PlantSimLab Basic Tutorial:

Running PlantSimLab:

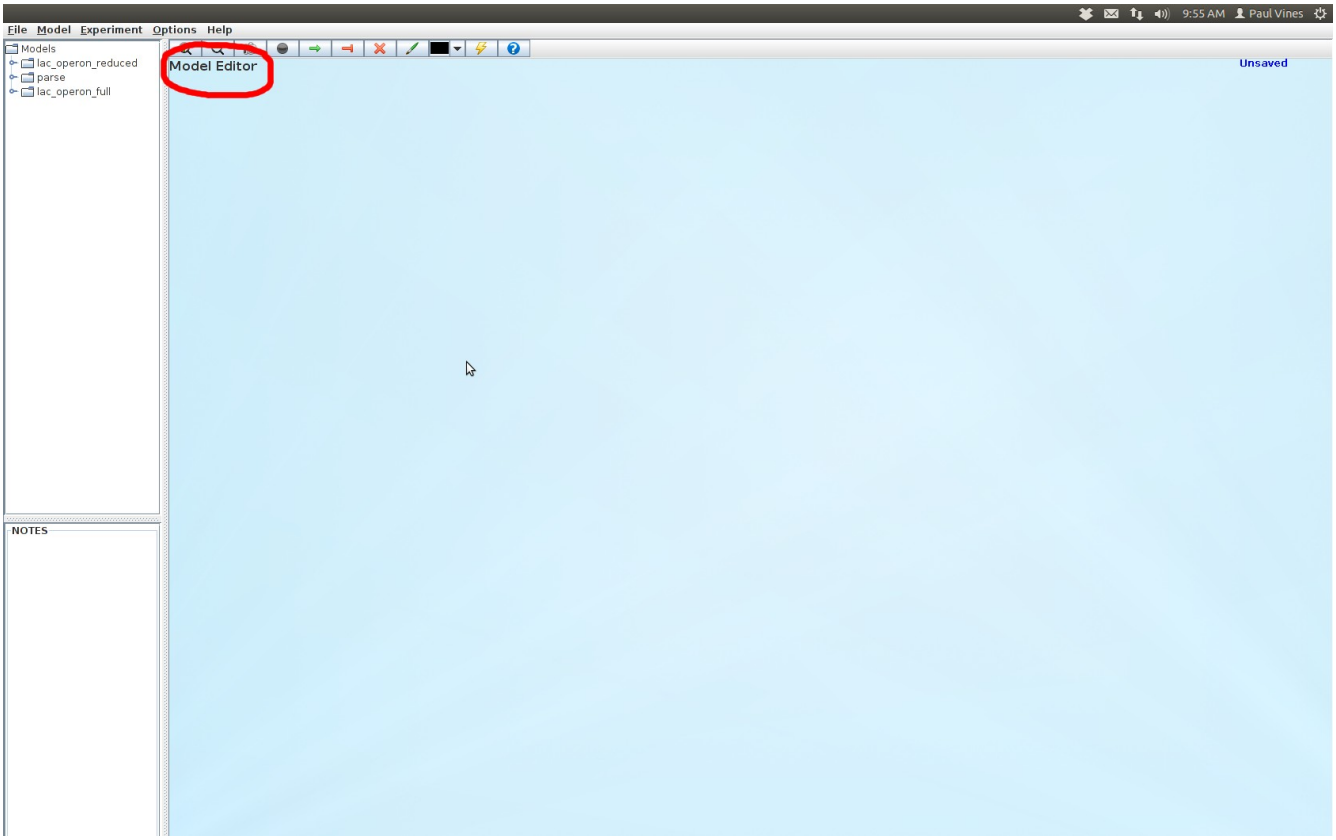
On Windows 32/64bit

- download .zip
- extract (should create a folder "PlantSimLab")
- go into PlantSimLab
- double-click PlantSimLab.jar (this should run it as a java program)
- If this does not work, you may not have java installed on your computer and need to download it.

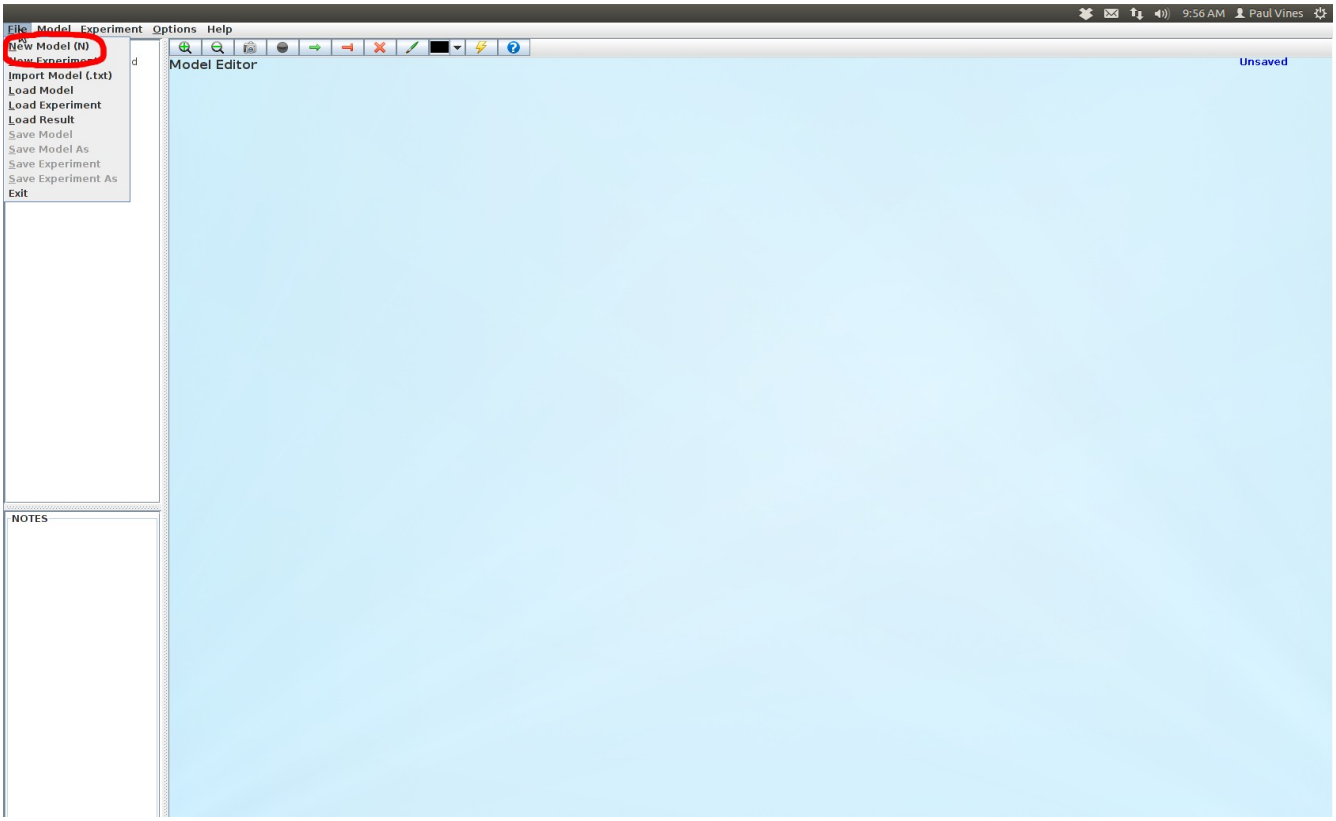
To run on Linux/Mac OS 64bit:

- download .zip
- extract
- go into PlantSimLab
- double-click on PlantSimLab.jar
    - if the interface does not have a blue background:
    - double-click on Run.sh
    - select "Run in Terminal"
    - terminal should pop up saying something about "Make Cyclone" and then the window will pop up a few seconds later
- Otherwise, you may need to install java

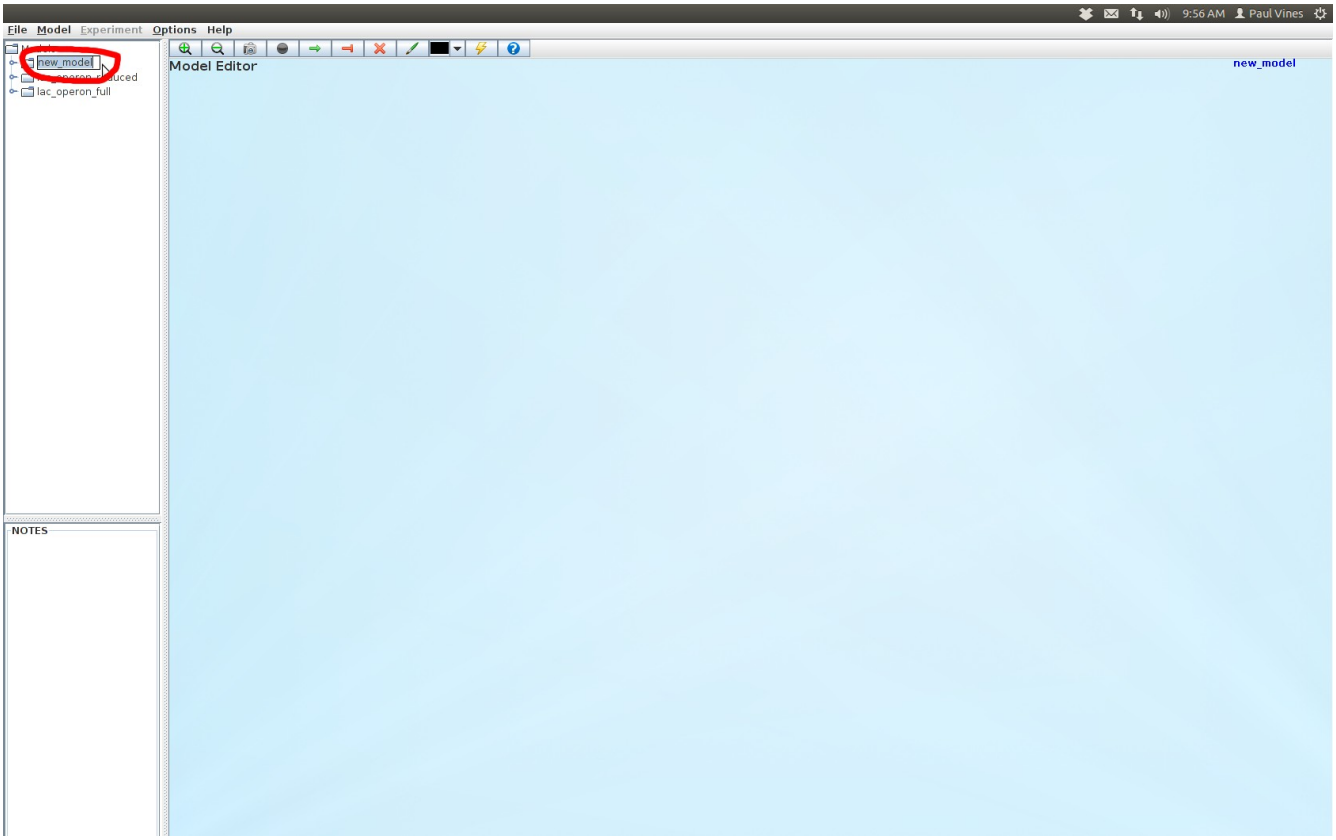
First-Time Walkthrough:



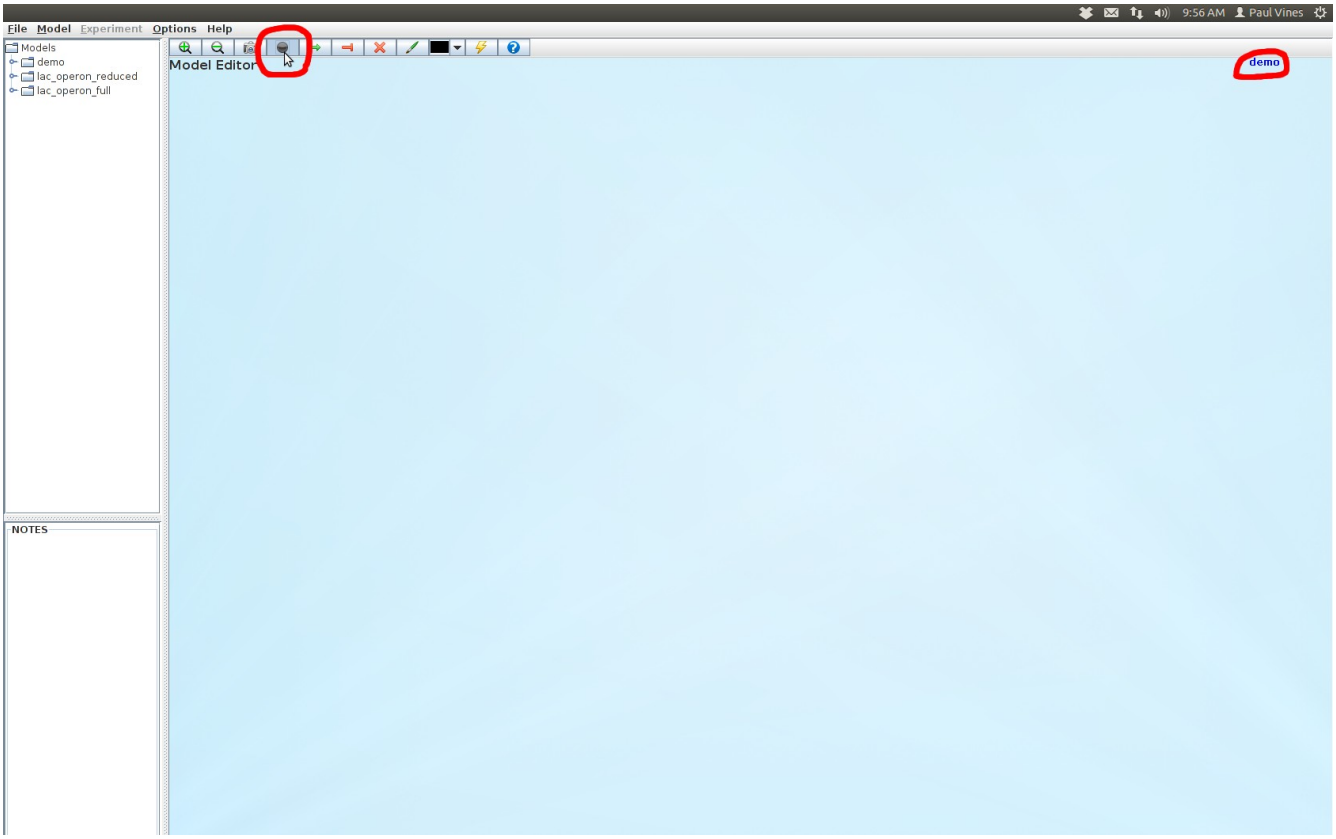
This is the first screen of PlantSimLab, it is the Model Editor panel as labeled in the top-left, and by the blue background. You currently have no models so the left-hand side file-hierarchy is empty except for the empty folder “Models.” Let's fix that:



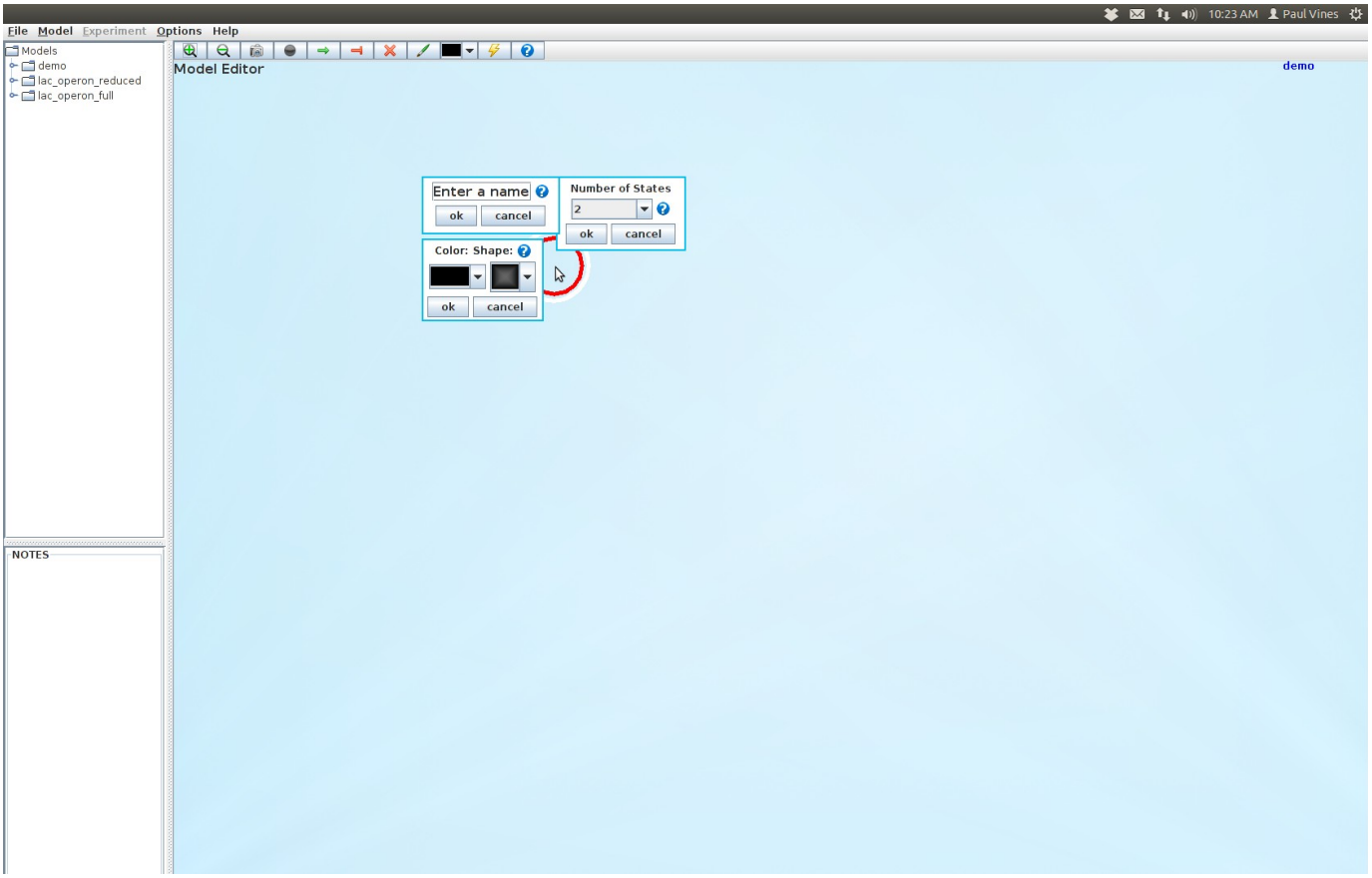
Go to the “File” menu and click on “New Model.”



