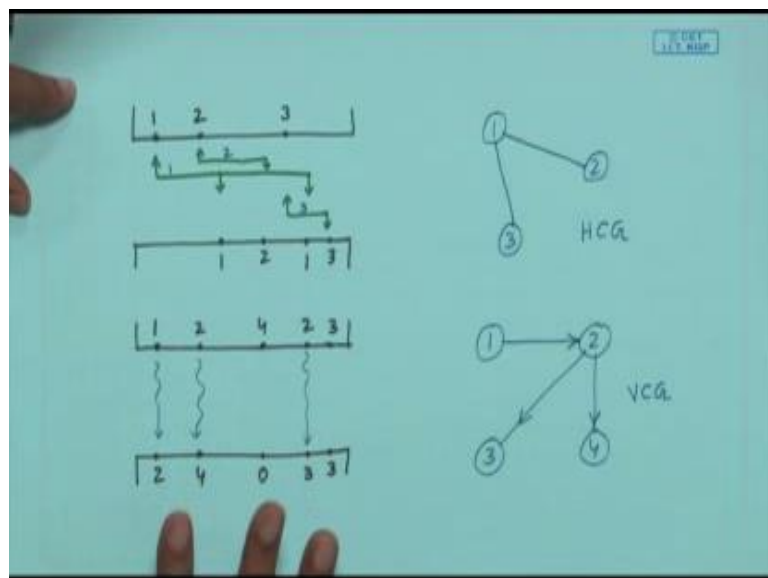


**VLSI Physical Design**  
**Prof. Indranil Sengupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 21**  
**Detailed Routing (Part II)**

So, let us continue the discussion so recall during our last lecture we had talked about the horizontal and vertical constraint graphs. Let us spend a few minutes on that once more before we proceed to a next part.

(Refer Slide Time: 00:38)



We see the concept the basic of horizontal constraint graph was like this. Suppose I have a channel, with some points like this. Say a net 1 was here, it was here, say another point was here. This mean I can represent it like this, that the net 1 as a horizontal span this.

Let us say net 2 has one point here and one point here so net 2 will be having a span a horizontal span here to here. Let us say I have another net 3 which is say from here to here so 3 will be having a horizontal span from here to here so while we talk about horizontal constraint graph so what was our idea so every net will be represented by a vertex and there will be an edge between 2 vertices if their horizontal spans are intersecting. Like here horizontal span of one this is 1, this is 2 and this is 3.

So, 1 and 2 are intersecting so there will be an edge between 1 and 2. 1 and 3 are also intersection so there will be an edge between 1 and 3, but 2 and 3 are disjoint, so no edge between 2 and 3. This is your horizontal constraint graph. Now vertical constraint graph as I said let us take another simple example. Let us say we have a generating pulse we have net 1 here, net 2 here. Let us say 2 here, 4 here, 4 here. This is no connection 0. Let us say we have another 2 here, 3 here and also 3 here let us say and 3 also here.

So, in this case you see the horizontal constraint here. One is on top of 2, 2 if on top of 4, 4 is on 0 you ignore this. 2 is on top of 3 and 3 on top of the same net you also ignore this. So here also there will be vertices corresponding to the nets. And now the edges will be having arrows. 1 on top of 2 so we represent it than arrow, 2 on top of 4 we represent it by an arrow. 2 on top of 3 and that is it. This is your vertical constraint graph. So the algorithm that we shall be talking about later now they will be using HCG and VCG to represent some constraints on the channel routing problem.

(Refer Slide Time: 03:57)

**Two-layer Channel Routing**

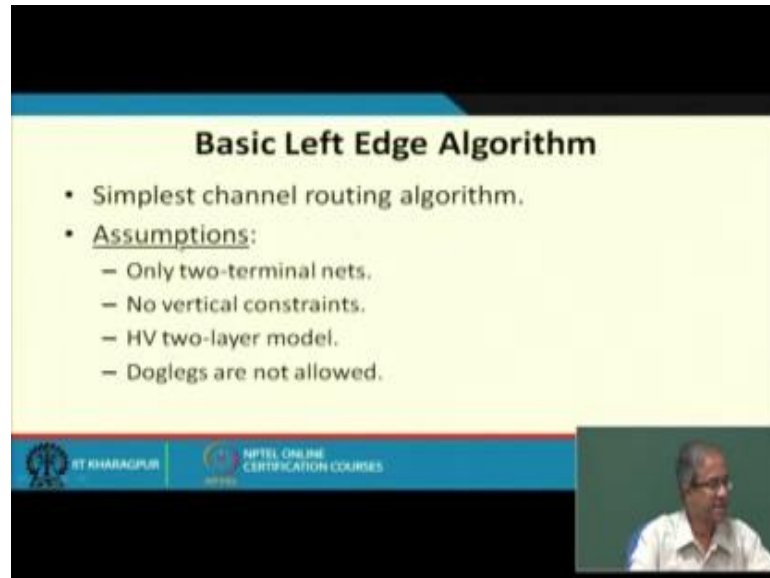
- Left-Edge Algorithms (LEA)
  - Basic Left-Edge Algorithm
  - Left-Edge Algorithm with Vertical Constraints
  - Dogleg Router
- Constraint-Graph Based Algorithm
  - Net Merge Channel Router
- Greedy Channel Router
- Hierarchical Channel Router

