Rotman Lens Simulation in XFdtd 7.0

Saving time by using XFdtd 7.0 scripts to import complex geometries and perform postprocessing of results

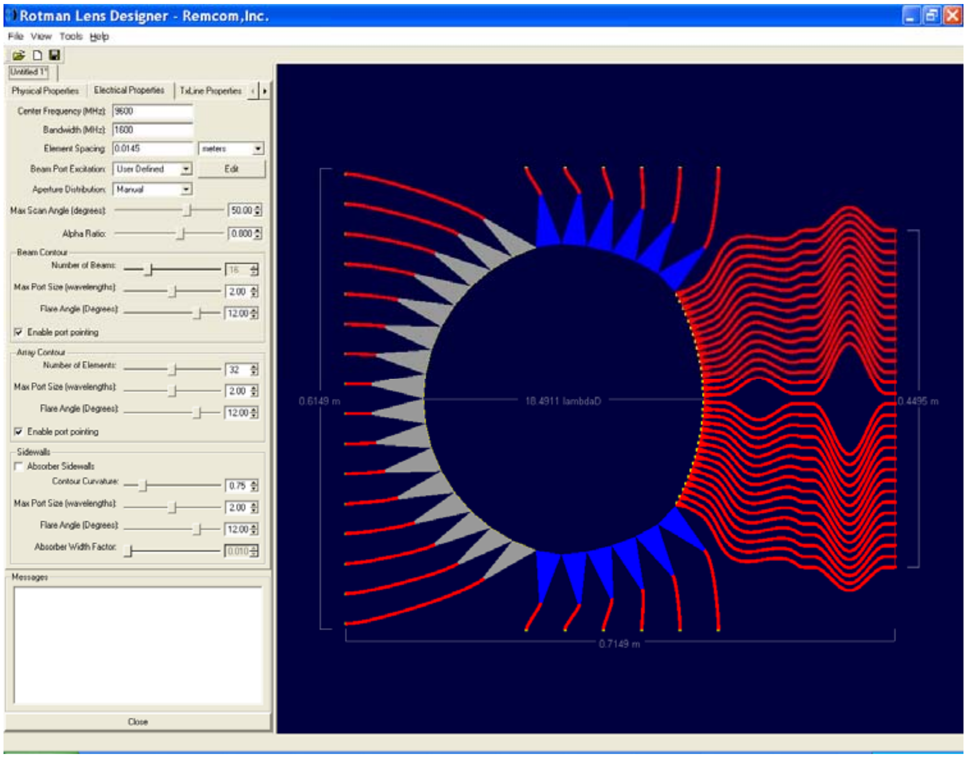
Remcom currently offers a product for quick prototyping of planar microstrip and stripline beamformers of the Rotman Lens type. This product is called Rotman Lens Designer, or RLD for short. It is based on a combination of theoretical design equations and simple geometrical optics which provides rapid design creation and optimization, but leaves at least a little doubt as to the realistic performance of the lens.

The XFdtd 7.0 product is a full wave simulator that is based on the Finite Difference Time Domain (FDTD) method which has been validated for numerous applications. It is a readily-available option for verifying the performance of the lenses developed by the RLD software.

A typical Rotman Lens device can have tens or hundreds of input ports which made the problem set-up quite tedious in earlier versions of the XFdtd software. However, with version 7.0 a sophisticated scripting language, based on Qt Script, is available which makes it possible to replace many time-consuming operations with a simple macro. Each macro can be saved and reused again and again.

For the specific case of the interaction between RLD and XFdtd 7.0, a file loading script has already been written and is available for any user of RLD. This script will read the RLD geometry file description (stored with a *.2ds file extension) and import the entire geometry, and add all port definitions required. Manually performing this task in XFdtd 6.0 could take over an hour. With the script functionality the process is handled by the software with minimal user interaction in less than five minutes.

To demonstrate with an example, consider the X-band Rotman Lens shown in Figure 1 which was designed in the RLD software. This lens has 16 beam ports at the left and 32 array ports at the right. There are also 6 dummy ports at the top and bottom of the lens for absorbing fields at the sidewalls of the geometry to eliminate unwanted reflections. This is a total of 60 ports which each must have a voltage source or resistor attached to them; a fairly significant task.



Center Frequency (MHz): 9600
Bandwidth (MHz): 1600
Element Spacing: 0.0145 meters
Beam Port Excitation: User Defined Edit
Aperture Distribution: Manual
Max Scan Angle (degrees): 50.00
Alpha Ratio: 0.000

Beam Contour
Number of Beams: 16
Max Port Size (wavelengths): 2.00
Flare Angle (Degrees): 12.00
☑ Enable port pointing

