Master Degree Project

Phase Locked Loop using LABVIEW

Author: Chetan Narashimhamurthy
Supervisor: Sven-Erik Sandström
Examiner: Sven-Erik Sandström
Semester: VT 2019
Subject: Electrical Engineering
Abstract

The phase-locked loop is an important concept in the field of wireless communication. PLL:s have wide-ranging applications in many electronic circuits. The history and the basic principle of the phase-locked loop are discussed. The different building blocks and their roles are also described along with some of the major applications of phase-locked loops. The thesis mainly describes how to build a phase-locked loop circuit using LabVIEW, as a laboratory experiment intended for a course in Radio Engineering. It was previously implemented in PSpice and this is described for comparison. The basic functions and features of LabVIEW are discussed. The primary circuit of a phase-locked loop is constructed in LabVIEW and its characteristics are noted. Some conclusions are drawn and future work on this phase-locked loop circuit using LabVIEW is suggested.
Acknowledgement

I would like to thank my thesis supervisor Professor Sven-Erik Sandström of the Faculty of Technology at Linnaeus University for providing me with this thesis topic. It would have also been extremely difficult to complete this thesis without his guidance. He was always ready to help and steer me in the right direction. I offer my gratitude to the Department of Physics and Electrical Engineering for providing the required tools for completing this thesis.

Finally, I must express my very profound gratitude to my parents and friends for catering me with unfailing support and continuous encouragement throughout my years of study and through the process of writing this thesis. This accomplishment would not have been possible without them. Thank you.

Chetan Narashimhamurthy
Contents

1 Introduction 1

2 Phase-Locked Loops 1
   2.1 History ............................................. 1
   2.2 PLL overview ...................................... 1

3 Components of PLL 2
   3.1 Phase Detector ...................................... 2
   3.2 Voltage Controlled Oscillator .................. 3
   3.3 Feedback Loop .................................... 3

4 Types of PLL 4

5 Applications 4

6 Previous Implementation 5

7 LabVIEW 7

8 Implementation 9
   8.1 Part 1 ............................................... 9
   8.2 Part 2 ............................................... 11

9 Conclusion 16

References 17
1 Introduction

The phase-locked loop (PLL) is a fundamental concept in the field of electronics and control systems. The primary purpose of this device is to let a signal track another signal. It synchronizes two signals in both phase and frequency. This circuit has many applications in electronic devices, for example in color television. A PLL is responsible for making different colors appear on the TV screen. It ensures that the color red looks red and that the blue color appears as blue on the TV screen. Another PLL will also be present to make sure everything is in the right position: if there is a standing person in the video, a PLL is responsible for making the head of that person to appear at the top and the feet at the bottom of the screen [1]. PLLs have applications in carrier recovery, clock recovery, frequency and phase demodulation, and amplifiers.

Relating to education, a course in Radio Engineering would typically have a laboratory experiment to build and study a PLL circuit using PSpice. Rather than using PSpice for the simulation, one could do this in LabVIEW instead. LabVIEW is a graphics-based programming language. This thesis consists of building a PLL in LabVIEW and understand the circuit.

2 Phase-Locked Loops

2.1 History

Phase-Locked Loops play a significant role in wireless communication. This has been in the making for a long time. There were some early mentions of PLL in the paper written by Appleton [2] in 1923, where he discussed synchronization of oscillators. The French scientist de Bellescize, who invented coherent demodulation, also proposed PLL in 1932 [3]. There were many references to phase locking over the years, but this was just in theory. The practical realization was not possible because the technology was not advanced at that time [4]. In the 1970s, Integrated Circuits (IC’s) were used worldwide, and this led to the global use of PLL applications [5].

2.2 PLL overview

![Block Diagram of a PLL](image)

Figure 2.1: Block Diagram of a PLL.

