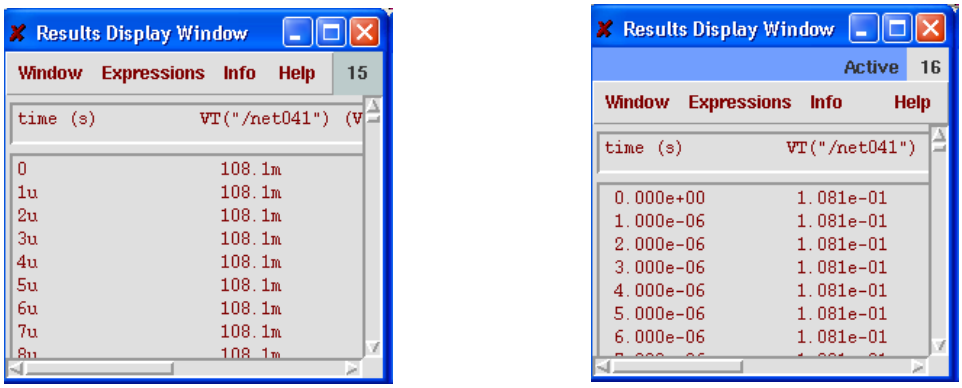
GUIDELINES FOR MOVING DATA FROM CADENCE TO MATLAB

A. Moving Small Amounts (< 1900 points) of Data from Cadence to Matlab

1. Simulate the circuit.
2. From the waveform window, enter the Calculator window.
3. Using something like VT or IT, select a certain waveform on the schematic to print to a file.
4. Select **printvs**. A window pops up (Fig. 1) where you can introduce the data range you want to save. A results display window appears with the data (Fig. 2a).
5. Usually you prefer to save the data in scientific notation. You can change the data representation by choosing **Expressions> Display Options**. The window of Fig. 2b appears.
6. Then, save the data to a file by Selecting from this window **Window>Print**. The window of Fig. 3 appears. Select **Print to File**, and write the filename.

Figure 1. Data range selection



(a) (b)

Figure 2. Data displayed (a) Engineering notation (b) Scientific notation

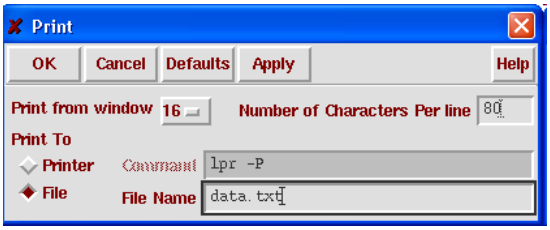


Figure 3. Printing data to a file

7. Now, you should have a text file (e.g. data.txt) with the X and Y values of the waveform data in two columns. The number of rows corresponds to the number of data points you selected above (let's say, 1000).
8. Now you can read this file in Matlab. For instance, you can use the following function:

```
function [x,y]=lv2m(file,size)
fid=fopen(file);
for i=1:size
x(i)=0;
y(i)=0;
end;
for i=1:size
s=fgetl(fid);
x(i)=str2num(s(1:12));
y(i)=str2num(s(13:size(s,2)));
end;
fclose(fid);
```

Then, for reading the file into X and Y vectors in Matlab, you can write:

```
file='d:\antonio\td\icsrd\data.txt'     % Path to file
[x1,y1]=lv2m(file,1000);
```

B. Moving Large Amounts (> 1900 points) of Data from Cadence to Matlab

1. Limit the output of the simulation to only what is necessary by specifying a print output start time in the transient setup (under options). Under the output… save all menu, make sure to de-select save all voltages and currents and only save those waveforms that you require. We will suppose that the node names of interest are "vc", "clk", and "data."
2. The command to print the simulation data is called ocnPrint(), which must be typed in the ICFB command window. For example, you might type (all on one line):

   ocnPrint(VT("/vc") VT("/clk") VT("/data") ?output "./data/myOutFile" ?numberNotation 'none ?precision 12)

   where "./data/myOutFile" is the path and name of the output file, 'none means no number notation (it's faster than using exponential or scientific notation) and 12 is the number of significant digits. The precision can be as high as 16.

3. For the example above, 4 columns will be written to the file myOutFile, the first being time. Open it using emacs on gauss – tesla doesn't have emacs – using the command:

   emacs myOutFile &

   The thing to notice is that the samples (in time) are not evenly spaced. Delete the header (and spaces) at the top of the file, so that ONLY data is in the file. At this point, I suggest you rename the file something like:

   mv myOutFile XXXX_time_vc_clk_data.dat

   where XXXX is a description of the circuit and the other words tell what data is in each column.

4. Read the data into Matlab using the load command and then separate the data into column vectors:

   ```
   infile = 'z:/cadence/data/time_vc_clk_data.dat';
   temp = load(infile, '-ascii');
   time = temp(:, 1);
   ```

```
vc = temp(:, 2);
clk = temp(:, 3);
data = temp(:, 4);
clear temp;
```

5. Resample the data at a uniform sampling rate using the spline() function in Matlab. First you have to generate a new time vector that is evenly spaced, e.g.,

   ```
   newtime = 0:tstep:tstop
   ```

   where you specify the step size (tstep) and stop time (tstop) that you want.

   Then use the spline command, as in:

   ```
   newvc = spline(time, vc, newtime);
   newclk = spline(time, clk, newtime);
   newdata = spline(time, data, newtime);
   clear time;
   clear vc;
   clear clk;
   clear data;
   ```

   Notice the use of clear commands in order to save memory.