VLSI Design Flow: RTL to GDS

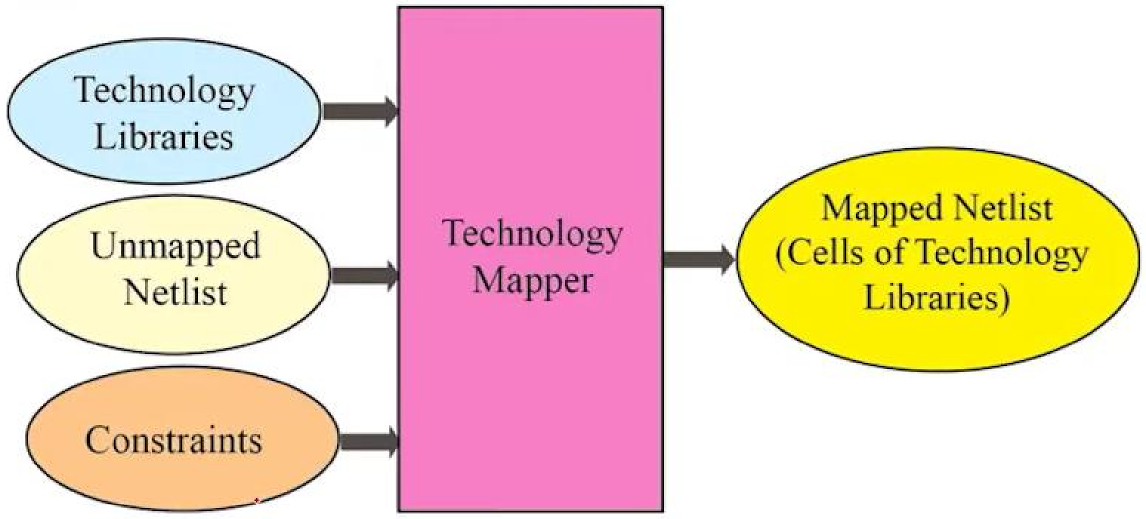
Dr. Sneh Saurabh

Department of Electronics and Communication Engineering
IIIT-Delhi

Lecture 34
Technology Mapping

Hello everybody, welcome to the course VLSI Design Flow RTL-2 GDS. This is the 27th lecture. In this lecture we will be discussing technology mapping. In the earlier lectures we had seen that logic synthesis consists of various sub tasks which are shown in this figure and in earlier lectures we had discussed RTL synthesis and logic optimization. So at the end of logic optimization we get a netlist which is in the form of generic logic gates or netlist which is in terms of generic logic gates and it is optimized. Now after we have got a netlist which is optimized and in terms of generic logic gates we need to carry out a task which is known as technology mapping.

So what does a technology mapping do? So technology mapping basically transforms the netlist which is represented in terms of generic logic gates to a netlist which is in terms of standard cells which are in the technology libraries. So in this lecture we will be looking at technology mapping. So first let us look into the framework of technology mapping. So the framework is shown in this figure.



So it takes these inputs and produces a map netlist which is the output. Now what are these inputs? So the first input is a set of technology libraries. So for technology mapping we need to provide to the logic synthesis tool the set of technology libraries from which the cells will be picked and will be used in the netlist. Now just to recap, when we looked into technology libraries we discussed that a technology library consists of various kinds of cells and these cells perform different logic functions. For example, combinational logic functions like AND gate, OR gate, AND or INVERT and multiplexer and so on.

And there can be sequential circuit elements also such as latches and flip-flops. Additionally for the same logic function a technology library can contain cells of different sizes. So for example, for a logic function of NOT gate or an inverter we can have an inverter of size 1x where x is some unit of area or and there can be an inverter of size 2x then there can be a third inverter which has a size of 4x and so on. Similarly for other kinds of cells for example NAND gate, NOR gate, OR gate, AND gate there will be multiple cells of different sizes which deliver the same functionality. Additionally the additionally this when we provide technology libraries to the logic synthesis tool we can provide technology libraries of various types.

So what we mean by type is that there can be some technology library which is implemented using transistors of, say low threshold voltage. So low threshold voltage or low VT. So the transistor in this case is a low VT meaning that it works at lower threshold voltage and it turns on much more easily. So when a cell is implemented using transistors which are of low threshold voltage then typically its speed is high, but it also consumes more power. And similarly there can be another technology library which is in which the cells are implemented using transistors of high VT.

