Description

In this example, we will build a host-target (master-slave) communication system to exchange scalar as shown in the diagram below.

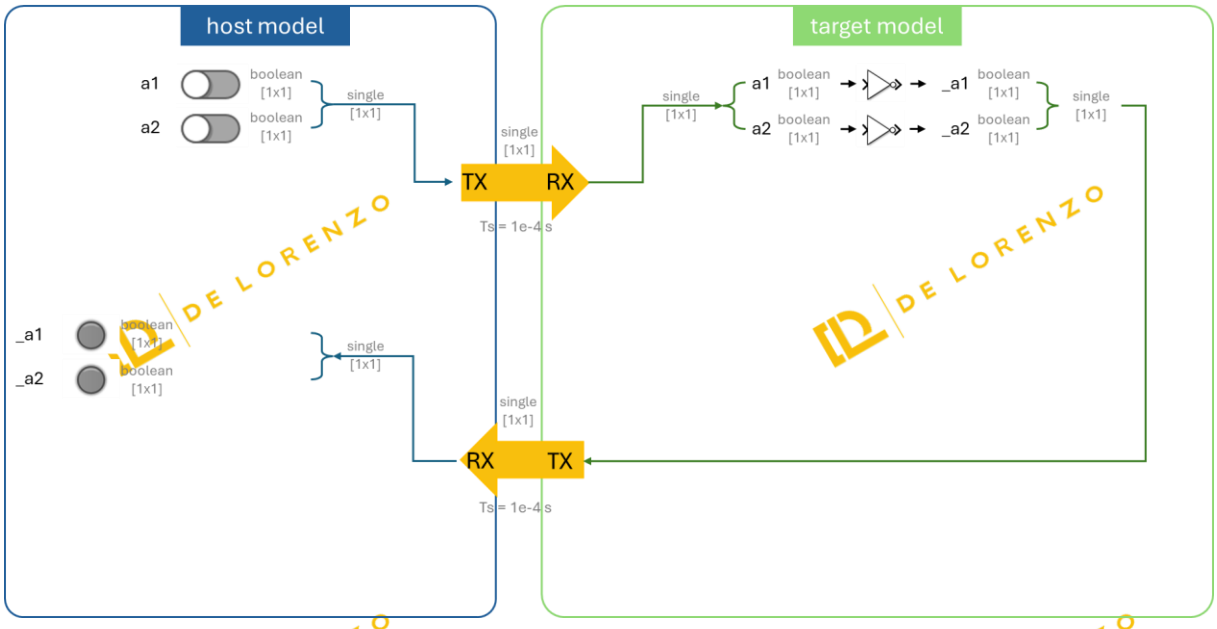
The host model running in the master (PC) will package and transmit data to the target model running in the slave device (DL RCPCORE), including 2 Boolean scalar values.

The DL RCPCORE (slave) will unpack the received data, perform simple processing (in this example, logic inversion), then package and return the processed data back to the master.

The master will then receive and unpack the data, display the results, and verify the correctness of the communication.

TX/RX Timing & Signal Dimensions

The host model performs a TX [1x1] at regular intervals of 1e-4 s;
The target model performs RX [1x1] at regular intervals of 1e-4 s;
The target model performs TX [1x1] at regular intervals of 1e-4 s;
The host model performs RX [1x1] at the same interval as the target TX, i.e. 1e-4 s.



Learning Objectives

- how to set up a simple host-target communication system
- understand and differentiate the host model and the target model through hands-on practice
- data serialization and parsing
- understand the significance of time in communication
- dashboard development

Prior Knowledge

- Create New Project
- Control Model Settings
- Host-Target Communication
    - Fundamentals

Demo Models

If you want to run the demo directly, please execute the following command:

openDLDemo('comm_scalar')

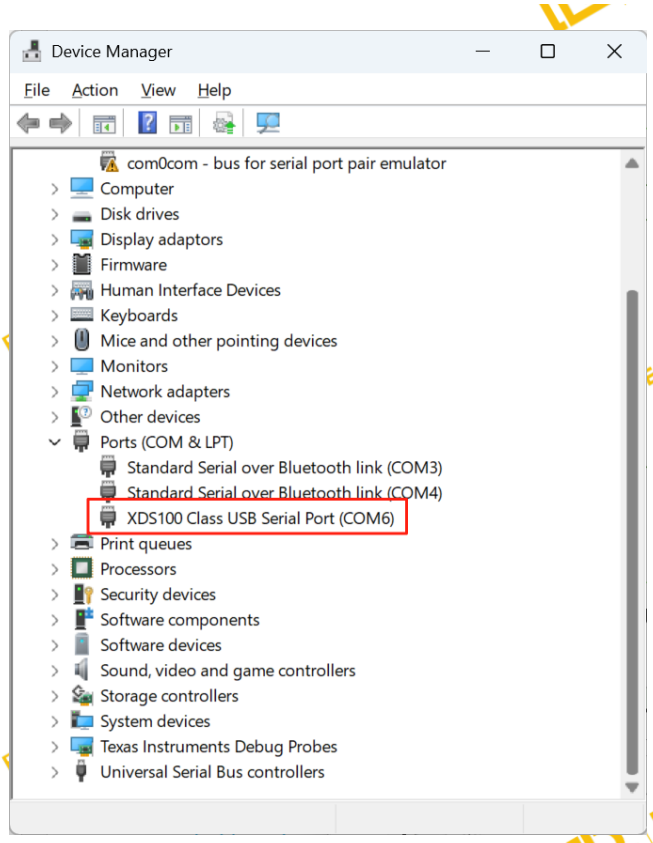
Step 0. Model creation & parameter settings

Create new host and target models using the *CreateNew* command.