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, let us move ahead so we shall be starting by discussing some of the two-layer channel routing problems and algorithm. So they simplest the simplest one and you can say the basic one on which many of the other algorithms are based. It is called a left edge algorithm. So we shall be looking at the left edge algorithm and some of its variations. Then we shall be looking at a more powerful kind of a routing algorithm where there is a concept of a constraint graph, then greedy channel router these are the thing we shall be

discussing, but you can have a hierarchical channel router also which we shall not be discussing as part of this lecture series, but anyway there is a hierarchical channel router taking that is also available. So we start with the left edge algorithm.

(Refer Slide Time: 04:51)



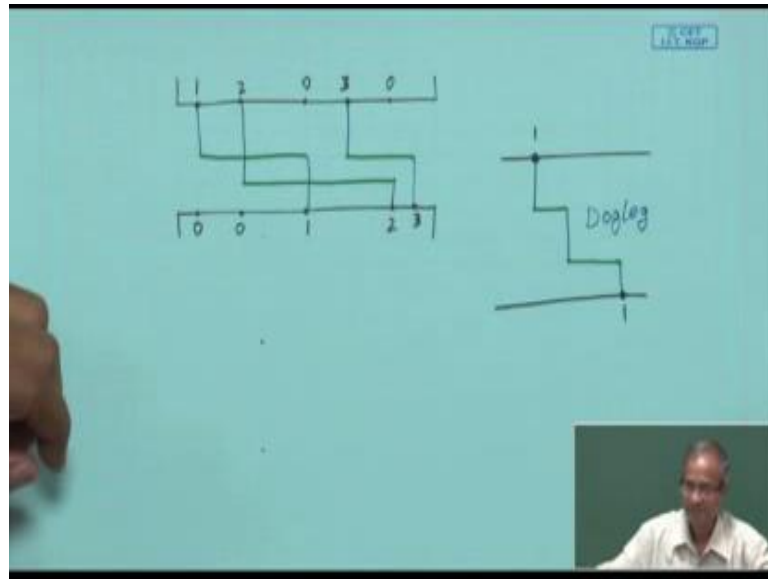
The slide is titled "Basic Left Edge Algorithm" and contains the following text:

- Simplest channel routing algorithm.
- Assumptions:
  - Only two-terminal nets.
  - No vertical constraints.
  - HV two-layer model.
  - Doglegs are not allowed.

The slide also features a logo for IIT Kharagpur and NPTEL Online Certification Courses, and a small video inset of a man speaking.

The basic left edge algorithm is the simplest as I said, but it has some very simple assumptions. The first assumption is that it does not handle multi terminal nets only 2 terminals. Like for example, terminal 1 we can have only 2 points marked as 1 only 2 points marked as 2, 2 points marked as 3 and 4 all 2 terminals nets and no vertical constraints. And it is a two-layer model mean all horizontal track on one-layer vertical on another layer. And there is something doglegs which are not allowed. Let us take a very simple example.

(Refer Slide Time: 05:35)



So, one channel instance which for this you can use the left edge algorithm is like this say 1 is here. 1 is here, but there is no connection here, there is no connection here. Let us say 2 is here 2 is here, but no connection here, no connection here. Something like this no vertical constraints. So while we allow the nets, so you can allow them like this one what is will track 2 you can have vertical tracks then we join them the trunks, something like this, right.

Now, the algorithm is and dogleg is something which is like this. Dogleg means if you are 2 points like say one on top layer one on bottom layer which you want to connect let us say both are same net. Then the horizontal segments instead of a single horizontal segments you can break it up into more than 2 horizontal segments on 2 layers. So this is called a dogleg, but I am not sure why it is called a dogleg because I never seen a dog with a leg like this, but this is called a dogleg, in the literature where the horizontal nets are split across 2 different tracks, right.

