PRGUI: A Visual Tool for Displaying Placement and Routing

James Foltz
Oklahoma State University

Summer Advisor: Dr. Kia Bazargan
University of Minnesota
NSF REU Program
Summer 2004

Abstract

Current two-dimensional (2D) Field Programmable Gate Arrays (FPGA's) suffer from large delays of global interconnects and this is even more stringent as circuit sizes (and therefore FPGA's) are continuously increasing. One possible solution to address this problem is the development of the three-dimensional (3D) FPGA's.

3D FPGA's can offer smaller average net delay compared to the 2D case and therefore they allow implementation of larger circuits while attaining the same performance. In order to fully analyze 3D architectures one needs appropriate CAD tools. This work is on developing a Placement and Routing Graphical User Interface (PRGUI), which can be used to visually display the placement and routing of circuits on 3D FPGA's.

Features of PRGUI include the ability to view the entire three-dimensional circuit or every layer individually. PRGUI has many filtering options for displaying the netlist and timing critical paths. These features, coupled with its abilities to zoom and rotate make PRGUI a very powerful program for displaying placement and routing on 3D FPGA's.

A very common and well used device in Electrical Engineering is a Field Programmable Gate Array (FPGA). An FPGA is a complex logic device, which can be programmed to implement circuits. It has a simplified architecture, including an array of configurable logic blocks, input and output pads, switch and connection blocks, and wires to route signals between the configurable logic blocks (CLB's). The way an FPGA works is very unique. Take a look at the figures below. Figure 1 shows a simple digital circuit, $(A \bullet B) + (C \bullet D) = Output$.

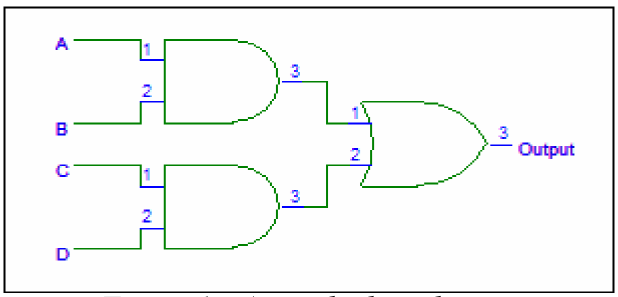

Figure 1: A simple digital circuit

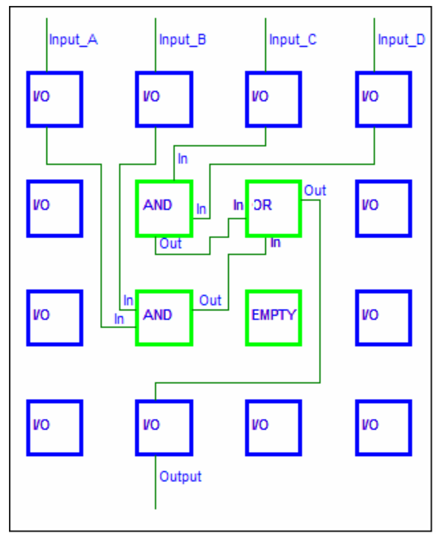

Figure 2: The circuit on a 2D FPGA

Figure 2 shows the same circuit implemented on a two-dimensional FPGA. You can clearly see the Input/Output Pads (I/O Pads), the CLB's, and the wires connecting all the components. An FPGA is always situated such that the CLB's are surrounded by Input/Output pads. Figure 2 is only one representation of how an FPGA could implement the simple digital circuit in Figure 1. For instance, the locations of the AND and OR gates could be switched, or any of the inputs or outputs could be moved to any of the open locations, or any other combination, as long as the CLB's are situated inside the I/O pads. There are many complex algorithms that determine the best and most efficient locations of the CLB's, I/O pads, and the wires.

FPGA's typically exist in a two-dimensional sense, as shown in Figure 2. A two-dimensional FPGA is comprised of rows and columns of CLB's, surrounded by I/O pads. However, FPGA technology has evolved, and now three-dimensional FPGA's are coming into play. My summer advisor, Dr. Kia Bazargan, and my grad student, Cristinel Ababei, are currently working on developing algorithms to determine the most efficient way to layout a three-dimensional FPGA. Due to the fact that this is a new technology, there is no way to physically view their work. My project this summer was to create a Graphical User Interface (GUI) to illustrate the placement and connections between CLB's and I/O pads in a three-dimensional FPGA.

For my project, I created a program in Java. Java was chosen over other languages for several reasons. It is simple to write code for, it is object-oriented, and it is platform independent, so the same code can be used on either a Windows or a UNIX computer. My program is titled PRGUI, which stands for Placement and Routing Graphical User Interface. PRGUI is quite adept at what it does, and has many abilities. PRGUI can display the entire three-dimensional FPGA at once, or it can display one layer in a two-dimensional view that the user specifies. It has many different filtering options for looking at the connections between the CLB's and I/O pads. PRGUI also has zooming in and zooming out features, as well as the ability to rotate the entire circuit and its connections ninety degrees clockwise or counterclockwise. It also has the ability to display the timing critical path information of the circuit.

