VLSI Design Flow: RTL to GDS

Dr. Sneh Saurabh
Department of Electronics and Communication Engineering
IIIT-Delhi

Lecture 26
Technology Library

Hello everybody, welcome to the course VLSI design flow RTL to GDS. This is the 21st lecture. In this lecture, we will be looking at technology library. In the earlier lectures, we had looked into logic synthesis and we have seen that how the transformation of an RTL to netlist consisting of generic logic gates is done. And we had also looked into logic optimization that is performed on netlist consisting of generic logic gates. Subsequently in the design flow, we want to map these generic logic gates into the cells of the given technology library.

And we also want to do various analysis and various verification steps, we want to carry out on our design. For example, we want to do timing analysis and power analysis and so on. Now, the generic logic gates does not contain the attributes of the cell such as timing, power and so on. However, the cells of a technology library contains those attributes and those attributes are inside the technology library.

And therefore, the tool needs to look into the technology library to convert a netlist that is consisting of generic logic gates to the cells of the given technology library. So, the information contained in technology library is very very important for the subsequent design flows not only for the implementation but also for the verification. So, in this lecture, we will be looking into some details of the technology library. Now, the libraries that we use in in VLSI design flow are of two types. The first one is the technology library and the second one is the physical library.

So these are the most commonly used libraries in our design flows. Now what are the differences between these two libraries? So let us first understand that. So the technology library was initially created basically for the logic synthesis purpose and initially it contained only say details such as functionality of a logic cell or a cell. For example, it says that okay this is an AND gate, this is an OR gate and those kind of functionality were specified for the standard cells in the technology library. But with time what happened is that this technology library evolved and it started supporting or it started containing many many useful information for the design flow.

For example, it contains information for timing verification, for physical implementation and also for carrying out test activities which we will see later in this course. So technology library contains a lot of information which are useful throughout the VLSI design flow. And since this technology libraries contain primarily the data or the information of timing, therefore sometimes we also call this technology libraries as timing library. But keep in mind that technology library also contains information of power and many other information that we will see subsequently in this lecture. So in which format do we specify this technology libraries? So the most common way in which this technology libraries or the information of the cells in a technology is encapsulated is in the liberty format.

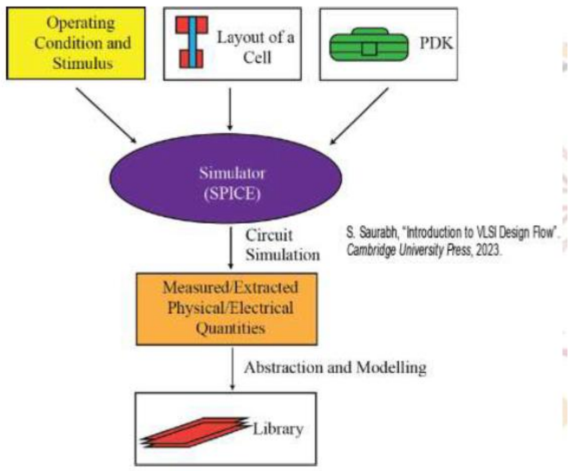
So the format in which the technology library is typically used in the design flow that format is known as the liberty format and the technology libraries typically have got the extension of .lib. So if you have a design infrastructure in that infrastructure you if you have said that .lib files then this .lib files most likely contains information of the technology library.

And these are ASCII files meaning that you can open it in an editor and just see that what characters and what things are written there. So these technology libraries are human readable. Now what about the physical libraries? So physical library contains abstract information about the layout of the cells and the technology for example the resistance per square and capacitance and so on. So it contains information of the technology such as say how many layers are there in the technology and also it contains the information in some abstract form about the layout of a cell that where the pins are and so on. So these information are there in physical library and the physical library is typically specified in a format which is known as library exchange format or LEF format and typically these the files the lef files will have an extension .lef.

The extension will be .lef and these are also human readable or ASCII files. Now this physical libraries are used in the physical design process and we will be looking into physical libraries in detail later in our course when we discuss physical design task. So in today's lecture we will be concentrating or focusing on the technology library. Now what is the motivation for using technology library? Why do we use technology library in our design flow? So the primary motivation is to simplify the design task. So thus the design task is basically finally getting a layout which delivers the required functionality.

Now this is a very complicated task and therefore we need to simplify it and we divide this design task into two tasks and therefore we are trying to simplify it or make it a two step process and there are many benefits of having this two step process and what are these two steps? So the first step is creating library. So the first step in designing is creating the library and the second step is using the library. Now what we mean by creating library, what we mean by using library we will just see. So creating library means that we are designing cells at the transistor level. So we are designing cells in a most optimum way for example NAND gate, AND gate, inverter and such cells are designed at the transistor level meaning that we make the topology, we make the layout of the cells, do spice simulation, verify the functionality and if we can make some improvement in the layout we do those things and make our cells very optimal in terms of what it delivers the functionality and also in terms of figure of merit meaning the delay and the power and the area and so on.

