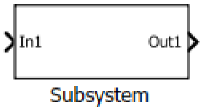
Contents

Subsystems

Subsystems

Library: Simulink/Ports & Subsystems/Subsystem



A subsystem is a collection of blocks, represented by a single block called the **Subsystem** block.

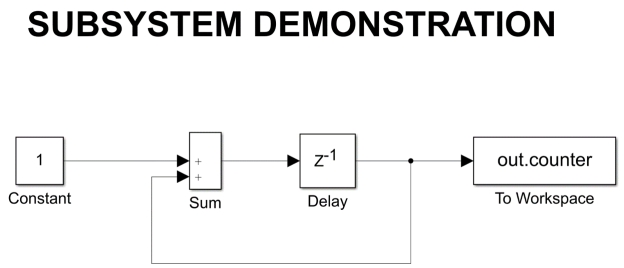
A **subsystem** block can be used to achieve one of the following:

1. Reduce the number of blocks displayed on the main workspace of the Simulink® model.
2. Establish a model hierarchy by layering blocks that perform certain tasks in the model.
3. Conveniently replicate a set of blocks (represented by the subsystem), that operate independently in the model.
4. Execute a subsystem in the model only when conditions are met. See also: Conditionally Executed Subsystem.

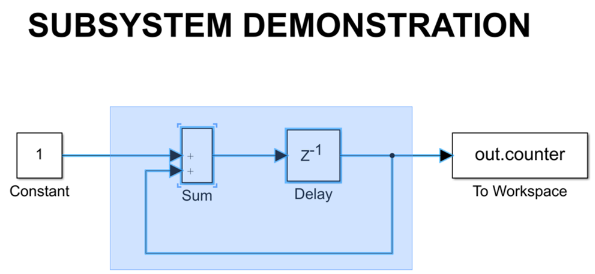
Creating a subsystem

A subsystem can be created in a Simulink® model by either creating one or copying an existing subsystem from another model. In this example, a simple counter that increments the previous state by 1 is illustrated. The model consists of a Constant block (input), a Sum block, and a Unit Delay block. Optionally, a To Workspace or Scope block can be used to log the signals.

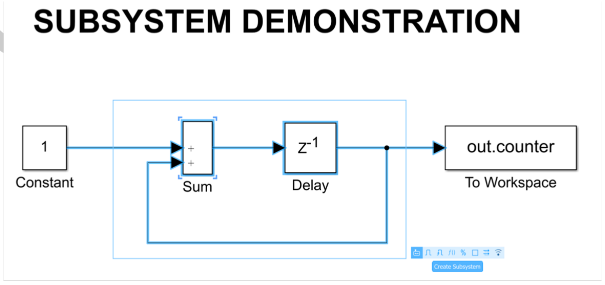
Grouping existing blocks into a subsystem

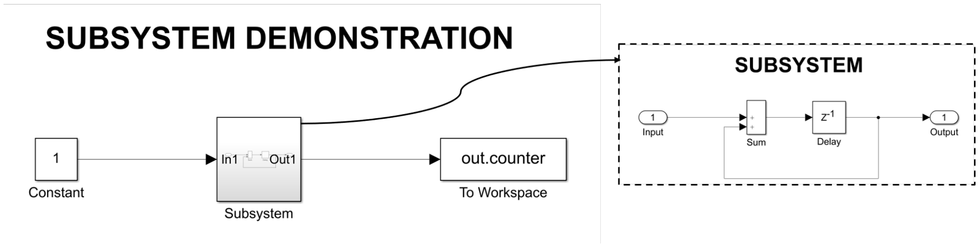


1. Include all the blocks in the model that are expected to be in the subsystem.



2. Click and hold the left mouse button to start a bounding box and drag the mouse to change its shape.
3. Encompass all the blocks in the model that are a part of the subsystem. Release the mouse button when all the blocks are selected.





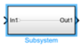
4. Hover the mouse on **...** at the bottom right of the bounding box. Click option **Create Subsystem** to form a subsystem from current selection. Double clicking the subsystem block shows the encompassed blocks with input and output ports.