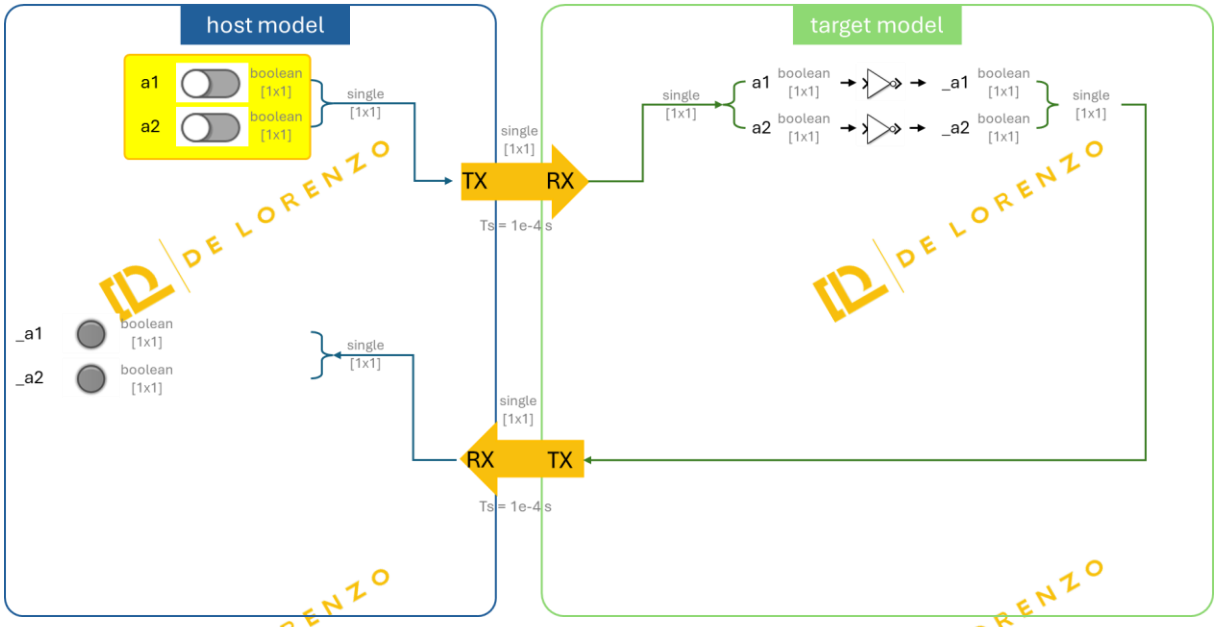
Click on ' 📌 Click here to edit model parameters' to edit the parameters.

Set the model base sample time **Ts** (which here corresponds to the transmission interval) and the **COM port** recognized by the hardware in Device Manager.

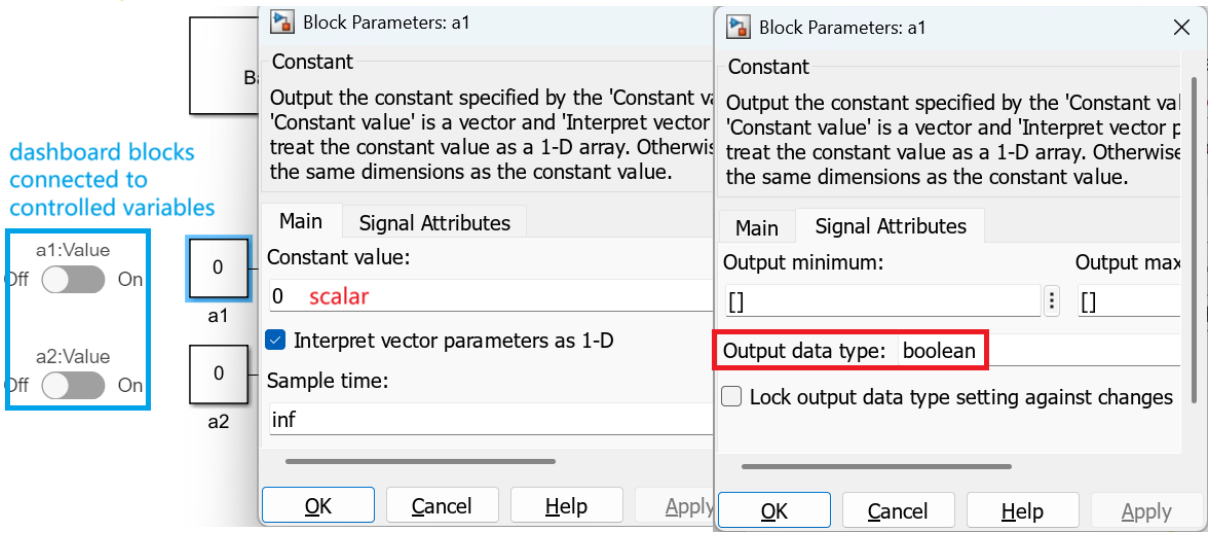
```
%% Set PWM Switching frequency
PWM_frequency    = 4e3;          %Hz          // Converter
T_pwm            = 1/PWM_frequency;  %s  // PWM switching
deadtime         = 5e-6;         %s //rising edge delay fo

%% Model Parameters
Ts               = 1e-4;        %s      // Sample time for
dataType         = 'single';            % Floating point

%% Hareware parameters
% Set controller parameters
target           = SetControllerDetails('DL RCPCORE', PWM_
target.comport   = 'COM6';       % Update the appropriate
converter        = SetConverterParameters('DL 2106T06');
```



