A VHDL-AMS buffer model using IBIS v3.2 data

IBIS Summit at DAC 2003

Marriott Hotel, Anaheim, CA

June 5, 2003

Luca Giacotto
Université Joseph Fourier
lgiacott@libero.it

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

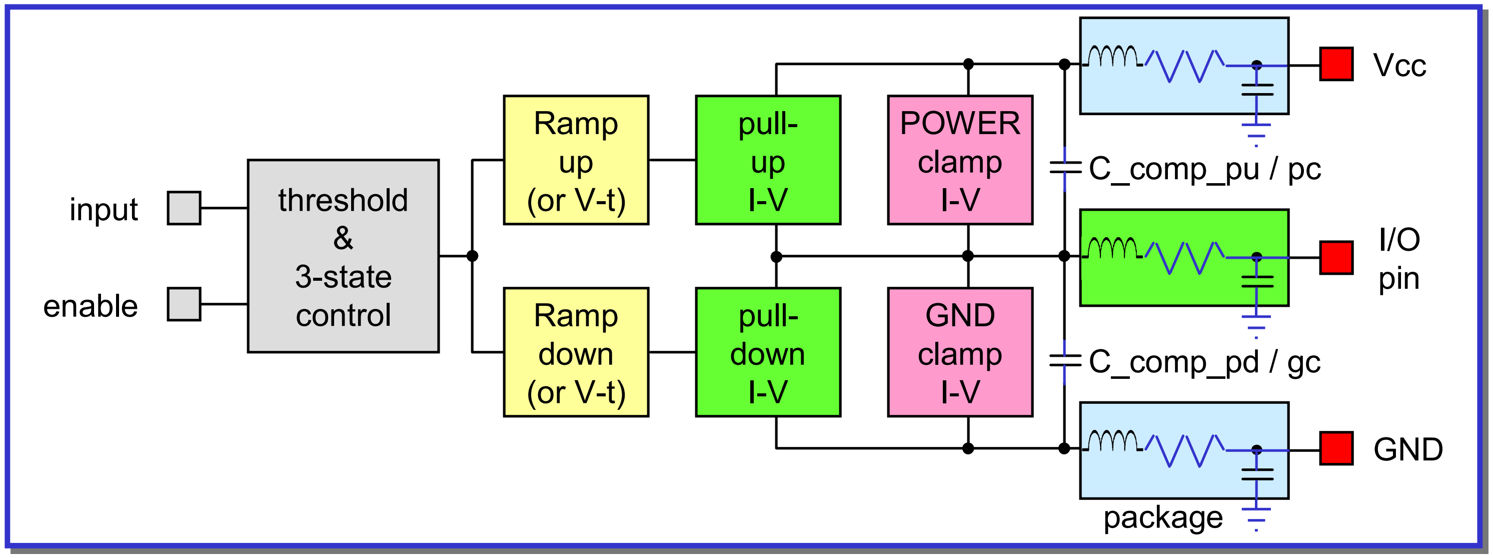
Outline

- **Motivation**
- **IBIS model block diagram review**
- **The system of two equations, two unknowns**
  - Equation
  - Solution
  - VHDL-AMS implementation
- **Waveform overlay with HSPICE B-element**
- **Solving the “DDR problem”**
  - the multi VT-tables approach
- **Summary**

Motivation

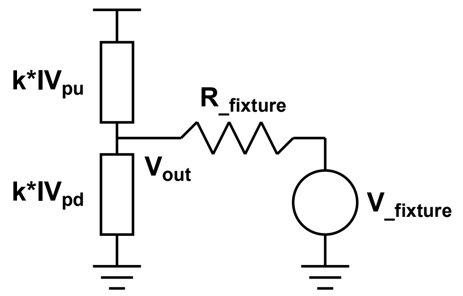
- **This presentation was written to explain the algorithms of a basic I/O buffer model written for IBIS in VHDL-AMS**
  - The presentation is accompanied by a VHDL-AMS file which is made available freely for anyone interested
  - This is done to encourage the use of the *-AMS extensions of IBIS for improved behavioral modeling
- **Demonstrate the usefulness of using the *-AMS extensions of IBIS with a practical example that solves an existing problem**
  - An enhanced version of the model demonstrates how problems can be solved by writing better algorithms
- **This presentation is NOT intended to be an introduction to the VHDL-AMS language**

