Arpeggio Tutorial Version 0.1 Oct. 13, 1999

Welcome

to Arpeggio, the advanced Time-Domain Full-Wave Electromagnetic simulator. This tutorial will guide you through the important steps for using Arpeggio.

License Agreement

This version (Ver. 0.1) of Arpeggio is a freeware. It can be freely distributed with its original form. Any modification is strongly prohibited. This version is not bug-free, please report any bug to the author.

System Requirement and installation

Before we start the tutorial, you have to install Arpeggio to your computer. The system requirements for using Arpeggio are

- Microsoft Windows 95/98 (NT not tested, but should work also)
- Pentium class CPU (X86 should work also, but it will be too slow)
- At least 800*640 resolution monitor (1024*768 is highly recommended)
- At least 32 MB RAM (This version only allows you to run up to 20 MB RAM).

Known System Issues and Bugs

This version (ver. 0.1) has been test for only few cases with single layer and low dielectric constant substrate microstrip circuit. It may have bugs for other circuit configurations. A known issue is there is no communication between the ProgressViewer and the FD-TD core running in the background in this version. If you have been waiting for long time without seeing any drawing in ProgressViewer, the FD-TD core might fail to launch. A check will be added in later version to make sure FD-TD core is successfully launched. Try to reboot the computer to see if the problem solved.

Theory of Arpeggio

Although you don't have to (and you don't care about) the details how Arpeggio works, you have to know some basic theories behind Arpeggio in order to successfully fulfill your job by using Arpeggio. You also have to know Can Do and Can't Do of Arpeggio. Knowing these is very important since it will give you sense when you should use Arpeggio and when you should not use it. And even when Arpeggio gives you the results which you didn't expect, knowing the basic theories might give you a sense of knowing what happened and possible way to resolve that.

Arpeggio adopts Finite-Difference Time-Domain (FD-TD) method to solve Maxwell equations in time domain. If you have knowledge on electromagnetic theory, you know that EM wave propagates once it is launched by a excitation source (such as a probe or an antenna). The EM wave propagates and will be 'scattered' once it hits an obstacle (or discontinuity in microwave term). The pattern of the 'scattered' wave depends on the geometry, dimensions, material properties ….. It is a very complicated problem and this is why you have to use EM simulators. Arpeggio simulates the wave propagation in a complex situation, it allows you to obtain many quantities you desire such as S-parameters, radiation pattern.

You can also think in this way: if you want to measure properties of an antenna you invent, how do you proceed? Will you put an antenna anywhere you like to measure it? Generally you cannot because there are many reflections from everywhere which might totally destroy the real response of the antenna (call Multi-path). Instead, you have to find an 'infinite large' space to measure the antenna. A common practice is to

measure the antenna in an antenna chamber, which is covered by good EM wave absorbers which absorb EM wave without reflecting them back. Then you pump in RF signals and measure whatever the quantities you want to get. If you want to get S parameters of a low pass filter, you connect input port with a 50 ohm source and output with a 50 ohm load (absorber) and most of the time, the circuit will be finite, so a better way is to cover the edges with absorbers. Then you excite the filter with source (in time domain), let the RF signal propagates toward the filter and gets the reflected wave at input port and transmitted wave at output port. S parameters can then be found as the ration between the reflected wave to the excitation wave and the reflected wave to the excitation wave. In this fashion, the space you are measuring the circuit in looks like an 'infinite large' space. But how 'infinitely large' depends on how good the absorbers are. Arpeggio use the same concept, it has a 'virtual infinitely large' space in the computer and simulates EM wave propagation, reflection, deflection…. in a computer. Remember, Arpeggio does all these in time domain, in order to get desired quantities in frequency domain, Arpeggio also take Fourier Transform and provide you the final result in frequency domain. Therefore, a very big difference between Arpeggio and any other Frequency-Domain based EM simulator is Arpeggio simulates the actual wave propagates while Frequency-Domain based EM simulators deal heavily with math, matrix manipulation, interpolation.

Can do and Can't Do

Arpeggio is a very powerful tool for high frequency circuit design. It has many excellent abilities compared to Frequency-Domain EM simulators. However, far from being perfect, Arpeggio also has difficulties in some other situations.

Can Do :

- A true 3D EM simulator, compared to its counterpart Frequency-Domain EM simulators which mostly can only solve an infinitely large substrate (2.5 D). This means Arpeggio can solve any shape as long as you can give the geometrical description input into Arpeggio. A patch antenna sitting on top of a finite-sized substrate, and bond wire connecting two chopped dielectrics with different heights, different dielectric constants and separated by a gap are two perfect examples that Arpeggio can do while other integral-equation based Frequency-Domain EM simulators can't do or have difficulties to do.

- With a single run, Arpeggio gives you the response over a very wide frequency band. Although most of frequency domain methods take much less time for obtaining the result at one frequency, but with the same bandwidth, and particularly if the circuit is resonant (this is how microwave circuits work), frequency domain methods need to sweep many times to get an accurate resonance. So overall speaking, they are pretty much the same for run time.

Can't Do :

- Time domain methods have a well known problem for simulating a lossy thin metal. Arpeggio can't incorporate metallic loss. This seems devastating for Arpeggio compared to any other Frequency-Domain method, but it is not because depending on the situation, frequency domain methods might not be able to provide accurate info for metallic loss either! Generally speaking, metallic loss is pretty difficult to simulator in practice.

- For highly resonant circuit (like high dielectric constant dielectric resonator), once the circuit is excited by a source, the waveform in time domain will be oscillating for a long time. This will increase the run time since it is important to wait until the waveform dies out (approach to zero). Otherwise, Fourier Transform will give an oscillating curve in Frequency Domain.

- For multi-mode structures, Arpeggio will have problem to distinguish the modes. Once you record a multi-modal response in time domain and perform the Fourier Transform, the frequency response

screws up. For some special situation, there are ways to separate the mode before performing Fourier Transform (in the updated version, I will show one example to you, stay tuned). Unfortunately, it is generally not easy to do that for most multi-mode structure.

- Arpeggio can not simulate evanescent structure.

- Arpeggio is a Full-Wave EM simulator, it does lots of number crunching which requires a good CPU and lots of memories if you want to simulate a giant circuit (in Full-Wave EM simulation world, but still small in the real world). It can not, and will not be able to run as fast as a circuit simulator such like Libra and MDS in the foreseeable future!! The suggestion is to get a decent computer if you want to take full advantages of Arpeggio (and any other Full-Wave EM simulator).

Procedure of Simulation

First, you have to know the circuit you want to simulate (of course). Draw a sketch on a paper and indicate the dimensions. Now you need to discretize the circuit in x-, y- and z-directions. The general rule of thumb is that the mesh size must be as small as the smallest material size can have several meshes, and the largest mesh size must be at least one-tenth of the wavelength of the highest frequency you want. The finer the meshes are, the more accurate result will be. But consider the memory requirement, you can't have the mesh size being too small! Generally, two-meshes is acceptable but more will be much better. Arpeggio also allow you to have more than one mesh size in each direction, but it is not recommended since it will reduce the accuracy and since it hasn't been fully tested in this version, stick with using only one mesh size in each direction as hard as you can.

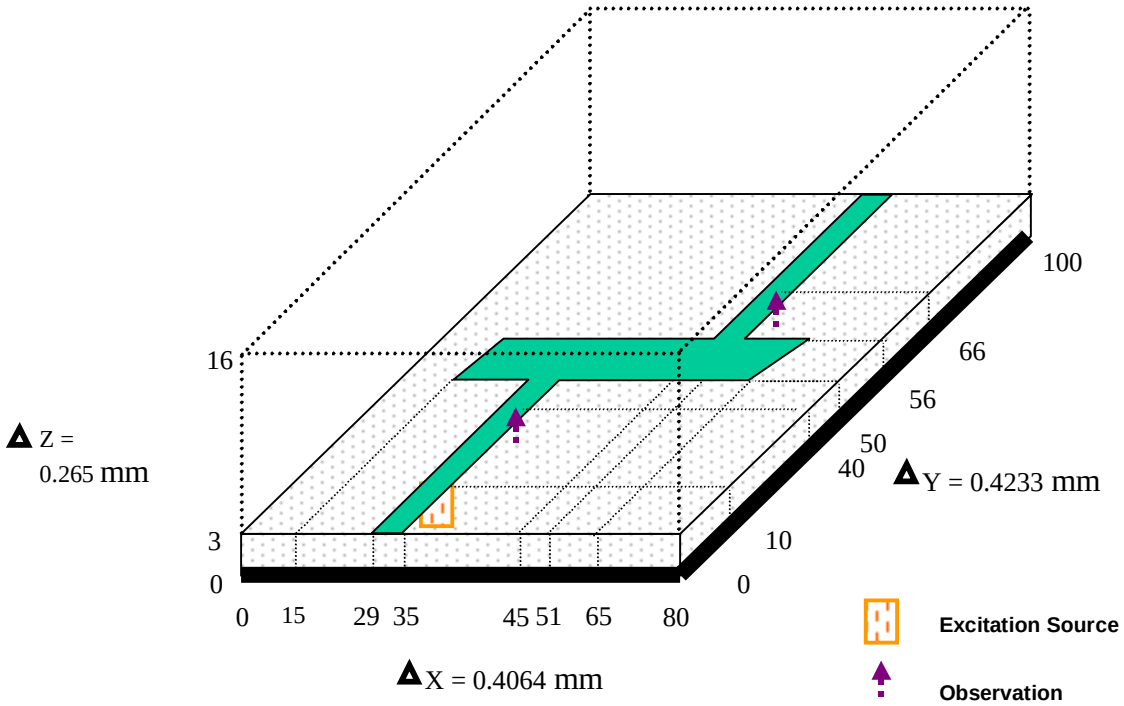
Having done with defining the mesh size and mesh index of the material inside the computational domain, you need to enter those into the ParaEditor Arpeggio provides. It should be very straightforward to use ParaEditor to enter the parameters. ParaEditor is a dialog-based text-input circuit parameter editor, in the later version of Arpeggio, you will be allowed to draw metal strip pattern also (so don't go away !!)

