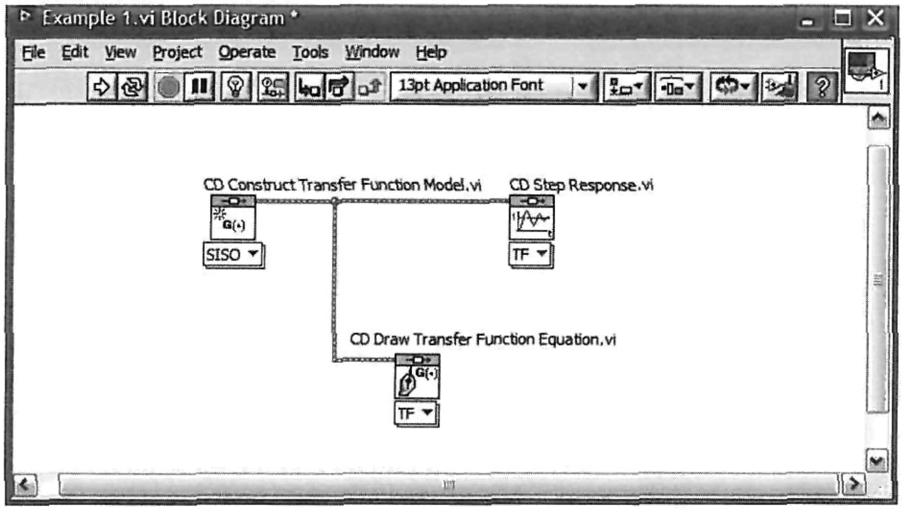
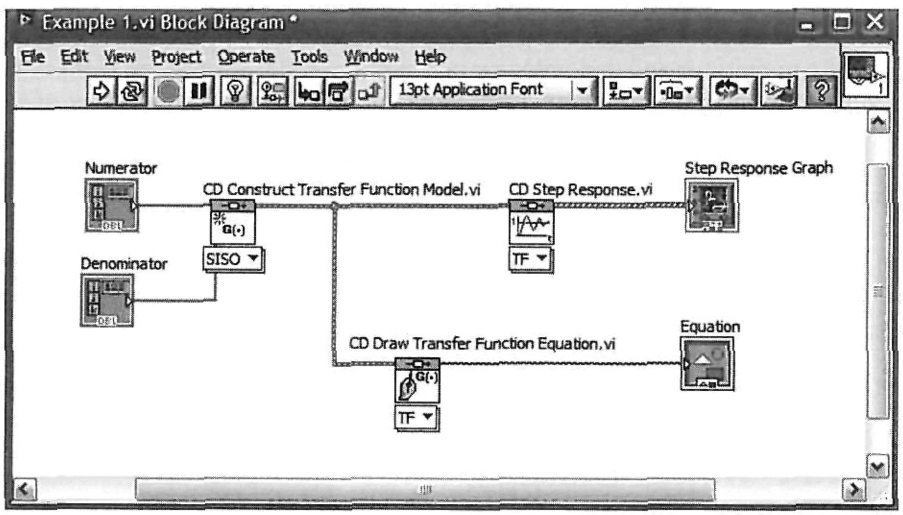


FIGURE D.6 Interconnected blocks

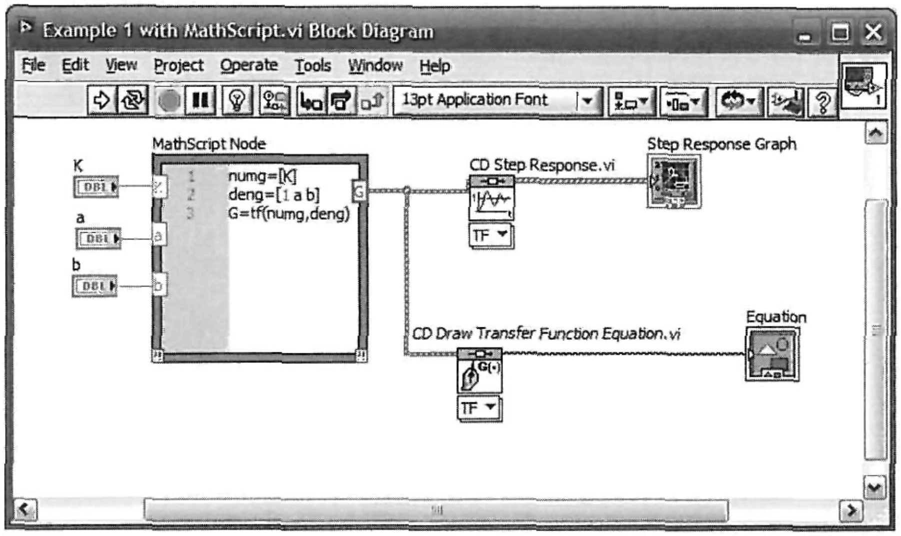
- **CD Step Response.vi** response plot indicator. Right-click on the **Step Response Graph** terminal and select **Create/Indicator**.
- **CD Draw Transfer Function Equation.vi** symbolic transfer function indicator. Right-click on the **Equation** terminal and select **Create/Indicator**. Your **Block Diagram** should now look similar to Figure D.7(*a*).

As an option, you can create transfer functions using a **MathScript** block if the MathScript RT Module is installed. This option is generally compatible with MATLAB's M-file code statements for creating your transfer function. Interfaces are then created to pass parameters to and from the M-file code. You should be familiar with MATLAB to use this option. The **MathScript** block is found in the **Programming/Structures/MathScript** palette. You create M-file code inside the **MathScript** block. Input and output interfaces are created and named identically



(*a*)

FIGURE D.7 **Block Diagram** window: **a.** with **Control Design** blocks and interfaces; (*figure continues*)



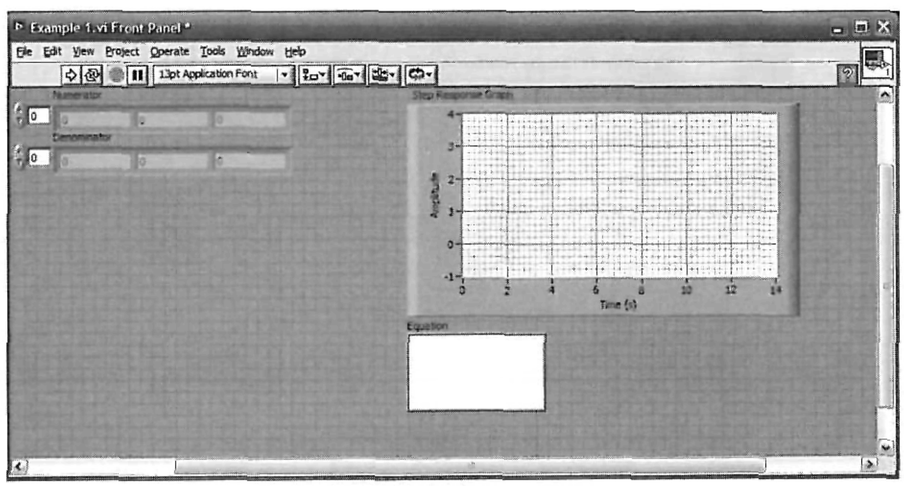
(*b*)

FIGURE D.7 (*Continued*) **b.** with **MathScript** block

to those within the M-file code. However, when using MathScript, you must create controls first on the **Front Panel**, rather than the **Block Diagram**. For example, to create the numeric interfaces for K, a, and b, right-click the **Front Panel** to produce the **Controls** palette. From this palette, produce each numeric control from **Modern/Numeric/Numeric Control**. The resulting controls are shown in Figure D.8(*b*). These interfaces are then wired to the appropriate terminals on the **Block Diagram**. Your **Block Diagram** should now look similar to Figure D.7(*b*).

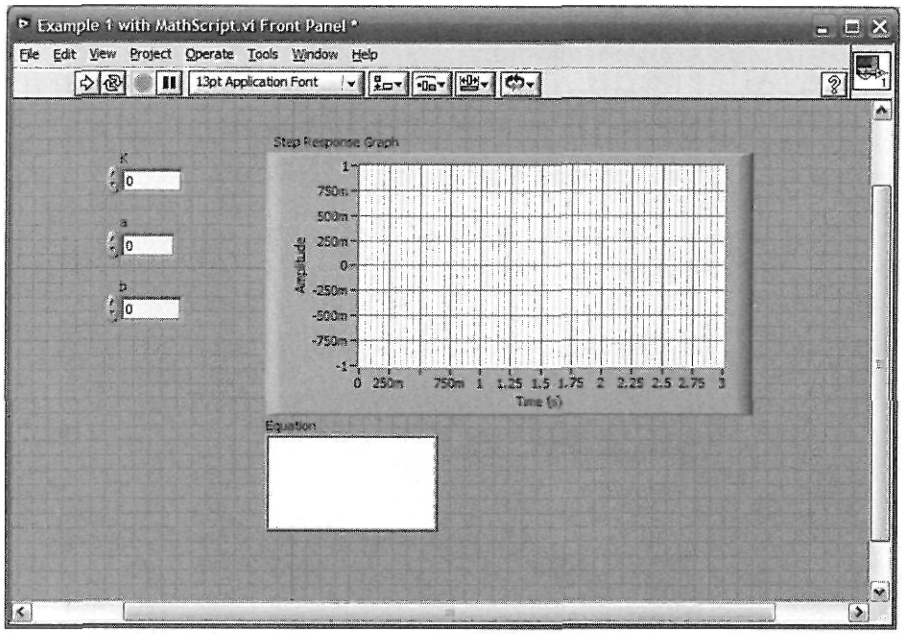
On the **Block Diagram** window menu bar, select **Window/Show Front Panel**. You will see the **Front Panel** shown in Figure D.8 created by your interfaces. You can double-click the labels above your interfaces either in the **Front Panel** window or the **Block Diagram** window to change the label to be more descriptive of your project.

7. **Set the controls** Using the **Front Panel** window, enter polynomial coefficients for the numerator and denominator in ascending order—lowest to highest. The selector to the left of the numerator and denominator shows the power of s for



(*a*)

FIGURE D.8 **Front Panel: a.** for **Block Diagram** shown in Figure D.7(*a*); (*figure continues*)



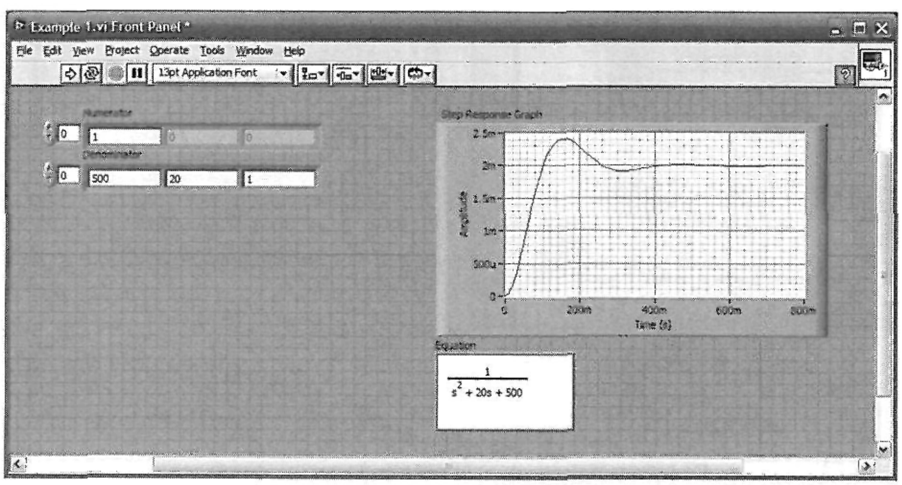
(*b*)

FIGURE D.8 (*Continued*) **b.** for **Block Diagram** shown in Figure D.7(*b*)

the left-most coefficient. Increasing the counter allows entry of higher-order coefficients not visible originally. To make all coefficients of a polynomial visible, let the mouse move on the right-hand edge of the polynomial indicator until the pointer becomes a double arrow and blue dots appear at the left and right edges of the entire polynomial indicator. You can then drag the right blue dot to expose more cells.

Familiarize yourself with the choices on the menu bar as well as those on the pop-up menus created when you right-click on any indicator or control. For example, under the **Edit** menu, among other choices, you can **Make Current Values Default** or **Reinitialize Values to Default**. Right-clicking the indicators or controls brings up a menu from which, among other choices, **Properties** can be selected to configure the indicator or control as desired.

8. **Run the program** Figure D.9 shows Example D.1 after execution. The figure shows the values entered, the equation, and the step response. Execution was initiated by clicking the arrow at the left of the toolbar.



