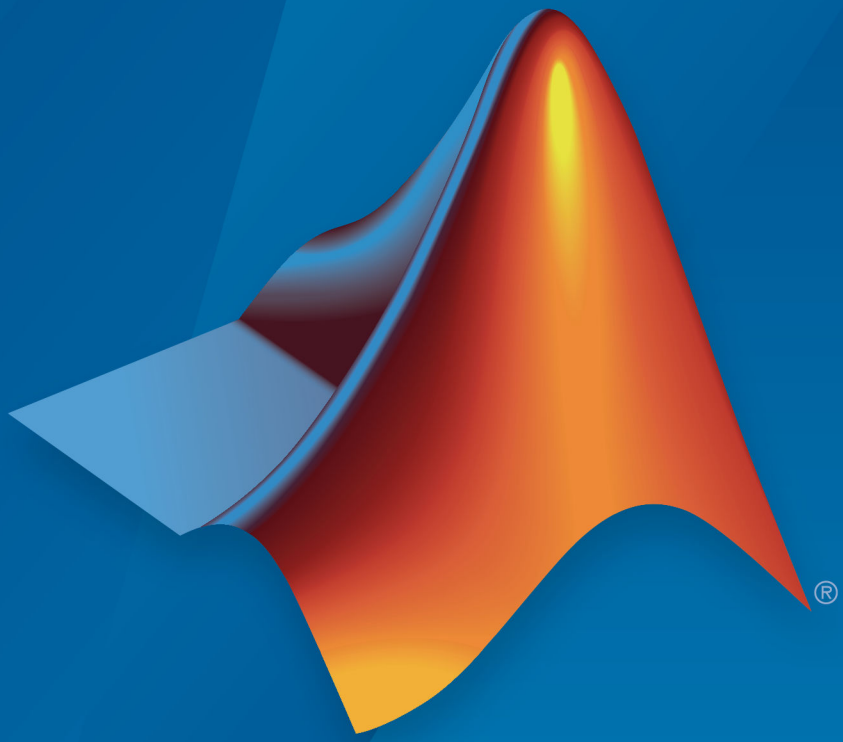
SerDes Toolbox™

User's Guide



MATLAB® & SIMULINK®

R2019a

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

SerDes Toolbox™ User's Guide

Revision History

March 2019	Online only	New for Version 1.0 (Release 2019a)

Contents

1 SerDes Toolbox Topics

2 Customize SerDes Systems

3 Customize IBIS-AMI Models

4 Industry Standard IBIS-AMI Models

SerDes Toolbox Topics

Fundamentals of SerDes Systems

Modern high-speed electronic systems are characterized by increased data speed integrated circuits (ICs). The input/output performance remains the bottleneck that limits the overall performance of a high-speed system. Serial data transfer is the most efficient way of communicating large data quickly between computer chips on printed circuit boards through copper cables and through short, medium, and long length fiber optics.

Thus, many systems now aggregate and serialize multiple input/ output (I/O) signals for transmission across fiber and copper cables and PCBs at a higher data rate, recovering and de-serializing the individual signals on the receiving end. These SerDes (Serializer/De-Serializer) implementations employ additional silicon real estate to perform sophisticated equalization required for reliable signal transmission at very high data speeds. This approach helps maximize throughput at the system level.

SerDes design is a complex, iterative process that typically starts with a baseline SerDes system that demonstrates the feasibility of a design approach. This system also establishes budgets for the different parts of the serial channel and associated transmitter (TX) and receiver (RX) equalization circuitry. The data that describes the desired behavior of each of the equalization filters in both the transmitter and the receiver is then back-annotated in the behavioral models with the correlation with simulations or measurements. The final step is to implement the training algorithms and control loops that will be executed by the chip during startup and from time to time when the channel needs to be retrained.

The SerDes system is then compiled into IBIS-AMI (Input/Output Buffer Information Specifications — Algorithmic Model Interface) models.

There are six sections of a SerDes system:

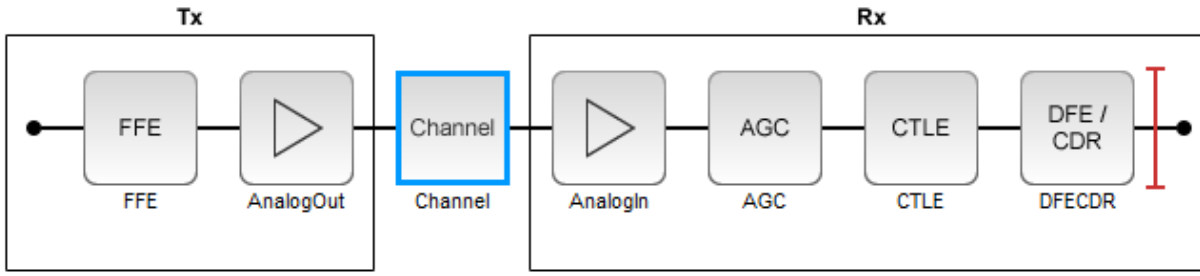
- TX equalization — This becomes the IBIS-AMI dll for the transmitter.
- TX AnalogOut — This becomes the analog model of the transmitter. It is part of the IBIS model for TX, and is typically represented by the I-V and V-T characteristics curves in the `.ibs` file.
- Channel — This becomes the model of the physical channel, including the TX and RX package models.
- RX AnalogOut — This becomes the analog model of the receiver. It is part of the IBIS model for RX, and is typically represented by the I-V and V-T characteristics curves in the `.ibs` file.

- RX equalization — This becomes the IBIS-AMI dll for the receiver.
- Training algorithms and control loops — These become the on-chip microcode that is executed inside of the chip during startup and when the channel needs to be retrained.

Customize SerDes System in MATLAB

Open the **SerDes Designer** app. In the **CONFIGURATION** tab of the app toolstrip, set **Symbol Time (ps)** to 125 and **Target BER** to `1e-12`.

In a new blank canvas, add an FFE block to the **Tx** side. Add an AGC, a CTLE and a DFECDR block to the **Rx** side.



Select the channel block. Set **Channel loss (dB)** to 13.

From the **EXPORT** tab of the app toolstrip, select **Generate MATLAB code for SerDes System**. A MATLAB® script open that represents the command line interface to the SerDes system.

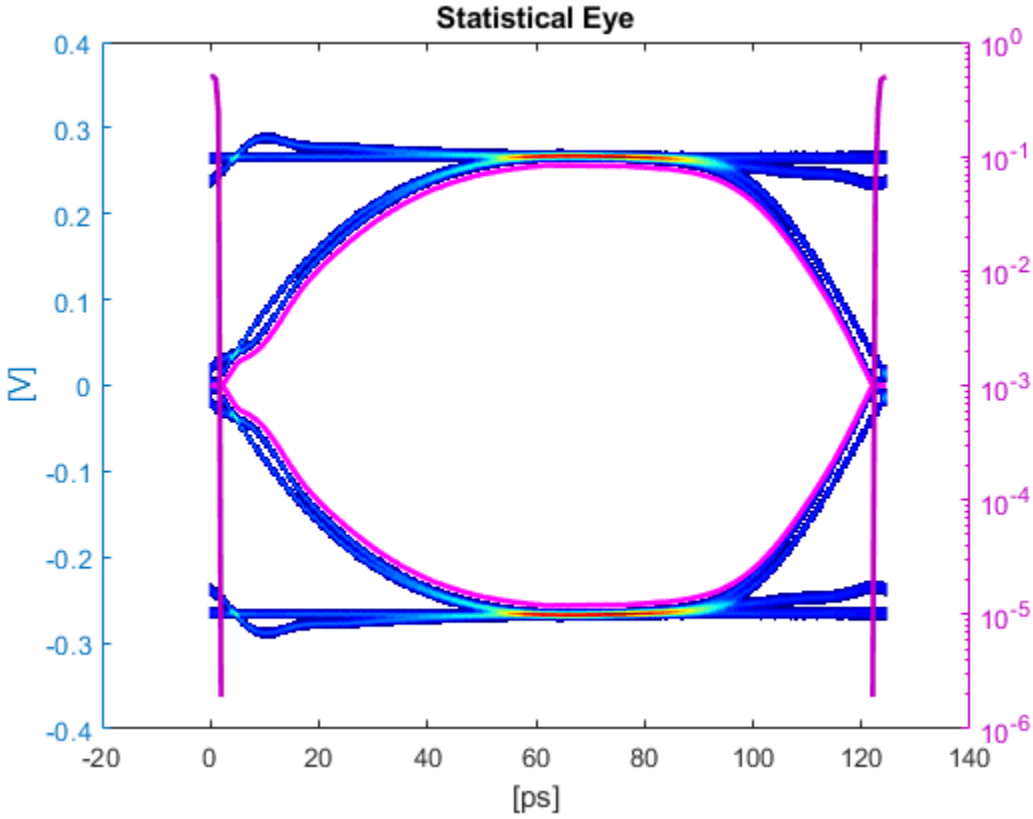
The MATLAB script contains the code to generate the transmitter and receiver building blocks and analog models. It also contains the channel information and SerDes system configuration. The script exposes every parameter that is part of the SerDes system. You can modify the parameters to further explore the SerDes system.

For example, to see the effect of **Channel loss** on the SerDes system, scroll down to the section of the MATLAB script that says `% Build ChannelData`. Replace the default code section with the following code:

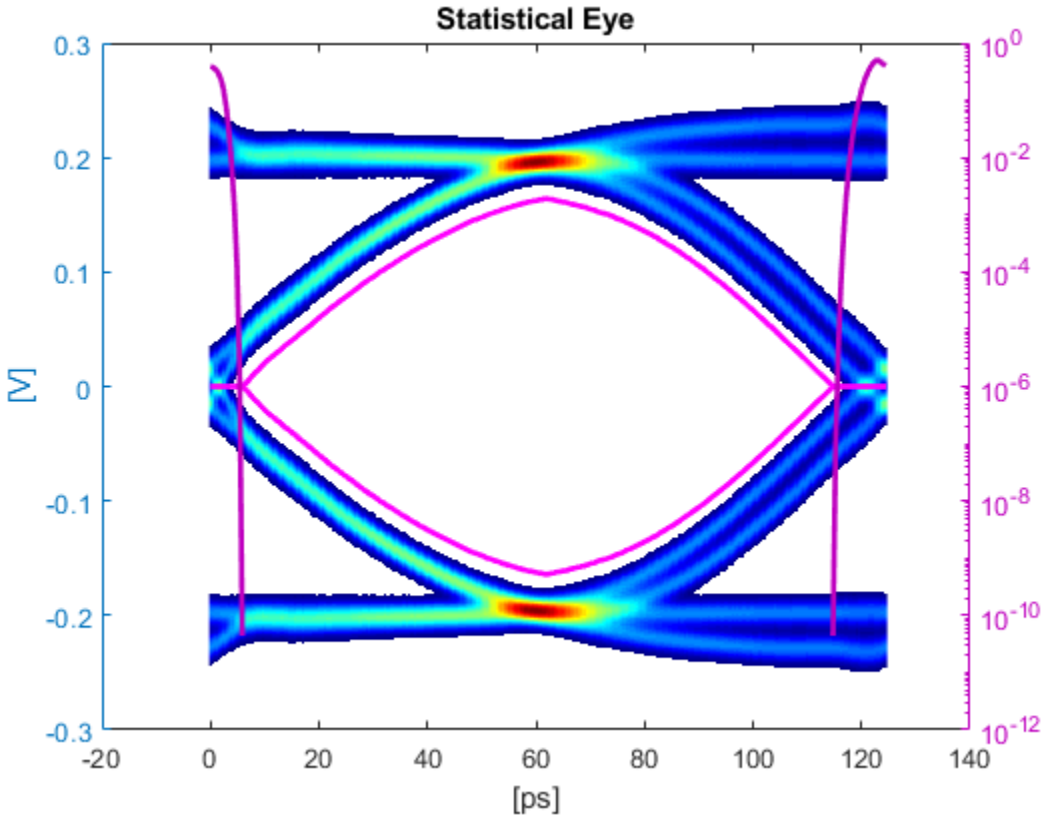
```
% Build ChannelData:
channelLoss = 5;
channel = ChannelData( ...
    'ChannelLossdB',channelLoss, ...
    'ChannelLossFreq',5000000000, ...
    'ChannelDifferentialImpedance',100);
```

Save the change and run the script. Keep changing the value of `channelLoss` to see the effect of changing channel loss.

The eye diagram when the **Channel loss** is set to 5 dB:



The eye diagram when the **Channel loss** is set to 16 dB:



After you finalize the SerDes system with your desired Channel Loss, you can export the MATLAB script of the SerDes system as a Simulink® model. From the Simulink canvas, you can perform further time-domain analysis, or export the system to a AMI model.

Customize SerDes System in Simulink

Overview of AMI Parameters

2

Customize SerDes Systems

Customizing SerDes Toolbox Datapath Control Signals

This example shows how to customize the control signals in a SerDes system datapath by adding new custom AMI parameters and using MATLAB® function blocks. This allows you to customize existing control parameters without modifying the built-in blocks in the SerDes Toolbox™ library.

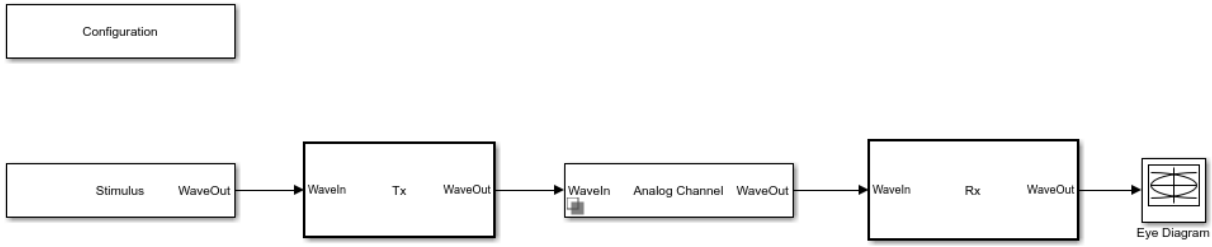
This example shows how to add a new AMI parameter to control the operation of the three transmitter taps used by the FFE block. The custom AMI parameter simultaneously sets all three taps to one of the ten values defined by the PCIe4 specification or allows you to enter three custom floating-point tap values. To know more about how to define a PCIe4 transmitter model, see “PCIe4 Transmitter/Receiver IBIS-AMI Model” on page 4-2.

PCIe4 Transfer Model

The transmitter model in this example complies with the PCIe4 specification. The receiver is a simple pass-through model. A PCIe4 compliant transmitter uses a 3-tap feed forward equalizer (FFE) with one pre-tap and one post-tap, and ten presets.

Open the model `adding_tx_ffe_params`. The SerDes system Simulink® model consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks.

```
open_system('adding_tx_ffe_params.slx')
```



- The Tx subsystem contains an FFE block to model the time-domain portion of the AMI model and an Init block to model the statistical portion.
- The Analog Channel block has the PCIe4 parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx analog model parameters.
- The Rx subsystem has a Pass-Through block and an Init block.

Add New AMI Parameter

Add a new AMI parameter to the transmitter which is available to both the Init and GetWave datapath blocks and functions. The parameter is also included in the Tx IBIS-AMI file.

Double-click the Configuration block to open the Block Parameters dialog box. Click the **Open SerDes IBIS-AMI Manager** button. Go to the **AMI-Tx** tab of the SerDeS IBIS-AMI Manager dialog box.

- Select the FFE parameter, then click **Add Parameter...** to add a new FFE sub-parameter.
- Set the Parameter name to `ConfigSelect`.
- Keep the **Current value** as `0`.
- In the Description, add `Pre/Main/Post tap configuration selector`.
- Keep the **Usage** as `In`.
- Set the **Type** to `Integer`.
- Set the **Format** to `List`.
- Under the **List Format details**, set **Default** to `0`.
- Set **List values** to `[-1 0 1 2 3 4 5 6 7 8 9]`
- Set **List_Tip values** to `["User Defined" "P0" "P1" "P2" "P3" "P4" "P5" "P6" "P7" "P8" "P9"]`

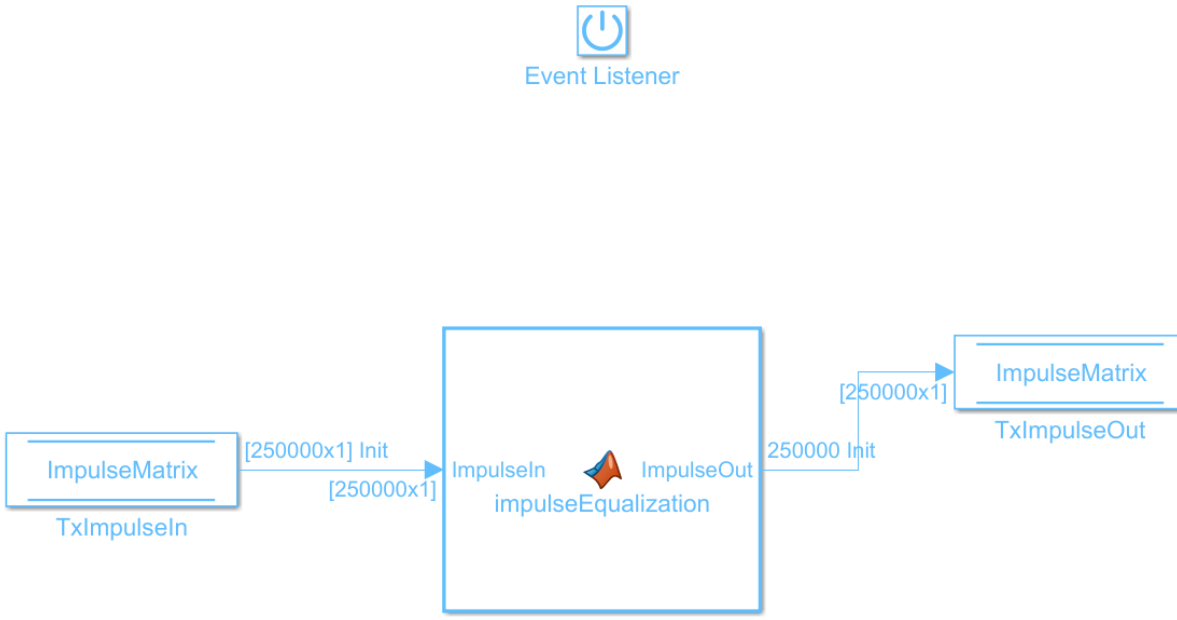
A new parameter **ConfigSelect*** is added to the **AMI-Tx** tab.

Modify Init

Modify the Initialize MATLAB function inside the Init block in the Tx subsystem to use the newly added **ConfigSelect***parameter. The **ConfigSelect*** parameter controls the existing three transmitter taps. To accomplish this, add a switch statement that takes in the values of **ConfigSelect*** and automatically sets the values for all three Tx taps, ignoring the user defined values for each tap. If a **ConfigSelect** value of `-1` is used, then the user-defined Tx tap values are passed through to the FFE datapath block unchanged.

Inside the Tx subsystem, double-click the Init block to open the Block Parameters dialog box and click the **Refresh Init** button to propagate the new AMI parameter to the Initialize sub-system.

Type **Ctrl-U** to look under the mask for the Init block, then double-click on the initialize block to open the Initialize Function.



Double-click on the impulseEqualization MATLAB function block to open the function in MATLAB. This is an automatically generated function which provides the impulse response processing of the SerDes system block (IBIS AMI-Init). The `%% BEGIN:` and `% END:` lines denote the section where custom user code can be entered. Data in this section will not get over-written when Refresh Init is run:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
FFEParameter.ConfigSelect; % User added AMI parameter
% END: Custom user code area (retained when 'Refresh Init' button is pressed)
```

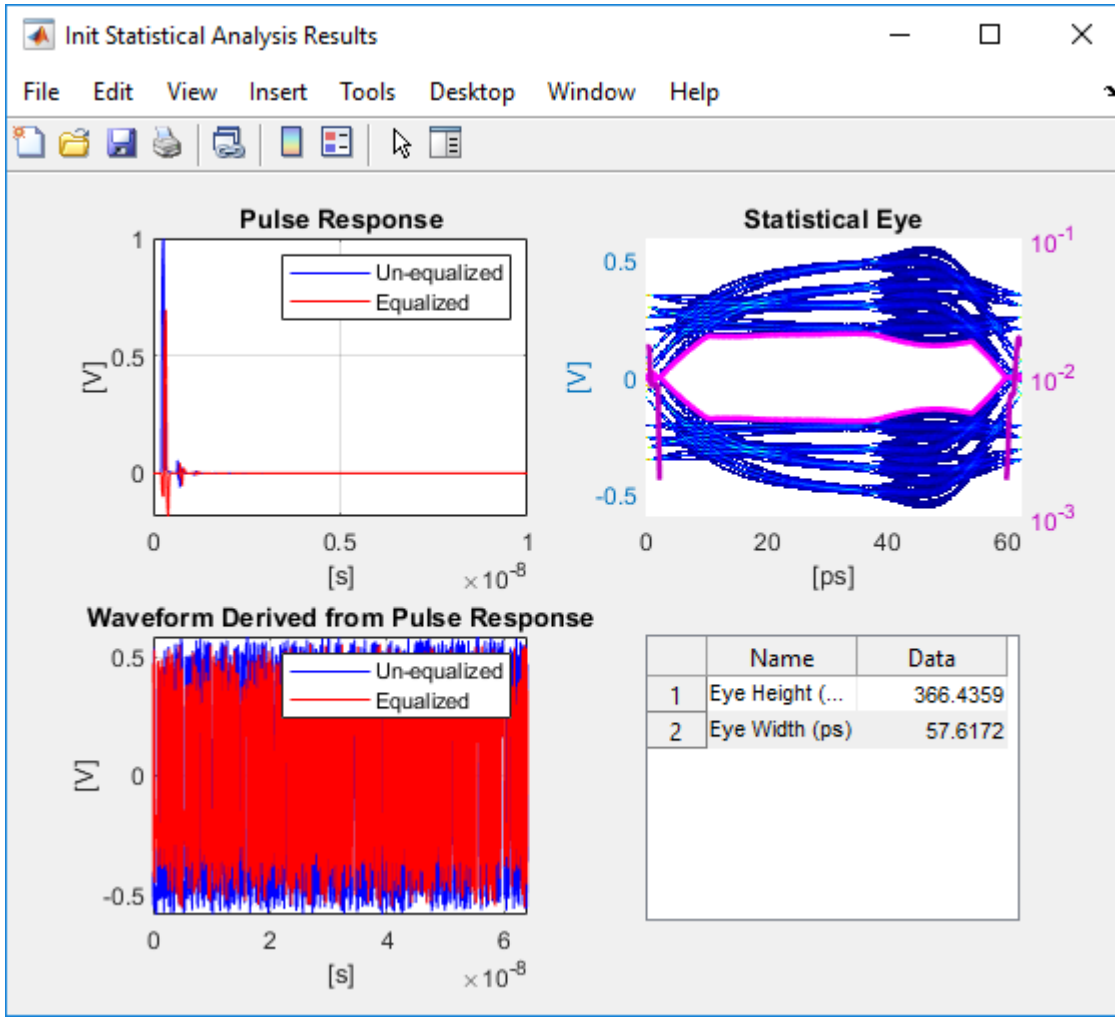
To add the custom ConfigSelect control code, scroll down the Customer user code area, comment out the `FFEParameter.ConfigSelect` line, then enter the following code:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
%FFEParameter.ConfigSelect; % User added AMI parameter
switch FFEParameter.ConfigSelect
case -1  % User defined tap weights
FFEInit.TapWeights = FFEParameter.TapWeights;
```

```
case 0  % PCIe Configuration: P0
FFEInit.TapWeights = [0.000 0.750 -0.250];
case 1  % PCIe Configuration: P1
FFEInit.TapWeights = [0.000 0.830 -0.167];
case 2  % PCIe Configuration: P2
FFEInit.TapWeights = [0.000 0.800 -0.200];
case 3  % PCIe Configuration: P3
FFEInit.TapWeights = [0.000 0.875 -0.125];
case 4  % PCIe Configuration: P4
FFEInit.TapWeights = [0.000 1.000 0.000];
case 5  % PCIe Configuration: P5
FFEInit.TapWeights = [-0.100 0.900 0.000];
case 6  % PCIe Configuration: P6
FFEInit.TapWeights = [-0.125 0.875 0.000];
case 7  % PCIe Configuration: P7
FFEInit.TapWeights = [-0.100 0.700 -0.200];
case 8  % PCIe Configuration: P8
FFEInit.TapWeights = [-0.125 0.750 -0.125];
case 9  % PCIe Configuration: P9
FFEInit.TapWeights = [-0.166 0.834 0.000];
otherwise
FFEInit.TapWeights = FFEParameter.TapWeights;
end
% END: Custom user code area (retained when 'Refresh Init' button is pressed)
```

