Pre/de-emphasis buffer modeling with IBIS

IBIS Summit at DATE05

München, Germany

March 11, 2005

Arpad Muranyi
Signal Integrity Engineering
Intel Corporation
arpad.muranyi@intel.com

Kuen Yew Lam
Signal Integrity Engineering
Intel Corporation
kuen.yew.lam@intel.com

Options for modeling pre/de-emphasis buffers in IBIS

- **Model the building blocks of the buffer with independent [Model]s and tell the user to wire them up**
  - This approach was used initially for many models but required manual editing of files and/or simulation schematics
- **The legacy [Driver Schedule] keyword provides a reasonable solution to model pre/de-emphasis buffers**

  http://www.eda.org/pub/ibis/summits/jan05/muranyi.pdf
  - Eliminates the need for manually connecting [Model]s to make a complete buffer
  - Uses no more than IBIS v3.2 syntax
  - Useful for tools not supporting the *-AMS extensions of IBIS
  - Reasonably good correlation with transistor level model
  - There are a few unsolved problems
- **The *-AMS language extensions of IBIS v4.1 provide means to solve the outstanding problems**
  - The issues around C_comp compensation can be solved
  - Switching into an unfinished edge, and
  - Data pattern dependent behavior can be added
  - Any other features and capabilities can be added as needed, such as
  - Frequency and/or voltage dependent C_comp, etc…

Pre/de-emphasis buffer review

In most of the current two-tap designs the "emphasis stimulus pattern" is a one bit delayed and inverted copy of the "input stimulus pattern"

This is not necessarily true for all pre/de-emphasis buffer designs. The delay may not be a one bit duration in each design, and multi-tap configurations would usually have a more complicated stimulus logic.

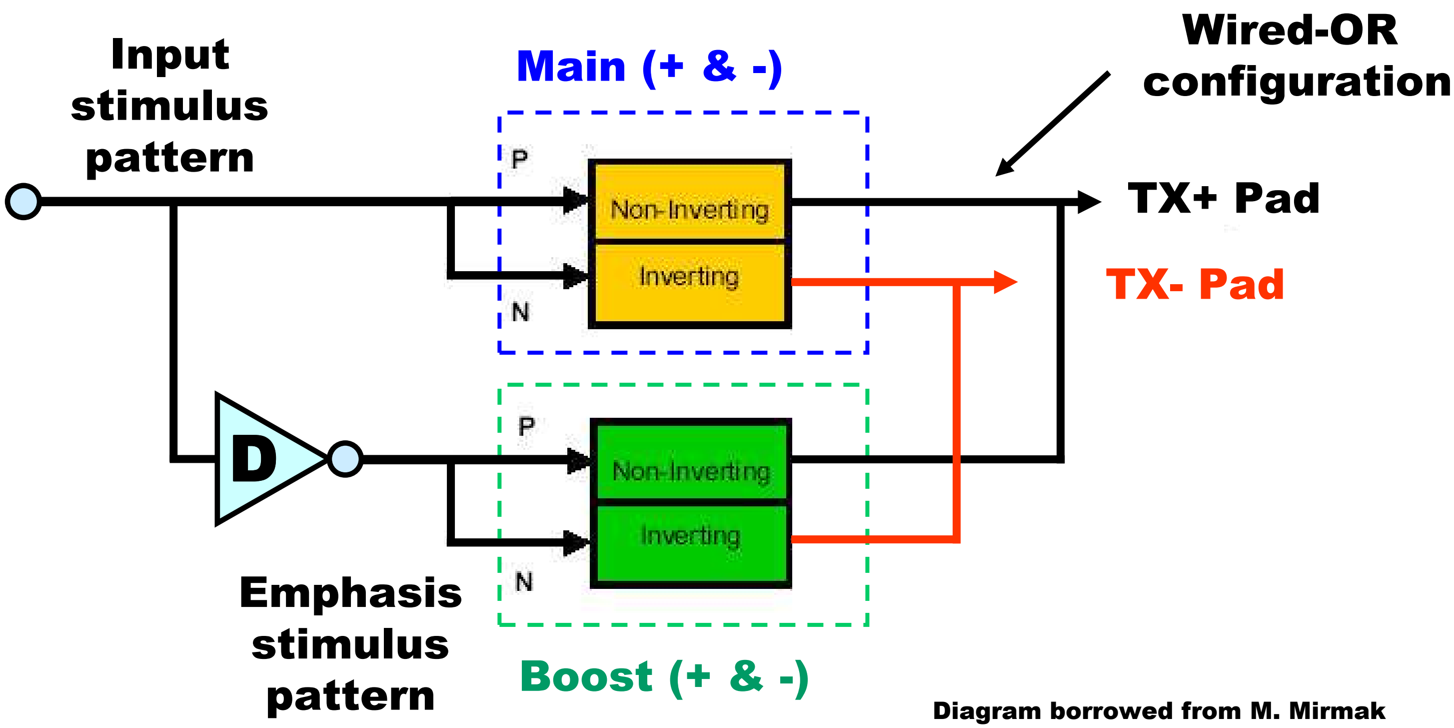
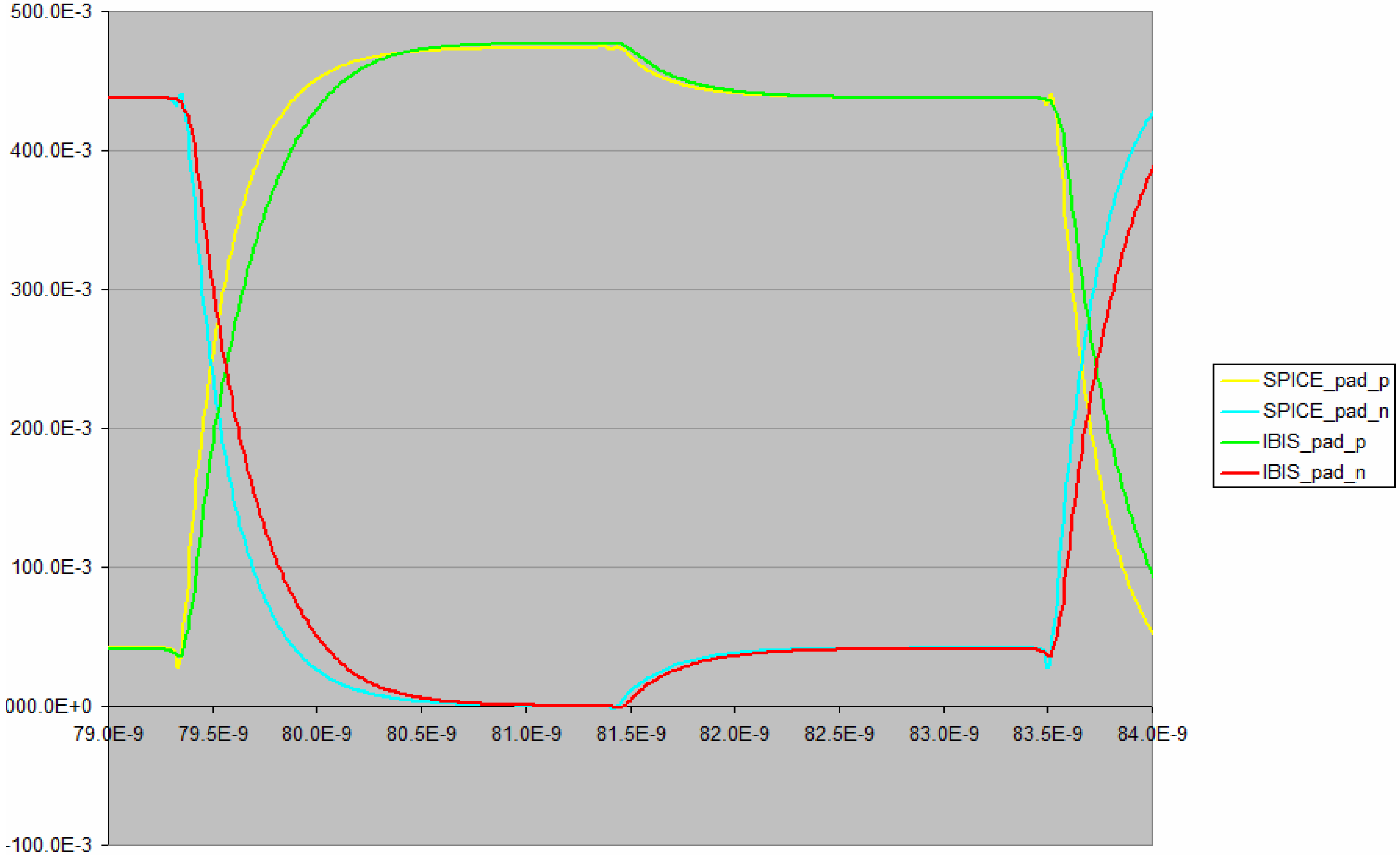


