

ANSOFT CORPORATION PRESENTS:

CONVERGE



AN APPLICATIONS WORKSHOP FOR
HIGH-PERFORMANCE DESIGN

Applying VHDL-AMS and other Advanced Modeling Techniques

Steve Chwirka

AE – EM products

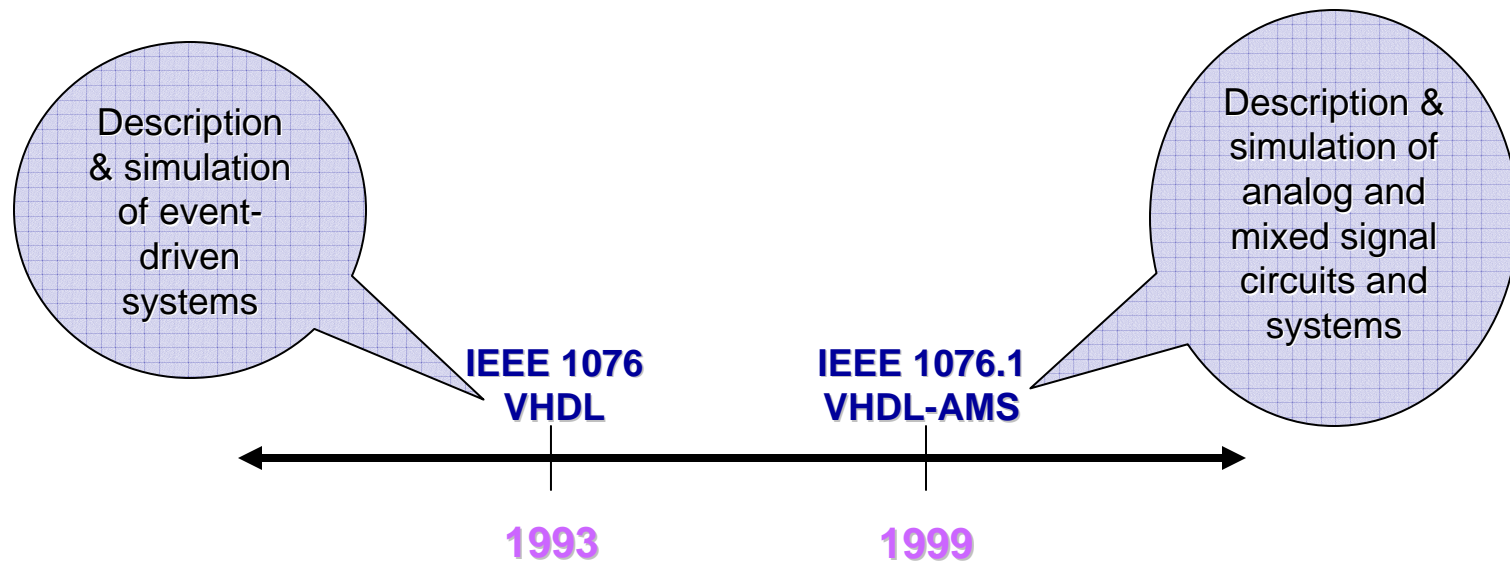
Applying VHDL-AMS and other Advanced Modeling Techniques



AN APPLICATIONS WORKSHOP FOR **HIGH-PERFORMANCE DESIGN**

What is VHDL-AMS

Very High Speed Integrated Circuit **H**ardware
Description **L**anguage – **A**nalog and **M**ixed-**S**ignal



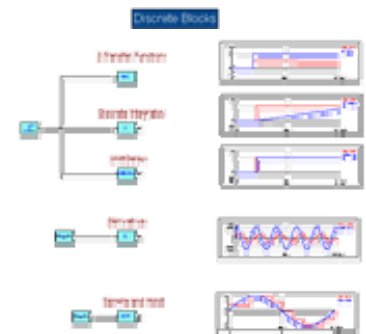
VHDL-AMS is a strict superset of IEEE Std. 1076

Why Use VHDL-AMS

- ▶ Standard Format Allows Model Portability
 - ▶ Different engineering groups within same company
 - ▶ With Sub-Contractors
 - ▶ Between different simulators
- ▶ Multi-level Modeling
 - ▶ Different levels of abstraction of model behavior
- ▶ Multi-domain Modeling
 - ▶ Electrical, Thermal, Magnetic, Mechanical, etc
- ▶ Mixed-signal Modeling
 - ▶ Supports analog and digital modeling



```
entity Resistor_Net at Resistor_Net
  port
    R1 : BIT_VECTOR (0 DOWNTO 0);
    R2 : BIT_VECTOR (0 DOWNTO 0);
    R3 : BIT_VECTOR (0 DOWNTO 0);
  end port;
  constant R1_Value : REAL := 1.0;
begin
  R1: ENTITY Resistor;
  generic map (R1_Value)
  port map (R1);
  R2: ENTITY Resistor;
  generic map (R2_Value)
  port map (R2);
  R3: ENTITY Resistor;
  generic map (R3_Value)
  port map (R3);
end Resistor_Net;
```



