**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD NUMBER:** 186.3 Draft 9e

**ISSUE TITLE:** File Naming Rules

**REQUESTOR:**  Walter Katz, Mike LaBonte, Signal Integrity Software, Inc.; Bob Ross, Teraspeed Labs

**DATE SUBMITTED:** November 29, 2016

**DATE REVISED:** February 16, 2017; April 14, 2017; TBD

**DATE ACCEPTED:**

**DEFINITION OF THE ISSUE:**

IBIS AMI modeling and Interconnect modeling may result in a large number of supporting files. This BIRD removes the restriction that all files referenced by the .ibs file be restricted to the directory containing the IBIS file. Instead, files referenced by the .ibs file shall be either in the same directory as the .ibs file or in a directory structure below the directory containing the .ibs file.

In addition, this BIRD relaxes other constraints on file names, including allowing both upper and lower case letters and allowing longer file names.

The IBIS specification must meet these requirements:

|  |  |
| --- | --- |
| Requirement | Notes |
| 1. Enable file names that are up to 256 characters in length.
 | IBIS in general will need to increase the number of characters permitted in each line. |
| 1. Enable files to be defined and located in a Hierarchical structure either in the same directory of the IBIS file or in a directory directly below it.
 | AMI supporting files and Interconnect model supporting files can be organized in a directory structure that makes it simple for IC vendors to organize and deliver their IBIS models to their customers. |
| 1. Enable file names with both lower and upper case characters.
 | IBIS should allow file naming rules similar to other modern tools and specifications. |
| 1. Enable file names to include a relative directory.
 |  |
| 1. Use consistent language to describe the parts of file names and paths.
 |  |

**SUMMARY OF PROPOSED CHANGES:**

For review purposes, the proposed changes are summarized as follows:

|  |  |  |
| --- | --- | --- |
| Specification Item | New/Modified/Other | Notes |
| Define terminology for the parts of file names in GENERAL SYNTAX RULES AND GUIDELINES. | New | Boost.org offers one source. Include an illustration. |
| Review 44 instances of “file name”, changing to “file” in some places and clarifying where relative paths are allowed. | Modified | Many places. |
| Change the file naming rules and IBIS File line length limits in GENERAL SYNTAX RULES AND GUIDELINES | Modified |  |

**PROPOSED CHANGES:**

*All page numbers refer to the IBIS version 6.1 Adobe PDF file.*

*----*

*Page 9, under* **3 GENERAL SYNTAX RULES AND GUIDELINES**

*Replace:*

This section contains general syntax rules and guidelines for ASCII .ibs files:

*With:*

*Unless noted otherwise, these subsections contain general syntax rules and guidelines for ASCII IBIS file types defined in this document. They are .ibs (Section 4), .pkg (Section 7), .ebd (Section 8), .ims (Section 12???), files and where applicable .am (Section 10.3)i and parameter passing files (Section 6.3).*

*Page 9, Add:*

## 3.1 FILE NAMING DEFINITIONS

The following terms related to file naming and file referencing for all ~~ASCII~~ formats are defined ~~here and for non-ASCII executable model files defined in the IBIS-AMI sections~~:

* **file name**: The name of a file without its location.
* **stem**: The portion of a file name before the last period, or the full file name if no period.
* **extension**: The portion of a file name after the last period, if any.
* **directory**: A directory contains a list of files. Directories may include other directories, forming the basis for a hierarchical filesystem.
* **path**: A sequence of root directory (optional), directory elements and file name that identify the location of a file. A path may be absolute or relative.
* **absolute path**: A path that unambiguously identifies the location of a file without reference to an additional starting location.
* **relative path**: A path that is not absolute, and so only unambiguously identifies the location of a file when resolved relative to an implied starting location.
* **root name**: For operating systems supporting multiple filesystem roots, a name to identify the filesystem.
* **root directory**: A standard designation for the root of a filesystem.
* **file reference**: A reference to a file, expressed as either a simple file name or a relative path, which includes a simple file name.

Figure 1 shows an example of a file path with its parts delineated.



1. - Example of file naming definitions

## 3.2 SYNTAX RULES

*On page 9,*

*Replace:*

1. To facilitate portability between operating systems, file names used in a .ibs file must only have lower case characters. File names should have a basename of no more than forty (40) characters followed by a period (“.”), followed by a file name extension of no more than three characters. The file name and extension must use characters from the set (space, “ ”, 0x20 is not included):

a b c d e f g h i j k l m n o p q r s t u v w x y z

0 1 2 3 4 5 6 7 8 9 \_ ^ $ ~ ! # % & - { } ) ( @ ‘ `

The file name and extension are recommended to be lower case on systems that support such names.

*With:*

1. File names should have a stem of no more than sixty (60) characters followed by a period (“.”), followed by a file name extension. The file name and extension must use characters from the set (space, “ ”, 0x20 is not included):

a b c d e f g h i j k l m n o p q r s t u v w x y z

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

0 1 2 3 4 5 6 7 8 9 \_ ^ $ ~ ! # % & - { } ) ( @ ‘ ` .

The character sequence “./” is not permitted in any reference to an IBIS file or to any other file format, effectively restricting the naming of files to those in the same directory as the referring file or a subdirectory of that directory. Absolute paths, those beginning with a root name or root directory, are not permitted in a reference to any file.

*DO EXTERNAL FILES (.ISS, .SP, .TS, .V, ETC. HAVE THE STEM LENGTH RESTRICTION, OR ARE THEY EXCLUDED? –PROBABLY NO STATEMENT IS NEEDED, AND THIS IS NOT CHECKED AND MAY BE EXPLICITLY EXCLUDED FOR SOME FILE TYPES*

*On page 9*

*Replace:*

4 A line of the file may have at most 120 characters, followed by a line termination sequence. The line termination sequence must be one of the following two sequences: a linefeed character or a carriage return followed by linefeed character.

*With:*

1. Except for .ami files, a line of the file may have at most 1024 characters, followed by a line termination sequence. The line termination sequence must be one of the following two sequences: a linefeed character or a carriage return followed by linefeed character.

