Verilog-AMS
- Verilog-AMS: analog and mixed signal extensions to Verilog
  - On track to become an IEEE Standard
- An earlier language standard was called Verilog-A (Verilog with Analog extensions)
  - Verilog-A is a subset of Verilog-AMS
- Important extensions of Verilog-AMS over Verilog-A
  - Both digital and analog signals can be included in same module
  - User-defined conversions modules are automatically inserted in netlist if analog signal connected to digital signal or vice-versa
  - More freedom in accessing digital/analog signals within a module
- These slides will look at some example models and point out important features
  - http://www.eda.org/verilog-ams
- Much of the same terminology used in VHDL-AMS

Sample Model: Voltage Integrator

```vhdl
module V_integrator(in,out);
  input in;
  output out;
  voltage in,out;
  // integration coefficient
  parameter real ki=1.0 exclude 0;
  parameter real dcval = 0;
  real k1;
  initial k1 = 1/ki;
  analog
    V(out) <+ k1*idt(V(in),dcval);
  endmodule
```

Natures and Disciplines

- A nature is a collection of attributes
  - Attributes characterize quantities solved for during simulation

```vhdl
nature Mycurrent
  units = "A";
  access = I;
  idt_nature = charge;
  abstol = 1e-12;
  huge = 1e6;
  endnature
```

- Example nature, attributes predefined by Cadence (see Chap 4)
- Name of the access function for this nature
- Nature to apply when idt (time integral) or idt_mod is applied
- Maximum allowed change in timestep
- Tolerance for convergence

Some predefined Natures

```vhdl
nature Current
  units = "A";
  access = I;
  idt_nature = Charge;
  endnature

nature Voltage
  units = "V";
  access = V;
  idt_nature = Flux;
  endnature
```

- Defined in “discipline.h” include file
  (tools/dflI/samples/spectreHDL/include, or Appendix C)
- Many others (Maneto_Motive_Force, Temperature, Power, Position, Acceleration)

Disciplines

- Disciplines used to bind natures with potential and flow
- Discipline with single nature called signal-flow discipline
- Discipline with multiple natures called conservative discipline. Nature bound to potential must be different from nature bound to flow.
- Can also have an empty discipline (a wire is an empty discipline)

Discipline Compatibility

Operations between nodes of different disciplines allowed if potential/flow natures are the same.

```
Discipline Compatibility
```

- From this flowchart, can connect electrical and voltage disciplines, electrical and current disciplines, and voltage and current disciplines.

Branches

- Branch is a path between nodes
- Can only be declared within a module
- Currents summed at branch node

```
Branches
```

- Define two different branch currents
Branches in Diode Model

```verilog
module diode (a, c);
  electrical a, c;
  branch (a, c) diode, cap;
  branch (a, a) anode;
  parameter real rs = 0, is=1e-14, tf=0, cjo=0, imax=1, phi=0.7;
  analog begin
    I(diode) <+ is*(limexp((V(diode)-rs*I(anode)/$vt) – 1));
    I(cap) <+ sqrt(tf*I(diode) - 2 * cjo * sqrt(phi * (phi * V(cap)))) ;
    if (I(anode) > imax) // Checks current through port
      $strobe(“Warning: diode is melting!”);
  endmodule
```

Special branch (port branch), used to monitor current through port

Branch currents summed

Thermal voltage, language builtin

Special Variables

<table>
<thead>
<tr>
<th>Environment Function</th>
<th>Description of Returned Value</th>
<th>Type of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$realtime</td>
<td>Current simulation time in seconds</td>
<td>Real</td>
</tr>
<tr>
<td>$temperature</td>
<td>Ambient temperature in degrees Kelvin</td>
<td>Real</td>
</tr>
<tr>
<td>$svt</td>
<td>Thermal voltage (kT/qT) at current simulation temperature</td>
<td>Real</td>
</tr>
<tr>
<td>$svt(temp)</td>
<td>Thermal voltage at temperature (temp, specified in degrees Kelvin)</td>
<td>Real</td>
</tr>
</tbody>
</table>