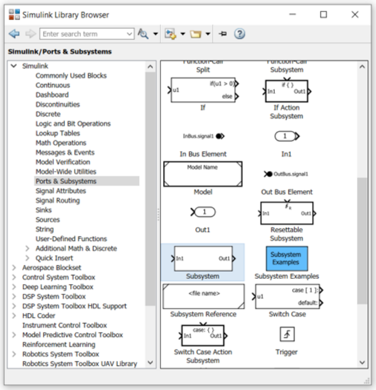
   Alternatively (try one of these)**:**

   1. Right click any one block in the selection. In the drop down menu, select **Create Subsystem**.
   2. Press **Ctrl + G** to create a subsystem from the selection.
   3. Navigate to **MODELING**. Go to **COMPONENT** and select **Create Subsystem** option.

Using a Subsystem block

This method can be used to first create a subsystem block and then add operation blocks inside the subsystem.





1. Navigate to **SIMULATION** in Simulink®.

2. Click on **Library Browser**.

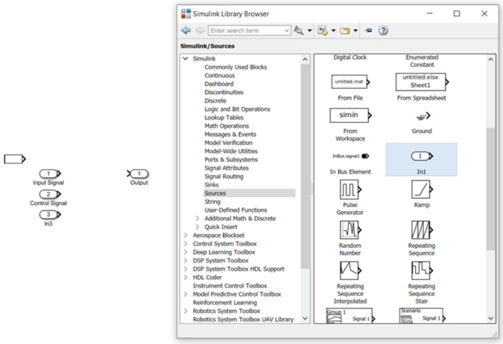
3. On the left hand column, select **Simulink**. Under **Simulink**, scroll to **Ports & Subsystems**.

4. On the right hand column, select the **Subsystem** block. Drag the block into the Simulink model workspace.

5. Double click the subsystem block to enter the block. Add remaining blocks to reconstruct the counter example.

Configuring a subsystem

Adding inputs and outputs



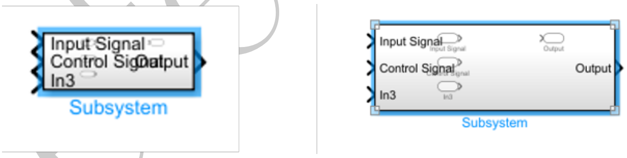
A subsystem is defined by its input and output ports. When a subsystem is created, Simulink® automatically creates inputs and outputs represented by **In** and **Out** respectively.

More input ports can be created by including **In1** block from **SIMULATION** > **Library Browser** > **Simulink** > **Sources** > **In1** inside the subsystem. Similarly, output ports can be created by including **Out1** block from **SIMULATION** > **Library Browser** > **Simulink** > **Sinks** > **Out1**.
Alternatively:

1. Hover mouse on the left hand side of the subsystem block. When the mouse pointer changes to +, click and drag the mouse to create a new input port.

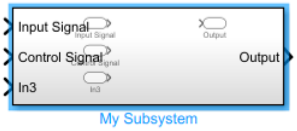
2. To create an output port, repeat step 1 on the right side of the subsystem block.

Resizing the subsystem block



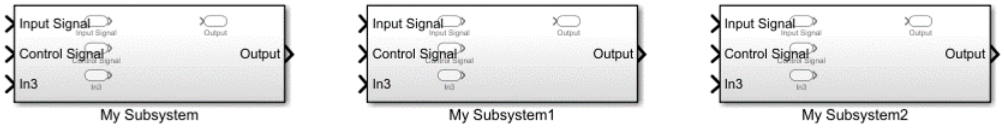
With increasing the number of input and output ports, the default block size may disfigure the port labels. This can be fixed by resizing the block. To resize the block, left click + hold any corner represented by □ and drag the mouse.

Renaming the subsystem block



To simplify multiple inputs and outputs, rename the ports with desired characters. This can be done by double-clicking the existing name of the port, e.g. "In1" and replacing it by a specific text. Upon renaming the port inside the subsystem, the corresponding port name is automatically changed on the subsystem block.

