

An Introduction to Simulink® in Simulation of Instrumentation and Process Control Systems

Overview

Simulink® is a graphical application program accompanied with MATLAB and working in the MATLAB environment. Although the MATLAB package is useful for linear system analysis Simulink is far more useful for control system simulation. Simulink enables the rapid construction and simulation of control block diagrams.

Learning Outcomes

After completion of these tutorials, you should be able to

- Get started with Simulink®
- Explain how a Simulink model works
- Create and run a Simulink model for simulation of a temperature measurement system
- Simulate measuring systems
- Solve differential equations with Simulink®
- Simulate zero-, first- and second-order systems

Contents

- Introduction – Getting Started with Simulink
- **Tutorial 1** –First Simulink Model and Simulation of Temperature Measurement System
- **Tutorial 2** – Simulation of a Resistance Transducer
- **Tutorial 3** – Simulation of Zero-order and First-order Systems
- **Tutorial 4** – Simulation of Second-order Systems

1. Products of MathWorks

<http://www.mathworks.com/>



Figure 1 Overview of MathWorks products

2. Toolboxes for Instrumentation and Process Control

List of products: http://www.mathworks.com/products/product_listing/index.html

MATLAB

Control System Design and Analysis

- Control System Toolbox
- System Identification Toolbox
- Fuzzy Logic Toolbox
- Robust Control Toolbox
- Model Predictive Control Toolbox
- Aerospace Toolbox

Test & Measurement

Data Acquisition Toolbox
Instrument Control Toolbox
Image Acquisition Toolbox
SystemTest
OPC Toolbox
Vehicle Network Toolbox

Simulink (R)

Physical Modeling

- [Simscape](#)
- [SimMechanics](#)
- [SimPowerSystems](#)
- [SimDriveline](#)
- [SimHydraulics](#)
- [SimElectronics](#)

Control System Design and Analysis

- [Simulink Control Design](#)
- [Aerospace Blockset](#)
- [Simulink Design Optimization](#)

Code Generation

- [Real-Time Workshop](#)
- [Real-Time Workshop Embedded Coder](#)
- [Stateflow Coder](#)
- [Simulink HDL Coder](#)

Rapid Prototyping and HIL Simulation

- [xPC Target](#)
- [xPC Target Embedded Option](#)
- [Real-Time Windows Target](#)

Control Engineering Lab

MATLAB (R2007b)

Math and Optimization

- [Symbolic Math Toolbox](#)

Control System Design and Analysis

- [Control System Toolbox](#)
- [System Identification Toolbox](#)

Test & Measurement

- [Data Acquisition Toolbox](#)

Simulink

Code Generation

- [Real-Time Workshop](#)

Rapid Prototyping and HIL Simulation


- [xPC Target](#)

3. Background of Simulink®

3.1 What Is Simulink?

Simulink, a companion program to MATLAB (the current version of MATLAB: R2008b), is an interactive system for simulating linear and nonlinear dynamic systems. It is a graphical mouse-driven program that allows you to model a system by drawing a block diagram on the screen and manipulating it dynamically. It can work with linear, nonlinear, continuous-time, discrete-time, multivariable, and multi-rate systems.

3.2 Simulink Library Browser

The first step is to start up MATLAB on the computer you are using. Type `>>simulink` in the Command Window or click the  button in the Toolbar menu of the MATLAB Window. The Simulink Library Browser window will appear as that shown in **Fig. 1**.