(Refer Slide Time: 07:12)

- Basic Steps:
  - Sort the nets according to the x-coordinate of the leftmost terminal of the net.
  - Route the nets one-by-one according to the order.
  - For a net, scan the tracks from top to bottom, and assign it to the first track that can accommodate it.
- In the absence of vertical constraints, the algorithm produces a minimum-track solution.

So, the left edge algorithm is very simple. It says so all the nets you sort by their x-coordinates route, the nets one by one according to the order. And for routing you see which is the first track available from the top which means you scan the tracks from top to bottom and assign it to the first track that can accommodate it, like this example if you again come back, you say one is the form the left this one comes first then 2 let us say I have 3 after that let us say I have net 3 here and 3 here.

So, because the x-coordinates of one comes first you route one to assign it to the first track. Then move on to the next net which is here x coordinates 2 comes first then route 2 you cannot assign it to the first track because they are overlapping, so you assign to the next track 2 is completed then comes 3, 3 is from here to here you see the first is available if the first track because one is finished. So 3 you can use the same tracker we use it so for connecting 3 your connection will be like this. So, 1 and 3 are sharing this same track , 2 closes the 7 track. So this is the basic left edge algorithm they may it works alright.

So some characteristics of this method is that well in the absence of vertical constraint which I have assumed so far there are no vertical constraints this algorithm is guaranteed to produce a minimum track solution. This can be very easily proved.

(Refer Slide Time: 09:02)

**Extension to Left-Edge Algorithm**

- Vertical constraints may exist, but there are no directed cycles in the VCG.
- Select a net for routing if both the following conditions are true:
  - a) The x-coordinate of the leftmost terminal is the least.
  - b) There is no edge incident on the vertex corresponding to that net in the VCG.
- After routing a net, the corresponding vertex and the incident edges are deleted from the VCG.
- Other considerations are the same as the basic left-edge algorithm.

BT BHARAGURU | NPTEL ONLINE CERTIFICATION COURSES

But you see in a practical problem ignoring vertical constraints is very restrictive. In general, there will be vertical constraint there will be columns with one pin on the top and one pin on the bottom. There will be vertical constraints and the basic left edge algorithm does not handle that kind of vertical constraint per say.

So, let us see how it can be handled. So some extensions we are proposing. So here I assuming that there can be vertical constraints, but again there is an assumption we assume there are no cycles in the VCG because cycles are a specific problem case, why?

(Refer Slide Time: 09:55)

NO DOGLEG

The whiteboard shows a routing grid on the left with three columns and three rows. A path is drawn from top-left to top-right, then down to bottom-right, then left to bottom-left, and finally up to top-left, forming a cycle. A large 'X' is drawn over this path. To the right, a directed graph with three nodes (1, 2, 3) is shown. Node 1 is at the top, node 2 is at the bottom right, and node 3 is at the bottom left. Directed edges connect 1 to 2, 2 to 3, and 3 to 1, forming a cycle.

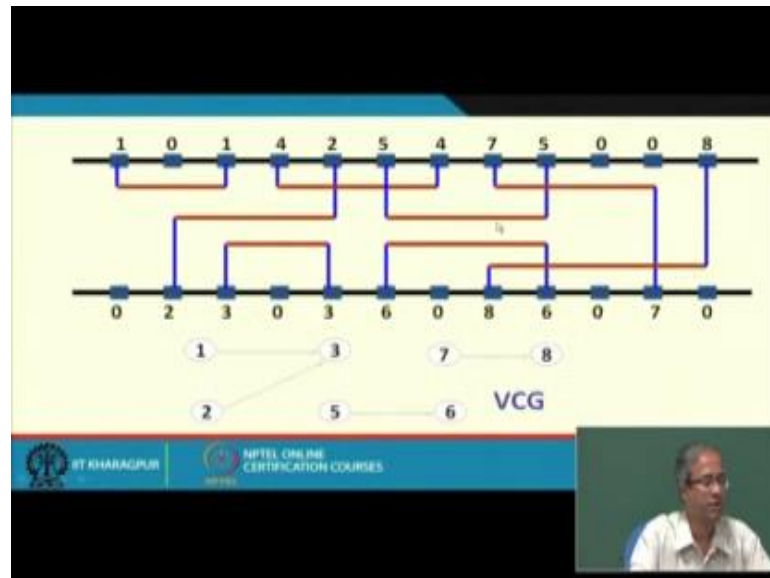
Let me try to intuitively justify. Suppose in a channel routing problem I have a column where 1 is here 2 is here. Well so far I am assuming here that I am allowing any dogleg kind of a connection the entire horizontal segments will be on the same track.

