System-on-Chip

3- Analog and Mixed-Signal Modelling
- From schematic to verilog
- Analog signal data (real) to/from 64bits
- Conventions
- Single bit wrapper
3- Analog and Mixed Signal Modelling

- Analog and mixed signal circuits are validated using transistor level simulation (ex: hspice or spectre):
  - At block level, as exhaustively as possible, including the load and the real supply operating conditions
  - At system level for a very limited number of use case scenarios, given the high computational effort required for system level simulations at transistor level
- Analog and mixed signal behavior is \textbf{modeled in Verilog} with the purpose to, validate the system in a wide range of use cases, using a digital Verilog simulator
3- Analog and Mixed Signal Modelling

• to design state machines that are synthesized (implemented with logic gates) automatically

- Block design
- RTL Simulation
- Digital synthesis (&DFT)
- Logic Simulation
- Back-end

• to model blocks with schematics that were design “by hand”

- Block Modelling
- Block Simulation
- System Simulation

• to validate the system

Chap. 2
Chap. 4
Chap. 6

System-on-Chip
Marcelino Santos, IST Universidade de Lisboa
3- Analog and Mixed Signal Modelling

• From schematic to verilog
  • Analog signal data (real) to/from 64bits
  • Conventions
  • Sigle bit wrapper
3- Analog and Mixed Signal Modelling

From schematic to verilog

- Digital and mixed circuits, designed with virtuoso schematic editor, can be automatically exported to verilog format with useful information on:
  - Hierarchy and Interconnections
- Verilog of pure digital circuits is ready to simulate in verilog (given the cells verilog definition)
- Verilog of circuits containing analog and mixed signal cells need to be edited in order to model in verilog the functionality implemented at transistor level

In the next topic: Analog signal data (real) to/from 64bits
Verilog Netlist Extraction with virtuoso Lab Tutorial
Verilog Netlist Extraction with virtuoso Lab Tutorial

System-on-Chip

Marcelino Santos, IST Universidade de Lisboa

Circuit exemple:

![Circuit Diagram]
Verilog Netlist Extraction with virtuoso Lab Tutorial

Select:
- Launch
- Simulation
- NC-Verilog

System-on-Chip
Marcelino Santos, IST Universidade de Lisboa
Verilog Netlist Extraction with virtuoso Lab Tutorial

In the new window, select:  
- Initialize Design
Verilog Netlist Extraction with virtuoso Lab Tutorial

In order to be able to setup the netlist options, select:
- Setup
  - Netlist
Verilog Netlist Extraction with virtuoso Lab Tutorial

Here you find some useful netlist options
It is now possible to generate the netlist. Select:
- Commands
  - Generate Netlist
Verilog Netlist Extraction with virtuoso Lab Tutorial

And check the echo in the command interpreter window.

<table>
<thead>
<tr>
<th>CELL_NAME</th>
<th>VIEW_NAME</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>gmex33</td>
<td>symbol</td>
<td><em>Stopping View</em></td>
</tr>
<tr>
<td>gmex33</td>
<td>symbol</td>
<td><em>Stopping View</em></td>
</tr>
<tr>
<td>RATCHARGERdigexample</td>
<td>schematic</td>
<td></td>
</tr>
<tr>
<td>COMMONLIBsxfflv</td>
<td>schematic</td>
<td></td>
</tr>
<tr>
<td>COMMONLIBsinvlv</td>
<td>schematic</td>
<td></td>
</tr>
<tr>
<td>COMMONLIBand2lv</td>
<td>schematic</td>
<td></td>
</tr>
<tr>
<td>COMMONLIBor2lv</td>
<td>schematic</td>
<td></td>
</tr>
</tbody>
</table>

--------- End of netlist configuration information  ---------

INFO (VL:VNET-10): The library 'RATCHARGER', cell 'RATCHARGERdigexample', and view 'schematic' has been netlisted successfully.