Diagram borrowed from M. Mirmak

C_comp issues

- **The IBIS specification says that C_comp should be placed into the "top level" model and should represent the total buffer capacitance**
- **This is easy for the model maker, but tool vendors need to answer some difficult questions:**
  - How is the C_comp compensation done?
    - independently, inside the Main and Boost [Model]s?
    - collectively?
  - If independently, how is the capacitive loading effect of the neighboring model(s) accounted for in the compensation algorithm?
  - How is the total C_comp divided between the Main and Boost buffers?
  - Is the C_comp compensation correct for each transition?
    - strong to strong bit
    - strong to weak bit
    - weak to strong bit
- **More C_comp related information:**
  - http://www.eda.org/pub/ibis/summits/apr04/mirmak2.pdf
  - http://www.eda.org/pub/ibis/summits/oct04/mirmak2.pdf
- **A constant C_comp value may not be accurate enough at GHz speeds**
  - Frequency and/or voltage dependence may be important, which can only be modeled with the IBIS v4.1 language extensions

Waveforms with independent C_comp compensation

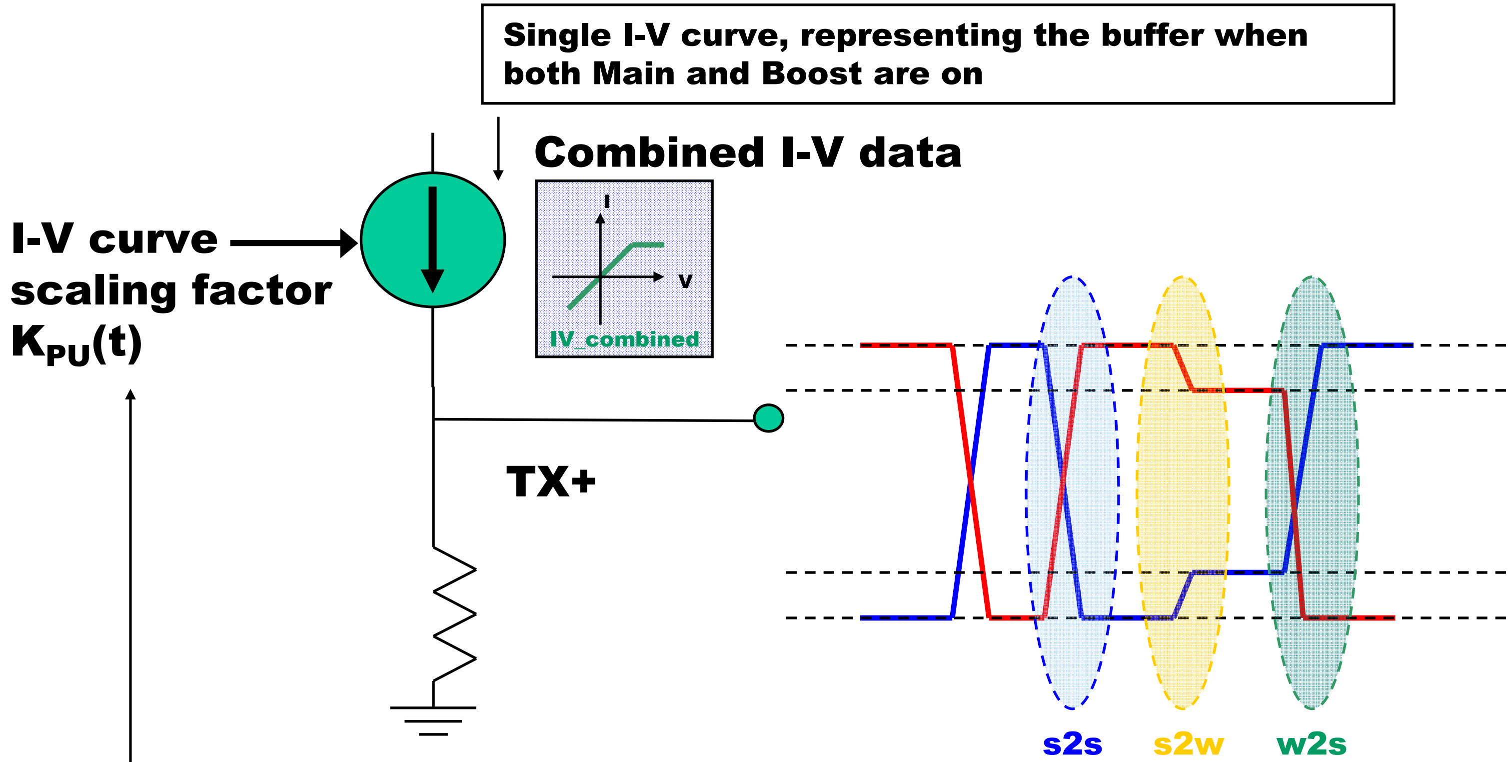


This simulation uses two separate VHDL-AMS models representing the "Main" and "Boost" blocks, in which the C_comp compensation is done independently. The reduced edge rate is a result of the two blocks loading each other.

Solving this problem with a modified algorithm

- **How about combining the main and boost buffers into one single model?**
  - **Have only one I-V curve, representing the Main + Boost I-V curves**
  - **Separate V-t curves for the different transition edges**
    - Strong to strong bit
    - Strong to weak bit
    - Weak to strong bit
  - **Use *-AMS to pick the right V-t (K_{PU}(t)) curves to use, and scale the IV curve accordingly**
  - **No need to change the C_comp compensation equations**

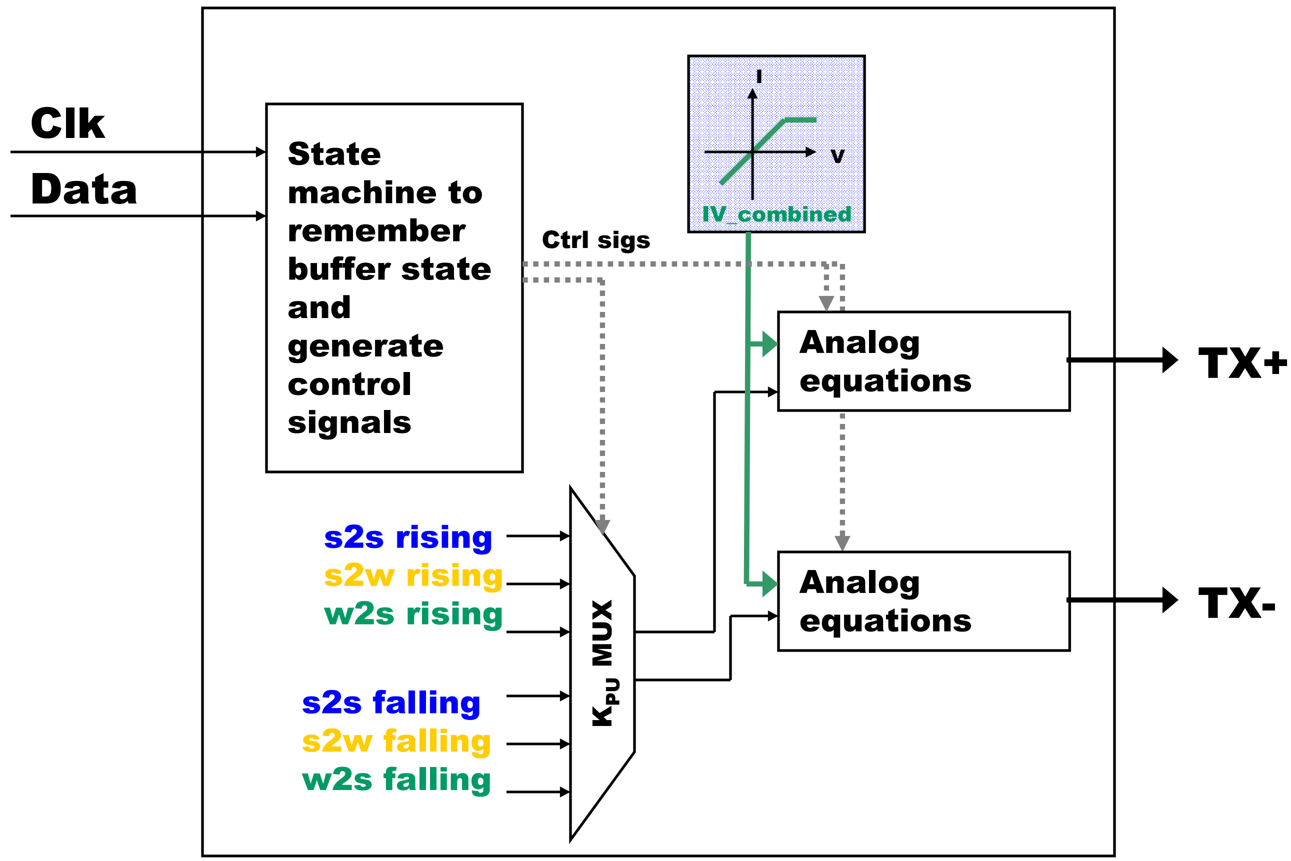
Combine the Main and Boost blocks into one model



3 types of state transitions, resulting in 6 different $K_{PU}(t)$ waveforms:

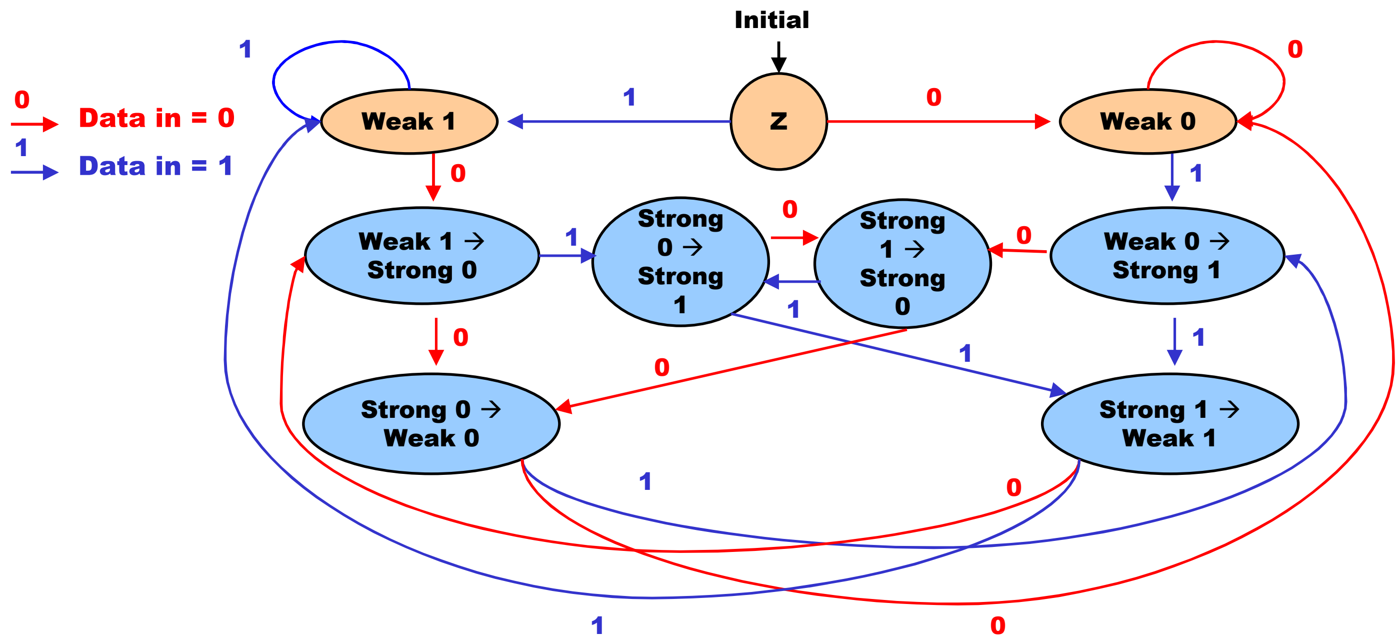
- **Strong-strong: Main + Boost switching in same direction**
- **Strong-weak: Boost switching (Main constant)**
- **Weak-strong: Main switching (Boost constant)**

Block diagram of combined model




```
-- One set of analog equations
--------------------------------------------------------------------------
Ipc_p_0    == -1.0            * Lookup("IV", Vpc_p_0, I_pc, V_pc); -- Power clamp eqn's
Ipu_p_0    == -1.0 * k_pu_p_0 * Lookup("IV", Vpu_p_0, I_pu, V_pu); -- Pull up eqn's
Igc_p_0    ==                   Lookup("IV", Vgc_p_0, I_gc, V_gc); -- Ground clamp eqn's
.....
```

State machine diagram for the logic

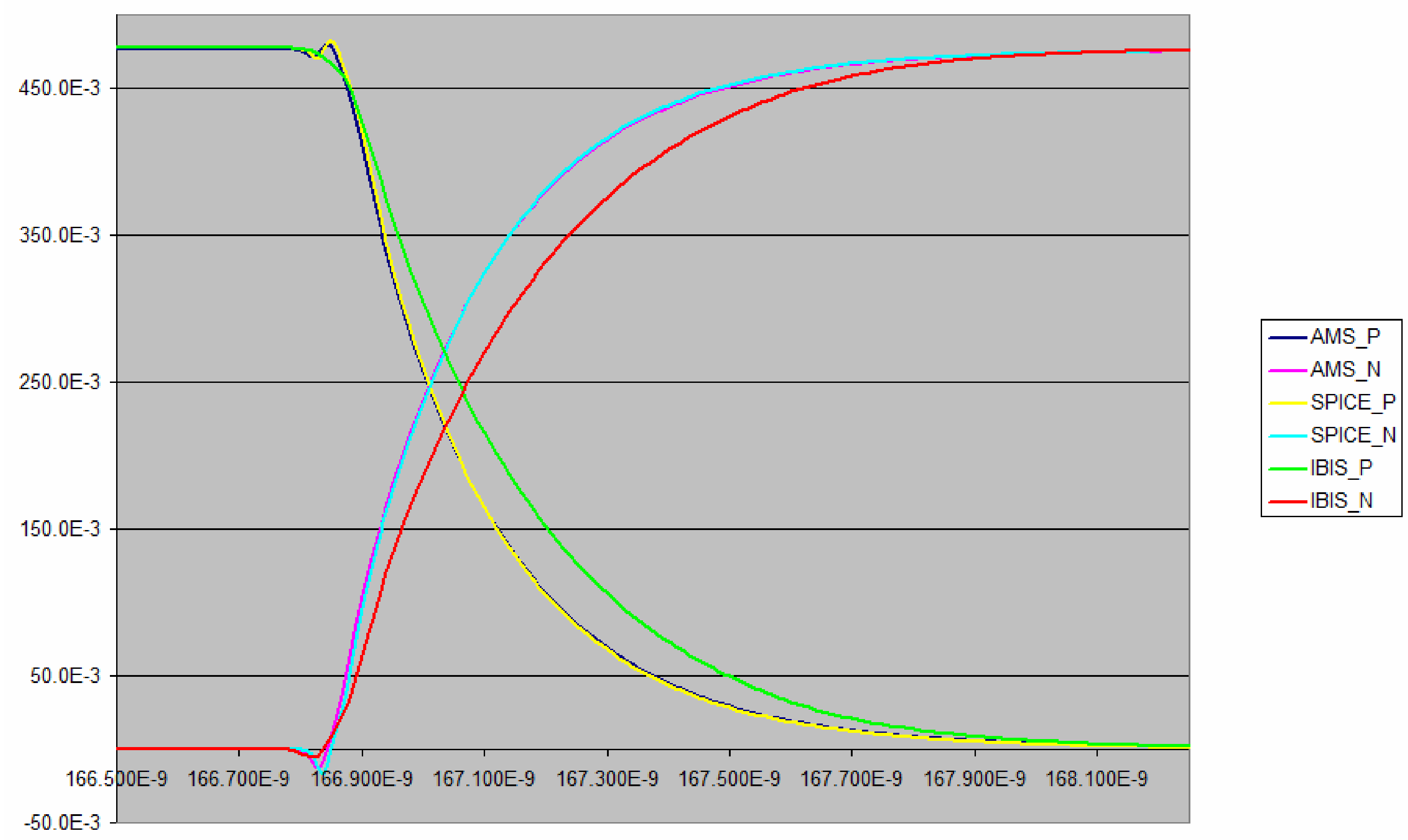


- Each blue bubble represents a buffer state transition (6 of them in total, one for each $K_{PU}(t)$ waveform)
- Orange bubbles represent no state changes
- State changes occur at clock edges

