

SharkSim SPICE to IBIS User Guide

Version 2.2.3 July 2011

Table of Contents

Inside This Manual	7
Overview	7
SharkSim Documentation	7
Customer Support	7
Naming Conventions	8
SPICE to IBIS Translation.....	9
Overview.....	9
Creating the SPICE Model Wrapper.....	9
Spectre Support.....	10
Differential Support	10
Input	10
Output	10
Output with Enable or Input/Output	10
Single-Ended to Differential Output	11
Differential to Differential Output	11
Single-Ended to Differential Output with Enable	11
Differential to Differential Output with Enable	12
Series Switch.....	12
Series Switch with Enable.....	12
Steps to Create a SPICE Model Wrapper	12
Opening the SPICE to IBIS Translation GUI	15
Defining Pins	15
Select Buffer Type	15
Specifying Process Corner	16
SPICE File Name.....	16
SPICE Subcircuit Call.....	16
Pin Definitions	16
Enable Pin	16
Excluding Core Power and Ground Pins	16

Excluding Vref Pin	17
Enable Differential to Differential Outputs.....	17
Differential Output Load.....	17
Override Voltages	17
Override Voltages for Series Switch.....	18
Process Settings	19
IO Voltage.....	19
Core Voltage	19
Temperature	19
Vref Voltage.....	19
IO Voltage Variation.....	19
C_comp Extraction	20
Extraction Method	20
Create Extraction Netlists	20
Run SPICE	20
View Data	20
Simulation Options	20
IV Data Simulation	21
VT Data Time Window.....	21
Embedded Clock.....	21
VT Reference Load.....	21
Test Load Data.....	22
Select Topology	22
Select Load.....	22
Define Test Load.....	22
Create Netlists.....	22
Run SPICE	22
View Test Data	22
Define Test Load.....	23
Select Test Load.....	23

Test Load Parameters..... 25

Input Source 25

Generate IBIS Data 27

 Create IV Netlists..... 27

 Create VT Netlists..... 27

 View IV Data 27

 View VT Data 27

 Run SPICE 27

Save and Load States 28

 Save Window State 28

 Load Window State..... 28

Create Buffer Model 29

 Overview..... 29

 Opening the Create Buffer Model GUI..... 29

 Model Type..... 29

 Differential Model Types..... 31

 Series Switch Models..... 31

 Voltage..... 31

 Temp 31

 C_comp..... 31

 Vinh 31

 Vinl 32

 Vref 32

 Cref 32

 Rref 32

 Vmeas 32

 Defining the Vmeas Test Load 32

 Set Enable 33

 Polarity..... 33

 Define Model Spec 33

Add Submodel	33
Receiver Thresholds.....	34
Add Test Data	34
Driver Schedule	34
Load IBIS IV Data.....	34
Load IBIS VT Data.....	34
Include All Process Corners.....	34
Define VT Loads.....	34
Preview Waveforms.....	35
Create the .MOD File.....	35
New Model.....	35
Add IBIS-AMI Calls.....	36
Overview.....	36
Opening the IBIS-AMI GUI	36
Adding IBIS-AMI Calls.....	36
Compiler	36
DLL or SO File	36
AMI Param File	36
MOD File.....	36
Generate IBIS File	37
Overview.....	37
Opening the Generate IBIS File GUI.....	37
Configure Header.....	37
Company Name	37
Component/Part Number	37
IBIS Version	37
File Name.....	37
File Revision	37
Date.....	38
Source	38

Copyright Year..... 38

Notes 38

Disclaimer 38

Save and Load State..... 38

Create IBIS File Header..... 38

Configure Component and Package..... 38

 Signal Integrity Location 38

 Timing Location..... 39

 Manufacturer..... 39

 Global Package Pin Values 39

 Define Pins..... 39

 Pin Mapping..... 39

 Define Differential Pins 39

 Define Series Pins 39

 Model Selector..... 40

 Create IBIS Component and Package 40

Load .MOD Components..... 40

Generate IBIS File 40

Advanced Features..... 41

 Overview..... 41

 SPICE Integration 41

 HSPICE Integration and Support..... 41

 Other SPICE Application Support..... 43

Inside This Manual

Overview

This document describes the SPICE to IBIS add-on module for SharkSim. The SPICE to IBIS module allows the user to generate IBIS data from a SPICE model and format the data into a compliant IBIS file.

SharkSim Documentation

The SharkSim application documentation is divided into separate User Guides as listed below.

SharkSim Editor and Viewer Guide. Describes how to load and view IBIS simulation models as well as general application features.

SharkSim Quality Checker Guide. Describes how to use the Quality Checker feature to validate IBIS simulation models.

SharkSim SPICE to IBIS Guide. Describes how to translate SPICE to IBIS simulation models using the built-in translation GUI features and also how to build a full IBIS file.

SharkSim Simulation and Correlation Guide. Describes how to simulate IBIS buffer models and view the results. Also describes how to use existing [Test Data] to perform full correlation.

Customer Support

Customers can view the HTML help file any time from within the application by selecting the **Help** icon. The user guides are also in PDF format in the application release directory.

Customers can also visit www.sharksim.com to get full support for the application through additional tutorials, white papers, videos, forums, and contact information of the SharkSim developers.

Naming Conventions

Many different naming conventions are used when talking about the IBIS Specification. For all of the SharkSim User Guides the following conventions have been followed:

IBIS File. This refers to the overall IBIS file with a .ibs extension.

IBIS Buffer Model. This refers to the individual keyword [Model] types in the overall IBIS file. The buffer models as listed by the [Model] keyword are the actual sections where the IBIS data is stored for each [Model] type.