Block diagram of an I/O buffer model



- **The logic front end controls the state of the output**
  - This can be done with purely digital equations
- **The PU and PD IV curves describe the steady state characteristics**
- **The Ramps or Vt curves describe the transient characteristics**
  - Ramps or Vt curves are used to scale the PU and PD IV curves with respect to time to account for the partially on/off transistors during transients
- **The POWER_cl and GND_cl IV curves describe the clamps and static on-die terminations**
  - These are always “ON”, no variations with respect to time are allowed
- **The passive package circuit is modeled separately from the buffer**

The system of two equations, two unknowns



$$0 = k_{pu}(t) \cdot IV_{pu}(V_{wfm1}(t)) - k_{pd}(t) \cdot IV_{pd}(V_{wfm1}(t)) - I_{out}(V_{wfm1}(t))$$

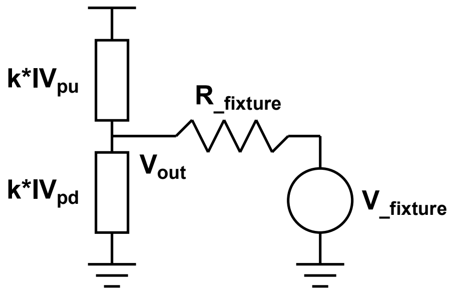
$$0 = k_{pu}(t) \cdot IV_{pu}(V_{wfm2}(t)) - k_{pd}(t) \cdot IV_{pd}(V_{wfm2}(t)) - I_{out}(V_{wfm2}(t))$$

where

$$I_{out} = \frac{V_{out} - V_{fixture}}{R_{fixture}}$$

and wfm1 and wfm2 are waveforms of the same switching direction (rising edges or falling edges) obtained with two different V$_{\_fixture}$ values (usually Vcc and GND)

Assumption

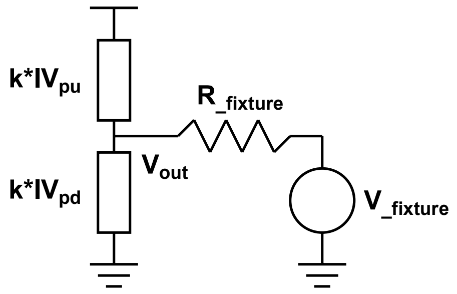


$$0 = k_{pu}(t)\cdot IV_{pu}(V_{wfm1}(t)) - k_{pd}(t)\cdot IV_{pd}(V_{wfm1}(t)) - I_{out}(V_{wfm1}(t))$$

$$0 = k_{pu}(t)\cdot IV_{pu}(V_{wfm2}(t)) - k_{pd}(t)\cdot IV_{pd}(V_{wfm2}(t)) - I_{out}(V_{wfm2}(t))$$

- **$k_{pu}(t)$ and $k_{pd}(t)$ are assumed to be the same for the two different waveforms.**
- **Strictly speaking this is not true, but the explanation for that is beyond the scope of this presentation.**

Solution



$$k_{pd}(t) = \frac{I_{out}(V_{wfm1}(t)) \cdot IV_{pu}(V_{wfm2}(t)) + I_{out}(V_{wfm2}(t)) \cdot IV_{pu}(V_{wfm1}(t))}{IV_{pd}(V_{wfm2}(t)) \cdot IV_{pu}(V_{wfm1}(t)) - IV_{pd}(V_{wfm1}(t)) \cdot IV_{pu}(V_{wfm2}(t))} = \frac{I_{fx1}(t) \cdot I_4(t) + I_{fx2}(t) \cdot I_3(t)}{I_2(t) \cdot I_4(t) - I_1(t) \cdot I_3(t)}$$

$$k_{pu}(t) = \frac{I_{out}(V_{wfm1}(t)) \cdot IV_{pd}(V_{wfm2}(t)) + I_{out}(V_{wfm2}(t)) \cdot IV_{pd}(V_{wfm1}(t))}{IV_{pd}(V_{wfm2}(t)) \cdot IV_{pu}(V_{wfm1}(t)) - IV_{pd}(V_{wfm1}(t)) \cdot IV_{pu}(V_{wfm2}(t))} = \frac{I_{fx1}(t) \cdot I_1(t) + I_{fx2}(t) \cdot I_2(t)}{I_2(t) \cdot I_4(t) - I_1(t) \cdot I_3(t)}$$