(*a*)

FIGURE D.9 **Front Panel** after execution: **a.** for block diagram in Figure 7(*a*); (*figure continues*)

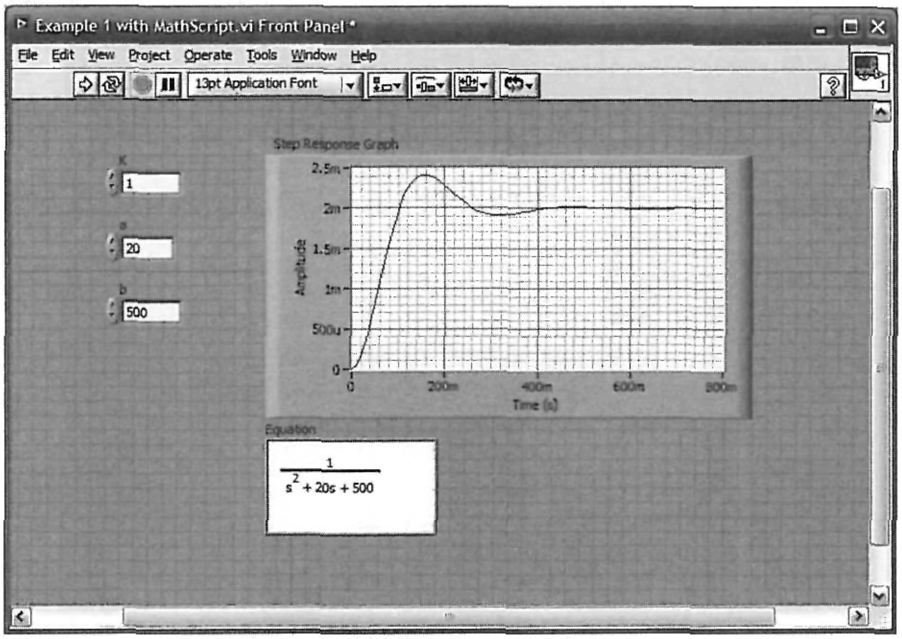
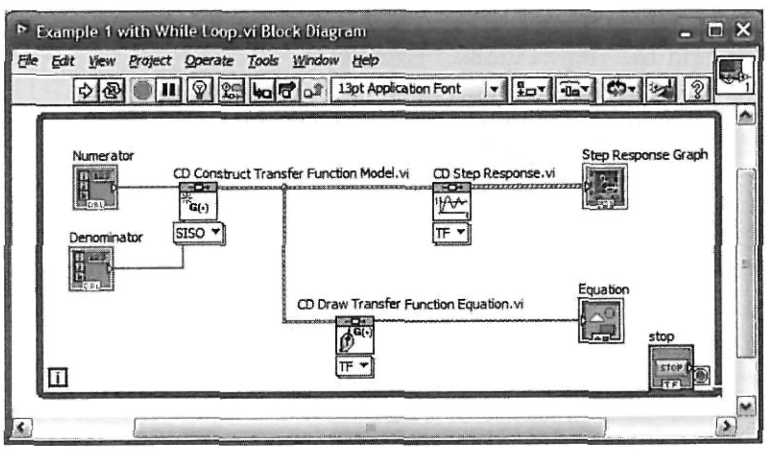


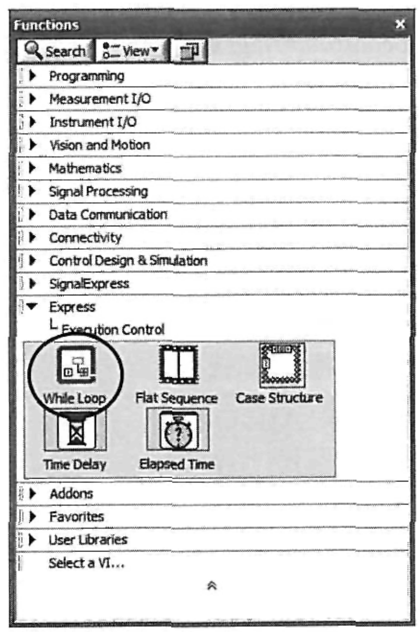
FIGURE D.9 (*Continued*) **b.** for block diagram in Figure 7(*b*)

(*b*)

The program can run continuously by clicking the curved arrows on the toolbar. Now, change values; hit the **Enter** key and see the results immediately displayed. Stop the program execution by clicking on the red hexagon on the toolbar. Another way of continuously running the program is to place a **While Loop** around the block diagram as shown in Figure D.10(*a*). The loop is accessed from **Functions/Express/Execution Control** as shown in Figure D.10(*b*). After



(*a*)



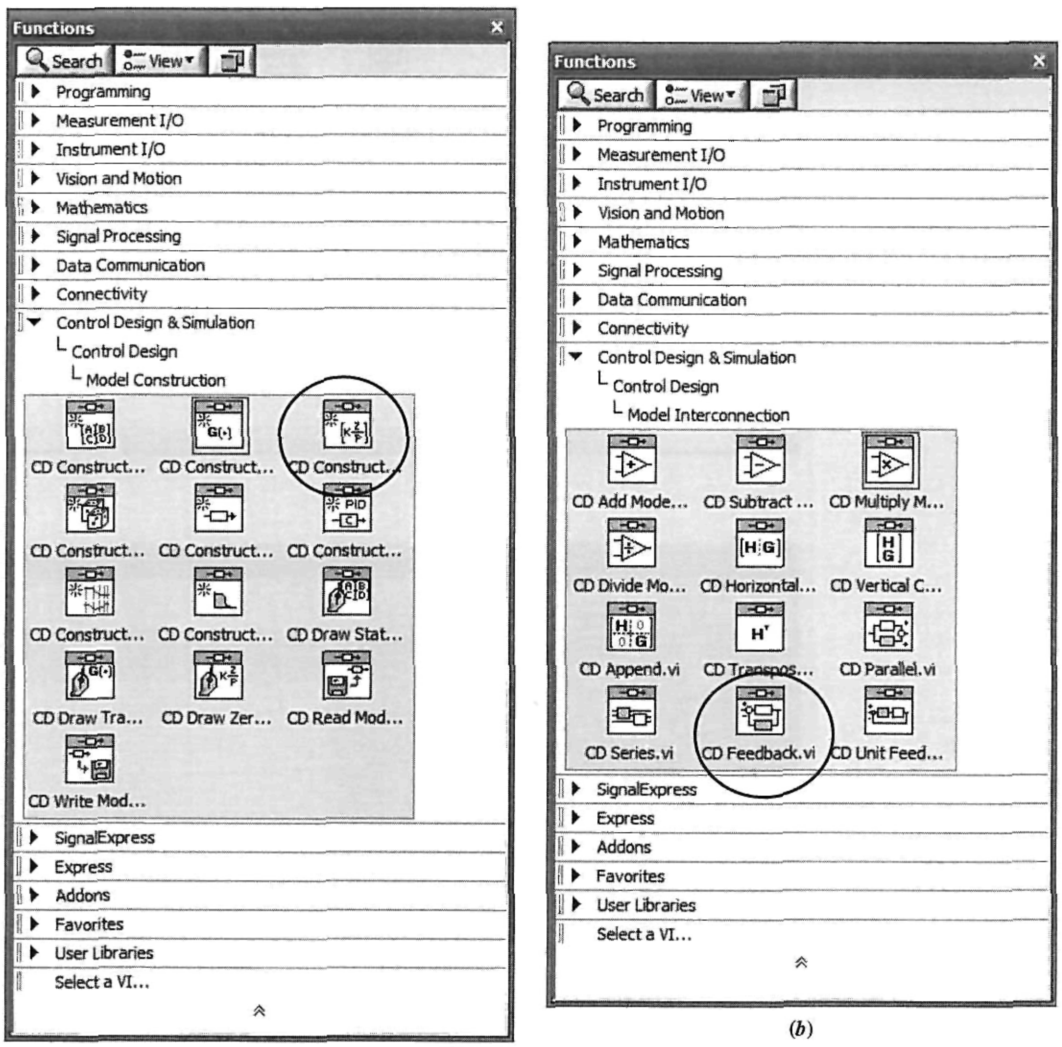
(*b*)

FIGURE D.10 **a.** Block diagram with **While Loop**; **b. Functions** palette showing **While Loop** location selecting the **While Loop**, drag the cursor across the block diagram to create the continuous loop. A **stop** button will appear on the block diagram as well as on the **Front Panel**. At the lower right is a **Loop Interation** icon, which can be used to control the **While Loop**. The reader should consult the on-line documentation for further information.

Example D.2

Closed-Loop Step Response

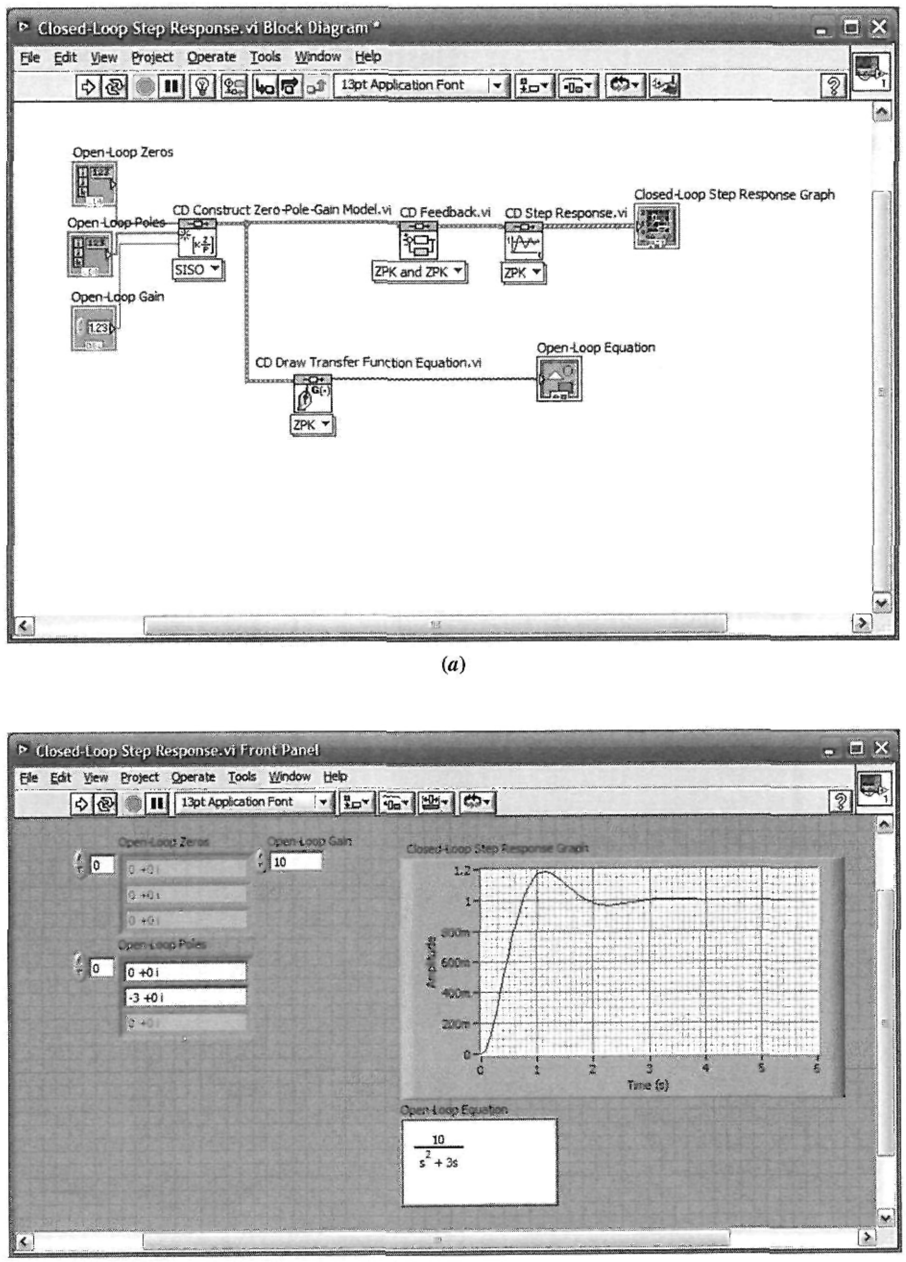
In this example, we show how to display the step response of a unity-feedback system. For variety, we represent the open-loop system as a ratio of zeros over poles



(*a*) (*b*)

FIGURE D.11 **a.** Obtaining zero-pole-gain transfer function from the **Functions** palette; **b.** Obtaining **Feedback** interconnection from **Functions** palette with a multiplying gain, analogous to MATLAB's **zpk** function. In the previous example, we represented the system as a ratio of polynomials, analogous to MATLAB's **tf** function.

1. **Select blocks** The zero-pole-gain transfer function is obtained from the **Functions** palette as shown in Figure D.11(a). We place this transfer function in the forward path of a unity-feedback system by following its block with a **Feedback** block obtained from the **Functions** palette as shown in Figure D.11(b).



