**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD NUMBER:** 201.1

**ISSUE TITLE:** Back-channel Statistical Optimization

**REQUESTOR:**  Walter Katz, Signal Integrity Software

**DATE SUBMITTED:** January 7, 2020

**DATE REVISED:** June 2, 2020

**DATE ACCEPTED:** July 17, 2020 (Superseded by BIRD215)

**DEFINITION OF THE ISSUE:**

IC vendors would like back-channel to support training in the Statistical flow iterating between the Tx and Rx doing optimization based on impulse responses.

This BIRD defines one new BCI reserved parameter and how it determines what the EDA tool needs to do to support training in the Statistical Flow.

This BIRD also defines one new executable model function AMI\_Impulse. After the initial calls to Tx and Rx AMI\_Init, and if Statistical Training is enabled, the EDA tool shall repeatably call the Tx and Rx AMI\_Impulse functions until the Rx AMI\_Impulse returns BCI\_State Converged.

**SOLUTION REQUIREMENTS:**

The IBIS specification must meet these requirements:

|  |  |
| --- | --- |
| Requirement | Notes |
| 1. Enable back-channel link training messages between the Tx and Rx executable models to enable the Rx or Tx executable model to control the equalization of the Tx or Rx during statistical simulations. |  |
| 1. Allow the user and tool to know when statistical link training has ended, and normal operation has begun. |  |

**SUMMARY OF PROPOSED CHANGES:**

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

|  |  |  |
| --- | --- | --- |
| Specification Item | New/Modified/Other | Notes |
| New AMI Reserved Parameter  BCI\_Training\_Mode | All are new AMI Parameters | All affect the operation of the AMI function AMI\_Impulse. |
| New executable function AMI\_Impulse | New |  |

**PROPOSED CHANGES:**

ADD TO SECTION 10.2.3 AFTER *Function* AMI\_Init and above *Function* AMI\_GetWave:

*Function:* **AMI\_Impulse**

*Required:* Yes

*Declaration:* long AMI\_Impulse (double \*impulse\_matrix,

char \*BCI\_parameters\_in,

char \*\*BCI\_parameters\_out,

char \*\*AMI\_parameters\_out,

void \*AMI\_memory)

*Arguments:*

impulse\_matrix

Same as impulse\_matrix argument defined under AMI\_Init above.

Note that since both AMI\_Init and AMI\_Impulse modify the impulse\_matrix in place, the EDA tool could maintain the original impulse\_matrix and use different memory for the impulse\_matrix input to the AMI\_Init and AMI\_Impulse functions.

Note that the AMI\_Impulse function uses the number\_of\_rows, aggressors, sample\_interval, and bit\_time that were passed to the AMI\_Init call.

BCI\_parameters\_in

The BCI\_parameters\_in argument is a pointer to a string. This pointer is returned in the BCI\_parameter\_out argument by a previous call to an AMI\_Impulse function in another executable model in the channel. Memory for the string is allocated and de-allocated by the previous call to an AMI\_Impulse function in another executable model in the channel. The string must be formatted as defined by the BCI\_Protocol. On the first call to the primary Tx AMI\_Impulse function, this pointer shall be the Null pointer (0).

BCI\_parameters\_out

The BCI\_parameters\_out argument is a pointer to a string pointer. Memory for the string is allocated and de-allocated by the algorithmic model. The model returns a pointer to the string as the contents of this argument. The string must be formatted as defined by the BCI\_Protocol.

The EDA tool must initialize the memory content at this address to zero (null pointer) prior to calling the AMI\_Impulse function, so that after the execution of the function it can determine whether or not the function returned a valid string pointer at that address.

AMI\_parameters\_out

Same as AMI\_Parameters\_out argument defined under AMI\_Init above.

**AMI\_memory**

This is the memory pointer which was allocated during the AMI\_Init call.

Return Value

1 for success

0 for failure

Algorithmic models shall return a failure code (0) if and only if the function call fails due to a program execution error. In all other cases the return code shall be "success" (1), even if the function cannot operate properly due to some functional problems. For example, if a function includes a CDR which is unable to get into a stable mode, the function shall still return a success code (1). Examples for returning a failure code (0) may include an invalid data type, a null pointer during run time, or anything that prevents the successful execution of the model’s code.

The authors of Algorithmic Models are encouraged to provide feedback to the EDA tool’s users through the various available messaging options about any difficulties the model encounters during execution, regardless of what the value of the function’s return code is.

ADD SECTION SUBTITLES FOR REPEATERS

Page 262 ABOVE “The time-domain simulation flow for a Repeater link shown in Figure 41 is defined below.”

**10.8.2 REPEATER TIME DOMAIN SIMULATION FLOW FOR A REPEATER LINK**

Page 264, ABOVE “The statistical simulation flow for a Repeater link shown in Figure 41 is defined below.”

