

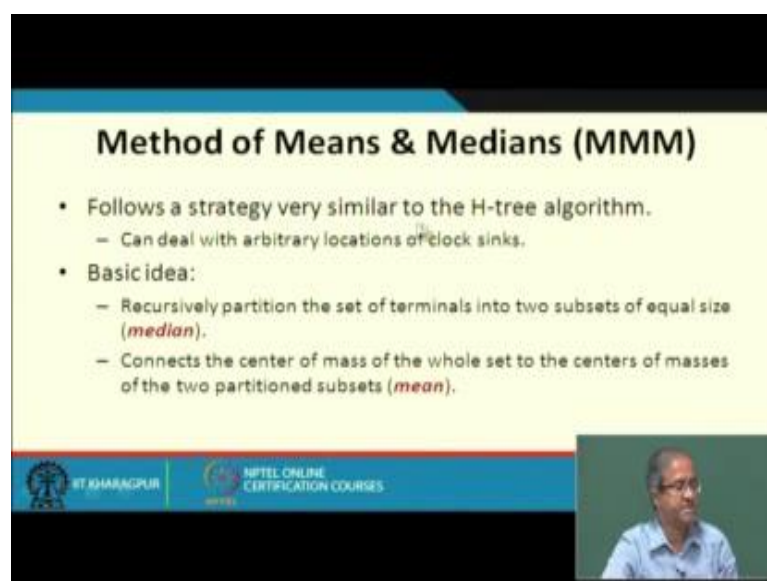
**VLSI Physical Design**  
**Prof. Indranil Sengupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 30**  
**Clock network Synthesis (Part 4)**

So, we continue with our discussion on clock tree design. We recall during our last lecture we talked about the H-tree and X-tree approaches for designing a clock tree. Both of these approaches were basically designing a 4 array tree structure, each node was connected to 4 other nodes at every stage. So, number of terminal points or sinks grew as a power of 4 - 4, 16, 64, 256 and so on. These 2 methods basically did not consider the exact location of the clock terminals per say. It independently designed or created the tree and produces a regular array of sink location across the surface of the chip, but the algorithms that we shall be trying to talk about now they take a different approach. They do not ignore the locations of the actual clock terminal points, what they do instead.

Let us look at where we require the clocks to be sent the actual clock terminal point or sinks. Now with respect to that let us try to systematically construct a tree. So, it that tree many look like H may not look like X, but it will be a tree nevertheless. So, this is the basic idea behind the methods that we shall be discussing now.


(Refer Slide Time: 02:16)



**Method of Means & Medians (MMM)**

- Follows a strategy very similar to the H-tree algorithm.
  - Can deal with arbitrary locations of clock sinks.
- Basic idea:
  - Recursively partition the set of terminals into two subsets of equal size (*median*).
  - Connects the center of mass of the whole set to the centers of masses of the two partitioned subsets (*mean*).

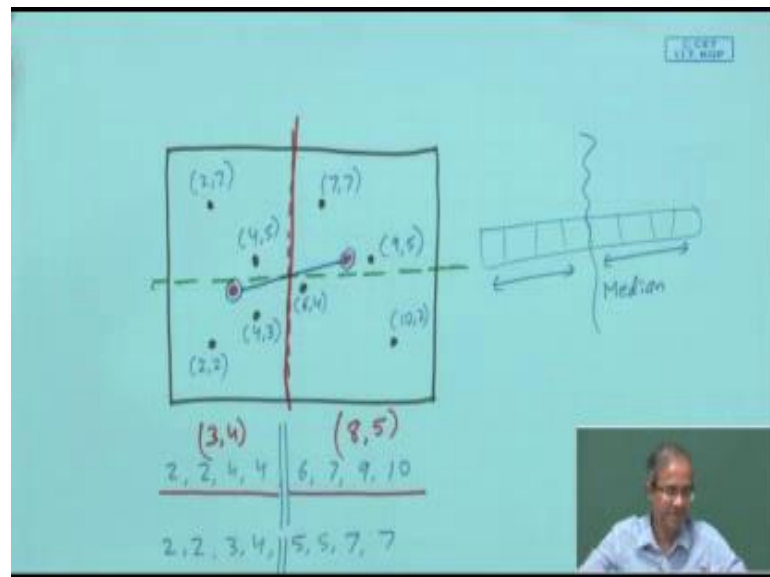
IT KHARAGPUR    NPTEL ONLINE CERTIFICATION COURSES



So, the first method that we talk about this is called the method means and medians or MMM. Here the first similarity is that it follows a strategy which is similar to the H-tree algorithm. Strategy in the sense that it can handle the sink locations anywhere you want. Because H-tree also as we move from one level to other you are getting locations in finer and finer and finer locations points as power of 4, where ever you need a clock print to be located to be sourced. So, you can carry out creating the tree up to a desired number of level you will be getting a point which is very close to the actual clock terminal point.

So, in this method, this MMM is called in short, the basic idea is you calculate median and mean successively. Well the way medians and means are calculated.

(Refer Slide Time: 03:48)



Let us try to explain that how you are calculating the median. Let us look at a very specific example I will illustrate with this. So, let us consider that I have a set of points. So, each of these points will be having some coordinates on a grid like structure if you consider, just I am noting down the coordinates - 2 2, 2 7, 4 3, 4 5, 6 4, 7 7, 10 2, 9 5. So, you see each of this points there having some x coordinates and y coordinates. If you see when you have a set of points or set of numbers, let us say I have a set of items. So, when I say I am taking a median. So, what is the definition of median?