(a)

(b)

FIGURE D.12 **a. Block Diagram** for Example D.2; **b. Front Panel** for Example D.2

If the **Model 2** input to the **Feedback** block is left unconnected, then a unity-feedback interconnection is assumed. Other options for interconnection, such as parallel and series, are shown on the palette of Figure D.11(*b*).

2. **Interconnect and label blocks** Producing the closed-loop step response is similar to Example D.1, except the step-response blocks are placed at the output of the **Feedback** block. The equation writer is wired to the system output as in Example D.1. All data types must be compatible and are shown selected with the pull-down menu at the base of the blocks. If you select **Automatic** in the pull-down menu, LabVIEW will select the correct form for you as you connect the blocks.

   The final **Block Diagram** and **Front Panel** for this example are shown in Figure D.12 (*a*) and (*b*), respectively. Notice that you enter open-loop poles, zeros, and gain on the **Front Panel** in place of polynomial open-loop numerator and denominator coefficients.

Example D.3

Root Locus Analysis and Design

We can obtain root locus plots by adding the **Root Locus** block obtained from the **Functions** palette as shown in Figure D.13. The **Root locus** block is connected to the output of the open-loop system and a **Root Locus Graph** indicator is formed at the output of the **Root Locus** block. The resultant **Block Diagram** and **Front Panel** are shown in Figure D.14(*a*) and (*b*) respectively.

Figure D.13 shows other characteristic blocks that can be added. For example, closed-loop poles and zeros, as well as damping ratio and natural frequency, can be displayed.

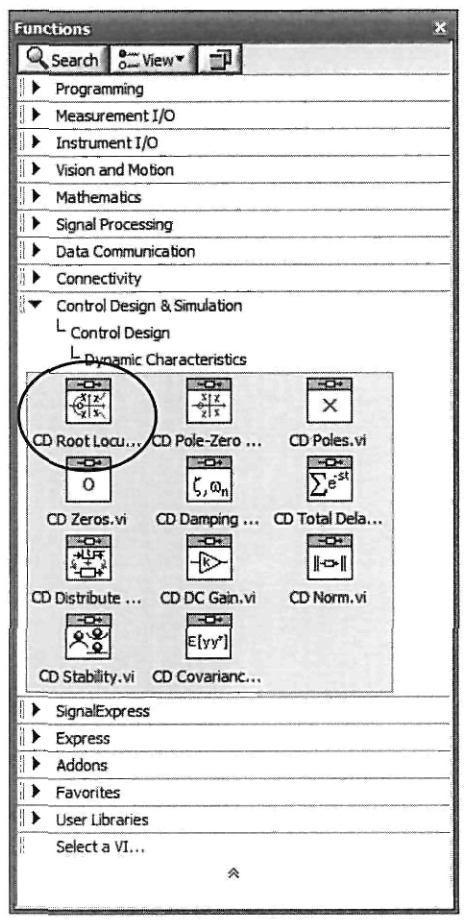


FIGURE D.13 **Functions** palette showing location of **Root Locus** block

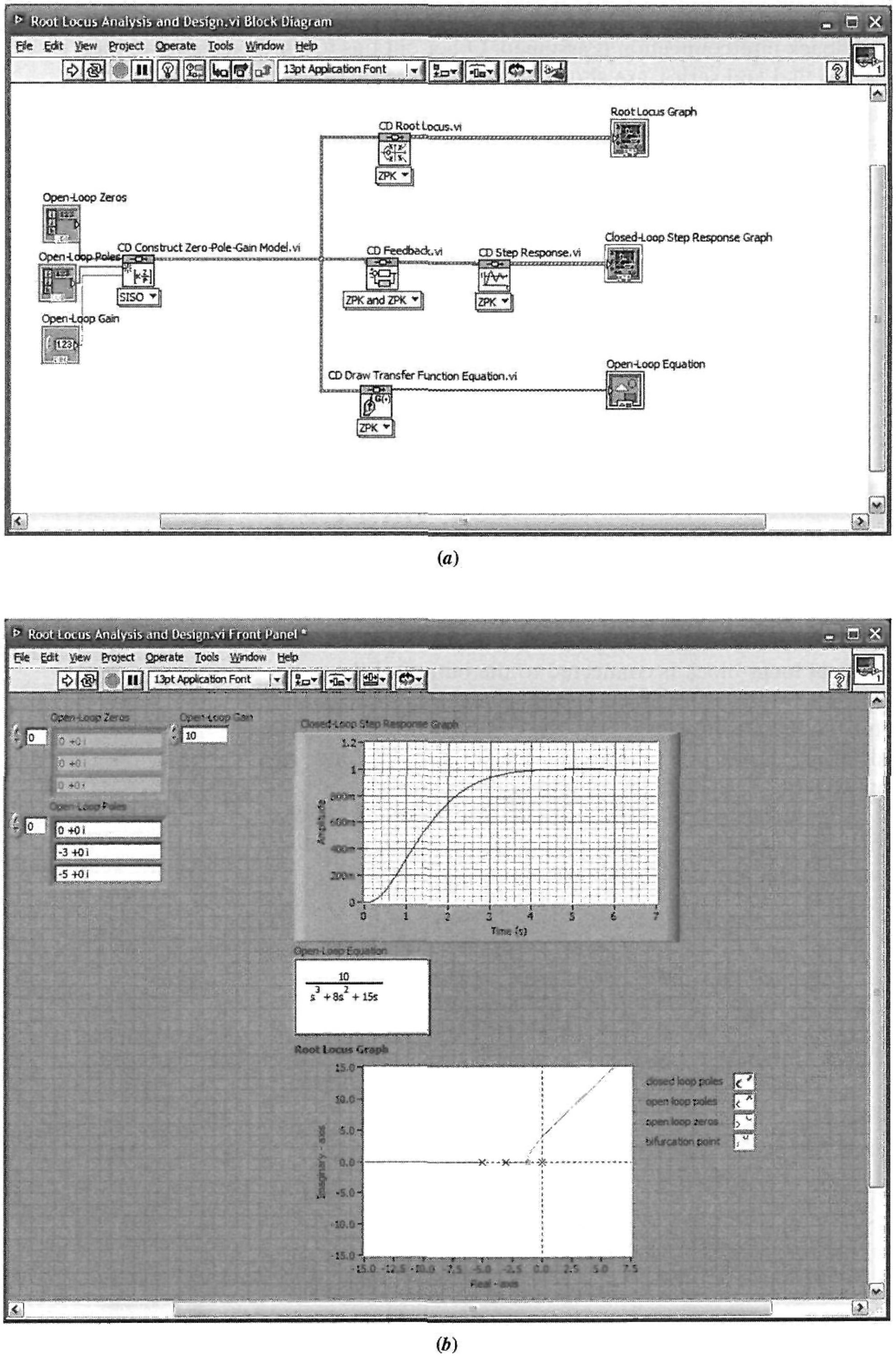


FIGURE D.14 Windows showing root locus analysis: **a. Block Diagram**; **b. Front Panel**

Example D.4

Open- and Closed-Loop Sinusoidal Frequency Analysis and Design

We can obtain open- and closed-loop sinusoidal frequency response curves by replacing the **Root Locus** block with the **Bode** block to yield the open-loop frequency response. A copy of the **Bode** block can be added at the output of the **Feedback** block to obtain the closed-loop frequency response. Figure D.15 shows where to obtain the **Bode** block.

Figure D.16 shows the **Block Diagram** and **Front Panel** with open- and closed-loop Bode analysis. In order to display the plots, the indicators shown at the outputs of the **Bode** blocks were created.

Figure D.15 shows other alternatives for frequency response analysis. For example, in addition to the Bode plots, you can create an indicator telling you the gain and phase margins by using the **Gain and Phase Margin** block. Figure D.17 shows that result.

Finally, if you need to use Nyquist or Nichols charts, the associated blocks are shown in Figure D.15 and can replace the **Bode** blocks.

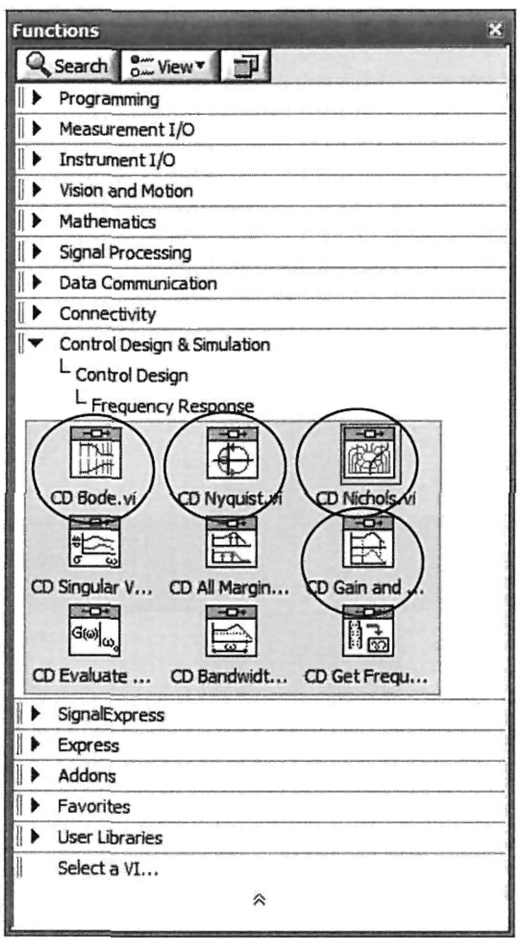
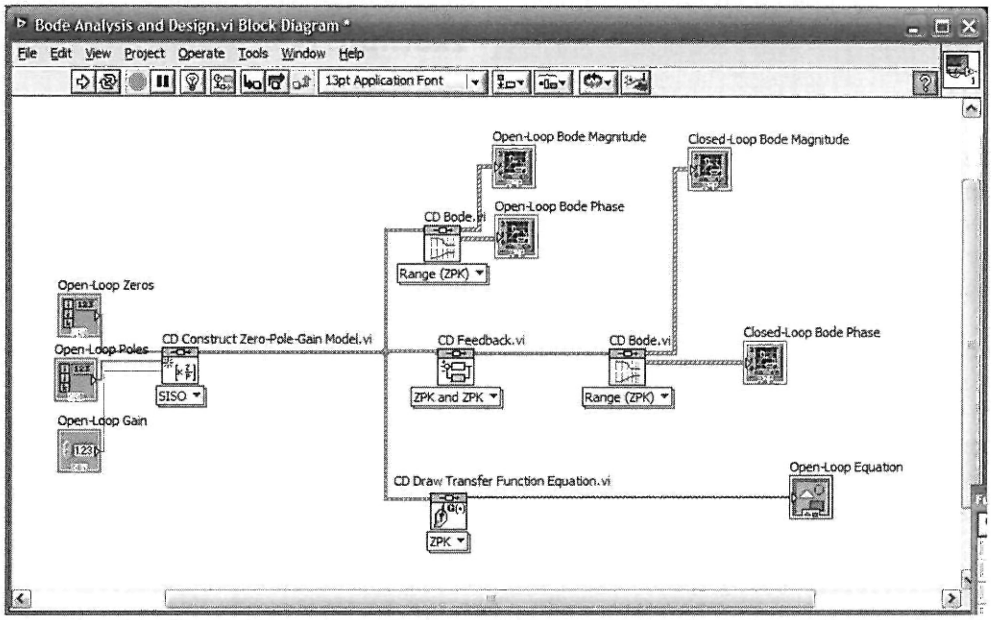
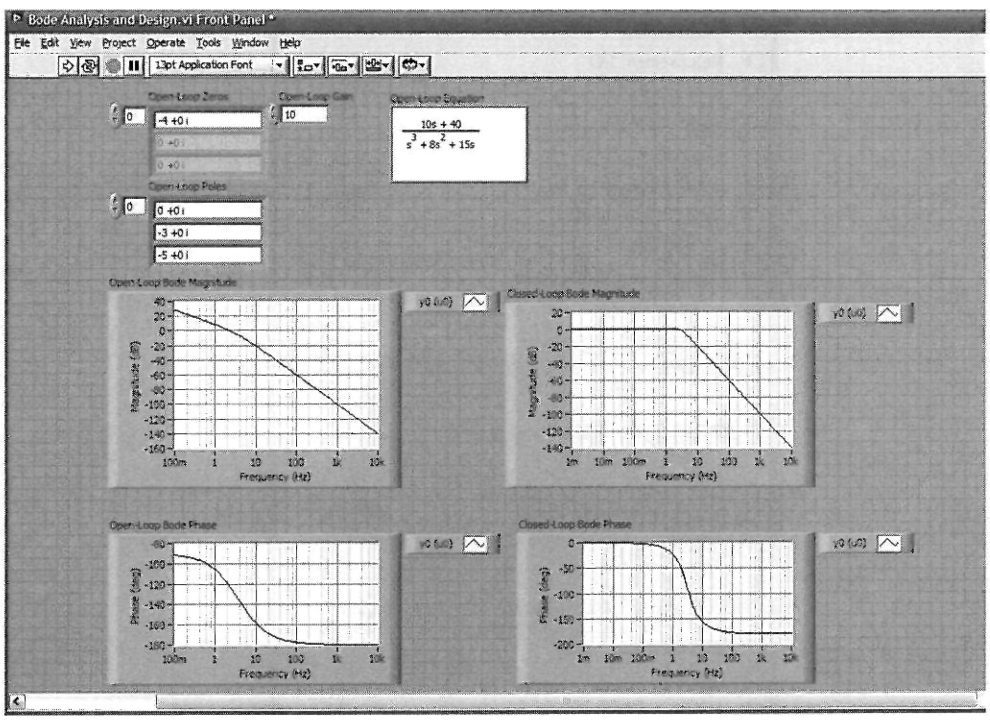