Some words should be mentioned about ABC (Absorbing Boundary Condition). The reason for using ABC is we only have limited computer resources such like memory. In order to simulate the 'infinite space" that I mentioned previously, a numerical absorber has to be placed around the 'box' which contains the circuit. ABC is the most important thing in Arpeggio since the choice of different ABC significantly affects the run time, memory requirement and accuracy. If you want to have a very good result, you have to use better ABC and the price you have to pay is it take many times more memory and run time (just like you have to pay more for getting better absorbers).

Finally, you are almost done. Now you can start simulation and wait until the simulation either terminates by itself (depends on the number of time steps you specify in ParaEditor) or you can stop it manually if the waveform has reached almost zero for a while. Then you can examine the results using Arpeggio's PostProcessor.

Start the Tutorial

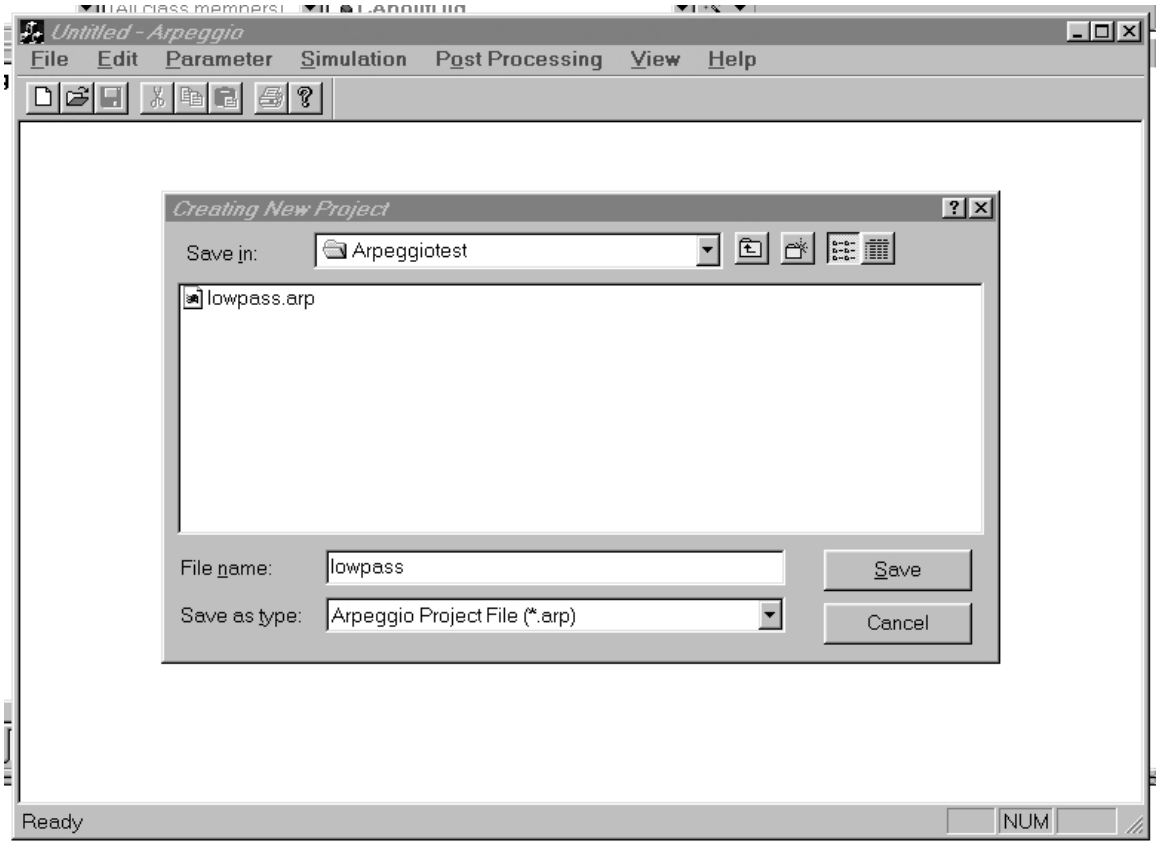
In this tutorial, you are going to learn how to use Arpeggio to simulate a microstrip low pass filter. This benchmark circuit is taken from an IEEE MTT journal paper "Application of the Three-Dimensional Finite-Difference Time-Domain Method to the Analysis of Planar Microstrip Circuits" by David M Sheen…et al, vol. 38, no. 7, July 1990, pp.849-857.



You can find a project for the low pass filter comes with the Arpeggio under the Tutorial folder in Arpeggio directory.

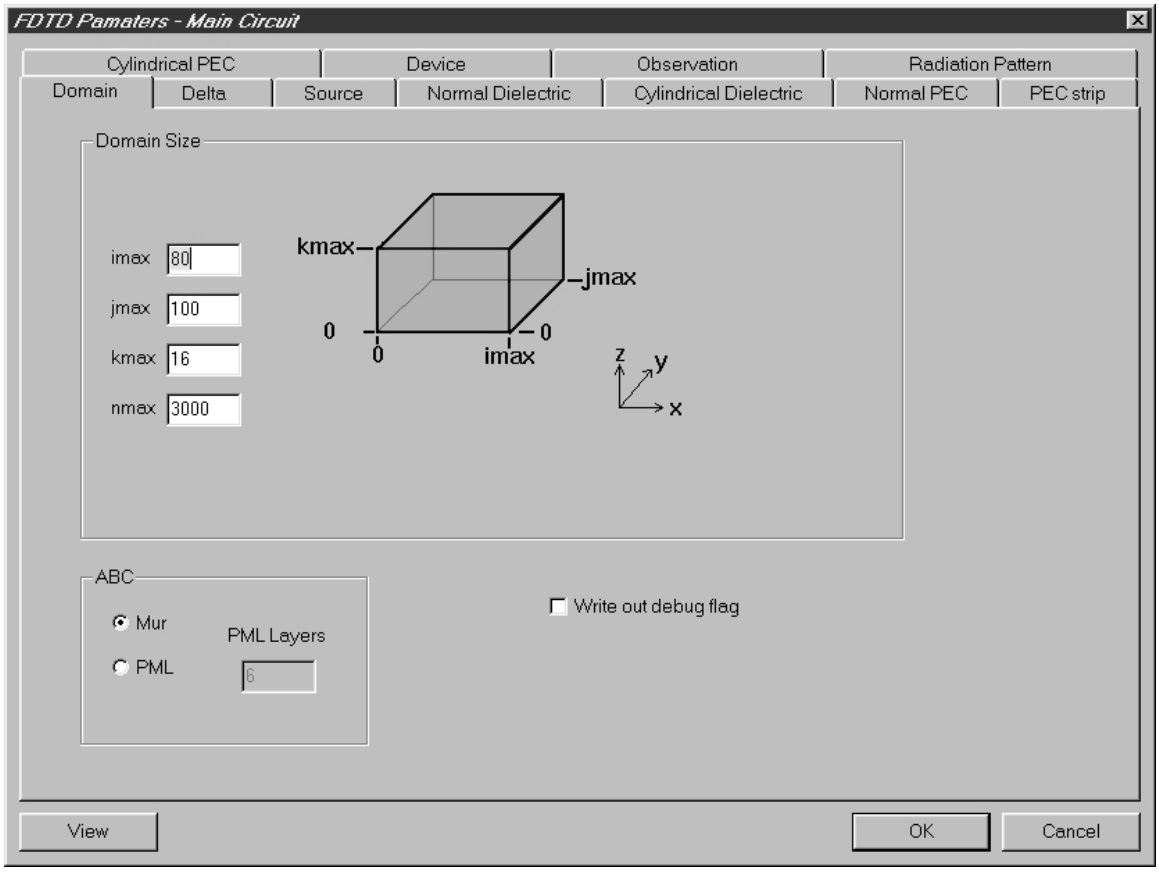
(1) Launch Arpeggio by double clicking on the icon. You will see a window showing on the screen.

(2) You have to create a project space for your simulation. Click on *File*-*New* and create a project in the directory you desire. In this case, a project called lowpass.arp is created in d:\arpeggiotest\lowpass\.

