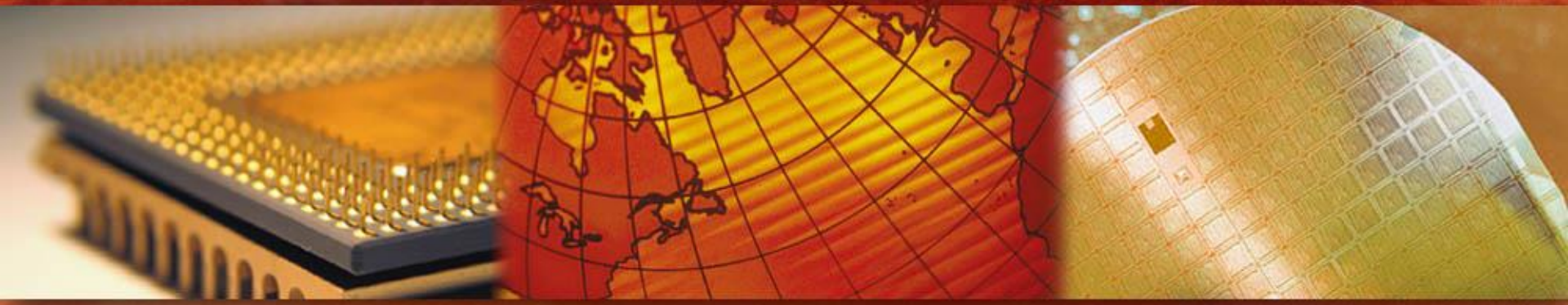
XtremeEDA

A Systematic Approach to Creating Behavioral Models

CDNLive, March, 2015

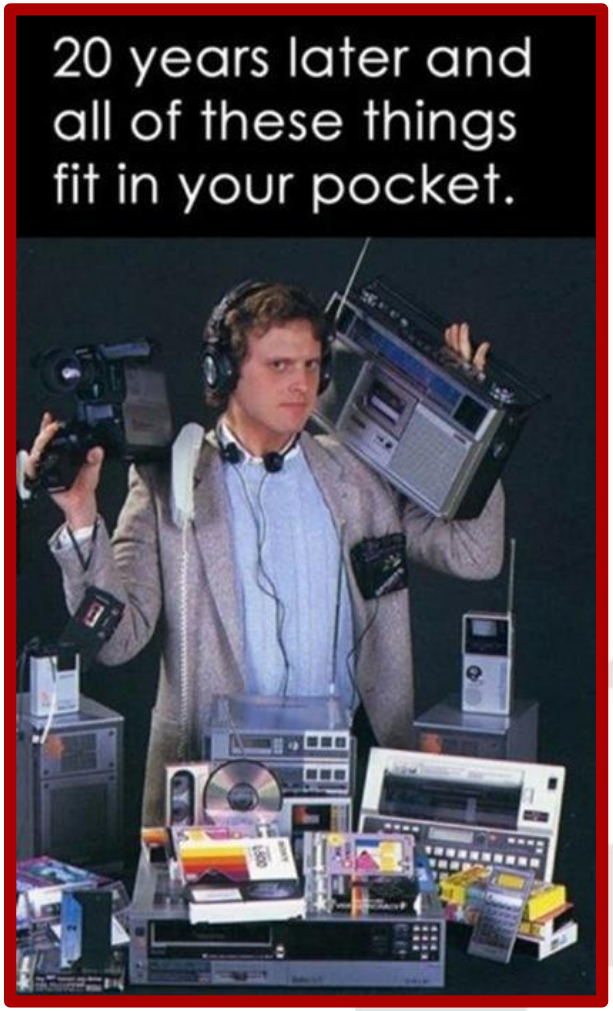
Bob Peruzzi, Joe Medero



Agenda

- Introduction
  - Mixed-Signal Systems on Chips
  - Link to White Paper
  - Model accuracy and trade-offs
  - Good and bad models
- Creating Models
  - Keep Cell Views Sync'd
  - Top-Down with SMG
  - Learn the Circuit
  - Plan the Model
  - Write the Model Code
- Validating Models
  - Accurate versus plan?
  - Detects errors?
  - Test benches
  - Collaborative Effort
- Maintaining Models Through Project Lifetime
  - AMS Design and Model Validator
- Models and Applications
  - Analog Models
  - Digital Models
  - Real Number Models
- Verification
  - Digital Verification with RNM
  - Analog Verification with RNM
  - Common Verification Platform
  - Common Verification Overview
- Summary

Introduction



– https://image-store.slidesharecdn.com/32f6395a-b7bf-40c7-ba98-88107898208f-original.jpeg

Introduction – Mixed-Signal Systems on Chips

- It's a mixed-signal SOC world out there. Why all this integration into bigger and more complicated mixed-signal SOCs? Because it makes financial sense.
- Evolution or extinction
- Survive by embracing the inevitable
  - Independent digital, independent analog, and closely-coupled analog and digital subsystems coexist on the same die
  - Verification of the full chip must be done from the top level
  - SPICE type electrical simulators cannot simulate the full chip
  - **Behavioral models must be used**
  - Mixed signal verification must follow the lead of UVM or similar digital approaches
  - Digital verification methodologies must adapt to include analog
- Our talk is about creating behavioral models

Introduction – Link to White Paper

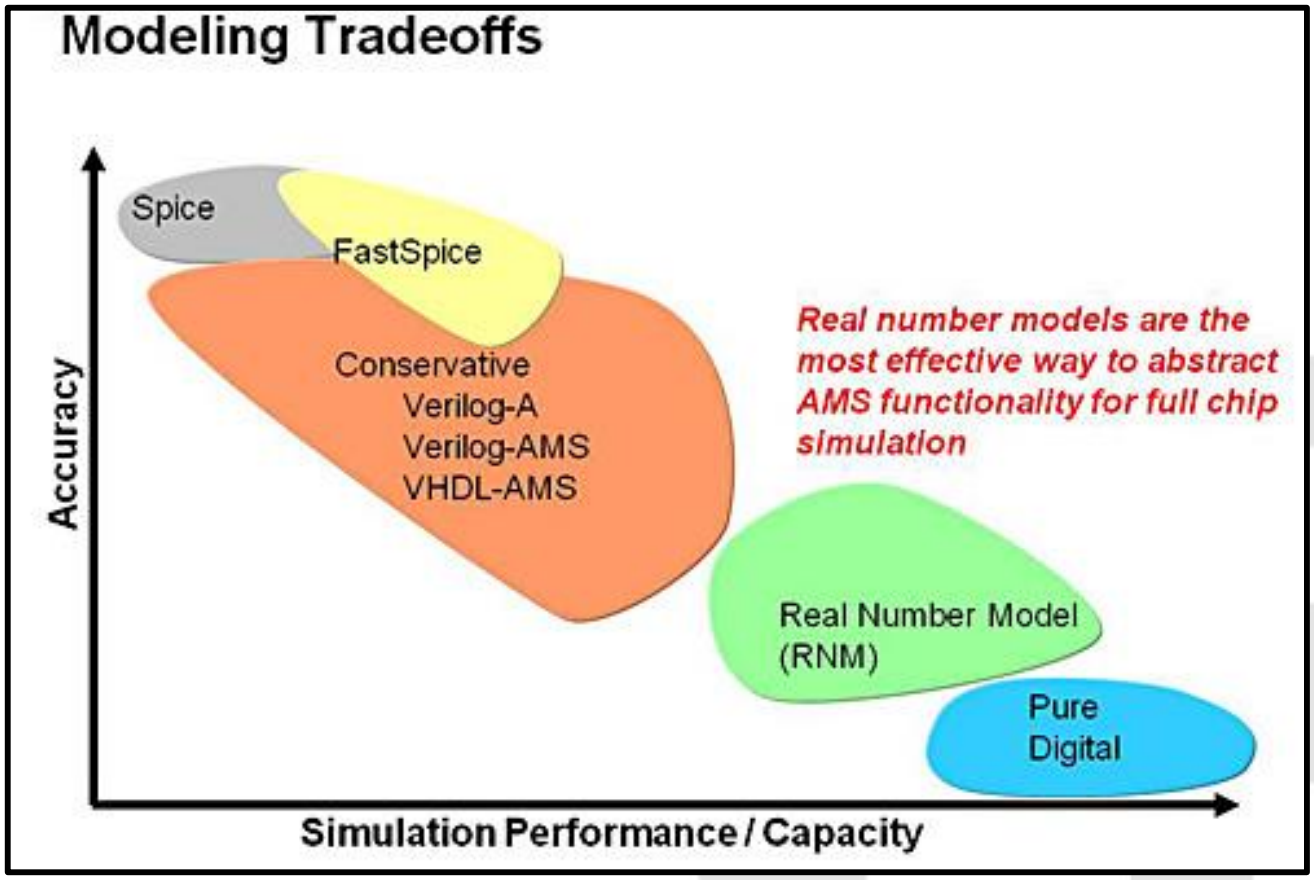
For the white paper "**A Systematic Approach to Creating Behavioral Models**" accompanying this presentation, follow this link:

http://tinyurl.com/Xtreme-EDA-Customer-Resources

A Systematic Approach to Creating Behavioral Models

Introduction – Model accuracy and trade-offs

- **Cadence Lava Lamp – Accuracy versus Performance**
  - http://community.cadence.com/CSSharedFiles/blogs/ms/2012/BS_Sim_Performance.jpg



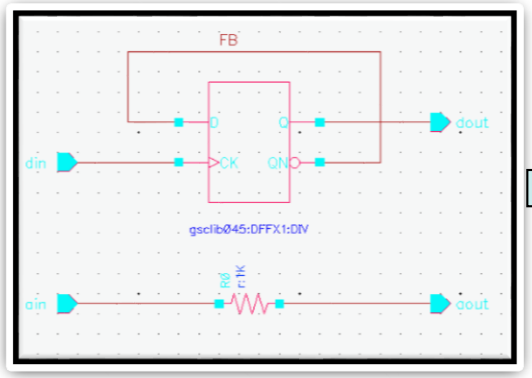
Introduction – Good and bad models

- What about behavioral models?
  - Model writer bridges the gap between analog design and digital verification
  - Model writer knows the analog behavior and knows how to target toward UVM
- "A Bad Model is Worse than No Model"
  - When model ignores faulty inputs
  - Incorrect behavior
- Our approach to creating, validating and maintaining good models is what this presentation is all about

Creating Models – Keep Cell Views Sync'd

- Open a schematic view of the block to be modeled
- Use "create Verilog-AMS" to automatically create the shell of a Verilog-AMS model, with I/O matching the schematic

