VLSI Physical Design with Timing Analysis

Dr. Bishnu Prasad Das

Department of Electronics and Communication Engineering

Indian Institute of Technology Roorkee

Week 07

Lecture 37

Clock Routing Algorithms – III

Welcome to the course on VLSI Physical Design with Timing Analysis. In this lecture, we will discuss some more clock routing algorithms. The content of this lecture includes exact zero skew algorithm. Then we will discuss about one another popular algorithm called Deferred Merge Embedding algorithm DME algorithm. So, it based on we need to build tree of segment algorithm and find the exact location of the clock tree. So, here what is the main concept is that we will create a clock tree which is having zero skew. So, what is the approach here? So, what we are doing here is that we are following a bottom-up process of matching the sub-tree nodes and the corresponding sub-trees similar to RGM. RGM stands for recursive geometric matching algorithm. So, what we are doing? We are going by a bottom-up approach where we are following similar to RGM and we are maintaining a zero clock skew in the clock tree. So, what is the improvement here is that we find say exact zero skew tapping points.

So, here we are finding a exact zero skew tapping point by considering the interconnect using the Elmore delay model. So, the main point here is that we are exploiting Elmore delay model to estimate the wire length and comparing to match the delay through the clock tree. So, we are not using the linear delay model. So, here what we are doing is that it maintains an exact delay balance. It maintains an exact delay balance even when two sub-tree with very different shore sensing delays are matched. So, what is happening here is that even if your sub-trees are of different delay, it will create an exact balance. So, how this is accomplished is that by elongating the wear basically we can have different interconnect length in two different branches as to make the source and sink delay same. So, if you can see here, we have two sub-tree, one is Ts1 and Ts2. So, their delays are not same till this S1 and delay till S2 is not the same. Then what we are doing is that we are finding a tapping point here such that my delay from this point through this sub-tree 1 Ts1 should be equals to the delay through sub-tree Ts2.

So, what we are doing here? So, this is the problem statement. We created a model for that one. So, delay till this point is denoted by t of Ts1. Similarly, delay till this point S2 is denoted by t of Ts2. Now our target is to find a point here such that both the branch delay is same. So, what we are doing here is that we are creating the Elmore model of the interconnect. And we are estimating the interconnect delay through the Elmore approach. So, here you have effective capacitance here is basically is Cs1 it is the sub-tree Ts1 is driving Cs1 and sub-tree Ts2 is driving C of s2. So, both capacitance are different. Now we have a two different piece of metal.

One is with length Z and another is with having the length 1-Z. So, here if you can see this is the length Z. So, now we are creating the delay from this point let us say this point is A till let us say this point is T. So, what is the delay from A to T and what is the delay from B to T? Both the things should be matched. If the both the things are matched then the skew at this point T is 0. So, we are doing the same method to achieve a 0 skew. So, we are basically choosing the interconnect length such that both the delays should be same from the tapping point. So, what is the concept here is that so I am finding a delay from tapping point TP to Ts1. Then I am finding a delay from tapping point TP to Ts2 and we are making sure that both are same. So, now we are doing the delay estimation. So, this first one is basically for Ts1 sub-tree. So, this is the delay of the Ts1 and this is the delay of the interconnect from Ts1 to TP. So, this portion is basically Ts1 to TP delay and delay till Ts1. Now these delays are additive so we can add them. So, now we can have the second delay.

This is basically delay from Ts2 to TP and this is the delay till so this is basically delay of Ts2. So, now we are matching the delay. Now after we match it I want the resistance and capacitance. Basically, what we are doing here is that we are calculating the delay from this tapping point. Here the clock is entering our design and it is going to all the leaf nodes actually leaf nodes or the sink nodes you can say. So, what we are doing here is that till this point what is the delay Ts1 and this is Ts2 till that delay is already calculated. Now we are finding so how much delay I should add in the both the branches such that delay from this point till this should be equal to delay from this point till Ts2. So, what we are doing here is that we are finding the interconnect length in the both the branches. So, let us the interconnect length is Z in this segment and the other segment is 1-Z. Now what we did is that we have created an equivalent PI network equivalent PI network for the interconnect.