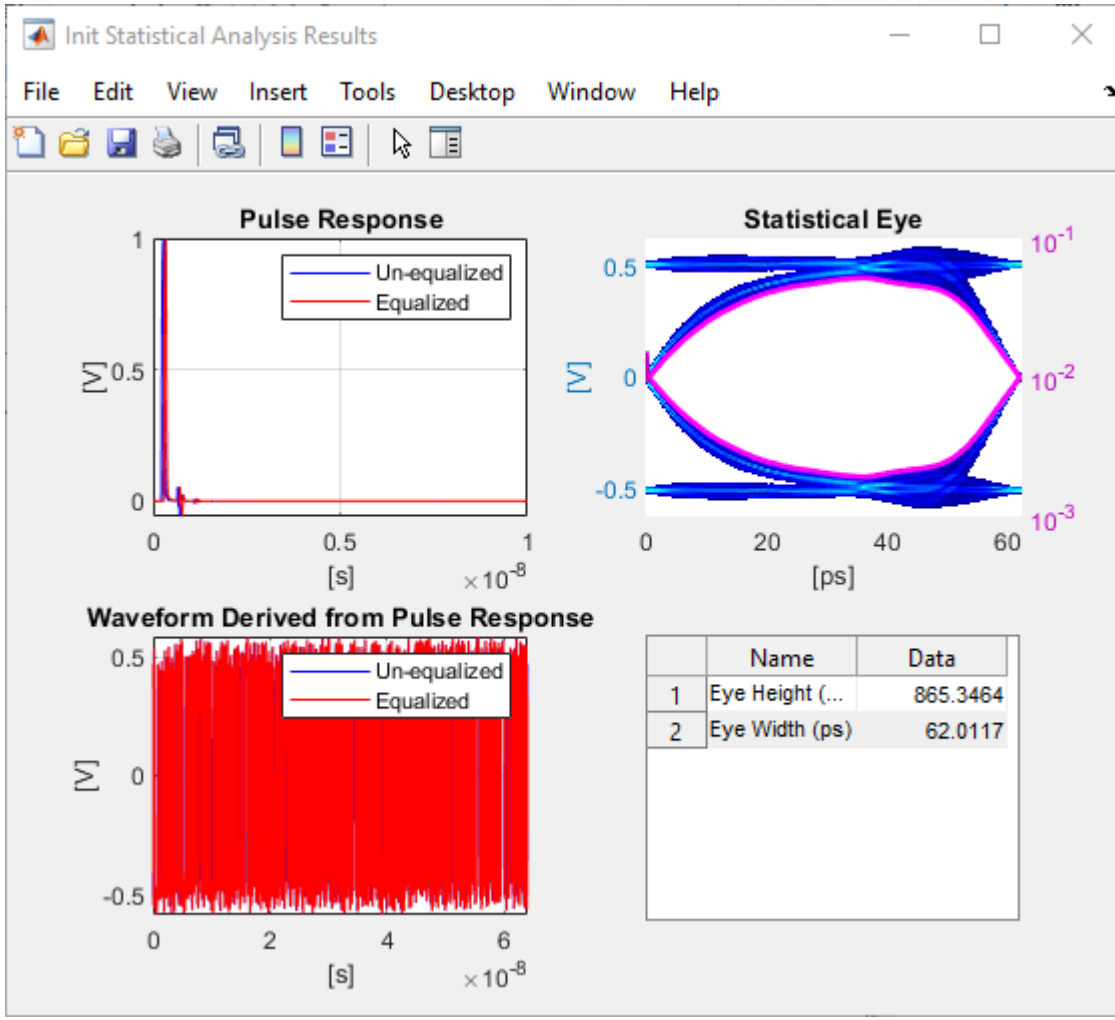
To test that the new FFE control parameter is working correctly, open the SerDes IBIS-AMI Manager dialog box from the Configuration block. In the **AMI-Tx** tab, edit the **ConfigSelect*** parameter to set **Current value** to P7. This corresponds to PCIe Configuration P7: Pre = -0.100, Main = 0.700 and Post = -0.200.

Run the simulation and observe the results of Init statistical analysis.



Next, set the **Current value** of the **ConfigSelect*** parameter to `User Defined`. This corresponds to user-defined tap weights: Pre = 0.000, Main = 1.000 and Post = 0.000.

Run the simulation and observe the results of Init statistical analysis.



Try different values of **ConfigSelect*** to verify proper operation. The statistical eye opens and closes based on the amount of equalization applied by the FFE. How much the eye changes, and the tap values that create the most open eye varies based on the loss defined in the Analog Channel block.

Modify GetWave

To modify GetWave, add a new MATLAB function that operates in the same manner as the Initialize function.

Inside the Tx subsystem, type **Ctrl-U** to look under the mask of the FFE block.

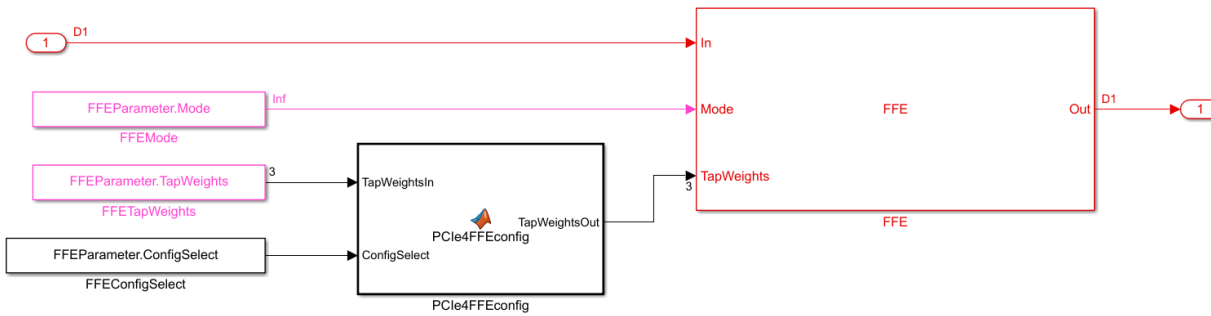
- Add a Constant block to the canvas from the Simulink/Sources library.
- Rename the Constant block as `FFEConfigSelect` and set the **Constant value** to `FFEParameter.ConfigSelect`.
- Add a MATLAB Function block to the canvas from the Simulink/User-Defined library.
- Rename the MATLAB Function block to `PCIe4FFEconfig`.
- Double-click the MATLAB Function block and replace the template code with the following:

```
% PCIe4 tap configuration selector
%   Selects pre-defined Tx FFE tap weights based on PCIe4 specified
%   configurations.
%
% Inputs:
%   TapWeightsIn: User defined floating point tap weight values.
%   ConfigSelect: 0-9: PCIe4 defined configuration (P0-P9).
%                 -1:  User defined configuration (from TapWeightsIn).
% Outputs:
%   TapWeightsOut: Array of tap weights to be used.
%
function TapWeightsOut = PCIe4FFEconfig(TapWeightsIn, ConfigSelect)

switch ConfigSelect
    case -1  % User defined tap weights
```

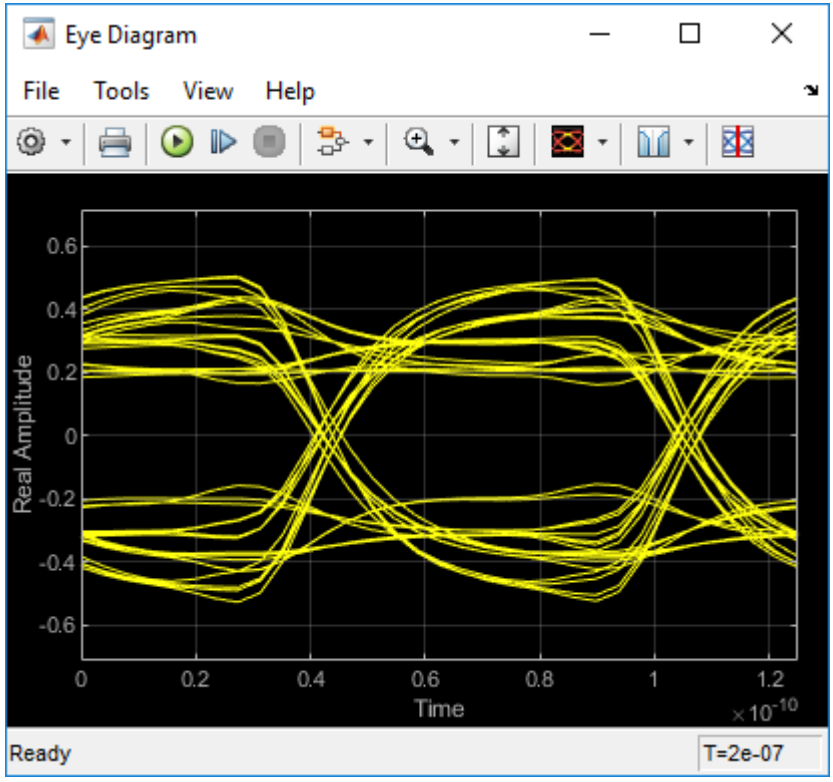
```
        TapWeightsOut = TapWeightsIn;
    case 0   % PCIe Configuration: P0
        TapWeightsOut = [0.000 0.750 -0.250];
    case 1   % PCIe Configuration: P1
        TapWeightsOut = [0.000 0.833 -0.167];
    case 2   % PCIe Configuration: P2
        TapWeightsOut = [0.000 0.800 -0.200];
    case 3   % PCIe Configuration: P3
        TapWeightsOut = [0.000 0.875 -0.125];
    case 4   % PCIe Configuration: P4
        TapWeightsOut = [0.000 1.000 0.000];
    case 5   % PCIe Configuration: P5
        TapWeightsOut = [-0.100 0.900 0.000];
    case 6   % PCIe Configuration: P6
        TapWeightsOut = [-0.125 0.875 0.000];
    case 7   % PCIe Configuration: P7
        TapWeightsOut = [-0.100 0.700 -0.200];
    case 8   % PCIe Configuration: P8
        TapWeightsOut = [-0.125 0.750 -0.125];
    case 9   % PCIe Configuration: P9
        TapWeightsOut = [-0.166 0.834 0.000];
    otherwise
        TapWeightsOut = TapWeightsIn;
end
```

Re-wire the FFE sub-system so that the FFETapWeights and FFEConfigSelect constant blocks connect to the inputs of the newly defined PCIe4FFEconfig MATLAB function block. The TapWeightsOut signal from the PCIe4FFEconfig block connects to the **TapWeights** port of the FFE block.

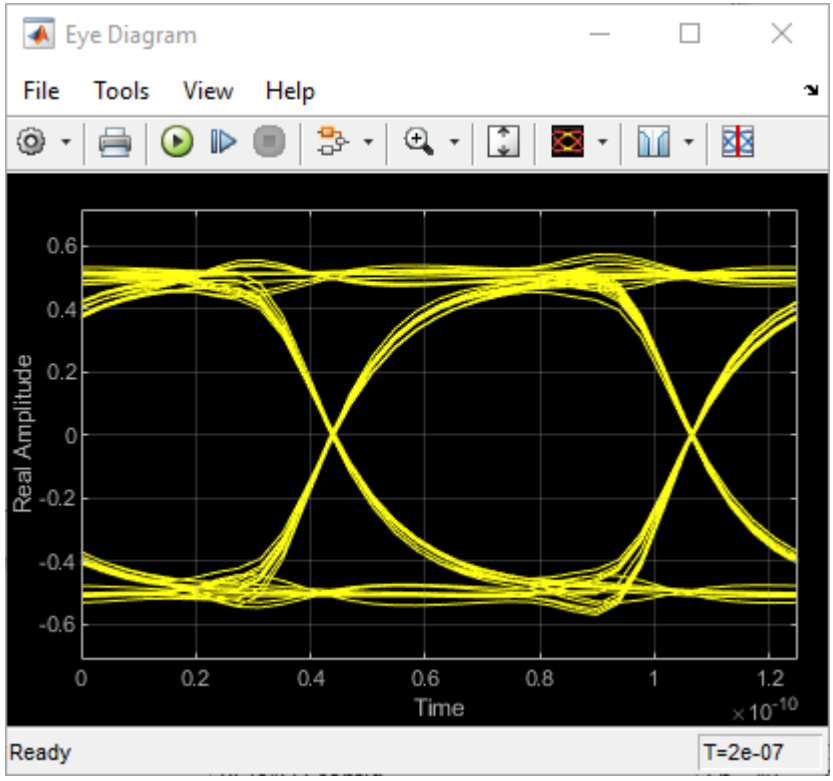


To test that the new FFE control parameter is working correctly, open the SerDes IBIS-AMI Manager dialog box from the Configuration block. In the **AMI-Tx** tab, edit the

ConfigSelect* parameter to set **Current value** to `P7`. This corresponds to PCIe Configuration P7: Pre = -0.100, Main = 0.700 and Post = -0.200. Observe the output waveform.



Next, set the **Current value** of the **ConfigSelect*** parameter to `User Defined`. This corresponds to user-defined tap weights: Pre = 0.000, Main = 1.000 and Post = 0.000. Observe how the output waveform changes.



Try different values of **ConfigSelect*** to verify proper operation. The time-domain eye opens and closes based on the amount of equalization applied by the FFE. How much the eye changes, and the tap values that create the most open eye varies based on the loss defined in the Analog Channel block.

Export the Tx IBIS-AMI Model

Verify that both Init and GetWave are behaving as expected, then generate the final IBIS-AMI compliant PCIe4 model executables, IBIS and AMI files.

Double-click the Configuration block to open the Block Parameters dialog box. Click the **Open SerDes IBIS-AMI Manager** button, then select the **Export** tab:

- Update the **Tx model name** to `pcie4_tx`.
- **Tx and Rx corner percentage** is set to `10`. This will scale the min/max analog model corner values by +/-10%.

- Verify that **Dual model** is selected as **Model Type** for the Tx. This will create model executables that support both statistical (Init) and time domain (GetWave) analysis.
- Set the Tx model **Bits to ignore** parameter to 3 since there are three taps in the Tx FFE.
- Set the **Models to export** to **Tx only**.
- Set the **IBIS file name (.ibs)** to `pcie4_tx_serdes.ibs`
- Click the **Export** button to generate models in the **Target directory**.

Test Generated IBIS-AMI Model

The PCIe4 transmitter IBIS-AMI model is now complete and ready to be tested in any industry standard AMI model simulator.

References

PCI-SIG.

3

Customize IBIS-AMI Models

Managing AMI Parameters

This example shows how to add, delete, modify, rename and hide AMI parameters in SerDes Toolbox. These parameters are then available to be used with the existing datapath blocks, user created MATLAB function blocks or optimization control loop. These parameters can be passed to or returned from the AMI model executables (DLLs) created by SerDes Toolbox.

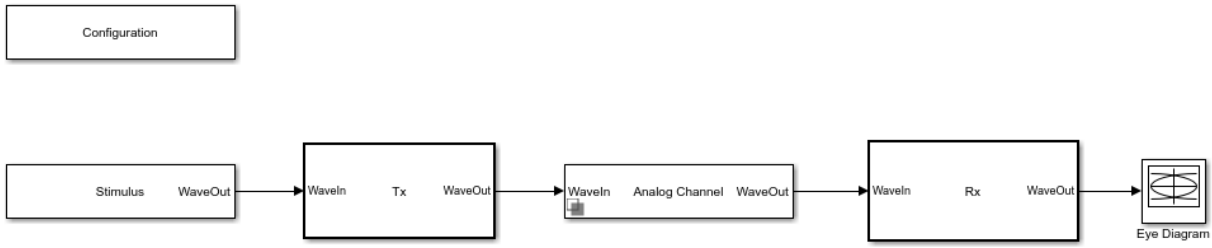
Example Setup

This example will be adding a new InOut Parameter 'Count' alongside the Pass-through datapath block. This parameter will count the number of passes through AMI_Init (which should be 1), then pass the result to AMI_GetWave where it will continue to count the total number of passes. While this may not be especially useful functionality for AMI model development, it will serve to demonstrate how new AMI parameters are added and used during model generation.

Inspect the Model

This example starts with a simple receiver model that only uses a pass-through block.

```
open_system('serdes_add_param.slx')
```



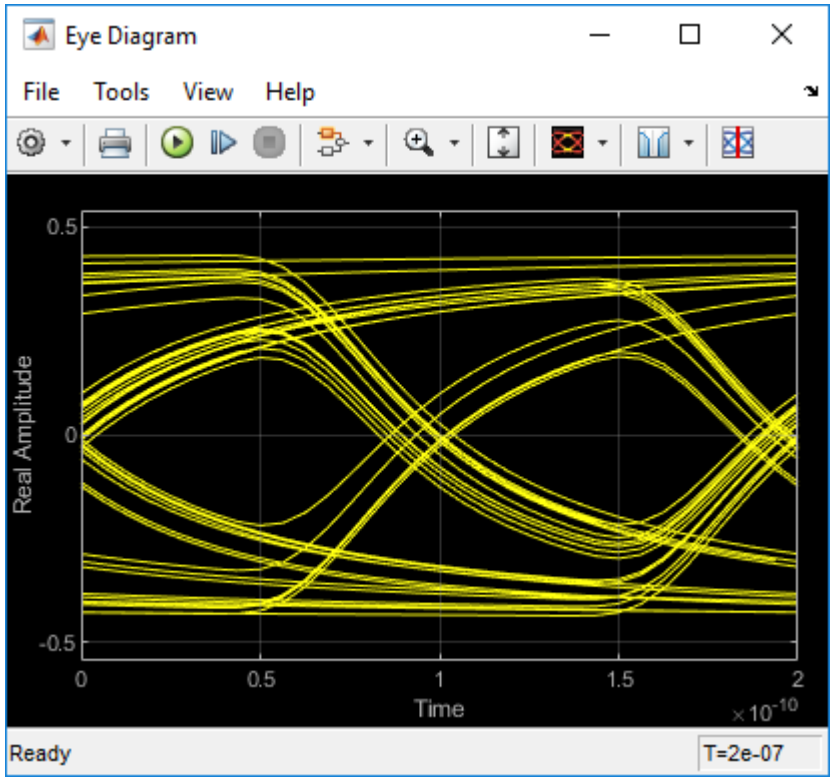
This Simulink SerDes System consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks.

- The Tx subsystem has the FFE datapath block to model the time domain portion of the AMI model and an Init block to model the statistical portion. The Tx subsystem will not be used in this example.
- The Analog Channel block has the parameter values for Target frequency, Loss, Impedance and Tx/Rx analog model parameters.

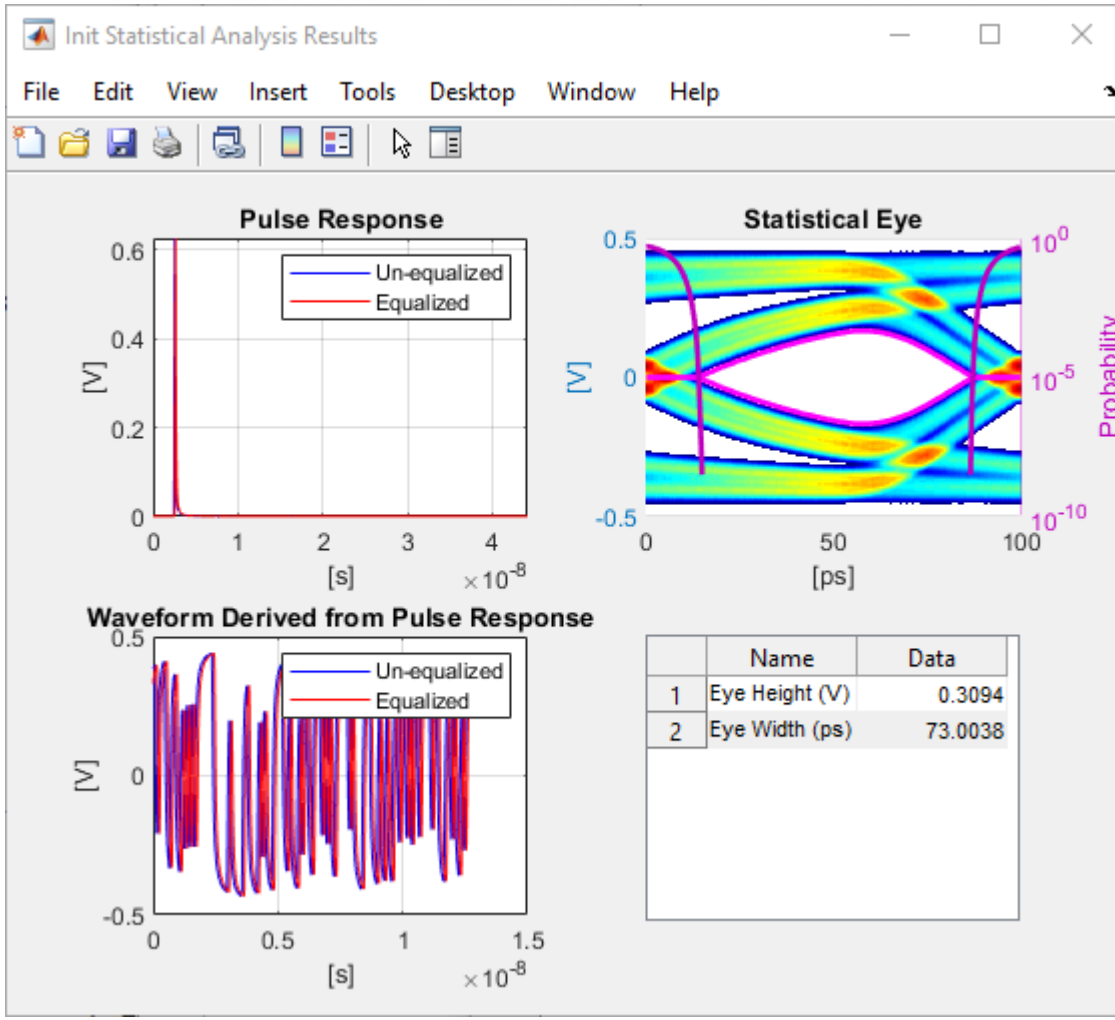
- The Rx subsystem has the Pass-Through datapath block and an Init block to model the statistical portion of the AMI model.

Run the Model

Run the model to verify that the base configuration is working as expected before editing. Two plots are generated. The first is a live time domain (GetWave) eye diagram that is updated as the model is running.



The second plot contains four views of the statistical (Init) results.



