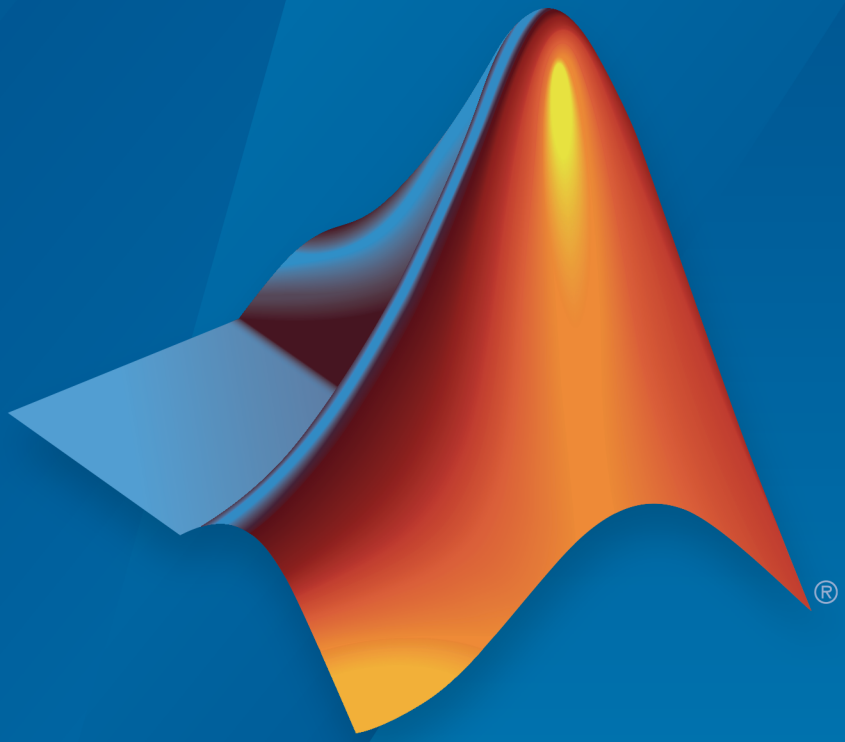
Simulink®

Getting Started Guide



MATLAB® & SIMULINK®

R2017a

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

Simulink® Getting Started Guide

© COPYRIGHT 1990–2017 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

September 2005	Online only	New for Version 6.3 (Release 14SP3)
March 2006	Online only	Revised for Simulink 6.4 (Release 2006a)
September 2006	Online only	Revised for Simulink 6.5 (Release 2006b)
March 2007	First printing	Revised for Simulink 6.6 (Release 2007a)
September 2007	Second printing	Revised for Simulink 7.0 (Release 2007b)
March 2008	Third printing	Revised for Simulink 7.1 (Release 2008a)
October 2008	Fourth printing	Revised for Simulink 7.2 (Release 2008b)
March 2009	Fifth printing	Revised for Simulink 7.3 (Release 2009a)
September 2009	Online only	Revised for Simulink 7.4 (Release 2009b)
March 2010	Online only	Revised for Simulink 7.5 (Release 2010a)
September 2010	Online only	Revised for Simulink 7.6 (Release 2010b)
April 2011	Online only	Revised for Simulink 7.7 (Release 2011a)
September 2011	Sixth printing	Revised for Simulink 7.8 (Release 2011b)
March 2012	Seventh printing	Revised for Simulink 7.9 (Release 2012a)
September 2012	Eighth printing	Revised for Simulink 8.0 (Release 2012b)
March 2013	Ninth printing	Revised for Simulink 8.1 (Release 2013a)
September 2013	Tenth printing	Revised for Simulink 8.2 (Release 2013b)
March 2014	Eleventh printing	Revised for Simulink 8.3 (Release 2014a)
October 2014	Twelfth printing	Revised for Simulink 8.4 (Release 2014b)
March 2015	Thirteenth printing	Revised for Simulink 8.5 (Release 2015a)
September 2015	Fourteenth printing	Revised for Simulink 8.6 (Release 2015b)
October 2015	Online only	Rereleased for Simulink 8.5.1 (Release 2015aSP1)
March 2016	Fifteenth printing	Revised for Simulink 8.7 (Release 2016a)
September 2016	Sixteenth printing	Revised for Simulink 8.8 (Release 2016b)
March 2017	Seventeenth printing	Revised for Simulink 8.9 (Release 2017a)

Contents

1 Introduction

Simulink Product Description 1-2
Key Features 1-2

Model-Based Design 1-3
What Is Model-Based Design? 1-3
Modeling, Simulation, and Analysis with Simulink 1-4
Interaction with MATLAB Environment 1-5

Basic Modeling Workflow 1-6
Define System 1-6
Model System 1-9
Integrate Model 1-11

Basic Simulation Workflow 1-13
Prepare for Simulation 1-13
Run and Evaluate Simulation 1-15

Documentation and Resources 1-17
Simulink Online Help 1-17
Simulink Examples 1-18
Website Resources 1-19

2 Simple Simulink Model

Create Simple Model 2-2
Model Overview for This Tutorial 2-2
Open New Model in Simulink Editor 2-3
Open Simulink Library Browser 2-5

Browse or Search for Specific Blocks 2-7
Add Blocks to Model 2-8
Move and Resize Blocks 2-10
Block Connections 2-11
Draw Signal Lines Between Blocks 2-11
Draw Branched Signal Lines 2-14
Define Configuration Parameters 2-16
Run Simulation 2-17
Observe Simulation Results 2-17

3 Model a Dynamic System

Model a Dynamic System 3-2
Define a House Heating System 3-2
Model a House Heating System 3-8
Integrate a House Heating Model 3-25

4 Basic Simulation Workflow

Simulate a Dynamic System 4-2
Prepare for Simulation 4-3
Run and Evaluate Simulation 4-6

1

Introduction

- "Simulink Product Description" on page 1-2
- "Model-Based Design" on page 1-3
- "Basic Modeling Workflow" on page 1-6
- "Basic Simulation Workflow" on page 1-13
- "Documentation and Resources" on page 1-17

Simulink Product Description

Simulation and Model-Based Design

Simulink is a block diagram environment for multidomain simulation and Model-Based Design. It supports system-level design, simulation, automatic code generation, and continuous test and verification of embedded systems. Simulink provides a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It is integrated with MATLAB®, enabling you to incorporate MATLAB algorithms into models and export simulation results to MATLAB for further analysis.

Key Features

- Graphical editor for building and managing hierarchical block diagrams
- Libraries of predefined blocks for modeling continuous-time and discrete-time systems
- Simulation engine with fixed-step and variable-step ODE solvers
- Scopes and data displays for viewing simulation results
- Project and data management tools for managing model files and data
- Model analysis tools for refining model architecture and increasing simulation speed
- MATLAB Function block for importing MATLAB algorithms into models
- Legacy Code Tool for importing C and C++ code into models

Model-Based Design

In this section...
"What Is Model-Based Design?" on page 1-3
"Modeling, Simulation, and Analysis with Simulink" on page 1-4
"Interaction with MATLAB Environment" on page 1-5

What Is Model-Based Design?

Model-Based Design is a process that enables fast and cost-effective development of dynamic systems, including control systems, signal processing, and communications systems. In Model-Based Design, a system model is at the center of the development process, from requirements development through design, implementation, and testing. The model is an executable specification that you continually refine throughout the development process. After model development, simulation shows whether the model works correctly.

When software and hardware implementation requirements are included with the model, such as fixed-point and timing behavior, you can generate code for embedded deployment and create test benches for system verification, saving time and avoiding manually coded errors.

Model-Based Design allows you to improve efficiency by:

- Using a common design environment across project teams
- Linking designs directly to requirements
- Integrating testing with design to continuously identify and correct errors
- Refining algorithms through multi-domain simulation
- Generating embedded software code
- Developing and reusing test suites
- Generating documentation
- Reusing designs to deploy systems across multiple processors and hardware targets

Modeling, Simulation, and Analysis with Simulink

With Simulink, you can move beyond idealized linear models to explore realistic nonlinear models, factoring in friction, air resistance, gear slippage, hard stops, and the other parameters that describe real-world phenomena. Simulink enables you to think of the development environment as a laboratory for modeling and analyzing systems that would not be possible or practical otherwise.

Whether you are interested in the behavior of an automotive clutch system, the flutter of an airplane wing, or the effect of the monetary supply on the economy, Simulink provides you with the tools to model and simulate almost any real-world problem. Simulink also provides examples that model a wide variety of real-world phenomena.

Tool for Modeling

Simulink provides a graphical editor for building models as block diagrams, allowing you to draw models as you would with pencil and paper. Simulink also includes a comprehensive library of sink, source, linear and nonlinear component, and connector blocks. If these blocks do not meet your needs, however, you can also create your own blocks. The interactive environment simplifies the modeling process, eliminating the need to formulate differential and difference equations in a language or program.

Models are hierarchical, so you can build models using both top-down and bottom-up approaches. You can view the system at a high level, then drill down to see increasing levels of model detail. This approach provides insight into how a model is organized and how parts interact.

Tool for Simulation

After you define a model, you can simulate its dynamic behavior using a choice of mathematical integration methods, either interactively in Simulink or by entering commands in the MATLAB Command Window. Commands are particularly useful for running a batch of simulations. For example, if you are doing Monte Carlo simulations or want to apply a parameter across a range of values, you can use MATLAB scripts.

Using scopes and other display blocks, you can see the simulation results while a simulation runs. You can then change parameters and see what happens for “what if” exploration. You can save simulation results in the MATLAB workspace for postprocessing and visualization.

Tool for Analysis

Model analysis tools include linearization and trimming tools you can access from MATLAB, plus the many tools in MATLAB and its application toolboxes. Because MATLAB and Simulink are integrated, you can simulate, analyze, and revise your models in either environment.

Interaction with MATLAB Environment

Simulink software requires MATLAB to run, and it depends on it to define and evaluate model and block parameters. Simulink can also use many MATLAB features. For example, Simulink can use the MATLAB environment to:

- Define model inputs.
- Store model outputs for analysis and visualization.
- Perform functions within a model, through integrated calls to MATLAB operators and functions.

Related Examples

- “Create Simple Model” on page 2-2
- “Basic Modeling Workflow” on page 1-6
- “Basic Simulation Workflow” on page 1-13

External Websites

- Simulink Overview
- Model-Based Design with MATLAB and Simulink

Basic Modeling Workflow

In this section...
“Define System” on page 1-6
“Model System” on page 1-9
“Integrate Model” on page 1-11

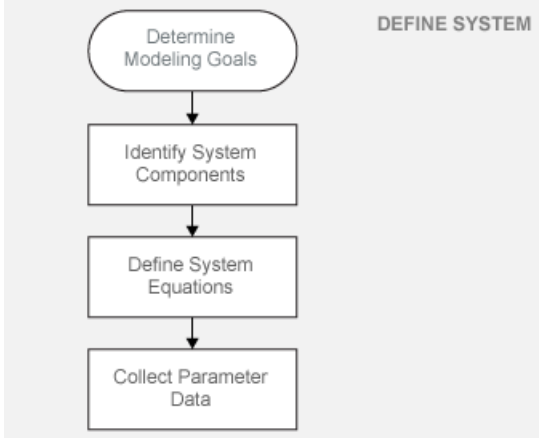
Modeling is a process in which you describe a dynamic system with mathematical equations and then create a simplified representation of the system with a model. The equations define the science of the system and the model uses the equations to define the time-varying behavior.

The steps in a typical modeling workflow include:

- Defining a system
- Modeling the system
- Integrating the system

Define System

Identify the components of a system, determine physical characteristics, and define dynamic behavior with equations. You perform these steps outside of the Simulink software environment and before you begin building your model.



Determine Modeling Goals

Before designing a model, you need to understand your goals and requirements. Ask yourself these questions to help plan your model design:

- What problem does the model help you solve?
- What questions can it answer?
- How accurately must it represent the system?

Some possible modeling goals:

- Understand how system components interact with each other.
- Explore controller and fault-tolerance strategies.
- Decide between alternative designs.
- Observe the response of systems that you cannot solve analytically.
- Determine how various inputs and changing model parameters affect the output.

Identify System Components

Once you understand your modeling requirements, you can begin to identify the components of the system.

- Identify the components that correspond to structural parts of the system. Creating a model that reflects the physical structure of a system, for example, motor controller or brake system, is helpful when you have to build part of the system in software and hardware.
- Identify functional parts that you can independently model and test.
- Describe the relationships between components, for example, data, energy, and force transfer.
- Draw a picture showing the connections between components. Include major parameters in your diagram. Creating a picture of the system can help you identify and model the parts that are essential to the behaviors you want to observe.

Define System Equations

After you identify the components in a system, you can describe the system mathematically with equations. Derive the equations using scientific principles or from the input-output response of measured data. Many of the system equations fall into three categories:

- For continuous systems, differential equations describe the rate of change for variables with the equations defined for all values of time. For example, the velocity of a car is given by the second order differential equation

$$\frac{dv(t)}{dt} = -\frac{b}{m}v(t) + u(t).$$

- For discrete systems, difference equations describe the rate of change for variables, but the equations are defined only at specific times. For example, the control signal from a discrete propositional-derivative controller is given by the difference equation

$$pd[n] = (e[n] - e[n-1])K_d + e[n]K_p.$$

- Equations without derivatives are algebraic equations. For example, the total current in a parallel circuit with two components is given by the algebraic equation

$$I_t = I_a + I_b.$$

Collect Parameter Data

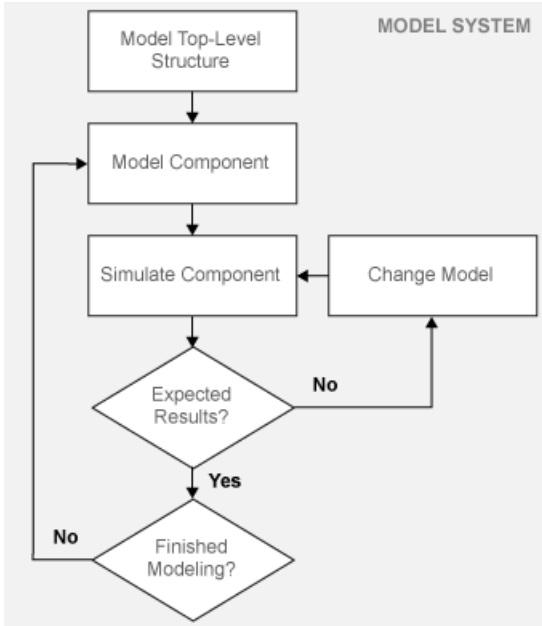
Create a list of equation variables and constant coefficients, and then determine the coefficient values from published sources or by performing experiments on the system.

Use measured data from a system to define equation coefficients and parameters in your model.

- Identify the parts that are measurable in a system.
- Measure physical characteristics or use published property values. Manufacturer data sheets are a good source for hardware values.
- Perform experiments to measure the system response to various inputs. You will later use this data to verify your model design with simulations.

Model System

Build individual model components that implement the system equations, and define the interfaces for passing data between components.



Model Top-Level Structure

A model in Simulink is a graphical representation of a system using blocks and connections between blocks. After you finish describing your system, its components, and equations, you can begin to build your model.

- Use the system equations to build a graphical model of the system with the Simulink Editor.
- If you place all of the model blocks in one level of a diagram, your diagram can become difficult to read and understand. One way to organize your model is to use subsystems. Examples of blocks you can use to create a subsystem include Subsystem, Atomic Subsystem, and Model.
- Identify input and output connections (for example, signal lines) between subsystems. Input and output values change dynamically during a simulation.

Model Component

Some questions to ask before you begin to model a component:

- What are the constants for each component and the values that do not change unless you change them?
- What are the variables for each component and the values that change over time?
- How many state variables does a component have?

After you create the top-level structure for your model, you can begin to model the individual components.

- Use the system equations to create a Simulink model.
- Add Simulink blocks in the Simulink Editor. Blocks represent coefficients and variables from the equations. Connect blocks to other blocks. Lines connecting blocks represent data transfer.
- Build the model for each component separately. The most effective way to build a model of a system is to consider components independently.
- Start by building simple models using approximations of the system. Identify assumptions that can affect the accuracy of your model. Iteratively add detail until the level of complexity satisfies the modeling and accuracy requirements.

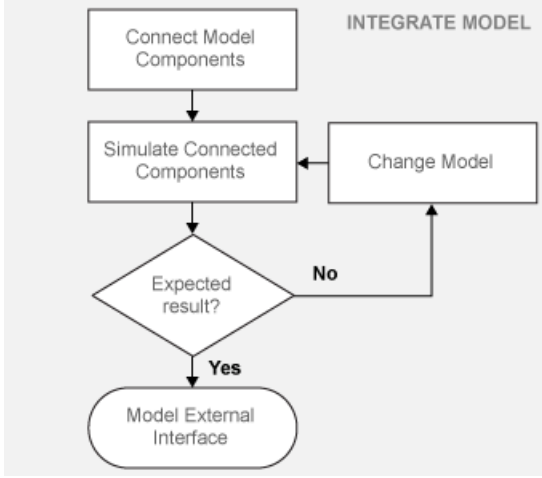
Simulate Component

After you build a model component, you can simulate to validate the design.

- Predict the expected output of the integrated model components.
- Add blocks to approximate actual input and control values. Add sink blocks to record and visualize results.
- Validate the model design by comparing the simulation output to your expected output.
- If the result does not match your prediction, change your model to improve the accuracy of your prediction. Changes include model structure and parameters.

Integrate Model

Connect component models and simulate the model response over time to validate the design.



Connect Model Components

After you build and validate each model component, you can connect them into a complete model, simulate the model, and analyze the results.

Some guidelines for connecting model components:

- Integrate model components by first connecting two of them (for example, connect a plant to a controller). After validating the pair by simulation, continue connecting components until your model is complete.
- Think about how each component you add affects the other parts of the model.

Simulate Connected Components

Validating your model determines if it accurately represents the physical characteristics of the modeled dynamic system. Some guidelines for validating subsystems:

- Predict the expected simulation results and outputs of the subsystems.
- Add realistic inputs using source blocks.
- Add sink blocks to record and visualize results.
- Simulate the subsystems and compare the simulated result with your expected result.

Model External Interface

Add blocks for connecting external signals into and out of your model.

Related Examples

- "Basic Simulation Workflow" on page 1-13

Basic Simulation Workflow

In this section...
“Prepare for Simulation” on page 1-13 “Run and Evaluate Simulation” on page 1-15

Simulation is a process in which you validate and verify a model by comparing simulation results with:

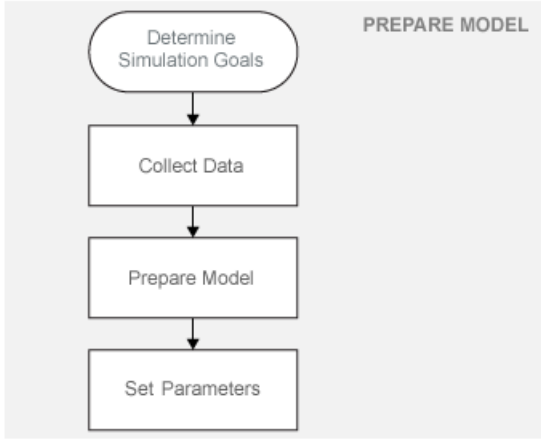
- Data collected from a real system.
- Functionality described in the model requirements.

Perform the simulation workflow after you've finished building your model and a simulation completes without errors. The steps in a typical simulation workflow include:

- Prepare for simulation.
- Run and evaluate simulation.

Prepare for Simulation

Define the external input and output interfaces.



Determine Simulation Goals

Before simulating a model, you need to understand your goals and requirements. Ask yourself these questions to help plan your simulations:

- What questions do you want the simulation to answer?
- How accurately does the model need to represent the system?

Some possible simulation goals:

- Understand input to output causality — For a given input set and nominal parameter values, look at how the inputs flow through the system to the outputs.
- Verify model — Compare simulation results with collected data from the modeled system. Iteratively debug and improve the design.
- Optimize parameters — Change parameters and compare simulation runs.
- Visualize results — Send simulation results to a plot or print in a report.

Collect Data

Collect input and output data from an actual system.

- Use the measured input data to drive the simulation.
- Compare measured output data with the model simulation results to verify the accuracy of your model.