VHDL-AMS implementation

```
------------------------------------------------------------------------
for index in Vwfm_pu'range loop
------------------------------------------------------------------------
  -- Calculate intermediate (current) variables
  ----------------------------------------------------------------------
  I1   :=        Lookup("IV", Vwfm_pd(index) - V_pd_ref, Iiv_pd, Viv_pd);
  I2   :=        Lookup("IV", Vwfm_pu(index) - V_pd_ref, Iiv_pd, Viv_pd);
  I3   := -1.0 * Lookup("IV", V_pu_ref - Vwfm_pu(index), Iiv_pu, Viv_pu);
  I4   := -1.0 * Lookup("IV", V_pu_ref - Vwfm_pd(index), Iiv_pu, Viv_pu);
  ----------------------------------------------------------------------
  -- Calculate intermediate (fixture) variables
  ----------------------------------------------------------------------
  Ifx1 := ((Vwfm_pu(index) - Vfx_pu) / Rfx_pu) + C_comp * dVwfm_pu(index);
  Ifx2 := ((Vfx_pd - Vwfm_pd(index)) / Rfx_pd) - C_comp * dVwfm_pd(index);
  ----------------------------------------------------------------------
  -- Set up the numerator of the equation depending on the direction of
  -- the transition, and set up denominator of the equation.
  ----------------------------------------------------------------------
  if (Edge = "K_pu_on") or (Edge = "K_pu_off") then
    num  := (Ifx1 * I1) + (Ifx2 * I2);
  elsif (Edge = "K_pd_on") or (Edge = "K_pd_off") then
    num  := (Ifx1 * I4) + (Ifx2 * I3);
  else
    num  := 0.0;
  end if;

  den  := (I1 * I3) - (I2 * I4);
  ----------------------------------------------------------------------
  Kout(index) := num / den;
------------------------------------------------------------------------
end loop;
------------------------------------------------------------------------
```