So, one on top of 2 means I have to assign here net 1 and some track and net 2 on a track which is below it. Why? Because in first column I have to make a connection like this, I have to make a connection like this so unless 2 is below one if 2 is above 1 there will be a short circuit. If while I make a blue, blue metal connection here and here, so 1 as to be among 2. Similarly say I have another column where 2 is here and 3 is here. So 2 is already lead out here and 3 must be below it. So 3 must be on another layer which is below it.

So, for this connection 2 will be connected to the net 2, and 3 will be connected to net 3, but suppose I have another column where 3 is on top and one is on bottom then we have a conflict. So already 1 is on top of 3 we cannot satisfy this. This cannot be satisfied. So the version of the algorithm they are considering now does not allow for this kind of cycle, where 1 to 2, 2 to 3 and 3 to 1, there is a cycle like this. 1 to 2, 2 to 3 like 2 to 3 and 3 to 1, this kind of a cycle is not allowed.

So, the only change modification where doing to the basic left edge algorithm is that, we select a net for routing just like in the left edge, if the x-coordinates of the left most terminal is the least just like the left edge algorithm and this is an additional constraint. There is no edge incident on the vertex corresponding to that net in the VCG. This, remember the statement I shall be explaining with help of an example. In shortly so once a net is routed the corresponding vertex and the edges are removed from the VCG and I repeat this process until all the nets are routed.

(Refer Slide Time: 12:36)



Let us take an example to illustrate so this is a channel routing problem which I am showing and this is a corresponding vertical constraint graph. You can check 1 is on top of 3, there is an edge, 2 is on top of 3, there is an edge, 7, 8 is here, 7 in top of 8, 5 6 is here. You see 4 does not appear here because if 4 there is no vertical constraint. There is no connection here, no connection here. 4 does not appear. So the idea this follows you see I cannot route 3 before. Because what will happen if I route 3 before, because I will be assigning the first track to 3.

Now, once I assign the first track with 3, I cannot route 1 because 1 is also having a terminal here. So there will be a short circuit. So that is why I start with only those nets which does not have any incoming edges. So there is no dependency on them. Like 1, 2, 7 and 5, this 4 nets can be handled just like the left edge algorithm without any problem. So let us do that. So if we sort by left edge, 1 comes first then 2, then 5 and then 7, 1 2, 5, 7 in that order you route. So what you get is something like this. So the net 1 is routed on the first track, 2 is routed on the second track, because there is an overlap. 4 does not appear so 4 you can also route. 4 is routed on the first track because this is available. 5 does a overlap so 5 is routed in the second track and 7 again it is available it is routed in the first track.

So, once they routed you delete 1, 2, 5 and 7 from the graph. So you are left with 3, 6 and 8 which does not have any incoming edges any more. So 3, 6, 8 you can handle

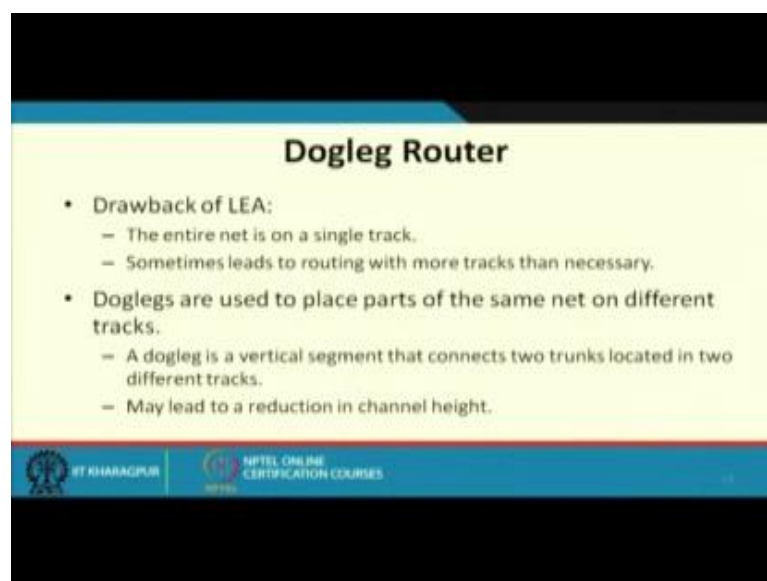


together. Among them 3 comes first, then 6, then 8 in that order you route. So first you route 3 so 3 gets routed here. Because you cannot use first track second track they are already occupied, then 5, 6 sorry 6. We route here. They also have to occupy on the third track because the errors are occupied then 8. And 8 requires a fourth track because you see this span of 8 is this much and the other 3 tracks are already occupied. So if that a solution and I require 4 tracks.