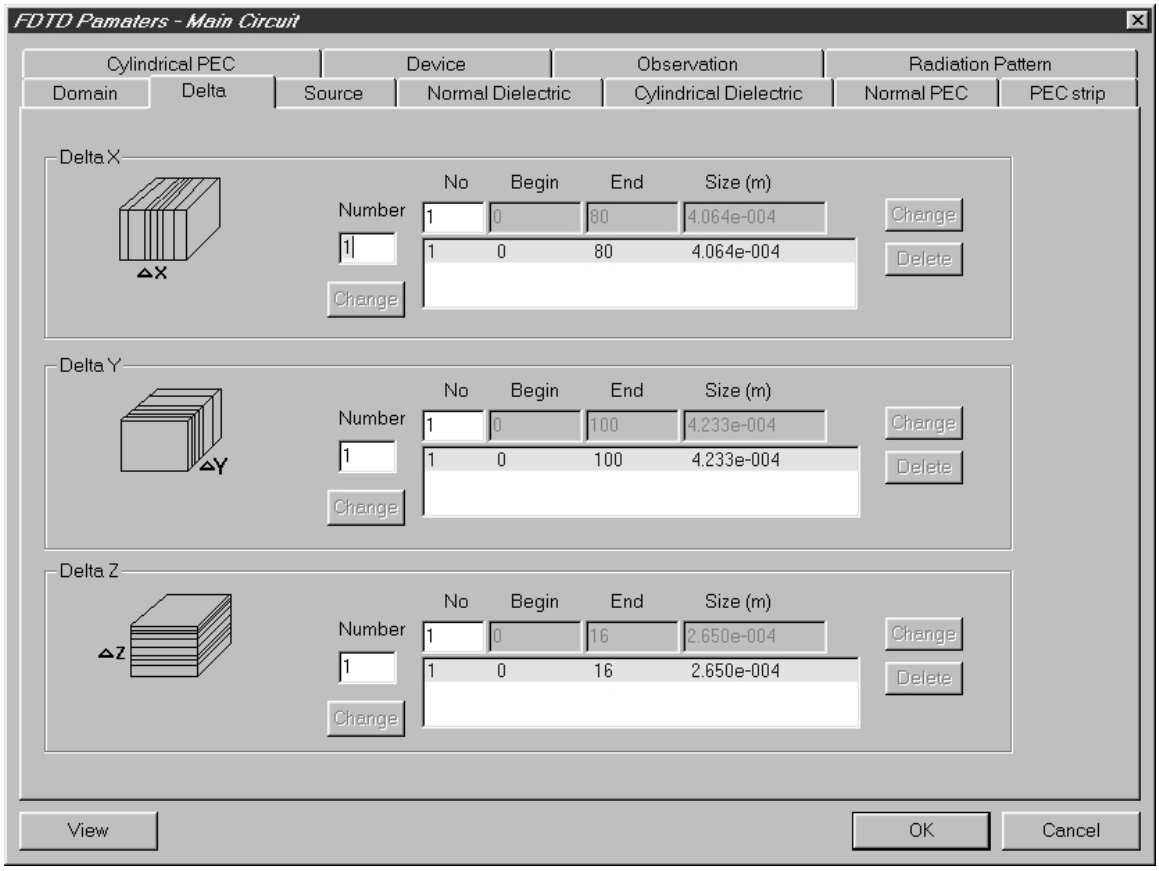


(3) After you press ***Save***, a new project is created for this tutorial. Since this is a new project, only ***Parameter*** – ***Main*** – ***Edit*** is enabled, meaning you can only edit parameters of Main circuit. Proceed to do so.

(4) The domain size for this case is 80*100*16 and we are planning to simulate for 3000 time steps (a rough guess). In this version, only first order Mur Absorbing Boundary Condition is allowed. The Mur ABC provide a fastest way of simulating an open space and its accuracy depends on some factors. Uncheck 'Write out Debug flag', check it only when you have problem and want to debug it.



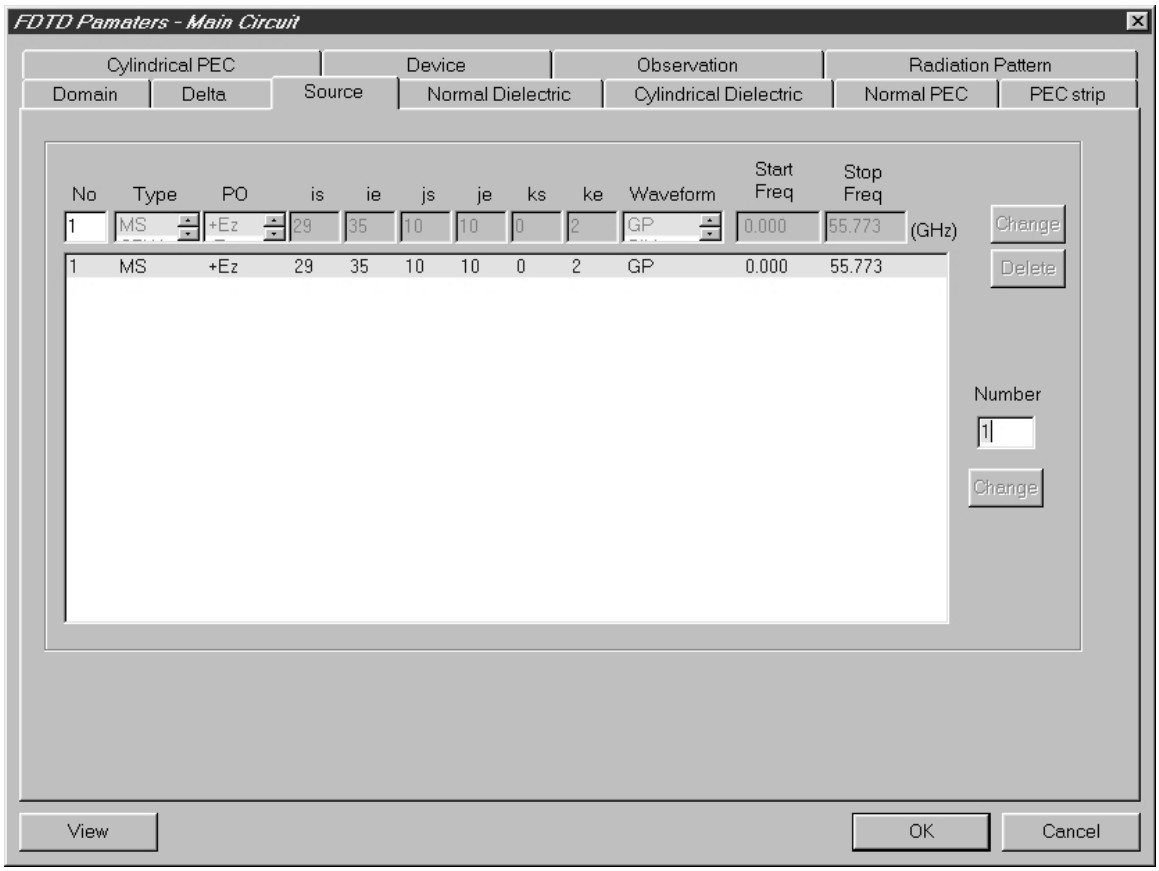
(5) Give mesh size to the entries



Note you have to press *Change* to make the input effective and those newly input values will be showing in the big area under the entries. Only data in the big area are real input to Arpeggio. You can also double click on the any item in the big area to select it into the entries to modify it.

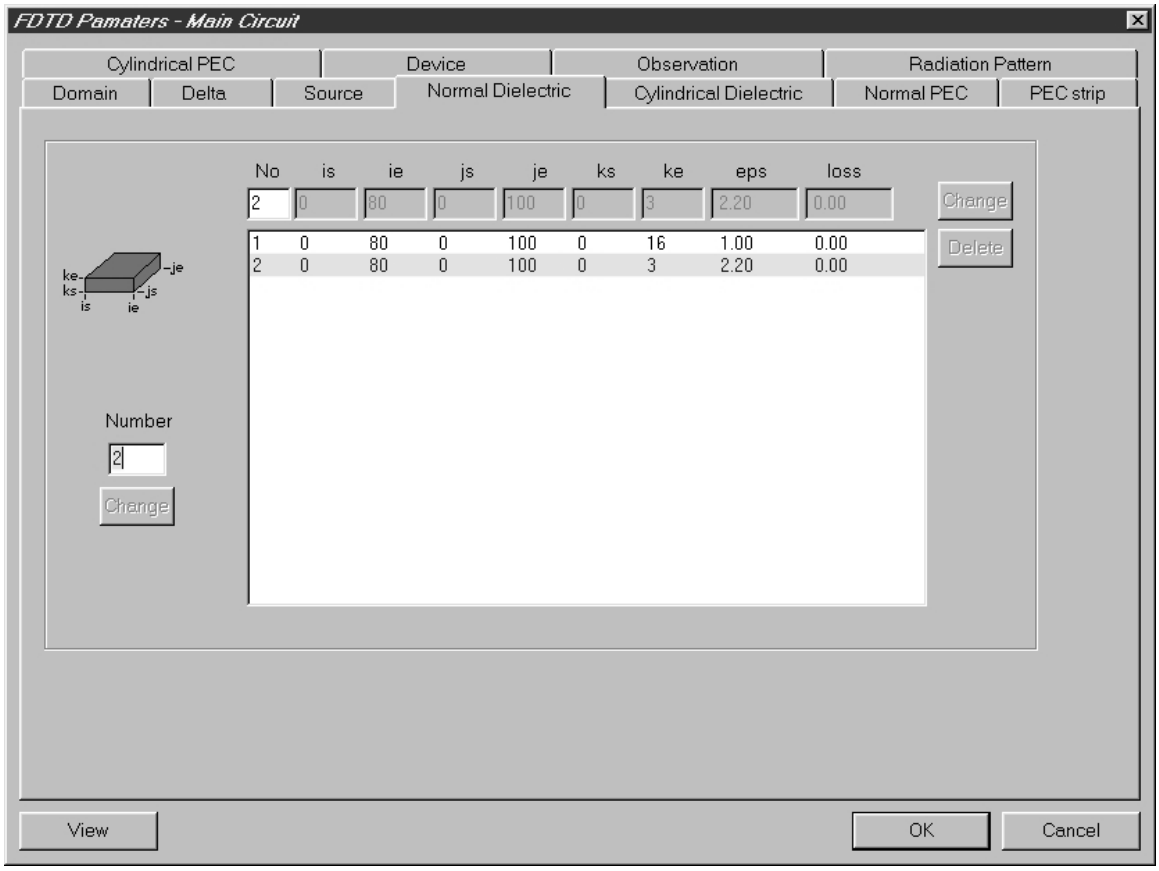
Remember Arpeggio has the ability of adopting different mesh size in each direction. But it is generally not recommended.