IBIS keywords [keyword]. The IBIS Specification specifies special keywords using the format [keyword] and the User Guides use the same format to highlight the fact that a keyword is being used.

Many engineers interchange ‘IBIS File’ and ‘IBIS Model’. Sometimes an engineer will be talking about the full IBIS file with .ibs extension yet will call it an IBIS model. The SharkSim User Guides have tried to distinguish between IBIS File and IBIS Buffer Model to make the terminology easier to understand.

SPICE to IBIS Translation

Overview

The SPICE to IBIS Translation section of the SPICE to IBIS module allows the user to generate IBIS data from a SPICE simulation model. The translation section supports single-ended and differential buffer model extraction as well as advanced features.

Creating the SPICE Model Wrapper

SharkSim allows the user to generate IBIS data from a SPICE simulation model. By default Synopsys HSPICE is supported but other SPICE simulation engines can be used as long as they are compatible with HSPICE. As of version 2.0 SharkSim also supports Mentor Graphics Eldo SPICE and Cadence Spectre SPICE as well.

In order for SharkSim to properly read in and use the SPICE simulation model the SPICE model needs to be put into a specific format and will be referred to as a ‘wrapper’. The user needs to create a subcircuit ‘wrapper’ around their current SPICE model.

The SPICE simulation model needs to have a subcircuit defined in the following general form:

```
.subckt subckt_name In Out Enable IOPower IOGround [CorePower] [CoreGround] [Vref]
*circuit definition and process data goes here
.ends subckt_name
```

Depending on the buffer type of the circuit some of the original nodes in the SPICE simulation model may not be needed. The subcircuit nodes have to be in the specified order for SharkSim to read it properly. Any node with a { } bracket is required and any node with a [] bracket is optional. For each buffer model generated a separate subcircuit wrapper is needed. There also needs to be a separate SPICE file for each process corner. The following supported buffer types have the node order listed out.

Spectre Support

All SPICE model wrappers in Cadence Spectre format need to begin and end with the ‘simulator lang=spectre’ command in order for SharkSim to properly read the netlists.

Differential Support

SharkSim supports SPICE to IBIS translation of differential buffers by treating all buffers as pseudo-differential designs. This makes the assumption that the differential current between the differential pins is negligible and the differential outputs can be modeled as single-ended outputs. This method works very well for most designs. SharkSim provides translation features for technologies such as LVDS and CML to properly extract the IBIS IV and VT data. However there can be some differential designs where this does not work due to the nature of the circuit design. It is always best to perform full correlation of the generated IBIS data to determine if the differential circuit is suitable for this translation method.

Input

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref] [Clk]
```

For an input buffer type the input, I/O power, and I/O ground nodes to be present. The same pin ordering applies to a differential input as SharkSim treats all differential signals as single-ended. The user may need to tie the inverting differential input pin to either logic high or low.

Output

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref] [Clk]
```

For an output buffer type the input, output, I/O power, and I/O ground nodes to be present. The same node order applies for an Open Sink output buffer.

Output with Enable or Input/Output

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out} {Enable} {IOPower} {IOGround} [CorePower] [CoreGround]
[Vref] [Clk]
```

For an output enabled buffer type the input, output, enable, I/O power, and I/O ground nodes to be present. The same node order applies for an Input/Output type buffer, a 3-State buffer, Open Sink 3-State buffer, and an Open Sink IO buffer.

Single-Ended to Differential Output

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out+} {Out-} {IOPower} {IOGround} [CorePower] [CoreGround]
[Vref] [Clk]
```

For a single-ended to differential output buffer type the single-ended input, non-inverting differential output, inverting differential output, I/O power, and I/O ground nodes to be present. The same node order applies for a Single-Ended to Differential Open Sink output buffer.

Differential to Differential Output

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In+} {In-} {Out+} {Out-} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref]
[Clk]
```

For a differential to differential output buffer type the non-inverting differential input, inverting differential input, non-inverting differential output, inverting differential output, I/O power, and I/O ground nodes to be present. The same node order applies for a Differential to Differential Open Sink output buffer.

Single-Ended to Differential Output with Enable

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out+} {Out-} {En} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref]
[Clk]
```

For a single-ended to differential 3-state output buffer type the single-ended input, non-inverting differential output, inverting differential output, enable, I/O power, and I/O ground nodes to be present. The same node order applies for a Single-Ended to Differential 3-State Open Sink output buffer.

Differential to Differential Output with Enable

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In+} {In-} {Out+} {Out-} {En} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref] [Clk]
```

For a differential to differential 3-state output buffer type the non-inverting differential input, inverting differential input, non-inverting differential output, inverting differential output, enable, I/O power, and I/O ground nodes to be present. The same node order applies for a Differential to Differential 3-State Open Sink output buffer.

Series Switch

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref] [Clk]
```

For a series switch type the input, output, I/O power, and I/O ground nodes to be present.

Series Switch with Enable

The SPICE simulation model subcircuit node order should be the following:

```
.subckt subckt_name {In} {Out} {En} {IOPower} {IOGround} [CorePower] [CoreGround] [Vref] [Clk]
```

For a series switch type with an enable the input, output, enable, I/O power, and I/O ground nodes to be present.

Steps to Create a SPICE Model Wrapper

The following example will show demonstrate how to create a subcircuit wrapper.

Create a top-level SPICE netlist file that can properly simulate the input or output buffer into the manufacturers specified test load. This netlist file should include all other necessary subcircuit calls and library/process calls if they are not embedded into the file. A separate file should be created for each of the three process corners including typical/minimum/maximum.

