**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**Draft 3**

**ISSUE TITLE:** IBIS-AMI New Reserved Parameters for Dependency Tables

**REQUESTOR:**  Walter Katz, Todd Westerhoff, SiSoft, Adge Hawes, IBM

**DATE SUBMITTED:** *September 24, 2012*

**ANALYSIS PATH/DATA THAT LED TO SPECIFICATION:**

Model developers and EDA vendors building IBIS-AMI models using the IBIS 5.0 specification have come across a number of modeling issues that are not addressed in IBIS 5.0. In order to deliver models and EDA tools that meet end-user demands for model accuracy and functionality, EDA vendors have defined "extensions" to add new capabilities to IBIS-AMI models. Unfortunately, EDA vendors have had to use proprietary (and different) syntax to add these capabilities to models, limiting model portability between different EDA tools.

This BIRD proposes new syntax for the .ami control file that improves model functionality and accuracy. Including this syntax in the IBIS standard will allow creation of accurate, compliant IBIS-AMI models that are readily portable between commercial EDA simulators.

Dependency Tables are Model\_Specific parameters that are defined with format Table, and Usage Info. The parameter name shall end in “Dependency\_Table”.

Dependency Tables define a relationship between one or more input parameters and one or more output parameters. Note that in this context, "input parameter" and "output parameter" refer to inputs and outputs of the relationship being defined, not to whether the parameters in question are used as inputs or outputs by the algorithmic model itself. Dependency Tables make IBIS-AMI models easier for the systems designer by allowing complex relationships between related parameters to be automatically defined as part of the model.

**ANY OTHER BACKGROUND INFORMATION:**

This BIRD is being requested by the following IBIS users and model developers, in conjunction with the authors:

Cisco Systems: Upen Reddy, Doug White

ANALYSIS PATH/DATA THAT LED TO SPECIFICATION

The new Dependency Table defined in this BIRD came from commercial IBIS-AMI model development efforts where new functionality was needed to meet customer expectations for model functionality, accuracy and performance.

## New Usage Dependency\_Table

Dependency Tables are used to specify relationships between different AMI parameters. This is desirable because sometimes the value to be used for an AMI parameter is dependent on the selected value of another AMI parameter. For example, consider the case of an output buffer with a user-selectable output strength setting. The strength selected by the user affects both the output’s voltage swing and impedance.

Let’s assume that the output strength is controlled by a Model\_Specific parameter "Tx\_Strength" that has 8 discrete settings from 0 to 7, with a nominal value of 4:

(Tx\_Strength (Usage In)(Type Integer)(Range 4 0 7)

 (Description "Output buffer strength setting")

)

The TX analog buffer’s impedance is specified using the equivalent circuit analog model parameters. The output impedance values for each of the 8 output strength settings are:

(Rs (Usage Info)(Type Float)

 (List 45.0 46.0 48.0 50.0 52.0 50.0 48.0 45.0)

 (Description "Tx output impedance")

)

The Tx analog buffer’s output swing is also specified using the equivalent circuit analog model parameters. The output voltage values for each of the 8 output strength settings are:

(Voh (Usage Info)(Type Float)

 (List 0.40 0.42 0.44 0.46 0.48 0.50 0.52 0.54)

 (Description "Tx output voltage")

)

The model user is to be presented with a single control parameter, Tx\_Strength, with the values of Rs and Voh automatically selected by the EDA tool based on the value the user specifies for Tx\_Strength. Dependency Tables specify relationships between parameters to make this possible.

Input parameters are placed in the left-most columns and output parameters are placed in the right-most columns of the table. The Labels row lists the Parameter names used in the table and explicitly defines whether each Parameter is used as an Input ("In") to the table, or is an output ("Out\_\*") whose value is derived from the table. The Labels row is required for Dependency Tables.

For the example just described, one possible Dependency Table declaration is:

(Tx\_Strength\_Dependency

(Usage Dependency\_Table)

(Parameter\_Names Tx\_Strength Rs Voh)

(Column\_Types In Out\_Match Out\_Match)

(Description “Rs and Voh dependency on Tx\_Strength”)

(Table

(0 45.0 0.40)

 (1 46.0 0.42)

 (2 47.0 0.44)

 (3 50.0 0.46)

 (4 52.0 0.48)

 (5 50.0 0.50)

 (6 48.0 0.52)

 (7 45.0 0.54)

)

)

This tells the EDA tool that a Tx\_Strength setting of 0 corresponds to Rs = 45.0 ohms, Voh = 0.40V, a setting of 1 corresponds to Rs = 46.0 ohms, Voh = 0.42V, and so on.

