GREEDY CHANNEL ROUTER IMPLEMENTATION IN FORTRAN

Ray S. Linton
5th Year Microelectronic Engineering Student
Rochester Institute of Technology

ABSTRACT

During this project, the Greedy Channel Router was implemented in FORTRAN. This will allow students to break their general routing problems into channel routing problems to be done by the program. Since many students know FORTRAN, it will also be possible to build on the program and develop a more general router in the future [1].

INTRODUCTION

The main goal of any routing scheme is to join all pins having the same electrical node number. It is generally desirable to route the wiring paths efficiently. This is especially true in the design of an integrated circuit since a smaller chip is less costly and operates at a faster speed. Thus, a second goal of a router should be to minimize the use of chip real estate. The router should be implemented on a computer in such a way as to make it reliable and repeatable, since a third goal is to achieve design automation.

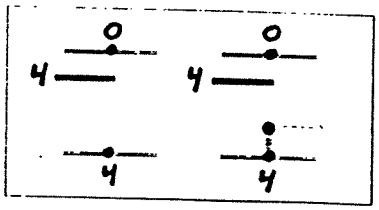
Specifically, a channel router wires between two parallel pin sets within a rectangular bounding box. These pins may belong to IC functional cells or gate arrays. Joining all pins with the same node will effectively join the cells. To accomplish this wiring, two levels of conduction paths are needed. One level is used for vertical wires while the second is used for horizontal wires. The horizontal and vertical wires do not short when they cross over each other due to an insulating layer between them. A via may be made in the insulating layer when it is desired to have a horizontal/vertical connection. The upper conduction path is usually a metal, such as aluminum. The lower conduction path may consist of a diffusion, doped polysilicon, or a metal. The insulating layer usually is silicon dioxide. The vertical wiring is done in a column, and the horizontal wiring is done in tracks. The number of tracks defines the channel width.

A channel router may be part of a more general area routing scheme. The area router first breaks a general problem into pieces or component channels. The channel router then performs each partial routing. The area router joins all the solutions together to form the final routing solution [2]. In this case, the area routing package is only as effective as the channel router.

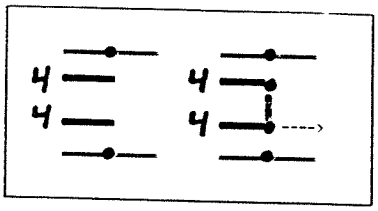
During this project, the Greedy Channel Router was implemented in VAX FORTRAN. The Greedy Router has several advantages over other channel routers. The algorithm uses a left to right, column by column approach. This avoids vertical wiring problems and makes the computer code easier to understand and modify. Other routers, such as Deutsch's Dogleg, have difficulty when there are vertical cycles in the pin net numbers. For instance, a cycle would be a pin set (top,bottom) of 1,2 followed by a pin set 2,1 in a future column. The node 1 has to cross the channel as well as the node 2. The Greedy Algorithm can deal with this situation, if necessary, by extending the right end of the channel and joining the nodes in a spill over region [2]. The algorithm is also unique in that it allows a node to occupy more than one track at a time. This allows conflicts to be resolved in a later column. If the channel proves to be too narrow to bring in a new node number from a pin, other channel routers do not complete. The Greedy Router can automatically widen the channel when it runs out of room. In fact, the authors claim that a proper implementation of the algorithm will yield a program that always completes a routing. Routers such as Deutsch's Dogleg, the Revised Dogleg, and the Least Cost Path algorithm do not have these automatic adaptability features. Other channel routers are also significantly more difficult to implement [3].

The input to the Greedy Router is a list of net numbers belonging to pins along the channel. A top and bottom pin in the channel defines a column. The router performs a series of six steps for each column. These steps are done in order of priority. Once the router completes a column, it never returns. Figure 1 illustrates these steps with before and after diagrams. Step one performs minimal pin connections by bringing the pins in the current column into the channel. If a track with a pin's net number exists, the pin may be connected to it. However, if an empty track is closer, a new track with the pin's net number will be started. Future steps may alter this connection if required. Step two attempts to join all tracks having a given net number together. This is called 'collapsing split nets.' The collapses which free the largest number of tracks are given priority. Step three attempts to jog all split nets, that could not be collapsed in step two, closer together. Step four jogs tracks towards their target edges. The target edge for a track is either the top or bottom channel edge, based upon where the next pin with the same net number as the track is located. Step five automatically widens the channel if there was no room in step one to bring in new pins. Step six extends all tracks which continue to the right into the next column. It is also necessary to do a wrap up step to determine if the routing is complete. If there are more pin sets, or there are un-collapsed split nets, the six steps must be performed again [4].

