

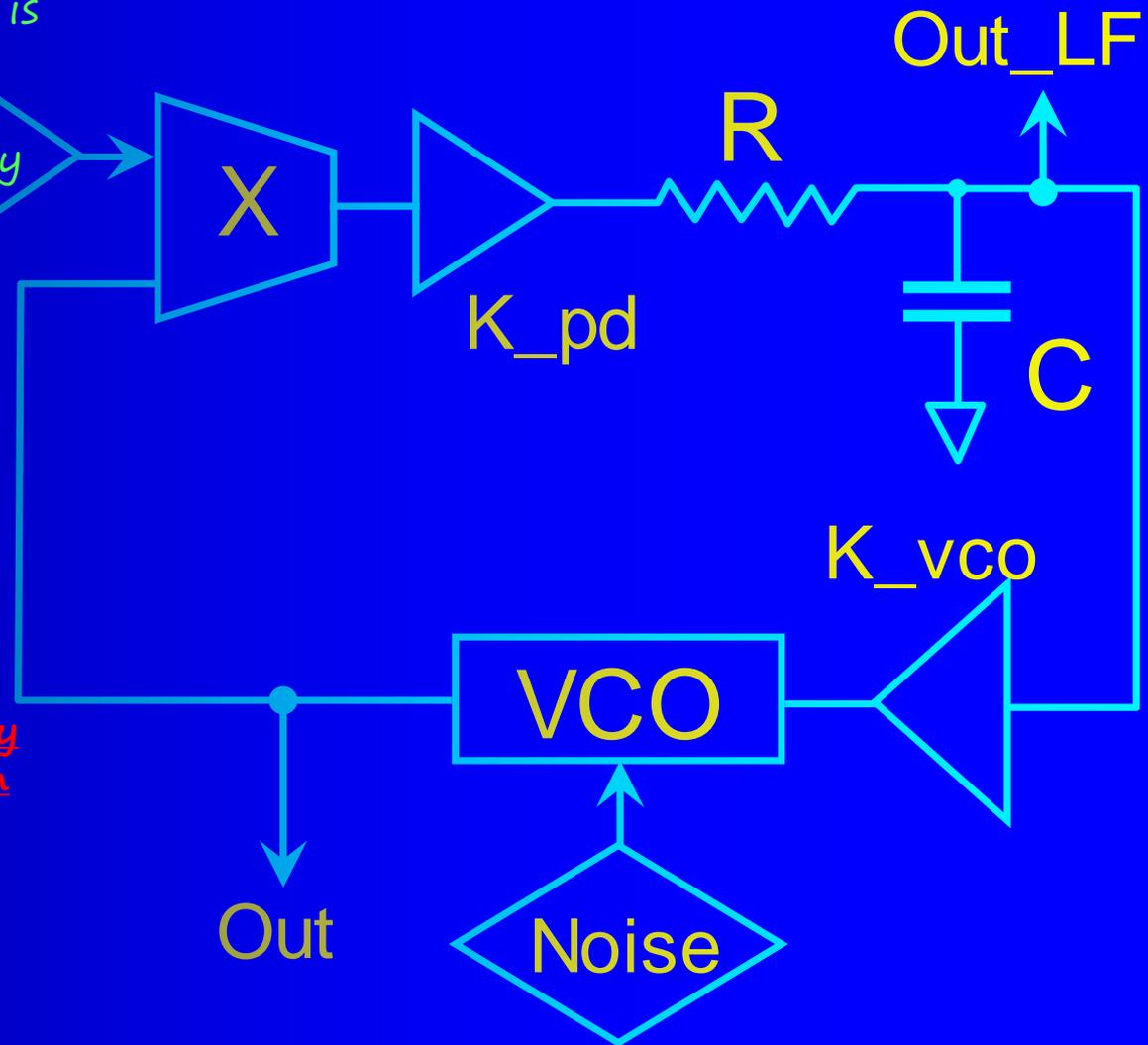
How to Model a PLL in Excel—part #1 – a quick and dirty, cook-book style recipe

A phase locked loop (or a PLL) is an electronic circuit which generates a oscillatory output synchronized with an oscillatory input. Just in telecommunications the PLL applications are very wide:

- signal synchronization
- signal recovery
- frequency synthesis
- demodulation

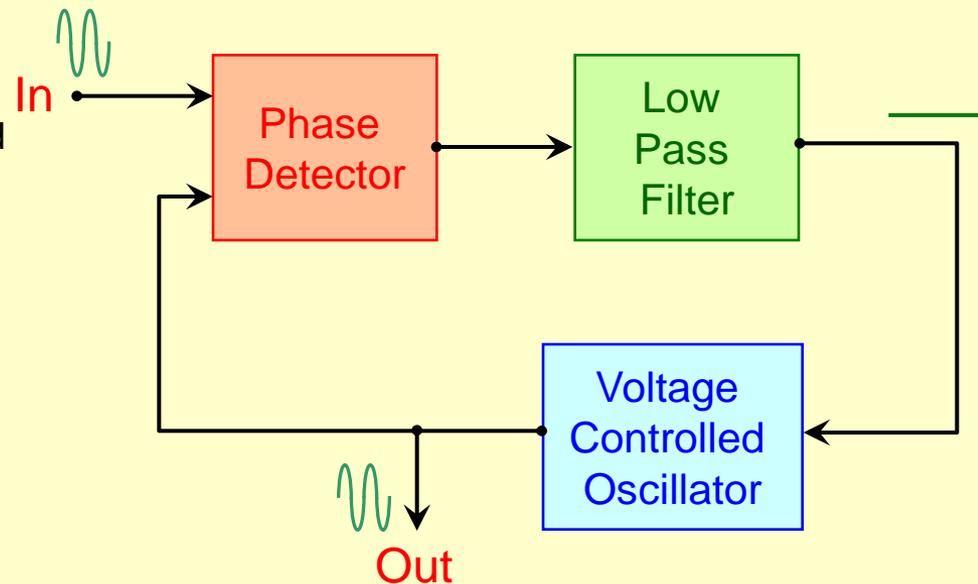
The first two presentations will NOT explain the PLL functionality in much detail, it will just explain the numerical method and show you how to implement your own fast and dynamic PLL model in minimum time and effort.

excelunusual.com



A basic PLL block diagram:

The general principle of a PLL is easy to understand:
- if the free running frequency of the voltage controlled oscillator is not too far from the input frequency any phase difference will be sensed by the phase detector, filtered and sent back to control the VCO frequency until the input and the output have exactly the same frequency and there is a constant phase difference between them.



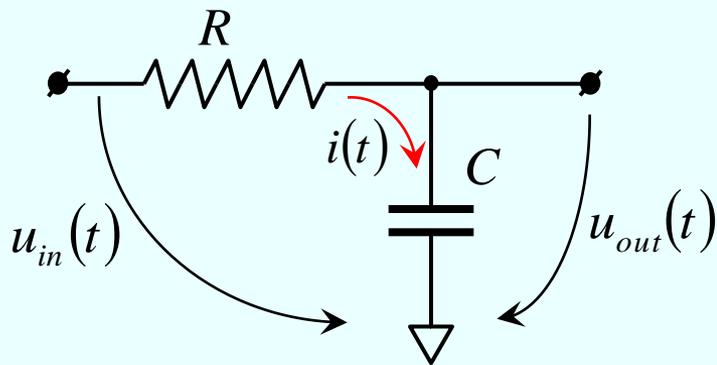
The phase detector:

- Most of the written materials out there say that the phase detector is nothing more than an analog multiplier or a XOR gate in the digital case. In this presentation we will not try to understand why. We will come back for more insight into this topic in future presentations. For now we will just use a plain multiplication to model a phase detector in Excel.

The low pass filter (LPF) a basic difference derivation:

- This is a little bit more difficult to model so you need to pay attention. If you finally understand the derivation of the numerical model you will be able to do much more complicated things by yourself.