The basic working principle of a PLL is that it tracks the phase and frequency of the feedback signal in relation to that of the reference signal, and ensures that they are in sync. It has four main functional blocks namely, Phase Detector (PD), Loop Filter, Voltage Controlled Oscillator (VCO), and a Feedback network, as shown in Figure 2.1. The phase detector is a comparator that compares the phases of the signals and produces an error voltage signal. The error voltage signal contains noise and high frequency signals. These
unwanted noise signals are removed using the loop filter. The filtered signal is sent to a Voltage Controlled Oscillator, which in turn produces an oscillating signal from the error voltage signal. The oscillation frequency depends on the value of the error voltage signal. The signal generated by the VCO is fed back to the phase detector which compares the phases of the two signals. The VCO generates high-frequency signals, so there could be a frequency divider circuit in the feedback loop. The VCO adjusts its frequency so that it matches its phase with the incoming signal [6]. The two signals are in the same frequency range when they are compared. The PD compares again, and if there is a difference in phase, it produces an error signal and the loop adjusts until the phase difference is small.

A PLL has different modes of operation. It is first said to be in Free running mode. In this mode, the reference signal is not applied. The VCO has an internal frequency called free running frequency that defines the response to zero input. When an input signal is applied, the PD produces an error voltage, which in turn drives the VCO. The VCO adjusts its frequency, so as to approach the input frequency. This is the second mode of operation called the Capture Mode. The VCO performs its function, and this leads to matching of the input and the VCO frequencies. This is known as the Tracking Mode. The VCO tracks the input frequency up to a specific range called pull-in range. The pull-in range of a circuit varies based on the design specifications. Once the input signal is out of this range, the circuit won’t be in locked phase [7].

3 Components of PLL

3.1 Phase Detector

![Circuit diagram of an analog phase detector.](image)

The phase detector is a comparator circuit. It compares an incoming signal with the feedback signal. If there is a phase difference between these signals, it produces an error signal proportional to the phase difference. The error signal generated is a voltage signal. This error voltage is later used to drive the Voltage Controlled Oscillator. The error signal contains high-frequency components and noise, so it is then filtered through a Low Pass Filter. The phase detector is linear when the phase difference is small. Different circuits are used as Phase detector based on the type of PLL. In analog PLLs, an analog PD is used. The most common circuit is the multiplier. A Digital PLL makes use of Digital PDs. The best example is the JK Flipflop circuit. In some PLLs, the PD is replaced by a Phase Frequency Detector. This circuit measures both the phase and the frequency difference between the two signals.
Figure 3.1 shows a basic phase detector circuit. The error signal voltage as derived in [8], is given by

\[ v_0(t) \approx K_d(\theta_1 - \theta_2) \] (1)

In this expression, \( v_0(t) \) is the voltage error signal, \( K_d \) is the phase detector gain factor, \( \theta_1 \) and \( \theta_2 \) are the phases of the two signals. The error voltage is proportional to the phase difference between the two signals.

### 3.2 Voltage Controlled Oscillator

![A voltage-controlled oscillator](image)

Figure 3.2: A voltage-controlled oscillator [8].

A voltage-controlled oscillator drives the PLL. They are considered as the brain of the PLL [9]. The oscillation frequency of the VCO is steered by an error voltage signal. The VCO runs at an in-built frequency called free running frequency when no error voltage is applied (\( v_c = 0 \)). The error voltage obtained from the PD is applied to the VCO. The transfer function of the VCO, as mentioned in [8], is given by

\[ \omega_0 = \omega_c + K_0 v_c \] (2)

In this expression, \( \omega_0 \) is the output frequency of the VCO, \( \omega_c \) is the free running frequency of the VCO, \( v_c \) is the error voltage signal after being filtered, and \( K_0 \) is the VCO gain factor. The primary role of the VCO is to adjust its frequency so that the phases of the two signals are matched. The frequency of the VCO can be adjusted as long as it lies in the capture range of the VCO. This leads to phase locking of the circuit.

Different types of oscillators are used based on the application of the PLL. Digital VCOs are used in many applications. Some of the common VCOs are ring, crystal, and Colpitt oscillators.

