

ECE 1387 - CAD for Digital Circuit Synthesis and Layout
Assignment #1 – FPGA Maze Router with Multi-Fanout Nets

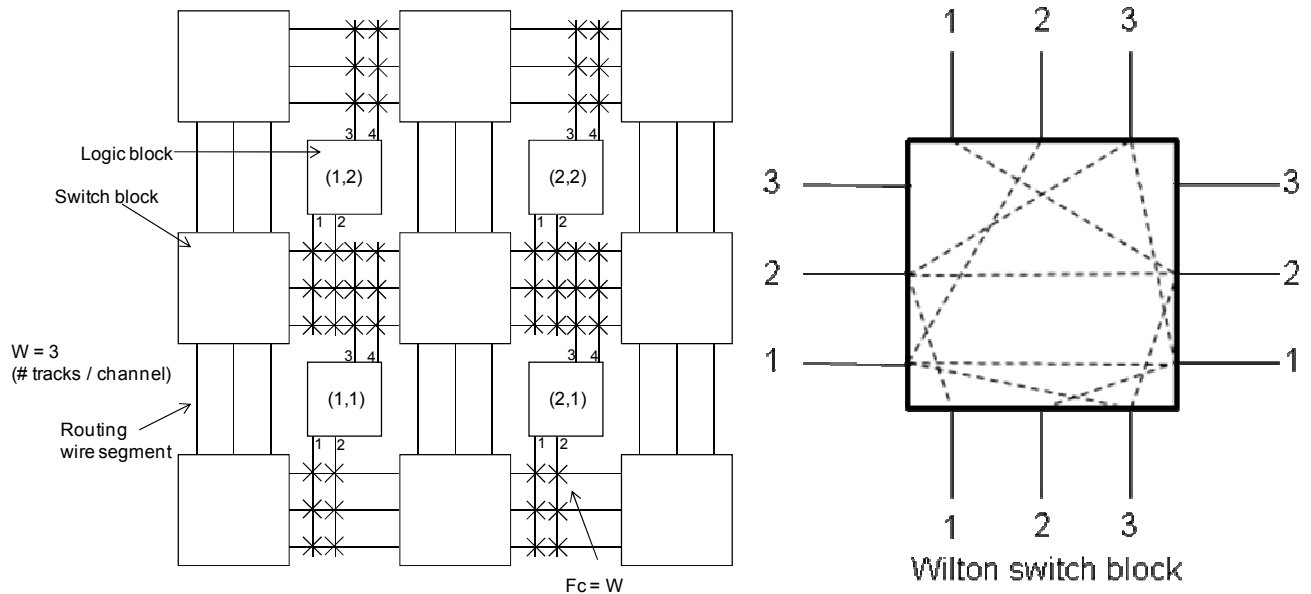
January 2010

J. Anderson

Assignment Date: January 27
Due Date: February 10 (before lecture begins)
Late Penalty: -1 mark per day late, with total marks available = 10

You are to write an implementation of the FPGA maze router described in class and innovate to minimize wire usage for multi-fanout nets. You must have your program display its progress with X11-based graphics. A C-based graphics package is provided on the course web page. You may use either Linux or Solaris.

You should use the FPGA architecture described in class, as illustrated in the figure below. The figure shows the pin numbering scheme, and the x,y position scheme. You should use the Wilton switch block, as shown below. In the Wilton switch block, the diagonal (turning) connections are rotated by one track. Note: the entire switch block is not shown, just some of the connections. Make sure that you understand how the complete switch blocks look, for any value of W.



Wilton switch block formal definition:

- East-West: track i connects to track i
- North-South: track i connects to track i
- West-North: track i on West connects to track $i+1$ on North
- West-South: track i on West connects to track $i-1$ on South
- East-North: track i on East connect to track $i-1$ on North
- East-South: track i on East connects to track $i+1$ on South

Your program should take input from a file that has the following format:

The first line consists of one integer, n , where n is an integer giving the $n \times n$ dimensions of the chip in logic blocks. The grid cells are numbered from 1 to n in each dimension.

The second line indicates the number of tracks per channel to use, W .

The next set of lines has the form "X1 Y1 P1 X2 Y2 P2". Each of these lines gives a pair of pins to be connected. The first pin is attached to the block at location X1,Y1, and uses pin number P1, where P1 = 1, 2, 3 or 4. The second pin is specified in

the same manner by X2, Y2 and P2. P1 is the **source** pin (the driver); P2 is the **sink** pin (the load). This list is terminated by the line: -1 -1 -1 -1 -1.

Example input file:

10	(10 x 10) grid
3	(3 tracks per channel (W = 3))
1 2 4 4 4 1	Pin 4 on block at (1,2) connects to pin 1 at (4,4)
3 3 4 8 9 3	Pin 4 on block at (3,3) connects to pin 3 at (8,9)
-1 -1 -1 -1 -1 -1	(end of pin pair list)

Your program must be able to display the routing solution for all of the connections in the test file using the graphics display. For debugging purposes, you may find it helpful to write your program so that it can display the progress of your algorithm as it routes each step for each connection; that is, you may wish to display each step of the router expansion (though this is not mandatory for this assignment). You should test your program on the following test files located on the course web page:

fcct1_10, fcct2_10, fcct3_10 and fcct4_10

Note that in addition to routing each circuit with the value of W given in the input file, you will need to find the smallest value of W for which the test circuits will route successfully.

Observe that, in the test files, there are some connections having the same driver pin. These cases represent multi-fanout nets; that is, nets with more than one load pin.

What to do and hand in:

1. A listing of your program.
2. The location of the executable file (on EECG or ECF), with instructions on how to run it. Please set the permissions so that I can run it.
3. A paper plot of the results from the four test files (using the value of W given in the test files). The graphics package allows you to do this. For this step, each driver/load connection on a multi-fanout net should be routed *independently*. Meaning that the paths from the driver pin to each load pin on a multi-fanout net should **not** share any routing wire segments (aside from the sharing the driver pin itself!).
4. The smallest number of tracks/channel (W) that your program could successfully route the test circuits in. That is, your program should be capable of varying the number of tracks per channel, and you are required to find the smallest number of tracks per channel that your program will successfully route the circuits in. Try different connection ordering schemes to reduce W. Explain any tricks you applied that were successful in reducing W.
5. Innovate to enhance your router to produce better solutions for the multi-fanout nets. In general, the total number of *used* routing wire segments can be reduced *sharing* routing segments among the paths connecting different source/load pin pairs on a given multi-fanout net. Reducing the number of used wire segments is beneficial to power consumption. In a table, report the total number of used routing segments, with and without your optimizations. In a table, report the minimum W for each test file, with and without your optimizations.
6. Hand in a two-page description of the flow of your software, describing the major routines and data structures, and how they interact. Where you were faced with choices in your implementation of the algorithm, indicate what choices you made, and why. Describe how you altered your router to handle multi-fanout nets, and the different approaches you investigated and why you selected the one that you did.