Phase-Locked Loops

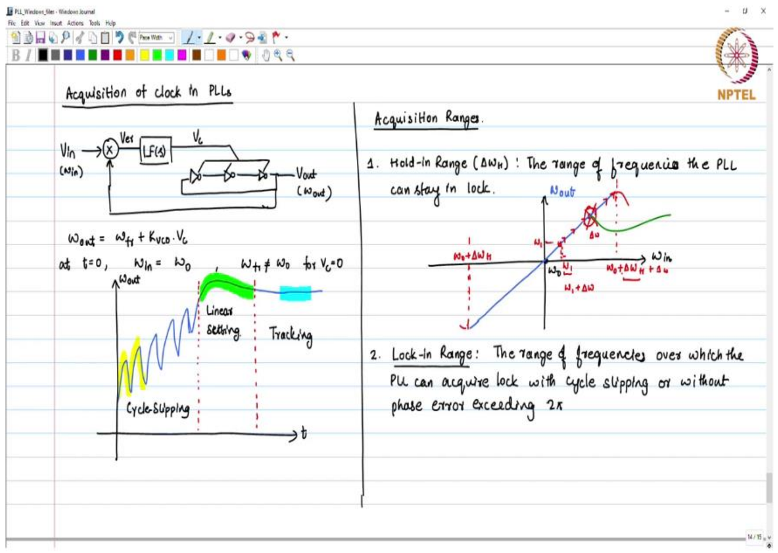
Dr. Saurabh Saxena
Department of Electrical Engineering
Indian Institute of Technology, Madras

Lecture – 10
Frequency Acquisition Range for PLLs

In the previous session, we have seen that how the PLL locks and the steady state phase error in the presence of phase step, frequency step and frequency ramp. But all that analysis was while using the small signal model of the PLL and we have also looked at the PLL and said that there is some hard limit on the error voltages because of the phase error detector which we are using.

In this session, we will look at the actual transient signals in the PLL when we have any phase or frequency step. The one which we have seen before was with the small signal analysis, now we will see how actually the transient signals will change, and what are the limits for different kinds of PLLs.

(Refer Slide Time: 1:21)



Acquisition of clock in PLLs

$\omega_{out} = \omega_{fr} + K_{vco} \cdot V_c$

at $t=0$, $\omega_{in} = \omega_0$, $\omega_{fr} \neq \omega_0$ for $V_c = 0$

Acquisition Ranges.

1. Hold-in Range ($\Delta\omega_H$) : The range of frequencies the PLL can stay in lock.

2. Lock-in Range: The range of frequencies over which the PLL can acquire lock with cycle slipping or without phase error exceeding 2π

We are going to study about the acquisition of clock in PLLs. So far, the example which we are using is a simple implementation where I have a mixer based phase error detector which gives me error voltage followed by the loop filter which can change depending on the type and order which

I want, and this controls the VCO, this is just a symbol you can think about right now. So, the frequency is controlled for this VCO and this is the output, V_{in}, V_{out}, this is V_c.

So, what happens when you design this PLL, we know that,

$$\omega_{out} = \omega_{fr} + K_{VCO}.V_c$$

$$\text{At } t = 0,\ \omega_{in} = \omega_0,\ \omega_{fr} \neq \omega_0 \text{ for } V_c = 0$$

So, the two frequencies, namely, the free running frequency and the input frequency are different. Here, the control voltage is equal to zero. So, the control voltage and the error voltage are initially equal to zero at time instant $t = 0$. You have a frequency error to begin with.

And now you want to understand whether the PLL is going to acquire the lock, which means whether the output frequency is going to be equal to the input frequency or not and what is going to happen during the due course of acquisition. So, typically when you go and design this PLL and you happen to observe the output frequency over a period of time, so I am just going to plot one kind of frequency acquisition which you may see, you may start with ω_{fr}, you may have some kind of frequencies doing like this and then for some time you may see something like this and after some time it will be very smooth. This is a typical behavior of a PLL acquiring the output frequency when you start your PLL.

In this particular case, we will just kind of vaguely divide this area of frequency acquisition into three parts. In this part what you see here is, I will show this with the simulations also, but right now, in this part what you are seeing here is that the frequency increases and then decreases, increases and then decreases. Why will it happen if I want to acquire a certain output frequency? Will it not be nice that it goes only in one direction and tracks it? That would be a nice thing but that does not happen, depending on the frequency error which you have.