Prepare Model

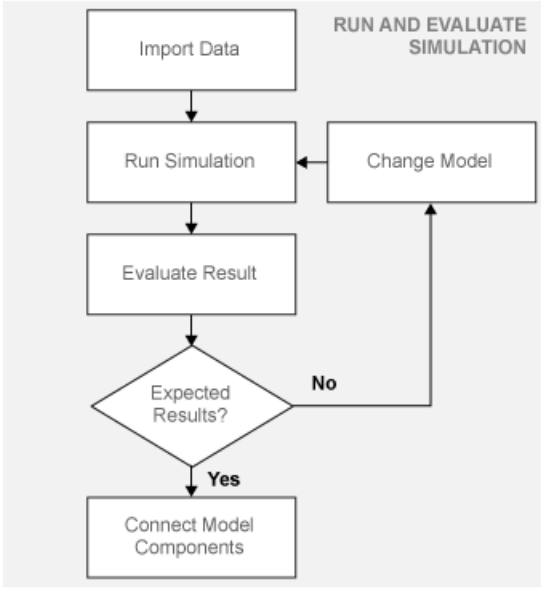
Preparing a model for simulation includes defining the external interfaces for input data and control signals, and output signals for viewing and recording simulation results.

Set Parameters

For the first simulation, use model parameters from the validated model. After comparing the simulation results with measured output data, change model parameters to more accurately represent the modeled system.

Run and Evaluate Simulation

Simulate your model and verify that the simulation results match the measured data from the modeled system.



Import Data

Simulink enables you to import data into your model.

- Use the Signal Builder block to import input signals from a Microsoft® Excel® file (XLSX, XLS) or a comma-separated value file (CSV). Simulink saves data imported from an Excel file using a Signal Builder block with the model and loads the data into memory when you open the model.
- For large data sets, use a MATLAB MAT-file with an Inport block.

Run Simulation

Using measured input data, run a simulation and save results.

Evaluate Result

Evaluate the differences between simulated output and measured output data. Use the evaluation to verify the accuracy of your model and how well it represents the system behavior. Decide if the accuracy of your model adequately represents the dynamic system you are modeling.

Change Model

Determine the changes to improve your model. Model changes include:

- Parameters — Some parameters were initially estimated and approximated. Optimize and update parameters.
- Adding structure — Some parts or details of the system were not modeled. Add missing details.

Related Examples

- “Basic Modeling Workflow” on page 1-6

Documentation and Resources

In this section...
“Simulink Online Help” on page 1-17
“Simulink Examples” on page 1-18
“Website Resources” on page 1-19

Simulink Online Help

Simulink software provides comprehensive online help describing features, blocks, and functions with detailed procedures for common tasks.

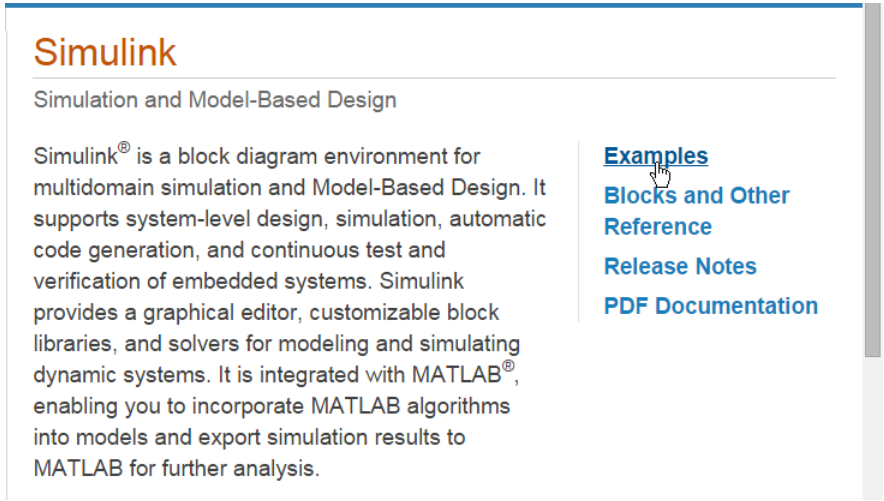
Access online help from **Help** menus and context-sensitive block labels.

- From the Simulink Library Browser toolbar, select the **Help** button .
- From the Simulink Editor menu, select **Help** > **Simulink** > **Simulink Help**.
- Right-click a Simulink block, and then select **Help**.
- From the model Configuration Parameters dialog box or a block parameters dialog box, right-click a parameter label, then select **What’s This?**

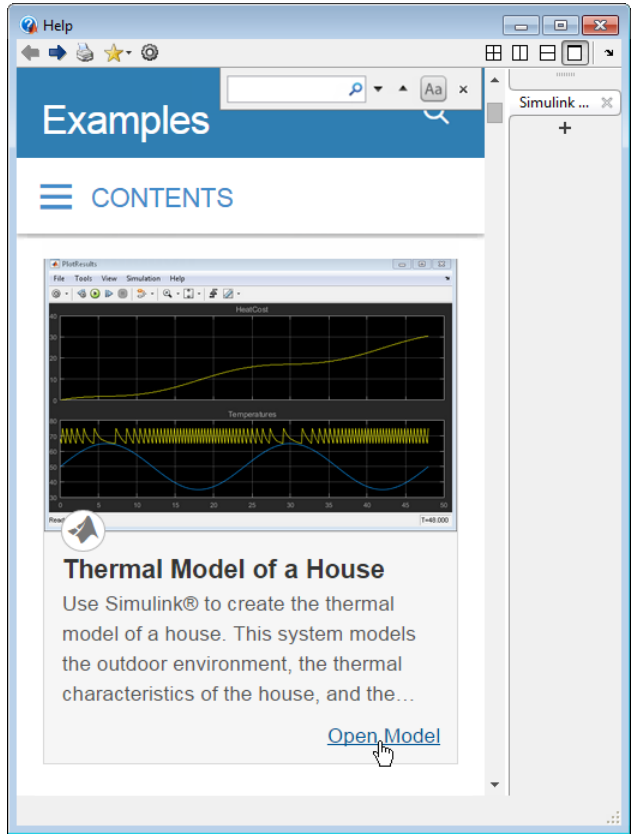
Simulink Examples

Simulink provides example models that illustrate key modeling concepts and Simulink features. To view a list of examples:

- From the Simulink Editor menu, select **Help** > **Simulink** > **Examples**.
- From the Help browser, open the Simulink product page, and then click **Examples** at the top right.



To open the Simulink model for an example, click the **Open Model** button.



Website Resources

You can access additional Simulink resources on the MathWorks website, including a description of capabilities, technical articles, tutorials, and hardware support.

http://www.mathworks.com/products/simulink

Related Examples

- “Model a Dynamic System” on page 3-2
- “Simulate a Dynamic System” on page 4-2

2

Simple Simulink Model

Create Simple Model

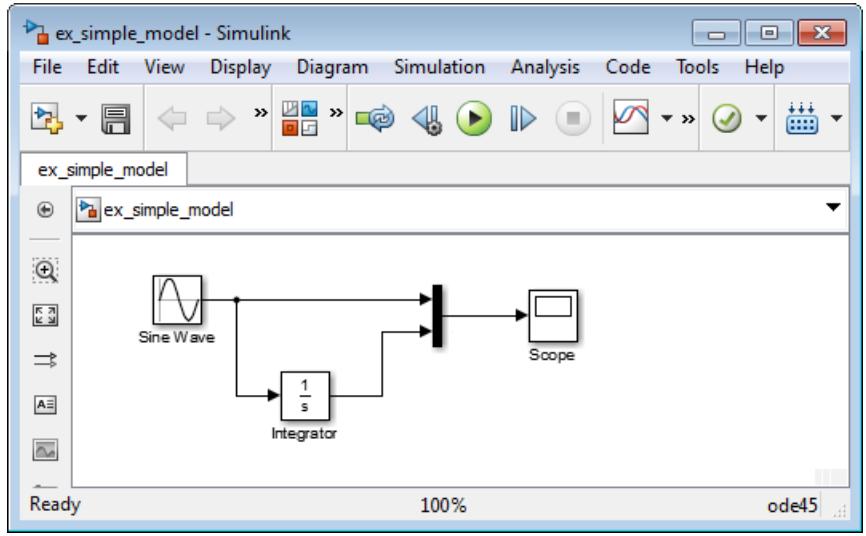
In this section...
"Model Overview for This Tutorial" on page 2-2
"Open New Model in Simulink Editor" on page 2-3
"Open Simulink Library Browser" on page 2-5
"Browse or Search for Specific Blocks" on page 2-7
"Add Blocks to Model" on page 2-8
"Move and Resize Blocks" on page 2-10
"Block Connections" on page 2-11
"Draw Signal Lines Between Blocks" on page 2-11
"Draw Branched Signal Lines" on page 2-14
"Define Configuration Parameters" on page 2-16
"Run Simulation" on page 2-17
"Observe Simulation Results" on page 2-17

Model Overview for This Tutorial

You can use Simulink to model a system and then simulate the dynamic behavior of that system. The basic techniques you use to create the simple model in this tutorial are the same techniques that you use for more complex models.

To create this simple model, you need four Simulink blocks. Blocks are the model elements that define the mathematics of a system and provide input signals:

- Sine Wave — Generate an input signal for the model.
- Integrator — Process the input signal.
- Bus Creator — Combine multiple signals into one signal.
- Scope — Visualize and compare the input signal with the output signal.



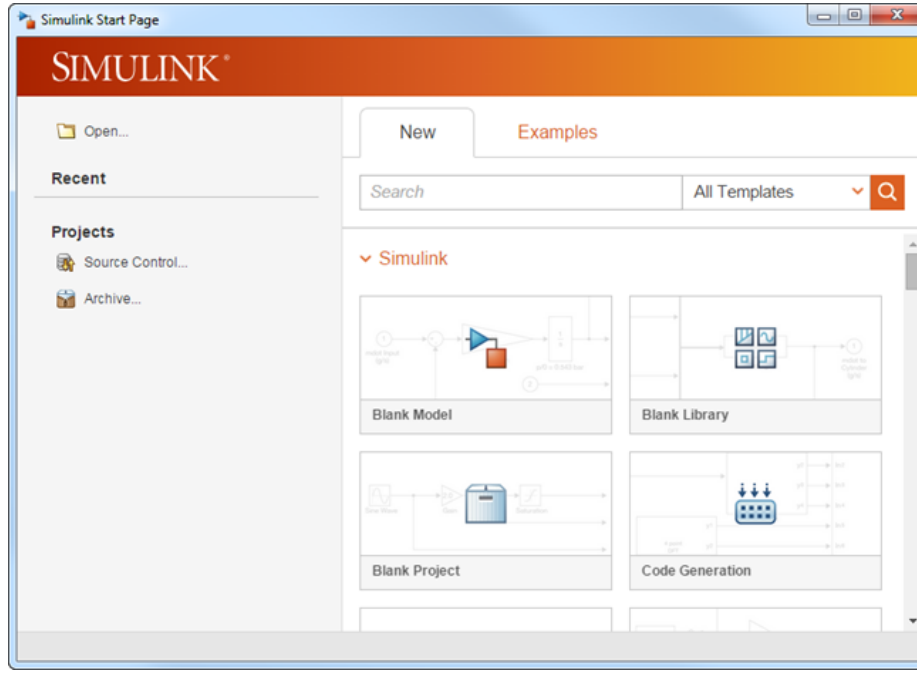
Simulating this model integrates a sine wave signal to a cosine signal and then displays the result, along with the original signal, in a Scope window.

Open New Model in Simulink Editor

Use the Simulink Editor to build your models.

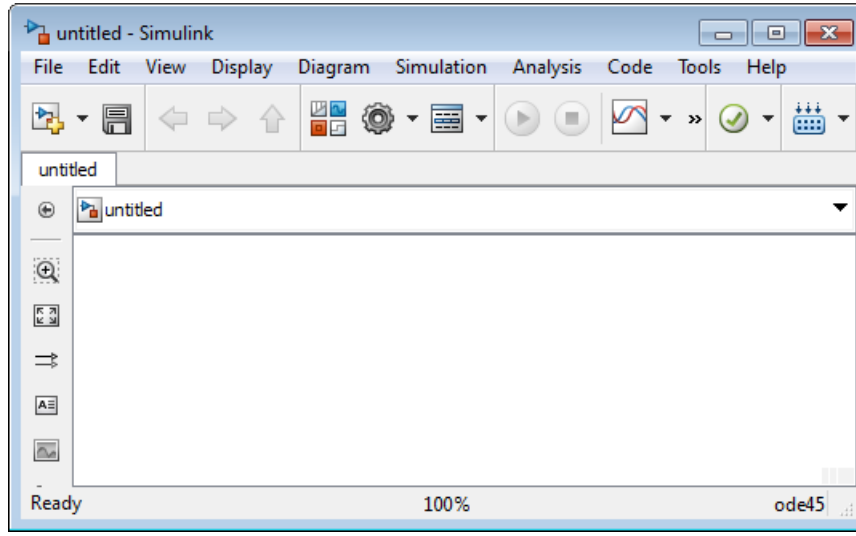
1 Start MATLAB. From the MATLAB Toolstrip, click the **Simulink** button .

A short delay occurs the first time you open Simulink.



2 Click the **Blank Model** template, and then click the **Create Model** button.

The Simulink Editor opens with a new block diagram.

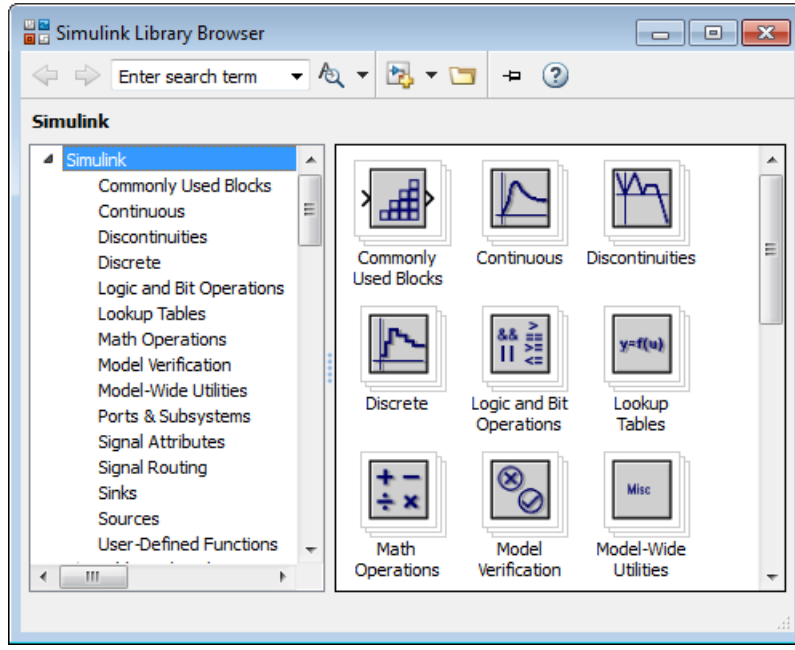


3 Select **File > Save as**. In the **File name** text box, enter a name for your model, For example, `simple_model`. Click **Save**. The model is saved with the file extension `.slx`.

Open Simulink Library Browser

From the Simulink Library Browser you can search for blocks to use in your model. Also, you can create a new Simulink model, project, or Stateflow® chart.

1 From the Simulink Editor toolbar, click the **Simulink Library** button .

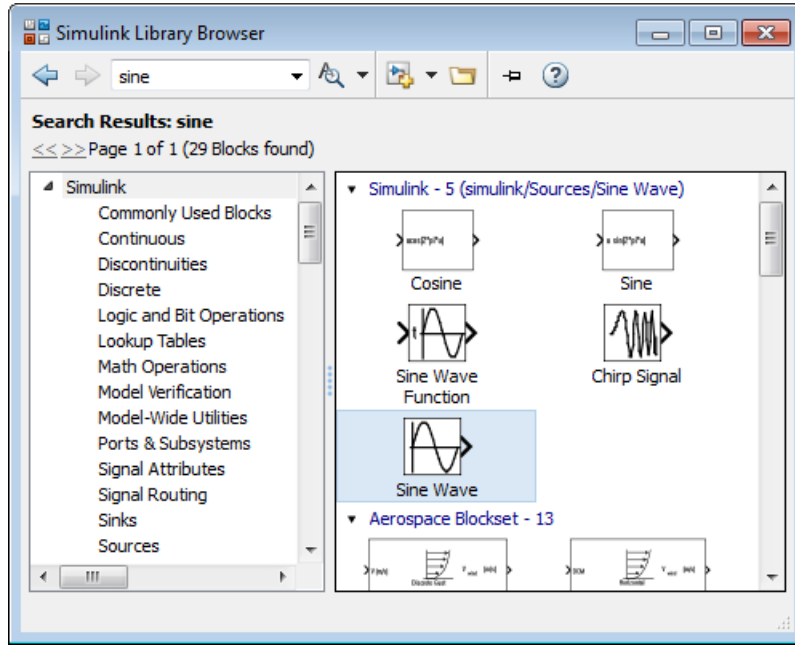


2 Set the Library Browser to stay on top of the other desktop windows. On the Library Browser toolbar, select the **Stay on top** button .

Browse or Search for Specific Blocks

To browse through the block libraries, select a MathWorks® product and then a functional area in the left pane. To search all of the available block libraries, enter a search term.

1. Search for a Sine Wave block. In the search box on the browser toolbar, enter `sine`, and then press the Enter key. Simulink searches the libraries for blocks with `sine` in their name or description, and then displays the blocks.



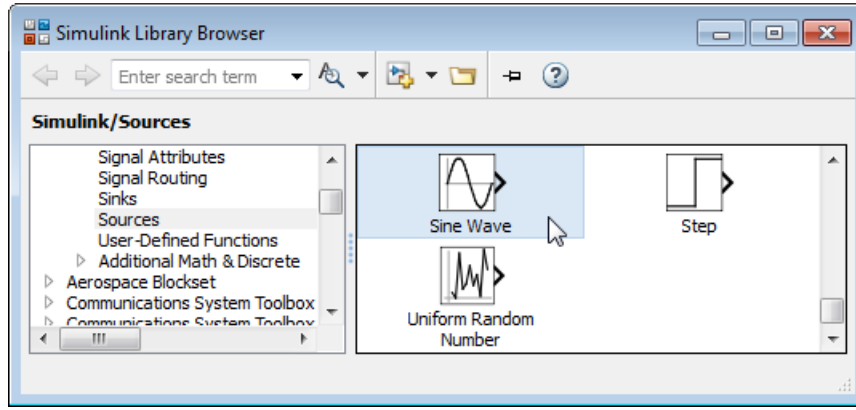
2. Get detailed information about a block. Right-click a block, and then select **Help for the <block name>**. The Help browser opens with the reference page for the block.
3. View block parameters. Right-click a block, and then select **Block Parameters**. The block parameters dialog box opens.

Add Blocks to Model

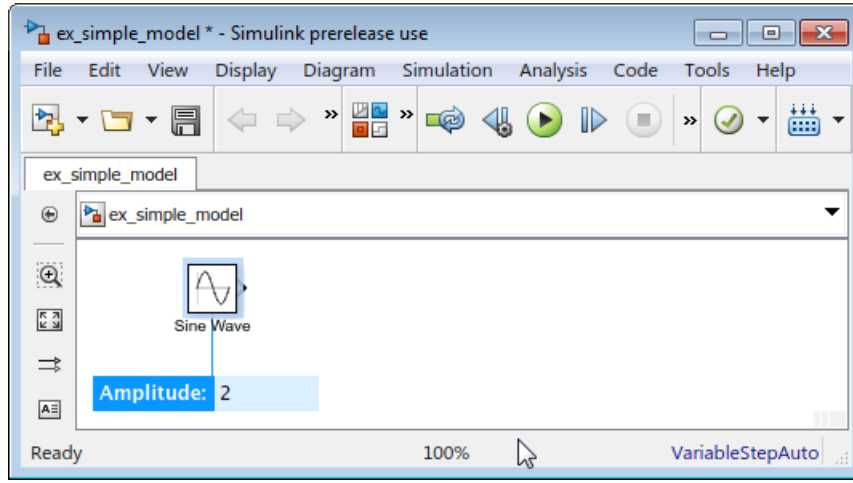
You build models by dragging blocks from the Simulink Library Browser window to the Simulink Editor or single-clicking your model and entering a search term.

To build the simple model, begin by copying blocks from the Simulink Library Browser to the Simulink Editor.

1. In the left pane of Simulink Library Browser, select the `Sources` library.
2. From the right pane, select the Sine Wave block.



