Topics in Systems Engineering-IC Design

Detailed Routing

Shoukath Ali Mohammad
System and Circuit Technology, Heinz Nixdorf Institute
Universität Paderborn
Paderborn, Germany
shoukath@mail.uni-paderborn.de

***Abstract*—This paper presents Detailed routing which is an important step in the process of routing in VLSI physical design. A brief review of this routing technique is provided. In VLSI physical design flow after the process of placement, the routing procedure determines the accurate paths for the interconnection of the terminals on the circuit board from the nets. These accurate paths of nets should meet the design rules provided by the chip foundries to make sure that the design can be manufactured. Detailed routing is to make all the connections of nets to achieve complete routability otherwise the chip may fail to perform its functionality.**

I. INTRODUCTION

Detailed routing is an important step in the process of integrated circuits (ICs) design. The very next step after Global routing is Detailed routing. Global routing is used to find out roughly sequence of regions through which each of the net must flow or pass through. Subsequently to Global routing we must take up the routing regions one at a time and complete the routing in each of them. This phase is called Detailed routing. In a typical VLSI chip interconnection is not done on a single layer. There are often several layers available. Interconnection is done on these layers. Typically, two of those layers are metal layers. And other layers may be diffusion or polysilicon layer. Layers of two different nets should not intersect on the same layer as this may lead to short circuit.

II. TERMINOLOGIES

Before any further details on detailed routing is discussed it is important to know some terminology.

- *Terminal:* A point of contact for closing a connection.
- *Net:* A logical representation of a terminal.
- *Branch:* The vertical connections in the routing area.
- *Trunk:* The horizontal connections in the routing area.
- *Tracks:* The paths on which the truncks are laid in the routing region.
- *Span*: The space occupied by the net on the routing track.

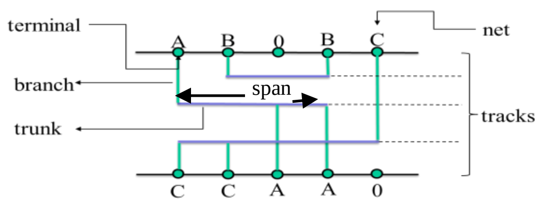


Fig.2.1: Simple diagram of routed terminals.

III. CONSTRAINTS IN DETAILED ROUTING

There are two types of constraints in detailed routing, Horizontal constraints and Vertical constraints. These constraints are to be fulfilled to perform routing. These constraints are represented using horizontal constraint graph and vertical constraint graph. These graphs are used to decide number of tracks to be used before routing and to decide the priority of terminals to be routed first.

A. Horizontal Constraints

When the horizontal span of two terminals occupy common area then it is called horizontal constraint.

Consider the nets A and B placed on the top and bottom of the routing area as shown in Fig.3.1. The net 0 represents no connection. In the above figure we can see that net A and B have common span that is they occupy common area in the 2nd and 3rd columns. To avoid this constraint each terminal is to be routed is assigned a separate track.

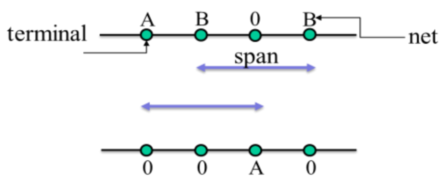


Fig.3.1: Example of horizontally constrained and unconnected terminals.

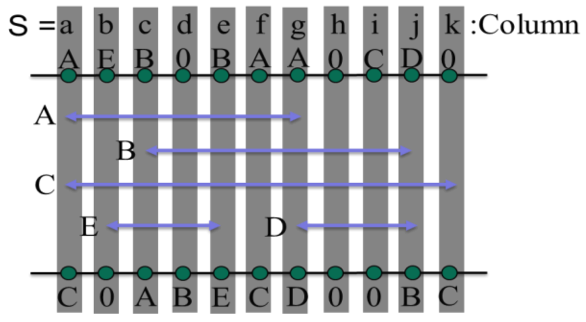


Fig.3.2: Example with all the tracks divided into columns. Reference [1]

These constraints can be represented by a horizontal constraint graph. A horizontal constraint graph is used to determine number of tracks to be used for routing in the initial stage.