How to Add a new Parameter

Open the Block Parameter dialog box for the Configuration block, then click on the **Open SerDes IBIS-AMI Manager** button and select the **AMI-Rx** tab.

1. Highlight the **PT** datapath block and press **Add Parameter...**

2. Change the **Parameter Name** to: `Count`

3. Verify that the **Current value** is set to `0` (this will be the starting point for our count).

4. Add a **Description** of: `Starting value of iteration count.`

There are four possible values for **Usage**:

- `In`: These parameters are required inputs to the AMI Executable.
- `Out`: These parameters are output from the AMI_Init and/or AMI_GetWave functions.
- `InOut`: These parameters are required inputs to the AMI Executable and can also return values from AMI_Init and/or AMI_GetWave.
- `Info`: These parameters are information for the User and/or the simulation tool.

5. Set the **Usage** to: `InOut`

There are six possible parameter **Types**:

- `Float`: A floating point number.
- `Integer`: Integer numbers without a fractional or decimal component.
- `UI`: Unit Interval (the inverse of the data rate frequency).
- `Tap`: A floating point number for use by Tx FFE and Rx DFE delay lines.
- `Boolean`: True and False values, without quotation marks.
- `String`: A sequence of ASCII characters enclosed in quotation marks.

6. Set the **Type** to: `Integer`

There are three possible parameter **Formats**:

- `Value`: A single data value.
- `List`: A discrete set of values from which the user may select one value.
- `Range`: A continuous range for which the user may select any value between Min and Max.

7. Set the **Format** to: `Value`

8. Hit **OK** to create the new parameter, then close the SerDes IBIS-AMI Manager.

Accessing a new Parameter from the Initialize Function

New parameters are accessed from the Initialize function (for statistical analysis) through the impulseEqualization MATLAB function block. This example has added an InOut parameter. To use the new InOut Parameter 'Count' in AMI_Init:

1. Inside the Rx subsystem, double click on the Init block to open the mask.

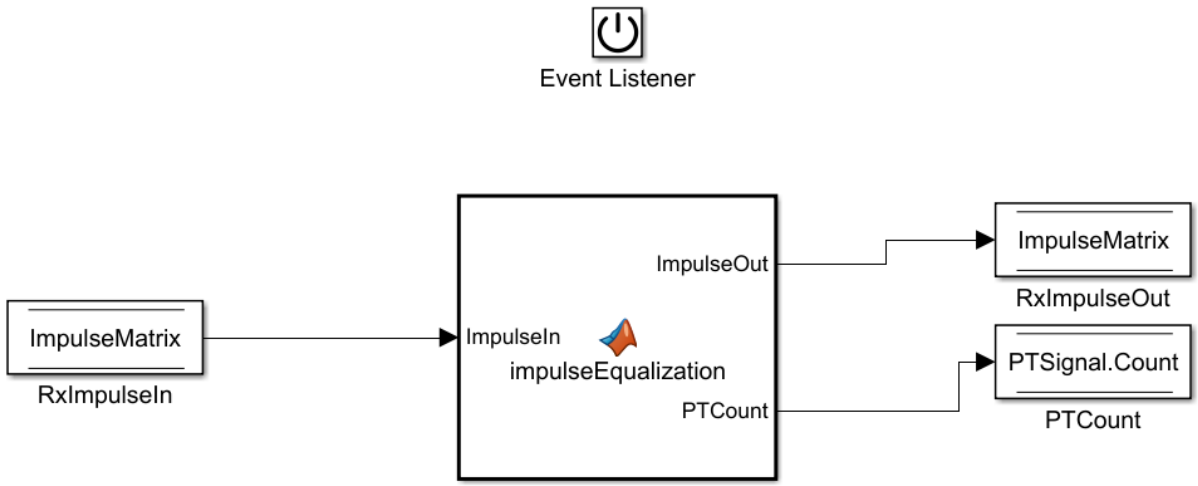
2. Press the **Refresh Init** button to propagate the new AMI parameter(s) to the initialize subsystem.

3. Click **OK** to close the mask.

4. Click on the Init block again and type **Ctrl-U** to look under the Init mask, then double-click on the initialize block to open the Initialize Function.

The impulseEqualization MATLAB function block is an automatically generated function which provides the impulse response processing of the SerDes system block (IBIS-AMI Init).

Note that the new Count parameter has been automatically added as an output of this MATLAB function as a Data Store Write block. No Data Store Read is required because the input parameters are passed in as a PTSignal Simulink.Parameter.



5. Double-click on the **impulseEqualization MATLAB function block** to open the function in MATLAB. The '%% BEGIN:' and '% END:' lines within this function block

denote the section where custom user code can be entered. Data in this section will not get over-written when Refresh Init is run:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)

% END: Custom user code area (retained when 'Refresh Init' button is pressed)
```

When **Refresh Init** was run, it added our new parameter to the Custom user code area so that it can be used as needed:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
PTCount = PTParameter.Count; % User added AMI parameter from SerDes IBIS-AMI Manager
% END: Custom user code area (retained when 'Refresh Init' button is pressed)
```

6. To add our custom code, scroll down to the Custom user code section, then enter `PTCount = PTCount + 1;` The Custom user code section should look like this:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
PTCount = PTParameter.Count; % User added AMI parameter from SerDes IBIS-AMI Manager
PTCount = PTCount + 1;  % Count each iteration through this function.
% END: Custom user code area (retained when 'Refresh Init' button is pressed)
```

7. Save the updated MATLAB function, then run the Simulink project to test the new code. Using the Simulation Data Inspector, verify that the value of Count after Init is now '1'.

Note that the final value for Count was written to the PTSignal data store so that it is now available in AMI_GetWave.

How Usage affects Parameters in Init

Depending on what Usage was selected, parameters show up in the Custom User code area of the impulseEqualization MATLAB function block in different ways:

Info Parameters

Info parameters are informational for the user or simulation tool and cannot be used by the model, therefore they will not show up in the Initialize code.

In Parameters

In parameters show up as a variable that can be used as needed. For example, an In parameter named 'InParam' that was added to the VGA block would show up as follows:

```
VGAParameter.InParam; % User added AMI parameter from SerDes IBIS-AMI Manager
```

Out Parameters

Out parameters show up as a variable with the value defined in the IBIS-AMI Manager. For example, an Out parameter named 'OutParam' that was added to the VGA block with a value of '2' would show up as follows:

```
VGAOutParam=2; % User added AMI parameter from SerDes IBIS-AMI Manager
```

In addition, a Data Store Write block named 'OutParam' is added to the Initialize Function for passing values out of Init to the user (via the AMI_Parameters_Out string) and for use in GetWave (if desired):

InOut Parameters

InOut parameters show up as a variable that can be used as needed. For example, an InOut parameter named 'InOutParam' that was added to the VGA block would show up as follows:

```
VGAInOutParam = VGAParameter.InOutParam; % User added AMI parameter from SerDes IBIS-AM
```

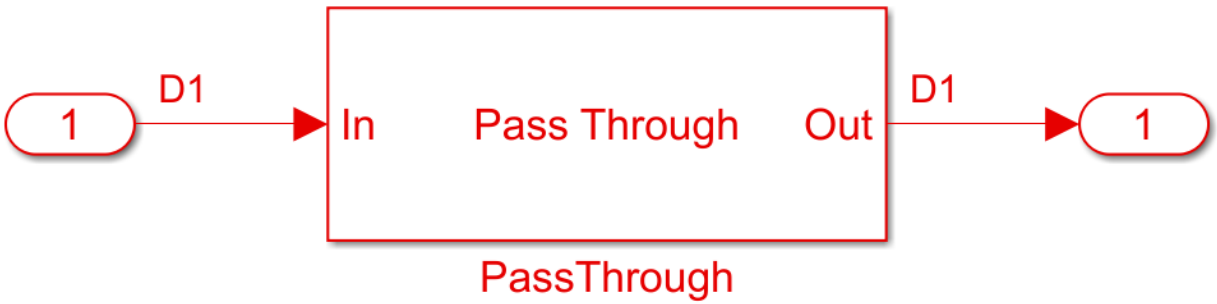
In addition, a Data Store Write block named 'OutParam' is added to the Initialize Function for passing values out of Init to the user (via the AMI_Parameters_Out string) and for use in GetWave (if desired):

Accessing a new Parameter from the GetWave Function

New parameters are accessed from the GetWave function (for time-domain analysis) by adding a Constant, Data Store Read or Data Store Write block to a datapath block. This

example has added an InOut parameter. To use the new InOut Parameter 'Count' in GetWave:

1. Inside the Rx subsystem, click on the Pass-Through datapath block and type **Ctrl-U** to look under the Pass-Through mask.



2. Add a Simulink/Signal Routing **Data Store Read** block to the canvas

- Name the Data Store Read block: `PTCount_Read`
- Double-click on the Data Store Read block and change the Data store name to: `PTSignal`
- On the Element Selection tab, in the Specify element(s) to select box type: `PTSignal.Count` and press the **Select>>** button to select the Count element.
- Resize the block to make all names and element properties visible.
- Click **OK** to close the dialog.

3. Add a Simulink/Math Operations **Sum** block to the canvas.

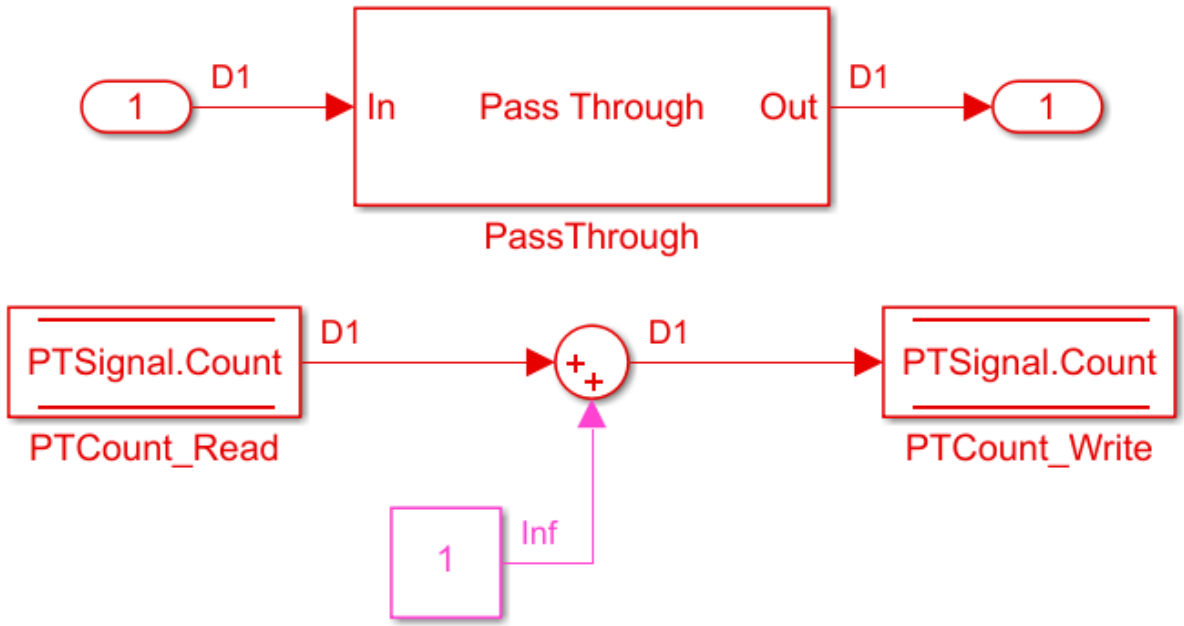
4. Add a Simulink/Sources **Constant** block to the canvas and set the value to `1`.

5. Add a Simulink/Signal Routing **Data Store Write** block to the canvas.

- Name the Data Store Write block: `PTCount_Write`
- Double-click on the Data Store Write block and change the Data store name to: `PTSignal`
- On the Element Selection tab, in the Specify element(s) to select box type: `PTSignal.Count` and press the **Select>>** button to select the Count element.
- Resize the block to make all names and element properties visible.

- Click **OK** to close the dialog.

6. Wire up each of the elements so that the Pass Through block now looks like the following:



7. Save, then run the Simulink project to test the new code.

By adding Value Labels to the output port of the Sum block, see that the value of Count after GetWave is 3.2e+04 (Samples Per Symbol * Number of symbols). After generating AMI model executables, the value of Count will be available in the Parameters out string in an AMI simulator.

How Usage affects Parameters in GetWave

New parameters are accessed from the GetWave function in different ways, depending on what Usage was selected.

Info Parameters

Info parameters are informational for the user or simulation tool and cannot be used by the model.

In Parameters

In parameter are used by adding a Constant block.

For more information, see “Customizing SerDes Toolbox Datapath Control Signals” on page 2-2.

Out Parameters

Out parameters are used by adding a Data StoreWrite block, which is covered as a sub-set of the InOut parameters below.

InOut Parameters

InOut parameters are used by adding a Data Store Read block and a Data Store Write block. The Data Store blocks also allow for passing the parameter values out of AMI_Init into AMI_GetWave.

How to Rename a Parameter

The parameters used by the SerDes Toolbox built-in System Objects can be modified or hidden but cannot be renamed.

User generated AMI parameters are renamed as follows.

Update the AMI Parameters

1. Open the Block Parameter dialog box for the Configuration block, then click on the **Open SerDes IBIS-AMI Manager** button.

2. Go to either the **AMI-Tx** or **AMI-Rx** tab where the parameter resides.

3. Highlight the parameter to be renamed and press **Edit Parameter...**

4. In the Parameter name field, changed the name as desired.

5. Hit **OK**, then **Close** the SerDes IBIS-AMI Manager

Update Init

1. Push into either the Tx or Rx subsystem block where the parameter is used.

2. Double click on the Init block to open the mask.

3. Press the **Refresh Init** button to propagate the AMI parameter name change to the initialize subsystem.

4. Click **OK** to close the mask.

5. Click on the Init block again and type **Ctrl-U** to look under the Init mask, then double-click on the initialize block to open the Initialize Function.

6. Double-click on the **impulseEqualization MATLAB function block** to open the function in MATLAB.

7. Scroll down to the section titled:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
```

8. Rename all instances of the parameter.

9. Save and close the MATLAB function block.

Update GetWave

Push into each datapath block where the renamed parameter was used and rename each instance of the parameter.

Verify Results

Run a simulation to verify that the project still operates with no errors or warnings.

How to Delete a Parameter

The parameters used by the SerDes Toolbox built-in System Objects can be modified or hidden but cannot be deleted.

User generated AMI parameters are deleted as follows.

Update the AMI Parameters

1. Open the Block Parameter dialog box for the Configuration block, then click on the **Open SerDes IBIS-AMI Manager** button.

2. Go to either the **AMI-Tx** or **AMI-Rx** tab where the parameter resides.

3. Highlight the parameter to be deleted and press **Delete Parameter**.

4. Confirm the deletion, then **Close** the SerDes IBIS-AMI Manager.

Update Init

1. Push into either the Tx or Rx subsystem block where the parameter was used.

2. Double click on the Init block to open the mask.

3. Press the **Refresh Init** button to remove any deleted Out or InOut parameter Data Stores from the initialize subsystem.

4. Click **OK** to close the mask.

5. Click on the Init block again and type **Ctrl-U** to look under the Init mask

6. Double-click on the initialize block to open the Initialize Function.

7. Double-click on the **impulseEqualization MATLAB function block** to open the function in MATLAB.

8. Scroll down to the section titled:

```
%% BEGIN: Custom user code area (retained when 'Refresh Init' button is pressed)
```

9. Delete or comment out all instances of the removed parameter.

10. Save and close the MATLAB function block.

Update GetWave

Push into each datapath block where the removed parameter was used and delete each instance of the parameter.

Verify Results

Run a simulation to verify that the project still operates with no errors or warnings.

How to Hide a Parameter

There may be times when a parameter is required for model functionality, but needs to be hidden from the user. For example, to keep a user from changing the FFE mode, remove this parameter from .ami file - effectively hardcoding the parameter to a single value. The

mode parameter is still present in the code so that the FFE continues to work as expected, but the user can no longer change the value.

To hide a parameter from both Init and GetWave:

1. Open the mask by double-clicking on the datapath block of interest.

2. Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.

3. Deselect the parameter(s) to be hidden.

A few things to keep in mind about hiding parameters:

- When hiding parameters, verify that the current parameter value(s) are correct. The current value will now always be used as the default value for that parameter.
- Hiding a parameter has no effect on the model executable. It only removes the parameter from the generated .ami file.
- If the hidden parameter is of type Out or InOut, it will still show up in the AMI_Parameters_Out string of the model executable.

How to Modify a Parameter

All the parameters used in SerDes Toolbox are modified via the SerDes IBIS-AMI Manager dialog by using the **Edit Parameter...** button. However, the parameter values that can be modified vary depending on which type of parameters they are.

For the built-in System Objects, only the following fields can be modified:

- Current Value
- Description
- Format
- Default
- List values (for Format List)
- Typ/Min/Max values (for Format Range)

For the user defined parameters all fields can be modified.

How to add Jitter Parameters

Jitter and noise parameters such as Tx_Rj, Tx_Dj, Tx_DCD, Rx_Rj, Rx_Dj and Rx_DCD or other reserved parameters such Rx_Receiver_Sensitivity are post-processing parameters

that are used by an IBIS-AMI compliant simulator to modify the simulation results accordingly. These parameters must be added manually by editing the .ami files in a text editor.

For example, to add Rx_Receiver_Sensitivity, Rx_DCD, Rx_Dj and Rx_Sj to a receiver .ami file, add the following lines to the Reserved_Parameters section of the .ami file:

```
(Rx_Receiver_Sensitivity (Usage Info)(Type Float)(Value 0.04))

(Rx_DCD (Usage Info) (Type UI) (Value 0.0125))

(Rx_Dj (Usage Info) (Type UI) (Value 0.0175))

(Rx_Rj (Usage Info) (Type UI) (Value 0.00375))
```

Remember that any manual edits to the .ami files will be over-written during model Export, so be careful to save any manually edited files in a separate location or turn off the creation of the AMI file(s) during the Export process.

For more information on IBIS reserved parameters see the IBIS specification.

References

IBIS 6.1 Specification

Industry Standard IBIS-AMI Models

PCIe4 Transmitter/Receiver IBIS-AMI Model

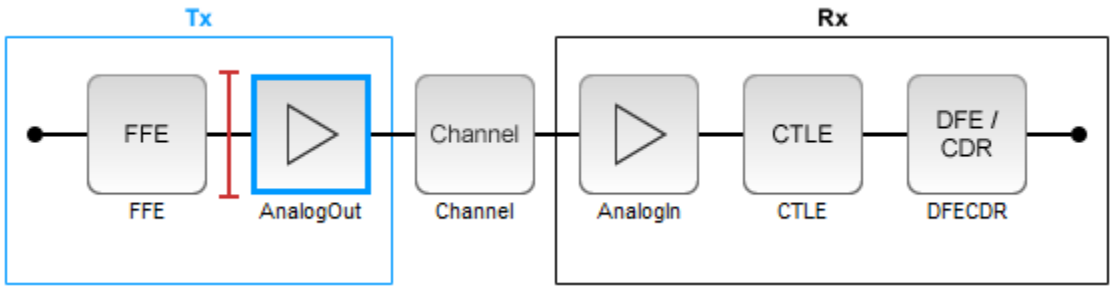
This example shows how to create generic PCIe Generation 4 (PCIe4) transmitter and receiver IBIS-AMI models using the library blocks in SerDes Toolbox™. The generated models conform to the IBIS-AMI and PCI-SIG PCIe4 specifications.

PCIe4 Tx/Rx IBIS-AMI Model Setup in SerDes Designer App

The first part of this example sets up the target transmitter and receiver AMI model architecture using the blocks required for PCIe4 in the SerDes Designer app. The model is then exported to Simulink® for further customization.

This example uses the SerDes Designer model `pcie4_txrx_ami`. Type the following command in the MATLAB® command window to open the model:

```
>> serdesDesigner('pcie4_txrx_ami')
```



A PCIe4 compliant transmitter uses a 3-tap feed forward equalizer (FFE) with one pre-tap and one post-tap, and ten presets. The receiver model uses a continuous time linear equalizer (CTLE) with seven pre-defined settings, and a 2-tap decision feedback equalizer (DFE). To support this configuration the SerDes System is set up as follows:

Configuration Setup

- **Symbol Time** is set to `62.5` ps, since the maximum allowable PCIe4 operating frequency is `16` GHz
- **Target BER** is set to `1e-12`.
- **Samples per Symbol**, **Modulation**, and **Signaling** are kept at default values, which are respectively `16`, `NRZ` (non-return to zero), and `Differential`.

Transmitter Model Setup

- The Tx FFE block is set up for one pre-tap and one post-tap by including three tap weights. Specific tap presets will be added in later in the example when the model is exported to Simulink.
- The Tx AnalogOut model is set up so that **Voltage** is `1.05` V, **Rise time** is `12` ps, **R** (output resistance) is `50` Ohms, and **C** (capacitance) is `0.25` pF according to the PCIe4 specification.

Channel Model Setup

- **Channel loss** is set to `23` dB according to the PCIe4 specification 16.0 GT/s root port long calibration channel insertion loss limit.
- **Target Frequency** is set to the Nyquist frequency, `8` GHz.
- **Differential impedance** is kept at default `100` Ohms.

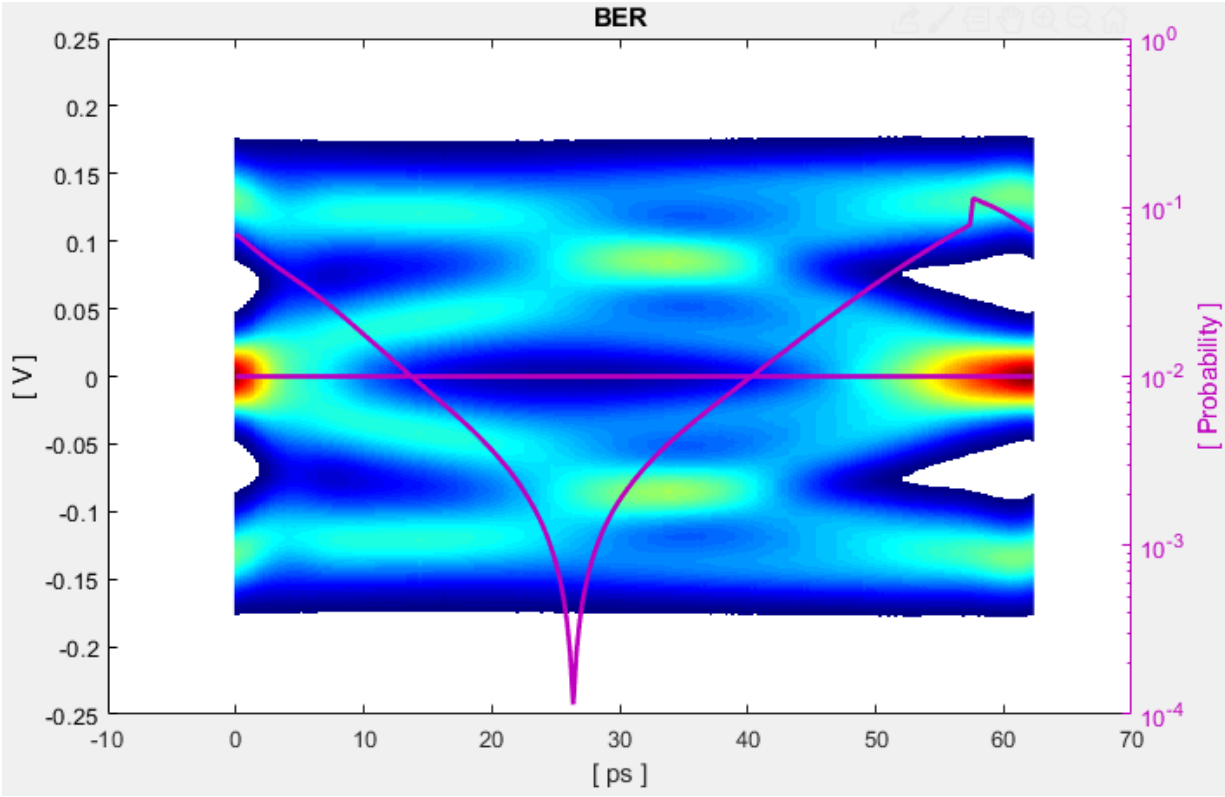
Receiver Model Setup

- The Rx Analogin model is set up so that **R** (input resistance) is `50` Ohms and **C** (capacitance) is `0.25` pF according to the PCIe4 specification.
- The Rx CTLE block is set up for 7 configurations. The **GPZ** (Gain Pole Zero) matrix data is derived from the transfer function given in the PCIe4 Behavioral CTLE specification.
- The Rx DFE/CDR block is set up for two DFE taps. The limits for each tap have been individually defined according to the PCIe4 specification to `+/-30` mV for tap1 and `+/-20` mV for tap2.