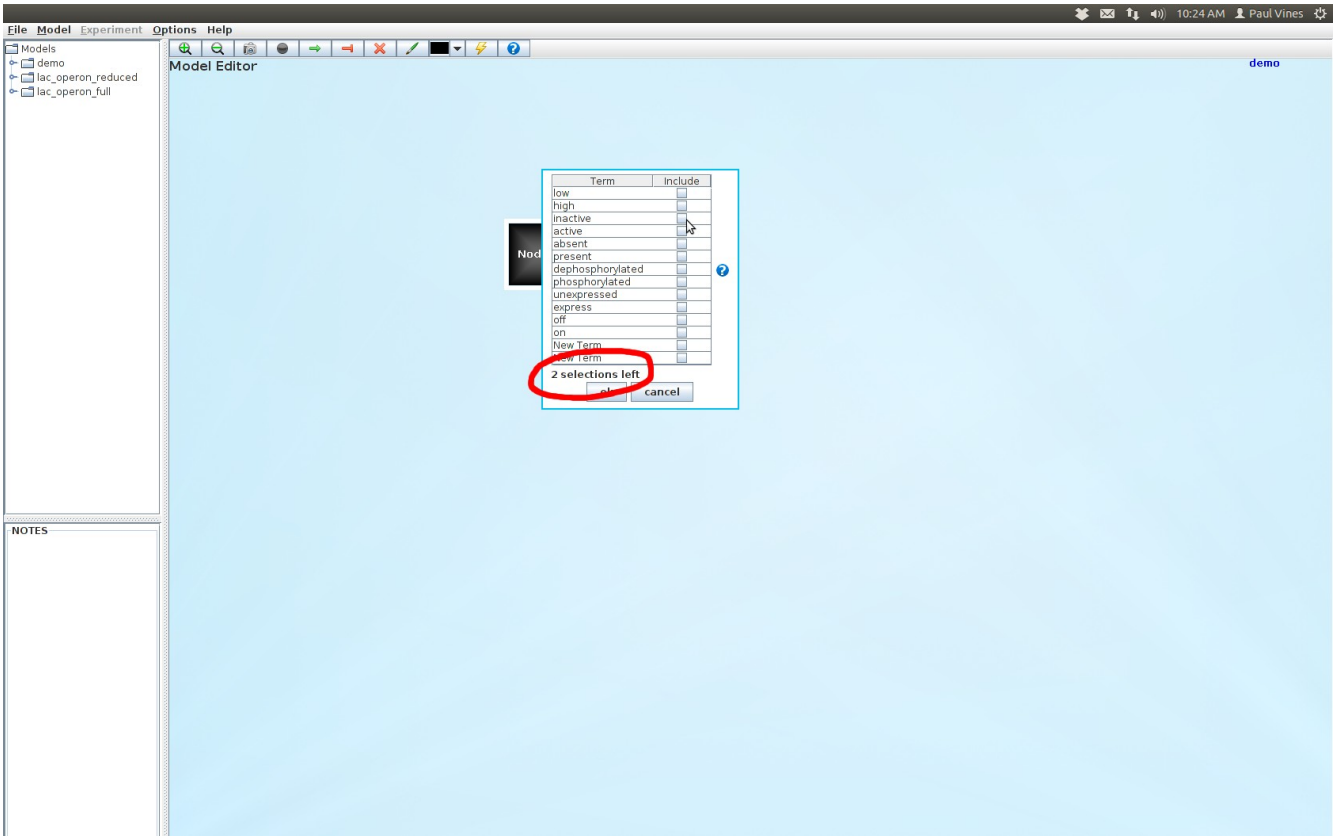
This has created a file called “new_model” on the left, waiting for you to edit its name. Change it to “demo” and hit Enter.



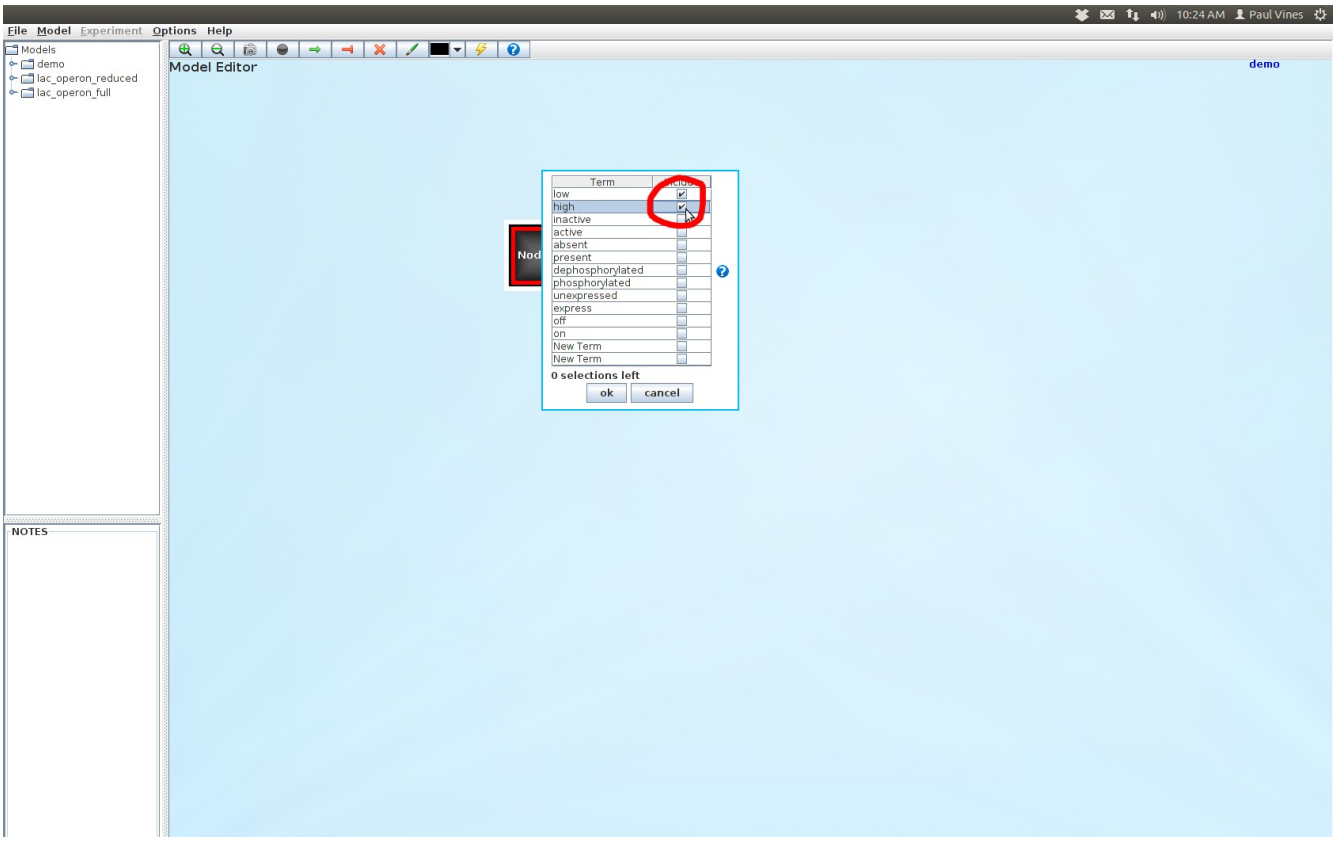
The name has now changed, and you can see you are currently editing the model named “demo” by the name in the top-right of the screen.
Click on the “Add Node” button in the middle of the toolbar, then click anywhere in the blue screen.



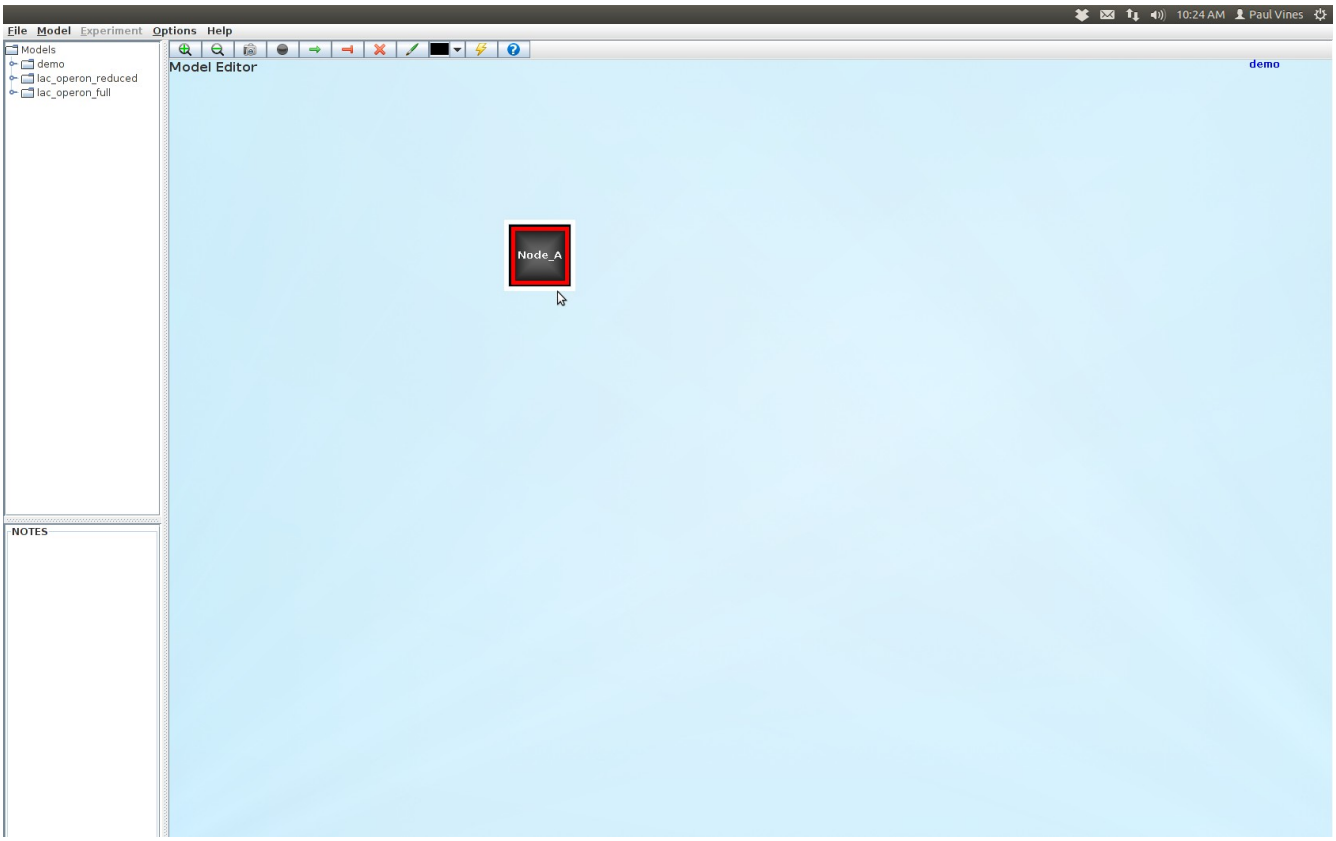
Where you click, it will add a node, popping up 3 panels to ask you to specify characteristics of the node: its name, number of states, and appearance.
Fill in “Node A” for the name
Click “OK” for the color and shape (to make a black square node).
Click “OK” for number of states to make a 2-state node.



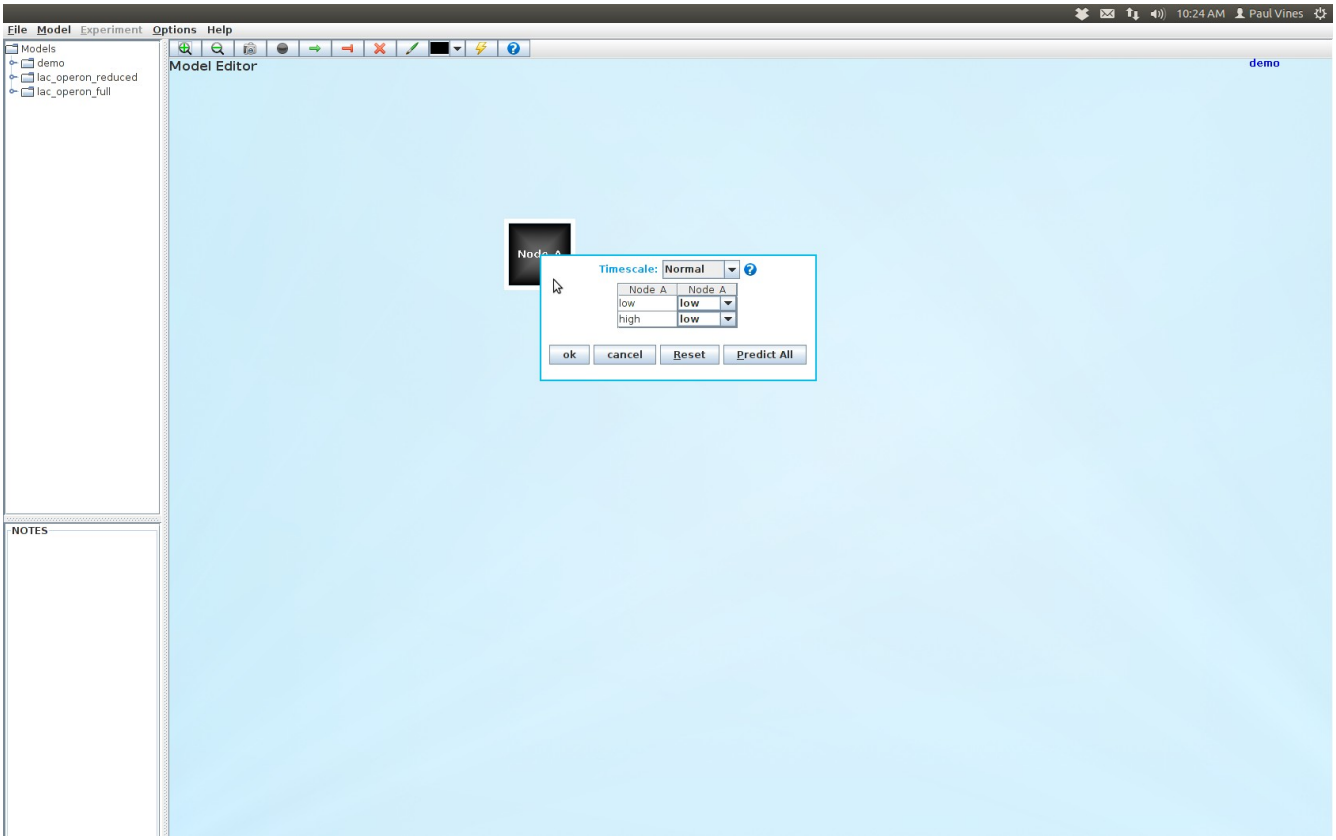
After selecting a number of states, another window will pop-up asking for you to select terms to use for state names. The number of selections required is shown at the bottom.



