- Signed in as Domenico Spina
- UGent CAS logout

UNIVERSITEIT GENT

Zephyr

- My Zephyr
- My profile
- My agenda

My Zephyr > EMWIKI EM Wiki > Wiki > Use ADS with Matlab

EM Wiki

Search Add a new page Recent Changes All Pages Export Wiki Show wiki menu

- Read
- Edit
- Page History

Use ADS with Matlab 2016-06-02 17:04

It is possible to access a design in ADS, to modify and run it, as well as to manipulate the results of a simulation, without actually opening ADS and, namely, in a completely automatic way. The following sections provide all the necessary information to implement those functions in Matlab.

Section 1: How to use ADS from command line

To use ADS in combination with Matlab, one has to set up the system variables to be able to run ADS from command line. Here, we list the necessary steps for a Windows computer, for ADS version 2012.08 and later. Changes and information about other versions can be found at the links supplied in each step, for the Momentum and the circuit simulation, respectively.

Step 1: Setting the system variables

1.a: Momentum

Open up a cmd prompt, and enter the following commands:

```
set HPEESOF_DIR= # for example: C:\Agilent\ADS2012_08

set ADS_LICENSE_FILE= # for example: 27000@license.intec.ugent.be

set PATH=%HPEESOF_DIR%\bin;%PATH%
```

Verify these steps by entering `adsMomWrapper -version`, which should return the version information of the installed Momentum engine.

Information for different version of ADS is given in the following Keysight Technologies knowledge base entry: http://edadocs.software.keysight.com/display/eesofkcads/Run+a+Momentum+simulation+from+command+line.

1.b: Circuit simulation

Open up a command prompt, and enter:

```
set SIMARCH= # for example: win32 on a 32-bit system, win32_64 on a 64-bit system

set HPEESOF_DIR= # off course, if you've performed Step 1 already, you don't need to do this anymore

set
PATH=%HPEESOF_DIR%\bin\%SIMARCH%;%HPEESOF_DIR%\bin;%HPEESOF_DIR%\lib\%SIMARCH%;%HPEESOF_DIR%\circuit\lib.%SIMARCH%;

set ADS_LICENSE_FILE= # off course, if you've performed Step 1 already, you don't need to do this anymore
```

Note that, for ADS2014.01, you should enter the following lines to correctly set the path:

```
set MOSAIC_ARCH=win32_64

set
```

```
PATH=%HPEESOF_DIR%\bin;%HPEESOF_DIR%\lib\%SIMARCH%;%HPEESOF_DIR%\circuit\lib.%SIMARCH%;%HPEESOF_DIR%\adsptolemy\lik
```

Verification of these steps can be performed by entering `hpeesofsim -version`.

Information for different version of ADS is given in the following Keysight Technologies knowledge base entry:
http://edadocs.software.keysight.com/display/eesofkcads/Command+line+Environments+to+run+ADS+or+hpeesofsim.

Step 2: Simulating an ADS design

2.a: Momentum

1. Create the layout and define the associated EM simulation setup in an ADS workspace.
2. From the EM Setup window, select **Generate: Simulation input files** and hit the **Go** button to write the required input files.
   Following files will be written to the /simulation////_ directory:

Input File	**Content**
proj_a	original geometries before layout pre-processing
proj.vpl (optional)	far field view plan, needed for far field computations only
proj.sti	frequency stimulus plan
proj.prt and proj.pin	S-Parameter port and associated pin information
proj.opt	simulation options (e.g. layout pre-processing, mesh and simulation related settings)
proj.ltd	substrate information
proj.cfg	configuration file (e.g. paths to user/site/supplied substrate database locations)

3. Either copy these files to a directory of choice, or work in the default directory. In the command line, or in the command you issue from within Matlab, change directory (`cd`) to the desired one. A complete Momentum simulation consists of different steps.

Simulation Step	**Engine Option**	**Output Files**
Substrate file pre-processing (e.g. expanding substrate for thick conductors)	-T	proj.sub including expanded layers for thick conductors
Substrate database generation (Green function calculation)	-DB	proj.qry that contains information on where the Green function database file is located
S-parameter model generation	-3D	proj.cti, proj.afs, proj.sam, proj.cvi, proj.sta, and others
Layout pre-processing and mesh generation	-M	proj.mmd, proj.mrp, proj.stm, and others
Far Field calculation	-FF	proj.FFe, proj.ant, proj.fff (3D, citi format)

4. You can either invoke each step individually, or when specifying the -O (capital letter O, not the number zero) option, the engine will run all steps from the beginning up to the one specified.
   For example, a typical simulation of an antenna will run all the listed steps, and would be issued as:

```
    adsMomWrapper -O -3D proj proj
```

5. The output of the simulator will apear in these files:

Output file	**Content**
proj.afs	.cti file with S-parameter data

2.b: Circuit Simulation

1. Create a schematic file in ADS (for instance, name it `tryout`) and then run it. This step is necessary for ADS to create the file `netlist.log` in the workspace folder and `tryout.ds` in `workspace folder/data`.
2. Copy both `tryout.ds` and `netlist.log` in another folder and open `netlist.log`: the first line should be like follows:
   ```
   Options ResourceUsage=yes UseNutmegFormat=no EnableOptim=no TopDesignName="....._lib:tryout:schematic"
   ```
   and it has to be modified by adding `ASCII_Rawfile=yes,` in order to allow ADS to export the results in a `.raw` file:
   ```
   Options ResourceUsage=yes UseNutmegFormat=no ASCII_Rawfile=yes EnableOptim=no
   TopDesignName="....._lib:tryout:schematic"
   ```
3. Then, the simulation can be run from command line by using the following command:
   ```
   hpeesofsim [-r output_rawfile_name] [netlist_inputfile_name]
   ```
   where `output_rawfile_name` is the path to the `.raw` file that will be created to store the results when the simulation is over, whereas `netlist_inputfile_name` is the path to the `netlist.log` file.
4. Finally, the .raw file can be read in Matlab by means of the read_raw function.

