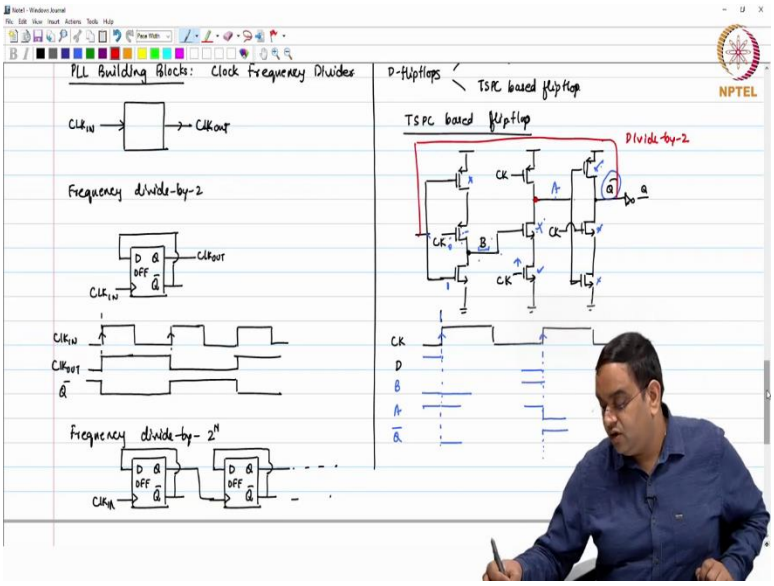
Phase-Locked Loops
Dr. Saurabh Saxena
Department of Electrical Engineering
Indian Institute of Technology Madras

Lecture – 57
Circuit-level Design of Clock Frequency Divider

(Refer Slide Time: 00:15)



Hello everyone. Welcome to this session. So, we have been discussing the PLL building blocks and in the PLL building blocks, the last one which we are going to see is the clock frequency divider. So, this is quite simple by the way as compared to the other blocks. So, what we need is we need a block where we give in the input clock let us call as CLK_{IN} and we get CLK_{OUT} and CLK_{OUT} should have frequency which is lesser than the input frequency and it should be some number which is fed externally.

So, to do that, one of the easiest ways of implementing a frequency divide-by-2, let us first look at that, frequency divide-by-2, quite often you use that, divide-by-2, divide-by-4 and so on. So, it can be implemented by using a D flip flop. So, you have D, Q, $\bar{Q}$, you have a clock and the output of this D flip flop is connected in feedback in this manner. And this is your CLK_{IN} and this is your CLK_{OUT}.

So, the D flip flop is surely connected to VDD and ground. Any circuit which we are using here has VDD and ground connection. So, if you look at it, when I give my CLK_{IN} like this, you can

start with the initial value $Q = 0$ and $\bar{Q} = 1$. So, this is CLK_{IN}, I am assuming that the initial value of Q or CLK_{OUT} is 0, that means $\bar{Q} = 1$. If $\bar{Q} = 1$ and you get a rising edge on this D flip flop, the input which is $\bar{Q}$, it is 1 right now. You get a rising edge on CLK_{IN}, so you get output 1 and it remains 1, the next output is going to change only at the next rising edge. So, when you get the next rising edge, till that time, because $\bar{Q} = 0$, at the next rising edge, your input was 0, so you get this. So, in this way, you can very well implement divide-by-2 clock.

Now, if you want any frequency division where frequency division is in the order of exactly frequency divide-by-2^N whether $N = 1, 2, 3, 4$. It is like divide-by-2, divide-by-4, divide-by-8, divide-by-16 and so on. You can cascade these blocks. So, I will just erase this part a bit, remove this also and connect it. I am going to cascade this, so this connection has to go like this, this is CLK_{IN} and similarly you can keep on going to have divide-by-2 or divide-by-4 and so on.

Now, these D flip flops which you are going to use for this particular division, these D flip flops which are commonly used. There are two kinds of flip flops which we use. One is you can say Strong-ARM latch based flip flop and the other is TSPC based flip flop. These are the two commonly used flip flops. TSPC based flip flop has a lesser power consumption in general as compared to strong-arm latch based flip flop and it is also quite fast, it can operate at a higher frequency.