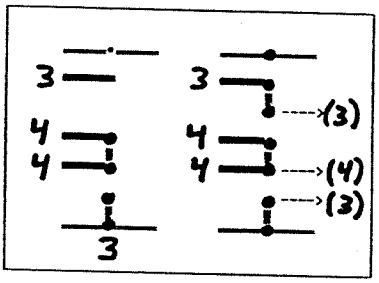
Figure 1: Greedy Router Steps



STEP 1: BASIC PIN CONNECTIONS
Connect new pins to an existing track or begin a new track. Use a minimum of vertical wire in all cases.

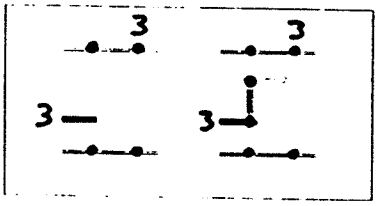


STEP 2: COLLAPSE SPLIT NETS
Reduce the number of tracks needed by connecting ones having the same electrical net. Optimize by doing net collapses which free the largest number of tracks.

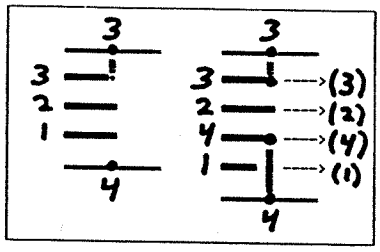


STEP 3: JOG SPLIT NETS
There will be split nets that step two could not collapse together. Jog the these tracks closer to each other if possible.

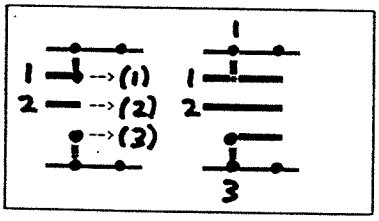
STEP 4: JOG TOWARDS TARGET EDGE
Move tracks closer to their target edges. A target edge is where the next pin with that track's net number exists.



STEP 5: ADD NEW TRACKS IF NEEDED
If step one failed to wire the new pins (due to lack of channel space), add a new track to the channel and wire the pins now.



STEP 6: EXTEND TRACKS
Terminate all tracks whose net number does not continue. Extend all active tracks to the next column.



WRAP UP: ARE WE DONE ?
If there are more pin sets to do we continue. If we are done, but there are still split nets, continue. Otherwise, we are finished.

RESULTS

The Greedy Channel Routing Algorithm was implemented in FORTRAN at Rochester Institute of Technology using a Digital Equipment Corporation VAX/VMS system. The software consists of a main program called ROUTER and six subroutines. The function of each is described in Figure 2.

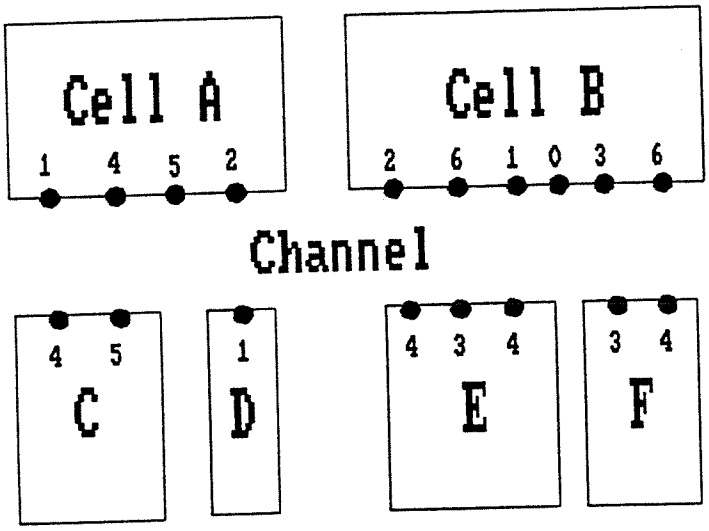
FIGURE 2: Parts of the Router Implementation

ROUTER	Main program, performs six steps plus wrap up. Calls subroutines.
HELPROUT	Help facility, displays help files on screen.
CHECKRT	Checks for pins of a given net number to the right of the current column.
EXTEND	Extends active tracks into the next column.
TRACKADD	Part of step five, widens the channel.
OUTSUB	Sends output matrix to a file when called [5].
OUTSCR	Sends output matrix to screen when called. Attempts to overlay steps on screen to permit viewing of the routing in progress.