There should be one top level instance call that does not include a package model. An example could look like the following for an output buffer type:

```
Xdie Input_A Sense_B Output_C VCC GND buffer
```

In this instance call there is an input pin Input_A, an output pin Output_C, an I/O power pin VCC, an I/O ground pin GND, and an extra pin called Sense_B. The subcircuit call is buffer. From the specified output buffer protocol order we need an input, output, I/O power, and I/O ground pins. We have an extra node, Sense_B that is not needed for the IBIS translation. Also, the protocol specifies that the node order needs to be input than output as well. In the next section we will create a new subcircuit wrapper to address this.

A subcircuit wrapper should be created around the current subcircuit call with the correct format and ordering.

Here is the old top-level subcircuit call:

```
Xdie Input_A Sense_B Output_C VCC GND buffer
.subckt buffer Input_A Sense_B Output_C VCC GND
* circuit definitions and process info goes here
.ends buffer
```

Now we will create a new wrapper:

```
Xdie Input_A Output_C VCC GND buffer2

.subckt buffer2 Input_A Output_C VCC GND

Vsource Sense_B 0 DC 0
Xdie Input_A Sense_B Output_C VCC GND buffer

.ends buffer2

.subckt buffer Input_A Sense_B Output_C VCC GND

* circuit definitions and process info goes here

.ends buffer
```

We did two things here: we changed the node name order and set the extra node that was not needed to an appropriate logic level so as not interfere with the IBIS data generation. A new subcircuit **buffer2** was created and the previous subcircuit call **buffer** was invoked from within this subcircuit.

It is important to note that the Xdie call should be commented out once the subcircuit wrapper has been defined and tested. SharkSim will automatically create the top level instance calls in the generated SPICE files and will call the subcircuit directly. Leaving in any instance level calls to the same subcircuit will result in errors.

There are many different way to create subcircuit calls to format an existing SPICE netlist to work properly with SharkSim and this is only one recommended method.

Opening the SPICE to IBIS Translation GUI

The SPICE to IBIS Translation GUI can be opened from the navigation menu item **Generate | SPICE to IBIS Translation**.

Defining Pins

The first section of the SPICE to IBIS Translation GUI is the **Define Pins** section where the user defines the SPICE simulation model node names and subcircuit used in the SPICE model wrapper files.

Select Buffer Type

The user can select from eleven different buffer model types for the SPICE to IBIS translation. The options are:

Input: An input buffer.

Output: An output buffer (no tri-state).

3-State Output: An output buffer with tri-state (enable pin required).

Input/Output: An input/output buffer (enable pin required).

Open Sink Output: An output buffer with no pullup. (no tri-state)

Open Sink 3 State: An output buffer with no pullup with tri-state (enable pin required).

Open Sink IO: An input/output buffer with no pullup (enable required).

Diff Output: A differential output buffer (no tri-state).

Diff 3State Output: A differential output buffer with tri-state (enable pin required).

Diff Open Sink Output: A differential output buffer with no pullup (no tri-state).

Diff 3S Open Sink Out: A differential output buffer with no pullup with tri-state (enable pin required).

Series Switch: A series switch (analog switch) buffer.

Series Switch with Enable: A series switch (analog switch) buffer (enable pin required).

Specifying Process Corner

The **only TYP corner** checkbox allows the user to only define data for the Typical process and not for the minimum and maximum which is the default.

SPICE File Name

The user needs to specify the SPICE circuit file name for each process corner. The user can enter the full path manually or use **the Load SPICE files** button to automatically browse and select each file.

SPICE Subcircuit Call

The user needs to enter in the SPICE subcircuit call name for each process corner. The user can select the **same subcircuit call** button to copy the Typical data field to the Minimum and Maximum data fields if they are the same.

Pin Definitions

Depending on the buffer type that the user selected different Pin data fields will be highlighted for the user to enter pin node names into. For example, if the user selected an Input type than the Input Pin, IO Power Pin, and IO Ground Pin data fields need to be filled in. These pin name fields should match the SPICE subcircuit wrapper node names.

Enable Pin

If the buffer type includes an Enable pin the user also needs to specify whether it is active-high or active-low using the drop-down box.

Excluding Core Power and Ground Pins

If the SPICE subcircuit does not include optional Core Power and Core Ground pins the user must check the **exclude core power** checkbox.

Excluding Vref Pin

If the SPICE subcircuit does not include the optional Vref pin the user must check the **exclude vref power** checkbox.

Enable Differential to Differential Outputs

If the buffer type is a differential output that is a differential to differential design the user needs to check the **enable diff-diff** output checkbox. This will enable the Input Pin (inverting) for the inverting differential input pin node name. It will also enable the Diff Input Vhi and Diff Input Vlo data fields that should be filled in with the proper differential input voltages. For example, a standard LVDS differential input Vhi voltage would be 1.4 Volts and a Vlo voltage would be 1.0 Volts.

Differential Output Load

If the user selects a differential output buffer type the Differential Output Load section will need to be filled in. This section allows the user to specify a load to be placed on the inverting differential output pin. The non-inverting differential output pin will have a DC source on it that will be swept across voltage to generate the IV IBIS data.

By specifying a load on the inverting differential output pin the correct buffer operation can be kept constant while the IBIS data is generated. For technologies like LVDS that require feedback the load should be equivalent to the manufacturers test load into a non-differential load. For LVDS this is typically 50 Ohms to Vos. For technologies like CML the load would simply be 50 Ohms to Vcc.

Override Voltages

The user can enter a different voltage value for the input and enable pins rather than the default IO voltage value to be used in the IBIS IV data extractions. In some circuit designs like an integrated input/output buffer the input node might need a different voltage than the IO voltage to get the proper IV curves.

Override Voltages for Series Switch

When specifying a Series Switch device the Vds voltage must be entered ($V_{ds} > 0$) in the Override Voltages section.