Let's not worry about filter theory and just pick a resistor and a capacitor and improvise a basic low pass filter. Why is that circuit a LPF? Because for very low frequencies the capacitor has a large impedance and the signal goes from input to output unchanged. At very high frequencies the capacitor is a short, all the input signal is grounded and the output is zero. For this circuit we can write (these formulas are high school level):



From Ohm's law: $R = \frac{u_{in}(t) - u_{out}(t)}{i(t)}$

From the definition of current intensity: $C = \frac{dq(t)}{du_{out}(t)}$

From the definition of capacitance: $i(t) = \frac{dq(t)}{dt}$

} =>

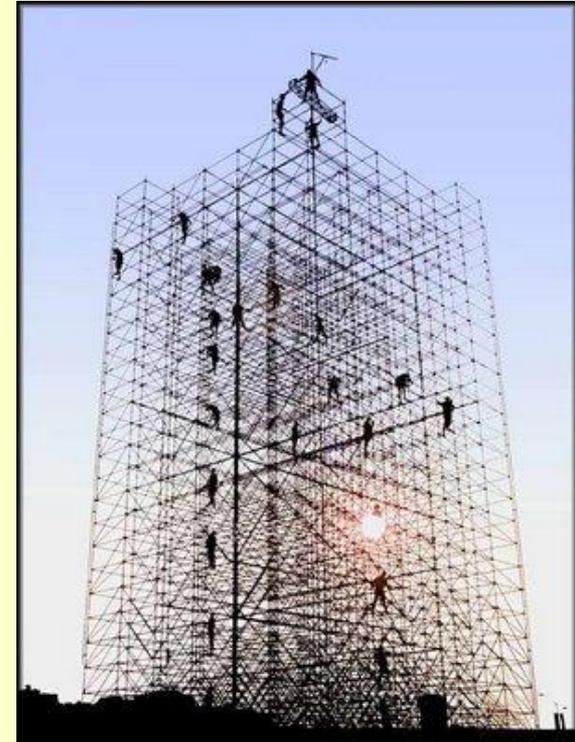
Eliminate the current intensity: $\Rightarrow \left\{ \begin{array}{l} R = \frac{u_{in}(t) - u_{out}(t)}{\frac{dq(t)}{dt}} \\ C = \frac{dq(t)}{du_{out}(t)} \end{array} \right. \Rightarrow \left\{ \begin{array}{l} R \cdot dq(t) = [u_{in}(t) - u_{out}(t)] \cdot dt \\ C \cdot du_{out}(t) = dq(t) \end{array} \right. \Rightarrow$

$\Rightarrow R \cdot C \cdot du_{out}(t) = [u_{in}(t) - u_{out}(t)] \cdot dt$

Don't tell me you had any difficulties with the derivations on this page. If you did you must go back to the high school physics book of problems and solve 100 of them.

Apply the previous formula on an elementary time step (dt):

- Numerical methods are very dumb but very efficient ways of solving very difficult problems. We essentially calculate what happen on a very short time interval. Numerical methods are not exact methods, there are approximations and intrinsic errors but the errors can be minimized by choosing smaller time steps.
- Because the interval is so short, the laws of physics are simple and we can often use linear approximations.
- Another factor that makes the numerical methods easy to work with is the fact that we already know all the results from previous time steps and we can use them to calculate the result of the current time step. It is all a recurrent chain of calculations, just like the skyscraper builders who use no cranes but use the previously build structure to attach the scaffolding on.
- Another very important factor in facilitating the calculations is the fact that the derivatives (tangents) to curves of certain functions can be approximated as secants, using the current and the previous value of the function.



To the right there are three common derivative approximations which dramatically simplify calculations in numerical methods. Pay attention to colors.

Derivative definition:

$$\frac{df(t)}{dt} = \lim_{dt \rightarrow 0} \left[\frac{f(t+dt) - f(t)}{dt} \right]$$

Central estimate:

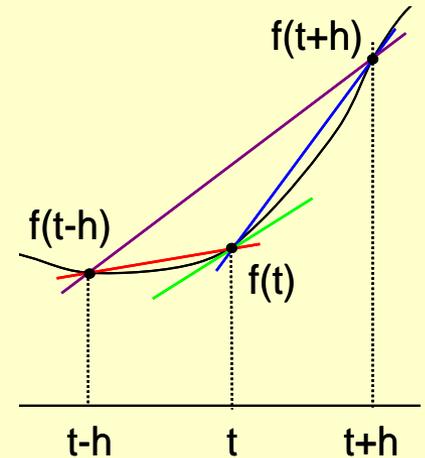
$$\frac{df(t)}{dt} \approx \frac{f(t+h) - f(t-h)}{2 \cdot h}$$

