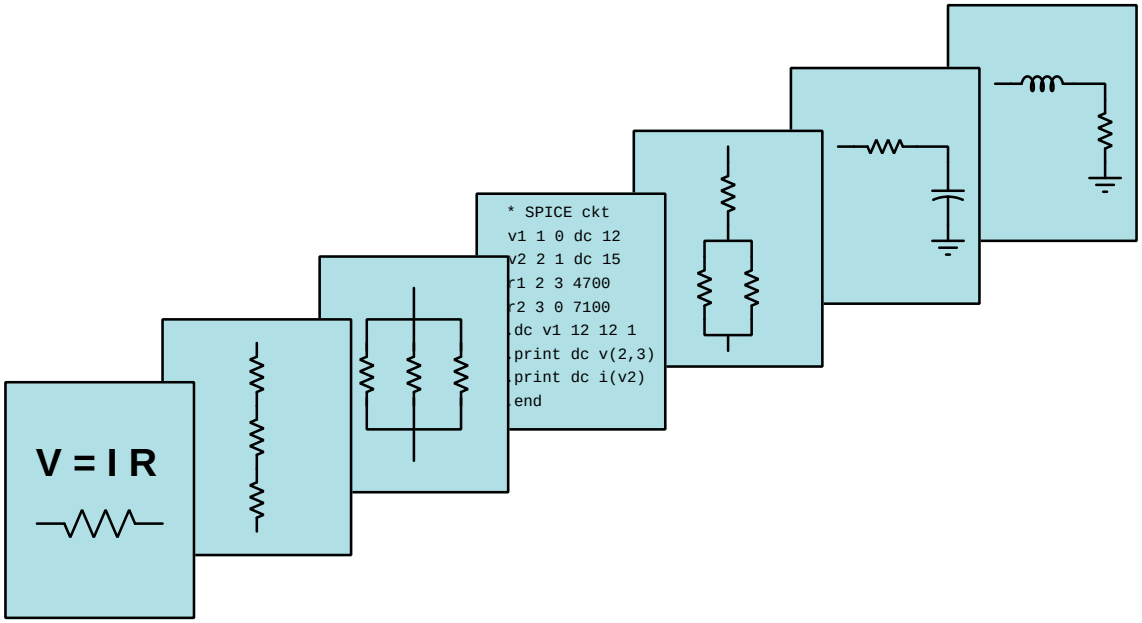
Modular Electronics Learning (ModEL) project



Phase-Locked Loops

© 2022-2025 by Tony R. Kuphaldt – under the terms and conditions of the Creative Commons Attribution 4.0 International Public License

Last update = 20 November 2025

This is a copyrighted work, but licensed under the Creative Commons Attribution 4.0 International Public License. A copy of this license is found in the last Appendix of this document. Alternatively, you may visit `http://creativecommons.org/licenses/by/4.0/` or send a letter to Creative Commons: 171 Second Street, Suite 300, San Francisco, California, 94105, USA. The terms and conditions of this license allow for free copying, distribution, and/or modification of all licensed works by the general public.

Contents

1 Introduction **3**
- 1.1 Recommendations for students . . . 3
- 1.2 Challenging concepts related to phase-locked loops . . . 5
- 1.3 Recommendations for instructors . . . 6

2 Case Tutorial **7**
- 2.1 Example: PLL phase detector type experiment . . . 8
- 2.2 Example: PLL filter cutoff experiment . . . 10

3 Tutorial **15**
- 3.1 Frequency determination . . . 16
- 3.2 Negative feedback . . . 17
- 3.3 Phase-locked loop principles . . . 20
- 3.4 Phase detector types . . . 21
  - 3.4.1 XOR gate phase detectors . . . 22
  - 3.4.2 Flip-flop phase detectors . . . 24
  - 3.4.3 Mixer phase detectors . . . 27
- 3.5 PLL frequency detection . . . 28
- 3.6 PLL frequency multiplication . . . 29

4 Questions **31**
- 4.1 Conceptual reasoning . . . 35
  - 4.1.1 Reading outline and reflections . . . 36
  - 4.1.2 Foundational concepts . . . 37
  - 4.1.3 Simple musical synthesizer design . . . 38
  - 4.1.4 CD4046B phase-locked loop applications . . . 39
  - 4.1.5 Tone-controlled switching system . . . 40
- 4.2 Quantitative reasoning . . . 41
  - 4.2.1 Miscellaneous physical constants . . . 42
  - 4.2.2 Introduction to spreadsheets . . . 43
- 4.3 Diagnostic reasoning . . . 46

A Problem-Solving Strategies **47**

B Instructional philosophy **49**
B.1 First principles of learning 50
B.2 Proven strategies for instructors 51
B.3 Proven strategies for students 53
B.4 Design of these learning modules 54

C Tools used **57**

D Creative Commons License **61**

E References **69**

F Version history **71**

Index **72**

Chapter 1

Introduction

1.1 Recommendations for students

Feedback loops are extremely important in electronic system design, with operational amplifiers being perhaps the preeminent example of feedback used to practical benefit. However, using amplifiers to automatically balance two voltage signals is only one application of feedback. Another, explored in this module, is the automatic balancing of two signal *frequencies* until equality is achieved. This is the basic concept of a *phase-locked loop* or *PLL*.

Phase-locked loops are widely used in signaling systems for such purposes as frequency division, frequency multiplication, and frequency demodulation (FM).

Important concepts related to phase-locked loops include **negative feedback**, **phase shift**, **frequency**, **oscillator** circuits, digital **counter** circuits, frequency **divider** circuits, **period**, counter **modulus**, **operational amplifiers**, **flip-flops**, **logic gates**, **mixers**, **filters**, **block diagrams**, **functional inversion**, and signal **synthesis**.

Here are some good questions to ask of yourself while studying this subject:

- How might an experiment be designed and conducted to explore the concept of phase detection? What hypothesis (i.e. prediction) might you pose for that experiment, and what result(s) would either support or disprove that hypothesis?
- How might an experiment be designed and conducted to explore the concepts of frequency division? What hypotheses (i.e. predictions) might you pose for that experiment, and what result(s) would either support or disprove those hypotheses?
- What effect(s) does negative feedback have in amplifier circuits?
- How does positive feedback differ from negative feedback?
- What does the *error* signal represent in a feedback system?
- What does it mean to *modulate* a signal?

- What does it mean to *demodulate* a signal?
- Why must operational amplifiers exhibit extremely high voltage gains in order to be as useful as they are?
- What does it mean for a function to be *inverted* in a feedback system?
- How is "error" typically sensed in a phase-locked loop feedback system?
- What do *filter* networks do, and how do they typically work?
- What does it mean for a PLL to "pull in"?
- What does it mean for a PLL to reach a condition of "lock"?
- How are PLL circuits useful in digital communication networks?
- How may a digital counter circuit be used to divide the frequency of a pulse signal?
- How may a PLL be used to multiply the frequency of a pulse signal?
- What causes *cycle slip* in a PLL?

1.2 Challenging concepts related to phase-locked loops

The following list cites concepts related to this module's topic that are easily misunderstood, along with suggestions for properly understanding them:

- **Negative feedback** – this non-intuitive concept is difficult to grasp due its recursive nature: the output of an amplifier is fed back to its input, which affects its output, which affects its input, *ad infinitum*. A particular challenge with negative feedback is comprehending why this does not irresistibly lead to oscillations, given the fed-back signal's tendency to counter whatever direction the output signal is changing in. One rationale for the oscillation hypothesis is that we as human beings tend to imagine changes in discrete steps rather than as continuous trends, and so it is understandable why a new student might assume negative feedback causes the amplifier to leap back and forth between the (only) two extremes of signal range they might be envisioning. In reality, as the output of an amplifier with negative feedback drifts in one direction, the fed-back signal has the effect of slowing that drift down until it settles at some value rather than oscillate: as the "error" approaches zero, so does the rate of drift.

- **Functional inversion** – one of the non-intuitive consequences of negative feedback is that whatever function is placed in the feedback loop ends up being inverted in the amplifier's overall response. For example, feeding back a divided signal results in the amplifier multiplying its input voltage. This can and will make sense if one takes the time to trace signal values in a feedback system, which is one great application for block diagrams!

1.3 Recommendations for instructors

This section lists realistic student learning outcomes supported by the content of the module as well as suggested means of assessing (measuring) student learning. The outcomes state what learners should be able to do, and the assessments are specific challenges to prove students have learned.

- **Outcome** – Demonstrate effective technical reading and writing

  Assessment – Students present their outlines of this module's instructional chapters (e.g. Case Tutorial, Tutorial, Historical References, etc.) ideally as an entry to a larger Journal document chronicling their learning. These outlines should exhibit good-faith effort at summarizing major concepts explained in the text.

  Assessment – Students present their "Conceptual Toolbox" Theory Journal entries listing important concepts applied in the Tutorial text(s). A "Conceptual Toolbox" serves as inspiration when solving problems, each concept suggesting a potential problem-solving action. Challenge students to collaboratively generate a Conceptual Toolbox for the day's topic during the class session and then use it when solving problems together, thus fostering the habit of reasoning from principles rather than recalling procedures!