### 3.3 Feedback Loop

A PLL has a negative feedback loop that carries the output of the VCO to the phase detector. Various applications can be devised by placing components in this loop. If there is a multiplier in the feedback path, their input frequency will be divided. If there is a divider in the feedback path, their input frequency will be multiplied [10].
4 Types of PLL

There are different types of PLL based on the components in their building blocks. A PLL with all analog blocks is known as Analog or Linear PLL. They were extensively used till the 1970s after which it was discovered that integrated circuits could perform the same function on a chip [11]. A PLL with all analog blocks, along with a digital phase detector, is known as a Digital PLL. Ex-OR gates and Edge triggered flip-flops are commonly used as digital phase detectors. The next type of PLL consists of digital blocks only, and it is called an All-digital PLL (ADPLL). In this type, even the loop filter and VCO are digital components. ADPLLs are more flexible, versatile and have many advantages over the analog PLL. Digital PLL circuits have higher speed, better performance and are smaller in size along with lesser cost when compared to the Analog PLL [12]. There is another type of PLL called Software PLL. Software, rather than hardware components are then used to implement the PLL [9].

5 Applications

FM demodulation is an important application of PLL. The FM wave which contains the modulated carrier signal is passed through the PLL. The VCO generates a signal that matches the phase and frequency of the FM wave. The signal generated by the VCO is based on the control signal that is applied to the VCO. So, it can be deduced that the control signal is the message of the FM wave. This is one of the methods to demodulate FM. The efficiency of this method depends on the bandwidth of the PLL; the PLL should be able to track the different fluctuations in the frequency of the FM signal [13].

A PLL can be used as a Band-Pass Filter to track Amplitude Modulated (AM) signals. The conventional Band-Pass Filters cannot be used in some applications because of their temperature dependence, and in some implementations, the carrier frequency of the AM signal varies. This PLL limits the transmitted signal to a specific boundary and also separates the AM signal spectrum from the interfering signals. This bandpass filter implemented by PLL can also be used to retrieve the carrier wave from the noisy signal in a coherent receiver [13].

A PLL is used in recovering the clock in the receiver [10]. It could also serve as an amplifier for AM signals because high-gain amplifiers are expensive. This is achieved by applying the signal to be amplified to the PLL input, and the VCO output is the amplified signal. This circuit also acts a limiter and a filter [13]. The frequency synthesizer is an essential concept in the field of communication. It generates a range of frequencies from a single frequency. A PLL is used to generate a stable and pure high-frequency signal that is used in frequency synthesis. It could also serve as a phase modulator.

Carrier recovery is another important concept in coherent receivers. The received signal contains noise and other unwanted signals. A PLL is used in recovering the input carrier signal. The received signal contains the burst of carrier signals along with other noise signals. This signal is matched with another signal from the PLL and phase locking occurs. The carrier signal is then demodulated from the PLL [10]. A PLL can be used as a Frequency Divider by placing a Frequency Multiplier in the feedback path. It can also be used as a Frequency Multiplier by placing a Frequency divider in the feedback path. In both cases, the input frequency remains constant, but the carrier frequency and the phase/frequency changes based on the component used in the feedback loop [13].
6 Previous Implementation

The simulation of a PLL was previously conducted on Pspice 9.1 for a lab exercise in the course Radio Engineering. SPICE stands for Simulation Program for Integrated Circuits Emphasis, which is a graphical programming language. This is an Analog circuit simulator. A PC version of Spice is called PSpice. It has analog and digital libraries containing all the standard components. This makes it an essential tool for many applications where the circuits are built and tested to see their performance. Based on those results, the hardware is manufactured.

![Figure 6.1: Integrator using PSpice.](image1)

A complete VCO does not exist in PSpice. So, the circuit was built from a sinus function as shown in Figure 6.1. It contains a DC voltage, Integrator and a Sinus function. The DC voltage of 1 V was passed through an Integrator and a Sinus function. An integrator is used to produce a phase ramp in PSpice, and the sinus function is used to produce a Sine wave. This is the free running frequency of the VCO. Figure 6.2 shows how the input voltage was multiplied with another DC voltage. This was done as a preparation for phase comparison.