Median means the number of elements to the left and the number of elements to the right must be equal. So, here also we are trying to calculate that kind of a thing. So, with respects to the x coordinates I can calculate the median with respect the y coordinates I

can calculate the median. Like you see if you look at the x coordinates 2 2, 4 4. So, I just writing down the x coordinates 2 2, 4 4, 6 7, 9 10. So, I am writing them sorted by their x coordinates. So, the first 4 I can keep in the first partition second in the other partition. So, with respect to the x coordinates if I compute the median. So, the middle point will be this. Similarly, by their y coordinates if you just sort by their y coordinates it will be first will be this 2, then 2 again there is another 2 and 3 4 and 5 5, 7 7. So, here again if you cut at the middle point 2 2, 3 4 will be in 1. So, vertically if you want to find out the median the median will be this.

So, if you calculate the median either with respect to be x or y. So, what you are doing you are dividing a set of points into 2 equal parts. So, the essential idea behind calculating median is this right.

(Refer Slide Time: 07:23)

**Method of Means & Medians (MMM)**

- Follows a strategy very similar to the H-tree algorithm.
  - Can deal with arbitrary locations of clock sinks.
- Basic idea:
  - Recursively partition the set of terminals into two subsets of equal size (*median*).
  - Connects the center of mass of the whole set to the centers of masses of the two partitioned subsets (*mean*).

So, this is the first step partition the set of terminals into 2 subsets of equal size. Then connect the center of mass of the whole set to the center of mass of the partitioned subsets. Let us also illustrate this. So, the first part is median next part is mean. Suppose I have partitioned it is in the red partition, let us consider this partition. So, I have 4 points on the left 4 points on the right. Take the average you take the 4 x coordinates 2 2, 4 4. 2 2, 4 4 what is the average is 3 - 6 7, 9 10. So, what is the average? Average is 8. So, your 3, 8 will be average. So, so 3 may be 10. And if you look at the y coordinates similarly look at the y coordinates 2 3 5 7. So, 5 10 17 by 4 approximately 4, this is x this is y

average and for the right side 4 2 6 11 18. So, it will approximately let us say 5 between 4. So, 3 4 and 8 5 are the mean. So, 3 1 2 3 4 this is the 3 4 8 5 3 4 5 6 7 8 1 2 3 4 5 this. So, the idea is as follows.

So, after calculating the median you divide it up into 2 subsets. Then in each of the subsets you find the means, with respect to the points you take the average of the x and ys in approximately to find out the center of mass or center of gravity whatever you call. Then you connect these 2 center of masses by a straight line. And you recursively go on doing this. So, again the left subset you divide into 2 parts, right subsets you divide in 2 parts this way you proceed, right. So, because you are successively computing means and medians, medians than means, again median again mean. That is why this method is called method of means and medians, right.

(Refer Slide Time: 10:09)

**How is the partitioning done?**

- Let  $L_x$  denote the list of clock points sorted according to their x-coordinate.
- Let  $P_x$  be the median in  $L_x$ .
  - Assign points in list to the left of  $P_x$  to  $P_L$ .
  - Assign the remaining points to  $P_R$ .
- Next, we go for a horizontal partition, where we partition a set of points into two sets  $P_{R1}$  and  $P_{R2}$ .
- This process is repeated iteratively.

IIT KHARAGPUR
 NPTEL ONLINE CERTIFICATION COURSES

So, let us see. So, whatever I have just worked out with that example is explained here. So, we sort the points by their x coordinates. Suppose we want to carry out the median or calculate the median with respect to x. If  $P_x$  is the average which the median. So, all points to the left of  $P_x$  is assign to a left subset  $P_L$ . All the points to right is assigned to  $P_R$ . Then for each of these we calculate as I had shown the mean and connect them. And you go recursively. First you do a vertical partition then a horizontal then vertical and this iteratively you go through it.

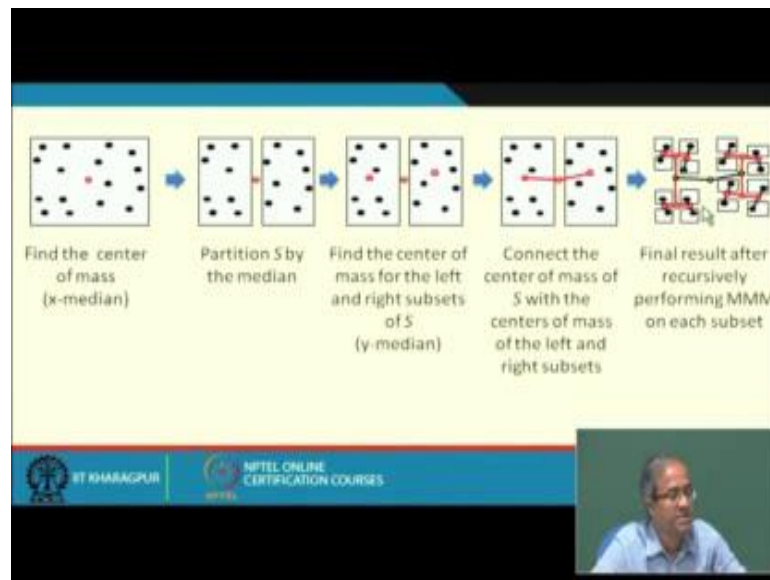
(Refer Slide Time: 11:03)

- The basic algorithm ignores the blockages and produces a non-rectilinear tree. Some wires may also intersect.
  - In the second phase, each wire can be converted so that it consists only of rectilinear segments and avoids blockages.

So, the process I will just explain with the help of an example. Now just like your H-tree and X-tree algorithms. So, we ignore the blockages we do not consider that there are obstacles on the layer on which were laying out the clock tree and because we are using the exact points where you need to feed the clocks as the vertices. So, my clock tree will look different from H-tree it will not be rectilinear. Some edges will be slanted points can be anywhere right, need not be regularly spaced.

So, each of this you can say non rectilinear wires slanted wires you can convert it by making them horizontal and vertical if you want this is an optional step, not required if you are laying the entire thing in a single layer.

