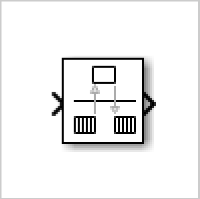
Prerelease Documentation

Rate Transition

Handle transfer of data between blocks operating at different rates



Libraries:
Simulink / Signal Attributes
HDL Coder / Signal Attributes

Description

The Rate Transition block transfers data from the output of a block operating at one rate to the input of a block operating at a different rate. Use the block parameters to trade data integrity and deterministic transfer for faster response or lower memory requirements. To learn about data integrity and deterministic data transfer, see Data Transfer Considerations (Simulink Coder).

Transition Handling Options

Transition Handling Options	Block Parameter Settings	Behavior
• Data integrity • Deterministic data transfer • Maximum latency	Select **Ensure data integrity during data transfer** and **Ensure deterministic data transfer**.	• Generated code transfers data such that the data bytes for the signal (including elements of a wide signal) are from the same time step. • Relative sample time (delay) from which the data is transferred from one rate to another is always the same. • Only ANSI$^{®}$-C code is used. • Target-specific critical-section protection is not needed.
• Data integrity • Nondeterministic data transfer • Minimum latency • Higher memory requirements	Select **Ensure data integrity during data transfer** and clear **Ensure deterministic data transfer**.	• Generated code transfers data such that data bytes for the signal (including elements of a wide signal) are from the same time step. • From one transfer of data to the next, the relative sample time (delay) for which the data is transferred can vary. Code that reads and writes the data runs more often than the case when you select **Ensure data integrity during data transfer** and **Ensure deterministic data transfer**. In the worst case scenario, the delay is

Transition Handling Options	Block Parameter Settings	Behavior
		equivalent to when you select **Ensure data integrity during data transfer** and **Ensure deterministic data transfer**, but the delay can be less, which can be important for some applications. • This option supports data transfers to and from asynchronous rates. • Only ANSI-C code is used. • Target- specific critical-section protection is not needed.
• Potential loss of data integrity • Nondeterministic data transfer • Minimum latency • Lower memory requirements	Clear **Ensure data integrity during data transfer** and **Ensure deterministic data transfer**.	• Code generator does not produce code for the Rate Transition block. • This option is acceptable for applications where atomic access of scalar data types is guaranteed and when the relative temporal values of the data are not important. • This option does not introduce delay.

Dependencies

The behavior of the Rate Transition block depends on:

- Sample times of the ports to which the block connects (see Effects of Synchronous Sample Times and Effects of Asynchronous Sample Times)
- Priorities of the tasks for the source and destination sample times (see Sample time properties)
- Whether the model specifies a fixed- or variable-step solver (see Compare Solvers)
- Setting of model configuration parameters **Device vendor** and **Device type** (see Effects of Device Configuration)

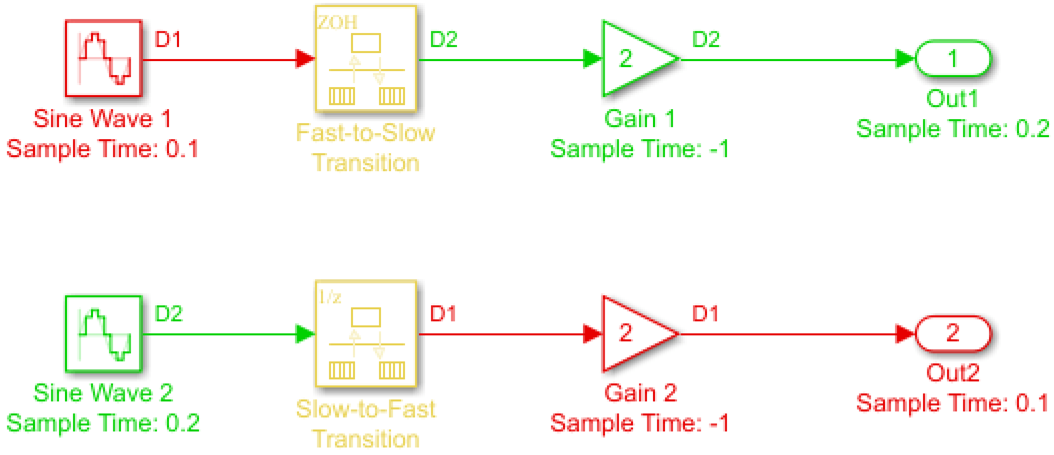
Block Labels

When you update a model diagram, a label appears on the Rate Transition block to indicate simulation behavior.