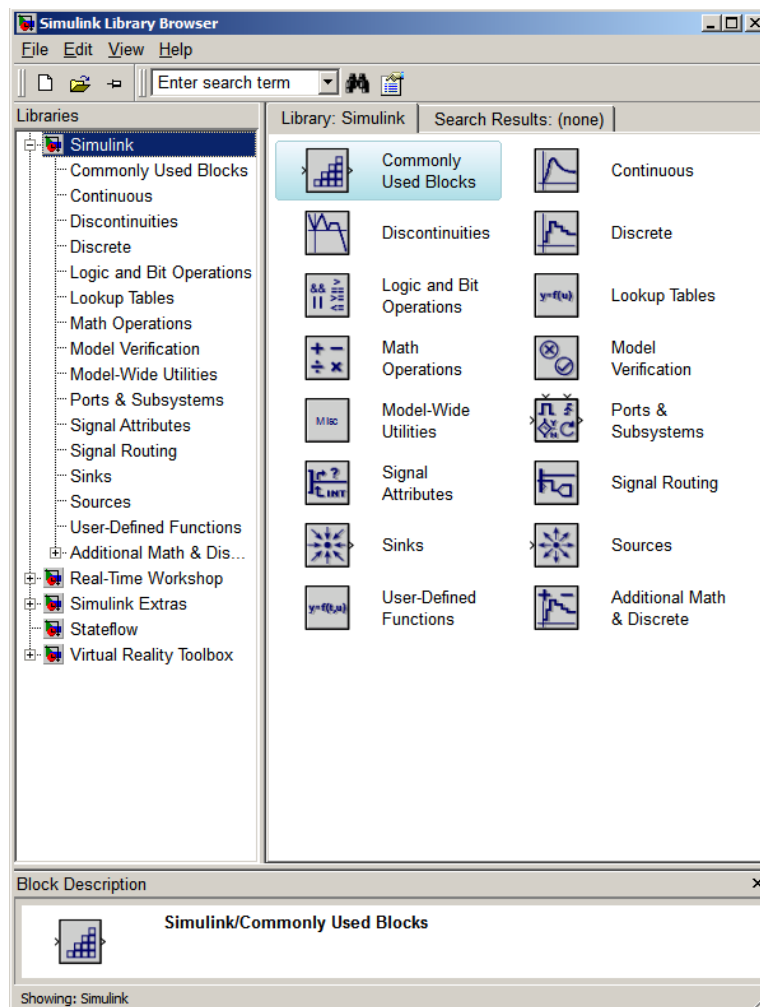


Figure 1 Simulink library browser (MATLAB R2008b (7.7), Simulink 7.2)

Simulink® 7.2 has a number of additional options. There are several groups of Simulink blocks in the Simulink icon such as Commonly Used Blocks, Continuous, Discontinuities, Math Operations, Sinks and Sources, etc. Selecting Commonly Used Blocks will provide a list of blocks shown in **Fig. 2**.