*DOES THIS RULE APPLY TO .AMI FILES? Yes*

*On page 10,*

*Replace:*

1. The use of tab characters is legal, but they should be avoided as much as possible. This is to eliminate possible complications that might arise in situations when tab characters are automatically converted to multiple spaces by text editing, file transferring and similar software. In cases like that, lines might become longer than 120 characters, which is illegal in .ibs files.

*With:*

1. The use of tab characters is legal, but should be avoided as much as possible. This is to eliminate possible complications that might arise in situations when tab characters are automatically converted to multiple spaces by text editing, file transferring and similar software. In cases like that, lines might become longer than 1024 characters, which is illegal in IBIS file types (except for .ami files).

*On page 10,*

*Replace:*

14. Only ASCII characters, as defined in ANSI Standard X3.4-1986, may be used in IBIS file types. This includes files with file extensions .ibs, .pkg, .ebd, .ami and any other files used for passing parameter values.

*With:*

14. Only ASCII characters, as defined in ANSI Standard X3.4-1986, may be used in IBIS file types and in other plain text files referenced by IBIS.

*On page 11 renumber “***3.1 KEYWORD HIERARCHY***” to “***3.3KEYWORD HIERARCHY***”.*

*--------------------------------------------------------------------------------*

*On page 18,*

*Replace:*

*Keyword:* **[File Name]**

*Required:* Yes

*Description:* Specifies the name of the .ibs file.

*Usage Rules:* The file name must conform to the rules in paragraph 3 of Section 3, "GENERAL SYNTAX RULES AND GUIDELINES". In addition, the file name must use the extension “.ibs”, “.pkg”, or “.ebd”. The file name must be the actual name of the file.

*With:*

*Keyword:* **[File Name]**

*Required:* Yes

*Description:* Specifies the file name of the file containing this keyword

*Usage Rules:* The file name must conform to the rules in item 3 of Section 3.2, "SYNTAX RULES". In addition, the file name shall use the extension “ibs”, “pkg”, “ebd”, or ”ims”. The file name shall be the actual name of the file.

---------------------------------------------------------------

*On page 99, for [External Model] Parameters, change (delete period (.) for correct extension definition, change file\_name to file\_reference in several locations, and “a the” to “a”):*

*Replace:*

Corner:

Three entries follow the Corner subparameter on each line:

corner\_name file\_name circuit\_name

The corner\_name entry is “Typ”, “Min”, or “Max”. The file\_name entry points to the referenced file in the same directory as the .ibs file.

Up to three Corner lines are permitted. A “Typ” line is required. If “Min” and/or “Max” data is missing, the tool may use “Typ” data in its place. However, the tool should notify the user of this action.

Models instantiated by corner\_name "Min" describe slow, weak performance, and models instantiated by corner\_name "Max" describe fast, strong performance.

The circuit\_name entry provides the name of the circuit to be simulated within the referenced file. For SPICE and IBIS-ISS files, this is normally a “.subckt” name. For VHDL-AMS files, this is normally an “entity(architecture)” name pair. For Verilog-AMS files, this is normally a “module” name.

No character limits, case-sensitivity limits or extension conventions are required or enforced for file\_name and circuit\_name entries. However, the total number of characters in each Corner line must comply with the rules in Section 3. Furthermore, lower-case file\_name entries are recommended to avoid possible conflicts with file naming conventions under different operating systems. Case differences between otherwise identical file\_name entries or circuit\_name entries should be avoided. External languages may not support case-sensitive distinctions.

Parameters:

Lists names of parameters that can be passed into an external model file. Each Parameters entry must match a name or keyword in the external file or language. The list of Parameters may span several lines by using the word Parameters at the start of each line. The Parameters subparameter is optional, and the external model must operate with default settings without any Parameters assignments.

Parameter passing is not supported in SPICE. VHDL-AMS and VHDL-A(MS) parameters are supported using “generic” names, and Verilog-AMS and Verilog-A(MS) parameters are supported using “parameter” names. IBIS-ISS parameters are supported for all IBIS-ISS parameters which are defined on the subcircuit definition line.

Parameters are locally scoped under each [External Model] keyword, i.e., the same parameter under two different [External Model] will have independent values.

The parameter(s) listed under the Parameters subparameter may optionally be followed by an equal sign and a numeric, Boolean or string literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file name, followed by an open parentheses and a the tree root name, a new open parentheses for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file name. The file reference may point to any file which contains one or more parameter trees. The files referenced must be located in the same directory as the .ibs file containing the reference. The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, files with no extensions (e.g, xyz) or with just a dot (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, and .ebd) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “.ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Model] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “.ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “.ami”.
3. Usage Dep parameters may also be referenced in an AMI parameter definition file under the following conditions:
* the [External Model] keyword is located under a [Model] keyword which also contains an [Algorithmic Model] keyword,
* the [External Model]'s parameter and the [Algorithmic Model] keyword point to the same “.ami” file,
* the AMI parameter definition file contains the parameter AMI\_Resolve\_Exists with a value of True.

If all of these conditions are satisfied, the EDA tool must execute the AMI\_Resolve function in the executable model defined by the [Algorithmic Model] keyword to resolve the value of any Usage Dep parameter before passing its value to the [External Model] (see Section 10.2.3).

*With:*

Corner:

Three entries follow the Corner subparameter on each line:

corner\_name file\_reference circuit\_name

The corner\_name entry is “Typ”, “Min”, or “Max”. The file\_reference entry points to a file that resides in the same directory as the .ibs file or in a relative path under that directory.

Up to three Corner lines are permitted. A “Typ” line is required. If “Min” and/or “Max” data is missing, the tool may use “Typ” data in its place. However, the tool should notify the user of this action.

Models instantiated by corner\_name "Min" describe slow, weak performance, and models instantiated by corner\_name "Max" describe fast, strong performance.

The circuit\_name entry provides the name of the circuit to be simulated within the referenced file. For SPICE and IBIS-ISS files, this is normally a “.subckt” name. For VHDL-AMS files, this is normally an “entity(architecture)” name pair. For Verilog-AMS files, this is normally a “module” name.