(6) Next, give the excitation source to the Arpeggio. It just like when you are using Network Analyzer, the internal circuit generates high frequency signal to excite the circuit you are trying to measure.



In this case, we are simulating a microstrip circuit, so we put an excitation source sheet just under the microstrip line. Since we are trying to find S-parameters over a frequency band, choose GP (Guassian Pulse) and give the input frequency from 0 to 55 GHz.

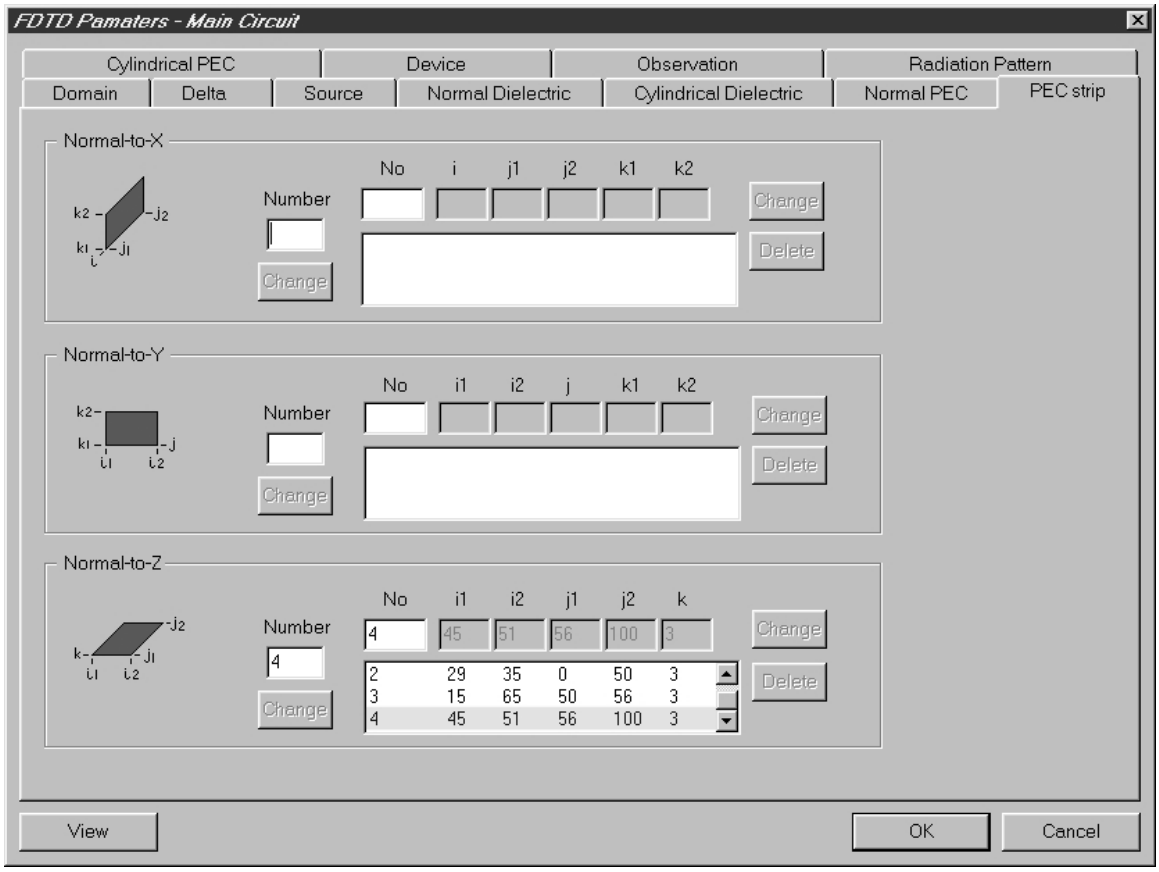
(7) Next, define the dielectrics used in the simulation. We have two dielectric, one is the entire space, which is the free space with dielectric of 1. The other one is the substrate for the circuit.



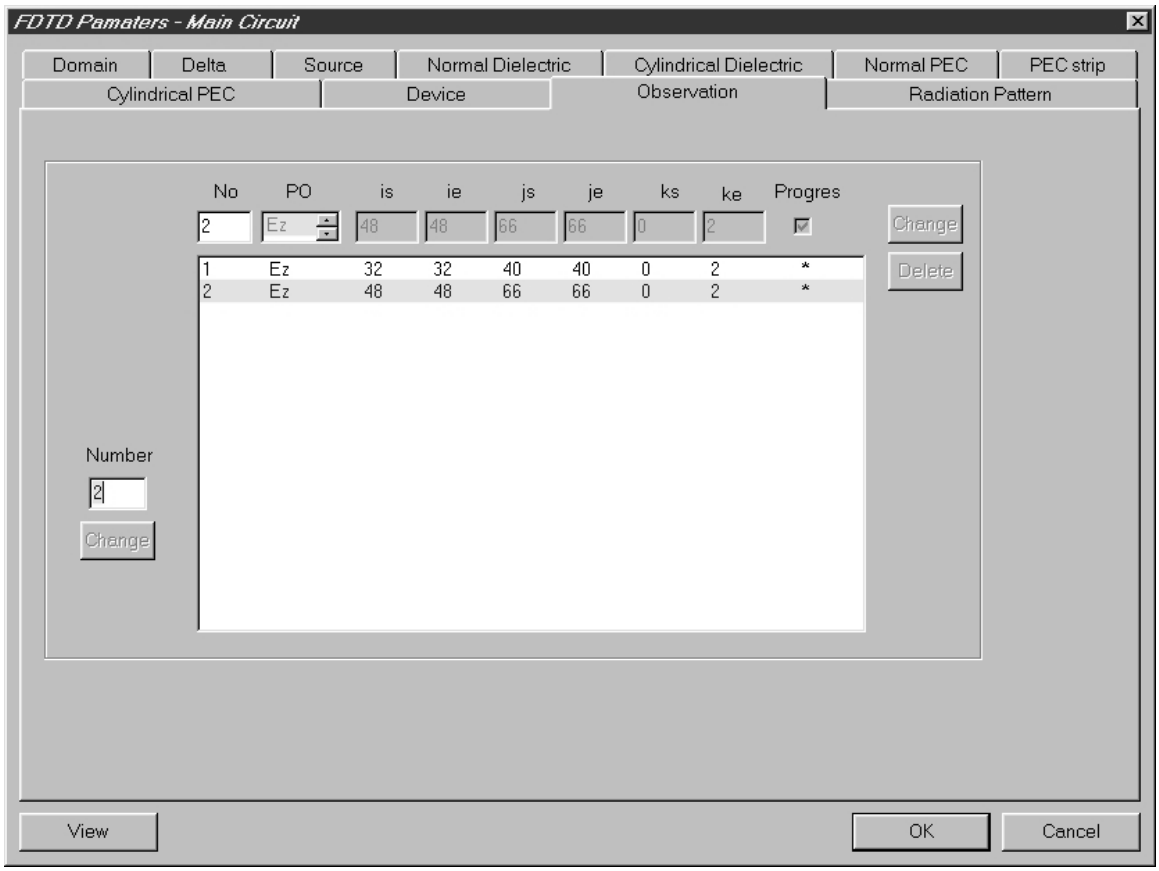
If you like, you can put in the loss of the dielectrics also. Noted that the later dielectric have higher priority and will overwrite all the dielectrics before it. In this case, from k=0 to 3, the dielectric will be number 2 even though number 1 is also specified. You can also specify number 1 as from k=3 to 16.

(8) Skip to PEC strip since there are no cylindrical dielectric and arbitrary PEC in our circuit.

We have four PEC (perfectly electrical conducting) strips (infinitely thin) to input, including the ground plane (0, 80, 0, 100, 0) (not shown in the figure). Notice the Normal Dielectric and the ground plane touch ABC, meaning they are infinitely large.