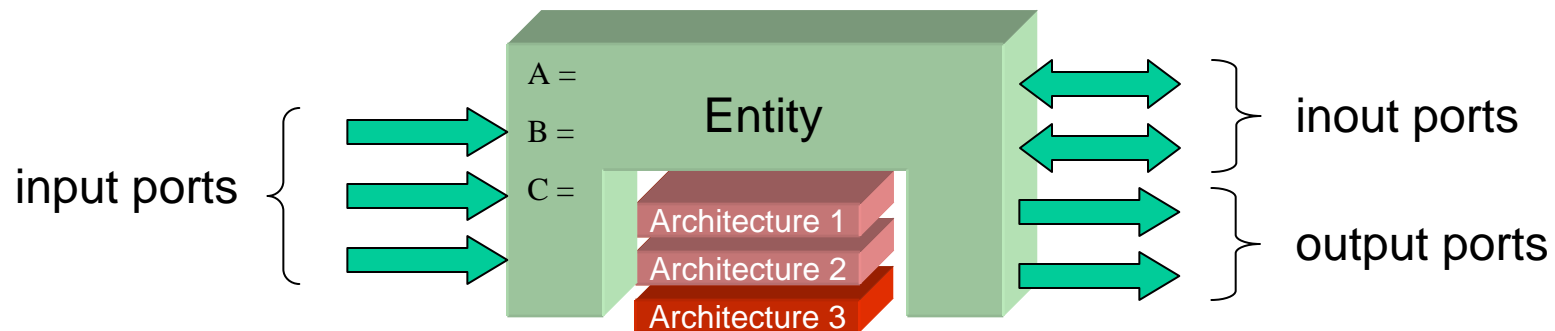
Basic VHDL-AMS structure

Entity

- ▶ Interface description of a subsystem or physical device
- ▶ Specifies input and output ports to the model
- ▶ Can specify input parameters to the model
(A =, B =, C = ..)

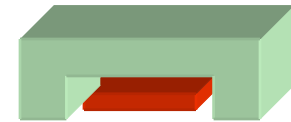
Architecture

- ▶ Can provide several levels of model description
- ▶ Modeling can deal with both analog (continuous) and digital (discrete) domains

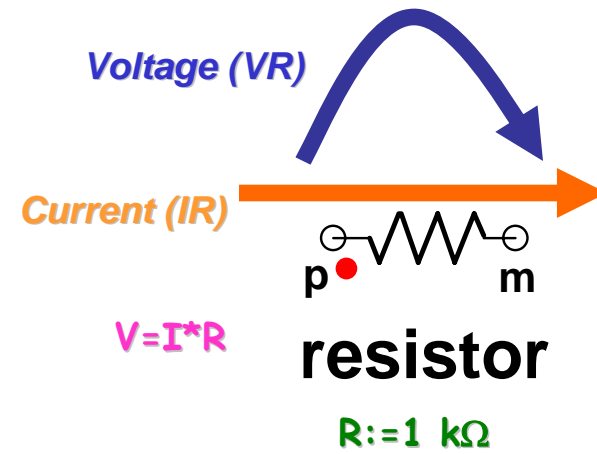


VHDL-AMS Simple Example

```
ENTITY resistor IS  
  GENERIC (R : REAL := 1.0e+03);  
  PORT (TERMINAL p,m : ELECTRICAL);  
END ENTITY resistor;
```



```
ARCHITECTURE behav OF resistor IS  
  QUANTITY VR ACROSS IR THROUGH p TO m;  
BEGIN  
  VR == IR * R;  
END ARCHITECTURE behav;
```



Applying VHDL-AMS Modeling

- ▶ This Example will utilize VHDL-AMS modeling for the DC/DC power converter design flow.
 - ▶ Component level (R, L, C, Diode, Switch, opamp, etc)
 - ▶ Create behavioral linearized average model for designing the feedback loop, and for system level simulations
 - ▶ Model digital components in the Modulation circuit block
 - ▶ System level control for - fuse, SSPC, circuit breakers
- ▶ In addition to VHDL-AMS modeling, other advanced modeling techniques will also be shown during this design process