So the first step is the while creating library is that we design cells at the transistor level. So if we determine its optimal layout for each of the cell for example there will be AND gate, OR gate, NAND, NOR, flip flops and so on, many types of cells will be there and we will be designing or making an optimal layout for all these individual cells that is what is done in the creating library step. And then after we have designed our cells, we extract useful information out of those cells and keep them in a format or write them in a format of .lib or liberty format or some other format. We keep the information of the individual cells and once we keep that information of that in some file or in some repository then that constitutes a library and then we have created a library. So now once we have created this library then this library can be used by multiple designs for example using the same library say we have created a library for 45 nanometer technology then using this library many designs can be taped out or the layout of that can be designed and then fabricated for example we can get a processor, different types of processors, we can get a security chip or we can get various other functionality or chips of different functionality using the standard cells of the same library. So once we have designed many cells and a rich set of library consisting of many cells, we extract information and put that information in some form of library then that library can be used multiple times for various designs and for creating many chips.


Operating Condition and Stimulus
Layout of a Cell
PDK
Simulator (SPICE)
Circuit Simulation
S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.
Measured/Extracted Physical/Electrical Quantities
Abstraction and Modelling
Library

Now how does it help? So it helps by reducing the cost. So the cost of developing a high quality library gets distributed over multiple designs for example if we create one library and the same library is used by say 100 designs for example the various processors or various kind of controllers and many other chips that we have made using the same library say hundreds of designs are based on the same library then the cost in developing that library is distributed over all the hundred designs. So the overall cost of developing a high quality library and highly optimized standard cells that get distributed over multiple designs so that is why it saves cost. So this one, the creating library is the first part. The second part is using library in our designs so we said that using the one library, hundred designs were made, separate designs were made. Now these designs are using the library so in using library what the designer need to do. The designer need to instantiate cells from the library to achieve the desired functionality. Suppose there were say thousand cells in a library. Now these standard cells will be picked by the designer and instantiated in the layout and a design will be created. So the focus of using library is instantiation rather than designing individual cells.

So it allows designer to focus just on the instantiations of the cells rather than going into the details of the standard cells and therefore the design time and the effort of the designer who is using this library that decreases. So it also reduces the chances of introducing bugs inside the cells so once the cell is created in a library, we do not change the internal structure, internal details of the cell and if we are not changing we cannot make errors so the person who is using the library cannot introduce any extra errors inside the cells because he or she is not making any change inside the library cells. The changes are only in the instantiations. So the scope of error that also decreases a lot in creating a say multimillion gate design. And it raises the abstraction levels of the tools that are used in the design flow from the transistor to the cell level. So what it means is that say suppose a tool is doing timing analysis now the timing analysis need not be done on a design at the transistor level but the tool can do the analysis at the cell level and using a higher level of abstraction.

So the design tools that will be used in using the library, those design tools need to work at the higher level of abstraction and that higher level of abstraction is the cell level instead of a lower level abstraction that is a transistor level. By raising the level of abstraction what happens is that the tasks for these tools that decreases or the complexity of the task decreases tremendously. For example for the STA tool, for the physical design tools and other synthesis tools the task become much easier if they are working at a higher level of abstraction. So these are the primary motivation of making the design design flow into two parts: the first part is creating the library and the second part is using the lab. Now let us understand that how a library is created. So library is created using a process which is known as library characterization and this library characterization can be done either at the foundry level, foundry who is developing the technology. The foundry can also develop the library, they can design the standard cells and then give the library to the designer and the designer will will basically instantiate those cells in the libraries and make a design.

So either this characterization can be done at the foundry in which the foundry creates the library or individual design house can also undertake creating the libraries on their own. Now let us look into the various tasks that are done during creation of a library. So in the library creation what is done. The first part is we need to design each cell optimally and verify that the functionality of the cell is correct for example AND gate is working as an AND gate and we have to design the cell optimally meaning that the area should be low, the timing should be as good as possible and a good tradeoff should be there between the timing and the area, and the power numbers should also be acceptable and those kind of things are considered during designing the cell. So once we have got the design cell then what is done is that we do SPICE simulations for the cells and while doing this SPICE simulation, we specify various operating conditions and various stimulus for example given an AND gate, what will be the stimulus at one input and what will the stimulus at the other input and so on. So the stimulus and the operating conditions are decided and then the information of the transistor. Now this SPICE simulation is done at the transistor level and during this simulation it needs information about the transistor, the process and so on.

So that information is contained in the PDK or process design kit that we had seen earlier in the lectures. So using this PDKs and the layout of the cell that we have designed and the given stimulus, SPICE simulation will be done and based on the SPICE simulation, we measure and extract the parameters of interest. For example what is the delay under a given stimulus and then what would be the slew, what will be the transition at the output or what will be the voltage swing, what will be the capacitance. So these kind of information is extracted from the SPICE simulator while doing SPICE simulation and once this information is gathered then an abstract model is built and written in a given format and which can be say the liberty format. So from the measured and extracted physical and electrical quantities, an abstract model is built meaning that model which can be used later on by various design tools such as synthesis, STA or static timing analysis and physical design tools and so on. So abstract model is built and that model is then written in a file which will be .lib file or is in the liberty format and then once we get this library files then this will be used in the design flows subsequently.