Consider the tracks placed in the Fig.3.2. in the order [A, E, B, 0, B, A, A, 0, C, D, 0] in the top and [C, 0, A, B, E, C, D, 0, 0, B, C] at the bottom. These tracks are divided into columns S = {a, b, c, d, e, f, g, h, I, j, k}. The span of each net is shown by the purple arrow mark. In the column S(a) both the nets A and C start their span, so they have horizontal constraint. Nets A and C pass through column S(b) and net E begins its span so in column b there are 3 constraints. Net B starts its span from column c and nets A, C and E pass through it. The total number of constraints in column S(c) is 4. Column d has nets A, B, C and E. So, then total number of constraints in S(d) is 4. Similarly, for S(e) number of constraints is 3, S(f) is 3, S(g) is 4, S(h) is 3, The maximum S(i) is 3, S(j) is 3 and S(k) is 1.

Number of constraints in a column is 4 so minimum number of tracks needed for initial routing are 4. These constraints can be represented by a horizontal constraint graph(HCG). This graph is an undirected graph where all the vertices represent nets and denoted by corresponding net assigned to the terminal. The connection between vertices is represented by edges and these edges represent constraints between the terminals. The horizontal constraint graph in Fig 3.3 is the representation of example in the Fig 3.2. From the HCG we can see that vertices E and D do not have an edge connection which directly reflects that nets E and D do not have horizontal constraints which can be seen in Fig3.2 as nets E and D share the same track.

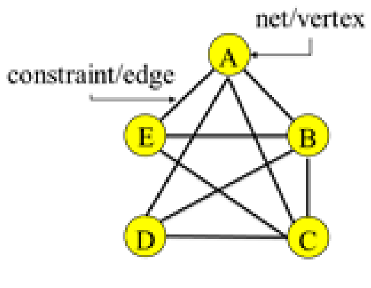


Fig.3.3: HCG. Reference [1]

B. Vertical Constraints

When two terminals have pins in the same column then it is called as vertical constraint.

Consider the scenario as shown in Fig.3.4. we have nets [A, E, B, 0, B, A, A, 0, C, D, 0] in the top of the routing area

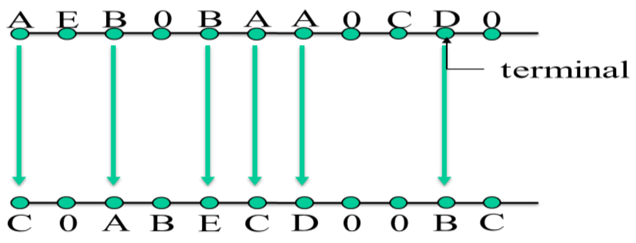


Fig.3.4: Example for vertical constraints and order of routing. Reference [1]

and nets [C, 0, A, B, E, C, D, 0, 0, B, C] in the bottom of the routing area. In the first column net A is above net C which is a vertical constraint. To overcome this constraint net A is assigned a track before assigning a track to net C. The direction of the arrow mark indicates that net C is to be assigned a track after assigning a track to net A. This procedure is followed to all the columns from left to right. From this we can see that net A is to be assigned a track before assigning a track to net D and net C. Net D must be assigned a track before net B. Net B should be assigned at rack before nets A and E.

The above conclusions can be represented by a graph. This graph has all vertical constraints in it. This graph is called as vertical constraint graph(VCG). This graph is a directed graph where all the vertices of the graph represent the nets and represented by corresponding nets assigned to the terminals and the edges represent the vertical constraints. The direction of the edges represents the priority of the assignment of the order of nets to tracks. The vertical constraint graph in Fig.3.5. represents the vertical constraints in Fig.3.4. There may be a case when two nets have equal priority, that is both the nets are above each other in different columns. When this happens, we do net splitting we do net splitting and add an additional track for connection.