So it can have a high VT. So for this the speed will be lower because these transistors require more voltage to turn on so the speed will be lower, but it will consume less power dissipation or consume less power. So when we provide technology libraries to the technology mapper we may provide a set of libraries meaning that some libraries can be of low VT some can be of high VT and so on. Now another input or the most important input that we need to give to the technology mapper is the un-mapped netlist. So what is an un-mapped netlist? So this is a netlist which is implemented in terms of generic logic gates.

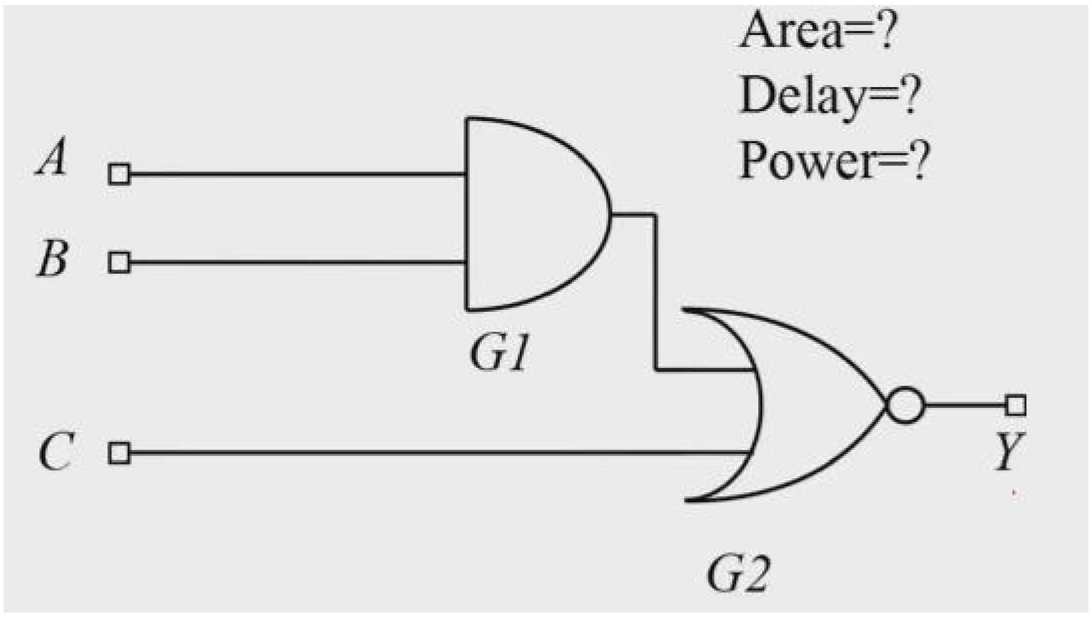
So just to recap, generic logic gates do not have a transistor level implementation. So the function is well defined for generic logic gates, but its transistor level implementation is not defined and therefore the area, the timing and the power dissipation these things cannot be computed for generic logic gates. So when we give a design to a technology mapper the design is in terms of netlist and that netlist is implemented in terms of generic logic gates. And the third important input that we need to give to the technology mapper is the constraints. So we have seen in the last lecture that we can represent constraints using SDC format Synopsys Design Constraints format or these constraints are in the.

sdc files. So we can give the sdc files as input to the technology mapper and use these so what information does the sdc files contain? So the sdc file contains information about the clock frequency, the environment of the design in which a design is expected to work and so on. And based on these constraints file the mapper can take its decision that way for which path which kind of cells needs to be instantiated in the netlist and so on. And also while performing optimization a technology mapper needs to know what is the objective or what is the objective it is required to meet. For example the objective needs to be that we want to minimize the area for our netlist under a given delay constraint or it can be that we want that the mapper should minimize the delay under a given area constraint. So these are some examples of the objectives that the tool needs to or the mapper needs to consider.

Now in the logic synthesis tool there are options like effort high, effort low and so on. And based on the effort level of the logic synthesis tool the objective of the technology mapper can change. For example if the effort is high for timing then the primary objective of the technology mapper will be to minimize the delay. So in addition to these all inputs what does a mapper do? A mapper produces a netlist consisting of library cells. And while producing this netlist the most important thing that the mapper should consider or ensure is that the unmapped netlist and the mapped netlist these two netlists must be functionally equivalent.