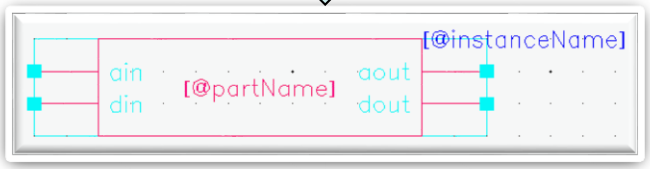
Schematic



Verilog-AMS

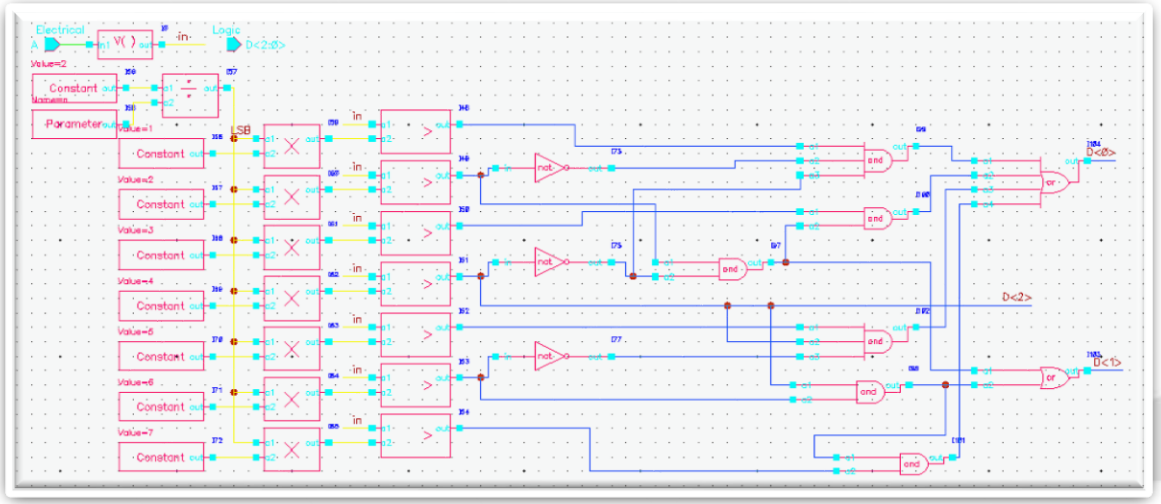
```
//Verilog-AMS HDL for "SandBox", "cell1" "verilogams"

`include "constants.vams"
`include "disciplines.vams"

module cell1 ( aout, dout, ain, din );

  output dout;
  output aout;
  input din;
  input ain;
endmodule
```

Symbol



Creating Models – Top Down with SMG

- Top-Down Approach: Designer works to match the model.
- SMG is a good way to create the initial model for the designer to match

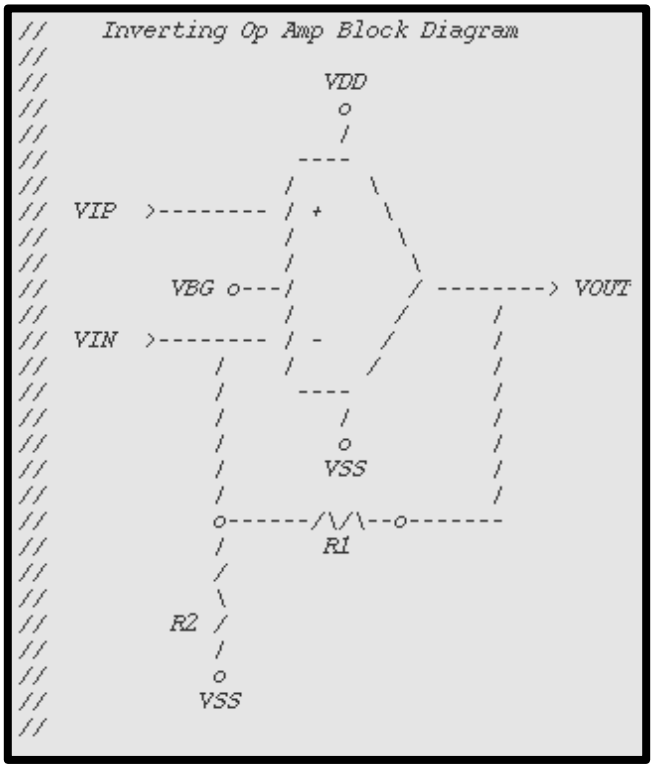


Graphical model is simulate-able, and may be compiled into Verilog-AMS (electrical or wreal) or System Verilog with real signals.

```
module flash_adc(A,D);

    input       A;
    output      [2:0] D;

    electrical A;

    parameter integer n = 8;


    real net055;
    real net064;
    real net050;
    real net018;
    real net053;
    real in;
    real LSB;
```

Creating Models – Learn the Circuit

- Study the schematic
- Interview its designer
- Develop a description of the circuit behavior. Narrative text, equations, tables, sketches of diagrams and waveforms
- Comment the description into the model file. If you use nedit, ASCII drawing is nearly as easy as Power Point.




```
// GAIN[2:0] / Gain(db) / Gain V/V
// ------------------------------------
//      000   /     0     /  1.0000
//      001   /     1     /  1.1220
//      010   /     2     /  1,2500
//      011   /     3     /  1.4125
//      100   /     4     /  1.5849
//      101   /    X [1]   /  0.0001
//      110   /     X     /  0.0001
//      111   /     X     /  0.0001
//
//  [1] Disallowed. Use assertion.
//
```

Creating Models – Learn the Circuit

- The designer blesses the description

- Henceforth, the designer is obligated to notify the model's owner of any changes to pinout or functional behavior

- The model owner is obligated to update the description as well as the model

Creating Models – Plan the Model Views

- Decide what to include and what to leave out from the model
  - Collaboration between stake holders
  - Model Designer is the bridge between Circuit Designer and Verification Lead
- Verification plan drives the models

- Some behaviors which may be left out of the model
  - Bandwidth of a fixed-bandwidth amplifier, unless required by V-plan
  - Digitally programmable bandwidth, for most testcases
  - Sensitivity to magnitude of VDD, VREF, IBIAS. (Just check they're within bounds.)

Creating Models – Plan the Model Views

PLL Realistic Model

- Program selection of capacitance
- Program frequency dividers M, N, N-fraction
- Oscillator spin-up from runt pulses. May require an injected "kick"
- Control loop
  - Frequency, phase acquisition
  - Lock
  - Recovery from disturbances
  - Phase jitter
- Frequency set-point changes
  - Re-programming
  - Same control loop

Dedicated Verilog-AMS testbench

- Low levels may be transistor level
- Characterize worst-case delays

Full chip testbench

- Maybe 2 testcases need this much detail

PLL RNM

- Program selection of capacitance
- Program frequency dividers M, N, N-fraction
- Wait for characterized worst case delay
  - Output ideal clock
  - Optionally inject disturbance
  - Optionally inject phase jitter
- Frequency set-point changes
  - Re-programming
  - For the characterized delay, ramp frequency transient

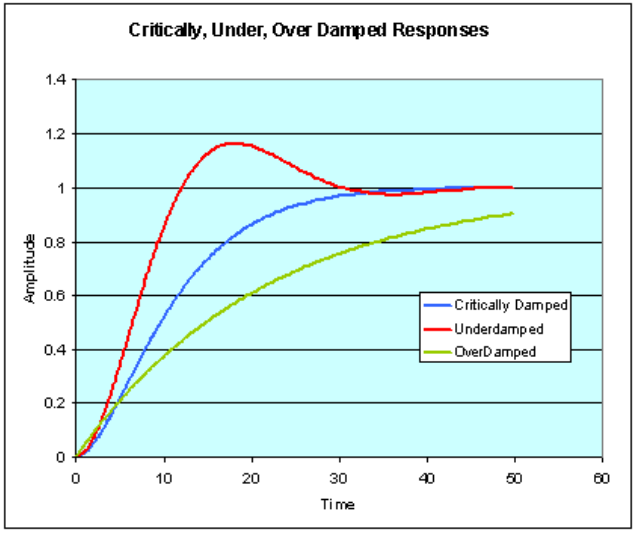
Full chip testbench

- Nearly every testcase

Creating Models – Plan the Model Views

AFE Path Startup Transient Realistically Modeled

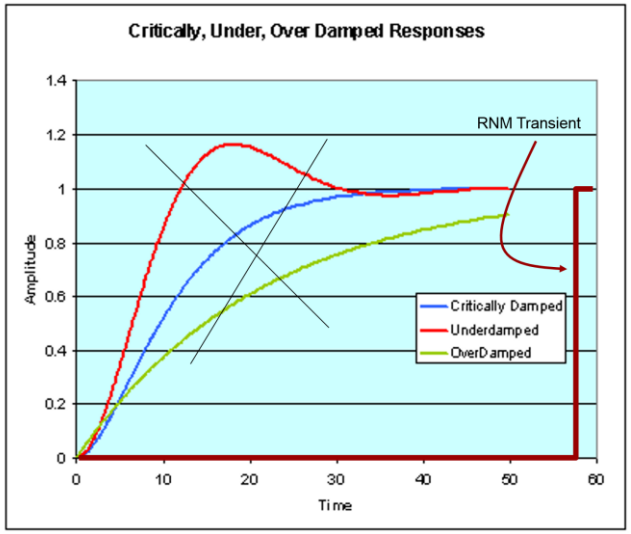
- Transient response up to the A/D input



- Static DC step without any AC applied
- Is this detail necessary?
  - Maybe for some testcases
  - Mostly not

AFE Path Startup Transient RNM Modeled

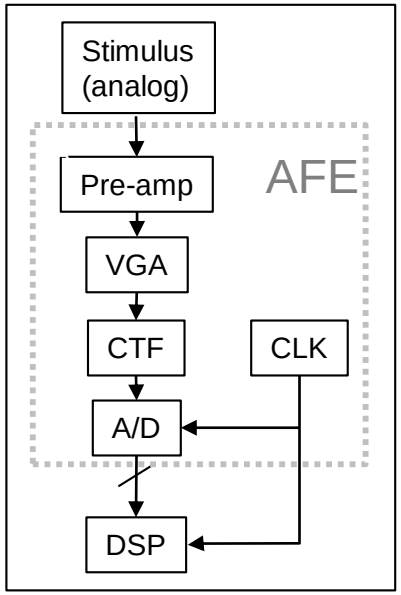
- Wait and step



- Key behavior is the wait-time
- With RNM the step will not cause a loss of simulation convergence.
- Is this good enough?
  - Depends on the V-Plan

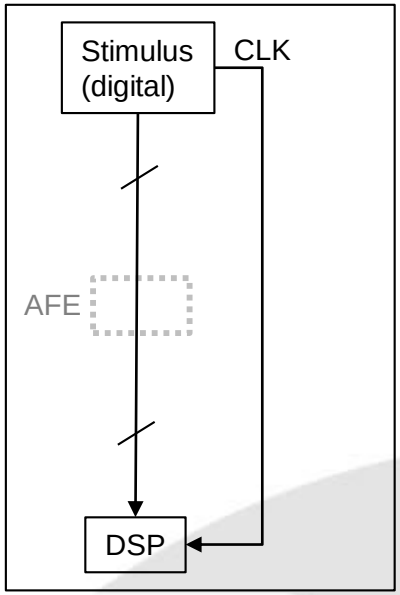
Creating Models – Plan the Model Views

AFE Realistic Models



- Needed for front end verification
  - Analog signal flow
  - Digital Control
  - Calibration, Compensation
  - Adaptation (feedback)

AFE Model for Digital Stimulus



- For verifying downstream digital
- Provides zero AFE coverage
- Is this okay?
  - Depends on the V-Plan

Creating Models – Plan the Model Views

- The lead verification engineer and circuit designer bless the plan.