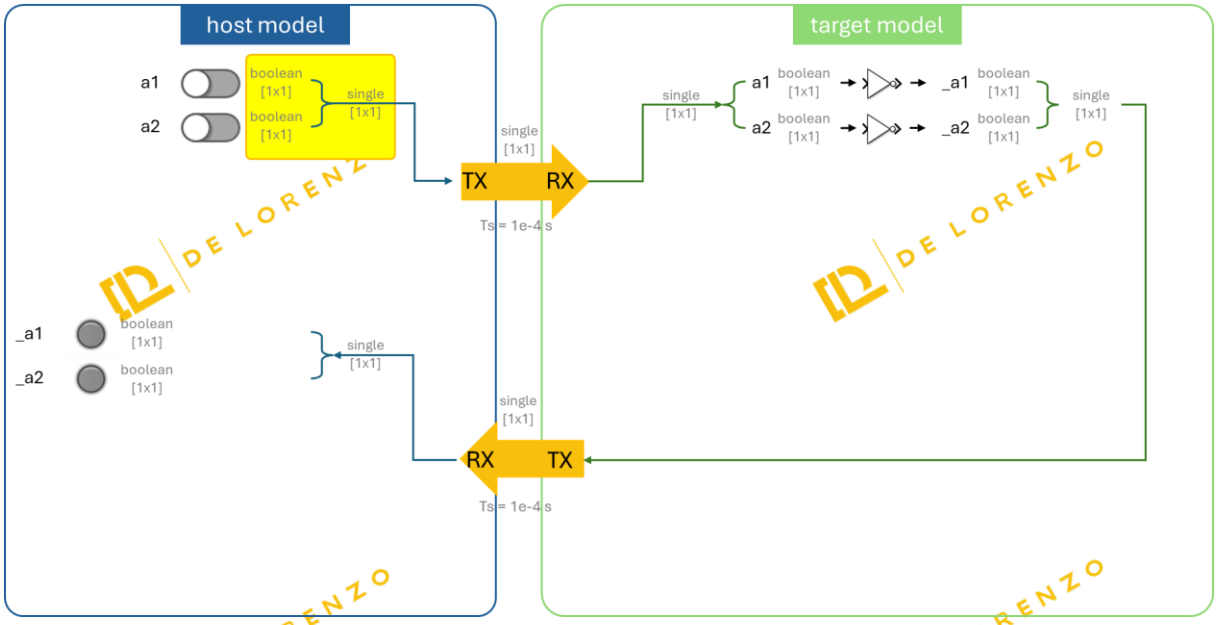
Step 1. Create controlled variables



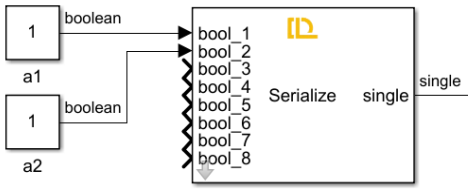
First, we add the corresponding control variables to the host model according to the requirements.