In column 1 net A is above net B and in column 3 net B is above net A. So, this leads to a cyclic conflict. To overcome this conflict net B is split into two and an extra track is assigned to net B.

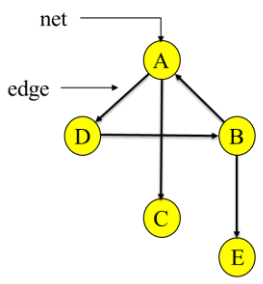


Fig.3.5: VCG. Reference [1]

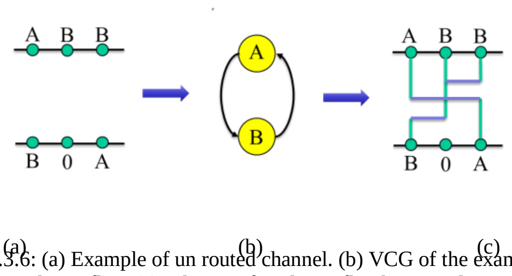


Fig.3.6: (a) Example of un routed channel. (b) VCG of the example showing cycle conflict. (c) solution of cycle conflict by net splitting and adding additional track. Reference [1]

IV. CHANNEL ROUTING

Channel Routing is a special case of detailed routing where pins are on either side of a channel and are routed on a horizontal routing region that has no obstacles. By convection the routing channels have uniform channel width as it follows row based layout and the terminals are placed on the top and bottom of the routing channel. Channel routing is done using Left-Edge algorithm or Dogleg algorithm.

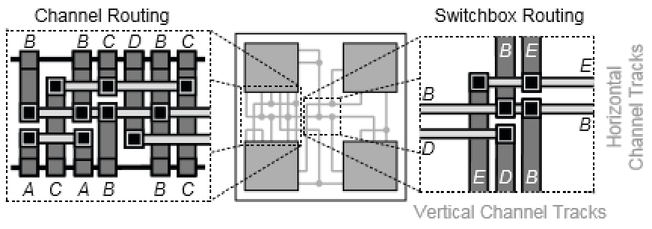


Fig.4.1: Example of two-layer channel and switchbox routing. The pins of each net A-D are all positioned perpendicular to the channel. Each layer has a preferred direction – one layer has only horizontal tracks and one layer has only vertical tracks. Vias are used if routing a net requires both horizontal and vertical tracks. Copied from [1]

A. *Left-Edge Algorithm:*

The left-edge algorithm is based on the vertical constraint graph, the zone representation and the left edge order. The left edge order and VCG are used to determine the assignment order of the terminals on the tracks. The zone representation determines the span of each terminal. The limitation of this algorithm is that only one horizontal segment or track is allowed for each terminal. The algorithm is as follows:

```
"Input: channel routing instance CR
Output: track assignments for each net
1. curr_track = 1 // start with topmost track
2. nets_unassigned = Netlist
3. while (nets_unassigned != Ø) // while nets still
//unassigned
4. VCG = VCG(CR) // generate VCG and zone
5. ZR = ZONE_REP(CR) // representation
6. SORT (nets_unassigned, start column) // find left-to-
//right ordering of all unassigned nets
7. for (i =1 to |nets_unassigned|)
8. curr_net = nets_unassigned[i]
9. if (PARENTS(curr_net) == Ø && // if curr_net has no
//parent
10. (TRY_ASSIGN (curr_net, curr_track)) // and does not
//cause conflicts on curr_track,
11. ASSIGN (curr_net, curr_track) // assign curr_net
12. REMOVE (nets_unassigned, curr_net)
13. curr_track = curr_track + 1 // consider next track"
```

Left-Edge Algorithm: (Copied from [1])