- Comment the plan into the model description.
  - Archive the model with description, and always revise the description if you change the model

Creating Models – Write the Model Code

- Write the model to fit its description.
  - This is the easy part.

- Validating the model is the toughest part.

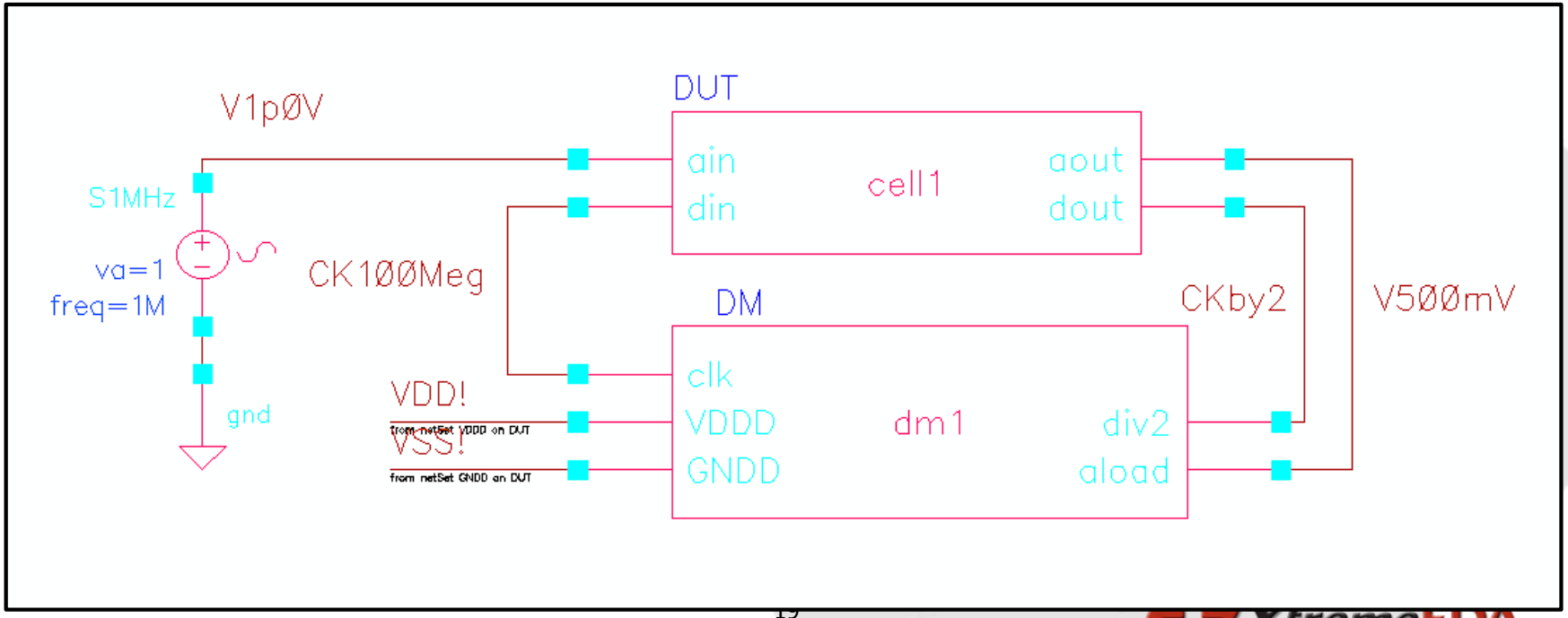
Validating Models – Accurate versus plan

- Validation answers two questions:
  - Does the model match the model plan?
  - Does the model respond correctly to all input scenarios, including illegal combinations and bad logic?

- Validation testbench
  - Separate from full chip testbench
  - Testbench schematic connects symbols for the (DUT) and driver-monitor (DMON).
  - Sometimes it makes sense to build up a validation testbench for more than one cell

Validating Models – Test Benches

- Validation ought to be independent and disinterested, but it seldom turns out that way.
- Circuit designer would be a likely candidate to validate the model, but may not know the modeling language or the intricacies and pitfalls of the AMS simulator
- Our approach to validation, beginning with testbench



Validating Models – Collaborative Effort

- **Does the model match the model plan?**

1. Circuit and model designers collaborate to describe analog and digital stimulus
2. Model designer creates DMON stimulus
3. Simulates schematic and model.
4. Debugs until they "match"
5. Delivers a turnkey simulation scenario to circuit designer
6. Circuit designer reruns simulation in a familiar environment
7. Circuit designer "blesses" model and stimulus

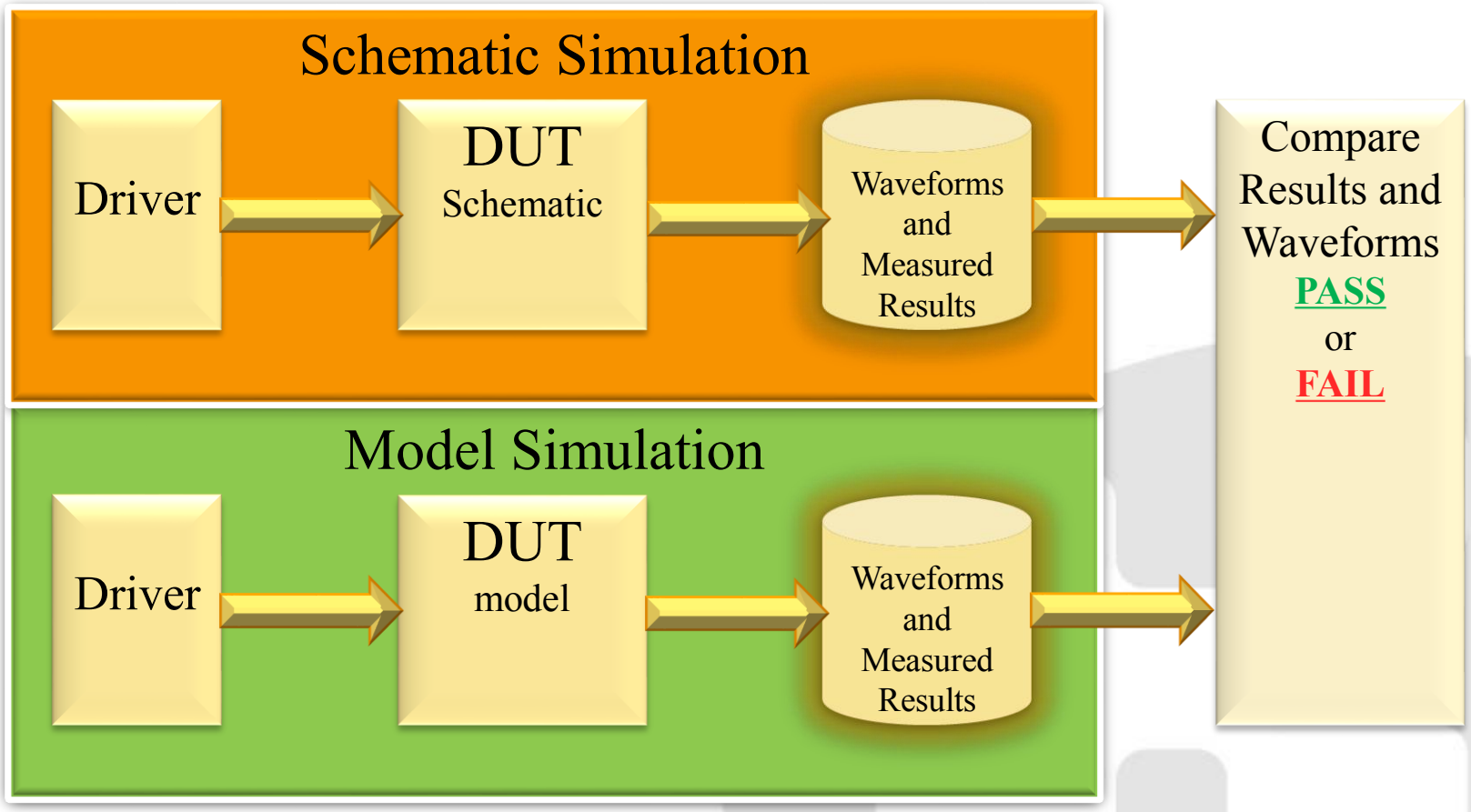
Validating Models

- **Does the model respond correctly to all input scenarios, including illegal combinations and bad logic?**

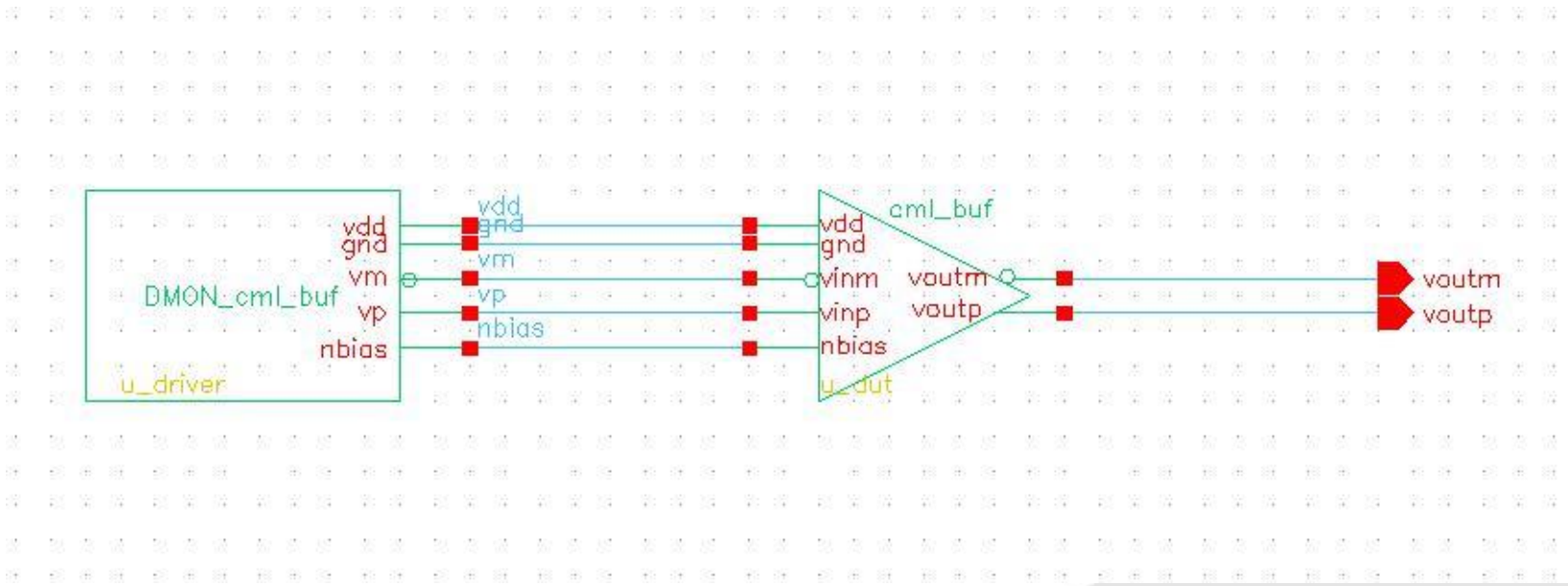
1. Use constrained random approach where possible
2. Include out-of-range analog
3. Include illegal combinations of logic
4. Include unknown ‘x’ states
5. Want to ensure the model doesn’t “pretend” everything is okay