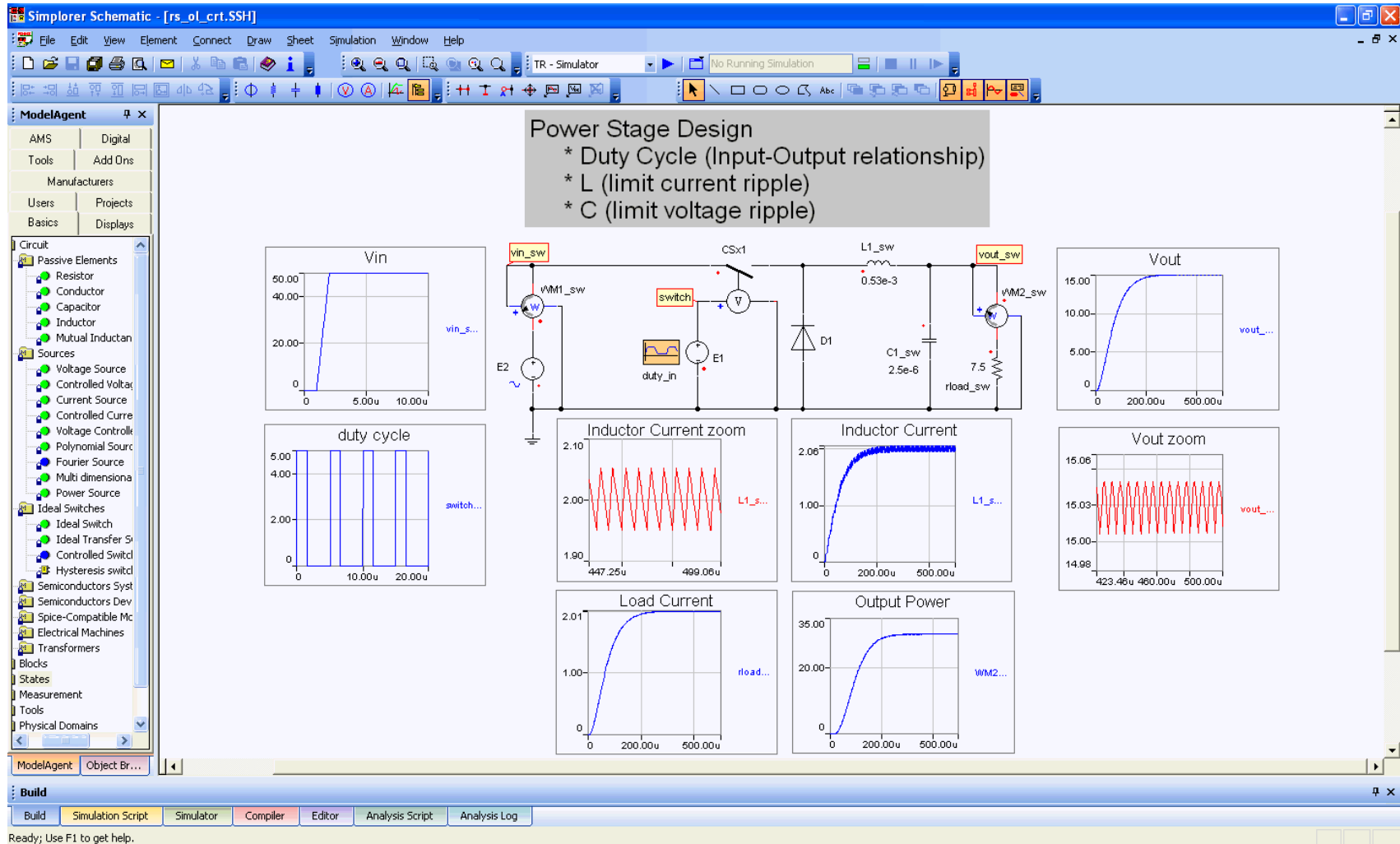


Design Process (Converter)

- ▶ Define Power Converter Specification
 - ▶ Input/Output Voltage requirements, Power levels
 - ▶ Mode of operation (CCM, DCM, Current or Voltage mode)
 - ▶ Ripple Voltage and Current specifications
 - ▶ Switching Frequency and target Efficiency
 - ▶ Load/line regulation
- ▶ Design Power Stage (open loop design)
- ▶ Design the feedback loops
- ▶ Design the modulation circuit
- ▶ Validate closed loop design