(Refer Slide Time: 12:15)

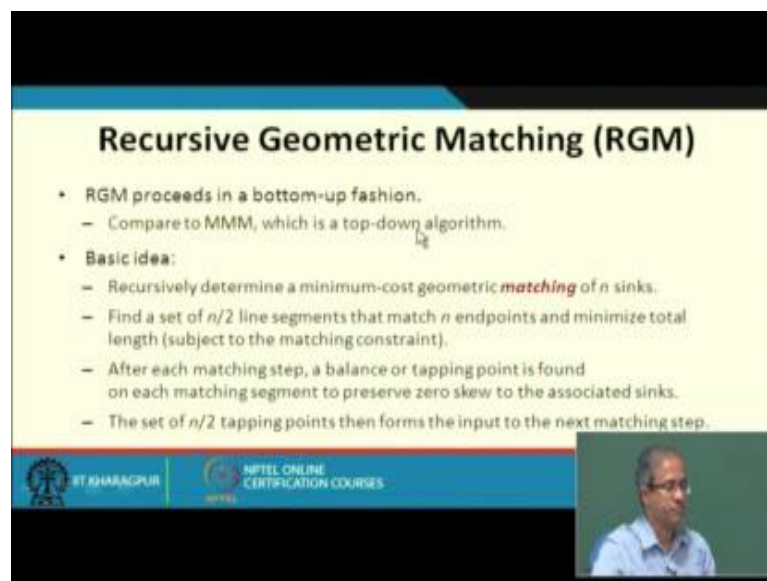


Let us see with the help of an example. So, I have worked out an example numerically, but here I am showing it diagrammatically what happens. There are 16 points just like the way I talked about. So, along the x direction you calculate the x-mean. So, the red point is your x-median. Median you calculate the x-median. Red point is the x median. So, what you do you vertically do a partition, partition the x with respect to the x-median you have to try it out. So, there will be 8 points from the left 8 points from the right you repeat the process for the left and right subsets, but now not with x, but with y. So, like in this example if you look at. So, now, on the left subset you have 4 points you repeat this same thing by y, so 7 5 3 2. So, now, if you do a partition, 2 3 will be in 1 5 7 will be in the other. So, there will be a partition like this.

Similarly, on this side 2 4 5 7, so the partition will be like this right. So, same thing will happen here in this example there I am working out. So, you compute the center of mass for left and right, but with respect y median now with respect to y. So, may be this is your y median, and this is your y median may not be at the same vertical level. You join this median with respect to the median of these 2. Then you do a partitioning of this and this, and continue recursively and I am showing you the final solution. So, if you repeat it again do with this, then do it like this, then do it like this, 4 more steps you get this right. Each subset will consist of a single point and they will be all connected by some arcs.

So, now you can see that the arcs or the edges that are connecting the points, they are not horizontal and vertical only. They can be of arbitrary and these shapes and sizes, but when you finally, convert them may be you can convert them into segment horizontal vertical later on, but initially you have got this. So, what you have got. So, you have got given an initial set of points, you have got a tree embedding which will ensure the, ensure that the lengths of all the arms are equal in sizes. Because the way you are calculating the means and medians this will be ensured that is your length of the segment of each of the tree, starting from the lowest level if you think you create a tree such that the 2 sizes are equal, and use 2 trees to combine to a higher level tree. Again 2 sides are equal that is the same way you are repeating. And this is a basically top down approach. Starting from the overall problem you are trying to divide them, partition them, go on partitioning till each partition consist of a single point, and when you get that you have already obtained the tree right.

(Refer Slide Time: 16:05)



**Recursive Geometric Matching (RGM)**

- RGM proceeds in a bottom-up fashion.
  - Compare to MMM, which is a top-down algorithm.
- Basic idea:
  - Recursively determine a minimum-cost geometric *matching* of  $n$  sinks.
  - Find a set of  $n/2$  line segments that match  $n$  endpoints and minimize total length (subject to the matching constraint).
  - After each matching step, a balance or tapping point is found on each matching segment to preserve zero skew to the associated sinks.
  - The set of  $n/2$  tapping points then forms the input to the next matching step.

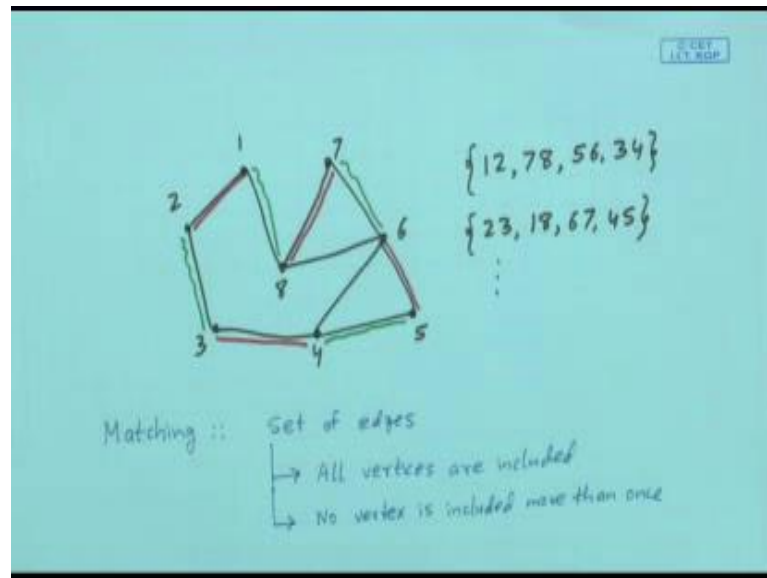
ST KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

NPTEL

NPTEL

So, we have an alternative method which is very similar, but it follows in a, follows a bottom up fashion. Unlike this MMM which is top down. Here we use the concept of a matching. So, matching is a concept that comes from graphs theory.

