Simulation of a Phase Locked Loop Using LabVIEW.

T.J. Moir
Massey University at Albany
Institute of Information and Mathematical Sciences
Albany
Auckland
New Zealand
Email: t.j.moir@massey.ac.nz
Abstract

A method for simulating an analogue phase-locked loop (PLL) is shown. The programming language given is LabVIEW (trademark National Instruments) but the techniques used are quite general and apply equally well to other programming languages and packages. LabVIEW seems an unlikely candidate to simulate a PLL as it is more often associated with process control systems and virtual instrumentation rather than communication systems. However it will be shown that in fact LabVIEW gives a powerful solution which is both easy to implement and with a user graphical interface comparable to any modern programming language. The simulation is fully interactive and demodulates ordinary FM like a radio receiver.

Keywords:
Phase-Locked Loop (PLL), Feedback system, LabVIEW

1. Introduction

The Phase-Locked Loop (PLL) is one of the most commonly used integrated circuits (ICs) in use in modern communications systems[1]. Although perhaps surprisingly first invented as early as 1932 by Bellscie[2] it never gain popularity until the early 1970s when cheap ICs were readily available. It quickly found application as a precision FM demodulator as a replacement for Foster-Seely discriminators. Digital communication systems quickly followed and the PLL has found application in such areas as modems, mobile communications, satellite receivers and television electronics. The PLL is used extensively in modern electronic systems but its design is often met with trepidation. This is perhaps understandable since to fully understand the operation of a PLL requires some knowledge of communication systems and control systems, two subjects which are treated in isolation. For example seldom are PLLs covered in a taught control course at undergraduate level whilst feedback and stability is only briefly mentioned when covering PLLs in a communication course. The purpose of this paper is to show how a realistic PLL can be simulated. The simulation uses LabVIEW as it is easy to build a graphical interface which is interactive. Although such packages as MATLAB and MATRIXx[3] could equally well be used the result will not be as interactive. For example in [3] although the simulation is realistic it is of the ‘one-shot’ variety where the program must be re-run to show the effect of any changes and cannot be made to run continuously in pseudo ‘real-time’ like the approach used here. The final goal is a single PLL program which can then be applied to a whole range of possible applications such as demodulation of FM, Frequency Shift Keying (FSK) or to further investigate such problems such as multi-path and additive noise (thresholding).
2. The Basic PLL

The block diagram of a generic PLL is shown in Figure 1 below.

![Block Diagram of Generic PLL](image)

**Figure 1. Block Diagram of Generic PLL**

We consider an input to the PLL which is FM. Whilst there are other types of input which can be used, this approach gives the opportunity of modulating various different baseband signals which can be later used to test the PLL. For example a step input or a sinusoidal frequency response are commonly used for testing all feedback control systems. The FM signal which must be simulated is an FM modulated sine wave (although square waves and other waveforms are also considered) and has the analogue form \( f(t) \) where

\[
f(t) = \cos(\omega_c t + \beta \sin(\omega_m t))
\]

In the above \( \beta = \Delta \omega / \omega_m \) is defined to be the usual FM modulation index with \( \omega_c \) and \( \omega_m \) respectively the carrier and baseband frequencies in rad/s. The depth of modulation is given by \( \Delta \omega \) rad/s.

In Figure 1 the PLL comprises a phase detector (PD), a voltage-controlled oscillator (VCO) and a filter. The PLL will demodulate the FM and give an output which is the original baseband signal. The PLL would normally operate on the intermediate frequency (IF) waveform of a radio receiver.

The theory of the PLL is well documented [1,4] and only an outline will be given here. The operation is easiest to see when there is no FM and only a fixed carrier frequency is present (ie \( \Delta \omega = 0 \)). When in lock the output of the VCO will be a
waveform which is in phase-quadrature with the incoming waveform. The phase detector (PD) is a linear multiplier (for our example) and gives an output which is approximately zero with an additive term at twice the carrier frequency $2\omega_c$. When properly designed (i.e., the bandwidth is chosen appropriately) the term at $2\omega_c$ will be attenuated by the filter (apart from some residual left-over) and the PLL output will sit at zero. When a baseband signal is presented (i.e., FM) the VCO output will track the variation in phase of the incoming FM and the PLL output (the input to the VCO) will be the rate of change of phase (i.e., instantaneous frequency). The VCO thus acts dynamically as an integrator and this is important when examining the open-loop frequency response.

The beauty of the PLL is that it can be analysed in a linear form independent of the carrier frequency. The block diagram of the closed loop PLL is shown in Figure 2. The VCO transfer function $H(s)$ is shown as a pure integrator and the phase detector as a summing junction providing negative feedback. It remains to find the filter dynamics $F(s)$.