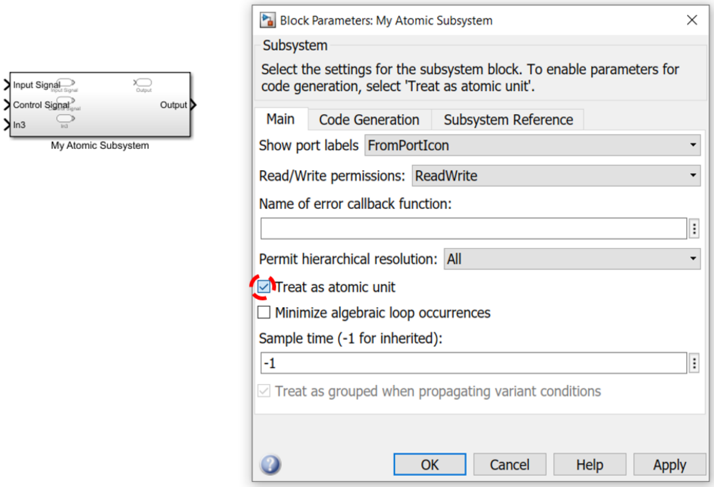
Hierarchy in subsystems



Every subsystem creates a layer of hierarchy in the model. In case, multiple copies of subsystem exist in the the same model, rename each subsystem block for convenience. Blocks inside the subsystems are not affected by multiple copies of the subsystem in the model.

Multiple instances of the same subsystem can be executed differently in a model by including conditions. See also: Conditionally Executed Subsystem.

Conditional Execution of subsystems



Block Parameters: My Atomic Subsystem

Subsystem

Select the settings for the subsystem block. To enable parameters for code generation, select 'Treat as atomic unit'.

Show port labels: FromPortIcon

Read/Write permissions: ReadWrite

Name of error callback function:

Permit hierarchical resolution: All

☑ Treat as atomic unit

□ Minimize algebraic loop occurrences

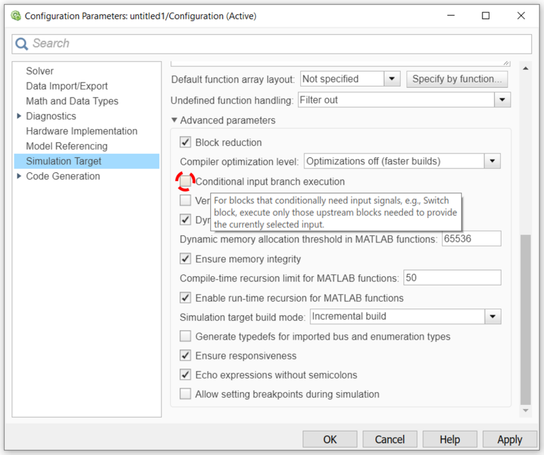
Sample time (-1 for inherited): -1

☑ Treat as grouped when propagating variant conditions

Figure 1: Treat regular subsystem as atomic subsystem

By default, all unconditional subsystems are virtual. All blocks in a virtual subsystem are executed at the same order as blocks outside the subsystem. To execute blocks of a subsystem before model execution, use Atomic Subsystems (SIMULATION > Library Browser > Simulink > Ports & Subsystems > Atomic Subsystem) or a conditionally executed subsystems. Alternatively, block parameters of a regular subsystem can be changed to Treat as atomic unit.

To reduce the time taken to execute a model, Simulink® by default avoids unnecessary execution of blocks connected to switch, multi-port switch, and conditionally executed subsystems. This behavior is called **conditional execution (CE)** behavior. In a conditionally executed subsystem, Simulink® only calculates the output of the subsystem when the subsystem executes otherwise the output is held (or reset) until the next execution. Upon disabling this setting, Simulink® calculates subsystem output irrespective of whether the subsystem executes however, the output of the subsystem remains the same.



Default function array layout: Not specified ▾ Specify by function...

Undefined function handling: Filter out ▾

Advanced parameters

☑ Block reduction

Compiler optimization level: Optimizations off (faster builds) ▾

☐ Conditional input branch execution

☐ Ver

☑ Dyr

For blocks that conditionally need input signals, e.g., Switch block, execute only those upstream blocks needed to provide the currently selected input.

Dynamic memory allocation threshold in MATLAB functions: 65536

☑ Ensure memory integrity

Compile-time recursion limit for MATLAB functions: 50

☑ Enable run-time recursion for MATLAB functions

Simulation target build mode: Incremental build ▾

☐ Generate typedefs for imported bus and enumeration types

☑ Ensure responsiveness

☑ Echo expressions without semicolons

☐ Allow setting breakpoints during simulation

Figure 2: Disable conditional execution of subsystems

To disable the setting, go to MODELING > Model Settings > Simulation Target > Advanced Parameters and deselect the Conditional input branch execution option.

