**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD ID#:**

**ISSUE TITLE:** *New AMI API to Resolve Dependent Model Parameter*

**REQUESTER:** *Fangyi Rao, Agilent Technologies, Inc.*

**DATE SUBMITTED:**

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

AMI model parameters that are used by EDA tools can depend on other model parameters and simulation parameters including data rate, IBIS corner and IBIS model name. The form of such dependency relation varies from IC vendor to IC vendor and from device to device. The number of possible variations among vendors and devices is infinite. Model vendors need a flexible mechanism to implement parameter dependency according to their proprietary formula and pass the dependent parameter values to EDA tools. It’s foreseeable that certain vendors need to conceal the dependency formula.

A new API is added to AMI and a new reserved parameter is introduced. The API declaration is

*long AMI\_ResolveDependentParam(double bit\_time,*

*char \* corner,*

*char \* model\_name,*

*char \* AMI\_parameters\_in,*

*char \*\* AMI\_parameters\_out);*

Argument definitions are

**bit\_time:** input argument, in second, equals 1/data rate.

**corner:** input argument, ibis model corner, allowed values are “typ”, “min” and “max”.

**model\_name:** input argument, ibis model name.

**AMI\_parameters\_in:** input argument, a string that contains name-value pairs of In and InOut parameters. The format of this string is the same as that of the AMI\_parameters\_in argument in AMI\_Init.

**AMI\_parameters\_out:** output argument, pointer to a string that contains name-value pairs of Out and InOut dependent parameters. The format of this string is the same as that of the AMI\_parameters\_out argument in AMI\_Init.

The new reserved parameter, ResolveDependentParam\_Exists, indicates whether the model implements the AMI\_ResolveDependentParam function and is defined as

(ResolveDependentParam\_Exists (Usage Info) (Type Boolean) (Default False)

(Description “Indicates whether DLL implements ResolveDependentParam.”) )

The usage of the new API is described below.

1. User selects ibis model and specifies corner and data rate.
2. EDA tool initializes AMI\_parameters\_out to NULL.
3. If ResolveDependentParam\_Exists is False, go to step 9.
4. If ResolveDependentParam\_Exists is True, EDA tool allocates memory for the AMI\_parameters\_in string and writes to it name-value pairs of all In and InOut parameters.
5. EDA tool calls AMI\_ResolveDependentParam before analog channel impulse characterization.
6. DLL computes dependent parameter values according to independent parameter values in AMI\_parameters\_in, bit\_time, corner and model\_name.
7. DLL allocates memory for the AMI\_parameters\_out string and writes to it name-value pairs of dependent parameters. Note that DLL must use malloc or calloc to allocate memory for AMI\_parameters\_out (see step 10).
8. EDA tool sets/adjusts analog model parameters if their values are returned by DLL in AMI\_parameters\_out.
9. EDA tool characterizes analog channel impulse responses, calls AMI\_Init and finishes the rest of the simulation.
10. EDA tool calls free(\*AMI\_parameters\_out) if AMI\_parameters\_out is not a NULL pointer to free memory.