Simulink Blocks

06 February 2021 07:55 PM

1) Model block
   - You can include one model in another model by using a model block.
   - The included model is called a referenced model, and the model that contains the Model block is called the parent model.
   - The Model block displays input and output ports that correspond to the top-level input and output ports of the referenced model. Using these ports allow you to connect the referenced model to other blocks in the parent model.
   - Advantages of model block
     - Modular development
     - Model protection
     - Accelerated simulation
     - Incremental code generation

2) What is work of unit delay block?
   - It holds and delays its input by the sample period we specify.
   - It gives error if we use to create a transition between blocks operating at different sample rates. It is used to solve the algebraic loop error.

3) What are the internal setting in the unit delay block?
   - Initial condition:- 0
   - Input processing:- Elements as channels(Sample based)
   - Sample time:- -1 (Inherited)

4) Doc Block
   - The DocBlock allows you to create and edit text that documents a model, and save that text with the model. Double-clicking an instance of the block creates a temporary file containing the text associated with this block and opens the file in an editor.
   - Use the editor to modify the text and save the file. Simulink® software stores the contents of the saved file in the model file.
   - The DocBlock supports HTML, Rich Text Format (RTF), and ASCII text document types. The default editors for these different document types are
   - Code generation possible with text doc only

5) Delay block
   - In delay block we can specifies delay length, initial condition and external reset using rising, falling, either, level and level hold.

Reset Mode	Behaviour
None	No reset
Rising	Reset on a rising edge

Falling	Reset on a falling edge
Either	Reset on either a rising or falling edge
Level	Reset in either of these cases: • When the reset signal is nonzero at the current time step • When the reset signal value changes from nonzero at the previous time step to zero at the current time step - 1 previous - 0 current
Level hold	Reset when the reset signal is nonzero at the current time step - 1 current

6) Memory block
   - The Memory block holds and delays its input by one major integration time step. When placed in an iterator subsystem, it holds and delays its input by one iteration. This block accepts continuous and discrete signals
   - We can not specifies the sample time in memory block
   - Memory block inherit sample from the driving block

7) Zero order hold block
   - It Convert an input signal with a continuous sample time to an output signal with a discrete sample time.
   - We can specifies sample time for this block

8) On/Off Delay Block
   - When the input becomes true, the output becomes true after a pre-set time delay. The output remains true as long as the input is true.
   - When the input is false or becomes false, the output becomes false with no delay.
   - When the Initial condition of previous input parameter is set to 1, and the input is true at t = 0, the output is true with no delay.
   - The input signal must be Boolean output is Boolean.

9) Signal Builder Block (test case generation)
   - Signal builder block allow you to make customizable signal using signal builder editor window or you can import external signal data vector using excel file or mat file.
   - After import the data vectors from the excel file we can get signals in signal builder block editor window and also the output of signal builder block

10) N-d lookup table
   - A lookup table is an array of data that maps input values to output values
   - N-d lookup table contain breakpoint data and table data
   - Using this data we can convert one signal to another signal
   - Now consider we are driving a car and we want to stop it we then press brake pedal. In this case we giving breakpoint data for pedal position and table data torque from sensor
   - So that pedal position is out brake point data ie. Our input and braking torque is table data ie. Output.

11) What are the commonly used blocks in the Simulink?

Bus Creator	Logical Operator	Constant

Bus Selector	Relational Operator	Delay block
Mux	Scope	Display
Demux	Switch	Subsystem
Constant	Integrator	Gain

12) When to use data store, read and write block ?
   - We can share data between multiple level of model using data store memory block.
   - The location of data store memory block determine which Data Store Read and Data Store Write blocks can access the data store.
   - If the Data Store Memory block is in the top-level system, Data Store Read and Data Store Write blocks anywhere in the model can access the data store.
   - If the Data Store Memory block is in a subsystem, Data Store Read and Data Store Write blocks in the same subsystem or in any subsystem below it in the model hierarchy can access the data store.

13) What is MAAB Guideline for switch block ?
   - Always give Boolean input to switch block and set the condition of switch block is ~=0.

14) Relay Block(Hysteresis)
   - The output for the Relay block switches between two specified values. When the relay is on, it remains on until the input drops below the value of the Switch off point parameter.
   - When the relay is off, it remains off until the input exceeds the value of the Switch on point parameter. The block accepts one input and generates one output.

