

Real Valued Modeling for Mixed Signal Simulation

Product Version IES 8.2
January 2009

Authors: Walter Hartong and Scott Cranston

Copyright Statement
© 2008 Cadence Design Systems, Inc. All rights reserved worldwide.

Contents

Real Valued Modeling for Mixed Signal Simulation	3
Purpose	3
Terms	3
Audience.....	4
Introduction.....	4
What is Real Valued Modeling?	5
Motivation behind Real Valued Modeling.....	5
Simulation Performance, Accuracy and Modeling Effort	7
Model creation and verification.....	10
Real value modeling approaches	10
VHDL real	10
SV real	11
Verilog-AMS wreal net	11
Wreal extensions.....	13
Wreal support in IES.....	19
Summary	19

Real Valued Modeling for Mixed Signal Simulation

Purpose

Real Valued Modeling (RVM) is a way by which users can perform verification of their analog or mixed-signal designs using discretely simulated real values. This allows simulation using only the digital solver, avoiding the slower analog simulation, thus, enabling intensive verification of mixed signal design in short period of time. The trade-off between simulation performance and accuracy has to be considered in the context. RVM also opens the possibility of linkage with other advanced verification technologies such as assertion-based verification without the difficulty of interfacing to the analog engine or defining new semantics to deal with analog values.

It is anticipated that users enable the RVM flow by migrating their analog models or transistor level design to real value modeling style. The purpose of this application note is to give an introduction into RVM and to show the performance gains and accuracy tradeoffs in doing so. Various examples included in this application note will illustrate the flow. Finally, the latest enhancements in IES8.2 to the RVM flow are also highlighted in this application note.

Terms

Real Valued Modeling (RVM) - simulation with the digital simulator using primarily real (floating-point) values instead of analog/electrical simulation.

Wreal Coercion - the process by which a Verilog wire can become a wreal net because of its hierarchical connections.

System On Chip (SOC) – A large system implemented on a single chip including various digital and analog functional blocks.

Connect Modules (CM) – Connection blocks between different signal domains, e.g. electrical and logic. The CMs are mostly inserted automatically during the elaboration process (Automatic inserted connect modules - AICM)

Audience

The target audiences for this application note are analog, digital as well as mixed signal engineers. Customers looking for high performance mixed signal verification – mainly for BigD/smallA full chip verification. Also, targeted for customers:

- Looking to perform high volume, digital-centric nightly regressions tests to verify their mixed signal SoCs
- Customers verifying Top-level SoCs that have a small to moderate amount of analog in the design

Introduction

Most of the system verification in analog, digital and mixed signal domain is based on simulation runs. To meet the verification goals, certain amount of simulation data and data accuracy are required, e.g. a detailed analysis of an RF low noise amplifier requires very high simulation accuracy but a single RF sinusoid period might be sufficient. On the other hand, a pin connectivity check for a large digital block has an extremely low sensitivity towards accuracy but may require a long transient simulation time to cover all sorts of events and states.

Consequently, a long full-chip simulation run using highest level of simulation accuracy would be desirable. The limiting factor in this context is simulation performance. The only practical way around this problem is a hierarchical verification approach that uses different level of design abstractions for different verification goals.

Real values modeling is an interesting add-on to classical mixed signal verification approaches, like a Verilog and Spice mixed signal simulation or a pure digital modeling of the analog block in the mixed signal design.

This document presents the motivation behind Real Valued Modeling and a brief comparison to other modeling and simulation approaches. It introduces the Verilog-AMS wreal net concept and the related VHDL and SystemVerilog based approaches. Cadence supports some proprietary extensions to wreal in the context of System Verilog and other Real Valued Modeling applications. Detailed examples are used to illustrate how to use Real Valued Modeling to model a typical analog block in a digital simulation context.

What is Real Valued Modeling?

The simulation approaches in analog and digital are fundamentally different due to the structure of the underlying equation system to solve. While the digital solver is solving logical expressions in a sequential manner based on triggering events, the analog simulator must solve the entire analog system matrix at every simulation step. Each element in the analog design can have an instantaneous influence on any other element in the matrix and vice versa. Thus, there is not an obvious signal flow in one or the other direction. Time and values are continuous. In digital, time and values are discrete. The solver can apply a well-defined scheme of signal flow and events to solve the system.