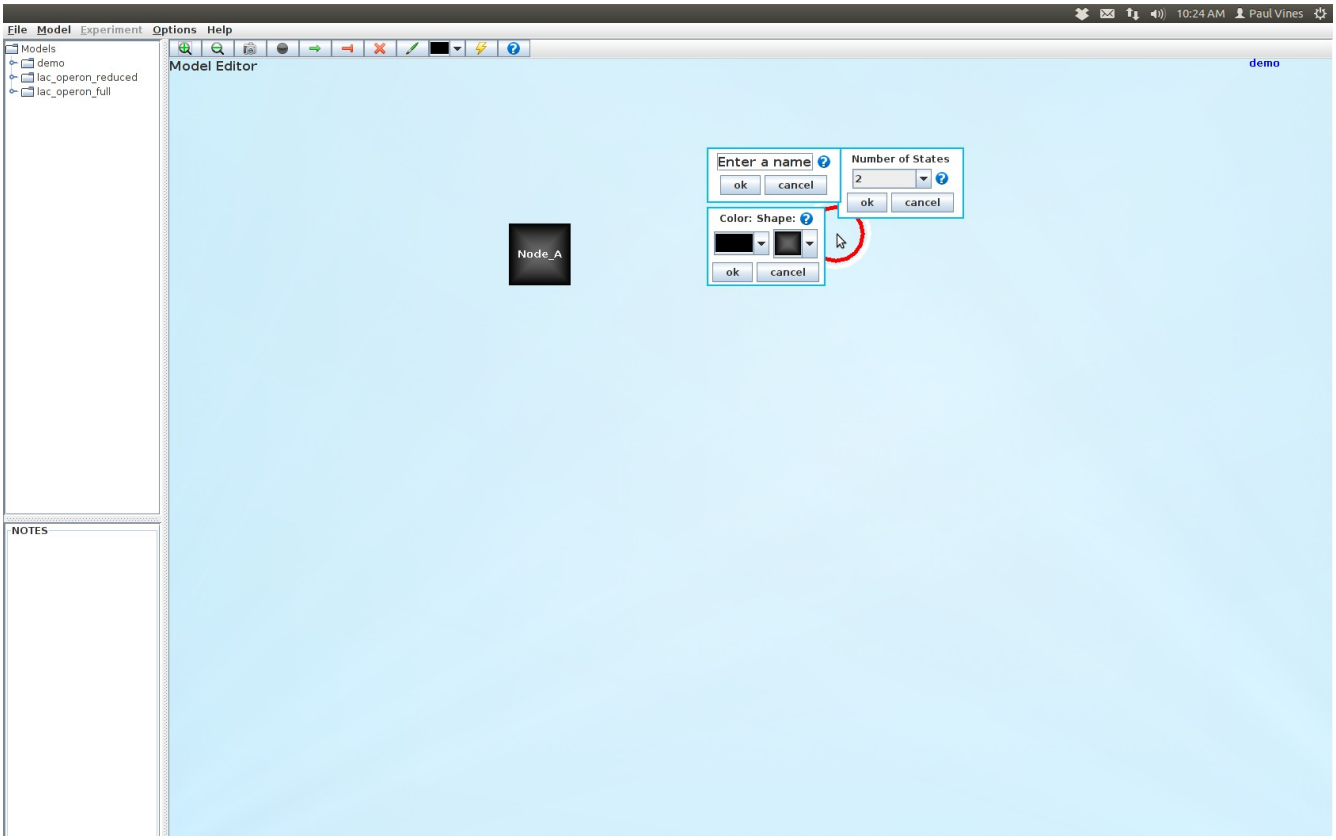
Check “low” and “high” and press “OK.”



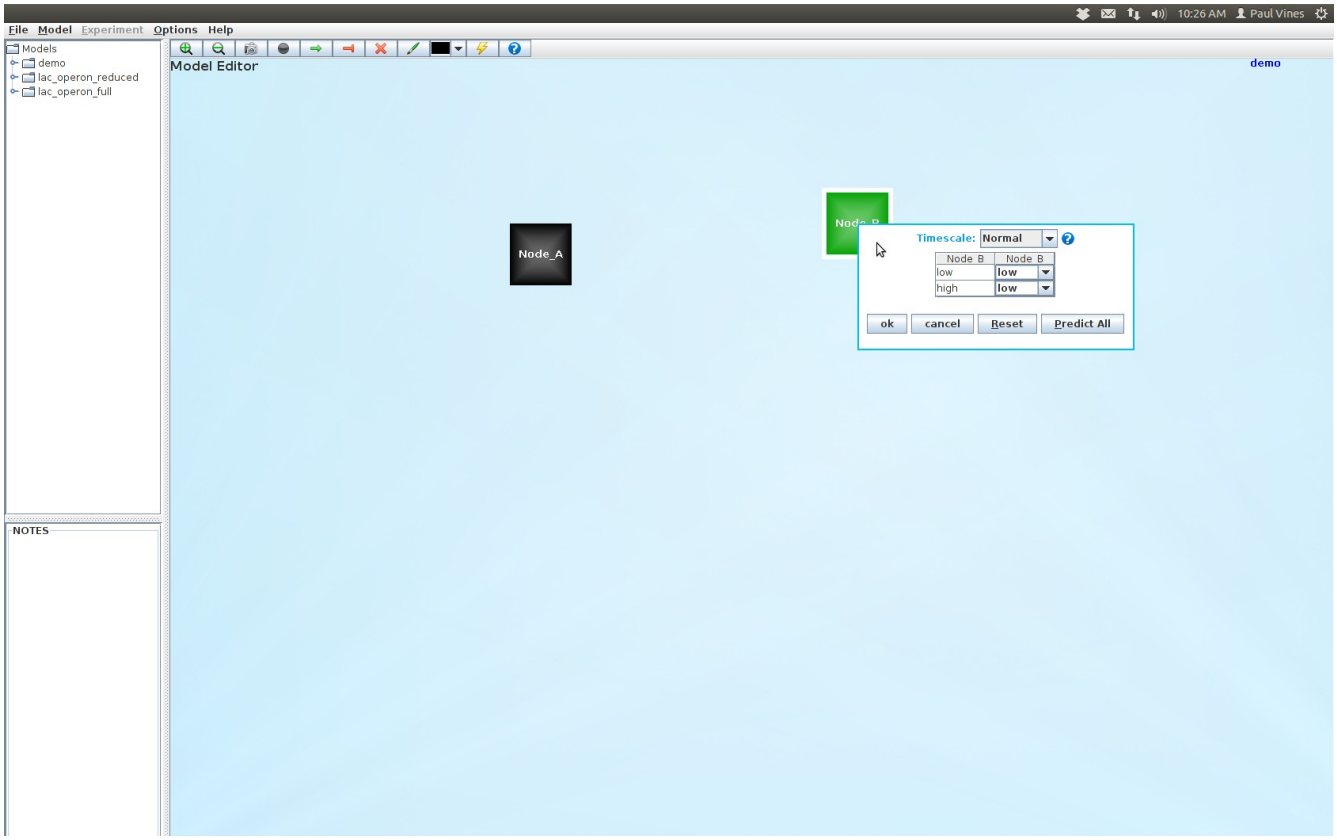
The node is now made, but should be flashing red. This is because its update table has not been created. Click on the node.



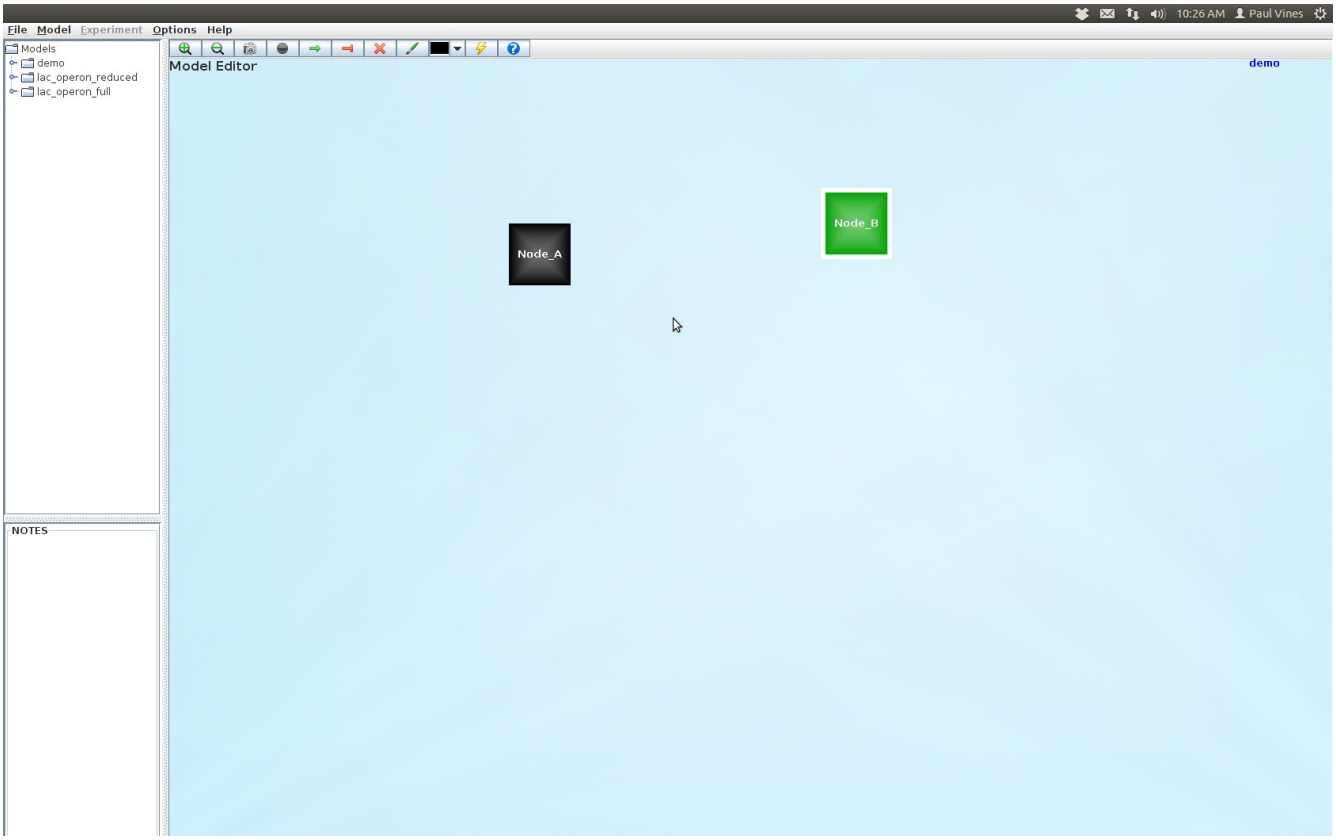
Another window will appear, containing a box labeled “timescale” and a short, two-column two-row table. We will ignore the “timescale” box for now, leaving it at Normal.
The table is the update table for this node, and describes its behavior in the network. The rightmost column shows what state the node will update to, when the cells left of it are true. In this case, there are only two rows because Node A is only affected by what state it is currently in, so you can designate what will happen when it is low, and what will happen when it is high.
Leave it as it appeared for now, and click “OK.”
The node should no longer be flashing red.
Now add another node by doing the same thing.



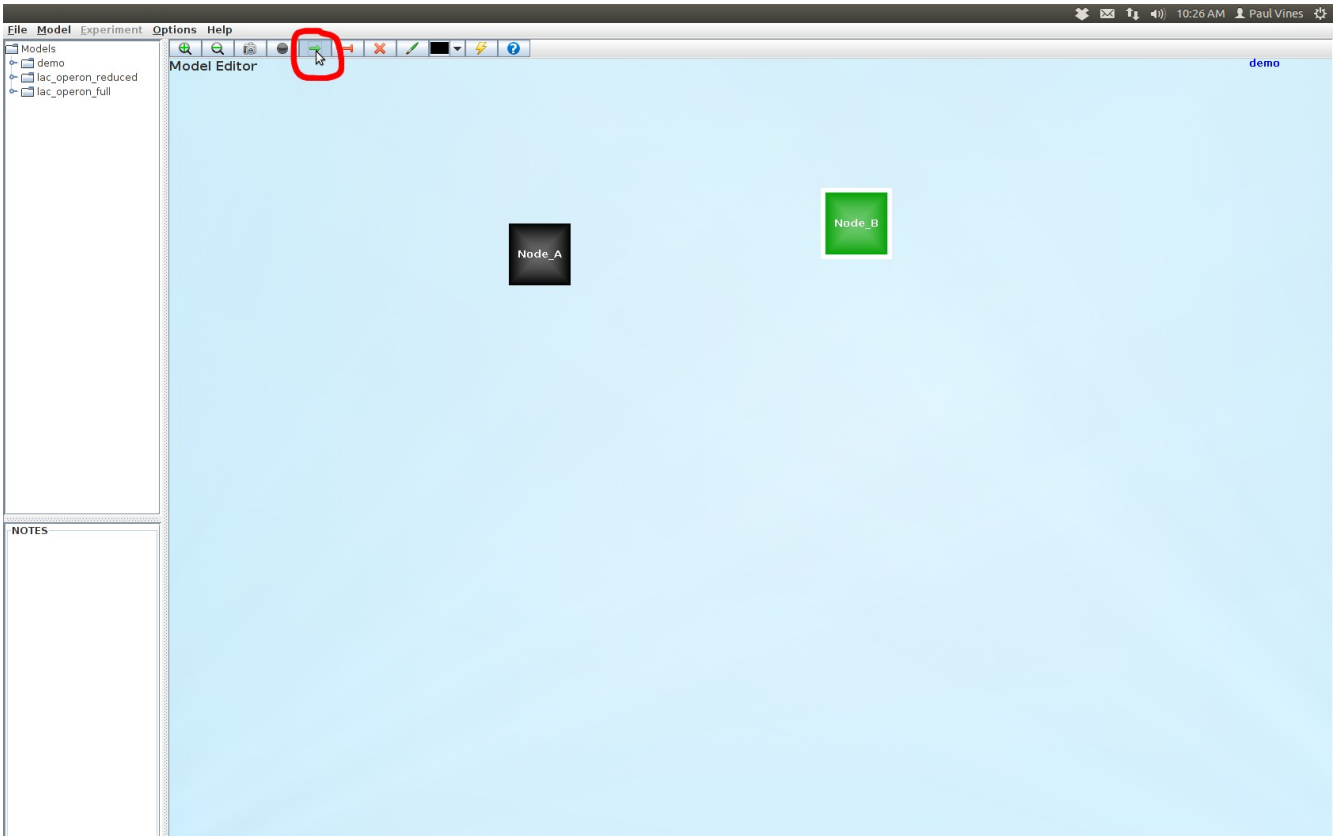
Enter “Node B” for the name, again select 2 states and select “low” and “high.” For color, use the dropdown menu to select green so we can distinguish the two nodes more easily.



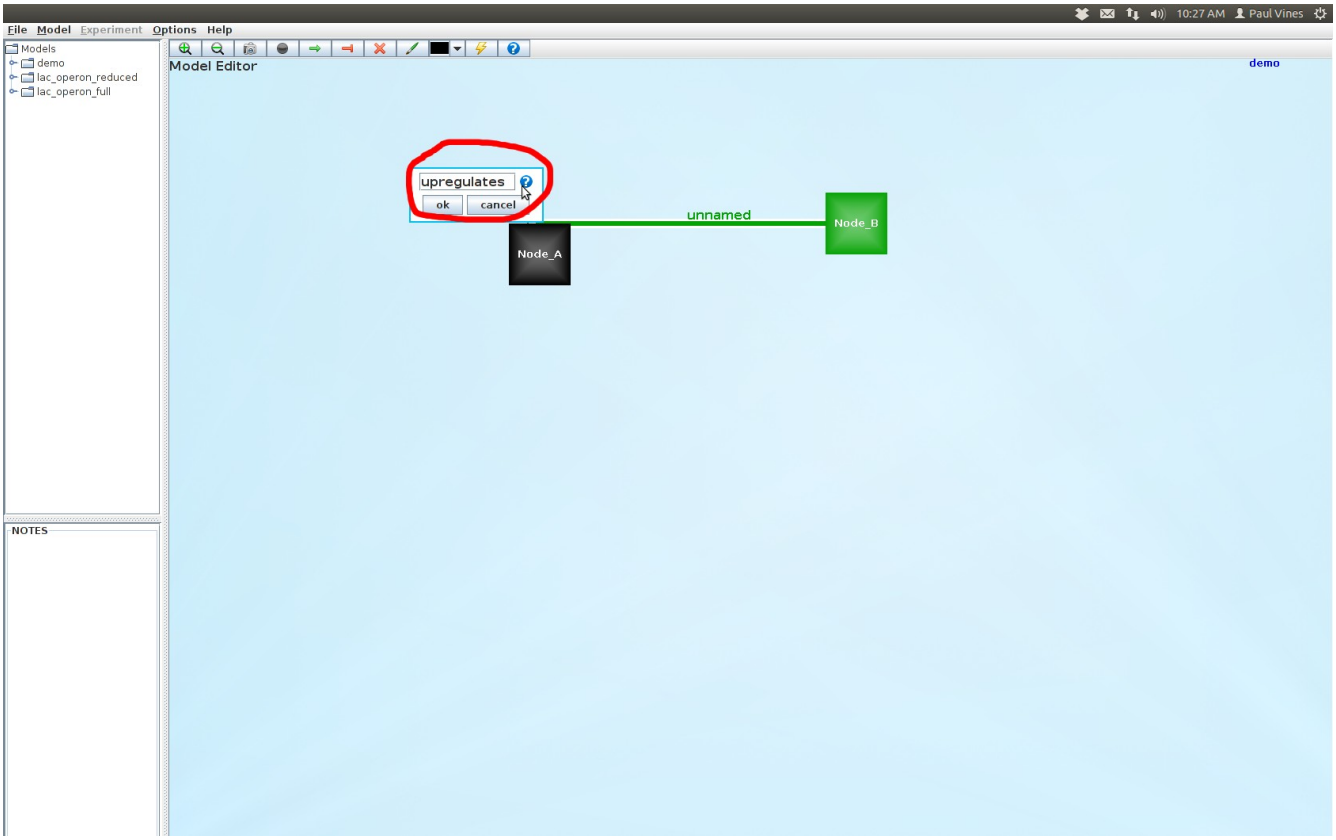
Now click on Node B and approve its table, which should have both low and high resulting in “low” on the right-hand column.