The filter dynamics are found by drawing the open-loop Bode plot which should have a similar form to the one shown in Figure 3. (although other forms are possible)
This type of PLL is sometimes known as a third order type II PLL as there are two integrators within the loop. The first integrator is the VCO and the second is an added electronic integrator. Since two integrators with negative feedback results in an oscillator, a phase lead (advance) stabilisation is needed. Hence the overall Bode plot has the form shown in Figure 3. This particular design is preferred as it has better tracking abilities than a type I PLL. The higher the gain at low frequencies results in good tracking and hence low error. (for the control system details see reference [5])

The unity gain bandwidth of the PLL should be chosen high enough to track adequately but not too high so as to let too much $2\omega_c$ through. By experience it has been found that a unity gain bandwidth of

$$f_\phi = \frac{2f_c}{10}$$

gives good results.

Figure 3. Open Loop PLL Bode Plot
3. LabVIEW Simulation

The graphical user interface of LabVIEW together with the block diagram approach makes it a good candidate for simulating a PLL. The simulation is divided into several steps. The generation of FM, simulating a first order linear time-invariant system, the VCO and phase detector and the composite design. The specification for the simulation is as follows:

- Sampling frequency 10kHz, carrier frequency 2kHz, modulation frequency as a +/- percentage of the carrier frequency up to say +/-10% (+/-200Hz), unity-gain crossover frequency 400Hz, phase-margin around 55 degrees.

3.1 Simulation of Frequency Modulation and the VCO

The FM signal generation is covered in one of the labVIEW examples and so is one of the easiest of the steps to follow. The block diagram is shown in Figure 4.

![Figure 4 FM Simulation in LabVIEW](image)

In common with the whole PLL simulation the FM generator (which is only a slight modification of a LabVIEW example program) uses scalar quantities rather than arrays. Figure 4 shows a sine wave as the modulating signal and a sine wave as the carrier signal. The case statement enables different kinds of baseband signals namely, square, triangle and sawtooth. The various waveform generators default to a vector output and so they must be converted to scalar form by indexing the array and taking the zeroth value (the first sample). The FM generation is similar to how FM is generated using two sinusoidal generators. The output of the baseband signal generator is fed into the ‘sweep’ input of the second which acts as the carrier frequency. The sine generator for the carrier only accepts scalar inputs for frequency and hence the need for a scalar simulation. Normalised frequency is used throughout defined as actual frequency (Hz)/Sampling Frequency (Hz). The sampling frequency of 10kHz used in the PLL simulation was chosen as it is more than ten times the unity gain bandwidth of 400Hz and about eight times the upper break-frequency of the open-loop Bode plot. The continuously changing amplitude of the first sine wave generator is multiplied by the fixed carrier frequency of 2kHz/10kHz and this changes the frequency of the second oscillator and produces FM. A dc offset at the first oscillator output is required so that when there is no baseband signal, unity is multiplied by the carrier frequency to give a continuous waveform. The amplitude of the first oscillator (baseband signal)
determines the depth of modulation. If the baseband signal had amplitude unity then by off-setting its output by unity and multiplying by 2kHz/10kHz the second oscillator will sweep from 0Hz to 4kHz which represents 100% FM modulation. The amplitude of the baseband oscillator X100 therefore represents percentage depth of modulation. It is nominally set to 10% throughout which represents a carrier frequency which is centred on 2kHz and sweeps +/- 200Hz.

It is worth mentioning the VCO operation in this section as it is nearly identical to the above FM generation. Figure 5 shows the LAbVIEW block diagram of the VCO and the phase detector.

The VCO simulation is the FM generator with no baseband input. As with the FM generator the VCO input has an offset of unity so that when the VCO input is zero, unity multiplies the VCO free-running frequency (2kHz/10kHz) and gives a continuous waveform. The scaling of the VCO is 2kHz/volt and hence the VCO gain is $K_v = 12566\, \text{rad/s/volt}$. The input unit is taken as volts to conform to a real VCO. This gain is inherent in the VCO and must be accounted for when calculating the overall gain of the loop. The phase detector is a linear multiplier as shown in Figure 5. Its two inputs come from the FM generator and the VCO output. The VCO output is a sine wave here but can be easily changed to a square wave as is the case in most ICs. The phase detector output feeds into the filter which is discussed next.

3.2 Filter Simulation

Excluding the dynamics of the VCO which is of the form of an integrator, it remains to simulate the filter transfer function which is of the form

$$F(s) = \frac{K}{s} \frac{1 + sT_1}{1 + sT_2}$$

for $T_1 > T_2$. 