So, if that does not happen, you see such kind of frequency increase and decrease and what we see here is that we have a phenomenon called cycle slipping. We will see this cycle slipping that how it happens. In this case, what you are seeing here is a kind of a linear settling of the output frequency. You would have seen linear settling for any first order or second order transfer function. You apply any small change and for that small change, what you observe is that there is some kind of peaking and then it settles. In the final phase, what you observe is that the thing has actually settled to the desired output frequency, it is not changing, if you make any small change to the system, it will either bring back or it will track the input changes. So, at a very broad level, this is what you have during frequency acquisition. We will understand that how these things happen and what we expect when we start our PLL from the frequency point of view.

Now, because so many things happen in the PLL depending on the initial frequency error which you see, the frequency acquisition of the PLL can happen in different ways as you see here, so what we do is we define different kind of acquisition ranges, that how you are acquiring the output frequency, based on that we define the acquisition ranges. The first acquisition range is defined as hold-in range. I will explain what it means. So, hold-in range is the range of frequencies the PLL can stay in lock, normally defined with the symbol $\Delta\omega_H$. This is the range of frequencies the PLL can stay in lock.

What does it mean? What it means is the following. So, let us say this is the ω_{in} frequency, it is not zero, you will just treat this point as ω_0. So, I have a PLL and I have the input frequency which is ω_0, if I vary ω_0 by a small amount, so this is ω_{out}, there is a difference between ω_{out} and ω_0, as I vary the input frequency, the output frequency will vary, whether it has the free running frequency same as the input or not but the output frequency will follow the input frequency, that range is going to be defined by hold-in range.

So here, on this ω_{in}, if you pick up any point, so I am writing this as $\omega_0 + \Delta\omega_H$, and this is $\omega_0 - \Delta\omega_H$, you pick up any point for the input frequency when the PLL is locked, how it reached there, that is not of concern right now, the concern is if I have the input and output frequencies of the PLL, if I pick up any point on this at which the input and output frequencies are same, if you make any change at this point, if you apply any disturbance to the PLL, the PLL will remain in lock, it will not lose the lock.

So, I will just give one example here. So, in this example, let us say I have this frequency $\omega_{in} = \omega_1$, so, ω_{out} will also be equal to ω_1. If that is the case, if I apply a small change in the phase for the PLL, applying a change in the phase is like applying a disturbance to the PLL, if I apply disturbance to the PLL in terms of phase change, not in terms of frequency change, the PLL will remain in lock, it will not lose lock. Second, if I apply a small change in ω_1, let us say, I made a small change in the input frequency as $\omega_1 + \Delta\omega$, the output frequency will track the input frequency, you apply a small change in the input frequency, you give the PLL enough time, the output frequency will follow the input frequency. So, till the point where I can go by applying small changes to the input frequency such that I will be able to use the small signal diagram for the PLL, till that time I can keep changing my input frequency, till that particular output frequency, you can track. If at $\omega_0 + \Delta\omega_H$, I apply $+\Delta\omega$, at that point, the output frequency does not follow the input frequency, the PLL loses lock. So, the extreme limit on either side which you can track by changing the input frequencies, that frequency range comes under hold-in range.

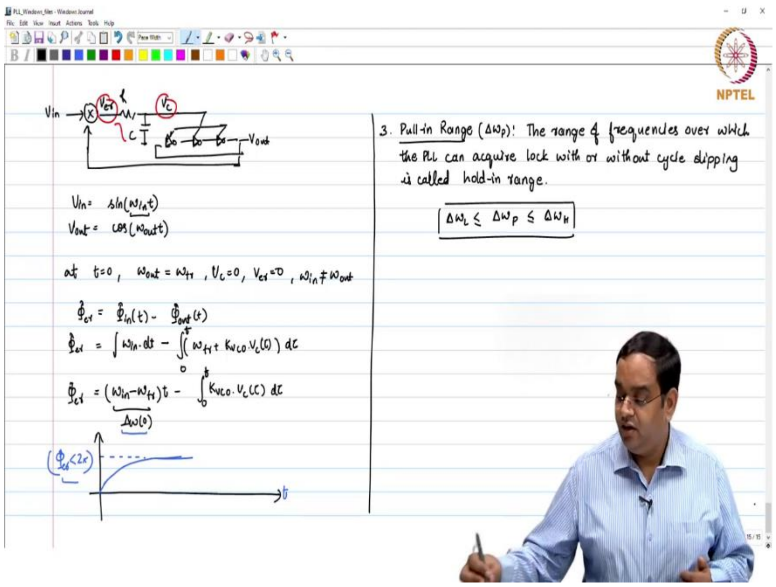
So, point #1, that ω_{out} should follow ω_{in} when you are applying small changes in the input frequency, ω_{out} keeps following ω_{in}. After a point it will happen that if you make a small positive change in ω_{in} on the positive side or a small negative change on the negative side, the PLL loses lock, that limit is the hold-in range. This is one way to understand. The other way to understand is, see here we are not talking about the free running frequency. The free running frequency can be anything, we are looking at ω_{out} as following ω_{in}.