Maintaining Models Through Project Lifetime

- Model Validation and Regressions using Cadence AMSDMV (AMS Design and Model Validator)

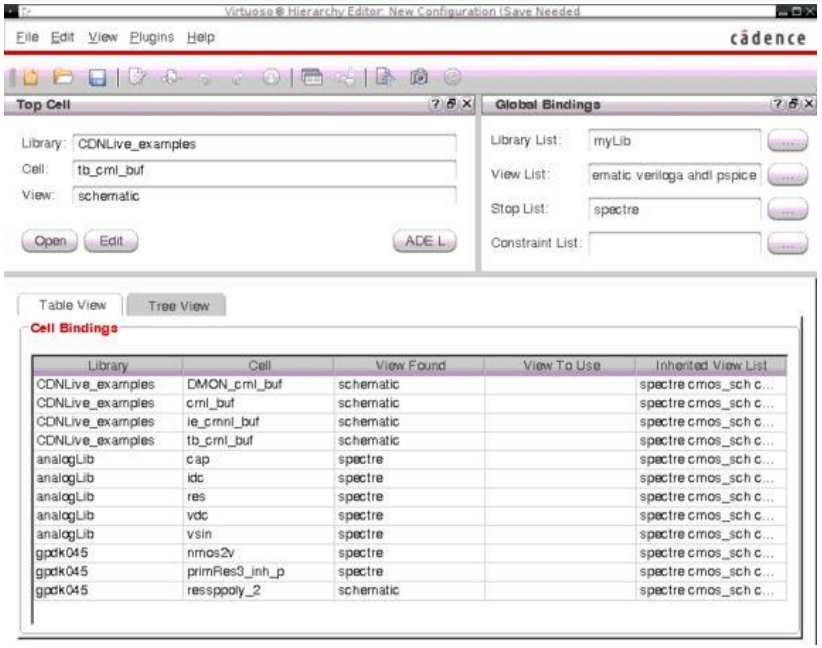


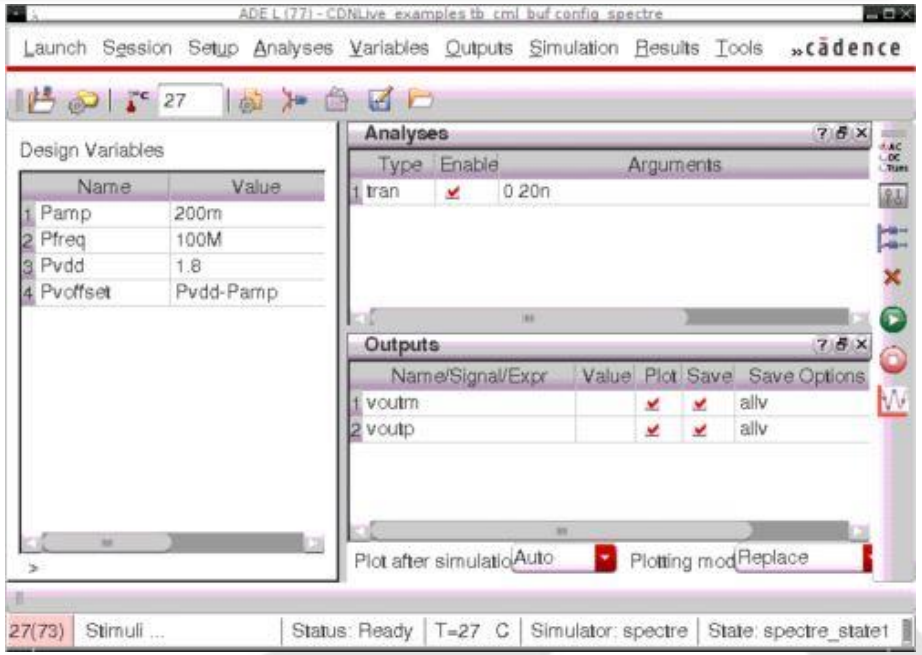
Virtuoso Testbench



- DUT: CML Buffer
- DMON
  - Generates the power supplies and bias current
  - Drives the differential input signals to the DUT

Configuration and ADE State for Spectre

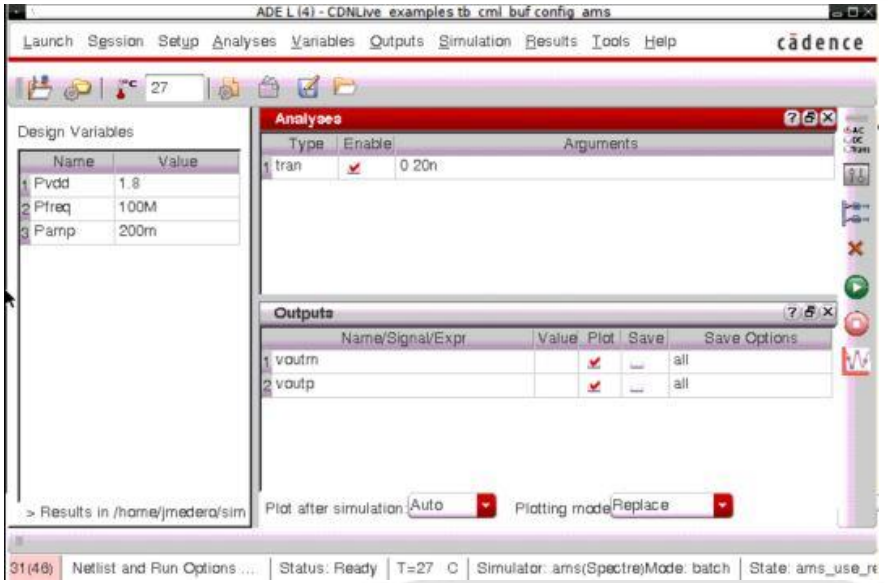




- The configuration file used is to show differences with the AMS configuration files.

XtremeEDA

Configuration and ADE State for AMS



Needed for Wreal
Leave blank for
Pure Verilog

XtremeEDA

CML BUF Dual Path Model (RNM or Verilog)

```
//Verilog-AMS HDL for "CDNLive_examples", "cmnlbuf" "wreal"


`include "xeda_defines.vh"
module cml_buf (
    output `XEDA_WREAL voutm,
    output `XEDA_WREAL voutp,
    input  `XEDA_WREAL vinm,
    input  `XEDA_WREAL vinp,

    input              nbias,
    input  `XEDA_WREAL gnd,
    input  `XEDA_WREAL vdd

);

`XEDA_CHK_PWR(vdd, gnd, nbias)

assign voutm = (pwrgood===1'b1) ? vinm : `XEDA_PWR_OFF_STATE;
assign voutp = (pwrgood===1'b1) ? vinp : `XEDA_PWR_OFF_STATE;


endmodule
```

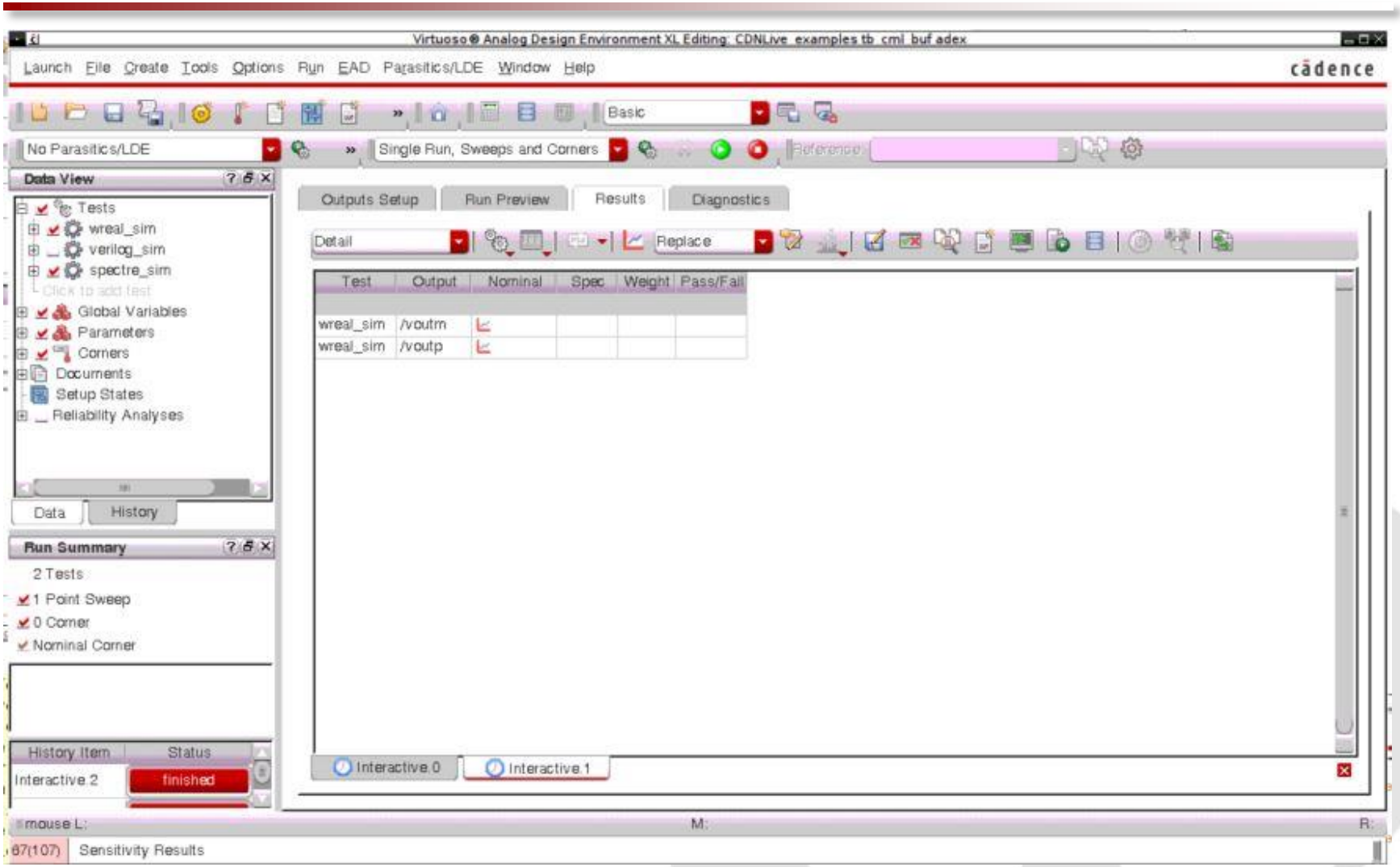
xeda_defines.vh (modified) - /home/jmedero/cadence/