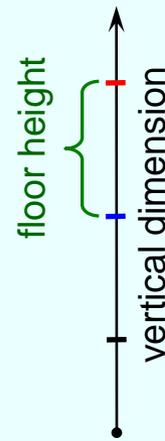
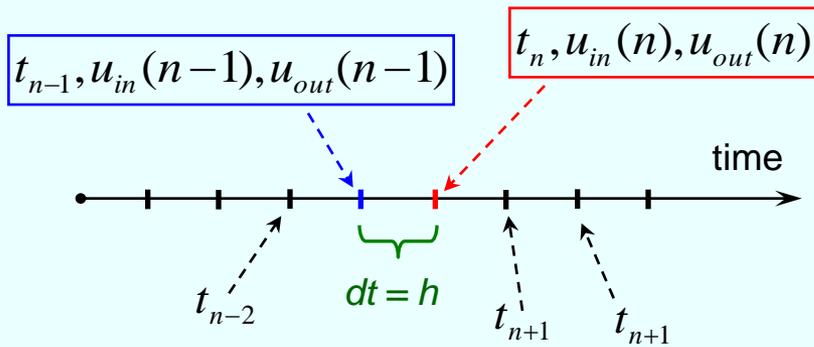
Forward estimate:

$$\frac{df(t)}{dt} \approx \frac{f(t+h) - f(t)}{h}$$

Backward estimate:

$$\frac{df(t)}{dt} \approx \frac{f(t) - f(t-h)}{h}$$





In order to make the numerical methods easier to understand I set up an analogy with a construction site. The analogy is somewhat idealistic. The construction methods described here are quite unorthodox, devised purposely for this presentation.

<u>Numerical Methods</u>	<u>An Original Construction Site</u>
Time domain – time step duration	Vertical dimension – floor height
Calculate one step at a time – allows for handy approximations and ease of calculations. The previous time steps are fully finished and their data can be displayed on a chart.	Build one floor at a time after completely finishing the floors underneath. The workers and the machines are concentrated on the current floor. The floors underneath are in use.
Formulas and calculations are simple. Any enthusiastic geek can program the methods instead of employing a highly paid mathematician using a high price software.	The workers have a solid and safe floor underneath. No need for hard to find “high iron” workers from Mohawk (part of Iroquois) tribe. Just about any worker will do the job since the skills to build within 10 feet of height are lower.
Calculations use only information from the previous 1-3 time steps.	Instead of using an expensive crane workers just use scaffolding anchored to the previous 2-3 floors
For large simulation times, the model could be paused, storing the last 2-3 steps and it can be restarted from the same point any time later, after a day, a month or a year.	The construction crew could take a long vacation and restart the work any time with no major consequences while the building underneath is in use.

$$R \cdot C \cdot du_{out}(t) = [u_{in}(t) - u_{out}(t)] \cdot dt$$

$$\begin{pmatrix} dt = h \\ t = n \\ t - dt = n - 1 \end{pmatrix}$$

The numerical equations to the left are equivalent and must give very close results. The former uses the backwards approximation of the derivative and the latter uses the forward approximation.



Let's see how by knowing everything about the previous time steps (up to time "n-1") we can translate the above equation into a numerical approximation equation with the purpose of finding the output voltage at the moment "n":

$$R \cdot C \cdot [u_{out}(n) - u_{out}(n-1)] = [u_{in}(n) - u_{out}(n)] \cdot h$$



$$R \cdot C \cdot [u_{out}(n) - u_{out}(n-1)] = [u_{in}(n-1) - u_{out}(n-1)] \cdot h$$

The numerical output formula by backward derivative estimate:

$$u_{out}(n) = \frac{h \cdot u_{in}(n) + R \cdot C \cdot u_{out}(n-1)}{R \cdot C + h}$$

The numerical output formula by forward derivative estimate:

$$u_{out}(n) = u_{out}(n-1) + \frac{h}{R \cdot C} \cdot [u_{in}(n-1) - u_{out}(n-1)]$$

- Both formulas lead to the same result. The shorter the time step the better the precision of either result. In the next section, before integrating the LPF formula into the PLL model, the two formulas will be tried on a spreadsheet example using rectangular periodic signal as input.

- Whenever you build a model break it in simple stages and don't proceed to the next stage until you fully test the current stage.