![Figure 6.2: Voltage multiplier circuit.](image2)

Figure 6.3 shows the complete circuit of the PLL. This circuit contains sinusoidal voltage, multiplier, voltage controlled voltage source (E1), resistors, capacitor, DC voltage source, integrator and sinusoidal generator. The multiplier acts as a phase detector.
multiplying the two signals to obtain a phase difference, which in turn produces an error voltage signal. A simplified phase detector has been used in this circuit. The RC-link that follows acts as a filter that removes high frequency components and noise from the signal. The upper-frequency limit was set to 10 $Hz$, and the capacitance was calculated using the formula

$$C_1 = \frac{1}{\omega_u R}$$

with $\omega_u = 2\pi f_u$. The capacitance obtained was 15.92 $\mu F$.

The combination of Voltage Controlled Voltage Source (E1), a gain device and the SIN function were used to implement a VCO. It was decided to have the output frequency as 1 $kHz$, so the gain was calculated using the formula $2\pi f = 2\pi \times 1000 = 6283$. The SIN function is a uni-directional component that allows the data to pass from the left to right. In Figure 6.3, the data flow is from right to left in the feedback loop, so the SIN function component is flipped for the circuit to run. The VCO circuit combination produced a new signal based on the error voltage signal. This output was compared with the reference signal, and when there was a phase difference, the VCO would drive the circuit so that the phase difference reduced to a zero or reached a bare minimum.

Figure 6.4 shows the input and output waveform. The green waveform is the input signal (V1), and the red waveform is the output signal (V2). It can be observed that the two signals have different phases in the beginning and that it settles to a constant after some time. The phase difference is constant and the phase locking is achieved.
7 LabVIEW

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) is a graphical programming language developed by National Instruments (NI). It uses icons instead of line commands to create applications. LabVIEW has advantages in both the software and hardware fields. It is easy to program in LabVIEW and it has an extensive graphical user interface. In conventional programming languages, some programs take a long time to execute, while LabVIEW is faster. This is because it is designed in a unique way, and has vast libraries and functions at its disposal. It takes measurements and analyzes the data. The results are then presented to the user [9]. The circuits and devices can be built on the software and tested before they are implemented in hardware. LabVIEW has special libraries like data acquisition (DAQ) and General Purpose Interface Bus (GBIP) that interact with hardware devices [14].
The LabVIEW interface consists of two windows called Front Panel and Block Diagram. The combination of Front Panel and Block Diagram is called a VI (Virtual Instrument), and the configuration of symbols that implement the VI is typically run from the Front Panel. The Front Panel is a user interface which is built with controls and indicators. Figure 7.1 shows the basic view of a Front panel. Controls are usually input mechanisms, and indicators are mostly output displays. These controls and indicators can be found in the Controls Palette, which can be found by right-clicking on an empty place in the front panel or by selecting Controls Palette in the View option. Figure 7.3 shows the controls palette. This has many categories, and each of them contains many controls and indicators.

![Figure 7.1: Front Panel.](image1)

The Block Diagram contains the graphical code in terms of symbols, wires and connectors. These are used to determine the function of the Front Panel. Figure 7.2 shows the basic view of a block diagram. The functions palette is a tool that is found in the block diagram that has all the components that are needed to design a graphical code. They can be found under View section or by right-clicking anywhere on an empty block diagram panel. Figure 7.3 also shows the functions palette, and each category contains different functions.

LabVIEW can be connected to a hardware and communication can be established between the graphical code and the hardware. The connection is made possible by driver softwares and there are drivers for a variety of specific hardware connections [15].

![Figure 7.2: Block diagram.](image2)
8 Implementation

In this project, there has been an attempt to build a PLL in LabVIEW. This circuit can later be used to perform PLL applications like FM demodulation. The problem with this circuit is that it doesn’t follow the block diagram as seen in the textbook. The circuit is broken down into two parts to make it easy to understand its function. In the first part, the use of shift registers to perform the integration of the signal is discussed. In the second part, the complete PLL circuit and its working is explained.

8.1 Part 1

The PLL circuit runs continuously, so there was a need to design a time variable. This was achieved by making use of loops in LabVIEW. A For loop was used in this design. This can be found in the Block Diagram VI, by right-clicking anywhere on the screen and searching For Loop, or under the section Programming/Structures. The loop appears on the Block diagram as seen in Figure 8.1. This loop involves N, which is the number of iterations, and i, which is the present iteration value of the loop. The components placed inside this loop will be run continuously till the last iteration.