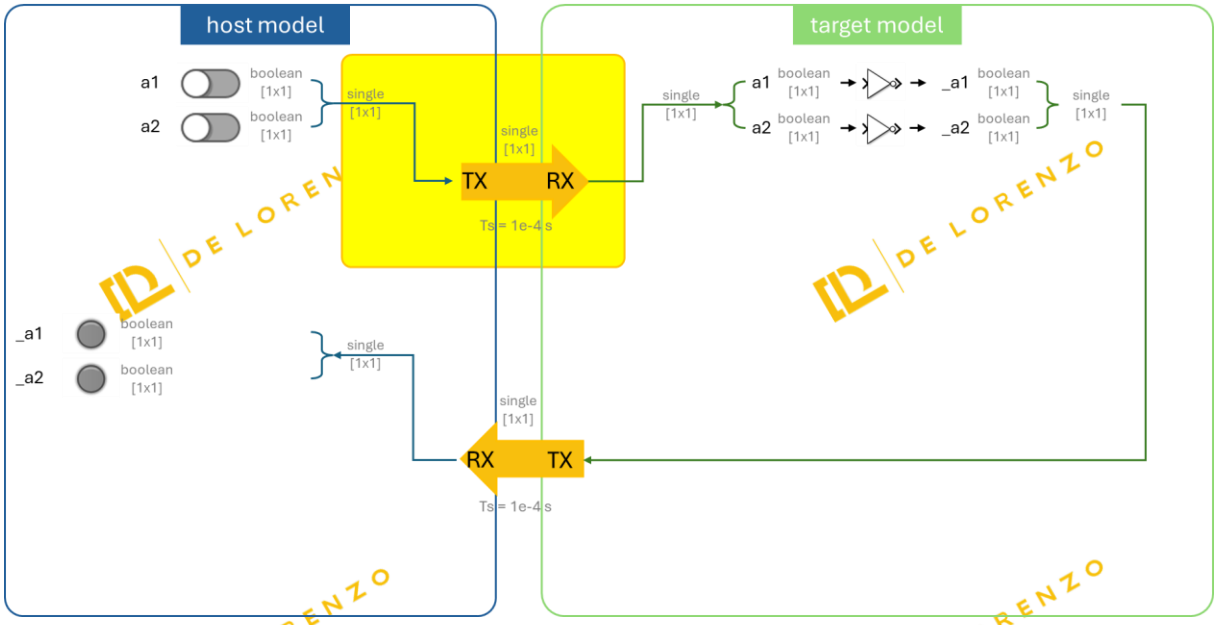
Step 2. Serialize multiple same-type signals into a single signal



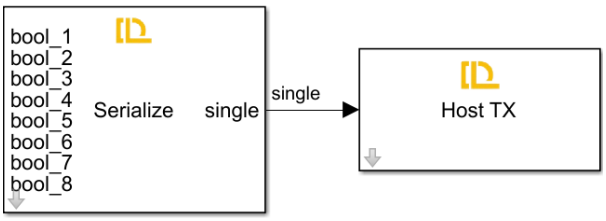
Use the **'Serialize Boolean into Single'** block to combine multiple **Boolean** signals into a **single**-precision value.