So, the algorithm is pretty simple. So you have the channel graph the horizontal and constraints and the vertical constraints graph. You have the channel definition. You consider only the nets at any given alternation. In every alternation we consider only the nets which does not have an incoming edge in the vct, which means there is no such case where that net a point a terminal is below and another point is above. No such case is there. So you consider only those nets and among them run the left edge algorithm. Once you do it remove them from the draft basically and repeat the process.

Again you see which of the nets that do not have any more incoming edges take them and run left edge algorithm again on them. Go repeating this will be getting a solution, but even that you can immediately see that if there was a cycle in the VCG you could not reveal you could not resolve that cycle even this technique. Because 1, 2, 3 they are in a cycle so neither you 1 nor 2 nor 3 you can pick up and route. So this algorithm does not allow the cycles in the VCG.

(Refer Slide Time: 16:44)



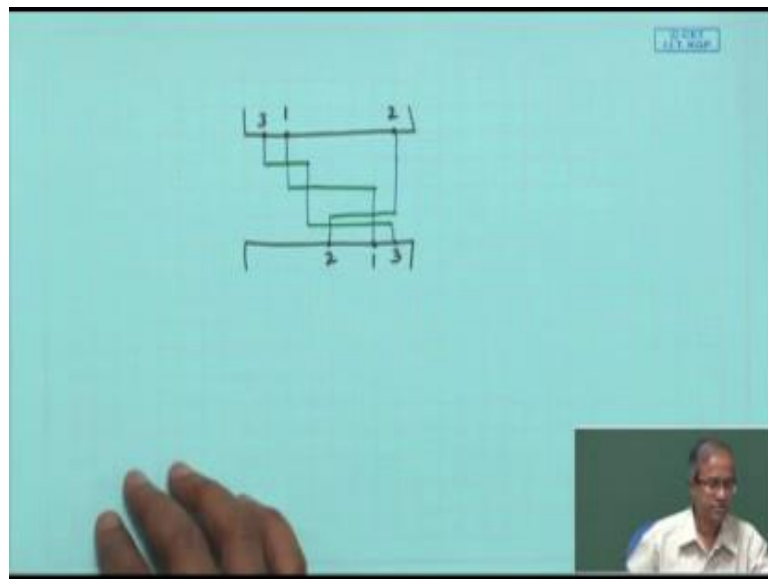
**Dogleg Router**

- Drawback of LEA:
  - The entire net is on a single track.
  - Sometimes leads to routing with more tracks than necessary.
- Doglegs are used to place parts of the same net on different tracks.
  - A dogleg is a vertical segment that connects two trunks located in two different tracks.
  - May lead to a reduction in channel height.

IFT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, the second extension we do here is that you see, dogleg we just mention what is dogleg. Dogleg mean the horizontal segments need not be laid on a single layer it can break it up into 2 trunks and 2 different tracks. The left edge algorithm and this extension as we are said that the entire horizontal segments were put on the same track, so it does not allow for dogleg. Now we shall see that if you allow for doglegs, then there can be some instances where you may require less number of tracks right let us take an example.

(Refer Slide Time: 17:32)



Let say I have a point one here, and a point one here. So they are connected like this let say. They are connected like this. Let us say I have another case a point a let us say 2 is here. In the point 2 we say here. So you can connect them like this, you can connect them like this. Now suppose I have a scenario, where I have a point 3 here, let say 3 is here and 3 is also here now here. You have 2 alternatives. Either you take another track like this, lay it out like this or what you can do, you can have connection like this. You have a segment like this, break it up here again bring it down, again break it up here then again bring it down. Well although this example does not illustrate, but I shall show example later that will require less number of tracks if you allow doglegs.

(Refer Slide Time: 19:37)

• Dogleg router allows multi-terminal nets and vertical constraints.  
– Multi-terminal nets are broken into a series of two-terminal nets.  
• Cannot handle cyclic vertical constraints.

ST KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, as I said doglegs if you use it may lead to a reduction in channel height. And the dogleg router algorithm that we shall be considering it allows multi terminal nets. And of course, vertical constraints, but no cycle in the vertical constraint graph still the cycle is not allowed right.