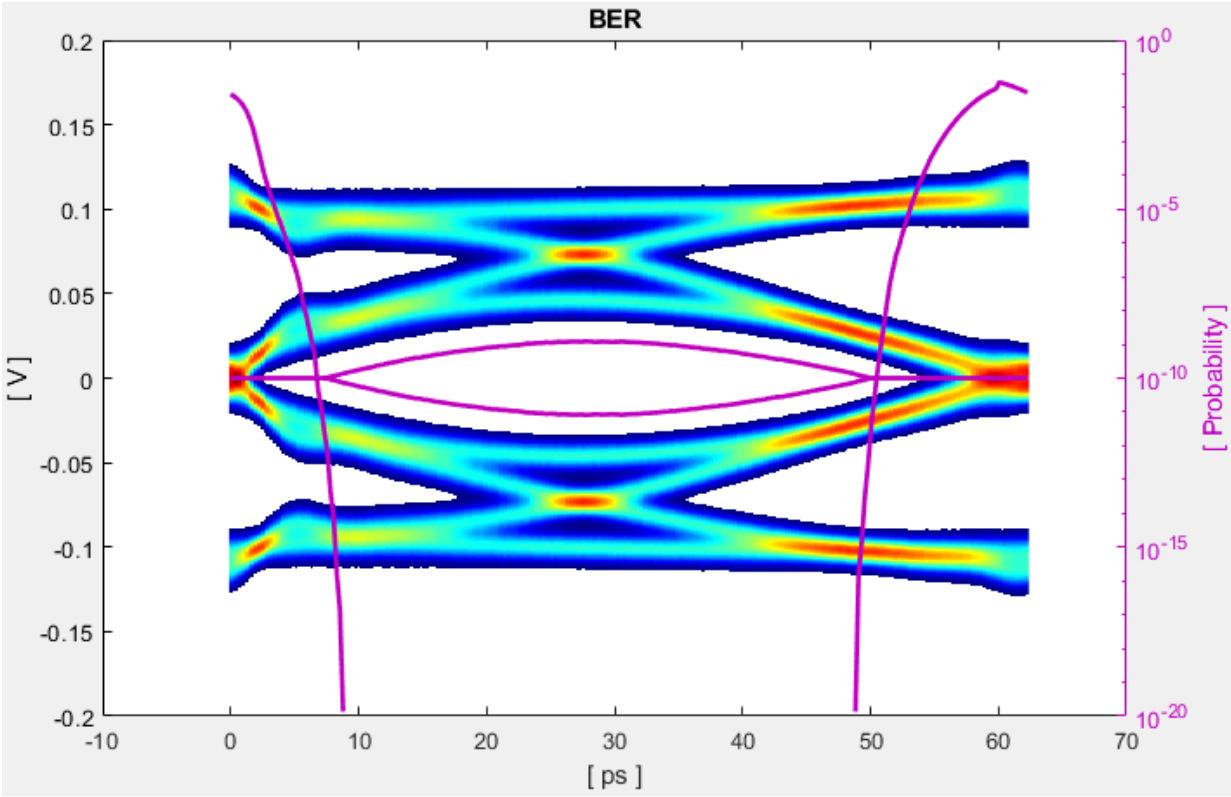
Plot Statistical Results

Use the SerDes Designer plots to visualize the results of the PCIe4 setup.

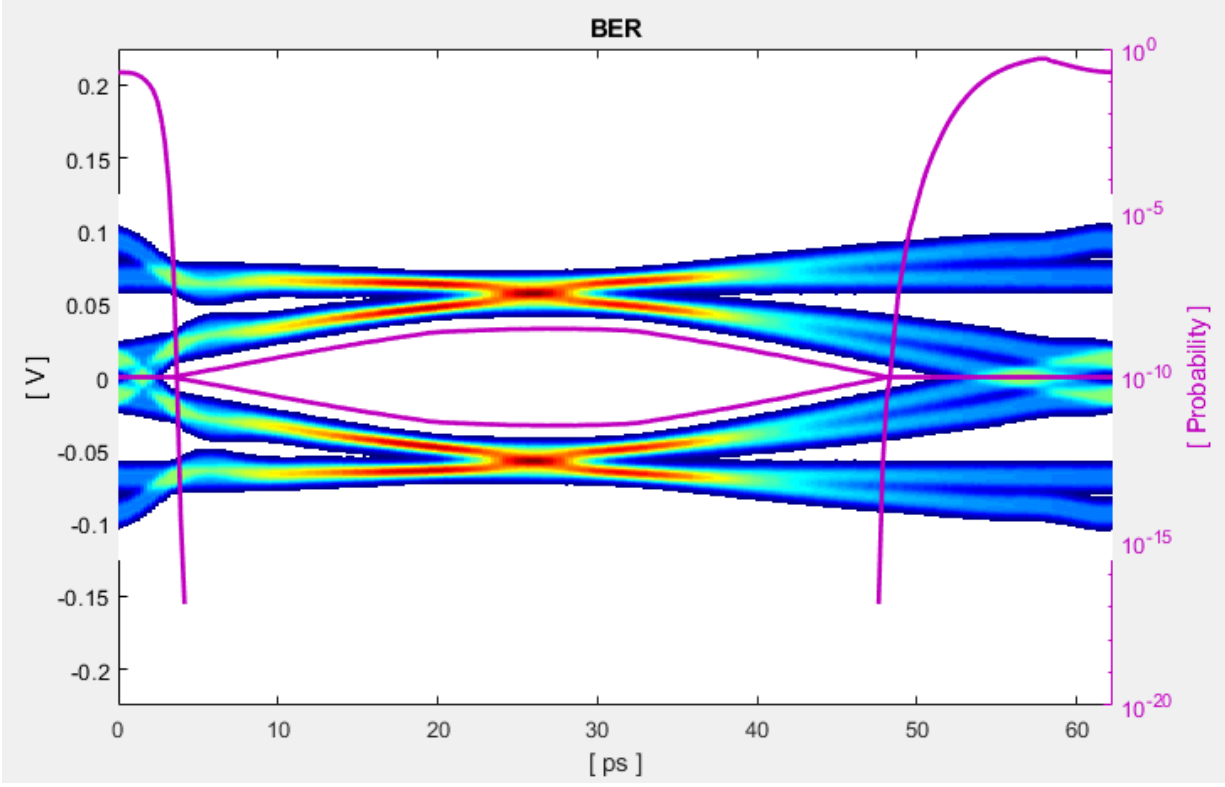
Add the **BER** plot from **ADD Plots** and observe the results.



Change the Rx CTLE **Configuration select** parameter value from `0` to `4` and observe how this changes the data eye.



Change the value of the Tx FFE **Tap weights** from `[0 1 0]` to `[-0.1 1 -0.1]` and observe the results.



Before continuing, reset the value of the Tx FFE **TapWeights** back to `[0 1 0]` and Rx CTLE **ConfigSelect** back to `0`. Resetting these values here will avoid the need to set them again after the model has been exported to Simulink.

Export SerDes System to Simulink

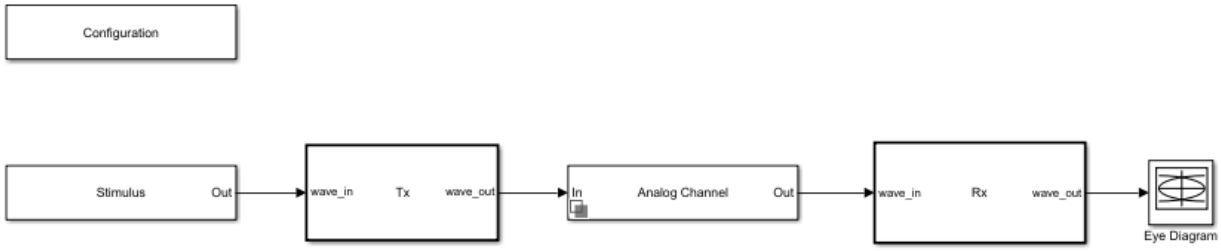
Click on the **Export** button to export the above configuration to Simulink for further customization and generation of the AMI model executables.

PCIe4 Tx/Rx IBIS-AMI Model Setup in Simulink

The second part of this example takes the SerDes system exported by the SerDes Designer app and customize it as required for PCIe4 in Simulink.

Review Simulink Model Setup

The SerDes System imported into Simulink consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks. All the settings from the SerDes Designer app have been transferred to the Simulink model. Save the model and review each block setup.

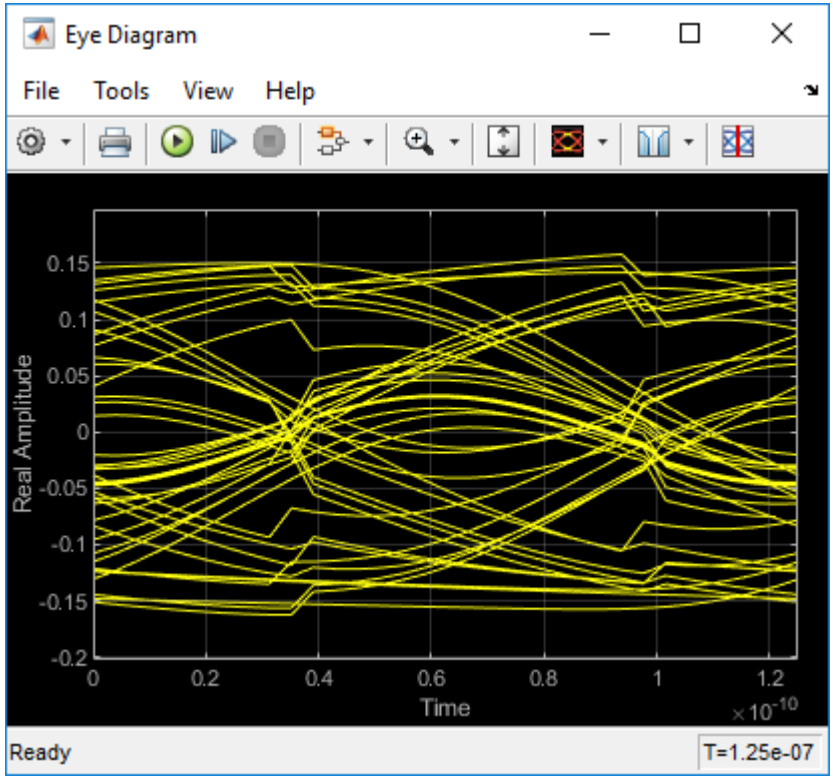


- Double click the Configuration block to open the Block Parameters dialog box. The parameter values for **Symbol time**, **Samples per symbol**, **Target BER**, **Modulation** and **Signaling** is carried over from the SerDes Designer app.
- Double click the Stimulus block to open the Block Parameters dialog box. You can set the **PRBS** (pseudorandom binary sequence) order and the number of symbols to simulate. This block is not carried over from the SerDes Designer app.
- Double click the Tx block to look inside the Tx subsystem. The subsystem has the FFE block carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.
- Double click the Analog Channel block to open the Block Parameters dialog box. The parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx analog model parameters is carried over from the SerDes Designer app.
- Double click on the Rx block to look inside the Rx subsystem. The subsystem has the CTLE and DFECDR blocks carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.

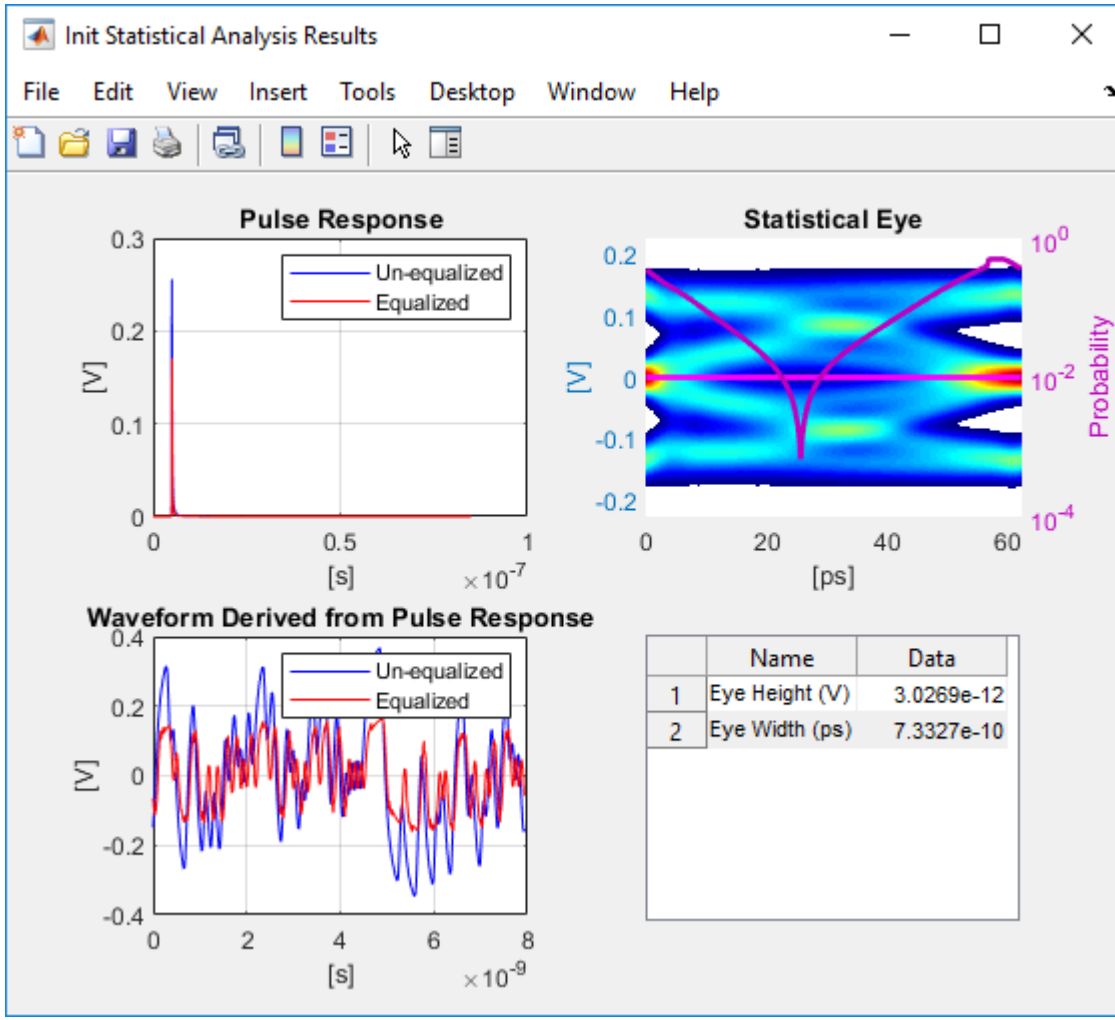
Run the Model

Run the model to simulate the SerDes System.

Two plots are generated. The first is a live time-domain (GetWave) eye diagram that is updated as the model is running.



The second plot contains four views of the statistical (Init) results, similar to what is available in the SerDes Designer App.



Update Tx FFE Block

- Inside the Tx subsystem, double click the FFE block to open the FFE Block Parameters dialog box.

- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Deselect the **Mode** parameter to remove this parameter from the AMI file, effectively hard-coding the current value of **Mode** in final AMI model to `Fixed`.

Review Rx CTLE Block

- Inside the Rx subsystem, double click the CTLE block to open the CTLE Block Parameters dialog box.
- **Gain pole zero** data is carried over from the SerDes Designer app. This data is derived from the transfer function given in the PCIE4 Behavioral CTLE specification.
- CTLE **Mode** is set to `Fixed`, which means an optimization algorithm built into the CTLE system object selects the optimal CTLE configuration at run time.

Update Rx DFECDR Block

- Inside the Rx subsystem, double click the DFECDR block to open the DFECDR Block Parameters dialog box.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Clear the **Phase offset** and **Reference offset** parameters to remove these parameters from the AMI file, effectively hard-coding these parameters to their current values.

Generate PCIe4 Tx/Rx IBIS-AMI Model

The final part of this example takes the customized Simulink model, modifies the AMI parameters for PCIe4, then generates IBIS-AMI compliant PCIe4 model executables, IBIS and AMI files.

Open the Block Parameter dialog box for the Configuration block and click on the **Open SerDes IBIS/AMI Manager** button. In the **IBIS** tab inside the SerDes IBIS/AMI manager dialog box, the analog model values are converted to standard IBIS parameters that can be used by any industry standard simulator. In the **AMI-Rx** tab in the SerDes IBIS/AMI manager dialog box, the reserved parameters are listed first followed by the model specific parameters following the format of a typical AMI file.

Update Transmitter AMI Parameters

Open the **AMI-Tx** tab in the SerDes IBIS/AMI manager dialog box. Following the format of a typical AMI file, the reserved parameters are listed first followed by the model specific parameters.

Inside the **Model_Specific** parameters, you can set the TX FFE tap values in three different ways:

- Leave the Tx FFE tap values at their default configuration - the user enters any floating point value for the pre/main/post taps values.
- Create a new AMI parameter to automatically select preset values - see the example (add link).
- Directly specify the ten preset coefficients as defined in the PCIe4 specification - shown below in this example.

When you directly specify the preset coefficients, you change the format of the three **TapWeights** and specify the exact value to use for each preset. Only these ten defined presets will be allowed, and all three taps must be set to the same preset to get the correct values.

Set Preshoot Tap

- Select **TapWeight -1**, then click the **Edit Parameter...** button to bring up the Add/Edit AMI Parameter dialog.
- Set the **Current Value** to `0.000`.
- Change the **Description** to `Preshoot tap value`.
- Change the **Format** from `Range` to `List`.
- Change the **Default value** to `0.000`.
- In the **List values** box enter: `[0.000 0.000 0.000 0.000 0.000 -0.100 -0.125 -0.100 -0.125 -0.166]`.
- In the **List_Tip values** box enter: `["P0" "P1" "P2" "P3" "P4" "P5" "P6" "P7" "P8" "P9"]`.
- Click **OK** to save the changes.

Set Main Tap

- Select **TapWeight 0**, then click the **Edit Parameter...** button to bring up the Add/Edit AMI Parameter dialog.
- Set the **Current Value** to `0.750`.
- Change the **Description** to `Main tap value`.
- Change the **Format** from `Range` to `List`.
- Change the **Default value** to `0.750`.

- In the **List values** box enter: `[0.750 0.833 0.800 0.875 1.000 0.900 0.875 0.700 0.750 0.834]`.
- In the **List_Tip values** box enter: `["P0" "P1" "P2" "P3" "P4" "P5" "P6" "P7" "P8" "P9"]`.
- Click **OK** to save the changes.

Set De-emphasis Tap

- Select **TapWeight 1**, then click the **Edit Parameter...** button to bring up the Add/Edit AMI Parameter dialog.
- Set the **Current Value** to `-0.250`.
- Change the **Description** to: `De-Emphasis tap value`.
- Change the **Format** from `Range` to `List`.
- Change the **Default value** to `-0.250`.
- In the **List values** box enter: `[-0.250 -0.167 -0.200 -0.125 0.000 0.000 0.000 -0.200 -0.125 0.000]`.
- In the **List_Tip values** box enter: `["P0" "P1" "P2" "P3" "P4" "P5" "P6" "P7" "P8" "P9"]`.
- Click **OK** to save the changes.

Export Models

Open the **Export** tab in the SerDes IBIS/AMI manager dialog box.

- Update the **Tx model name** to `pcie4_tx`.
- Update the **Rx model name** to `pcie4_rx`.
- Note that the **Tx and Rx corner percentage** is set to `10`. This will scale the min/max analog model corner values by +/-10%.
- Verify that **Dual model** is selected for both the Tx and the Rx AMI Model Settings. This will create model executables that support both statistical (Init) and time domain (GetWave) analysis.
- Set the **Tx model Bits to ignore value** to `3` since there are three taps in the Tx FFE.
- Set the **Rx model Bits to ignore value** to `20,000` to allow sufficient time for the Rx DFE taps to settle during time domain simulations.
- Set **Models to export** as **Both Tx and Rx** so that all the files are selected to be generated (IBIS file, AMI files and DLL files).

- Set the **IBIS file name** to be `pcie4_serdes`.
- Press the **Export** button to generate models in the Target directory.

Add Jitter to AMI files

- Browse to the Target directory specified in the SerDes IBIS/AMI Manager and open the two .ami files in a text editor.
- In the Tx AMI file, add the following Reserved Parameters:

```
(Tx_Rj (Usage Info) (Type Float) (Range 0 0 2e-12))

(Tx_Dj (Usage Info) (Type Float) (Range 0 0 3e-11))

(Tx_DCD (Usage Info) (Type Float) (Range 0 0 3e-11))
```

These ranges allow the user to fine-tune the jitter values to meet PCIe4 jitter mask requirements.

- In the Rx AMI file, add the following Reserved Parameters:

```
(Rx_Rj (Usage Info) (Type Float) (Range 0 0 1e-11))

(Rx_Dj (Usage Info) (Type Float) (Range 0 0 3e-11))

(Rx_DCD (Usage Info) (Type Float) (Range 0 0 3e-11))
```

These ranges allow the user to fine-tune the jitter values to meet PCIe4 jitter mask requirements.

- Save and close both files.

Test Generated IBIS-AMI Models

The PCIe4 transmitter and receiver IBIS-AMI models are now complete and ready to be tested in any industry standard AMI model simulator.

References

1 PCI-SIG, https://pcisig.com.

DDR5 SDRAM Transmitter/Receiver IBIS-AMI Model

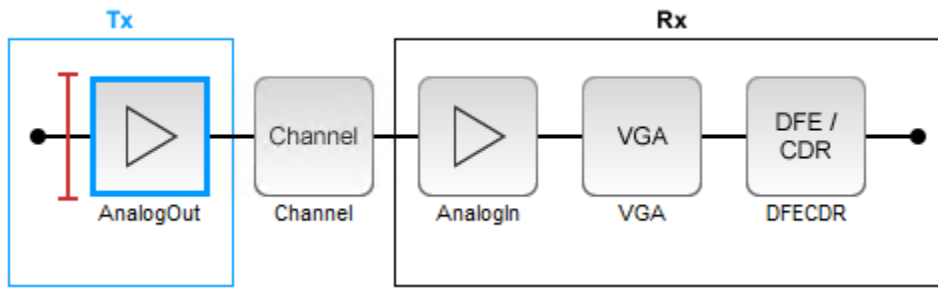
This example shows how to create generic DDR5 transmitter and receiver IBIS-AMI models using the library blocks in SerDes Toolbox™. Since DDR5 DQ signals are bidirectional, this example creates Tx and Rx models for the SDRAM. The generated models conform to the IBIS-AMI specification.