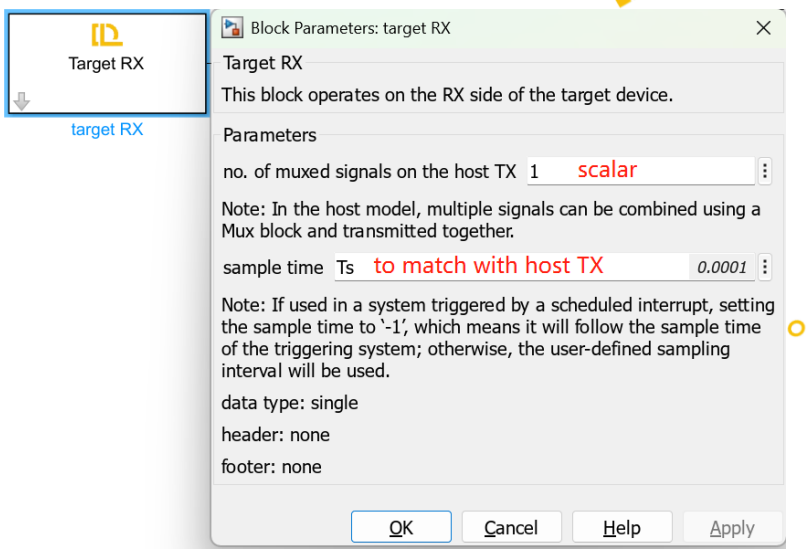
Step 3. Host-to-target transmission



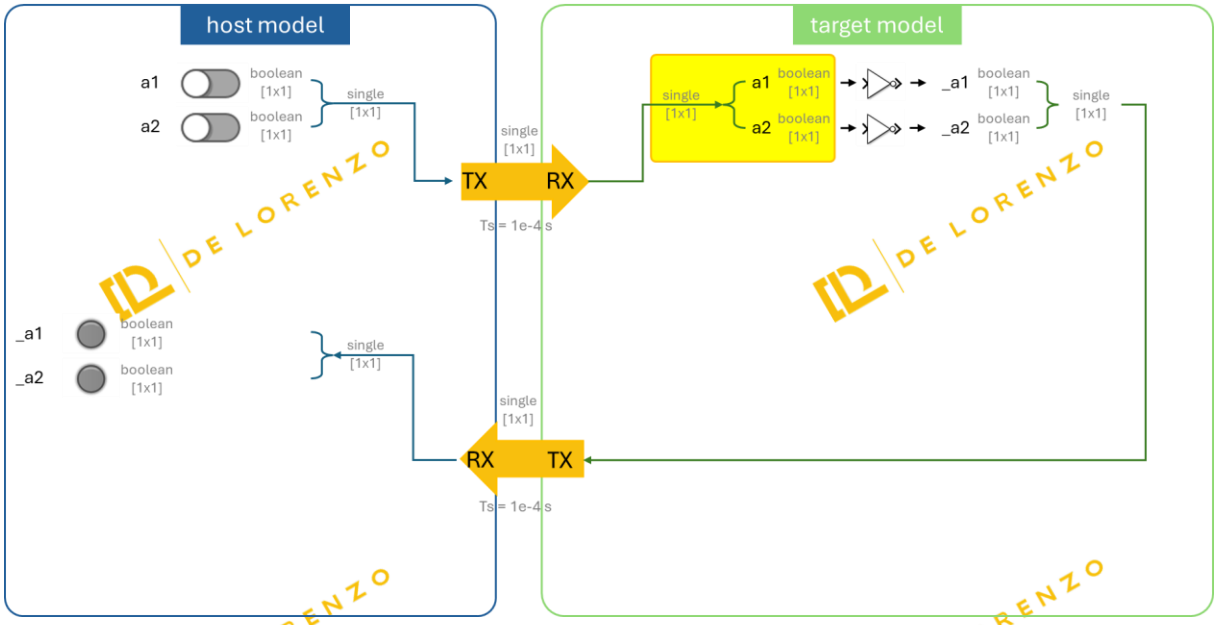
In the **host model**, place the **'Host TX'** block. The transmission rate will follow the sampling interval of the **'host RX'** block which is Ts = 1e-4 s.



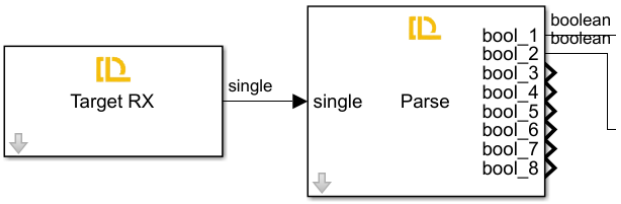
In the **target model**, place the **'Target RX'** block. Since only a single scalar is transmitted from the host side, the number of columns (i.e., the number of single-type signals) at the RX end on the target is set to 1. The reception interval is set to Ts = 1e-4 s.



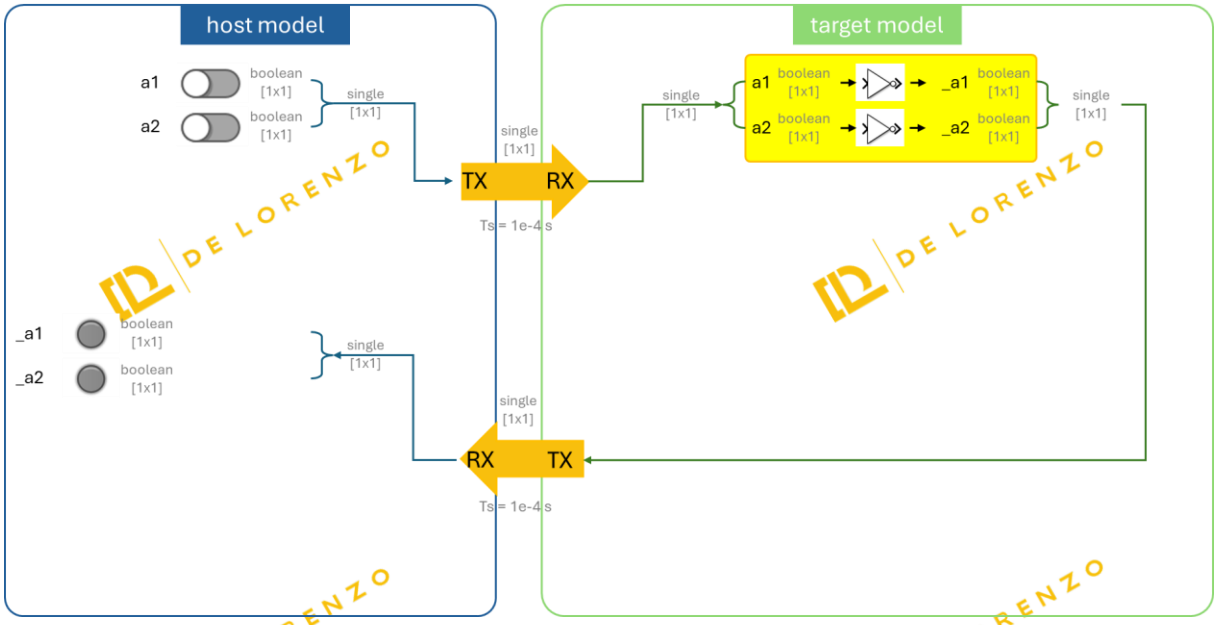
Step 4. Parse to recover the controlled variables



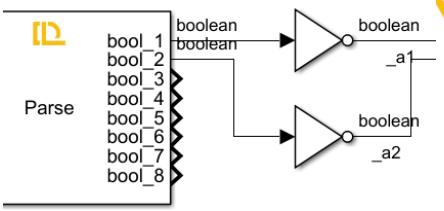
Use the **'Parse Single into Boolean'** block to convert a **single**-type signal into multiple **Boolean** signals and extract the controlled variables from the corresponding channels.