So the functional equivalence must be preserved. So whatever we mean by functional equivalence is that for a circuit if we are giving a sequence of 0s and 1s at the input then it produces a sequence of 0s and 1s at the output. So for the unmapped netlist and also for the mapped netlist the sequences produced at the output should be exactly the same for a given set of input sequences of 0s and 1s. So that is what we mean by functional equivalence. Now let us take an example to understand what is done in mapping.

Let us consider this netlist which is in terms of generic logic gates meaning that G1 and G2 are generic logic gates.



Since G1 and G2 are generic logic gates the area, delay and power for this netlist cannot be computed. However the functionality is very well defined and we can represent this functionality as Y is equal to A dot B meaning that this is because of this AND gate we have this signal as A dot B and then we have a NOR gate.

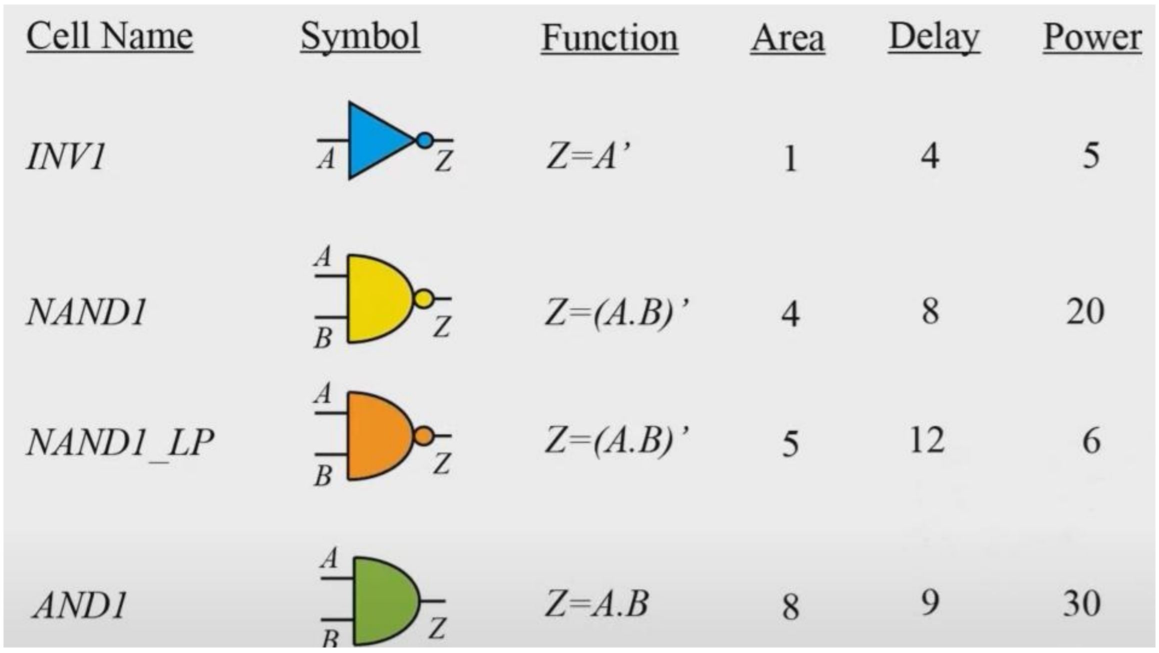
Logic Function Y = (A .B + C)`

So that is why we have A dot B plus C as a whole complement. So the function for this netlist is very well defined but other PPA measures like area, delay and power are not known.

And for this netlist let us assume that a set of libraries is given in which these are the cells. Now we are taking only a few cells just for illustrative purposes; typically a technology library cell contains hundreds of cells. But for illustrative purposes let us assume that the technology library contains only these four cells. So we have an inverter we have NAND gate which is written as NAND1 and then is another NAND gate which is NAND1 underscore LP just to signify that it is implemented in say high threshold voltage transistors and therefore it will consume low power and that is why we have underscore LP. And then let us assume that there is another gate which is AND1.