DDR5 SDRAM Tx/Rx IBIS-AMI Model Setup in SerDes Designer App

The first part of this example sets up and explores the target transmitter and receiver architectures using the blocks required for DDR5 in the SerDes Designer app. The SerDes system is then exported to Simulink® for further customization and IBIS-AMI model generation.

Type the following command in the MATLAB® command window to open the `ddr5_sdram` model:

```
>> serdesDesigner('ddr5_sdram')
```



The SDRAM has a DDR5 transmitter (Tx) using no equalization. The SDRAM also has a DDR5 receiver (Rx) using a variable gain amplifier (VGA) with 7 pre-defined settings and a 4-tap decision feedback equalizer (DFE) with built-in clock data recovery.

Configuration Setup

- **Symbol Time** is set to `208.3` ps, since the target operating rate is `4.8`Gbps for DDR5-4800.
- **Target BER** is set to `100e-18`.
- **Signaling** is set to `Single-ended`.

- **Samples per Symbol** and **Modulation** are kept at default values, which are `16` and `NRZ` (nonreturn to zero), respectively.

Transmitter Model Setup

- The DDR5 SDRAM has no transmit equalization, so only an analog model is required.
- The Tx AnalogOut model is set up so that **Voltage** is `1.1` V, **Rise time** is `100` ps, **R** (output resistance) is `48 o`hms, and **C** (capacitance) is `0.65` pF. The actual analog models used in the final model will be generated later in this example.

Channel Model Setup

- **Channel loss** is set to `5` dB, which is typical of DDR channels.
- **Single-ended impedance** is set to `40` ohms.
- **Target Frequency** is set to `2.4` GHz, which is the Nyquist frequency for 4.8 GHz

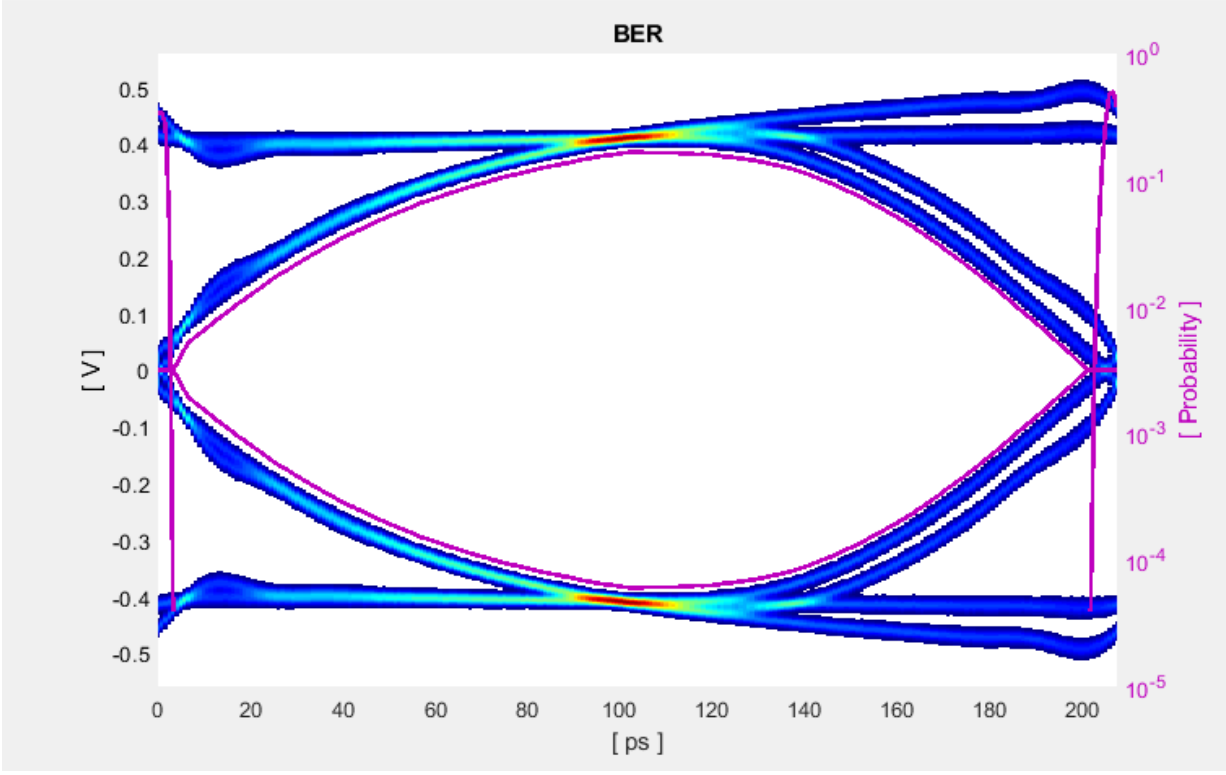
Receiver Model Setup

- The Rx AnalogIn model is set up so that **R** (input resistance) is `40` ohms and **C** (capacitance) is `0.65`pF. The actual analog models used in the final model will be generated later in this example.
- The VGA block is set up with a **Gain** of 1 and the **Mode** set to on. Specific VGA presets will be added later in this example after the model is exported to Simulink.
- The DFECDR block is set up for four DFE taps by including four **Initial tap weights** set to 0. The **Minimum tap value** is set to `[-0.2 -0.075 -0.06 -0.045]` V, and the **Maximum tap value** is set to `[0.05 0.075 0.06 0.045]` V.

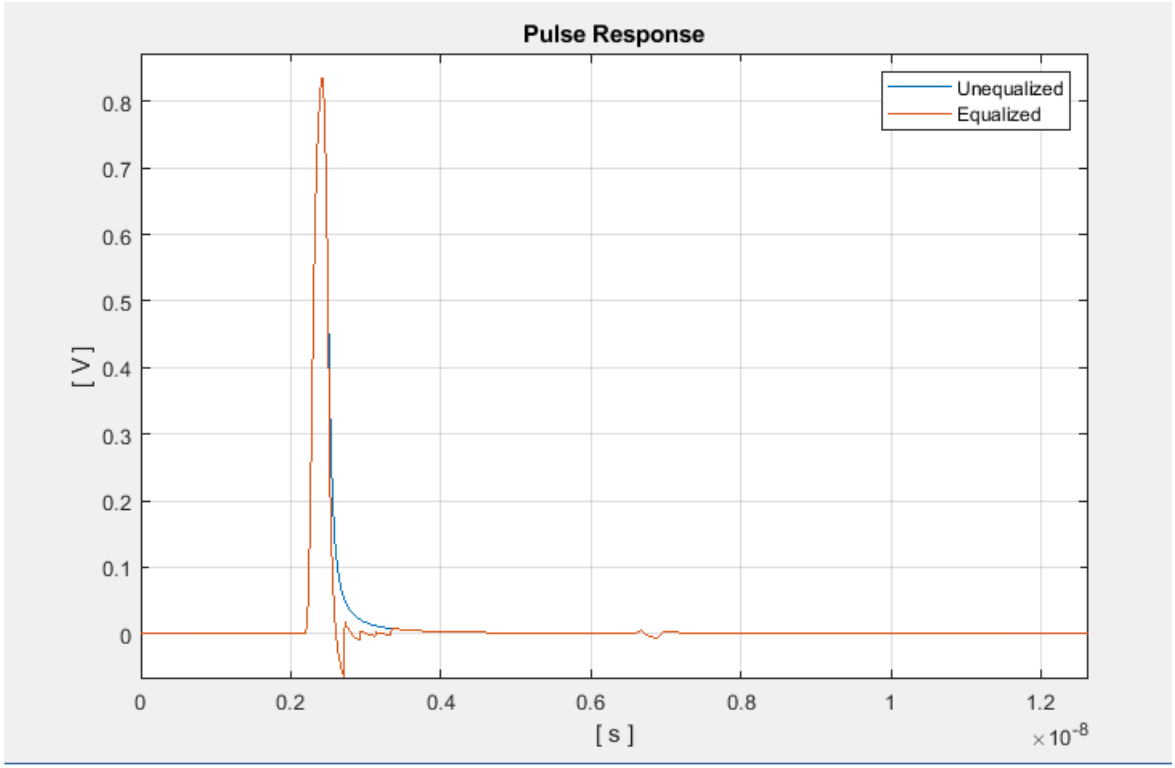
Plot Statistical Results

Use the SerDes Designer **Add Plots** button to visualize the results of the DDR5 SDRAM setup.

- Add the BER plot from **Add Plots** and observe the results.



- Add the Pulse Response plot from **Add Plots** and zoom into the pulse area to observe the results.



Export SerDes System to Simulink

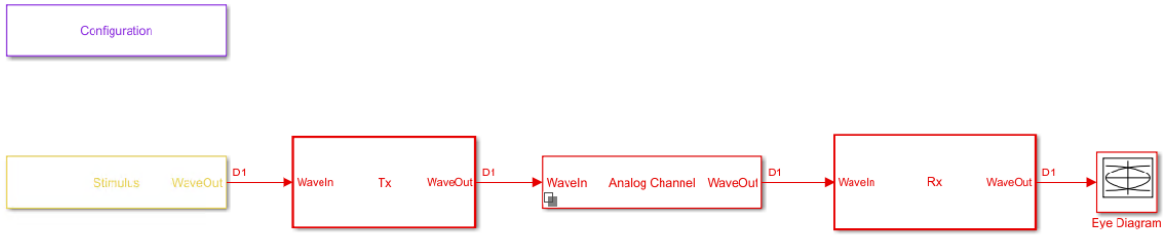
Click **Save** and then click on the **Export** button to export the configuration to Simulink for further customization and generation of the AMI model executables.

DDR5 SDRAM Tx/Rx IBIS-AMI Model Setup in Simulink

The second part of this example takes the SerDes system exported by the SerDes Designer app and customizes it as required for DDR5 in Simulink.

Review the Simulink Model Setup

The SerDes System imported into Simulink consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks. All the settings from the SerDes Designer app have been transferred to the Simulink model. Save the model and review each block setup.

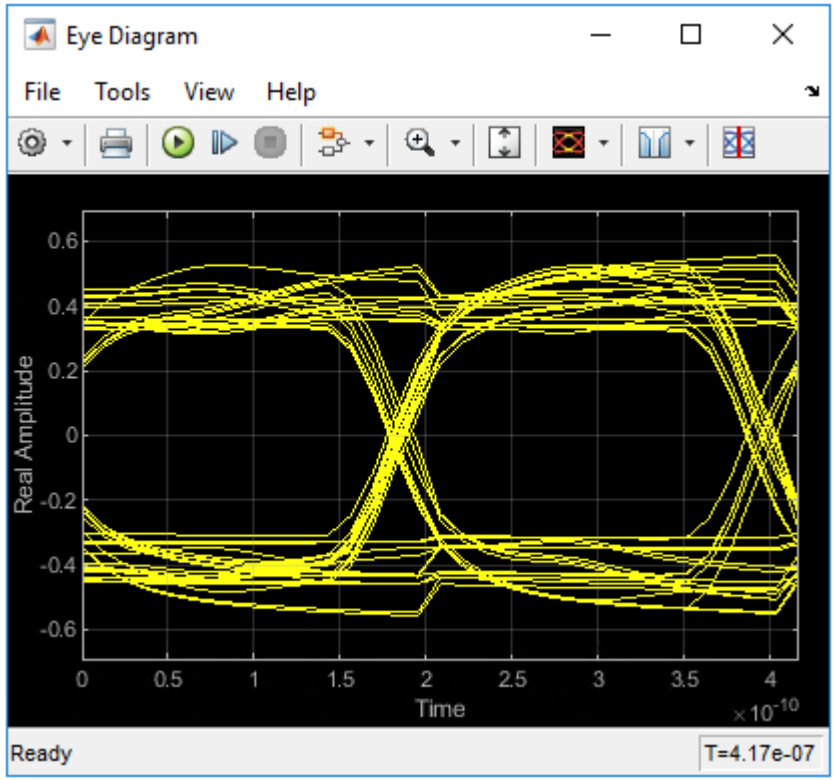


- Double-click the Configuration block to open the Block Parameters dialog box. The parameter values for **Symbol time**, **Samples per symbol**, **Target BER**, **Modulation** and **Signaling** are carried over from the SerDes Designer app.
- Double-click the Stimulus block to open the Block Parameters dialog box. You can set the **PRBS** (pseudorandom binary sequence) order and the number of symbols to simulate. This block is not carried over from the SerDes Designer app.
- Double-click the Tx block to look inside the Tx subsystem. Since there is no algorithmic model for the transmitter, the Tx subsystem is simply a pass through from the WaveIn to WaveOut ports.
- Double-click the Analog Channel block to open the Block Parameters dialog box. The parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx **Analog Model** parameters are carried over from the SerDes Designer app.
- Double-click on the Rx block to look inside the Rx subsystem. The subsystem has the VGA and DFECDR blocks carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.

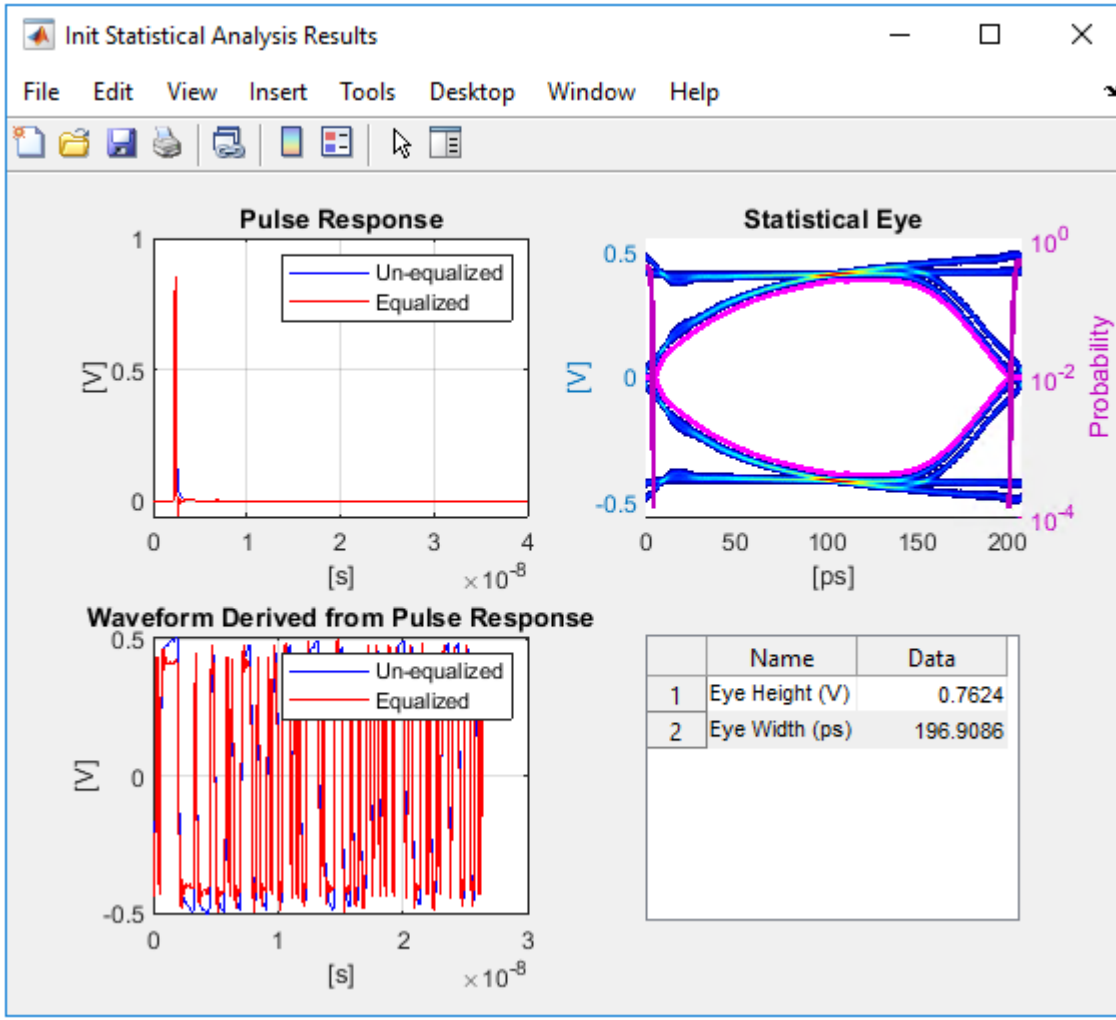
Run the Model

Run the model to simulate the SerDes system.

Two plots are generated. The first is a live time domain (GetWave) eye diagram that is updated as the model is running.



After the simulation has completed the second plot contains four views of the statistical (Init) results, similar to what is available in the SerDes Designer app.



Review Rx VGA Block

- Inside the Rx subsystem, double-click the VGA block to open the VGA Block Parameters dialog box.
- The **Mode** and **Gain** settings are carried over from the SerDes Designer app.

Update Rx DFECDR Block

- Inside the Rx subsystem, double-click the DFECDR block to open the DFECDR Block Parameters dialog box.
- The **Initial tap weights**, **Minimum DFE tap value**, and **Maximum tap value** RMS settings are carried over from the SerDes Designer app. The **Adaptive gain** and **Adaptive step size** are set to `3e-06` and `1e-06`, respectively, which are reasonable values based on DDR5 SDRAM expectations.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Deselect **Phase offset** and **Reference offset** to remove these parameters from the AMI file, effectively hard-coding these parameters to their current values.

Generate DDR5 SDRAM IBIS-AMI Models

The final part of this example takes the customized Simulink model, modifies the AMI parameters for a DDR5 SDRAM, and then generates IBIS-AMI compliant DDR5 SDRAM model executables, IBIS and AMI files.

Open the Block Parameter dialog box for the Configuration block and click on the **Open SerDes IBIS-AMI Manager** button. In the **IBIS** tab inside the SerDes IBIS-AMI manager dialog box, the analog model values are converted to standard IBIS parameters that can be used by any industry-standard simulator.

Review Transmitter (Tx) AMI Parameters

Open the **AMI-Tx** tab in the SerDes IBIS-AMI manager dialog box. Notice that there are no model-specific parameters since the DDR5 SDRAM Tx does not have any equalization.

Update Receiver (Rx) AMI Parameters

Open the **AMI-Rx** tab in the SerDes IBIS-AMI manager dialog box. The reserved parameters are listed first followed by the model-specific parameters adhering to the format of a typical AMI file.

Set the VGA Gain:

- Highlight **Gain.**
- Click the **Edit Parameter**... button to launch the Add/Edit AMI Parameter dialog box.
- In the **Description** box, type `Rx Amplifier Gain`.
- Make sure **Format** is set to **List** and set **Default to** `1`.

- In the **List values** box, enter `[0.5 0.631 0.794 1 1.259 1.585 2]`
- In the **List_Tip values** box, enter `["-6 dB" "-4 dB" "-2 dB" "0 dB" "2 dB" "4 dB" "6 dB"]`
- Click **OK** to save the changes.

Set First DFE Tap Weight

- Highlight **TapWeight 1**.
- Click the **Edit Parameter**... button to launch the Add/Edit AMI Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.2`, and **Max** = `0.05`
- Click **OK.**

Set Second DFE Tap Weight

- Highlight **TapWeight 2**.
- Click the **Edit Parameter**... button to launch the Add/Edit AMI Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.075`, and **Max** = `0.075`
- Click **OK.**

Set Third DFE Tap Weight

- Highlight **TapWeight 3**.
- Click the **Edit Parameter**... button to launch the Add/Edit AMI Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.06`, and **Max** = `0.06`
- Click **OK.**

Set Fourth DFE Tap Weight

- Highlight **TapWeight 4**.
- Click the **Edit Parameter**... button to launch the Add/Edit AMI Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.045`, and **Max** = `0.045`
- Click **OK.**

Export Models

Open the **Export** tab in the SerDes IBIS-AMI manager dialog box.

- Update the **Tx model name** to `ddr5_sdram_tx.`
- Update the **Rx model name** to `ddr5_sdram_rx`.
- Note that **Tx and Rx corner percentage** is set to `10`. This scales the minimum/maximum analog model corner values by +/-10%.
- Verify that **Dual model** is selected for both the Tx and the Rx AMI model settings. This creates model executables that support both statistical (Init) analysis and time-domain (GetWave) simulation.
- Set the Rx model **Bits to ignore** value to `250000` to allow sufficient time for the Rx DFE taps to settle during time domain simulations.
- Set the **Models to export** to **Rx only** and ensure that all files have been selected to be generated (**IBIS file**, **AMI file(s)** and **DLL file(s)**).
- Set the **IBIS file name** to `temp_ddr5_sdram`.
- Click the **Export** button to generate models in the Target directory.

Add Jitter to AMI files

- Browse to the Target directory specified in the SerDes IBIS-AMI Manager. Open the Rx AMI file and add the following Reserved Parameters:

```
(Rx_Receiver_Sensitivity (Usage Info) (Type Float) (Value 0.04))

(Rx_Clock_Recovery_DCD (Usage Info) (Type UI) (Value 0.0125))

(Rx_Clock_Recovery_Dj (Usage Info) (Type UI) (Value 0.0175))

(Rx_Clock_Recovery_Rj (Usage Info) (Type UI) (Value 0.00375))
```

- These ranges allow the user to fine-tune the jitter values to meet DDR5 jitter mask requirements.
- Save and close the file.

Update DDR5 Analog Models

To accommodate different topologies, loading configurations, data rates and transfers, DDR5 requires variable output drive strength and input on-die termination (ODT). While the same algorithmic AMI model is used, multiple analog models are required to cover all these use cases. The generation of these analog models is out of scope for this example, so a completed IBS file with the following analog models in it is available in the current example directory:

- POD11_IO_ZO34_ODTOFF: 34 ohm output impedance with no input ODT.
- POD11_IO_ZO48_ODTOFF: 48 ohm output impedance with no input ODT.
- POD11_IN_ODT34_C: Input with 34 ohm ODT.
- POD11_IN_ODT40_C: Input with 40 ohm ODT.
- POD11_IN_ODT48_C: Input with 48 ohm ODT.
- POD11_IN_ODT60_C: Input with 60 ohm ODT.
- POD11_IN_ODT80_C: Input with 80 ohm ODT.
- POD11_IN_ODT120_C: Input with 120 ohm ODT.
- POD11_IN_ODT240_C: Input with 240 ohm ODT.

To generate this complete IBIS file, the following changes were made to `temp_ddr5_sdram.ibs` using a text editor:

- Created one pin with a **signal_name** of DQ1_sdram and **model_name** of dq.
- Added two drivers with **Model_type** of I/O and named them POD11_IO_Z034_ODTOFF and POD11_IO_Z048_ODTOFF, respectively.
- Added seven receiver models and named them:

a) POD11_IN_ODT34_C

b) POD11_IN_ODT40_C

c) POD11_IN_ODT48_C

d) POD11_IN_ODT60_C

e) POD11_IN_ODT80_C

f) POD11_IN_ODT120_C

g) POD11_IN_ODT240_C

- Added VI curves and **Algorithmic Model** sections to all above mentioned models.
- Added a **Model Selector** section that references all above mentioned models.

Test Generated IBIS-AMI Models

The DDR5 transmitter and receiver IBIS-AMI models are now complete and ready to be tested in any industry-standard AMI model simulator.