So, let us look at this TSPC based flip flop. So, what you have here is you have clock signals coming like this, input is D, this is clock CK here and the output of this goes to another stage where the next two stages, you give clock here also and at this place and the output goes here, I call this as B. The output of this stage goes to another stage NMOS and this is here. So, this here and this is clock. The output here is inverted.

So, if you want actually Q with respect to D, you can add another inverter and you can use that as D, this is Q. So, if you want to look at the operation, let us just look at it. You have clock and D. Let us say you are going to get the rising edge of the clock like this. Consider the case when clock is 0, so, during the time when clock is 0, look at the signals B and A.

So, when clock is 0, at that time, if $D = 1$, $D = 1$ that means your clock is 0 and $D = 1$, so, this transistor is turned off. $D = 1$ means $B = 0$. $B = 0$ and clock actually is 0 which is like this

particular signal $A = 1$ irrespective of the input. At clock 0, $A = 1$. Now, $A = 1$, this transistor is cut-off, whatever $\bar{Q}$ value you have, it will be there and this transistor is also off.

Now, if D was 1 and we see that your clock goes high. When clock goes high, at that time, this PMOS transistor was initially active but because this was off, so, it was not doing anything. B still remains at 0. B remains at 0, clock goes high, this transistor turns on, A goes low. These two get turned on, A goes low. If A goes low, your PMOS, this PMOS gets active. This NMOS gets active but this NMOS is off because A is low.

So, your $\bar{Q}$ signal actually goes high. High value because clock is 0, Q we do not know what the Q value is. It depends on what the previous value you had, $\bar{Q}$. Now, when you get the rising edge on the clock, what is going to happen is B was 0. So, at that particular point because B was 0 and clock is high, this transistor is still off, this transistor is on but nothing will happen. A will still remain 1, A does not change.

If A remains 1, clock goes high, your $\bar{Q}$ actually goes low. B still remains low because $D = 1$. Now, consider the case when $D = 0$. In case $D = 0$, at that time, your B signal is going to be high here and if B signal is high, A signal is surely also going to be high, $\bar{Q}$ whatever state you have from the previous, it will continue and then when clock goes high because B was 1 earlier, that time your A signal will go low. If A signal goes low, then your $\bar{Q}$ signal will go high.

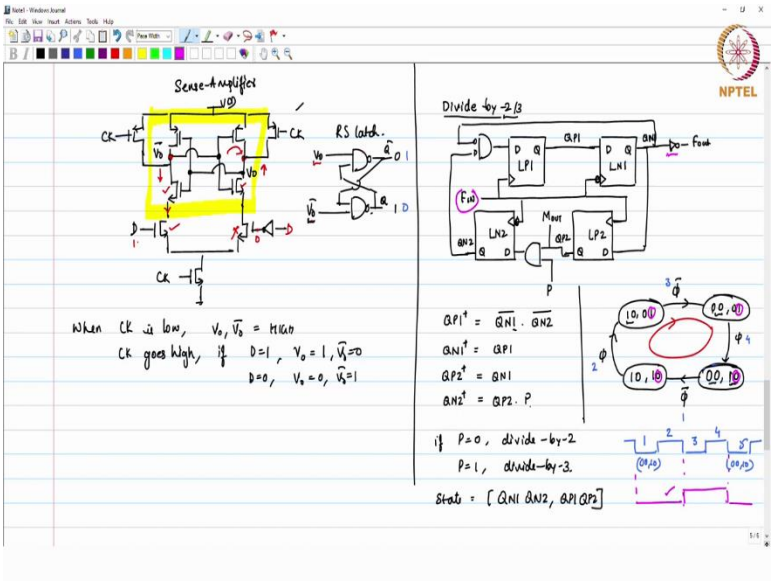
So, what you are doing here is you are sampling the input at the rising edge of the clock and here you are actually keeping this as $\bar{Q}$. Now, in order to have a D flip flop based clock divider divide-by-2, you can very well connect it in this manner. This becomes a divide-by-2 circuit with this feedback, you do not have any D here. If you want to cascade this, you can very well cascade this.