Conditionally Executed Subsystem

A **conditionally executed (CE) subsystem** executes one or more times every time step in a model when an input condition signal to the subsystem satisfies the user set condition. The input signal that controls the subsystem execution is called a **control signal**. A control signal can either be continuous or discrete.

A CE subsystem can be categorized as one of the following:

1. Enabled Subsystem.
2. Triggered Subsystem.
3. Enabled and Triggered subsystem.
4. Control Flow Subsystem.

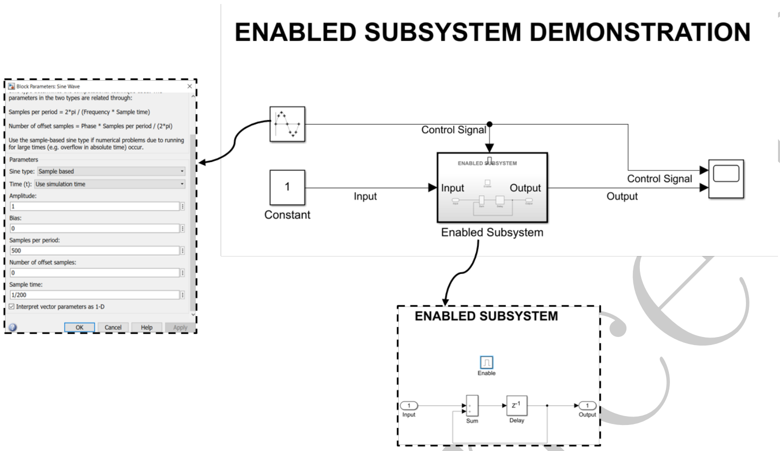
Conditionally executed subsystems can be very useful when building complex models where the execution of certain components depend upon others.

Enabled Subsystem

An **enabled subsystem** begins execution once every time step when the control signal crosses zero and becomes positive and continues execution as long as the signal is positive.

Creating an Enabled Subsystem

In this example, a simple counter (subsystem) is executed as long as the sine wave (control signal) is positive. When the control signal is negative, the output of the subsystem is held (by default) until enabled again.



Block Parameters: Sine Wave

parameters in the two types are related through:

Samples per period = 2*pi / (Frequency * Sample time)

Number of offset samples = Phase * Samples per period / (2*pi)

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: Sample based

Time (t): Use simulation time

Amplitude: 1

Bias: 0

Samples per period: 500

Number of offset samples: 0

Sample time: 1/200

☑ Interpret vector parameters as 1-D

Figure 3: Enabled Subsystem Demonstration

1. Include a Sine Wave block from (SIMULATION > Library Browser > Simulink > Sources > Sine Wave).

2. Include a Enabled Subsystem block from (SIMULATION > Library Browser > Simulink > Ports & Subsystems > Enabled Subsystem). Reconstruct the subsystem from the counter example.

3. Connect Constant block to the input port of the Enabled Subsystem.

4. Connect Sine Wave block to the enable signal of the Enabled Subsystem. (Optional) The enable block can be changed to **reset**, where the output of subsystem starts from initial value of the subsystem when enabled and hold the output while disabled.

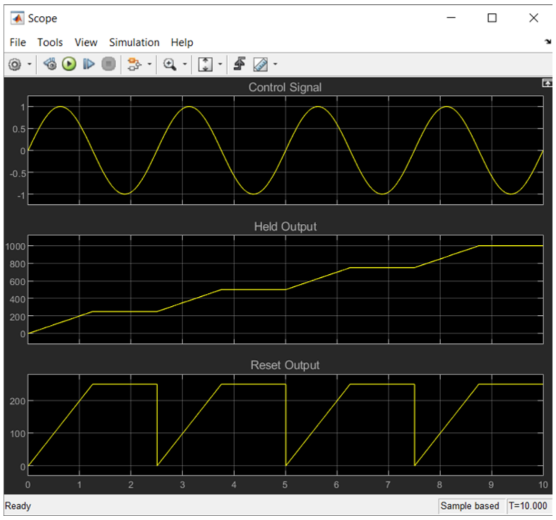


Figure 4: Enabled subsystem output

Triggered Subsystem

A **triggered subsystem** execution is based on trigger events initiated by the control signal. In contrast to an enabled subsystem, a triggered subsystem executes only once when a trigger event occurs. Between trigger events, the previous output of the subsystem is held.