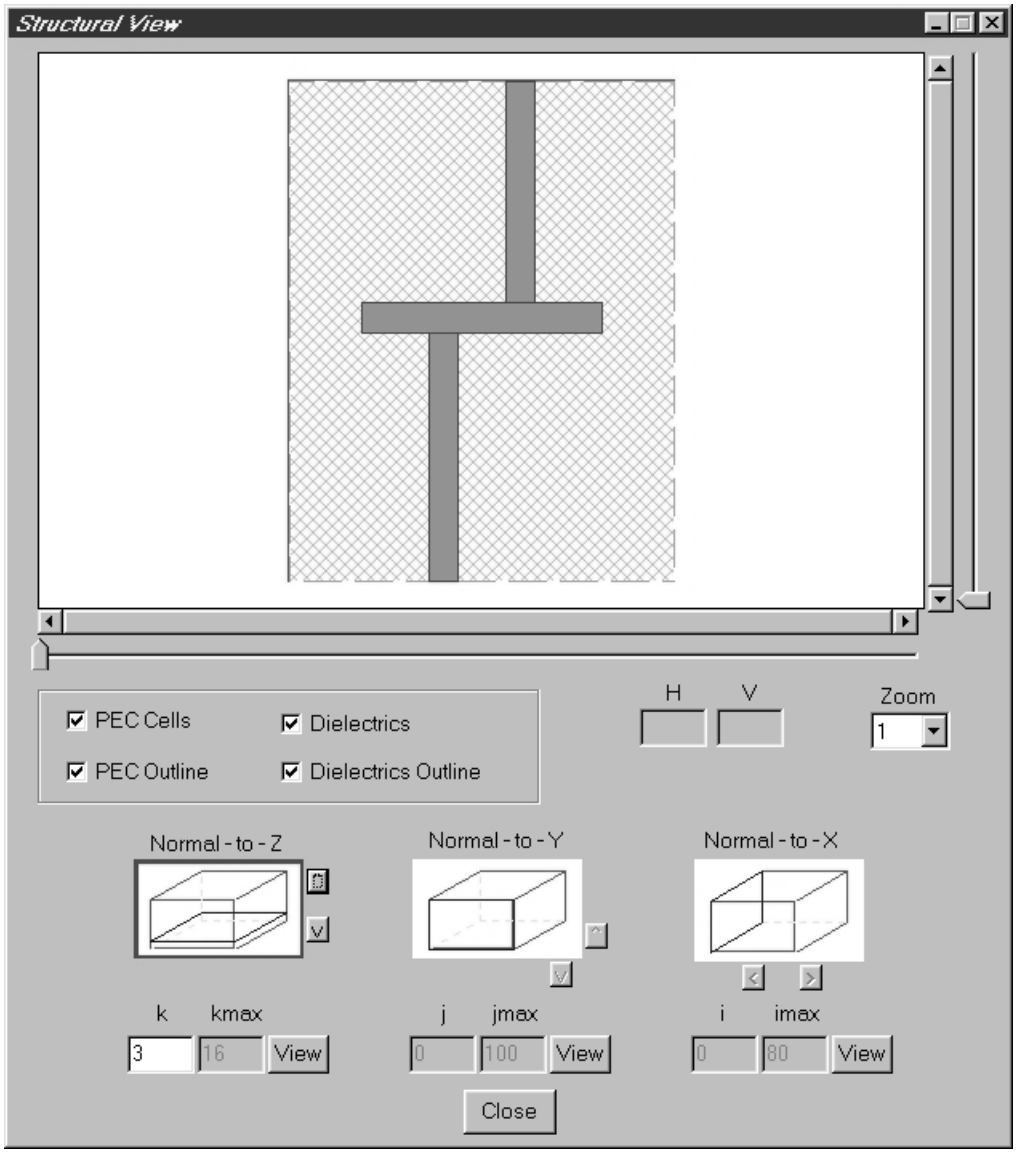


(9) Jump to Observation. Since this circuit is a two-port circuit, and we want to find S11 and S21, we need to record two voltages, one at port 1 and the other one at port 2 (just remember how we use Network Analyzer to get S11 and S21, same thing here).



The general rule of thumb is put the observation point 10 cells away from the discontinuities. In our case, the discontinuities are at 50 and 56 (y-direction). So put observations at 40 and 66. Also notice for microstrip circuit, the voltage is defined as the integration of electrical field from ground plane up to the top strip, so we put 0 for ks and 2 for ke (microstrip sits at k = 3). Arpeggio let you observe up two waveforms when running the simulation in the ProgressViewer, here check Progress for both observation voltages.

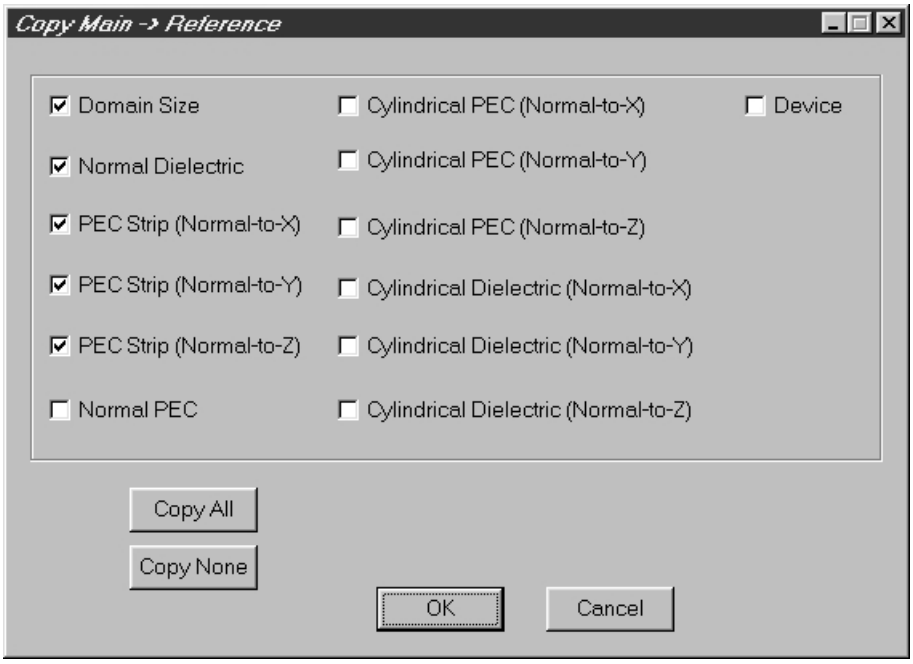
(10) You are done with ParaEditor. Now you can view the circuit by clicking on *View*.



You can play with this dialog to examine the circuit you just edit.

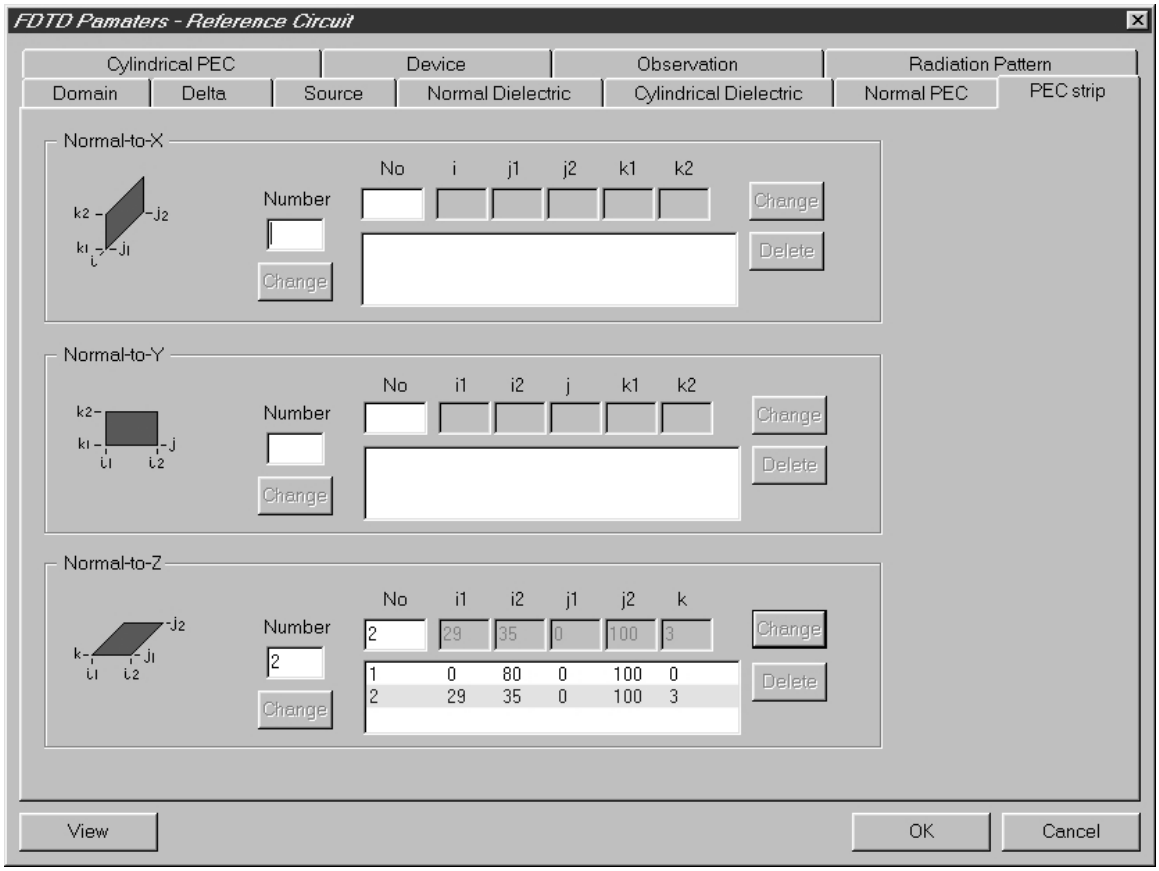
(11) When you are done with examining the circuit, press *Close* and then *OK*, return to the main window.