FIGURE D.15 **Functions** window showing frequency response blocks, such as **Bode**, **Nyquist**, **Nichols**, and **Gain and Phase Margin** blocks

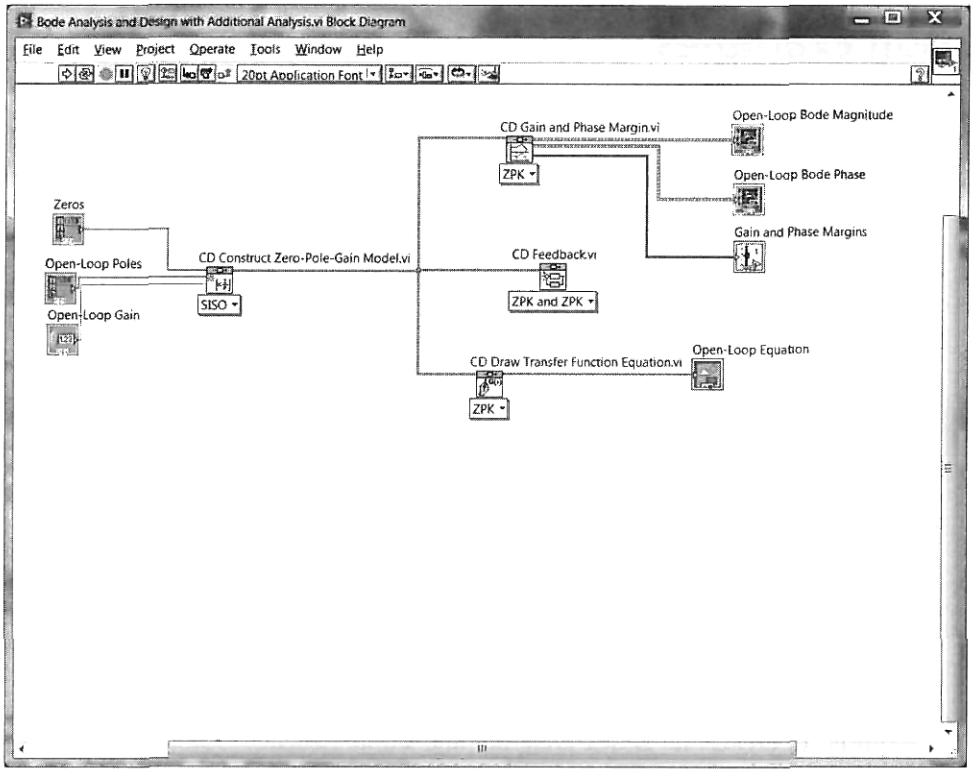


(*a*)

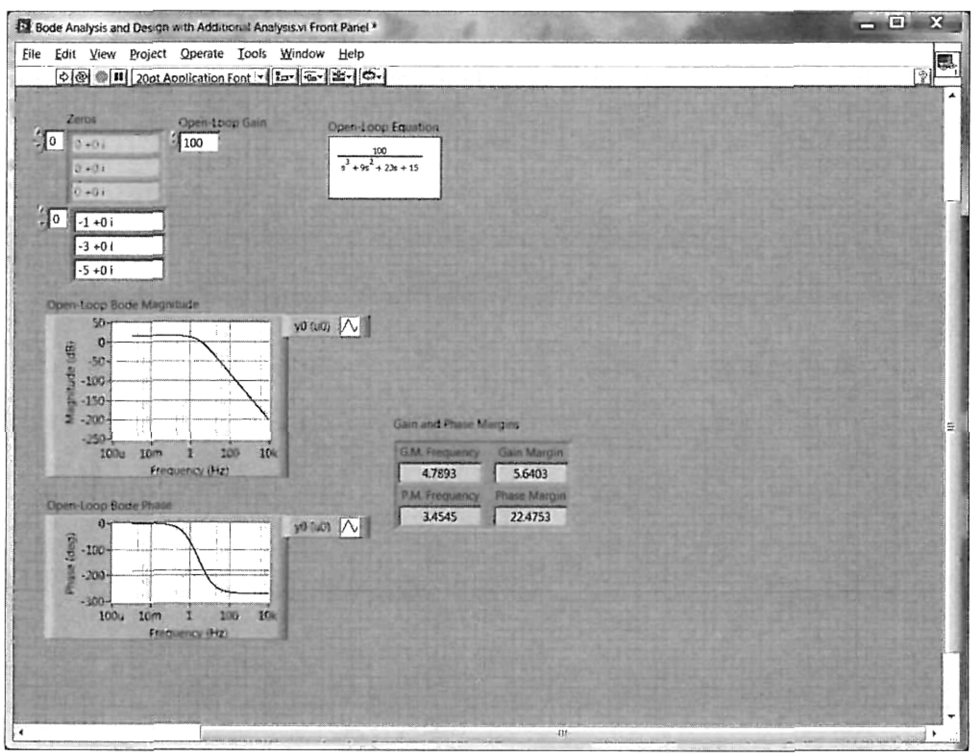


(*b*)

FIGURE D.16 Bode analysis via LabVIEW: **a. Block Diagram**; **b. From Panel**



(*a*)



(*b*)

FIGURE D.17 Bode analysis with gain and phase margin: **a. Block Diagram**; **b. Front Panel**

D.5 Simulation Examples

Whereas the LabVIEW block sequence for design and analysis is analogous to following the code statement sequence in a MATLAB M-file, the LabVIEW block sequence for simulation is analogous to following the block sequence of a Simulink diagram.

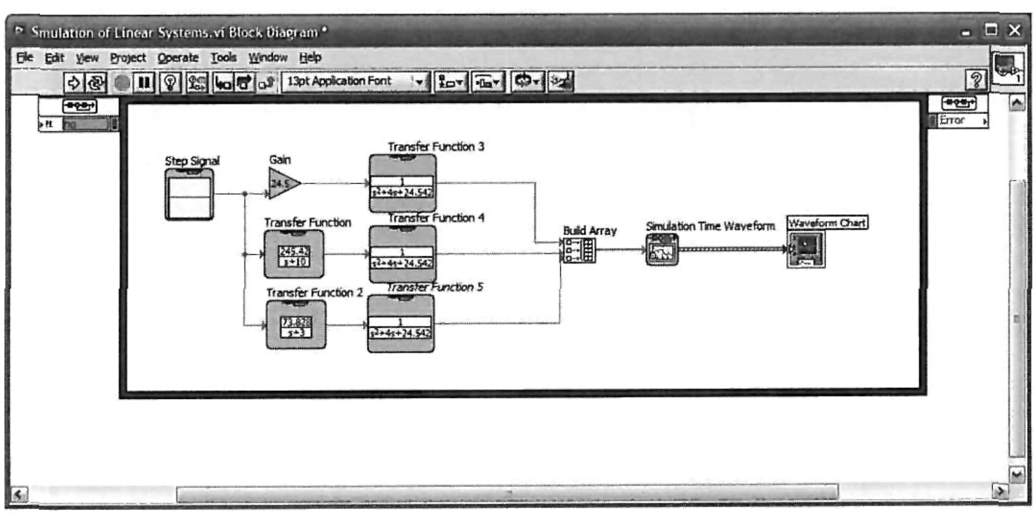
In this section, we show examples of simulation using LabVIEW. For control system simulation, icons for the block diagram are taken from the **Simulation** subpalette under the **Control Design and Simulation** palette. Our examples will parallel the examples shown in Appendix C which uses Simulink.

Example D.5

Simulation of Linear Systems

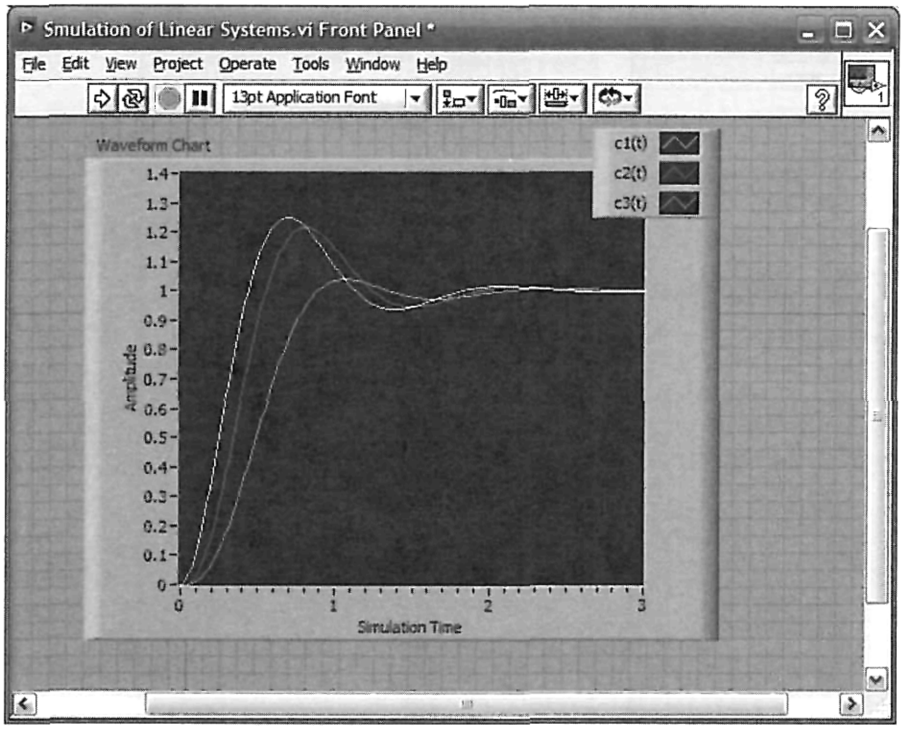
Create Block Diagram and Front Panel Figure D.18 shows the **Block Diagram** and **Front Panel** for simulating a linear system. The simulation reproduces Example C.1 in Appendix C, which uses Simulink. Blocks are selected from the **Simulation** subpalette under the **Control Design and Simulation** palette and must be placed within the **Simulation Loop** obtained from **Functions/Control Design and Simulation/Simulation/Simulation Loop**. We now enumerate the detailed steps required to create the **Block Diagram** and **Front Panel:**

1. Transfer functions are obtained from **Functions/Control Design and Simulation/Simulation/Continuous Linear Systems/Transfer Function**. Right-click on each transfer function and select **Configuration** to enter the parameter values shown in Figure D.18(a) or equivalently in Figure C.5
2. The gain block is obtained from **Functions/Control Design and Simulation/Simulation/Signal Arithmetic/Gain**. Right-click on the gain block and select **Configuration** to enter the parameter value.