Left-edge algorithm has the following work flow. Line 1 is the topmost mask where the work flow starts. In line 2 nets_unassigned collects all the un assigned nets. Lines 3 to 5 are used to generate the VCG and zone representation of the un assigned nets. Then in line 6 all the terminals of the unassigned nets are taken in the left to right order. Now each unassigned net of the left to right order is assigned to the current track in line 1 if n has no predecessors in the VCG and it does not have conflicts with the previously assigned terminals in their corresponding tracks in lines 7 to 11. After a net has been assigned a path then remove it from the net_unassigned in line 12. When all the unassigned terminals are visited, increment the track index in line 13. Continue this procedure until all terminals are assigned tracks in the routing region.

Consider the following example in Fig.4.2 and apply left-edge algorithm to it. First curr_track is set to 1. All the terminals of the unassigned nets are collected. Now a vertical constraint graph and a zone representation are formed. After this a left to right order of all the un assigned nets is noted as [B, A, C, D, E, F, G, H, I, J].

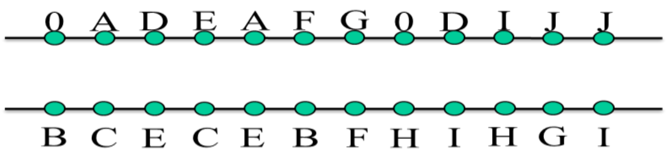


Fig.4.2: Example to demonstrate left-edge algorithm. Reference [1]

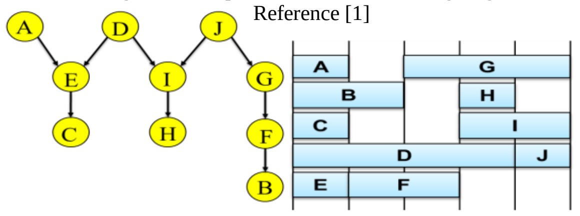


Fig.4.3: VCG. Reference [1], Fig:4.4: Zone representation. Reference [1]

Nets A, D and J are on the top of the VCG graph which have no predecessors. So, these nets are selected and assigned to curr_track = 1. Now in the left to right order A is first so it is assigned the track. Net D cannot be assigned a track yet as it has horizontal constraint with net A. Net J has no conflicts with net A so it is assigned to curr_track = 1 and track is assigned on the same track that of net A. Since nets A and J are assigned tracks they are removed from the zone representation, VCG and the left to right order data.

The updated left to right order is [B, C, D, E, F, G, H, I] Now curr_track is updated to 2. The next net with no predecessors is net D and net G. Since net D is first in left to right order it is assigned a track. Net G has horizontal constraint with net D, so it is not assigned any track. Net D is deleted from zone representation, VCG and left to right order list. The updated left to right order list is [B, C, E, F, G, H, I].

curr_track is updated to 3. The next nets with no predecessors are net E, G and I. Net E is first assigned track because it is first from the left. Next in the order is net G. As net G has no conflicts with net E it is added to curr_track = 3 and assigned on the same track. Net I have constraint with net G, so it is not assigned on this track. Nets E and G are removed from VCG, zone representation and left to right order list. The updated left to right order list is [B, C, F, H, I].

curr_track is updated to 4. The nets with no predecessors are C, F and F. As net C is first from left it is assigned on the track 4. Next is net F which has no constraints with net C, so it is also assigned on the same track. Net I also have no conflict with both the nets C and F which can be seen in the zone representation, so it is also assigned on the same track. Now nets C, F and I are removed from VCG, zone representation and left to right order list.

Now curr_track is updated to 5. The left to right order list is now [B, H]. As net B is first from left order it is assigned to the updated track. Next the only remained net is H. Net H have no conflict with net B, so it is also assigned on the same track. Now nets B and H are removed from the VCG, zone representation and left to right order list.

Now there is no net left which is not assigned, and all the nets assigned to the tracks have neither horizontal nor vertical conflicts. The routing of example in Fig.4.2 is complete using Left-edge algorithm.