(Refer Slide Time: 16:36)



So, let us first explain what is a matching, in case you have you are not familiar with the definition. Suppose I have a graph there are 8 vertices, say they are connected by some edges. So, there are 8 vertices. So, we are trying to define something called a matching. Matching, what is a matching? Matching is nothing, but a set of edges, then there are some requirements. Your first requirement is that you have to select a set of edges such that all vertices are included and, Secondly, no vertex is included more than once. So, for this example there can be a so, much different kind of matchings. Let us say one possible matching may be, you take this edge you take this edge. You take this edge and you take this edge. So, you can try some other matching also like, another matching may be, you take this edge, you take this edge, you take this edge. So, there can be multiple matchings which are possible.

So, in this case I have showed 2 possible matchings. So, one matching was consisting of the edges 1 2, 7 8, 5 6 and 3 4. Marked by red and the other one was marked by green. Let us say another matching 2 3, 1 8, 6 7, 4 5. So, there can be multiple such matchings. Now here what you are doing at the lowest level we have a set of points, these are the clock terminals. We try to find out our matching for which the sum of the total, some of the edges is minimum. You see in the method of means and medians we are starting from the top all points partitioning them progressively finally, we are arriving at the solution, but now we are moving in the reverse direction.



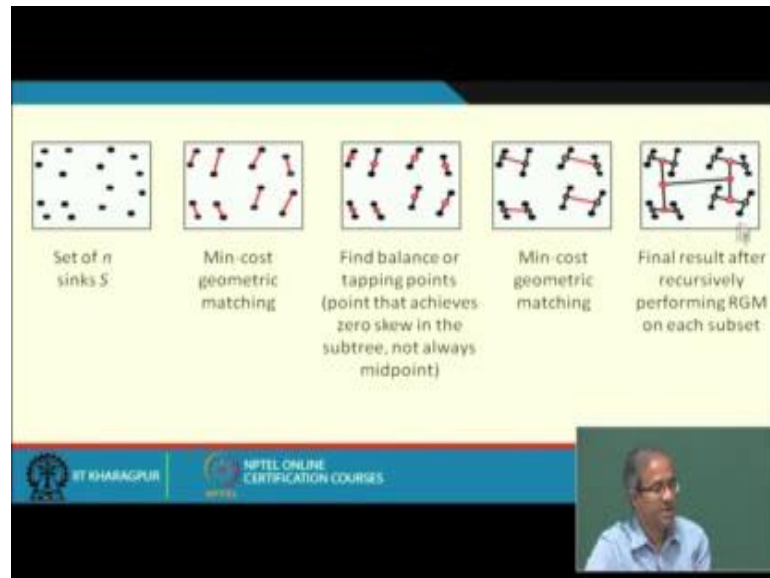
We are starting with all points we are finding a matching means we are connecting 2 points  $2, 2$ . Such that the total length of the wires is minimum, so among all the possible matchings we are finding one matching which is of minimum size. Then for each of these matchings we have found out like for example, if I take the red one, suppose you have take the red one, but each of these matchings well find the middle points.

So, now forget the original 8 vertices, now these are my new vertices. Let us call it  $v_1, 2$ , let us call it  $7, 8$ , these are my 4 vertices  $5, 6$ , and  $3, 4$ . You repeat the process on these 4. So, on these 4 you again repeat the process merge these 2 vertices and you so, now, again you compute some kind of matching. So, one matching may be between this and this. And the other matching can be between this and this let say. So, you connect the vertices correspondingly. So, the process I will be showing with an example.

So, this will be clear, but now let us see you came back to the slide and see what is written here. The same thing as I have illustrated. So, we recursively determine a minimum cost matching of  $N$  sinks. And you are assuming  $N$  is a even, which means you are finding a set of  $N$  by 2 line segments that match  $N$  endpoint. You see the example I took we had a graph, we are taking the subset of the edges, but now I have a just a set of points. I am assuming that there is an  $h$  between every pair of vertices. So, every connection is possible. Out of that I am finding the smallest method, right.

So, we are finding out  $N$  by 2 line segments, that covers all the  $N$  endpoints. And after each matching as I have shown, we are finding out a middle point, which is some time called as a tapping point, so as to preserve 0 skew. So, these  $N$  by 2 tapping points will form the input to the next step. Just I had said.

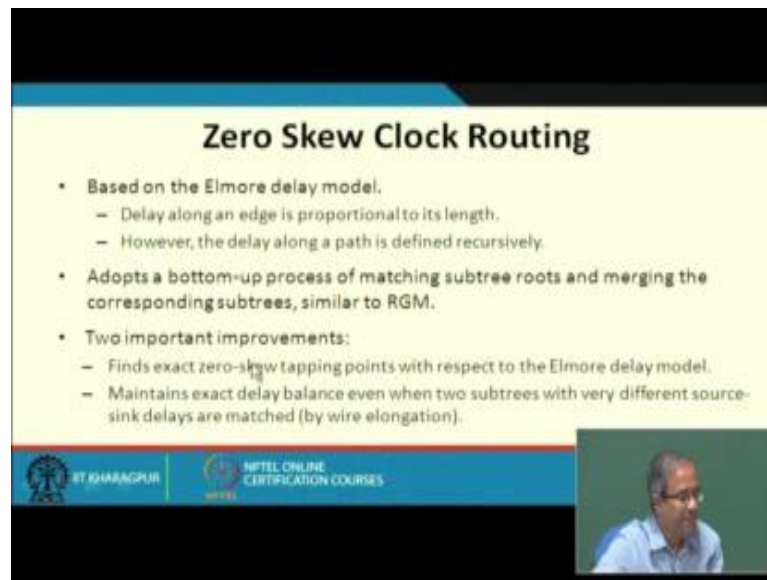
(Refer Slide Time: 22:40)



So, let us just work it out again. Here we consider same example. There is a set of 16 points. So, instead of starting with a partitioning like the MMM algorithm, we start by finding out the smallest matching. Suppose the smallest matching we found is this. This is the matching. So, I have connecting them by the corresponding edges. Now after this matching for each of the lines that you have drawn shown as red, we find the tapping points. So, the tapping points are shown as these red circles. You go on repeating. For these 4 and 4, 8 tapping points, again find out a matching. Suppose my matching is now this, smallest matching. So, again find out the middle points or the tapping points for these 4 edges. So, again do a matching go on repeating this.