No character limits, case-sensitivity limits or extension conventions are required or enforced for file\_reference and circuit\_name entries. However, the total number of characters in each Corner line must comply with the rules in Section 3. Furthermore, lower-case file\_reference entries are recommended to avoid possible conflicts with file naming conventions under different operating systems. Case differences between otherwise identical file\_reference entries or circuit\_name entries should be avoided. External languages may not support case-sensitive distinctions.

Parameters:

Lists names of parameters that can be passed into an external model file. Each Parameters entry must match a name or keyword in the external file or language. The list of Parameters may span several lines by using the word Parameters at the start of each line. The Parameters subparameter is optional, and the external model must operate with default settings without any Parameters assignments.

Parameter passing is not supported in SPICE. VHDL-AMS and VHDL-A(MS) parameters are supported using “generic” names, and Verilog-AMS and Verilog-A(MS) parameters are supported using “parameter” names. IBIS-ISS parameters are supported for all IBIS-ISS parameters which are defined on the subcircuit definition line.

Parameters are locally scoped under each [External Model] keyword, i.e., the same parameter under two different [External Model] will have independent values.

The parameter(s) listed under the Parameters subparameter may optionally be followed by an equal sign and a numeric, Boolean or string literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file reference, followed by an open parenthesis and a tree root name, a new open parenthesis for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file reference. The file reference may point to any file which contains one or more parameter trees. ~~The files referenced must be located in the same directory as the .ibs file containing the reference.~~ The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, file names using only a stem (e.g., xyz) or a stem and an ending period and no extension (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, .ebd, and .ims) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Model] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “ami”.
3. Usage Dep parameters may also be referenced in an AMI parameter definition file under the following conditions:
* the [External Model] keyword is located under a [Model] keyword which also contains an [Algorithmic Model] keyword,
* the [External Model]'s parameter and the [Algorithmic Model] keyword point to the same “.ami” file,
* the AMI parameter definition file contains the parameter AMI\_Resolve\_Exists with a value of True.

If all of these conditions are satisfied, the EDA tool must execute the AMI\_Resolve function in the executable model defined by the [Algorithmic Model] keyword to resolve the value of any Usage Dep parameter before passing its value to the [External Model] (see Section 10.2.3).

*------------------------------------------------------------------*

*On page102, for [External Model] Converter Parameters, change (delete period (.)*

*Replace:*

The Converter\_Parameters subparameter must contain one parameter name per line, which must be followed by an equal sign and a constant numeric literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file name, followed by an open parentheses and a the tree root name, a new open parentheses for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file name. The file reference may point to any file which contains one or more parameter trees. The files referenced must be located in the same directory as the .ibs file containing the reference. The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, files with no extensions (e.g, xyz) or with just a dot (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, and .ebd) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “.ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Model] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “.ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “.ami”.
3. Usage Dep parameters may also be referenced in an AMI parameter definition file under the following conditions:
* the [External Model] keyword is located under a [Model] keyword which also contains an [Algorithmic Model] keyword,
* the [External Model]'s parameter and the [Algorithmic Model] keyword point to the same AMI parameter definition file,
* the AMI parameter definition file contains the parameter AMI\_Resolve\_Exists with a value of True.

If all of these conditions are satisfied, the EDA tool must execute the AMI\_Resolve function in the executable model defined by the [Algorithmic Model] keyword to resolve the value of any Usage Dep parameter before passing its value to the [External Model] (see Section 10.2.3).

*With:*

The Converter\_Parameters subparameter must contain one parameter name per line, which must be followed by an equal sign and a constant numeric literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file reference, followed by an open parenthesis and a tree root name, a new open parenthesis for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file reference. The file reference may point to any file which contains one or more parameter trees. ~~The files referenced must be located in the same directory as the .ibs file containing the reference.~~ The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, file names using only a stem (e.g., xyz) or a stem and an ending period and no extension (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, .ebd, and .ims) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Model] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “ami”.
3. Usage Dep parameters may also be referenced in an AMI parameter definition file under the following conditions:
* the [External Model] keyword is located under a [Model] keyword which also contains an [Algorithmic Model] keyword,
* the [External Model]'s parameter and the [Algorithmic Model] keyword point to the same AMI parameter definition file,
* the AMI parameter definition file contains the parameter AMI\_Resolve\_Exists with a value of True.

If all of these conditions are satisfied, the EDA tool must execute the AMI\_Resolve function in the executable model defined by the [Algorithmic Model] keyword to resolve the value of any Usage Dep parameter before passing its value to the [External Model] (see Section 10.2.3).

---------------------------------------------------------

*On page120, for [External Circuit] Parameters, change (delete period (.) for correct extension definition, change file\_name to file\_reference in several locations, and “a the” to” a”):*

*Replace:*

Corner:

Three entries follow the Corner subparameter on each line:

corner\_name file\_name circuit\_name

The corner\_name entry is “Typ”, “Min”, or “Max”. The file\_name entry points to the referenced file in the same directory as the .ibs file.

Up to three Corner lines are permitted. A “Typ” line is required. If “Min” and/or “Max” data is missing, the tool may use “Typ” data in its place. However, the tool should notify the user of this action.

The circuit\_name entry provides the name of the circuit to be simulated within the referenced file. For SPICE and IBIS-ISS files, this is normally a “.subckt” name. For VHDL-AMS files, this is normally an “entity(architecture)” name pair. For Verilog-AMS files, this is normally a “module” name.

No character limits, case-sensitivity limits or extension conventions are required or enforced for file\_name and circuit\_name entries. However, the total number of characters in each Corner line must comply with Section 3. Furthermore, lower-case file\_name entries are recommended to avoid possible conflicts with file naming conventions under different operating systems. Case differences between otherwise identical file\_name entries or circuit\_name entries should be avoided. External languages may not support case-sensitive distinctions.

Parameters:

Lists names of parameters that may be passed into an external circuit file. Each Parameters entry must match a name or keyword in the external file or language. The list of Parameters can span several lines by using the word Parameters at the start of each line. The Parameters subparameter is optional, and the external circuit must operate with default settings without any Parameters assignments.