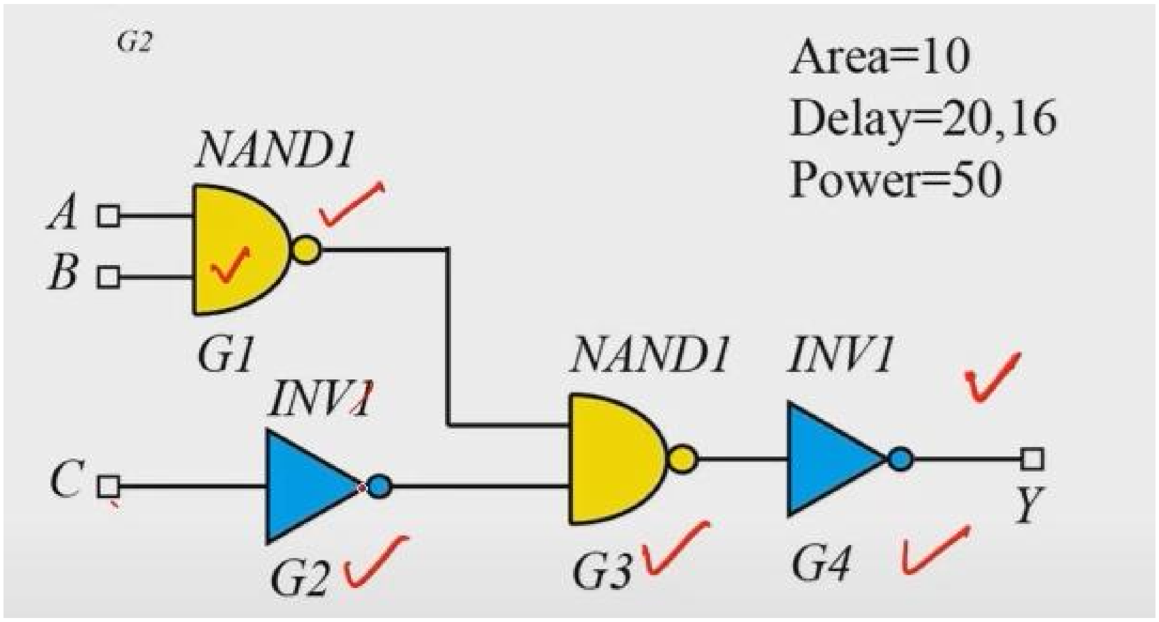
Now for these logic standard cells or cells in the libraries the PPA measures are unknown. For example the area, the delay and power numbers these things will be specified in the technology library. So in the format that we have discussed earlier. So let us assume that these numbers are given for example for inverter area is 1 in some unit and delay is 4 in some arbitrary unit and power is again is 5 in some arbitrary unit. And in the same unit the area for the NAND1 is 4 for NAND1 LP is 5 and NAND1 is 8 and so on.

So let us assume that these PPA numbers for the cells in the library are given to us. Now let us understand how this net list in terms of generic logic gates will be mapped to the cells in the technology library. Now for given this unmapped net list one of the solutions that can be produced is shown here. So what the mapper can do is that it can instantiate these cells NAND gate NAND1, NAND1 and this inverter and implement the same functionality which is delivered by the unmapped net list. So first let us check whether this net list which is shown which is mapped is functionally equivalent to the given unmapped net list.



Cell Name	Symbol	Function	Area	Delay	Power
INV1		*Z=A'*	1	4	5
NAND1		*Z=(A.B)'*	4	8	20
NAND1_LP		*Z=(A.B)'*	5	12	6
AND1		*Z=A.B*	8	9	30

So for the given unmapped net list we have seen that the function y is equal to a dot b plus c whole complement. Now for this net list for the mapped net list let us compute what function it delivers.



Logic Solution ***$Y = (((A.B)'.\, C')')' = (A.B + C)'$***

So we can write y is equal to a dot b whole complement because this is a NAND gate and then it is taking a product with c complement because c is there and there is an inverter so this signal is c complement. So this signal will be a dot b whole complement into c complement and then whole complement and then again we complement it because there is another inverter at the. Now since there are two inverters, one inverter here and the other bubble here.

So this corresponds to two inverters so not or not is the same signal. So we can remove these two complement operators and we can just simply write y is equal to a dot b whole complement dot c complement. Now are these two equivalent so to establish their equivalence we have to invoke De Morgan's law. Now you might be aware of this formula x plus y whole complement is equal to x complement dot y complement this is from De Morgan's theorem. So now if we look into this part this is similar to this where x is same as a dot b x is same as a dot b and y is same as c and therefore we can write this as a dot b plus c like or if we use the LHS of this a plus a dot b plus c whole complement and this is the nothing but the given function which was here.

