Cadence® AMS Tutorial

Dr. George L. Engel

November 2016

This document is intended to be a brief tutorial on how to use the Cadence® AMS (Advanced Mixed-Signal) analyzer to simulate a digital-to-analog converter (a high-level behavioral model). The testbench is composed of a Verilog module (4-bit counter) and a VerilogA module (ideal 4-bit DAC). The DAC output is a full-scale sawtooth waveform.

You should do the following:

Use a vnc client (like "tightvnc") to connect to "vlsi" server. Open up a terminal window and type 'cds'. You must first install the AMS libraries before trying to run AMS. Make sure your project directory is set to ece585 and then type

```
setup_ece585_ams
```

You should then launch the ic tools by typing icd

1.) From the CIW window start the Library Manager. Create a new Verilog cell view in your Lib library called *dac_counter*. Use the code provided. Create a symbol for the module.

2.) Create a VerilogA module called *ideal_dac_4bits*. Use the code provided. Create a symbol for the module.

3.) Create a testbench schematic in the LibTest library called *ideal_dac_AMS_tb*. Wire the Verilog counter module to the VerilogA DAC as shown in the schematic distributed as part of this tutorial.

Actually steps 1-3 are for completeness. You really don't have to type in the code or the testbench. My "setup_ece585_ams" script actually copied those things into your account but you may delete them and re-type them in if you wish.

You will now need to create a config view for the *ideal_dac_AMS_tb* schematic. Go back to the Library Manager and create a cell view for the Hierarchy Editor tool (i.e. a config view). This will launch the Hierarchy Editor. You should tell the Hierarchy Editor that you are creating a config view for the *ideal_dac_AMS_tb* schematic. Click on choose template and select the AMS_spectre template.

Delete mylib from the Lib list and instead type in

```
Lib LibTest
```

Update and Save the config view. We are almost ready to simulate but we have to bring up the AMS menu. Over on the right of the Hierarchy Editor window you will see PlugIns. Pull this window down and select AMS. You should see another main menu item called AMS suddenly appear. Pull down on the AMS menu item and select Setup Run Directory. You should choose the AMSruns directory within your project directory for saving your results. Click on the button that will make sure that this directory is always used for saving results of this simulation.

Return and select the AMS menu again. This time select Plot and choose to save and plot everything. You should return once again to the AMS menu and this time select Run Simulation. Enter how long you would like to run the simulation. I suggest 1m (a milli-second). The tool will do a design prep (compile and elaborate all of the Verilog and VerilogA modules) and then will launch SimVision. In SimVision, hit the run button. Wait for a few seconds and you should have your results. Use SimVision for viewing your results.

If you get a message saying that AMS cannot be run because a simulation is already running then **exit** the cadence tools and in your project directory type the command

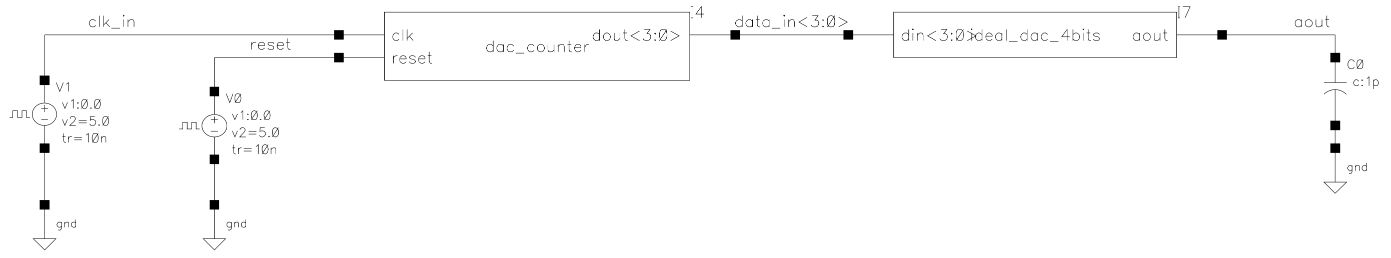
```
rm_ams
```

This script will remove the any orphaned AMS lock files. Re-start cadence by typing

```
icd
```

and everything should be fixed.