RVM is a mixed approach borrowing concepts from both domains. The values are continuous – floating-point (real) numbers – as in the analog world. However, time is discrete, meaning the real signals change values based on discrete events. In this approach, we apply the signal flow concept, so that the digital engine is able to solve the RVM system without support of the analog solver. This guarantees a high simulation performance that is in the range of a normal digital simulation and orders of magnitudes higher than the analog simulation speed.

There are three different HDL language standards that support RVM, namely:

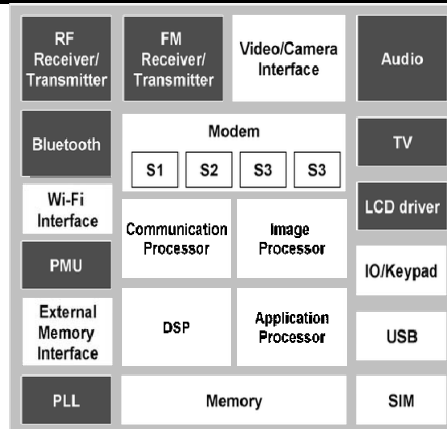
- **wreal** ports in Verilog-AMS
- **real** in VHDL
- **real** in SystemVerilog

It is important to note that the real-wire (**wreal**) is defined only in the Verilog-AMS LRM. Thus, a **wreal** can only be used in a Verilog-AMS block. However, it is the digital kernel only that solves the **wreal** system. There are no major performance drawbacks when using these types of Verilog-AMS modules in a digital simulation context.

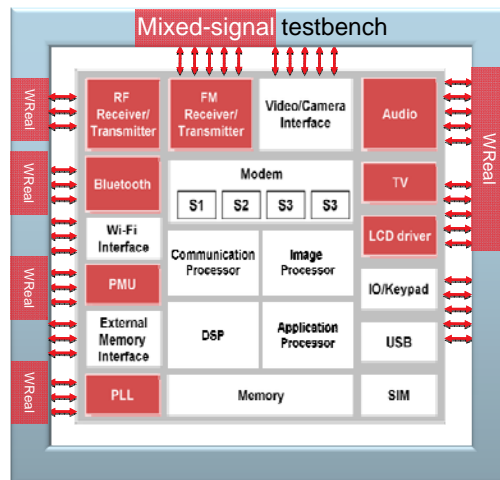
Motivation behind Real Valued Modeling

The typical SoC Verification flow involves top-level simulation of components at various levels of abstraction. For example, a verification engineer may need to integrate components from schematics, SystemVerilog, and Verilog (or VHDL)-AMS in a single top-level SoC verification. The following picture illustrates this scenario (the dark gray blocks are analog or mixed-signal blocks):

Real Valued Modeling for Mixed Signal Simulation



Functional complexity in terms of modes of operation, extensive digital calibration, and architectural algorithms is now overwhelming the traditional verification methodologies. Simulation at this top-level is extremely costly (both in terms of time and licenses cost) since a significant amount of the SoC is simulated inside the analog engine. Finding a way to reduce the time and expense to verify this SoC, while trading off some accuracy that is not needed at this high level of integration, is extremely valuable. This is the target application of Real Valued Modeling. By replacing the analog portions of the SoC with functionally equivalent digital models, which do not require the analog engine, we achieve a significant speed-up in simulation performance and reduction in the license cost. Meanwhile, typical analog simulation problems such as convergence issues are totally eliminated. Following is a modified version of the above picture with the analog portions of the design replaced with functionally equivalent real valued models (the red blocks):



It is obvious that this top level verification strategy is not a replacement for detailed block or multiple-block, cluster-level verification with full analog simulation accuracy. Even for the top-level verification, there might be rare cases where the RVM approach

Real Valued Modeling for Mixed Signal Simulation

does not provide enough accuracy for a particular verification goal, e.g. cross talk of a global net into an analog sub block. Moreover, the model verification task comparing the model against the transistor level reference for each RVM model used in the top-level verification is essential to qualify the overall verification result.