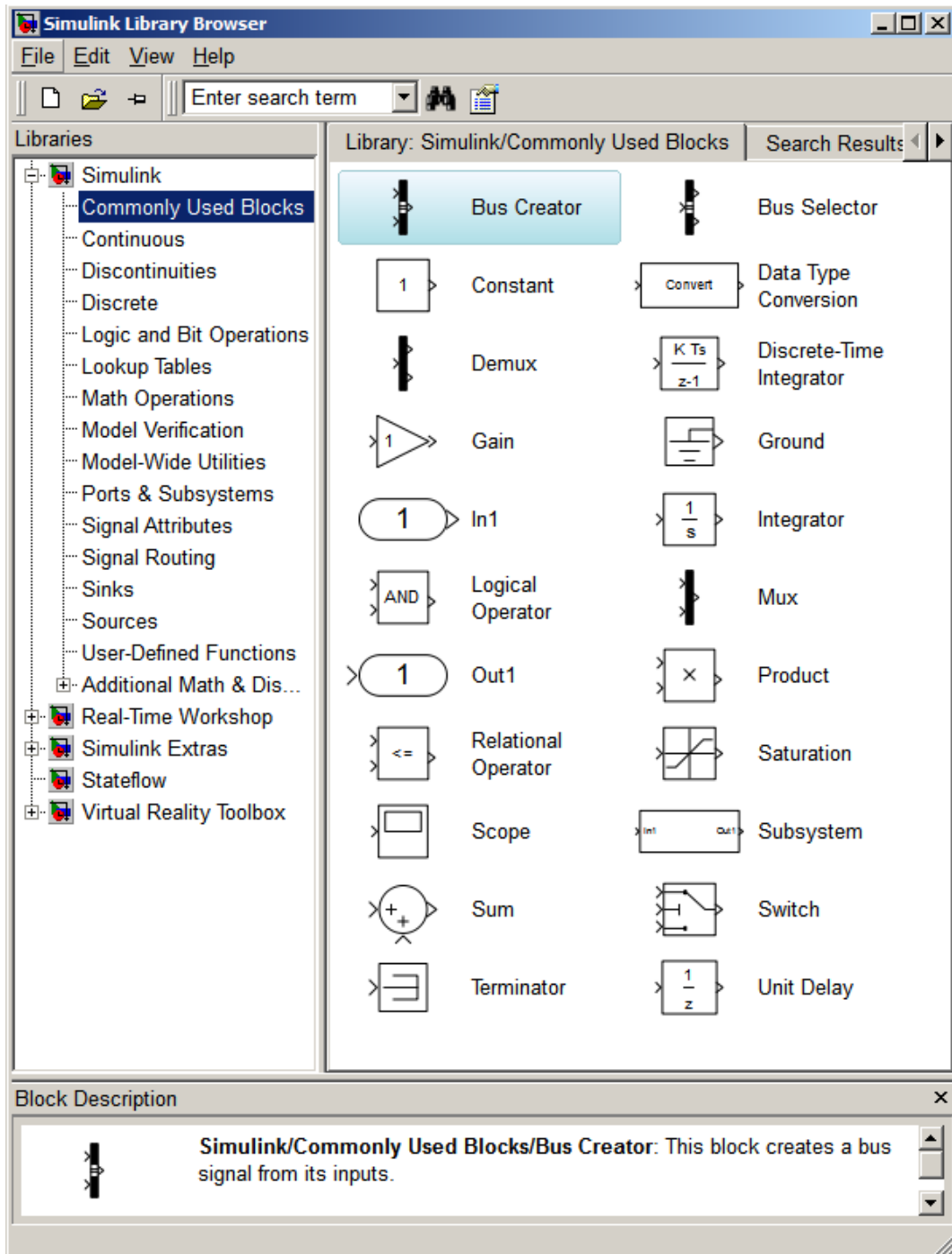


Figure 2 A list of blocks in Commonly Used Blocks group

Selecting Continuous will provide a list of blocks shown in **Fig. 3**. The ones that we often use are Transfer Fcn, State-space and Integrator.

Selecting the Sources icon yields the library shown in **Fig. 4**. The most commonly used sources are Clock (which is used to generate a time vector), Step (which generates a step input), and Constant (that generate a constant function).

The Sinks icon as shown in **Fig. 5** provides a set of Sinks blocks that are used to display simulated results. The most often used blocks may be To Workspace (to which a variable passed is written to a vector in the MATLAB Workspace), Scope (to represent data graphically).