File Edit Search Preferences Shell Macro Windows

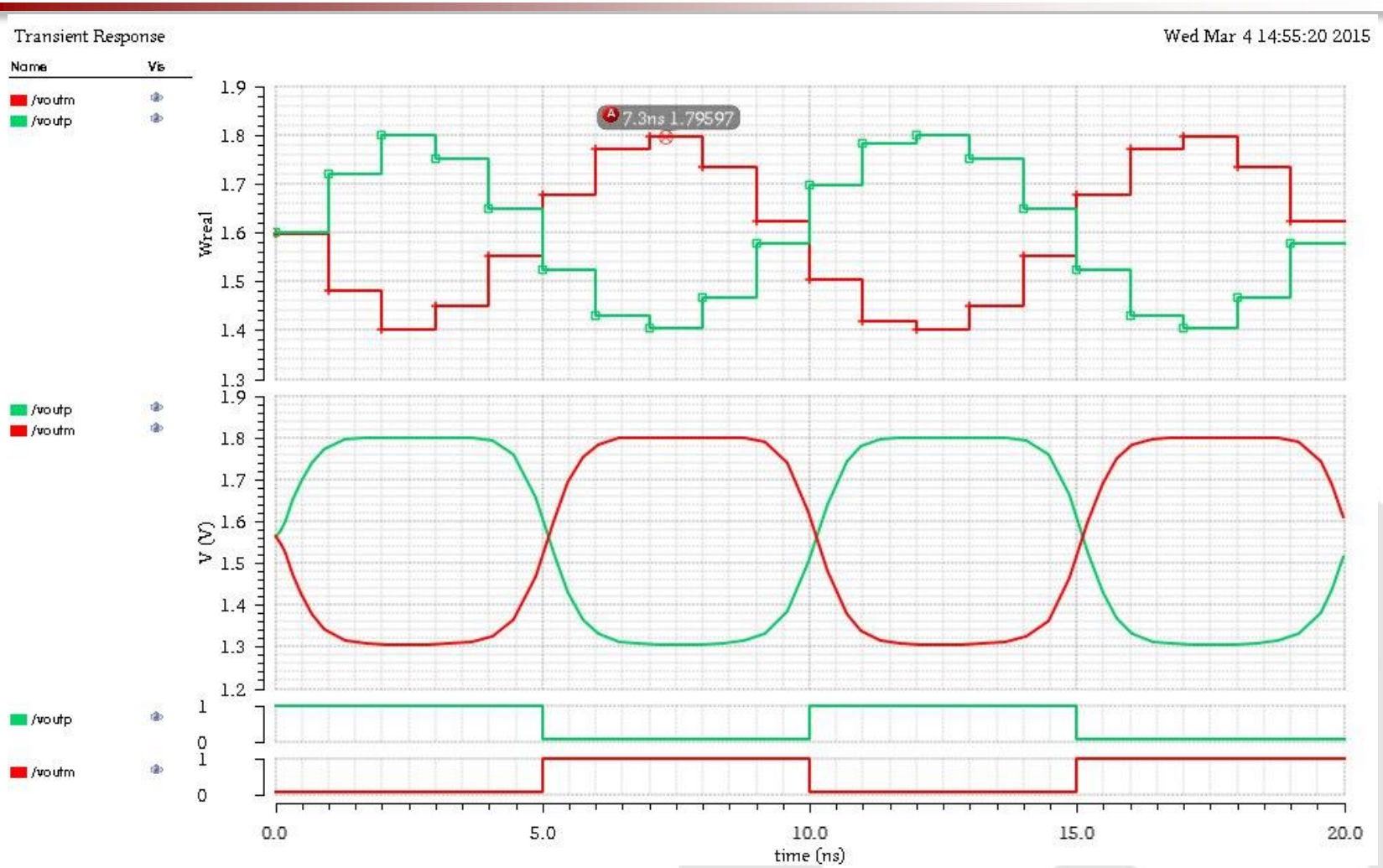
```
// XEDA DEFINES
`define XEDA_POWER_AWARE
`define XEDA_VDD_MIN        1.7
`define XEDA_VDD_MAX        1.9
`define XEDA_PWR_OFF_STATE  0

`ifdef XEDA_USE_REAL
   `define XEDA_WREAL wreal
`else
   `define XEDA_WREAL
`endif

`define XEDA_CHK_PWR(vdd, gnd, nbias) \
   wire pwrgood; \
   `ifdef XEDA_POWER_AWARE \
      `ifdef XEDA_USE_REAL \
         assign pwrgood = (( (vdd-gnd) > `XEDA_VDD_MIN  \
                         && (vdd-gnd) < `XEDA_VDD_MAX) \
                         && (nbias === 1'b1)) \
                         ? 1'b1 : 1'b0; \
      `else \
         assign pwrgood =  (vdd===1'b1) && (gnd===1'b0) && (nbias===1'b1); \
      `endif \
   `else \
      assign pwrgood = 1'b1; \
   `endif
```

- Option: -define XEDA_USE_REAL
  - When defined uses Wreal
    - `XEDA_WREAL = wreal
  - When not it is a digital model
    - `XEDA_WREAL = “”