Note that Type is not required to specify the column types in the Dependency Table since the Type is automatically determiniuned by the Type of the Parameter assigned to each Column in the Parameter\_Names leaf.

The following rules apply to the use of Dependency Tables:

1. The Parameter names used in the Dependency Table must already be declared in the .ami file, either in the Reserved\_Parameter or Model\_Specific sections
2. The values listed in the table for each Parameter must be a legal Format for that Parameter based, and should, but is not required, to be a legal value based on the definition of the Parameter in the .ami file.
3. Dependency Tables can only be declared in the Model\_Specific section of the .ami file
4. A Dependency Table defines the relationship between one or more input parameters and one or more output parameters.
5. Two or more Dependency Tables can be linked together (i.e. output variables in one table can be used as input variables in another table) to specify complex relationships
6. Dependency Tables are evaluated in the order they are declared in the .ami file.

The following elements define a Dependency Table:

• Usage is Dependency\_Table

• Parameter\_Names Parameter name of each column

• Column\_Types Column usage declaration

• Data rows

Parameter\_Names declares the parameters defined in the table. Column\_Types declares whether the parameters are inputs or outputs to the relationship being defined. A parameter is defined as an input to the table if its column type is In. A parameter is defined as an output from the table if its column type is Out\_Match, Out\_Closest, Out\_Range or Out\_PWL.

The table data rows contain the input parameter values that, when matched according to the defined criteria, provide the values to be used for the output parameters.

Example:

(Tx\_Strength (Usage In)(Type String)

 (List "0" "1" "2" "3" "4" "5" "6" "7")

 (Description "Output buffer strength setting")

)

(Rs (Usage Info)(Type Float)

 (List 45.0 46.0 48.0 50.0 52.0 50.0 48.0 45.0)

 (Description "TX output impedance")

)

(Voh (Usage Info)(Type Float)

 (List 0.40 0.42 0.44 0.46 0.48 0.50 0.52 0.54)

 (Description "TX output voltage")

)

(Tx\_Strength\_Dependency (Usage Dependency\_Table)

(Parameter\_Names Tx\_Strength Rs Voh)

(Column\_Types In Out\_Match Out\_Match)

(Description “Rs and Voh dependency on Tx\_Strength”)

(Table

(0 45.0 0.40)

(1 46.0 0.42)

(2 47.0 0.44)

(3 50.0 0.46)

(4 52.0 0.48)

(5 50.0 0.50)

(6 48.0 0.52)

(7 45.0 0.54)

)

)

The Dependency Table contains as many data rows as needed to specify all the combinations of input parameters to be tested (matched) and their corresponding output parameter values.

If there is no row that matches the values in the dependent parameters then the value of the dependent parameters shall be their default value as specified in their parameter declarations.

Dependency Tables support one or more columns of input parameter data and one or more columns of output parameter data. Except for the last Input column, the values of the Input Parameters must exactly match the values listed in a data row to select the Output Parameter values for that row. The last Input column in a Dependency Table (or the only Input column in the single Input case) is treated differently, however, in that piecewise linear interpolation and non-exact value matches are supported on the Input values listed in that column.

The type of special treatment used when matching the last column of input parameters is specified with the Output parameter, and as such, the method of matching used can be different for each output parameter.

The types of parameter matching supporting on the last input parameter column

are:

Out\_Match

Output parameter value is set to the row where the last input parameter column is an exact string match or it is a numeric match to within floating Point resolution. If no matching criteria are met, the output parameter is set to the default value as defined above.

Out\_Closest

Output parameter value is set to the row where the last input parameter column matches closest numerically to the value of that parameter. This requires the last input parameter column to contain numeric data. If there are two equal choices, the larger value will used for selection. If it does not contain numeric data, Out\_Match will be used instead. If no matching criteria are met, the output parameter is set to the default value as defined above.