RVM also opens the door to use other logic verification methodologies, such as

- Metric-driven verification (randomization, coverage, assertion-based)
- Higher-level verification languages such as SystemVerilog and *e*
- CPF (low-power)

These verification techniques are not yet fully supported for electrical systems and they require a high simulation performance to check sufficient number of data cycles – both issues can be addressed by RVM.

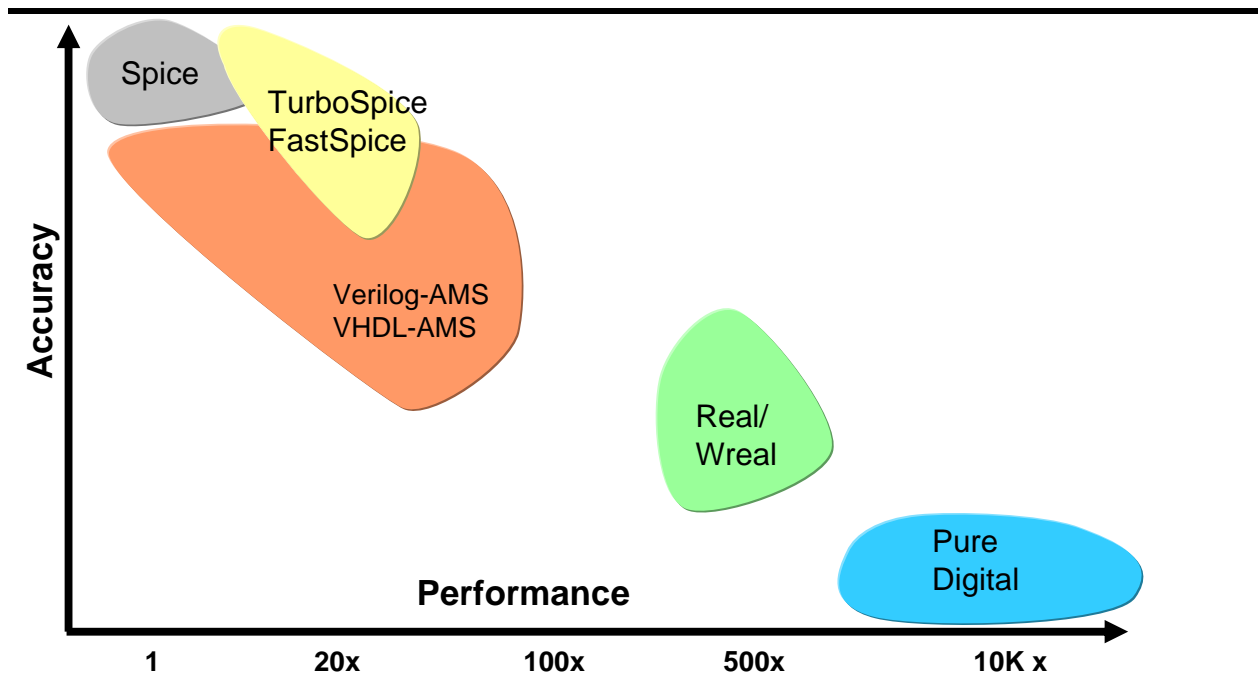
Simulation Performance, Accuracy and Modeling Effort

For the analog sub system, common levels of abstraction are:

- Extracted transistor netlist including parasitic elements
- Transistor level representation, Spice level simulation
- Fast spice simulation
- Analog behavioral modeling
- Real number modeling
- Pure digital model

The gain in simulation performance and the reduction in accuracy are highly dependent on the application. There is no general recommendation on what level of abstraction might be useful or not. All have particular advantages and disadvantages and thus a useful application area. The following pictures show a general trend in the accuracy/performance tradeoff. The numbers are generic and can vary significantly for different applications.