Label	Block Behavior
`ZOH`	Acts as a zero-order hold
`1/z`	Acts as a unit delay
`Buf`	Copies input to output under semaphore control
`Db_buf`	Copies input to output by using double buffers
`3buf`	Copies input to output by using triple buffers
`Copy`	Unprotected copy of input to output
`NoOp`	Does nothing
`Mixed`	Expands to multiple blocks with different behaviors
`RT`	Indicates data transfer between partitions when using the Schedule Editor. For more information, see Using the Schedule Editor.

Label	Block Behavior
Memory	Indicates memory mode. The block is in memory mode when **Ensure deterministic data transfer (maximum delay)** is cleared.

The block behavior label shows the method that ensures safe transfer of data between tasks operating at different rates. You can use the sample-time colors feature (see View Sample Time Information) to display the relative rates that the block bridges. Consider this example model:



Sample-time colors and the block behavior labels show:

- The Rate Transition block at the top of the diagram acts as a zero-order hold in a fast-to-slow transition.
- The Rate Transition block at the bottom of the diagram acts as a unit delay in a slow-to-fast transition.

For more information, see Data Transfer Representation and Processing (Simulink Coder).

Effects of Synchronous Sample Times

This table summarizes how each label appears when the sample times of the input and output ports (inTs and outTs) are periodic, or synchronous.

Block Settings		Block Label		
Rate Transition	Conditions for Rate Transition Block	With Data Integrity and Determinism	With Only Data Integrity	Without Data Integrity or Determinism
inTs = outTs (Equal)	inTsOffset < outTsOffset	None (error)	Buf	Copy or NoOp (see note that follows the table)
	inTsOffset = outTsOffset	Copy or NoOp (see note that follows the table)	Copy or NoOp (see note that follows the table)	
	inTsOffset > outTsOffset	None (error)	Db_buf	
inTs < outTs (Fast to slow)	inTs = outTs / N inTsOffset, outTsOffset = 0	ZOH	Buf	
	inTs = outTs / N inTsOffset ≤ outTsOffset	None (error)		
	inTs = outTs / N inTsOffset > outTsOffset	None (error)	Db_buf	
	inTs ≠ outTs / N	None (error)		
inTs > outTs	inTs = outTs * N	1/z	Db_buf	

Block Settings		**Block Label**		
Rate Transition	**Conditions for Rate Transition Block**	**With Data Integrity and Determinism**	**With Only Data Integrity**	**Without Data Integrity or Determinism**
(Slow to fast)	`inTsOffset`, `outTsOffset` = 0			

	`inTs` = `outTs * N`
	`inTsOffset` ≤ `outTsOffset`
		None (error)		

	`inTs` = `outTs * N`
	`inTsOffset` > `outTsOffset`
		None (error)		

	`inTs` ≠ `outTs * N`	None (error)		

KEY

- `inTs, outTs`: Sample times of input and output ports, respectively
- `inTsOffset, outTsOffset`: Sample time offsets of input and output ports, respectively
- `N`: Integer value > 1

When you select model configuration parameter **Block reduction**, `Copy` reduces to `NoOp`. No code generation occurs for a Rate Transition block with a `NoOp` label. To prevent a block from being reduced when block reduction is on, add a test point to the block output (see Configure Signals as Test Points).

Effects of Asynchronous Sample Times

This table summarizes how each label appears when the sample time of the input or output port (`inTs` or `outTs`) is not periodic, or asynchronous.

Block Settings	**Block Label**		
	With Data Integrity and Determinism	**With Only Data Integrity**	**Without Data Integrity or Determinism**
`inTs` = `outTs`	`Copy`	`Copy`	`Copy`
`inTs` ≠ `outTs`	None (error)	`Db_buf`	

KEY

- `inTs, outTs`: Sample times of input and output ports, respectively

Effects of Device Configuration

If the settings of model configuration parameters **Device vendor** and **Device type** specify hardware that supports atomic data load and store operations, the code generator optimizes the generated rate transition code when the target hardware supports atomic load and store operations for the data type of the signal being transferred. The code generator takes advantage of the hardware data load and store capability by replacing double-buffering code between asynchronous tasks with code that performs a single memory copy.