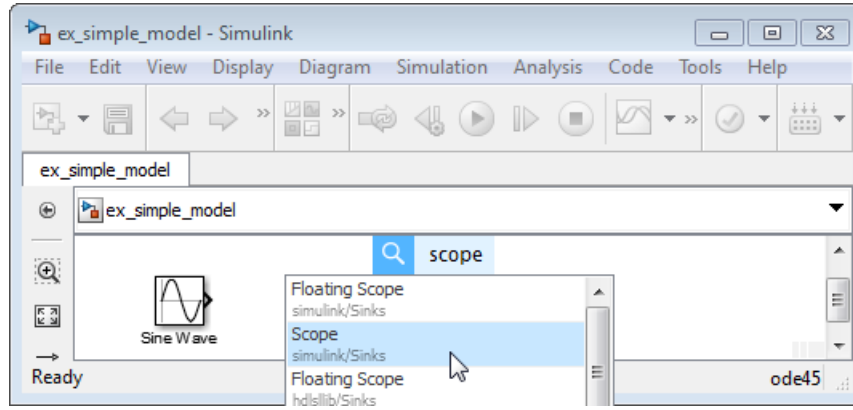
3. Drag the Sine Wave block to the Simulink Editor. A copy of the Sine Wave block appears in your model with a text box for entering the value of the **Amplitude** parameter. In the text box enter `2`.



4 Add a Scope block using this alternative procedure:

   a Click within the block diagram.

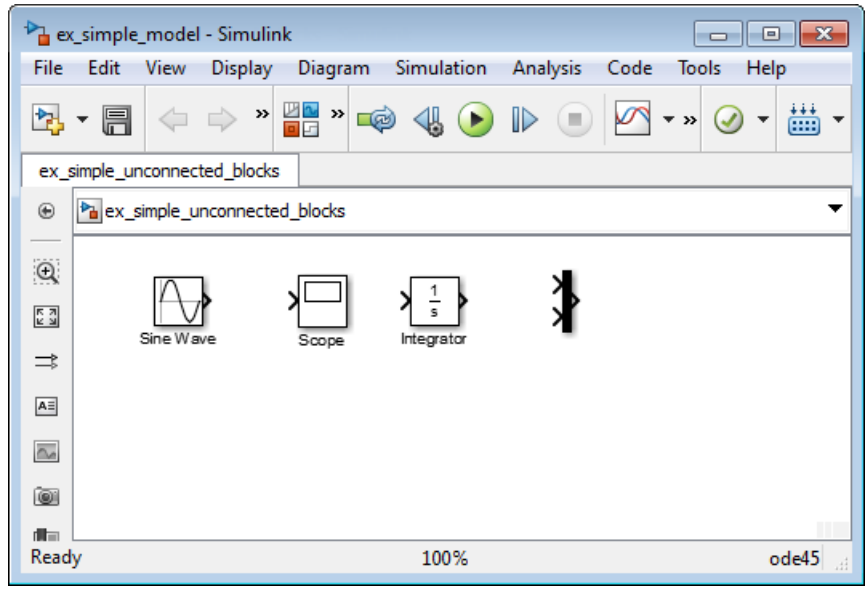
   b After the search icon appears, type `scope`, and then from the list, select `Scope` from the `Simulink/Sinks` library.



5 Add the following blocks to your model using one of the approaches you used to add the Sine Wave and Scope blocks.

Library	Block
Simulink/Continuous	Integrator
Simulink/Signal Routing	Bus Creator

Your model should now have the blocks you need for the simple model.



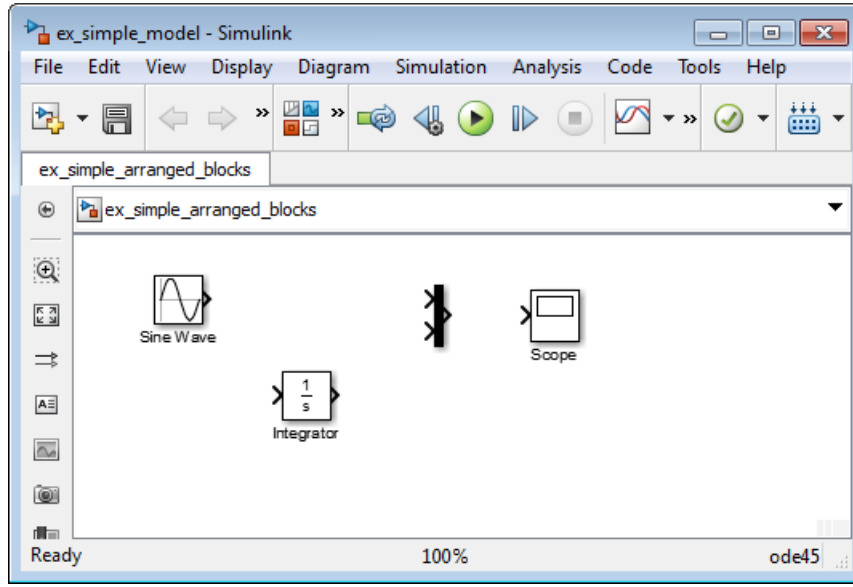
Move and Resize Blocks

Before you connect the blocks in your model, arrange the block inputs and outputs to make signal connections as straightforward as possible.

1 Move the Scope block after the Bus Creator block output. You can either:

- Click and drag the block.
- Select the block, and then press the arrow keys on your keyboard.

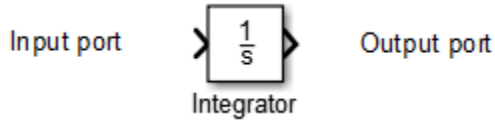
2 Move the blocks until your model looks similar to the following figure.



Block Connections

Most blocks have angle brackets on one or both sides. These angle brackets represent input and output ports:

- The > symbol pointing into a block is an *input port*.
- The > symbol pointing out of a block is an *output port*.



You connect block output ports to input ports with lines. The lines represent signals with time-varying values.

Draw Signal Lines Between Blocks

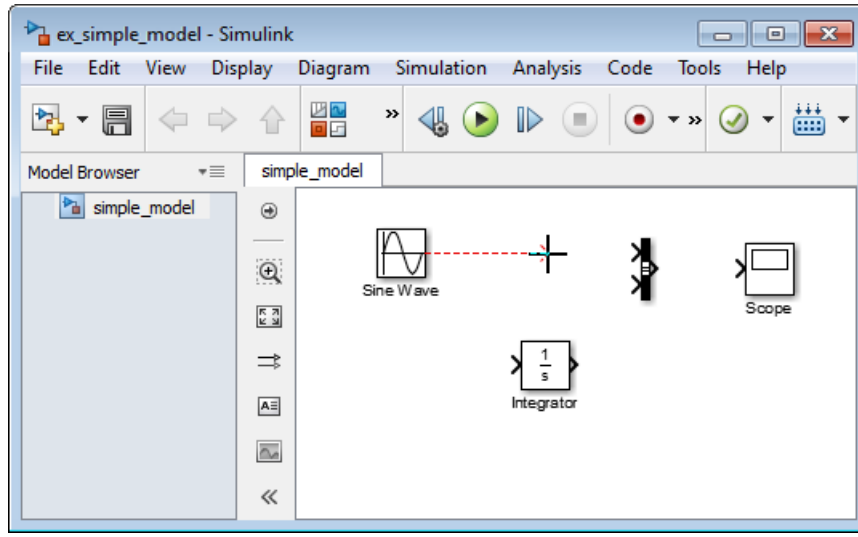
Connect the blocks by drawing lines between output ports and input ports.

1 Position the cursor over the output port on the right side of the Sine Wave block.

The pointer changes to a cross hair (+).

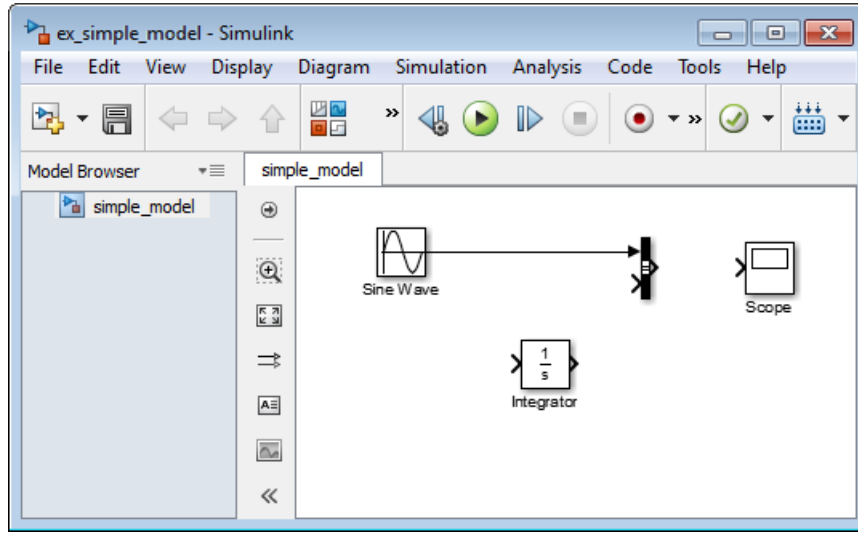
2 Click, and then drag a line from the output port to the top input port of the Bus Creator block.

While holding down the mouse button, the connecting line appears as a red dotted arrow.



3 Release the mouse button when the pointer is over the output port.

Simulink connects the blocks with a line and an arrow indicating the direction of signal flow.



4 Connect the output port of the Integrator block to the bottom input port on the Bus Creator block using a **Ctrl** key shortcut:

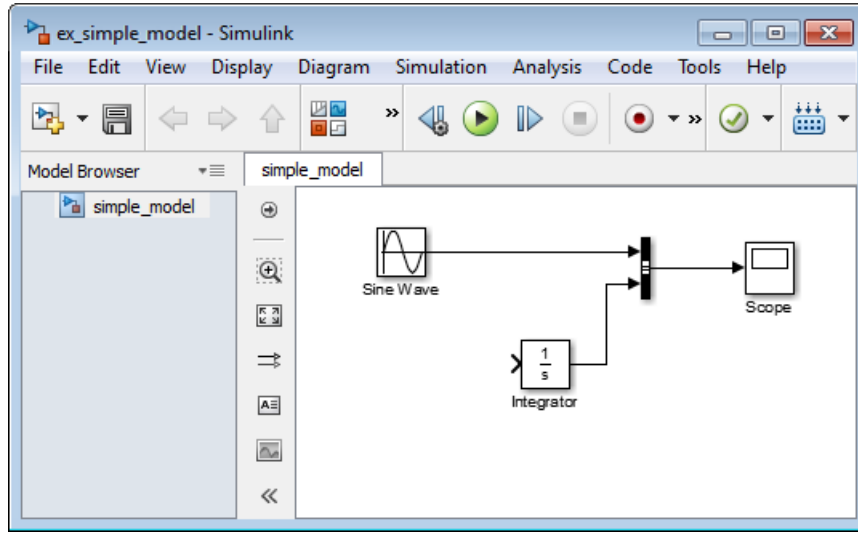
   a Select the Integrator block.

   b Press and hold the **Ctrl** key.

   c Click the Bus Creator block.

   The Integrator block connects to the Bus Creator block with a signal line.

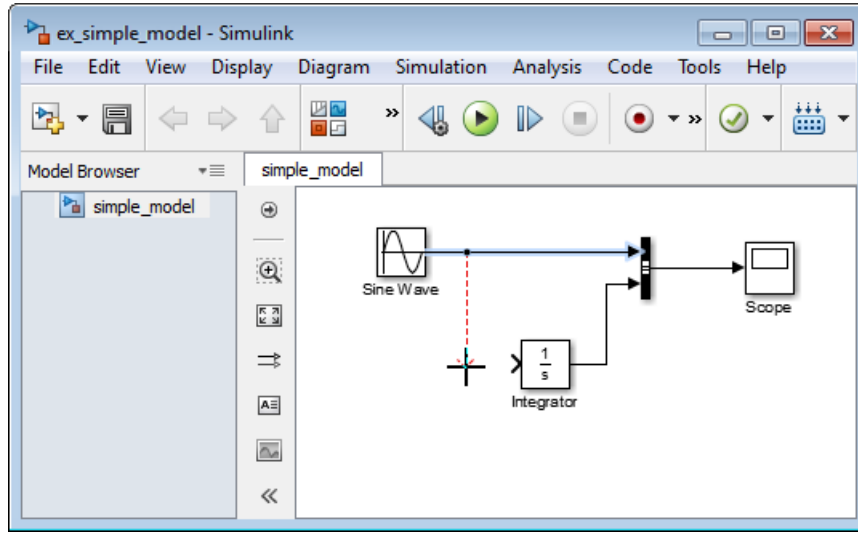
5 Click and drag the output port from the Bus Creator block to the Scope block.



Draw Branched Signal Lines

Your simple model is almost complete. To finish the model, connect the Sine Wave block to the Integrator block. This connection is different from the other connections, which all connect output ports to input ports.

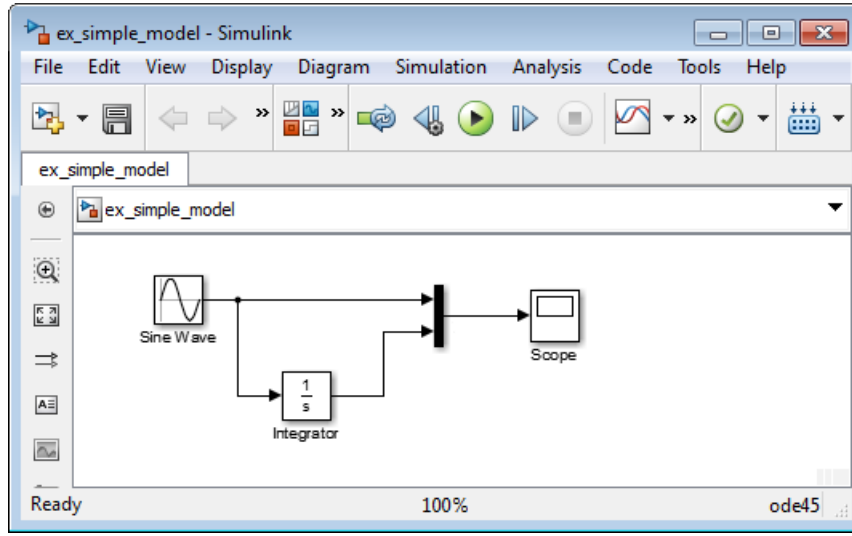
1. Position the cursor where you want to start a branch line. In this example, position the cursor over the signal line between the Sine Wave and Bus Creator blocks.
2. Right-click and drag the cursor away from the line to form a dotted line segment.



3 Continue to drag the cursor to the Integrator input port, and then release the mouse button.

The new line, called a *branch line*, carries the same signal that passes from the Sine Wave block to the Bus Creator block.

4 Drag line segments to straighten and align with blocks. Your model is now complete.



Define Configuration Parameters

Before you simulate the behavior of your model, you can modify the default values for the configuration parameters. The configuration parameters include the type of numerical solver, start time, stop time, and maximum step size.

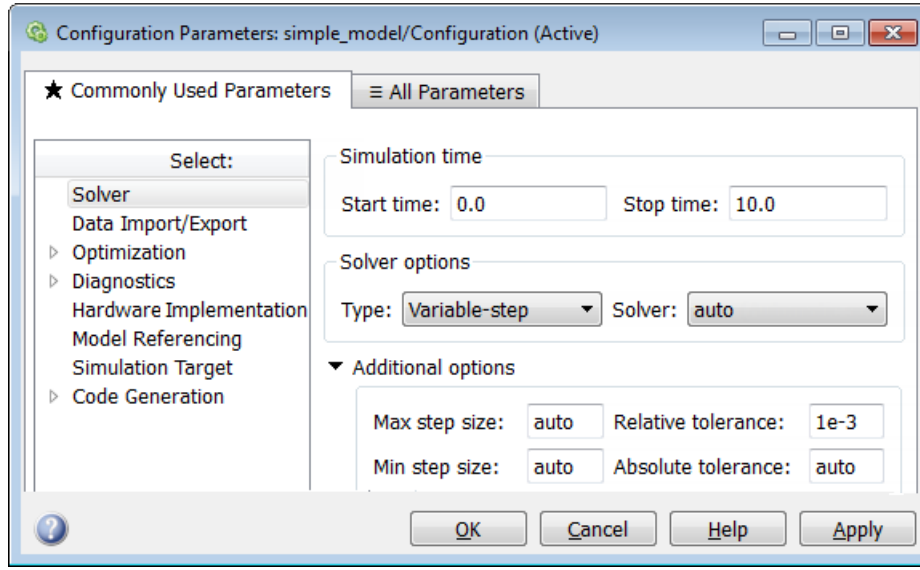
1 From the Simulink Editor menu, select **Simulation** > **Model Configuration Parameters**. The Configuration Parameters dialog box opens to the **Solver** pane.

Tip: Alternatively, you can open the Model Configuration Parameters dialog box by clicking the parameters button on the Simulink Editor toolbar.

---

2 In the **Stop time** field, enter `20`. With the **Solver** parameter set to `auto`, Simulink determines the best numerical solver for simulating your model.

3 Select **Additional options**. In the **Max step size** field, enter `0.2`.



4 Click **Apply** or **OK**.

Run Simulation

After you define the configuration parameters, you are ready to simulate your model.

1 From the Simulink Editor menu bar, select **Simulation** > **Run**.

The simulation runs, and then stops when it reaches the stop time specified in the Model Configuration Parameters dialog box.

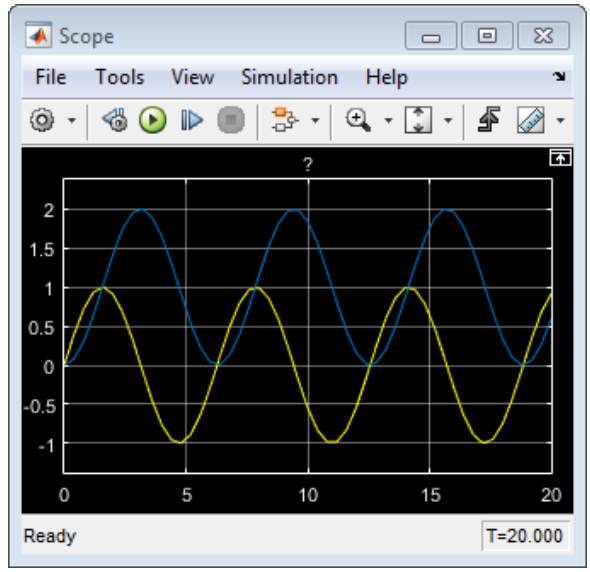
Tip: Alternatively, you can control a simulation by clicking the **Run** simulation button and **Pause** simulation button on the Simulink Editor toolbar or Scope window toolbar.

Observe Simulation Results

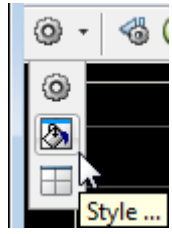
After simulating a model, you can view the simulation results in a Scope window.

1 Double-click the Scope block.

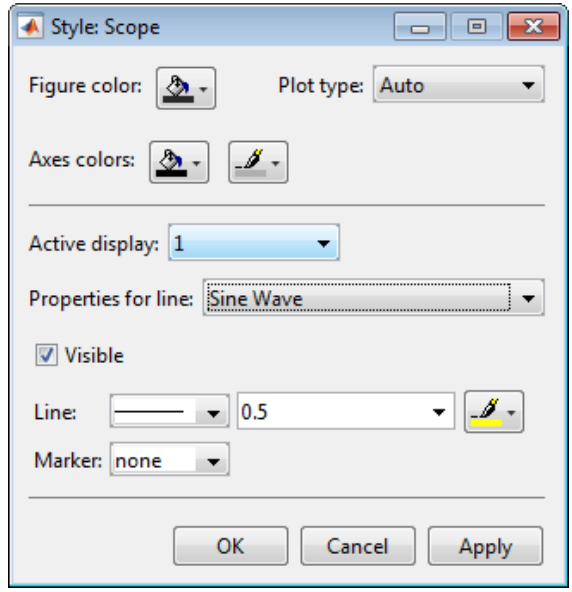
The Scope window opens and displays the simulation results. The plot shows a sine wave signal with the resulting cosine wave signal.