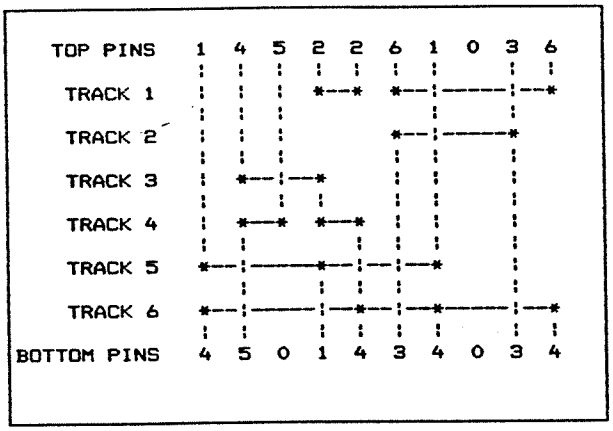
There are four help screens a user may view when the program is started. These describe the program, the input variables, the output format, and the references used. Next, the user is prompted for various output options. The user then enters routing parameters and the pin set net numbers along the channel. The program performs the routing and displays the results as desired. Only text characters are used in the output to avoid problems with terminal compatibility. The characters representing vertical lines, horizontal lines, and vias may be easily changed in the main ROUTER program if desired.

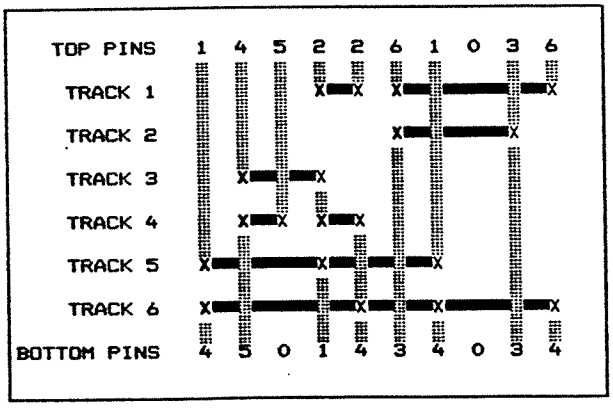
The router has been successfully implemented. Figure 3 shows a simple channel routing problem, the input pin sets, and the text based output. A visualization of the two conductor levels is also shown. Note that all pins with the same net number have been joined. Different net numbers do not short at any location. The program is capable of adding tracks if needed. The program will also expand the channel to the right by adding null pin sets if there are tracks that could not be joined in the user defined columns.

Figure 3: A Simple Routing Problem



COLUMN NUMBER	1	2	3	4	5	6	7	8	9	10
TOP PINS	1	4	5	2	2	6	1	0	3	6
BOTTOM PINS	4	5	0	1	4	3	4	0	3	4





The output of the program may be altered by changing input parameters such as the initial channel width, the minimum jog length, and the steady net constant. In this way, users will be able to optimize the routing done to conform to their applications [6]. The program code includes many comments so that a future programmer may make modifications. A user's guide and a programmer's guide are included in the appendix. This, along with the on-line help screens, make the package fairly easy to use.

SUMMARY

The Greedy Channel Router was successfully implemented in FORTRAN. Students will now be able to break their general routing problems into channel routing problems and use the program. This will save effort and serve as a valuable educational tool concerning routers. If the user chooses to display every step on the screen, he/she may watch the program making the wiring decisions. It is easier to understand an algorithm if one can make up an example and watch it run. Since many students learn FORTRAN, it will be possible for users to make modifications and extensions to the program in the future.

REFERENCES

[1] Amar Mukherjee, Introduction to nMos and CMOS VLSI Systems Design, (Prentice-Hall Publishers, New Jersey, 1986), p. 314-324.

[2] Amar Mukherjee, Introduction to nMos and CMOS VLSI Systems Design, (Prentice-Hall Publishers, New Jersey, 1986), p. 323-324.

[3] Rivest, R. L., and C. M. Fiduccia, "A Greedy Channel Router," 19th Design Automation Conference, June 1982 (Las Vegas), pp. 418-424.

[4] Amar Mukherjee, Introduction to nMos and CMOS VLSI Systems Design, (Prentice-Hall Publishers, New Jersey, 1986), p. 326-333.

[5] R. Pearson, Initial OUTSUB Output Subroutine, Rochester Institute of Technology, 1988.

[6] Rivest, R. L., and C. M. Fiduccia, "A Greedy Channel Router," 19th Design Automation Conference, June 1982 (Las Vegas), pp. 418-424.