15) Saturation block
   - The Saturation block produces an output signal that is the value of the input signal bounded to the upper and lower saturation values.
   - The upper and lower limits are specified by the parameters Upper limit and Lower limit.

16) Virtual, non virtual and semi virtual blocks

Block Name	Condition Under Which Block Is Virtual
Bus Assignment	Virtual if input bus is virtual.
Bus Creator	Virtual if output bus is virtual.
Bus Selector	Virtual if input bus is virtual.
Demux	Always virtual.
Enable	Virtual unless connected directly to an Outport block.
From	Always virtual.
Goto	Always virtual.
Goto Tag Visibility	Always virtual.
Ground	Always virtual.
Inport	Virtual *unless* the block resides in a conditionally executed or atomic

	subsystem *and* has a direct connection to an Outport block.
Mux	Always virtual.
Outport	Virtual when the block resides within any subsystem block (conditional or not), and does *not* reside in the root (top-level) Simulink window.
Selector	Virtual only when **Number of input dimensions** specifies 1 and **Index Option** specifies Select all, Index vector (dialog), or Starting index (dialog).
Signal Specification	Always virtual.
Subsystem	Virtual unless the block is conditionally executed or the **Treat as atomic unit** check box is selected. You can check if a block is virtual with the Is Subsystem Virtual block property. See Block-Specific Parameters.
Terminator	Always virtual.
Trigger	Virtual when the output port is *not* present.

17) Scope of GoTo Block
    - The Goto block passes its input to its corresponding From blocks. The input can be a real- or complex-valued signal or vector of any data type. From and Goto blocks allow you to pass a signal from one block to another without actually connecting them.
    - local (default) — From and Goto blocks that use the same tag must be in the same subsystem. A local tag name is enclosed in brackets ([]).
    - scoped — From and Goto blocks that use the same tag must be either:
      In the same subsystem.
    - At any level in the model hierarchy below the Goto Tag Visibility block that does not entail crossing a nonvirtual subsystem boundary. In other words, they must be within the boundary of an atomic, conditionally executed, or function-call subsystem or a model reference.
    - A scoped tag name is enclosed in braces ({}).
    - global — From and Goto blocks using the same tag can be anywhere in the model except in locations that span nonvirtual subsystem boundaries.

18) Gain or Multiply block
    - The Gain block multiplies the input by a constant value (gain). The input and the gain can each be a scalar, vector, or matrix.
    - You specify the value of gain in the Gain parameter. The Multiplication parameter lets you specify element-wise or matrix multiplication. For matrix multiplication, this parameter also lets you indicate the order of the multiplicands.

19) Product Block
    - The Product block outputs the result of multiplying two inputs: two scalars, a scalar and a nonscalar, or two nonscalars that have the same dimensions.

20) Switch and Multiport Switch
    - Switch
        - The Switch block passes through the first input or the third input signal based on the value of the second input. The first and third inputs are data input. The second input is a

control input. Specify the condition under which the block passes the first input by using the Criteria for passing first input and Threshold parameters.

- Multiport Switch
  - The Multiport Switch block determines which of several inputs to the block passes to the output. The block bases this decision on the value of the first input. The first input is the control input and the remaining inputs are the data inputs. The value of the control input determines which data input passes to the output.

21) Truth Table
    - Truth tables implement combinatorial logic design in a concise, tabular format. Typical applications for truth tables include decision making for:
      - Fault detection and management
      - Mode switching
    - Truth tables are supported only in Simulink®.
    - You can add a Truth Table block directly to your Simulink model, or you can define a truth table function in a Stateflow® chart, state, or subchart. Truth Table blocks in a Simulink model execute as a Simulink block, while truth table functions in a Stateflow chart execute only when you call the truth table function. The location of the function determines the set of states and transitions that can call the function.

22) Rate Limiter
    - The Rate Limiter block limits the first derivative of the signal passing through it. The output changes no faster than the specified limit.

23) Bus Assignment block
    - The Bus Assignment block assigns the values of a signal to bus elements. Use a Bus Assignment block to change bus element values without adding Bus Selector and Bus Creator blocks that select bus elements and reassemble them into a bus.
    - Connect the bus signal to the first input port. To other input ports, connect one or more signals whose values you want to assign to a bus element. Use the Block Parameters dialog box to specify the bus elements to be replaced. The block displays an assignment input port for each such element. For an example of a model that uses a Bus Assignment block, see Assign Signal Values to a Bus.