C_comp compensation

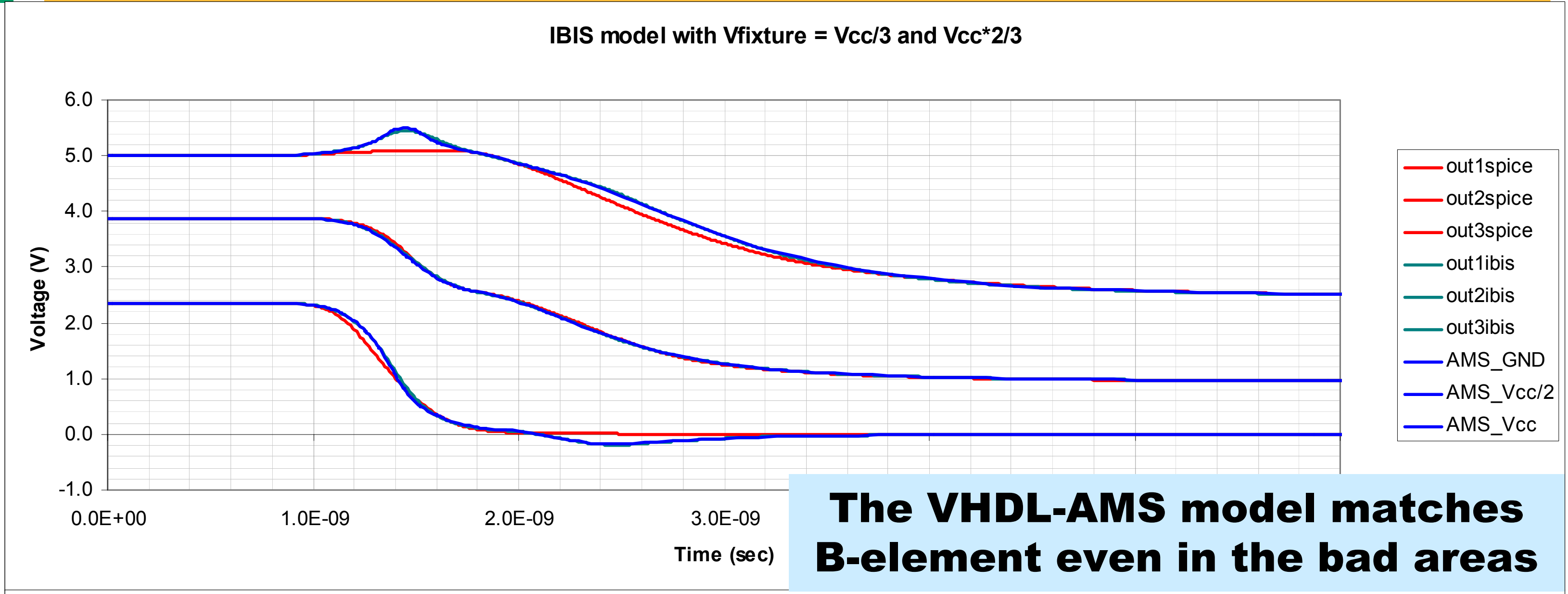
Overview of VHDL-AMS I/O buffer example

- **“Entity” section**
  - “Generics” – various IBIS parameters defined as variables: $C_{\_comp}$, $V_{\_fixture}$, $R_{\_fixture}$, V_{puref}, V_{pdref}, IV tables, Vt tables, etc…
  - One non-IBIS parameter to define mesh size for processed Vt tables and scaling coefficients
- **“Architecture” section**
  - Define “ports”, “signals”, “quantities”, “constants”, and functions
    - PWL lookup function
    - Common time axis generator and interpolator for Vt curves
    - Vt curve to scaling coefficient converter
  - “Process” sections process events on digital signals (input, enable)
  - “Break” statements ensure that the analog equations are calculated properly when events occur
  - Simultaneous “if” statements select the appropriate scaling coefficients for each particular state
  - Analog equations of output current due to IV curves and $C_{\_comp}$ capacitors

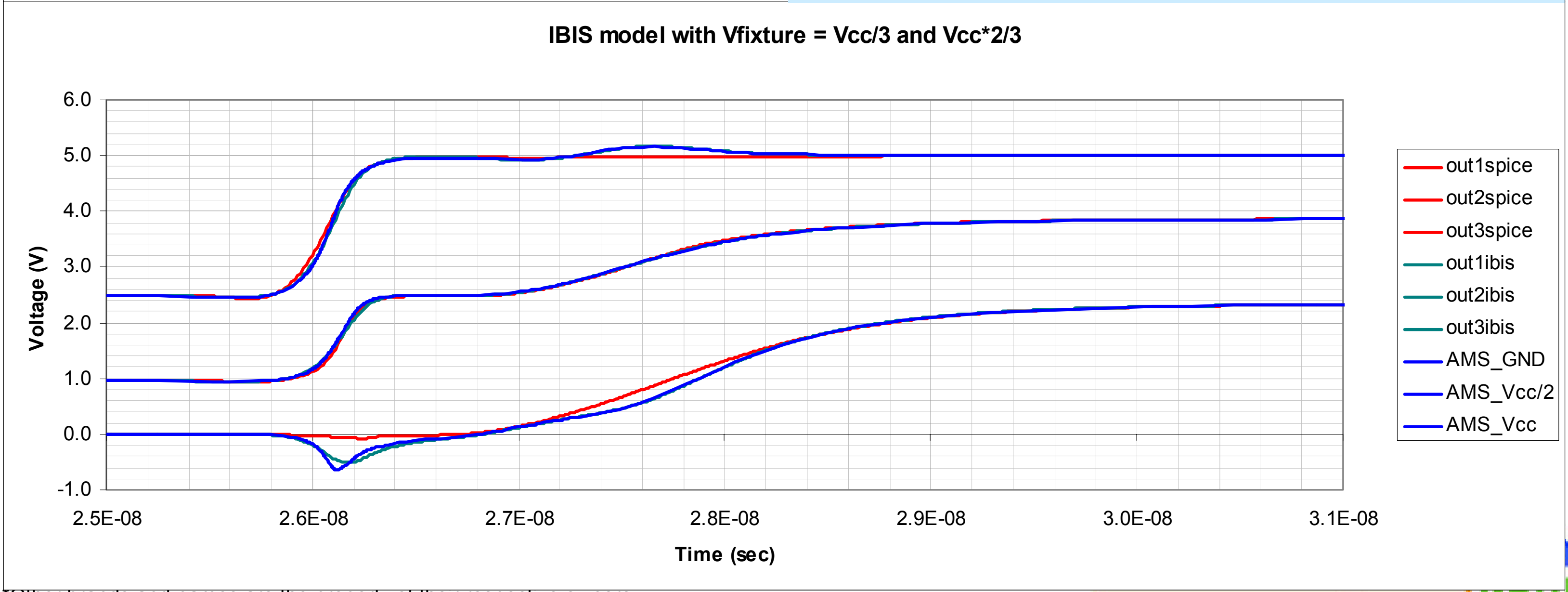
Detailed study of VHDL-AMS file

- If we have enough time, we can open the VHDL-AMS file and go through each statement and function in detail
- If you are reading this presentation on your own, please refer to the files referenced on the last summary page