Now why are SPICE simulations using PDK is not directly used in power delay computation. So let us understand that what will happen if we directly use the SPICE simulation in our design flow. We are saying that in the design flow or when we are using the library, the design tools work at a higher level of abstraction. They work at the cell level rather than at the transistor level. Now what if those tools were working at the transistor level, they were doing SPICE simulations using PDKs then what will happen. So in that case the tools will take lot of time because SPICE simulations are time taking and they are time taking because this SPICE simulation are very accurate. They build differential equationsfor a given circuit and then those differential equations are solved iteratively using iterative technique and these SPICE simulations are time taking.

Now for each instances in our design we try to do this kind of SPICE simulation then it will lead to a huge run time and that will not be acceptable. So what is done is that we use library models in our design flows. So what we do is that we extract information that we gathered for individual cells during SPICE simulation and model in the library as we have seen in the library characterization step. So the EDA tools use library models instead of SPICE simulation for computing delay, slew, power, voltage variations etc. and these are orders of magnitude maybe 100 or 1000 times faster than what would be the run time if we directly use SPICE simulations. So this is the primary motivation of using library models in our design flow is it speeds up all the verification and the implementation task. Now to allow this to be done seamlessly we have to consider some important attributes that these library models should have and what are these attributes. So the first attribute is regarding the speed and accuracy trade off. So now we are creating an abstract model. So we are using some abstract model or we extract information and model in the library.

So we are not modeling everything that we get from the SPICE simulation in our library. We model only important things that we consider will be required in computing say delay and other things. So we are actually losing some information from SPICE simulation but the trick is that we should lose that information which is redundant meaning that the accuracy loss by removing some information should be acceptable compared to the SPICE simulation which is considered as the golden result. Then the second important attribute of a library model is the robustness. So what we mean by robustness is that suppose there was an inverter in our library.

Now that inverter will be used at say 100 different places in our design, in our layout. So the number of instances of an inverter can be very huge and what we want is that for each instance of the inverter, the result should be or the delay number or power number should be computed accurately in comparison to the SPICE simulation. So the accuracy of the model should not be dependent on the instantiation or the environment in which we put our standard cells. So the library models for these standard cells should be such that the these models work fairly accurately at all kinds of instances in our design. That is what is meant by robustness and in our library model should be robust enough to handle various instantiation scenario.

And then this library model should be portable. It should not be that this library model is usable only in this tool. It should be usable in say static timing analysis. It should be usable in say synthesis, physical design and so on. And not only various stages of design flow, it should also be portable from one vendor to another.

So we use the EDA tools from various vendors. So the ideal library model should be one which should which works with all EDA vendors or the tools of EDA vendors. And then the library should have variety of cells and there should also be some uniformity. What we mean by variety and uniformity let us see. So by variety we means that we should have multiple cells of the same function.

For example, if we have say an inverter so we should have inverter of say size 1x, size 2x, size 4x and so on. So as we increase the size of the inverter then the drive strength will increase. And as a result it will become say faster, it can drive the load faster but at the same time it will incur more area on the wafer. So now the library should be rich enough and it should contain cells of various drive strength and the same functionality. Why we want to do this? We want to do this so that there is sufficient free degree of freedom for the synthesis and physical design tools.

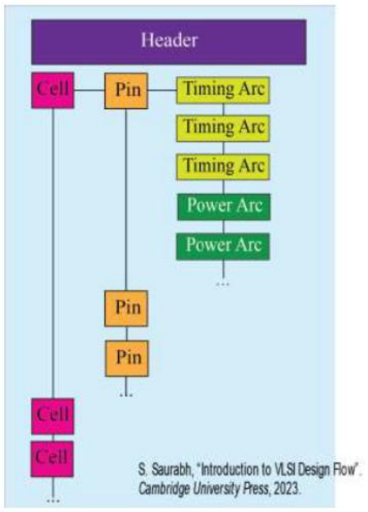
So if there is a critical path then probably the synthesis tool can instantiate say inverter of size 4. Then it will be driving much faster than the case for an inverter of drive strength say 1x or its 4 times smaller than 4x. If an instance is not in a critical path then probably it can use the inverter of size 1x. So we give the freedom to the synthesis and implementation tool to choose whichever library cell it wants to pick based on the trade-offs of various figures of merit. And that is why we want the library to be as rich as possible.

So then the tool will have many choices for various instances and it will pick the choice based on its requirement. And then there are we want libraries of various flavors meaning that some library will deliver low power, some library will develop deliver high performance, some library will be very compact, have very compact cell and will give high density or there can be libraries of different threshold voltage which we call as as low VT library or low threshold voltage transistor and then there can be high threshold voltage transistors and so on. So depending on our design requirement we can use different libraries for the same technology. And what about the uniformity? So now we have created various types. For the same functionality for example, inverter we created three different sizes of inverters. Now while creating these cells we also maintain some uniformity and the most commonly used uniformity is uniformity in height meaning that though the size of the cell will change for example, 4x will be four times bigger than the transistor than the inverter of 1x size, but we do not change the height of the cell. We keep it fixed. So what do we change? We change the width.