**10.8.3 REPEATER STATISTICAL DOMAIN SIMULATION FLOW FOR A REPEATER LINK**

REPLACE:

## AMI Reserved Parameter Definitions For Link Training Communications

In this section, the parameters BCI\_Protocol, BCI\_State, BCI\_ID, BCI\_Message\_Interval\_UI and BCI\_Training\_UI are documented, to enable link training communication. These Reserved Parameters are in the AMI file and positioned under the Reserved\_Parameters branch.

WITH

## AMI Reserved Parameter Definitions For Link Training Communications

With the information provided in this section, IC vendors and EDA tool vendors can develop models that support Back Channel Training and the enhancements EDA tools will need to support these models. The following Reserved Parameters are in the AMI file and positioned under the Reserved\_Parameters branch.

ADD AFTER *Parameter* BCI\_Training\_UI SECTION AND ABOVE 10.9.1:

*Parameter:* **BCI\_Training\_Mode**

*Required:* No, and illegal before AMI\_Version 7.1

*Direction:* Rx, Tx

*Descriptors*:

Usage: In

Type: String

Format: Value, List

Default: <string literal>

Description:<string>

*Definition:* This parameter tells the EDA tool if the model supports Statistical Optimization Only, Time Domain Optimization Only, or is “Both” (both Statistical and Time Domain Optimization). The only allowed values of BCI\_Training\_Mode are “Impulse”, “GetWave”, or “Both”. If “Both” is present, then “Impulse” and “GetWave” shall also be present.

Allowed Formats:

(Value “Impulse”)

(Value “GetWave”)

(List “Impulse”)

(List “GetWave”)

(List “Impulse” “GetWave”)

(List “Impulse” “GetWave” “Both”)

Illegal Formats:

(Value “Both”)

(List “Both” “Impulse”)

(List “Both” “GetWave”)

*Usage Rules:* The user/EDA tool can only choose a BCI\_Training\_Mode value if it is available on both the Tx and the Rx (and must be set the same for Tx and Rx). If BCI\_Training\_Mode is not present, then it shall default to “GetWave”.

To run a BCI Statistical simulation, the Tx and Rx model (or the primary Tx, terminal Rx, and all repeater Rx and Tx models) must have BCI\_Training\_Mode as either “Impulse” or “Both”.

To run a BCI Time Domain simulation, the Tx and Rx model (or the primary Tx, terminal Rx, and all repeater Rx and Tx models) must have BCI\_Training\_Mode as either “GetWave” or “Both”.

*Example:*

(BCI\_Training\_Mode (Usage In)(Type String)(List “Both” “Impulse” “GetWave”)

(Description "This Device supports Back-channel Statistical and Time Domain Optimization”))

AFTER SECTION 10.9.2, ADD NEW SECTIONS 10.9.3 AND 10.9.4

**10.9.3 STATISTICAL TRAINING/ANALYSIS FLOW FOR CHANNELS WITH NO REPEATER**

The EDA tool shall make the following calls to the Tx and Rx AMI\_Init and AMI\_Impulse functions if (BCI\_Training\_Mode “Impulse” or “Both”).

1. Tx AMI\_Init is called with
   1. (BCI\_State “Training”) (BCI\_Protocol “<name>”) (BCI\_ID “<my\_ ID>”) (BCI\_Training\_Mode “Both” or ”Impulse”)
   2. If (BCI\_Training\_Mode “GetWave”) follow the flow in Section 10.9.1.
   3. Impulse Matrix contains Impulse Response of channel
   4. If the Tx executable model does not implement the BCI\_Protocol and BCI\_Training\_Mode, it returns “Error” in BCI\_State.
2. Rx AMI\_Init is called with
   1. (BCI\_State “Training”) (BCI\_Protocol “<name>”) (BCI\_ID “<my\_ID>”) (BCI\_Training\_Mode “Both” or ”Impulse”)
   2. Impulse Matrix contains Impulse Response output of Tx AMI\_Init
   3. If the Rx executable model does not implement the BCI\_Protocol and BCI\_Training\_Mode, it returns “Error” in BCI\_State.
   4. The EDA tool may analyze the results of Rx AMI\_Init.
3. Tx AMI\_Impulse is called with the same impulse\_matrix used in the call to Tx AMI\_Init. The value of BCI\_parameters\_in shall be Null (0) on the first call to Tx AMI\_Impulse and the value of BCI\_parameters\_out of the previous call to Rx AMI\_Impulse on subsequent calls to Tx AMI\_Impulse.
4. The Rx AMI\_Impulse is called using the impulse\_matrix output of Tx AMI\_Impulse. The value of BCI\_parameters\_in shall be set to the value of BCI\_parameters\_out of the previous call to Tx AMI\_Impulse.
5. Steps 3 and 4 are repeated until the Rx AMI\_Impulse returns AMI Reserved Parameter BCI\_State “Converged”, “Failed”, or “Error”
   1. “Converged”: The impulse\_matrix and AMI\_parameters\_out returned by the receiver AMI\_Impulse function are used by EDA tool to complete the simulation. The EDA tool may proceed with Time Domain simulation.
   2. “Fail”: This tells the EDA tool that the training failed to converge. The EDA tool can terminate the simulation or proceed using the outputs of AMI\_Impulse or AMI\_Init to complete the simulation at its own risk. The EDA tool may proceed with Time Domain simulation.
   3. “Error”: Tells the EDA tool that an error was detected that prevented optimization to continue. The EDA tool can terminate the simulation or proceed using the outputs of AMI\_Impulse or AMI\_Init to complete the simulation at its own risk. The EDA tool may proceed with Time Domain simulation.
6. If the User selected to do Time Domain simulation after Impulse training
   1. If BCI\_Training\_Mode is “Impulse” the Tx and Rx AMI\_GetWave shall not do training and shall return BCI\_State “Off”.
   2. If BCI\_Training\_Mode is “Both” the time-domain training outlined in section 10.9.1 is invoked.