- **Outcome** – Design a circuit utilizing a phase-locked loop

  Assessment – Sketch a schematic or block diagram of a circuit employing a PLL to control a device remotely using audio tone signals; e.g. pose problems in the form of the "Tone-controlled switching system" Conceptual Reasoning question.

- **Outcome** – Independent research

  Assessment – Read and summarize in your own words reliable source documents on the subject of phase-locked loops. Recommended readings include *CD4046B Phase-Locked Loop: A Versatile Building Block for Micropower Digital and Analog Applications* by David K. Morgan (Texas Instruments document number SCHA002A, February 2003.

Chapter 2

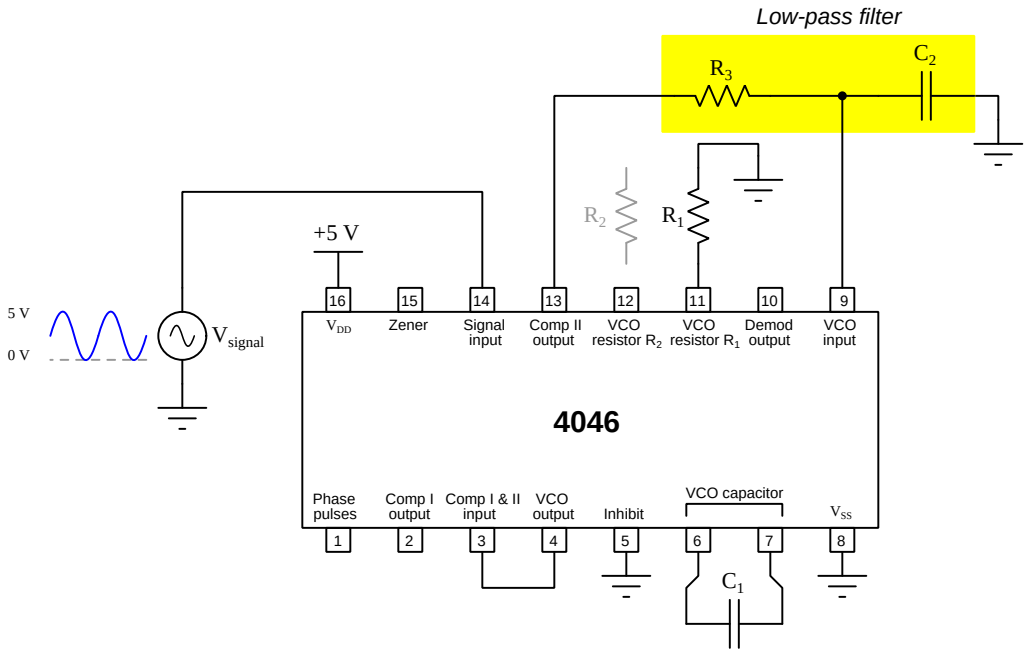
Case Tutorial

The idea behind a *Case Tutorial* is to explore new concepts by way of example. In this chapter you will read less presentation of theory compared to other Tutorial chapters, but by close observation and comparison of the given examples be able to discern patterns and principles much the same way as a scientific experimenter. Hopefully you will find these cases illuminating, and a good supplement to text-based tutorials.

These examples also serve well as challenges following your reading of the other Tutorial(s) in this module – can you explain *why* the circuits behave as they do?

2.1 Example: PLL phase detector type experiment

Here we see a CD4046B phase-locked loop IC configured to lock into a 2 kHz sine-wave signal:

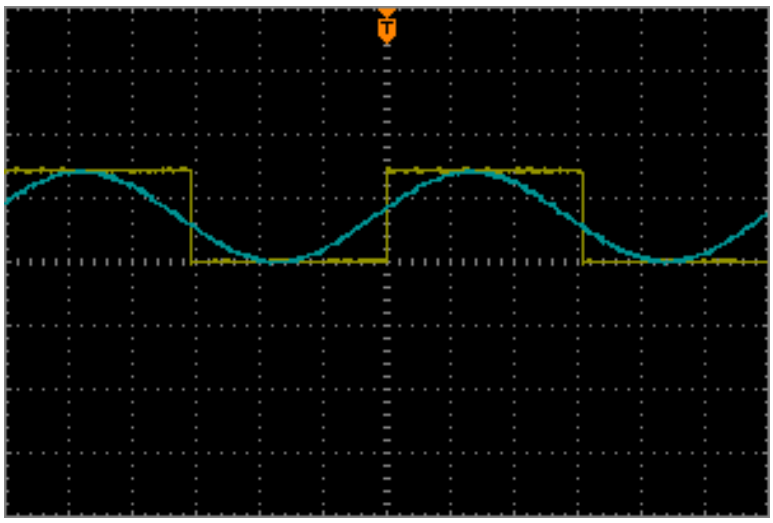


- $R_1 = 100\ \text{k}\Omega$
- $R_2 = \textit{omitted}$
- $R_3 = 15\ \text{k}\Omega$
- $C_1 = 0.0047\ \mu\text{F}$
- $C_2 = 2\ \mu\text{F}$

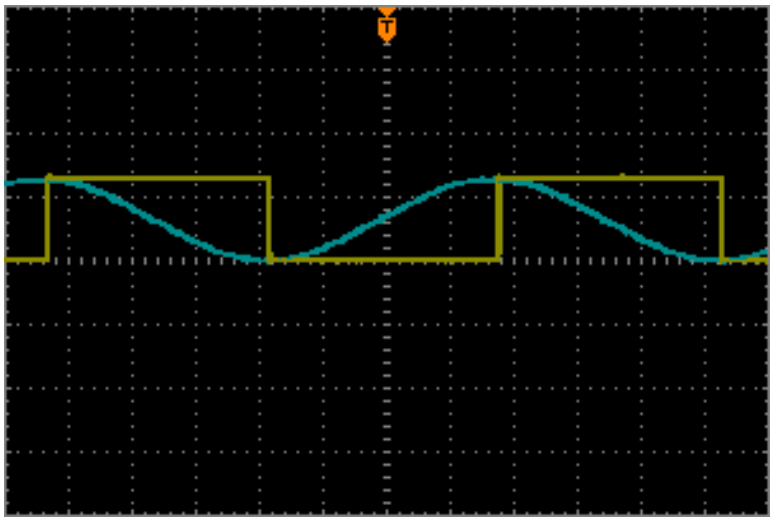
A convenient way to test how long it takes for the PLL to "lock" is to briefly connect another 0.0047 μF capacitor in parallel with C_1 to cause the VCO's frequency to suddenly fall. Following disconnection of this extra capacitor, you may then monitor the VCO output and the input signal on a dual-trace oscilloscope to observe the two waveforms gradually return to a synchronized condition.

The low-pass filter's cutoff frequency definitely impacts the ability of the PLL to achieve lock. Increasing C_2's capacitance increases the circuit's time-to-lock because it dampens the VCO's response to out-of-phase conditions reported by the phase detector.

The 2 kHz sine wave signal (applied between pin 14 and ground) appears in blue while the VCO's square-wave output signal (measured between either pin 3 or pin 4 and ground) appears in green. Note how the phase shift between input and VCO waveforms is very nearly 0^o:

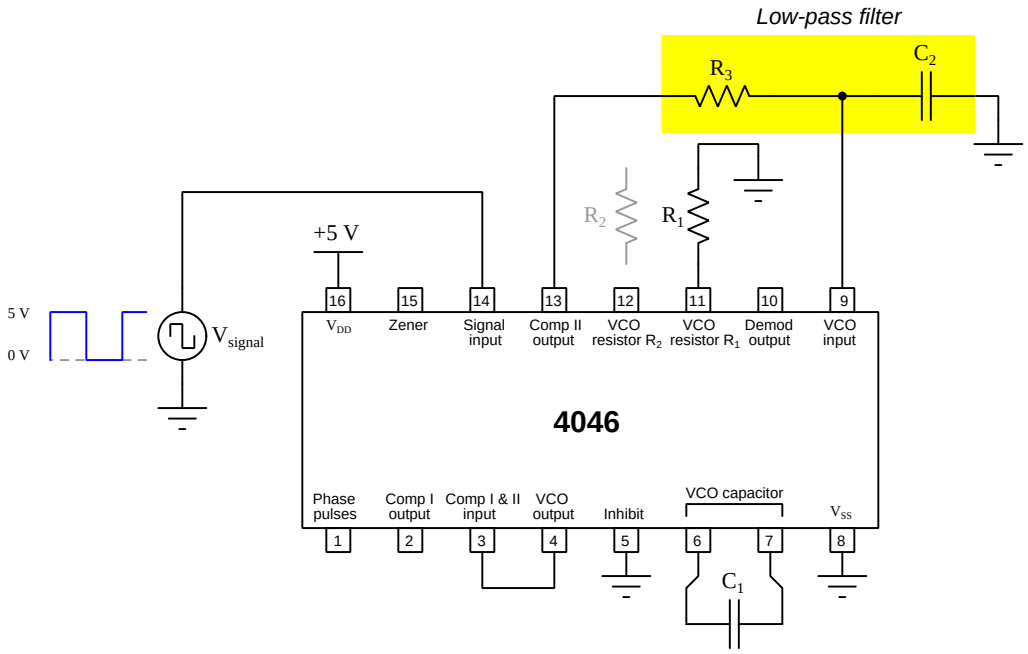


Re-configuring the CD4046 to use Comparator I rather than Comparator II (i.e. connecting R_3 to pin 2 rather than pin 13, reducing C_1 to 0.001 μF, and reducing C_2 to 0.0047 μF) resulted in these input and VCO waveforms. Note how the phase shift between input and VCO waveforms is very nearly 90^o:



2.2 Example: PLL filter cutoff experiment

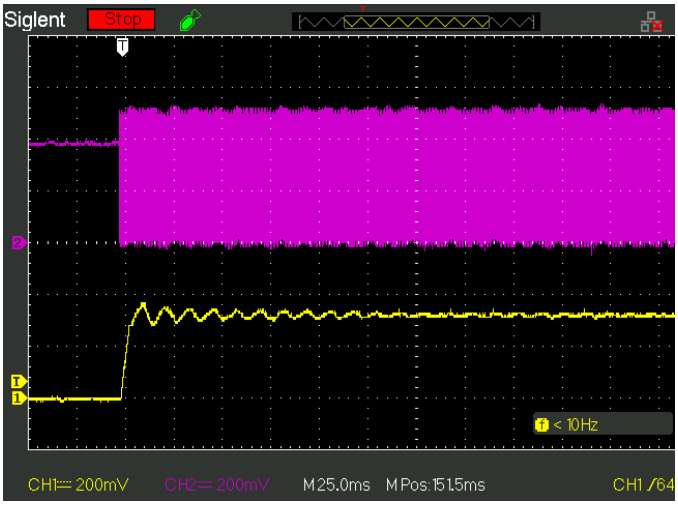
Here we see a CD4046B phase-locked loop IC configured to lock into square-wave signals suddenly applied to the signal input terminal:



- $R_1 = 100\ \text{k}\Omega$
- $R_2 = \textit{omitted}$
- $C_1 = 0.00068\ \mu\text{F}$
- $C_2 = 1\ \mu\text{F}$
- $R_3 = \textit{variable between 4.7 k}\Omega\textit{ and 47 k}\Omega$

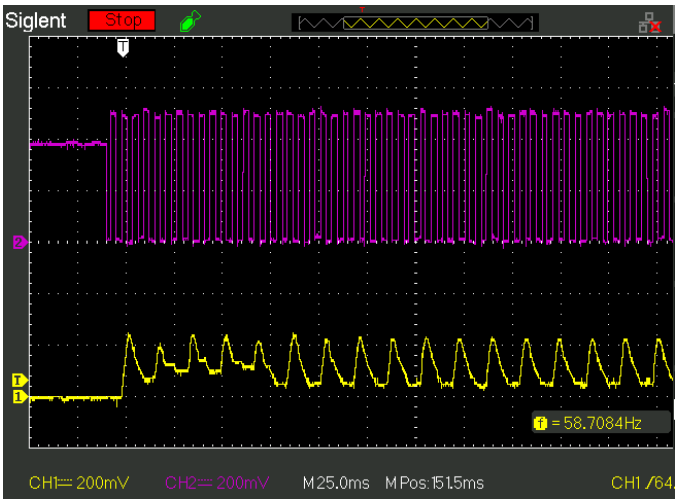
In this experiment we will vary the value of the low-pass filter network's resistor (R_3) value in order to change that filter's cutoff frequency, and observe the effect this has on the PLL's ability to achieve "lock" on two different frequencies of square-wave input signal. An oscilloscope monitors the incoming V_{signal} on channel 2 (violet) as it is turned on, versus the VCO input voltage on channel 1 (yellow) which is the output of the low-pass filter network.

$R_3 = 4.7$ kΩ and $f_{input} = 1.3$ kHz



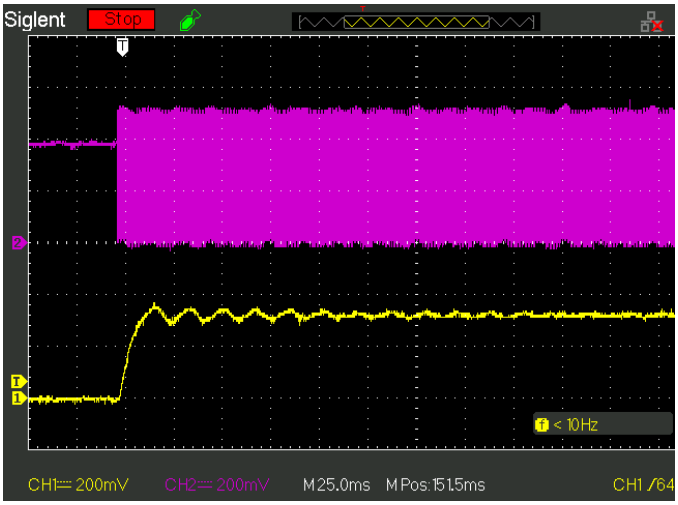
Using a 4.7 kΩ resistor for R_3 sets the filter's cutoff frequency at 33.863 Hz. With this cutoff we can see the VCO's input voltage "hunt" above and below its ultimate DC settling value. This means the VCO's output frequency will vary up and down as the PLL seeks to match the input and output phases.

$R_3 = 4.7$ kΩ and $f_{input} = 175$ Hz



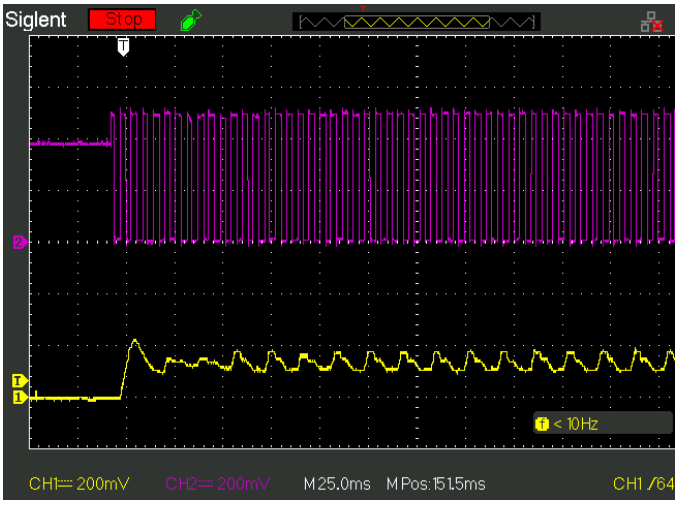
Applying a much lower square-wave signal frequency of 175 Hz and using the same low-pass filter components values reveals a complete lack of lock, with the PLL continually "hunting" back and forth.

$R_3 = 10$ kΩ and $f_{input} = 1.3$ kHz



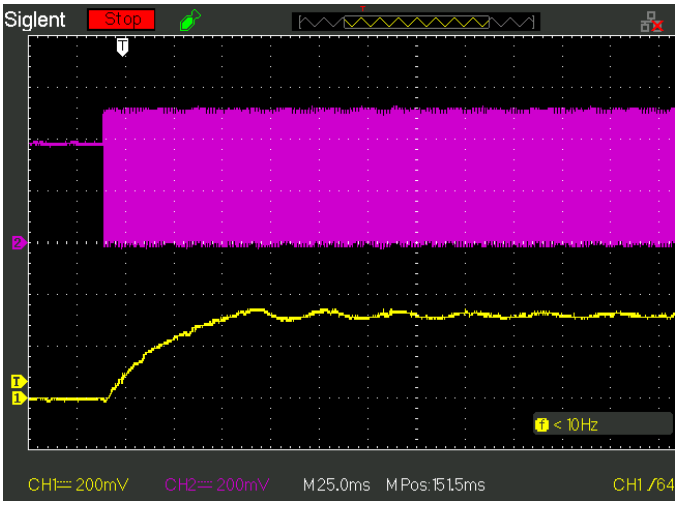
Using a 10 kΩ resistor for R_3 sets the filter's cutoff frequency at 15.915 Hz. With this cutoff we can see the VCO's input voltage still "hunts" above and below its ultimate DC settling value, but it does so at a slower rate than it did when R_3 was 4.7 kΩ. This means the VCO's output frequency will vary up and down at a more leisurely pace as the PLL seeks to match the input and output phases, and it also means the "lock" state will take longer to achieve.

$R_3 = 10$ kΩ and $f_{input} = 175$ Hz

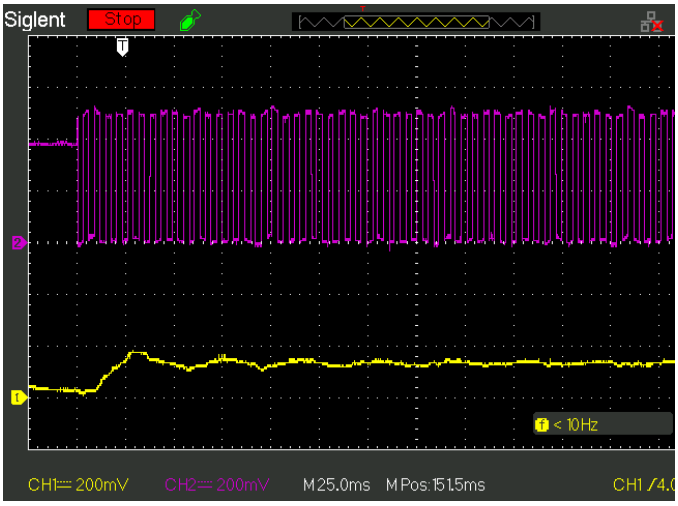


Applying a much lower square-wave signal frequency of 175 Hz and using the same low-pass filter components values reveals another failure to lock, with the PLL continually "hunting" back and forth though not as severely as before with the 4.7 kΩ R_3 value.