PRGUI is very long and complex, consisting of 2,856 lines of code. On the following page Figure 3 shows the classes of the program, the public variables within each class, the methods used in each class, and the Java libraries imported into the program.

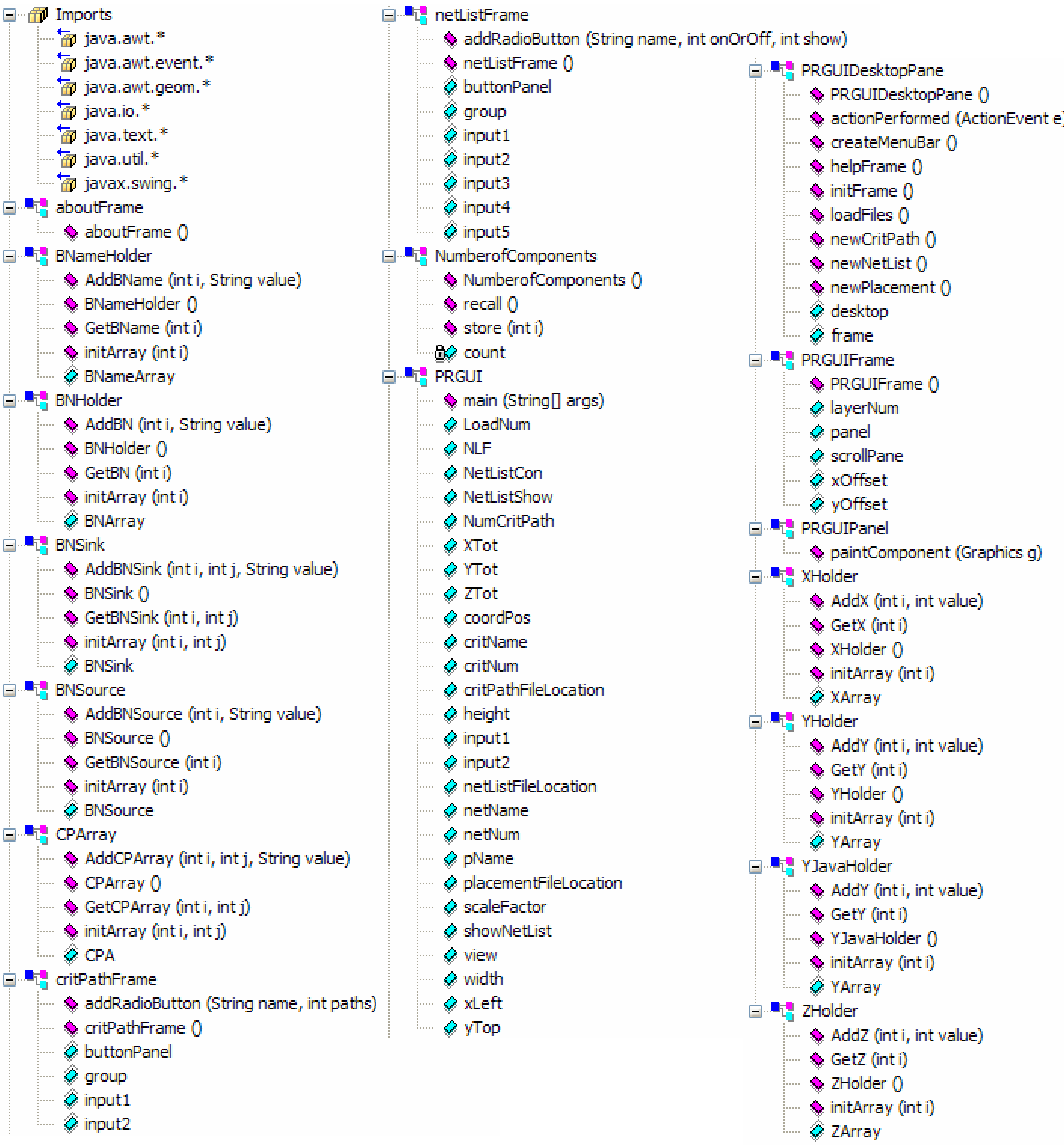

Figure 3: A graphical view of the data in PRGUI
Classes are shown by a , Methods are shown by a , Public Variables are shown by a
Screenshots taken from JCreator, copyright Xinox Software, http://www.jcreator.com

The following information is a basic summary of each class, what it contains, and what its job is inside PRGUI.