DDR5 Controller Transmitter/Receiver IBIS-AMI Model

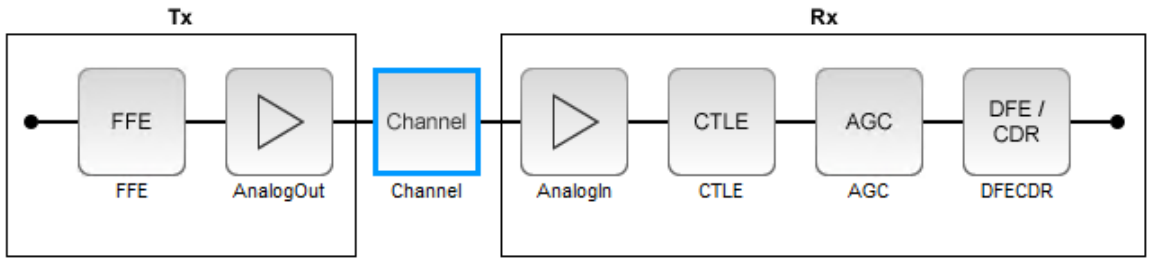
This example shows how to create generic DDR5 transmitter and receiver IBIS-AMI models using the library blocks in SerDes Toolbox™. Since DDR5 DQ signals are bidirectional, this example creates Tx and Rx models for the controller. The generated models conform to the IBIS-AMI specification.

DDR5 Controller Tx/Rx IBIS-AMI Model Setup in SerDes Designer App

The first part of this example sets up and explores the target transmitter and receiver architectures using the blocks required for DDR5 in the SerDes Designer app. The SerDes system is then exported to Simulink® for further customization and IBIS-AMI Model generation.

Type the following command in the MATLAB® command window to open the `ddr5_controller` model:

```
>> serdesDesigner('ddr5_controller')
```



The controller has a DDR5 transmitter (Tx) using 5-tap feed forward equalization (FFE). The controller also has a DDR5 receiver (Rx) using a continuous time linear equalizer (CTLE) with 8 pre-defined settings, an automatic gain control (AGC), and a 4-tap decision feedback equalizer (DFE) with built-in clock data recovery.

Configuration Setup

- **Symbol Time** is set to `208.3` ps, since the target operating rate is `4.8` Gbps for DDR5-4800.

- **Target BER** is set to `100e-18`.
- **Signaling** is set to `Single-ended`.
- **Samples per Symbol** and **Modulation** are kept at default values, which are respectively `16` and `NRZ` (nonreturn to zero), respectively.

Transmitter Model Setup

- The Tx FFE block is set up for one pre-tap, one main-tap, and three post-taps by including five tap weights. This is done with the array [0 1 0 0 0], where the main tap is specified by the largest value in the array. Tap ranges will be added later in the example when the model is exported to Simulink.
- The Tx AnalogOut model is set up so that **Voltage** is `1.1` V, **Rise time** is `100` ps, **R** (output resistance) is `50 o`hms, and **C** (capacitance) is `0.65` pF. The actual analog models used in the final model will be generated later in this example.

Channel Model Setup

- **Channel loss** is set to `5` dB, which is typical of DDR channels.
- **Single-ended impedance** is set to `40` ohms.
- **Target Frequency** is set to `2.4` GHz, which is the Nyquist frequency for 4.8 GHz

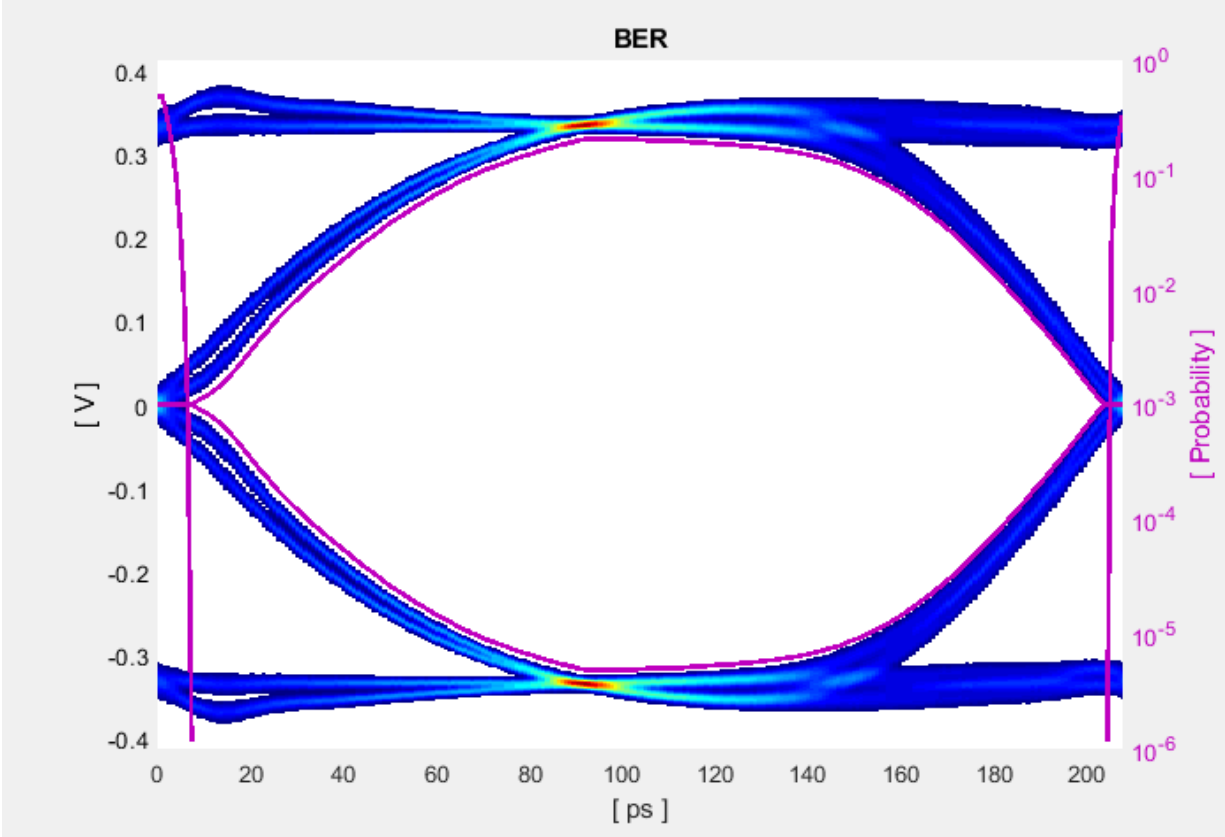
Receiver Model Setup

- The Rx AnalogIn model is set up so that **R** (input resistance) is `40` Ohms and **C** (capacitance) is `0.65`pF. The actual analog models used in the final model will be generated later in this example.
- The CTLE block is set up for `8` configurations. The **Specification** is set to `DC Gain and AC Gain`. **DC Gain** is set to `[0 -1 -2 -3 -4 -5 -6 -7]` dB. Peaking frequency is set to `2.4` GHz. All other parameters are kept at their default values.
- The AGC block has the default Target RMS voltage of `0.3` Volts.
- The DFECDR block is set up for four DFE taps by including four **Initial tap weights** set to 0. The **Minimum tap value** is set to `[-0.2 -0.075 -0.06 -0.045]` V and the **Maximum tap value** is set to `[0.05 0.075 0.06 0.045]` V.

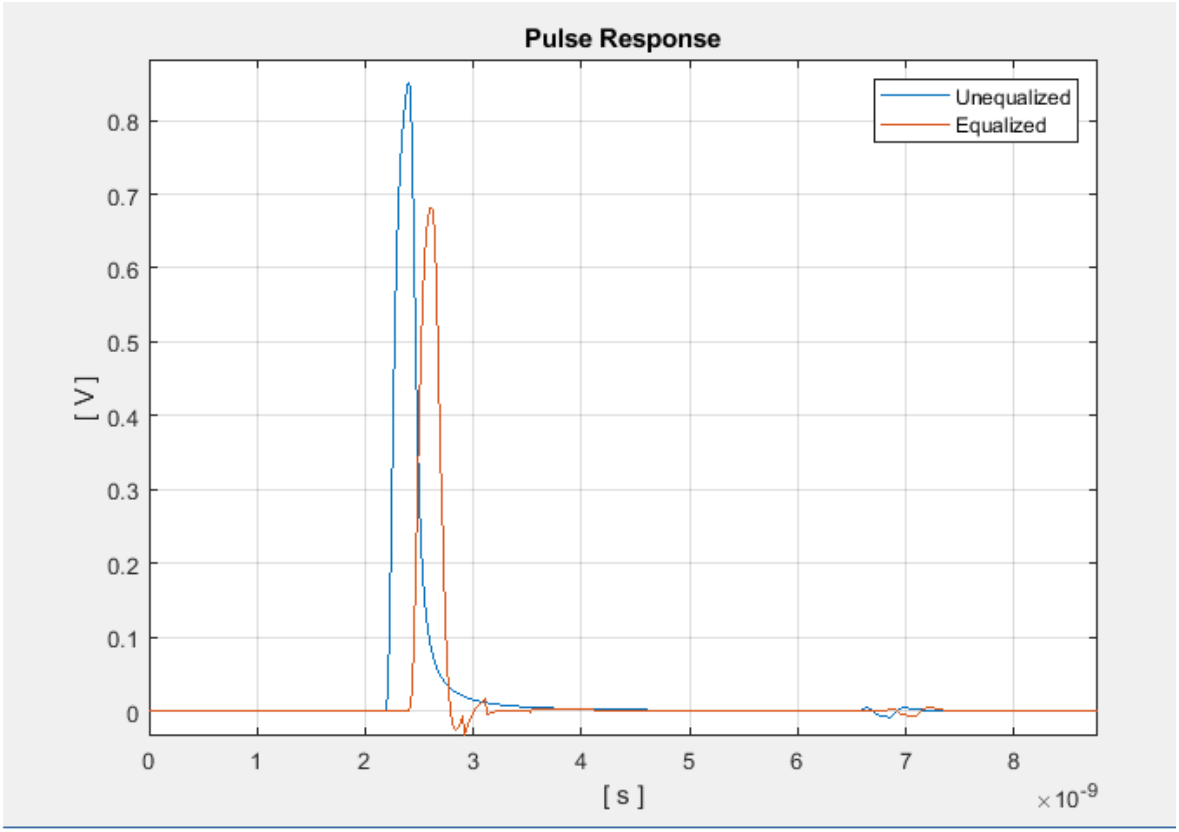
Plot Statistical Results

Use the SerDes Designer **Add Plots** button to visualize the results of the DDR5 Controller setup.

Add the BER plot from **Add Plots** and observe the results.



Add the Pulse Response plot from **Add Plots** and zoom into the pulse area to observe the results.



Export SerDes System to Simulink

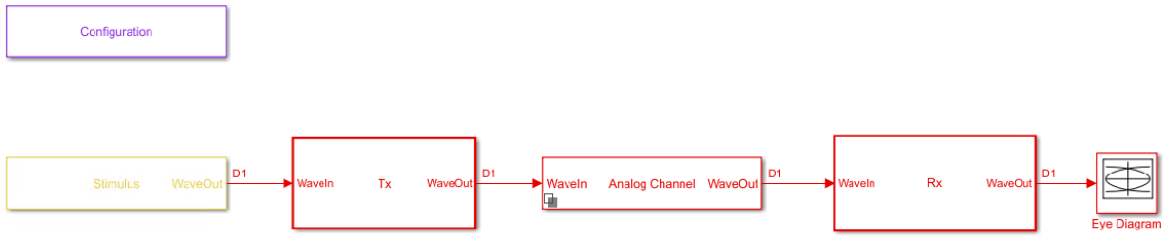
Click **Save** and then click on the **Export** button to export the configuration to Simulink for further customization and generation of the AMI model executables.

DDR5 Controller Tx/Rx IBIS-AMI Model Setup in Simulink

The second part of this example takes the SerDes system exported by the SerDes Designer app and customizes it as required for DDR5 in Simulink.

Review the Simulink Model Setup

The SerDes System imported into Simulink consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks. All the settings from the SerDes Designer app have been transferred to the Simulink model. Save the model and review each block setup.

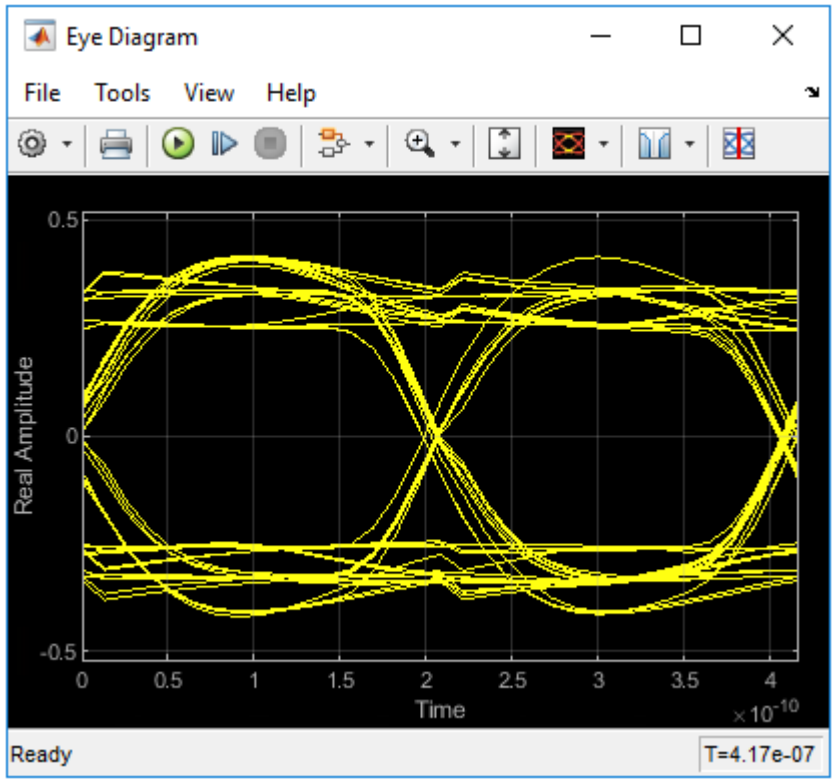


- Double-click the Configuration block to open the Block Parameters dialog box. The parameter values for **Symbol time**, **Samples per symbol**, **Target BER**, **Modulation,** and **Signaling** are carried over from the SerDes Designer app.
- Double-click the Stimulus block to open the Block Parameters dialog box. You can set the **PRBS** (pseudorandom binary sequence) order and the number of symbols to simulate. This block is not carried over from the SerDes Designer app.
- Double-click the Tx block to look inside the Tx subsystem. The subsystem has the FFE block carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.
- Double-click the Analog Channel block to open the Block Parameters dialog box. The parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx **Analog Model** parameters are carried over from the SerDes Designer app.
- Double-click on the Rx block to look inside the Rx subsystem. The subsystem has the CTLE, AGC and DFECDR blocks carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.

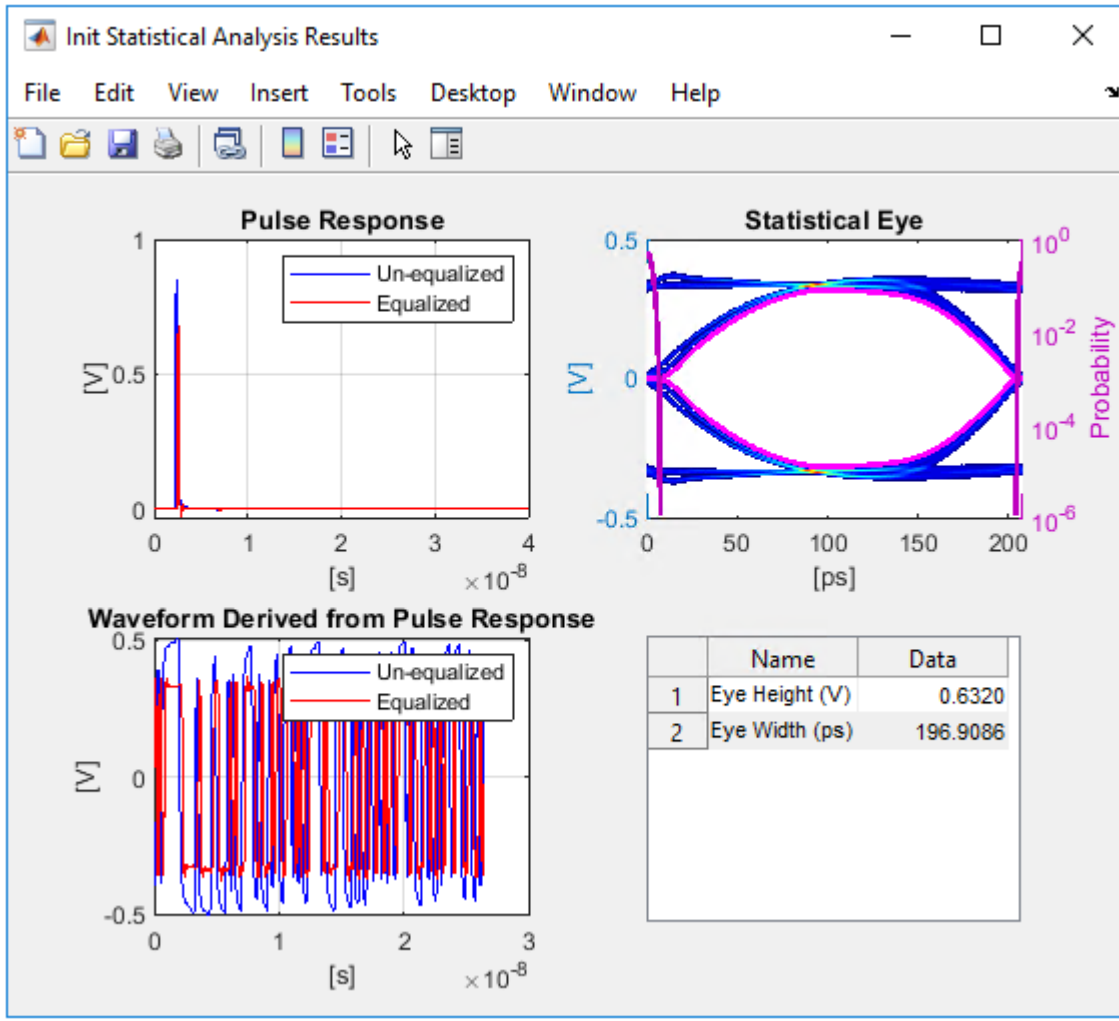
Run the Model

Run the model to simulate the SerDes system.

Two plots are generated. The first is a live time domain (GetWave) eye diagram that is updated as the model is running.



After the simulation has completed the second plot contains four views of the statistical (Init) results, similar to what is available in the SerDes Designer app.



Review Tx FFE Block

- Inside the Tx subsystem, double-click the FFE block to open the FFE Block Parameters dialog box.
- The **Tap Weights** are carried over from the SerDes Designer app.

Review Rx CTLE Block

- Inside the Rx subsystem, double-click the CTLE block to open the CTLE Block Parameters dialog box.
- **DC gain**, **AC gain**, and **Peaking frequency** are carried over from the SerDes Designer app.
- CTLE **Mode** is set to `Adapt`, which means an optimization algorithm built into the CTLE system object selects the optimal CTLE configuration at run time.

Review Rx AGC Block

- Inside the Rx subsystem, double-click the AGC block to open the AGC Block Parameters dialog box.
- The **Target RMS voltage** is carried over from the SerDes Designer app.
- The **Maximum gain** is set to `10` and **Averaging length** (the number of bits over which the average is calculated) is set to `100`. These values are reasonable for a generic controller model.

Update Rx DFECDR Block

- Inside the Rx subsystem, double-click the DFECDR block to open the DFECDR Block Parameters dialog box.
- The **Initial tap weights**, **Minimum DFE tap value,** and **Maximum tap value** RMS settings are carried over from the SerDes Designer app. The **Adaptive gain** and **Adaptive step size** are set to `3e-06` and `1e-06`, respectively, which are reasonable values based on DDR5 Controller expectations.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Deselect **Phase offset** and **Reference offset** to remove these parameters from the AMI file, effectively hard-coding these parameters to their current values.

Generate DDR5 Controller IBIS-AMI Models

The final part of this example takes the customized Simulink model, modifies the AMI parameters for a DDR5 Controller, and then generates IBIS-AMI-compliant DDR5 Controller model executables, IBIS and AMI files.

Open the Block Parameter dialog box for the Configuration block and click on the **Open SerDes IBIS-AMI Manager** button. In the **IBIS** tab inside the SerDes IBIS-AMI

manager dialog box, the analog model values are converted to standard IBIS parameters that can be used by any industry-standard simulator.

Update Transmitter (Tx) AMI Parameters

Open the **AMI-Tx** tab in the SerDes IBIS-AMI manager dialog box. The reserved parameters are listed first followed by the model-specific parameters adhering to the format of a typical AMI file.

Set Pre-Emphasis Tap

- Highlight **TapWeight -1**
- Click the **Edit Parameter...** to launch the Add/Edit Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.2`, and **Max** = `0.2.`
- Click **OK** to save the changes**.**

Set Main Tap

- Highlight **TapWeight 0**.
- Click the **Edit Parameter**... button to launch the Add/Edit Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `1`, **Min** = `0.6`, and **Max** = `1`.
- Click **OK**.

Set First Post-Emphasis Tap

- Highlight TapWeight **1**.
- Select the **Edit Parameter**... button to launch the **Add/Edit Parameter** dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.2`, and **Max** = `0.2`.
- Click **OK**.

Set Second Post-Emphasis Tap

- Highlight TapWeight **2**.
- Select the **Edit Parameter**... button to launch the **Add/Edit Parameter** dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min**= `-0.1`, and **Max** = `0.1`.
- Click **OK**.

Set Third Post-Emphasis Tap

- Highlight TapWeight **3**.
- Select the **Edit Parameter**... button to launch the **Add/Edit Parameter** dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.1`, and **Max** = `0.1`.
- Click **OK**.

Update Receiver (Rx) AMI Parameters

Open the **AMI-Rx** tab in the SerDes IBIS-AMI manager dialog box. The reserved parameters are listed first followed by the model-specific parameters adhering to the format of a typical AMI file.

Set First DFE Tap Weight

- Highlight **TapWeight 1.**
- Click the **Edit Parameter**... button to launch the Add/Edit Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.2`, and **Max** = `0.05`.
- Click **OK**.

Set Second DFE Tap Weight

- Highlight **TapWeight 2**.
- Click the **Edit Parameter**... button to launch the Add/Edit Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.075`, and **Max** = `0.075`.
- Click **OK**.

Set Third DFE Tap Weight

- Highlight **TapWeight 3**.
- Click the **Edit Parameter**... button to launch the Add/Edit Parameter dialog box.
- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.06`, and **Max** = `0.06`.
- Click **OK**.

Set Fourth DFE Tap Weight

- Highlight **TapWeight 4**.
- Click the **Edit Parameter**... button to launch the Add/Edit Parameter dialog box.

- Make sure **Format** is set to **Range** and set **Typ** = `0`, **Min** = `-0.045`, and **Max** = `0.045`.
- Click **OK**.

Export Models

Open the **Export** tab in the SerDes IBIS-AMI manager dialog box.