The PLL is a feedback circuit, so the output signal is fed back to the input which means that the current output signal is obtained based on the current input signal and the previous output signal. This raised a need to store the previous output value and use it for the next iteration. This was achieved by using a shift register. By right-clicking on the For loop, a shift register is added. This was done, and the register was initialized with the value 0 at the left side of the loop. The loop iteration i was combined with the initial value of the shift register and this sum was connected to the right end of the shift register on the loop which can be seen in the Figure 8.1.
Figure 8.1: Block diagram of integration function.

The primary function of this circuit is to set up a feedback path and store data for the integration. The shift register adds the present iteration input and output from the previous iteration. This can be verified by plotting the output by adding a waveform graph on the front panel of the VI. This was done by right-clicking on the front panel and searching *Waveform Graph*, or under the section *Modern»Graph*. An icon with the same name appears on the block diagram. In Figure 8.1, there is a square bracket connection at the tunnel where the waveform graph is connected. This is because auto-indexing is enabled. When it is enabled, the output is read one element at a time instead of all the elements in the array. The loop count can be set either in the block diagram or the front panel. In this project, it was decided to set them in the front panel. The value was set to 500. There is also a wait function in the block diagram. This specifies the amount of time the VI has to wait to execute the next iteration. This function can be found under *Programming»Timing*. The time is set in milliseconds, and in this circuit, the wait time has been set to 1 ms.

The circuit is executed by pressing the arrow button in the Front Panel VI and the waveform obtained is shown in Figure 8.2. The block diagram shows the addition function which can be verified in the waveform. So, it was concluded that the shift register stores the value from an iteration from the right side of the loop and sends it to the left side, where it is used as an initial value for the next iteration. We use this principle to perform approximate integration of the signal by the process of summation.
8.2 Part 2

This part deals with a complete circuit for the PLL. Figure 8.3 shows the full circuit of the PLL that was built on the block diagram. This circuit has two signals: a reference signal and a VCO output signal. The reference signal was obtained from the iteration count of the loop. The loop count is set to 2000 in the block diagram as seen in the figure. This was done by creating a control at the loop count in the front panel, by right-clicking on the loop count and choosing Create » Constant. The reference signal is a ramp signal. It is a constant value with an increment for every iteration. The loop count was set to 2000 in order to get a good range for the output. This means that the input will have a significant range. This was normalized by multiplying the reference with the time increment $dt$, where $dt$ is small in order to produce an accurate integration. This performs the function of integrating the reference signal producing a phase ramp. It can be observed that the complex form of the signals is being used in the circuit because it becomes easier to obtain the phase difference by just multiplying the two complex signals.

The reference signal was converted to a complex form by converting the signal to the polar form. This was achieved by connecting it to the imaginary part of the Re/Im to Polar function, which was found under Mathematics » Numeric » Complex in the functions palette. The real part of the function was set to zero. The phase of the signal can be determined by the imaginary part of the complex form which led to considering the real part as zero. The exponential function, which could be found under Mathematics » Elementary & Exponential Functions » Exponential Functions, converted the complex reference signal into a complex exponential $Z1$. It has already been discussed that the VCO runs at free running frequency when error voltage is not applied to it. Eq. 2 shows that the VCO output is a combination of PD output, Free running and gain factor. This was applied in the circuit diagram. The VCO value was needed to be integrated. This was done by multiplying the value with the integration step and making use of shift registers. This generated the total integrated phase, and it was connected to the right end of the shift register to use it for the next iteration step. The VCO output was generated in the same way as the reference signal. The VCO output is referred to as $Z2$. 

[Figure 8.2: Graphical presentation of the integration.]
Figure 8.3: Block diagram of the complete PLL.