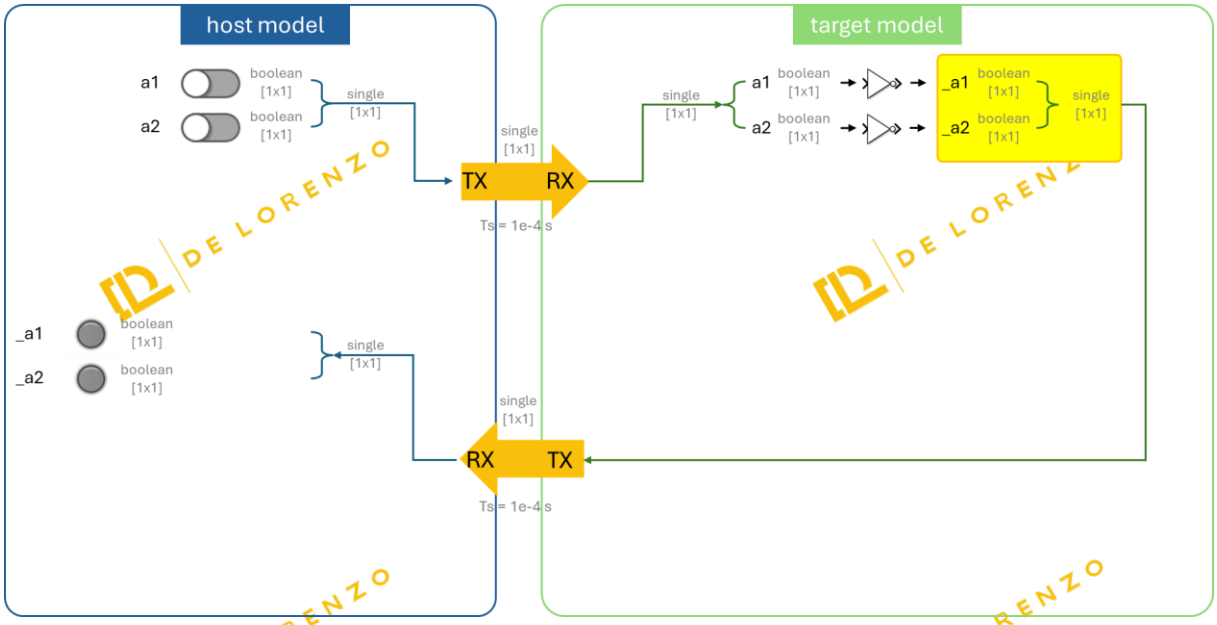
Step 5. Simple Signal Processing



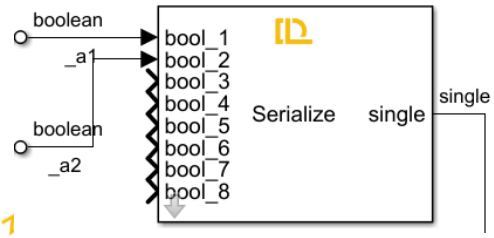
Invert Boolean signals a1 and a2 to obtain _a1 and _a2.



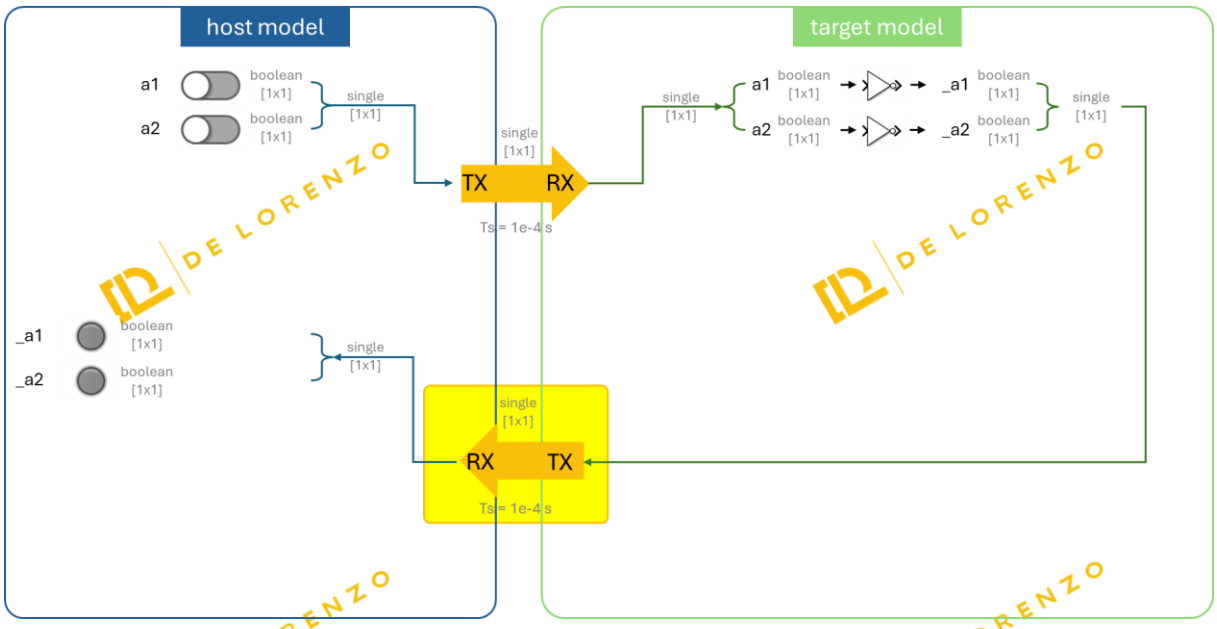
Step 6. Serialize multiple same-type signals into a single signal