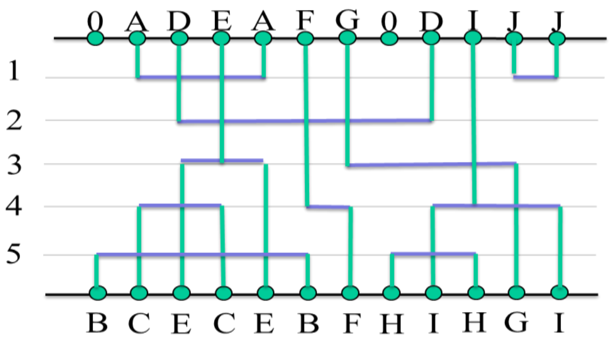


Fig.4.5: Solution of example in fig.4.2 with all the nets assigned tracks using Left-edge algorithm. Reference [1]

B. *Dogleg Algorithm*

This algorithm is based on left-edge algorithm. It improves left-edge algorithm by net splitting. Net splitting reduces the routing area on a track. This algorithm reduces the conflicts in VCG when encountered by cycles. In many cases this algorithm also reduces the number of tracks to be used. Dogleg represents L-shape. In Fig.4.6 we can see the net B split in the shape of L in two tracks.

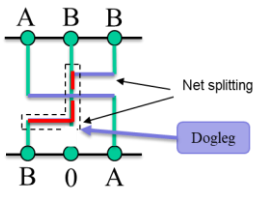


Fig.4.6: Demonstrating dogleg by net splitting. Reference [1]

This algorithm removes the cycles in the VCG. While applying dogleg algorithm it assumes that for the nets to which net splitting is to be done has no extra vertical tracks

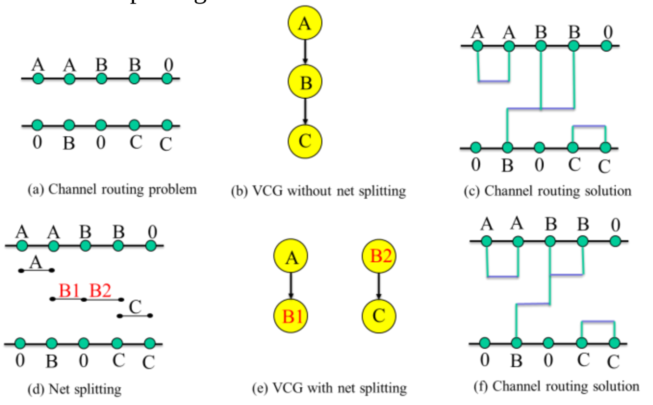

A A B B 0
0 B 0 C C
(a) Channel routing problem
A
B
C
(b) VCG without net splitting
A A B B 0
0 B 0 C C
(c) Channel routing solution
A A B B 0
A
B1 B2
C
0 B 0 C C
(d) Net splitting
A
B1
B2
C
(e) VCG with net splitting
A A B B 0
0 B 0 C C
(f) Channel routing solution

Fig.4.7: (a), (b), (c) demonstrating left-edge algorithm and (d), (e) and (f) demonstrating dogleg algorithm. Reference [1]

available. Consider the example shown in Fig.4.7. The channel routing problem (a) indicates the channel routing region with nets placed on the terminals as shown and the first VCG (b) is formed without net splitting. The corresponding channel routing solution (c) is formed using left-edge algorithm. In the next row the same example is shown with a difference in notation (d). Here since there are total three net B's it is split the span of it into two parts. The middle net B is denoted as B1 when approached from net B from the left and denoted as B2 when approaching net B to the right. This splitting of span of net into B1 and B2 is used to denote the net splitting during routing on the tracks. Now from this span split notation VCG (e) is drawn using net splitting. The VCG (e) is split into two parts because of net splitting net B. From the VCG nets are routed to the terminals on the tracks. In the channel routing solution (f) using dogleg algorithm is completely different from the channel routing solution (c) using left-edge algorithm. In (c) most of the second track is used by net B and more than half of track 1 is unused. But in