$R_3 = 47$ kΩ and $f_{input} = 1.3$ kHz



$R_3 = 47$ kΩ and $f_{input} = 175$ Hz



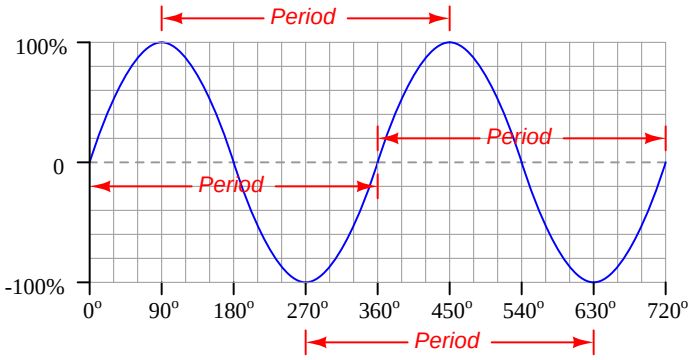
Using a 47 kΩ resistor for R_3 sets the filter's cutoff frequency at 3.3863 Hz. With this cutoff we can see the VCO's input voltage gradually grows and then "hunt" at an even more leisurely pace before settling into its ultimate DC settling value.

Chapter 3

Tutorial

3.1 Frequency determination

A simple definition of *frequency* may be stated as "how often an AC signal completes a full cycle", typically measured in *cycles per second* which is the same as the unit *Hertz* (Hz). Many ways exist to measure the frequency of a signal, including indirectly by measurements of voltage or current (using *reactive* components such as inductors and capacitors whose impedances vary with frequency), and directly by means of an oscilloscope (visually identifying the length of the signal's period and then reciprocating that value, converting *seconds per cycle* into *cycles per second*) as illustrated below:



$$f = \frac{1}{t_{period}}$$

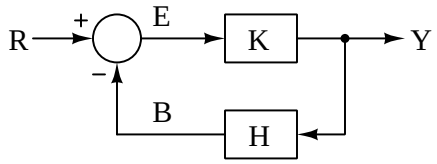
Many AC circuits operate at constant or near-constant frequency, AC electric power grids being a large-scale example of this. Other AC circuits such as audio-frequency amplifiers must operate over a range of signal frequencies, in this case the range of human hearing (approximately 20 Hz to 20 kHz). Others yet experience signals whose frequencies vary over relatively narrow ranges, such as radio communications circuits where a "carrier" signal maintains a fixed frequency but is *modulated* to produce other frequency values varying with the information conveyed.

In systems where a signal's frequency encodes information, the determination of frequency and conversion of that frequency value into other forms such as analog voltage values is very important. A circuit known as a *phase-locked loop* or *PLL* uses negative feedback much like an operational amplifier circuit to match one frequency against another, and in so doing provide a means for signal conversion, frequency multiplication, and other useful feats. This Tutorial will explore the basic concepts of phase-locked loops, starting with a brief review of negative feedback theory.

3.2 Negative feedback

Feedback is when a signal output by a circuit is "fed back" into one of its inputs for various purposes. *Positive* or *regenerative* feedback is when the fed-back signal acts in such a way as to reinforce itself to amplify the original effect. *Negative* or *degenerative* feedback is when the fed-back signal acts in such a way as to work against or cancel out the original effect. Applications of positive feedback include electronic oscillators and latch circuits. Applications of negative feedback include precise signal amplification and phase-locked loops, and will be the type of feedback explored in this Tutorial.

A popular form of diagram to represent feedback systems in engineering is the *block diagram*, so called because each segment of the system is represented by a block with arrows going in (inputs) and arrows coming out (outputs) at each. Below we see a block diagram representing a negative feedback system:



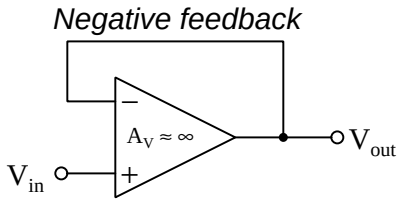
Block diagrams are mathematical abstractions of the real physical system, each block representing some mathematical function and each variable representing some quantity passed between functions. The letters used to symbolize each are fairly standard within engineering: R is the *Reference* or input signal to the system, K represents the *gain* of the amplifier or "compensator", Y represents the output signal from the system, H represents the feedback network, B represents the *Back*-fed signal, and E represents the *Error* (or *difference*) between the reference signal R and the back-fed signal B.

Cardinal Rule of Negative Feedback

> *Given sufficiently aggressive compensation, the fed-back signal will match the reference signal to eliminate "error" in a negative feedback system.*

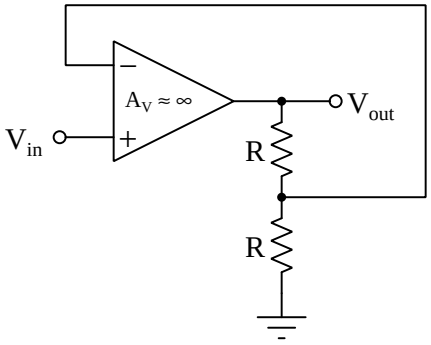
A "thermostat" room temperature control system serves as a helpful illustration of this principle. So long as the thermostat system is working and has sufficient capacity to bring a room to a desired "setpoint" temperature (R), the room temperature (B) will be held nearly equal to setpoint such that there will be negligible error (E) between the desired room temperature and the actual room temperature. In this analogy, the block diagram's K element is the heating or cooling system, Y represents the heat rate added to or removed from the room, and H represents various thermal "loads" on the room (e.g. outdoor temperature, wind chill, fresh air exchange rate). In a capable and properly-functioning system, $B \approx R$ and therefore $E \approx 0$.

Operational amplifier (or *opamp*) circuits exploit this property of negative feedback to very precisely amplify all kinds of electronic signals. If the output terminal of a standard opamp is connected directly to its inverting ($-$) input terminal, its output voltage will become almost exactly equal to the voltage input at its noninverting ($+$) terminal:



$V_{out} \approx V_{in}$ for this "voltage follower" network

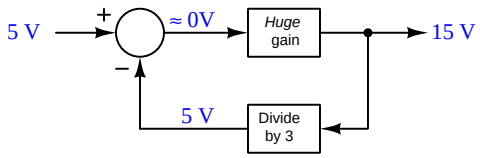
If, however, we place a voltage divider between the opamp's output terminal and its inverting input terminal, V_{out} must achieve a *greater* voltage level in order to match V_{in} to eliminate error. This creates a *voltage amplifier* whose gain is set by the divider network's ratio:



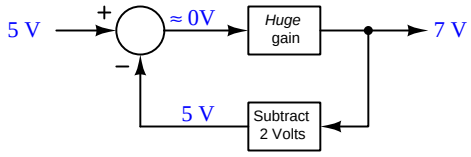
$V_{out} \approx 2V_{in}$ when the two resistors are equal

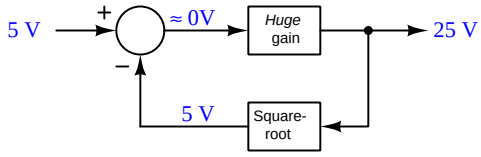
A very useful property of negative feedback, besides forcing one signal to become equal to another, is *functional inversion* where the relationship of the compensating device's (opamp, K) output signal (Y) to the input signal (reference, R) becomes the inverse of the feedback network's (H) characteristic. This is why placing a voltage *divider* network in the opamp's feedback path resulted in the amplifier exhibiting a voltage *gain* from V_{in} to V_{out}.

Functional inversion is non-intuitive, so it merits further elaboration. We will illustrate some examples using standard block-diagram notation. Below we see a negative-feedback amplifier system with a 3:1 voltage-divider network in the feedback signal path. This forces the opamp to produce an output voltage three times as great as the input:



Here the high-gain opamp *did what it had to* to make the weakened back-fed signal practically equal to the reference signal, and in so doing created a strengthened output. If our feedback network had instead *subtracted* a certain amount of voltage, the result would have been an output with that same voltage offset *added*. Likewise, if the feedback were somehow made to be a *square-root* function, the output would become the *square* of the reference signal:



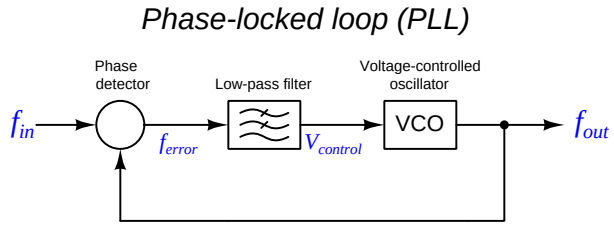


Functional inversion is more than just a curiosity. This principle allows us to build more than twice as many types of signal-manipulation networks than we could otherwise. Using negative feedback, any network characteristic we might discover can be exploited to produce precisely the opposite effect.