So, if you can see here the total capacitance at this node will be the sum of these two that will be multiplied with R of w1 to find the delay till this point. So, now if you can go here so R of w1 into C of w1 by 2 plus Cs1 is the effective capacitance at that node multiplied by its resistance and similarly the delay till TS1. Now what we are doing is that now we apply the same concept in the other branch. So, then whenever we are doing the equivalent delay to both the branches then we are interested in finding the resistance and capacitance of each of the line interconnect segment. So, R of w1 is basically we need to find and C of w1 we need to find. So, how it will depend? It depends upon alpha. Alpha is basically resistance per unit length times Z into the length. So, it will be basically resistance will be this much.

$$R(w_1) = \alpha.Z.L(s_1, s_2)$$

Similarly, C of w1 is

$$C(w_1) = \beta.Z.L(s_1, s_2)$$

So, this is for one branch.

Similarly, we can apply the same thing in the second calculation in this one second calculation. So, R of w2

$$R(w_1) = \alpha.(1-Z).L(s_1, s_2)$$

Similarly in C of w2

$$C(w_1) = \beta.(1-Z).L(s_1, s_2)$$

So, if you can see this, this is the total resistance calculation and one concept here is that your resistance is basically rho l by A. So, area of the metal is constant it only depends upon the length. What is the length of the metal? Your resistance will scale as per the length of the metal. Assuming that we are drawing the metal why it is not depending upon the area? Because our one dimension is constant because the width of the metal is constant throughout the basically clock trace synthesis and it is the minimum length of the metal allowed in that technology. So, that is why it does not depend upon the area. So, this is one concept related to resistance. Similarly, the same concept can be applicable for capacitance your C equal to epsilon A by D.

So, here epsilon will depend upon the dielectric and similarly basically this area will have two components w into l by d, but this we are worried about the length or the length of the metal because width of the metal is drawn is the minimum width of the metal like similar analysis we did it for the resistance. So, that is why your beta is capacitance for unit length. Now what we are doing is that combining the above two equations it is a position of zero skew tapping point. So, we can solve this equation to find a length which is creating a zero skew tapping point. So, here there is two cases possible if your z is lies between 0 to 1 then the tapping point is located along the segment connecting the roots of the two sub trees if it is means z is lies between 0 to 1.

What is the other case if it is not the case then what we have to do is that it is not possible to route the both the metals such that we can get zero skew. So, what we have to do we have to basically use a longer metal and do the analysis again. So, where must be elongated that means that you are using a longer piece of metal in the both the branch to meet the zero skew condition. So, that is the idea. So, it is very simple and all are basically linear equation kind of thing.

So, anybody can understand this. So, now we will discuss a new algorithm called deferred and merge embedding algorithm for clock tree synthesis. It has a different concept behind the clock tree synthesis. What is happening here is that we are basically delaying the choice of tapping point for the sub trees of the clock tree. We are basically delaying that one. And we need a tree topology as also input to the algorithm. So, what is the fundamental limitation of the existing or the previous algorithm is that it is basically determines the location of the internal node of the clock tree very early. So, all the tapping points are determined early before intelligent decisions are even possible. So, the choice of selection of the internal nodes of the clock tree is done early in the previous algorithms. However, in this algorithm, they have a systematic approach to choose this tapping point or the internal nodes of the clock tree. So, we will discuss how this is happening. It is very interesting. Basically, it has some of the geometrical notion here. We will discuss that in detail. What is happening here is that we have a term called Manhattan arc. We have to first understand what is Manhattan arc. In the Manhattan geometry, so two sinks in the general position will have infinite number of midpoints creating a tilted line segment or Manhattan arc. We will understand this with an diagram. Let us say we have two sink nodes S1 and S2. These are the two sink nodes available to us.

Then what is this Manhattan arc for these sink nodes? So, this red line is the Manhattan arc for the S1 and S2. Why that is the basically the Manhattan arc? It is saying that it is the locus of all the Manhattan midpoints is creating the Manhattan arc. For example, from S1 to S2, I can connect in multiple different ways. All the lines are passing through these red lines and this red line is having the midpoint of that routing. So, this red line is called the Manhattan arc. What are the possible routing possible here? Let us say I can do the routing in this manner. This is one possible routing, then this is the midpoint. Another possible routing is that from here it is going like this. So, this is the midpoint. How this is the midpoint? If I can calculate the segments here, so here it is taking one segment, two segments, three segments to this point.