Ports

Input

collapse all

Port_1 — Input signal
scalar | vector | matrix | N-D array

Input signal that transitions to a new sample rate, specified as a scalar, vector, matrix, or N-D array. To learn about the block parameters that enable you to trade data integrity and deterministic transfer for faster response or lower memory requirements, see Transition Handling Options.

Data Types: `single` | `double` | `half` | `int8` | `int16` | `int32` | `int64` | `uint8` | `uint16` | `uint32` | `uint64` | `Boolean` | `fixed point` | `enumerated` | `bus`

Output

collapse all

Port_1 — Output signal
scalar | vector | matrix | N-D array

Output signal is the input signal converted to the sample rate you specify. The default configuration ensures safe and deterministic data transfer. To learn about the block parameters that enable you to trade data integrity and deterministic transfer for faster response or lower memory requirements, see Transition Handling Options.

Data Types: `single` | `double` | `half` | `int8` | `int16` | `int32` | `int64` | `uint8` | `uint16` | `uint32` | `uint64` | `Boolean` | `fixed point` | `enumerated` | `bus`

Parameters

collapse all

Ensure data integrity during data transfer — Ensure data integrity
on (default) | `off`

Selecting this parameter results in generated code that ensures data integrity when the block transfers data. If you select this parameter and the transfer is nondeterministic (see **Ensure deterministic data transfer**), depending on the priority of input rate and output rate, the generated code uses a proper algorithm using single or multiple buffers to protect data integrity during data transfer.

Otherwise, the Rate Transition block is reduced or generates code by using a copy operation to affect the data transfer. This unprotected mode consumes less memory. However, the copy operation is interruptible, which can lead to loss of data integrity during data transfers. Select this parameter if you want the generated code to operate with maximum responsiveness (that is, nondeterministically) and maintain data integrity. For more information, see Control Data Transfer Behavior in Generated Code (Simulink Coder).

Programmatic Use

Block Parameter: `Integrity`
Type: character vector
Values: `'off'` | `'on'`
Default: `'on'`

Ensure deterministic data transfer (maximum delay) — Ensure deterministic data transfer
on (default) | `off`

Selecting this parameter results in generated code that transfers data at the sample rate of the slower block, that is, deterministically. If you do not select this parameter, data transfers occur when new data is available from the source block and the receiving block is ready to receive the data. You avoid transfer delays, thus ensuring that the system operates with maximum responsiveness. However, transfers can occur unpredictably, which is undesirable in some applications. For more information, see Control Data Transfer Behavior in Generated Code (Simulink Coder).

Programmatic Use

Block Parameter: `Deterministic`
Type: character vector
Values: `'off' | 'on'`
Default: `'on'`

Initial conditions — Initial conditions

`0` (default) | scalar | vector | matrix | N-D array

This parameter applies only to slow-to-fast transitions. It specifies the initial output of the Rate Transition block at the beginning of a transition, when there is no output signal due to the absence of input signal coming from the slow block connected to the input of the Rate Transition block. Simulink® does not allow the initial output of this block to be `Inf` or `NaN`. The value you specify must be a scalar, or have the same dimensions as the input signal.

Programmatic Use

Block Parameter: `InitialCondition`
Type: character vector
Values: finite scalar
Default: `'0'`

Output port sample time options — Mode for specifying output port sample time

`Specify` (default) | `Inherit` | `Multiple of input port sample time`

Specifies a mode for setting the output port sample time.

- `Specify` — Allows you to use the **Output port sample time** parameter to specify the output rate to which the Rate Transition block converts its input rate.
- `Inherit` — Specifies that the Rate Transition block inherits an output rate from the block to which the output port is connected.
- `Multiple of input port sample time` — Allows you to use the **Sample time multiple (>0)** parameter to specify the Rate Transition block output rate as a multiple of its input rate.

If you select `Inherit` and blocks connected to the output port also inherit sample time, the fastest sample time in the model applies.

Programmatic Use

Block Parameter: `OutPortSampleTimeOpt`
Type: character vector
Values: `'Specify' | 'Inherit' | 'Multiple of input port sample time'`
Default: `'Specify'`

Output port sample time — Output rate

`-1` (default) | scalar | vector

Enter a value that specifies the output rate to which the block converts its input rate. The default value (`-1`) specifies that the Rate Transition block inherits the output rate from the block to which the output port is connected. See Specify Sample Time for information on how to specify the output rate.

Dependencies

To enable this parameter, set **Output port sample time options** to `Specify`.