HSPICE B-element and VHDL-AMS model



The VHDL-AMS model matches B-element even in the bad areas



Structuring the VHDL-AMS file

The VHDL-AMS code presented can be structured to define sub-“entities” and/or “packages”.

- an entity “transistor” (to instantiate PU and PD);
- an entity “clamp” (to instantiate the two clamps);
- a package with the functions.

It results a main entity containing only the concurrent statements of the digital logic and the instantiation of the sub-entities.

Advantages:

- small structures: easier to maintain;
- shorter code (= fewer bugs);
- different kind of buffers (I/O, IN, Open-Collector, …) obtained by selecting the appropriate sub-entities to instantiate.

Entities and packages can be concatenated into a single file for distribution, eventually.

Solving an existing problem with VHDL-AMS

- DDR style termination (to a voltage of Vcc/2) results in inaccurate waveforms when the $V_{\_fixture}$ values used in the IBIS file are at Vcc and GND
- Further studies revealed that simulation waveforms are even worse when the actual simulation uses termination voltages outside the range that the $V_{\_fixture}$ values cover in the IBIS file
- This problem has been presented in a previous IBIS summit
  - http://www.eda.org/pub/ibis/summits/sep01/muranyi1.pdf
  - Please note that the original presentation contained an error which has been corrected in an update on February 13, 2003 which has not been presented in public to date
  - Even though the problem was first observed with the HSPICE B-element, it turns out that is a general problem inherent to the 2-equations, 2-unknown algorithm

Multi VT-tables with VHDL-AMS

In the equations to solve for the K-tables only two rising (and two falling) VT-tables can be used.

But… we can use whichever pair of VT-tables we like better.

Two approaches could be implemented :

- computing all the possible K-tables during the initialization phase;
- computing the necessary K-tables “on the fly”.

In any case, adding a simple decision logic to the code discussed on the previous pages enables us to choose the best VT-table pair.

Let’s see how…

Multi VT-tables with VHDL-AMS (2)

- Using the initial voltage-values of the VT-tables we can define some "switching zones".
- Depending on which zone contains the actual voltage value at the buffer pad (when the transition is about to start), we can decide which VT-table pair describes the current situation most accurately.

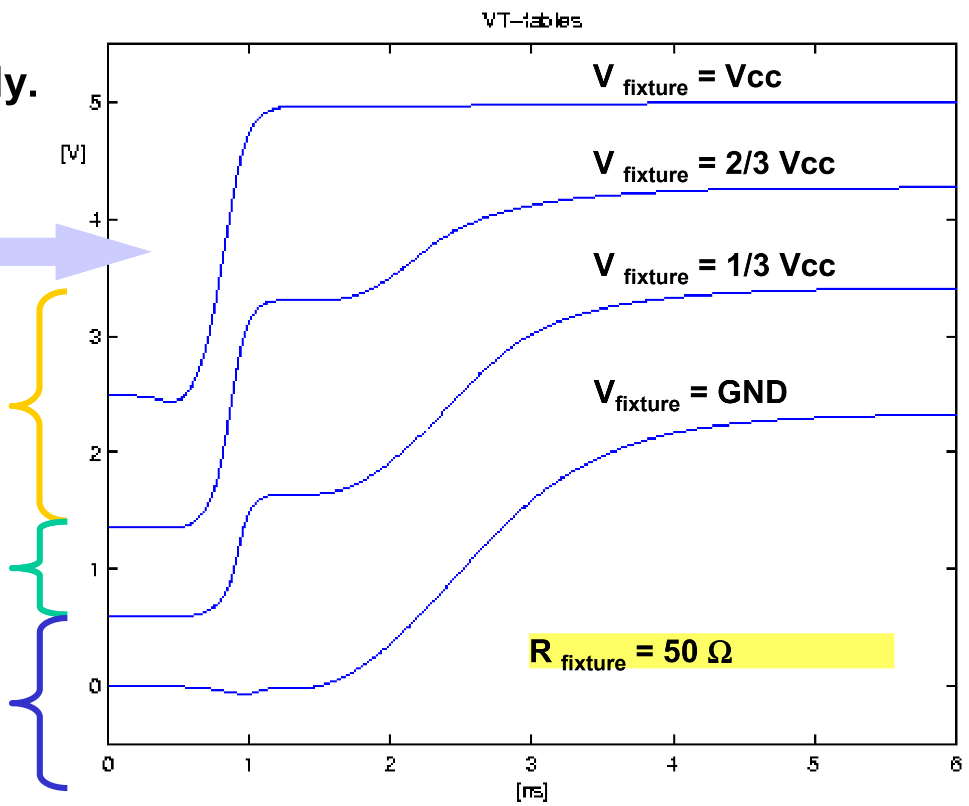
The four rising VT-tables from our example-file