(Refer Slide Time: 19:54)

**Dogleg Example**

No dogleg  
3 tracks

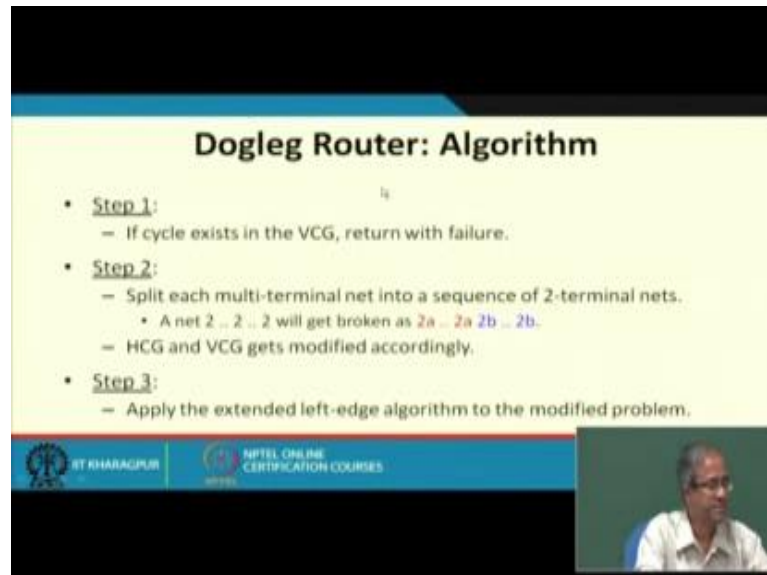
With dogleg  
2 tracks

ST KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, let us take an example of a dogleg. This is a channel routing problem and this is a conventional method using the extended left Lee algorithms which requires 3 tracks, but if you allow dogleg you said net number 2 is split between 2 tracks. And we are sharing

the first and third tracks, where two different parts of the segments and you can completely the routing in 2 tracks only so your number of tracks, getting reduced.

(Refer Slide Time: 20:29)



**Dogleg Router: Algorithm**

- **Step 1:**
  - If cycle exists in the VCG, return with failure.
- **Step 2:**
  - Split each multi-terminal net into a sequence of 2-terminal nets.
    - A net  $2 \dots 2 \dots 2$  will get broken as  $2a \dots 2a \ 2b \dots 2b$ .
  - HCG and VCG gets modified accordingly.
- **Step 3:**
  - Apply the extended left-edge algorithm to the modified problem.

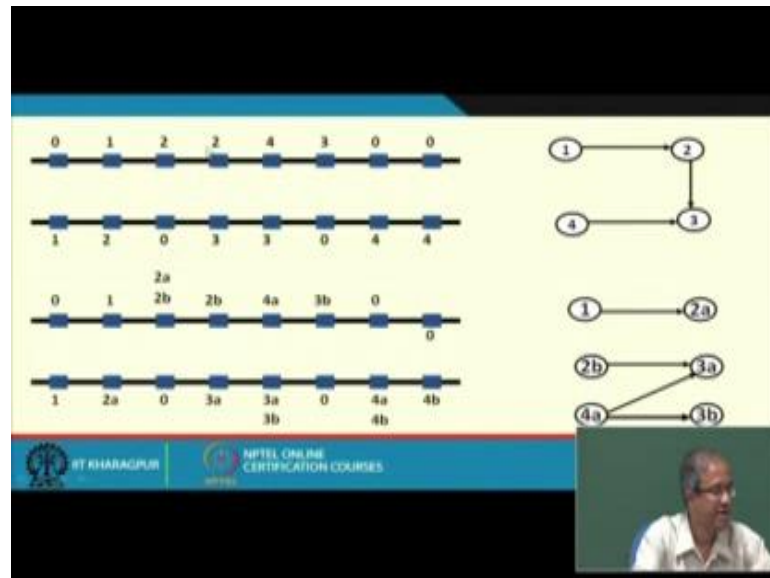
ST KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, the basic algorithm works like this. It checks if there is a cycle in VCG of course, it cannot handle the cycle it returns with failure. And for multi terminal nets because dogleg can exist only at multi terminal nets. For 2 terminal nets normally you do not use dogleg, already I showed an example earlier, but normally if 2 terminal nets it does not use dogleg. Use a dogleg only there are 3 or more terminal required.