Power Stage Design



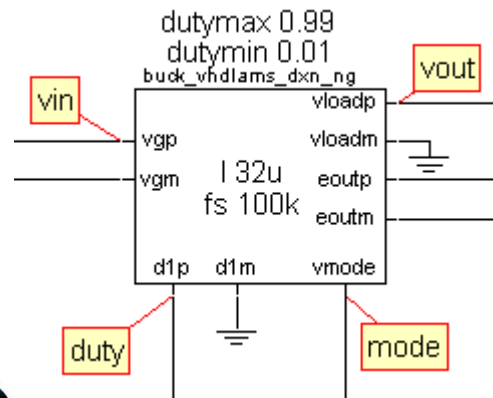
VHDL-AMS Behavioral Model

```

VhdlAms1.vhd buck_vhdlams_dxn_ng.vhd
LIBRARY ieee;
USE ieee.electrical_systems.all;
USE ieee.fundamental_constants.all;
USE ieee.math_real.all;
LIBRARY basic_vhdlams;

ENTITY BUCK_VHDLAMS_DXM_NG IS
  PORT (
    TERMINAL dlp : ELECTRICAL;
    TERMINAL dlm : ELECTRICAL;
    TERMINAL vgp : ELECTRICAL;
    TERMINAL vgm : ELECTRICAL;
    TERMINAL eoutp : ELECTRICAL;
    TERMINAL eoutm : ELECTRICAL;
    TERMINAL vloadp : ELECTRICAL;
    TERMINAL vloadm : ELECTRICAL;
    TERMINAL vmode : ELECTRICAL;
    QUANTITY L : IN REAL := 100.0E-6;
    QUANTITY Fs : IN REAL := 1.0E5;
    QUANTITY dutymin : IN REAL := 0.01;
    QUANTITY dutymax : IN REAL := 0.99
  );
BEGIN
END ENTITY BUCK_VHDLAMS_DXM_NG;

```



```

ARCHITECTURE behav_dxn_ng OF buck_vhdlams_dxn_ng IS
  QUANTITY dlpd2 : REAL;
  QUANTITY den : REAL;
  QUANTITY x : REAL;
  QUANTITY d : REAL;
  QUANTITY d2 : REAL;
  QUANTITY d3 : REAL;

  QUANTITY dl ACROSS dlp TO dlm;
  QUANTITY vload ACROSS vloadp TO vloadm;
  QUANTITY vgp ACROSS vgp TO vgm;
  QUANTITY eoutp ACROSS eoutp TO eoutm;
  QUANTITY mode ACROSS i THROUGH vmode TO ELECTRICAL_REF;
  QUANTITY iin THROUGH vgp TO vgm;
  QUANTITY il THROUGH eoutm TO eoutp;

BEGIN

  x == (2.0*il*Fs*L/(den*d+1.0E-6));
  den == vgp - vload;
  d2 == dlpd2 - d;
  d3 == 1.0 - dlpd2;
  mode == x;

PROCEDURAL IS
  BEGIN
    IF (dl < dutymin) THEN
      d := dutymin;
    ELSIF (dl > dutymax) THEN
      d := dutymax;
    ELSE
      d := dl;
    END IF;

    IF (x >= 1.0) THEN
      dlpd2 := 1.0;
    ELSIF (x <= d) THEN
      dlpd2 := d;
    ELSE
      dlpd2 := x;
    END IF;

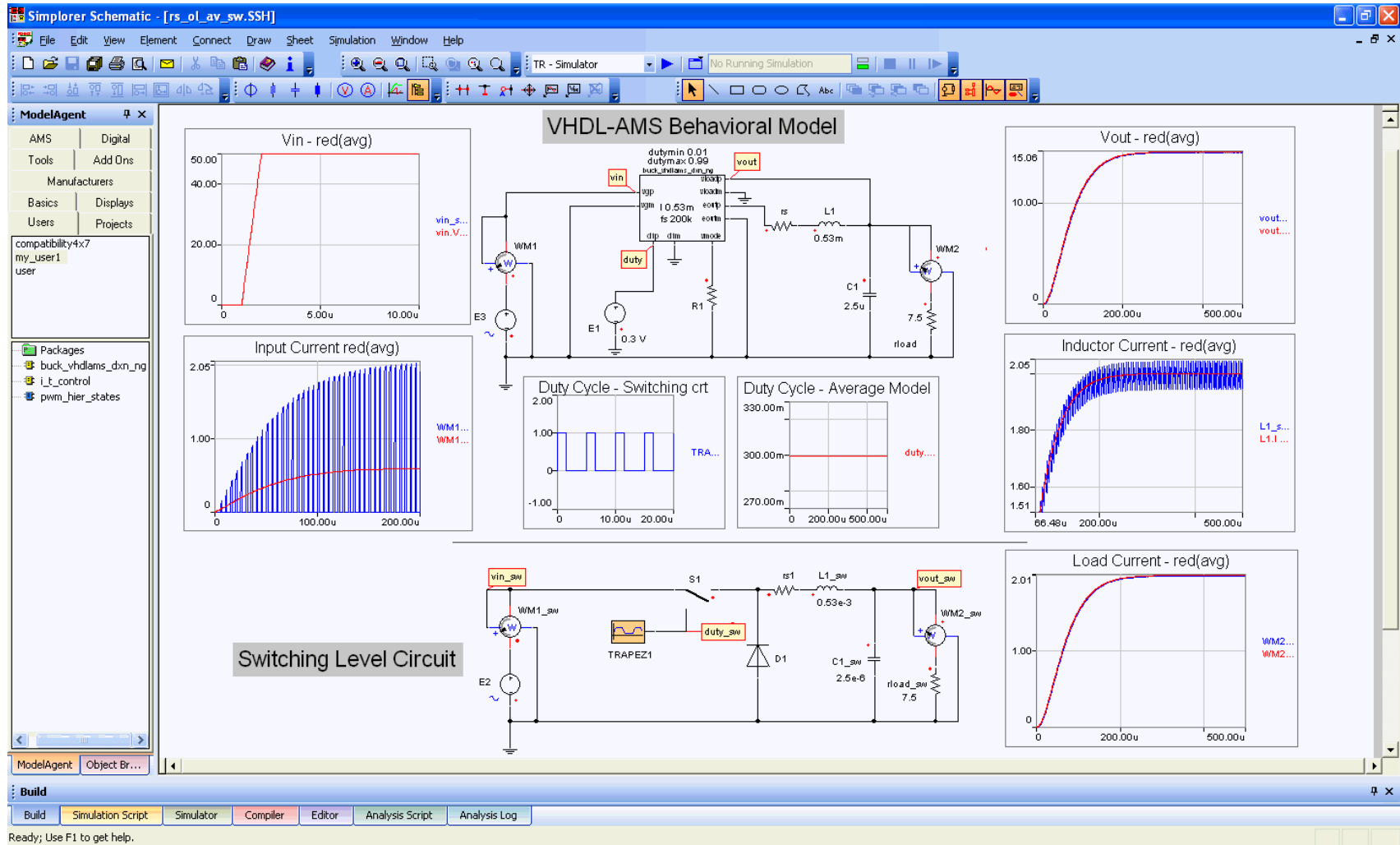
  END PROCEDURAL;

  eoutpm == vload*(1.0 - dlpd2) + vgp*d;
  iin == il*d/dlpd2;

END ARCHITECTURE behav_dxn_ng;