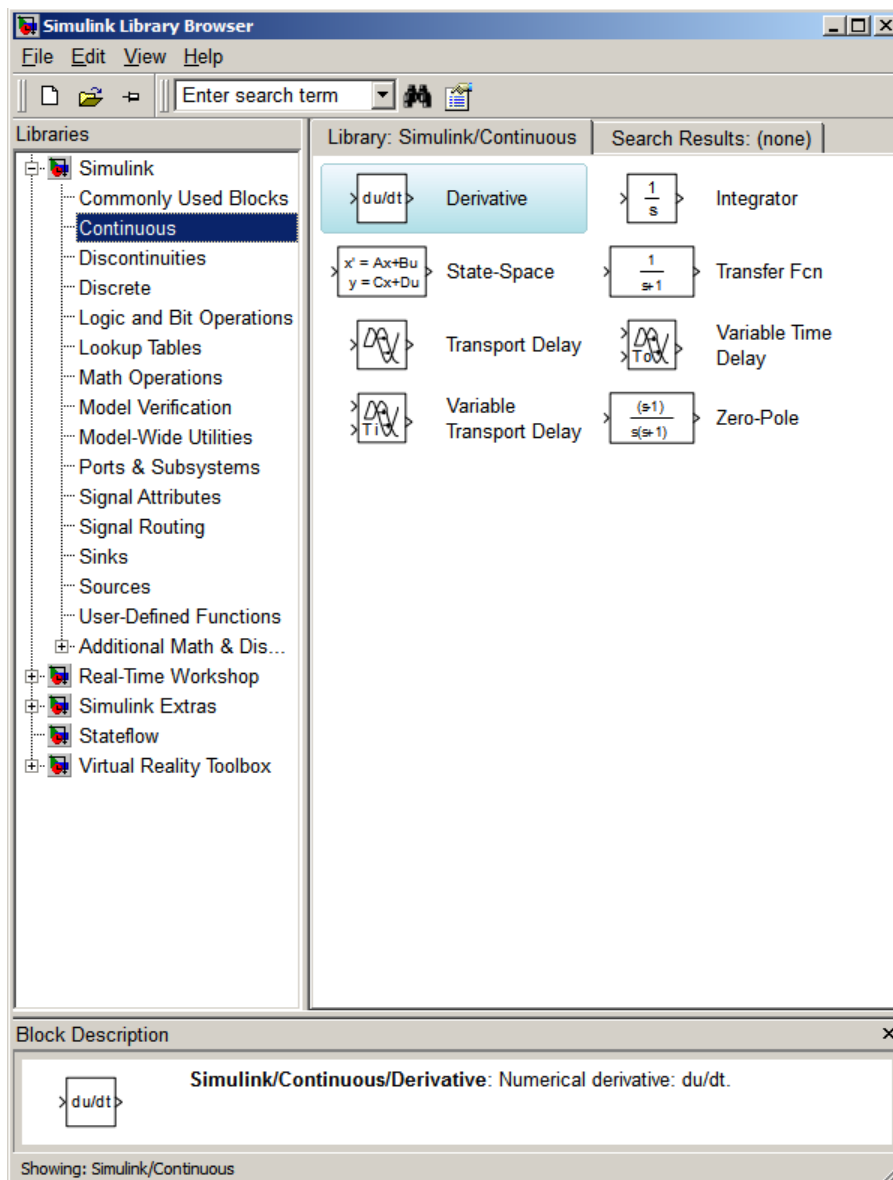


Figure 3 A list of blocks in Continuous group

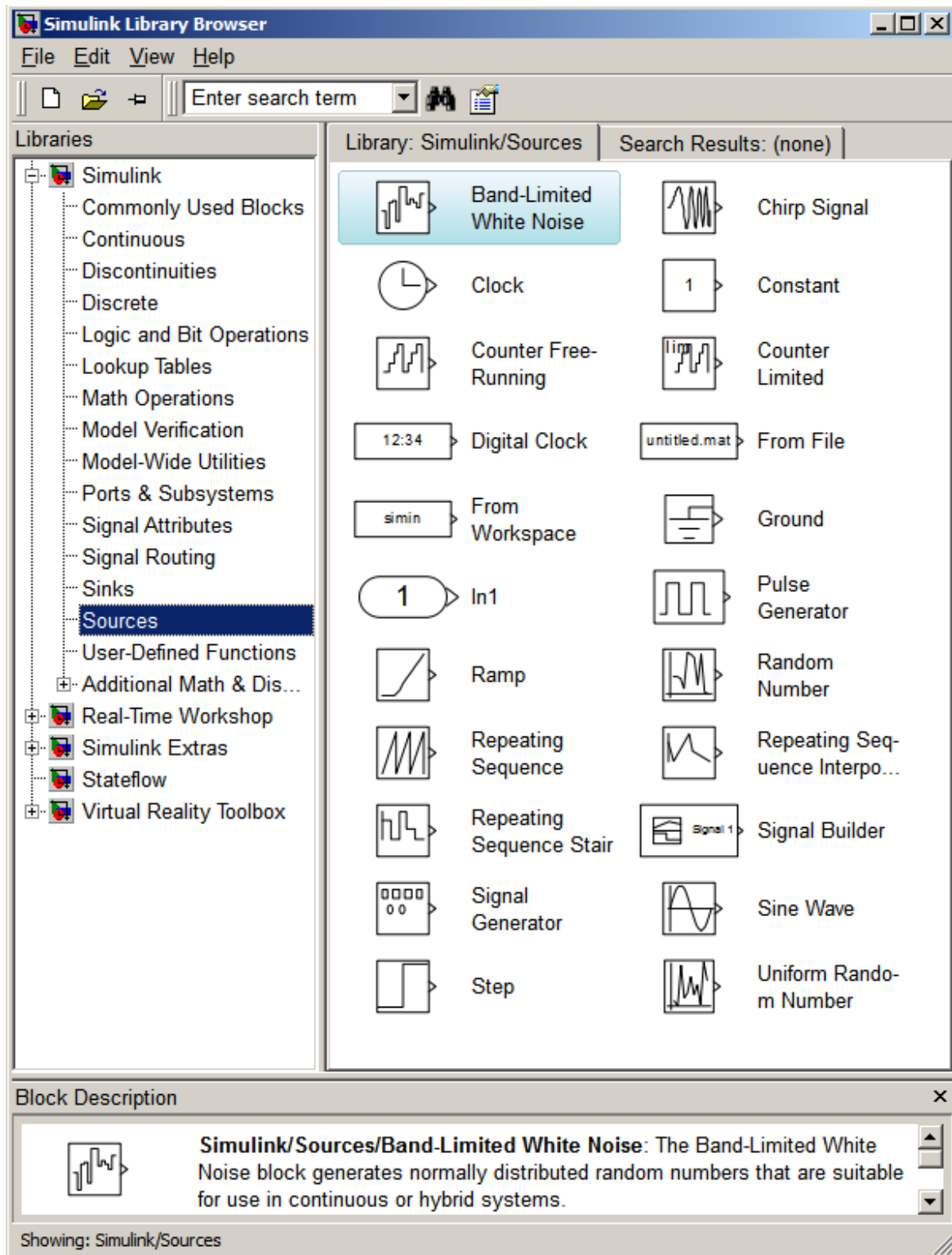


Figure 4 A list of blocks in the Sources group

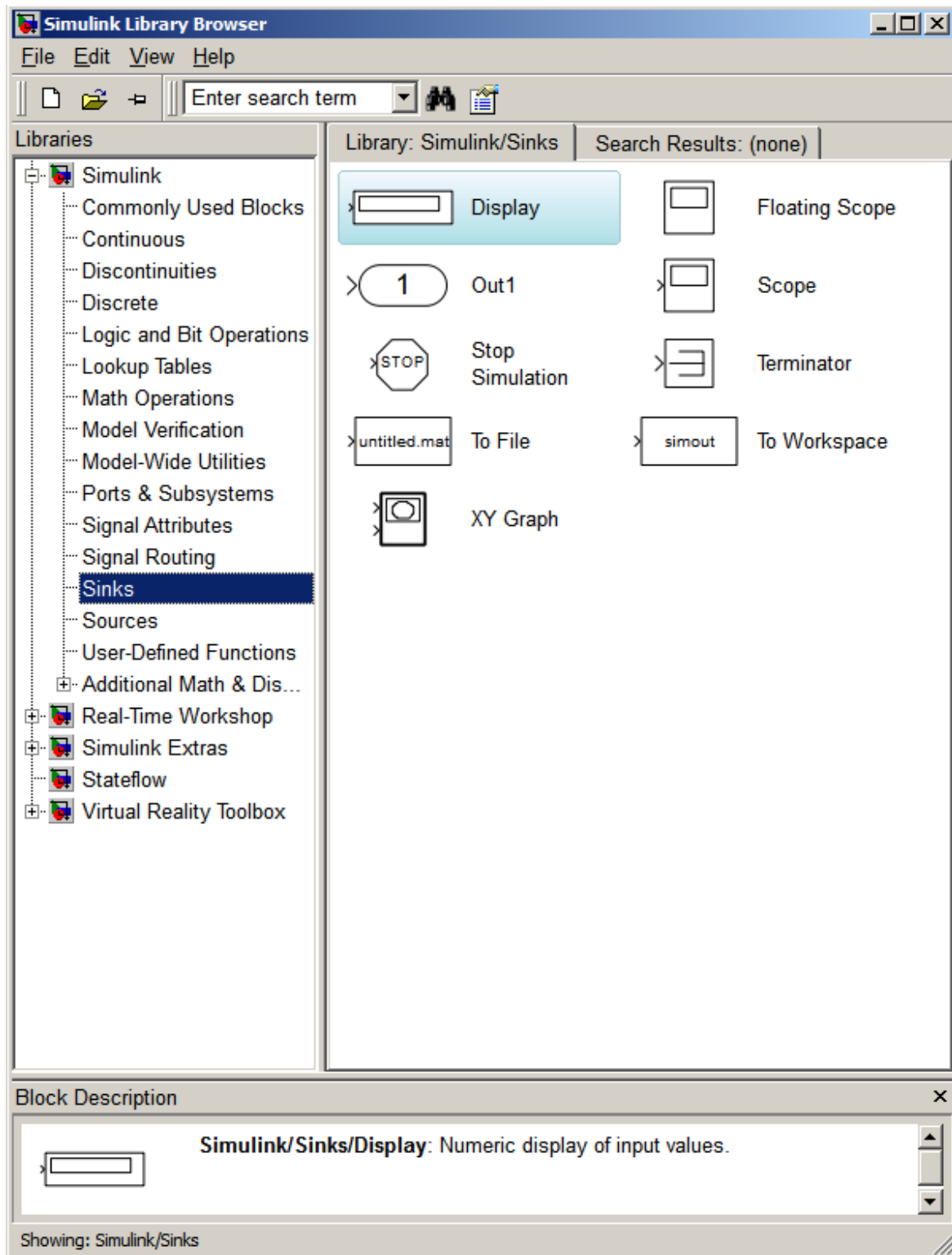


Figure 5 A list of blocks in Sinks group

3.3 Concept of Signal and Logic Flow

In Simulink, data/information from various blocks is sent to another block by lines connecting the relevant blocks. Signals can be *generated* and fed into blocks (dynamic / static). Data can be fed into functions. Data can then be dumped into *sinks*, which could be scopes, displays or could be saved to a file. Data can be connected from one block to another, can be branched, multiplexed etc. In simulation, data is processed and transferred only at *Discrete* times, since