2 On the Scope window toolbar, click the **Style** button.



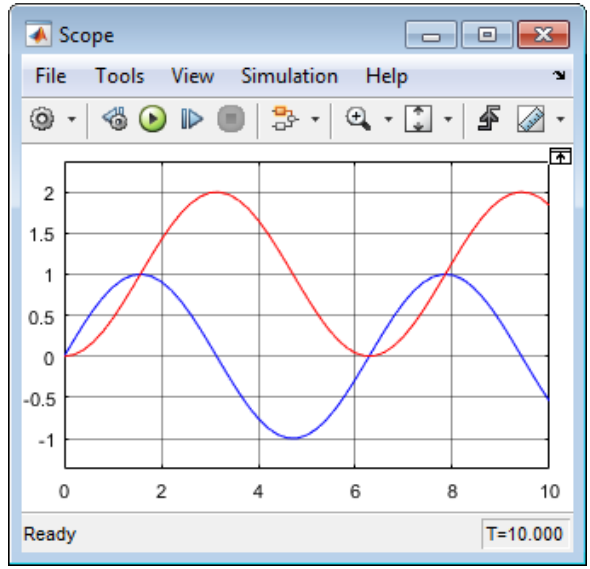
A Style dialog box opens with display options.



3 Change the appearance of the display. For example, select white for the **Figure color** and **Axes color** background (icons with a pitcher).

4 Select black for the **Axes color** ticks, labels, and grid colors (icon with a paintbrush).

5 Change signal line colors for the Sine Wave to blue and the Integrator to red. To see your changes, click **OK** or **Apply**.



See Also

Blocks

Bus Creator | Integrator | Scope | Sine Wave

Related Examples

- "Build and Edit a Model in the Simulink Editor"
- "Model a Dynamic System" on page 3-2
- "Simulate a Dynamic System" on page 4-2
- Getting Started with Simulink

3

Model a Dynamic System

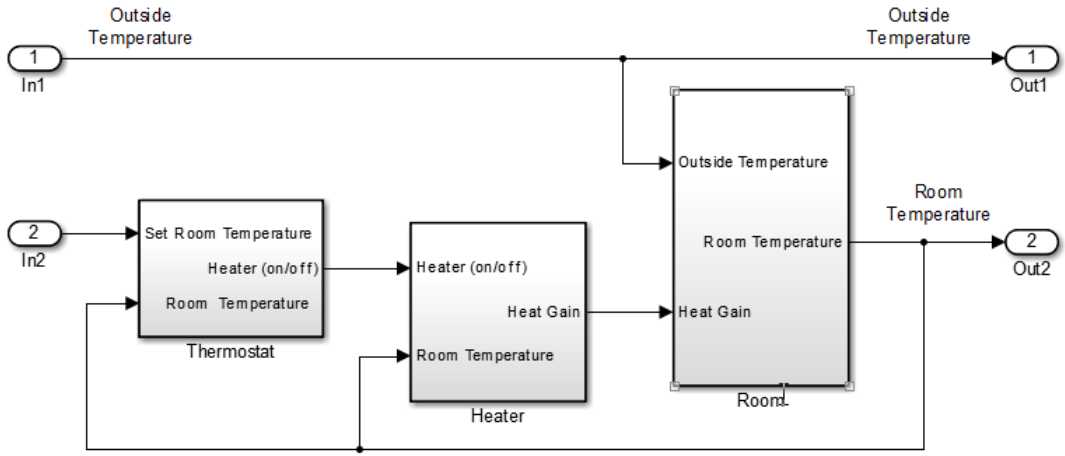
Model a Dynamic System

In this section...
“Define a House Heating System” on page 3-2
“Model a House Heating System” on page 3-8
“Integrate a House Heating Model” on page 3-25

This tutorial shows how to model a dynamic system using Simulink software. The model is a house heating system that includes a heater (plant), thermostat (controller), and room (environment).

To review a completed model, in the MATLAB Command Window, enter

```
open_system([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_modeling'])
```



Define a House Heating System

Define requirements and mathematical equations. Collect data for model parameters and validating the simulation results. For more information on how to define a system, see Basic Modeling Workflow: “Define System” on page 1-6.

Determine Modeling Goals

Before designing a model, consider your goals and requirements. The goals for modeling the house heating system are:

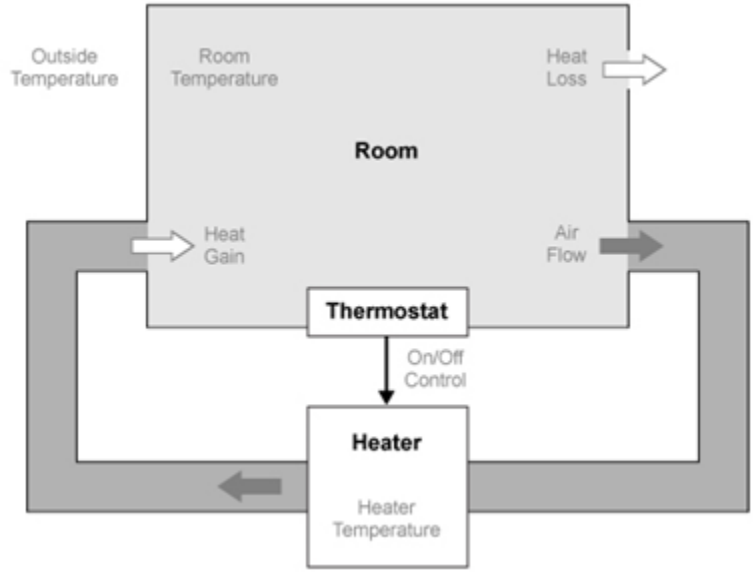
- Observe how the changing outdoor temperature affects the indoor temperature.
- Using experimentally measured outdoor and indoor temperatures, compare simulation results with measured values.

Identify System Components

The house heating system in this tutorial defines a heating system and its relationship to a room. It includes:

- Thermal characteristics of a house
- Thermal characteristics of a heater
- A thermostat to control the heater
- Outdoor environment
- Indoor environment

After you select a thermostat setting, the thermostat turns the heater on and off depending on the difference between the outside temperature and the room temperature.



The model for this system includes three components: heater, thermostat, and room.

Define System Equations

Three variables define the house heating model:

- Thermal energy transferred from the heater (Q_{gain}) to a room
- Thermal energy transferred from the room (Q_{loss}) to the outdoor environment
- Temperature of the room (T_{room})

A differential equation defines each of the variables, but since heat transfer is defined in terms of changing temperature, only room temperature is a state variable.

Rate of Heat Gain Equation

The temperature of the air in the heater is T_{heater} and the room temperature is T_{room}. Thermal energy gain to the room is by convection of heated air from the heater. Heat gain for a mass of air in the heater $m_{heaterair}$ is

$$Q_{gain} = m_{heaterair} c_{air} (T_{heater} - T_{room}).$$

A fan takes room air, and passes it through the heater and back to the room. The rate of thermal energy gain from the heater is

$$\frac{dQ_{gain}}{dt} = \frac{dm_{heaterair}}{dt} c_{air} (T_{heater} - T_{room}).$$

Since the mass of air per unit time from the heater is constant, replace $dm_{heaterair} / dt$ with a constant $M_{heaterair}$ and simplify the equation to

$$\frac{dQ_{gain}}{dt} = M_{heaterair} c_{air} (T_{heater} - T_{room}).$$

Rate of Heat Loss Equation

Thermal energy loss from the room is by conduction through the walls and windows:

$$Q_{loss} = \frac{kA(T_{room} - T_{outside})t}{D}.$$

The rate of thermal energy loss is

$$\frac{dQ_{loss}}{dt} = \frac{kA(T_{room} - T_{outside})}{D}.$$

Replacing kA / D with $1 / R$ where R is the thermal resistance simplifies the equation to

$$\frac{dQ_{loss}}{dt} = \frac{(T_{room} - T_{outside})}{R}.$$

Changing Room Temperature Equation

Define the rate of temperature change in the room by subtracting the rate of heat loss from the rate of heat gain:

$$\frac{dT_{room}}{dt} = \frac{1}{m_{roomair} c_{air}} \left(\frac{dQ_{gain}}{dt} - \frac{dQ_{loss}}{dt} \right).$$

Collect Data

Most of the parameter values needed for the house heating model are published in standard property tables. The flow rate for the heater is from a manufacturer data sheet.

List the variables and coefficients from your equations and check for dimensional consistency between the units. Since the unit of time for the model is hours, convert published values for the thermal property of materials from units of seconds to hours.

Equation Variables and Constants

You can use the constant names and values in this table when building the model.

Equation Variable or Coefficient	Description	Units
A	Area of wall or window surface `A_wall = 914, A_window = 6`	square meter
D	Depth of wall or window `D_wall = 0.2, D_window 0.01`	meter
Q	Thermal energy transferred	joule
dQ/dt	Rate of thermal energy transferred	joule/hour
k	Thermal conductivity; property of a material to conduct heat transfer `k_fiberglass = 136.8, k_glass = 2808`	joule/meter · hour · degree
r	Thermal resistivity; property of a material to resist heat transfer `r = 1/k`	meter · hour · degree/joule
R	Thermal resistance $R = D/kA = (T_1 - T_2)Q$	hour · degree/joule

Equation Variable or Coefficient	Description	Units

	`R_wall = 1.599e-6`, `R_window = 5.935e-7`
	
	`R_equivalent = (R_wall * R_window)/(R_wall + R_window) = 4.329e-7`	

m	Mass of air in the room or heater `m_room_air = 1470` The mass of the heater `m_heater_air` is not needed for this model.	kilogram
dm/dt	Rate of air mass passing through the heater	kilogram/hour
M	Constant rate of air mass passing through the heater `M_heater_air = 3600`	kilogram/hour
c	Specific heat capacity `c_air = 1005.4`	joule/kilogram · degree
T_{heater}	Constant air temperature from heater `T_heater = 50`	degree Celsius
T_{room}	Air temperature of room Initial air temperature of room `T_roomIC = 20`	degree Celsius

Model a House Heating System

Model the top-level structure and individual components. Organize your model into a hierarchical structure that corresponds to the components of the system. See Basic Modeling Workflow: “Model System” on page 1-9.

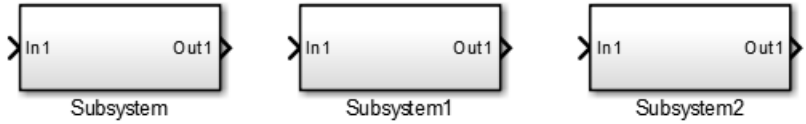
Model Top-Level Structure

At the top level of the house heating model, use Subsystem blocks to organize your model and create the structure. The model includes the subsystems Thermostat, Heater, and Room.

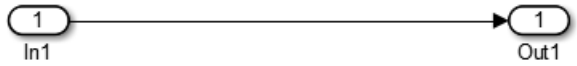
1. From the MATLAB Toolstrip, click the **Simulink** button, or in the **Command Window**, enter

   ```
   simulink
   ```

2. Click the Blank Model template, and then click the **Create Model** button.
3. From the Simulink Editor toolbar, click the **Library Browser** button.
4. Add Subsystem blocks. Drag three Subsystem blocks from the Ports & Subsystems library to the new model in the Simulink Editor.



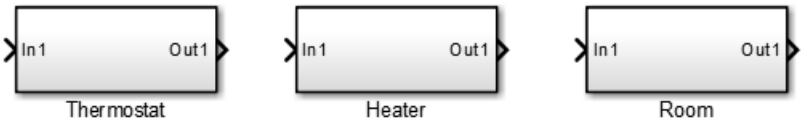
5. Open a Subsystem block. Double-click the block.



   Each new Subsystem block contains one Inport (In1) and one Outport (Out1) block. These blocks define the subsystem interface with the next higher level in a model hierarchy.

   Each Inport block creates an input port on the Subsystem block, and each Outport block creates an output port. Add more blocks for additional input and output signals.

6 On the Simulink Editor toolbar, click the **Up to Parent** button to return to the top level. Rename the Subsystem blocks as shown. Double-click a block name and type the new name.



Model Heater Component

Let's start by modeling the heater system component. The heater model:

- Takes the current temperature from the room and a control signal from the thermostat as inputs
- Calculates the heat gain from the heater
- Outputs the heat gain when the room temperature is lower than the selected room temperature

To model the heater subsystem, model the "Rate of Heat Gain Equation" on page 3-4 with Simulink blocks:

$$\frac{dQ_{gain}}{dt} = M_{heaterair} c_{air} (T_{heater} - T_{room}).$$

Subtract Room Air Temperature from Heater Air Temperature

The temperature difference is the current room temperature subtracted from the constant temperature of the heater (`T_heater`). See "Equation Variables and Constants" on page 3-6.

1 Open the Heater subsystem.

2 In the Heater subsystem, type Sum to display a list of blocks with Sum in the name. Double-click the Sum block on the list. When prompted for a list of signs, type `|-+` to place - and + input ports on the block, and press **Enter**.

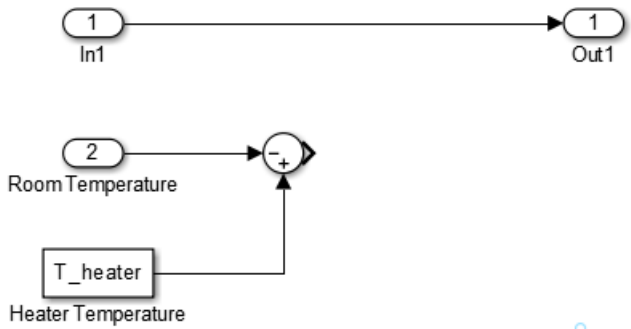
The vertical bar (|) changes the position of input ports by inserting spaces between the ports. A vertical bar at the beginning of the sign list, places a space at the top of the block and shifts the ports counter clockwise.

3 Add a Constant block to model the constant air temperature from the heater. Set the block **Constant value** parameter to `T_heater`. You will define the value of `T_heater` in the Model Workspace.

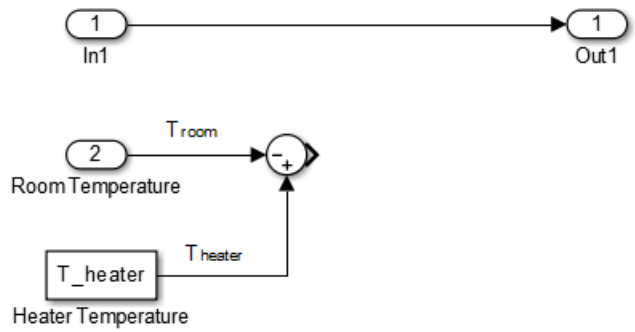
   If the block displays `-C-`, resize the block to display the variable name.

4 Add an Inport block to connect the room temperature signal to another part of your model.

5 Rename the blocks and connect them as shown in the figure.



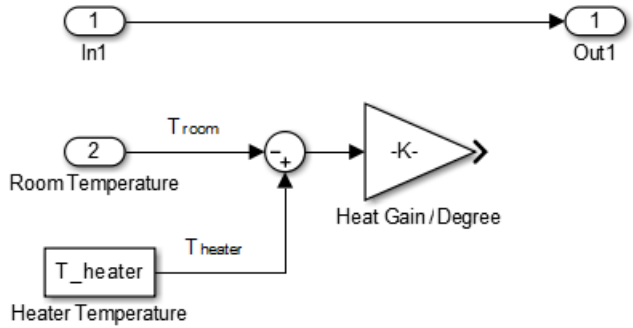
6 Add labels to the signal lines to help trace model components to the equations and model requirements. Double-click above a signal line and enter a label.



Multiply Temperature Difference by Heat Gain Per Degree

The heat gain per degree is the constant rate of mass from the heater (`M_heater_air`) multiplied by the specific heat capacity of air (`c_air`). See “Equation Variables and Constants” on page 3-6.

1. Add a Gain block to the Heater subsystem. Set the **Gain** parameter to `M_heater_air*c_air`. You will define the values of these variables in the Model Workspace.
2. Connect the output of the Sum block to the input of the Gain block.
3. Rename the Gain block to `Heat Gain/Degree`.



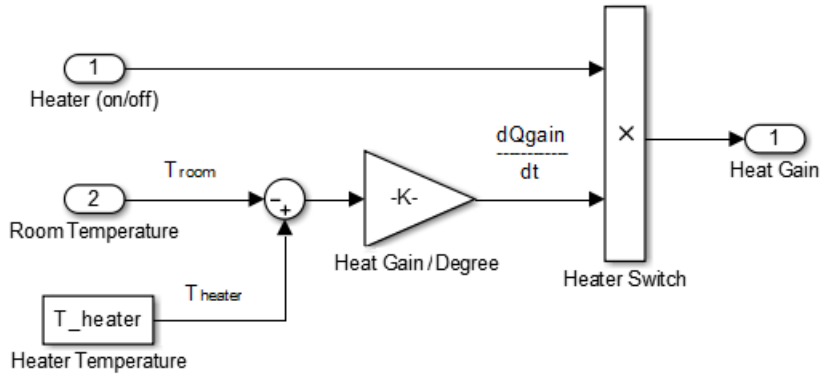
Model a Heater Switch

The thermostat sends an on/off signal equal to 1 (on) or 0 (off) to the heater. Because the input signal is binary, you can use a Multiplier block to model a switch.

1. Remove the connection between the In1 and Out1 blocks. Select the line and press **Delete**.
2. Add a Product block. Resize the block vertically to align the block in your diagram. Connect the In1 block to the first block input and the block output to the Out1 block. Rename the blocks as shown.

3 Connect the output from the Gain block to the second input. Move all the connected blocks together. Draw a selection box around the blocks you want to move, and then drag them to the new location.

4 Rename blocks and add labels to signals as shown in the figure.



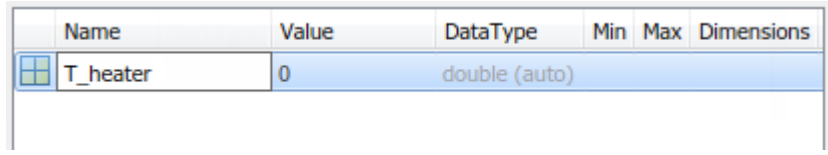
The Inport and Outport blocks create ports that connect this subsystem to other subsystems in your model.

Define Heater Model Parameters

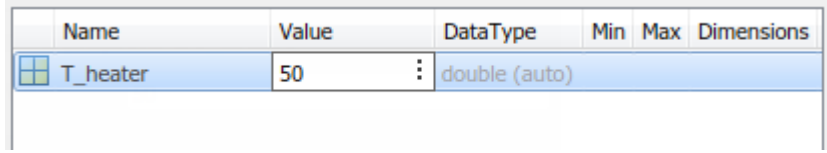
You can define variables in the MATLAB Workspace and then enter their names in the block parameter dialog boxes. However, a more robust method is to use the Simulink Model Workspace because variable values are saved with the model.

1 In the Simulink Editor, select **View** > **Model Explorer** > **Model Workspace**.

2 In Model Explorer, select **Add** > **MATLAB Variable**. In the middle pane, click the new variable `Var` and enter the variable name for a block parameter. For this example, enter `T_heater`.



3 Click the `0` value and enter the value for this variable. For this example, enter `50` degrees.

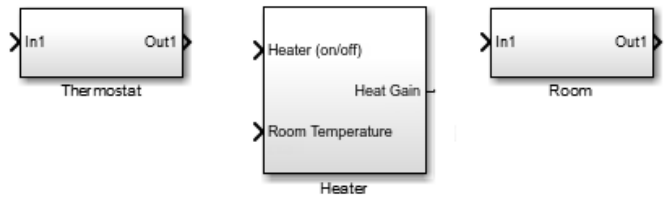


4 Using the same approach, add the variable `M_heater_air` with a value of `3600` kilogram/hour and `c_air` with a value of `1005.4` joule/kilogram · degree.

Prepare Heater Model for Simulation

Set up the heater model for simulation. Think about the expected behavior and how you can test that behavior with a simulation. When the thermostat output is 1, the expected output from the gain is (50 – 25) x 3600 × 1005.3 = 9.05×10^7.

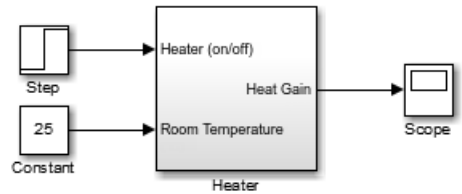
1 From the Heater subsystem, click the **Up to Parent** button to navigate to the top level of your model. You can resize the Heater block as shown in the figure.



Notice the Heater block has a second input port and that each port corresponds to an Inport block or Outport block in the subsystem.

2 Add a Constant block to represent the room temperature, and set the value to `25` (degrees Celsius). Add a Step block for a temporary Heater (on/off) signal. Set **Step time** to `4`.

3 Add a Scope block and connect it to the Heat Flow output. Connect the rest of the blocks as shown.



Simulate Heater Model and Evaluate Results

Use the default simulation settings to validate your model design.

1. Open the Configuration Parameters dialog box. In the Simulink Editor, select **Simulation** > **Model Configuration Parameters**.