(a)

FIGURE D.18 Simulation of linear systems: **a. Block Diagram**; (*figure continues*)



(*b*)

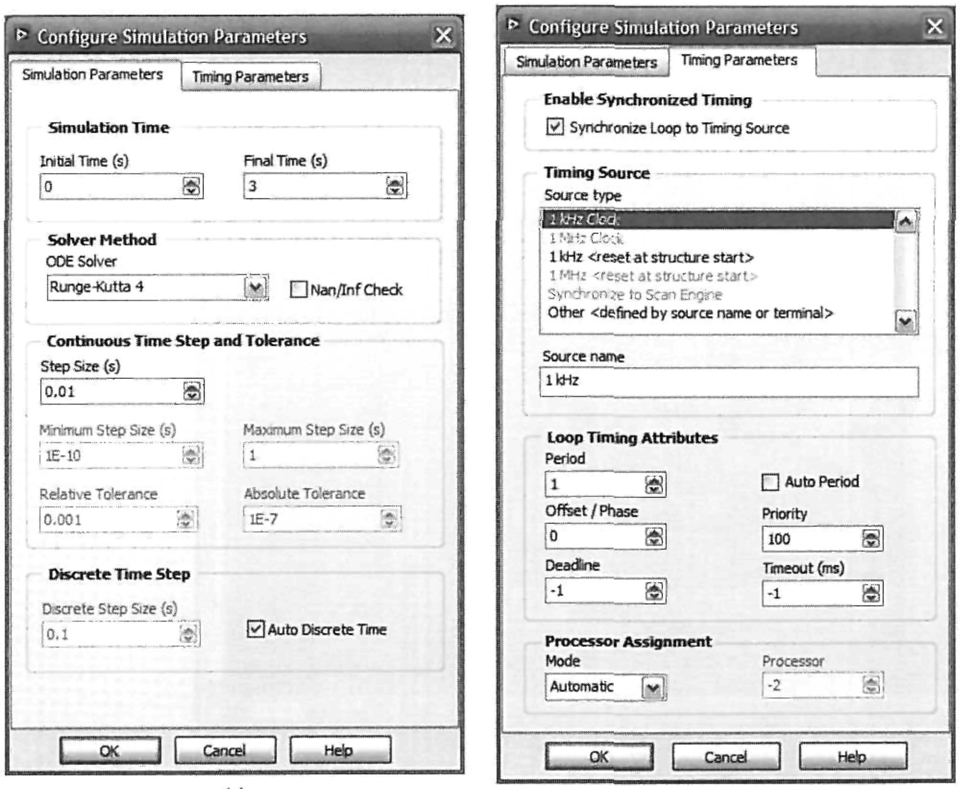
FIGURE D.18 (*Continued*) **b. Front Panel**

3. The step-input block is obtained from **Functions/Control Design and Simulation/Simulation/Signal Generation/Step Signal**. Right-click on the gain block and select **Configuration** to enter the parameter value.
4. In order to display the three step-response curves simultaneously, we use a **Build Array** block obtained from **Functions/Programming/Array/Build Array**. Drag the bottom of the icon to expose the correct number of inputs three for this case).
5. To create the display, we use the **Simulation Time Waveform** block obtained from **Functions/Control Design and Simulation/Simulation/Graph Utilities/ Simtime Waveform**. Right-click the output of the **Simtime Waveform** block and select **Create/Indicator** to produce the **Waveform Chart** icon and the **Front Panel** display.

Configure simulation loop Finally, set the simulation parameters by right-clicking the **Simulation Loop** and selecting **Configure Simulation Parameters** . . . Set the parameters as shown in Figure D.19.

Configure graph parameters On the **Front Panel**, right-click the graph and select **Properties** to configure graph parameters if required. Select the legend and expand it vertically to expose all three plot identities. The titles in the legend can be changed to reflect meaningful labels for the plots.

Run the simulation Perform the simulation by clicking the arrow at the extreme left of the toolbar on the **Front Panel** window. You can erase curves between trials by right-clicking the display and selecting **Data Operations/Clear Chart**.



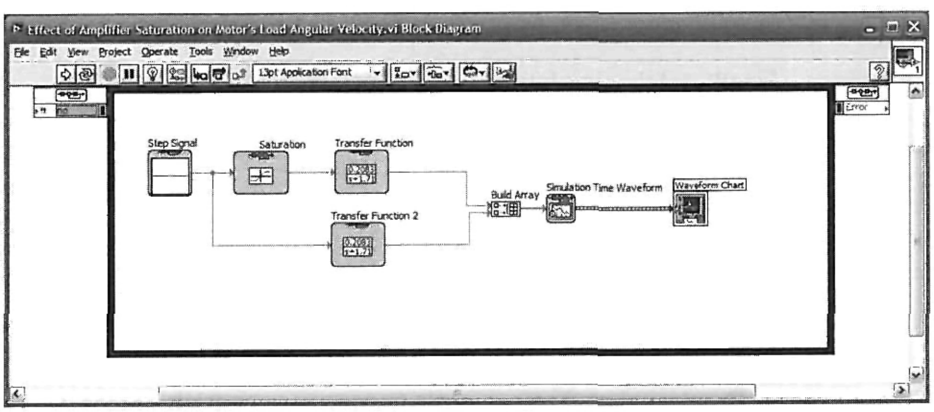
(*a*) (*b*)

FIGURE D.19 Configuring the **Simulation Loop** parameters: **a.** Simulation parameters; **b.** Timing parameters

Example D.6

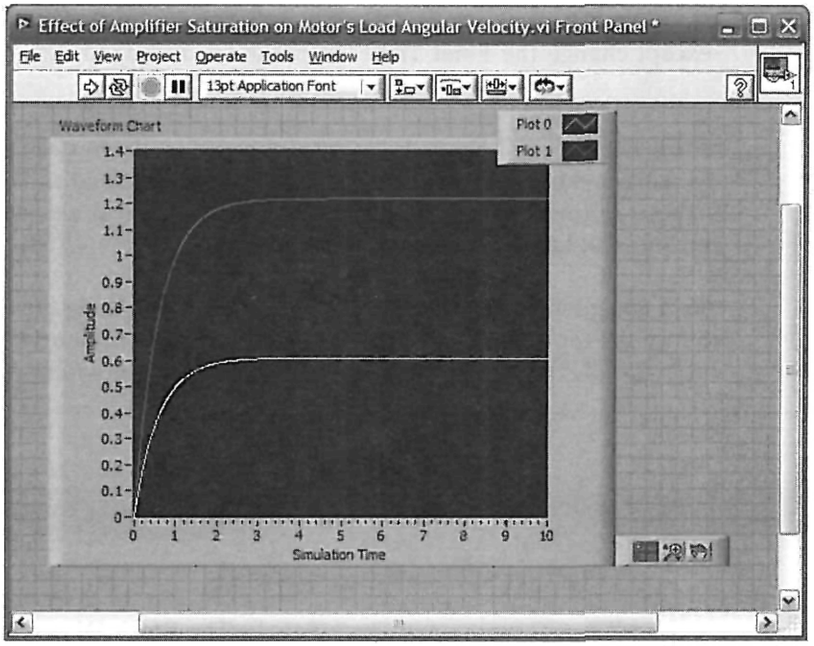
Effect of Amplifier Saturation on Motor's Load Angular Velocity

Create Block Diagram and Front Panel The **Block Diagram and Front Panel** for simulating a dc motor with and without saturation are shown in Figure D.20. The **Saturation** block is obtained from **Control Design & Simulation/Simulation/Nonliner Systems/Saturation**.



(*a*)

FIGURE D.20 Simulation of a dc motor with and without saturation: **a. Block Diagram**; (*figure continues*)



(*b*)

FIGURE D.20 (*Continued*) **b. Front Panel**

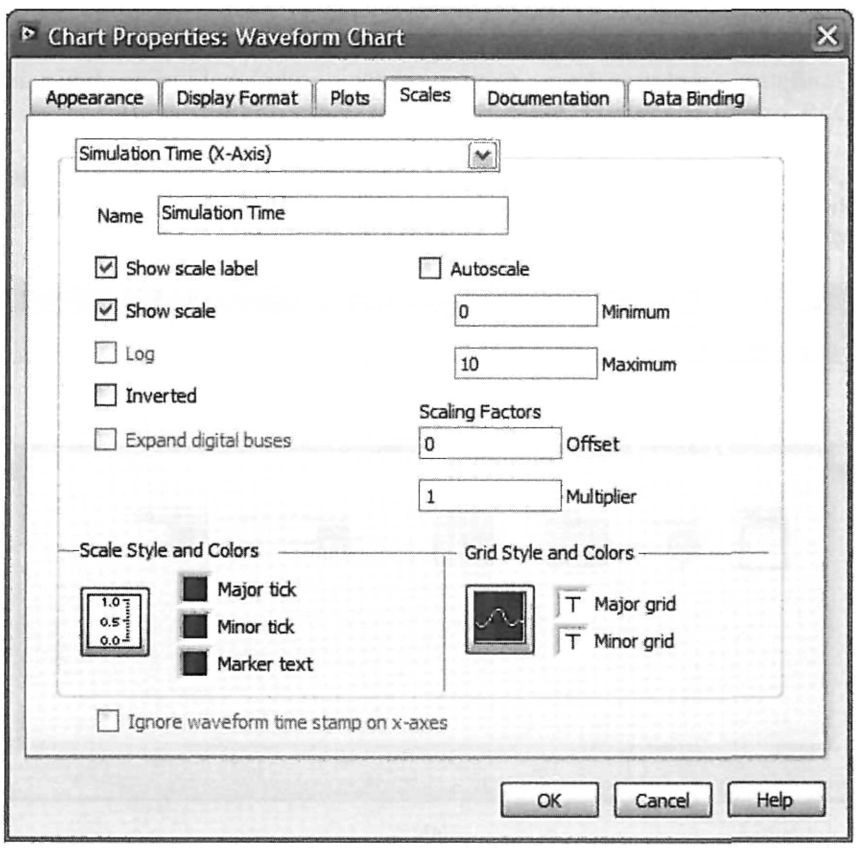


FIGURE D.21 **Chart Properties: Waveform Chart** Window

Configure simulation loop Configure the simulation loop as shown in figure D.19, except change the **Final Time (s)** in Figure D.19(*a*) to 10.

Configure graph parameters On the **Front Panel**, right click the graph and select **Properties** to configure graph parameters. Select the **Scales** tab and enter 10 in the **Maximum** box as shown in Figure D.21. Select the legend and expand it vertically to expose both plot identities. The titles in the legend can be changed to reflect meaningful labels for the plots.

Run the simulation Perform the simulation by clicking the arrow at the extreme left of the toolbar on the **Front Panel** window. You can erase curves between trials by right-clicking the display and selecting **Data Operations/Clear Chart**.

Example D.7

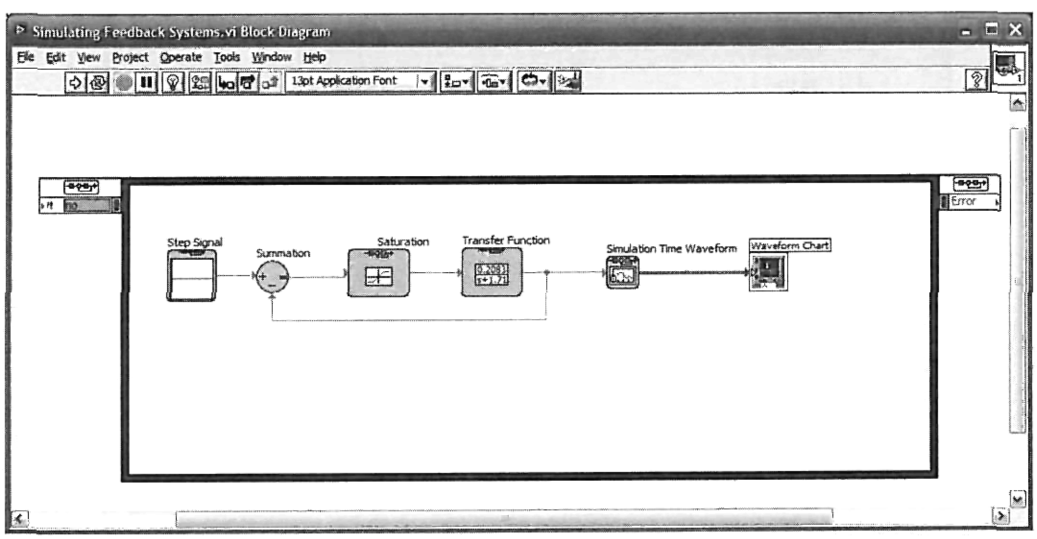
Simulating Feedback Systems

Create Block Diagram and Front Panel The **Block Diagram** and **Front Panel** for simulating feedback systems is shown in Figure D.22. The **Summation** block is obtained from **Control Design & Simulation/Simulation/Signal Arithmetic/Summation**.

Configure Summation and other blocks Right-click the **Summation** block and select **Configuration** . . . Repeat for other blocks.