Next we will see how to build negative-feedback systems where *frequency* rather than voltage is the compared signal quantity.

3.3 Phase-locked loop principles

The following block diagram shows the basic configuration of a *phase-locked loop* or *PLL*:



Three components form the basis of any PLL, the *phase detector*, the *filter*, and the *VCO*:

- **Phase detector** – also known as a **phase comparator**, this implements *subtraction* function necessary for any negative-feedback system into a phase-locked loop system. Instead of outputting the difference (error) between two voltage signals, a phase detector outputs a signal proportional to the *phase shift* separating two AC signals.
- **Low-pass filter** – blocks high-frequency signals from passing through, but allows low-frequency and DC (zero-frequency) signals to pass with little attenuation. This focuses the feedback loop's self-correction on longer-term averages of error rather than immediate calculations of error.
- **Voltage-controlled oscillator** – outputs an AC signal whose frequency is proportional to the VCO's DC input voltage.

If $f_{out} < f_{in}$, the phase detector's filtered signal prompts the VCO to increase its output frequency. If $f_{out} > f_{in}$, the VCO receives a decreasing control voltage which forces its output frequency to decrease. Once $f_{out} = f_{in}$, the phase-locked loop is said to be in a condition of *lock*. In this lock condition, the two signals may still have some small amount of phase shift between them, but their frequencies will match precisely.

As one might guess, getting two frequencies to match each other by sensing their relative phase shift and driving an oscillator accordingly is not as simple a task as getting two DC voltage signals to match each other as in the case of a negative-feedback opamp circuit. Different phase-comparator circuit designs exist, each with its own practical application(s). Similarly, the cutoff characteristics of the low-pass filter also greatly affect a PLL's ability to achieve lock.

When we measure phase shift between two signals of differing frequency, we find that phase angle constantly drifts, cycling from 0^o (in-phase) to 180^o (perfectly out-of-phase) and back to 0^o at a rate equal to the difference between those two frequencies[1]. This means a simple phase detector will indicate one signal alternately leading and then lagging the other, the detector's output sending a "confused" message to the VCO, telling it to speed up and then slow down. For this reason, a PLL in the process of "pulling in" to a condition of lock may output a frequency that wavers on its way to matching the input frequency.

[1] e.g. for 5 kHz and 5.2 kHz, the phase shift repeats its cycle at a rate of 200 Hz.

The challenge of achieving a "lock" condition by measuring phase shift between two unequal-frequency signals also means PLLs have much longer settling times than a typical DC opamp circuit. Whereas an opamp circuit need only subtract one DC signal from another to determine which way to drive its output, and may drive its output at any rate less than its slew rate limit, a PLL can neither "decide" nor "act" as rapidly.

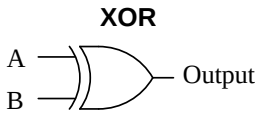
The frequency range over which a PLL may be expected to achieve lock is called the *capture range*, while the frequency range over which a PLL may be expected to maintain lock once achieved is called the *lock range*. For all PLLs, lock range is broader than capture range. This is to say, a PLL struggles more to initially achieve a locked condition than it does to hold on to that lock once achieved.

3.4 Phase detector types

An essential component within any phase-locked loop (PLL) is the *phase detector*. Its purpose is to measure the phase shift existing between two different waveforms and provide that measurement to the voltage-controlled oscillator (VCO) for corrective action, so that the two waveforms may be brought into phase with each other, or at least brought into a condition of equal frequency. Multiple types of phase detectors exist, so we will explore some of them in the following subsections.

3.4.1 XOR gate phase detectors

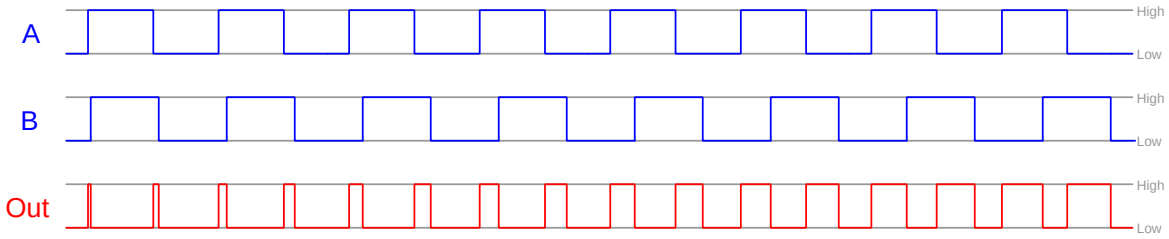
The simplest possible phase detector for digital (pulse) signals is an Exclusive-OR gate, as shown below:



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Output will be "high" whenever pulse signals A and B do not precisely match

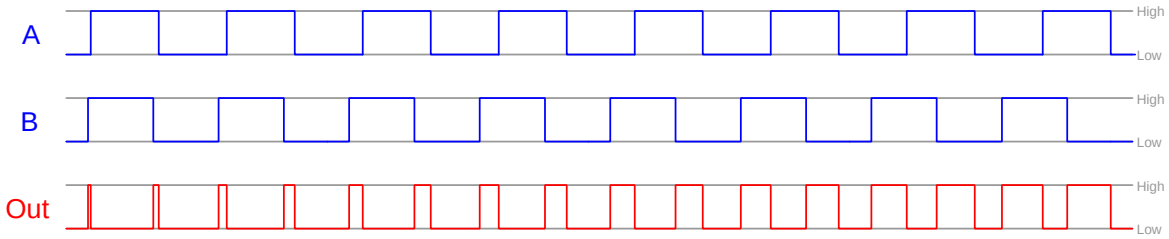
As the XOR gate's truth table reveals, the gate outputs a "low" state if the input states are identical and a "high" state if they differ. This means any two pulse signals compared by an XOR gate will generate a pulse signal at the gate's output whose duty cycle reflects the degree to which the input pulses are out of phase with each other. In the timing diagram shown below we have two pulse signals of slightly differing frequency, signal A having a higher frequency than signal B:



Frequency of A is higher than frequency of B

It should be clear from this timing diagram how the duty cycle of the XOR gate's output increases as the two waves become further and further out of phase with each other. If the XOR gate's output pulse is passed through a low-pass filter, as is typical within a PLL system, this varying duty cycle will become filtered to be a DC voltage whose average value is proportional to the duty cycle. This DC signal drives the VCO to self-correct, eventually reaching a state of equilibrium where the two pulse waveforms must equal in frequency. In order for this feedback to actually be self-correcting in the timing diagram example shown above, the filtered XOR gate output signal must drive signal B: this way, the increasing duty cycle (as signal B falls further behind A) will act to boost the VCO's output frequency and bring signal B's frequency up to meet signal A's.

Here, though, we see a disadvantage of the XOR gate as a PLL phase detector. Suppose we encounter a situation where the feedback signal (B) happens to run at a *higher* frequency than signal A. Observe what happens to the XOR's output pulse duty cycle as these two waves fall further and further out of phase with each other:



Despite the fact that the lead/lag phase relationship is reversed between signals A and B, the XOR gate outputs the exact same pulse signal with progressively increasing duty cycle! This is because the XOR gate only has the ability to detect *differences* between the two signal states, and cannot indicate which signal is ahead of the other. This is problematic in the same PLL where the filtered XOR output signal drives the VCO generating signal B. If an increasing duty cycle instructs the VCO to run at a higher frequency, it will actually make the phase mismatch *worse*. In other words, as soon as the feedback signal's frequency runs faster than the other frequency, the XOR phase detector's feedback action becomes *positive* rather than negative.

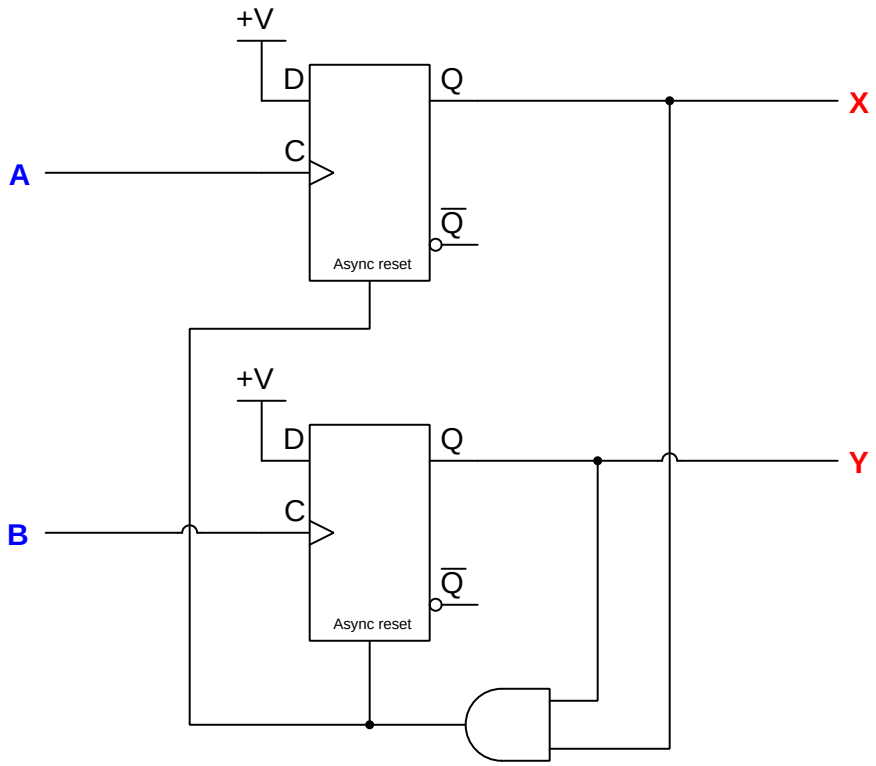
The situation is not as hopeless as it might seem, though, because this action will eventually cause signal B to fall behind signal A in phase shift. It's akin to two people racing each other around a circular track: if one pulls ahead of the other and continues to gain speed, the faster racer will eventually approach the slower racer *from behind*. This action, where the undesirable behavior of the XOR phase detector causes the phase shift to pass through 180^o and then "catch up from behind" to the other signal is called *cycle slipping*. In a PLL using an XOR phase detector, every slipped cycle provides another opportunity for the loop to stabilize itself with negative feedback.