Real Valued Modeling for Mixed Signal Simulation

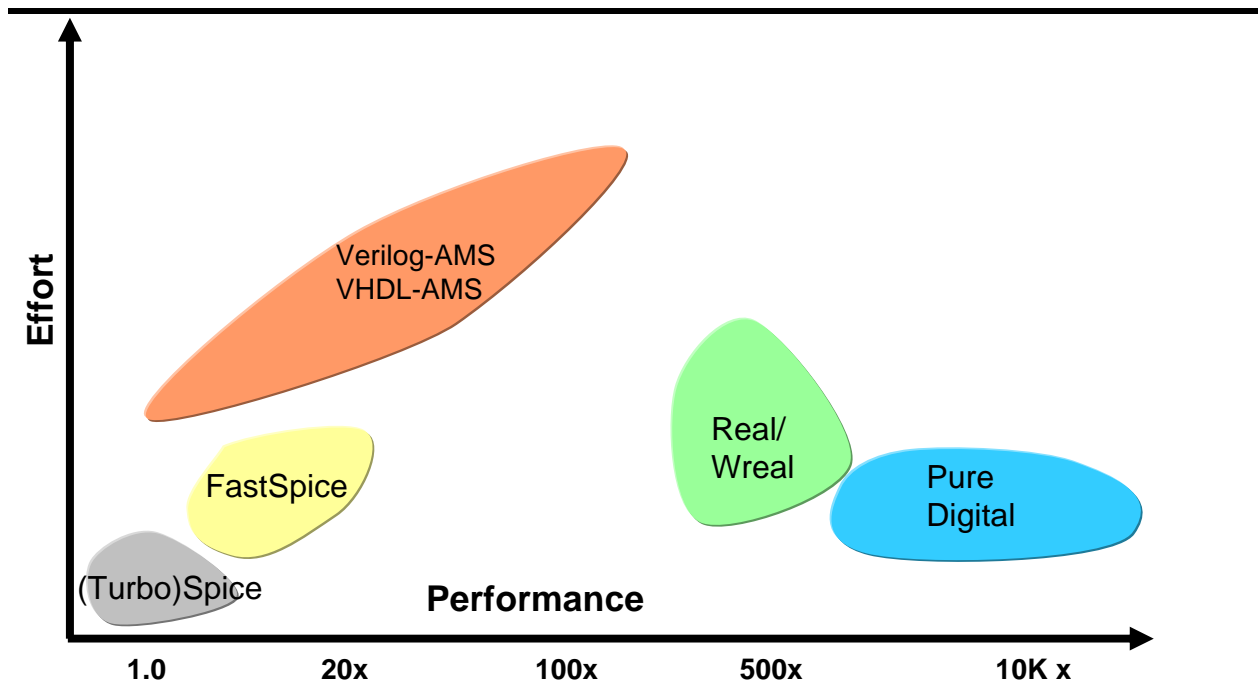


Spice level simulations are used as golden reference simulation. Fast and Turbo Spice engines reach the same accuracy but can also trade-off some accuracy versus speed (mainly fast spice). Analog behavioral modeling provides a large range of capabilities, reaching from high accuracy to high performance. However, it should be noted that a low performance, low accuracy model is a potential risk for inexperienced modelers. Especially convergence issues caused by over-idealized models might slow down the simulation significantly.

As mentioned before, RVM provide high simulation performance but restrict the model accuracy at the same time. Finally, pure digital model are very inaccurate but may be sufficient for some verification tasks, like connectivity checks.

The other effect to consider in this context is the effort required to setup a simulation or create the model. The following chart illustrates the general trends.

Real Valued Modeling for Mixed Signal Simulation



Spice simulations are reference simulations that are performed anyway, thus, there is no overhead. Turbo spice setup requires only one single option (-turbo) – again no overhead. Since FastSpice simulations require a detailed control on speed vs. accuracy, on a block-level basis, some amount of setup effort and understanding of the design is required.

Analog behavioral model creation effort can range from hours to days and possibly weeks for a good behavioral model. RVM are inherently restricted to the signal flow approach and analog convergence is not an issue. Consequently, the modeling effort is significantly lower compared to analog behavioral models and the same applies for pure digital models. Secondly, more importantly, the design engineer does not need to learn a new language to create the real value models. Therefore, the ramp-up time to create these models is much lower than creating analog behavioral models.

Behavioral models can also be differentiated by the modeling goal. A performance-oriented model needs to precisely capture critical behavior necessary to efficiently explore the design space and make implementation trade-offs. Functional models capture the actual circuit behavior only to the detail needed to verify the correct design functionally. Both types of models are found in top-down and bottom modeling. However, functional verification is far more common in bottom-up modeling where they are used to perform the final design verification prior to tape-out. For example, a functional verification model can be used to study dynamic closed-loop functionality between the RF and digital baseband ICs. Whereas a performance-oriented model of an RF block could be used for cascaded measurements such as IP3 and explore system-level metrics such as BER and EVM.