Process Settings

The **Process Settings** section allows the user to define the IO buffer process settings needed for the IBIS data extraction.

IO Voltage

The user should define the IO voltage of the buffer for the process corners.

Core Voltage

The user should define the Core voltage of the buffer for the process corners if applicable.

Temperature

The user should define the IO temperature of the buffer for the process corners.

Vref Voltage

The user should define the Vref voltage of the buffer for the process corners if applicable.

IO Voltage Variation

The user should define the IO voltage variation of the buffer. For example, if the Minimum IO buffer voltage is 2.97 Volts, the Typical IO buffer voltage is 3.3 Volts, and the Maximum IO buffer voltage is 3.63 Volts the IO Voltage Variation would be 10%.

C_comp Extraction

SharkSim allows the user to select an extraction method to determine the buffer capacitance that is specified as C_comp in the IBIS specification.

Extraction Method

The user can select from the following extraction methods:

Transient Sawtooth: This method essentially uses a transient waveform to record the current and using $C = [I_{\text{current}} / (dV/dt)]$ the capacitance is extracted.

Frequency Sweep: This method uses an AC frequency sweep on the output sweeping the buffer capacitance and calculates C versus frequency and voltage. The default frequency sweep goes from 1 Mhz to 1 Ghz and an HTML table and 2D contour plot is generated for the user.

C_comp extraction is not supported for Series Switch buffer model types since IBIS Specification requires die capacitance to be included in Terminator buffers.

Create Extraction Netlists

This button will create the extraction netlists to be simulated with.

Run SPICE

This button will run SPICE with the C_comp extracted netlists.

View Data

This button will open a data viewer that will show the results of the C_comp extracted netlist simulations. The user has the ability to save both the data table and the waveform image for later use.

Simulation Options

The Simulation Options section allows the user to define the simulation settings to be used in the SPICE simulator.

IV Data Simulation

SharkSim allows the user to specify whether a **DC Analysis** or **Transient Analysis** should be used to generate the IV data curves. If the SPICE subcircuit wrapper includes an embedded clock the user must select the **Transient Analysis** option. It is recommended to start with the **DC Analysis** (default) and see if the simulation completes or not. Sometimes there can be convergence issues and using the **Transient Analysis** can overcome those issues. There should be no difference in the accuracy of the data based upon the simulation method used.

VT Data Time Window

For output buffer types that require VT data to be generated the user needs to specify the simulation length to be used to extract the data. Care should be taken in selecting this value as the VT data needs to reach steady state but the overall time window length should be less than $\frac{1}{2}$ clock period to avoid overclocking.

Embedded Clock

The user can specify the settings for the use of an embedded clock in the SPICE subcircuit wrapper by selecting the **enable clock** checkbox and filling in the following data fields:

Clock Pin: The node name of the clock pin as defined in the SPICE subcircuit wrapper.

Clock Delay: The initial delay of the clock signal.

Clock Edge: The edge rate of the clock signal.

Clock Pwidth: The pulse width of the clock signal.

VT Reference Load

For output buffer types that require VT data to be generated the user needs to specify the reference load to be applied to the output buffer pin.

R_load: This is the resistor that is used in the reference load. This should match the system impedance of the target system which is typically 50 Ohms.

V_typ/min/max: This is the reference voltage for the VT data and is typically Vcc but depends on the manufacturers test load and circuit design.

R_diff: By enabling the R_diff the user can specify a differential resistor be put across a differential output for the VT data to be generated.

Test Load Data

SharkSim allows the user to generate multiple sets of correlation data to be used in the IBIS file with the [Test Data] keywords. The Test Load Data section allows the user to select a test load, run the simulation, view the results, and save the data in the IBIS [Test Data] format that can be used later on when the IBIS file is generated.

Select Topology

The user can select either a **Single-Ended** or **Differential** topology. By selecting a Differential topology only the differential signal will be simulated. The user can select a Single-Ended topology for a differential output to generate the single-ended test load data for the differential output.

Select Load

The user can select from a **Standard Load** or a **Transmission Line Load**.

Define Test Load

See the following Define Test Load section for detailed information.

Create Netlists

By selecting this button the Test Load Data netlists will be generated.

Run SPICE

By selecting this button the Test Load Data netlists will be simulated in the SPICE simulator.

View Test Data

By selecting this button the Test Load Data results will be loaded into a data viewer. The user can view the results and save the data in IBIS [Test Data] format

as well as save screen shot images of the waveforms by right-clicking on the waveform image and selecting **Save as** option.

Define Test Load

The user can select the Define Test Load button under the Test Load Data section and a new GUI will be brought up allowing the user to specify a test load for the SPICE simulation model to be simulated into. The user can specify multiple loads but once a load has been defined the user needs to run the simulations and save the data. Each time a new load is defined the previous test load data is over-written.

Select Test Load

There are 27 different test load options to choose from. These are predefined test loads that are a subset of the available test load combinations allowed by the IBIS specification. This subset of test loads should accommodate almost any design including single-ended or differential buffers. Below are the different available loads to choose from:

Capacitive: A capacitor to ground.

RC Series: A series resistor and capacitor to ground.

RC Parallel: A resistor and capacitor to ground.

LC Series: A series inductor and capacitor to ground.

Pullup: A resistor to a voltage source.

Pullup with Cap: A resistor to a voltage source and a capacitor to ground.

Pulldown: A resistor to a voltage source.

Pulldown with Cap: A resistor to a voltage source and a capacitor to ground.

Thevenin: A resistor to a voltage source V_{term1} and resistor to a voltage source V_{term2} .

Rdiff: A differential resistor placed between the inverting and non-inverting output pins of a differential buffer.