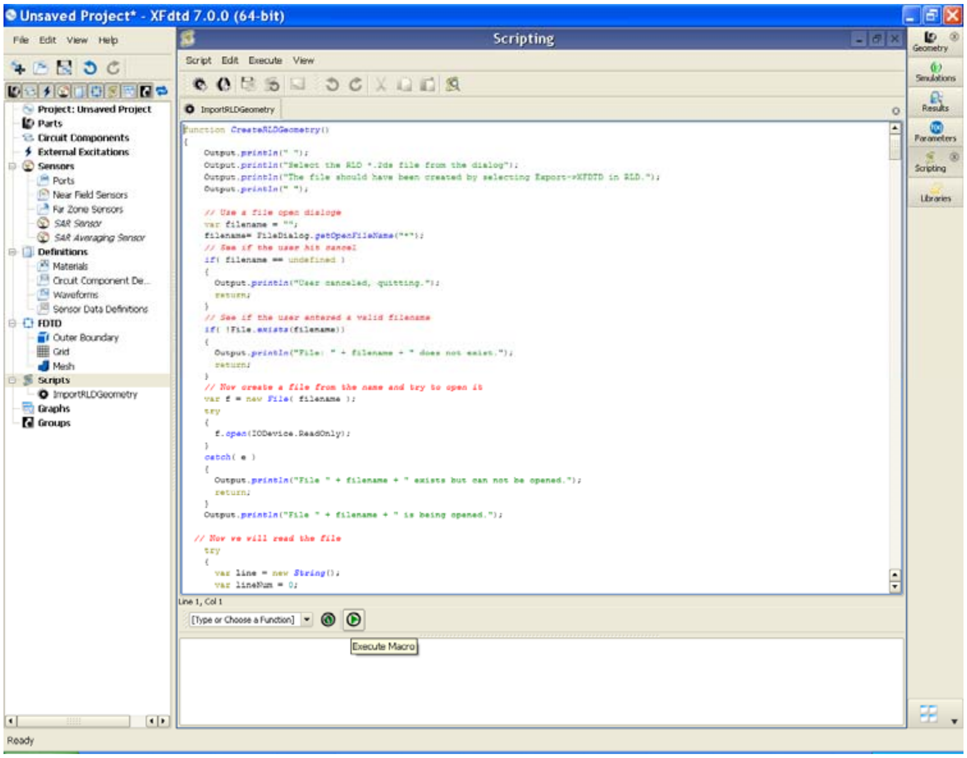
Array Contour
Number of Elements: 32
Max Port Size (wavelengths): 2.00
Flare Angle (Degrees): 12.00
☑ Enable port pointing

Sidewalls
☐ Absorber Sidewalls
Contour Curvature: 0.75
Max Port Size (wavelengths): 2.00
Flare Angle (Degrees): 12.00
Absorber Width Factor: 0.010

Messages

Figure 1: The X-band lens shown in RLD.

Importing this lens into XFdtd 7.0 requires opening the script file and running it. A portion of the script is shown in the script editor in XFdtd 7.0 in Figure 2. By pressing the "Play" button, the script will begin execution. The script will query the user for the filename of the exported RLD geometry file. Once the file is selected, the script will begin adding each geometry piece to the XFdtd Parts list. For each part entered, a material is also created by the script and assigned to the part, saving the user from performing this task. After all parts are added, the script will prompt for the number of ports of each type (beam, array, and dummy) as shown in Figure 3. Note also in Figure 3 the partially-visible listing of parts at the left of the screen and the geometry in the background. The script will proceed to enter in all the ports with the selected component type. By default the component added is a voltage source with a series resistor, but this setting may be changed in the script. In Figure 4, one of the ports is shown selected with the location and component settings added by the script shown. The script will also set a default waveform to be used by the ports.

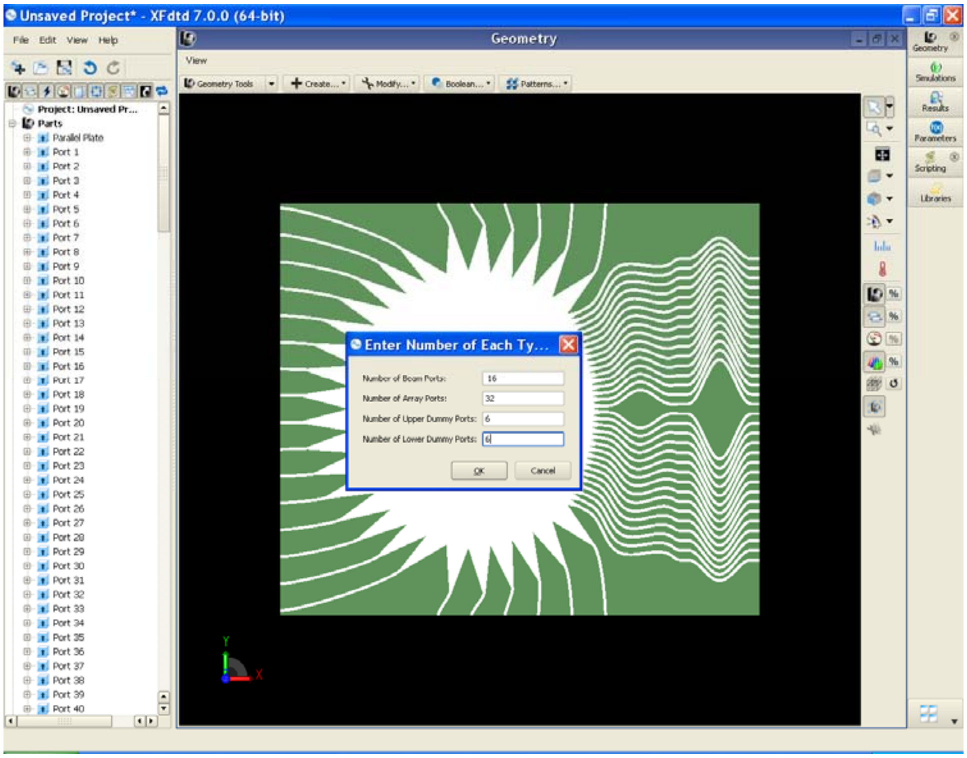


```
function CreateRLDGeometry()
{
    Output.println(" ");
    Output.println("Select the RLD *.2ds file from the dialog");
    Output.println("The file should have been created by selecting Export->XFDTD in RLD.");
    Output.println(" ");

    // Use a file open dialoge
    var filename = "";
    filename= FileDialog.getOpenFileName("*");
    // See if the user hit cancel
    if( filename == undefined )
    {
      Output.println("User canceled, quitting.");
      return;
    }
    // See if the user entered a valid filename
    if( !File.exists(filename))
    {
      Output.println("File: " + filename + " does not exist.");
      return;
    }
    // Now create a file from the name and try to open it
    var f = new File( filename );
    try
    {
      f.open(IODevice.ReadOnly);
    }
    catch( e )
    {
      Output.println("File " + filename + " exists but can not be opened.");
      return;
    }
    Output.println("File " + filename + " is being opened.");

  // Now we will read the file
    try
    {
      var line = new String();
      var lineNum = 0;
```