```

Comparison of Behavioral Model and Switching Circuit (open loop)



Using VHDL-AMS Behavioral Model for Feedback Design

Simplorer Schematic - [rs_cl_avg.SSH]

File Edit View Element Connect Draw Sheet Simulation Window Help

AC - Simulator No Running Simulation

ModelAgent

- AMS Digital
- Tools Add Ons
- Manufacturers
- Basics Displays
- Users Projects

compatibility4x7
my_user1
user

Packages

- buck_vhdlams_dxn_ng
- i_t_control
- pwm_hier_states

Control to Output Response

Gain

Phase

Opamp Response

Gain

Phase

Feedback Design Response

Gain

Phase

Loop Response

Gain

Phase

Note additional -180 phase shift from inverting opamp

Build

Build Simulation Script Simulator Compiler Editor Analysis Script Analysis Log

Ready; Use F1 to get help.

Modulation Block Using State Flow Modeling

The screenshot displays the Simpler Schematic interface for a project named [rs_pwm_mod_state2.ssh]. The main workspace is divided into several sections:

- ModelAgent:** A sidebar on the left containing a tree view of components and states. The 'States' section is expanded, showing states like State 01, State 10, State 11, State 33, and State flexible.
- PWM logic state model:** A central diagram showing state flow blocks. It includes a 'clock pulse turns switch on' block with a condition $volk.V \geq \text{threshold}$ and a 'comparator state model' block with conditions $v_{ramp}.V \geq v_{c1}.V$ and $v_{ramp}.V < v_{c1}.V$. Transitions are labeled TRANS1, TRANS2, TRANS11, and TRANS21. Parameters include $\text{maxlim} \# \text{maxtime}$ and Set a time delay .
- Circuit Schematic:** A detailed circuit diagram on the right showing the implementation of the PWM logic. It includes an input vin_ctrl , a switch $VM1_sw$, a diode $D1$, an inductor $L1_sw$ (0.53m), a capacitor $C2$ (2.5u), a load $rload_sw$, and an op-amp $OPV51$ configured as a comparator. The op-amp's non-inverting input is connected to a voltage divider (resistors $r11$, $r21$, $r31$) and its inverting input is connected to a ramp generator (resistors $r41$, $r11$, $r31$). The output of the op-amp is labeled vc_sw .
- Waveforms:** Two plots at the bottom. The left plot, 'Switch Drive Including limiting', shows a square wave signal $vm_ctrl.V$ (blue) and a ramp signal $v_{ramp}.V$ (red) over time from 189.29u to 360.52u. The right plot, 'Comparator output', shows a square wave signal $s_compout$ (black) and a ramp signal $v_{ramp}.V$ (red) over time from 177.24u to 279.85u.

