Discussion Points on “Flow Correction” BIRD

04 January 2011

Statement of the issues

- BIRD120 states the issues as follows:

1. Section 2.3 of Section 10 (NOTES ON ALGORITHMIC MODELING INTERFACE AND PROGRAMMING GUIDE) describes a flawed reference flow. While the intent was to support non-LTI algorithms in the AMI_GetWave functions of the AMI models, Step 4 and Step 5, as described in Section 2.3 will only yield correct results with LTI AMI_GetWave algorithms.

2. In addition, Sections 2.1 and 2.2 allude to the existence of LTI (statistical) and non-LTI (Time Domain) flows, the specification contains only one detailed reference flow in Section 2.3 which does not differentiate between LTI Statistical, LTI Time Domain and non-LTI Time Domain flows.

3. Also, the IBIS ATM subcommittee, in attempting to incorporate Use_Init_Output into the correct flows concluded that Use_Init_Output added unnecessary complications to the flows, and decided to deprecate this AMI parameter.

Concerns with BIRD 120's solution to issue #2

Issue 2

```
In addition, Sections 2.1 and 2.2 allude to the existence of LTI
(statistical) and non-LTI (Time Domain) flows, the specification
contains only one detailed reference flow in Section 2.3 which does not
differentiate between LTI Statistical, LTI Time Domain and non-LTI Time
Domain flows.
```

- **Concerns with BIRD 120 approach**
  1. BIRD 120 attempts to solve this problem by defining a new flow whereby the outputs of **AMI_Init** may or may not be used by the EDA tool depending on the existence of **AMI_GetWave**.
     - This approach is intended to prevent "double-counting" of equalization that might be modeled in **AMI_Init** and **AMI_GetWave**, depending on the particular model implementation.
     - Unfortunately, this limits the model developer's flexibility to model different behaviors (LTI and non-LTI) in different functions.
  2. De-convolution is necessary with BIRD 120 in certain circumstances, namely when Tx **AMI_Init** returns an IR, Tx **AMI_GetWave** exists, and Rx **AMI_GetWave** does not exist.
     - De-convolution can produce undesirable mathematical artifacts
- **An alternative solution is presented in the following slides.**