The parameter(s) listed under the Parameters subparameter may optionally be followed by an equal sign and a numeric, Boolean or string literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file name, followed by an open parentheses and a the tree root name, a new open parentheses for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. The file reference may point to any file which contains one or more parameter trees. The files referenced must be located in the same directory as the .ibs file containing the reference. The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “.ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Circuit] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “.ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “.ami”.

*With:*

Corner:

Three entries follow the Corner subparameter on each line:

corner\_name file\_reference circuit\_name

The corner\_name entry is “Typ”, “Min”, or “Max”. The file\_reference entry points to a file that resides in the same directory as the .ibs file or in a relative path under that directory.

Up to three Corner lines are permitted. A “Typ” line is required. If “Min” and/or “Max” data is missing, the tool may use “Typ” data in its place. However, the tool should notify the user of this action.

The circuit\_name entry provides the name of the circuit to be simulated within the referenced file. For SPICE and IBIS-ISS files, this is normally a “.subckt” name. For VHDL-AMS files, this is normally an “entity(architecture)” name pair. For Verilog-AMS files, this is normally a “module” name.

No character limits, case-sensitivity limits or extension conventions are required or enforced for file\_reference and circuit\_name entries. However, the total number of characters in each Corner line must comply with Section 3. Furthermore, lower-case file\_reference entries are recommended to avoid possible conflicts with file naming conventions under different operating systems. Case differences between otherwise identical file\_reference entries or circuit\_name entries should be avoided. External languages may not support case-sensitive distinctions.

Parameters:

Lists names of parameters that may be passed into an external circuit file. Each Parameters entry must match a name or keyword in the external file or language. The list of Parameters can span several lines by using the word Parameters at the start of each line. The Parameters subparameter is optional, and the external circuit must operate with default settings without any Parameters assignments.

The parameter(s) listed under the Parameters subparameter may optionally be followed by an equal sign and a numeric, Boolean or string literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file reference, followed by an open parenthesis and a tree root name, a new open parenthesis for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. The file reference may point to any file which contains one or more parameter trees. ~~The files referenced must be located in the same directory as the .ibs file containing the reference.~~ The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, file names using only a stem (e.g., xyz) or a stem and an ending period and no extension (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, .ebd, and .ims) do not contain parameter trees and are not permitted as parameter definition files.Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Circuit] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “ami”.

----------------------------------------------------------

*On page122, for [External Circuit] Converter Parameters, change (delete period (.) for correct extension definition and change file\_name to file\_reference in several locations add .ims format, and change a the to a):*

*Replace:*

The Converter\_Parameters subparameter must contain one parameter name per line, which must be followed by an equal sign and a constant numeric literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file name, followed by an open parentheses and a the tree root name, a new open parentheses for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file name. The file reference may point to any file which contains one or more parameter trees. The files referenced must be located in the same directory as the .ibs file containing the reference. The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, files with no extensions (e.g, xyz) or with just a dot (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, and .ebd) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “.ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Circuit] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “.ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “.ami”.

*With:*

The Converter\_Parameters subparameter must contain one parameter name per line, which must be followed by an equal sign and a constant numeric literal or a reference to a parameter name which is located in a parameter tree. The reference must begin with a file reference, followed by an open parentheses and a tree root name, a new open parentheses for any branch names (including the Reserved\_Parameters or Model\_Specific branch names if present in the tree) and the parameter name, and a matching set of closing parentheses. Spaces are allowed in the reference following the file reference. The file reference may point to any file which contains one or more parameter trees. ~~The files referenced must be located in the same directory as the .ibs file containing the reference.~~ The file names of parameter definition files must follow the rules for file names given in Section 3, “GENERAL SYNTAX RULES AND GUIDELINES”. In addition, file names using only a stem (e.g., xyz) or a stem and an ending period and no extension (e.g., xyz.) are permitted. IBIS file formats except .ami (e.g., .ibs, .pkg, .ebd, and .ims) do not contain parameter trees and are not permitted as parameter definition files. Parameter definition files may only contain parameter trees using the tree syntax described in IBIS in Section 10.3 with the following exceptions and additions:

The following rules apply to parameter trees located in parameter definition files whose file name extension is not “ami”.

1. The parameter tree must not contain the Reserved\_Parameters branch.
2. The parameter tree must contain the Model\_Specific branch.
3. The parameter tree may only contain Usage Info parameters.

The following rules must be observed when [External Circuit] parameters or converter parameters reference parameters located in external parameter definition files.

1. Usage Info parameters may be referenced in any external parameter definition file with or without the “ami” extension.
2. Usage In parameters may be referenced in any parameter definition file whose file name extension is “ami”.

------------------------------------------------------------------------------

*On pages 125-129, in the Multi-lingual Section, replace the Examples test to change file\_name to file\_reference and adjust column alignment:*

*Replace:*

*Examples:*

Example of Model B as an [External Circuit] using SPICE:

[External Circuit] BUFF-SPICE

Language SPICE

|

| Corner corner\_name file\_name circuit\_name (.subckt name)

Corner Typ buffer\_typ.spi bufferb\_io\_typ

Corner Min buffer\_min.spi bufferb\_io\_min

Corner Max buffer\_max.spi bufferb\_io\_max

|

| Parameters - Not supported in SPICE

|

| Ports List of port names (in same order as in SPICE)

Ports A\_signal int\_in int\_en int\_out A\_control

Ports A\_puref A\_pdref A\_pcref A\_gcref

|

| D\_to\_A d\_port port1 port2 vlow vhigh trise tfall corner\_name

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.6 0.4n 0.3n Max

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.6 0.4n 0.3n Max

|

| A\_to\_D d\_port port1 port2 vlow vhigh corner\_name

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Typ

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Min

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Max

|

| Note, the A\_signal port might also be used and int\_out not defined in

| a modified .subckt.

|

[End External Circuit]

Example [External Circuit] using IBIS-ISS:

[External Circuit] BUFF-ISS

Language IBIS-ISS

|

| Corner corner\_name file\_name circuit\_name (.subckt name)

Corner Typ buffer\_typ.spi bufferb\_io\_typ

Corner Min buffer\_min.spi bufferb\_io\_min

Corner Max buffer\_max.spi bufferb\_io\_max

|

| List of parameters

Parameters sp\_file\_name = paramfile.par(RootName(Model\_Specific(TstoneFile)))