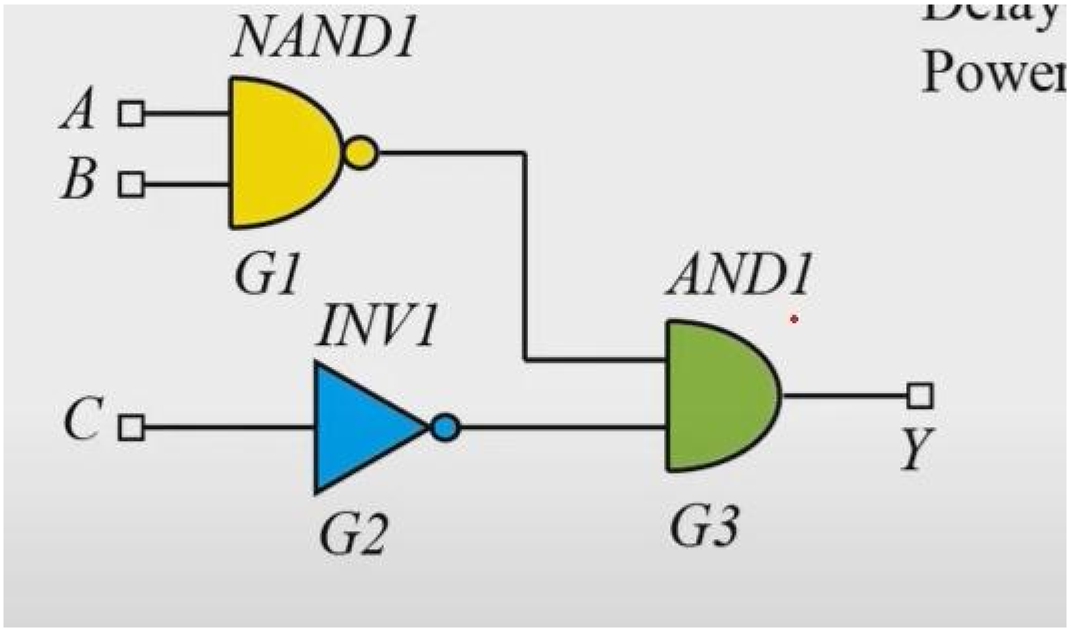
So from this analysis we can establish that the functionality delivered by the unmapped netlist and the mapped netlist both are the same. Now let us also evaluate the PPA numbers for the map netlist. So for example what is the area of this netlist map netlist. So in this case the area of NAND 1 is 4 and for the inverter the area is 1. So 4 plus 4 plus 1 plus 1 is equal to 10.

So 10 is the area of this map netlist and then what is the delay. Assume that the signal is coming at a, b, c at say 0 time instant. So what will be the arrival time of the signal at 1. So there are 3 paths the signal can arrive from this path it can arrive from this path and the third path. So from a and b what will be the delay.

If we look into the delay then the delay of NAND 1 is 8 and the delay of this again is 8 and the inverter is 4. So 8 plus 8, 16 plus 4, 20. So that is for the path from a to y and b to y that will be the same and it will be 20 units. And what will be the delay from c to y. The delay of the inverter is 4 and this one is in the NAND gate it is 8 and for the other inverter it is again 4.

So 8 plus 4, 12 plus 4, 60. So the arrival time of the signal y through c will be 60. So in a circuit the worst case delay matters. So if we consider the worst case arrival time in this case it will be 20. And then we can also evaluate the power numbers. Now it is given that the power consumed by NAND 1 is say 20 and for the inverter it is 5.