As a final remark, it is important to stress that since the `netlist.log` file contains all the parameters of the circuit, it is possible to readily modify it before running a simulation from command line, in case the configuration of the circuit itself has to be changed without opening ADS. However, the properties of any model created from a layout and then imported in the schematic cannot be modified in this way. Therefore, in case the properties of a layout need to be changed, the only possibility is to modify the corresponding simulation files and then run `Momentum` from command line. Since this procedure is less straightforward than modifying the `netlist.log` file, it will be dealt with in the following next two Sections.

Section 2: How to modify the design parameters in the layout

As mentioned before, the file `proj_a` contains all the information about the geometry of the design in ADS. More specifically, for each figure that has been drawn in the project layout there is a line in `proj_a` containing the 2D coordinates of the points that identify it (e.g. its corners, in case it is a rectangle). To modify the geometry of the project in Matlab, we need to use four functions: proj_a_import, proj_a2struct, struct2proj_a and proj_a_write. The first one imports the `proj_a` file in Matlab. Then, `proj_a2structure` reads the information in the file and turns it into a data structure. The data structure may be read, modified and `struct2proj_a` allows any change to be imported back in the structure. Finally, `proj_a_write` writes the structure back in the `proj_a` file. As an example, suppose that the first figure has to be shifted along the `x` axis by a quantity `shift`. The corresponding sequence of instructions could be as follow:

```
proj_a=proj_a_import(proj_aPath);             %imports the file proj_a stored in proj_aPath
inputPatch=proj_a2struct(proj_a);             %information on the geometry are imported into a data structure
uP1=inputPatch.pol(1).points;                 %points corresponding to the first figure are imported into uP1, which is organized as an array of two columns
uP1(:,1)=uP1(:,1)+shift;                      %shift is applied to the x coordinate of all points
inputPatch.pol(1).points=uP1;                 %changes are applied to the data structure
proj_a=struct2proj_a(inputPatch);             %modifies proj_a
proj_a_write(proj_a,proj_aPath);              %writes the structure back in the original location of the proj_a file
```

Another possibility is to modify the information on the substrate. Those are contained in the `proj.ltd` file. For instance, suppose that a value of permittivity has to be changed, and that that information is in the 14th line of `proj.ltd`. The line itself and the instructions to modify it are the following:

```
  MATERIAL BF PERMITTIVITY=1.56 LOSSTANGENT=0.012 PERMEABILITY=1 LOWFREQ=1000 VALUEFREQ=1e+09 HIGHFREQ=1e+12
```

```
substrateData=importdata(proj.ltd);           %all lines of proj.ltd are imported into a data structure
epsLine=substrateData{14};                    %14th line is selected to be modified
epsInd=find(epsLine==' ');
epsLine=[epsLine(1:epsInd(4)+13) '1.64' epsLine(epsInd(5):end)];    %permittivity is replaced by the new value, being 1.64
substrateData{14}=epsLine;                    %modified line is placed back into the structure
fid=fopen(proj.ltd,'w');                      %proj.ltd is opened and all the lines, included the modified one, are printed back
fprintf(fid, '%s\n', substrateData{:});
fclose(fid)
```

Section 3: How to extract the simulated values from Momentum

When the simulation in ADS is finished, the results are stored in the file `proj.cti`, in the same folder where the Simulation input files are. Is the simulation includes a far field calculation, also the file `proj.ctf` is created. Both files can be imported in Matlab by means of the function citicell_import. The function needs two inputs, which are the path to the `proj.cti` file and a value for the debug (1 if wanted, otherwise 0):

```
citiImport=citicell_import(proj.cti,0)        %import data into data structure without debug
```

The data structure `citiImport` is organized as follows:

- `citiImport{1}` is the Data Cell;
- `citiImport{2}` is the Variables Cell;
- `citiImport{3}` is the Information Cell.

Each cell of `citiImport` contains different subcells:

- `citiImport{1}{block}{1}`, `citiImport{1}{block}{2}` and `citiImport{1}{block}{3}(index,parameter)` are the data block names, data block types and data block values, respectively. `block` is the number of the data block, `index` is the index of the datapoint within the block and `parameter` is the data point parameter number (e.g. for complex numbers, `parameter=1` points to the real part, whereas `parameter=2` points to the imaginary part);
- `citiImport{2}{varno}{1}`, `citiImport{2}{varno}{2}` and `citiImport{2}{varno}{3}(index)` are the variable names, variable types and variable values, respectively, being `varno` the number of the variable;
- `citiImport{3}{1}` and `citiImport{3}{2}` are the CITI file title/name and the CITI file version, respectively.