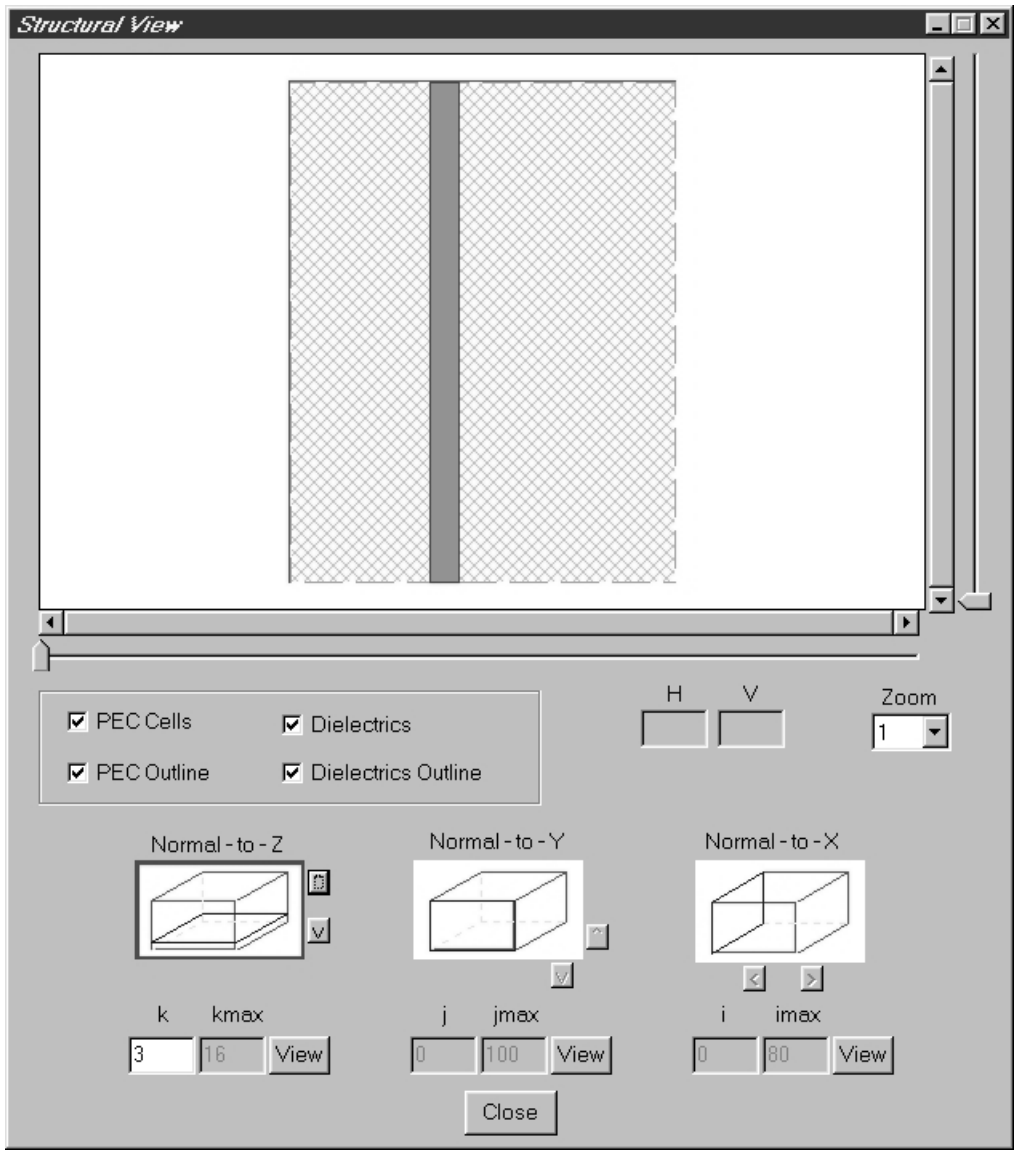
(12) Next, you need to edit the Reference Circuit for obtaining the incident wave to calculate the S parameters. Go to ***Parameter***-***Reference***-***Copy***.



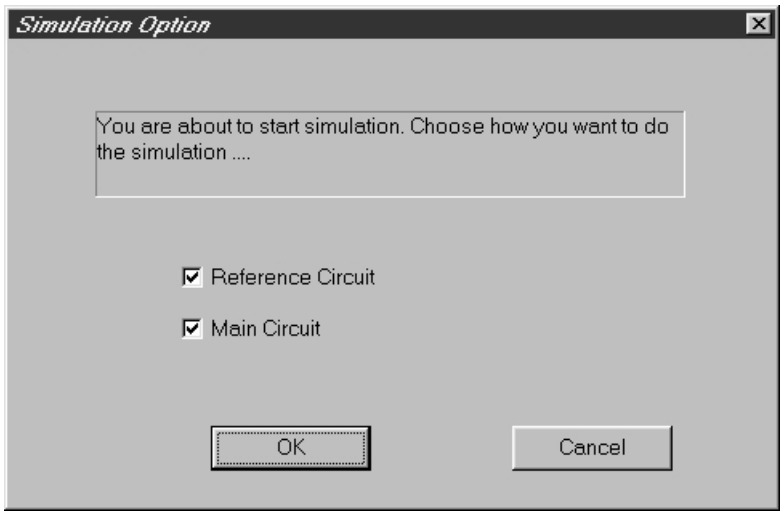
This dialog provides an easy way for you to construct a 'template' for Reference Circuit since mostly likely the Reference Circuit will be very similar to the Main Circuit. You only have to modify some parameters in most of the case. You can select anything you want to copy from the Main Circuit you just edited. Press *OK*.

(13) Go to ***Parameter***-***Reference***-***Edit*** and you will see the same ParaEditor dialog. For several parameters, you can't edit such like size of the meshes, source, and observation. They are generated for you in the previous copy process. Normally, the Reference Circuit is just a uniform circuit, so in this case, we only need to edit the PEC strip. There are only two PECs in Reference Circuit, a ground plane and a strip line.



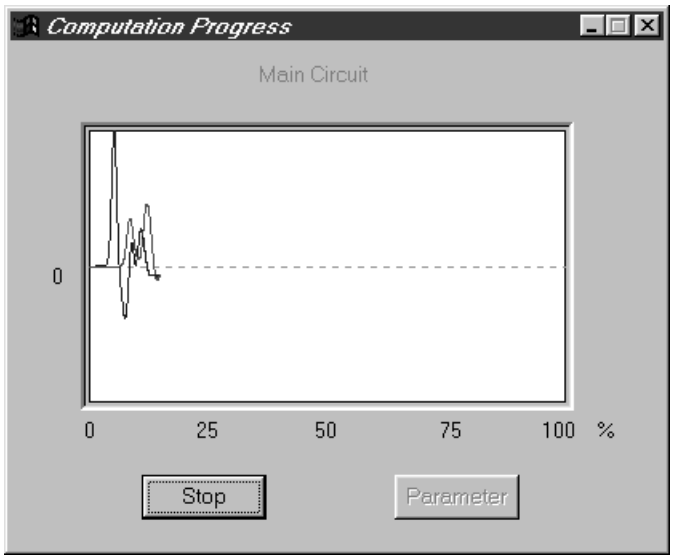

Structural View
PEC Cells
Dielectrics
PEC Outline
Dielectrics Outline
H
V
Zoom
1
Normal - to - Z
Normal - to - Y
Normal - to - X
k
kmax
3
16
View
j
jmax
0
100
View
i
imax
0
80
View
Close

(14) When everything is done, press *Close*, *OK* and go back to Main Window. Now we have done all the Circuit Parameters editing, we can start simulating the circuit. Go to ***Simulaton***-***Start***.