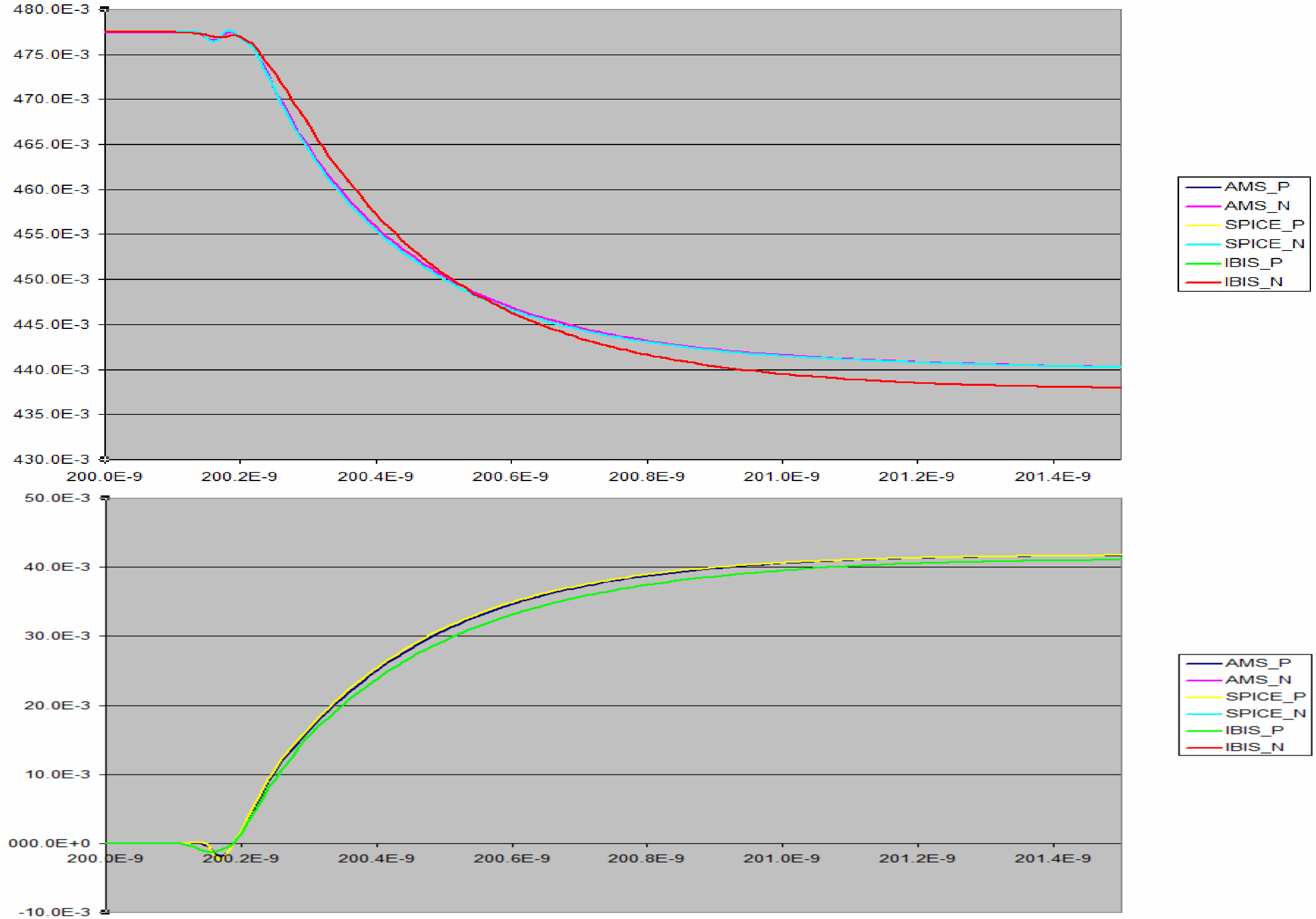
Data extraction

- **I-V curves**
  - *Only ONE I-V curve generated, for when both Main and Boost are on*
  - *Can re-use existing IBIS data (Sum Main and Boost I-V)*
  - *No need to worry about double-counting Internal terminations (between Main and Boost buffers, as in previous techniques)*
- **V-t curves**
  - *Generate V-t curves for the SIX different transition types*
  - *No need to worry about double-counting Internal terminations*
- **Same C_comp extraction methodology as before, but C_comp doesn't need to be split between buffer blocks**