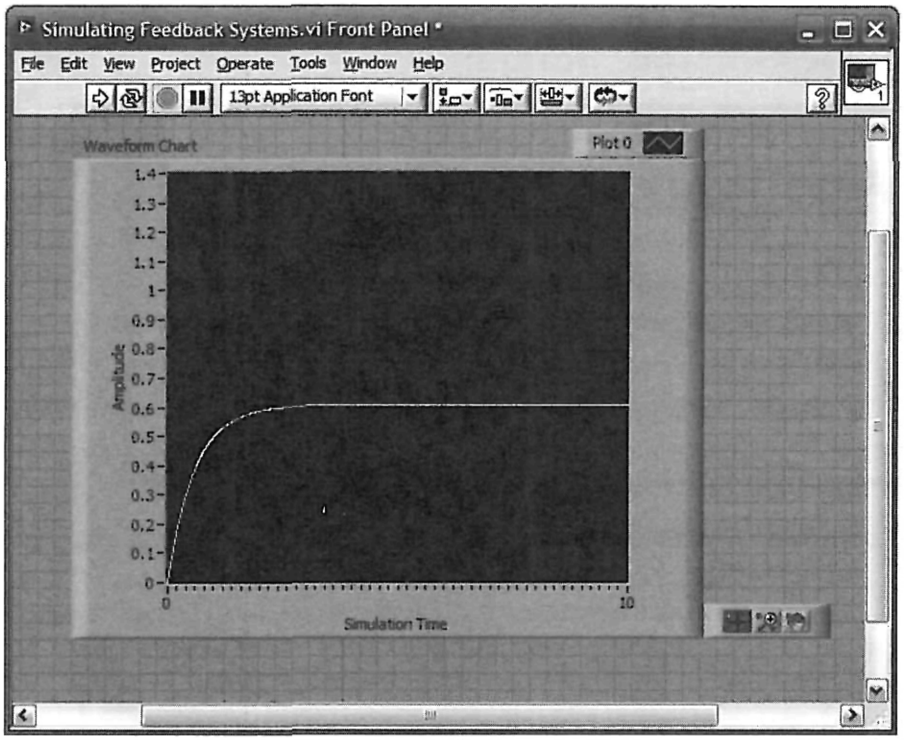
Configure simulation loop Configure the simulation loop as shown in Figure D.19, except change the **Final Time (s)** in Figure D.19(*a*) to 10.

Configure graph parameters On the **Front Panel**, right click the graph and select **Properties** to configure graph parameters. Select the **Scales** tab and enter 10 in the **Maximum** box as shown in Figure D.21.



(*a*)

FIGURE D.22 Simulation of feedback systems: **a. Block Diagram**; (*figure continues*)



(*b*)

FIGURE D.22 (*Continued*) **b. Front Panel**

Run the simulation Perform the simulation by clicking the arrow at the extreme left of the toolbar on the **Front Panel** window. You can erase curves between trials by right-clicking the display and selecting **Data Operations/Clear Chart**.

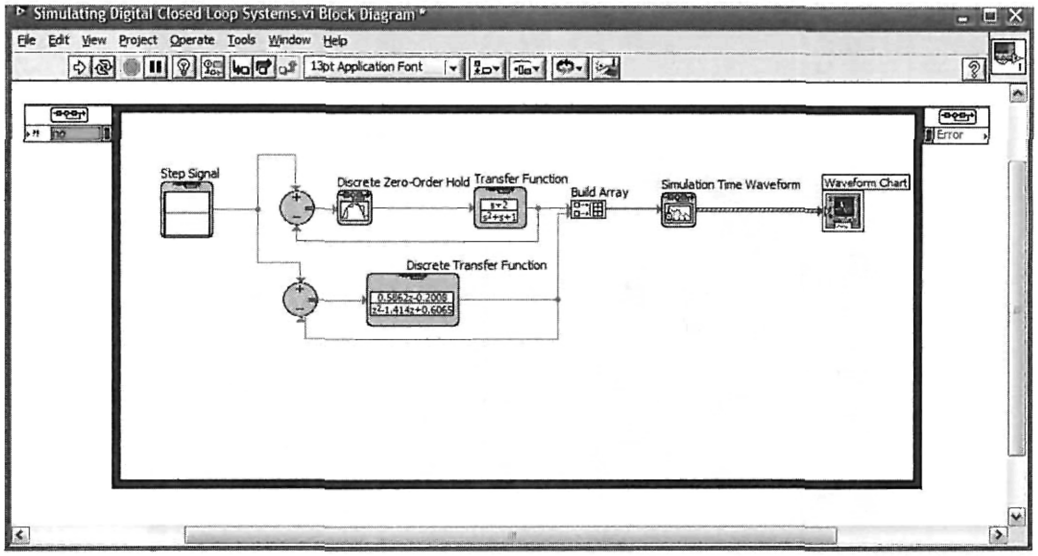
Example D.8

Simulating Digital Systems with the Simulation Palette

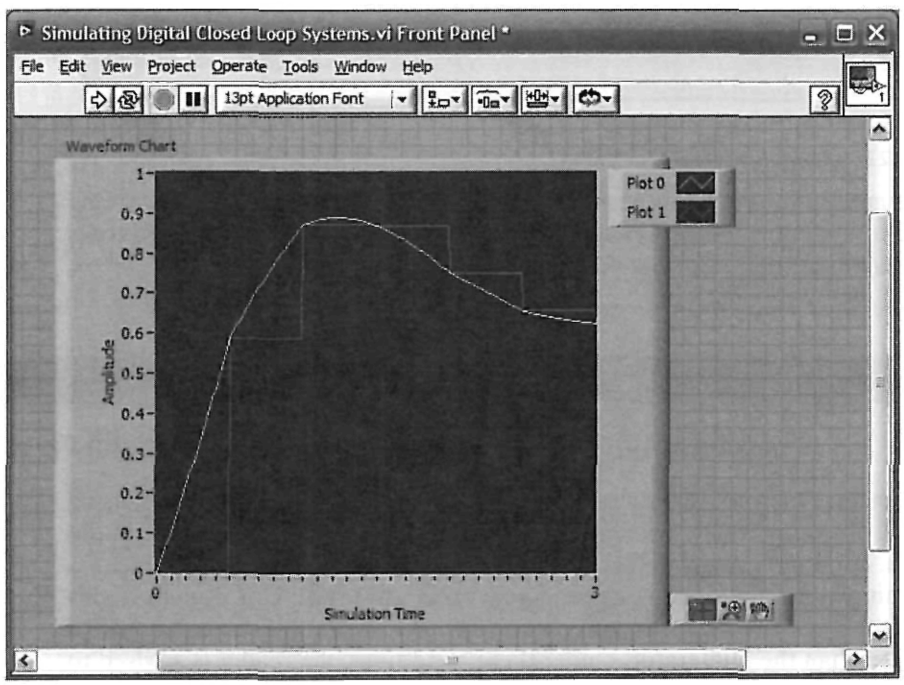
Digital systems, such as Example C.4 in Appendix C, can be simulated using LabVIEW. However, there are restrictions on the transfer functions used in the simulation. LabVIEW requires that all inputs to the transfer functions be present at the beginning of the simulation or else a cycle error will result. Unfortunately, this requirement limits the use of transfer functions to those with a denominator of higher order than the numerator. Under these conditions, the reader is advised to use either MATLAB or the **Control Design** palette rather than the **Simulation** palette of the **Control Design & Simulation** function.

Our first digital example will simulate a digital feedback system using the **Simulation** palette with proper transfer functions. The next example will simulate Example C.4 in Appendix C, which does not have proper transfer functions, using LabVIEW's **Control Design** palette.

Create Block Diagram and Front Panel The **Block Diagram** and **Front Panel** for simulating digital systems is shown in Figure D.23. The **Discrete Zero-Order Hold** block is obtained from **Control Design & Simulation/Simulation/Discrete Linear Systems/Discrete Zero-Order Hold**. The **Discrete Transfer Function** is obtained



(*a*)



(*b*)

FIGURE D.23 Simulation of digital systems with **Simulation** palette: **a. Block Diagram**; **b. Front Panel**

from **Control Design & Simulation/Simulation/Discrete Linear Systems/Discrete Transfer Function**.

Configure Discrete Zero-Order Hold and other blocks Right click the **Discrete Zero-Order Hold** block and select **Configuration . . .** Set the sample period to 0.5 second. Configure the transfer functions as shown on the **Block Diagram**. Configure the **Step Signal** to be a unit step.

Configure simulation loop Configure the simulation loop as shown in Figure D.19.

Configure graph parameters On the **Front Panel**, right click the graph and select **Properties** to configure graph parameters. Select the **Scales** tab and enter three in the **Maximum** box for both the x-and y-axes as shown in Figure D.21. Select the legend and expand it vertically to expose both plot identities. The titles in the legend can be changed to reflect meaningful labels for the plots.

Run the simulation Perform the simulation by clicking the arrow at the extreme left of the toolbar on the **Front Panel** window. You can erase curves between trials by right-clicking the display and selecting **Data Operations/Clear Chart**.

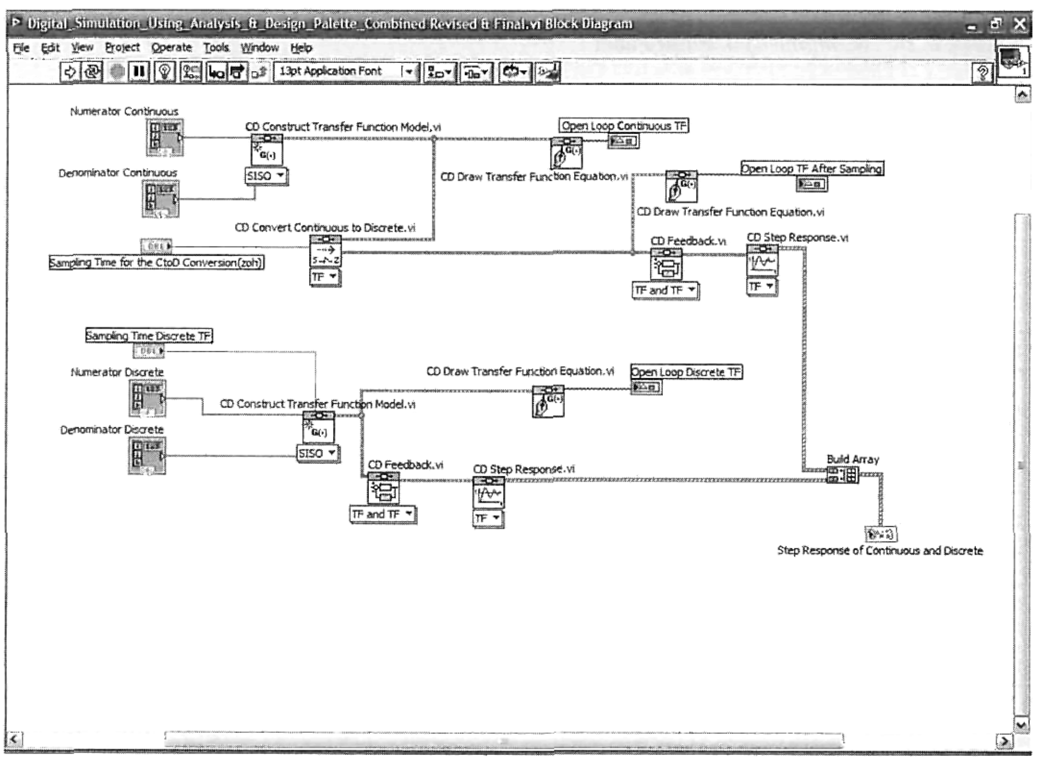
The simulation shows the difference in responses obtained by (1) modeling the digital system as a zero-order hold cascaded with a linear system (Plot 0), or (2) modeling the system with a digital transfer function (Plot 1).

Example D.9

Simulating Digital Systems with the Control Design Palette

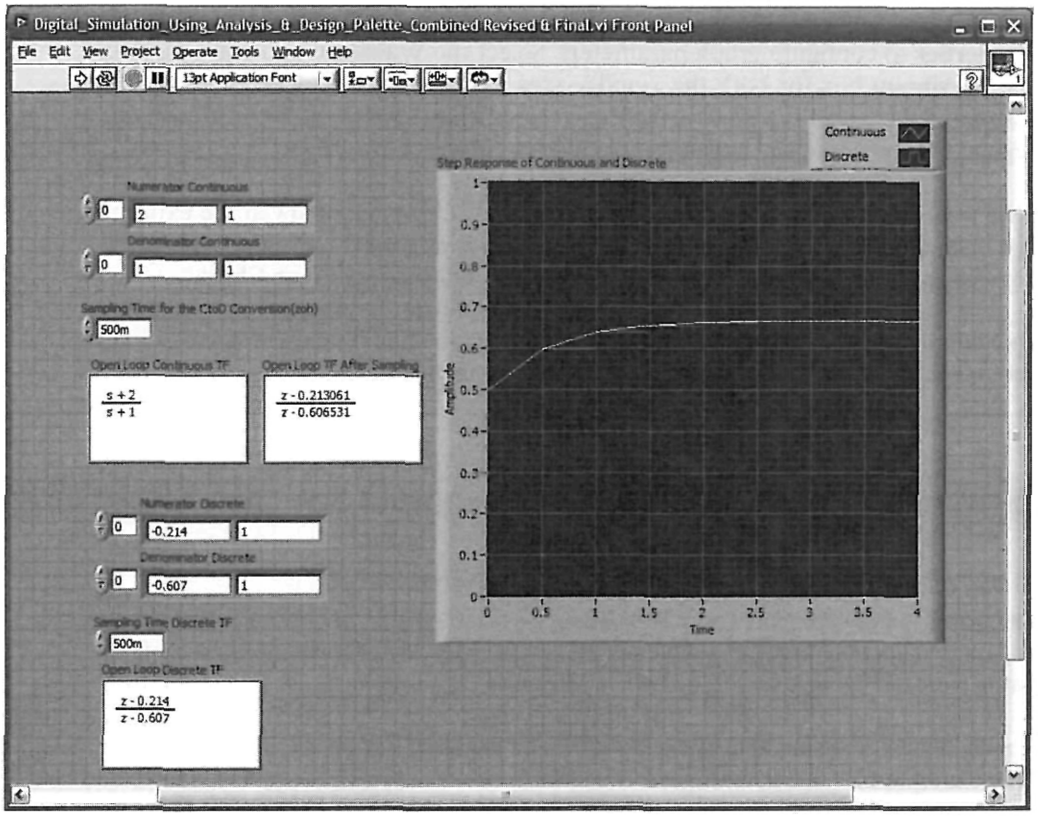
In order to avoid cycle errors in LabVIEW, we use the **Control Design** palette when we have transfer functions for which the numerator and denominator are of the same order. This example reproduces Simulink Example C.4.