As one might conclude, this tendency for an XOR phase detector to create "cycle slip" in a phase-locked loop system may result in the loop failing to achieve lock for several complete cycles, like a racer who keeps passing their opponent on a circular track before finally adjusting their speed to match.

Thus, XOR detectors suffer from only being effective when the controlled (feedback) signal's phase differs from the input signal's in a particular orientation, either leading or lagging depending on how the VCO responds to a non-matching output from the XOR gate. This is why, when in operation within a PLL, simple XOR detectors can never bring the two frequency signals perfectly in-phase with each other. One must always lead the other by some amount (usually 90^o) in order that the XOR gate's output will always have a self-correcting effect (i.e. negative feedback) on the system.

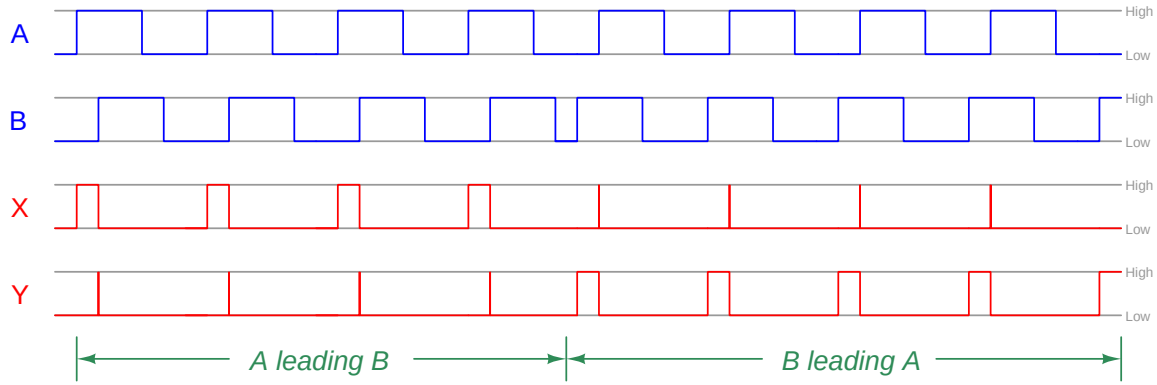
3.4.2 Flip-flop phase detectors

A different design of phase detector intended to overcome the limitations of the XOR gate is shown below, using two D-type flip-flops whose clock inputs are driven by the two waveforms to be compared. An AND gate monitors the Q outputs of both flip-flops and immediately resets them whenever both Q outputs are high:



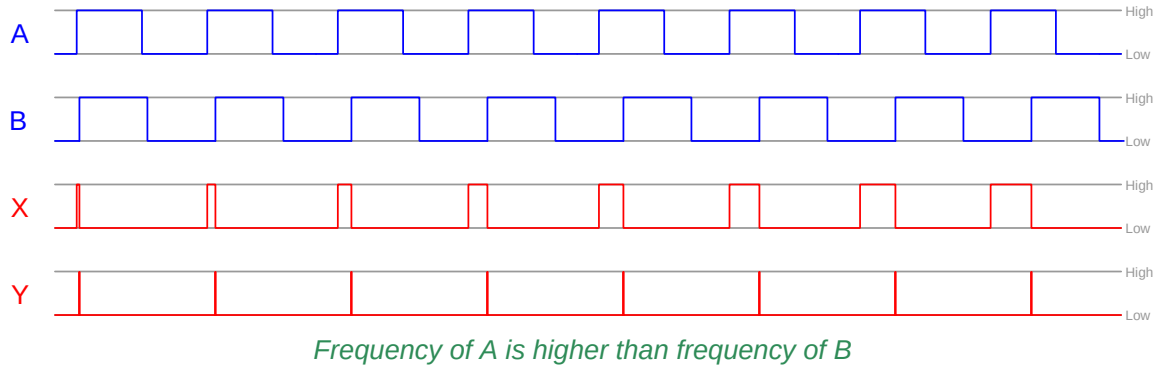
Note how the D inputs of both flip-flops are tied "high" so that each flip-flop's Q output is sure to go "high" as well at the rising edge of its input pulse signal. Each flip-flop Q output, therefore, remains in a "high" state until it gets reset, and that resetting happens as soon as the *other* flip-flop goes high as well. This means the flip-flop whose Q output remains "high" for the longest amount of time is the one being clocked by the *leading* pulse waveform. For example, if output pulse X remains "high" longer than output pulse Y, then signal A must be leading signal B. If signal A begins to lag behind signal B, then output Y will remain "high" longer than output X.

In the timing diagram shown below we begin with pulse signal A leading ahead of pulse signal B, and then approximately half-way along the time axis we see pulse signal B suddenly lead ahead of pulse signal A:



As you can see, whenever A leads B we see substantial pulse durations on output X but not output Y. When input signal B takes the lead, these signal durations swap between outputs X and Y with only the Y output generating substantial-length pulses. The dual outputs of this phase detector therefore provide unambiguous indication of which signal leads and which lags.

Next we see another timing diagram showing what happens when signal A's frequency is greater than signal B's:

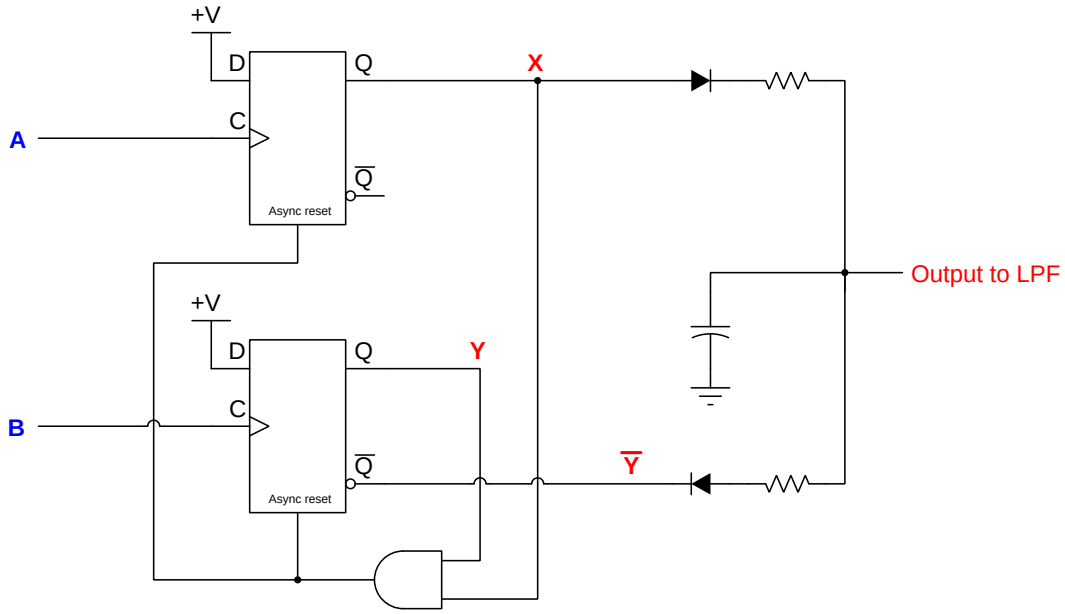


With its higher frequency (i.e. shorter period), signal A keeps pulling further and further ahead of signal B which means the phase shift between the two pulse signals grows larger over time. This causes output signal X's pulse durations to gradually increase over time while output signal Y only briefly "spikes[2]" at each re-set. If a phase detector within a phase-locked loop system detected a

[2]This "spike" signal seen on output Y is of course the other input of the AND gate going to a "high" state just before the AND gate's output goes "high" as well to immediately reset both D-type flip-flops. The duration of this transient pulse is the combined propagation delay time of the AND gate and of the flip-flops' asynchronous reset functions.

growing phase shift such as this, the result would be correction by the PLL's oscillator to either increase the frequency of signal B or decrease the frequency of signal A (whichever one was the output of the VCO) until the two signals were locked in-phase with each other. In other words, in an operating PLL we would not expect to see one of the pulse signals continue to pull ahead of the other without some eventual correction as the PLL "pulls in" to an (eventual) locked condition.