TLine with Cap: An ideal lossless transmission line with a capacitor to ground.

TLine with Rcvr: An ideal lossless transmission line with an IBIS input buffer model.

Series TLine with Cap: A series resistor with an ideal lossless transmission line and a capacitor to ground.

Series TLine with Rcvr: A series resistor with an ideal lossless transmission line and an IBIS input buffer model.

TLine with RC Parallel: An ideal lossless transmission line with a resistor and capacitor to ground.

TLine with Pullup: An ideal lossless transmission line with a resistor to a voltage source.

TLine with Pullup and Cap: An ideal lossless transmission line with a resistor to a voltage source and a capacitor to ground.

TLine with Pullup and Rcvr: An ideal lossless transmission line with a resistor to a voltage source and an IBIS input buffer model.

TLine with Pulldown: An ideal lossless transmission line with a resistor to a voltage source.

TLine with Pulldown and Cap: An ideal lossless transmission line with a resistor to a voltage source and a capacitor to ground.

TLine with Pulldown and Rcvr: An ideal lossless transmission line with a resistor to a voltage source and an IBIS input buffer model.

TLine with Thevenin: An ideal lossless transmission line with a resistor to a voltage source V_{term1} and resistor to a voltage source V_{term2} .

TLine with Thevenin and Rcvr: An ideal lossless transmission line with a resistor to a voltage source V_{term1} and resistor to a voltage source V_{term2} and an IBIS input buffer model.

TLine with Rdiff: Two ideal single-ended lossless transmission lines with a differential resistor placed between the inverting and non-inverting output pins of a differential buffer.

TLine with Rdiff and Cap: Two ideal single-ended lossless transmission lines with a differential resistor placed between the inverting and non-inverting output pins of a differential buffer and a capacitor to ground on each differential output.

Input Test Load: An ideal voltage source (defined in Input Source section) with a series resistor and an ideal lossless transmission line connected to the SPICE input.

Black Box Load: Any SPICE compatible subcircuit load connected to the output buffer pin with an IBIS input buffer model.

Test Load Parameters

There are some extra test load parameters beyond the load variables (such as resistor R1, etc.) that need to be defined.

Test Data Name: The name of the [Test Data] that will appear in the IBIS file.

Test Load Name: The name of the [Test_load] that will appear in the IBIS file.

Driver Model: The name of the **Driver_model** that will appear in the IBIS file. This has to be the name of an output type buffer in the same IBIS file.

Receiver Model: If used this has to be the name of an input type buffer in an IBIS file that is specified in Receiver File.

Receiver File: If used this has to be a compliant IBIS file with the appropriate input type buffer specified.

Black Box Model: If used this has to be an HSPICE compatible subcircuit file with a .sp extension.

Input Source

The user must specify the input source parameters to be used as the input stimulus for the test load data generation.

Vlow: Input low voltage.

Vhigh: Input high voltage.

Delay: Input source delay time.

Edge: Input source edge rate. (Same for both rising and falling.)

Pulse Width: Input source pulse width.

Period: Input source period.

Sim Length: Overall test data generation simulation length.

Source: The user can select between three different input source types:

Edge: A rising and falling edge.

Pulse: A repeating 0101... pattern.

PRBS: A PRBS pattern as specified by user.

PRBS Pattern Load: The user can specify the PRBS pattern by entering it manually in the data field or by loading a .txt text file with only a '0' or '1' used for the pattern definition.

Generate IBIS Data

This section of the GUI creates the necessary SPICE simulation files to generate the IBIS data and view the sample data as well.

Create IV Netlists

By selecting this button the user will create the necessary SPICE files for simulation of the IBIS IV data.

Create VT Netlists

By selecting this button the user will create the necessary SPICE files for simulation of the IBIS VT data.

View IV Data

By selecting this button the user will be allowed to load the SPICE simulated data files and view the IV data.

View VT Data

By selecting this button the user will be allowed to load the SPICE simulated data files and view the VT data.

Run SPICE

By selecting this button the user will run the SPICE tool to simulate and generate the IBIS data.

Save and Load States

This section of the GUI creates a save state of the current GUI data fields and also allows the user to load the state file in the GUI.

Save Window State

By selecting this button the user will create an ASCII text state file that captures all of the current settings of the GUI window so the user can load this state file when the GUI is opened again and start where they left off from. This generated text file should not be manually modified by the user otherwise it will not load properly in the GUI.

SharkSim saves three different state files. One is for the SPICE to IBIS Translation GUI window. The second state file can be used in the Create Buffer GUI window so the same information does not have to be re-entered. The third state file can be used in the Set VT Loads GUI window under Create Buffer GUI so the same information does not have to be re-entered.

Load Window State

By selecting this button the user can load the state file when the GUI is opened again and start where they left off from.

Create Buffer Model

Overview

Once the IBIS data has been generated the user can use the Create Buffer Model GUI to automatically take the raw simulated data and create a modular component called a .mod file that contains the IBIS buffer model information under the keyword [Model]. Having an open and flexible format like this will allow users to easily re-use buffer components and share data when building complete IBIS files.

Opening the Create Buffer Model GUI

The Create Buffer Model GUI can be opened from the navigation menu item **Generate | Create Buffer Model**.

Model Type

Select from the drop-down box the type of IBIS buffer to be created. Note that these selections are not specific IBIS buffer types but the .mod model generated will be in an IBIS specific buffer type.

Input: An IBIS input model type with clamp IV data will be generated.

Output: An IBIS output model type with no clamp IV data but pullup/pulldown IV data and VT data will be generated.

3-State Output: An IBIS 3-State Output model type with pullup/pulldown/clamp IV data and VT data will be generated.