ADE XL Simulation Environment



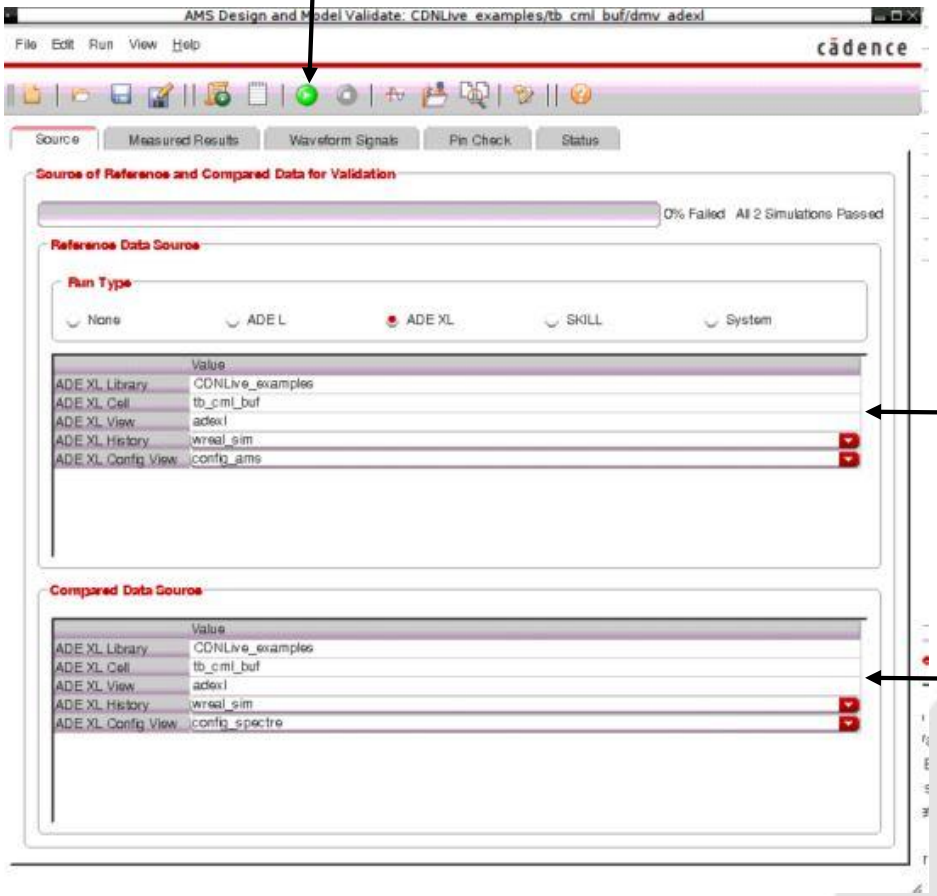
Simulation Results



XtremeEDA

AMSDMV Setup

Press the green button to run sims



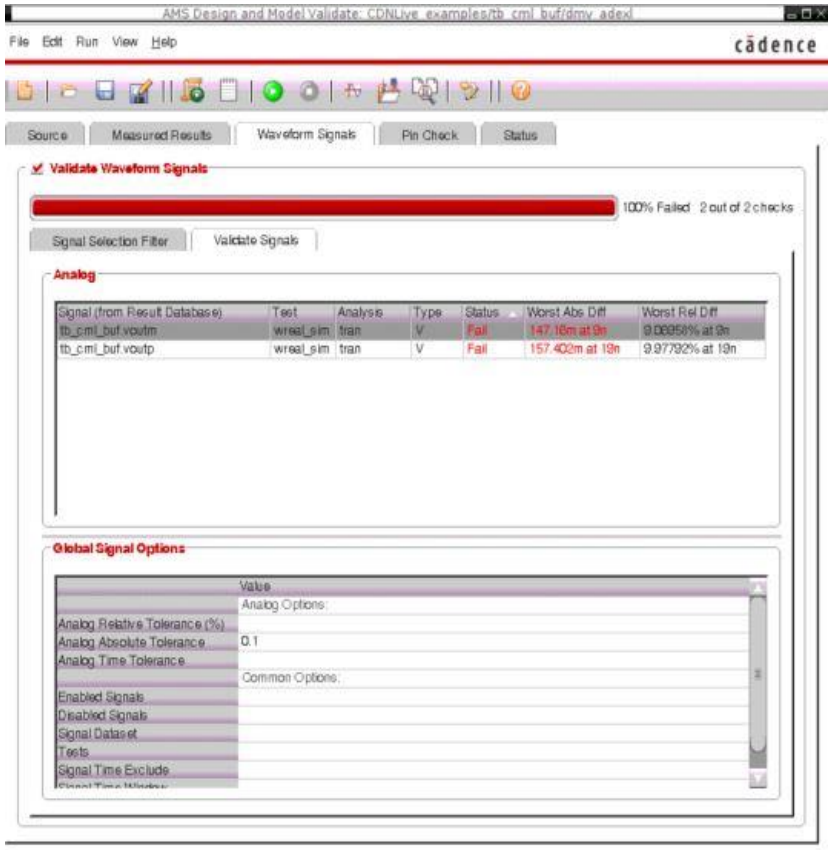
Simulation complete message

Reference Simulation

Compare Simulation

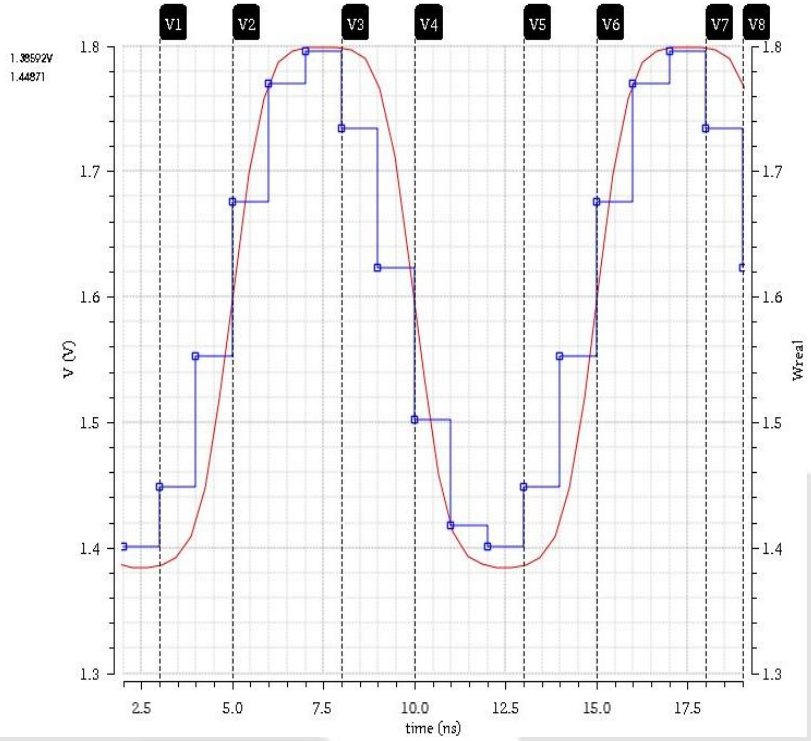
XtremeEDA

AMSDMV Failed to Match Case



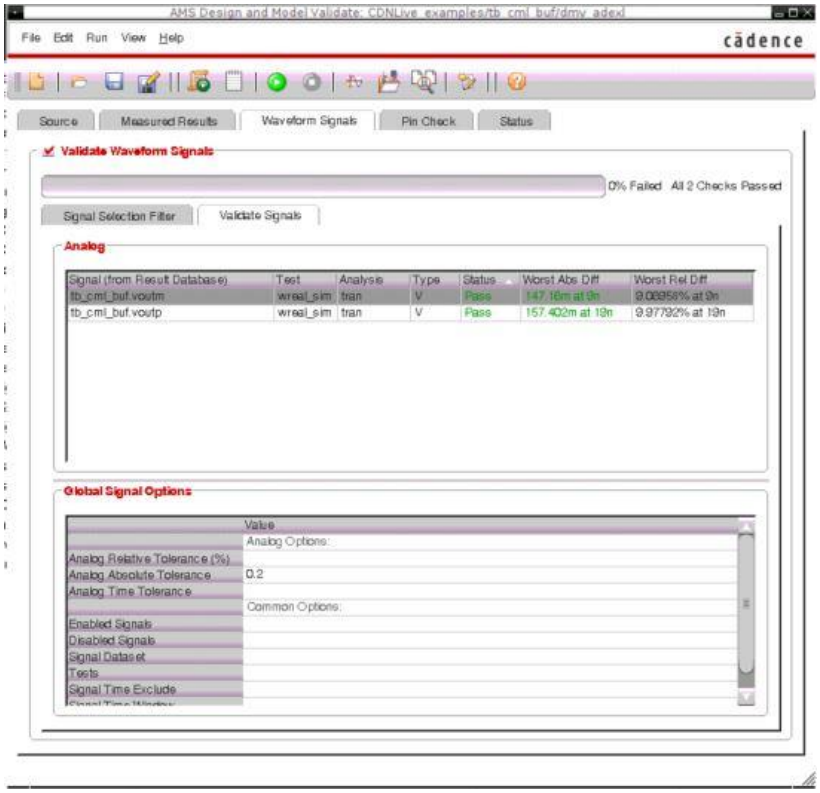
- Absolute Tolerance Set to 100mV

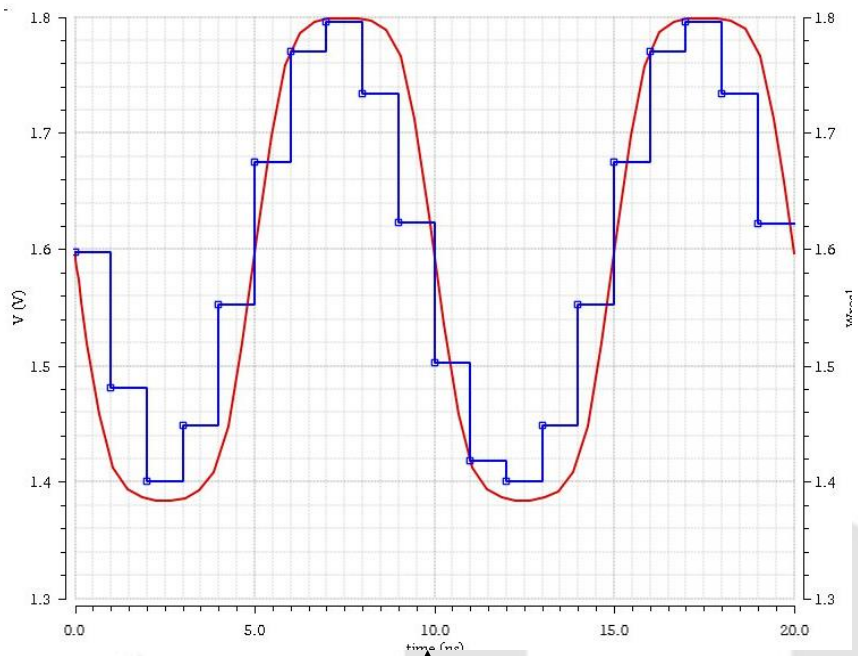
- Failure Reports



	Start	End
1	3n	5n
2	8n	10n
3	13n	15n
4	18n	19n

AMSDMV Pass



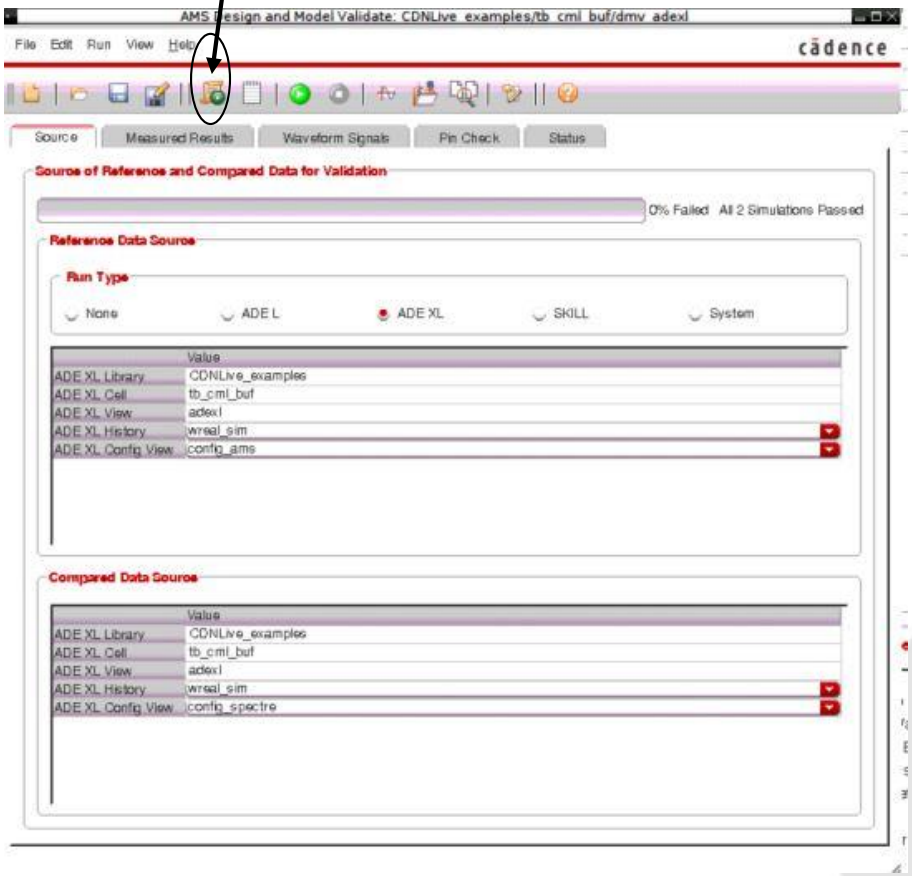


- Absolute Tolerance Set to 200mV
- Alternatively, we could have set the E2R connect module to provide better accuracy.

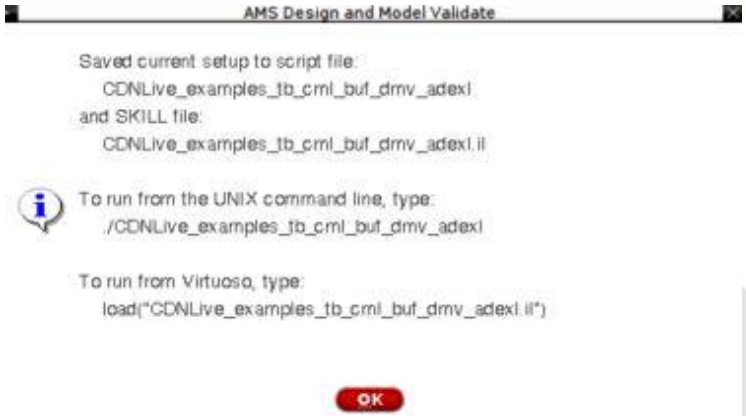
- No Error Points Shown

AMSDMV Save Regression Scripts

Button used to create command line Scripts for future regressions



Message shows the location of the scripts and how to use them.



XtremeEDA

Models and Applications

- Analog Verification Models
  - System Level/Architecture Models
    - VerilogA, VerilogAMS, analog library components, SMG Library
    - Proof of concept – schematics/topologies not completely defined
    - Determining specifications for each block
    - Example: ADC, …
  - Analog Mixed Signal Models
    - VerilogA, Verilog AMS, VHDL AMS, Verilog/VHDL, etc.
    - Provides simulation speedup
      - Top level simulations are able to run in reasonable time.
      - Individual blocks can be tested thoroughly
      - When including the digital blocks it provides valuable feedback between analog and digital interface functionality and performance.
      - Example: Test bench for a frequency synthesizer
    - Cadence SMG (Schematic Model Generator) allows automatic generation of these types of models.

Models and Applications

- Digital Verification Models
  - Verilog, VHDL, System Verilog, System C, …
  - High Level representation of the Analog block
    - Normally written early by the digital team
    - Does not include the full Analog functionality
    - Interconnectivity and hierarchical differences can be prevalent
    - Dynamic changes to analog blocks and functionality is not tracked.
    - Specifications may not be up to date and therefore lagging the design.
    - Metrics are not indicative of overall project status
      - They can give the RTL and verification teams a false sense of design completeness and coverage that is not present if the full analog system/model is included.

Models and Applications

- **Real Number Models (RNM)**
  - Wreal, System Verilog, …
  - Use in Analog, Digital, and Mixed-Signal environments
    - Virtuoso, Incisive (irun) with directed and UVM tests.
  - Allows for tests to be shared between Analog and Digital Environments.
  - Allows for complete system verification
    - Code Coverage
    - Assertions (Formal, Simulation)
    - Functional Coverage
    - Constrained Random Simulation
    - Power Aware Design and Verification
      - CPF/UPF, multi-power domain analysis, isolation, in-rush current, level-shifting, state initialization, state retention, clock-gating, reset, power and current distribution, Power-State Machine verification.