- aboutFrame: This class creates the JInternalFrame that is shown when the user selects Help → About, and has the information contained there
- BNameHolder: This class contains an array that stores the Block Names of the various blocks, in order of appearance in the .p input file
- BNHolder: This class contains an array that stores the Block Numbers of the various blocks, in order of appearance in the .p input file
- BNSink: This class contains a two-dimensional array that stores the Block Names of the Sink blocks of each Net, in order of appearance in the .net input file
- BNSource: This class contains an array that stores the Block Names of the Source blocks of each Net, in order of appearance in the .net input file
- CPArray: This class contains a two-dimensional array that stores the Critical Path information, in order of appearance in the .kcp input file
- critPathFrame: This class creates the JInternalFrame that is shown when the user selects Critical Paths → Show Critical Paths… from the menu, and has the information contained there
- netListFrame: This class creates the JInternalFrame that is shown when the user selects NetList → Filtering And Viewing Options… from the menu, and has the information contained there
- NumberofComponents: This class merely has an integer, count, that contains the number of blocks in the entire circuit
- PRGUI: PRGUI is the main method, and it contains the all the public variables that are called from multiple classes, including the location of the files imported, the dimensions of the FPGA, zoom factor, rotation information, and more. It also creates an instance of the JDesktopPane and makes it visible
- PRGUIDesktopPane: This class creates and defines the JDesktopPane used in PRGUI. Within this class, all of the Menus are created, and their actions are also defined. This class is also where ALL information is read into the program. It is the only class that has the IOException thrown.
    - actionPerformed: This method carries out the specified action when a specific option is selected from the JMenuBar
    - createMenuBar: This method creates the JMenuBar and its components within it
    - helpFrame: This method reads in "help.txt" and displays it in a JTextArea, within its own JInternalFrame
    - initFrame: This method creates the JInternalFrame that is PRGUI
    - loadFiles: This method creates the JInternalFrame shown when the user selects File → Load Circuit Files…
    - newCritPath: This method reads in the .kcp file chosen by the user and processes the data contained within the file
    - newNetList: This method reads in the .net file chosen by the user and processes the data contained within the file
    - newPlacement: This method reads in the .p file chosen by the user and processes the data contained within the file

- PRGUIFrame: This class defines the JInternalFrame created in PRGUIDesktopPane.initFrame()
- PRGUIPanel: This class draws EVERYTHING in PRGUI. It draws everything in both 2D and 3D modes, including the blocks, nets, critical paths, etc. It has many if statements that specify what it draws, that depend on the public variables in PRGUI
- XHolder: This class contains an array that stores the X location of the various blocks, in order of appearance in the .p input file
- YHolder: This class contains an array that stores the Y location of the various blocks, in order of appearance in the .p input file. This number is converted to Java coordinates – the .p file contains XY coordinates in the 1st Quadrant, whereas Java uses XY coordinates in the 4th Quadrant.
- YJavaHolder: This class contains an array that stores the Y location of the various blocks, in order of appearance in the .p input file. This class is different than the YHolder class because it does NOT change the coordinates to Java code. It is used for the rotation multiplication needed to rotate the circuit
- ZHolder: This class contains an array that stores the Z location of the various blocks, in order of appearance in the .p input file

PRGUI is very complex. When it is first opened, PRGUI displays an empty JDesktopPane. This is shown below in Figure 4. As can be seen, the menus across the top read

Figure 4: A newly opened instance of PRGUI

File, View, NetList, Critical Paths, Zoom, Rotate, and Help. The first thing that should be done to use PRGUI is to load circuit files. Under the File menu, there are two options, as shown in Figure 5: Load Circuit Files… and Quit. Quit is self explanatory, it exits the program. When

Figure 5: File Menu

Load Circuit Files… is selected, the user is greeted with a new window, shown in Figure 6. From this menu, the user can see the current files loaded, though in Figure 6, no files are loaded, so the file information is blank. When one of the buttons is selected, a new window appears, allowing the user to browse their computer for the proper file to load into PRGUI. This window is shown in Figure 7. After the circuit files are loaded, the default circuit settings for PRGUI are displayed. This can be seen in Figure 8 on the next page. By default, PRGUI shows qualitative information about the circuit

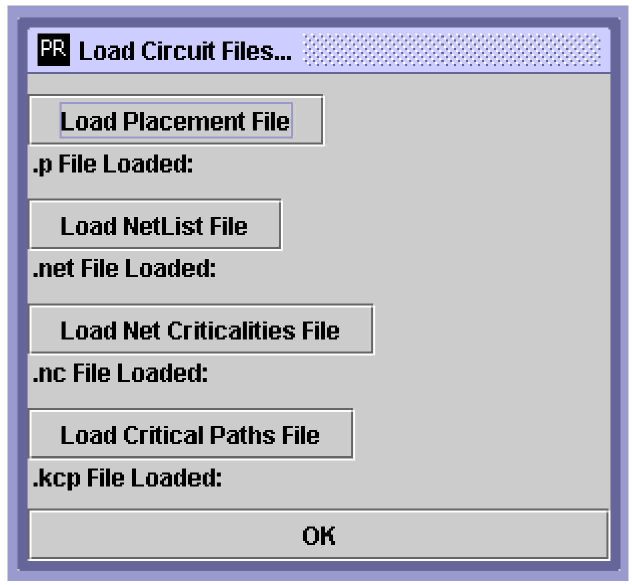

Figure 6: Load Circuit Files…

it is currently displaying. PRGUI displays the names of the files currently loaded, the total number of cells (blocks) in the circuit, the total number of nets (connections) in the circuit, the total number of layers in the circuit, and it displays the entire circuit in a three-dimensional view, with used blocks being white in color, and unused blocks being light grey in color. The coordinates of three corners can