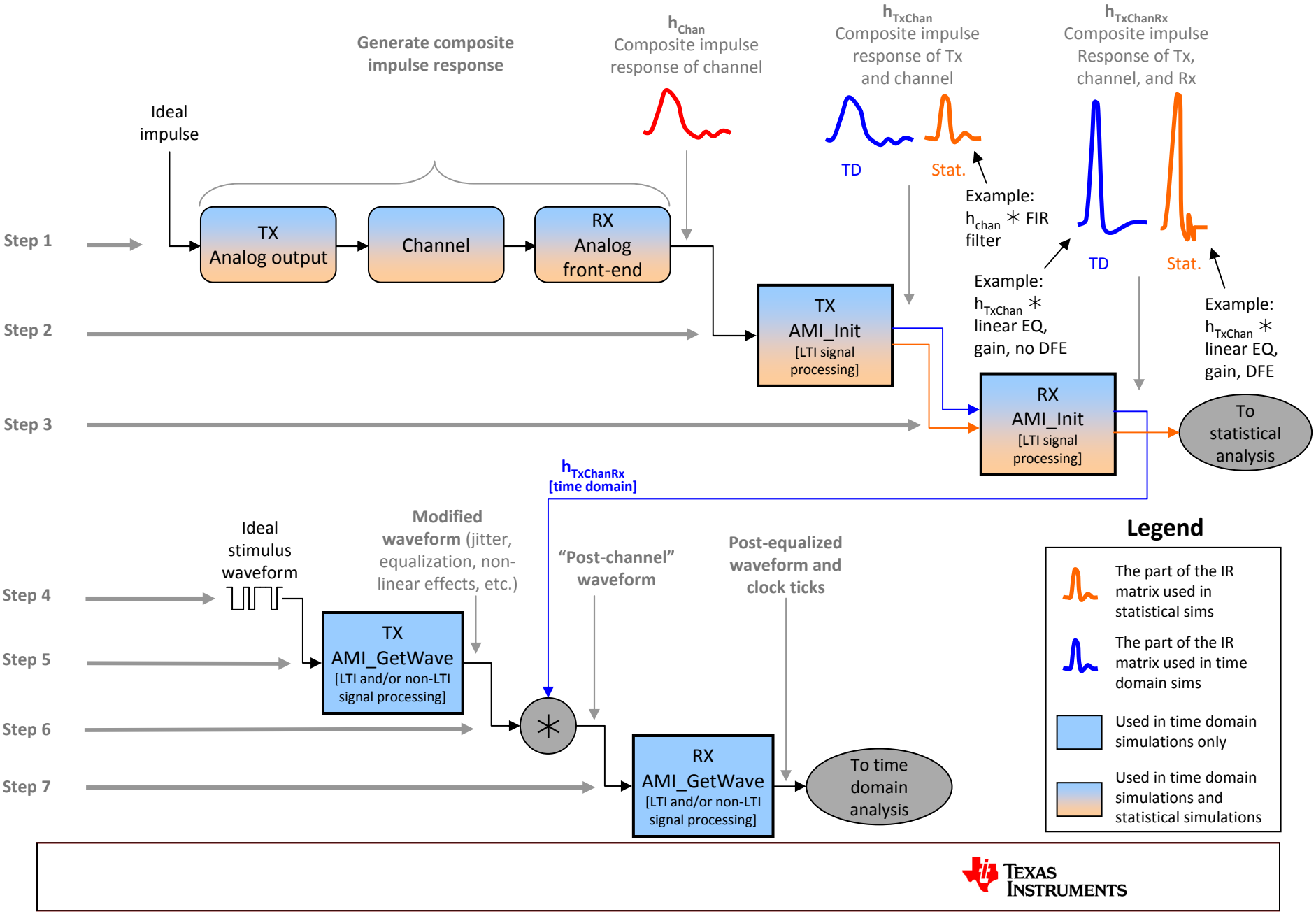
Background for issue #2

- IBIS-AMI functions **AMI_Init** and **AMI_GetWave** are conducive to different types of modeling.
  - **AMI_Init** strictly involves impulse response manipulation, which makes it ideal for modeling LTI signal processing blocks (linear filters, bandwidth limitations, etc.).
  - **AMI_GetWave** strictly involves waveform manipulation, which makes it ideal for modeling LTI and/or non-LTI signal processing blocks (signal compression, jitter, etc.).
- The consensus opinion on how the *statistical* and *time domain* simulation modes should be defined appears to be:
  - *Statistical* simulations involve only LTI behavior which comes from the impulse response manipulation done by the **AMI_Init** functions.
  - *Time domain* simulations involve LTI and/or non-LTI behavior, and thus both impulse response manipulation and waveform manipulation are allowed.
- The challenge then is to provide the AMI models and the EDA tool with the correct inputs for both modes.
  - This is particularly challenging when the same signal processing step (i.e. a Tx FIR filter) is modeled both in **AMI_Init** as a frequency-domain filter *and* in **AMI_GetWave** as a time-domain filter.
  - In the current version of the standard, such an implementation would lead to the Tx FIR filter being “double-counted,” once in **AMI_Init** and once again in **AMI_GetWave**.