Dual Path Model (RNM or Verilog)

```
//Verilog-AMS HDL for "CDNLive_examples", "cmnlbuf" "wreal"


`include "xeda_defines.vh"
module cml_buf (
    output `XEDA_WREAL voutm,
    output `XEDA_WREAL voutp,
    input  `XEDA_WREAL vinm,
    input  `XEDA_WREAL vinp,

    input              nbias,
    input  `XEDA_WREAL gnd,
    input  `XEDA_WREAL vdd

);

`XEDA_CHK_PWR(vdd, gnd, nbias)

assign voutm = (pwrgood===1'b1) ? vinm : `XEDA_PWR_OFF_STATE;
assign voutp = (pwrgood===1'b1) ? vinp : `XEDA_PWR_OFF_STATE;


endmodule
```

xeda_defines.vh (modified) - /home/jmedero/cadence/

File Edit Search Preferences Shell Macro Windows

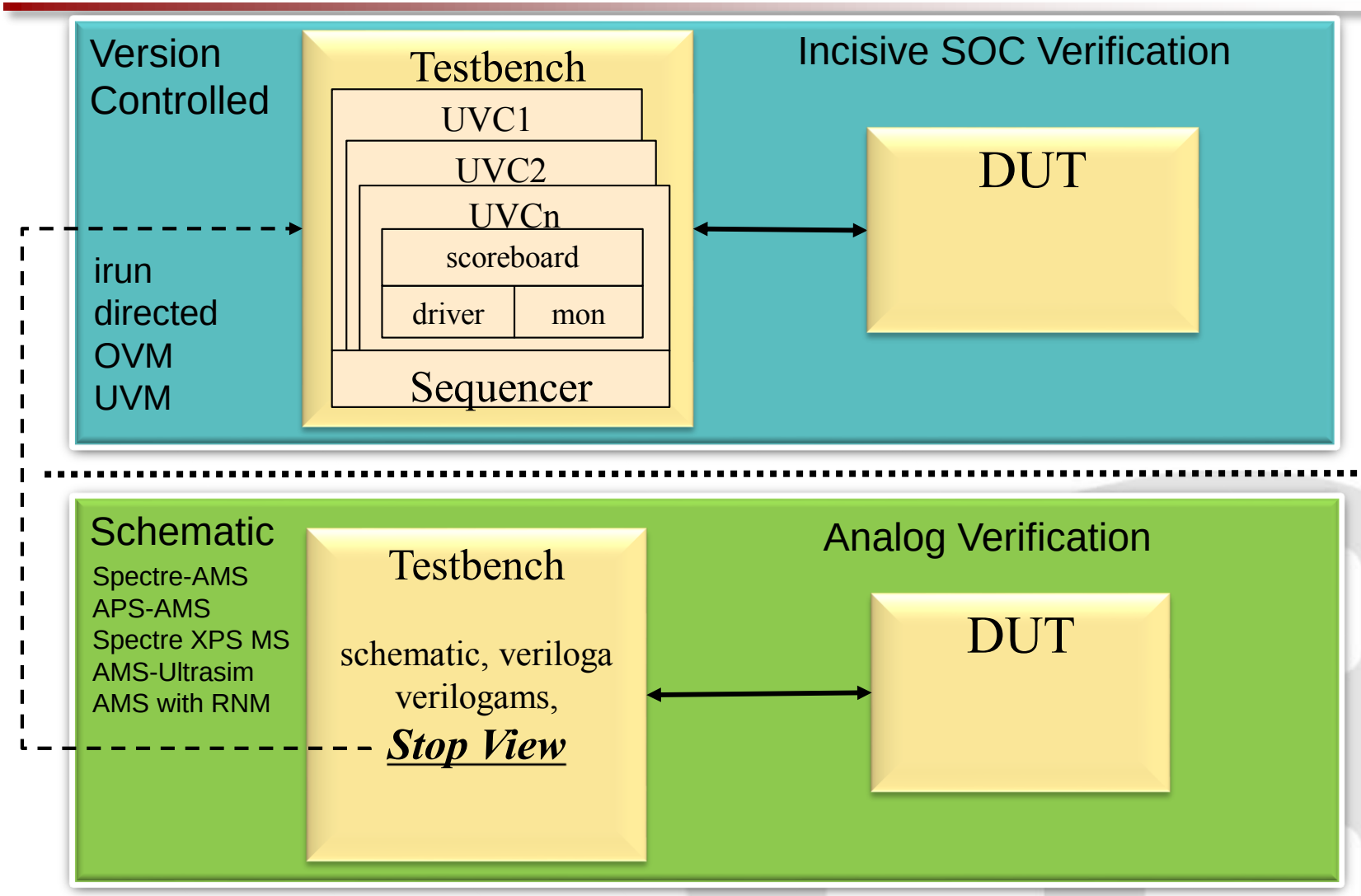
```
// XEDA DEFINES
`define XEDA_POWER_AWARE
`define XEDA_VDD_MIN        1.7
`define XEDA_VDD_MAX        1.9
`define XEDA_PWR_OFF_STATE  0

`ifdef XEDA_USE_REAL
   `define XEDA_WREAL wreal
`else
   `define XEDA_WREAL
`endif

`define XEDA_CHK_PWR(vdd, gnd, nbias) \
   wire pwrgood; \
   `ifdef XEDA_POWER_AWARE \
      `ifdef XEDA_USE_REAL \
         assign pwrgood = (( (vdd-gnd) > `XEDA_VDD_MIN  \
                         && (vdd-gnd) < `XEDA_VDD_MAX) \
                         && (nbias === 1'b1)) \
                         ? 1'b1 : 1'b0; \
      `else \
         assign pwrgood =  (vdd===1'b1) && (gnd===1'b0) && (nbias===1'b1); \
      `endif \
   `else \
      assign pwrgood = 1'b1; \
   `endif
```

- The code is written to provide a pure digital path or a wreal path.
  - Digital path used by the digital verification team without impact on simulation performance.
  - Wreal used by Analog team or selected digital sims for functional coverage.
- Wreal and digital path code is shared so code coverage metrics are the same.
- XEDA_POWER_AWARE switch checks power if defined.
- The XEDA_CHK_PWR macro is an example of checking power consistently and to clean up the main code.
  - Assertions can also be added to the macro if needed.

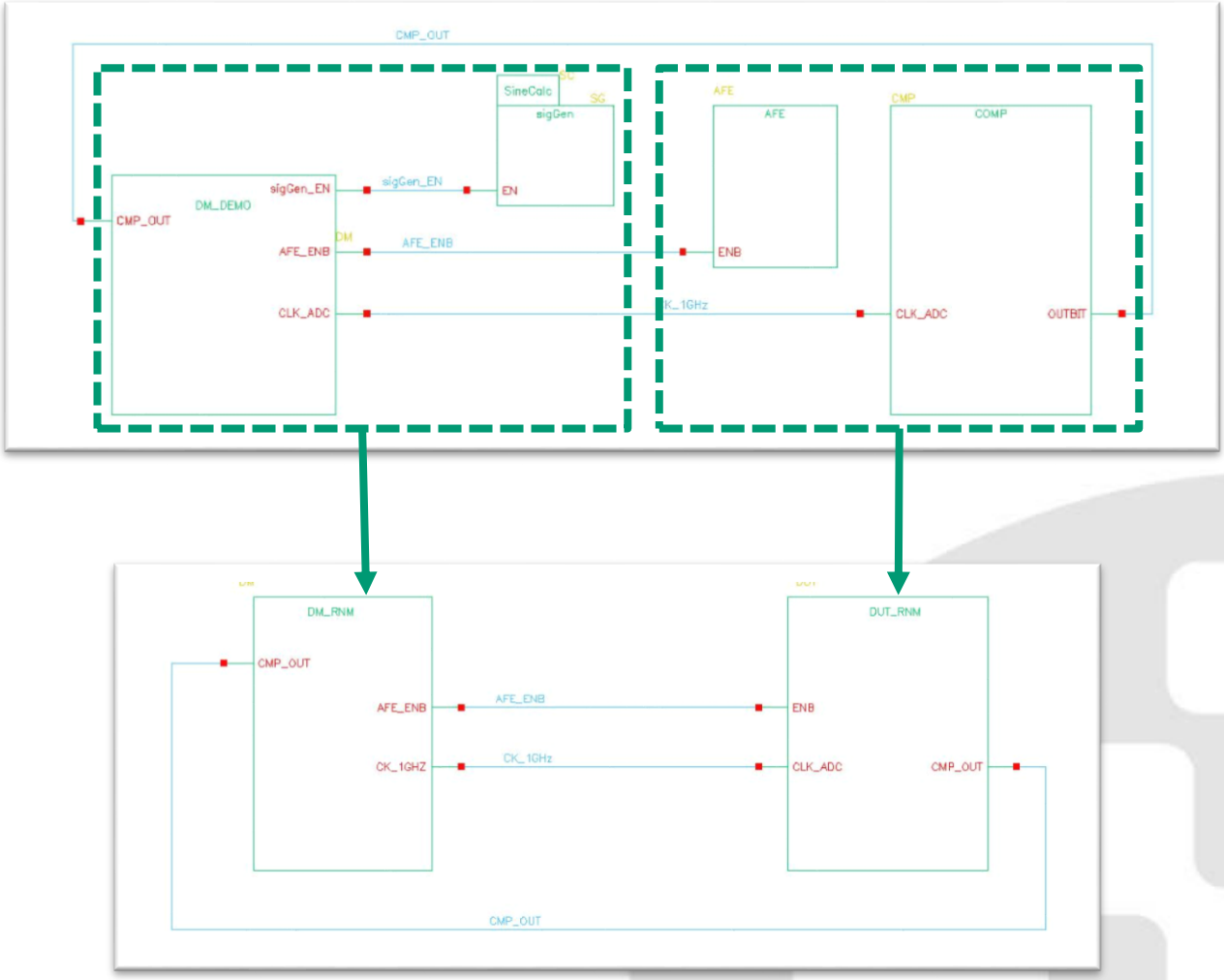
RNM Verification

- RNM allows now the Analog and Digital teams to perform verification on the individual platforms to verify the system quickly.
- If an issue is found in the RNM models used by the digital verification team debugging time may take too long.
  - Digital teams have limited analog background and are not familiar with the circuits and operation.
  - Analog teams are unfamiliar with the verification environment and this limits the help they can provide.
    - The Analog Engineer then tries to create a test bench that mimics the issues found in the system verification.
      - Time consuming and no guarantee that the system is well simulated.
- How do we solve these issues?
  - By providing a Common Verification Platform in the analog environment that uses the digital verification teams platform and tests to speed up test bring up time and to allow quick cooperation in debugging issues.