**10.9.4 STATISTICAL TRAINING/ANALYSIS FLOW FOR CHANNELS WITH REPEATERS**

1. It is required that all AMI models from the primary Tx to the terminal Rx support (BCI\_Training\_Mode “Impulse” or “Both”).
2. The AMI\_Init flow is identical to the flow defined for the statistical simulation flow for a Repeater link shown in Figure 41 in Section 10.8.2 and documented in Section 10.8.3.
3. This same flow is repeated with the calls to AMI\_Init replaced by calls to AMI\_Impulse.
4. The BCI training terminates when BCI\_State returns “Converged”, “Fail”, or “Error” in the AMI\_parameters\_out of the terminal Rx AMI\_Impulse function.

Note that the EDA tool shall set the value of BCI\_parameters\_in to the value of BCI\_parameters\_out of the previous call to an AMI\_Impulse function in the channel.

CHANGE SECTION NUMBER FROM SECTION 10.9.3 TO 10.9.5 AND ADD BCI\_Training\_Mode to TABLES 33-35:

### Summary Tables for Usage, Type and Format

Table 33 – General Rules and Allowable Usage for BCI Reserved Parameters

| **Reserved Parameter** | **General Rules** | | **Allowable Usage** | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Required** | **Default2,4** | **Info** | **In** | **Out** | **Dep1** | **InOut** |
| BCI\_Message\_Interval\_UI | Yes3 | -- | X |  |  |  |  |
| BCI\_ID | Yes3 | -- |  | X |  |  |  |
| BCI\_Protocol | No | -- |  | X |  |  |  |
| BCI\_State | Yes3 | -- |  |  |  |  | X |
| BCI\_Training\_UI | Yes3 | -- |  | X |  |  |  |
| BCI\_Training\_Mode | No5 | “GetWave” |  | X |  |  |  |
| Notes:   1. Illegal for AMI\_Version 6.0 and earlier 2. “Default” in this context means “behavior if Reserved Parameter is absent” 3. Required if BCI\_Protocol is present 4. “--" means that an entry must be provided if the parameter is present; no default is assumed or permitted 5. Illegal for AMI\_Version 7.0 and earlier, required if BCI\_Protocol supports Statistical Training | | | | | | | |

**Table 34 – Allowable Data Types for BCI Reserved Parameters**

| **Reserved Parameter** | **Data Type** | | | | |
| --- | --- | --- | --- | --- | --- |
| **Float** | **UI** | **Integer** | **String** | **Boolean** |
| BCI\_Message\_Interval\_UI |  |  | X |  |  |
| BCI\_ID |  |  |  | X |  |
| BCI\_Protocol |  |  |  | X |  |
| BCI\_State |  |  |  | X |  |
| BCI\_Training\_UI |  |  | X |  |  |
| BCI\_Training\_Mode |  |  |  | X |  |

Table 35 – Allowable Data Formats for BCI Reserved Parameters

| **Reserved Parameter** | **Data Format** | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Value** | **Range** | **Corner** | **List** | **Increment** | **Steps** | **Gaussian** | **Both-Dirac** | **DjRj** | **Table** |
| BCI\_Message\_Interval\_UI | X |  |  |  |  |  |  |  |  |  |
| BCI\_ID | X |  |  |  |  |  |  |  |  |  |
| BCI\_Protocol | X |  |  | X |  |  |  |  |  |  |
| BCI\_State |  |  |  | X |  |  |  |  |  |  |
| BCI\_Training\_UI | X |  |  |  |  |  |  |  |  |  |
| BCI\_Training\_Mode | X |  |  | X |  |  |  |  |  |  |

ADD TO TABLE 39 ABOVE BCI\_Training\_UI IN SECTION 10.12 FOR FIRST SUPPORTED AMI VERSION:

BCI\_Training\_Mode 7.1

**BACKGROUND INFORMATION/HISTORY:**

BIRD201

Contributor Michael Mirmak, Intel Corp

BIRD201.1

Technical clarifications and organizational changes to declare where content is positioned