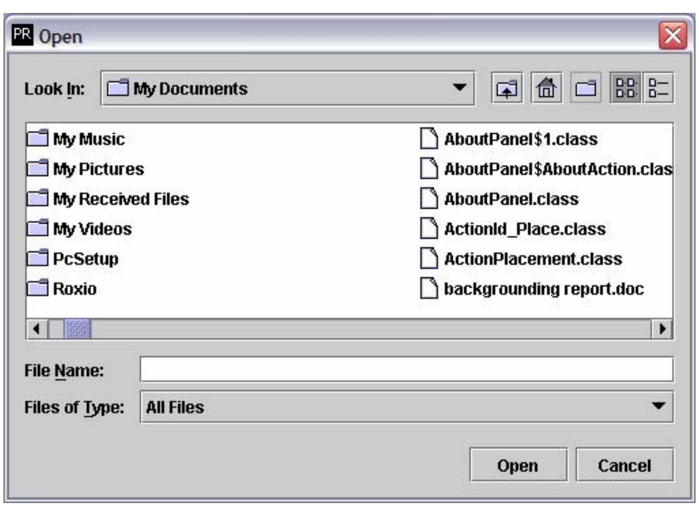

Figure 7: Select a file to load

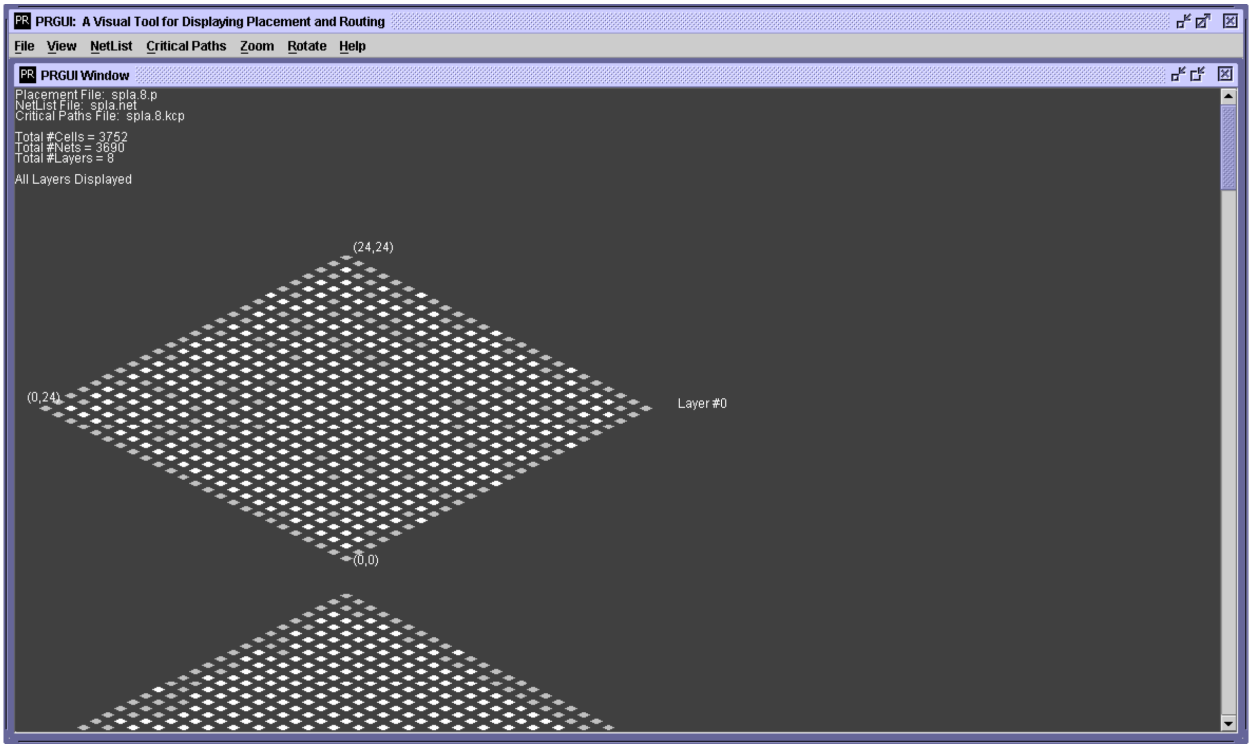

Figure 8: PRGUI in default setting mode, just after the loading of files

be seen in Figure 8. These are important so the user can understand the orientation of the circuit after it rotates.

Figure 9 shows the options under the NetList menu. The two options are Filtering And Viewing Options… and Hide NetList. The NetList is the viewing of the Nets, otherwise know as the connections in the circuit. It does not display proper routing information – what it does is if Block A is connected to Block B, a straight line is drawn connecting them. In reality this is not the way the FPGA is wired, as the wires exist only between blocks, so the connections must be routed through the circuit as such. When Filtering and Viewing Options… is selected, a new window pops up. This is shown on the next page in Figure 10. By default, the Hide NetList option is selected. There are many filtering options for the NetList. The following is a list of the options and an overview of what each one displays.