So, you will get finally, a solution which looks like this. So, you see the final thing that you get looks like a tree looks like an edge, but it is not a very regular edge a skewed edge. The edges are not horizontal and vertical, but usually for this problem this bottom up approach gives better result than the top down approach. Of course, there can be some example you could be work out where this MMM method will give result which is better than this top down one bottom of one. So, both works and these are better than H-tree approach in the general case. Where you are directly trying to connect the sink points, but again I am repeating for 2 step routing where you have a grid like structure at the bottom and you have a regular clock tree on the top then you can go for h, h type of routing because there you have the terminal points regularly spaced fine.

(Refer Slide Time: 25:06)



**Zero Skew Clock Routing**

- Based on the Elmore delay model.
  - Delay along an edge is proportional to its length.
  - However, the delay along a path is defined recursively.
- Adopts a bottom-up process of matching subtree roots and merging the corresponding subtrees, similar to RGM.
- Two important improvements:
  - Finds exact zero-skew tapping points with respect to the Elmore delay model.
  - Maintains exact delay balance even when two subtrees with very different source-sink delays are matched (by wire elongation).

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

*(Video inset shows a man speaking)*

Let us talk about 0 skew clock routing, but we are not going into detail here, we shall be coming back to that later on when we look at estimation of delays and noise and circuits later on. So, the basic principle is that here we are not considering the geometric length of the wires. So, we are actually estimating the rc delays not just the length. Delay along an edge is proportional to its length, but along a path it can be defined recursively. So, here I shall be showing you some simple calculation how it is done. So, the basic principle is like the bottom up approach.

The RGM approach that we just now talked about, but the only difference is that is in the determination of the tapping points. So, when you determine the tapping points, we do not find the middle point, but we use some kind of a realistic delay model this Elmore delay model is one of the most widely used delay models, which gives quite realistic delays, but also at the same time does not consume too much computation. So, we use this Elmore delay model to find the tapping points. And sometimes you may have to carry out wire elongation using sneaking to match the delays across subtrees.

Because the Elmore delay model you can find that one of the parts is much faster than the other, to make it slower you may have to make it longer to make them equal.

(Refer Slide Time: 27:02)

- The point set is recursively partitioned into two subsets, and trees are constructed in a bottom-up manner.
  - Assume, inductively, that every sub-tree has achieved zero skew.
  - Given two zero-skew sub-trees, merge them by an edge to achieve zero skew on the new tree.
    - Necessary to decide the position of the connecting points (taps).
    - Uses Elmore delay model for the purpose.

ST KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, the principle is the same as in the bottom up thing there is no difference in that, but the only difference is in selection of the tapping point rest is the same.

(Refer Slide Time: 27:18)

Tapping point  $t_0$

Subtree  $T_{11}$      Subtree  $T_{12}$

Tapping point  $t_0$ , where Elmore delay to sinks is equalized

The diagram illustrates the Elmore delay model for merging two subtrees,  $T_{11}$  and  $T_{12}$ , at a tapping point  $t_0$ . The left side shows a tree structure with a root node  $t_0$  and two subtrees  $T_{11}$  and  $T_{12}$ . The distance from  $t_0$  to the sink  $s_1$  is  $z$ , and the distance to the sink  $s_2$  is  $1-z$ . The right side shows the equivalent circuit model. The top branch has a resistor  $R(w_1)$  in series with a capacitor  $C(w_1)$  in parallel with a sink  $s_1$ . The bottom branch has a resistor  $R(w_2)$  in series with a capacitor  $C(w_2)$  in parallel with a sink  $s_2$ . The total resistance of each branch is  $\frac{R(w_i)}{2}$  and the total capacitance is  $\frac{C(w_i)}{2}$ . The delay to sink  $s_1$  is  $t_{11}$  and the delay to sink  $s_2$  is  $t_{12}$ .

ST KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, the either is as follows; that means, you have 2 subtrees, there is a connection between then you are trying to find out the tapping point. So, the length of one of them is  $z$  other is  $1$  minus  $z$ . So, each of these lines you are modeling in terms resistances and capacitances. And this Elmore delay model is one method using which you can make a

realistic estimate and to determine for what value of z these 2 delays are approximately equal, you select the tapping point accordingly.

(Refer Slide Time: 27:56)

**Elmore Delay**

- ON transistors look like resistors.
- Pullup or pulldown network modeled as RC ladder.
- Elmore delay of RC ladder is shown.

$$t_{pd} \approx \sum_{i=1}^N R_{i-to-source} C_i$$

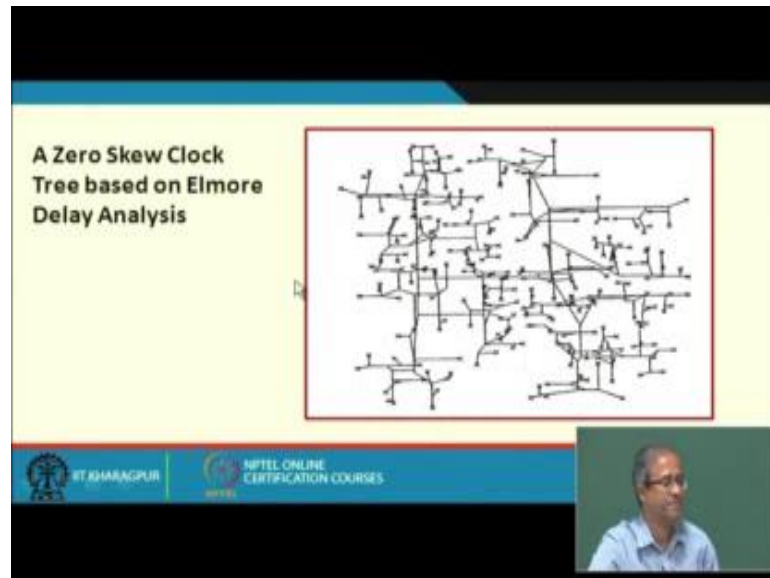
$$= R_1 C_1 + (R_1 + R_2) C_2 + \dots + (R_1 + R_2 + \dots + R_N) C_N$$

The diagram shows an RC ladder network with a voltage source on the left, followed by a series of resistors  $R_1, R_2, R_3, \dots, R_N$  and shunt capacitors  $C_1, C_2, C_3, \dots, C_N$  connected to ground. The Elmore delay formula is presented above the circuit diagram.