Figure 2: XFdtd 7.0 includes a comprehensive scripting capability. Here the script editor is shown with a portion of a script for importing RLD geometries.



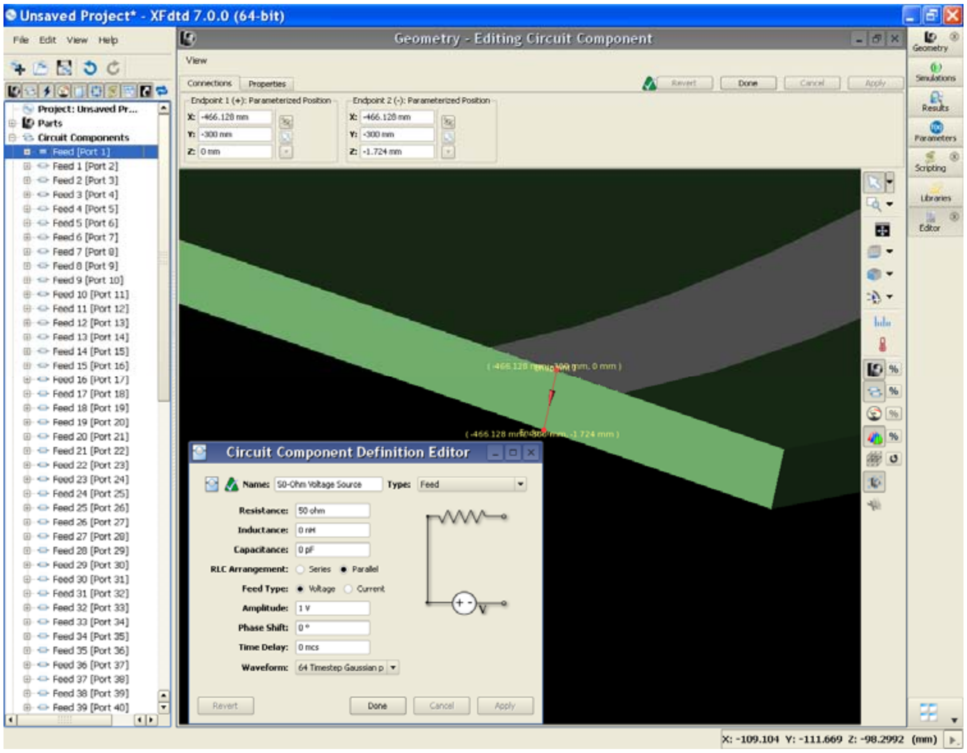
Number of Beam Ports: 16

Number of Array Ports: 32

Number of Upper Dummy Ports: 6

Number of Lower Dummy Ports: 6

Figure 3: Here the imported geometry is visible in the background. The menu in the foreground is asking for the number of each type of port before adding them to the project.



