A problem in AMI flow “7j”

Arpad Muranyi
Mentor Graphics Corporation

ATM Teleconference June 22, 2010

The following discussion is in reference to AMI flow "7j" (dated June 7, 2010) that can be found on the ATM web site under the Work Archive section:

http://www.vhdl.org/pub/ibis/macromodel_wip/archive/20100607/arpadmuranyi/AMI%20Flows%207j/AMI_Flows_7j.pdf

The problem exists when Rx Init contains optimization

- **With Boolean settings TTF (GetWave_Exists = T, Init_Returns_Impulse = T, Use_Init_Output = F) the TX AMI model maker says:**

  "Don't use the modified impulse response that is generated by Tx Init because Tx GetWave will do that modification (again) the same way or better."

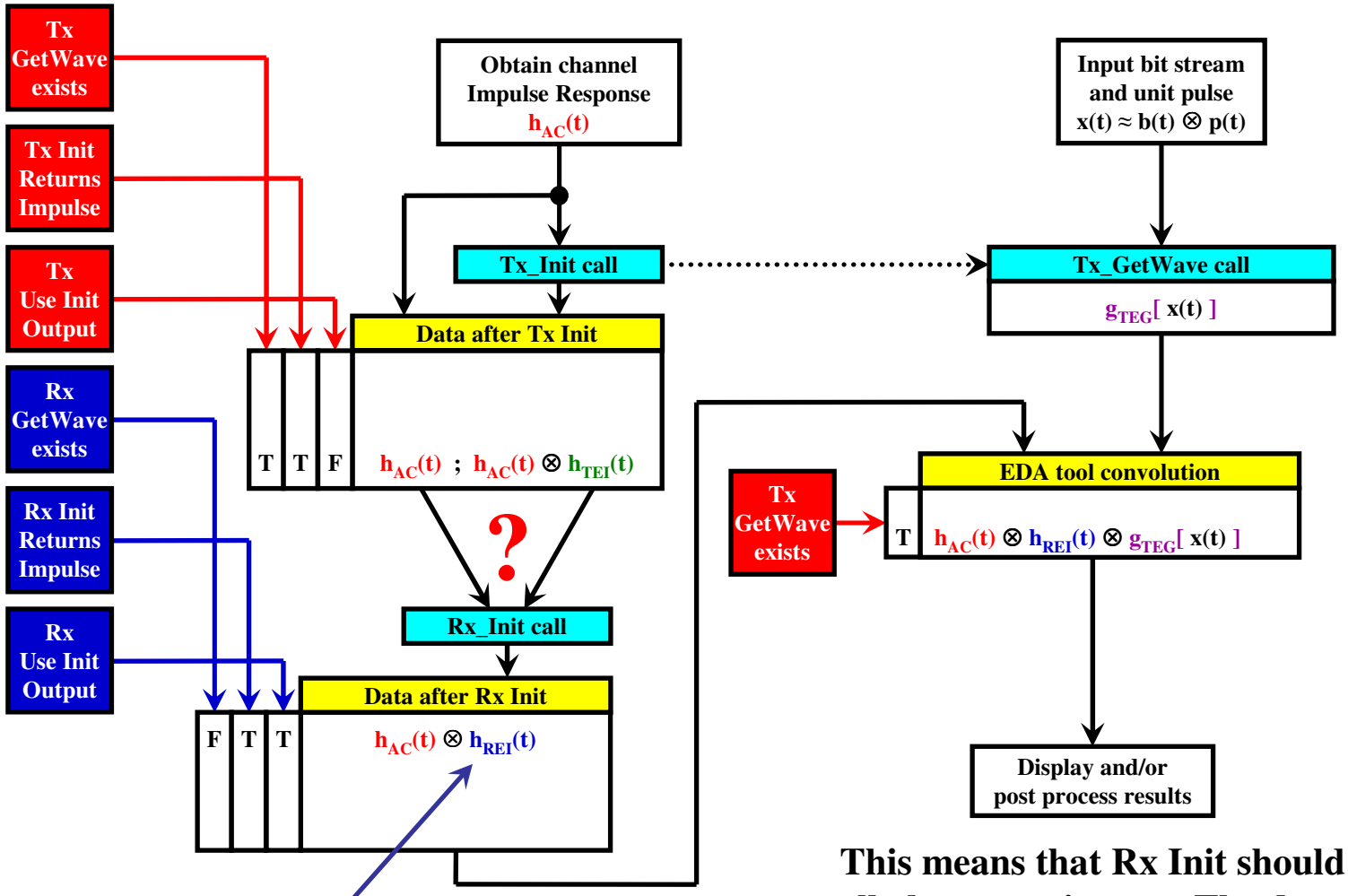
Problem:

- **The flow control selectors, including "Use_Init_Output" are on the left side in flow 7j. According to this flow, when Tx Use_Init_Output = False, the input of Rx Init is the same as the input of Tx Init, i.e. it receives an unmodified impulse response. (This was done this way to eliminate the need for de-convolution).**
- **When Rx Init contains an (optional) optimizer, the optimizer will find incorrect settings for the Rx Init filter(s) this way.**
- **On the other hand, if we found a way to feed a modified impulse response to Rx Init (the output from Tx Init), de-convolution will be necessary to eliminate double counting, even though the (optional) optimizer in Rx Init would find correct filter settings, and its output would be correct this way.**

Question:

- **What rule for the Boolean flow controls would give the correct input for Rx Init?**

From AMI flow #7 - TD simulations with Tx GetWave only



Due to optimization, $h_{REI}(t)$ depends on $h_{TEI}(t)$!

This means that Rx Init should really have two inputs. The data to be operated on should contain $h_{AC}(t)$ only, but the data which is used for optimization should include $h_{AC}(t) \otimes h_{TEI}(t)$.

Notes:

1. *Use_Init_Output is optional. If not declared it defaults to TRUE.*
2. *When GetWave_Exists = FALSE, both Use_Init_Output and Init_Returns_Impulse must be TRUE*
3. *For statistical simulations Use_Init_Output is ignored and is treated as if it was TRUE*

Optional solutions

- **Ignore Tx Use_Init_Output = F when there is optimization in Rx Init. This ensures that the Rx Init optimizer finds correct filter settings, but it requires de-convolution. Problem: Since there is no information in the model or its parameters to tell the EDA tool about the presence or absence of an optimizer in Rx Init, the EDA tool does not have a way to determine when to ignore Tx Use_Init_Output and do the de-convolution.**
- **Ignore Tx Use_Init_Output = F for the TTF FTT and TTF TTT cases all the time. This way four (4) cases would require de-convolution whether or not the Rx Init contains an optimizer.**
- **Move the Boolean flow controls to the GetWave side of the flow diagram (so that the output of Tx Init goes into Rx Init unconditionally). This way two more cases would require de-convolution (TTF TFX and TTF TXF) whether or not the Rx Init contains an optimizer.**
- **Feed two impulse responses (inputs) into Rx Init by doubling the size of the currently used impulse_matrix. The main input would be the data which is modified by Rx Init (modified or unmodified impulse response, depending on the value of the Use_Init_Output Boolean). The second input would always come from the output of Tx Init (modified impulse response) to be used by the Rx Init optimizer. This method has the benefit of not needing any de-convolution, nor any special rules about ignoring or using Tx Use_Init_Output, and the latest flow "7j" wouldn't need to be changed (other than indicating the double input to Rx Init).**

Other possibilities:

- **Don't allow self optimization for the Tx case TTF. This rule can make certain models incompatible with each other.**
- **Introduce the Init_Returns_Filter Boolean. This feature could also make certain models incompatible with each other.**