Common Verification Platform

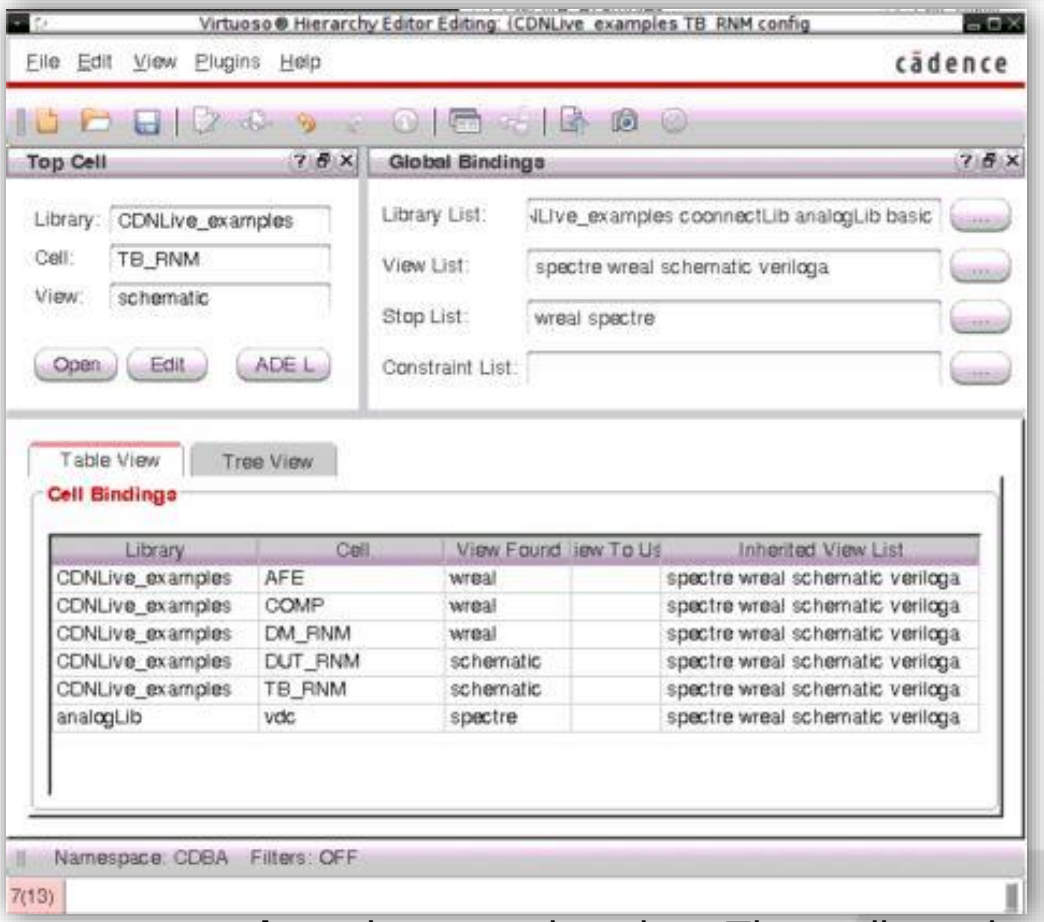


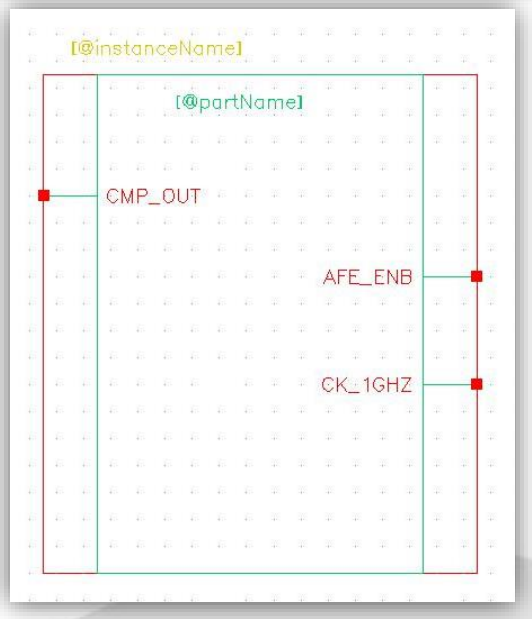
Updating Analog Testbench for CVP



XtremeEDA

Configuration File

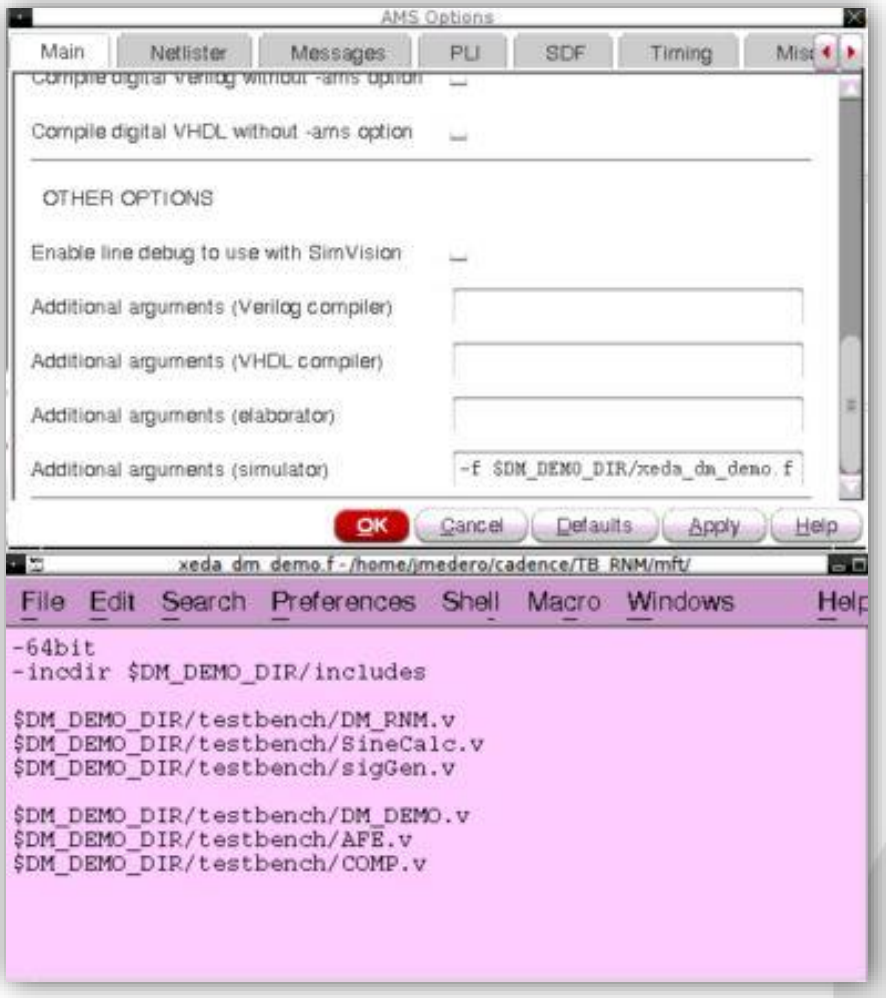




wreal is just a symbol.

wreal used as stopping view: The netlister does not look inside the block.
The model needs to be provided at run time.

Finding the instances

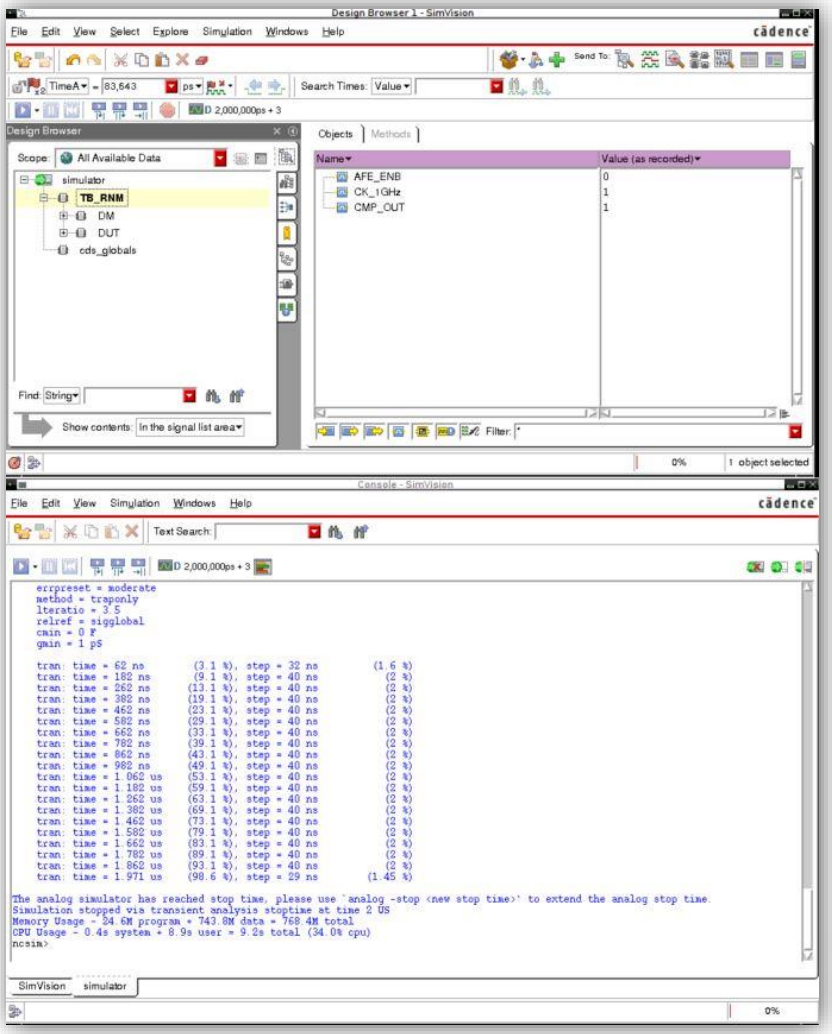


- Use the AMS Options Menu to specify the manifest and command files to run.
- $DM_DEMO_DIR is set with the project setup scripts.
- The xeda_dm_demo.f is shown at the left.

ADE Simulation Netlist and Run Options

- From ADE select the netlister and simulation engine.
  - In this case:
    - Netlister: OSS
    - Simulator: irun
- Can select to run Batch or Interactive
  - Interactive selected to debug issues.

Digital Control from ADE AMS Simulation



- In interactive mode
  - Have access to digital debug environment
  - Can load saved svcf files from the digital team to quickly load the signals and zoom on issues.

Using Virtuoso to trace signals

Virtuoso Visualization & Analysis Browser

Can use Analog Tracer to select signals from the schematic

XtremeEDA

Using Simvision to trace signals

Simvision

And/or can use Simvision to work with the digital verification team

CVP Highlights

- Specification and Verification driven designs
  - The hierarchy is built with the same structure and verification purpose.
  - Planning, Performance Metrics, Functional Coverage, Code Coverage, Assertions, Power Management is owned by all.
- The top level (or block level) test bench can be instantiated in an Analog Schematic (Virtuoso).
  - A "stopping view" is selected when the digital TB is to be used.
  - The selection of tests, required files, etc are added to the AMS options form allowing the same "irun" command to run in Virtuoso.
  - Engineers don't have to spend extra time generating the tests and familiarity with the debugging environments increases the efficiency of the team (self-checking methods can be used).

CVP Goal

- Taking separate verification efforts and direction

- And bringing them together towards a common goal.

Summary

- Behavioral models must be accurate and targeted toward efficient design verification
- There is no totally automatic way to create behavioral models, it is a collaborative effort between the circuit designer, the verification lead and the model writer
- It is the collaborative nature of our procedure that increases the likelihood of error-free models, and error-free silicon

When it comes to bridging the analog + UVM design verification gap, AMS, and digital verification, think XtremeEDA. With over 60 engineers in North America, we are a leading provider of expert front end design and verification engineering services.