Parameters C1\_value

Parameters R1\_value = paramfile.par(RootName(Model\_Specific(R1)))

|

Converter\_Parameters MyVlow = 0.0

Converter\_Parameters MyVHigh = 3.3

Converter\_Parameters MyVinl = paramfile.par(RootName(Model\_Specific(Vinl)))

Converter\_Parameters MyVinh = paramfile.par(RootName(Model\_Specific(Vinh)))

Converter\_Parameters MyTrise = paramfile.par(RootName(Model\_Specific(Trf)))

Converter\_Parameters MyTfall = paramfile.par(RootName(Model\_Specific(Trf)))

|

| Ports List of port names (in same order as in ISS)

Ports A\_signal int\_in int\_en int\_out A\_control

Ports A\_puref A\_pdref A\_pcref A\_gcref

|

| D\_to\_A d\_port port1 port2 vlow vhigh trise tfall corner\_name

D\_to\_A D\_drive int\_in my\_gcref MyVlow MyVhigh MyTfall MyTrise Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.6 0.4n 0.3n Max

|

| A\_to\_D d\_port port1 port2 vlow vhigh corner\_name

A\_to\_D D\_receive int\_out my\_gcref MyVinl MyVinh Typ

|

| Note, the A\_signal port might also be used and int\_out not defined in

| a modified .subckt.

|

[End External Circuit]

Example [External Circuit] using VHDL-AMS:

[External Circuit] BUFF-VHDL

Language VHDL-AMS

|

| Corner corner\_name file\_name entity(architecture)

Corner Typ buffer\_typ.vhd bufferb(buffer\_io\_typ)

Corner Min buffer\_min.vhd bufferb(buffer\_io\_min)

Corner Max buffer\_max.vhd bufferb(buffer\_io\_max)

|

| Parameters List of parameters

Parameters delay rate

Parameters preemphasis

|

| Ports List of port names (in same order as in VHDL-AMS)

Ports A\_signal A\_puref A\_pdref A\_pcref A\_gcref A\_control

Ports D\_drive D\_enable D\_receive

|

[End External Circuit]

Example [External Circuit] using Verilog-AMS:

[External Circuit] BUFF-VERILOG

Language Verilog-AMS

|

| Corner corner\_name file\_name circuit\_name (module)

Corner Typ buffer\_typ.v bufferb\_io\_typ

Corner Min buffer\_min.v bufferb\_io\_min

Corner Max buffer\_max.v bufferb\_io\_max

|

| Parameters List of parameters

Parameters delay rate

Parameters preemphasis

|

| Ports List of port names (in same order as in Verilog-AMS)

Ports A\_signal A\_puref A\_pdref A\_pcref A\_gcref A\_control

Ports D\_drive D\_enable D\_receive

|

[End External Circuit]

Example [External Circuit] using SPICE:

| Interconnect Structure as an [External Circuit]

|

|

[External Circuit] BUS\_SPI

Language SPICE

|

| Corner corner\_name file\_name circuit\_name (.subckt name)

Corner Typ bus\_typ.spi Bus\_typ

Corner Min bus\_min.spi Bus\_min

Corner Max bus\_max.spi Bus\_max

|

| Parameters - Not supported in SPICE

|

| Ports are in same order as defined in SPICE

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

| No A\_to\_D or D\_to\_A required, as no digital ports are used

|

[End External Circuit]

Example [External Circuit] using IBIS-ISS:

| Interconnect Structure as an [External Circuit]

|

|

[External Circuit] BUS\_SPI

Language IBIS-ISS

|

| Corner corner\_name file\_name circuit\_name (.subckt name)

Corner Typ bus\_typ.spi Bus\_typ

Corner Min bus\_min.spi Bus\_min

Corner Max bus\_max.spi Bus\_max

|

| List of parameters

Parameters sp\_file\_name

Parameters C1\_value R1\_value

|

| Ports are in same order as defined in IBIS-ISS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

| No A\_to\_D or D\_to\_A required, as no digital ports are used

|

[End External Circuit]

Example [External Circuit] using VHDL-AMS:

[External Circuit] BUS\_VHD

Language VHDL-AMS

|

| Corner corner\_name file\_name entity(architecture)

Corner Typ bus.vhd Bus(Bus\_typ)

Corner Min bus.vhd Bus(Bus\_min)

Corner Max bus.vhd Bus(Bus\_max)

|

| Parameters List of parameters

Parameters r1 l1

Parameters r2 l2 temp

|

| Ports are in the same order as defined in VHDL-AMS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

Example [External Circuit] using Verilog-AMS:

[External Circuit] BUS\_V

Language Verilog-AMS

|

| Corner corner\_name file\_name circuit\_name (module)

Corner Typ bus.v Bus\_typ

Corner Min bus.v Bus\_min

Corner Max bus.v Bus\_max

|

| Parameters List of parameters

Parameters r1 l1

Parameters r2 l2 temp

|

| Ports are in the same order as defined in Verilog-AMS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

[End External Circuit]

The scope of the following keywords is limited to the [Component] keyword. They apply to the specific set of pin numbers and internal nodes only within that [Component].

*With::*

*Examples:*

Example of Model B as an [External Circuit] using SPICE:

[External Circuit] BUFF-SPICE

Language SPICE

|

| Corner corner\_name file\_reference circuit\_name (.subckt name)

Corner Typ buffer\_typ.spi bufferb\_io\_typ

Corner Min buffer\_min.spi bufferb\_io\_min

Corner Max buffer\_max.spi bufferb\_io\_max

|

| Parameters - Not supported in SPICE

|

| Ports List of port names (in same order as in SPICE)

Ports A\_signal int\_in int\_en int\_out A\_control

Ports A\_puref A\_pdref A\_pcref A\_gcref

|

| D\_to\_A d\_port port1 port2 vlow vhigh trise tfall corner\_name

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_drive int\_in my\_gcref 0.0 3.6 0.4n 0.3n Max

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.6 0.4n 0.3n Max

|

| A\_to\_D d\_port port1 port2 vlow vhigh corner\_name

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Typ

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Min

A\_to\_D D\_receive int\_out my\_gcref 0.8 2.0 Max

|

| Note, the A\_signal port might also be used and int\_out not defined in

| a modified .subckt.