Input/Output: An IBIS Input/Output model type with pullup/pulldown/clamp IV data and VT data will be generated.

Open Sink: An IBIS Open Sink model type with no pullup IV but with pulldown IV and VT data will be generated.

Open Sink 3State: An IBIS Open Sink model type with no pullup IV but with pulldown/clamp IV and VT data will be generated.

Open Sink IO: An IBIS Open Sink IO model type with no pullup IV but with pulldown/clamp IV and VT data will be generated.

Open Source: An IBIS Open Source model type with no pulldown IV but with pullup IV and VT data will be generated.

Open Source 3State: An IBIS Open Source model type with no pulldown IV but with pullup/clamp IV and VT data will be generated.

Open Source IO: An IBIS Open Source IO model type with no pulldown IV but with pullup/clamp IV and VT data will be generated.

Input ODT: An IBIS input buffer type with clamp IV data including the termination will be generated.

Diff 3S Open Sink Out: An IBIS Open Sink output model type with no pullup IV but with pulldown/clamp IV and VT data will be generated as a single-ended output buffer.

Diff 3State Output: An IBIS 3-State Output model type with pullup/pulldown/clamp IV and VT data will be generated as a single-ended output buffer.

Diff Open Sink Output: An IBIS Open Sink output model type with no pullup/clamp IV but with pulldown IV and VT data will be generated as a single-ended output buffer.

Diff Output: An IBIS Output model type with no clamp IV but with pullup/pulldown IV and VT data will be generated as a single-ended output buffer.

Driver Schedule Top: An IBIS output or input/output model type with the [Driver Schedule] data added to create a top level model.

Driver Schedule Top: An IBIS output or input/output model type with the [Driver Schedule] data added to create a top level model.

Series Current Switch: An IBIS Series Current Switch model type with [Series MOSFET] IV curves only.

Series Switch: An IBIS Series Current Switch model type with [Series MOSFET] IV curves and [On] and [Off] keywords.

Terminator IV Input: An IBIS input model type with Ground Clamp and Power Clamp IV curves only.

Differential Model Types

All differential models generated by SharkSim are generated as single-ended buffer types according to the IBIS specification. This assumes that the inverting and non-inverting differential outputs are symmetrical as the same buffer will be used for both. The user can specify the same buffer in the [Diff Pin] section of the IBIS file so most commercial EDA tools will properly simulate the full differential buffer behavior.

Series Switch Models

Analog switches in IBIS are defined using [Series MOSFET] IV curves. Currently SharkSim only supports DC IV generation. There are two different methods available to generate Series Switch models using Terminator and Series Pin Mapping features. Please see Application Note #004 titled “IBIS Series Switch Support in SharkSim” for more specific information.

Voltage

The user should enter the IO buffer voltage that the IBIS data was generated under.

Temp

The user should enter the IO buffer temperature that the IBIS data was generated under.

C_comp

The user should enter the IO buffer capacitance.

Vinh

If applicable the user should enter the input high voltage for a receiving operation.

V_{inl}

If applicable the user should enter the input low voltage for a receiving operation.

V_{ref}

If applicable the user should enter the reference voltage for an output buffer according to the manufacturer's AC Timing test load.

C_{ref}

If applicable the user should enter the capacitance for an output buffer according to the manufacture's AC Timing test load.

R_{ref}

If applicable the user should enter the resistance for an output buffer according to the manufacture's AC Timing test load.

V_{meas}

If applicable the user should enter the voltage reference point for an output buffer according to the manufacture's AC Timing test load.

Defining the V_{meas} Test Load

The parameters V_{meas}/R_{ref}/C_{ref}/V_{ref} are used in IBIS output buffer types to define the manufacturer's AC timing test load. This is necessary for most commercial EDA simulation tools to properly account for board level flight time calculations that are used in system level timing calculations for setup and hold time. The IBIS specification requires V_{meas} to be present and a valid combination of the R_{ref}/C_{ref}/V_{ref} keywords. An example valid combination would be only C_{ref} as a capacitive load to ground. Another example would be R_{ref} and V_{ref} which would imply a resistive load to a reference voltage.

SharkSim will automatically put in the V_{meas} voltage of $\frac{1}{2} * VCC$ where VCC= Typical voltage as specified in the data field (if present), C_{ref} of 0 pF, V_{ref} of 0V, and R_{ref} of 10e9 Ohms if these data fields are left blank. It is recommended that

the user always fill in all of these fields manually to avoid the incorrect timing load being entered in the IBIS file.

Set Enable

If the buffer type requires an enable the user has to define whether it should be active-low or active-high.

Polarity

If the output buffer is an inverting output than the user can enable this checkbox and the keyword 'Polarity Inverting' will be added to the model data so an EDA simulator will know to invert the simulated output to match the actual device functionality.

Define Model Spec

The user can select this button to open a new GUI that will allow them to enter in values for the available Model Spec parameters allowed by the IBIS specification. The user can also save a set of Model Spec parameters and re-use them later for other buffer model creation by using the **Save** and **Load** buttons. To save the Model Spec parameter values to the actual .mod file to be generated the user needs to select the **Save to Model** button.

Add Submodel

The user can define a submodel call to be placed in the [Model] buffer by selecting the **Add Submodel** button. The user needs to enter the **Submodel name** and select the **Mode**.

SharkSim does not directly support the creation of a submodel [Model] call since it is essentially a selection of valid IBIS buffer model types but re-named as Sub Model. The user can create Sub Model calls using standard buffer types and then re-name them as Sub Model in the final IBIS file.

Receiver Thresholds

The user can define [Receiver Threshold] keywords for the model type by selecting the **Receiver Thresholds** button.