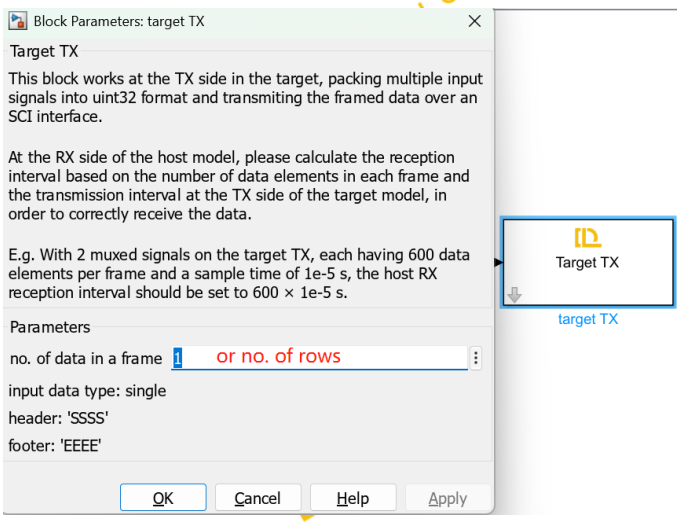
Use the **'Serialize Boolean into Single'** block to combine multiple **Boolean** signals into a **single**-precision value.