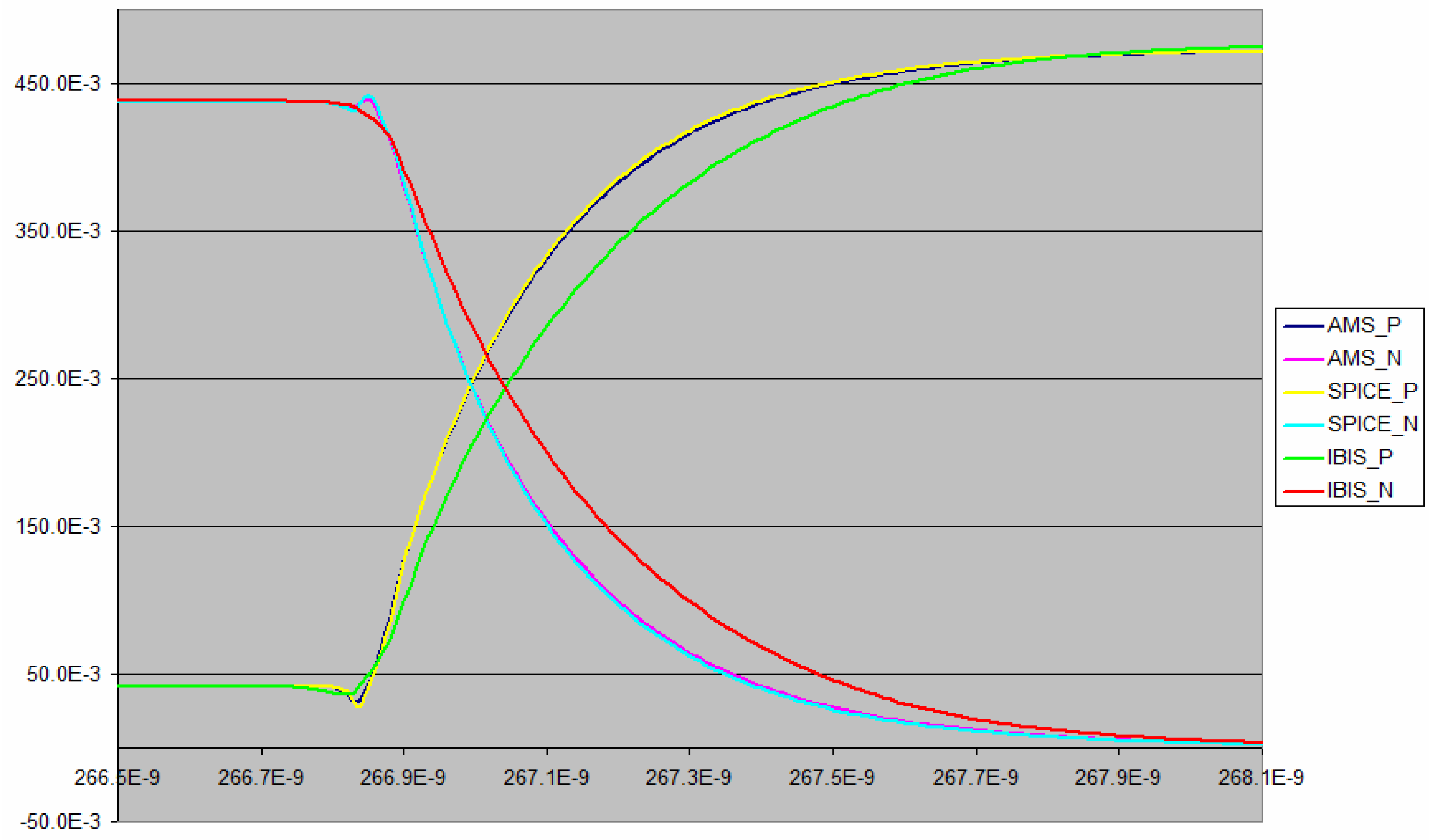
Strong bit to strong bit transition overlay