So, let us see this Elmore delay model I am illustrating with the help of the simple example. Suppose I have a transmission line where the resistances and capacitances are shown as distributed. R C 1 R 2 C 2 R 2 C 3 like this. So, the according to the Elmore model the delay can be estimated as follows C N, from the source C is only the delays of R 1, but C 2 C is the delay of R 1 plus R 2. C 3 will delay due to R 1 plus R 2 plus R 3. So, like that it is like a recursive formulation R 1 C 1 R 1 plus R 2 C 2 and at the end R 1 R 2 R N all added together multiplied by C N. So, for an RC ladder it is this. So, for a general simo circuit. So, any on transistoric and model as a resistance, and some pull up and pull down network any such things you can model as RC RC RC kind of a ladder and there you can make a quite accurate estimate of the dealing.

So, even for transmission line depending on which layer the line is running whether some metal layer polysilicon layer diffusion layer you can have the corresponding resistance and capacitance estimates, so we shall be coming to this later on, but for the being you have to just assume that there are ways of estimating R N C and Elmore delay model is a it is a quite accurate method to make a reasonably good estimate of that, and using that you find the tap points.

(Refer Slide Time: 29:38)

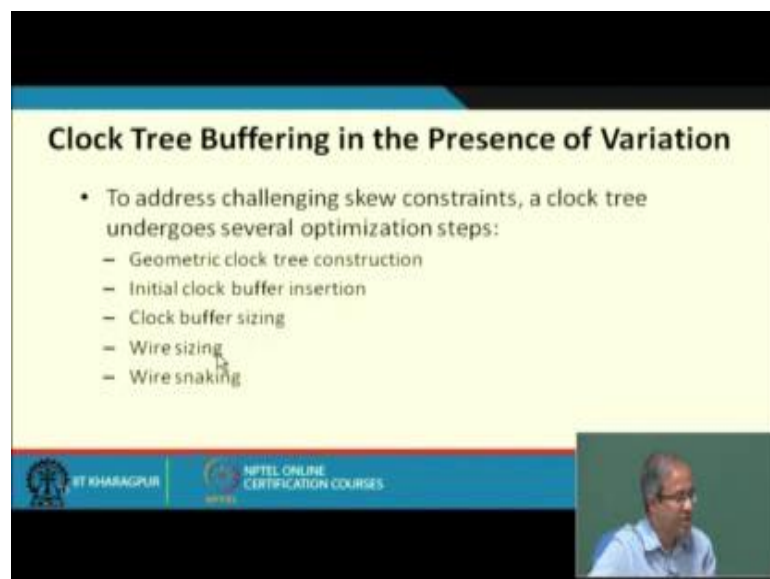


A Zero Skew Clock Tree based on Elmore Delay Analysis

The slide features a complex, irregular clock tree diagram with numerous nodes and branches, illustrating a zero skew clock network. The diagram is enclosed in a red border. Below the diagram, there are logos for IIT Kharagpur and NPTEL Online Certification Courses, and a small video inset of a speaker.

So, this is just a diagram which is shown that using Elmore delay model. So, how does a 0 skew clock network look like for a typical example. You see that the edges are quite haphazard and so on.

(Refer Slide Time: 29:55)



Clock Tree Buffering in the Presence of Variation

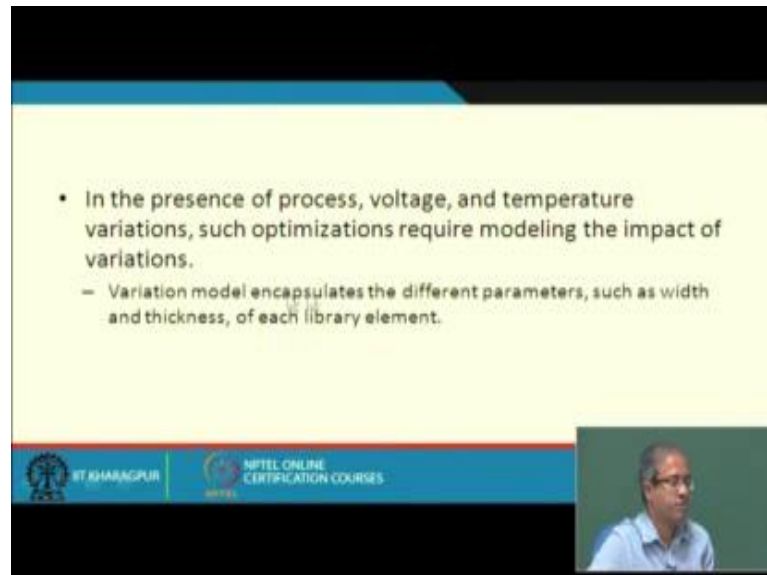
- To address challenging skew constraints, a clock tree undergoes several optimization steps:
  - Geometric clock tree construction
  - Initial clock buffer insertion
  - Clock buffer sizing
  - Wire sizing
  - Wire snaking

The slide contains a bulleted list of optimization steps for clock tree buffering. Below the list, there are logos for IIT Kharagpur and NPTEL Online Certification Courses, and a small video inset of a speaker.

So, when you talk about clock tree buffering in the presence of variation. So, you may need to carry out several optimization steps, like some of them I already talked about, geometric clock tree, initial buffering insertion, sizing of the buffer. Some of the buffer

you may have to make smaller larger, the width of the wires snaking you may have to increase the length of the wire by taking a detour along path.