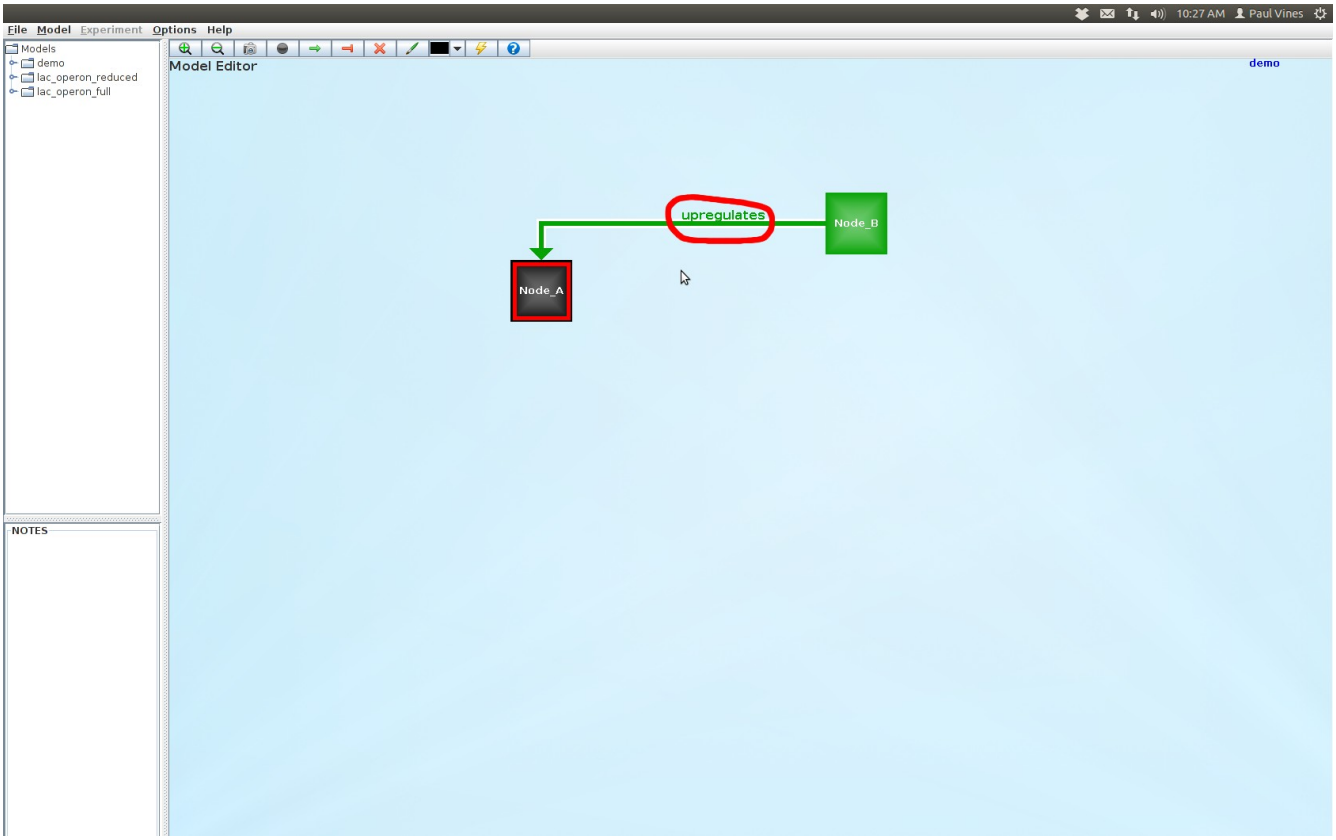
Our model should look like this.



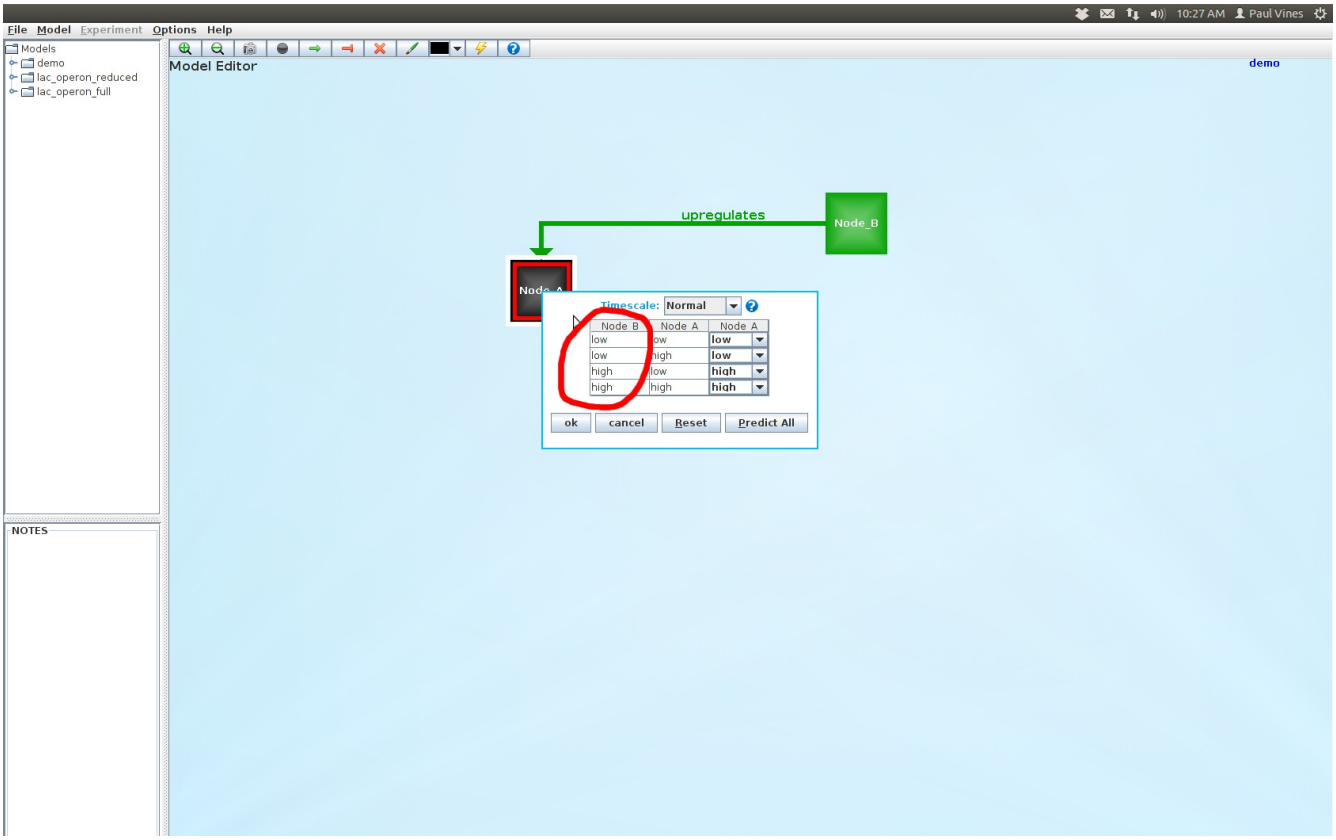
Now we will add an edge describing an interaction. We are going to make it an activating edge (the green arrow) from B to A. So click on the green arrow, then click on node B. Then move the cursor over to node A. As you do so, you should see a green line extending out from node B. Click on node A, and the line will finish and a box asking for a name will appear.



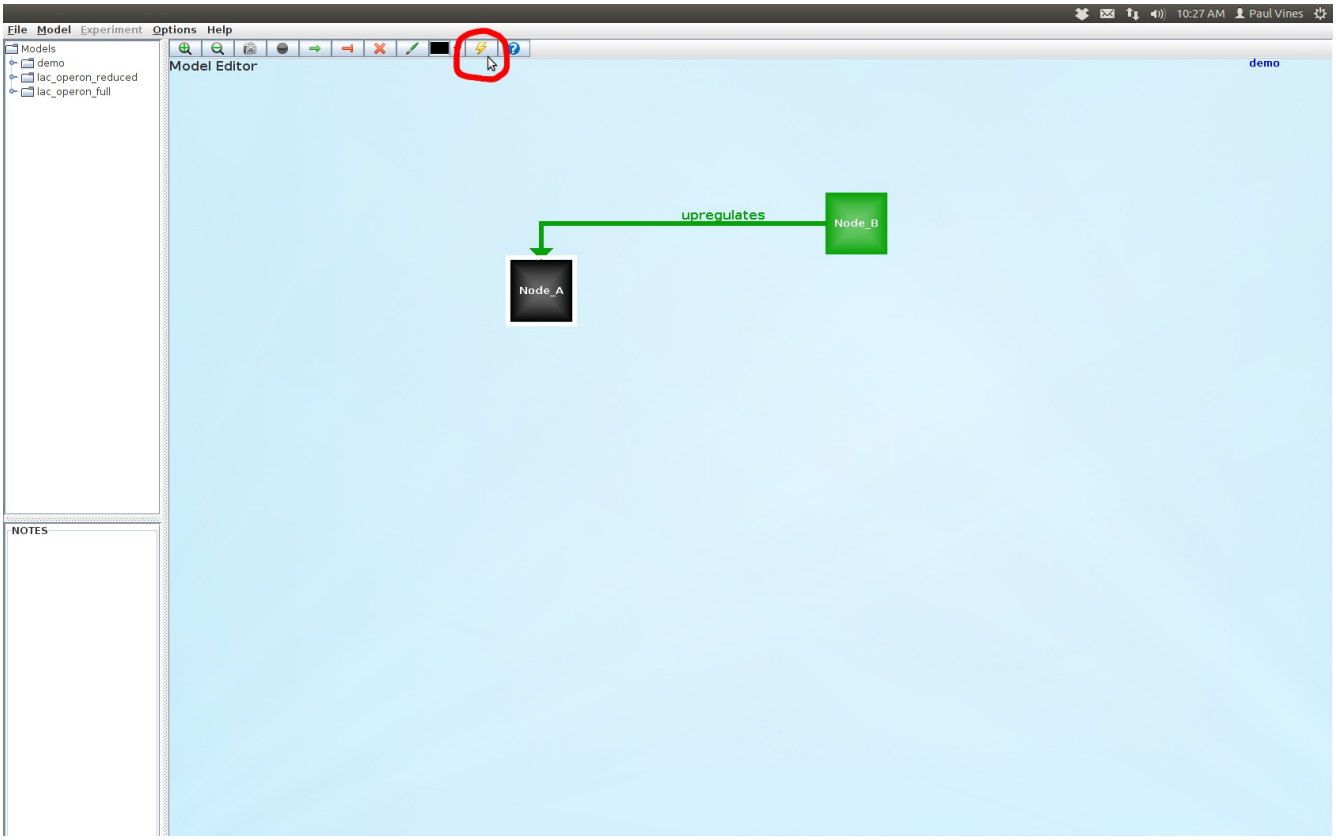
Enter “upregulates” in the box, since that is what this edge will be doing (although the name has no actual effect on the edge's behavior).



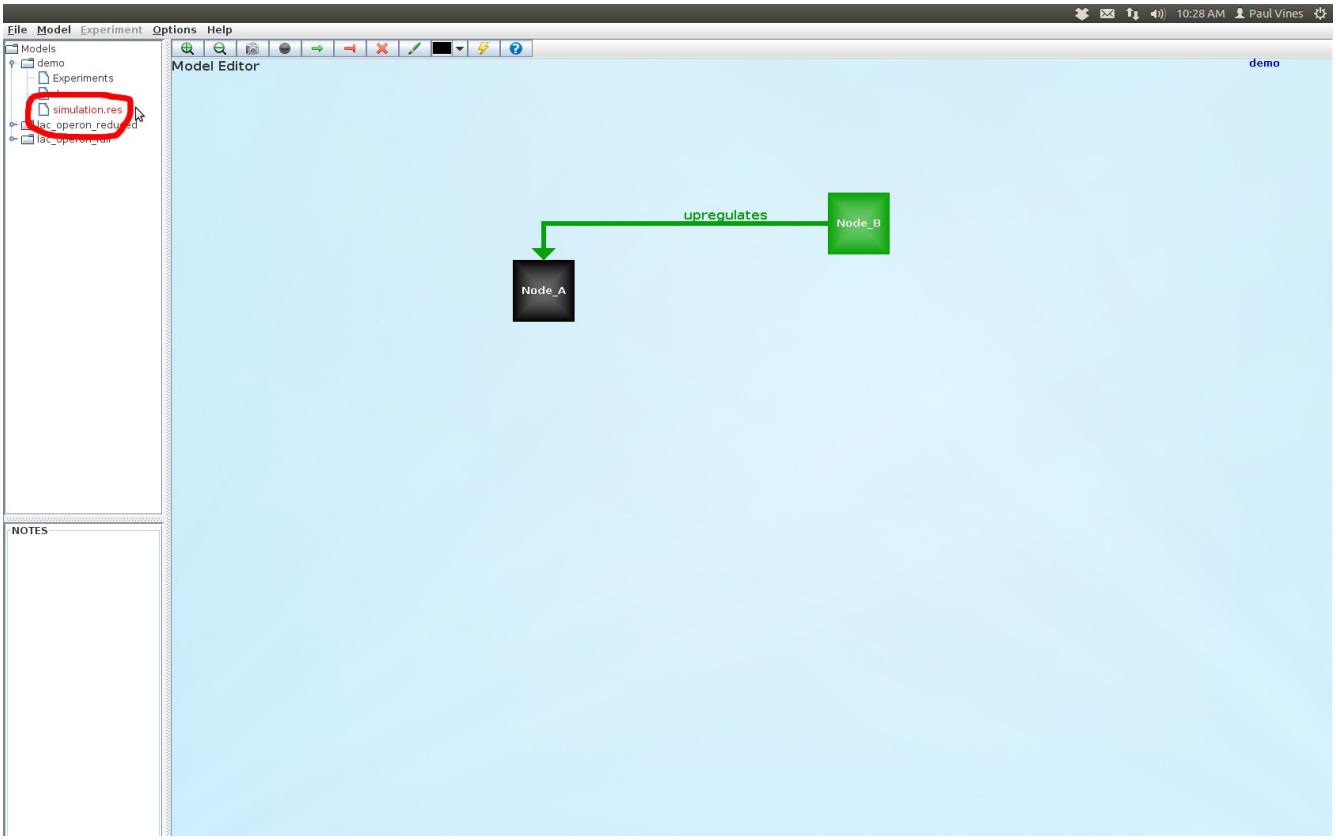
The name of the edge should now be “upregulates.” Also, node A should not be flashing red again, because we added a new input to it, so click on it to edit its table.



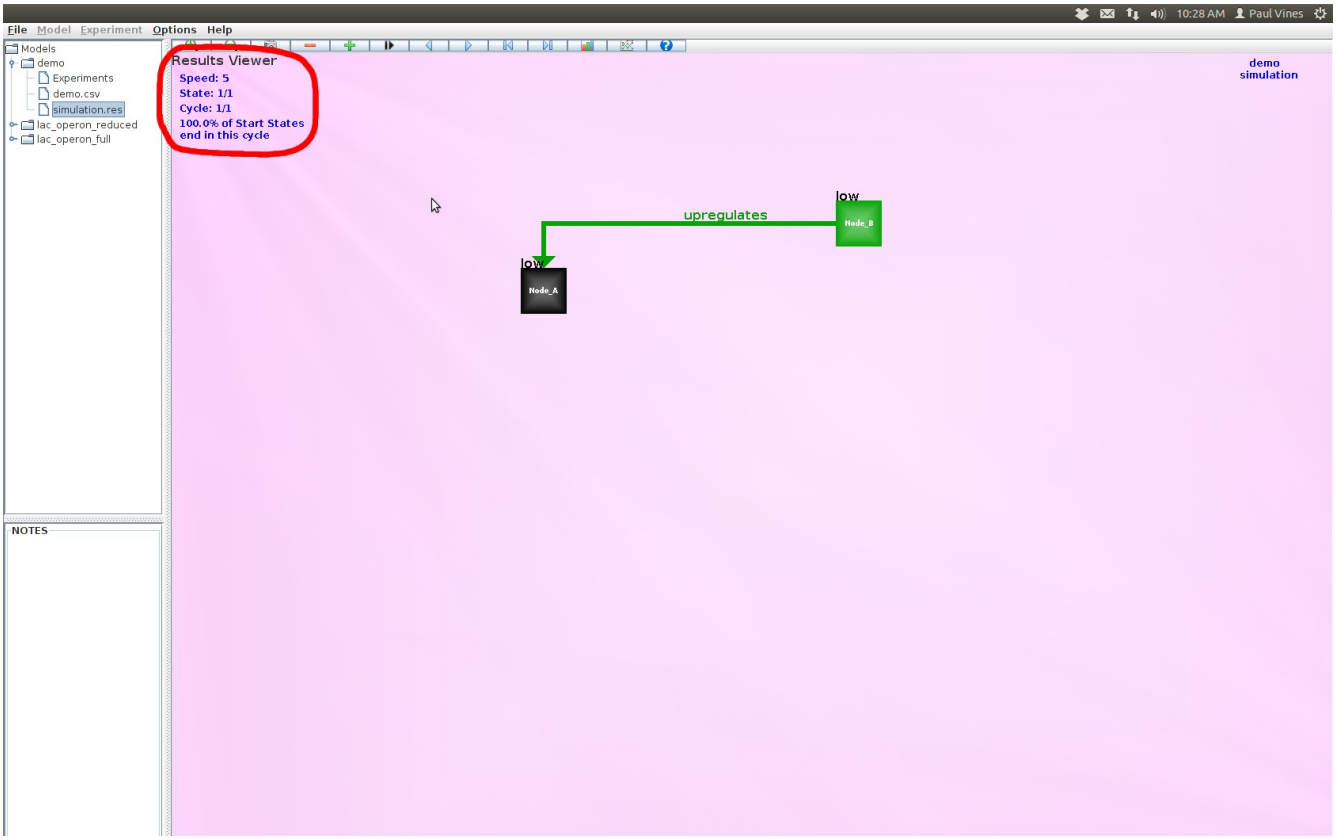
Now we can see there is an additional column, representing the state of node B, as well as two more rows to properly represent all combinations of states node A and B can be in.
Because we chose an activating edge, the program has automatically predicted what we want the table to look like: when B is high, A is activated, so it will become/stay high. When B is low, A is not activated, so it will become/stay low. This is what we want, so simply click “OK.”