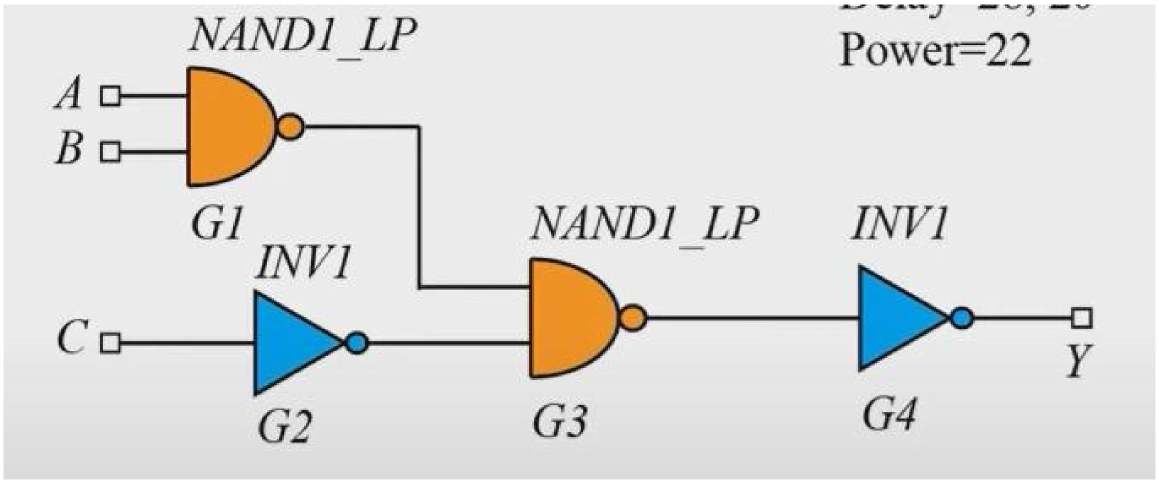
So the total power is 20 plus 20, 40 plus 10 that is 50. So these are the PPA numbers. Now this is one of the solutions that the mapper can produce. Let us look at another solution that can be produced for the same problem. So in this case we are using a NAND gate and then followed by an AND gate.



So now we have to first check whether it is functionally equivalent to the given function. So we can write for the map netlist we can write y is equal to a dot b whole complement right because of this NAND gate and then we take the AND of c bar or c complement right that is 1. And again if we invoke De Morgan's law then we can easily write it as a dot b plus c whole complement which is nothing but the given function right in these two are equivalent right. So the map netlist is still producing the same function right or it is implementing the same function as the given netlist right and therefore this mapping solution is functionally correct. Now let us also compute the PPA numbers for this circuit.

Now for this circuit the for the a what will be the area this has got an area of 4 area of 4 this AND gate has got an area of 8 right and this inverter has got an area of what. So 8 plus 4 12 plus 1 13, 13 is the 8 and what is the arrival time worst case delay or arrival time right. So the delay from a to y or b to y both are same and that will be the sum of the delay of this AND gate is 9 and delay of NAND 1 is 8 so 8 plus 9 is 17 and what about the other path from c to y it is delay is 4 right plus 9 plus 4 is 30 right. But what will matter is the worst case and worst case delay is 70 right. Now what about the power number so for the NAND gate the power is 20 right for the AND gate the power is 30 and for the inverter the power is 5 right.

So we have the power as 55 right. Now these are two possible solutions right there could be other solutions also let us look into a third solution right.



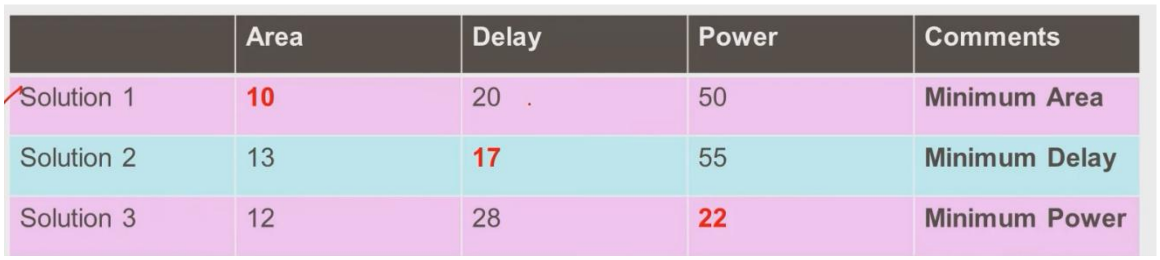
Now in the third solution the solution that was similar to what we had as the first solution only thing is that now we are using NAND 1 LP rather than NAND. Now functionally NAND 1 and NAND 1 LP both are delivering the same function that is NAND right but NAND 1 LP has low power but higher delay right. This is typical for a high VT high VT transit or as logic cells which are implemented in terms of high VT transistor right. Now the functional equivalence is or we have already established the functional equivalence for NAND 1 right.