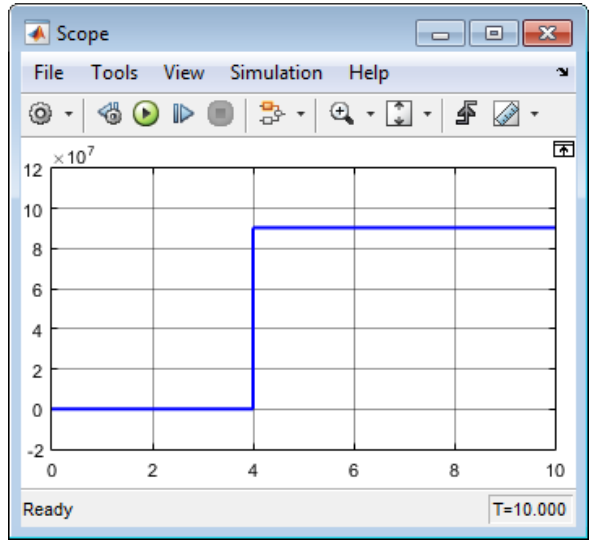
   Verify that **Stop time** is set to `10`, **Type** is set to `Variable-step`, and **Solver** to `Auto`.

2. Double-click the Scope block to open it.

3. Simulate the model. Select **Simulation** > **Run**, or click the **Run** button .

   As the simulation runs, the Scope plots the results.

4. View the scope trace. Right-click the `y`-axis and select **Configuration Properties properties**. On the Display tab, set **Y-limits (Minimum)** to `-2e7` and **Y-limits (Maximum)** to `12e7`.



5. Determine if this result is what you expected.

   When the heater on/off signal flips from 0 to 1 at 4 hours, the heater outputs 9.05×10^7 joule/hour. The simulation validates the expected behavior.

6 Remove Constant, Step, and Scope blocks you added for testing the Heater component.

Model Thermostat Component

You can model a thermostat without using system equations. Requirements for this component:

- When the room temperature is below the set temperature, the control signal equals 1. When the room temperature is above the set temperature, the control signal equals 0.
- There is a hysteresis of 2 degrees Celsius around the temperature set point. If the thermostat is on, the room temperature must increase 2 degrees above the set temperature before turning off.

This component models the operation of a thermostat, determining when the heating system is on or off. It contains only one Relay block but logically represents the thermostat in the model.

Subtract Set Room Temperature from Room Temperature

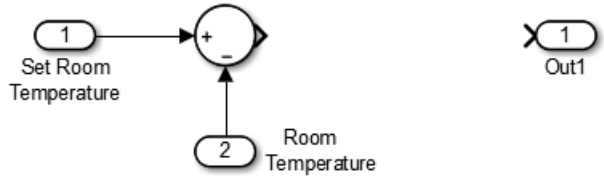
If the set room temperature is warmer than the room temperature, the thermostat model sends an "on" signal to the heater model. To determine if this is the case, begin by subtracting the room temperature from the set temperature.

1 Open the Thermostat subsystem. Add a Sum block to it. Set the **List of signs** parameter to `|+-` to place + and – input ports on the block.

2 Connect the Inport block to the + input of the Sum block. The Inport block sets the room temperature.

3 Add a second Inport block and connect it to the – input of the Sum block. This second Inport block is the current room temperature from the room subsystem. Move the output port to the top of the block. Right-click the block and select **Rotate & Flip** > **Counterclockwise**. If you want, you can reshape the block as shown in the figure by dragging the handles.

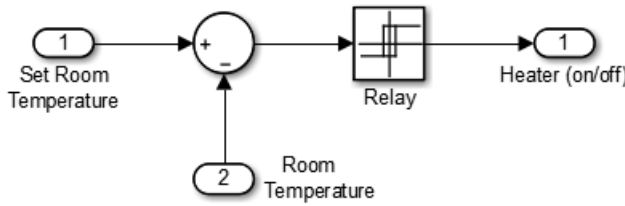
4 Rename the blocks as shown.



Model Thermostat Signal

Model the signal from the thermostat with a hysteresis value of 2 s Celsius.

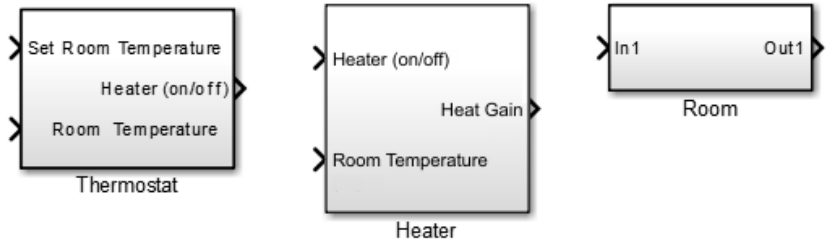
1. In the Thermostat subsystem, add a Relay block. Set the **Switch on point** parameter to `2`, and the **Switch off point** parameter to `-2`.
2. Connect and rename the blocks as shown in the figure.



Prepare Thermostat Model for Simulation

Prepare the Thermostat subsystem for simulation. Think about the expected behavior of the thermostat and how you can test that behavior with a simulation. When the room temperature rises above the thermostat setting by 2 degrees, the thermostat output is 0. When the room temperature moves below the thermostat setting by 2 degrees, the thermostat output is 1.

1. From the Thermostat subsystem, click the **Up to Parent** button to navigate to the top level of your model. Resize the Thermostat block as shown in the figure.



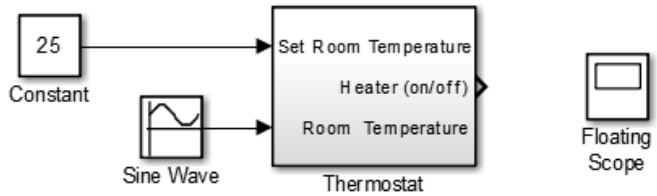
   Notice the Thermostat subsystem now has a second input port. Each input port corresponds to an Inport block in the subsystem.

2. Add a Constant block for setting the room temperature. Set the **Constant** parameter to `25` (degrees Celsius).

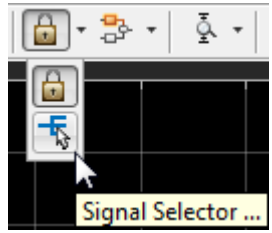
3 Add a Sine Wave block to represent the changing room temperature. Set the **Amplitude** parameter to `10`, the **Bias** to `20`, and the **Frequency** to `0.5`. These parameters give a variation above and below the temperature set point of 25.

4 Add a Floating Scope block to view signals.

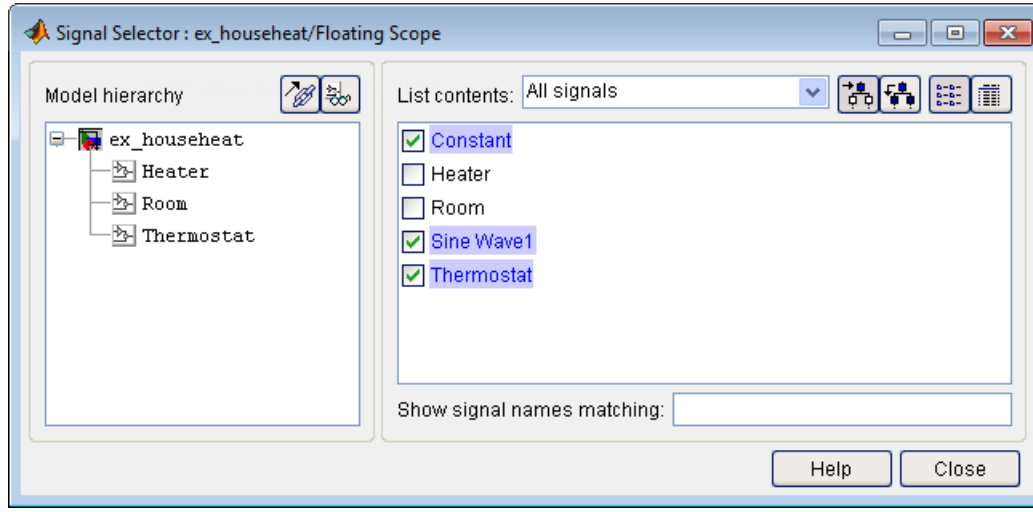
5 Connect the blocks as show in the figure.



6 Double-click the Floating Scope. From the toolbar, click the **Signal Selector** button.



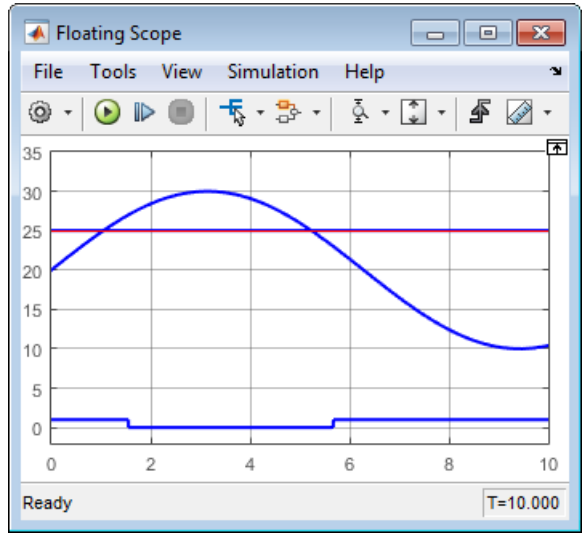
7 In the Signal Selector dialog box, select the **Constant**, **Sine Wave**, and **Thermostat** check boxes, and then click **Close**.



Simulate Thermostat Model and Evaluate Results

Use the default simulation settings to validate your model design.

1. Open the Configuration Parameters dialog box. In the Simulink Editor, select **Simulation** > **Model Configuration Parameters**. Verify **Stop time** is set to `10`, **Type** to `Variable-step`, and **Solver** to `ode45`.
2. Simulate the model. As the simulation runs, the Floating Scope plots the results.
3. Open the Floating Scope to view the scope trace. Right-click the display and select **Configuration Properties**. Select the **Display** tab, and then set **Y-limits (Minimum)** to `-2` and **Y-limits (Maximum)** to `35`.



4 Determine if this result is what you expected.

Initially the relay is on, and the room temperature is below the set temperature. When the room temperature increases above the set temperature, the relay does not initially switch to 0 until the room temperature increases by 2 more degrees. Simulation validates the expected behavior.

Model Room Component

Inputs to the room component are heat flow from the heater component and the external air temperature. The room component uses these inputs to compute heat loss through the walls, heat loss through the windows, and the current room temperature.

To design the room subsystem, use the “Rate of Heat Loss Equation” on page 3-5 and the “Changing Room Temperature Equation” on page 3-5.

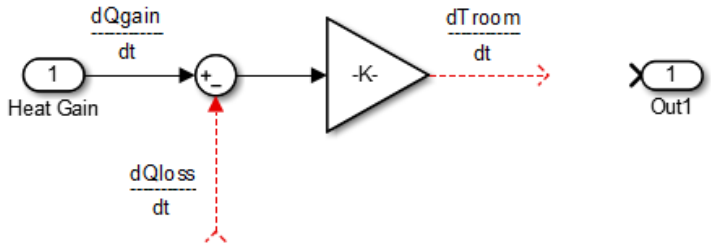
Model Changing Room Temperature

The rate of temperature change in the room (dTroom/dt) is defined by the equation

$$\frac{dT_{room}}{dt} = \frac{1}{m_{roomair}\, c_{air}} \left(\frac{dQ_{gain}}{dt} - \frac{dQ_{loss}}{dt} \right)$$

The term `dQgain/dt` is a signal from the Heater subsystem.

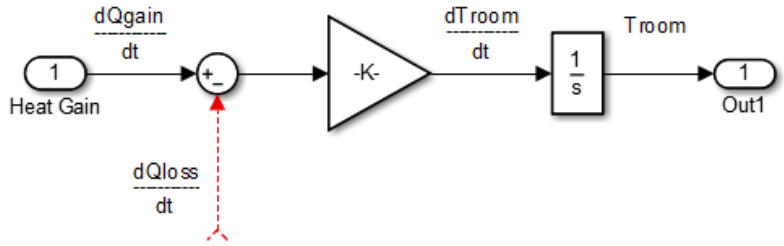
1. Open the Room subsystem block. In the Room subsystem, add a Sum block. Set the **List of signs** parameter to `|+–`.
2. Connect In1 to the + input. The input is the heat gain (`dQgain/dt`) from the heater component. The – input connects to the heat loss (`dQloss/dt`) from the room.
3. Add a Gain block. Set the **Gain** parameter to `1/(m_room_air*c_air)`. Connect the output of the Sum block to the input of the Gain block. Label signals as shown in the figure. Dotted signal lines are signals you will connect later.



Model Room Temperature

The output of the Gain block is the change in room temperature (`dTroom/dt`). To get the current room temperature (`Troom`), integrate the signal.

1. Add an Integrator block. Set the **Initial condition** parameter to `T_roomIC`.
2. Connect the output of the Integrator block to Out1 as shown.

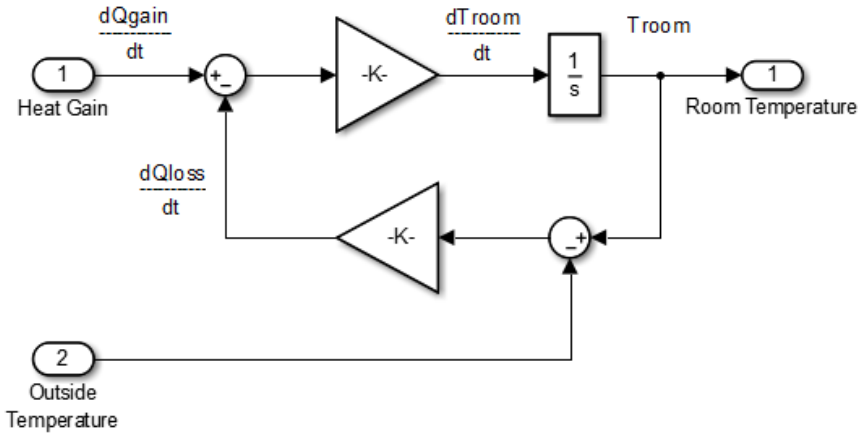


Model Heat Loss Through Walls and Windows

This equation is the rate of thermal energy loss through the walls and windows:

$$\frac{dQ_{loss}}{dt} = \frac{(T_{room} - T_{outside})}{R}.$$

1. In the Room subsystem, add a Sum block. Set the **List of signs** parameter to `|+-`. Right-click the block and select **Rotate & Flip** > **Flip Block**.
2. Click the signal line for `Troom`, press **Ctrl**, and then click and drag a branch signal line. Connect the line to the + input on the Sum block.
3. Add another Inport block and connect it to the – input of the Sum block. Rename it to Outside Temperature.
4. Add another Gain block. Set the **Gain** parameter to `1/R_equivalent`. Right-click the block and select **Rotate & Flip** > **Flip Block**.
5. Connect the blocks as shown in the figure.

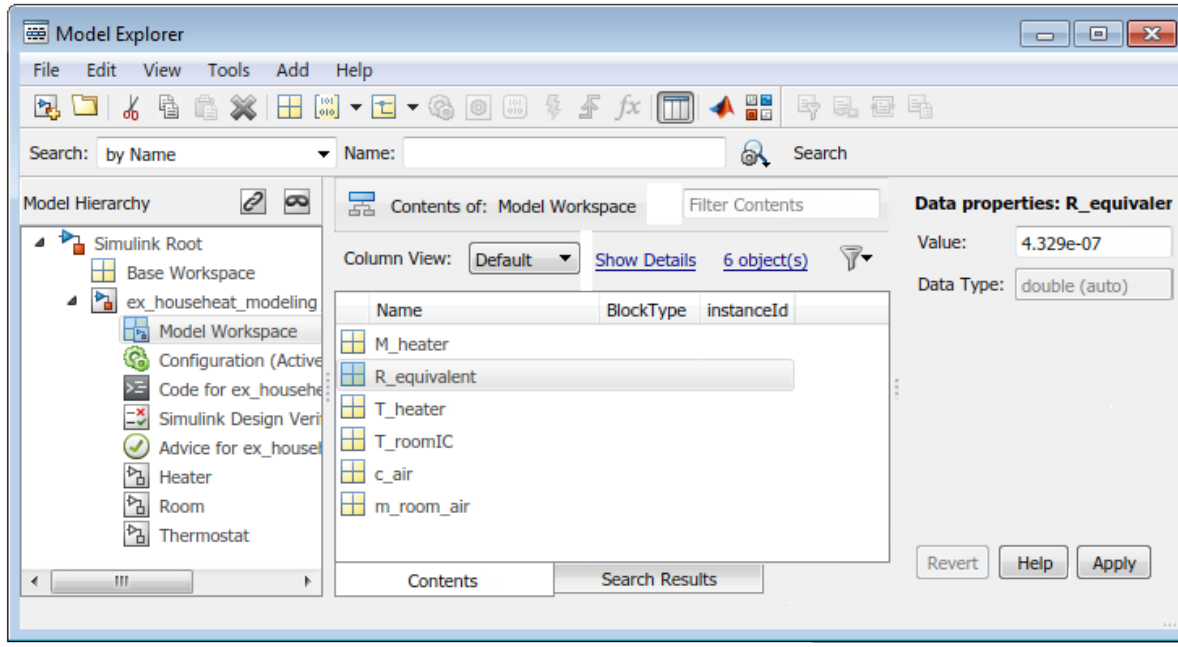


Define Room Model Parameters

You can define parameters in the MATLAB Workspace and then enter their names in the block parameter dialog boxes. However, a more robust method is to use the Simulink Model Workspace, which saves parameter values with the model.

1. In the Simulink Editor, select **View** > **Model Explorer** > **Model Workspace**.
2. In the Model Explorer, select **Add** > **MATLAB Variable**.
3. In the middle pane, click the new variable `Var` and enter the name `m_room_air`. In the right pane, enter the value `1470` (kilograms).

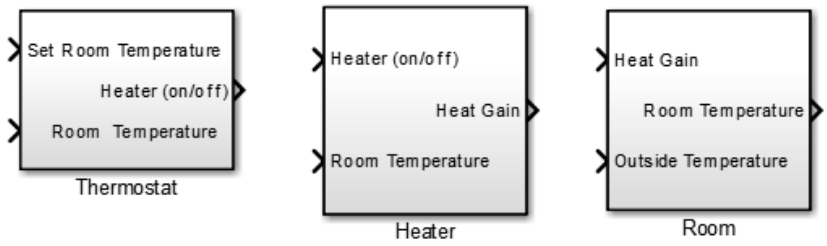
4 Add the variables `T_roomIC` = `20` (degrees Celsius) and `R_equivalent` = `4.329e-7` (hour · degree/joule).



Prepare Room Model for Simulation

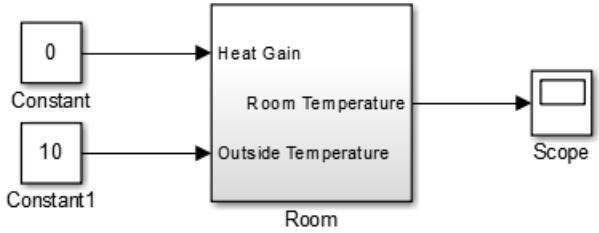
Prepare the Room subsystem for simulation. Think about the expected behavior and how you can test that behavior with a simulation. When the heater is off (Heat Gain = 0) and the initial temperature of the room (20) is above the outside temperature (10), heat loss should continue until the room temperature is equal to the outside temperature.

1 From the Room subsystem, click the **Up to Parent** button to navigate to the top level of your model. Resize the Room block as shown in the figure.



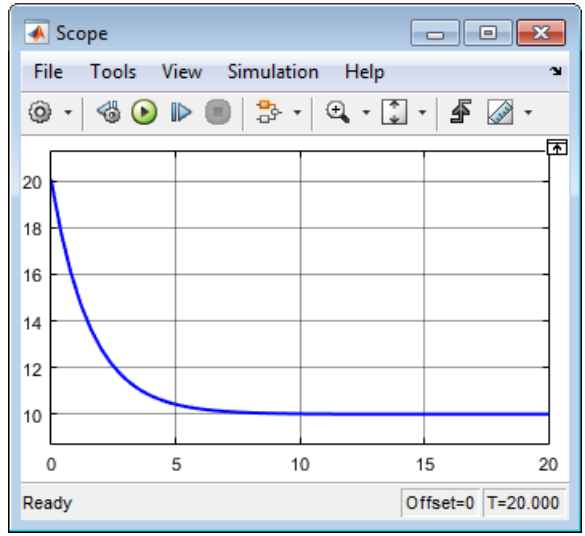
The Room block now has a second input port. Each inport port corresponds to an Inport block in the subsystem.