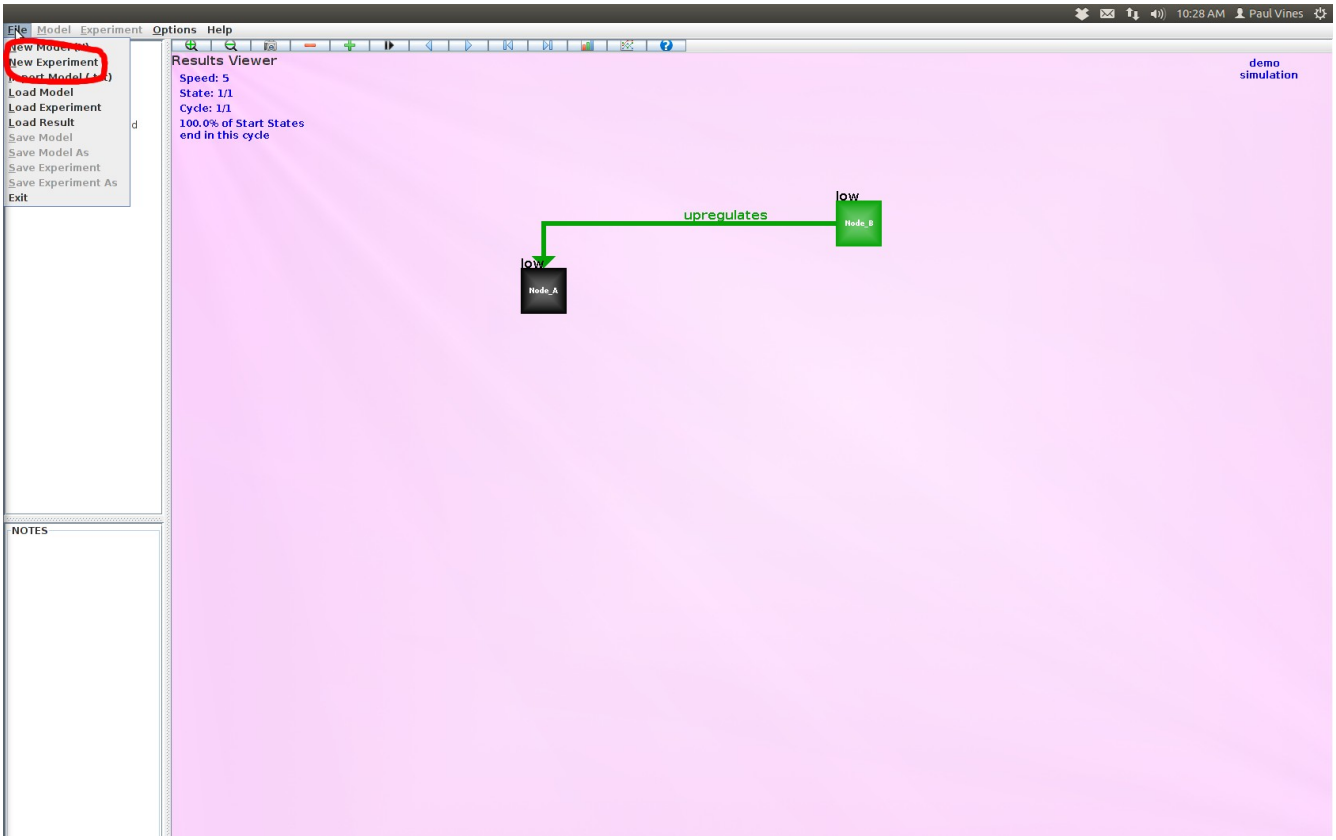
Now we are going to simulate our model to find what its equilibrium points are. Click on the yellow thunderbolt to do this.



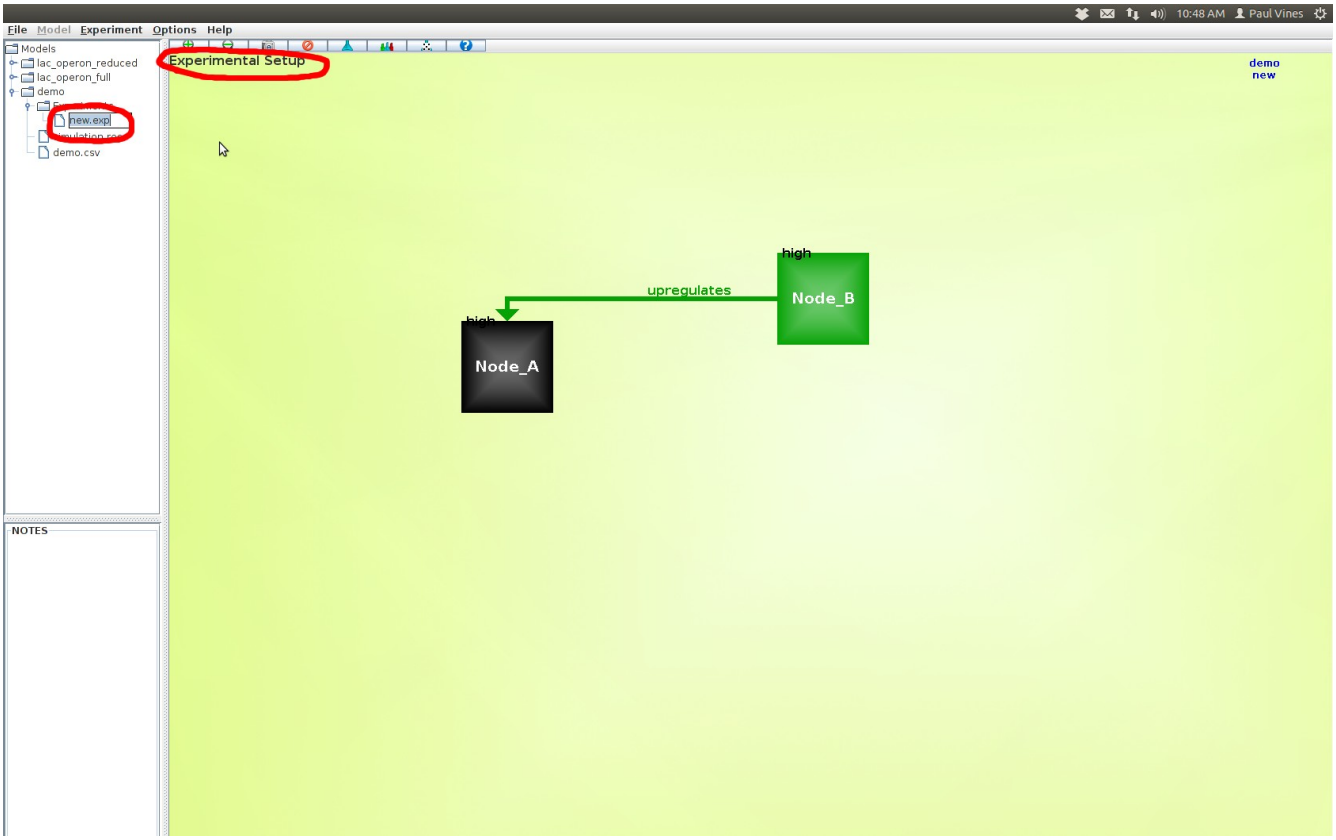
Now if we look over/expand the folder for “demo” we will see a file called “simulation.res” blinking red. It is blinking red because it is new, and it ends in .res because it is a results file. It is the result of our model simulation. Double-click on it to go to the Results Viewer mode.



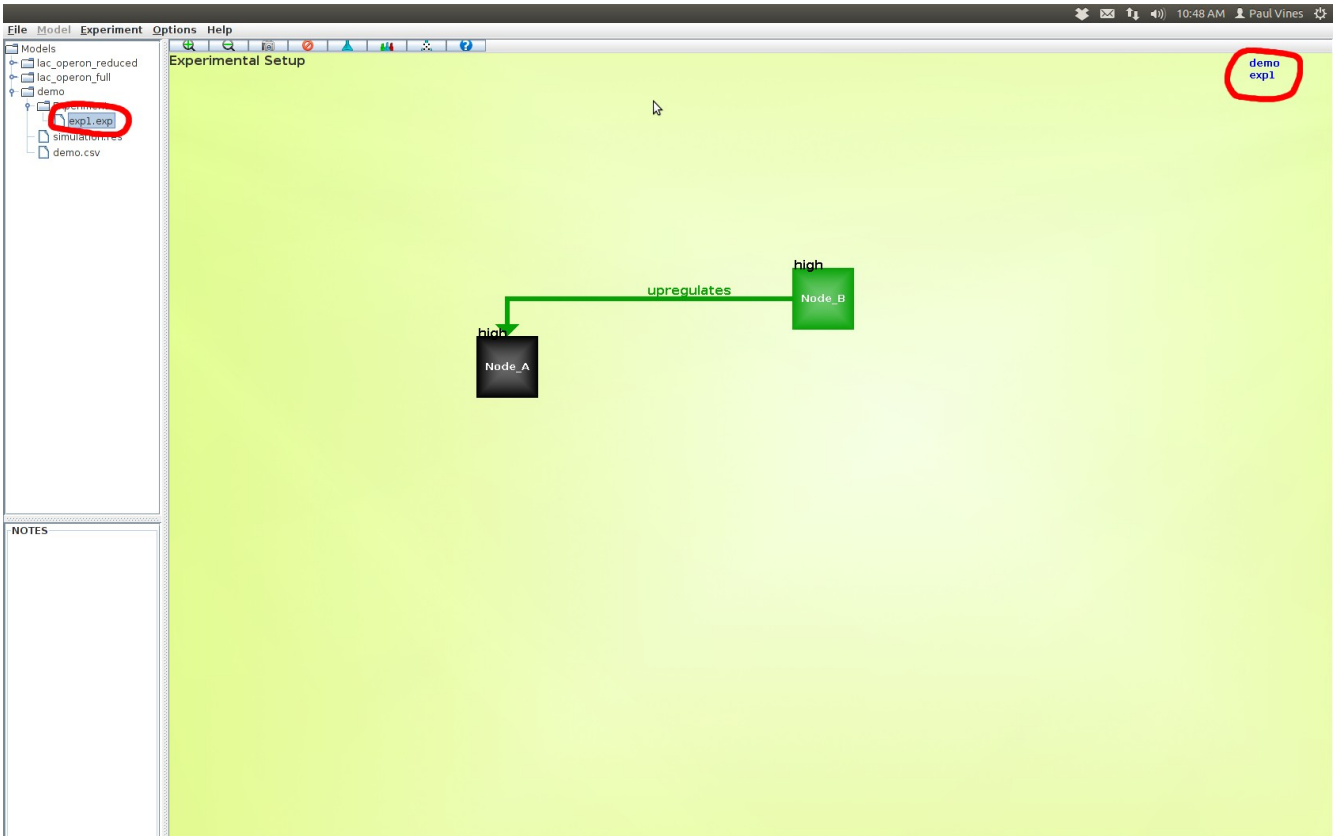
Here we are in Results Viewer mode, as shown by the label in the top-left and the pink background.
The top left contains other data as well, the Speed of display (we will get to that later).
The currently viewed state and number of total states in the cycle.
The currently viewed cycle and number of total cycles in the model.
The percentage of all possible states that wind up in this cycle.
In the middle, we can see out model nodes are now a different size and have words saying they are “low” and “low.” Because cycles says 1/1 and states says 1/1, that means our model only has a single point of equilibrium (which is why 100% of states wind up there) and that equilibrium is a steady state, not an oscillation.
By looking at the levels of the nodes, we can see the steady state for this model is A = low and B = low.



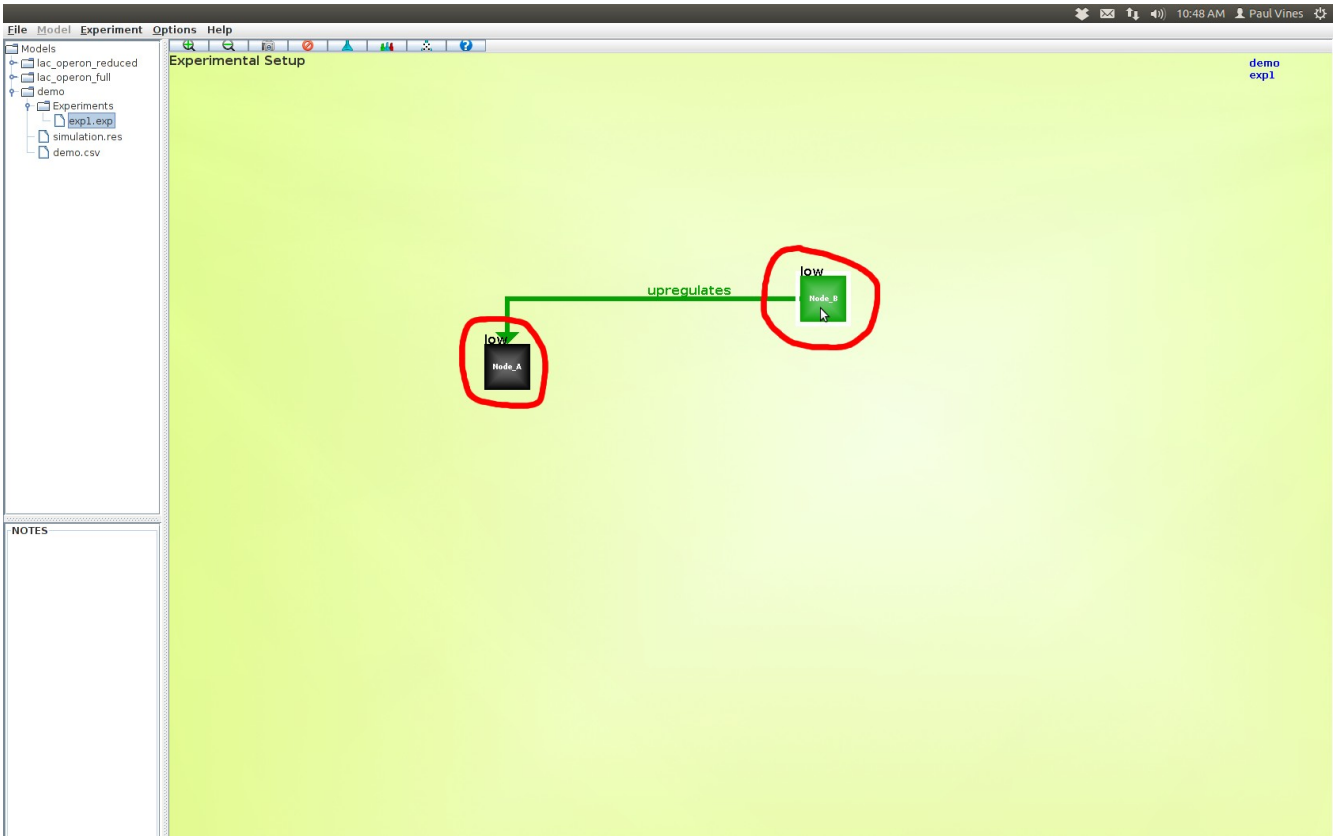
Now we are going to try running experiments to see how our model behaves when it starts from a specific state. Click on ”File” then “New Experiment”



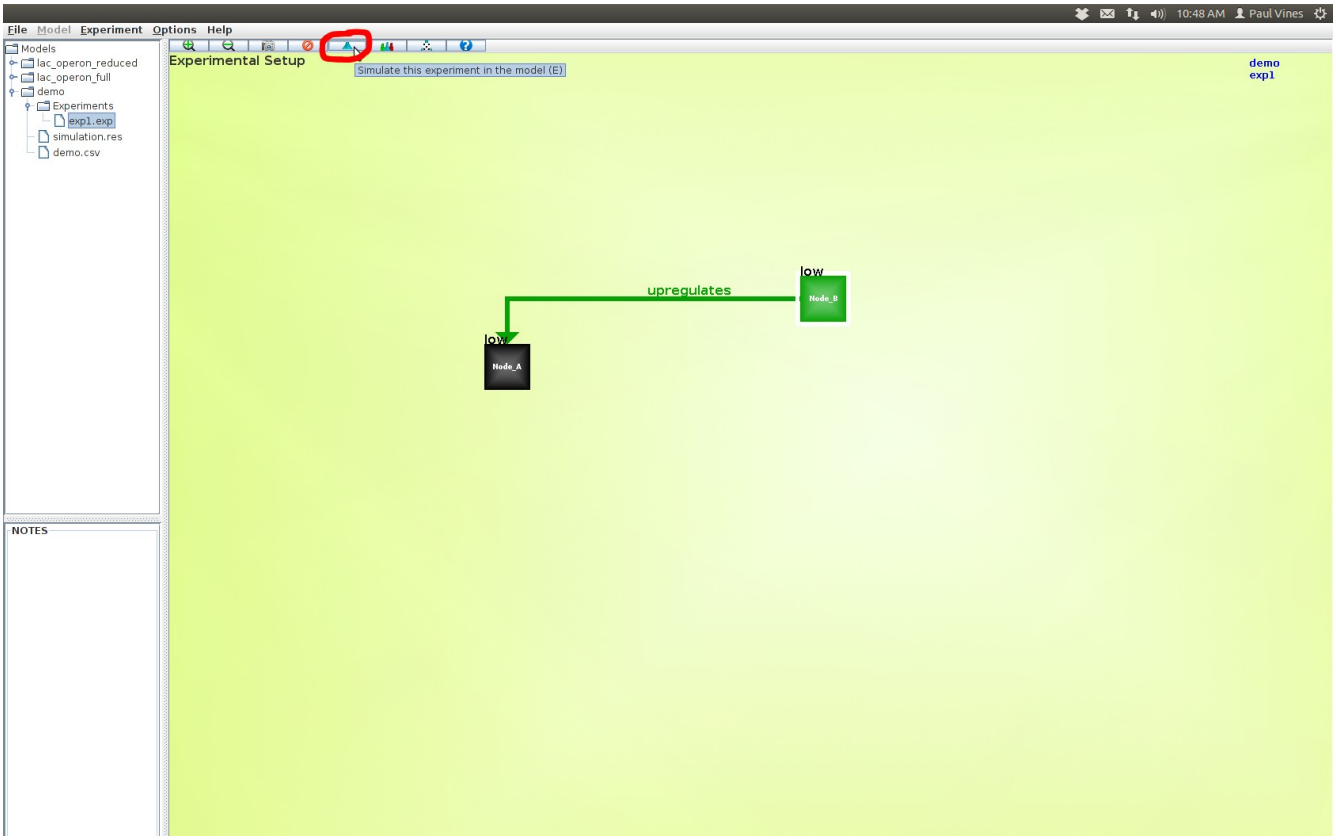
We are now in the Experimental Setup, designated by a green background, and are going to edit the name for our experiment to be “exp1.”