So if we replace it with NAND 1 LP it will still be functionally equivalent right. So we need not worry about the functional equivalence of this map netlist right. But let us compute the PPA numbers for this map netlist. So for NAND 1 LP the area is 5 right. So this has got an area of 5 and the inverter has got an area of 1 right.

So the area is 12 right. Now what about the deal? So the worst path or the maximum delay will be in this path right. And what is the delay? Delay is 12 for this right. And then we have a delay of 12 for this also and delay of 4 for this inverter and 4 of this inverter. So for from A to Y the delay is 12 plus 12, 24 plus 4, 20. And from C to Y the delay is 12 plus 4, 16 plus 4, 20 right.

And what about the power numbers for the NAND 1 LP the power is 6, 6 and for the inverter the power is 5 right. So 5 and 5. So the total power is 12 plus 10 that is 20 right. So we have seen that with a given unmapped netlist like this and a given technology library or the cells in the technology libraries there can be multiple solutions which will yield functionally equivalent net results right.

But their QR will be varying. So to get a sense of the PPN numbers for three different implementations let us see a summary of this right. For the first implementation we see that the area is 10, delay is 20 and power is 50. Similarly I have tabulated the PPN numbers for all the three solutions. Now what we see is that the solution 1 delivers the minimum area right. Out of these three solutions the solution 1 gives us the minimum area right.



	Area	Delay	Power	Comments
Solution 1	**10**	20	50	**Minimum Area**
Solution 2	13	**17**	55	**Minimum Delay**
Solution 3	12	28	**22**	**Minimum Power**

And what about the delay for if we look into the delay we see that solution 2 is giving us the minimum delay right. And if we look into the power numbers then we see that the solution 3 is delivering us the power minimum power number right. So what this example illustrates is that for an unmapped netlist or a netlist which is implemented in terms of generic logic gates there could be multiple solutions which will be functionally equivalent but their PPA will be varying. So the task of the mapper is to choose a mapping solution or produce a mapping solution which meets its objective meaning whether it is trying to minimize area, whether it is trying to minimize delay or whether it is trying to minimize power. So this example basically illustrates the opportunities and the challenges of technology mapping.

Now let us briefly discuss how technology mapping is done. There are typically two types of approaches for technology mapping. The first one is structural mapping. In structural mapping what is done is that suppose there is a portion of a circuit of the unmapped netlist then this portion of the netlist is matched or compared with the various library cells and this map matching or comparison is done structurally. To match this portion of a circuit with the cells in the library what needs to be done is that both this portion of the netlist and the libraries or the cells in the libraries both are represented using some base functions and that base function can be NAND gate and an inverter or there could be other types of base functions which could be used by the technology mapper.

Because the portion of the netlist and the library cells or the cells in the libraries both are represented in terms of say these base functions to input NAND gate and inverter and suitable matches are found for each portion of our netlist and later on a min cost cover is found among the subset of the matches. So that is what is done in a structural mapping. Now a given Boolean function can be represented in many ways structurally. For example, suppose there is a four input AND gate right we can represent in terms of say two input AND gate like this. This is one structural representation of a four input AND gate in terms of two input AND gates.

Other representation can be, so for the same logic function the structure it can be represented in a form of or it can be represented structurally in many ways and therefore structural mapping needs to consider this and therefore many some of the matches which could be possible those are missed by structural mapping and that problem is solved using Boolean mapping. So in Boolean mapping what is done is the functional equivalence of the portion of the netlist and the library cells that are actually compared using Boolean methods. For example, both can be represented using say BDD and we have seen earlier that the BDDs are canonical and therefore the equivalence of two functions can be tested very easily. So if we use the Boolean method then some of the problems of the structural mapping can be solved and therefore these days Boolean mapping is preferred. Now the exact algorithms or rather the algorithmic description of technology mapping is outside the scope of this course.

So we will not be discussing these things further. However, if you want to know more details of technology mapping I will suggest that you can go to these references. So let me summarize what we have done in this lecture. So in this lecture we briefly discuss technology mapping. So we discussed what inputs are given to the technology mapper and what output it produces and we also discussed what are the opportunities in terms of PPA and the challenges of a technology mapper. In the next lecture we will be discussing timing driven optimization. Thank you very much.