all computers are discrete systems. Thus, a SIMULATION time step (otherwise called an INTEGRATION time step) is essential, and the selection of that step is determined by the fastest dynamics in the simulated system.

3.4 Connecting Blocks

A Simulink model is created based on the idea of block diagram and it consists of several blocks. Necessary blocks for a Simulink model can be selected from the Simulink Library Browser and they are connected together. To connect blocks, *left-click* and drag the mouse from the output of one block to the input of another block. **Figure 6** shows the steps involved. The Simulink model in **Fig. 6** consists of the following blocks: Clock, To Workspace1, Step Input, Transfer Fcn and To Workspace2.

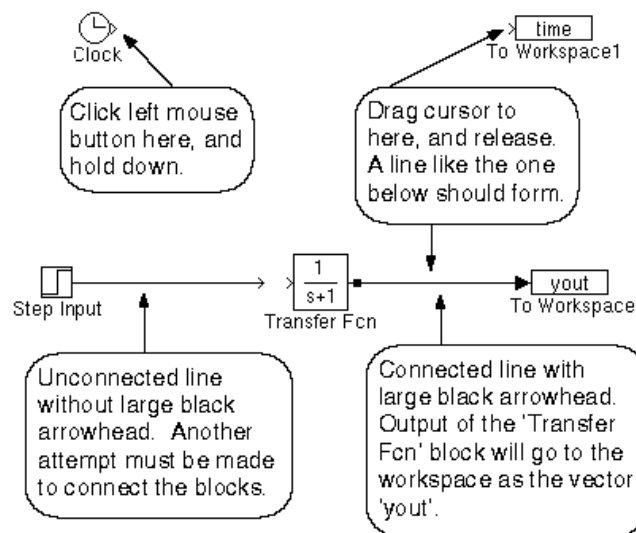


Figure 6 Connecting blocks

4. Procedure for Simulink Simulation

With Simulink we can develop block diagram algorithms to solve ODEs or do scientific computation. Although computer simulations can be used to model a large variety of systems, it can be seen that all computer simulations must embody the following components:

1. Structure of the mathematical model: This is the complete set of differential equations that describe the system behaviour (dynamics) and reflect the fundamental physical laws governing the behaviour of the system.

2. Model parameter values: Model parameters refer to numerical constants that usually do not change over the course of the simulation. Typical parameters for mechanical systems are mass, damping coefficient, and spring stiffness.