Similarly, if I calculate the segments here, this is the first segment, this is the second segment, this is the third segment, this line is the third segment. So, that is why this is the midpoint. Similarly for the upper case, this is the first point, this is the second segment or second grid you can say, this is the third grid. Similarly, if you can see this is the first, this is the second and this is the third. So, this is the basically if you can say these red lines is the midpoint of all possible routing connecting S1 and S2. So, that one is your Manhattan arc. So, this part is very important to remember. So, what is the definition of this Manhattan arc? And there is one more point is there. So, if your sinks are aligned, aligned means sinks are in the same axis. In the first case, your sinks are horizontally located. So, then the Manhattan arc basically if you can see is a point, only one midpoint is existing.

So, the length of the Manhattan arc is 0. So, in this case the Manhattan arc is 0, arc distance or length equals to 0 and same case here also. This is the only midpoint possible, this is the only midpoint possible. So, whenever the sinks are aligned, the Manhattan arc has 0 length. Now what we are doing is that we are basically this algorithm deferred and merge embedding algorithm has two phases. The first phase is a bottom-up approach and the second phase is the top-down approach.

We will discuss one phase at a time. So, what is the concept here is that we are determining all possible location of the internal load of G consistent with a minimum cost 0 tree. We are creating all the internal nodes. If you can see here, we have this Manhattan arc is the midpoint where we can get many possible solutions. So, this Manhattan arc is basically representing the all possible solution to the node G and for creating the minimum cost or 0 skew tree actually. So, what is the output of this algorithm? It is basically it is not giving the final clock tree synthesis output finally. It has a tree of line segment with each line segment being the locus of possible placement of an internal nodes. So, we have multiple possible solution to create a 0 skew tree the first phase is giving. So, we have multiple possible solution that is given by the first phase and we can choose any one of them based on our convenience. So, then in the second phase what is happening is that it is a top-down approach chooses the exact location of internal nodes in the tree. So, it exactly locates which is actually on the grid and it is also satisfy the 0 skew constraint.

So, then the final output is fully embedded minimum cost ZST for the topology G. So, this is the beauty of this algorithm. It creates a minimum cost 0 skew tree at the final output. So, there is two concept here. One we discussed that Manhattan arc. Then there is another concept is that tilted rectangular regions. So, what is it is the collection of point it is a collection of point in a fixed distance of Manhattan arc. So, it is a collection of point within a fixed distance of the Manhattan arc. So, it is creating first the Manhattan arc then after creating the Manhattan arc we are creating the tilted rectangular region. So, we have discussed how to create the Manhattan arc. One of the important point is that the Manhattan arc slope of the Manhattan arc should be the slope of I am writing in short Manhattan arc should be plus minus 1. This is important. I will show that how the slope is plus minus 1. So, now we are doing a collection of point within a fixed distance of Manhattan arc.

So, we will discuss that in the next slide. The core of TRR, the main core of TRR is a subset of its point at the maximum distance from its boundary. It is basically we will see this in the next slide. This is my Manhattan arc. I am writing Manhattan arc in short MA. So, now we will discuss this is the Manhattan arc. Now we are creating the tilted rectangular region or TRR for the Manhattan arc. So, what is the basically point here is that for the Manhattan arc we are creating the tilted TRR for sink nodes S1 and S2 with a radius of 2 units. So, if you can see here, so what it is saying that this tilted rectangular region is basically in this case is 2 unit from the Manhattan arc. If you can see here, this is the one of the point in the Manhattan arc and this is another point in the TRR. So, the distance is one grid, this is one grid and this is another grid.

So, 2 units. Any point you see here, let us say this is a point in the grid. So, it has also distance from the Manhattan arc is 1 and this is 2. Similarly, you can calculate for all other points. So, what it is saying that all the basically distance from the Manhattan arc is with a radius of, it is not the radius, it is the better word will be the distance from the Manhattan arc should be 2 units because here we are calculating the basically Manhattan distance.