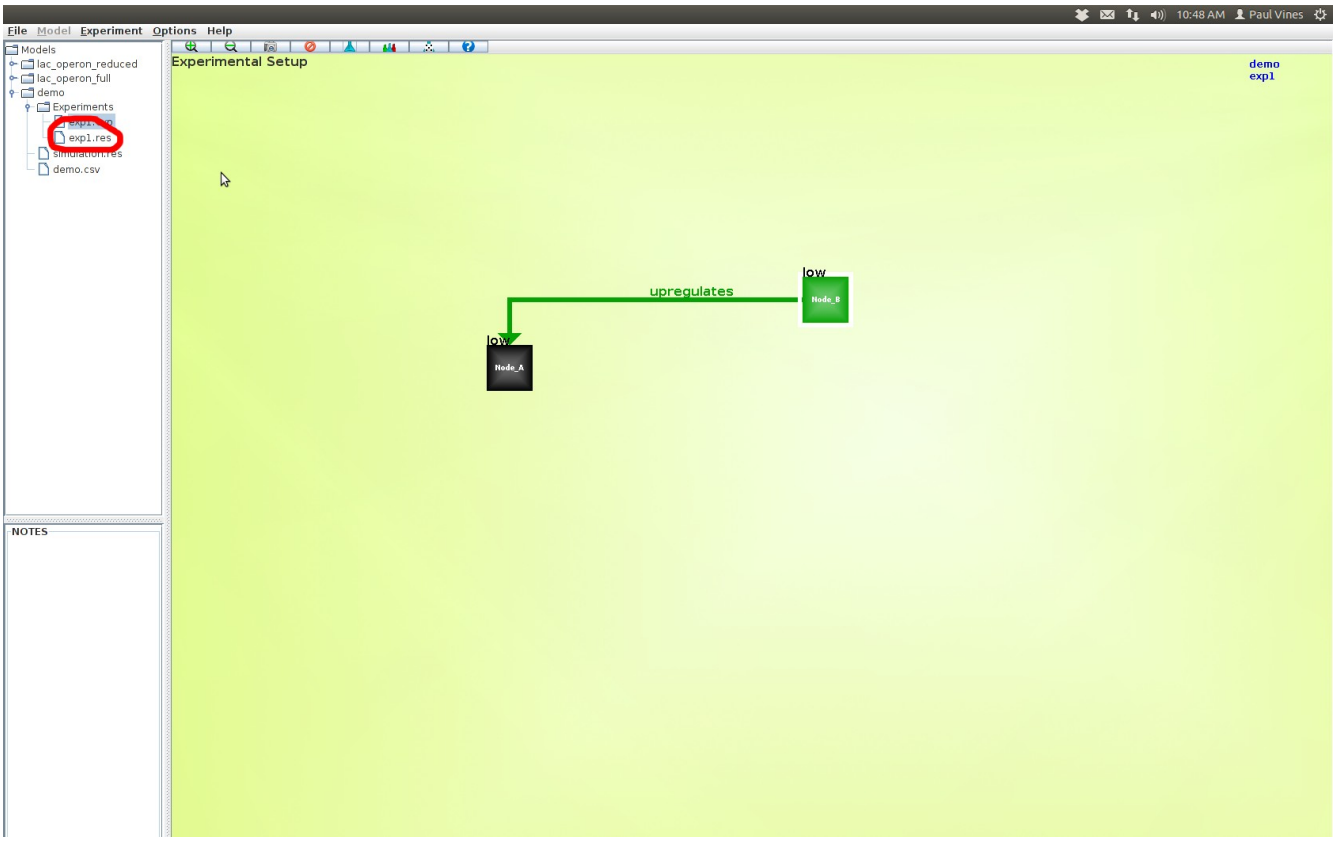
In the top-left we can now see both what model we are in “demo” and what experiment “exp1.” And we can see our nodes are again labeled with state names (in this case “high” instead of “low”).



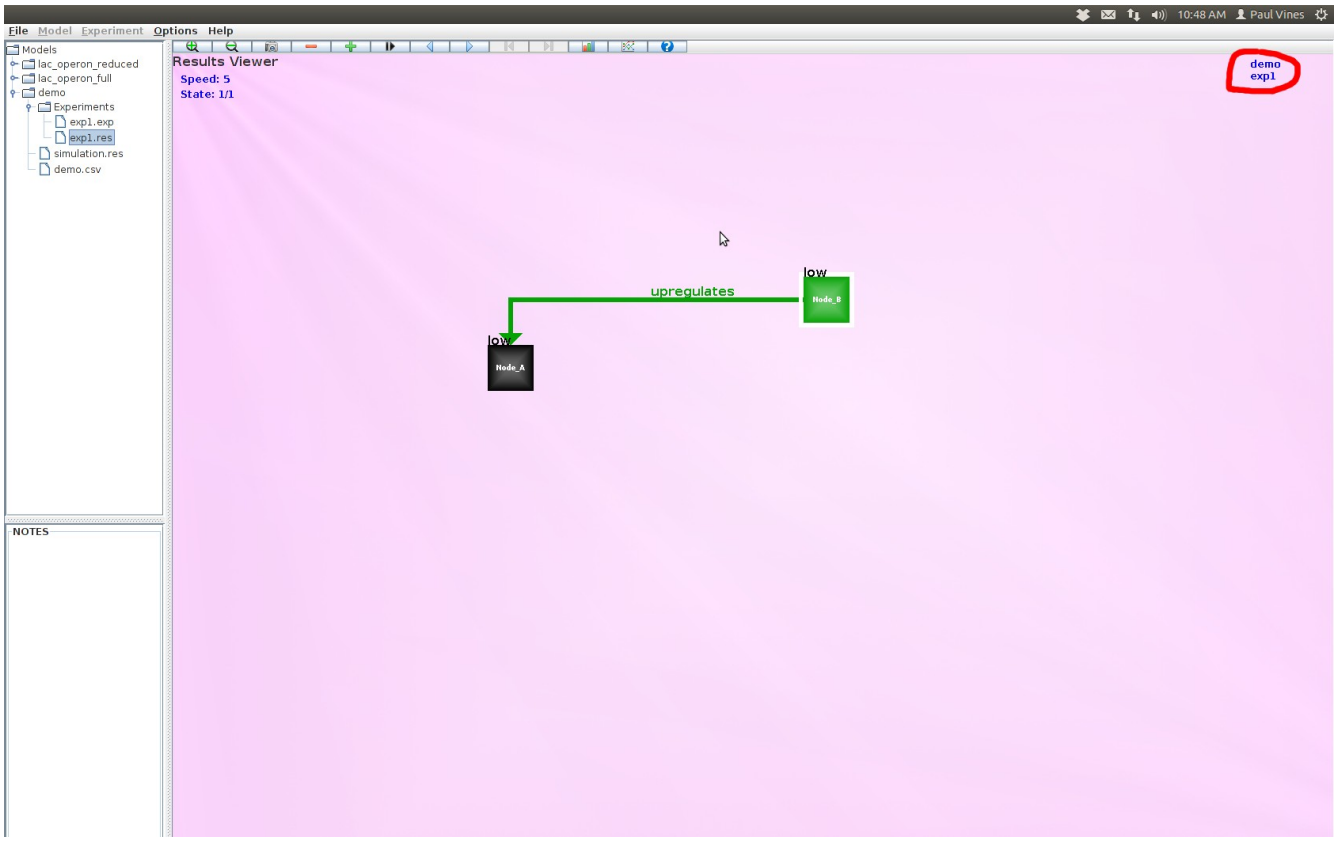
We can change the states of our nodes by double-clicking or clicking and dragging on them. To make them go low, click towards the edge and drag to the center, to make them high again, drag out away from the center.



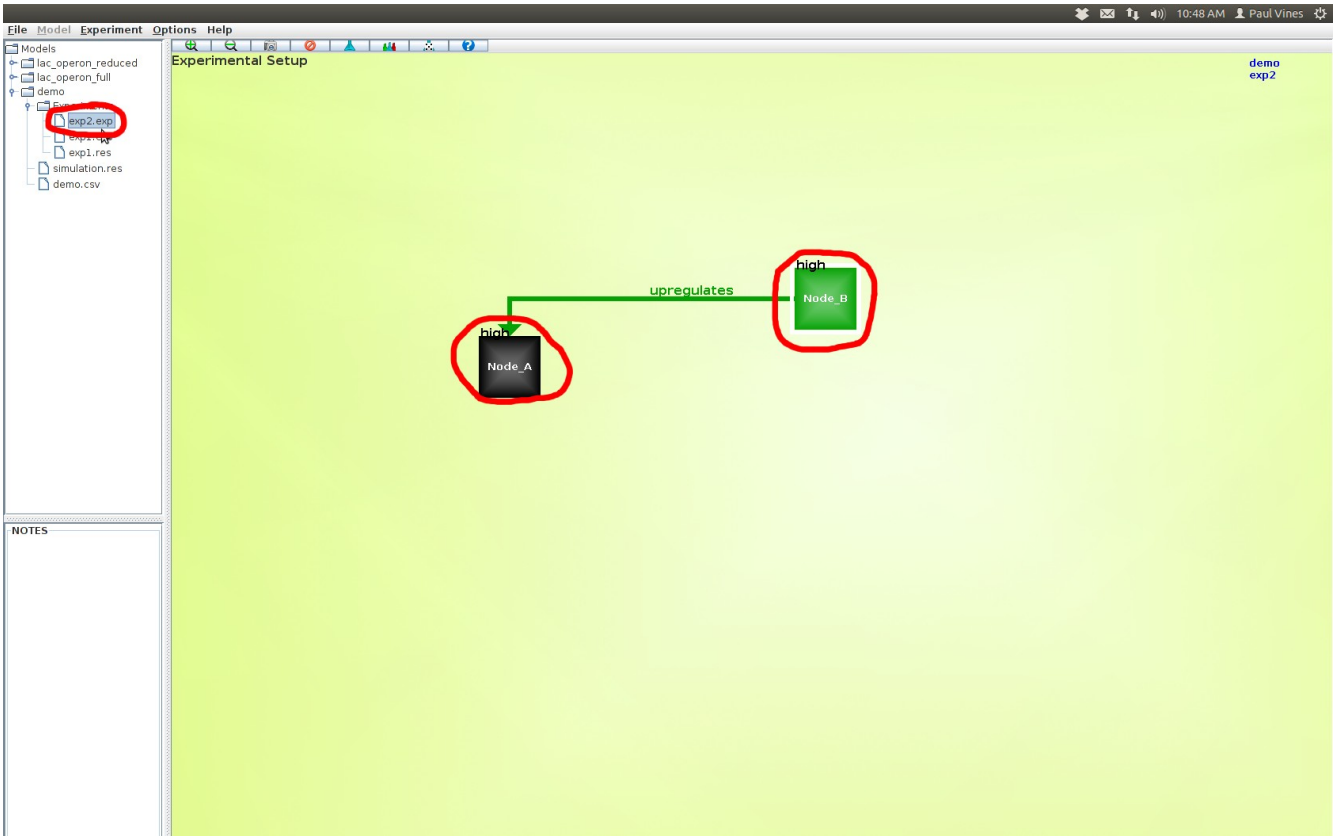
This has set our experiment's starting state as A = low, B = low. Now we will run our experiment by clicking the big blue flask.



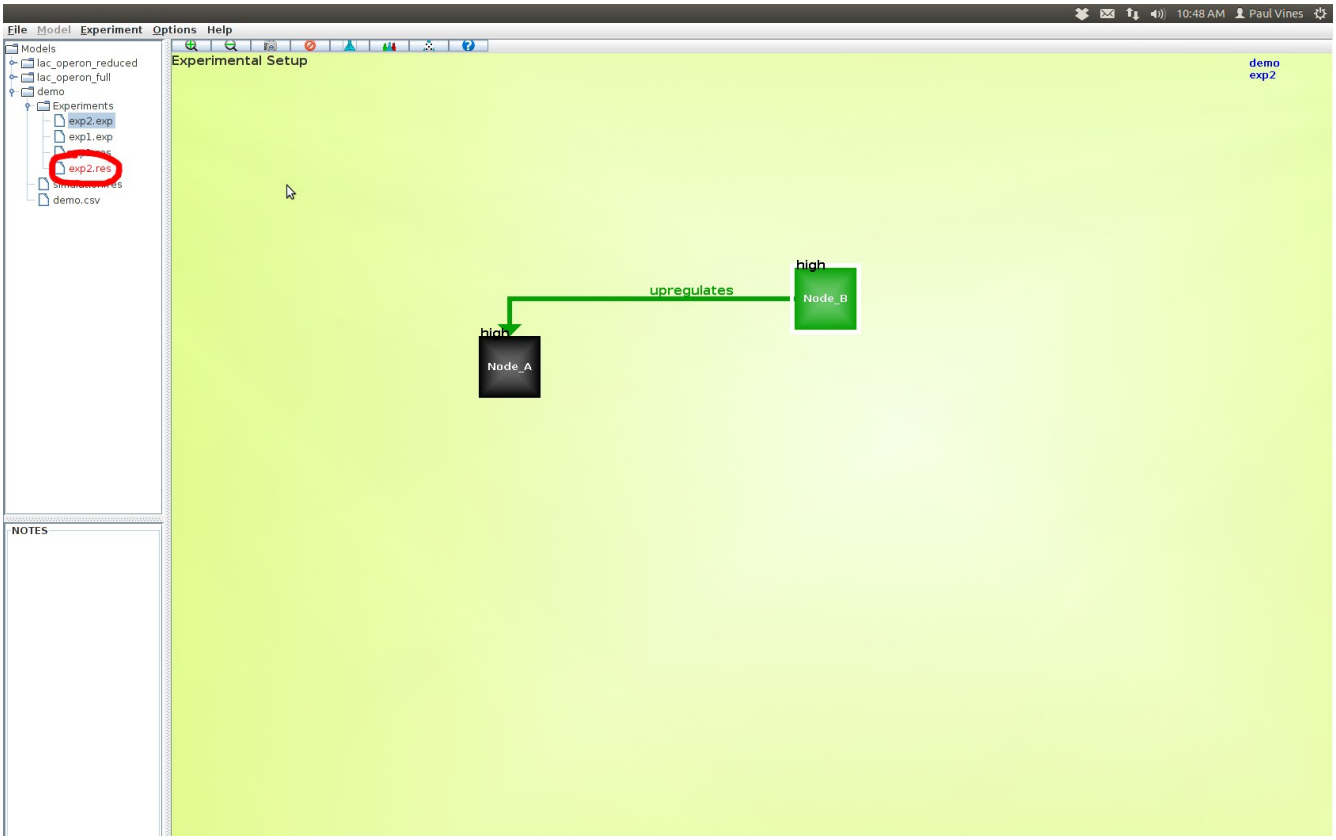
As with simulating the whole model, a new file as appeared, this one named “exp1.res.” Double-click on it.