Real Valued Modeling for Mixed Signal Simulation

Model creation and verification

A detailed understanding of the reference design – in most cases the transistor level circuit – is required for the model creation process. This includes transistor level simulations that are mostly driven from the simulation environment (ADE). On the other hand, the model is created for a specific verification purpose with its performance and accuracy requirements. While analog designers mainly own the first skill set, the mixed signal verification engineers understand the verification requirements better. Therefore, a close cooperation between both parties is mandatory.

Today, either the analog designers, digital designers, verification engineers and/or dedicated mixed signal modelers are creating real-valued models. It mostly depends on the skill-set of the people in the organization and the structure of design and verification teams, as to which approach works best for any specific company and/or the project team.

Real value modeling approaches

As mentioned above, the real value modeling capabilities are supported in different standard HDL languages, Verilog-AMS, VHDL and SystemVerilog. This application note is focused mainly on the Verilog-AMS wreal capability. The general idea and motivation of real number modeling is equivalent in the other languages while the feature set and the specific implementation may vary.

VHDL real

Real valued signals are supported in VHDL as internal signals as well as port types. The following example demonstrates the use of the real type in VHDL.

```
LIBRARY IEEE;
USE ieee.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;

ENTITY dac14 IS
  PORT (SIGNAL  clk:IN std_logic;
        SIGNAL  d  : IN std_logic_vector (0 TO 13);
        SIGNAL  lsb, supply : IN real;
        SIGNAL  aout: OUT real := 0.0 );
END ENTITY dac14;

ARCHITECTURE bhv OF dac14  IS
BEGIN
  conversion : PROCESS (clk)
    VARIABLE xval : real:=0.0;
    VARIABLE din_val : integer := 0;
```

Real Valued Modeling for Mixed Signal Simulation

```
BEGIN
IF (clk'EVENT AND clk='1') THEN
    din_val := TO_INTEGER(UNSIGNED(d));
    xval := real(din_val) * lsb;
    IF xval > supply THEN
        xval := supply;
    END IF;
    aout <= xval;
END IF;
END PROCESS;
END ARCHITECTURE bhv;
```

In contrast to the real and wreal concept in verilog VHDL uses real signals internally and as port types. It is possible to define resolution function for real signals with multiple drivers.

SV real

In SystemVerilog, there is no concept of “real-valued” nets. However, to provide a similar capability as real signals, the creators allowed real variables (in fact, all variables) to be driven just as nets are. For example, one can drive a real variable with a continuous assign, as in the following:

```
var real r, xr;

assign r = xr;
```

Now the variable r will be updated whenever the value of xr changes. Since non-collapsed ports are modeled as continuous assigns, a salutary effect of this is the ability to connect variables as a “reader” to port, either:

- Connected to an output port, or
- Declared as an input port

The chief drawback to this capability is the restriction to a single driver. An error will result if the user attempts to use more than one continuous driver to a single variable, or to procedurally assigning to a continuously driven variable.

Verilog-AMS wreal net

In traditional Verilog, real values were modeled using 64-bit vectors, which encoded the real value in IEEE floating point format. Two system tasks, \$realtobits and \$bitstoreal, were provided to encode and decode the real values in the 64 bit vectors. However, this

Real Valued Modeling for Mixed Signal Simulation

use model does not support reconfigurable models under a schematic based environment. In this environment, the user models a real value with a single, scalar entity, which does not map into the traditional Verilog representation of a 64-bit vector real. This also proved difficult in the mixed-language world with VHDL reals not mapping cleanly to the 64-bit vectors used in traditional Verilog.

In Verilog-AMS, the concept of a truly real-valued net/wire was introduced, called "wreal" – a real valued wire. These nets represent a real-valued physical connection between structural entities.