Add Test Data

The user can add as many .testdata files to a model type by selecting the **Add Test Data** button.

Driver Schedule

The user can define the [Driver Schedule] keyword for the model type by selecting the **Driver Schedule** button. A new GUI will open allowing the user to enter the model name and the turn on and turn off times for any number of buffer models in the overall IBIS file.

Load IBIS IV Data

The user can select HSPICE DC Data, HSPICE AC Data, or Raw Text Data in the **IV Data** drop-down box and select the **Load IV Data** button to browse and upload the necessary files.

Load IBIS VT Data

The user can select either HSPICE AC Data or Raw Text Data in the **VT Data** drop-down box and select the **Load VT Data** button to browse and upload the necessary files. Only one set of VT .lis files can be uploaded for a model.

Include All Process Corners

The user can select the **include min/max corners** checkbox to use all process corner data.

Define VT Loads

Once the user has loaded the VT Data the user must select the **Define VT Loads** button to define the VT load data.

R_fixture: The resistor used to generate the VT data.

V_fixture: The Rising and Falling Waveform to VCC voltage values used for generating the VT data.

V-T Curve start points: The user can specify when to start using the data for each process corner VT data set. This is useful to cut out initial lead-in time that is not necessary.

V-T Curve Length: The user can specify the overall VT time window to use the VT data.

Preview Waveforms

Once the necessary IV and VT data has been loaded and defined the user can select the Preview Waveforms button to view the IV, VT, and Test Data waveforms.

Create the .MOD File

By selecting the **OK** button the user will create the .mod file for the IBIS data loaded. A popup message will appear when the .mod file has been created.

New Model

By selecting this button the user will clear the current screen and can start entering data for a new .mod file generation. This button must be selected after each .mod generation or the data will be appended causing errors.

Add IBIS-AMI Calls

Overview

Once the IBIS data has been formatted into the .MOD file component the user can add IBIS-AMI calls to the existing .MOD file.

Opening the IBIS-AMI GUI

The Add IBIS-AMI Calls GUI can be opened from the navigation menu item **Generate | Add IBIS-AMI Calls**.

Adding IBIS-AMI Calls

The IBIS Specification allows under a [Model] section to include calls to an external IBIS-AMI file which includes an executable file (DLL for Windows and SO for Linux) and a parameter file (.ami extension). The user can add the compiler version, the DLL/SO file name, and the .ami parameter file name to an existing .MOD file.

The GUI allows the user to enter only one IBIS-AMI call at a time. To add both a Windows and a Linux call the steps need to be repeated using the GUI.

Compiler

Full version of the compiler used to compile the IBIS-AMI executable file.

DLL or SO File

The name of the DLL or SO file including extension.

AMI Param File

The name of the .ami parameter file including extension.

MOD File

The full path and name of the .MOD file to add the IBIS-AMI call to.

Generate IBIS File

Overview

Once the IBIS data has been formatted into the .MOD file component the user can use the Generate IBIS file GUI to combine as many .MOD files together along with the other necessary IBIS header information to create a complete IBIS file.

Opening the Generate IBIS File GUI

The Create Buffer Model GUI can be opened from the navigation menu item **Generate | Generate IBIS File**.

Configure Header

By selecting the **Configure Header** button the user can fill out the necessary information to be placed in the header section of the final IBIS file.

Company Name

Name of the company for the IBIS file.

Component/Part Number

The user can put in a generic or specific component part number.

IBIS Version

The current default IBIS version is 4.2 but the user has the option to select from previous versions 4.0 and 4.1 for compatibility with older CAD tools as well as the new 5.0 version.

File Name

The file name of the final IBIS file. This will also be the actual .ibs file name as well.

File Revision

The revision number of the IBIS file.

Date

The data when the IBIS file was generated.

Source

The user can select from four options: SPICE to IBIS Translation, Lab Measurement, Datasheet or Other.

Copyright Year

Current year for copyright considerations.

Notes

The user can enter as many notes as desired to leave comments for the end user about IBIS model usage and any issues.

Disclaimer

The user can put in a disclaimer in this section. The user can also load a disclaimer from a text .txt file by using the **Load Disclaimer** button.

Save and Load State

The user can save the current state of the GUI window to be used later by using the load state button.

Create IBIS File Header

By selecting the **Done** button the IBIS File Header information will be saved.

Configure Component and Package

By selecting the **Configure Component and Package** button the user can fill out the necessary information to be placed in the component and package section of the final IBIS file.

Signal Integrity Location

The IBIS specification allows the option of either the pin or die for waveform measurements.

Timing Location

The IBIS specification allows the option of either the pin or die for timing measurements.

Manufacturer

The name of the company of the device.

Global Package Pin Values

The user can enter in the R_pkg/L_pkg/C_pkg pin parasitic values that will be used globally for all pins in the IBIS file.

Define Pins

The user can load a .pin file that contains the IBIS pin list and pin parasitic values by using the **Load Pin File** button. An example .pin file is located in the release directory.

Pin Mapping

The user can load a .pinmap file that contains the IBIS pin mapping by using the **Load Pin Map File** button. An example .pinmap file is located in the release directory.

Define Differential Pins

The user can load a .diffpin file that contains the IBIS differential pin list declaration by using the **Load Diff Pin File** button. An example .diffpin file is located in the release directory.

Define Series Pins

The user can load a .seriespin file that contains the IBIS series pin list by using the **Load Series Pin File** button. An example .seriespin file is located in the release directory.

Model Selector

The user can load a .modelselect file that contains the IBIS [Model Selector] declarations by using the **Load Model Select File** button. An example .modelselect file is located in the release directory.

Create IBIS Component and Package

By selecting the **Done** button the IBIS Component and Package information will be saved.