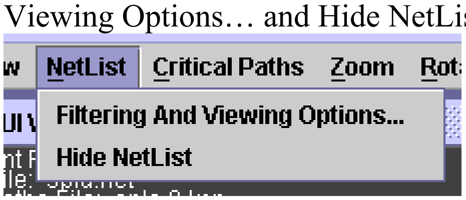

Figure 9: NetList Menu

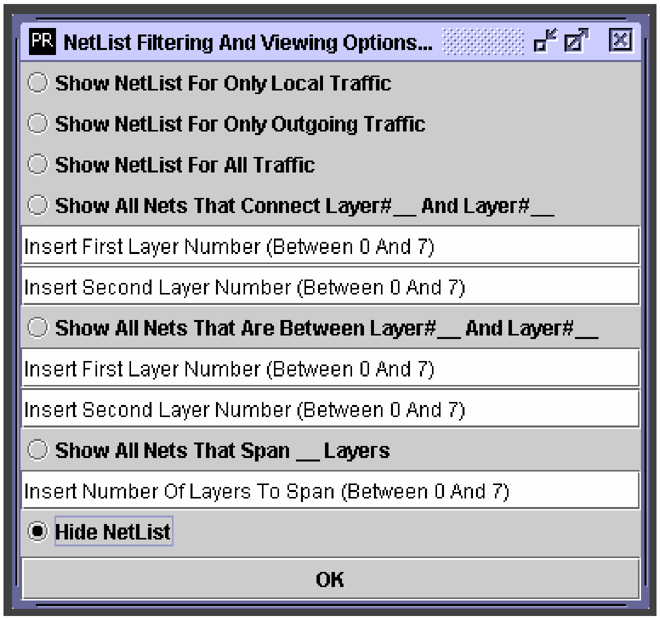

Figure 10: Filtering And Viewing Options…

- Show NetList For Only Local Traffic: If this option is selected, PRGUI will show all the connections in the circuit that begin and end on the same layer.
- Show NetList For Only Outgoing Traffic: If this option is selected, only the nets that begin and end on different layers will be displayed.
- Show NetList For All Traffic: If this option is selected, all the nets in the entire circuit are displayed.
- Show All Nets That Connect Layer #__ And Layer#__: If this option is selected, the user inputs two layer numbers, and the nets that connect the two layers specified are displayed.
- Show All Nets That Are Between Layer#__ And Layer#__: If this option is selected, the user inputs two layer numbers, and the nets that begin and end on layers between the two numbers specified are displayed.
- Show All Nets That Span__ Layers: If this option is selected, the user inputs a number, and the nets that connect layers that far apart are displayed.

Figure 11, below, shows a circuit in three-dimensional view with Show NetList For All Traffic shown. The local traffic is colored cyan, and the outgoing traffic is colored yellow.