(Refer Slide Time: 30:28)



• In the presence of process, voltage, and temperature variations, such optimizations require modeling the impact of variations.

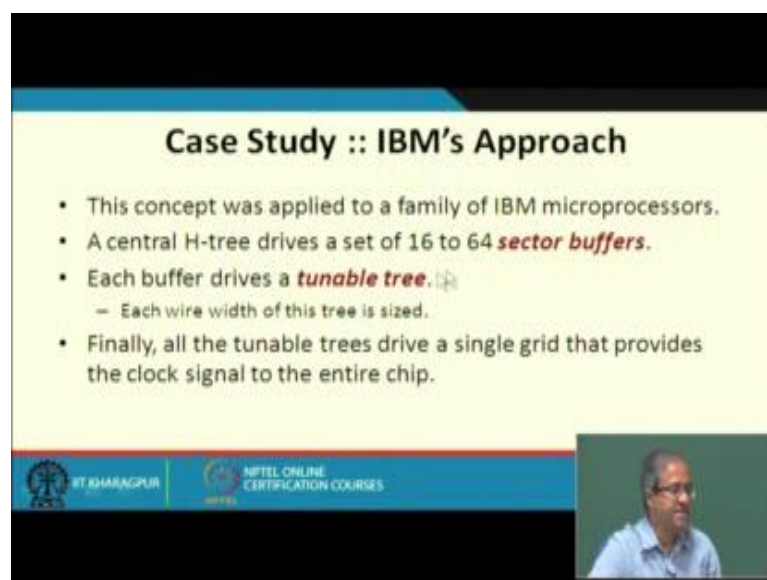
- Variation model encapsulates the different parameters, such as width and thickness, of each library element.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The slide features a yellow background with a blue header and footer. A small inset video of a speaker is visible in the bottom right corner.

So, there will be process variations. So, if you can model the impact of the variation. So, you can come up with a clock routing solution, which will give you a bound of the on the total skew even the presence of such variations.

(Refer Slide Time: 30:57)



### Case Study :: IBM's Approach

- This concept was applied to a family of IBM microprocessors.
- A central H-tree drives a set of 16 to 64 *sector buffers*.
- Each buffer drives a *tunable tree*.
  - Each wire width of this tree is sized.
- Finally, all the tunable trees drive a single grid that provides the clock signal to the entire chip.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

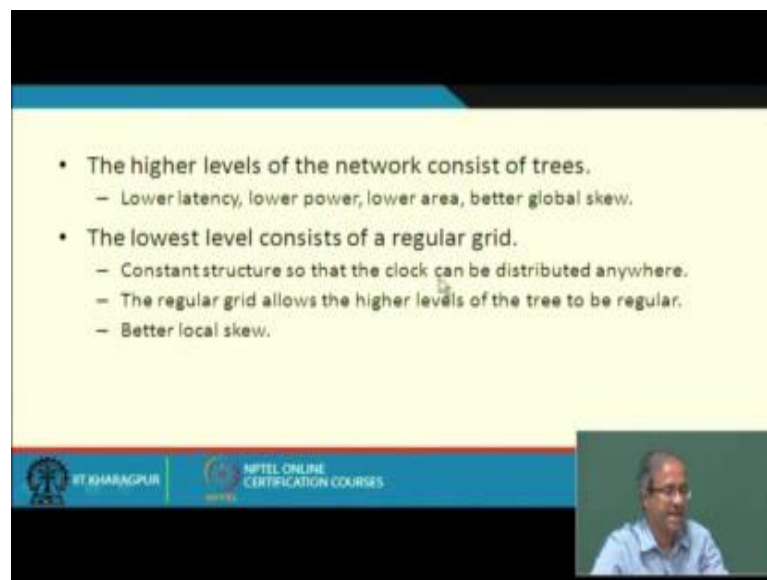
The slide features a yellow background with a blue header and footer. A small inset video of a speaker is visible in the bottom right corner.

So, we quickly look at 2 case studies where the clock trees were designed like IBM for some of the microprocessors what they did. They divided the entire chip area into 16 to

64 regions these were called sectors. So, they constructed a top level H-tree the drive it is a 2 level or 3 level H-tree that will drive a set of 16 to 64 sector buffers. So, the leaf of the sector buffer will lead you to one of the sectors. And each of the sector you have lot of clock things to connect right. They are handled independently.

So, these sector buffers are handling each of the sectors it is called a tunable tree. For each of this sectors there is a tunable tree where the wire lengths and sizing of the trees are tuned. So, that delays are equalized and these tunable trees drive a grid. That grid will provide the clock tree signal to the entire chip, just like I have said it is a 2 level approach.

(Refer Slide Time: 32:10)

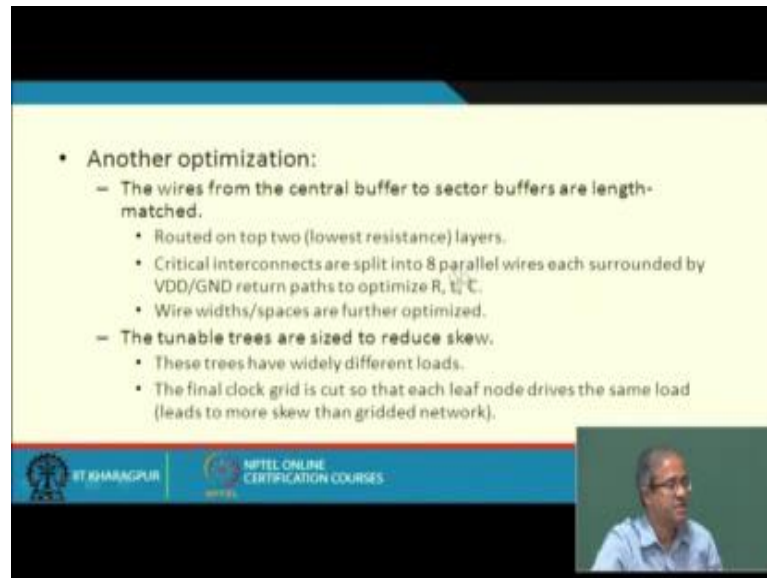


- The higher levels of the network consist of trees.
  - Lower latency, lower power, lower area, better global skew.
- The lowest level consists of a regular grid.
  - Constant structure so that the clock can be distributed anywhere.
  - The regular grid allows the higher levels of the tree to be regular.
  - Better local skew.