2. Add a Constant block and connect it to the Heat Gain input. Set the **Constant value** parameter to `0` (degrees Celsius) to mean that the heater is turned off.
3. Add another Constant block and connect it to the Outside Temperature input. Set the **Constant value** parameter to `10` (degrees Celsius).
4. Add and connect a Scope block to view the changing room temperature.



Simulate Room Model and Evaluate Results

1. In the toolbar, set the **Simulation stop time** to `20`.
2. Simulate the model.
3. Open the Scope and click the **Autoscale** button to view the scope trace.



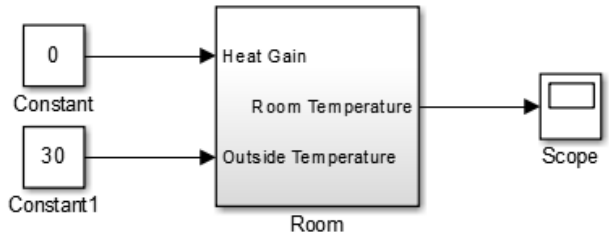
4 Determine if this result is what you expected.

The room temperature starts at the initial room temperature set in the Integrator block. Because the heat gain is 0, the signal decays to the outside temperature (10). The simulation validates the expected behavior.

Prepare Room Model for Second Simulation

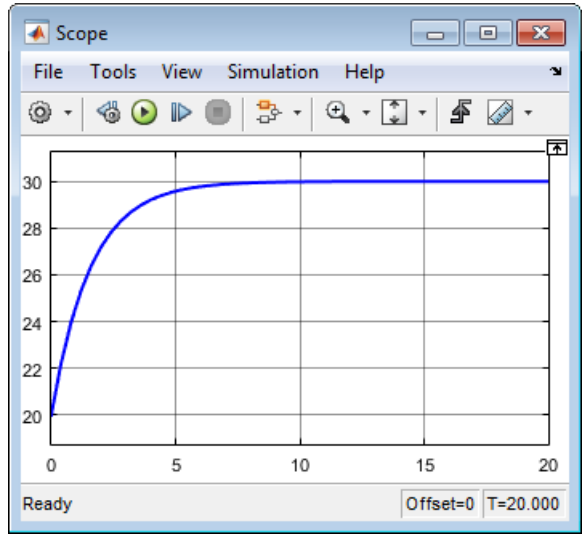
Set the constant outside temperature to a value above the initial room temperature (20).

1 In the Constant block that is connected to the Outside Temperature input, set **Constant value** to 30 (degrees Celsius).



Simulate Model and Evaluate Results

1. Simulate the model.
2. Open the Scope and click the **Autoscale** button to view the scope trace.



3. Determine if this result is what you expected.

   Room temperature starts at the initially set temperature of 20, but with the heater off (heat gain = 0) the room temperature rises to the outside temperature.

   This result was unexpected because you didn't explicitly model this behavior. However, the equation that models the heat loss also models the heat gain when the outside temperature is above the inside room temperature.

Integrate a House Heating Model

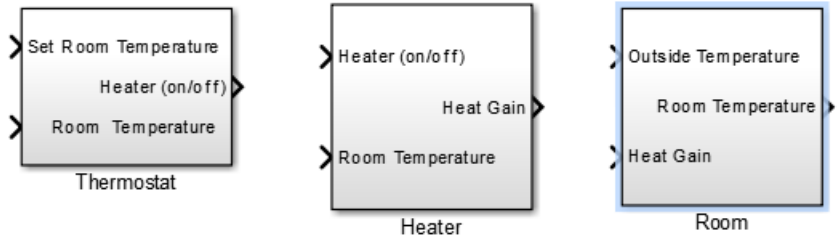
Connect model components, add realistic input, and then simulate the model behavior over time to validate the design. See Basic Modeling Workflow: "Integrate Model" on page 1-11.

Integrate Heater and Thermostat Components

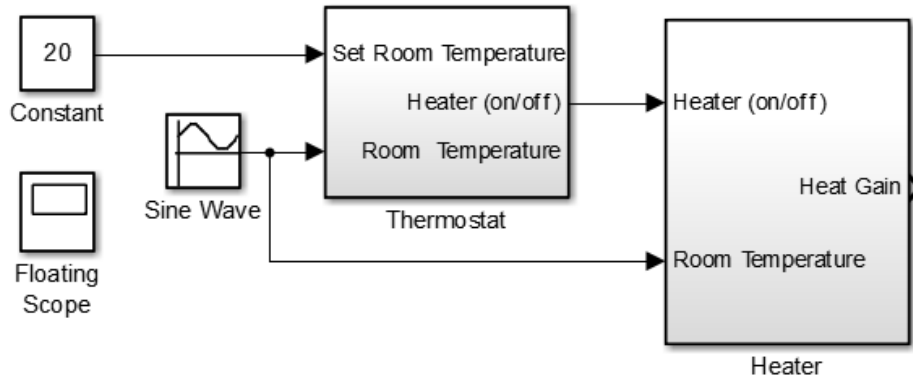
To simulate the heater and thermostat subsystems without the Room subsystem, you need a signal for the changing room temperature. Use a Constant block to set the thermostat temperature and a Sine Wave block for a realistic outside temperature signal.

Prepare Model for Simulation

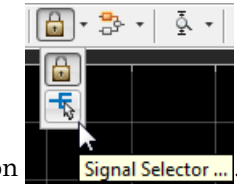
1. Open your model with the completed subsystems. Remove any blocks you added to test the separate components.
2. Open the Room subsystem. Double-click the Inport block labeled Heat Gain. In the Inport dialog box, set **Port number** to 2. The Heat Gain port moves to the bottom of the Room subsystem.



3. Connect the Heater (on/off) signal from the Thermostat subsystem output to the Heater subsystem input.
4. Add a Constant block to set the thermostat room temperature. Set **Constant** value to 20 (degrees Celsius).
5. Add a Sine Wave block to represent the changing room temperature. Set the parameters **Amplitude** to 10 (degrees Celsius), **Bias** to 15, and **Frequency** to 0.5.
6. Add a Floating Scope block and connect the blocks as shown in the figure.



7 On the Floating Scope window, click the **Configuration Parameters** button . In the **Main** tab, click the **Layout** button. Select two boxes. A second empty graph appears below the first.

8 From the toolbar, click the **Signal Selector** button 

.

9 In the Signal Selector dialog box, from the **Select signals for object** drop-down list, select `Display 1`. Select the **Heater** check box.

10 From the **Select signals for object** drop-down list, select `Display 2`. Select the **Constant** and **Sine Wave** check boxes. Click **Close**.

11 In the Floating Scope window, right-click the top display and select **Configuration Properties**. Set **Y-limits (Minimum)** to `0` and **Y-limits (Maximum)** to `20e7`.

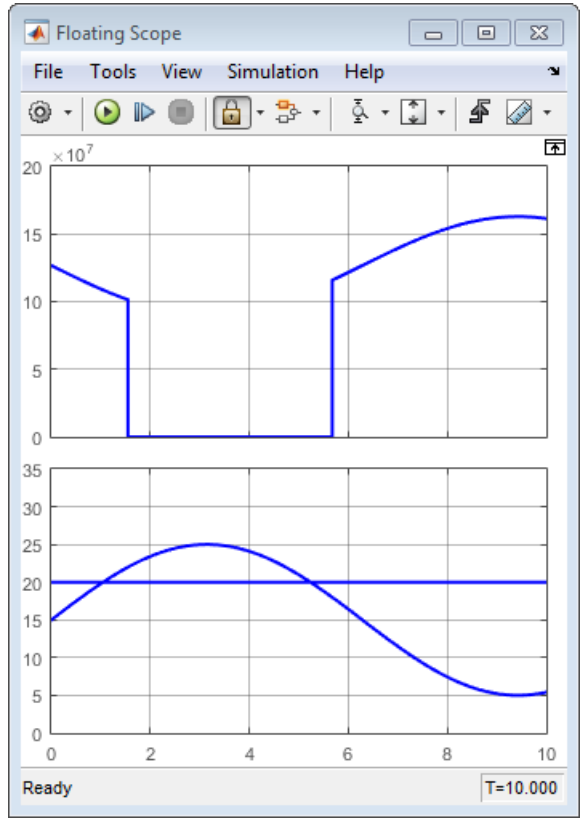
12 Right-click the bottom display and select **Configuration Properties**. Set **Y-limits (Minimum)** to `0` and **Y-limits (Maximum)** to `35`.

Simulate Model and Evaluate Results

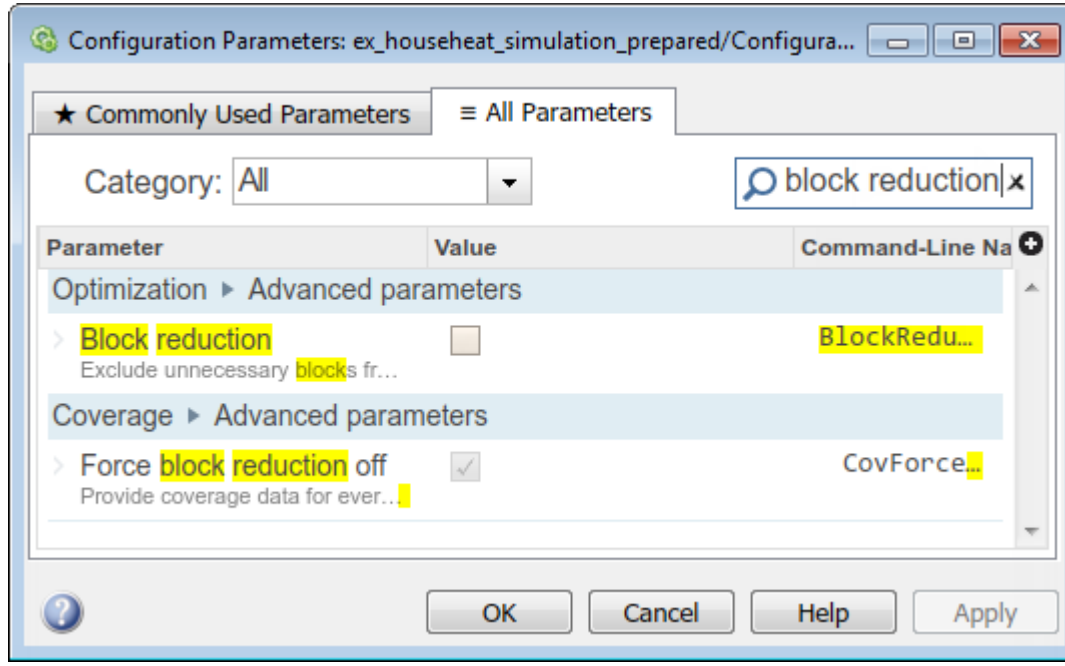
Simulate the model using the default stop time of 10 and solver ode45 solver.

1 Simulate the model.

2 Open the Floating Scope and view the simulation results. The top graph is the heater gain while the lower graph shows the changing room temperature modeled with a sine wave.



Note: If you get an error indicating Simulink cannot display a signal due to block reduction optimization, clear the **Block reduction** check box in the Configuration Parameters dialog box.



3 Determine if this result is what you expected.

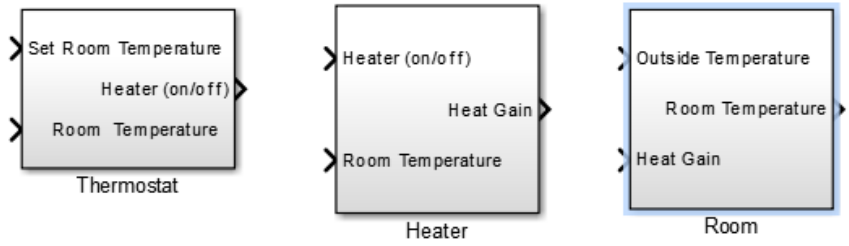
From about 0 to 1.5 hours, the heater is turned on. Heat gain is not constant but changes because heat gain is a function of the difference between the heater air temperature and the room air temperature. From 1.5 to 5.6 hours, the heater is turned off and the heat gain (top graph) is zero. The simulation confirms the expected behavior.

Integrate Room Component

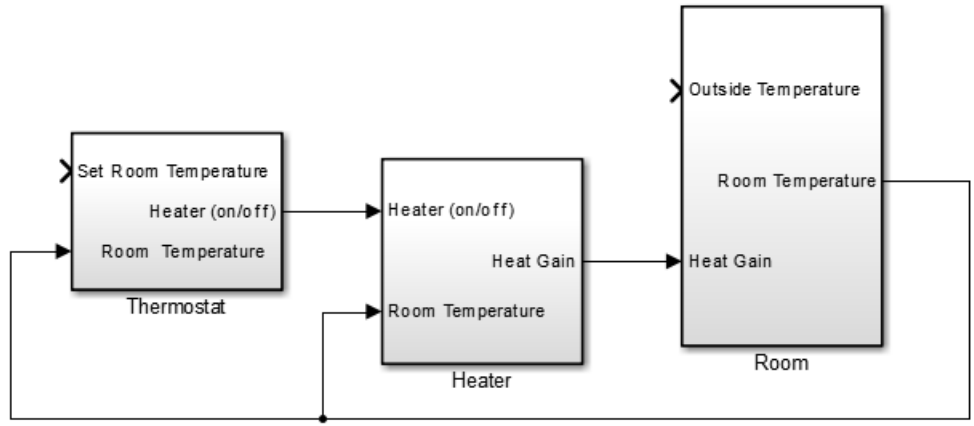
To simulate the Heater and Thermostat subsystems with the Room subsystem, you need a signal for the changing outside temperature. Simulating the model allows you to observe how the thermostat setting and outdoor temperature affect the indoor temperature.

Prepare Model for Simulation

1 Open your model with completed subsystems. Remove any blocks you added to test the separate components.



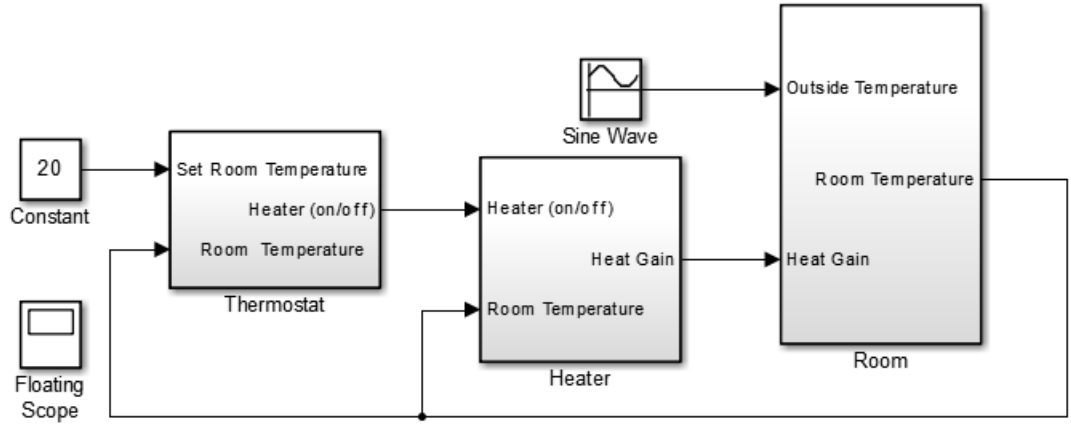
2 Connect the subsystems as shown.

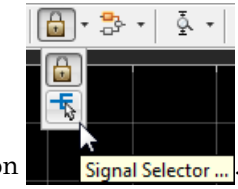


3 Add a Constant block for setting the room temperature. Set **Constant value** parameter to `20` (degrees Celsius).

4 Add a Sine Wave block to represent the changing outside temperature. Set **Amplitude** to `5`, **Bias** to `12`, **Frequency** to `2*pi/24`, and **Phase** to `180`.

5 Add a Floating Scope block to view simulation results.



6 

In the Floating Scope, click the **Signal Selector** button . In the Signal Selector dialog box and in the left pane, select the top model hierarchy. In the right pane, select the Room and Sine Wave signals.

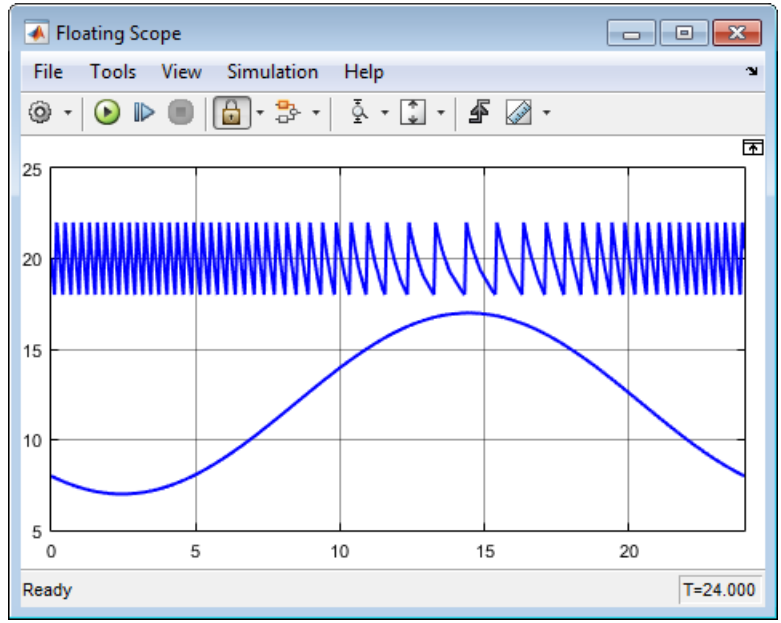
7 In the Floating Scope, right-click the graph and select **Configuration Properties**. Set **Y-limits (Minimum)** to 5 and **Y-limits (Maximum)** to 25.

Simulate Model and Evaluate Results

1 Set the simulation stop time to 24 (hours) to represent a day.

2 Simulate the model.

3 Open the Floating Scope and view results.



4 Determine if the simulation result matches your expectation.

When the outside temperature is below the set room temperature, the room temperature fluctuates 2 degrees above and below the set temperature. Since the thermostat subsystem includes a 2 degree hysteresis, this simulation result is expected.

5 You can compare your results with an example model. In the MATLAB Command Window, enter

```
open_system([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_modeling_prepared'])
```

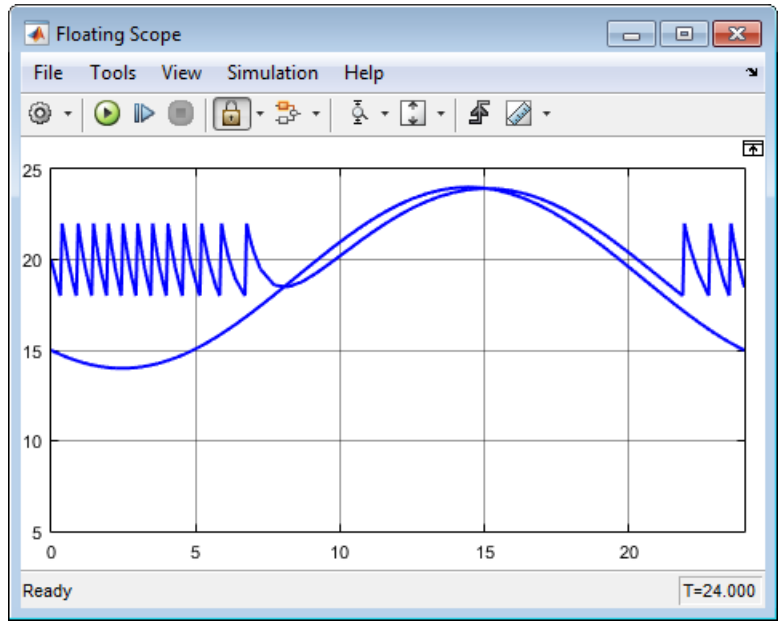
or click `ex_househeat_modeling_prepared.slx`.

Refine Model Parameters

With Simulink models, you can interactively change model parameters and then observe changes in the behavior of your model. This approach allows you to evaluate your model quickly and validate your design.

Change the outside temperature in the Sine Wave block so that upper values are above the thermostat temperature.

1. In the Sine Wave dialog box, set **Amplitude** to 5 and **Bias** to 19. These settings show what happens when outside temperature is higher than inside temperature.
2. Simulate the model and view the results.