- Update the **Tx model name** to `ddr5_controller_tx`
- Update the **Rx model name** to `ddr5_controller_rx`
- Note that **Tx and Rx corner percentage** is set to `10`. This scales the minimum/maximum analog model corner values by +/-10%.
- Verify that **Dual model** is selected for both the Tx and the Rx AMI model settings. This creates model executables that support both statistical (Init) analysis and time-domain (GetWave) simulation.
- Set the Tx model **Bits to ignore** to `5` since there are five taps in the Tx FFE.
- Set the Rx model **Bits to ignore** to `250000` to allow sufficient time for the Rx DFE taps to settle during time domain simulations.
- Verify that both Tx and Rx are set to export and that all files have been selected to be generated (**IBIS file**, **AMI file(s)** and **DLL file(s)**).
- Set the **IBIS file name** to `temp_ddr5_controller.ibs`
- Click the **Export** button to generate models in the Target directory.

Add Jitter to AMI files

- Browse to the Target directory specified in the SerDes IBIS-AMI Manager. Open the Rx AMI file and add the following Reserved Parameters:

```
(Rx_Receiver_Sensitivity (Usage Info) (Type Float) (Value 0.04))

(Rx_Clock_Recovery_DCD (Usage Info) (Type UI) (Value 0.0125))

(Rx_Clock_Recovery_Dj (Usage Info) (Type UI) (Value 0.0175))

(Rx_Clock_Recovery_Rj (Usage Info) (Type UI) (Value 0.00375))
```

- These ranges allow the user to fine-tune the jitter values to meet DDR5 jitter mask requirements.

- Save and close the file.

Update DDR5 Analog Models

To accommodate different topologies, loading configurations, data rates and transfers, DDR5 requires variable output drive strength and input on-die termination (ODT). While the same algorithmic AMI model is used, multiple analog models are required to cover all these use cases. The generation of these analog models is out of scope for this example, so a completed IBS file with the following analog models in it is available in the current example directory:

- POD11_IO_ZO50_ODTOFF: 50 ohm output impedance with no input ODT.
- POD11_IN_ODT40_C: Input with 40 ohm ODT.
- POD11_IN_ODT60_C: Input with 60 ohm ODT.

To generate this complete IBIS file, the following changes were made to temp_ddr5_controller.ibs using a text editor:

- Created one pin with a signal_name of DQ1_controller and model_name of dq.
- Changed the driver Model_type to I/O and named it POD11_IO_Z050_ODTOFF.
- Added two receiver models and named them POD11_IN_ODT40_C and POD11_IN_ODT60_C, respectively.
- Added VI curves and Algorithmic Model sections to all above mentioned models.
- Added a Model Selector section that references the above mentioned models.

Test Generated IBIS-AMI Models

The DDR5 transmitter and receiver IBIS-AMI models are now complete and ready to be tested in any industry-standard AMI model simulator.

CEI-56G-LR Transmitter/Receiver IBIS-AMI Model

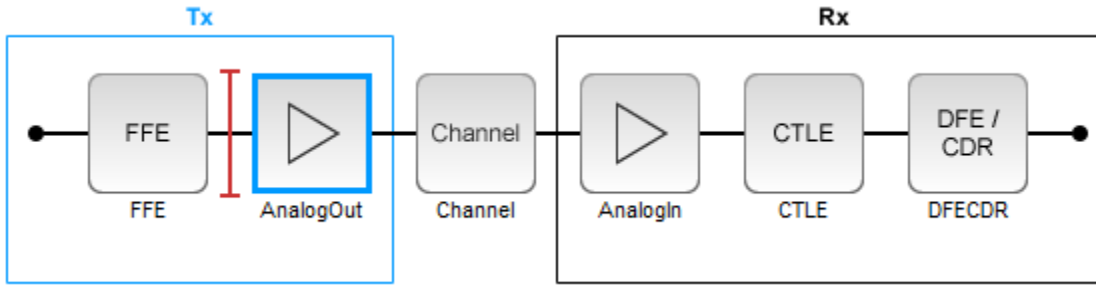
This example shows how to create generic CEI-56G-LR transmitter and receiver IBIS-AMI models using the library blocks in SerDes Toolbox™. The generated models conform to the IBIS-AMI and OIF-CEI-04.0 specifications.

CEI-56G-LR Tx/Rx IBIS-AMI Model Setup in SerDes Designer App

The first part of this example sets up the target transmitter and receiver AMI model architecture using the datapath blocks required for CEI-56G in the SerDes Designer app. The model is then exported to Simulink® for further customization.

This example uses the SerDes Designer model cei_56G_lr_txrx. Type the following command in the MATLAB® command window to open the model:

```
>> serdesDesigner('cei_56g_lr_txrx')
```



A CEI-56G-LR compliant transmitter uses a 4-tap feed forward equalizer (FFE) with two pre-taps and one post-tap. The receiver model uses a continuous time linear equalizer (CTLE) with 17 pre-defined settings, and a 12 to 18 tap decision feedback equalizer (DFE). To support this configuration the SerDes System is set up as follows:

Configuration Setup

- **Symbol Time** is set to 35.71 ps, for a symbol rate of 28 GBaud and a PAM4 rate of 56 Gbps.
- **Target BER** is set to 100e-6, which assumes a compliant receiver with FEC.
- **Modulation** is set to PAM4.

- **Samples per Symbol** and **Signaling** are kept at default values, which are respectively 16 and Differential.

Transmitter Model Setup

- The Tx FFE block is set up for two pre-taps and one post-tap by including four tap weights, as specified in the OIF-CEI-04.0 specification. This is done with the array [0 0 1 0], where the main tap is specified by the largest value in the array.
- The Tx AnalogOut model is set up so that **Voltage** is 1.0 V, **Rise time** is 2.905 ps, **R** (single-ended output resistance) is 50 Ohms, and **C** (capacitance) is 0.16 pF.

Channel Model Setup

- **Channel loss** is set to 20 dB.
- **Differential impedance** is kept at default 100 Ohms.
- **Target Frequency** is set to the Nyquist frequency, 14 GHz.

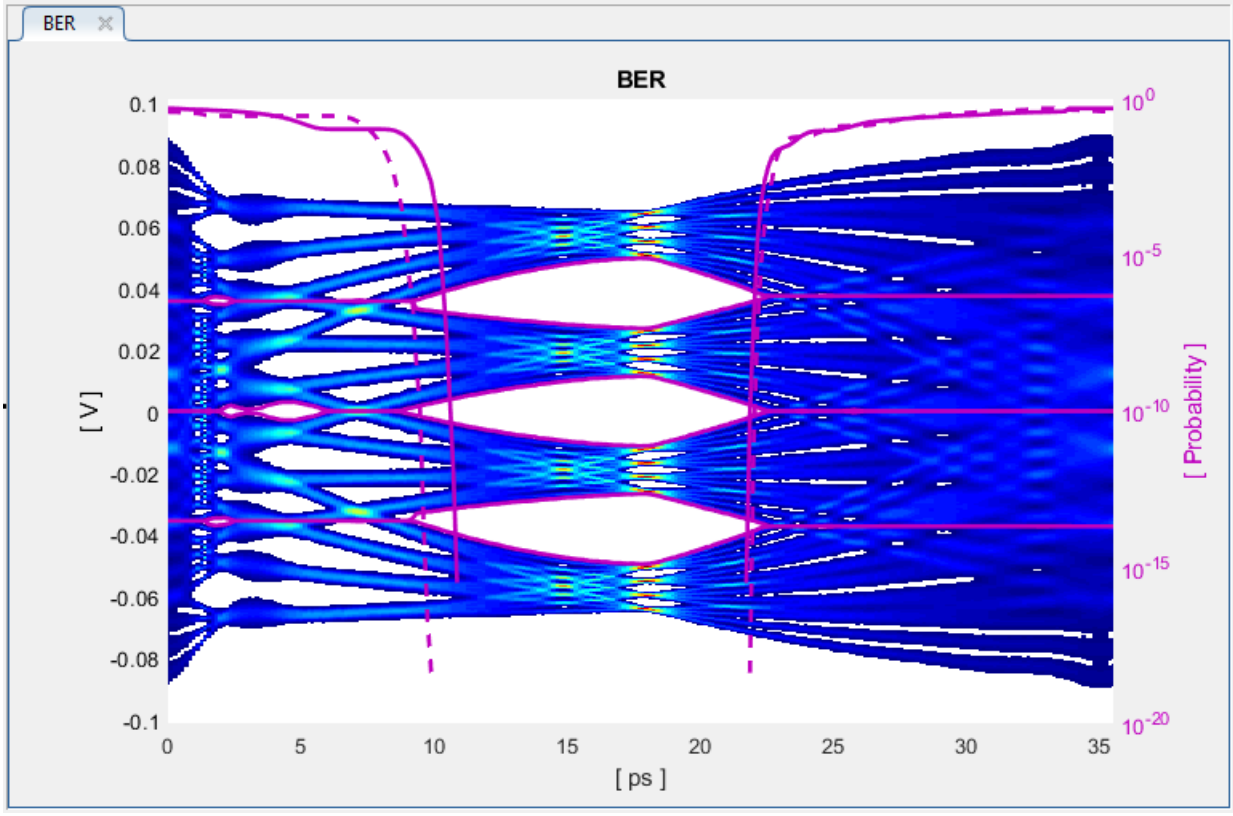
Receiver Model Setup

- The Rx AnalogIn model is set up so that **R** (single-ended input resistance) is 50 Ohms and **C** (capacitance) is 0.16 pF.
- The Rx CTLE block is set up for 147 configurations using the **GPZ** (Gain Pole Zero) matrix.
- The Rx DFE/CDR block is set up for 18 DFE taps. The limits for the taps are set to `-0.7` to `0.7`.

Plot Statistical Results

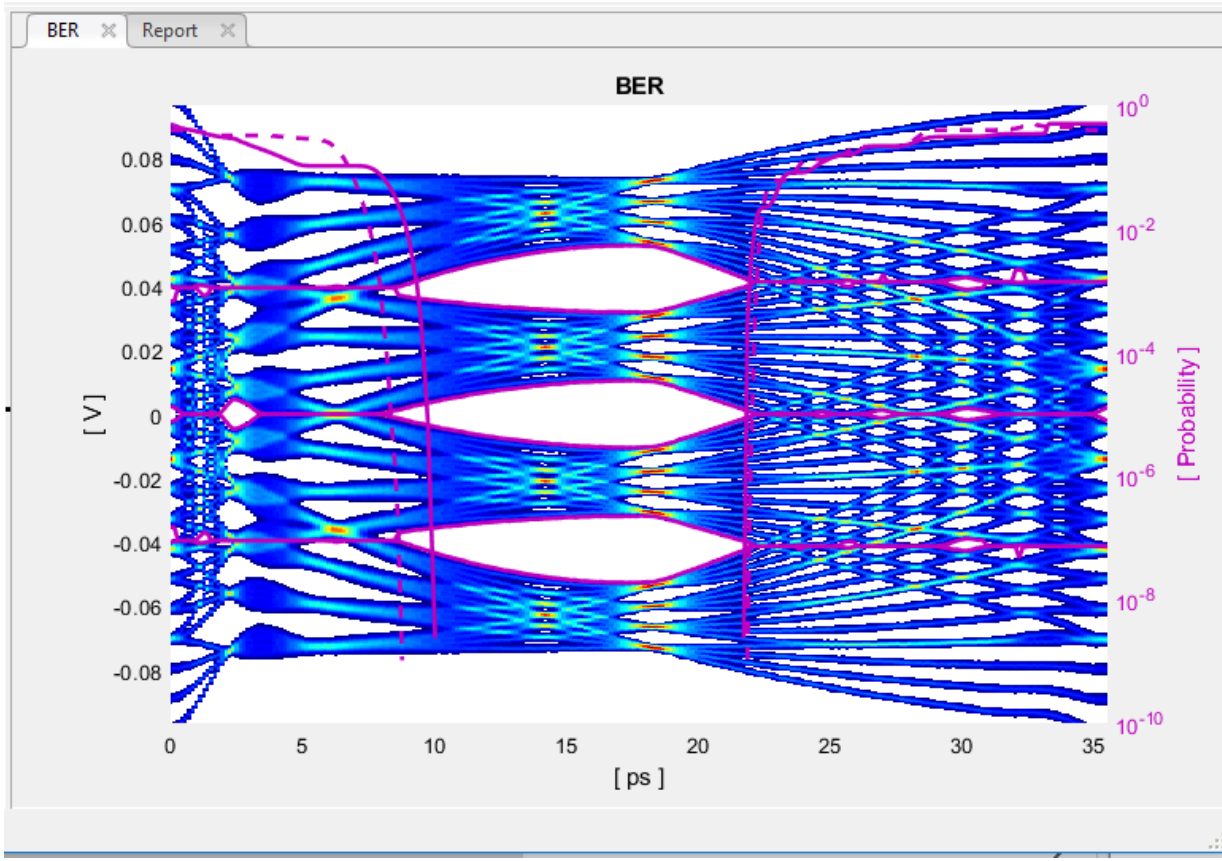
Use the SerDes Designer plots to visualize the results of the CEI-56G-LR setup.

Add the BER plot from **Add Plots** and observe the results.



Add the report from Add Plots and observe that the CTLE Config is 129.

Change the Rx CTLE **Mode** parameter to **`fixed`** and the **ConfigSelect** parameter value from 129 to 8 and observe how this changes the data eye.



Before continuing, reset the value of Rx CTLE **Mode** back to `adapt`. Resetting here will avoid the need to set it again after the model has been exported to Simulink.

Export SerDes System to Simulink

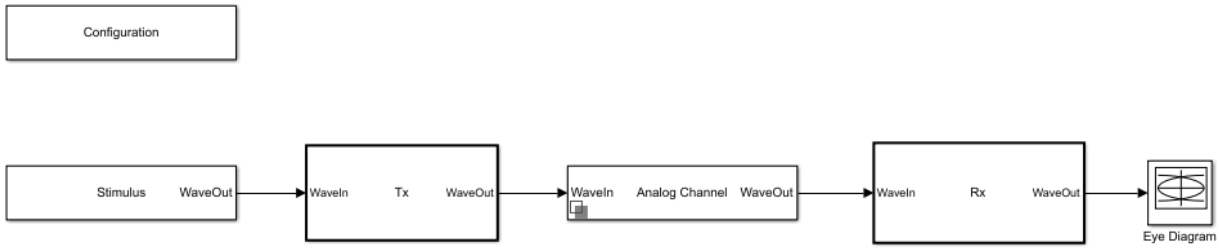
Click on the **Export** button to export the above configuration to Simulink for further customization and generation of the AMI model executables.

CEI-56G-LR Tx/Rx IBIS-AMI Model Setup in Simulink

The second part of this example takes the SerDes system exported by the SerDes Designer app and customizes it as required for CEI-56G-LR in Simulink.

Review Simulink Model Setup

The SerDes System exported into Simulink consists of Configuration, Stimulus, Tx, Analog Channel and Rx blocks. All the settings from the SerDes Designer app have been transferred to the Simulink model. Save the model and review each block setup.

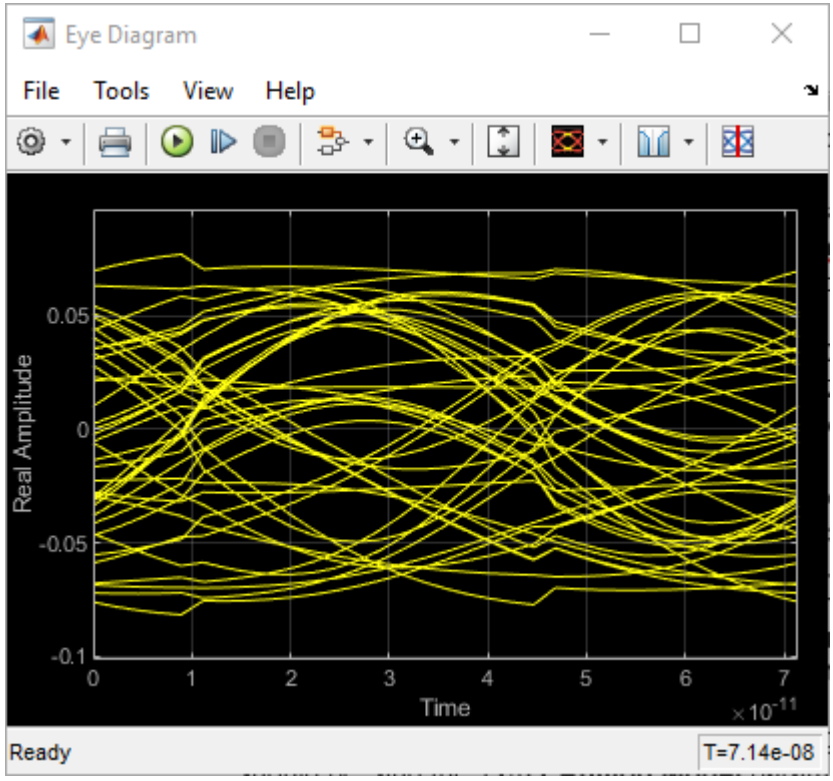


- Double click the Configuration block to open the Block Parameters dialog box. The parameter values for **Symbol time**, **Samples per symbo**l, **Target BER**, **Modulation** and **Signaling** are carried over from the SerDes Designer app.
- Double click the Stimulus block to open the Block Parameters dialog box. You can set the **PRBS** (pseudorandom binary sequence) order and the number of symbols to simulate. The settings for this block are not carried over from the SerDes Designer app.
- Double click the Tx block to look inside the Tx subsystem. The subsystem has the FFE block carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.
- Double click the Analog Channel block to open the Block Parameters dialog box. The parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx analog model parameters are carried over from the SerDes Designer app.
- Double click on the Rx block to look inside the Rx subsystem. The subsystem has the CTLE and DFECDR blocks carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.

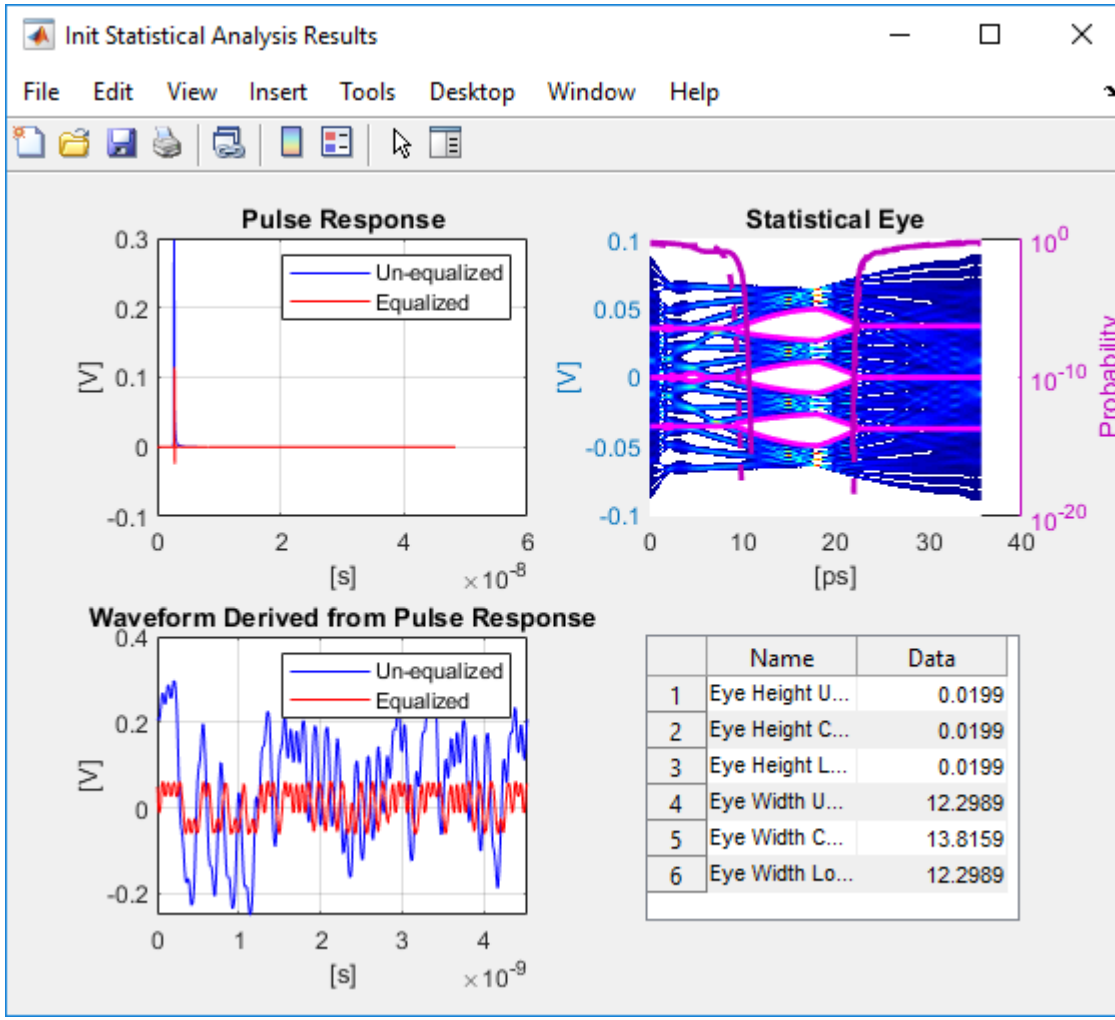
Run the Model

Run the model to simulate the SerDes System.

Two plots are generated. The first is a live time domain (GetWave) eye diagram that is updated as the model is running.



After the simulation has completed the second plot contains four views of the statistical (Init) results, similar to what is available in the SerDes Designer App.



Update Tx FFE Block

- Inside the Tx subsystem, double click the FFE block to open the FFE Block Parameters dialog box.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.

- Deselect the **Mode** parameter to remove this parameter from the AMI file, effectively hard-coding the current value of **Mode** in the final AMI model to Fixed.

Review Rx CTLE Block

- Inside the Rx subsystem, double click the CTLE block to open the CTLE Block Parameters dialog box.
- **Gain pole zero** data is carried over from the SerDes Designer app.
- CTLE **Mode** is set to Adapt, which means an optimization algorithm built into the CTLE system object selects the optimal CTLE configuration at run time.

Update Rx DFECDR Block

- Inside the Rx subsystem, double click the DFECDR block to open the DFECDR Block Parameters dialog box.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Deselect the **Phase offset** and **Reference offset** parameters to remove these parameters from the AMI file, effectively hard-coding these parameters to their current values.

Generate CEI-56G-LR Tx/Rx IBIS-AMI Model

The final part of this example takes the customized Simulink model, modifies the AMI parameters for CEI-56G-LR, then generates IBIS-AMI compliant CEI-56G-LR model executables, IBIS and AMI files.

Open the Block Parameter dialog box for the Configuration block and click on the **SerDes IBIS-AMI Manager** button. In the **IBIS** tab inside the SerDes IBIS-AMI manager dialog box, the analog model values are converted to standard IBIS parameters that can be used by any industry standard simulator. In the **AMI-Tx** and **AMI-Rx** tabs in the SerDes IBIS-AMI manager dialog box, the reserved parameters are listed first followed by the model specific parameters following the format of a typical AMI file.

Export Models

Select the **Export** tab in the **SerDes IBIS-AMI manager** dialog box.

- Update the **Tx model name** to `cei_56g_lr_tx`
- Update the **Rx model name** to `cei_56g_lr_rx`

- Note that the Tx and Rx **corner percentage** is set to 10%. This will scale the min/max analog model corner values by +/-10%.
- Verify that **Dual model** is selected for both the Tx and the Rx. This will create model executables that support both statistical (Init) and time domain (GetWave) analysis.
- Set the **Tx model Bits to ignore value** to 4 since there are four taps in the Tx FFE.
- Set the **Rx model Bits to ignore value** to 200000 to allow sufficient time for the Rx DFE taps to settle during time domain simulations.
- Verify that **Both Tx and Rx** are set to Export and that all files have been selected to be generated (IBIS file, AMI files and DLL files).
- Set the **IBIS file name** to be `cei_56g_lr_serdes.ibs`
- Press the **Export** button to generate models in the **Target directory**.