The other way to look at it is, you pick up any point on this input frequency span and at that particular point, ω_{out} is also equal to ω_{in}, how it reached there, that is not a concern right now, that does not matter actually, so ω_{out} reaches ω_{in} and then you apply some disturbance at that point, some phase disturbance or control voltage or error disturbance, whatever it is but it should be a small disturbance, not like a big disturbance, so that you can use your small signal analysis, changes are not big, the PLL holds the lock there, it does not lose its lock.

So, for example, if I pick up this point with the one big cross here, and I apply a small change of $\Delta\omega$, it is not going to happen that the PLL loses lock which means that the output frequency can be anywhere, it does not follow the input frequency, that will not happen if that particular input frequency is within the hold-in range. So, you can think that the hold-in range is a very wide frequency range over which the PLL can hold its lock, even in the presence of small disturbances.

The second acquisition range is defined as lock-in range. So, what is this lock-in range? Well, the range of frequencies over which the PLL can acquire lock without cycle slipping or without phase error exceeding 2π. Now, it is interesting to see that how this happens.

(Refer Slide Time: 15:51)



$V_{in} = \sin(\omega_{in} t)$

$V_{out} = \cos(\omega_{out} t)$

at $t=0$, $\omega_{out} = \omega_{fr}$, $V_c = 0$, $V_{er} = 0$, $\omega_{in} \neq \omega_{out}$

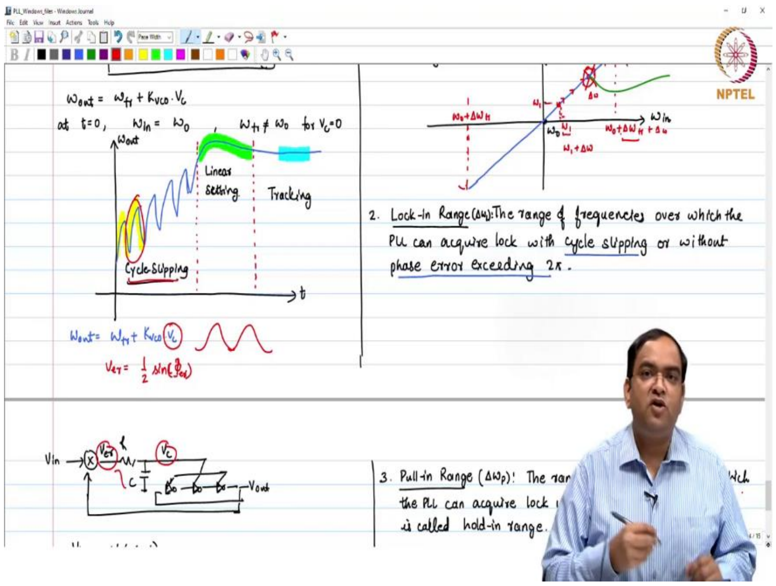
$\Phi_{er} = \Phi_{in}(t) - \Phi_{out}(t)$

$\Phi_{er} = \int \omega_{in} \cdot dt - \int_0^t (\omega_{fr} + K_{vco} \cdot V_c(\tau))\, d\tau$

$\Phi_{er} = \underbrace{(\omega_{in} - \omega_{fr})}_{\Delta\omega(0)} t - \int_0^t K_{vco} \cdot V_c(\tau)\, d\tau$

3. Pull-in Range ($\Delta\omega_P$): The range of frequencies over which the PLL can acquire lock with or without cycle slipping is called hold-in range.

$\boxed{\Delta\omega_L \leq \Delta\omega_P \leq \Delta\omega_H}$



$\omega_{out} = \omega_{fr} + K_{vco} \cdot V_c$

at $t=0$, $\omega_{in} = \omega_0$, $\omega_{fr} \neq \omega_0$ for $V_c = 0$

$\omega_{out} = \omega_{fr} + K_{vco} V_c$

$V_{er} = \frac{1}{2} \sin(\Phi_{er})$

2. Lock-in Range ($\Delta\omega_L$): The range of frequencies over which the PLL can acquire lock with cycle slipping or without phase error exceeding 2π.

3. Pull-in Range ($\Delta\omega_P$): The ran… …hich the PLL can acquire lock … is called hold-in range.

So here, what we need to understand is that what we mean by phase error exceeding 2π. So, these two terms, namely, cycle slipping and phase error exceeding 2π, mean the same. So, we want the phase error should not increase more than 2π, so what it means is, let us see with an example. So, I will pick up an example for the same mixer and loop filter based PLL. So, recall this, so you have a simple loop filter and a VCO, this was the PLL which you saw, so, R, C, V_{er}, V_c and this is V_{out}, this is V_{in}.