Strong bit to weak bit transition overlay



Weak bit to strong bit transition overlay

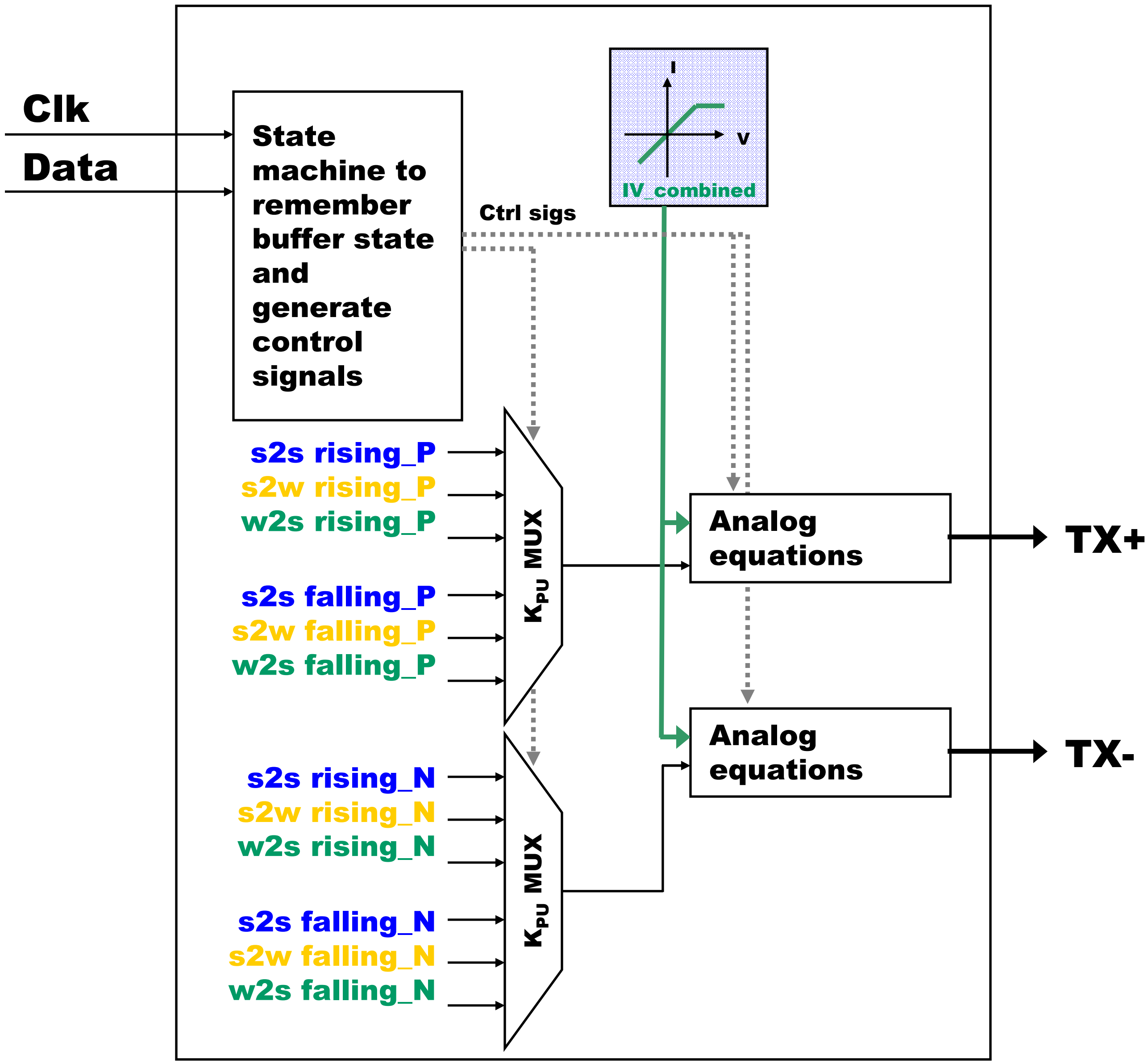


Notes on correlation results

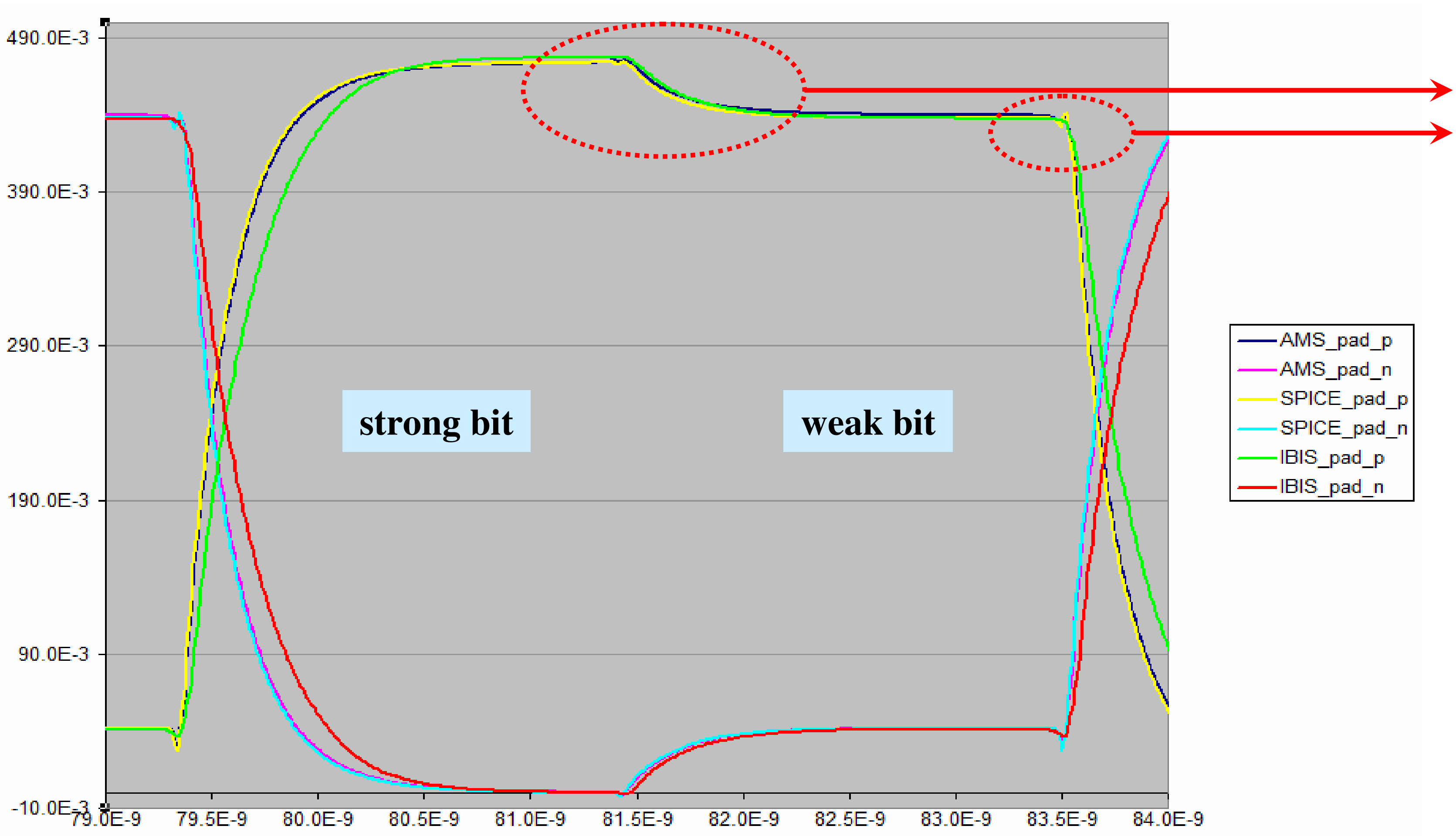
- **Excellent match between SPICE and *-AMS model on all transitions**
  - **No tweaking of the I-V & V-t curves and C_comp was necessary**
  - **Original C_comp compensation algorithm can still be used**

    http://www.eda.org/pub/ibis/summits/jun03b/muranyi1.pdf (pg. 9)
  - **This *-AMS model assumes a perfectly symmetric differential buffer in which the V-t characteristics are identical for the P and N ouputs**
  - **A small change in the code can account for the asymmetry effects also (next page)**
- **However, this was done with the clock slowed down, such that the V-t curves have settled**
  - **In this case, clock was slowed down from 480 MHz to 30 MHz**
  - **At full speed, some "switching into an unfinished edge" exists**
- **Effects, such as switching into an unfinished edge, or data pattern dependent behaviors are not addressed in this presentation**