Add Jitter to Tx AMI file

- Browse to the Target directory specified in the SerDes IBIS-AMI Manager and open the Tx .ami file in a text editor.
- In the Tx AMI file, add the following Reserved Parameters at the end of the Reserved Parameters section:

```
(Tx_Rj (Usage Info) (Type UI) (Range 0 0 0.05))

(Tx_Dj (Usage Info) (Type UI) (Range 0 0 0.1))

(Tx_DCD (Usage Info) (Type UI) (Range 0 0 0.1))
```

These ranges allow the user to fine-tune the jitter values to meet CEI-56G-LR jitter mask requirements.

- Save and close the file.

Test Generated IBIS-AMI Models

The CEI-56G-LR transmitter and receiver IBIS-AMI models are now complete and ready to be tested in any industry standard AMI model simulator.

References

IBIS 6.1 Specification, https://ibis.org/ver6.1/ver6_1.pdf

USB3.1 Transmitter/Receiver IBIS-AMI Model

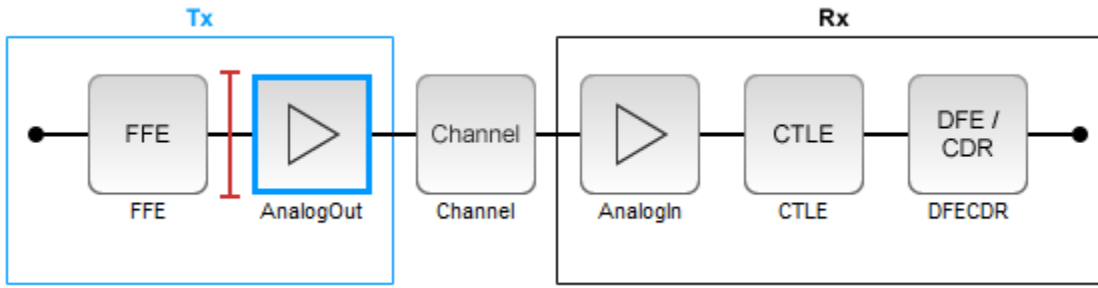
This example shows how to create generic Universal Serial Bus version 3.1 (USB3.1) transmitter and receiver IBIS-AMI models using the library blocks in SerDes Toolbox™. The generated models conform to the IBIS-AMI and USB3.1 specifications.

USB3.1 Tx/Rx IBIS-AMI Model Setup in SerDes Designer App

The first part of this example sets up the target transmitter and receiver AMI model architecture using the datapath blocks required for USB3.1 in the SerDes Designer app. The model is then exported to Simulink® for further customization.

This example uses the SerDes Designer model usb3_1_txrx_ami. Type the following command in the MATLAB® command window to open the model:

```
>> serdesDesigner('usb3_1_txrx_ami')
```



A USB3.1 compliant transmitter uses a 3-tap feed forward equalizer (FFE) with one pre-tap and one post-tap. The receiver model uses a continuous time linear equalizer (CTLE) with seven pre-defined settings, and a 1-tap decision feedback equalizer (DFE). To support this configuration the SerDes System is set up as follows:

Configuration Setup

- **Symbol Time** is set to 100 ps, since the maximum allowable USB3.1 operating frequency is 10 GHz.
- **Target BER** is set to 1e-12 as specified in the USB3.1 specification.
- **Samples per Symbol**, **Modulation**, and **Signaling** are kept at default values, which are respectively 16, NRZ (non-return to zero), and Differential.

Transmitter Model Setup

- The Tx FFE block is set up for one pre- and one post-tap by including three tap weights, as specified in the USB3.1 specification. This is done with the array [0 1 0], where the main tap is specified by the largest value in the array.
- The Tx AnalogOut model is set up so that **Voltage** is 1.00 V, **Rise time** is 60 ps, **R** (single-ended output resistance) is 50 Ohms, and **C** (capacitance) is 0.5 pF.

Channel Model Setup

- **Channel loss** is set to 15dB.
- **Differential impedance** is kept at default 100 Ohms.
- **Target Frequency** is set to the Nyquist frequency, 5 GHz.

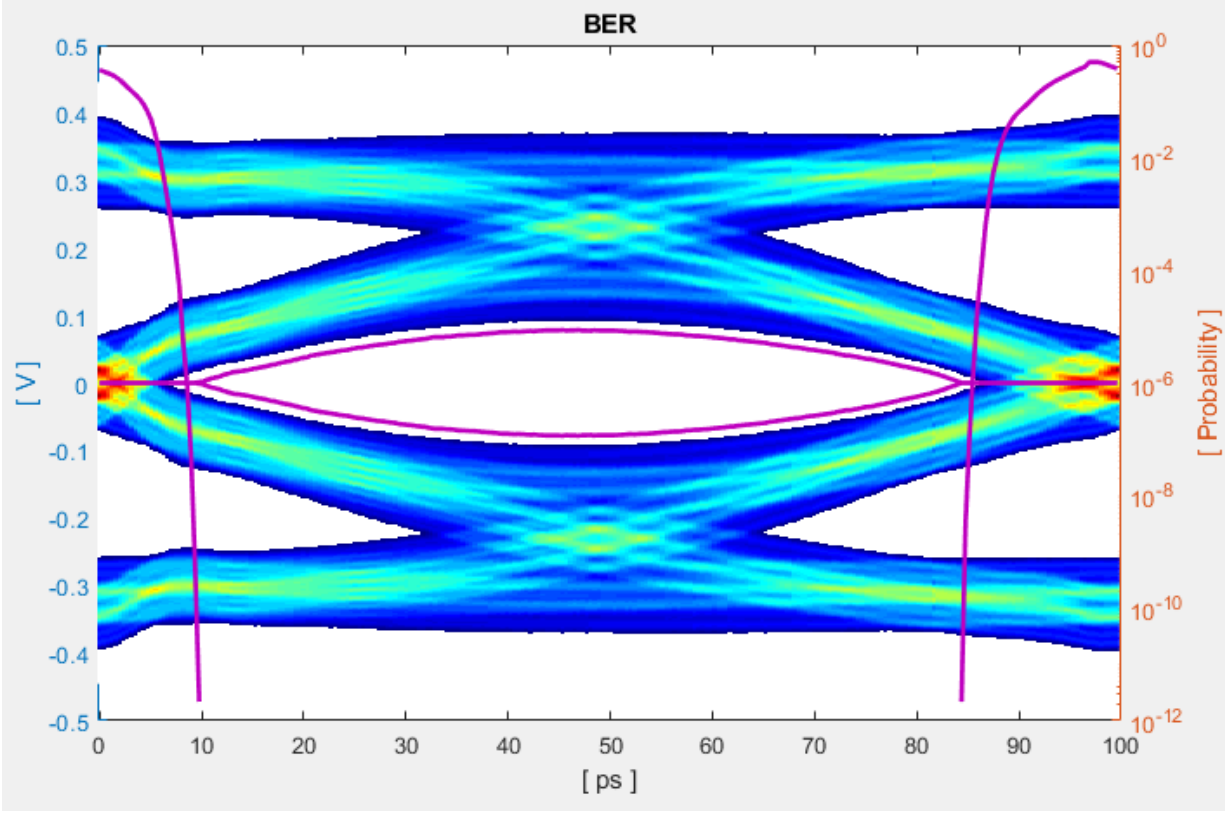
Receiver Model Setup

- The Rx AnalogIn model is set up so that **R** (single-ended input resistance) is 50 Ohms and **C** (capacitance) is 0.5 pF.
- The Rx CTLE block is set up for 7 configurations. The **GPZ** (Gain Pole Zero) matrix data is derived from the transfer function given in the USB3.1 Behavioral CTLE specification.
- The Rx DFE/CDR block is set up for one DFE tap. The limits for the tap are as defined by the USB3.1 specification: `+/-50 mV`.

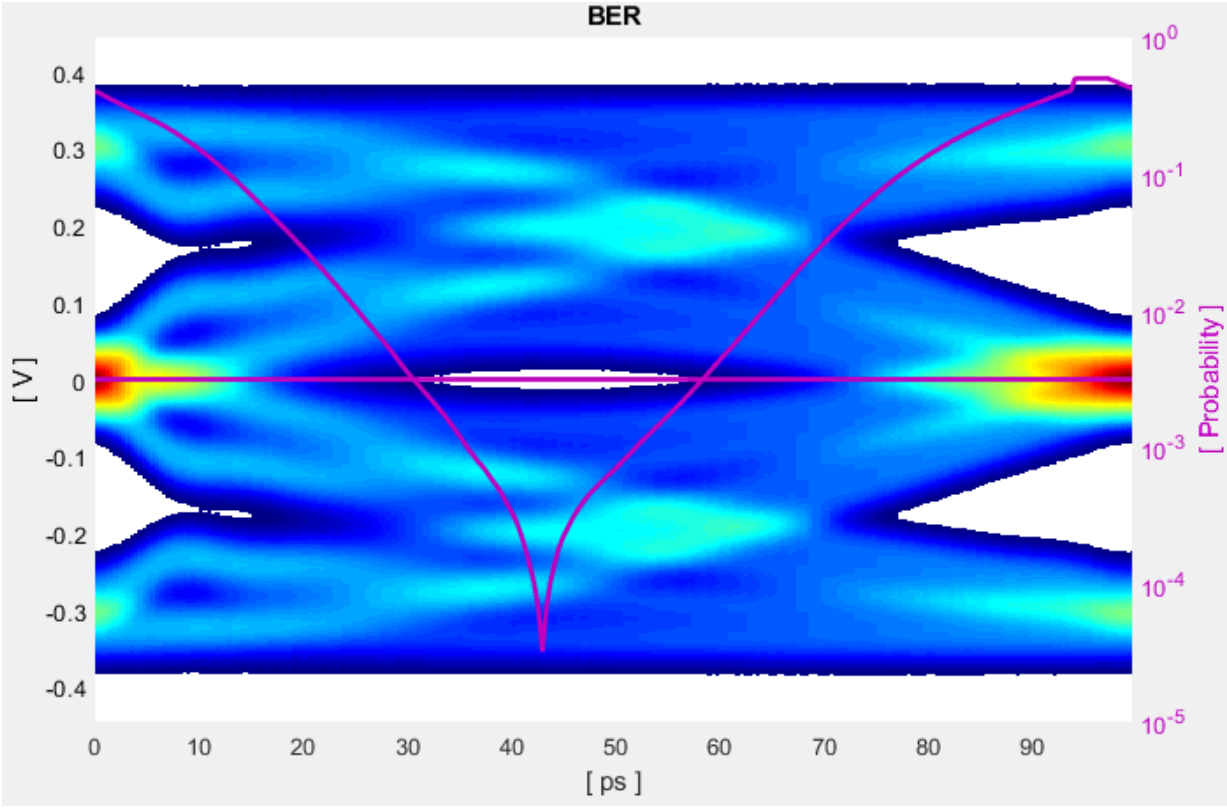
Plot Statistical Results

Use the SerDes Designer plots to visualize the results of the USB3.1 setup.

Add the BER plot from **ADD Plots** and observe the results.



Change the Rx CTLE **Mode** parameter from `adapt` to `fixed` and change the **ConfigSelect** parameter value from 6 to 0 and observe how this changes the data eye.



Before continuing, reset the value of Rx CTLE **Mode** back to `adapt`. Resetting the value here will avoid the need to set it again after the model has been exported to Simulink.

Export SerDes System to Simulink

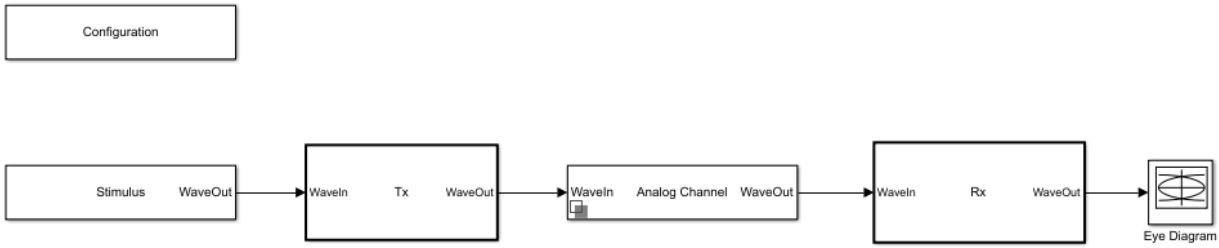
Click on the **Export** button to export the above configuration to Simulink for further customization and generation of the AMI model executables.

USB3.1 Tx/Rx IBIS-AMI Model Setup in Simulink

The second part of this example takes the SerDes system exported by the SerDes Designer app and customizes it as required for USB3.1 in Simulink.

Review Simulink Model Setup

The SerDes System imported into Simulink consists of the Configuration, Stimulus, Tx, Analog Channel and Rx blocks. All the settings from the SerDes Designer app have been transferred to the Simulink model. Save the model and review each block setup.

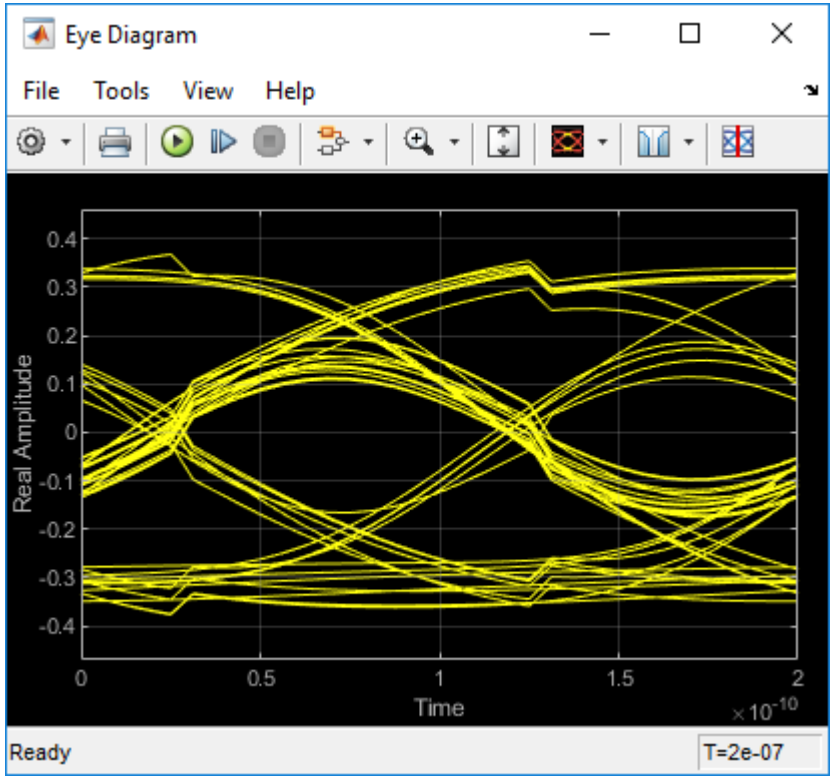


- Double click the Configuration block to open the Block Parameters dialog box. The parameter values for **Symbol time**, **Samples per symbol**, **Target BER**, **Modulation** and **Signaling** are carried over from the SerDes Designer app.
- Double click the Stimulus block to open the Block Parameters dialog box. You can set the **PRBS** (pseudorandom binary sequence) order and the number of symbols to simulate. This block is not carried over from the SerDes Designer app.
- Double click the Tx block to look inside the Tx subsystem. The subsystem has the FFE block carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.
- Double click the Analog Channel block to open the Block Parameters dialog box. The parameter values for **Target frequency**, **Loss**, **Impedance** and Tx/Rx **Analog Model** parameters are carried over from the SerDes Designer app.
- Double click on the Rx block to look inside the Rx subsystem. The subsystem has the CTLE and DFECDR blocks carried over from the SerDes Designer app. An Init block is also introduced to model the statistical portion of the AMI model.

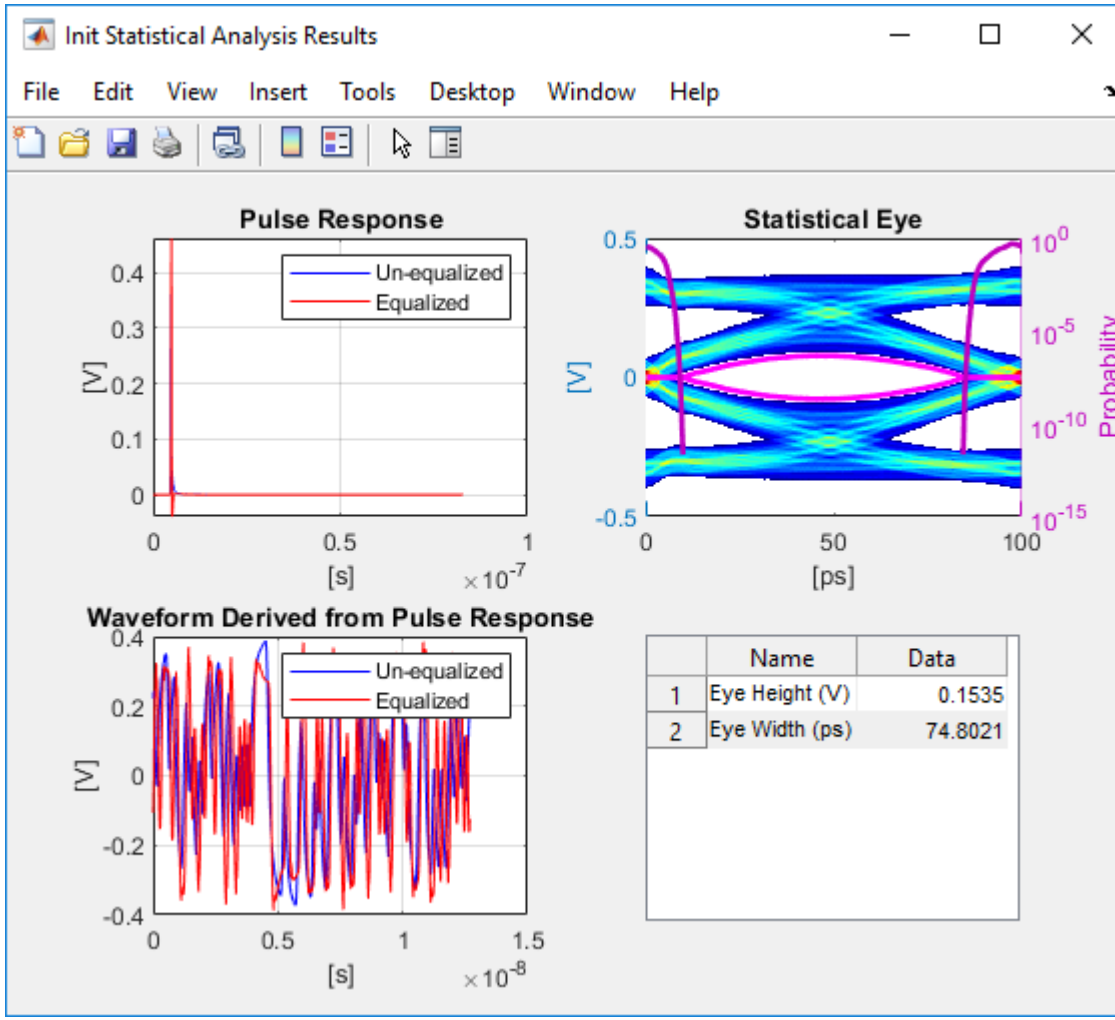
Run the Model

Run the model to simulate the SerDes System.

Two plots are generated. The first is a live time domain (GetWave) eye diagram that is updated as the model is running.



After the simulation has completed the second plot contains four views of the statistical (Init) results, similar to what is available in the SerDes Designer App.



Update Tx FFE Block

- Inside the Tx subsystem, double click the FFE block to open the FFE Block Parameters dialog box.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.

- Deselect the **Mode** parameter to remove this parameter from the AMI file, effectively hard-coding the current value of **Mode** in the final AMI model to Fixed.

Review Rx CTLE Block

- Inside the Rx subsystem, double click the CTLE block to open the CTLE Block Parameters dialog box.
- **Gain pole zero** data is carried over from the SerDes Designer app. This data is derived from the transfer function given in the USB3.1 Behavioral CTLE specification.
- CTLE **Mode** is set to Adapt, which means an optimization algorithm built into the CTLE system object selects the optimal CTLE configuration at run time.

Update Rx DFECDR Block

- Inside the Rx subsystem, double click the DFECDR block to open the DFECDR Block Parameters dialog box.
- Expand the **IBIS-AMI parameters** to show the list of parameters to be included in the IBIS-AMI model.
- Deselect the **Phase offset** and **Reference offset** parameters to remove these parameters from the AMI file, effectively hard-coding these parameters to their current values.

Generate USB3.1 Tx/Rx IBIS-AMI Model

The final part of this example takes the customized Simulink model, modifies the AMI parameters for USB3.1, then generates IBIS-AMI compliant USB3.1 model executables, IBIS and AMI files.

Open the Block Parameter dialog box for the Configuration block and click on the **SerDes IBIS-AMI Manager** button. In the **IBIS** tab inside the SerDes IBIS-AMI manager dialog box, the analog model values are converted to standard IBIS parameters that can be used by any industry standard simulator. In the **AMI-Tx** and **AMI-Rx** tabs in the SerDes IBIS-AMI manager dialog box, the reserved parameters are listed first followed by the model specific parameters following the format of a typical AMI file.

Export Models

Select the **Export** tab in the SerDes IBIS-AMI manager dialog box.

- Update the **Tx model name** to `usb3_1_tx`

- Update the **Rx model name** to `usb3_1_rx`
- Note that the Tx and Rx **corner percentage** is set to 10%. This will scale the min/max analog model corner values by +/-10%.
- Verify that **Dual model** is selected for both the Tx and the Rx. This will create model executables that support both statistical (Init) and time domain (GetWave) analysis.
- Set the **Tx model Bits to ignore value** to 3 since there are three taps in the Tx FFE.
- Set the **Rx model Bits to ignore value** to 20000 to allow sufficient time for the Rx DFE taps to settle during time domain simulations.
- Verify that **Both Tx and Rx** are set to Export and that all files have been selected to be generated (IBIS file, AMI files and DLL files).
- Set the **IBIS file name** to be `usb3_1_serdes.ibs`
- Press the **Export** button to generate models in the **Target directory**.

Add Jitter to AMI files

- Browse to the Target directory specified in the SerDes IBIS-AMI Manager and open the two .ami files in a text editor.
- In the Tx AMI file, add the following Reserved Parameters:

```
(Tx_Rj (Usage Info) (Type UI) (Range 0 0 0.012))

(Tx_Dj (Usage Info) (Type UI) (Range 0 0 0.17))
```

These ranges allow the user to fine-tune the jitter values to meet USB3.1 jitter mask requirements.

- In the Rx AMI file, add the following Reserved Parameters:

```
(Rx_Rj (Usage Info) (Type UI) (Range 0 0 0.015))

(Rx_Dj (Usage Info) (Type UI) (Range 0 0 0.3))

(Rx_Receiver_Sensitivity (Usage Info) (Type Float) (Range 0.025
0.015 0.1))
```

The jitter parameter ranges allow the user to fine-tune the jitter values to meet USB3.1 jitter mask requirements.

- Save and close both files.

Test Generated IBIS-AMI Models

The USB3.1 transmitter and receiver IBIS-AMI models are now complete and ready to be tested in any industry standard AMI model simulator.

References

USB, https://www.usb.org

IBIS 6.1 Specification, https://ibis.org/ver6.1/ver6_1.pdf