Now, in this particular example, we have,

$$V_{in} = \sin(\omega_{in} t)$$

$$V_{out} = \cos(\omega_{out} t)$$

I just chose the amplitude as 1. So, let us say,

$$\text{At } t = 0,\ \omega_{out} = \omega_{fr},\ V_c = 0,\ V_{er} = 0, \text{ and } \omega_{in} \neq \omega_{out}$$

So, there is a frequency error. So, if there is a frequency error, there is a phase error also, we know that. What is phase error? The phase error is given by,

$$\varphi_{er} = \varphi_{in}(t) - \varphi_{out}(t)$$

$$\varphi_{er} = \int \omega_{in}.dt - \int_0^t \left(\omega_{fr} + K_{VCO}.V_c(\tau)\right) d\tau$$

$$\varphi_{er} = (\omega_{in} - \omega_{fr})t - \int_0^t K_{VCO}.V_c(\tau)\, d\tau$$

So, this is the phase error. Now, what we are talking about is the following, that when you start your PLL and you have a frequency error which is $\omega_{in} - \omega_{fr} = \Delta\omega(0)$ to begin with, the PLL acquires the lock with $\omega_{out} = \omega_{in}$ without the phase error exceeding 2π.

So, this is a closed loop transfer function, we do not know what is going to happen, how it is going to change. Well, to begin with, both the signals are aligned, so you may start with some phase error and if finally, you attain a certain value of the phase error and then the phase error remains constant, the PLL acquires lock, good. If the PLL does not acquire lock, then what you will see is that the phase error will exceed 2π. So, this limit on the phase error with respect to time is actually less than 2π. If, from the beginning, from the time instant $t = 0$, as the input frequency is applied and the free running frequency is not the same as the input frequency, you begin with a frequency error, and if you acquire the lock without the phase error exceeding 2π during the frequency acquisition, then the upper limit on that frequency error is called as lock-in range. So, what is the maximum value of $\Delta\omega(0)$ which you can have in the beginning over which the PLL can acquire lock without this phase error exceeding 2π, that is called as lock-in range.

Next is the pull-in range. So, you will be interested in knowing that if the phase error exceeds 2π, what is the problem or if the phase error exceeds 2π, can it even acquire lock? Well, yes, it is possible. So, by the way, lock-in range is normally written with $\Delta\omega_L$ and pull-in range is normally defined with $\Delta\omega_P$. So, pull-in range is the range of frequencies over which the PLL can acquire the lock with or without cycle slipping.

Now, when will this happen, that is something which we can understand only with the help of an example. We will surely look at it. So, sometimes, it may happen that the phase error exceeds 2π and if the phase error exceeds 2π, the PLL may still acquire the lock. Now, just think about it, this is cycle slipping, now you may understand what cycle slipping is. During cycle slipping, the phase error is exceeding 2π, as you see here.

Now, just do some back calculation. I know that,

$$\omega_{out} = \omega_{fr} + K_{VCO}.V_c$$

If ω_{out} is having a ripple as shown to you here, that means this ripple can only come with the control voltage, so the control voltage is also having some kind of ripple which is like this and as you see, it is having peaks and nulls. So, the control voltage is having peaks and troughs, so you have a maximum value, you have a minimum value, that is what you have with the control voltage. Now, control voltage in the Type-I PLL is coming as just a filtered version of the error voltage. So, the low frequency signal is passed. So, if control voltage is having peaks and troughs like this, then your error voltage will also have peaks and troughs. The error voltage can have such kind of periodic peaks and troughs only when the phase error changes 2π, because error voltage is nothing but $V_{er} = \frac{1}{2}\sin(\varphi_{er})$ at any given time. So, if you are seeing periodic waves in case of error voltage, that means the phase error is spanning through 0 to 2π and that is how this particular acquisition happens with cycle slipping.

So, whether it happens with cycle slipping or without cycle slipping, whatever is the method, that comes under pull-in range. Hold-in range is one step further, it says, I do not care about how you reach there, but if you are there and you can hold the lock, that means even for small disturbances, it is not going to make the PLL to lose lock, so that is hold-in range. So, if you compare these ranges, what you will find is the following:

$$\Delta\omega_L \leq \Delta\omega_P \leq \Delta\omega_H$$

So, next, we will see that how this pull-in happens in an actual circuit in transient domain and how this hold-in also works, so that is what we discuss about the different ranges in the PLLs. Thank you.