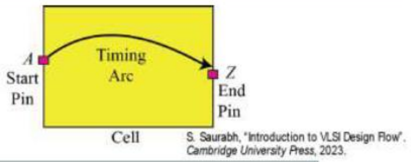
For example, if this is the layout for inverter of size 1x then the size of or the layout of 4x inverter will be something like this. The height will remain the same. The width will be increased by four times. So why do we put this kind of constraint on the height of the cell ,we do this so that our so that the the physical design tasks become very easy. So if we keep the height of the standard cell uniform then we can arrange standard cell in a kind of in various rows of a constant height and so on and making layout and routing much easier than if the height was allowed to be changed in arbitrary manner. So it eases the task of physical design. Now what does a library contain now let us go into the library and understand that what things are there in the library.

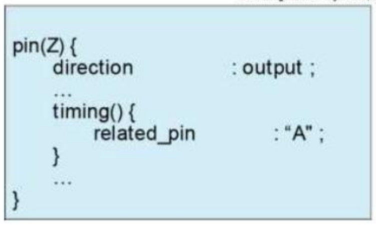
So it contains information of process parameters voltage and temperatures and these three parameters are basically clubbed together and we call them as PVT condition. So we encounter this term PVT a lot many times in design flow and what PVT refer to PVT refer to process, voltage and temperature combination of these three things and this information is there in the library. Now library contains many cells and therefore it contains cell data and in cells we have various pins and then pins will have functionality and other information of timing, area and power for the cells will also be there at the cell level and then what type of cells are there in the library. So library typically contains cells of hundreds of different logic functions for example there will be combinational and sequential standard cells for example AND gate, OR gate and so on, flip-flops, latches and these types of gates will be there in the libraries. Then there will be IO pads those are special kind of cells through which signals come inside our chip those are known as IO pads or input output pads and there can be also cells which are implementing memory functions and there can be bigger macros also for example half adder or adders, multipliers and those things, those information or the cells of these type can also be there in the technology library. Now when do we use libraries. So we use libraries throughout our design flow, from RTL to GDS and it is used in synthesis, timing and power analysis, verification, design for test, physical design.

So in rest of the lectures that we will be discussing, most of them will actually be using some information from the library and therefore we have started looking into library before we move to the other design task. Now what is the format in which we keep the information in the library. So the format is liberty format and as we discussed it is an ASCII or text format. Now in liberty format, data is primarily stored as attributes and so there is a mapping between an attribute name and the value. So there will be a name and value. For example there will be an attribute name which is time unit and its value is say 10 picosecond so this is a name value pair. Attribute name is time unit and the value is 10 picosecond. Now in library, the information is organized as hierarchy of groups so let us see what is this hierarchy of groups . So this is the structure of information in liberty format. At the top there is a header and what does header consists of ? So at the top level there is a header and header contains PVT information, scaling factors, units, it contains information that are valid for all the cells and pin and arcs inside the library and then it contains the list of cells. So these are list of cells, so we have header and then we have list of cells. Now what does a cell contain? Now cell contains the information which is relevant for the cell for example, its area, the leakage power at the cell level and then it will contain list of pins, a cell will contain the list of pins. Now what does a pin contain/ Now pin contain the information which is relevant for the pin level for example what is its direction, what is its name, what is capacitance, then what is its functionality meaning whether it is an AND gate, OR gate and so on, those functionality is specified at the pin level and then it contains the list of timing arcs and also list of power arcs. So this is how the information is organized in a technology library. Now timing arcs are used to perform timing analysis and computing delays of the arcs and power arcs are used to perform power analysis.


Header
Cell
Pin
Timing Arc
Timing Arc
Timing Arc
Power Arc
Power Arc
Pin
Pin
Cell
Cell
S. Saurabh, "Introduction to VLSI Design Flow". Cambridge University Press, 2023.

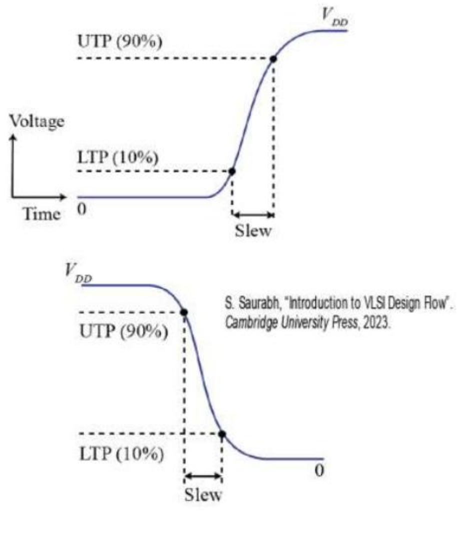
So we will be looking into timing arcs in more detail in this lecture. Now what are timing arcs so timing arcs are used to model timing attributes for combinational and sequential cells in the library, for example, consider this cell. This cell may be say inverter, AND gate, OR gate or whatever. So for this cell, for example, if we consider it as inverter then there is a pin A there is a pin Z. Now we know that there is a delay between the input A and Z so that information will be modeled using timing arc. Now more formally what does a timing arc has? Timing arc has a start pin, in this case A and an end pin, in this case Z. So the timing arc is specified at the end pin meaning that at the pin Z. So this is a snippet of technology library. So there will be a pin Z specified and inside this pin Z there is information about the timing arc. Now timing arc is specified, the information about this timing arc will be specified at the end pin, so end pin in this case is Z, so it will be specified at the pin Z and the start pin is specified using the attribute related_pin. So inside this timing arc, we have an attribute called related_pin whose value is A. So what does this A signify? This A signifies the start. So this is how the timing arcs exist in our technology library.