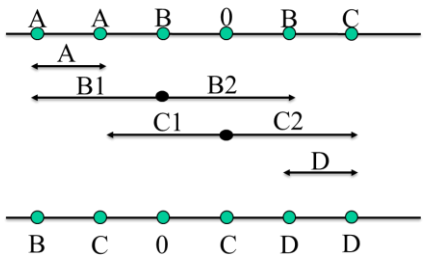


Fig.4.8: Complex example for dogleg algorithm

(f) the left-over space of track 1 is efficiently utilized by net splitting.Consider a more complex example shown in Fig.4.8 with net [A, A, B, 0, B, C] on the top and net [B, C, 0, C, D, D] at net is also denoted in Fig.4.8. As there are three net B's and three net C's. Net B is split the span of it into two parts. The middle net B is denoted as B1 when approached from net B from the left and denoted as B2 when approaching net B to the right. Net C is split the span of it into two parts. The middle net C is denoted as C1 when approached from net C from the left and denoted as C2 when approaching net C to the right. Now from this VCG is generated. Because of net splitting two VCG's are formed seen in Fig.4.9. Since net splitting is done now rest of the routing can be done using left-edge algorithm. Divide the tracks into columns as S = {a, b, c, d, e, f} to determine zone representation. Column S(a) has net A and net B starting from it. So, net B1 have horizontal constraints which implies S(a) = {A, B1}. In column S(b) has net A ending and net B1 passing through and net C1 start. So, net A, net C1and net B1 have horizontal constraint and S(b) = {A, B1, C1}. In column S(c) has net C1 passing, net B1 ending and net B2 start. Net C1, net B1 and net B2 have horizontal constraints in S(c) with S(c) = {B1, C1, B2}. Column S(d) has net B1 passing through, net C1 ending at and net C2 start. Net B1, net C1 and net C2 have horizontal constraints, so S(d) = {B1, C1, C2}. Column S(e) has net C2 passing through, net B2 ending at and net D start. Net C2, net B2 and net D have horizontal constraints, so S(e) = {C2, B2, D}. Column S(f) has net C2 ending at and net D ending. Net C2 and net D have horizontal constraints, so S(f) = {C2, D}. It can be concluded that maximum number of constraints in any column is three, so number of tracks to be used is three. First curr_track is set to 1. All the terminals of the unassigned nets are collected.

Now a vertical constraint graph and a zone representation are formed. After this a left to right order of all the un assigned nets is noted as [A, B1, C1, B2, C2, D]. Nets A, B2 and C2 are on the top of VCG with no predecessors. So, these nets are selected and assigned to curr_track = 1.

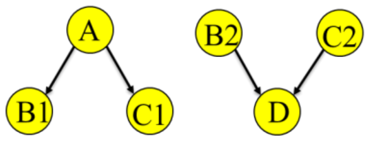


Fig.4.9: VCG using net splitting

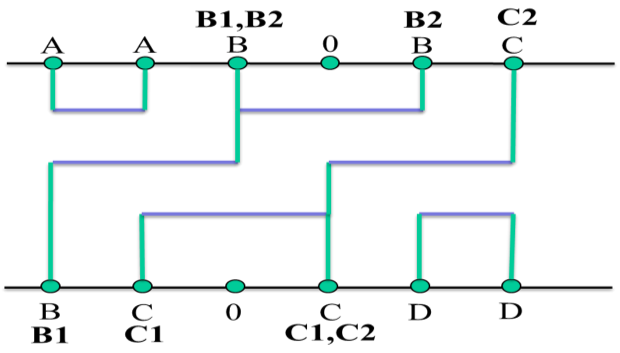


Fig.4.9: Channel routing solution

Now net the left to right order A is first so it is assigned the track. Net B2 has no conflicts with net A so it is assigned to curr_track = 1 and track is assigned on the same track that of net A. Net C2 cannot be assigned a track yet as it has horizontal constraint with net B2. Since nets A and B2 are assigned tracks they are removed from the zone representation, VCG and the left to right order data. The updated left to right order is [B1, C1, C2, D].