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



North Carolina State University

UPDATED	Nov 10 10:50:39 2011
BY	gle
SIZE	A
REV	
SHEET	1 of 1

```
//Verilog HDL for "Lib", "dac_counter" "verilog"

`define   BITS    4

module dac_counter (reset, clk, dout);
   input     reset, clk ;
   output    [`BITS-1:0] dout ;

   reg       [`BITS-1:0] dout ;
   always @(posedge clk or posedge reset) begin
      if(reset) dout <= 0 ;
      else dout <= dout + 1 ;
   end

endmodule
```

```
//
//
// VerilogA description of an ideal DAC (4 bits)
// Unipolar
//
//

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

`define  BITS   4

module ideal_dac_4bits(din, aout);

   input       [`BITS-1:0] din ;
   output      aout;

   electrical  [`BITS-1:0] din ;
   electrical  aout ;

   parameter real    vth = 2.5 ;
   parameter real    tpd = 0.0 from [0:inf) ;
   parameter real    tr = 1n from [0:inf) ;
   parameter real    tf = 1n from [0:inf) ;
   parameter real    Vref = 1.0 ;

   real      Dvalue ;
   real      weight ;

   genvar    j ;

   analog begin
      Dvalue = 0.0 ;
      weight = 0.5 ;
      for (j = `BITS-1 ; j >= 0 ; j = j - 1) begin
         Dvalue = Dvalue + (V(din[j]) > vth ? weight : 0.0) ;
         weight = weight  / 2.0 ;
      end

      V(aout) <+ transition(Vref * Dvalue, tpd, tr, tf) ;
   end

endmodule
```

```
#!/bin/tcsh

# Make sure user has PHOME set to ~/cds/ece585/

#
# Script to install AMS libraries in ~/cds/ece585/
# Copying from SRC
#

# Source

setenv SRC ~gengel/cds/ece585

# Destination

setenv  DST  $HOME/cds/ece585

# Move to the destination directory

cd $DST

# If connectLib exists remove it

if (-e ./connectLib ) then
  echo "connectLib directory exists .. removing"
  rm -rf connectLib
endif

# Make connectLib

echo "Creating connectLib directory"
mkdir connectLib

# If AMSruns exists remove it

if (-e ./AMSruns ) then
  echo "AMSruns directory exists .. removing"
  rm -rf AMSruns
endif

# Make AMSruns directory

echo "Creating AMSruns directory"
mkdir AMSruns


# If worklib exists remove it

if (-e ./worklib) then
  echo "worklib directory exists .. removing"
  rm -rf ./worklib
endif

# Make worklib directory

echo "Creating worklib directory"
mkdir ./worklib

# If a hdl.var or ams.env file exists remove it

if (-e ./ams.env ) then
  echo "ams.env exists in project directory ... removing"
  rm -f ams.env
endif

if (-e ~/ams.env ) then
  echo "ams.env exists in home directory ... removing"
  rm -f ~/ams.env
endif

if (-e ./hdl.var ) then
  echo "hdl.var exists in project directory ... removing"
  rm -f ./hdl.var
endif

if (-e ~/hdl.var ) then
  echo "hdl.var exists in home directory ... removing"
  rm -f ~/hdl.var
endif

# Place ams.env in home directory

echo "Moving ams.env to home directory"
cp $SRC/ams.env ~/

# Place hdl.var file in home directory

echo "Moving hdl.var to home directory"
cp $SRC/hdl.var ~/

# Copy the dac, the counter, and the testbench

cd $DST/Lib

cp -r $SRC/Lib/ideal_dac_4bits ./
echo "copying ideal_dac_4bits ..."

cp -r $SRC/Lib/dac_counter ./
echo "copying dac_counter ..."

# Move to LibTest

cd $DST/LibTest

cp -r $SRC/LibTest/ideal_dac_AMS_tb ./
echo "copying ideal_dac_AMS_tb ..."

# Compile  connectLib

echo "Compiling AMS connection libraries"

~cdsadmin/AMS/AMI05connect/ece585_compile_ams

# Backup cds.lib and copy cds.lib file

cd $DST
mv cds.lib cds.lib.bak
echo "Copying cds.lib file"
cp $SRC/cds.lib ./

# Exit

echo "AMS successfully installed!"
```