Endpoint 1 (+): Parameterized Position
X: -466.128 mm
Y: -300 mm
Z: 0 mm

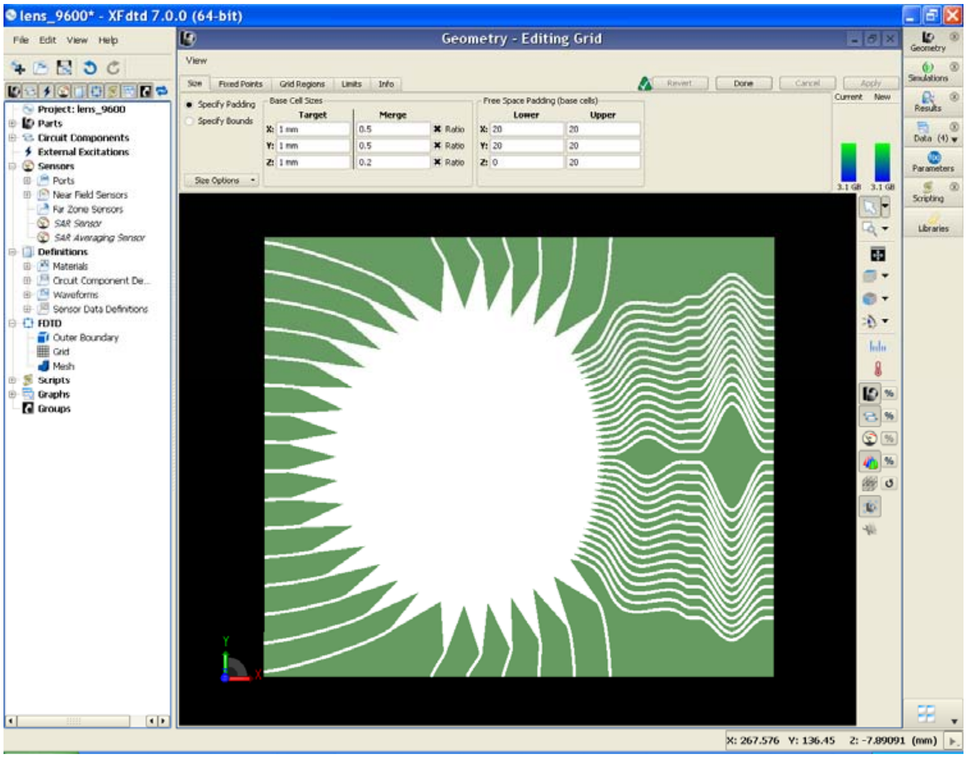
Endpoint 2 (-): Parameterized Position
X: -466.128 mm
Y: -300 mm
Z: -1.724 mm

Name: 50-Ohm Voltage Source
Type: Feed
Resistance: 50 ohm
Inductance: 0 nH
Capacitance: 0 pF
RLC Arrangement: ○ Series ● Parallel
Feed Type: ● Voltage ○ Current
Amplitude: 1 V
Phase Shift: 0 °
Time Delay: 0 mcs
Waveform: 64 Timestep Gaussian p

X: -109.104 Y: -111.669 Z: -98.2992 (mm)

Figure 4: For each port location, a circuit component is added automatically by the script. Typically the component is made up of a voltage source with a series resistor, but these options may be modified. Here one of the ports is selected after import and the component editor is shown to display the component settings.

As the script imports the geometry, adds the materials for each part, adds the ports, and sets the waveform for the simulation, the only remaining step for the user is to create the grid and run the simulation. In Figure 5 the grid definition is shown. For this example the mesh size is set to 1 mm and default spacing between the geometry and the outer boundary of the problem space is used for all faces except under the lens. For the bottom boundary of the lens a perfectly conducting boundary will be used to function as the ground plane. The setting of the PEC boundary is performed with the Outer Boundary Editor under "FDTD" on the Project Tree at the left of the screen. At this point the project should be saved and given a name.



● Specify Padding
○ Specify Bounds

Size Options

Base Cell Sizes

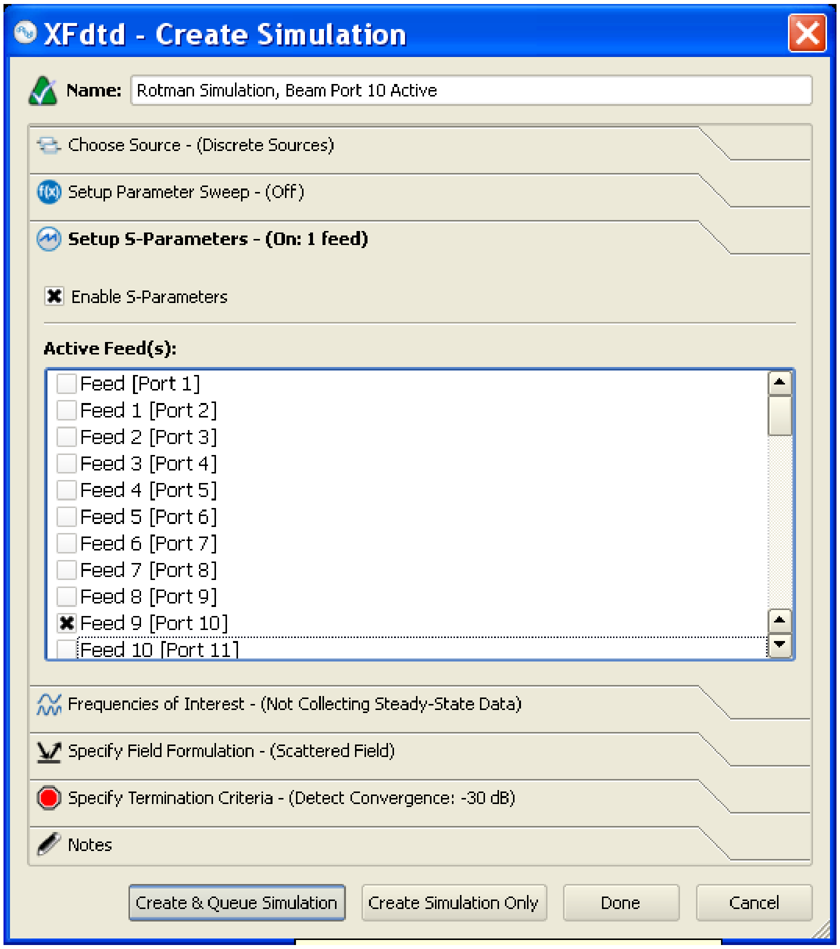
	Target	Merge	
X:	1 mm	0.5	✖ Ratio
Y:	1 mm	0.5	✖ Ratio
Z:	1 mm	0.2	✖ Ratio

Free Space Padding (base cells)

	Lower	Upper
X:	20	20
Y:	20	20
Z:	0	20

Figure 5: Creating the grid for the imported geometry involves setting the mesh dimensions and the padding on each boundary. In this case, the lower boundary padding under the lens is set to zero so that a Perfect Conductor may be used there to serve as the ground plane.