Proposed solution to statistical / time domain flow problem

- Why not have the AMI_Init functions return two impulse responses: a *statistical modified impulse response* and a *time domain modified impulse response*?
  - *Statistical modified impulse response:* Channel IR modified by Tx **AMI_Init** and/or Rx **AMI_Init** which is appropriate/correct for statistical simulations in the sense that it may include LTI behavior and/or linear approximations to non-LTI behavior.
    - Some, all, or none of this behavior may be simultaneously modeled in **AMI_GetWave**.
    - For example, a model developer may chose to model a Tx FIR filter in Tx **AMI_GetWave**. In order for this behavior to be present in statistical simulations, in which **AMI_GetWave** is not called, the same filter must be modeled in **AMI_Init**; but in order to avoid double-counting this filter in time domain simulations, it is necessary to include the filter in the statistical modified IR and not the time domain modified IR.
  - *Time domain modified impulse response:* Channel IR modified by Tx **AMI_Init** and/or Rx **AMI_Init** which is appropriate/correct for time domain simulations in the sense that it does not contain any filtering/behavior which is also modeled in **AMI_GetWave**.
    - None of the behavior modeled in the time domain modified IR is also modeled in **AMI_GetWave**.
    - For example, some linear filters may be modeled in **AMI_GetWave**. In such cases, these filters would not also be part of the time domain modified IR, otherwise it would be double-counted in a time domain simulation.
- This allows the correct modified impulse response for both simulation modes to be available at all times.
- Model developers would retain the flexibility offered by the current standard to model different behaviors (LTI and non-LTI) in different functions whilst maintaining a consistent and correct model for all simulation modes.
- This concept is illustrated on the following slide.

Solution to Statistical / Time domain flow problem



Implementation details

- The existence of both time domain and statistical modified IRs requires a minor modification to the data structure used in **AMI_Init** function calls.
- More specifically, a back-to-back impulse response matrix can be used to represent this dual IR instead of the single impulse response matrix used currently.
- The two matrices must have an identical format, i.e. they must have the same row_size, aggressors, sample_interval, and bit_time parameters.
- The second matrix is stored the same way in memory as the first one.
  - A single-dimensional array of floating point numbers which is formed by concatenating the columns of the impulse response matrix, starting with the first column and ending with the last column.
- The second matrix is placed immediately after the first matrix in memory.
  - This way the second matrix can be retrieved/identified using the same method as the first matrix with an offset corresponding to the length of the first matrix.

Example: Simultaneous time domain and statistical mode simulations

- With this proposal it is possible to run statistical mode and time domain mode simulations simultaneously.
- The statistical modified IRs returned by the Tx **AMI_Init** and Rx **AMI_Init** functions would be used for statistical analysis.
- The time domain modified IRs returned by the Tx **AMI_Init** and Rx **AMI_Init** functions, together with the waveforms produced by Tx **AMI_GetWave** and/or Rx **AMI_GetWave** would be used for time domain analysis.
- If desired, all possible linear approximations to the equalization/filtering done in the Tx and Rx can be included in the statistical modified IRs produced by the **AMI_Init** functions, thus producing a coherent and complete statistical simulation.
- LTI and non-LTI behavior modeled in **AMI_Init** and **AMI_GetWave** can be effectively “cascaded” in time domain mode simulations without the risk of double-counting equalization.

Example: Optimization in Rx AMI_Init

- Some models perform “optimization” in the Rx **AMI_Init** function.
  - Optimization could mean that certain parameters/variables in the Rx **AMI_GetWave** function (i.e. DFE tap weights, analog equalizer settings, etc.) are initialized to optimum or near-optimum values during the call to the Rx **AMI_Init** function.
- In order for optimization to be effective, the Rx **AMI_Init** function must be aware of the equalization (EQ) done by the Tx.
- Model developers may choose to model Tx EQ as a frequency-domain filter in **AMI_Init** or as a time-domain filter in **AMI_GetWave**.
- To ensure that Rx models which perform optimization benefit from “seeing” the EQ done by the Tx model, it is recommended that Tx models always provide a modified IR to the Rx which includes the EQ done by the Tx; and this is possible with the proposal outlined in this presentation.
  - In cases where the Tx model developer chooses to model Tx EQ as a frequency-domain filter in **AMI_Init**, then both the time-domain and frequency-domain modified IRs returned by Tx **AMI_Init** will contain EQ.
  - In cases where the Tx model developer chooses to model Tx EQ as a time-domain filter in **AMI_GetWave**, then he/she can include EQ in the statistical modified IR (for the benefit of Rx models which perform optimization) and *not* include EQ in the time domain modified IR, since the EQ will be modeled in **AMI_GetWave** instead.