A **trigger event** can be one of the following:

1. rising: A trigger event occurs when control signal crosses negative value or zero to a positive value.
2. falling: A trigger event occurs when control signal crosses positive value or zero to a negative value.
3. either: A trigger event occurs when control signal crosses zero to become either positive or negative valued.

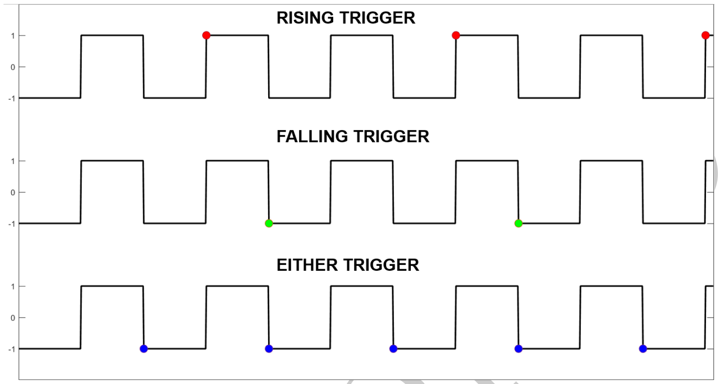
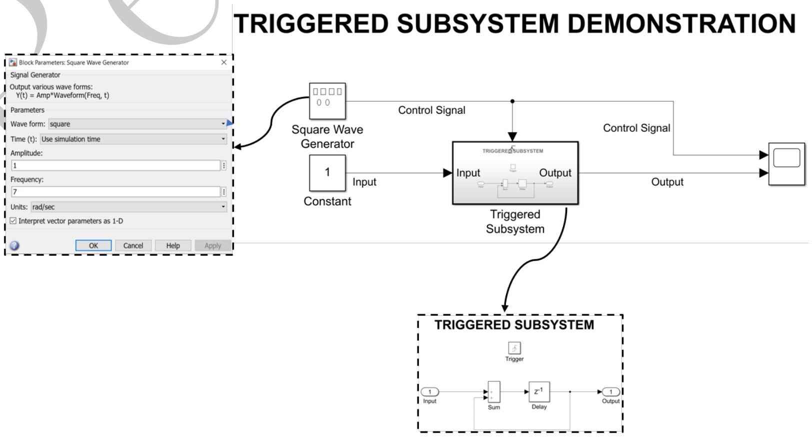


Figure 5: Different trigger events generated by square wave

This figure illustrates different trigger events generated by a square wave. Since the rising and falling edge of a square wave lasts for only one time-step, the first edge is not detected as a trigger event. Hence, for rising and falling trigger, every alternate edge is detected. Similarly, the either trigger is activated at a falling edge with a previous rising edge undetected.

Creating a Triggered Subsystem



Block Parameters: Square Wave Generator

Signal Generator

Output various wave forms:
Y(t) = Amp*Waveform(Freq, t)

Parameters

Wave form: square

Time (t): Use simulation time

Amplitude: 1

Frequency: 7

Units: rad/sec

☑ Interpret vector parameters as 1-D

Figure 6: Triggered Subsystem Demonstration

In this example, a simple counter (subsystem) is executed at trigger events initiated by a square wave (control signal). Outputs of the three types of trigger events can be visualized here.

1. Include a Wave Generator block from (SIMULATION > Library Browser > Simulink > Sources > Wave Generator). Set Waveform Type to square.

2. Include a Triggered Subsystem block from (SIMULATION > Library Browser > Simulink > Ports & Subsystems > Triggered Subsystem). The Trigger Type option is by default set to rising. Reconstruct the subsystem from the counter example.