|

[End External Circuit]

Example [External Circuit] using IBIS-ISS:

[External Circuit] BUFF-ISS

Language IBIS-ISS

|

| Corner corner\_name file\_reference circuit\_name (.subckt name)

Corner Typ buffer\_typ.spi bufferb\_io\_typ

Corner Min buffer\_min.spi bufferb\_io\_min

Corner Max buffer\_max.spi bufferb\_io\_max

|

| List of parameters

Parameters sp\_file\_name = paramfile.par(RootName(Model\_Specific(TstoneFile)))

Parameters C1\_value

Parameters R1\_value = paramfile.par(RootName(Model\_Specific(R1)))

|

Converter\_Parameters MyVlow = 0.0

Converter\_Parameters MyVHigh = 3.3

Converter\_Parameters MyVinl = paramfile.par(RootName(Model\_Specific(Vinl)))

Converter\_Parameters MyVinh = paramfile.par(RootName(Model\_Specific(Vinh)))

Converter\_Parameters MyTrise = paramfile.par(RootName(Model\_Specific(Trf)))

Converter\_Parameters MyTfall = paramfile.par(RootName(Model\_Specific(Trf)))

|

| Ports List of port names (in same order as in ISS)

Ports A\_signal int\_in int\_en int\_out A\_control

Ports A\_puref A\_pdref A\_pcref A\_gcref

|

| D\_to\_A d\_port port1 port2 vlow vhigh trise tfall corner\_name

D\_to\_A D\_drive int\_in my\_gcref MyVlow MyVhigh MyTfall MyTrise Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.3 0.5n 0.3n Typ

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.0 0.6n 0.3n Min

D\_to\_A D\_enable int\_en my\_gnd 0.0 3.6 0.4n 0.3n Max

|

| A\_to\_D d\_port port1 port2 vlow vhigh corner\_name

A\_to\_D D\_receive int\_out my\_gcref MyVinl MyVinh Typ

|

| Note, the A\_signal port might also be used and int\_out not defined in

| a modified .subckt.

|

[End External Circuit]

Example [External Circuit] using VHDL-AMS:

[External Circuit] BUFF-VHDL

Language VHDL-AMS

|

| Corner corner\_name file\_reference entity(architecture)

Corner Typ buffer\_typ.vhd bufferb(buffer\_io\_typ)

Corner Min buffer\_min.vhd bufferb(buffer\_io\_min)

Corner Max buffer\_max.vhd bufferb(buffer\_io\_max)

|

| Parameters List of parameters

Parameters delay rate

Parameters preemphasis

|

| Ports List of port names (in same order as in VHDL-AMS)

Ports A\_signal A\_puref A\_pdref A\_pcref A\_gcref A\_control

Ports D\_drive D\_enable D\_receive

|

[End External Circuit]

Example [External Circuit] using Verilog-AMS:

[External Circuit] BUFF-VERILOG

Language Verilog-AMS

|

| Corner corner\_name file\_reference circuit\_name (module)

Corner Typ buffer\_typ.v bufferb\_io\_typ

Corner Min buffer\_min.v bufferb\_io\_min

Corner Max buffer\_max.v bufferb\_io\_max

|

| Parameters List of parameters

Parameters delay rate

Parameters preemphasis

|

| Ports List of port names (in same order as in Verilog-AMS)

Ports A\_signal A\_puref A\_pdref A\_pcref A\_gcref A\_control

Ports D\_drive D\_enable D\_receive

|

[End External Circuit]

Example [External Circuit] using SPICE:

| Interconnect Structure as an [External Circuit]

|

|

[External Circuit] BUS\_SPI

Language SPICE

|

| Corner corner\_name file\_reference circuit\_name (.subckt name)

Corner Typ bus\_typ.spi Bus\_typ

Corner Min bus\_min.spi Bus\_min

Corner Max bus\_max.spi Bus\_max

|

| Parameters - Not supported in SPICE

|

| Ports are in same order as defined in SPICE

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

| No A\_to\_D or D\_to\_A required, as no digital ports are used

|

[End External Circuit]

Example [External Circuit] using IBIS-ISS:

| Interconnect Structure as an [External Circuit]

|

|

[External Circuit] BUS\_SPI

Language IBIS-ISS

|

| Corner corner\_name file\_reference circuit\_name (.subckt name)

Corner Typ bus\_typ.spi Bus\_typ

Corner Min bus\_min.spi Bus\_min

Corner Max bus\_max.spi Bus\_max

|

| List of parameters

Parameters sp\_file\_name

Parameters C1\_value R1\_value

|

| Ports are in same order as defined in IBIS-ISS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

| No A\_to\_D or D\_to\_A required, as no digital ports are used

|

[End External Circuit]

Example [External Circuit] using VHDL-AMS:

[External Circuit] BUS\_VHD

Language VHDL-AMS

|

| Corner corner\_name file\_reference entity(architecture)

Corner Typ bus.vhd Bus(Bus\_typ)

Corner Min bus.vhd Bus(Bus\_min)

Corner Max bus.vhd Bus(Bus\_max)

|

| Parameters List of parameters

Parameters r1 l1

Parameters r2 l2 temp

|

| Ports are in the same order as defined in VHDL-AMS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

Example [External Circuit] using Verilog-AMS:

[External Circuit] BUS\_V

Language Verilog-AMS

|

| Corner corner\_name file\_reference circuit\_name (module)

Corner Typ bus.v Bus\_typ

Corner Min bus.v Bus\_min

Corner Max bus.v Bus\_max

|

| Parameters List of parameters

Parameters r1 l1

Parameters r2 l2 temp

|

| Ports are in the same order as defined in Verilog-AMS

Ports vcc gnd io1 io2

Ports int\_ioa vcca1 vcca2 vssa1 vssa2

Ports int\_iob vccb1 vccb2 vssb1 vssb2

|

[End External Circuit]

The scope of the following keywords is limited to the [Component] keyword. They apply to the specific set of pin numbers and internal nodes only within that [Component].