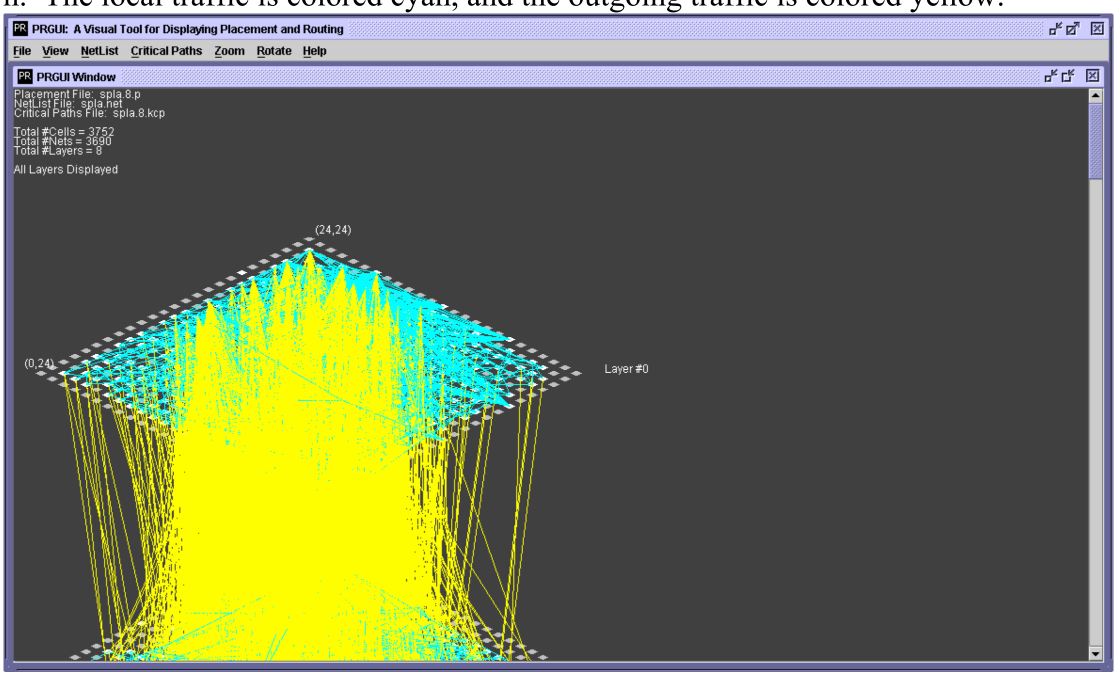

Figure 11: A circuit in 3D view with all nets shown

Figure 12 shows the options available under the Critical Paths menu. The two options are to Show Critical Paths… and to Hide Critical Paths. Critical paths are very important in analyzing a circuit. The most critical path in the circuit is the path with the longest input to output delay. The delay comes both from processing the logic and from signals traveling through the wires. The operating frequency is determined from the critical path, as the whole circuit must go as slow as the critical path for the circuit to work properly. When a user wishes to view the critical path information, and selects Show Critical Paths… from the menu, the window shown in Figure 13 pops up. The default setting is currently selected, to Hide Critical Paths. The first option for viewing the critical paths is to Show The__ Most Critical Path. When a user selects this option and inputs which critical path they wish to see. As can be seen in Figure 13, the file currently loaded has information for the ten most critical paths, so the user can view the first through the tenth critical path. The other option for viewing the critical paths is to Show The First__ Most Critical Paths. When this option is selected, the user inputs how many of the first critical paths they wish to view. In this case, they can view up to the first ten critical paths at once. When a critical path is shown, it is displayed in red, and the blocks it connects are highlighted in red. The first most critical path is displayed in the circuit shown in Figure 14 on the next page.

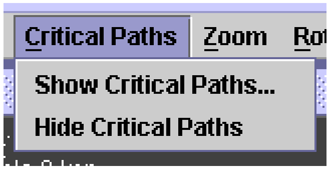

Figure 12: Critical Paths Menu

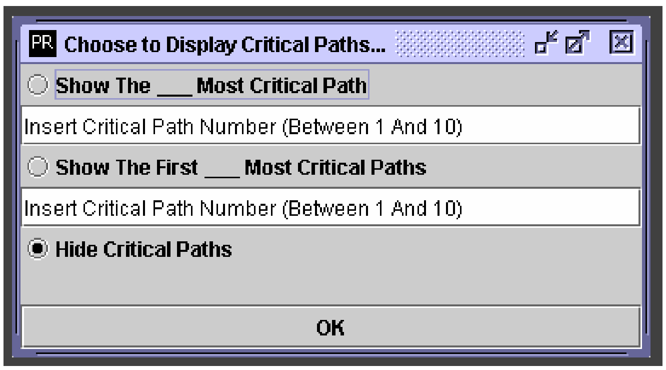

Figure 13: Show Critical Paths…

Another important ability PRGUI has is it can display a single layer in a two-dimensional view. This capability is useful for users wishing to get a more in-depth view of the connections

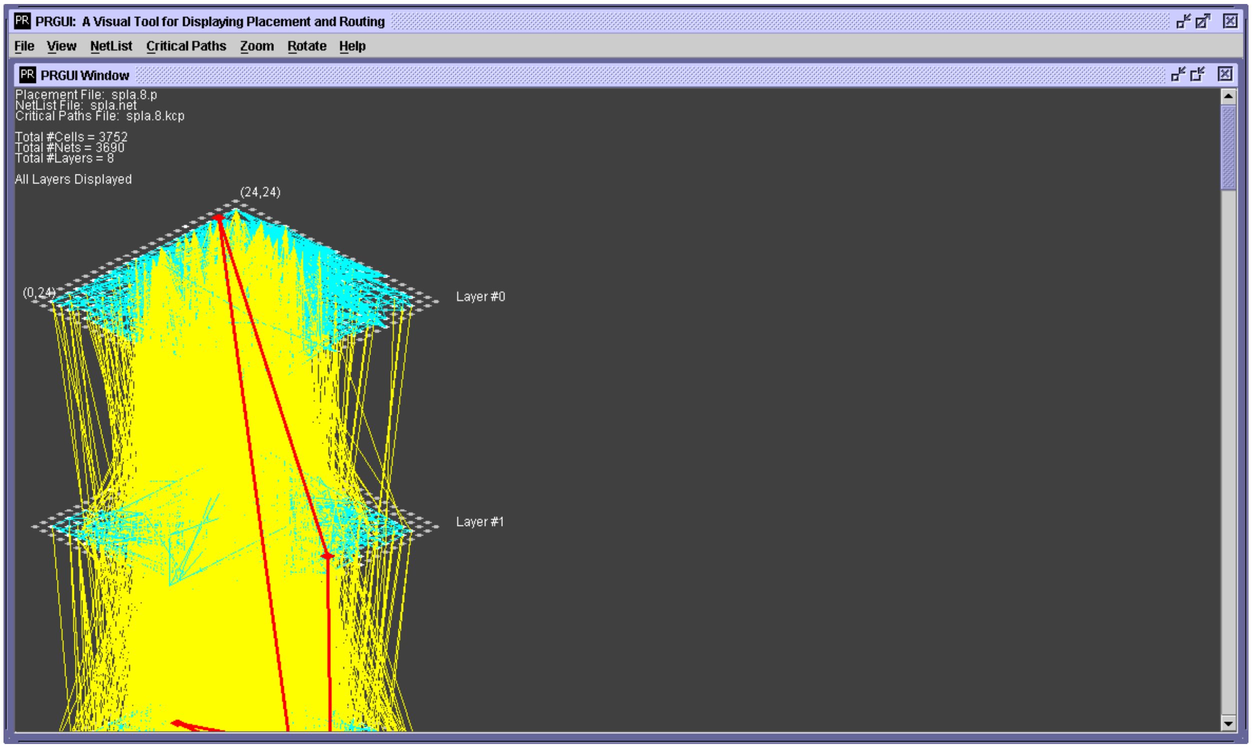

Figure 14: A circuit in 3D view with all nets and the first most critical path shown