At the bottom of the window, there is a 'Build' menu with options: Build, Simulation Script, Simulator, Compiler, Editor, Analysis Script, and Analysis Log. The status bar at the very bottom reads 'Ready; Use F1 to get help.'

Modulation logic Modeling
Using State flow Blocks

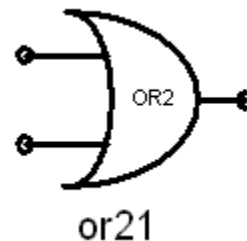
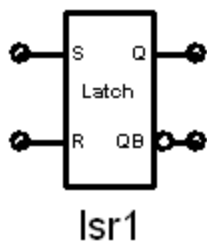
VHDL-AMS Digital Modeling

```

ENTITY lsr IS
  GENERIC (T_PROP : REAL := 0.0);
  PORT (S : IN BIT := '0';
        R : IN BIT := '0';
        Q : OUT BIT := '0';
        QB : OUT BIT := '1');
END ENTITY lsr;

ARCHITECTURE behav of lsr IS
  CONSTANT del : TIME := T_PROP * 1 sec;
BEGIN
  PROCESS (S,R)
  BEGIN
    IF (S='1') THEN
      Q <= '1' AFTER del;
      QB <= '0' AFTER del;
    ELSIF (R = '1') THEN
      Q <= '0' AFTER del;
      QB <= '1' AFTER del;
    END IF;
  END PROCESS;
END behav;

```

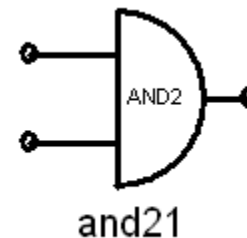


```

ENTITY or2 IS
  GENERIC (TP_LH : REAL := 0.0;
          TP_HL : REAL := 0.0);
  PORT (x1,x2 : IN BIT := '0';
        y : OUT BIT := '0');
END ENTITY or2;

ARCHITECTURE behav OF or2 IS
  CONSTANT lh : TIME := TP_LH * 1 sec;
  CONSTANT hl : TIME := TP_HL * 1 sec;
  SIGNAL OR_OUT : BIT;
BEGIN
  OR_OUT <= x1 OR x2 ;
  y <= OR_OUT AFTER hl WHEN OR_OUT = '0' ELSE
      OR_OUT AFTER lh;
END behav;