The following example illustrates the use of a real wire type as in input port of a VCO. The real value "vin" is used in an always block – this is possible due to the event based nature of a wreal net – calculating the output frequency of the VCO.

```
module vco(vin, clk);
  input vin;
  output clk;
  wreal vin;
  reg clk;
  real clk_delay, freq;

  always @(vin)
    begin
      freq = center_freq + vco_gain*vin;
      clk_delay = 1.0/(2*freq);
    end

  always #clk_delay
    clk = ~clk;
endmodule
```

The Verilog-AMS LRM lists the following restrictions on wreal nets:

- they can have at most one driver
- they can only connect to other wreals, wires, or real-valued expressions
- scalar only, no support for arrays

The example above shows that the wreal functionality is useful for practical designs. However, the limitations introduced by the LRM definitions have a substantial impact on the usability. As a result, Cadence extended the wreal support beyond the LRM limitations enabling a huge variety of applications with the extended wreal features. The following section will describe these enhancements in detail.

Real Valued Modeling for Mixed Signal Simulation

Wreal extensions

In order to provide some benefit to the wreal construct, Cadence had made some significant extensions, beyond the Verilog-AMS LRM restrictions. These extensions are:

- Electrical to wreal and wreal to electrical connect modules
- Support for wreal arrays
- Support for wrealXState and wrealZState
- Support for multiple wreal drivers and resolution functions
- Support for wreal table models
- Ability to connect a wreal to a VHDL real signal or SystemVerilog real variable
- Automatic “type-casting” to wreal when a wire is hierarchically connected to a wreal, SystemVerilog real variable, or VHDL real signal

The following sections will provide a quick overview into those enhancements and the application area.

R2E/E2R connect modules

Similar to the mechanism of automatically inserted connect modules (AICM) between the continuous and discrete domains, these connect modules are available between wreal and electrical. These are called E2R and R2E. They are also inserted automatically, if needed, during the elaboration process.

The current connect modules (CM) look like follows. The Real to Electrical (R2E) CM implementation is relatively straightforward. The newly introduced X and Z states (see below) are taken into account. The input wreal value is assigned to a voltage source with a serial resistance. Transition operators help to avoid abrupt changes in the behavior that might convergence issues in the analog solver.

```
// R2E_2.vams - Efficient Verilog-AMS discrete wreal to
electrical connection module
//
// REVISION HISTORY:
// Created: 09/01/08, gangchen

`include "disciplines.vams"
`timescale 1ns / 100ps

connectmodule R2E_2 (Din, Aout);
  input Din;
  wreal Din;          // input wreal
  \logic Din;
  output Aout;
  electrical Aout;   // output electrical
```

Real Valued Modeling for Mixed Signal Simulation

```
parameter real vsup = 1.8          from (0:inf); // supply
voltage
parameter real vdelta = vsup/64   from (0:vsup]; // voltage
delta
parameter real vx = 0              from [0:vsup]; // X output
voltage
parameter real vz = vx            from [0:vsup]; // Z output voltage
parameter real tr = 10p           from (0:inf); // risetime of
analog output
parameter real tf = tr            from (0:inf); // falltime of
analog output
parameter real ttol_t = (tr+tf)/20 from (0:inf); // time tol of
transition
parameter real tdelay = 0         from [0:inf); // delay time
of analog output
parameter real rout = 200         from (0:inf); // output
resistance
parameter real rx = rout          from (0:inf); // X output
resistance
parameter real rz = 10M          from (0:inf); // Z output
resistance

real Vstate, Rstate;
real Vout, Rout;

initial begin
  if (Din === `wrealXState)
    begin Vstate = vx; Rstate = rx; end
  else if (Din === `wrealZState)
    begin Vstate = vz; Rstate = rz; end
  else
    begin Vstate = Din; Rstate = rout; end
end

always @(Din) begin
  if (Din === `wrealXState)
    begin Vstate = vx; Rstate = rx; end
  else if (Din === `wrealZState)
    begin Rstate = rz; end
  else if (Din-Vstate >= vdelta || Vstate-Din >= vdelta)
    begin Vstate = Din; Rstate = rout; end
end

assign Din = Din;

analog begin
  Vout = transition(Vstate, tdelay, tr, tf, ttol_t);
  Rout = transition(Rstate, tdelay, tr, tf, ttol_t);
  I(Aout) <+ (V(Aout) - Vout) / Rout;
end
```

Real Valued Modeling for Mixed Signal Simulation

endmodule

In contrast to the R2E, the E2R (Electrical to Real) operation is not so obvious since the continuous analog value needs to be translated into an event based wreal signal. For this purpose, we created a new system function, *absdelta*, which creates events based on input value changes. In the CM below the *absdelta* function issues an event and thus updates the wreal value whenever V(Ain) changes more than the value *vdelta*.