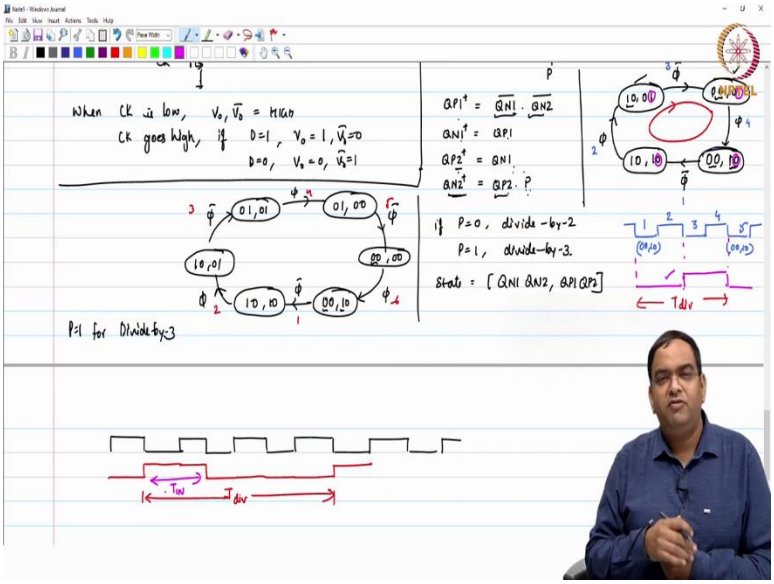
This kind of frequency divider is used at the highest frequency because you see that you are using like a minimum number of transistors to divide the clock by 2. The problem with this logic is as the frequency of operation is reduced, these node voltages which are actually held at 1 or 0, they are not connected to VDD or ground all the time with a direct path.

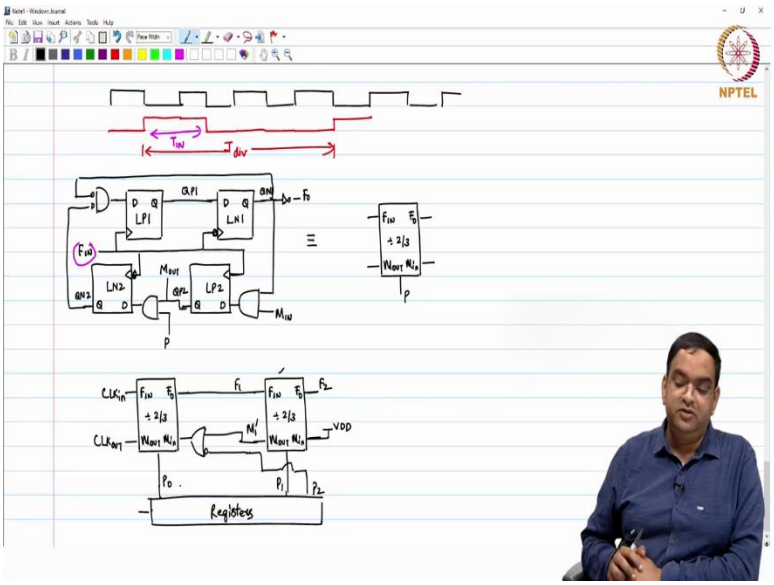
So, these nodes will start discharging or you have sub-threshold current for a long off duration or long on duration for different transistors. Due to the sub-threshold current, the leakage will be there and then you will have a different value at these internal nodes. So, normally we do not use

these TSPC based flip flops for frequencies lesser than 100 MHz, but this is very much technology dependent. For lower frequencies, it is not advisable. The reason is because the nodes discharge over a long on or off duration.

(Refer Slide Time: 12:14)







The other flip flop which we talked about is the D flip flop which is strong-arm latch based flip flop. So, strong-arm latch based flip flop, it has a sense amplifier as I will show you here, so this is strong-arm latch. So, you have a cross-coupled inverter connected in this manner. This connects here and this one connects here. You have an input pair and a clock. So, if I am going to give D here, there is an inverter here, I give here $\overline{D}$. It is like a differential input, you have clock here and you have a reset thing at the output which is also with the help of a clock only.