```



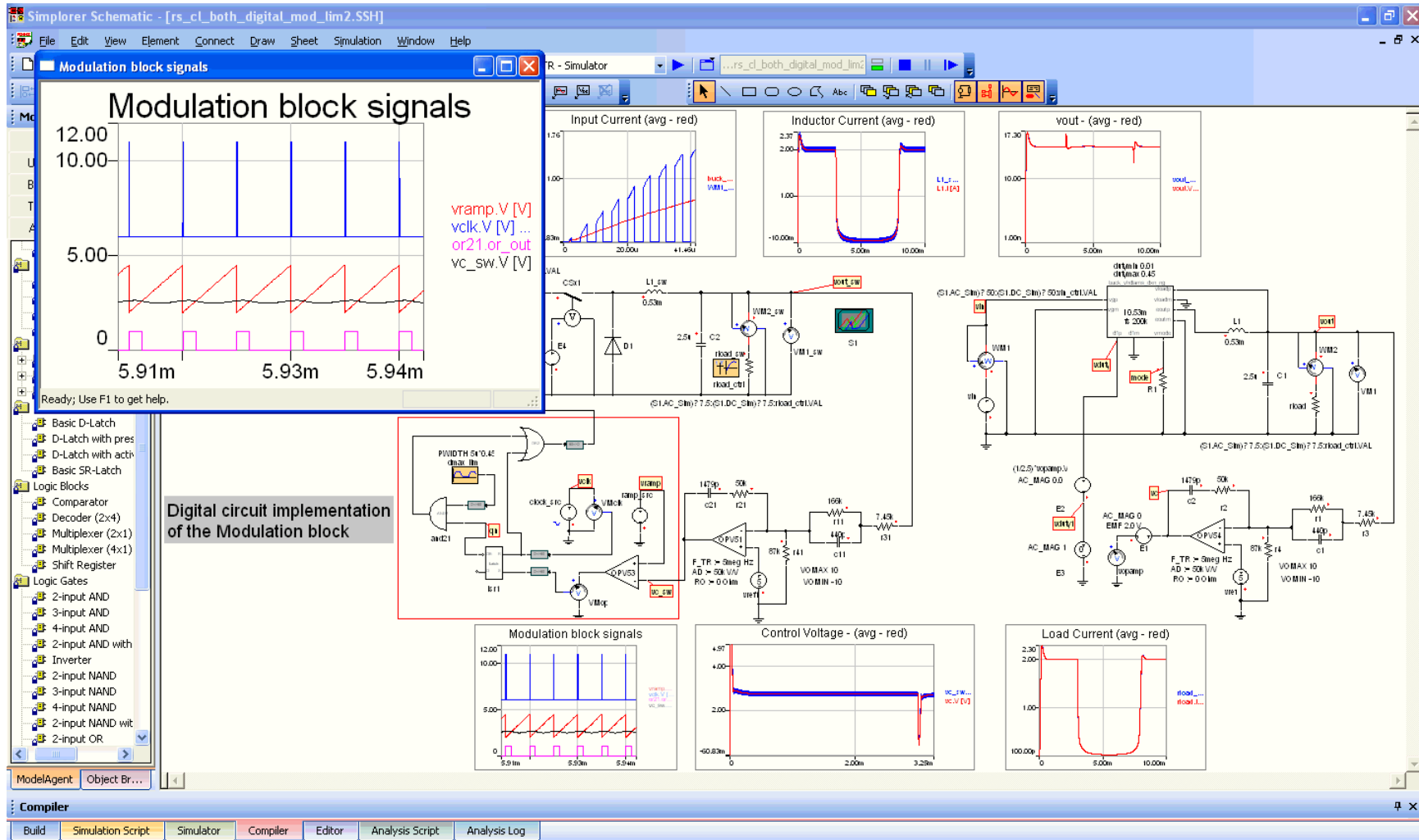
```

ENTITY and2 IS
  GENERIC (TP_LH : REAL := 0.0;
          TP_HL : REAL := 0.0);
  PORT (x1,x2 : IN BIT := '0';
        y : OUT BIT := '0');
END ENTITY and2;

ARCHITECTURE behav OF and2 IS
  CONSTANT lh : TIME := TP_LH * 1 sec;
  CONSTANT hl : TIME := TP_HL * 1 sec;
  SIGNAL AND_OUT : BIT;
BEGIN
  AND_OUT <= x1 AND x2 ;
  Y <= AND_OUT AFTER hl WHEN AND_OUT = '0' ELSE
      AND_OUT AFTER lh;
END behav;

```

Digital VHDL-AMS Modeling for the Modulation Block

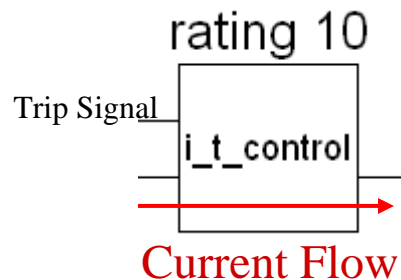


System Modeling (SSPC, RPC, Fuse, Circuit Breakers – control)

```

----- ENTITY DECLARATION i_t_control -----
ENTITY i_t_control IS
    GENERIC (
        rating : real := 1.0
    );
    PORT (
        TERMINAL vinp : electrical;
        TERMINAL voutm : electrical;
        TERMINAL trip : electrical
    );
END ENTITY i_t_control;