Out\_Range

Output parameter value is set to the row where the last input parameter column is less than or equal to the value of that parameter and the next numerically larger row is greater than that value. If there is not a next numerically larger row, then the output parameter value is set to the row where the last input parameter column is less than or equal to the value of that parameter. This requires the last input parameter column to contain numeric data. If it does not contain numeric data, Out\_Match will be used instead. If no matching criteria are met, the output parameter is set to the default value as defined above.

Out\_PWL

Output parameter value is set to the linear interpolated value between the row where the last input parameter column is less than or equal to the value of that parameter and the next numerically larger row is greater than that value. If there is not a next numerically larger row, then the output parameter value is set to the extrapolated value using the next numerically smaller row. If this row does not exist, then the output parameter value is set to the value where the last input parameter column is less than or equal to the value of that parameter. This requires the last input parameter column to contain numeric data. If it does not contain numeric data, Out\_Match will be used instead. If no matching criteria are met, the output parameter is set to the default value as defined above.

As an example, let’s return to the example where the TX output buffer has a variable strength setting, but instead of 8 values, we’ll increase the number of strength selections to 71:

(Tx\_Strength (Usage In) (Type Integer) (Range 35 0 70)

 (Description "Output buffer strength setting")

)

Note that we could have defined Tx\_Strength as a list with 71 values, but that’s impractical - it’s more convenient to define it as a range of Integers.

Let’s assume that Rs and Voh have the same minimum and maximum values as before, such that it makes sense to specify them as Format Range as well:

(Rs (Usage Info) (Type Float) (Range 48.0 45.0 52.0)

 (Description "TX output impedance")

)

(Voh (Usage Info) (Type Float) (List 0.46 0.40 0.54)

 (Description "TX output voltage")

)

It would be possible to define a Dependency Table for all 71 possible settings of Tx\_Strength, but let’s assume that, in this case, it’s just as accurate to specify the values of Rs and Voh as piecewise-linear data spaced in increments 10 units of Tx\_Strength apart:

(Tx\_Strength\_Dependency

(Usage Dependency\_Table)

(Parameter\_Names Tx\_Strength Rs Voh)

(Column\_Types In Out\_PWL Out\_PWL)

(Description “Rs and Voh dependency on Tx\_Strength”)

(Table

( 0 45.0 0.40)

(10 46.0 0.42)

(20 47.0 0.44)

(30 50.0 0.46)

(40 52.0 0.48)

(50 50.0 0.50)

(60 48.0 0.52)

(70 45.0 0.54)

)

)

This tells the EDA tool that a Tx\_Strength setting of 15 corresponds to Rs = 46.5 Ohms, Voh = 0.43V, a setting of 27 corresponds to Rs = 49.1 ohms, Voh = 0.434V, and so on.

"Corner" is an internal IBIS parameter that is determined by the EDA Tool for each model instance. This is the same thing that is used by the EDA tool to determine the values of parameters, I-V and V-T tables, and External Models.

In AMI models Corner, shall be treated as (Type String) and have values “Typ”, “Min” and “Max”, that can be used to create an input column in a Dependency Table. Its column shall have (Type String). Its allowed values are “Typ”, “Min” and “Max”. “Typ” represents typical operating conditions, “Min" describes slow, weak performance, and "Max" describes the fast, strong performance.

(Tx\_Strength\_Dependency

(Usage Dependency\_Table)

(Type String Float Float)

(Parameter\_Names Corner Rs Voh)

(Column\_Types In Out\_Match Out\_Match)

(Table

(“Min” 45.0 0.40)

(“Typ” 46.0 0.42)

(“Max” 47.0 0.44)

 )

)

"bit\_time" is a intrinsic AMI parameter of Type Float and Usage Info that can be used to create an input column for data rate in a Dependency Table. Its Value is equal to the length of one data bit in seconds.

"BAUD" is a intrinsic AMI parameter of Type Float and Usage Info that can be used to create an input column for data rate in a Dependency Table. It’s Value is equal to 1/[bit\_time].

"GBAUD" is a intrinsic AMI parameter of Type Float and Usage Info that can be used to create an input column for data rate in a Dependency Table. It’s Value is equal to 1/([bit\_time] \* 1e9).

"[Model]" is a intrinsic AMI parameter of Type String and Usage Info that can be used to create an input column to declare which IBIS [Model] each row refers to in a Dependency Table.