on one specific layer. Figure 15 shows the options available under the View menu. By default, PRGUI displays all the layers in a three-dimensional view. However, when Show 2D Layer… is selected, the user is greeted with the window shown in Figure 16. This window asks the user to specify which layer they wish to view, and shows them the layers available in the current circuit. In Figure 16, the FPGA has eight layers, so the user can input a layer from zero to seven. After a layer number is inputted and OK is selected, PRGUI displays the chosen layer in a two-dimensional view. Whatever settings were previously selected in three-dimensional view mode remain intact in the two-dimensional view. For example, Figure 14 shows a circuit with all the nets shown and the first most critical path displayed. Using the same circuit with

Figure 15: View Menu

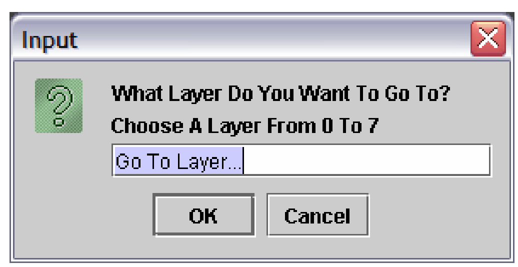

Figure 16: Show 2D Layer…

those settings and viewing the first layer of the circuit in two-dimensional view mode yields the circuit shown in Figure 17, below. As can be seen, it is similar in many ways to the three-

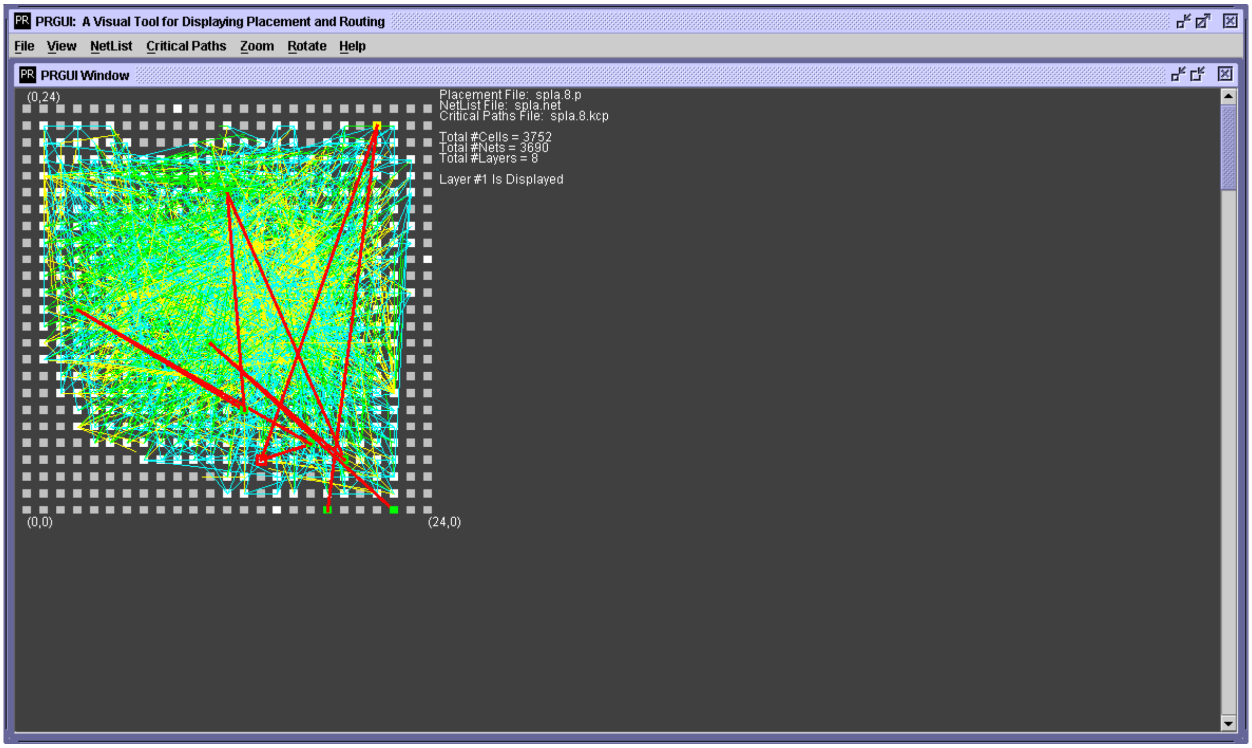

Figure 17: A circuit in 2D view with all nets and the first most critical path shown

dimensional view mode. The circuit information is still displayed, the coordinates are displayed, the used blocks are white and the unused blocks are light gray, the local nets are cyan, and the critical path is still red. However, there are many differences as well. The entire critical path is shown, not just the part on the current layer. The blocks it runs to are still highlighted, but those on a layer above the one currently being displayed are highlighted yellow, those below are highlighted green, and those on the current layer are highlighted red. The outgoing nets are also different too. If a connection is going to or coming from a layer above the layer being displayed, half of the connection is shown on the current layer in yellow. If a connection is going to or coming from a layer below the layer being displayed, half of the connection is shown on the current layer in green. Since only half of the connection is displayed, it appears to just suddenly stop. However, if the user goes to the other layer that the connection is going to or coming from, from that point the connection would be completed to its beginning or destination. This is due to the limitations of a two-dimensional view.