A
Start Pin
Timing Arc
Z
End Pin
Cell
S. Saurabh, "Introduction to VLSI Design Flow". Cambridge University Press, 2023.



```
pin(Z) {
      direction              : output ;
      ...
      timing() {
            related_pin          : "A" ;
      }
      ...
}
```

Now the timing arcs can be of two types: the first one is delay arc and the second type is of constraint arc. So constraint arc are related to setup check and hold check. So we will be looking into both types of arc in more detail. Now what does this timing arc contains? So this timing arc contains attributes of timing for a given arc and what are these attributes? So these attributes can be related to slew or it can be related to say delay. So what is slew, let us understand that first. So slew of a signal quantifies how steeply or sharply transition occurs from 0 to 1 or 1 to 0 meaning that you have 0 and you have 1 so how steeply it goes, how fast it goes from 0 to 1 or 1 to 0, that is measured by the slew of a signal. So slew is measured by defining two transition points. Now in technology library, we define slew by assuming that our signal is rising in a linear manner. So if say it is a 0, this is a 1, we say that from 0 to 1, the signal rises in some manner and the slew characterizes basically the average kind of slope of this line. Now to measure this slope what is done? During characterization of a timing library, first two transition points are chosen. These transition points are known as lower threshold percentage, LTP and upper threshold percentage or UTP and this information is present in the header of the technology library. So what are these two transition points for example during library characterization we can say that lower threshold percentage is say 10% and these are with respect to the VDD of the signal. So one point is the 10% of the VDD, this point it may be and the upper threshold point can be this point which can be say 90%. So we say that there is a signal and we have chosen during library characterization say at 10% as LTP and 90% as UTP, then this kind of threshold is known as 10-90 threshold because we have to taken two points 10% and 90%.

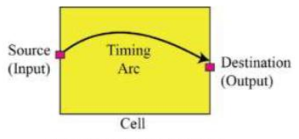

V_{DD}
UTP (90%)
Voltage
LTP (10%)
Time
0
Slew
V_{DD}
S. Saurabh, "Introduction to VLSI Design Flow". Cambridge University Press, 2023.
UTP (90%)
LTP (10%)
0
Slew

So based on this definition or this choice of our threshold point or transition point how do we define slew? So we define that the slew can be for different for the rising and the falling case. So in this case, since the signal is rising, we say that this is a slew and this is the time taken for a signal to reach from 10% to 90% so whatever is the time duration or time taken for a signal to go from 10% to 90% that is defined as a slew of the signal. Similarly we can define fall slew as the time taken by the signal to go from say UTP to LTP, so in this case it is 90% to 10%. Similarly the slew threshold can be between 20-80. So LTP can be chosen as 20, UTP can be chosen as 80, so then it is known as 20-80 threshold or it can be 30-70 threshold. So whatever the threshold will be that will be predefined in the header of the technology file and from there we can get the information or the tool can get the information that what was the definition of slew used by the tool while characterizing. Now this rise slew, fall slew are also known as rise transition time, fall transition time or rise time and fall time also. So there are various names all refer to the same thing meaning how sharp the transition occurs from 0 to 1 or 1 to 0.

Now the other definition is related to the delay of the timing arcs. Now what does delay quantifies? It quantifies how much time it takes for the change in input to propagate to the output and therefore this delay is also known as propagation delay. So suppose we had a a timing arc from this source to the destination. So if the delay can depend on the direction of the transition meaning that whether the transition at the output is rising or it is falling, the delay can be different for the rise transition or the fall transition and therefore we define two types of delays. One is the rise delay and rise delay is the delay which refers to the delay observed when output is rising so note that the rise delay is with respect to how the signal is transitioning at the output not at the input so rise delay is output rising and fall delay is output falling so the convention is that we define a delay arc based on the transition at the output of the arc not at the input that is the convention. Then again as we had chosen threshold for the slew similarly we choose threshold for the delay also for characterizing. So during characterization, a predefined threshold points on the input waveform and the output waveform are chosen.