3. Initial conditions: Initial conditions are important for determination of the solution of ODEs. Therefore in the simulation program initial conditions must be set.

4. Inputs: Typically a system responds to one or more inputs. The simulation must embody the inputs as well. In simulation tools, there are some blocks that generate test input signals such as a step function, a ramp function or a sine-wave function.

5. Outputs: Although a simulation does not require that the user explicitly define outputs, it is assumed that the goal of a computer simulation of a dynamic system is the time history of specific physical variables in the system under study. The time history of output variables can be stored to the computer hard drive for later analysis or displayed as graphs on the screen. The simulation tools often have blocks for displaying simulated outputs.

6. Simulation solution control parameters: Simulation solution control parameters define the values and choices made by the designer/engineer of the simulation tool who dictates how the numerical methods behind the simulation operate. These include values that determine the step size, output interval, error tolerance, and choice of numerical integration algorithm.

Example: in order to calculate $y = au + b$ we develop a block diagram as follows:

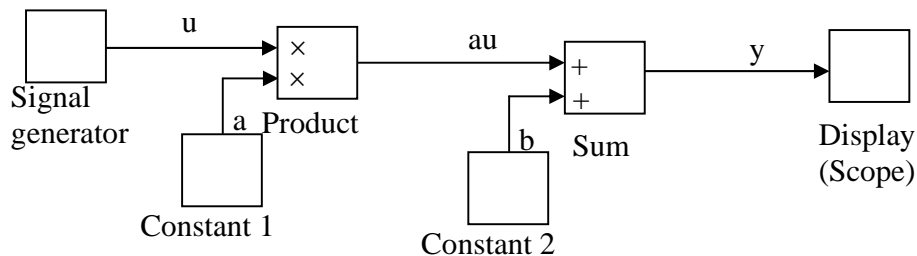
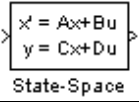
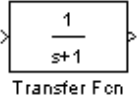

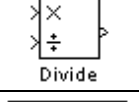
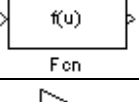
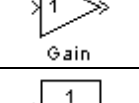
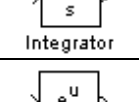
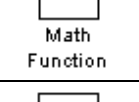
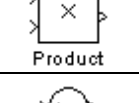
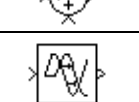

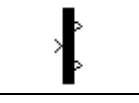




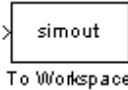






Figure 7 An example of block diagram algorithm

5. Commonly Used Simulink Blocks

Table 1 summarises a list of commonly used Simulink blocks which we will the most often use in our course.

Table 1 Summary of Commonly Used Simulink Blocks

Block icon	Name	Use
Continuous		
	State-Space	Implement a linear state-space system
	Transfer Fcn	Implement a linear transfer function
Math Operations		
	Derivative	Merge scalar, vector or matrix signals
	Divide	Multiply or divide inputs
	Function	Apply a specified expression to the input
	Gain	Multiplies the input by a constant value (gain)
	Integrator	Integrate the input signal
	Math Function	Perform a mathematical function
	Product	Multiply inputs
	Sum	Add or subtract inputs
	Transport Delay	Delay the input by a given amount of time
Signal Routing		
	Demux	Split vector signals into scalars or smaller vectors

	Mux	Extract and output the elements of a bus or vector signal
Sinks		
	Scope	Display signals generated during a simulation
	To Workspace	Write data to the workspace
	XY Graph	Display an X-Y plot of signals using a MATLAB figure window
Sources		
	Clock	Generate a time vector
	Constant	Generate a constant
	Ramp	Output a ramp signal
	Sine Wave	Generate a sine wave signal
	Step	General a step signal

6. Summary

Simulink is a very powerful block diagram simulation language. Simple simulations can be set up rapidly. The aim of this tutorial was to provide enough of an introduction to get you started on the development of simulations for dynamic systems. With experience, the development of these simulations will become second nature. It is recommended that you perform the simulations shown in these tutorials as well as the follow-up exercises, to rapidly acquire these simulation skills.