Two very important abilities of PRGUI are the options to zoom in and out and rotate. These options are very useful for the user to get a more in depth view of a particular part of a circuit, or to get a better overall view of the circuit. Figure 18 shows the Zoom menu. As can be seen, the zoom commands are associated with the keyboard shortcuts CTRL-Z for Zoom In and CTRL-X for Zoom Out. PRGUI can zoom in, zoom out, and rotate with any of its connections shown. It is very adept at what it does. If the circuit is zoomed in enough, the block numbers are displayed on each block. This is shown below in Figure 19. Figure 20 shows the options available under the Rotate menu.

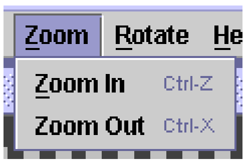

Figure 18: Zoom Menu

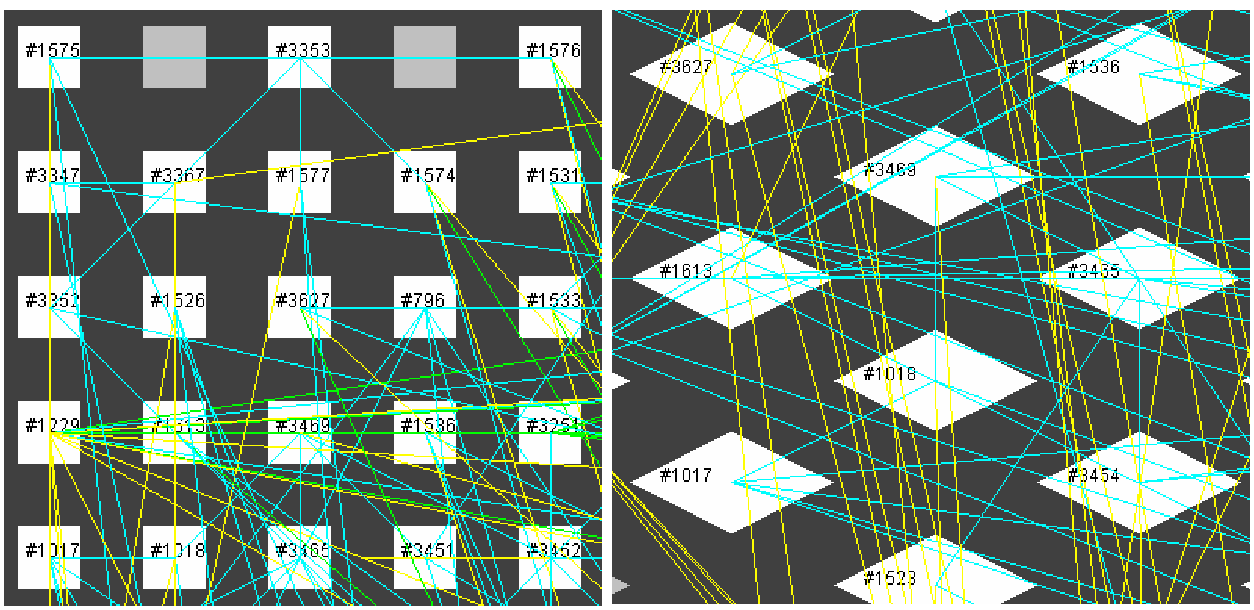

Figure 19: Left: Block Names visible in 2D Right: Block Names visible in 3D

There are only two options for rotation. The circuit can be rotated either ninety degrees clockwise or counterclockwise. When used in combination with

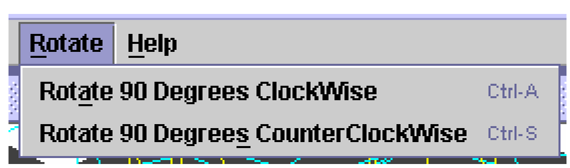

Figure 20: Rotate Menu

the zoom options, rotation is a very powerful tool for viewing and analyzing a circuit from many different aspects.

One final menu in PRGUI is the Help menu. It has two options, shown in Figure 21. The options are Help and About. Help brings up a help menu with a text file detailing the options available in PRGUI. About brings up a menu that displays personal information about the author, and the circumstances for which PRGUI was created. These options are useful for users wishing to know more about PRGUI.

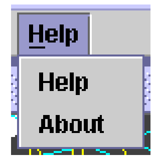

Figure 21: Help Menu

PRGUI is an excellent way for a user to visually see the layout of a three-dimensional FPGA. It has many filtering options for displaying the netlist, and for displaying the critical paths. It has the ability to display the entire circuit in a three-dimensional view, or to view one layer at a time in a two-dimensional view. With these options and the zoom and rotation features, PRGUI is a very powerful Graphical User Interface for displaying placement and routing information in a three-dimensional FPGA.