Using two inputs for Rx Init has the most benefits

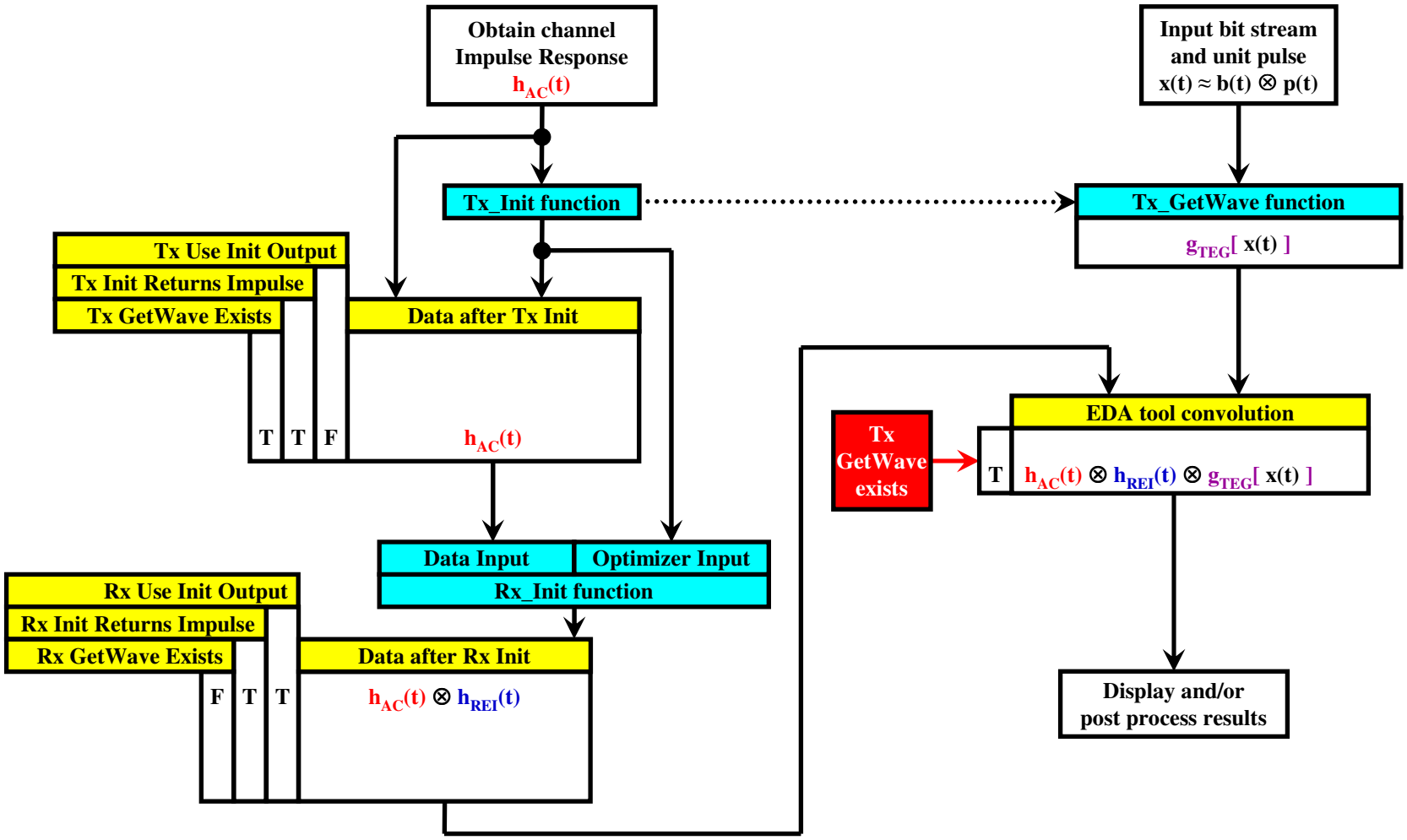
Currently, the main input to the Init function is an "impulse_matrix", containing the impulse response of the "primary channel" and "the impulse responses from aggressor drivers to the victim receiver". The related function arguments, row_size, aggressors, sample_interval, bit_time help the code in the Init function to properly parse the impulse_matrix into its components (primary and cross talk impulse responses).

It would be very easy to extend this format so that the entire matrix is doubled in size to give Rx Init one version of the impulse response for its filter and another for its optimizer.

Benefits:

- **No change to the footprint of the DLL is necessary. Section 3.1.2.1 in Section 10 (on the impulse_matrix), will need to be updated to explain that the first impulse_matrix ("filter matrix") represents the data that is to be modified by the Rx Init function's filter, and the second impulse_matrix ("optimizer input matrix") is used as a read-only input by the Rx Init function's optimizer if it has one. If the optimizer works on the same data that it needs to modify, i.e. when Tx Use_Init_Output = True, then the data in the "main" and the "secondary" matrices will be identical. If Rx Init does not have an optimizer, it will not read the second input.**
- **No need for any additional control parameters, or fancy rules about when to ignore what**
- **No changes are needed to the existing flow (7j)**
- **Each case works without de-convolution**

TD simulations with Tx GetWave only & Rx Init optimization



Notes:

1. *Use_Init_Output is optional. If not declared it defaults to TRUE.*
2. *When GetWave_Exists = FALSE, both Use_Init_Output and Init_Returns_Impulse must be TRUE*
3. *For statistical simulations Use_Init_Output is ignored and is treated as if it was TRUE*