```



```

----- ARCHITECTURE DECLARATION arch_i_t_control -----
ARCHITECTURE arch_i_t_control OF i_t_control IS
    CONSTANT res : real := 1.0e-3; --rating**(-1.5858) * 1.1143;
    CONSTANT thres_start : real := rating*1.2;

    QUANTITY vacross ACROSS ithrough THROUGH vinp TO voutm;
    QUANTITY vtrip ACROSS itrip THROUGH trip TO electrical_ref;

    QUANTITY icurve : real;
    QUANTITY vcharge : real;

BEGIN
    ithrough == vacross/res;

    IF (ithrough >= thres_start) USE icurve == 1.0/(726.3 * (ithrough/rating)**(-4.7306));
    ELSE icurve == 0.0;
    END USE;

    vcharge == (1.0)*icurve'INTEG;

    IF vcharge'ABOVE(1.0) USE vtrip == -5.0;
    ELSE vtrip == 5.0;
    END USE;

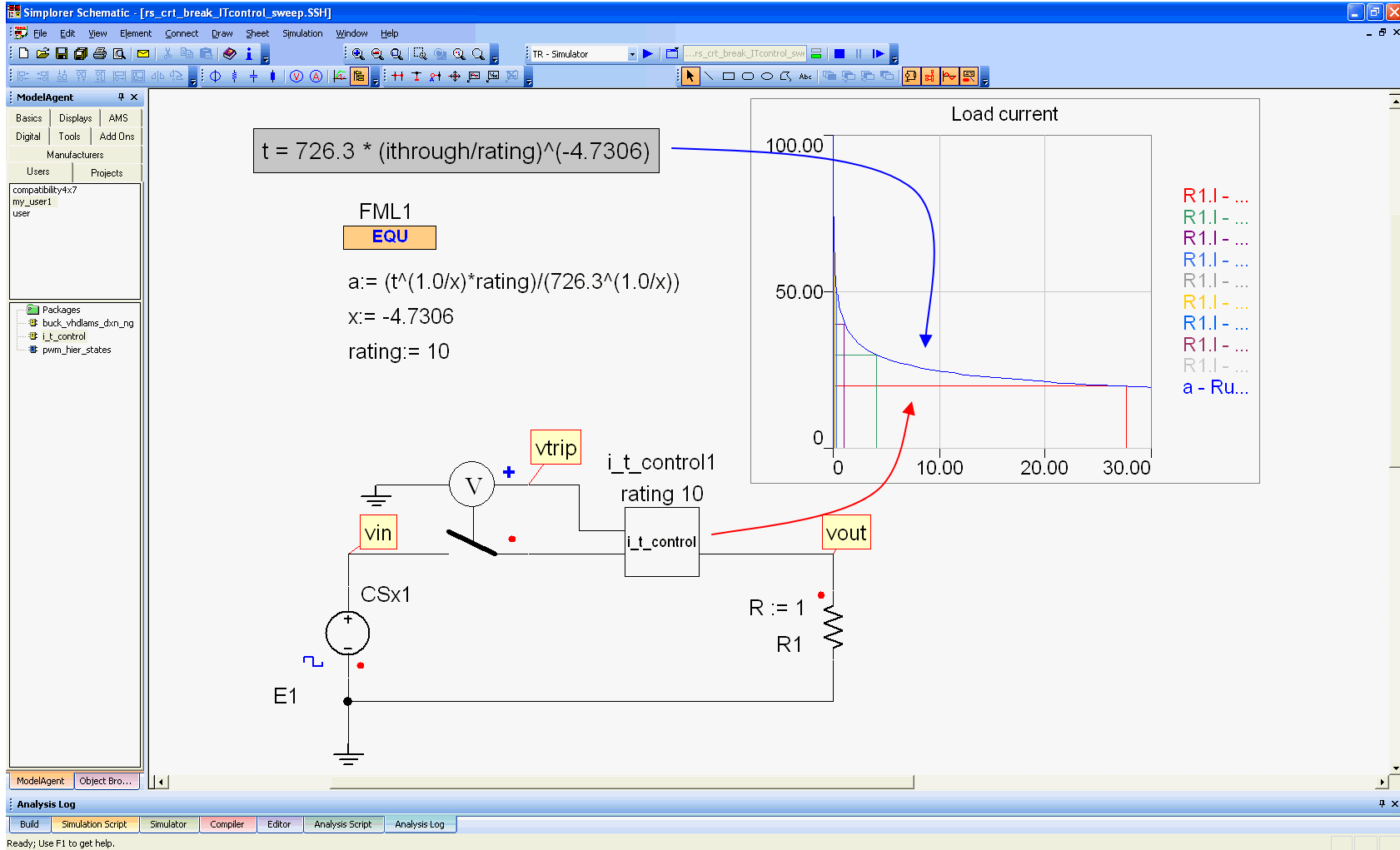
END ARCHITECTURE arch_i_t_control;

```

Main Relationship
 For Trip Characteristic Timing



Simulation Validation of the VHDL-AMS Trip Control Model



Conclusions

- ▶ VHDL-AMS is a very powerful Modeling language for not only mixed signal (Analog/Digital) but also for mixed technology (Electrical, Mechanical, Thermal, Magnetic, etc.) type designs.
- ▶ VHDL-AMS also allow the users to model based on characteristic equations of a device and not macro-modeling (emulation of equations using dependent sources, and other electrical components) such as done in other type tools.
- ▶ VHDL-AMS is an IEEE standard and eliminates the problems associated with other non-standard or proprietary modeling languages.
 - ▶ Don't have to learn multiple languages
 - ▶ Models will port to other engineering groups, or sub contractors that may be using other vendors tools
 - ▶ Gives Companies a wider selection of vendor tools without the loss of IP