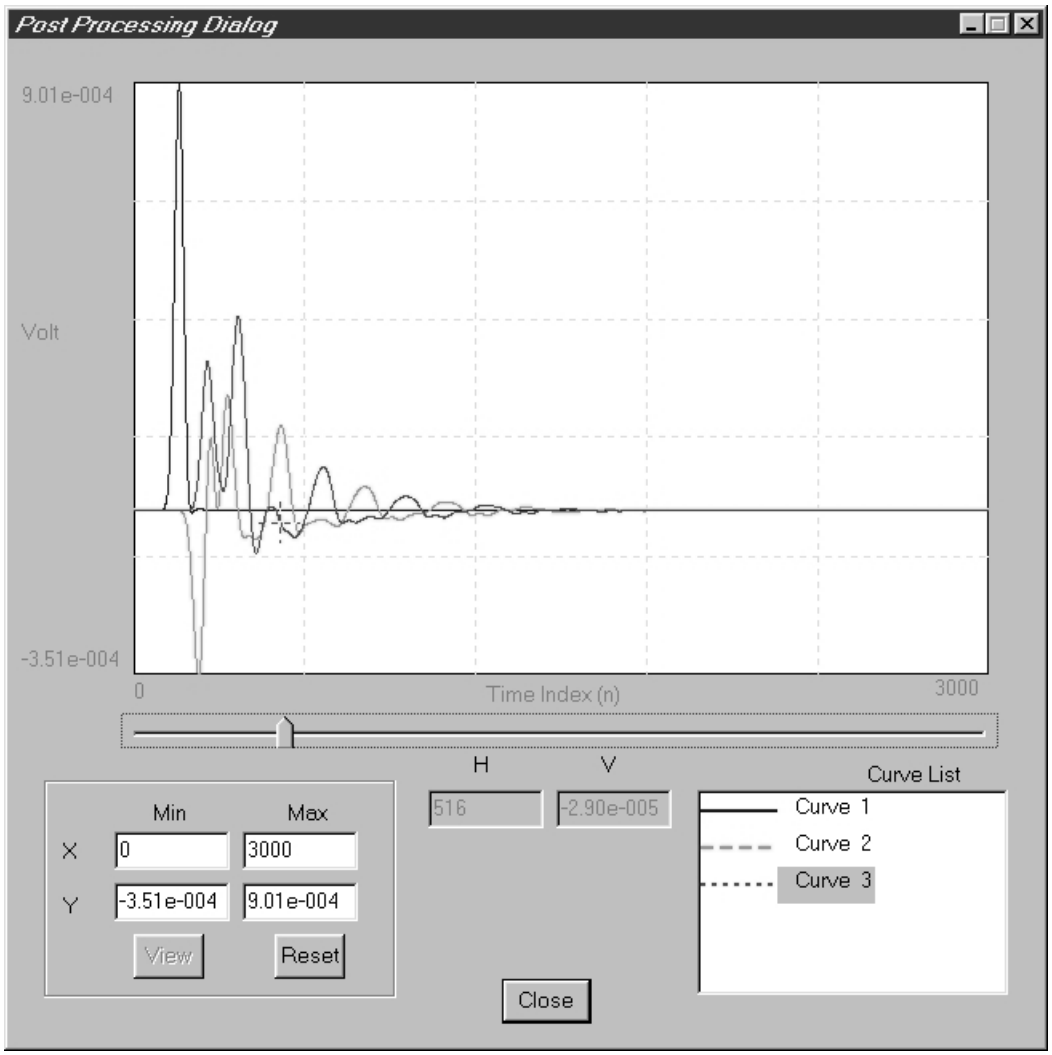
If you haven't done any simulation, just press OK to start simulation. But if you already simulated either Reference Circuit or Main Circuit, uncheck the one you don't want to simulate again. In this case, just press *OK*.

(15) You will notice some activities going on, and you are supposed to see the progress of the simulation on the screen



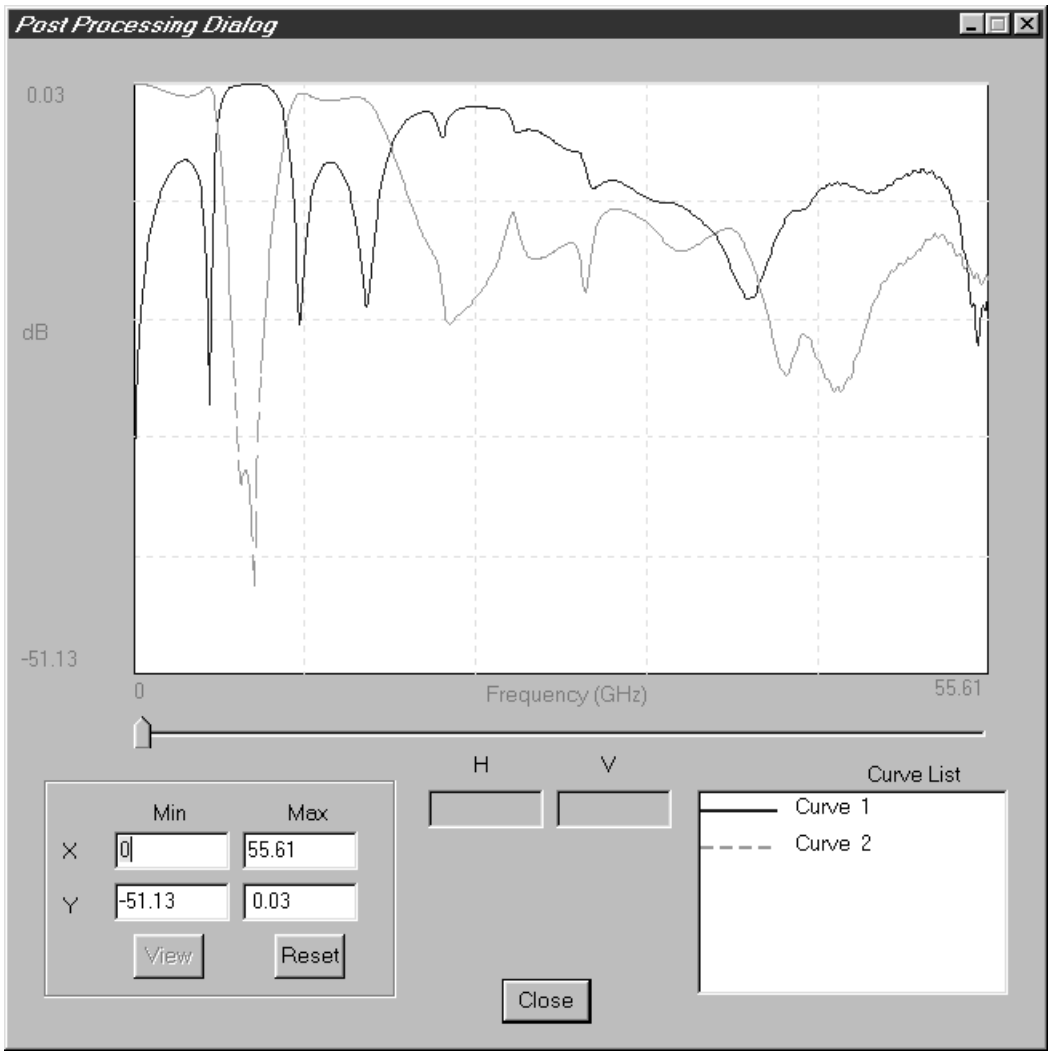
This function is very powerful, it allows you to examine not only the progress of the simulation, but also provides a way to examine the success of the simulation since you might make some mistakes in editing the circuit but didn't find them in ParaEditor. With this ProgressViewer, you can easily see if there is something wrong and you can stop the simulation as early as possible. If you are not familiar any time-domain method, you might wonder how you are able to know if the simulation is good or not by just looking at the time domain waveform. Yes, generally you can't, but for something really wrong, such like the waveform blows up, or signal stays at zero or some really strange waveforms, you know something is not right, and you can stop the simultion, go back to ParaEditor to debug it. When you are getting used to Arpeggio, you will have some sense on how the time-domain waveform should look like roughly.

(16) When the simulation is done, the main window pops up, now you are ready to see the results. Go to *Post Processing*-*Time Domain Waveform*, you can examine the response in time domain.



It is always a good practice to examine the time domain waveform before see other results.

You can see the S parameter by pressing *Parameter*-*S Parameters* as



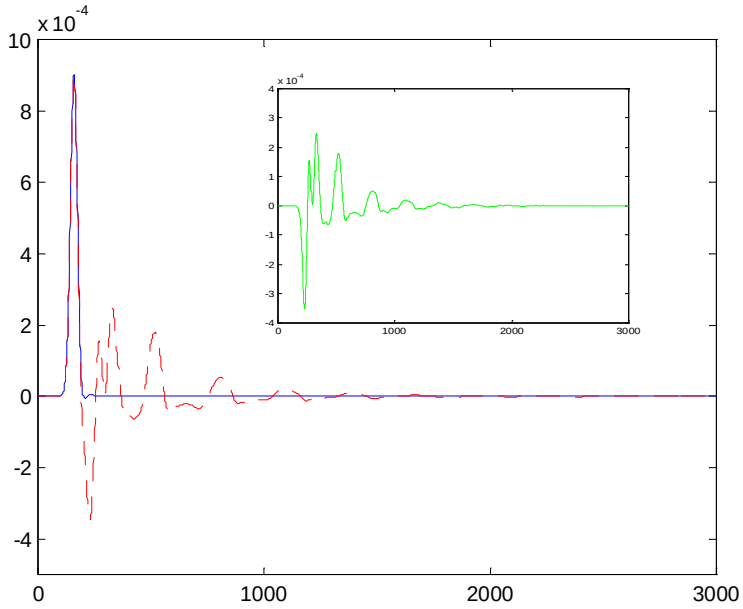
Congradulation, you just successfully done a complete simulation with Arpeggio !!!!

Useful hints

After done with the above tutorial, you might have many questions. Here are some useful hints for you

[Q] Why we need to simulate both Main Circuit and Reference Circuit?

[H] If we want to get the S parameters, we need to know the incident wave. The Main circuit is the circuit with everything there, and you excite the circuit with a source you specify. At port 1, the time domain waveform will be incident+reflected. In order to get reflected wave to calculate the reflected wave, we need the incident wave only (without any discontinuity).



This figure shows the time-domain waveforms at port 1 that we recorded in the previous low pass filter tutorial. The solid blue line is for Reference Circuit and dashed red line is for Main Circuit. It should be clear here that the waveform for Main Circuit that we recorded contains incident wave and reflected wave. Where in the small window, you can see the reflected wave (after subtracting Reference Circuit waveform from Main Circuit waveform).

[Q] How to choose Reference Circuit?

[H] It depends on the circuit you want the S parameters for. For example, in the previous example, the Reference Circuit is just a uniform microstrip line. And this is usually the case.

[Q] How to decide the size of the meshes?

[A] Just look the explanation in the tutorial, that should be enough.