![Figure 5 VCO and Phase Detector](image)
The above equation is a second linear time-invariant (LTI) transfer function which can be split into two first-order transfer functions in cascade. LabVIEW in its basic form has many .vi blocks which are available for filtering etc but none are directly relevant to implementing the above equation. It was therefore decided to build a .vi block which would implement any first order LTI system and cascade them to construct F(s). It is a matter of choice as to whether to implement one second-order LTI or two first-order LTI systems but it was decided to go for the simplest solution. Consider the general first-order system C(s) where

\[ C(s) = \frac{g (b_0 + b_1 s)}{(s + a)} \]

One possible signal flow graph for C(s) is shown in Figure 6.

Figure 6. Signal flow graph of generic first-order system

Where y is the system output, u is the system input and x is a state variable. For LabVIEW to implement this an integration algorithm is required. There are many techniques for integration but perhaps the most popular method is to use Trapezoidal integration. A Trapezoidal integrator is represented by the Bilinear transform. In difference equation form it becomes

\[ y_k = y_{k-1} + \frac{T}{2} [u_k + u_{k-1}] \]

where T is the sampling interval (0.1ms) and has the LabVIEW block diagram shown in Figure 7.
In the diagram delta t is the step size T, and gain is 0.5 set externally. When combined with the flow graph a general first-order system is constructed and has the form of Figure 8.
external While loop. When implementing feedback in LabVIEW a signal cannot be connected directly back as in a block diagram. Instead it must be stored in a register and the previous value fed back as shown with the state variable x in Figure 8. This is because a digital system cannot respond instantaneously as there must be at least a one step time computational delay for information to pass from input to output. The above LabVIEW program can be converted into a sub .vi and used as many times as necessary provided it is defined as re-entrant. It can be used as an integrator, phase-lead or as a low-pass filter.

3.3. Composite PLL Simulation

The LTI block used in the previous section can be used to construct an integrator and phase-lead compensator for the PLL (ie the Filter). It is necessary to calculate the gain values and any parameters. One possible design for a bandwidth of 400Hz gives a phase-lead L(s) of

\[
L(s) = 10 \frac{(s + 794.2)}{(s + 7942)}
\]

so that for the LTI block a=7942, g=10, b0=794.2 and b1=1. The integrator has a gain which can be found to be approximately 2X10^6 but this does not account for any existing (hidden or implicit) gains already in the loop. These hidden gain terms consist of the VCO gain (12566) and the Phase detector gain (0.5), a total of 6283.2. Dividing this value into the overall gain gives a remaining gain of 318.141 and it is this gain which must be used on the integrator ie 318.14/s. This is shown in Figure 9 below.

![Figure 9. Part of the PLL showing the filter dynamics.](image-url)
The gain adjust parameter is nominally set to unity but can be varied to see the effect of varying the overall gain of the loop. The complete PLL FM demodulator is show in Figure 10 below.

The transient response of the loop is found by FM modulating a square wave. An external first-order filter set at a 900Hz cut-off frequency was constructed using the LTI block and is an integral part of the PLL block. The cut-off frequency can be varied from the front panel. The resulting transient response is shown in Figure 11 below for a 10Hz baseband signal. A depth of modulation of +/- 200Hz was used on the 2kHz carrier. The FM modulation index for this example is therefore $\beta = \Delta \omega / \omega_m = 20$.

If the external filter is set to a high value (4kHz) so that it is ineffective and the gain of the loop is increased significantly the effect of twice the frequency of the carrier feed-through can be illustrated in Figure 12.

![Figure 10 Main simulation loop for PLL and FM generator.](image-url)
Figure 11 Transient response of PLL to a 10Hz square wave.

Figure 12 Illustrates feed-through term for high bandwidth and no external filtering.
It is important to note that in the above simulations there is no hard limiter present as would be in a real radio receiver. Since the FM has a constant amplitude of unity there is no need. However, if additive noise or co-channel interference is added to the simulation it is then necessary to add either a hard limiter or some form of Automatic Gain Control (AGC). This is because the phase detector is a linear multiplier and hence any change in amplitude of the FM signal will result in a change of loop-gain. This can lead to instability as the gain can be either too low or too high. (see figure 3)

4. Conclusions and further work

A simulation of a PLL using LabVIEW has been described in some depth. The simulation differs from previous studies in that it is fully interactive and not a ‘one-shot’ type simulation. Unlike a real PLL all of the parameters can be varied in real-time to see the effect. For example the gain is easily increased or decreased and the external filter cut-off can be varied. At present for simplicity the PLL is designed for a given fixed carrier frequency and bandwidth but that too can be made interactive with the design equations entered as equations in LabVIEW and continuously updated. The PLL behaves in an identical fashion to an IC PLL and as such is ideal for investigating such properties as co-channel interference in FM, thresholding, multi-path and so on.

5. References.