End netlisting NOW 12 16:08:24 2021
Verilog Netlist Extraction with virtuoso Lab Tutorial

The netlist can now be displayed and saved. Select:
- Results
- Netlist...
Select the library and cell name in the new window and click View
Verilog Netlist Extraction with virtuoso Lab Tutorial

The netlist can be now saved with:
- File
  - Save

Marcelino Santos, IST Universidade de Lisboa
3- Analog and Mixed Signal Modelling

- From schematic to verilog
- Analog signal data (real) to/from 64bits
- Conventions
- Sigle bit wrapper
3- Analog and Mixed Signal Modelling

Analog signal data (real) to/from 64bits

• Analog signals fall in three categories:
  1. Irrelevant – analog signals that are internal to a module and that do not control any signal in the digital system representation
  2. With digital behaviour – can be modelled as 0 or 1, Ex: a voltage or current reference is presente or it is not present. It’s value is validated at transistor level. In verilog, the relevant information is if it is ok or nor ok.
  3. With analog behaviour – Ex: input or output voltage of a voltage regulator
3- Analog and Mixed Signal Modelling

Analog signal data (real) to/from 64bits

• Analog signals fall in three categories:
  1. Irrelevant – analog signals that are internal to a module and that do not control any signal in the digital system representation - ignore
  2. With digital behaviour – can be modelled as 0 or 1, Ex: a voltage or current reference is presente or it is not present. It’s value is validated at transistor level. In verilog, the relevant information is if it is ok or nor ok. – model as 0 or 1 according to behaviour
  3. With analog behaviour – Ex: input or output voltage of a voltage regulator – model as 64 bit usign $realtobits and $bitstoreal

System-on-Chip
Marcelino Santos, IST Universidade de Lisboa
3- Analog and Mixed Signal Modelling

Analog signal data (real) to 64bits

inout [63:0] avdd, // analog supply
inout [63:0] dvdd, // digital supply
....
real rl_agnd;
real rl_dgnd;
....
parameter AVDDVAL = 1.8;
parameter DVDDVAL = 1.2;
....
initial assign rl_avdd = supply_ok ? AVDDVAL : 0.0;
nitial assign rl_dvdd = supply_ok ? DVDDVAL : 0.0;
....
assign avdd = $realtobits (rl_avdd);
assign dvdd = $realtobits (rl_dvdd);
inout [63:0] vin, // supply pin

....
real rl_vin;

....
parameter VINMIN = 2.5;
parameter VINMAX = 5.5;

....
assign vin_ok = (rl_vin < VINMAX) && (rl_vin > VINMIN) ? 1'b1 : 1'b0;

....
initial assign rl_vin = $bitstoreal (vin);
3- Analog and Mixed Signal Modelling

• From schematic to verilog
• Analog signal data (real) to/from 64bits
• Conventions
• Sigle bit wrapper
Analog and Mixed Signal Modelling

Conventions

Analog signals that require verilog “analog” representation are represented with 64 bits: `signame[63:0]`. Each one has also an associated real signal: `rl_siganme`

```verilog
// initial assign rl_var1    = $bitstoreal (var1_64b);
// assign var2_64b           = $realtobits (rl_var2);
assign avdd = $realtobits (rl_avdd);
assign dvdd = $realtobits (rl_dvdd);
assign dgnd = $realtobits (rl_dgnd);
initial assign rl_vin = $bitstoreal (vin);
initial assign rl_agnd = $bitstoreal (agnd);
```

real rl_vin;
real rl_avdd;
real rl_dvdd;
real rl_agnd;
real rl_dgnd;

64 bits value obtained from the real value forced

real value obtained from the 64bits forced
3- Analog and Mixed Signal Modelling

Resistive divider example

```verilog
module RESDIV_64b;
  input [63:0] vfb, // input of ideal resistive divider (no parasitic capacitance)
  output [63:0] vfbdiv, // output of resistive divider with no current
  input [63:0] agnd;

  real rl_vfb;
  real rl_vfbdiv;
  real rl_agnd;

  parameter R1 = 1.0e6;
  parameter R2 = 9.0e6;

  initial assign rl_vfbdiv = R1 * (rl_vfb - rl_agnd) / (R1 + R2);

  //-- Signal conversion ---------------------

  initial assign rl_vfb = $bitstoreal (vfb);
  initial assign rl_agnd = $bitstoreal (agnd);
  assign vfbdiv = $realtobits (rl_vfbdiv);

endmodule // RESDIV_64b
```

```verilog
module RESDIV_64b_tb;

  real rl_vfb;
  real rl_vfbdiv;
  real rl_agnd;
  wire [63:0] vfb; // input of ideal resistive divider (no parasitic capacitance)
  wire [63:0] vfbdiv; // output of resistive divider with no current
  wire [63:0] agnd;

  RESDIV_64b uut{
    .vfb (vfb), // input of ideal resistive divider (no parasitic capacitance)
    .vfbdiv (vfbdiv), // output of resistive divider with no current
    .agnd (agnd)
  };

  initial
  begin
    rl_agnd = 0.0;
    rl_vfb = 0.0;
    #10 rl_vfb = 1.0;
    #10 rl_vfb = 4.0;
    #10 rl_vfb = 2.0;
    #10 $finish;
  end

  //-- Signal conversion ---------------------

  // output to convert
  initial assign rl_vfbdiv = $bitstoreal (vfbdiv);

  // input forced
  assign agnd = $realtobits (rl_agnd);
  assign vfb = $realtobits (rl_vfb);

endmodule
```
3- Analog and Mixed Signal Modelling