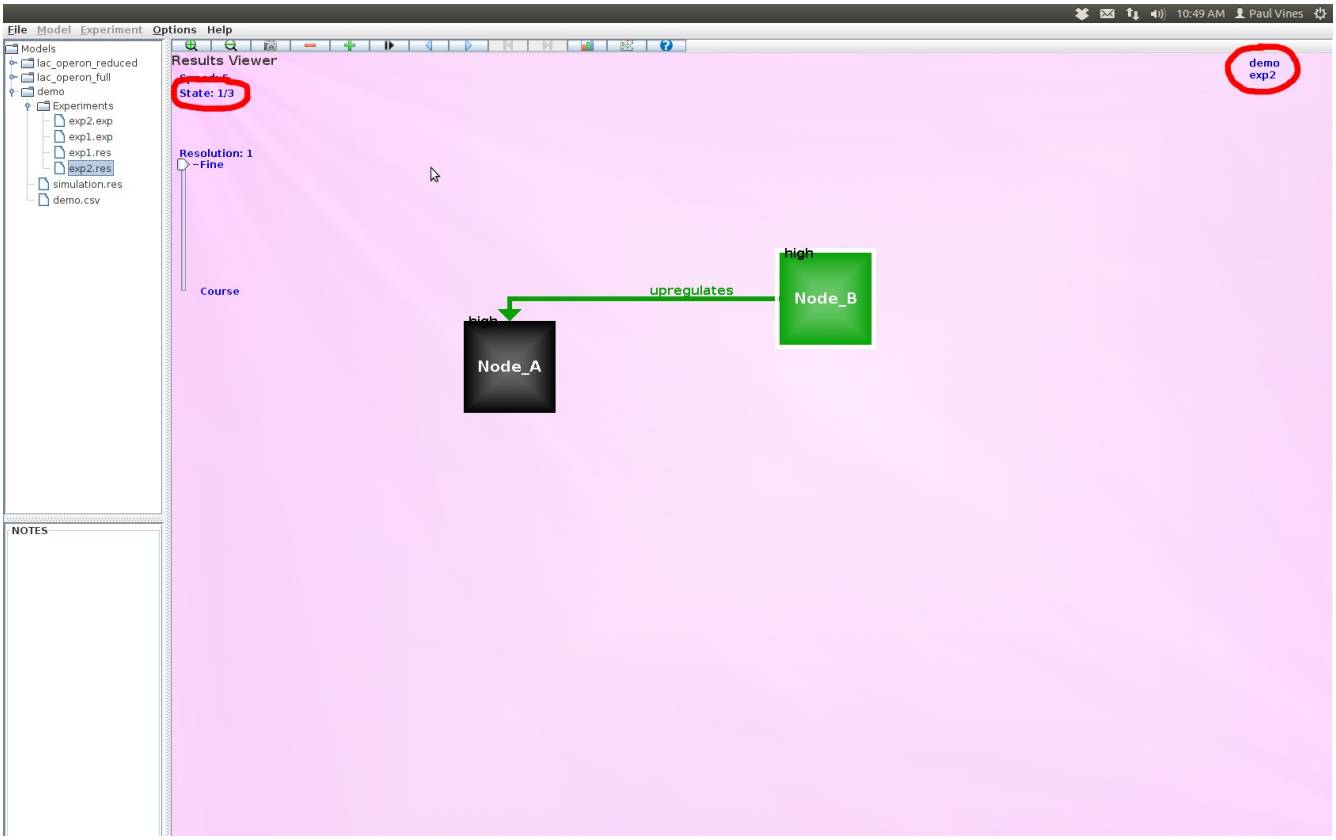
We are in a similar view as before, except now we have no “cycle” data, because every experiment can only possibly wind up in a single equilibrium point (because our model is deterministic). The experiment name is listed along with the model's in the top-right as in the experimental view.
Since our start state was A = low, B = low, and that was our model's single steady-state, this experiment obviously is not going to be very exciting, because we started in our steady-state so we will just stay there.
Let's try a different start for exp2...



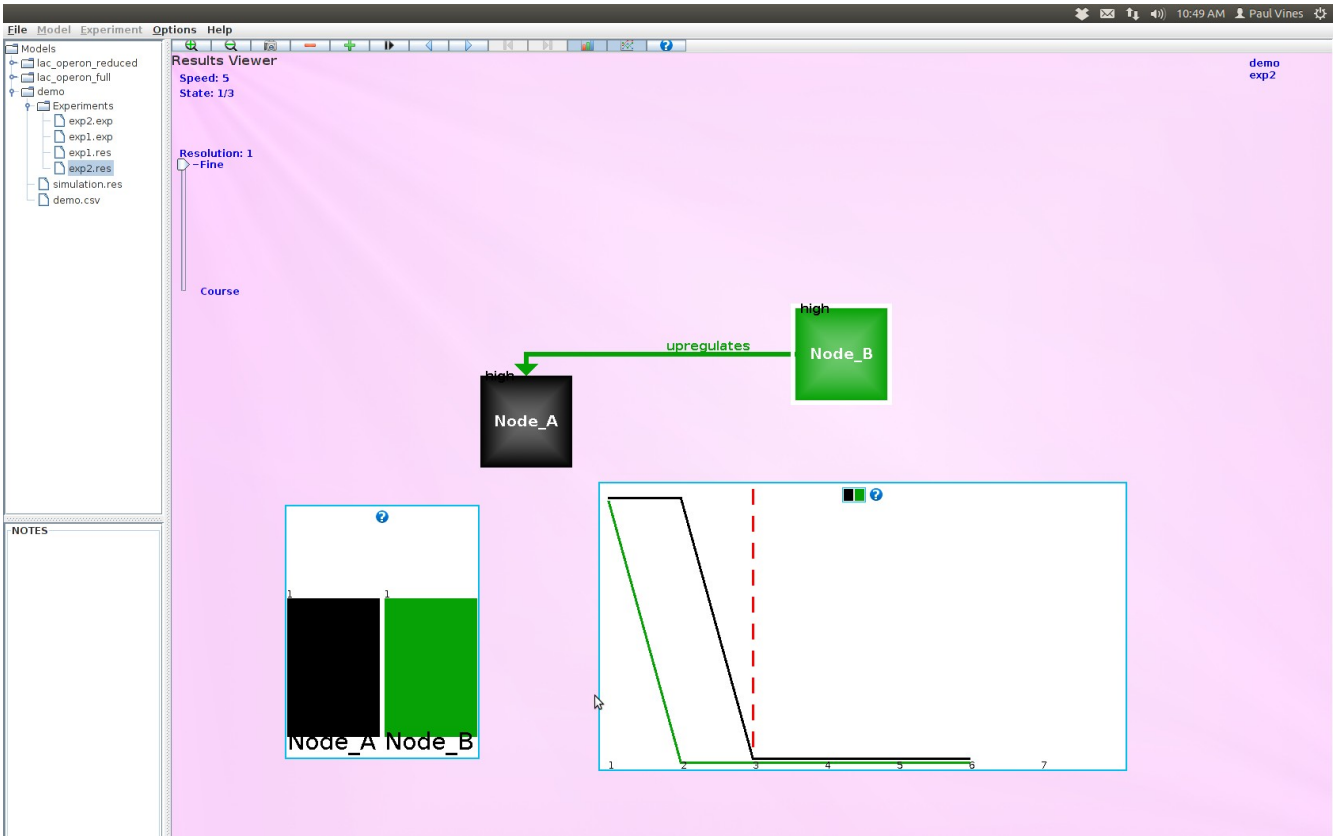
Make a new experiment and call it “exp2.” This time we will set both A and B to “high” and run that experiment and then view its results.



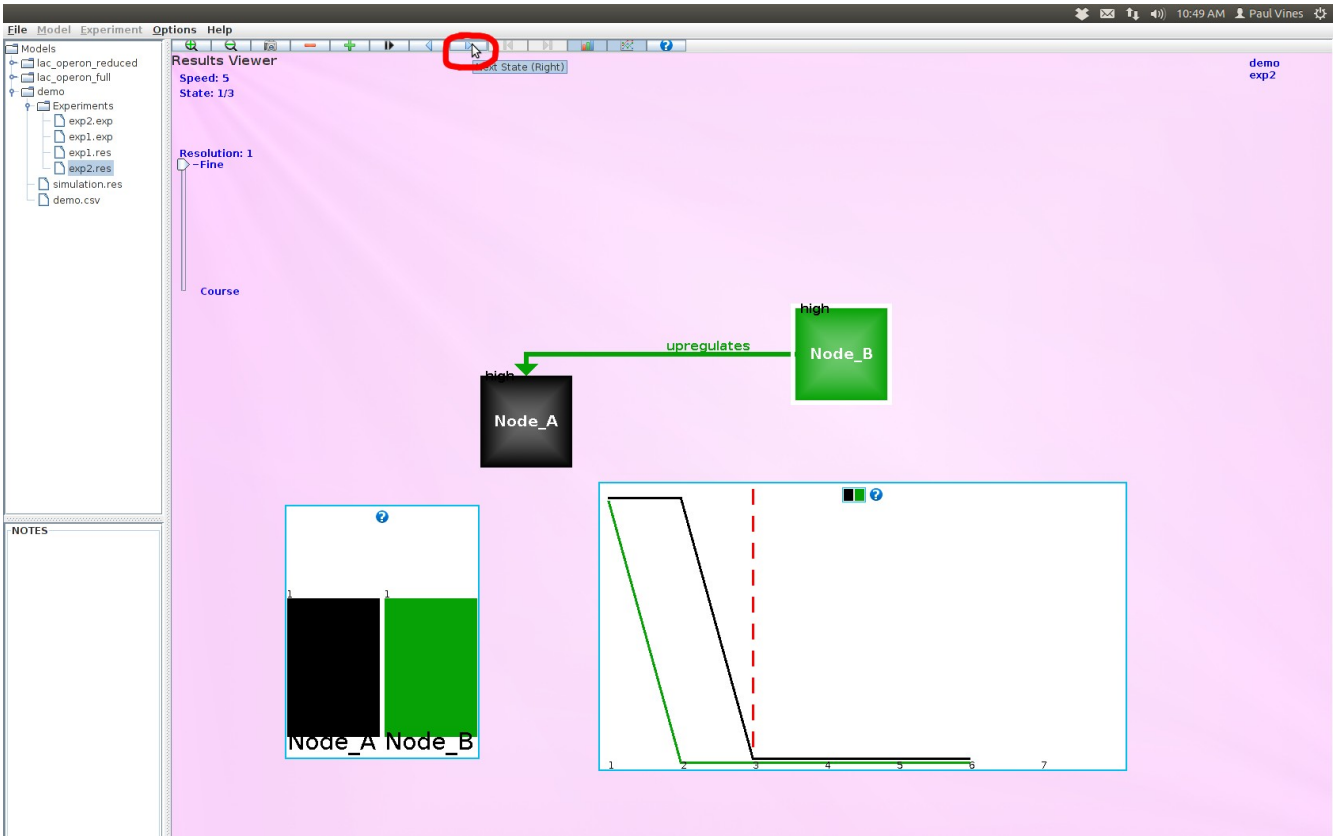
This time we have “State: 1/3” meaning our experiment has 3 states in it, and we are viewing the first one.



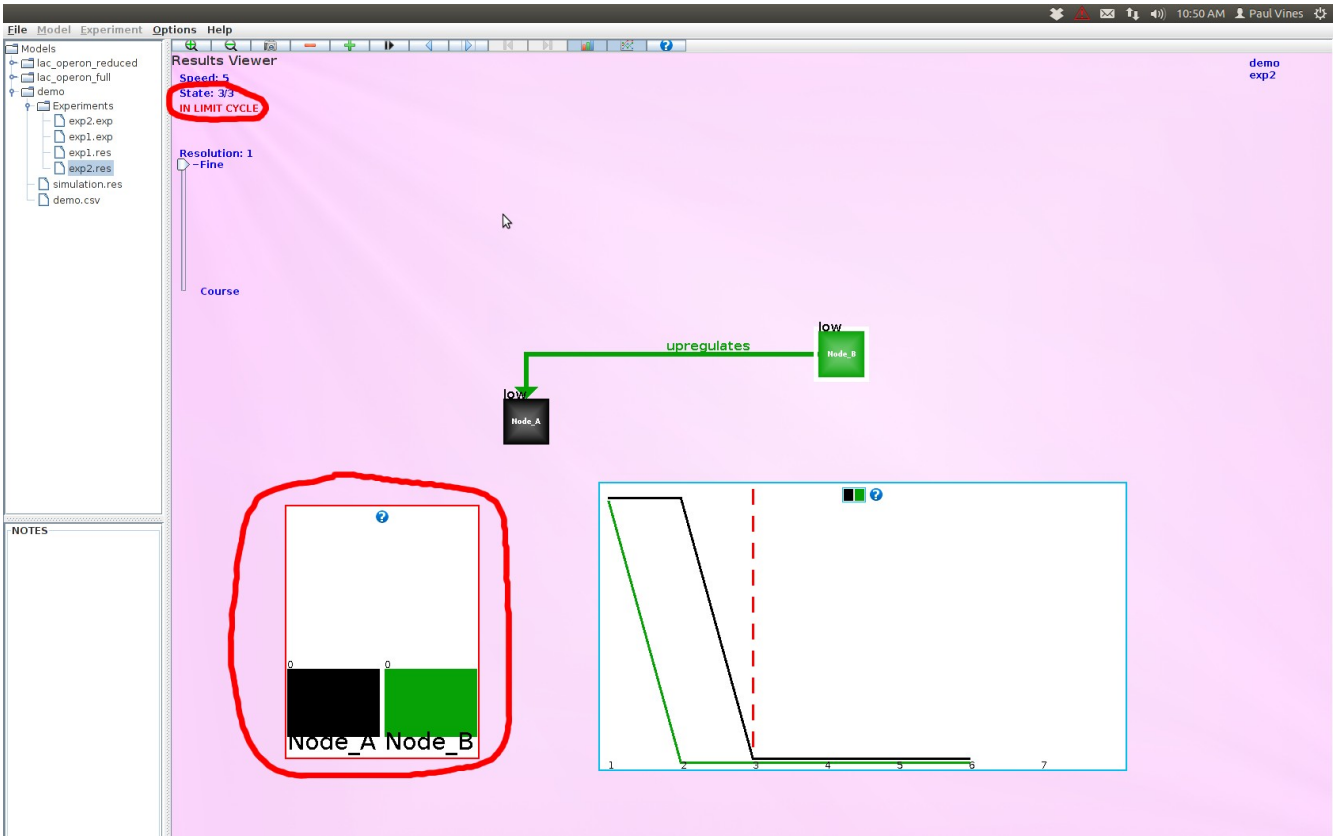
In addition to the view of the model, we can also view results in the form of bar and line graphs. The bar graphs simply show the levels of each node in the current state on a simple bar chart. The line-graph shows the level of each node at each state in the experiment's path to its equilibrium. This essentially mimics time-course data, except that it is still discrete, not continuous.



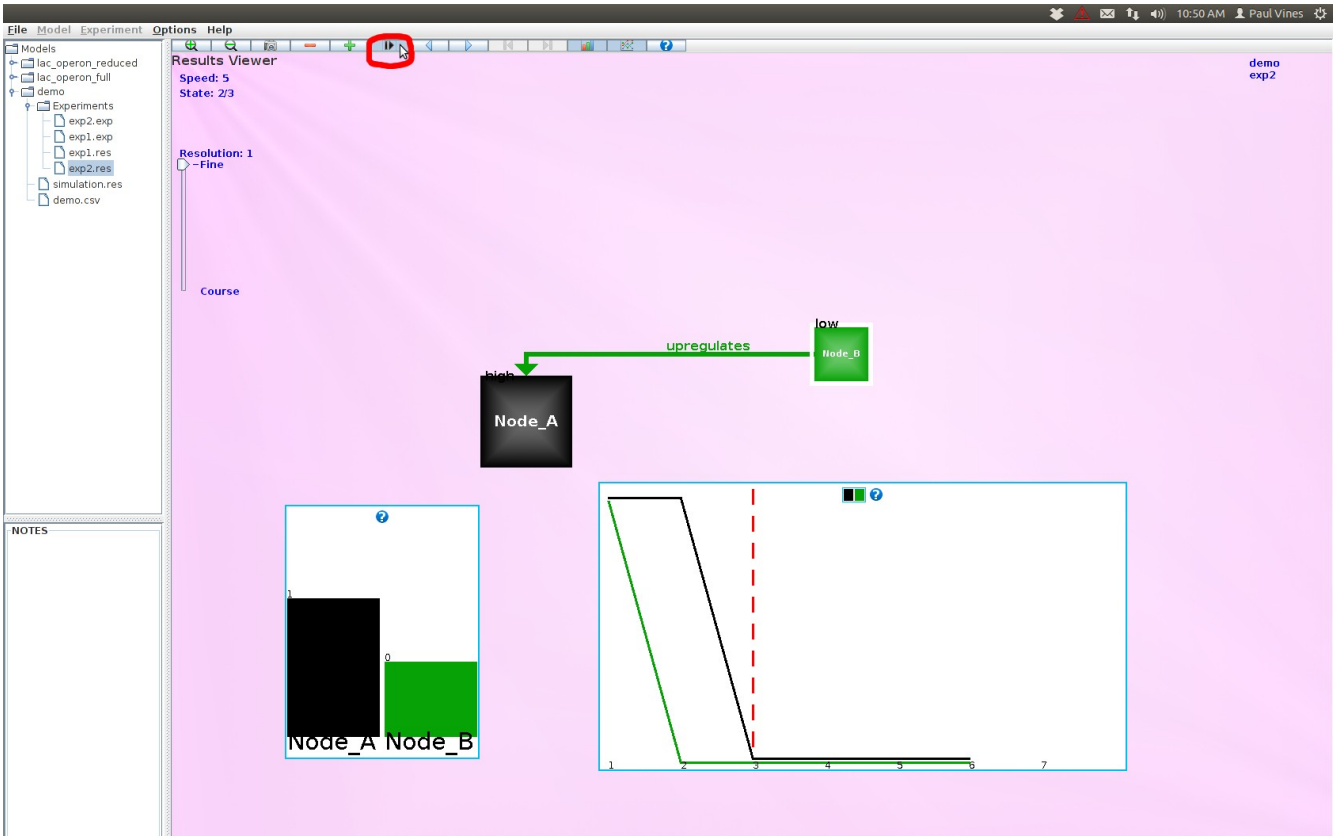
If we push the single right-arrow at the top, we move forward one state. So, we are currently viewing state 1, which was A = high, B = high where we started.