The reference signal $Z_1$ and the VCO output $Z_2$ were obtained. The conjugate of $Z_2$ was obtained using the complex conjugate function which was found under Mathematics » Numeric » Complex in the functions palette. $Z_1$ and $Z_2^*$ were multiplied which implements the function of a phase detector. This product was plotted on the waveform graph, and it was named as PD O/P. There were some more waveforms that were plotted. They were the reference signal named as Ramp $Z_1$, the complex exponential $Z_1$, the VCO named $Z_2$ and the other called Ramp $Z_2$. Phase unwrapping was performed, and its output was plotted as Ramp $Z_2$. The VCO output was converted to real and imaginary components from the complex form using Complex to Re/Im converter. The imaginary value was connected to a phase unwrap function, which was found under Signal Processing » Signal Operation. This function removes phase discontinuities in the signal. The PD works in the range $-\pi$ to $+\pi$ and if the signal was found to be out of this range causing discontinuities, the phase unwrap would remove these discontinuities. All the graphs in the block diagram plot only the real value of the signal. It can be noticed that there are thin and thick wires in the circuit. Thin wires represent scalar entities, which means they transfer one value per iteration. Thick wires represent vector entities, which means they transfer the complete array in every iteration.

The gain, Free running, and integration step $dt$ were converted to control functions so that they could be altered in the front panel. This was done because these values needed to be changed while the circuit was running in order to understand the working of the circuit. Figure 8.4 shows the front panel VI of the circuit. It consists of five waveform graphs and four values that have already been discussed.
The four control values were varied to understand the working of the circuit. The loop count was set to 2000 to have a bigger range for the circuit. It has already been discussed that the integration step $dt$ has to be small, so it was set to 0.02. The gain and Free running were set to 1. The VI was saved, and it was run.

Figure 8.5: Front panel with $G=1$, $wf=1$.

Figure 8.5 shows the waveforms. It can be observed that the reference $Z1$ and the VCO $Z2$ have the same waveform as far as one can discern. This means they are matched with no phase difference. It can be observed that PD waveform has a transient and then settles. This is because the circuit needs some time in the beginning to enter the lock phase. The values of gain and running frequency were varied to understand the limits of the circuit. There is an option called Run continuously which is next to the Run on the front panel. This option runs the circuit until its physically stopped and not when the for
loop condition is completed. This is used so that the values could be varied in the process to observe the behavior of the circuit.

The gain was first varied to understand its effect on the circuit. It was run continuously by keeping the free-running value constant. Figure 8.6 shows the front panel when the gain was set to 10. Comparing this to Figure 8.5, it can be concluded that the gain determines the rate at which the locking takes place. When the gain was high, the PD went to zero quickly, and the phase locking was fast.

Figure 8.6: Front panel with G = 10, wf = 1.

The next step was to determine the characteristics of free-running value. The gain was next set to 1.99. The circuit was run continuously, and the free running was varied. It was increased gradually, and the circuit seemed to be in lock. Figure 8.7 shows the waveforms when the free running value was 7.3. PD O/P shows that the circuit is not in locking. So it was concluded that the free-running value of 7.2 was the limit for the phase locking when the gain was set to 1.99.
The gain was set to 10, and the circuit was run continuously by gradually increasing the free-running value. The circuit stopped locking when this value reached 32.4, hence it was concluded that the range for the free running value for the circuit with a gain of 10 is 32.4.

These observations led to the conclusion that the range of the locking of a circuit depends on the gain of the circuit. Higher gain led to a wider range of phase locking.
9 Conclusion

This project deals with the basic functioning of the Phase Locked Loop and its applications. The history of PLL was discussed with glimpses of the evolution of PLL. The working of LabVIEW was outlined. The PLL circuit which was first built using PSpice was recreated using LabVIEW. The present circuit in LabVIEW differs in appearance from the one in PSpice, but the working is found to be similar. It was concluded from the previous chapter that higher gain leads to a larger locking range. Designing a circuit with a high gain will increase its cost. So suitable gains should be used to design a PLL based on the lock-in range that is needed for a particular application.

This is a basic circuit that performs the functions of a PLL. In the future, this circuit could be used as a platform to design more circuits that can tweaked to perform some applications of PLL like FM Demodulation, Amplification, or Carrier recovery.
References


Appendix 1

Linnaeus University IFE, Elec. Eng.