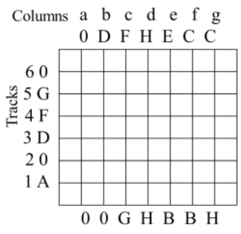
Now curr_track is updated to 2. The next net with no predecessors are net B1, C1 and net C2. Since net B1 is first in left to right order it is assigned a track. Net C1 has horizontal constraint with net B1, so it is not assigned any track. net the left to right order A is first so it is assigned the track. Net C2 has no conflicts with net B1 so it is assigned to curr_track = 2 and track is assigned. Net B1 and C2 are deleted from zone representation, VCG and left to right order list.

The updated left to right order list is [C1, D]. curr_track is updated to 3. The next nets with no predecessors are net C1 and D. Net C1 is first assigned track because it is first from the left. Next in the order is net D. As net D has no conflicts with net C1 it is added to curr_track = 3 and assigned track. Now nets C1 and D are removed from the VCG, zone representation and left to right order list.

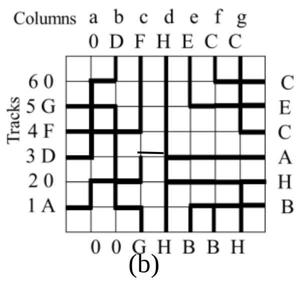
Now there is no net left which is not assigned,and all the nets assigned to the tracks have neither horizontal nor vertical conflicts. The routing of example in Fig.4.8 is complete using dogleg algorithm and the routed solution is shown in Fig.4.9.

V. SWITCHBOX ROUTING

Switchbox routing is a grid based model. Switchbox has fixed dimensions. Routing can be done on all four sides of the routing region. It is more complex than channel routing. Nets can be assigned horizontal tracks or vertical columns and can cross each other on several layers. When switchbox routing fails then a new track is added, and switchbox routing is repeated once again. Algorithm for switchbox routing can be derived from channel routing algorithms. An example of switchbox routing can be seen in Fig.4.1.



(a)



(b)

Fig.5:(a) 8x7 switchbox grid, (b) routed switchbox grid. Reference [1]

"Algorithms for switchbox routing can be derived from channel routing algorithms. switchbox routing algorithm has the following key improvements.

1. Pin assignments are made on all four sides.
2. A horizontal track is automatically assigned to a pin on the left.
3. Jogs are used for the top and bottom pins as well as for horizontal tracks connecting to the rightmost pins.**"** (Copied from [1])

Consider the following example with TOP = [0 D F H E C C], LEFT = [A 0 D F G 0], BOT = [0 0 G H B B H], RIGHT = [B H A C E C] as shown in Fig.5(a). Since it is a grid based model dimensions are fixed and it is an 8x7 grid based on the input. Column wise approach is applied to reach each net at top and bottom. Before the routing starts horizontal tracks are assigned to the terminals on the left most column (a). Now all the nets to the left are extended to the column (b). Net D is extended to track 6 as net D on the top is closer to track 6. A jog is applied to nett A and assigned to track 2. As there are no nets left to assign to top or bottom all the unassigned nets are extended to the next column (c). Net D is connected to the top of net D on track 6. Net G is assigned to track 1 as net G at the bottom is close to track 1.

All the unassigned tracks are Jog is applied to net A and is assigned to track 3. Net G is connected to net G at the bottom. Net A is the only unassigned net, so it is extended to the next column (d). Net H is connected from top to bottom net H. Unassigned nets A and H are extended to the next column (e). Connect net B to track 1 with the net B next to it. Connect the net E from top to track 5 as it is closer to net E on the right grid. All the un assigned nets A, B, H and E are extended to next column (f). Connect the net B of the bottom terminal to the net B on the right. Connect net C to track 6 as it is closer to net C on the right. Extend the unassigned nets B, H, A, E and C to column (g). Connect the net at the bottom terminal net H on track 2. Connect net C on the top to net C at right on track6. Assign net C to track 4.