This is connected to VDD. The output of this you can call it as V_o and $\overline{V_o}$. V_o and $\overline{V_o}$ they actually go to an RS latch. So, you have a NAND based, this is the RS latch which we use. You can have a different method of implementation but the operation should be similar. These are R and S, this is Q and $\bar{Q}$. So, here R and S is V_o and $\overline{V_o}$. Now, you look at it, the strong arm, so, let us call, this one is normally called as sense amplifier, this complete is a strong arm latch, sense amplifier and this is RS latch.

So, when your clock is low, clock is low, your V_o and $\overline{V_o}$ both these nodes are held high, they are reset to VDD. When you have this particular, these things reset to VDD, your NAND gates are not responding to anything, whatever values you have at Q and $\bar{Q}$, that will be there. Now, when clock goes high, it depends on what was the value at D. If previously D was 1 or logic high, then you have $\overline{D}$ which is coming here as 0.

So, when $D = 1$ and this node is 0, at that time what is going to happen is these two transistors get active, this is 1, so, it will start conducting, this is off. When this conducts, your $\overline{V_o}$ value will go down, your $\overline{V_o}$, as your $\overline{V_o}$ value goes down, with the PMOS transistor this value is held high. So, as CK goes high, if your $D = 1$, $V_o = 1$ or VDD and $\overline{V_o} = 0$. As soon as your $\overline{V_o}$ is actually equal to 0, what happens here is your Q, this if $\overline{V_o}$ goes to 0, this output, NAND gate output, this output goes to 1 and this output will go to, if this output goes to 1, then what will happen is you have 1 and 1, this output will go to 0.

So, you can, let us say this is Q and this is $\bar{Q}$ in our case. Similarly, if D was 0, at that time, $V_o = 0$, $\overline{V_o} = 1$ and if $V_o = 0$, in that case, $\bar{Q} = 1$ and $Q = 0$. So, it is changing the output Q and $\bar{Q}$ only during the rising edge of the clock. It holds that value after that time, no more changes will be taken into account because this cross-coupled latch holds the value. It is a regenerative latch, it holds the value to 1 and 0 unless it is reset and brought to the same level and then again it will decide.

So, these are the two flip flops which are commonly used. For the highest frequency with lower power consumption, we will prefer TSPC. As the frequency goes lower, then we will prefer this. You do not have any problem here with any node for high or low frequency. In this particular case, you have direct connection to VDD or ground for the outputs. So, these are the two flip flops which we use. What we have seen so far is a divide-by-2 or 2^N.

Quite often you need frequency division not only in 2^N, you need frequency division like 4, 5, 6, 7 or something kind of that. So, those things can be realized by using divide-by-2/3. Divide-by-2/3 is a basic circuit and based on this basic circuit, you can cascade such circuits to realize different frequency division value. So, let us look at this basic divide-by-2/3. There is a lot of literature which you can read how to cascade this divide-by-2/3 circuits and how the frequency will change.

So, we will look at basic divide-by-2/3 circuit. So, here you use latches like this. You will have D, Q and a clock gating circuit. So, let me just make this as a clock here, this, this and this. So, there are four such latches. Similarly, you have another latch, D, Q and clock and this will come at the output like this. There is nothing like $\bar{Q}$ here and two of these latches are actually clocked at rising edge of the clock and the other two latches are clocked at falling edge of the clock.

So, these two are at the falling edge of the clock and the other two are at rising edge. So, I will just give the clock input here. Then the output of the one which is clocked at the rising edge, I call that as LP1 and LP2, and the one which is at the falling edge, I call that as LN1 and LN2. This is input frequency, the output of these latches are $QP1$ and $QN1$, this one is $QN2$ and this one here will be $QP2$. The output of this connects here and then you have a NAND gate which connects it like this.