To create a simulation, select the Simulation menu at the right of the screen and press New Simulation. An example is shown in Figure 6. At the top, a descriptive name should be added to differentiate this simulation from others run later. Further notes may be added at the bottom of the screen to record various details of the simulation. In this case, beam port 10 will be selected as the active port for the simulation. Convergence settings for determining the completion of the run are set under Specify Termination Criteria. At this point, the simulation may be started by pressing "Create & Queue Simulation" or the necessary files may be written without starting the simulation by pressing "Create Simulation". If the project is going to be run on a remote computer, the "Create Simulation" option should be chosen.



XFdtd - Create Simulation

Name: Rotman Simulation, Beam Port 10 Active

Choose Source - (Discrete Sources)

Setup Parameter Sweep - (Off)

Setup S-Parameters - (On: 1 feed)

☒ Enable S-Parameters

Active Feed(s):

□ Feed [Port 1]
□ Feed 1 [Port 2]
□ Feed 2 [Port 3]
□ Feed 3 [Port 4]
□ Feed 4 [Port 5]
□ Feed 5 [Port 6]
□ Feed 6 [Port 7]
□ Feed 7 [Port 8]
□ Feed 8 [Port 9]
☒ Feed 9 [Port 10]
□ Feed 10 [Port 11]

Frequencies of Interest - (Not Collecting Steady-State Data)

Specify Field Formulation - (Scattered Field)

Specify Termination Criteria - (Detect Convergence: -30 dB)

Notes

Figure 6: When creating a simulation, a descriptive name may be added to identify the settings. Here the 10th port is set as the active port. The simulation may be started by pressing the Create & Queue Simulation button, or created for export to a remote computer by pressing Create Simulation Only.

After simulation, which takes about one hour on a hardware mini-cluster, the results may be viewed. The software will provide the complete set of S-parameter results for every port in the structure with the selected port active. Some of the S-parameter results versus frequency are shown in Figures 7 and 8. Figure 7 shows the return loss at the input port while Figure 8 shows the S-parameter magnitude received at several of the array ports. For comparison, Figure 9 shows the same S-parameter plots as Figure 8, except this time computed by RLD. Since RLD is based on theoretical equations it does not consider many of the effects such as sidewall reflections that the full wave solutions will. However, the levels of the S-parameter are quite similar to those generated by XFdtd and do serve the purpose of giving the relative performance of the design.

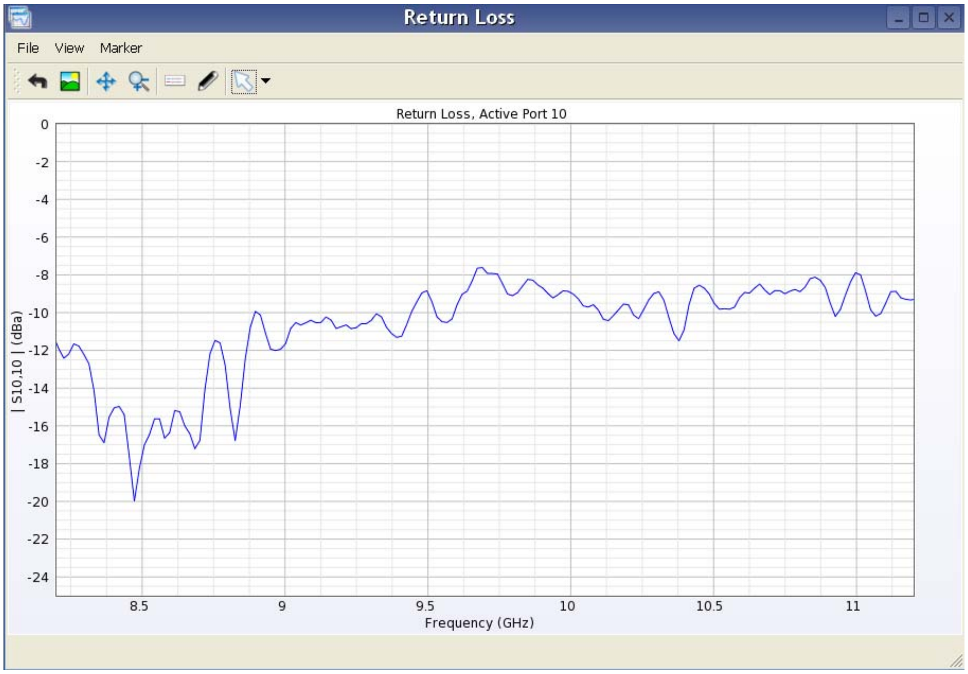


Figure 7: Plotted here is the return loss versus frequency for beam port 10 of the lens, which is one of the middle ports on the left side.