Load .MOD Components

The user can select the Load .mod Components button to browse and select as many .mod component files as desired to be included in the complete IBIS file. By selecting the **Clear** button the user can remove the .mod component to be added to the complete IBIS file.

Generate IBIS File

By selecting the Generate IBIS File the user can generate the complete IBIS file. A popup message will appear stating that the IBIS file has been saved.

Advanced Features

Overview

This section describes some of the advanced features of the SharkSim application in further detail.

SPICE Integration

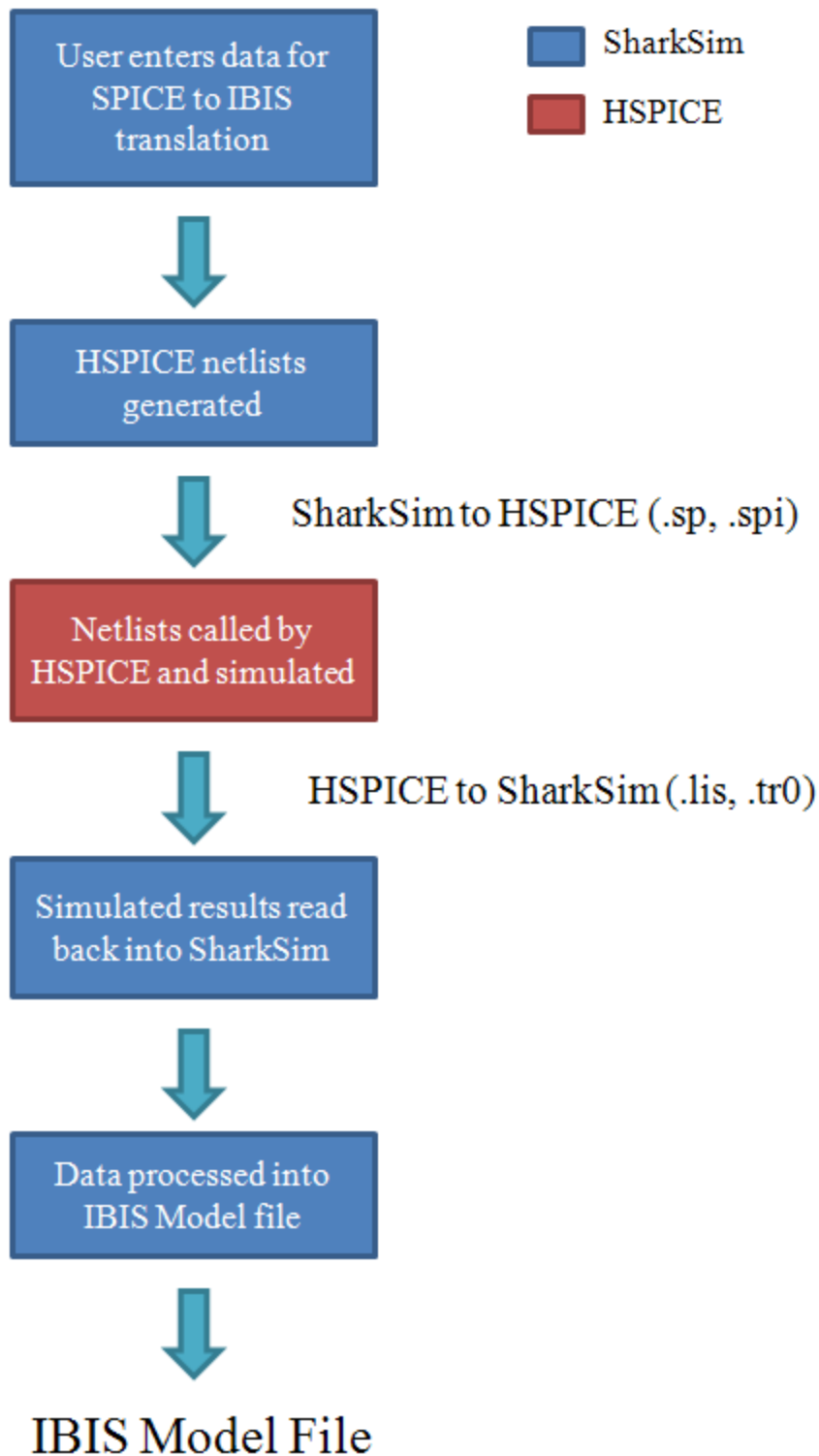
The SharkSim application allows the user to call external SPICE applications to take advantage of some of the advanced features in the application such as the SPICE to IBIS translation features and the IBIS Simulation and Correlation features. SharkSim supports Synopsys HSPICE by default and can be configured to support other SPICE applications as well. A separate license is required for all external SPICE applications.

HSPICE Integration and Support

The SharkSim application can use the Synopsys HSPICE engine to automatically create HSPICE netlists, run simulations, and then extract data from the .lis files and process the data into a formatted IBIS file.

The overall flow is that SharkSim generates different types of HSPICE netlist files in .sp format and will then call the HSPICE simulator externally to simulate the generated netlists. The output data files in .lis format will be read in by the SharkSim application and the data will be processed and formatted in various model formats.

The flow diagram below describes the interoperability between SharkSim and Synopsys HSPICE for the generation of IBIS files.



Other SPICE Application Support

The SharkSim application can use any external SPICE application to run simulations and process the data into various model formats. The user can set the SPICE application executable path in the **Preferences** section of the application.

Currently SharkSim supports Synopsys HSPICE compatible executable calls and formats by default. With version 2.0 Mentor Graphics Eldo and Cadence Spectre is also supported. If support is needed for a non-compatible SPICE application the user should contact SharkSim support.