More precisely the *absdelta* function generates events for the following times and conditions (*absdelta* (*expr*, *delta* [, *time_tol* [, *expr_tol*]]):

- At time zero.
- When the analog solver finds a stable solution during initialization
- When the *expr* value changes more than *delta* plus or minus *expr_tol*, relative to the previous *absdelta* event (but not when the current time is within *time_tol* of the previous *absdelta* event).
- When *expr* changes direction (but not when the amount of the change is less than *expr_tol*.

```
// E2R.vams - basic Verilog-AMS electrical to discrete wreal
connection module
// last revised: 08/01/06, jhou
//
// REVISION HISTORY:
// Created: 08/01/06, jhou

`include "disciplines.vams"
`timescale 1ns / 100ps

connectmodule E2R (Ain, Dout);
input Ain;
electrical Ain; //input electrical
output Dout;
wreal Dout; //output wreal
`logic Dout; //discrete domain

parameter real vdelta=1.8/64 from (0:inf); // voltage
delta
parameter real vtol=vdelta/4 from (0:vdelta); // voltage
tolerance
parameter real ttol=10p from (0:1m]; // time
tolerance

real Dreg; //real register for A to D wreal conversion
```

Real Valued Modeling for Mixed Signal Simulation

```
assign Dout = Dreg;

//discretize V(Ain) triggered by absdelta function
always @(absdelta(V(Ain), vdelta, ttol, vtol))
    Dreg = V(Ain);

endmodule
```

The proper setting of the CM parameters are critical for correct simulation results. While the supply voltage level mainly influences the electrical to logic connection threshold, the E2R settings can be independent. E.g. if one models a 5 mV bias voltage into a block in wreal and the net is connected an electrical part as well, a good setting for a vdelta parameter might be 100uV –independent of what the supply voltage might be.

Consider the default resistance in the R2E CM as well. If a wreal net is connected to an electrical part that consumes significant current (e.g. power net) the 200 Ohm default resistance of the R2E CM might have an unwanted influence on the signal level.

Wreal Arrays

Similar to busses, a wreal array groups multiple real values into a single, indexible entity. The following example demonstrates the definition of a wreal array:

```
module ams_tb;
    wreal w[3:2];
    sub_design d1(w);

    initial begin
        #10 $display("%f,%f",w[2],w[3]);
    end
endmodule

module sub_design(r);
    output wreal r [1:0];
    assign r[1] = 2.7182818;
    assign r[0] = 3.142818;
endmodule
```

Wreal X and Z State

The concept of an unknown – X and high impedance – Z state that is used in the 4-state logic concept is useful for wreal signals as well. The meaning of X and Z is equivalent for the wreal case. Two new keywords ``wrealZState` and ``wrealXState` are used to define the related conditions in the wreal context as shown in the following example:

Real Valued Modeling for Mixed Signal Simulation

```
module foo(x);
  inout x;
  wreal x;
  always @(x) begin
    if(x == `wrealZState)
      x = 1.234;
    if(x == `wrealXState)
      x = `wrealZState
  end
endmodule;
```

Multiply-Driven Wreals

While the resulting value of multiply driven 4-state logic nets are relatively obvious, it is not obvious what the result value on a wreal net should be if one driver provides 2.7 and another -4.87. Should the output be the sum, the average, an X? Since there is no - single answer to that question an user-selectable resolution function is provided.

Currently (IES 82 FCS) six resolution functions are supported.

- **DEFAULT** – Single active driver only, support for Z state
- **4STATE** – Similar to verilog 4-State resolution for digital nets
- **SUM** – Resolves to a summation of all the driver values
- **AVG** – Resolves to the average of all the driver values
- **MIN** – Resolves to the least value of all the driver values
- **MAX** – Resolves to the greatest value of all the driver values

The resolution function are selectable with “*-wreal_resolution <res_func>*” argument to ncelab or irun.