Thus, all the nets are connected from all the four sides using switchbox routing. The solution of a switch box after routing is shown in Fig.5(b).

VI. OVER THE CELL ROUTING (OTC)

OTC is used when routing on more than three layers. It only occupies the cell area which is empty. Higher metal layers like Metal2, Metal3 or any other metal layer is used for routing while for others polysilicon and Metal1 layers are used. Standard cell routed area do not form obstacle to these metal layers. Routing can be performed on the entire chip area.

In the previous all the sections have dealt with only two-layer routing which means one layer is fixed for routing horizontal tracks and other for routing vertical columns. However, in advancing standard cell designs more than two layers are used for which the previously explained algorithms may not be sufficient. The routing area of inner region between the cell area deliberately Metal1 and Poly layers are used. As OTC is performed on higher metal layers, routing is performed over the cell that is routing is performed on the entire chip as these layers are no obstacles to standard cells. Routing for two metal layers using OCT can be seen in Fig.6.

OTC routing is performed as per the following the steps.

- Step 1: choose the terminals with their corresponding nets to be routed beyond the channel area.
- Step 2: Route these nets in the OTC area.
- Step 3: Select all the remaining nets and route them inside the channel routing area.

Over the time many algorithms were developed to efficiently solve the OTC routing. One of the earliest algorithms is the chameleon router developed by Braun et al. It works as follows. To minimize the total routing area, it generates topologies in either two or more layers for the nets and then assigns tracks to them. The key feature of Chameleon is that it considers the technology parameters on each layer. For example, designer can specify the distance between two wire segments in each layer and even the wire width.

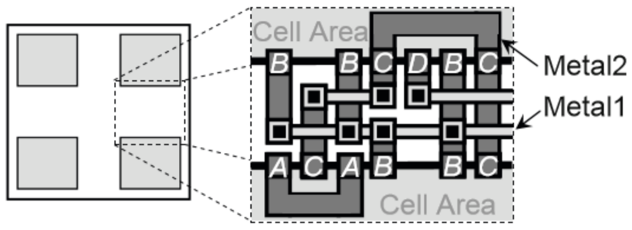


Fig.6: OTC routing with two metal layers. Copied from [1]

VII. SUMMARY

Detailed routing is invoked after global routing. Most of the algorithms explained are based on two routing layers. one layer is for routing of horizontal tracks and other for routing of vertical connections. For any of the routing algorithm in detailed routing to be performed channel routing nets have terminals at any of the opposite sides of the routing area, generally referred as top and bottom. Channel routing is implemented by two-layer routing algorithms namely left-edge algorithm and dogleg algorithm. Switchbox routing is a grid based routing model which has net terminals on all the fore directions of the channel routing region namely left, right, top and bottom. Channels are joined as switchboxes. In channel routing the width of the routing region is based on the number of tracks used for routing whereas for switchbox routing the dimensions of routing region is fixed. When routing is to be done over more than three layers Over-the-cell routing(OTC) is used. With need for low cost, high performance and low power consumption and dissipation horizontal and vertical constraints must be satisfied. In these algorithms must be constantly developed to meet the needs.

ACKNOWLEDGMENT

I would like to express my gratitude to Dr.rer.nat. Wolfgang Müller, for his support and guidance during the presentation and report writing for the seminar. I would also like to thank everyone who have directly or indirectly helped me through the seminar.

REFERENCE

[1] Andrew B. Kahng Jens Lienig Igor L. Markov and Jin Hu, “VLSI Physical Design: From Graph Partitioning to Timing Closure”, pp. 169-188