Programmatic Use

Block Parameter: `OutPortSampleTime`
Type: character vector
Values: scalar | vector
Default: `'-1'`

Sample time multiple(>0) — Sample time multiple

1 (default) | positive scalar

Enter a positive value that specifies the output rate as a multiple of the input port sample time. The default value (`1`) specifies that the output rate is the same as the input rate. A value of `0.5` specifies that the output rate is half of the input rate. A value of `2` specifies that the output rate is twice the input rate.

Dependencies

To enable this parameter, set **Output port sample time options** to `Multiple of input port sample time`.

Programmatic Use

Block Parameter: `OutPortSampleTimeMultiple`
Type: character vector
Values: scalar
Default: `'1'`

Block Characteristics

Data Types	`Boolean` | `bus` | `double` | `enumerated` | `fixed point` | `half` | `integer` | `single`
Direct Feedthrough	`yes`
Multidimensional Signals	`yes`
Variable-Size Signals	`no`
Zero-Crossing Detection	`no`

Extended Capabilities

collapse all

C/C++ Code Generation

Generate C and C++ code using Simulink® Coder™.

- Under certain conditions, generated code relies on `memcpy` or `memset` functions (`string.h`) .
- Cannot use inside a triggered subsystem hierarchy.

- Generated code for concurrent programs is platform specific and assumes that the deployment platform and development platform are the same.

HDL Code Generation

Generate VHDL, Verilog and SystemVerilog code for FPGA and ASIC designs using HDL Coder™.

HDL Coder™ provides configuration options that affect HDL implementation and synthesized logic.

Best Practices

- When the Rate Transition block is operating at a fast input rate and slow output rate, it is good practice to follow the Rate Transition block with a unit delay. Doing so prevents the code generator from inserting an extra bypass register in the HDL code. See also Multirate Model Requirements for HDL Code Generation (HDL Coder).
- To upsample the input signal without incurring a unit delay, configure Rate Transition block parameters as follows:
  - Clear **Ensure data integrity during data transfer**.

    Clearing this parameter hides the **Ensure deterministic data transfer (maximum delay)** parameter.
  - Configure the output port sample time of the block to be an integer multiple of the input port sample time. Specify a fractional value of `1/n` for **Sample time multiple**, where `n` is an integer. If **Sample time multiple** uses a value `1/n`, you can choose any value for the block parameter **Output port sample time** options.

> **Note**
>
> When downsampling the input signal, leave the **Ensure data integrity during data transfer** and **Ensure deterministic data transfer (maximum delay)** parameters selected. This mode generates an additional bypass register in the HDL code.

See also Usage of Rate Change and Constant Blocks (HDL Coder).

Restrictions

- The sample rate cannot be 0 or `Inf` for block input or output ports.

HDL Architecture

This block has one default HDL architecture.

HDL Block Properties

ConstrainedOutputPipeline	Number of registers to place at the outputs by moving existing delays within your design. Distributed pipelining does not redistribute these registers. The default is `0`. For more details, see ConstrainedOutputPipeline (HDL Coder).
InputPipeline	Number of input pipeline stages to insert in the generated code. Distributed pipelining and constrained output pipelining can move these registers. The default is `0`. For more details, see InputPipeline (HDL Coder).
OutputPipeline	Number of output pipeline stages to insert in the generated code. Distributed pipelining and constrained output pipelining can move these registers. The default is `0`. For more details, see OutputPipeline (HDL Coder).
AsyncRTAsWire	Map Asynchronous Rate Transition as Wire. When **Output port sample time** has noninteger values, it is considered as an asynchronous rate. Enable the **AsyncRTAsWire** option to generate a wire when such

	asynchronous rates are present for the Rate Transition block. **Dependencies:** This option is available only when the **Ensure data integrity during data transfer** and **Ensure deterministic data transfer** parameters are turned `off`. When using asynchronous rate transition as wire, set **Clock inputs** option to `Multiple`.

Note

While mapping the asynchronous rate transition as wire, you may see the numerical mismatches between Simulink and Code Generation results.

Complex Data Support

This block supports code generation for complex signals.

Fixed-Point Conversion

Design and simulate fixed-point systems using Fixed-Point Designer™.

Version History

Introduced before R2006a

See Also

Probe | Weighted Sample Time

Topics

Specify Sample Time

View Sample Time Information

Data Transfer Representation and Processing (Simulink Coder)