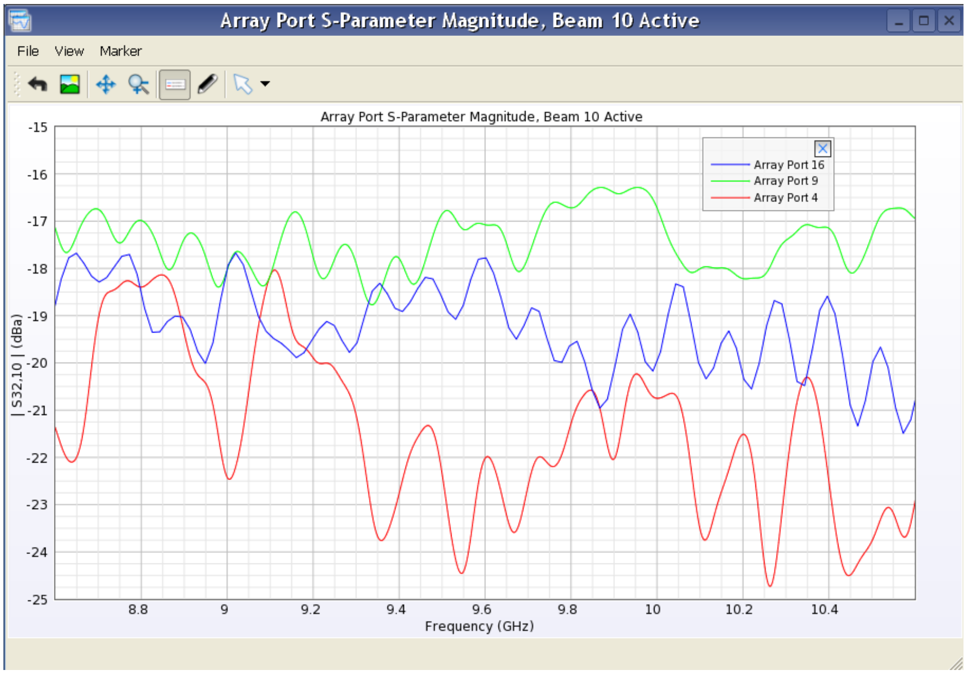


Figure 8: In this plot the S-parameter magnitude for 3 of the array ports is plotted versus frequency for port 10 active.

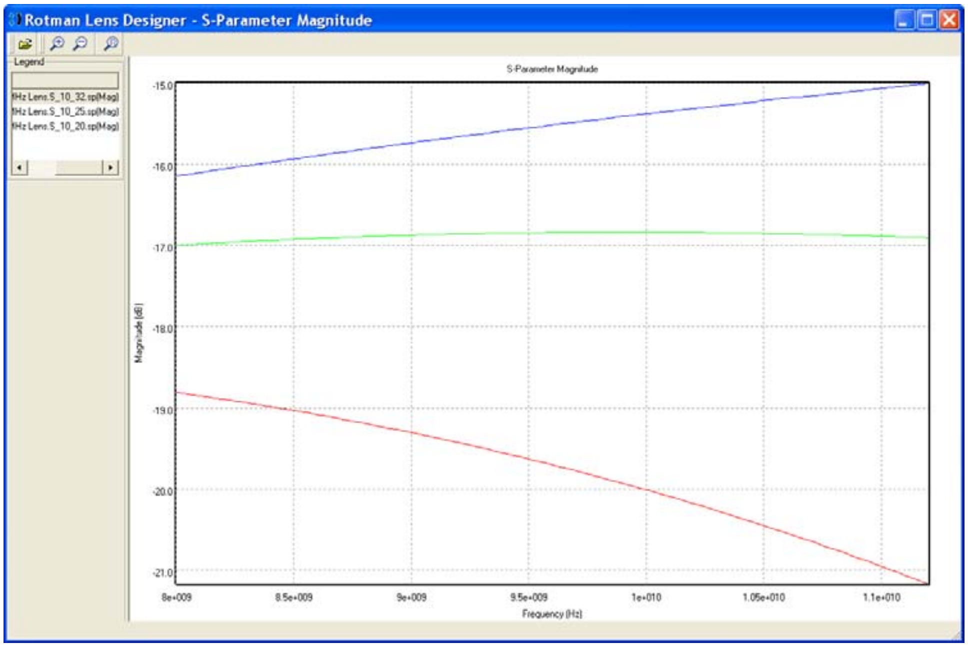


Figure 9: S-parameter magnitudes computed by RLD for the same array ports as in Figure 8. As can be seen the RLD calculation does not consider many of the higher order effects. However, the relative levels of the S-parameter magnitude are near those computed by XFdtd.

The RLD software produces beam patterns which show the relative position and shape of each beam produced by the lens when an individual port is active. Such a pattern is shown in Figure 10 for beam port 10 active. XFdtd 7.0 does not compute the array factor plot directly as a default output quantity. However, since the array factor in RLD is computed based on the complex voltages present at each of the array ports, a script can be used to recreate this plot in XFdtd. The script will collect the time-domain voltage data at each of the output array ports, convert it to the frequency domain through an FFT operation, then select complex voltage values at the center frequency (or another frequency entered by the user) and construct a linear array pattern. The results are saved to a text file and to an XFdtd 7.0 graph. Again, this script is available to any user of XFdtd or RLD who wishes to use it. The resulting graph created by XFdtd is shown in Figure 11. The XFdtd array factor plot has a similar beam location as RLD, but there are differences in the sidelobes generated, as is expected.