So what you do is split each multi terminal net into a sequence of 2 terminals nets. Like you see if you had a net like a 2 somewhere later again a 2, somewhere later again a 2, it breaks it up into 2 sub nets. You can say you called the first term is 2a and second one is 2b like a, I should give an example how it is and once you do it you modify the horizontal and vertical constraints graphs accordingly and on this modified graph we apply the extended left edge algorithm.

But here you consider the nets 2a and 2b as to separate nets, not a single net. Which means the first segments you can layout independently the second segments you can also layout independently. This is one example.

(Refer Slide Time: 21:57)

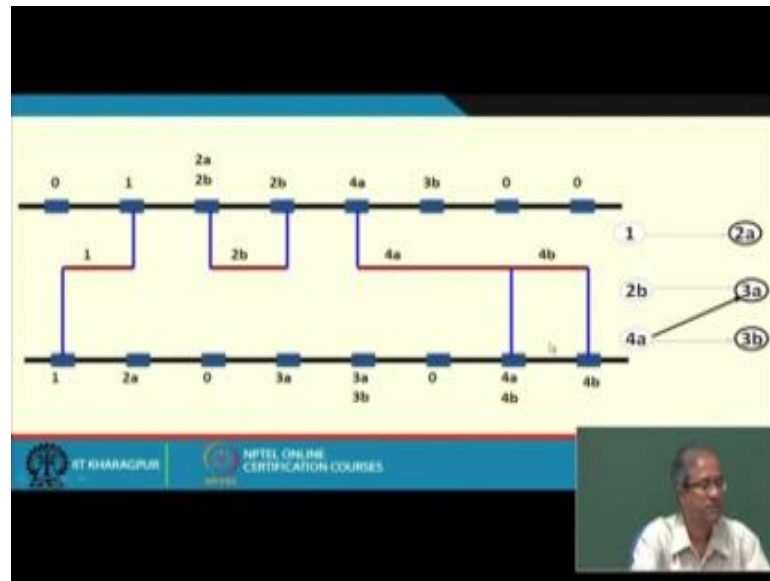


Let us take an example channel like this. So there are 4 nets and this is the VCG. So somewhere one is on top of 2, that is why this edge is there. 2 is on top of 3, this edge is there and 4 is on top of 3 this edge is there. Now you see 2 is a 3 terminal net, 3 is also a 3 terminal net, 4 is also a 3 terminal net. So we use this concept to nets to 3 and 4, but 1 is a 2 terminal net.

So, we modify our channel description as follows. So it is 2, 2 and 2. We divide into 2a and 2a and same terminal is also called 2b. There are 2 names 2a and also 2b. The second one I call 2b. Similarly, 3a, 3a and 3a, 3b, this one equals 3b. Similarly, 4a, 4a and 4b, 4b and once you do this and VCG; obviously, gets modified. So now, our constraints are one is on top of 2a this one, 2a this is no connection nothing 2b is on top of 3a this arrow. 4a is on top of 3a and 3b 4a is on top of both 3a and 3b nothing else. So this is now my VCG.

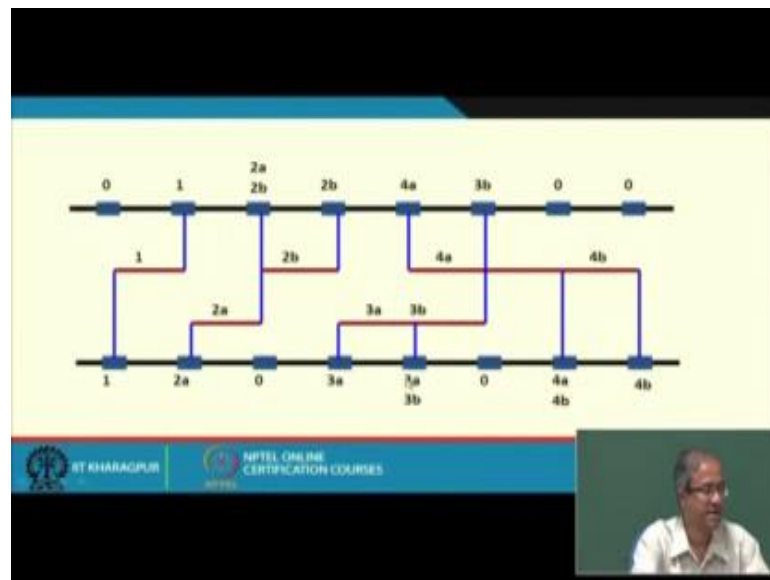
So, this is my modified channel description and this is my VCG. So I run the modified left edge algorithm on this VCG and this channel.