So, high level network consists of an H-tree which provides lower latency, lower power area is optimized global skew is also very good; the lower level consists of a grid as I had said earlier also. The advantage of a regular grid is that you can take the clock signal anywhere you want, because the grid is available everywhere. And this also ensures much better local skew because grid is a regular structure as such. So, when you have multiple points driving the grid structure across the grid the skew differences it is also very less this skew is good in that sense.



(Refer Slide Time: 32:52)



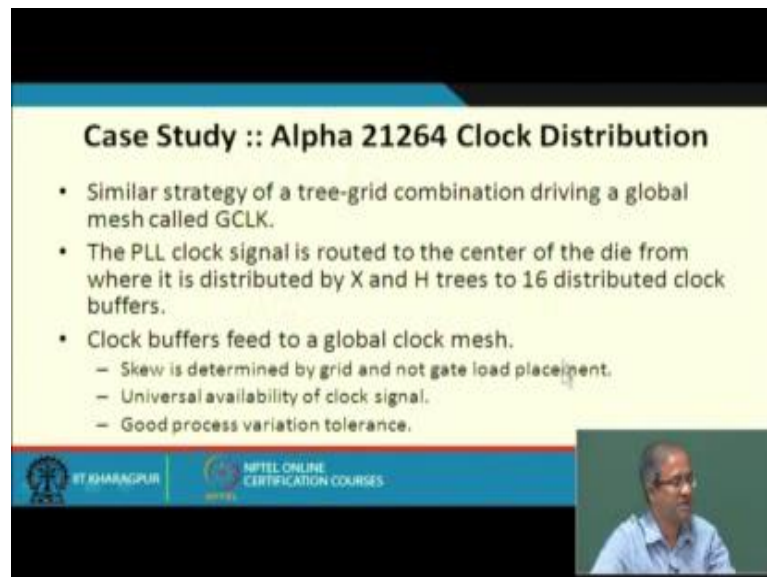
• **Another optimization:**

- The wires from the central buffer to sector buffers are length-matched.
  - Routed on top two (lowest resistance) layers.
  - Critical interconnects are split into 8 parallel wires each surrounded by VDD/GND return paths to optimize  $R, \tau, C$ .
  - Wire widths/spaces are further optimized.
- The tunable trees are sized to reduce skew.
  - These trees have widely different loads.
  - The final clock grid is cut so that each leaf node drives the same load (leads to more skew than gridded network).

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, there is some other optimizations which are also carried out, like the layers we use the metal layers and I am not going with detail of these. I got some of the critical wires instead of a single line that 8 parallel wires which were laid out such that the delay is reduced and so on.

(Refer Slide Time: 33:17)



**Case Study :: Alpha 21264 Clock Distribution**

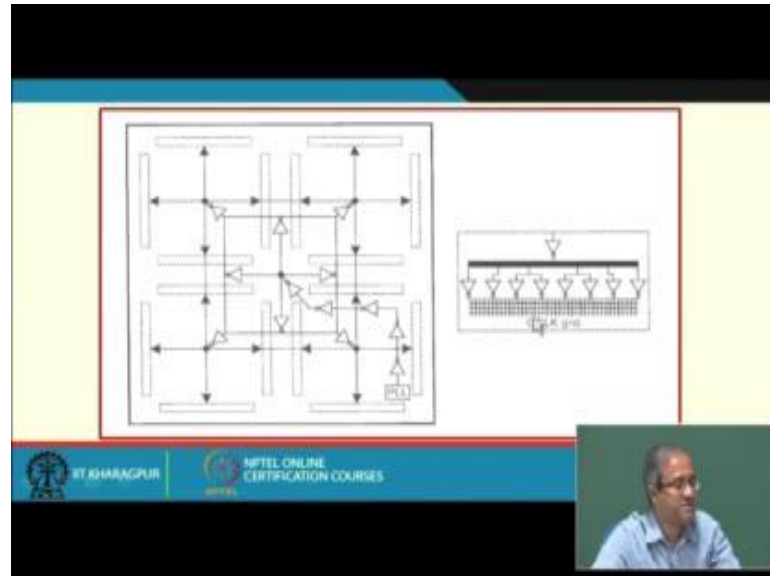
- Similar strategy of a tree-grid combination driving a global mesh called GCLK.
- The PLL clock signal is routed to the center of the die from where it is distributed by X and H trees to 16 distributed clock buffers.
- Clock buffers feed to a global clock mesh.
  - Skew is determined by grid and not gate load placement.
  - Universal availability of clock signal.
  - Good process variation tolerance.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Just a very quick look at another case study - digital corporation's alpha 21264 processor. So, there we used a similar kind of a 2 level configuration a tree followed by a grid. So, the global mesh we call it GCLK. So, here again they use an H-tree or an X-tree

to drive 16 clock buffers. And this clock buffers drive a global clock mesh. So, the basic approach between IBM and alpha approach are similar.

(Refer Slide Time: 33:56)



That dramatically it is shown like this. So, there is a tree structure. Drive the 16 buffer the 16 buffers are shown by this narrow rectangle. And each of this buffers they drive a grid. So, this grid will be local to that region right. So, once you have this grid from the grid you can take the clock points or tapping points to anywhere in the chip you want. So, this we come to the end of a discussion on clock tree routing. In our next lecture we shall be talking about power and ground routing.

Thank you.