What would the “pre-layout” package and on-die interconnect examples look like with the BIRD 163 syntax?

Arpad Muranyi

IBIS Interconnect Teleconference

April 9, 2014

Extending the proposed

<reserved buffer terminal name> : <pin name>

syntax in BIRD 163 could support everything in the SiSoft proposal(s)

Additional reserved names, wildcards, etc… could be added to the list of reserved buffer terminal names already mentioned in BIRD 163

Instead of supporting pin names only after the colon ":" character, we could also add support for [Model] names, and perhaps other items

These extensions to BIRD 163 could support the same capabilities described in SiSoft's proposal(s) and presentation(s)

For example, mapping a 4-port Touchstone model between two pins and two pads for any differential pair which has DQS as the [Model] name would look like this:

```
[Circuit Call] DQS_pkg
| This is the package for all DQS diff-pairs
Parameters  TS_FileName = 'Typ.s4p'
| mapping   port   pin/pad/node
Port_map    1      pin_pos:DQS
Port_map    2      signal_pos:DQS
Port_map    3      pin_neg:DQS
Port_map    4      signal_neg:DQS
[End Circuit Call]
…
[External Circuit] DQS_pkg
…
[End External Circuit]
```

where "pin_pos" and "pin_neg" refers to any pin which is part of a differential pair driven by any [Model] called "DQS", and "signal_pos" and "signal_neg" refer to the signal terminals of any differential [Model] pair called "DQS". With this example, the tool would use [External Circuit] DQS_pkg for the package behind all differential pin pairs delivering signals from instances of [Model] DQS.

We could also have:

```
[Circuit Call] Default_Differential_pkg
Parameters  TS_FileName = 'Default.s4p'
| mapping   port   pin/pad/node
Port_map    1      pin_pos_Default
Port_map    2      signal_pos_Default
Port_map    3      pin_neg_Default
Port_map    4      signal_neg_Default
[End Circuit Call]
```

In this example, the tool would apply [External Circuit] Default_Differential_pkg to all differential pairs unless another [Circuit Call] more narrowly matched the user's I/O buffers by [Model] name or pin name.

We could of course consider buffer type, or perhaps other items as a matching criterion too.

The "generalized" or wildcard names in the previous examples tell the tool unambiguously how to connect package models to I/O buffers and netlist them for simulation, given a clear set of precedence rules for choosing the [Circuit Call] matched by "_Default" vs. "pin_" vs. specific pin names. The IBIS file becomes more compact and perhaps more readable, and it can offer both default and more specialized broadband models.

However, we will run into trouble when we introduce generalized node names which do not unambiguously link the ports of a broadband model to a particular location in the component.

Coupled broadband models will have that problem, such as the following SiSoft example:

```
[Begin ISS Model] Rx_Xtalk
Language Touchstone
File PDF .2 Rx_Min.s12p .6 Rx_Typ.s12p .2 Rx_Max.s12p
Ports   Pin_Mod+.Rx.1   Pin_Mod-.Rx.1   Buf_Mod+.Rx.1   Buf_Mod-.Rx.1
Ports V_Pin_Mod+.Rx.2 V_Pin_Mod-.Rx.2 V_Buf_Mod+.Rx.2 V_Buf_Mod-.Rx.2
Ports   Pin_Mod+.Rx.3   Pin_Mod-.Rx.3   Buf_Mod+.Rx.3   Buf_Mod-.Rx.3
[End ISS Model]
```

While the EDA tool might have the concept of "victim" vs. "aggressor" net (in post- and/or pre-layout contexts), there certainly is no simple way to relate aggressor paths "1" and "3" with specific I/O buffers in the layout unambiguously without user intervention, regardless of the post- or pre-layout context.