And you have this particular one coming from this side. To control the divide-by-2/3, we use a NAND gate. We will see how the division happens by 2/3 by the way, just hold for a minute. So, you have one connection coming from here and another connection is external connection P. This $QP2$ is also quite often written as M_{OUT}, we will see what it does. So, now, with this, in the presence of these latches, this is like a state machine which you will see. So, we have,

$$QP1^{+} = \overline{QN1}.\overline{QN2}$$

$$QN1^{+} = QP1$$

$$QP2^{+} = QN1$$

$$QN2^{+} = QP2.P$$

If $P = 1$, then it will have, otherwise it is going to be 0. Now, in this case, you will have, if $P = 0$, then the output frequency you can take at any node, these are all having the same frequency at the output.

So, if you want you can take this as F_{out} or even M_{OUT} signal or $QP2$ signal is also going to have the same frequency. Then here if $P = 0$, you divide by 2. If $P = 1$, you divide by 3. How is it going to do? Let us look at the state machine for this. So, to begin with, let us choose these states, let me define the state first. So, state is $[QN1\ QN2, QP1\ QP2]$, these are the states.

So, we start with a state, let us say $QN1$ is 0 which is here and $QN2$ is also 0. So, 0, 0, if $QN1$ this is just like initial state, you can also fix that in the beginning. $QN1$ is 0, $QN2$ is 0, which is going to say that $QP1$ can be 1 and $QP2$ is going to be 0. So, [00, 10] is the initial state. From here, you are going to look at when you get the falling edge on the clock. So, when you get the falling edge on the clock, I am going to denote that as $\bar{\varphi}$, φ is the clock, you get falling edge on the clock.

So, based on the previous state, at the falling edge on the clock, only your $QN1$ and $QN2$ will change, $QP1$ and $QP2$ remain same. So, this remains same, you have 10. $QN1^+ = QP1$ which is 1 and $QN2^+ = QP2.P$. P is always 0. So, this will remain at 0 at $\bar{\varphi}$. Then in the next time when you get, at the falling edge this happens, then at the rising edge which is φ your $QN1$ $QN2$ remain same, it is the time for $QP1$. So, what you see is $QP1^+ = \overline{QN1}.\overline{QN2}$, so, it will remain 0, this will become 0 and $QP2^+ = QN1$ which is 1.

Then you get another falling edge in the clock cycle, so, when you have falling edge, your $QP1$ and $QP2$ remain same. $QN1^+ = QP1$, $QP1$ is actually 0, so this is 0 and $QN2$, it remains 0. Then you get a rising edge on the clock and rising edge on the clock you know that this state will be retained and what happens to, $QP1^+ = \overline{QN1}.\overline{QN2}$, so this becomes 1 and then you have $QP2^+ = QN1$ which is also 0.

So, this cycle actually repeats here. If you look at this cycle, what you are going to see here is this is happening at the falling edge and rising edge. With every falling and rising edge, you count half a clock period. So, 1, 2, 3, 4. After 4 half clock periods, the state repeats here. So, what it means is that if you have a clock like this, you can start from any state here, if you have a clock here, so on the first falling edge 1, then rising edge 2, then falling edge 3, then rising edge 4.

So, after this the state repeats. So, if it is [00,10] here, it is going to come [00,10] state in the 5th. So, this is how you are, so after every two clock periods, the output is repeating. Now, the other thing which you can see here is whether you take $QP2$. So, $QP2$ here, it is 0, 0, then it becomes 1 and 1 which means that if I look at the digital signal at $QP2$, $QP2$ is 0 for two of these consecutive and then it is 1.

So, this is, this particular signal is half the frequency of your F_{IN}. And if you want to get this divide by, you want to have, you can invert it at $QN1$ and you can take this as F_{out} if you are just doing divide-by-2. Now, what happens in divide-by-3, it is better that I do it here itself because the state diagram is here. So, I am going to start with the state let us say start with the same state [00, 10].