Zone "C"

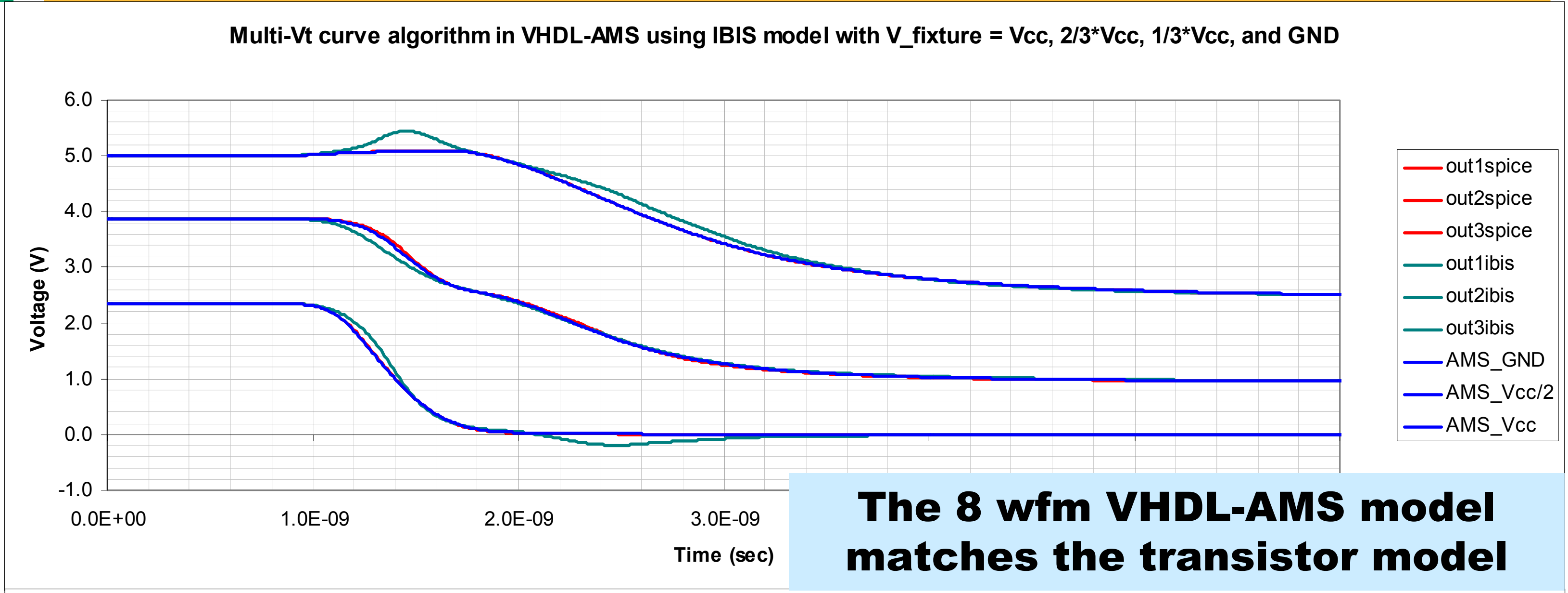
Zone "B"

Zone "A"