Delay Definition

- Quantifies how much time it takes for the change in input to propagate to the output

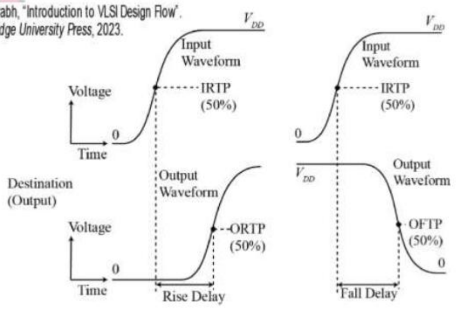


S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.

Delay can depend on the direction of transition (rising/falling)

- Rise delay: output rising
- Fall delay: output falling

S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.



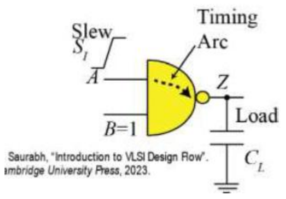
- Predefined threshold points on the input waveform and the output waveform.
- For the input signal:
  - input rise threshold percentage (IRTP)
  - input fall threshold percentage (IFTP).
- For the output signal:
  - output rise threshold percentage (ORTP)
  - output fall threshold percentage (OFTP)

So separate thresholds are there for the input and the output and these thresholds are mentioned in the header of the technology library. Now the input signal can be either rising or falling so for the case when the input signal is rising we define input rise threshold percentage IRTP and for the following case, input fall threshold percentage IFTP. So these threshold points are mentioned in the technology library. For example, let us consider this input signal. So this was the input signal and corresponding to this input signal, the output was this signal. Now at the input side we have defined a threshold as 50% so we have chosen an IRTP as 50%, since input is rising, so we have to take the input rise threshold percentage or IRTP, suppose it was 50% so this point refers to the IRTP point. Similarly for the output signal, we define output rise threshold percentage and output fall threshold percentage. Now this is the output signal and suppose we have defined output rise threshold percentage as 50%. Now since this signal is rising, we have to use this threshold.

So now how do we define delay? Now we define delay as the time elapsed between the input crossing the input threshold point and the output crossing the output threshold point. So the delay between this or the time elapsed or time interval between these two points, this is defined as the delay. So in this case the output is rising, so this is a rise delay. Similarly if we consider that the input was rising and the output was falling so since the input was rising we have to take IRTP as the threshold point, this one and since the output is falling, so we have to take this one, assume that this one was also specified as 50% in the technology library. So we get the 50% point or 50% of VDD point, this and the time elapsed between these two, this is known as the fall delay, why fall delay? Because output is falling so this is how we define delay in our technology library.

Now let us look that how does the delay and the slew of a gate is characterized in the technology library. Now let us consider this NAND gate, which drives a load of value CL this is a capacitive load and it gets a signal which is rising and it has got a slew of S_I. Now if we consider the delay of this timing arc and its output slew meaning the slew at this output pin Z, then these two things delay and output slew depends on the input slew that how fast the signal is transitioning at the input and the load, the capacitive load that is there at the output. So the delay and output slew depends on the input slew and the capacitive load. So the delay and slew depends on these two things. So if we look into this relationship between these delay and these input slew and output load, we will find that delay and output slew is a nonlinear function. Now this is a nonlinear relationship and modeling the same, capturing this information in the technology library is a bit difficult. So how do we handle this case? So we model the delay and the output slew as two-dimensional discrete point tables like this. So we choose a set of capacitive load CL1, CL2 and CLn, we have chosen n capacitive load and we choose a M input slew S_i1, S_i2 and S_im. So these are the input slew. Now for each discrete slew and capacitive load, will have a separate delay and the output slew value. So this is how we approximately model this nonlinear relationship of delay and output slew on the input slew and the output load and if in the design, we get a load which is something between CL2 and CL3 and we get a input slew which is something between S_i2 and S_i3, so we will choose these four values corresponding to these numbers and use some technique of interpolation maybe linear interpolation or so on to compute the delay and output. So this is how the information is modeled inside the library. It is modeled as discrete two dimensional table and this model is known as NLDM, nonlinear delay model. Now we will have different tables for delay and output slew.

CMOS : Characteristics of Slew and Delay

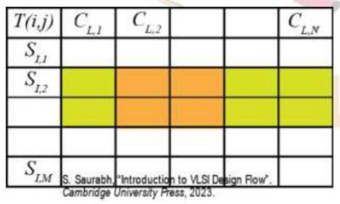


Saurabh, "Introduction to VLSI Design Flow". ambridge University Press, 2023.