3. Connect the Wave Generator block to the control input port of the Triggered Subsystem.

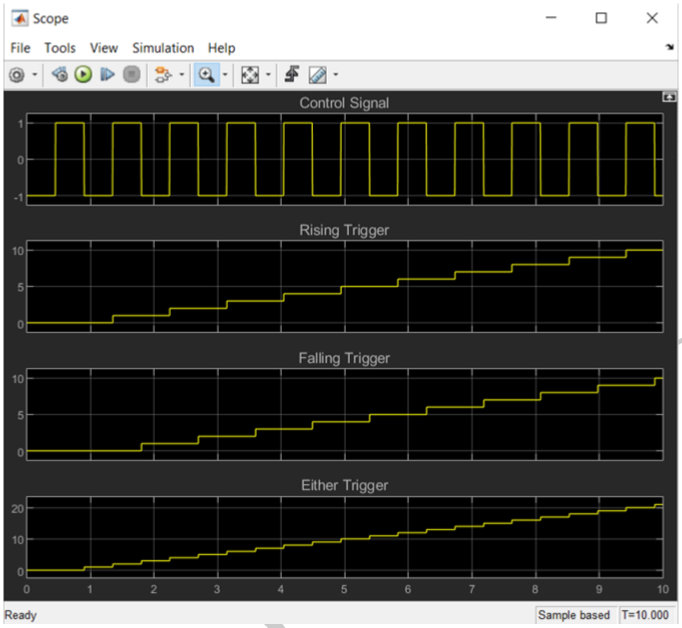


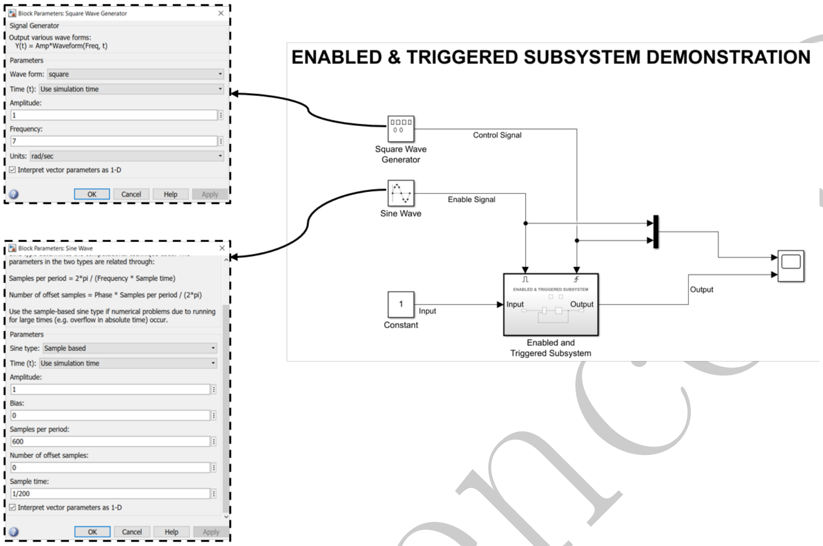
Figure 7: Triggered Subsystem Output

The sample time of blocks inside the **triggered subsystem** can only be continuous (0) or inherited (-1) to ensure that the blocks execute only when the subsystem executes.

Enabled and Triggered Subsystem

An **enabled and triggered subsystem** is executed by two control signals, enable signal and trigger signal. The block executes once every time-step if a trigger event occurs and enable signal is a positive value. The execution is dependent on the satisfaction of both control inputs.

Creating an Enabled and Triggered Subsystem



Block Parameters: Square Wave Generator

Signal Generator

Output various wave forms:
  Y(t) = Amp*Waveform(Freq, t)

Parameters

Wave form: square

Time (t): Use simulation time

Amplitude: 1

Frequency: 7

Units: rad/sec

☑ Interpret vector parameters as 1-D

Block Parameters: Sine Wave

parameters in the two types are related through:

Samples per period = 2*pi / (Frequency * Sample time)

Number of offset samples = Phase * Samples per period / (2*pi)

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: Sample based

Time (t): Use simulation time

Amplitude: 1

Bias: 0

Samples per period: 600

Number of offset samples: 0

Sample time: 1/200

☑ Interpret vector parameters as 1-D

Figure 8: Enabled and Triggered Subsystem Demonstration

In this example, a simple counter (subsystem) is executed when an enable signal is positive and trigger event occurs. The trigger event is set to rising by default. This type of subsystem is used in applications that require multiple conditions to control its execution.

1. Include an Enabled and Triggered Subsystem block from (SIMULATION > Library Browser > Simulink > Ports & Subsystems > Enabled and Triggered Subsystem). Reconstruct the subsystem from the counter example.

2. Include a Wave Generator block from (SIMULATION > Library Browser > Simulink > Sources > Wave Generator). Set Waveform Type to square. Connect the block to trigger port of the subsystem.