*------------------------------------------------------------*

*On page 139, in the Package Modeling Section, change this paragraph:*

*Replace:*

Use the [Package Model] keyword within a [Component] to indicate the package model for that component. The specification permits .ibs files to contain the following additional list of package model keywords. Note that the actual package models can be in a separate <package\_file\_name>.pkg file or can exist in the .ibs files between the [Define Package Model] ... [End Package Model] keywords for each package model that is defined. For reference, these keywords are listed in Table 15. Full descriptions follow. EDA tools that do not support these keywords will ignore all entries between the [Define Package Model] and [End Package Model] keywords.

*With:*

Use the [Package Model] keyword within a [Component] to indicate the package model for that component. The specification permits .ibs files to contain the following additional list of package model keywords. Note that the actual package models can be in a separate <stem>.pkg file or can exist in the .ibs files between the [Define Package Model] ... [End Package Model] keywords for each package model that is defined. For reference, these keywords are listed in Table 15. Full descriptions follow. EDA tools that do not support these keywords will ignore all entries between the [Define Package Model] and [End Package Model] keywords.

--------------------------------------------------------------

*On page 140 in the Package Modeling Section, change these paragraphs:*

*Replace:*

Usage Rules for the .Pkg File:

Package models are stored in a file whose name looks like:

<filename>.pkg.

The <filename> provided must adhere to the rules given in Section 3, "GENERAL SYNTAX RULES AND GUIDELINES". Use the “.pkg” extension to identify files containing package models. The .pkg file must contain all of the required elements of a normal .ibs file, including [IBIS Ver], [File Name], [File Rev], and the [End] keywords. Optional elements include the [Date], [Source], [Notes], [Disclaimer], [Copyright], and [Comment Char] keywords. All of the elements follow the same rules as those for a normal .ibs file.

*With:*

Usage Rules for the .Pkg File:

Package models are stored in a file whose name looks like:

<stem>.pkg.

The <stem> provided shall adhere to the rules given in the [File Name] keyword. Use the “pkg” extension to identify files containing package models. The .pkg file must contain all of the required elements of a normal .ibs file, including [IBIS Ver], [File Name], [File Rev], and the [End] keywords. Optional elements include the [Date], [Source], [Notes], [Disclaimer], [Copyright], and [Comment Char] keywords. All of the elements follow the same rules as those for a normal .ibs file.

-------------------------------------------------------------

*On page 154, in the Electrical Board Description Section under Usage Rules, change:*

*Replace:*

A .ebd file is intended to be a stand-alone file, not referenced by or included in any .ibs or .pkg file. Electrical Board Descriptions are stored in a file whose name looks like <filename>.ebd, where <filename> must conform to the naming rules given in Section 3 of this specification. The .ebd extension is mandatory.

*With:*

A .ebd file is intended to be a stand-alone file, not referenced by or included in any .ibs or .pkg file. Electrical Board Descriptions are stored in a file whose name looks like <stem>.ebd, where <stem> shall conform to the naming rules given in the [File Name] keyword. The ebd extension is mandatory.

------------------------------------------------------------

*On page 163, in the Electrical Board Description Section,*

*Replace:*

*Keyword:* **[Reference Designator Map]**

*Required:* Yes, if any of the path descriptions use the Node subparameter

*Description:* Maps a reference designator to a component or electrical board description contained in a .ibs or .ebd file.

*Usage Rules:* The [Reference Designator Map] keyword must be followed by a list of all of the reference designators called out by the Node subparameters used in the various path descriptions. Each reference designator is followed by the name of the .ibs or .ebd file containing the electrical description of the component or board, then the name of the component itself as given by the .ibs or .ebd file’s [Component] or [Begin Board Description] keyword respectively. The reference designator, file name and component name terms are separated by white space. By default the .ibs or .ebd files are assumed to exist in the same directory as the calling .ebd file. It is legal for a reference designator to point to a component that is contained in the calling .ebd file.

The reference designator is limited to ten characters.

*Example:*

[Reference Designator Map]

|

| External Part References:

|

| Ref Des File name Component name

u23 pp100.ibs Processor

u24 simm.ebd 16Meg X 36 SIMM Module

u25 ls244.ibs NoName 74LS244a

u26 r10K.ibs My\_10K\_Pullup

*With:*

*Keyword:* **[Reference Designator Map]**

*Required:* Yes, if any of the path descriptions use the Node subparameter

*Description:* Maps a reference designator to a component or electrical board description contained in a .ibs or .ebd file.

*Usage Rules:* The [Reference Designator Map] keyword must be followed by a list of all of the reference designators called out by the Node subparameters used in the various path descriptions. Each reference designator is followed by the file reference name of the .ibs or .ebd file containing the electrical description of the component or board, then the name of the component itself as given by the .ibs or .ebd file’s [Component] or [Begin Board Description] keyword respectively. The reference designator, file name and component name terms are separated by white space. The referenced .ibs or .ebd files can exist in the same directory as the calling .ebd file or in a relative path under this directory. It is legal for a reference designator to point to a component that is contained in the calling .ebd file.

The reference designator is limited to ten characters.

*Example:*

[Reference Designator Map]

|

| External Part References:

|

| Ref\_Des File\_reference Component\_name

u23 pp100.ibs Processor

u24 simm.ebd 16Meg X 36 SIMM Module

u25 ls244.ibs NoName 74LS244a

u26 r10K.ibs My\_10K\_Pullup

------------------------------------------------------------

*On page 171, 172,*

*Replace:*

Executable:

Three entries follow the Executable subparameter on each line:

Platform\_Compiler\_Bits File\_Name Parameter\_File

The Platform\_Compiler\_Bits entry provides the name of the operating system, compiler and its version and the number of bits the executable model file is compiled for. It is a string without white spaces, consisting of three fields separated by an underscore (“\_”). The first field consists of the name of the operating system followed optionally by its version. The second field consists of the name of the compiler followed by optionally by its version. The third field is an integer indicating the platform architecture. If the version for either the operating system or the compiler contains an underscore, it must be converted to a hyphen “-”. This is so that an underscore is only present as a separation character in the entry.