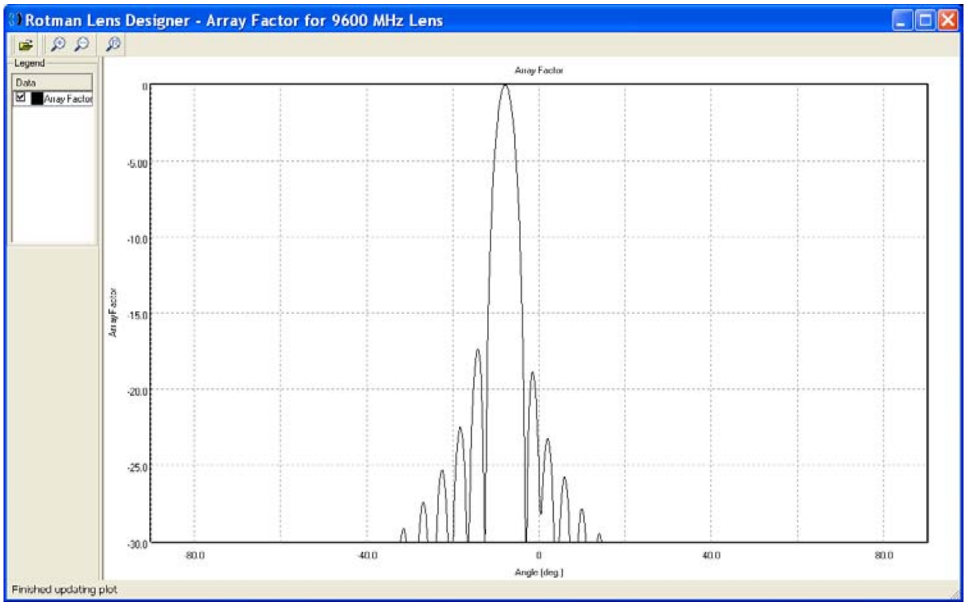


Figure 10: Here the array factor created by having beam port 10 active is shown, as computed by RLD. The beam location is seen to be centered around 8 degrees to the left.

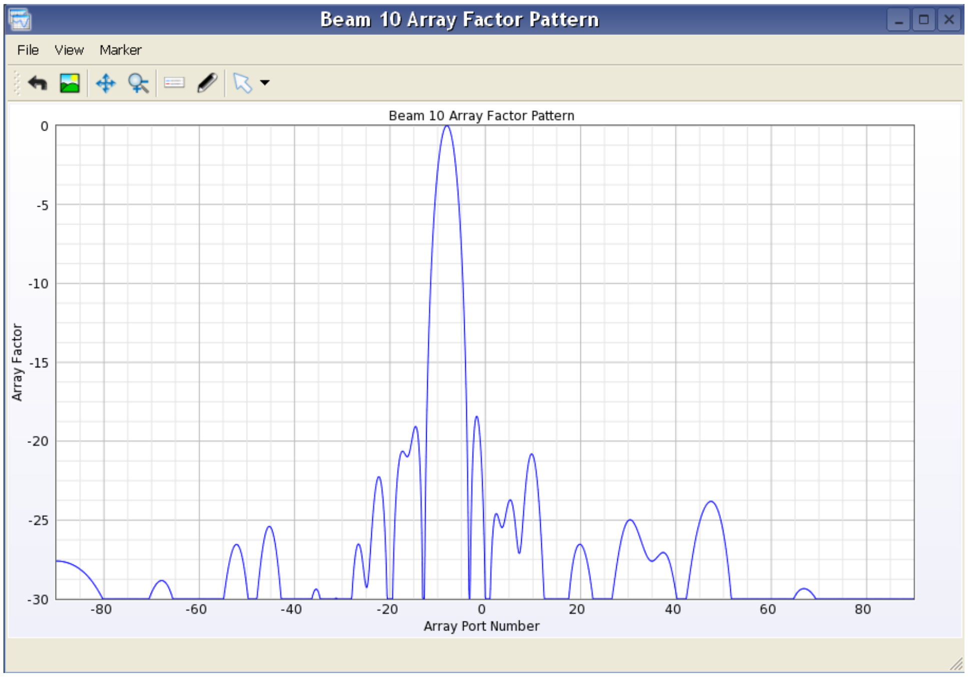


Figure 11: Here the beam 10 array factor is shown after running a script in XFdtd 7.0 to generate it.

A further script was written for XFdtd 7.0 to extract the S-parameter magnitude and phase across the array ports. These plots are of interest since they show the performance of the lens at a single frequency as a function of position across the lens. The resulting output of the magnitude and phase of the S-parameters at the center frequency of the lens is shown in Figures 12 and 13.