3. Determine if the results match your expectations.

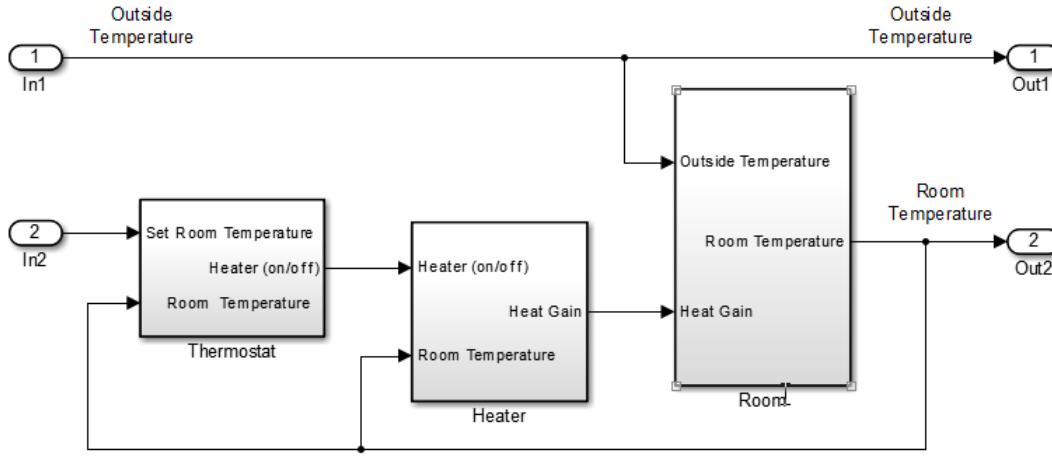
   When the outside temperature is above the set thermostat temperature, the room temperature is about equal to the outside temperature. In this case, heat loss is a loss of heat from the outside environment into the room.

Model External Interface

Model the external interface for further testing and possible use in a larger model. In Simulink, you model the external interface using Inport and Outport blocks.

1. Add Inport blocks to read data from the outside temperature and thermostat set temperature into your model.

2 Add Outport blocks to connect the outside temperature and room temperature to a larger model or to visualize results.



Specify Physical Units

By specifying physical units for model signals, you ensure the consistency of calculations across model components. In Simulink, you specify signal units through Inport and Outport blocks.

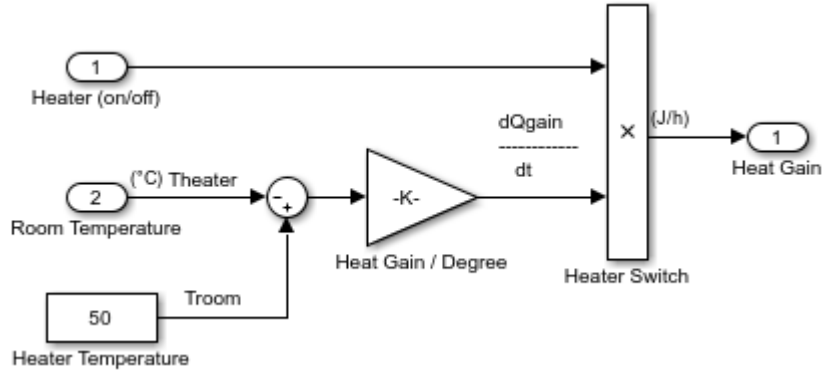
1 Double-click the `In1` block to open the Block Parameters dialog box. Select the **Signal Attributes** tab.

2 In the **Unit** box, start typing `degree`. From the list of symbols and names, select `°C degree_Celsius`.

For the remaining temperature Inport and Outport blocks, set the **Unit** parameter to `°C degree_Celsius`.

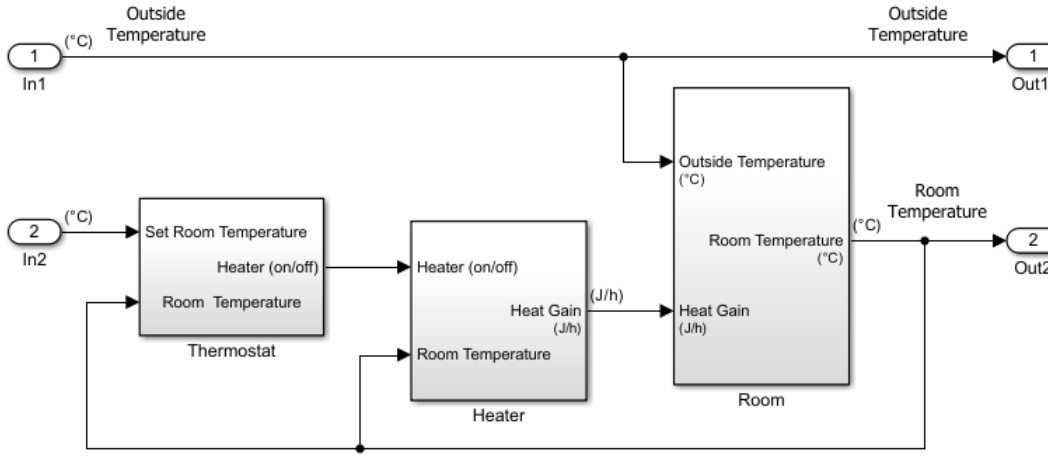
3 Display units on block ports. From the menu, select **Display** > **Signals & Ports** > **Port Units**.

4 Double-click the Heater Subsystem block. Double-click the Heat Gain Outport block to open the Block Parameters dialog box. Select the **Signal Attributes** tab.



5 In the **Unit** box, start typing `joule/hour`. From the list of symbols and names, select `joule/h joule/hour`.

6 Compile the model and perform unit consistency checking. Press **Ctrl+D**.



Your next step is to verify the correctness of the model by comparing simulations with real system data. See “Simulate a Dynamic System” on page 4-2.

More About

- “Model-Based Design” on page 1-3
- “Basic Modeling Workflow” on page 1-6

- “Basic Simulation Workflow” on page 1-13

4

Basic Simulation Workflow

Simulate a Dynamic System

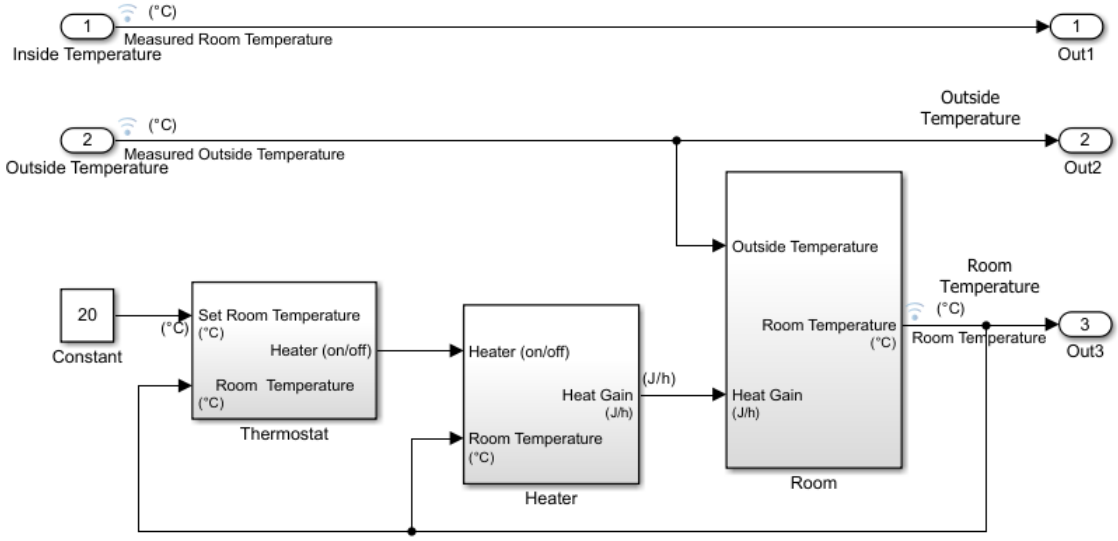
In this section...
“Prepare for Simulation” on page 4-3
“Run and Evaluate Simulation” on page 4-6

This tutorial shows how to simulate the model of a dynamic system using Simulink software, and then use the results to improve the model. After you prepare the model for simulation, you can use an interface to input measured system data and set room temperature.

To review an example model that is already prepared for simulation, in the MATLAB Command Window, enter

```
load([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_measured_data.mat'])

open_system([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_simulation_prepared'])
```



Prepare for Simulation

Verify that a simulation represents the behavior of the system you modeled. Begin by experimentally measuring physical characteristics of the system that have comparable signals in your model:

- Collect data from physical system
- Prepare model for simulation

For a detailed description of the workflow, see “Prepare for Simulation” on page 1-13.

Collect and Plot System Data

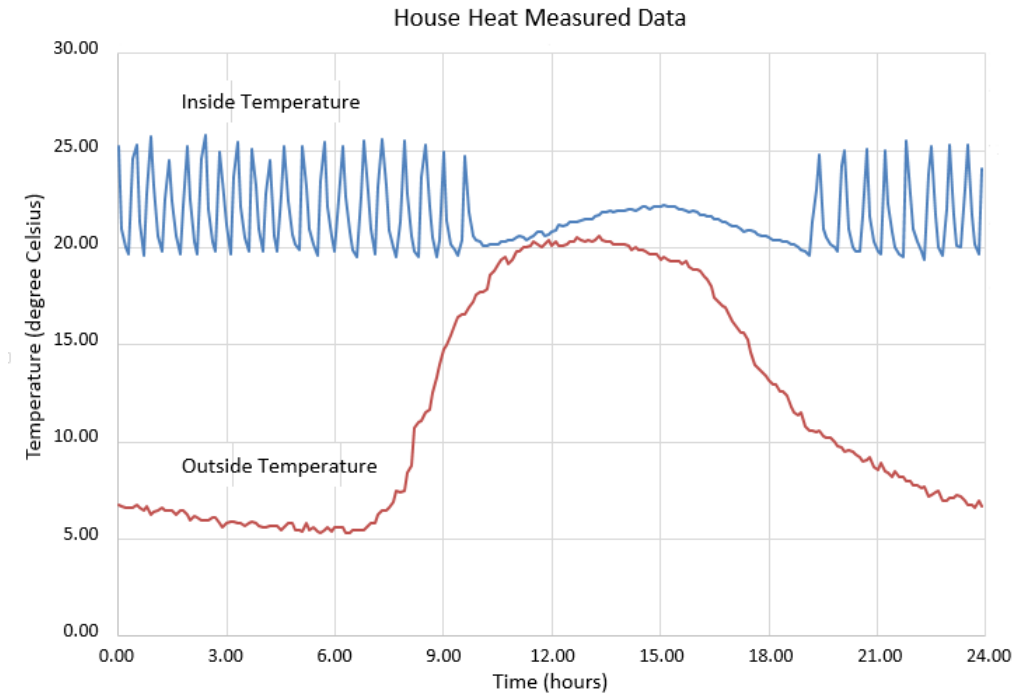
Measure the dynamic characteristics from an actual house heating system. You will use the measured data with model simulations to verify the behavior and accuracy of your model.

1. Measure the outside and inside temperatures of a house every 6 minutes for 24 hours.
2. Enter the measured data into a Microsoft Excel worksheet or open an example spreadsheet. In the MATLAB Command Window, enter

```
winopen([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_measured_data.xls'])
```

	A	B	C
1	Time (hours)	Inside Temperature	Outside Temperature
2	0.00	25.20	6.60
3	0.10	21.00	6.50
4	0.20	20.00	6.60
5	0.30	19.70	6.60
6	0.40	24.60	6.60
7	0.50	25.30	6.40
8	0.60	21.30	6.60
9	0.70	19.80	7.00
10	0.80	22.80	6.70

3 Review a plot of the measured data. The inside temperature data shows temperature spikes when the hot air heater turns on. This pattern is typical for a hot air heating system.

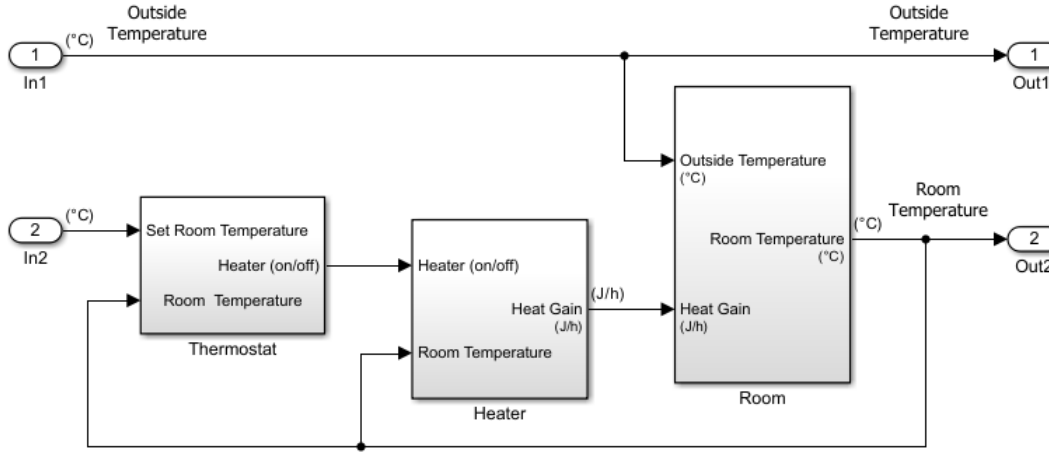


Prepare Model for Simulation

Prepare a model for simulation by adding an external interface for data input and input control signals.

1 Use the model you created in the tutorial "Model a Dynamic System" on page 3-2 or open the example model. In the MATLAB Command Window, enter

```
open_system([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_modeling'])
```

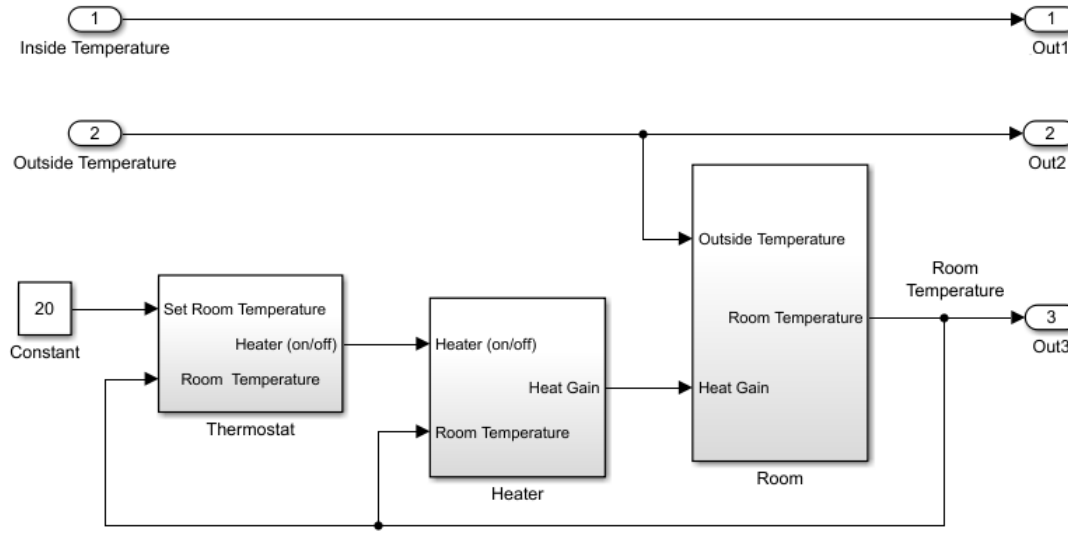


2 Replace the Inport block `In2` with a Constant block and set the **Constant** parameter to `20`. The Constant block sets the thermostat temperature.

3 Add an Inport block. Set **Port number** to `1`. This action also sets the **Port number** of the outside temperature signal to `2`.

4 Rename the first Inport block to `Inside Temperature`. Rename the second Inport block to `Outside Temperature`.

5 Add an Outport block and connect it to the first Inport block (Inside Temperature). The outport blocks are needed for saving (logging) the signals.



6 Save the model.

Run and Evaluate Simulation

Verify the accuracy of the model and optimize parameters. Some parameters to consider for optimization are heater hysteresis, temperature offset, and the resistance of the house to heat loss. Follow these steps to verify your model:

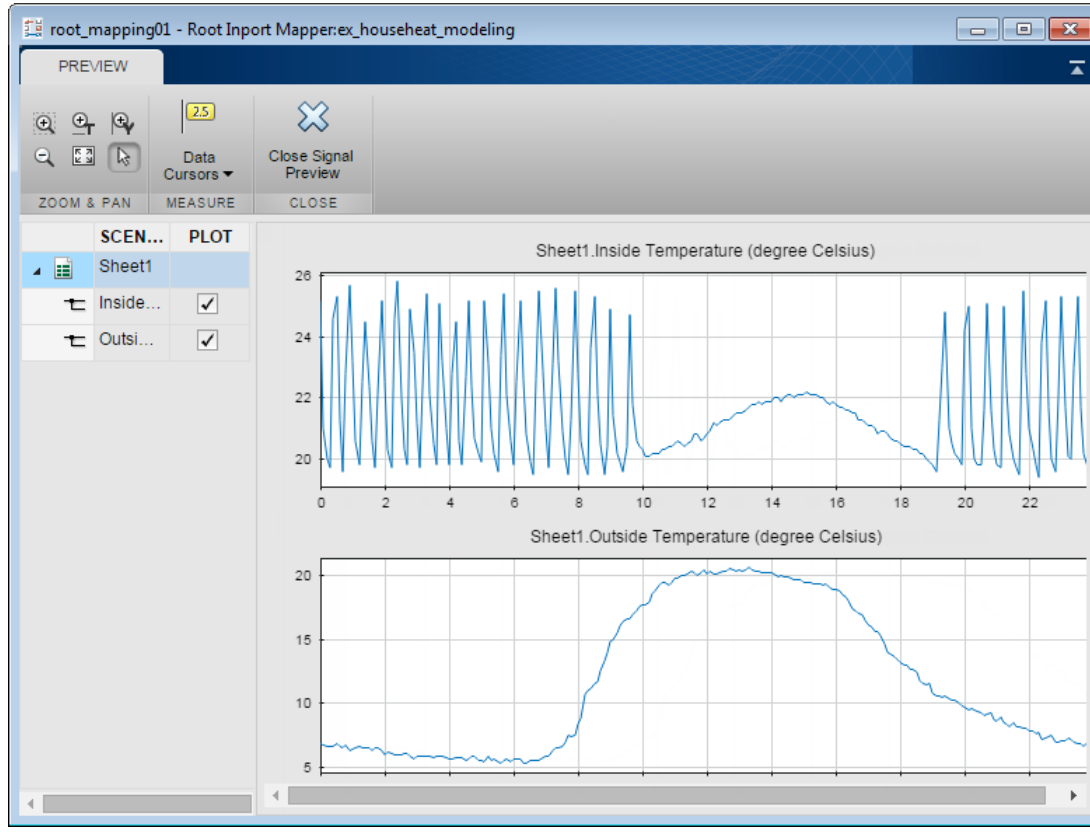
- Import Data
- Run Simulation
- Evaluate Simulation Result
- Change Model Parameters
- Rerun Simulation

For a detailed description of the workflow, see “Run and Evaluate Simulation” on page 1-15.

Import Data with Root Inport Mapping

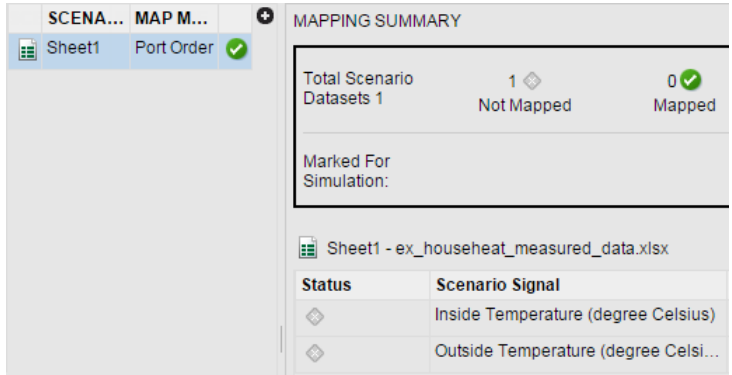
You can use the Root Inport Mapper tool to bring measured signal data from an Excel spreadsheet into a Simulink model.