The architecture entry can be either “32” or “64”. Examples of Platform\_Compiler\_Bits:

Linux\_gcc3.2.3\_32

Solaris5.10\_gcc4.1.1\_64

Solaris\_cc5.7\_32

Windows\_VisualStudio7.1.3088\_32

HP-UX\_accA.03.52\_32

The EDA tool will check for the compiler information and verify if the executable model file is compatible with the operating system and platform.

Multiple occurrences, without duplication, of Executable are permitted to allow for providing executable model files for as many combinations of operating system platforms and compilers for the same algorithmic model.

The File\_Name provides the name of the executable model file. The executable model file should be in the same directory as the.ibs file.

The Parameter\_File entry provides the name of the AMI parameter definition file, which shall have an extension of .ami. This must be an external file and should reside in the same directory as the .ibs file and the executable model file. See Section 10.3 for details.

*With:*

Executable:

Three entries follow the Executable subparameter on each line:

Platform\_Compiler\_Bits Executable\_Model\_File AMI\_Parameter\_File

The Platform\_Compiler\_Bits entry provides the name of the operating system, compiler and its version and the number of bits the executable model file is compiled for. It is a string without white spaces, consisting of three fields separated by an underscore (“\_”). The first field consists of the name of the operating system followed optionally by its version. The second field consists of the name of the compiler followed by optionally by its version. The third field is an integer indicating the platform architecture. If the version for either the operating system or the compiler contains an underscore, it must be converted to a hyphen “-”. This is so that an underscore is only present as a separation character in the entry.

The architecture entry can be either “32” or “64”. Examples of Platform\_Compiler\_Bits:

Linux\_gcc3.2.3\_32

Solaris5.10\_gcc4.1.1\_64

Solaris\_cc5.7\_32

Windows\_VisualStudio7.1.3088\_32

HP-UX\_accA.03.52\_32

The EDA tool will check for the compiler information and verify if the executable model file is compatible with the operating system and platform.

Multiple occurrences, without duplication, of Executable are permitted to allow for providing executable model files for as many combinations of operating system platforms and compilers for the same algorithmic model.

The Executable\_Model\_File provides the executable model file reference. This must be an external file. The executable model file reference shall reside in the same directory as the .ibs file or in a relative path under that directory. See Section 10.2 for details.

The AMI\_Parameter\_File entry provides the AMI parameter definition file reference, which shall have an extension of ami. This must be an external file. The .ami file shall reside in the same directory as the .ibs file or in relative paths under that directory. See Section 10.3 for details

--------------------------------------------------------

*On page 210, change file name to root name:*

*Replace:*

(mySampleAMI | AMI parameter definition file name

*With*

(mySampleAMI | AMI parameter definition root name

*On page 211, change by deleting last sentence:*

The file names or directory names may be written with or without a path, but in either case, they must be expressed relative to the location of the .ami file in which the Supporting\_Files parameter is found~~. (The AMI executable models and the AMI parameter definition files are all required to be in the same directory as the .ibs file in which they are declared).~~

----------------------------------------------------------

*On pages 212 and 213,*

*Replace:*

*Parameter:* **DLL\_ID**

*Required:* No, and illegal before AMI\_Version 6.0

*Direction:* Rx, Tx

*Descriptors*:

Usage: In

Type: String

Format: Value

Default: <string literal>

Description:<string>

*Definition:* The EDA tool is responsible for recognizing this parameter name and replacing the value declared in the .ami file with a string that contains a unique alphanumeric identifier. The algorithmic model is responsible for using DLL\_ID as the base name for any data files that the model creates, either for use as temporary storage or for recording output data. The use of DLL\_ID helps guarantee that multiple instances of the same model (or different models from the same vendor) do not mix up data as a result of collisions between temporary or permanent file names.

*With:*

*Parameter:* **DLL\_ID**

*Required:* No, and illegal before AMI\_Version 6.0

*Direction:* Rx, Tx

*Descriptors*:

Usage: In

Type: String

Format: Value

Default: <string literal>

Description:<string>

*Definition:* The EDA tool is responsible for recognizing this parameter name and replacing the value declared in the .ami file with a string that must conform to the rules in item 3 of Section 3.2, "SYNTAX RULES". The algorithmic model is responsible for using the DLL\_ID string as part of the name for any data files that the model creates, either for use as temporary storage or for recording output data. The use of DLL\_ID helps guarantee that multiple instances of the same model (or different models from the same vendor) do not mix up data as a result of collisions between temporary or permanent file names.

**BACKGROUND INFORMATION/HISTORY:**

Five drafts of this BIRD were discussed and revised in the Advanced Technology Modeling Task group. The group voted to submit the BIRD to the IBIS Open Forum November 29, 2016.

BIRD86.1:

1. The language in Version 6.1 states that all files must reside in the same directory. The statements are intentional and therefore need to be corrected in all locations.
2. Another change is to correctly list extensions WITHOUT the period – as defined by “period” extension. This issue has and inconsistency has been in all versions of IBIS.
3. Anticipate BIRD189 approval and add the .ims file type for Interconnect Model Set files.
4. Clarify that the extension after the “last period” in a file name shall not contain a “/”.
5. Font size for Times New Roman set to 12.

BIRD186.2

[File Name] Description is changed. [File Name]: Usage Rules: may and must 🡪 shall.

BIRD186.3

Font corrections in four places: Courier to Times New Roman

Package Modeling Section 7: <package\_file\_name>.pkg and <filename>.pkg 🡪<base name>.pkg

Electrical Board Description Section 8: <filename>.ebd 🡪 <base name>.ebd

Note, “file name extension” is not changed because it describes a portion that is defined in a file name.

BIRD186.3

Boost.org terminology adopted plus some IBIS additions such as using “file reference” instead of file\_name for column headings. <base name> 🡪 <stem>, relative path used to describe files in directories under the referencing file. Red text and strikeouts are used in many places to indicate the location of changes and the changes. Most of the changes are now organized as cut-and-paste replacements of the original paragraphs in IBIS Version 6.1.

Some clarifications are made in Section 3 because some rules apply to all file types, and some are restricted to ASCII formatted file types. Also, some rules are not stated for external files referenced by IBIS file types.

Several editorial changes made including changing “dot” to “period”.