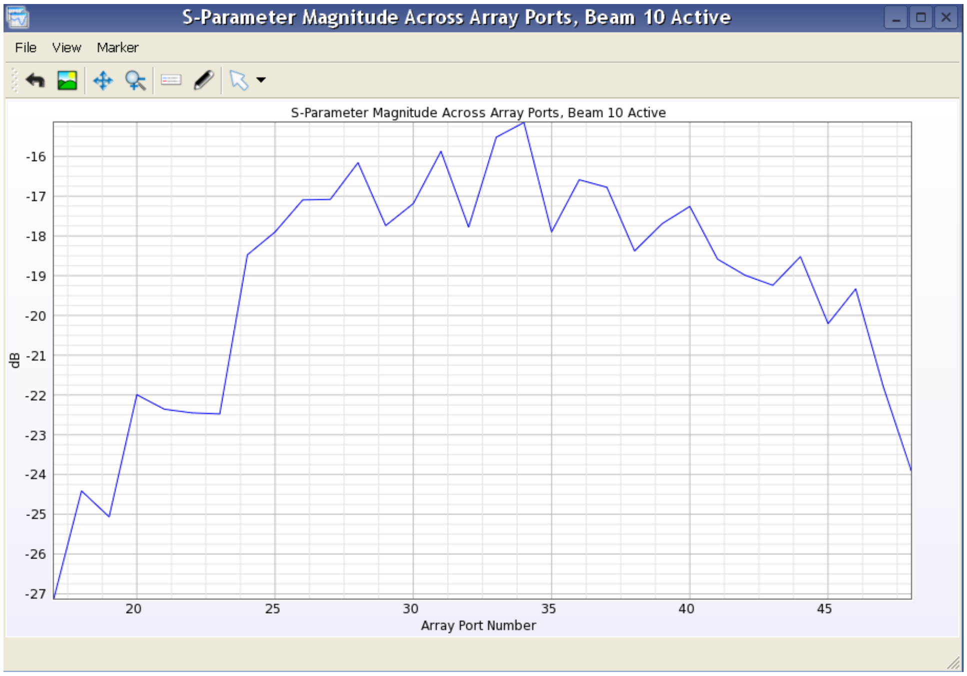


Figure 12: This plot shows the S-parameter magnitude at the center frequency across the array ports as computed by XFdtd. The plot is constructed by a script run in XFdtd that pulls the data from the S-parameter vs Frequency plots of each port.

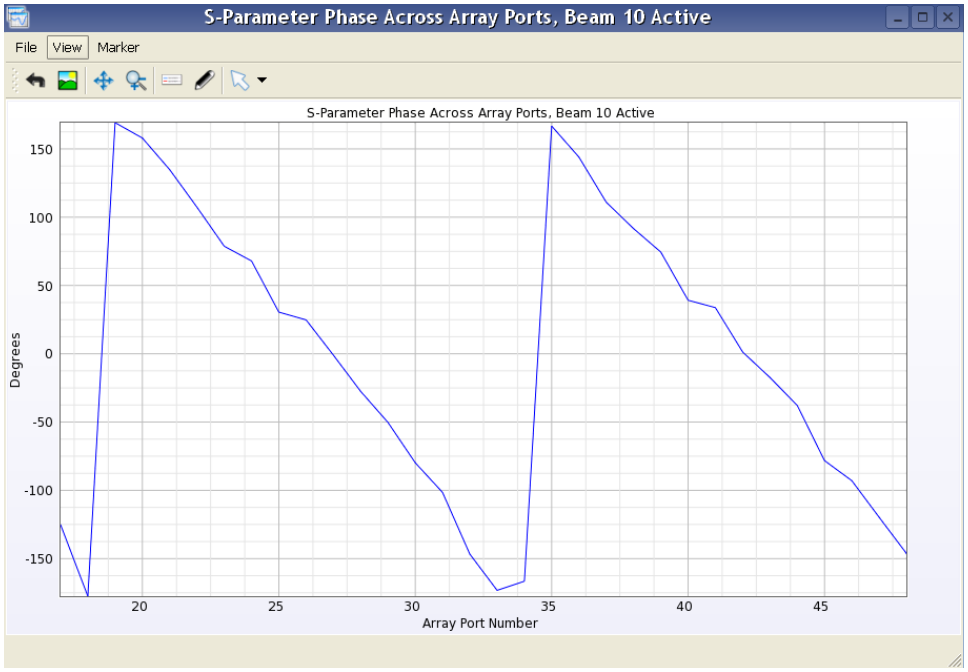

S-Parameter Phase Across Array Ports, Beam 10 Active
File
View
Marker
S-Parameter Phase Across Array Ports, Beam 10 Active
150
100
50
0
-50
-100
-150
Degrees
20
25
30
35
40
45
Array Port Number

Figure 13: Here the phase of the S-parameters is plotted across the array ports at the center frequency. The graph is made by a script in XFdtd that extracts the phase data at each port from the S-parameter data vs. Frequency.

Conclusion

By using scripts in XFdtd 7.0, many difficult or repetitive tasks may be eliminated and replaced by fast-running macros that save a great deal of time and user patience. In this example, a complex Rotman Lens design was imported from the RLD software, set up for simulation in XFdtd 7.0, simulated, and the results analyzed in less than 90 minutes with the simulation time of the full wave solver taking about 60 of those minutes. The actual hands-on time for the user to set up the problem is less than 15 minutes. The use of the scripts to perform post-processing of the results can generate different data formats not supported directly by the software, or can aid in comparing results from different simulations. The end result is a large boost to productivity resulting from the ability to simulate more designs in less time.