The following table shows the results of a wreal being driven by two drivers using different resolution functions:

D1	D2	Default	4state	sum	avg	min	max
x	x	x	x	x	x	x	x
x	z	x	x	x	x	x	x
x	1.1	x	x	x	x	x	x
z	z	z	z	z	z	z	z
z	1.1	1.1	1.1	1.1	1.1	1.1	1.1
2.2	1.1	x	x	3.3	1.65	1.1	2.2
1.1	1.1	x	1.1	2.2	1.1	1.1	1.1

Real Valued Modeling for Mixed Signal Simulation

Note that the resolution function is currently (IES82FCS) a global setting that applies to all wreal nets in the design. Moving forward, a net-based setting for different resolution functions may be considered.

Wreal Table Model

Another useful enhancement is the \$table_model function known from analog Verilog-A and Verilog-AMS blocks. The function is now available for real values as well. Please find details of the \$table_model function and the different options to the function in the related documentation (Cadence Verilog-AMS Language Reference).

```
module vco(VCO_OUT, VDD, VCTRL);
  wreal VCTRL;
  wreal VDD;
  logic VCO_OUT;

  always @(VDD, VCTRL) begin
    tableFreq = $table_model(VDD, VCTRL,
      "./vcoFreq.tbl");
    halfper = 1/(2*tableFreq);
  end

  always begin
    #Thalfper osc0P = !osc0P;
  end
  assign VCO_OUT = osc0P;
endmodule
```

The example above calculates the VCO output frequency according to the table in the text file "vcoFreq.tbl". This file may look like:

#VDD	VCNTL	Freq
1	0	9.81E+06
1	0.4	1.05E+09
1	0.8	1.41E+09
1.1	0	9.88E+06
1.1	0.45	1.29E+09
1.1	0.9	1.52E+09
1.2	0	9.95E+06
1.2	0.5	1.37E+09
1.2	1	1.69E+09

Assuming that the VDD=1.1 and VCNTL=0.45 the output frequency would be 1.29 GHz. For values between the definition points (e.g. VCNTL=0.687) a linear or third order spline interpolation is used. The exact behavior, including the behavior outside the

Real Valued Modeling for Mixed Signal Simulation

defined range (e.g. VCNTL=1.5), can be controlled by additional options. For more information, see details in the \$table_model documentation.

Enhanced Wreal Coercion

During the elaboration phase, each segment of a wire or a net gets associated with the appropriate type – electrical, logic type, wreal etc. The LRM only allows wreal ports and nets to connect to wires. These wires will get resolved to the type wreal in this case. This process is also called coercion to wreal.

We have enhanced this functionality in several ways. Wires can be coerced to wreal by connection to SystemVerilog real variables and VHDL real signals.

Wreal support in IES

The wreal simulation capabilities are available since many years in the AMS Designer, however, most of the advanced capabilities that make the usage most attractive have been added in the IES82 release. The table below shows the main features that have been added over the past years.

	Basic wreal support	Wreal/Electrical CMs	Wreal arrays	Wreal X/Z	Multiple drivers & resolution	Table model	Connection to SV real and VHDL real	Coercion with SV/VHDL real
<IES61	x							
IES 61	x	x						
IES 81	x	x						
IES 82	x	x	x	x	x	x	x	x

Summary

This Application Note has introduced some of the Real Valued Modeling technologies provided in the IES 8.2 release, and an example of how they can be applied to a real world modeling problem.

As highlighted in this application note, full-chip SoC simulations now require much more accuracy than digital Verilog/VHDL alone can offer. Real number models are being used more often to discretely model the analog portions of the design. The real number

Real Valued Modeling for Mixed Signal Simulation

models are easily portable between design and verification environments. Real models work in Virtuoso and Incisive environments and can be used as the hand-off between analog and digital designers. So, overall this methodology can provide customers significant simulation speed improvements while enabling them to perform top level verification of their mixed signal SoCs. Replacing the analog and mixed signal blocks in the SoC, under verification, with real and wreal models, the verification engineer can also run nightly regression runs since the verification would only use digital simulators thus avoiding the analog convergence issues.