Assignments

- Procedural assignments, used to modify integer, reals
  - `sum = a + b`
- Branch contribution statement
  - `V(n1, n2) <+ expr1;`
- Multiple branch assignments can be applied to same node
  - `V(n1, n2) <+ expr1;`
  - `V(n1, n2) <+ expr2;`
  - is equivalent to:
    - `V(n1, n2) <+ expr1 + expr2;`

More on branch assignments

- Simulation of a branch assignment
  - Simulator evaluates right hand expression
  - Simulator adds the value of the right hand expression to any previously retained value for the node (a summation)
  - At end of simulation cycle, summed value assigned to source branch
- If assigning a flow quantity, and previously assigned value was a potential, then potential value is discarded (and vice-versa)
  - When a branch changes between assigned flow and potential quantities, this is known as a switch branch.

Control Structures

- Begin/end blocks
- If/else
- Case
- Repeat Loop (loop fixed number of times)
- While/Loop (conditional loop)
- For/Loop
- Generate (similar to generate capability in VHDL but not quite as powerful)

Details of these constructs in Verilog-A ref manual, chapter 5.

Conservative/Signal-Flow Interface

To model a mixed conservative/signal-flow system, must have an interface model between the two since terminals of the two have incompatible disciplines.
Many constants defined in constants.h (appendix C in verilog reference manual).

`define M_E 2.7182818284590452354
`define M_LOG2E 1.4426950408889634074
`define M_LOG10E 0.43429448190325182765
`define M_LN2 0.69314718055994530942
`define M_LN10 2.30258509299404568402
`define M_PI 3.14159265358979323846
`define M_TWO_PI 6.28318530717958647652
`define M_PI_2 1.57079632679489661923

etc... Refer to them in code via: `M_LN2

Note the backquote in front of the constant name.

Module Example: Capacitor

module cap(p,n);
inout p,n;
electrical p,n;
parameter real c=0 from [0:inf);
analog I(p,n) <+ c*ddt(V(p,n));
endmodule;

Electrical discipline, bidirectional terminals.

Time derivative operator, implements dv/dt
Predefined as part of the language.

Model Example: Sine Wave Generator

module V_sine_generator(out);
output out;
voltage out;
parameter real freq = 1K from (0:inf),
ampl = 1,
offset = 0;
analog
begin
V(out) <+ ampl * sin(`M_TWO_PI * freq * $realtime) + offset;
end
endmodule;

Specifies maximum time between allowed between adjacent points in simulation. Forces simulation tracking of signals to accuracy required by model for correct operation.

Analog Operators

• Built-in functions that operate on more than just the current value of their arguments – they maintain internal state
  – Limited Exponential function ($limexp)
  – Time derivative operator (ddt)
  – Time Integral operator ($idt)
  – Circular integrator operator ($idmod)
  – Delay operator (delay)
  – Transition filter ($transition)
  – Slew filter ($slew)
  – Laplace transform filters ($laplace_zp, $laplace_zd, $laplace_np, $laplace_nd)
  – Z-transform filters ($zi_zp, $zi_zd, $zi_np, $zi_nd)

Miscellaneous Functions

• $strobe (aka $display) – formatted output statement
• $pwr – specify model power consumption
• File IO
  – $open, $fclose
  – $cref, $sdisplay
• $finish – simulator exit
• $stop – simulator exit
• Can also have user-defined local functions within a module

Mechanical Model: Friction

module damper1d(n1,n2);
module spring1d(n1,n2);
inout n1,n2;
kinematic n1,n2;
parameter real k = 10 from [0:inf);
// spring constant given in N/m
parameter real l = 0.1 from [0:inf);
// length of spring in m
analog
F(n1,n2) <+ k*(Pos(n1,n2) - l);
endmodule

Recall that a mechanical spring is akin to a resistor in the electrical world. Spring constant is equivalent to conductance.