- In general, the **delay** (D) and **output slew** (S_{OUT}) of a given **timing arc** depend on:
  - ➢ Input Slew (S_{IN})
  - ➢ Output Load (C_L)

- The relationship may be non-linear:
  - ➢ $D = f(S_{IN}, C_L)$
  - ➢ $S_{OUT} = g(S_{IN}, C_L)$
  - ➢ f, g are non-linear function (typically monotonically increasing with S_{IN} and C_L)

- Modelled approximately as two dimensional discrete point tables



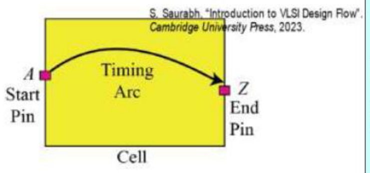
$T(i,j)$	$C_{L,1}$	$C_{L,2}$			$C_{L,N}$
$S_{I,1}$					
$S_{I,2}$	🟨	🟧	🟧	🟨	🟨
	🟨	🟧	🟧	🟨	🟨
$S_{I,M}$					

S. Saurabh, "Introduction to VLSI Design Flow". Cambridge University Press, 2023.

- Intermediate values are ***interpolated*** from closest match
- Different tables for delay and output-slew

Now let us look at how is this delay model or nonlinear delay model written in a technology library. Suppose we had timing arc between A and Z. Now corresponding to this arc, we will have a timing arc something like this. So at the header, there will be say two variables, one corresponding to slew and other corresponding to load and then we have the index_1 and index_2 which is showing that slew is varying or taking values of 10, 20, 30 and load is taking a value of 1.2, 5.0, 15.0 and 37.5. Now corresponding to this slew and load, there is a rise delay on the timing arc between Z and A, this timing arc because related pin is A. And there is a cell_rise attribute which says that it is a rise delay at the output and we have a table which specifies that there are for different combinations of slew and load, we have different values. Since there are three slew numbers and four load numbers, there are 3*4=12 entries in this timing arc.

Non-linear Delay Model (NLDM)


S. Saurabh, "Introduction to VLSI Design Flow". Cambridge University Press, 2023.
A
Start Pin
Timing Arc
Z
End Pin
Cell

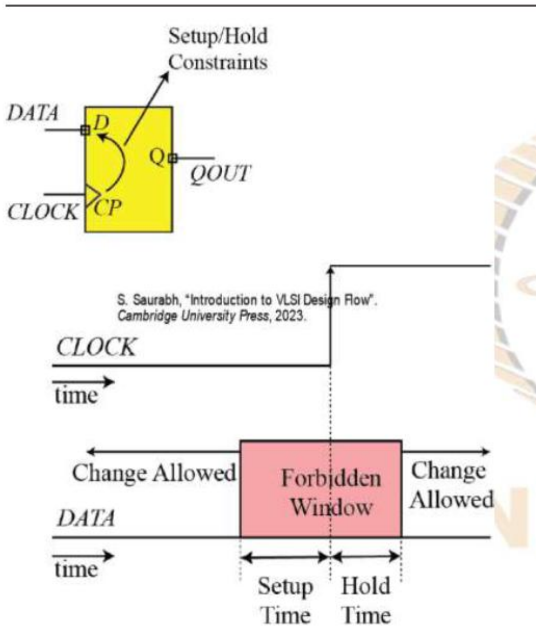
```
u_table_template(index_1) {
            variable_1 : input_net_transition ;
            variable_2 : total_output_net_capacitance ;
            index_1( "10, 20, 30" ) ;
            index_2( "1.2, 5.0,15.0, 37.5) ;
}
....
pin(Z) {

            timing()    {
                        related_pin           : "A" ;
                        timing_sense        : positive_unate ;
                        cell_rise(index_1) {
                                    values( "   4, 5, 7, 12,   ...3x4 table);
                        }
                        ...
            }
}
```

S. Saurabh, "Introduction to VLSI Design Flow".*Cambridge University Press*, 2023.

Now the modeling of delay and slew that is done using the NLDM model is not always accurate especially at the advanced process nodes, there is a significant accuracy loss and there are many reasons for that so one of them is that in NLDM model, we are assuming that the output is driven or the cell drives an output which is totally capacitive. Now what if there is a high resistance at the output and also because of Miller effects and also due to nonlinear effects, the assumption of the slew being one number that becomes inaccurate. What I mean to say is that suppose there was a nonlinear signal. Now for a nonlinear signal assuming that the signal has got one slope and that can be represented using slew that may not be accurate and there are accuracy losses because of that. Therefore at advanced process node, what we do is that we use other delay models for example based on current source models. So at advanced process nodes, we use two types of current source model: the first one is known as composite current source model or CCS and the second one is effective current source model or ECS. So these delay models are more accurate than the NLDM model and at advanced process nodes, we use these type of models.

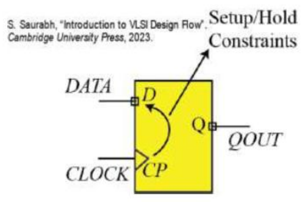
Now let us look into another type of attribute which is important for the cells or especially for the sequential cells in our design and these are related to the setup time and hold time. Now before going into the modeling of setup and hold time let us understand what is a setup time and hold time. Now suppose there is a flip-flop. So for a flip-flop to work in deterministically, a kind of relationship should be satisfied between data and the clock signal meaning that whenever the clock is say rising, the data should not change within some window of that. So there is some kind of forbidden timing window around the clock edge where we do not allow the data to change. So we define a forbidden window around the clock signal. Suppose this was the clock signal so we had our clock signal, which is rising and this is say data signal. Now if a data signal changes very close to this clock edge, then the data that will be captured by the flip-flop that cannot be reliably said to be 1 or 0. It can be something which we don't know and that state is known as kind of metastable state and we try to avoid that situation where we do not know what is the state of a flip-flop. So we define this window around this clock edge by two attributes and these attributes are known as setup time and hold time of the flip-flop.