Example: Tx AMI_Init returns an IR, Tx AMI_GetWave exists, and Rx AMI_GetWave does not exist

- With BIRD 120, this rare but perfectly-legal scenario would require the EDA tool to de-convolve the input to Rx **AMI_Init** and the output of Rx **AMI_Init** to get the impulse response of the Rx filter alone, h_{Rx}.
  - This allows the EDA tool to then convolve h_{Rx} with the channel impulse response, h_{Chan}, to get h_{ChanRx}.
  - h_{ChanRx} does not include any impulse response modification done by Tx **AMI_Init**; therefore it can be convolved with the output of Tx **AMI_GetWave** without the risk of double-counting any Tx equalization.
  - If de-convolution were not performed in this case, then the same equalization done in Tx **AMI_Init** and again Tx **AMI_Getwave** would be double-counted in a time domain simulation.
- With *this* proposal, de-convolution is not necessary since the Tx **AMI_Init** function would produce two modified IRs, one which includes EQ (for the benefit of the Rx) and one which does not include EQ (since the EQ will be modeled in the Tx **AMI_GetWave** function).
  - A well-designed Rx model consisting only of AMI_Init would use the statistical modified IR provided to it as an input, and it would estimate the LTI equalization/filtering that needs to be done in the Rx.
  - It would convolve the *input* statistical modified IR with said equalization/filtering to produce the *output* statistical modified IR.
  - It would convolve the *input* time domain modified IR with said equalization/filtering to produce the *output* time domain modified IR.
  - Rx **AMI_GetWave** does not exist, so time domain simulations would effectively stop after the waveform modified by Tx **AMI_GetWave** is convolved with the time domain modified IR.

Recapping the benefits of this solution

- This proposal accomplishes the main goals of BIRD 120 and does so without sacrificing the flexibility currently afforded to the model developer *and* without the need for de-convolution in any situation.
  - No need to run time domain and statistical mode simulations separately, just process the appropriate modified IRs (see example on slide 8).
  - The availability of both modified IRs makes it possible and simple for the Rx model to do "optimization" if it is designed to do so (see example on slide 9).
  - Double-counting of equalization can be easily avoided since the time domain and statistical modified IRs are separate and simultaneously available in every simulation. Furthermore, de-convolution is not necessary to avoid double-counting (see example on slide 10).
  - Flexibility to model different signal processing blocks (LTI and non-LTI) in different functions (**AMI_Init** and **AMI_GetWave**) is preserved going from IBIS 5.0 to IBIS 5.1.
- Additional benefits include:
  - 5.0 models should still function in this framework. All the EDA tool needs to do is duplicate the IR to form the back-to-back IR matrix.
  - In theory, 5.0 models and 5.1 models could be used simultaneously. 5.0 models would only make use of one of the IRs, while 5.1 models could potentially make use of both.

Backup

Example: Split and non-split models

- Consider two different Tx models:
  1. A “split” model because it models a portion of the Tx behavior in **AMI_Init** (for example, an FIR filter) and a portion in **AMI_Getwave** (for example, jitter).
  2. A non-split model because it models all its behavior in **AMI_Getwave** (for example, an FIR filter and jitter).
- It makes sense for the Tx model developer to include the following in the statistical and time domain modified IRs:

Tx model	What should be included in the statistical modified IR	What should be included in the time domain modified IR
1	FIR filter	FIR filter
2	Equivalent FIR filter	No FIR filter

- In both cases, the Rx **AMI_Init** function will utilize the statistical modified IR to do its optimization, since this is the IR which includes the Tx FIR filter.
- In both cases, the Tx model developer is inclined to include an FIR filter in the statistical modified IR for two reasons:
  1. It will make for the most accurate model in statistical simulations.
  2. It will allow the Rx to perform optimization effectively.