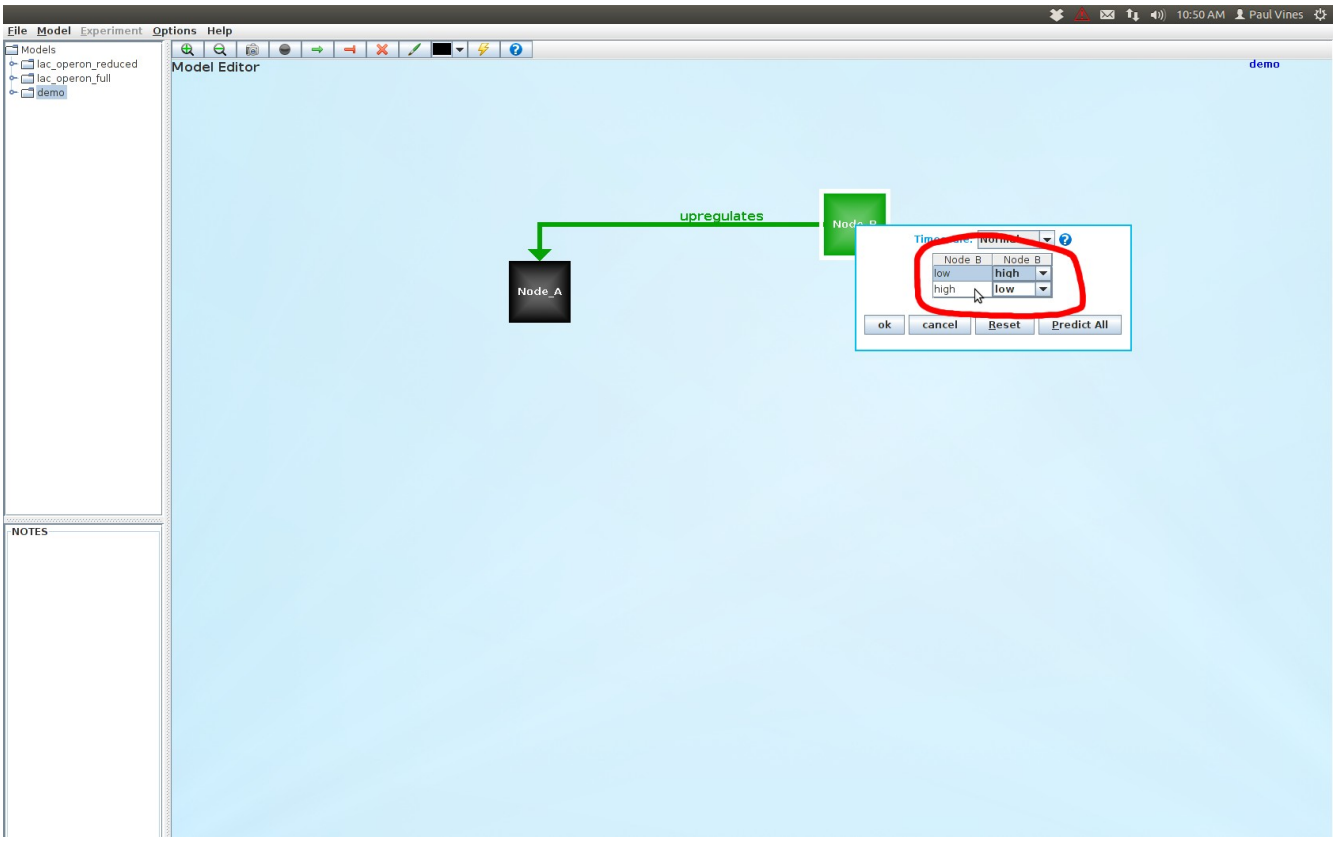
After pushing the arrow once, we are now in state 2, with A = high, B = low.



Pushing it again, we view state 3, A = low, B = low. There is now a red message beflore “State: 3/3” that says “IN LIMIT CYCLE,” that's because this is the steady state of the model, so this is where the experiment ends. This point can be seen on the line graph by looking for the vertical dashed red line.

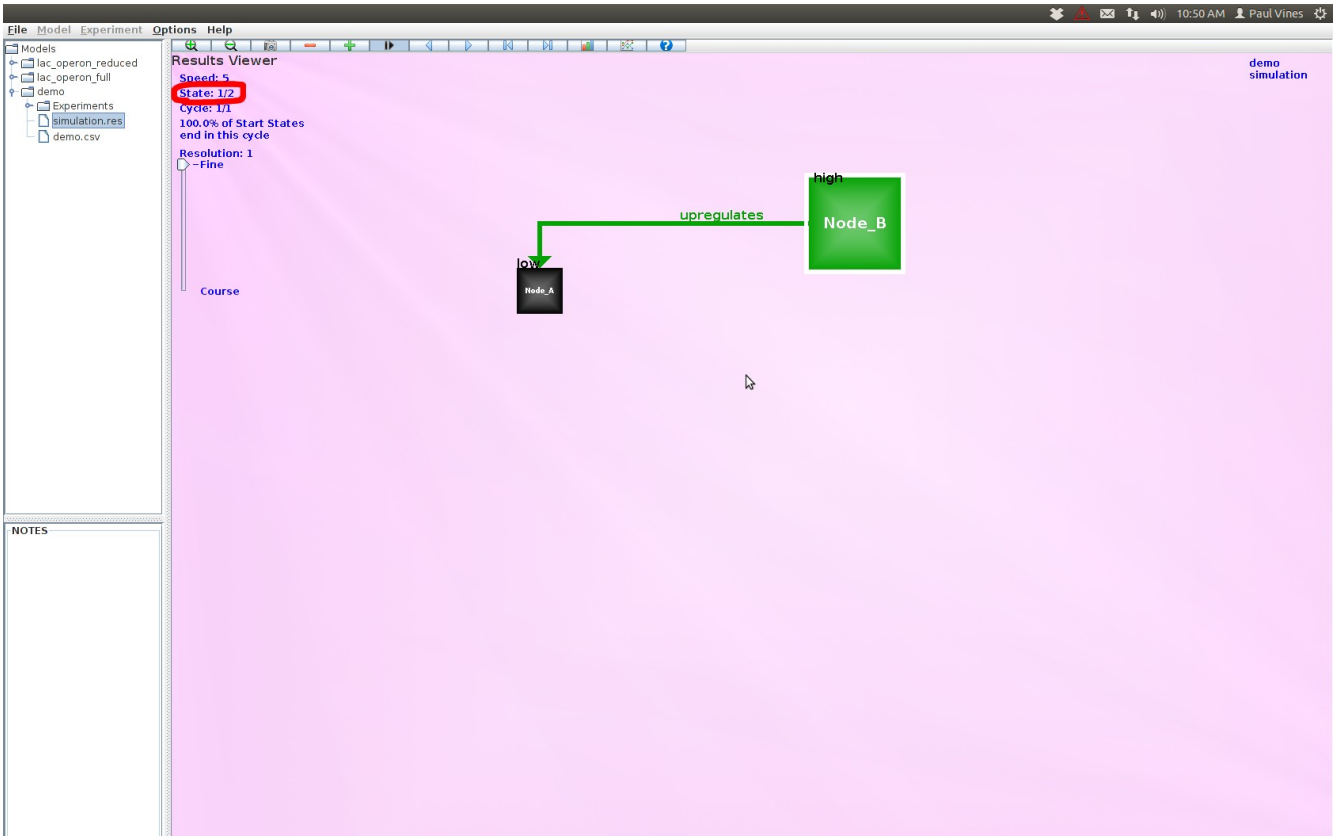


We can also view the transitions between these states as a movie. Click the arrow again to go back to state 1/3, and then toggle the “pause” button.
The nodes now gradually increase and decrease size as they transition through states. This can be a handy way to visualize network dynamics, particularly oscillations.
The speed can be increased or decreased by clicking on the “+” and “-” buttons.
But now, let's change our model to get one of these oscillations. Double-click on the “demo” folder or “demo.csv” to go back to the model editor for demo.

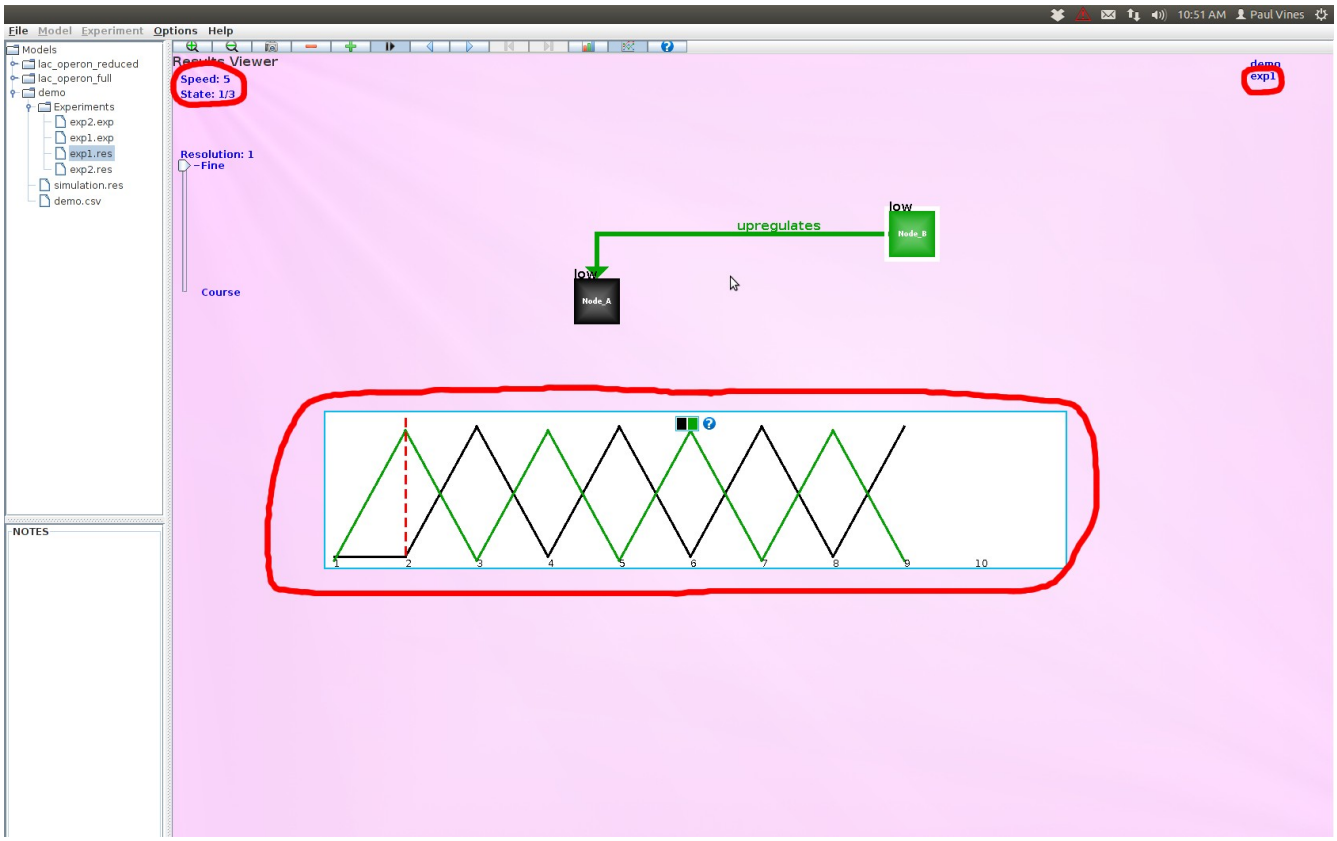


Now click on node B to bring up its table. Change the entry in row 1 to be “high” instead of “low.” This way, when B is low, it becomes high, and when B is high it becomes low, so it will eternally oscillate.

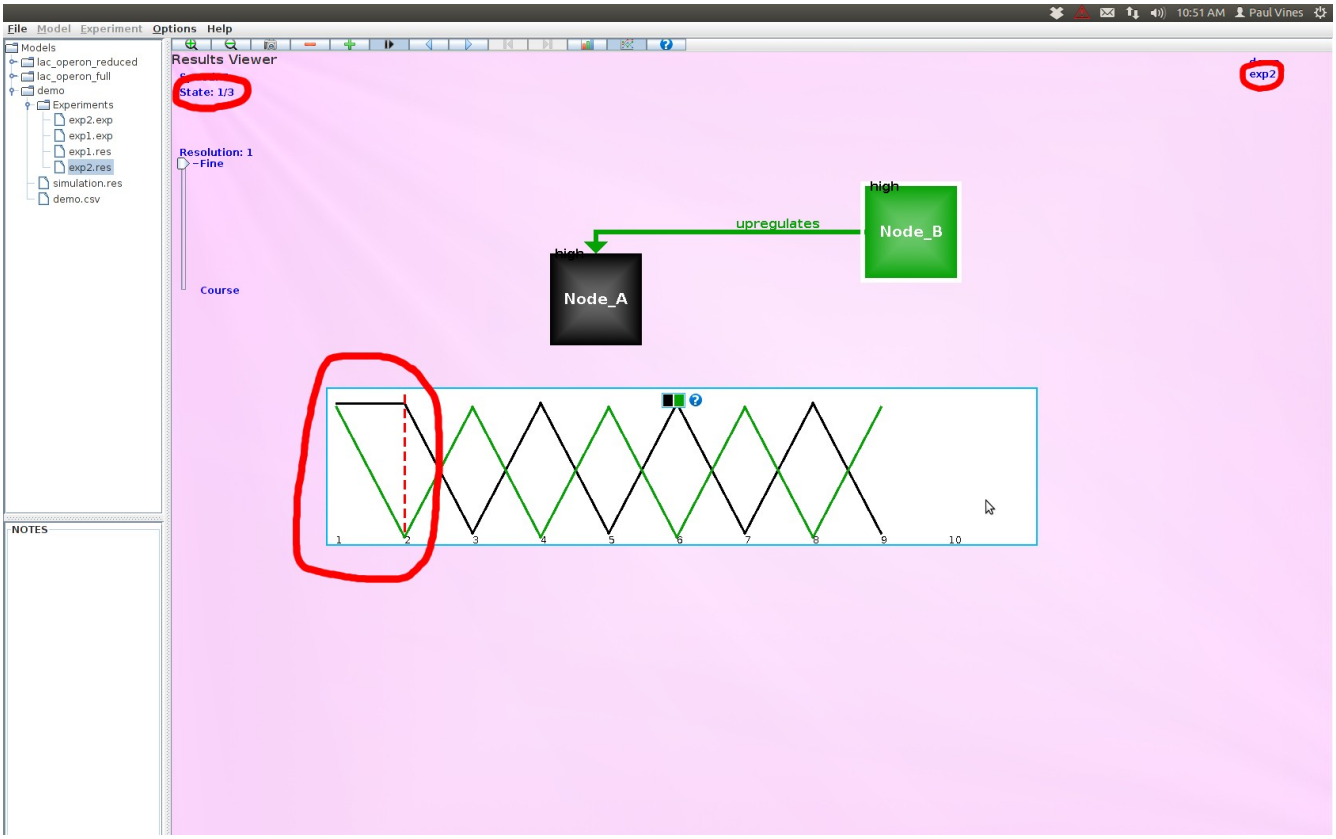
Now simulate the model by clicking the thunderbolt again, and then view the simulation.res file.



Now there are 2 states, but still only 1 cycle. You can switch between the two states, and see how they would lead to each other. Because whenever B is high, A will then become high and B will then become low. And whenever B is low, A will then become low and B will then become high. So at each step B will oscillate and A will follow along because of the edge from B to A.
This can be easily seen by viewing the line-graph.
After satisfying yourself that this is correct, go back to the experiments and re-simulate them (nothing is automatically re-run when you edit the model, so make sure you do not accidentally change the model but look at old results!).



Exp1's results now show this same cycle; we can see the start state is the A = low, B = low as before, but then B becomes high, and the experiment winds up in the oscillation.



As with exp1, exp2 shows how the start state begins at A = high, B = high, but then B oscillates down to low and then the simulation in drawn into the same oscillatory equilibrium as exp1.

So, that concludes the introduction to PlantSimLab. There are more features like Knockout experiments, stochastic simulations of experiments, custom terms, and a natural-language parser, but these are all more advanced features that will be covered in a later tutorial, or could reasonably be explored by a user.

Remember: the little blue question-marks can provide helpful hints about what something does!