3. Include a Sine Wave block from (SIMULATION > Library Browser > Simulink > Sources > Sine Wave). Connect the block to enable port of the subsystem.

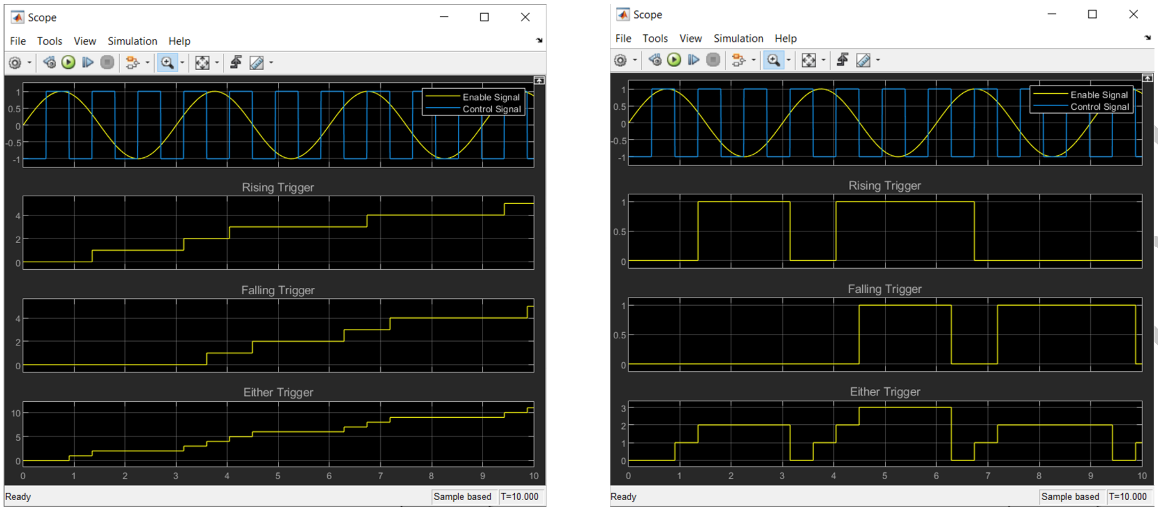


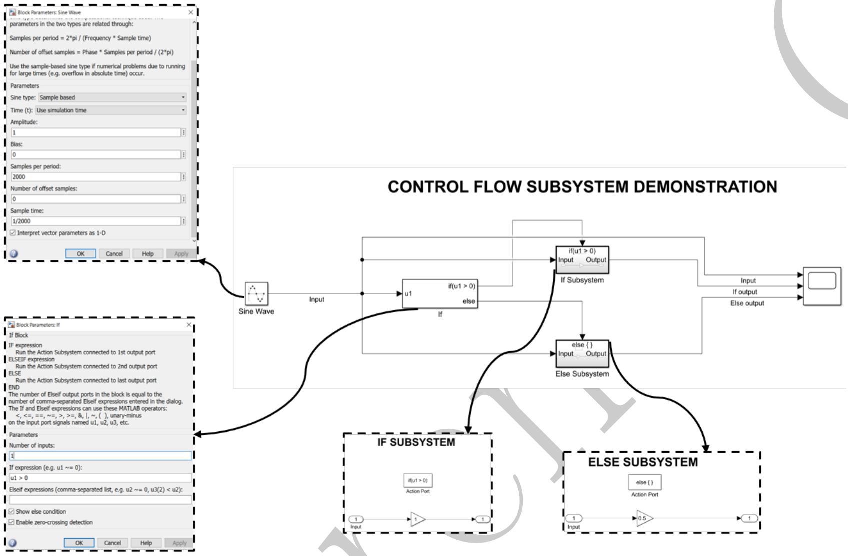
Figure 9: Enabled and Triggered Subsystem output with enable held (left) and enable reset (right)

Control Flow Subsystem

A **control flow subsystem** executes one or more times every time-step depending upon the control flow logic when enabled. It is equivalent to control flow statements of a programming language, e.g. if-then, while, do, and for statements.

Creating a Control Flow Subsystem

This example illustrates the Simulink® equivalent of the *if-else* statement in programming. The control signal and the input signal u is a sine wave. When the if statement, i.e. $u > 0$ is satisfied, the output is the same as the input signal else, the output is 0.5 times the input signal.