(Refer Slide Time: 23:40)



So, if you just work out, you can see that in this VCG, the nets that you are allowed to start with our 1a 2b and 4a because they have no incoming edges. So let us route 1a, 2b and 4a. They all can be placed on the same track because there is no overlap. Now once these are done we can work with 2a, 3a and 3b.

(Refer Slide Time: 24:04)



So, 2a will be lead on the next strap because already 2b as occupied here 3a 3b. Similarly, 3a was on this track this same point is there on the same track, but once you do this and if you connect the vertical and you will see, but although 4 track, 4 and tracks

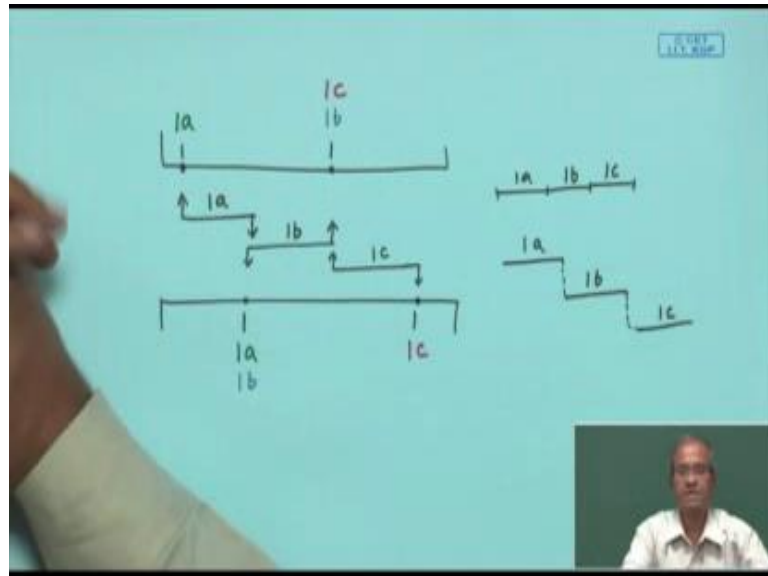
we are on the same horizontal track, the 2b we have broken up into a dogleg. And you are able to get a 2 track solution which using I mean without using dogleg you can check, but you cannot get a two-layer solution 2 track solution will be requiring minimum of 3 tracks for this.

So, here there are a few things that I need to just remind you, at well we have a channel routing problem. We express a channel as a set of points which are on the top and the top boundaries. The channel can be rotated ninety degrees also no problem same problem formulation can be can used for that case also. So in the basic left edge algorithm we only considered the 2 terminal nets and no vertical constraints. It was a purely greedy algorithm. We sorted the nets with respect to the left most x-coordinates and you place them to the earliest track possible the first track that is available we go on placing like this. Then we have an extension to that where we allowed vertical constraints, but no cycles. So we go on doing something which is very similar, but the only thing that we are not doing is that they are not considering all nets at the same time we only consider the nets which do not get involved in a vertical constraint at that point in time.

So, only the nets which do not have any incoming arrows there, you only take those set of nets and you route them using left edge algorithm, removed them from the graph repeat the process. So in this way you are assured that any step the set of nets that you are considering though once you place them there will be no other net we will tell that while I want to be on top of you there will no such constraint that may coming later. So once you place a net there will be no requirement that you have to insert a net on top of it may be below it, but not on top.

And the last algorithm that we have said that was an extension of this extended version again. So where we are considering multi terminal nets also. And each multi terminal net segments we are working up into 3 into 2 or more parts.

(Refer Slide Time: 27:09)



For example, if I have a net like this. So far I have say 1 here let say a 4 terminal net, 1 here and 1 here. So what I do here is that, the first segment I call them 1a and 1a. We said in the second segment I call is as 1b and 1b. The third segment I call is that, I call is at 1c and 1c. So actually I am splitting my problems into 3 sub problems, where this is my net 1a, here to here. This is my net 1b here to here. And this is my net 1c here to here.

Now, here once you do this we apply the left extended left edge algorithm after that, but the advantage that you gain is that because you are considering 1a, 1b and 1c as 3 independent nets. So it is possible that they are all laid out on the same track 1a, 1b, 1c like this on the same track. Or it may also possible that 1a is on one track 1b is on another track and 1c is on another track. So you can have a connection like this. So you can potentially have a dogleg kind of a scenario if you allow multiple segments of the same nets treated like this. So shall be continuing with our discussion on detailed routing in the next lecture.

Thank you.