Lab. 5 Radio Engineering: Simulation of PLL in LabVIEW

**TASK:** To build a PLL-circuit (Phase Locked Loop) and to investigate its function

**THEORY:** Pozar, Chapter 8

**EQUIPMENT:** NI LabVIEW 2015 SPI

**INTRODUCTION TO LABVIEW:**

LabVIEW is a graphical programming language. It uses icons instead of line commands to create applications. Open NI LabVIEW. Click on File and choose “New VI”. The two new windows that appear are called Front Panel and Block Diagram. The Front Panel is a user interface which is built with controls and indicators. Controls are usually input mechanisms and indicators are mostly output displays. The Block Diagram contains the code in terms of symbols that are run from the Front Panel. The combination of Front Panel and Block Diagram is called a VI (Virtual Instrument), and the configuration of symbols that implement the VI is normally run from the Front Panel by pressing the execute button (arrow). Figure 1 shows the diagram of Front Panel and Block Diagram.
PROCEDURE:

1. Integration using shift registers

The main principle of PLL (Phase-Locked Loop) is that it tracks the phase and frequency of a signal with respect to the reference signal and keeps it steady. Figure 2 shows how to build the integration part of a PLL. The Integration is approximately a summation and this is implemented with a shift register. To obtain the Functions menu which contains different tools to build the circuit, right click anywhere in the block diagram. There is a search option which you can use to look for a particular function.

Choose the FOR loop, which can be found under Programming/Structures. At the top left of the loop rectangle, you can set the Loop count. N is the limit for the loop count with $i$ as its counter. Right-click on N and choose Create-Control. This is done so that the Loop Count can be set in the Front Panel. Right-click on the loop and select “Add Shift Register”. Initialize the register on the left with 0. Connect the registers using the add function which can be found under Programming/Numeric. The value $i$ of the counter is now added to the register. Add time delay to the circuit. In the front panel, add Waveform Graph. This can be found under Modern/Graph. An icon appears in the Block Diagram as well. Now connect the output of the add function to the waveform graph. A tunnel is formed when the output of the add function is connected to the loop. A tunnel feeds data in and out of the structures. In our case, the data is fed out of the loop. The circuit shows there is a square bracket in the tunnel. This means the tunnel is auto-indexed. An auto-indexed tunnel receives one element at the end of every iteration. There are thick and thin wires in the circuit. Thin wires inside the loop represent scalar wires. They pick the latest value of the array during an iteration. Thick wires outside the loop represent vector wires. They pick all the values of the array at the same time during an iteration. Save the circuit and press RUN.
In the Front Panel you can see the output value on the Waveform Graph. The waveform suggests that the function of the circuit diagram is a summation. This summation can represent integration if a small step and not a unit step is used. This process is very important for the complete PLL.

2. Complete PLL

Figure 3 shows the complete circuit of the PLL that needs to be built in the Block Diagram. The Functions menu contains all the components that are needed to complete the circuit. Complex forms are used to determine the reference signal $Z_1$ and the output $Z_2$ from the VCO (Voltage Controlled Oscillator) as it is easy to extract the phase from complex numbers. To obtain the complex form, the signal has to be fed into the Imaginary part of the “Re/Im to Complex” converter. This converts the phase into a complex sinusoid. The reference used is a ramp signal. The output of the VCO is obtained from the shift register. From the previous part, it can be concluded that the shift register performs the task of phase integration by summation when the step is too small. This integrated phase is compared with the phase of the reference. The phase difference is obtained by $\arg(Z_1Z_2^*)$, so the complex conjugate of $Z_2$ has to be calculated. The Phase difference (PD O/P) determines if the loop is locked. If the phase difference gradually reduces and settles down, loop is said to be locked.
The VCO output $Z_2$ is plotted in two ways. One is direct plot coming from the exponent. For the other output, it is first converted to polar form and then “Unwrap Phase” is used. This unwraps the phase so that it extends outside of the interval $[-\pi, \pi]$.

Figure 3. Block Diagram of the complete PLL circuit.

Set Count = 2000, Integration Step $dt = 0.02$, Free Running $\omega_f = 1$ and Gain =1. Run the circuit. Increase the gain while running and observe the changes. What is the use of higher gain?

Select a suitable gain slightly higher than 1 and choose Run Continuously. Now vary the $\omega_f$. Check the range of values for which the circuit remains in locked state. Determine the lock range.

**EXAMINATION:** Standard written report