In order for this dual-output phase detector to be useful within a phase-locked loop system, we need to convert its twin pulsing outputs into an analog voltage signal that may be low-pass filtered and then used to drive the voltage-controlled oscillator (VCO) to either a higher or a lower frequency if the two pulse signals are not phase-aligned. A simple way to do this is to couple output X through a diode and inverted output Y through another diode so that a "high" signal on X may source current to a capacitor while a "high" signal on Y (i.e. a "low" signal on $\overline{Y}$) may sink current from that same capacitor. In this way, the varying pulse durations of X and Y drive the capacitor's DC voltage up and down, respectively:



The RC time constant of the capacitor network must be set such that several cycles of imbalance between the two pulse signals are necessary to saturate the capacitor's voltage at either fully +V or ground potential. In other words, the RC network produces a *gradual* increase or decrease of average capacitor voltage that can drive the voltage-controlled oscillator to achieve a lock condition.

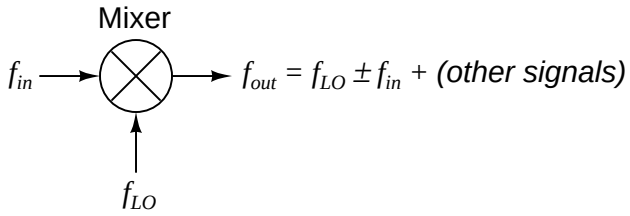
PLLs using this type of flip-flop-based phase detector do not suffer the anomalous positive-feedback behavior inherent to PLLs using XOR phase detectors. This means a PLL equipped with a flip-flop detector is actually able to synchronize the two signals together in phase and not just in frequency. The flip-flop phase detector is also not fooled by waveforms having different duty cycles from each other, as the XOR detector is. However, since flip-flop phase detectors function

by detecting differences in time between the *edges* of two pulse signals they are more susceptible to noise than XOR-based phase detectors.

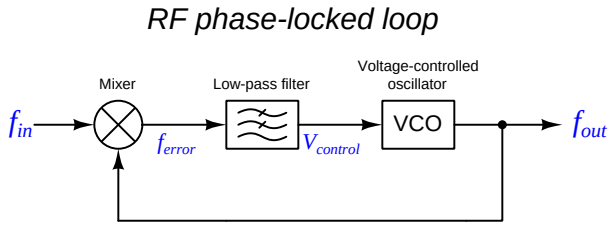
3.4.3 Mixer phase detectors

Phase-locked loops (PLLs) find a great many applications within radio-frequency systems, for example in demodulating FM radio signals. In such applications where the signals in question are usually quite weak and usually not square-wave in shape we cannot utilize logic-gate technology to detect phase shift. Instead, we must use circuitry more suitable to the signal type, so for RF signals it is common to use *mixers*.

A "mixer" is a circuit that multiplies the instantaneous voltage values of two AC signals to obtain an output signal. The trigonometric identity $\cos x \cos y = \frac{\cos(x-y)+\cos(x+y)}{2}$ informs us that two sinusoidal signals of differing frequency will produce two new frequencies when multiplied together, one that is the sum of the two mixed signals' frequencies $(x+y)$ and one that is the difference $(x-y)$. Therefore, when an input signal is mixed with the output signal of a local oscillator (LO), the result will be sum and difference frequencies, as well as some other artifact frequencies usually present in the output of non-ideal mixer circuits:



If the input and local oscillator frequencies sent to a mixer happen to be precisely the same frequency, then the sum frequency will be twice their value and the difference frequency will be zero. A frequency of zero simply means DC. This means that a mixer's output will contain DC when there is a match in the two input frequency values, and this in turn means that a mixer may be used as a phase detector for radio-signal PLLs. The low-pass filter which is a standard feature of all PLLs serves nicely to screen out all signals from the mixer's output that are not DC:

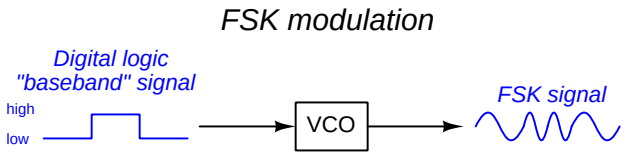


The DC voltage level present in the filtered mixer output signal is a function of phase shift. When the two input waveforms are perfectly in-phase with each other the DC voltage will be positive in sign, but when they are 180^o apart in phase the filtered DC output will be negative. A phase shift of exactly 90^o results in zero DC voltage. Like the XOR-based phase detector, a mixer cannot discern which signal is leading or lagging, and therefore mixer-based PLLs are susceptible to *cycle slip*.

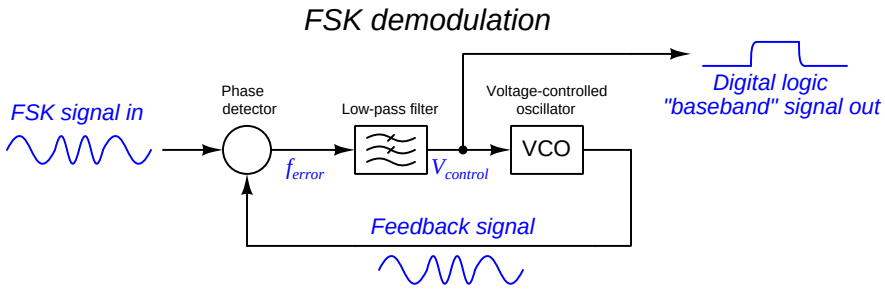
3.5 PLL frequency detection

Phase-locked loops work well as frequency detectors: determining when a signal's frequency has reached some specific value. The $V_{control}$ signal between the PLL's low-pass filter and its VCO will rise and fall as the input frequency f_{in} rises and falls, and so any analog comparator circuit set to detect a particular voltage level for $V_{control}$ will by extension detect when a particular frequency value is reached in f_{in}.

A practical application for this is demodulation of frequency shift-keyed (FSK) signals. An FSK signal is one where high/low digital logic states are encoded as two different AC frequencies instead of two different DC voltage levels. FSK signaling is often used in applications where DC voltages cannot be communicated over long distances, for example via analog telephone systems designed to convey human speech which occupies a narrow range of the audible spectrum. Modulating a digital logic signal into FSK format is as simple as making that digital signal the input to a VCO, where a "low" logic state produces one output frequency value and a "high" logic state produces a different output frequency:



If an FSK signal is fed into a phase-locked loop, the $V_{control}$ signal will exhibit the demodulated digital data:



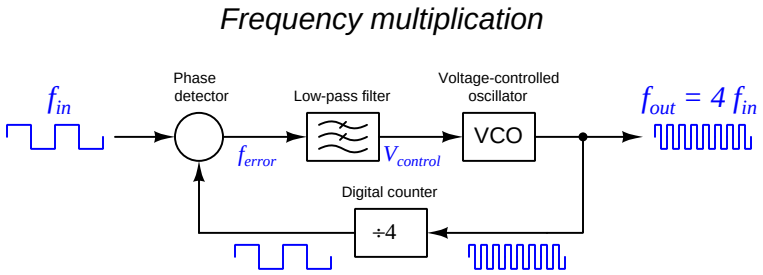
As the original digital signal toggles between high and low states, the modulator's FSK output switches between high and low frequencies. At the receiving end, our PLL circuit detects these frequency values and represents them in the form of a signal toggling between high and low voltage states – i.e. the digital data *demodulated* from FSK.

Another practical application of phase-locked loops is in Ethernet communication networks, where the *preamble* starting section of every Ethernet message frame (a series of alternating bit states: `101010...10101011`) is used to synchronize all receivers' clocks to the timing of the transmitter. The relatively long length (64 bits) of this preamble section give the receivers' PLLs time to lock in to the clock rate of the transmitter.

3.6 PLL frequency multiplication

We know from negative feedback theory that any mathematical function placed within the negative-feedback loop will become inverted from the system's input to its output. We call this *functional inversion* and it is what accounts for opamp amplifier circuits having a *gain* value as a result of the feedback signal being *attenuated*.

Since phase-locked loops operate using negative feedback, the functional inversion works here as well. In the case of a PLL, the function in question is one manipulating the *frequency* of the fed-back signal before being compared against f_{in}. Digital *counter* circuits are known to divide frequency by whole-number ratios, and so if we were to place such a frequency divider into the feedback path of a PLL the result would be f_{out} being that *multiple* of f_{in}. The following example shows this in action with a *divide-by-four* counter:



With the 4:1 frequency-divider placed within the PLL's feedback loop, the output frequency of the VCO must be exactly four times that of the input received by the phase detector, once the PLL has achieved a "locked" state.

This frequency-multiplying technique is often used on-board microprocessors, FPGAs and other high-speed digital ICs to generate internal clock signals that are some precise multiple of an external clock signal sent to that IC from other components on a printed circuit board. Such internal frequency-multiplied clock signals are necessary when the digital IC in question must index through multiple steps or operations for every one pulse of the lower-frequency external clock signal sent to it.

If the digital counter's frequency-division ratio is adjustable[3], then the circuit becomes a *frequency synthesizer* capable of outputting any specified harmonic of the input frequency.