Block Parameters: Sine Wave

parameters in the two types are related through:

Samples per period = 2*pi / (Frequency * Sample time)

Number of offset samples = Phase * Samples per period / (2*pi)

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: Sample based

Time (t): Use simulation time

Amplitude: 1

Bias: 0

Samples per period: 2000

Number of offset samples: 0

Sample time: 1/2000

☑ Interpret vector parameters as 1-D

Block Parameters: If

If Block

IF expression
    Run the Action Subsystem connected to 1st output port
ELSEIF expression
    Run the Action Subsystem connected to 2nd output port
ELSE
    Run the Action Subsystem connected to last output port
END
The number of Elseif output ports in the block is equal to the number of comma-separated Elseif expressions entered in the dialog.
The If and Elseif expressions can use these MATLAB operators:
    <, <=, ==, ~=, >, >=, &, |, ~, ( ), unary-minus
on the input port signals named u1, u2, u3, etc.

Parameters

Number of inputs: 1

If expression (e.g. u1 ~= 0): u1 > 0

Elseif expressions (comma-separated list, e.g. u2 ~= 0, u3(2) < u2):

☑ Show else condition

☑ Enable zero-crossing detection

Figure 10: If-else Subsystem Demonstration

1. Include a Sine Wave block from (SIMULATION > Library Browser > Simulink > Sources > Sine Wave).

2. Include an If block from (SIMULATION > Library Browser > Simulink > Ports & Subsystems > If). Select Number of inputs: as 1 and If expression: as $u > 0$

3. Include two If Action Subsystem block from (SIMULATION > Library Browser > Simulink > Ports & Subsystems > If Action Subsystem).

4. Connect action port of the action subsystems to if and else port of the If block respectively.

5. Connect the input port of both action subsystem blocks to the Sine Wave block.

6. Inside the If Action Subsystem, include gain with value 1 between input port and the output port.

7. Inside the If Action Subsystem, include gain with value 0.5 between input port and the output port.

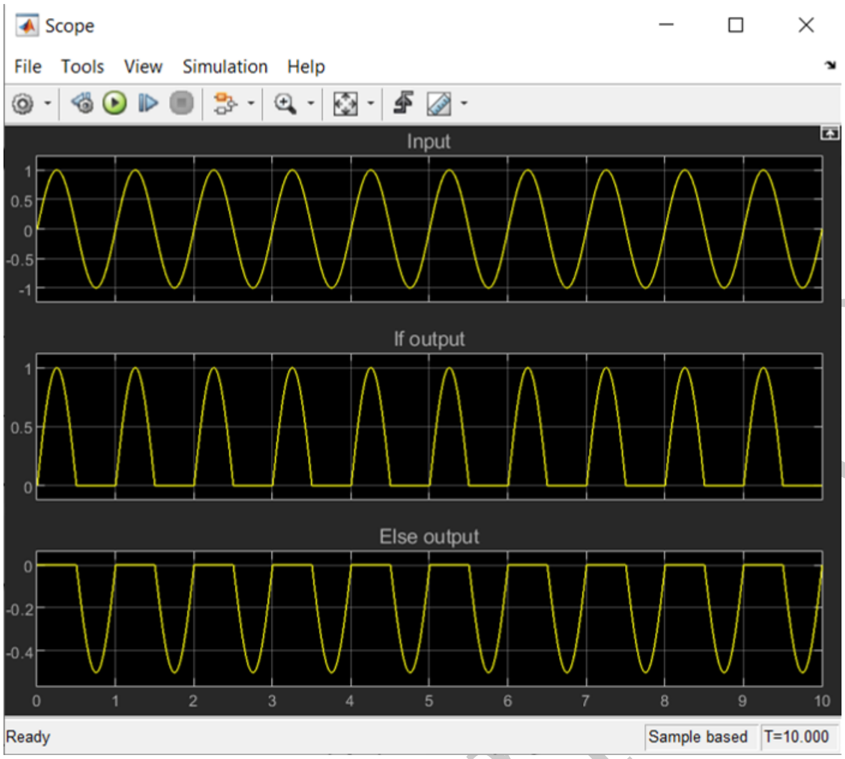


Figure 11: If-else Subsystem output