Create Block Diagram and Front Panel The **Block Diagram** and **Front Panel** for this example are shown in Figure D.24. Wire the blocks as shown.



(a)

FIGURE D.24 Simulation of digital systems with **the Control Design** palette: **a. Block Diagram**; (*figure continues*)



(*b*)

FIGURE D.24 (*Continued*) **b. Front Panel**

Most of the blocks were previously discussed in Example D.1 and D.2. Digital transfer functions are created using the same blocks as continuous systems, but with a nonzero **Sampling Time(s)** input.

The **CD Convert Continuous to Discrete.vi**, is obtained from **Functions/Control Design & Simulation/Control Design/Model Conversion/CD Convert Continuous to Discrete.vi**.

The **Build Array** is obtained from **Functions/Programming/Array/Build Array**. Expand the **Build Array** block to show two inputs.

Configure parameters for Build Array Right-click on **Build Array** and select **Concatenate Inputs**. Right-click again on **Build Array** and select **Create/Indicator**.

Right-click the indicator on the front panel and select **Replace**. Using the resulting palettes as shown in Figure D.25, select the **XY Graph**.

On the front panel expand the legend to show two graphs. Title the legend components as shown. Change the x- and y-axes' starting and ending points as desired by right-clicking the graph and selecting **Properties**. In the **Properties** window, select **Scales** and enter the desired information.

Right-click the graph on the front panel and select **Data Operations** and make your current values the default. Also, right-click again and choose to reinitialize to your default values. You may also choose to clear the current plot.

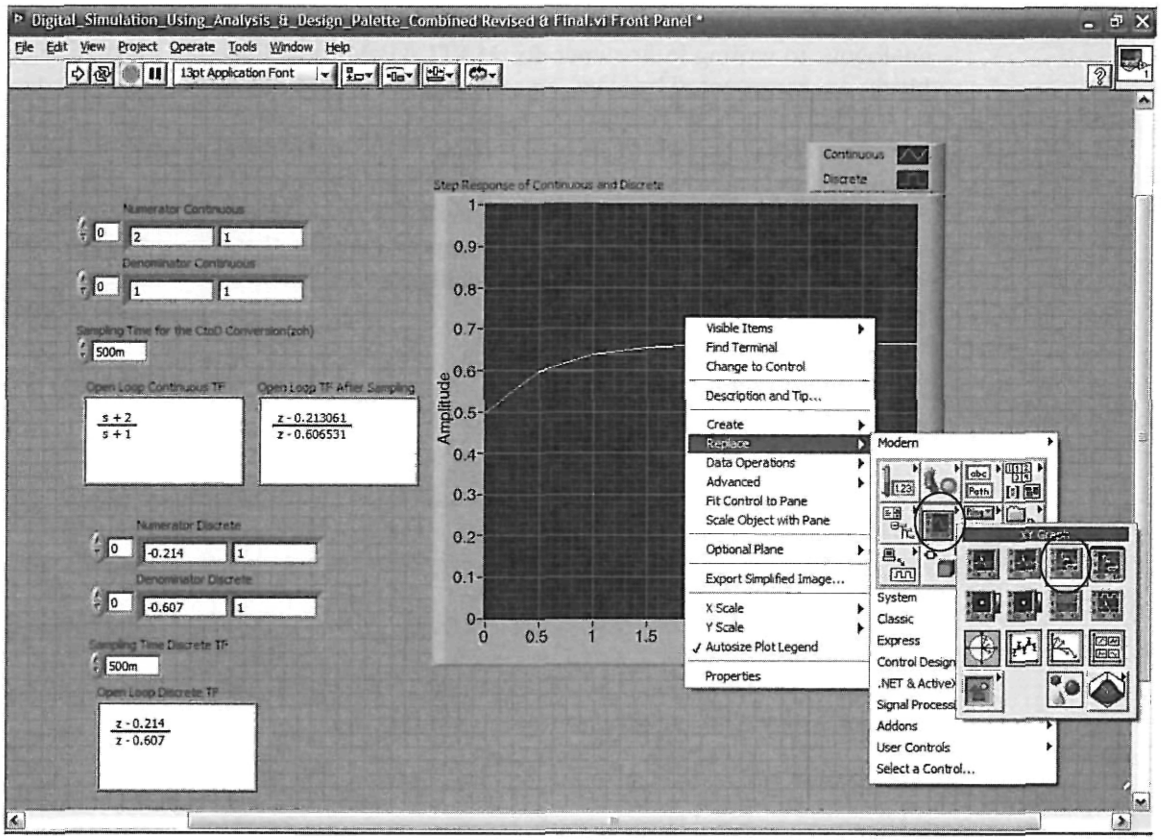


FIGURE D.25 Choosing **XYGraph**

Configure parameters for CD Convert Continuous to Discrete.vi Right-click and create a control for **Sample Time(s), Numerator**, and **Denominator** as described in Example D.1. Set the values as shown on the **Front Panel**.

Configure parameters for CD Construct Transfer Function Model.vi as a discrete model Right-click and create a control for **Sample Time(s)**, **Numerator**, and **Denominator** as described in Example 1. Set the values as shown on the **Front Panel**.

Configure parameters for all CD Draw Transfer Function Equation.vi Right-click and create a control for **Equation** as described in Example D.1. Set the values as shown on the **Front Panel**.

Run simulation See Example D.1 for a description. The results are shown in Figure D.24(*b*).

Summary

This appendix presented LabVIEW as an alternative to MATLAB for analysis, design, and simulation. Our discussion was divided into analysis and design, and simulation.

Analysis and design is performed by interconnecting code blocks, which is analogous to writing in-line code for MATLAB M-files. Since the LabVIEW code blocks are represented by icons, an advantage of using LabVIEW is that you do not have to know specific code statements.

Simulation is performed by interconnecting code blocks and is analogous to Simulink flow diagrams.

LabVIEW has more extensive applications than those covered here. You can create virtual instruments on your computer monitor that can operate external hardware and transmit and receive telemetric data. It is left to the interested reader to pursue these advanced topics.

Bibliography

National Instruments. *Getting Started with LabVIEW.* National Instruments, Austin, TX. 2003–2007.

National Instruments. *LabVIEW Fundamentals.* National Instruments, Austin, TX. 2003–2007.

National Instruments. *LabVIEW™ Control Design User Manual.* National Instruments, Austin, TX. 2004–2008.

National Instruments. *Introduction to LabVIEW in 3 Hours for Control Design and Simulation.* National Instruments Course Notes, Austin, TX.

Glossary

Acceleration constant $\lim_{s \to 0} s^2 G(s)$

Actuating signal The signal that drives the controller. If this signal is the difference between the input and output, it is called the *error*.

Analog-to-digital converter A device that converts analog signals to digital signals.

Armature The rotating member of a dc motor through which a current flows.

Back emf The voltage across the armature of a motor.

Bandwidth The frequency at which the magnitude frequency response is -3 dB below the magnitude at zero frequency.

Basis Linearly independent vectors that define a space.

Bilinear transformation A mapping of the complex plane where one point, s, is mapped into another point, z, through $z = (as + b)/(cs + d)$.

Block diagram A representation of the interconnection of subsystems that form a system. In a linear system, the block diagram consists of blocks representing subsystems, arrows representing signals, summing junctions, and pickoff points.

Bode diagram (plot) A sinusoidal frequency response plot where the magnitude response is plotted separately from the phase response. The magnitude plot is dB versus log ω, and the phase plot is phase versus log ω. In control systems, the Bode plot is usually made for the open-loop transfer function. Bode plots can also be drawn as straight-line approximations.

Branches Lines that represent subsystems in a signal-flow graph.

Break frequency A frequency where the Bode magnitude plot changes slope.

Breakaway point A point on the real axis of the s-plane where the root locus leaves the real axis and enters the complex plane.

Break-in point A point on the real axis of the s-plane where the root locus enters the real axis from the complex plane.

Characteristic equation The equation formed by setting the characteristic polynomial to zero.

Characteristic polynomial The denominator of a transfer function. Equivalently, the unforced differential equation, where the differential operators are replaced by s or λ.

Classical approach to control systems *See* **frequency domain techniques**.

Closed-loop system A system that monitors its output and corrects for disturbances. It is characterized by feedback paths from the output.

Closed-loop transfer function For a generic feedback system with $G(s)$ in the forward path and $H(s)$ in the feedback path, the closed-loop transfer function, $T(s)$, is $G(s)/[1 \pm G(s)H(s)]$, where the + is for negative feedback, and the – is for positive feedback.

Compensation The addition of a transfer function in the forward path or feedback path for the purpose of improving the transient or steady-state performance of a control system.

Compensator A subsystem inserted into the forward or feedback path for the purpose of improving the transient response or steady-state error.

Constant M circles The locus of constant, closed-loop magnitude frequency response for unity feedback systems. It allows the closed-loop magnitude frequency response to be determined from the open-loop magnitude frequency response.

Constant N circles The locus of constant, closed-loop phase frequency response for unity feedback systems. It allows the closed-loop phase frequency response to be determined from the open-loop phase frequency response.

Controllability A property of a system by which an input can be found that takes every state variable from a desired initial state to a desired final state in finite time.

Controlled variable The output of a plant or process that the system is controlling for the purpose of desired transient response, stability, and steady-state error characteristics.

Controller The subsystem that generates the input to the plant or process.

Critically damped response The step response of a second-order system with a given natural frequency that is characterized by no overshoot and a rise time that is faster than any possible overdamped response with the same natural frequency.

Damped frequency of oscillation The sinusoidal frequency of oscillation of an underdamped response.

Damping ratio The ratio of the exponential decay frequency to the natural frequency.

Decade Frequencies that are separated by a factor of 10.

Decibel (dB) The decibel is defined as 10 $\log P_G$, where P_G is the power gain of a signal. Equivalently, the decibel is also 20 $\log V_G$, where V_G is the voltage gain of a signal.

Decoupled system A state-space representation in which each state equation is a function of only one state variable. Hence, each differential equation can be solved independently of the other equations.

Digital compensator A sampled transfer function used to improve the response of computer-controlled feedback systems. The transfer function can be emulated by a digital computer in the loop.

Digital-to-analog converter A device that converts digital signals to analog signals.

Disturbance An unwanted signal that corrupts the input or output of a plant or process.

Dominant poles The poles that predominantly generate the transient response.

Eigenvalues Any value, λ_i, that satisfies $\mathbf{AX_i} = \lambda_i \mathbf{x_i}$ for $\mathbf{x_i} \neq 0$. Hence, any value, λ_i, that makes $\mathbf{x_i}$ an eigenvector under the transformation $\mathbf{A}$.

Eigenvector Any vector that is collinear with a new basis vector after a similarity transformation to a diagonal system.

Electric circuit analog An electrical network whose variables and parameters are analogous to another physical system. The electric circuit analog can be used to solve for variables of the other physical system.

Electrical admittance The inverse of electrical impedance. The ratio of the Laplace transform of the current to the Laplace transform of the voltage.

Electrical impedance The ratio of the Laplace transform of the voltage to the Laplace transform of the current.

Equilibrium The steady-state solution characterized by a constant position or oscillation.

Error The difference between the input and the output of a system.

Euler's approximation A method of integration where the area to be integrated is approximated as a sequence of rectangles.

Feedback A path through which a signal flows back to a previous signal in the forward path in order to be added or subtracted.

Feedback compensator A subsystem placed in a feedback path for the purpose of improving the performance of a closed-loop system.