[Q] Where to put the excitation source and where to put the observation points?

[A] Just follow the tutorial, this is how I put them usually.

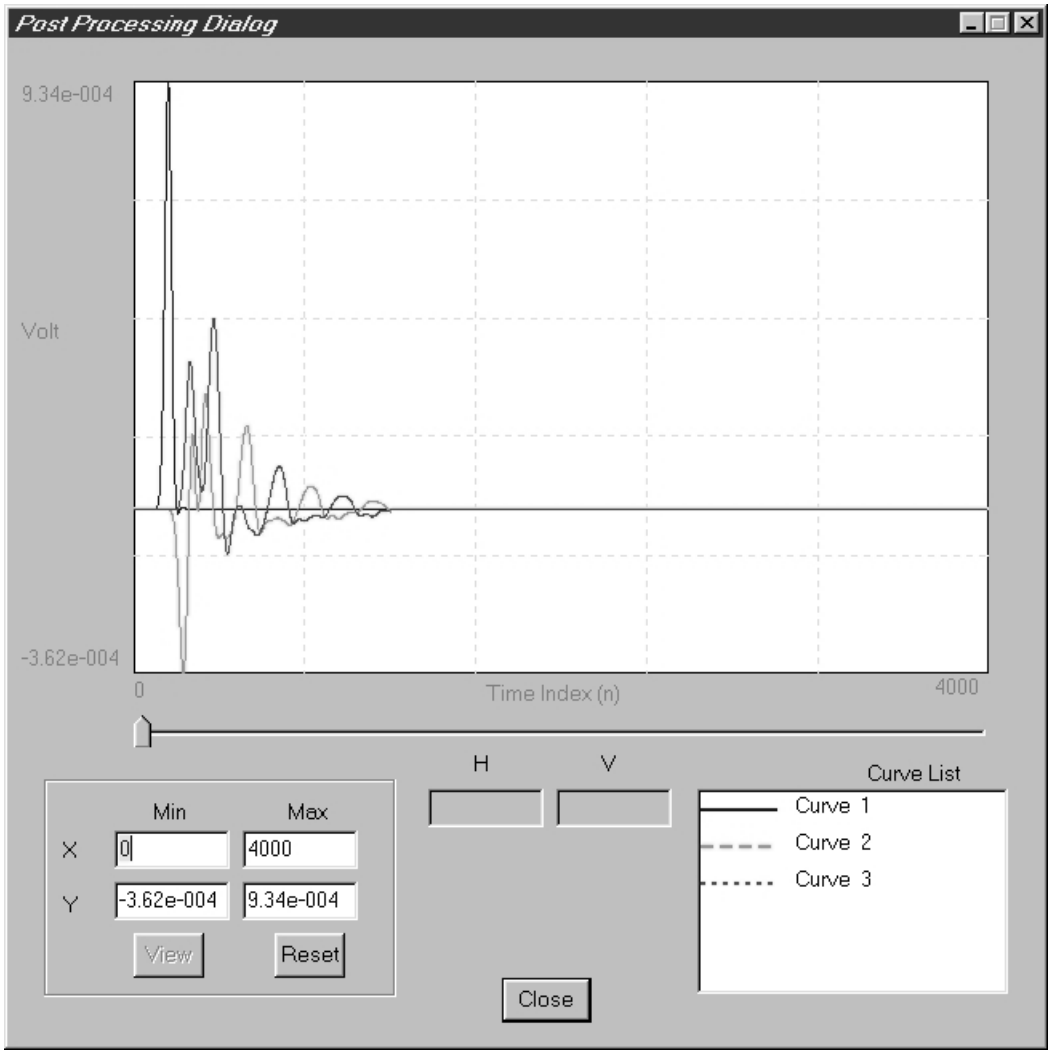
[Q] How many time step do I need for the simulation?

[A] Generally this is a difficult question to answer. It depends on the characteristics of the circuit you simulate. For example, if the circuit is highly resonant, you need to put a very large number (like 30000 or even more). Normally, 4000 will be the first try. After getting used to Arpeggio, you might have a sense on how the waveform will be and you might have a guess for how many time steps you need! Arpeggio allows you to stop the simulation manually and still getting results. In the later version of Arpeggio, you will be allowed to extend the number of time steps on-the-fly.

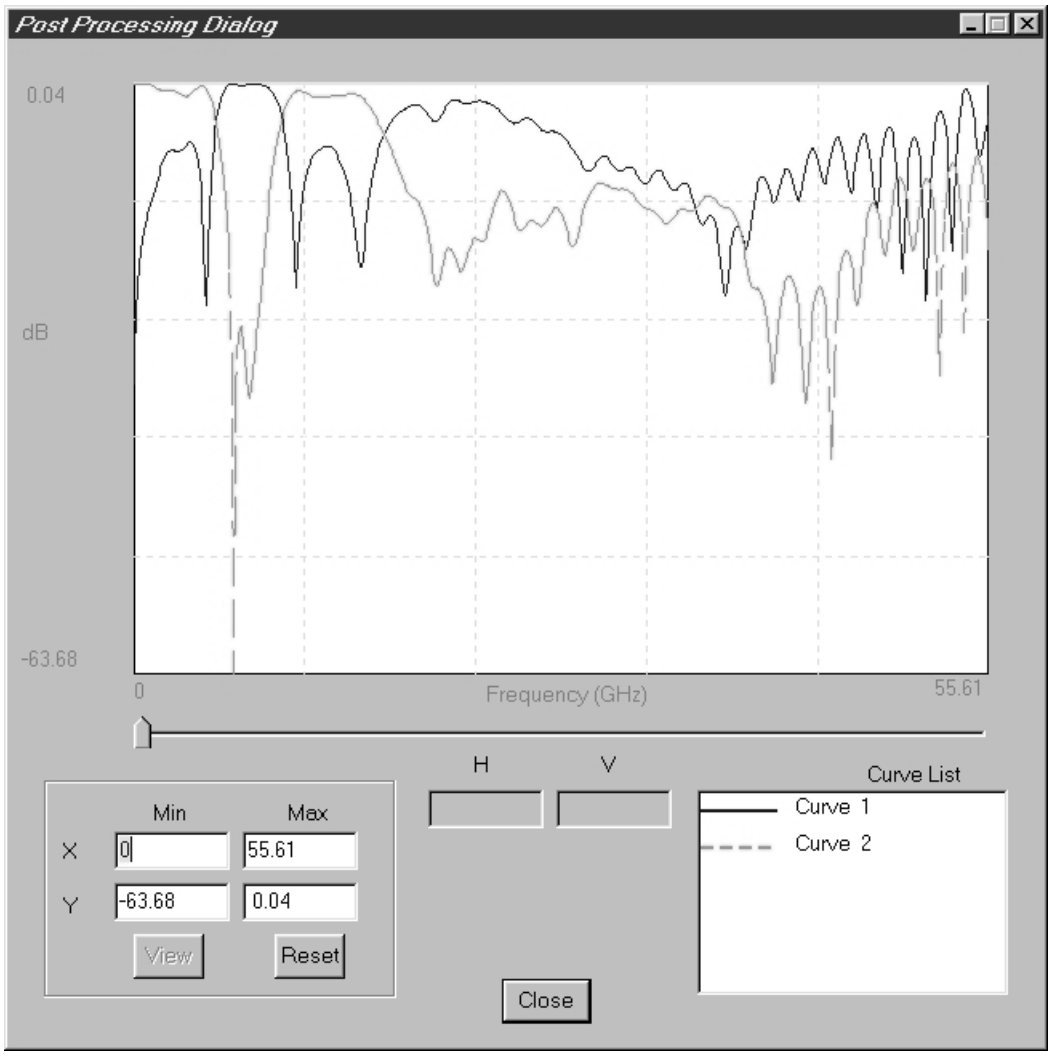
[Q] When can I stop the simulation?

[A] You have to wait until the waveform becomes almost zero. Otherwise the frequency response will be strange. If you wonder why and want to know more on this, please look at any book dealing with DFT (Discrete Fourier Transform).

Here is an example, with the same low-pass filter in the previous section, we stop the simulation earlier at around time step of 1200



And here is the S parameters



You can compare this with previous one. You can see the trends are very similar while in this case, it is oscillatory especially at higher frequency. If we stop the simulation even earlier, you should see it affects low frequencies also.

In this example, you should note that small difference in time domain may cause big difference in frequency, especially for phase. This is also the reason why ABC is so important since we need to transform the time domain waveform to frequency domain, and even if ABC just cause a little reflection in time domain, but the response in frequency domain may be quite different.

[Q] Which ABC should I choose ?

[A] It depends on the situation. First Order Mur is only good for near normal incident wave case like TEM wavegiude or Quasi-TEM waveguide like Microstrip line, and low dielectric constant circuit (say below 4). The accuracy will be lower with lots of waves hitting the ABC with large oblique incident angle, and/or high dielectric constant substrate. PML is the super high performance ABC suitable for most of the case, but it is slow compared to Mur's ABC. In the later version of Arpeggio, more ABCs will be added to compromise the run time and accuracy.

[Q] You mentioned that if the circuit touches ABC, it is infinitely large. But if the circuit is finite, how far should I put the circuit away from the ABCs if the circuit parts don't touch the ABC?

[A] It depends on the ABC you choose. If you choose PML, the shortest distance that I tried before was 5 cells. But if you choose Mur's ABC, the suggestion is 15 to 20 cells.