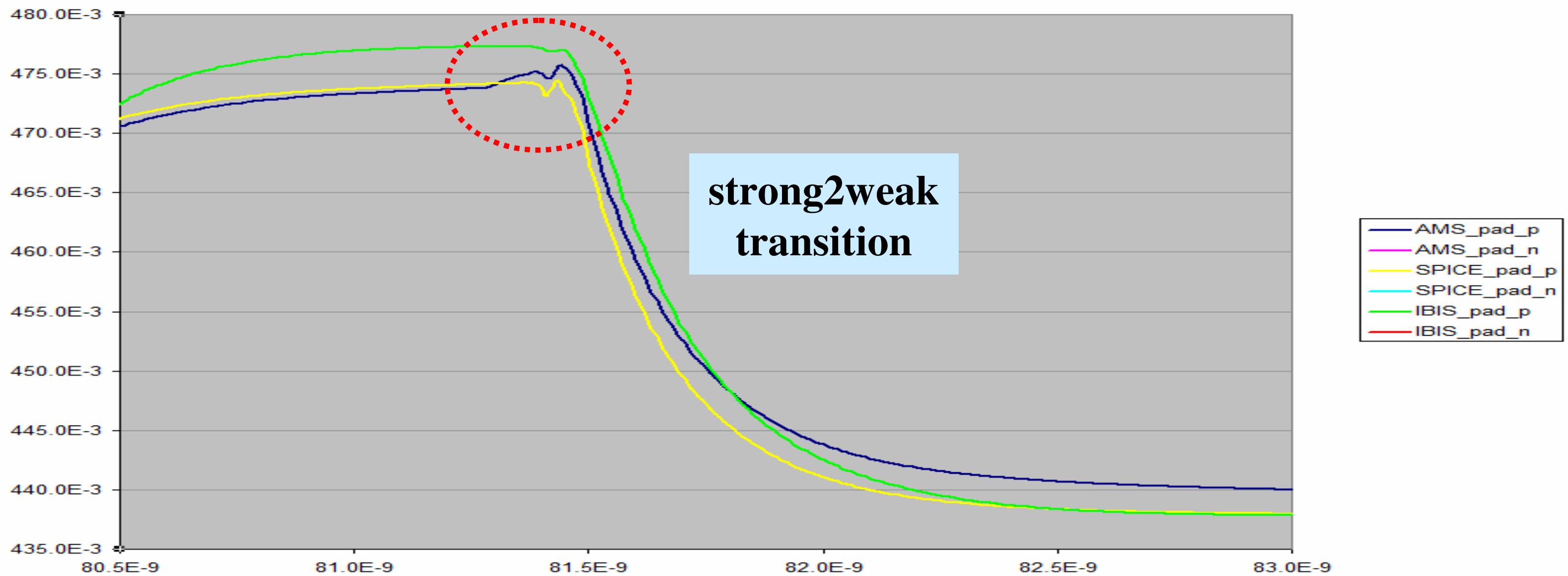
Block diagram with asymmetric differential capabilities

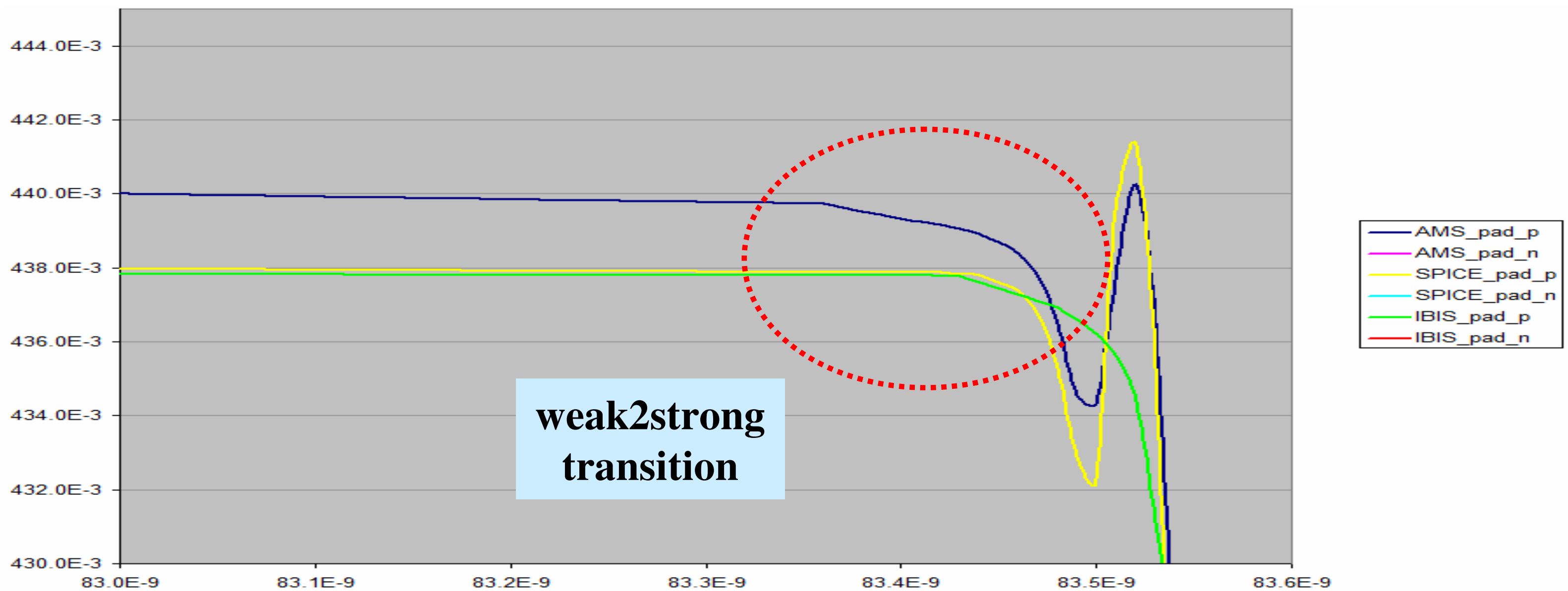


Simulation results at full speed (480MHz)



Details reveal some discontinuities due to unfinished edge





Conclusions

- **This study complements and completes the initial work:**

  http://www.eda.org/pub/ibis/summits/apr04/muranyi.zip
- **The VHDL-AMS model of this presentation simulates ~2.5x faster than the model developed above**
- **This model can also include the full differential buffer characteristics, discussed at:**

  http://www.eda.org/pub/ibis/summits/oct03/muranyi.pdf
- **Data required for this new approach**
  - **I-V curve is obtained for Main + Boost driving together**
  - **V-t curves need to be to be generated for each switching edge**
  - **C_comp, measured as usual for the complete buffer**
- **Next steps**
  - **Solve switching into an unfinished edge problem**
  - **Add data pattern dependent behavior effects**
  - **Add frequency and/or voltage dependent C_comp**
  - **Test with other interfaces**