Forced response For linear systems, that part of the total response function due to the input. It is typically of the same form as the input and its derivatives.

Forward-path gain The product of gains found by traversing a path that follows the direction of signal flow from the input node to the output node of a signal-flow graph.

Frequency domain techniques A method of analyzing and designing linear control systems by using transfer functions and the Laplace transform as well as frequency response techniques.

Frequency response techniques A method of analyzing and designing control systems by using the sinusoidal frequency response characteristics of a system.

Gain The ratio of output to input; usually used to describe the amplification in the steady state of the magnitude of sinusoidal inputs, including dc.

Gain margin The amount of additional open-loop gain, expressed in decibels (dB), required at 180° of phase shift to make the closed-loop system unstable.

Gain-margin frequency The frequency at which the phase frequency response plot equals 180°. It is the frequency at which the gain margin is measured.

Homogeneous solution *See* **natural response.**

Ideal derivative compensator *See* **proportional-plus-derivative controller.**

Ideal integral compensator *See* **proportional-plus-integral controller.**

Instability *The characteristic of a system defined by a natural response that grows* without bounds as time approaches infinity.

Kirchhoff's law The sum of voltages around a closed loop equals zero. Also, the sum of currents at a node equals zero.

Lag compensator A transfer function, characterized by a pole on the negative real axis close to the origin and a zero close and to the left of the pole, that is used for the purpose of improving the steady-state error of a closed-loop system.

Lag-lead compensator A transfer function, characterized by a pole-zero configuration that is the combination of a lag and a lead compensator, that is used for the purpose of improving both the transient response and the steady-state error of a closed-loop system.

Laplace transformation A transformation that transforms linear differential equations into algebraic expressions. The transformation is especially useful for modeling, analyzing, and designing control systems as well as solving linear differential equations.

Lead compensator A transfer function, characterized by a zero on the negative real axis and a pole to the left of the zero, that is used for the purpose of improving the transient response of a closed-loop system.

Linear combination A linear combination of n variables, x_i, for $i = 1$ to n, given by the following sum, S:

$$S = K_n X_n + K_{n-1} X_{n-1} + \cdots + K_1 X_1$$

where each K_i is a constant.

Linear independence The variables x_i, for $i = 1$ to n, are said to be linearly independent if their linear combination, S, equals zero *only* if every $K_i = 0$ and *no* $x_i = 0$. Alternatively, if the x_i's are linearly independent, then $K_n x_n + K_{n-1} x_{n-1} + \cdots + K_1 x_1 = 0$ cannot be solved for any x_k. Thus, no x_k can be expressed as a linear combination of the other x_i's.

Linear system A system possessing the properties of superposition and homogeneity.

Linearization The process of approximating a nonlinear differential equation with a linear differential equation valid for small excursions about equilibrium.

Loop gain For a signal-flow graph, the product of branch gains found by traversing a path that starts at a node and ends at the same node without passing through any other node more than once, and following the direction of the signal flow.

Major-loop compensation A method of feedback compensation that adds a compensating zero to the open-loop transfer function for the purpose of improving the transient response of the closed-loop system.

Marginal stability The characteristic of a system defined by a natural response that neither decays nor grows, but remains constant or oscillates as time approaches infinity as long as the input is not of the same form as the system's natural response.

Mason's rule A formula from which the transfer function of a system consisting of the interconnection of multiple subsystems can be found.

Mechanical rotational impedance The ratio of the Laplace transform of the torque to the Laplace transform of the angular displacement.

Mechanical translational impedance The ratio of the Laplace transform of the force to the Laplace transform of the linear displacement.

Minor-loop compensation A method of feedback compensation that changes the poles of a forward-path transfer function for the purpose of improving the transient response of the closed-loop system.

Modern approach to control systems *See* **state-space representation.**

Natural frequency The frequency of oscillation of a system if all the damping is removed.

Natural response That part of the total response function due to the system and the way the system acquires or dissipates energy.

Negative feedback The case where a feedback signal is subtracted from a previous signal in the forward path.

Newton's law The sum of forces equals zero. Alternatively, after bringing the ma force to the other side of the equality, the sum of forces equals the product of mass and acceleration.

Nichols chart The locus of constant closed-loop magnitude and closed-loop phase frequency responses for unity feedback systems plotted on the open-loop dB versus phase-angle plane. It allows the closed-loop frequency response to be determined from the open-loop frequency response.

Nodes Points in a signal-flow diagram that represent signals.

No-load speed The speed produced by a motor with constant input voltage when the torque at the armature is reduced to zero.

Nonminimum-phase system A system whose transfer function has zeros in the right half-plane. The step response is characterized by an initial reversal in direction.

Nontouching-loop gain The product of loop gains from nontouching loops taken two, three, four, and so on at a time.

Nontouching loops Loops that do not have any nodes in common.

Notch filter A filter whose magnitude frequency response dips at a particular sinusoidal frequency. On the s-plane, it is characterized by a pair of complex zeros near the imaginary axis.

Nyquist criterion If a contour, A, that encircles the entire right half-plane is mapped through $G(s)H(s)$, then the number of closed-loop poles, Z, in the right half-plane equals the number of open-loop poles, P, that are in the right half-plane minus the number of counterclockwise revolutions, N, around -1, of the mapping; that is, $Z = P - N$. The mapping is called the *Nyquist diagram* of $G(s)H(s)$.

Nyquist diagram (plot) A polar frequency response plot made for the open-loop transfer function.

Nyquist sampling rate The minimum frequency at which an analog signal should be sampled for correct reconstruction. This frequency is twice the bandwidth of the analog signal.

Observability A property of a system by which an initial state vector, $x(t_0)$, can be found from $u(t)$ and $y(t)$ measured over a finite interval of time from t_0. Simply stated, observability is the property by which the state variables can be estimated from a knowledge of the input, $u(t)$, and output, $y(t)$.

Observer A system configuration from which inaccessible states can be estimated.

Octave Frequencies that are separated by a factor of two.

Ohm's law For dc circuits the ratio of voltage to current is a constant called resistance.

Open-loop system A system that does not monitor its output nor correct for disturbances.

Open-loop transfer function For a generic feedback system with $G(s)$ in the forward path and $H(s)$ in the feedback path, the open-loop transfer function is the product of the forward-path transfer function and the feedback transfer function, or $G(s)H(s)$.

Operational amplifier An amplifier—characterized by a very high input impedance, a very low output impedance, and a high gain—that can be used to implement the transfer function of a compensator.

Output equation For linear systems, the equation that expresses the output variables of a system as linear combinations of the state variables.

Overdamped response A step response of a second-order system that is characterized by no overshoot.

Partial-fraction expansion A mathematical equation where a fraction with n factors in its denominator is represented as the sum of simpler fractions.

Particular solution *See* **forced response.**

Passive network A physical network that only stores or dissipates energy. No energy is produced by the network.

Peak time, T_p The time required for the underdamped step response to reach the first, or maximum, peak.

Percent overshoot, $\%OS$ The amount that the underdamped step response overshoots the steady-state, or final, value at the peak time, expressed as a percentage of the steady-state value.

Phase margin The amount of additional open-loop phase shift required at unity gain to make the closed-loop system unstable.

Phase-margin frequency The frequency at which the magnitude frequency response plot equals zero dB. It is the frequency at which the phase margin is measured.

Phase variables State variables such that each subsequent state variable is the derivative of the previous state variable.

Phasor A rotating vector that represents a sinusoid of the form $A\cos(\omega t + \phi)$.

Pickoff point A block diagram symbol that shows the distribution of one signal to multiple subsystems.

Plant or process The subsystem whose output is being controlled by the system.

Poles (1) The values of the Laplace transform variable, s, that cause the transfer function to become infinite; and (2) any roots of factors of the characteristic equation in the denominator that are common to the numerator of the transfer function.

Position constant $\lim_{s\to 0} G(s)$

Positive feedback The case where a feedback signal is added to a previous signal in the forward path.

Proportional-plus-derivative (PD) controller A controller that feeds forward to the plant a proportion of the actuating signal plus its derivative for the purpose of improving the transient response of a closed-loop system.

Proportional-plus-integral (PI) controller A controller that feeds forward to the plant a proportion of the actuating signal plus its integral for the purpose of improving the steady-state error of a closed-loop system.

Proportional-plus-integral-plus-derivative (PID) controller A controller that feeds forward to the plant a proportion of the actuating signal plus its integral plus its derivative for the purpose of improving the transient response and steady-state error of a closed-loop system.

Quantization error For linear systems, the error associated with the digitizing of signals as a result of the finite difference between quantization levels.

Raible's tabular method A tabular method for determining the stability of digital systems that parallels the Routh-Hurwitz method for analog signals.

Rate gyro A device that responds to an angular position input with an output voltage proportional to angular velocity.

Residue The constants in the numerators of the terms in a partial-fraction expansion.

Rise time, T_r The time required for the step response to go from 0.1 of the final value to 0.9 of the final value.

Root locus The locus of closed-loop poles as a system parameter is varied. Typically, the parameter is gain. The locus is obtained from the open-loop poles and zeros.

Routh-Hurwitz criterion A method for determining how many roots of a polynomial in s are in the right half of the s-plane, the left half of the s-plane, and on the imaginary axis. Except in some special cases, the Routh-Hurwitz criterion does not yield the coordinates of the roots.

Sensitivity The fractional change in a system characteristic for a fractional change in a system parameter.

Settling time, T_s The amount of time required for the step response to reach and stay within $\pm 2\%$ of the steady-state value. Strictly speaking, this is the definition of the 2% settling time. Other percentages, for example 5%, also can be used. This book uses the 2% settling time.

Signal-flow graph A representation of the interconnection of subsystems that form a system. It consists of nodes representing signals and lines representing subsystems.

Similarity transformation A transformation from one state-space representation to another state-space representation. Although the state variables are different, each representation is a valid description of the same system and the relationship between the input and the output.

Stability That characteristic of a system defined by a natural response that decays to zero as time approaches infinity.

Stall torque The torque produced at the armature when a motor's speed is reduced to zero under a condition of constant input voltage.

State equations A set of n simultaneous, first-order differential equations with n variables, where the n variables to be solved are the state variables.

State space The n-dimensional space whose axes are the state variables.

State-space representation A mathematical model for a system that consists of simultaneous, first-order differential equations and an output equation.

State-transition matrix The matrix that performs a transformation on $\mathbf{x}(0)$, taking $\mathbf{x}$ from the initial state, $\mathbf{x}(0)$, to the state $\mathbf{x}(t)$ at any time, $t \geq 0$.

State variables The smallest set of linearly independent system variables such that the values of the members of the set at time t_0 along with known forcing functions completely determine the value of all system variables for all $t \geq t_0$.

State vector A vector whose elements are the state variables.

Static error constants The collection of position constant, velocity constant, and acceleration constant.

Steady-state error The difference between the input and the output of a system after the natural response has decayed to zero.

Steady-state response *See* **forced response.**

Subsystem A system that is a portion of a larger system.

Summing junction A block diagram symbol that shows the algebraic summation of two or more signals.

System type The number of pure integrations in the forward path of a unity feedback system.

System variables Any variable that responds to an input or initial conditions in a system.

Tachometer A voltage generator that yields a voltage output proportional to rotational input speed.

Time constant The time for e^{-at} to decay to 37% of its original value at $t = 0$.

Time-domain representation *See* **state-space representation.**

Torque-speed curve The plot that relates a motor's torque to its speed at a constant input voltage.

Transducer A device that converts a signal from one form to another, for example from a mechanical displacement to an electrical voltage.

Transfer function The ratio of the Laplace transform of the output of a system to the Laplace transform of the input.

Transient response That part of the response curve due to the system and the way the system acquires or dissipates energy. In stable systems it is the part of the response plot prior to the steady-state response.

Tustin transformation A bilinear transformation that converts transfer functions from continuous to sampled and vice versa. The important characteristic of the Tustin transformation is that both transfer functions yield the same output response at the sampling instants.

Type *See* **system type**.

Undamped response The step response of a second-order system that is characterized by a pure oscillation.

Underdamped response The step response of a second-order system that is characterized by overshoot.

Velocity constant $\lim_{s \to 0} sG(s)$

***z*-transformation** A transformation related to the Laplace transformation that is used for the representation, analysis, and design of sampled signals and systems.

Zero-input response That part of the response that depends upon only the initial state vector and not the input.

Zero-order sample-and-hold (z.o.h.) A device that yields a staircase approximation to the analog signal.

Zeros (1) Those values of the Laplace transform variable, s, that cause the transfer function to become zero; and (2) any roots of factors of the numerator that are common to the characteristic equation in the denominator of the transfer function.

Zero-state response That part of the response that depends upon only the input and not the initial state vector.