1. Open any Inport block. Click the **Connect Input** button to open the Root Inport Mapper.
2. On the toolstrip, click **From Spreadsheet**.
3. In the From Spreadsheet dialog box, click the browse button. Browse to and select the file *`matlabroot`*`\help\toolbox\simulink\examples\ex_househeat_measured_data.xls`. Click **Open**. Click **OK** to import the spreadsheet.
4. From the **Signals** drop-down list, select **Preview Signals**.
5. On the left side, expand the tree view of `Sheet1`. Select the Inside Temperature and Outside Temperature check boxes.



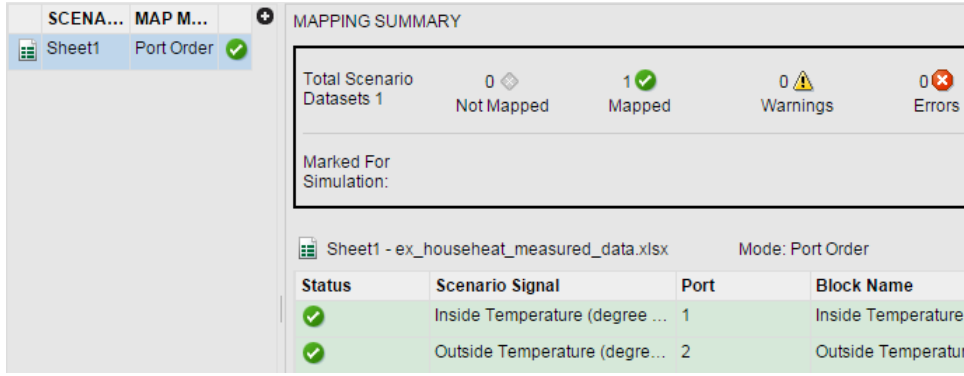
6. Click **Close Signal Preview**.

7 On the left side, select `Sheet1`. The **Scenario Signal** column shows the two signals from the Excel spreadsheet and an icon indicating the signals are unmapped.

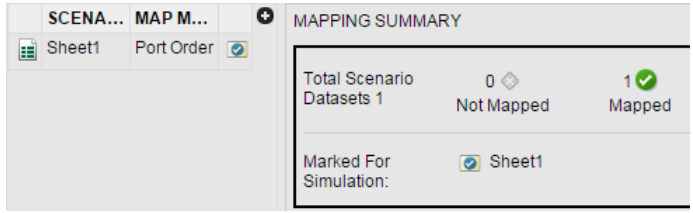


8 On the toolstrip, select the **Port Order** option. From the **Options** drop-down list, select the **Update Model** check box.

9 Click **Map to Model**, and then from the drop-down list, select **Map Unconnected**. The mapping summary shows the signals from the Excel spreadsheet mapped to the Input port blocks.



10 Click **Mark for Simulation**. The mapping summary shows `Sheet1` is marked for simulation and a `Dataset` object is created in the MATLAB Workspace.



11 Save the signal data in a MAT-file. In the MATLAB Command Window, type

```
save('ex_househeat_measured_data.mat', 'Sheet1')
```

Configure Model to Load Signal Data

Signal data mapped to input ports is located in a MATLAB workspace variable. With each new MATLAB session, you have to manually reload the data or let the model preload function do it for you.

1 From the Simulink Editor menu, select **File > Model Properties > Model Properties**.

2 Select the **Callbacks** tab.

3 In the Model callbacks section, select PreLoadFcn.

4 In the Model pre-load function box, enter

```
load('ex_househeat_measured_data.mat')
```

5 Click **OK**.

Configure Model to Save Simulation Results

Configure your model to save (log) signal data during a simulation. You can then view logged signals from a simulation using the Simulink Data Inspector.

1 In the model, select **Simulation > Model Configuration Parameters**. In the left pane, select **Data Import/Export**.

2 In the right pane, clear the **Time** and **Output** check boxes.

3 Select the **Signal logging** check box.

4 Select the **Record logged workspace data in Simulation Data Inspector** check box.

5 Click **OK**.

Select Signals to Save

Identify signals to display in the Simulink Data Inspector, name the signals if they are unnamed, and set the logging parameters.

1. Right-click the Inside Temperature signal line and select **Properties**.
2. In the **Signal name** box, enter `Measured Room Temperature`. Select the **Log signal data** check box. A logging badge appears above the signal line.
3. Name and select logging for these signals.

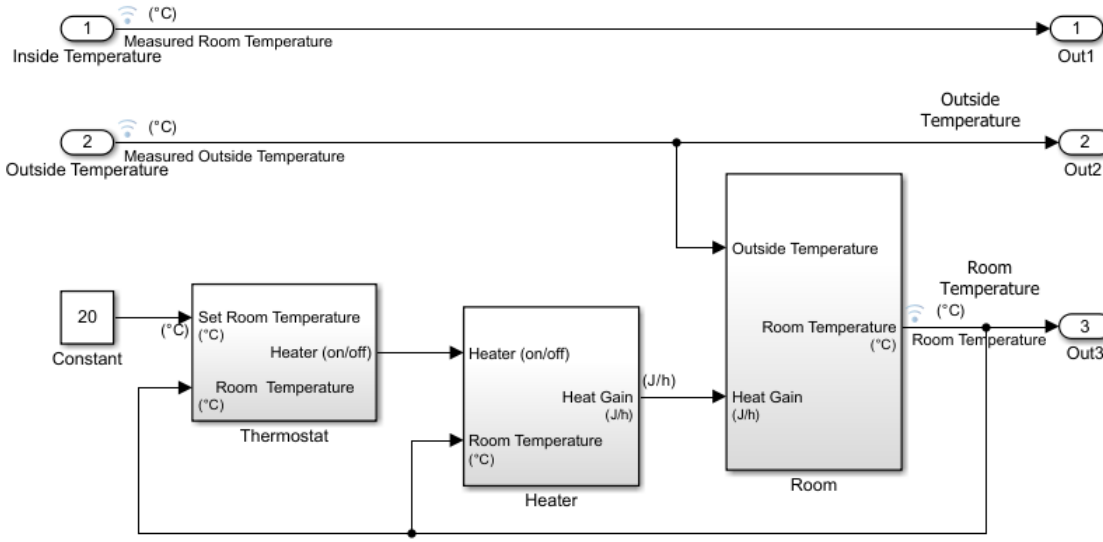
Location of signal	Signal name
Outside Temperature from output port 2.	`Measured Outside Temperature`
Room Temperature from Room subsystem output port	`Room Temperature`

Run Simulation

After importing data and enabling logging of data for the signals, you can run a simulation.

1. Use the model you prepared for simulation or open the example model. In the MATLAB Command Window, enter

```
open_system([matlabroot,...
'/help/toolbox/simulink/examples/ex_househeat_simulation_prepared'])
```



2 In the model, select **Simulation** > **Model Configuration Parameters**. In the left pane, select **Solver**. Set **Stop time** to `24` (hours), **Type** to `Variable-step`, and **Solver** to `ode45`. Click **OK**.

3 Click the **Run** button .

The model simulation runs from `0.0` to `24.0` hours using the outside temperature data from the root import block as input.

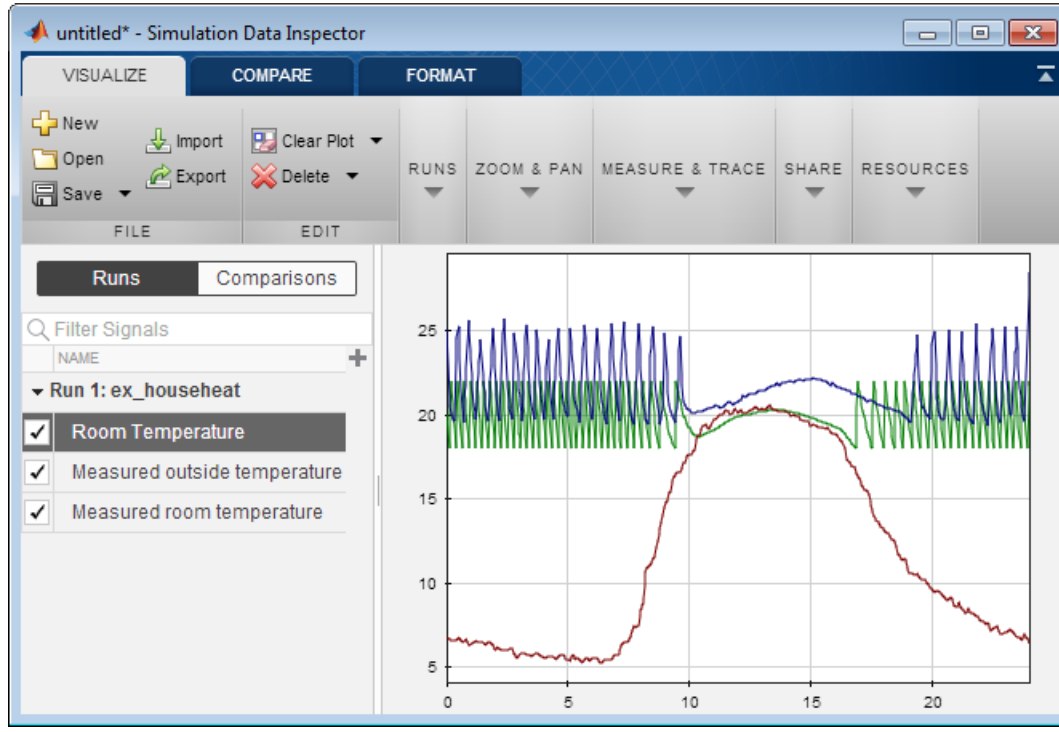
Compare Simulation Results with Measured System Data

Use the Simulink Data Inspector to compare the simulated output signals with measured data.

1 In the Simulink Editor toolbar, click the **Simulation Data Inspector** button .

A separate run appears in the **Runs** pane each time you simulate the model.

2 Select all the signal check boxes. As you select a signal, a plot is drawn on the graph.



The top signal is the Measured Room Temperature. The middle signal is the simulated Room Temperature. The bottom signal is the Measured Outside Temperature.

Determine Changes to Model

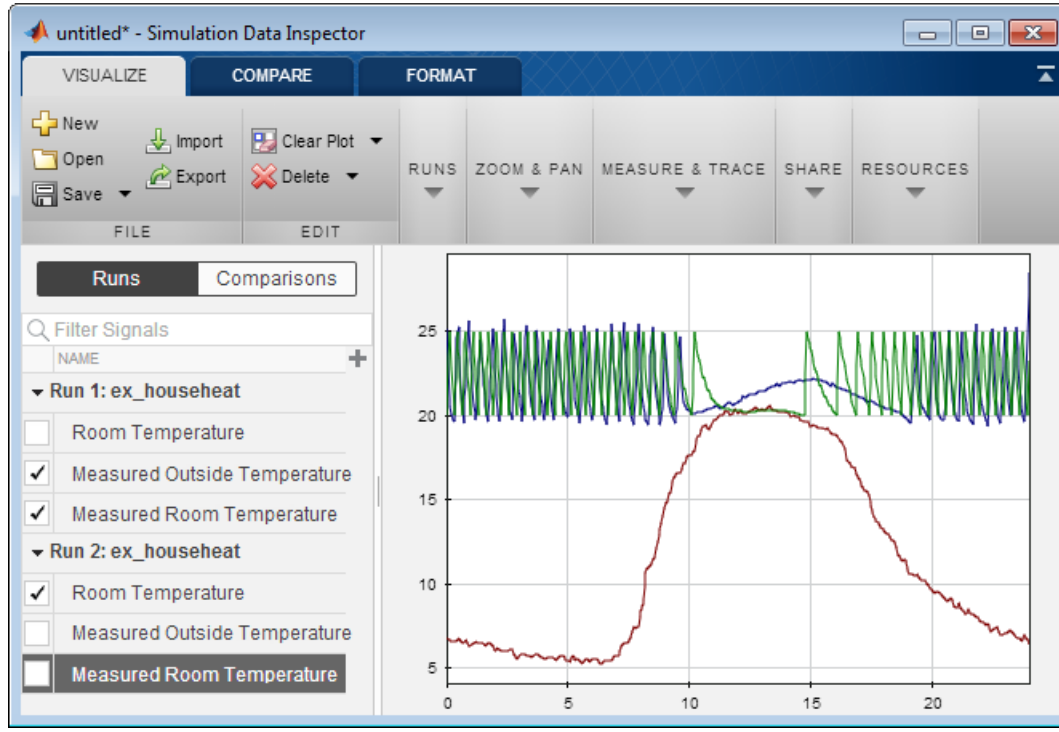
One obvious change to the model is the hysteresis of the thermostat. The simulated room temperature oscillates 18–22 degrees around the temperature set point of 20 degrees. The measured room temperature oscillates 20–25 degrees with the same set point.

1. Open the Relay block in the Thermostat subsystem.
2. Change **Switch on point** from `2` to `0` because the difference between the room temperature and set point is 0.
3. Change **Switch off point** from `-2` to `-5`. When the room temperature is 5 degrees above the set point, you want to turn off the heater. The set point is –5 degrees below the room temperature.

Compare Results Between Simulations

Use the Simulation Data Inspector to compare differences between two simulations that use different model parameters. This comparison shows how changes improve the accuracy of your model.

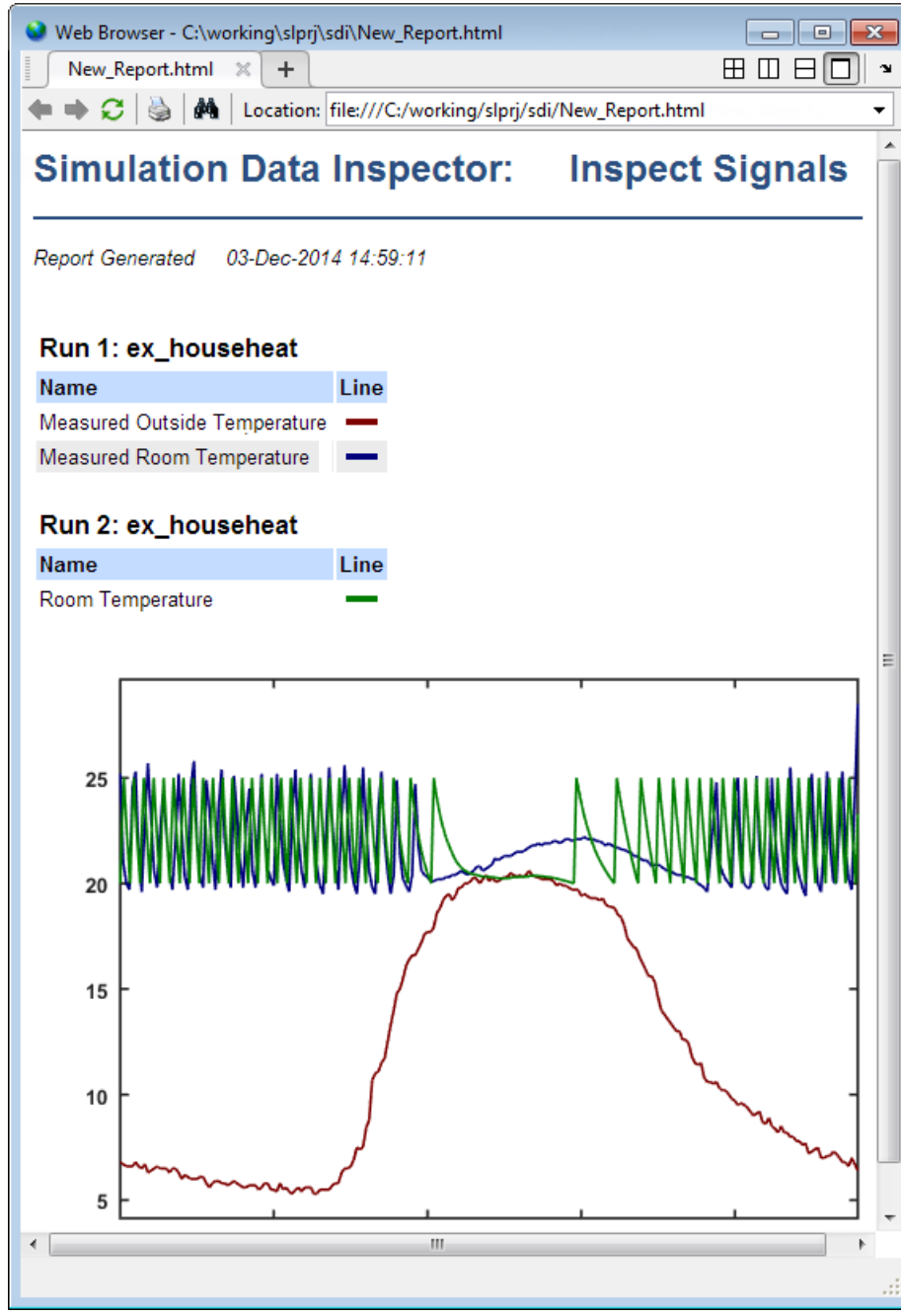
1. Simulate the model.
2. Open the Simulation Data Inspector.
3. Expand the list of logged signals by selecting the arrow to the left of the run. For **Run1**, select the `Measured Outside Temperature` and `Measured Room Temperature` check boxes. For **Run2**, select the `Room Temperature` check box.
4. Review the signals. The minimum and maximum values for the simulated room temperature now match the measured room temperature values.



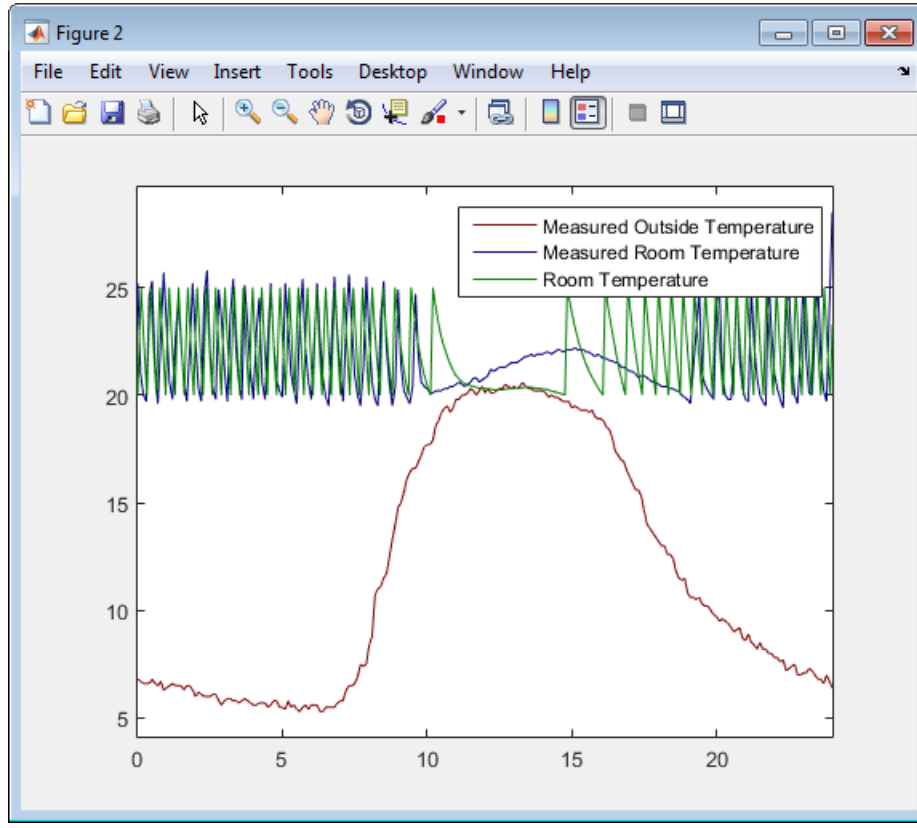
Report and Plot Simulation Results

Create reports and plots from the Simulation Data Inspector.

1. In the Simulation Data Inspector toolbar, click **Create Report**.
2. In the Create Report dialog box, select **Inspect Signals**. Click **Create Report**. The report opens in a web browser.
3. View the report.


Web Browser - C:\working\slprj\sdi\New_Report.html
New_Report.html
Location: file:///C:/working/slprj/sdi/New_Report.html
Simulation Data Inspector: Inspect Signals
Report Generated 03-Dec-2014 14:59:11
Run 1: ex_househeat
Name
Line
Measured Outside Temperature
Measured Room Temperature
Run 2: ex_househeat
Name
Line
Room Temperature
25
20
15
10
5

4 In the Simulation Data Inspector, click **Send to Figure**. A plot of the simulated signals opens in a MATLAB figure window.



More About

- "Model-Based Design" on page 1-3
- "Basic Modeling Workflow" on page 1-6
- "Basic Simulation Workflow" on page 1-13