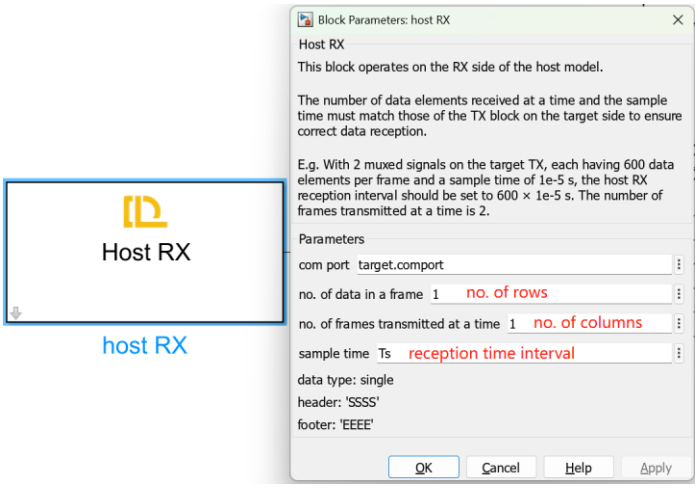


Step 7. Target-to-host transmission



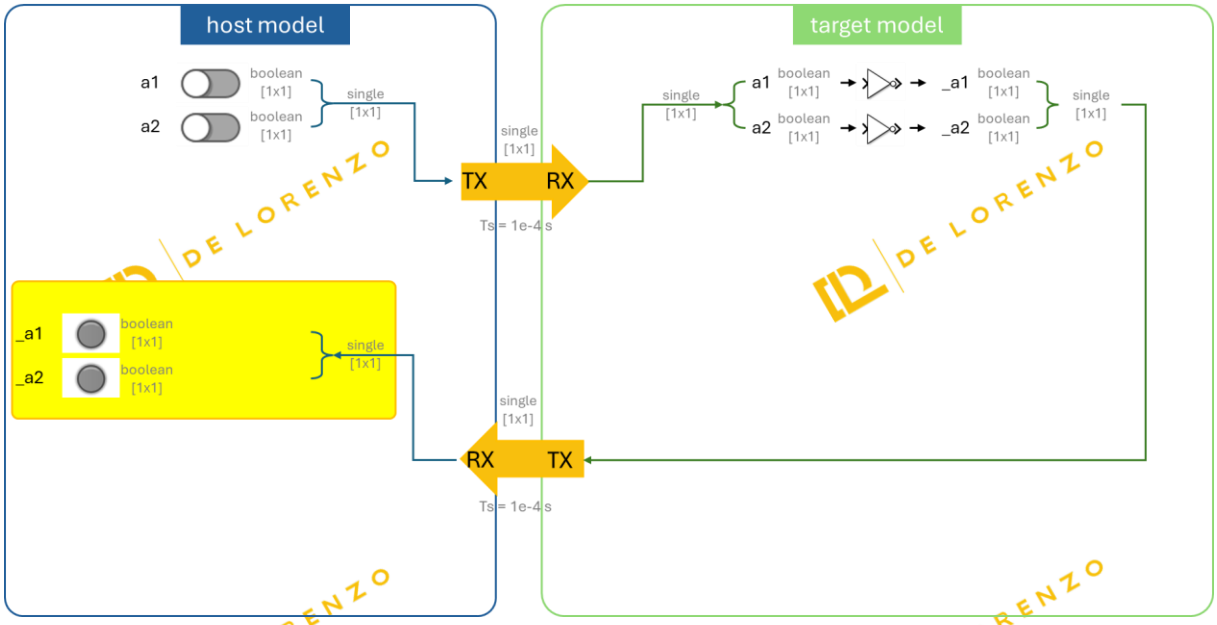
In the **target model**, place the **'Target TX'** block. The transmission rate will follow the target RX interval. Since the transmission and reception intervals on both the host and target sides are identical and only a single scalar is being transferred, the number of rows in the data — that is, the amount of data contained in one frame — is set to 1.



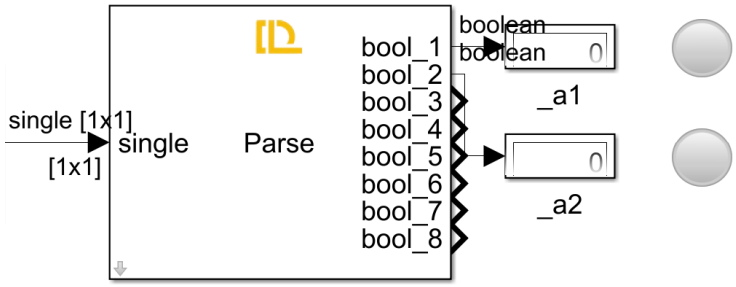


In the host model, place the **'Host RX'** block. For receiving a single scalar value, set both the number of rows and the number of columns to 1. Ensure that the reception interval matches the transmission interval.

Step 8. Parse to recover the processed variables



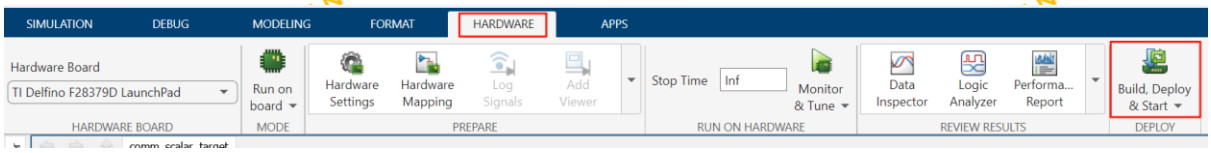
Use the **'Parse Single into Boolean'** block to convert a **single**-type signal into multiple **Boolean** signals and extract the controlled variables from the corresponding channels.



Step 9. Debugging

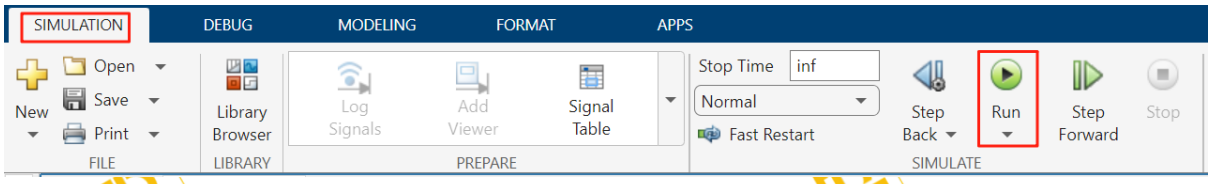
In the **target model**, compile and deploy the model to the DL RCPCORE hardware.

Wait for the 'Ready' message in the lower-left corner. If an error occurs, check and resolve it.

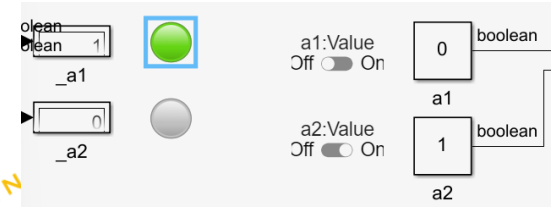


In the **host model**, after confirming that the target model has been deployed and is running on the DL RCPCORE, run the host model to begin debugging.

If an error occurs, check and resolve it.



Modify the control variables and verify whether the values processed by the target model are correctly sent back.



***Note**: Pay attention to the lamp color settings.*