We are finding the Manhattan distance. Manhattan distance means that either horizontal line or vertical line. So, basically this is my TRR. Now there is a concept called merging segment. So, this is very important. What is this merging segment says that is a node V in the topology denoted by MS of V is the locus of feasible locations of V consistent with exact 0 skew with minimum wire length, exact 0 skew with minimum wire length.

Now we have basically 4 points are there. So, I have S1, S2 are 2 points. This is my TRR for this corresponding to S1 and S2 and this one second one is basically TRR corresponding to S3 and S4. So, what we are finding is that we have basically two things MS of U1 merging segment of U1 and merging segment of U2. So, U1 is corresponding to the Manhattan arc S1 and S2 and MS U2 is corresponding to the Manhattan arc S3 and S4. So, what we are doing here is that we are finding a common segment of the TRR of both the rectangles. What is the common name? So, that is basically called merging segment U3. So, main point is that is the locus of feasible location of U3 with 0 skew and minimum wire length. So, this U3 is the points if you go pass through that point or the segment you will have basically 0 skew with minimum wire length. There are two concepts.

One is 0 skew and the second one is minimum wire length. Now what we are doing is that we are basically this is the actually sink nodes S1, S2 like that total 8 sink nodes are there or the basically the points where the clock should be connected. Then you are doing two at a time. If you can see here one important point observation is that first of all we will take two sink nodes at a time then we will create the Manhattan arc. So, here if you can see this is the Manhattan arc and this is the Manhattan arc and this is the Manhattan arc and this is the Manhattan arc. Basically, if you can see here this slope of the Manhattan arc is either is plus minus 1.

So, here it is basically 1, S8 and S7 is 1 and S2 and S1 it is minus 1. Basically, plus minus 1 slope should be plus minus 1. So, now what is our concept here is that we are basically creating two TRR. So, this is the TRR and this is the TRR and this is the TRR and this is the TRR and this S0 is very important. It is the point where the clock is entering to our design or the chip. So, now what is the concept here? We are creating the points between the two TRR which is having the same skew or zero skew and minimum balance.

So, this is the point corresponding to let us say this is I will denote this way TRR1, TRR2, TRR3 and TRR4. So, this segment is corresponding to TRR1 and TRR2 and this segment is corresponding to TRR3 and TRR4. Now we have these two segments which is important to us. Now what we are doing is that we created TRR for these two minimum locations like this is my TRR5 and this is my TRR6, TRR5 and TRR6. Then what we are doing we are finding the common segment which is giving me the same delay at this point is this one. Now what we are doing we are now working in the top-down phase. So, top-down phase what we are doing is that we are finding the all possible location of the child node V given the location of its parent node par. So, now we are connecting the parent node with the child nodes. So, you can see here this is my parent TRR, this is my ms of V. Then we are connecting the final placement of the parent node.

How I can connect this parent node with the child nodes. So, there is algorithm is there. I will go to the actual example how it is happening. So, what we created we are going in the top-down format. So, now I have a clock signal entering my design or the chip. Now I need to connect to my midpoint of the basically segment what we find by the intersection of the TRR.

So, now what is happening is that from this point I am again creating a TRR. The TRR is a point where any point I go the distance is same. Now I will see that at what point that line segment, this line segment is crossing the TRR. And it should not be outside that. So, this is the point where it is crossing and I cannot go to this point.

I cannot use this point because this is outside the rectangle. And this point is the best point because other points are not on grid. So, now what is happening is that I draw till this point. I choose this point. I choose this point for my clock tree routing. Now what I have to do, again I do the same approach to create a TRR around that tapping point. So, this is the tapping point here. This is the tapping point here. Then this is the TRR created around the tapping point let us say TRR A. This is TRR B. Now where it is touching. So, here it is touching. So, here it is touching but this is not on grid. So, we have to check the point where it is on grid actually. So, similarly I will go here. It is not on grid.

So, I will go till this point actually. So, this is my tapping point. This is my tapping point. So, this is my tapping point. So, these are my tapping points actually. So, now I will do the same procedure repeatedly to create my clock tree such that my two objective, one is zero skew and second one is minimum wire length can be achieved. So, this is a procedure called Deferred and Merge Embedding algorithm which is more popular in industry.

Thank you for your attention.