Now what is a setup time? so setup time is the minimum amount of time, the data signal should be held steady before the clock edge. So if this is the clock edge before this clock edge, we say that the data signal should get stabilized within this window. It shouldn't change before this clock edge for some time and what is that time. That time is known as the setup time. So before the clock edge, the data shouldn't change for the setup time duration. In this window, we are not allowing this data to change. If data changes within this window, the value at the output pin Q of the flip-flop cannot be deterministically said to be 0 or 1. So this is a forbidden window for change before the clock edge and similarly we define that the minimum amount of time, the data signal should be held steady after the clock edge so that data is sampled correctly and deterministically. So after the clock edge we don't allow the data to change within for some window and that window is till this time within this time we don't allow this the data to change if data changes within this small period after the clock edge that is defined by the clock hold time that then we say then the output Q may take any value or non we cannot deterministically say whether it is 0 or 1 so we define hold time as that minimum amount of time the data signal should be held steady after the clock edge so that data is sampled correctly and deterministically.



Now these setup and hold times are modeled inside our library. Why do we want to model this inside our library because these are attributes of the standard cells, for example, the flip-flops and setup and hold time can be measured only with the help of say spice simulations. So only one who is doing the library characterization, they can know that what are the constraints that should be imposed on the on flip-flop so that the data is captured reliably so that's why we need to model setup and hold time in our libraries. So in general, the setup and hold time constraint depends on the data slew meaning that how fast the data signal is changing from 0 to 1 or 1 to 0 and the clock slew, how fast or how steeply the data or signal is going from 0 to 1 at the clock side. So the setup and hold time depends on the slew at the data and the slew at the clock.

Library : Modelling Setup/Hold Constraints


S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.
Setup/Hold Constraints
DATA
D
Q
QOUT
CLOCK
CP

- Intermediate values are interpolated from closest match

```
u_table_template(index_1) {
            variable_1 : constrained_pin_transition;;
            variable_2 : related_pin_transition;
            index_1( "10, 20, 30" ) ;
            index_2( "10, 20, 30, 40) ;
}
....
pin(D) {

            timing()    {
                        related_pin          : "CP" ;
                        timing_type : "setup_rising";
                        rise_constraint(index_1) {
                                    values( "   4, 5, 7, 12,
...3x4 table);
                        }
                        ...
            }
}
```

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

Now in general, again it is a non-linear function and similar to NLDM, we can model it as two dimensional discrete point tables and there are different tables for setup and hold. So similar to NLDM what we do is that in this case also we have a two dimensional table consisting of two input variables and these two input variables are data slew and clock slew and what are the output variables? The output variables are the setup time and the hold time. So let us take an example of a library and then we can understand. Suppose this was our cell, the way this was a flip-flop. The technology library can model the setup time in the way that is shown in this figure. So we will have constrained pin transition and we will have related pin transition. So there are two types of transition: one is related pin time transition, other is constrained pin transition referring to the transition at the data pin and the clock and then the setup time is specified on the pin D, the data pin and the related pin is the clock pin and that is why there is an arc between these two and then since there are three values for constrained pin and four values for related pin there will be a 3x4 table for the rise constraint for the setup case. Similarly there will be another table for hold case, hold hold time. So similar to what we have rise constraint there can be a fall constraint corresponding to data falling. So this is how the setup time and hold times are modelled inside the library. Now if we had the slew at the data signal and the clock signal, take some intermediate values which is not modelled according to the selected points then some interpolation technique will be used as we have seen for the delay case. Now these are some of the information or some of the important information that are modelled inside a technology library.

In addition to this, there are lots of other information which are contained in technology library. So the other information is related to power. So when we will be discussing power analysis, then we will be looking into power models more carefully. Then there are models for crosstalk and SI analysis and noise analysis and there are information of power, ground pins for the standard cells or the cells in our library and then there are some arcs which are conditional arcs meaning that those arcs get active only when some condition holds at the other inputs so those are known as conditional arcs and those are defined by sdf_cond and when conditions and then there are some attributes. So other attributes which will be there in the library which can be very much vendor specific. So in addition to the attributes which are supported natively by liberty, there are other attributes which can be vendor specific and there are ways in which we can extend the syntax to include vendor specific information in our technology library.

So if you want to go deeper into the concepts that we discussed in this lecture you can refer to these materials. So to summarize, in this lecture we have looked into technology library. We have looked into the motivation of using technology library in design flows. We looked into the characterization and the format in which technology library is represented. We also looked into nonlinear delay model and how setup and hold time of flip-flops are modeled. From next lecture onward we'll be looking into timing analysis or static timing analysis which is one of the most important task that is carried out throughout VLSI design flow. Thank you.