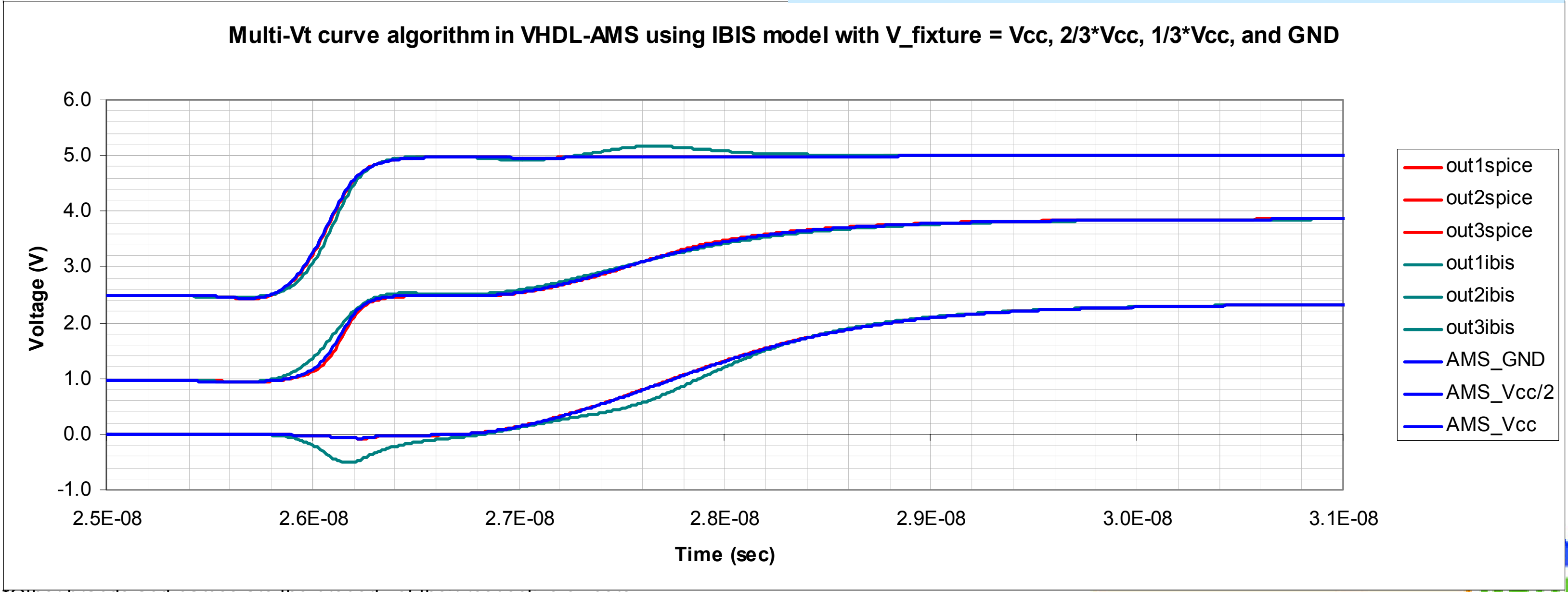
For the falling transition, a similar strategy has been implemented.



Transistor model, 4 wfm IBIS, and 8wfm IBIS



The 8 wfm VHDL-AMS model matches the transistor model



Multi VT-tables with VHDL-AMS (3)

The assumptions underlying the described algorithm are:

- All the VT-tables have the same $R_{\_fixture}$ (50 Ω) ;
- The actual loading impedance is about 50 Ω ;
- The transition happens when the previous transition is (almost) over.

The “standard” algorithm uses similar assumptions. Future work may lead to improvements in order to relax some of these limitations.

Summary

- **A basic VHDL-AMS implementation of a behavioral I/O buffer model using IBIS data has been shown**
  - http://www.eda.org/pub/ibis/summits/jun03a/IBIS_basic_IO.vhd
  - Feel free to download and use the file any way you want
- **An improved version of the file has been introduced to solve an existing problem that is inherent in the most commonly used IBIS algorithms**
  - http://www.eda.org/pub/ibis/summits/jun03a/IBIS_multiVt_IO.vhd
  - Feel free to download and use the file any way you want
  - Support for multi Vt curve IBIS models in EDA tools is a must to eliminate this problem
  - IBIS model makers should consider generating IBIS models with multiple sets of Vt curves using several $V_{\_fixture}$ values in addition to the usual Vcc and GND