You get a falling edge on the clock, when you get falling edge on the clock, your $QN2$ and $QN1$ only change. So, I will, $QP1$ and $QP2$ will not change. So, $QN1$, this remains 10, $QN1^+ = QP1$ which is 1 and $QN2^+ = QP2.P$. Here $P = 1$ for divide-by-3. So, $P = 1$, so, this is 0, you go to

[10,10]. Then you get a rising edge here, you will get the same state because P only affects the falling edge, it does not affect the rising edge.

So, I will get [10,01], same here. Then you get the falling edge. So, falling edge comes, $QP1$ remains same, $QP1$ $QP2$ remain same, $QN1$ and $QN2$. So, now, you see $QN1^{+} = QP1$ which is 0 and $QN2^{+} = QP2.P.\ QP2 = 1$ here, $P = 1$, so, this becomes [01,01]. So, your fourth state is different from here. Then you get another rising edge of the clock. So, rising edge of the clock $QN1$ state remains same, $QP1^{+} = \overline{QN1}.\overline{QN2}$, so this is also going to be 0 and $QP2^{+} = QN1$ which is 0.

Then you get falling edge on the clock, when you get falling edge on the clock, what you see here is this remains 00, $QN2^{+} = QP2.P$, so you get 00. Then you get rising edge of the clock, this remains 00. And your $QP2^{+} = QN1$ and $QP1^{+} = \overline{QN1}.\overline{QN2}$. So, if you look at it, we count how many half clock cycles, 1, 2, 3, 4, 5, 6. So, after six half clock cycles, these signals actually repeat.

So, when they repeat, you know that if it is repeating after 6 half clock cycles, then it is like repeating after 3 actual clock cycles, every φ and $\bar{\varphi}$ is your half clock period here. So, if I have my clock cycle like this. Let us start to look at $QP2$ here which is M_{OUT}. So, $QP2 = 1$, falling edge, I should start with the falling edge, 1, so $QP2 = 1$, it remains 1, then it becomes 0. Then it remains 0, it is 0 here and it is 0 here. So, 1, 2, 3, 4. So, for four of this, it remains 0, then it again goes to 1.

So, this is the period of the divided clock. Previously, this is the period of the divided clock. So, what you see here is that the divided clock is you can do a division by 2 or division by 3 and the divided clock can be any of these signals for division by 2 and division by 3 circuits. Then the important thing here is that you do not have a 50% duty cycle, this is only your T_{IN}.

So, the duty cycle of the divided clock here for divide-by-3 is not 50%. Quite often it is okay not to have it at 50%. It will work fine if things only depend on the rising edge of the clock. So, now if we want to have any division ratio, what you can do is you can cascade these divide-by-2/3 circuits. There is a NAND gate which is going to, this is M_{IN}. Now, you will divide by 3 only when $M_{IN} = 1$ and $P = 1$.

You will not divide by 3 otherwise. This is needed when you cascade these blocks. So, I will just, this block is equivalent, so, well, you have this also, this is F_{out} here, you have F_{IN}, F_{out}, this is divide-by-2/3, you have M_{OUT} and M_{IN}, these are also the inputs. So, you cascade this block. So, I am going to cascade these two blocks, this is just an example, and the output from here, it goes to this and this is something you get with a NOR gate.

Your M_{OUT}, first it goes here, this I call this as M_1'. This M_{IN} is always connected to VDD. So, that means whatever you want divide-by-2/3 in this circuit, it is you can get it, this is F_1, F_2. You can get finally divided clock at the same flip flop at where you are feeding the inputs. And then what you have here is this P_0. You have P also, so I will just write this as P.

So, this is P_0 and then you have here P_1 and this particular signal is going to add as P_2. So, based on this P_0, P_1 and P_2, you can load this using some registers. So, based on this P_0, P_1 and P_2, you can get different values of the division like 4, 5, 6, 7 in this case. This is an example, you can work it out that how this is going to give you divide by 4, 5, 6, or 7. So, when you want a larger division ratio and such kind of numbers, you can actually cascade such blocks, load these control signals P_0, P_1 and so on and you can get the division ratio which you like.

So, that brings us to an end of the discussion on the building blocks of PLL. And next we are going to see what the other problems with the PLL are or what we can do to improve one or the other performance metric of the PLL as a whole. Thank you.