• From schematic to verilog
• Analog signal data (real) to/from 64bits
• Conventions
• Sigle bit wrapper
Sigle bit wrapper

Real values (64 bit) are mandatory to model analog behaviour:
- Block validation including analog detail
- High speed system simulation with analog information
- Can not be used for system integration

Sigle bit model is required for system integration:
- Verilog to schematic
- Verilog to backend (verilog and LEF pins must be the same)

- Solution: perform integration with 64bit model inside a single bit wrapper
3- Analog and Mixed Signal Modelling

Sigle bit wrapper

Single bit wrapper for 64 bit model requires:
- Single bit D/A at each 64bit module inputs
- Single bit A/D at each 64bit module output

Requires default values for each pin: real value to be forced if input is 0 and real value to be forced if input is 1

Requires threshold to decide output binary value as a function of the 64bit variable real value
3- Analog and Mixed Signal Modelling

Sigle bit wrapper: example

```verilog
module RESDIV(
    input vfb,       // input of ideal resistive divider (no parasitic capacitance)
    output vfbdiv,   // output of resistive divider with no current
    inout  agnd     
);

real         rl_vfb;
real         rl_vfbdiv;
real         rl_agnd;
wire [63:0]   vfb_64b;
wire [63:0]   vfbdiv_64b;
wire [63:0]   agnd_64b;

parameter Vfb_high= 1.8; // rl_vfb value when vfb=1
parameter Vfb_low= 0.0;  // rl_vfb value when vfb=0
parameter Vfbdivthreshold = 0.16; // single bit A/D threshold

RESDIV_64b resdiv (.vfb(vfb_64b), .vfbdiv(vfbdiv_64b), .agnd(agnd_64b)); //64 bits instance

initial assign rl_agnd = 0.0;

initial assign rl_vfb = (vfb == 1'b1) ? Vfb_high : Vfb_low; //rl_vfb = 1 bit D/A conversion of input vfb
assign vfbdiv = (rl_vfbdiv > Vfbdivthreshold) ? 1'b1 : 1'b0; //output vfbdiv = 1 bit A/D conversion
```

---

System-on-Chip  Marcelino Santos, IST Universidade de Lisboa
Sigle bit wrapper: example

module RESDIV_tb;

reg vfb; // input of ideal resistive divider (no parasitic capacitance)
wire vfbdiv; // output of resistive divider with no current
wire agnd;

RESDIV uut(
    .vfb (vfb), // input of ideal resistive divider (no parasitic capacitance)
    .vfbdiv (vfbdiv), // output of resistive divider with no current
    .agnd (agnd)
);

assign agnd = 1'b0;

initial begin
    vfb = 0;
    #10 vfb = 1'b1;
    #10 vfb = 1'b0;
    #10 $finish;
end
endmodule
3- Analog and Mixed Signal Modelling

Sigle bit wrapper: example

- Internal real values based on default parameters
- Single bit values at the pins of the wrapper
3- Analog and Mixed Signal Modelling

• to design state machines that are synthesized (implemented with logic gates) automatically

- Block design
  - RTL Simulation
    - Digital synthesis (&DFT)
      - Logic Simulation
        - Back-end
          - Chap. 6

• to model blocks with schematics that were design “by hand”

- Block Modelling
  - Block Simulation
    - System Simulation
      - Chap. 3

• to validate the system

- System-on-Chip
  - Marcelino Santos, IST Universidade de Lisboa