[3] The frequency division ratio, or *modulus*, of a digital counter is the total number of allowed binary output states. For example, a four-bit binary counter naturally has a modulus of sixteen, since there are sixteen possible binary output values it may produce (`0b0000` through `0b1111`). Thus, a four-bit binary counter completes one full cycle of its count sequence for every sixteen cycles of the clock signal. However, by adding external logic to a digital counter either resetting its count to zero prematurely, or presetting its count to some non-zero value rather than letting it "roll around" to zero naturally, we may truncate its count range and therefore its modulus.

Chapter 4

Questions

This learning module, along with all others in the ModEL collection, is designed to be used in an inverted instructional environment where students independently read[1] the tutorials and attempt to answer questions on their own *prior* to the instructor's interaction with them. In place of lecture[2], the instructor engages with students in Socratic-style dialogue, probing and challenging their understanding of the subject matter through inquiry.

Answers are not provided for questions within this chapter, and this is by design. Solved problems may be found in the Tutorial and Derivation chapters, instead. The goal here is *independence*, and this requires students to be challenged in ways where others cannot think for them. Remember that you always have the tools of *experimentation* and *computer simulation* (e.g. SPICE) to explore concepts!

The following lists contain ideas for Socratic-style questions and challenges. Upon inspection, one will notice a strong theme of *metacognition* within these statements: they are designed to foster a regular habit of examining one's own thoughts as a means toward clearer thinking. As such these sample questions are useful both for instructor-led discussions as well as for self-study.

[1] Technical reading is an essential academic skill for any technical practitioner to possess for the simple reason that the most comprehensive, accurate, and useful information to be found for developing technical competence is in textual form. Technical careers in general are characterized by the need for continuous learning to remain current with standards and technology, and therefore any technical practitioner who cannot read well is handicapped in their professional development. An excellent resource for educators on improving students' reading prowess through intentional effort and strategy is the book textitReading For Understanding – How Reading Apprenticeship Improves Disciplinary Learning in Secondary and College Classrooms by Ruth Schoenbach, Cynthia Greenleaf, and Lynn Murphy.

[2] Lecture is popular as a teaching method because it is easy to implement: any reasonably articulate subject matter expert can talk to students, even with little preparation. However, it is also quite problematic. A good lecture always makes complicated concepts seem easier than they are, which is bad for students because it instills a false sense of confidence in their own understanding; reading and re-articulation requires more cognitive effort and serves to verify comprehension. A culture of teaching-by-lecture fosters a debilitating dependence upon direct personal instruction, whereas the challenges of modern life demand independent and critical thought made possible only by gathering information and perspectives from afar. Information presented in a lecture is ephemeral, easily lost to failures of memory and dictation; text is forever, and may be referenced at any time.

General challenges following tutorial reading

- Summarize as much of the text as you can in one paragraph of your own words. A helpful strategy is to explain ideas as you would for an intelligent child: as simple as you can without compromising too much accuracy.
- Simplify a particular section of the text, for example a paragraph or even a single sentence, so as to capture the same fundamental idea in fewer words.
- Where did the text make the most sense to you? What was it about the text's presentation that made it clear?
- Identify where it might be easy for someone to misunderstand the text, and explain why you think it could be confusing.
- Identify any new concept(s) presented in the text, and explain in your own words.
- Identify any familiar concept(s) such as physical laws or principles applied or referenced in the text.
- Devise a proof of concept experiment demonstrating an important principle, physical law, or technical innovation represented in the text.
- Devise an experiment to disprove a plausible misconception.
- Did the text reveal any misconceptions you might have harbored? If so, describe the misconception(s) and the reason(s) why you now know them to be incorrect.
- Describe any useful problem-solving strategies applied in the text.
- Devise a question of your own to challenge a reader's comprehension of the text.

General follow-up challenges for assigned problems

- Identify where any fundamental laws or principles apply to the solution of this problem, especially before applying any mathematical techniques.
- Devise a thought experiment to explore the characteristics of the problem scenario, applying known laws and principles to mentally model its behavior.
- Describe in detail your own strategy for solving this problem. How did you identify and organized the given information? Did you sketch any diagrams to help frame the problem?
- Is there more than one way to solve this problem? Which method seems best to you?
- Show the work you did in solving this problem, even if the solution is incomplete or incorrect.
- What would you say was the most challenging part of this problem, and why was it so?
- Was any important information missing from the problem which you had to research or recall?
- Was there any extraneous information presented within this problem? If so, what was it and why did it not matter?
- Examine someone else's solution to identify where they applied fundamental laws or principles.
- Simplify the problem from its given form and show how to solve this simpler version of it. Examples include eliminating certain variables or conditions, altering values to simpler (usually whole) numbers, applying a limiting case (i.e. altering a variable to some extreme or ultimate value).
- For quantitative problems, identify the real-world meaning of all intermediate calculations: their units of measurement, where they fit into the scenario at hand. Annotate any diagrams or illustrations with these calculated values.
- For quantitative problems, try approaching it qualitatively instead, thinking in terms of "increase" and "decrease" rather than definite values.
- For qualitative problems, try approaching it quantitatively instead, proposing simple numerical values for the variables.
- Were there any assumptions you made while solving this problem? Would your solution change if one of those assumptions were altered?
- Identify where it would be easy for someone to go astray in attempting to solve this problem.
- Formulate your own problem based on what you learned solving this one.

General follow-up challenges for experiments or projects

- In what way(s) was this experiment or project easy to complete?
- Identify some of the challenges you faced in completing this experiment or project.

- Show how thorough documentation assisted in the completion of this experiment or project.
- Which fundamental laws or principles are key to this system's function?
- Identify any way(s) in which one might obtain false or otherwise misleading measurements from test equipment in this system.
- What will happen if (component X) fails (open/shorted/etc.)?
- What would have to occur to make this system unsafe?

4.1 Conceptual reasoning

These questions are designed to stimulate your analytic and synthetic thinking[3]. In a Socratic discussion with your instructor, the goal is for these questions to prompt an extended dialogue where assumptions are revealed, conclusions are tested, and understanding is sharpened. Your instructor may also pose additional questions based on those assigned, in order to further probe and refine your conceptual understanding.

Questions that follow are presented to challenge and probe your understanding of various concepts presented in the tutorial. These questions are intended to serve as a guide for the Socratic dialogue between yourself and the instructor. Your instructor's task is to ensure you have a sound grasp of these concepts, and the questions contained in this document are merely a means to this end. Your instructor may, at his or her discretion, alter or substitute questions for the benefit of tailoring the discussion to each student's needs. The only absolute requirement is that each student is challenged and assessed at a level equal to or greater than that represented by the documented questions.

It is far more important that you convey your *reasoning* than it is to simply convey a correct answer. For this reason, you should refrain from researching other information sources to answer questions. What matters here is that *you* are doing the thinking. If the answer is incorrect, your instructor will work with you to correct it through proper reasoning. A correct answer without an adequate explanation of how you derived that answer is unacceptable, as it does not aid the learning or assessment process.

You will note a conspicuous lack of answers given for these conceptual questions. Unlike standard textbooks where answers to every other question are given somewhere toward the back of the book, here in these learning modules students must rely on other means to check their work. The best way by far is to debate the answers with fellow students and also with the instructor during the Socratic dialogue sessions intended to be used with these learning modules. Reasoning through challenging questions with other people is an excellent tool for developing strong reasoning skills.

Another means of checking your conceptual answers, where applicable, is to use circuit simulation software to explore the effects of changes made to circuits. For example, if one of these conceptual questions challenges you to predict the effects of altering some component parameter in a circuit, you may check the validity of your work by simulating that same parameter change within software and seeing if the results agree.

[3] *Analytical* thinking involves the "disassembly" of an idea into its constituent parts, analogous to dissection. *Synthetic* thinking involves the "assembly" of a new idea comprised of multiple concepts, analogous to construction. Both activities are high-level cognitive skills, extremely important for effective problem-solving, necessitating frequent challenge and regular practice to fully develop.

4.1.1 Reading outline and reflections

"Reading maketh a full man; conference a ready man; and writing an exact man" – Francis Bacon

Francis Bacon's advice is a blueprint for effective education: reading provides the learner with knowledge, writing focuses the learner's thoughts, and critical dialogue equips the learner to confidently communicate and apply their learning. Independent acquisition and application of knowledge is a powerful skill, well worth the effort to cultivate. To this end, students should read these educational resources closely, journal their own reflections on the reading, and discuss in detail their findings with classmates and instructor(s). You should be able to do all of the following after reading any instructional text:

☑ Briefly SUMMARIZE THE TEXT in the form of a journal entry documenting your learning as you progress through the course of study. Share this summary in dialogue with your classmates and instructor. Journaling is an excellent self-test of thorough reading because you cannot clearly express what you have not read or did not comprehend.

☑ Demonstrate ACTIVE READING STRATEGIES, including verbalizing your impressions as you read, simplifying long passages to convey the same ideas using fewer words, annotating text and illustrations with your own interpretations, working through mathematical examples shown in the text, cross-referencing passages with relevant illustrations and/or other passages, identifying problem-solving strategies applied by the author, etc. Technical reading is a special case of problem-solving, and so these strategies work precisely because they help solve any problem: paying attention to your own thoughts (metacognition), eliminating unnecessary complexities, identifying what makes sense, paying close attention to details, drawing connections between separated facts, and noting the successful strategies of others.

☑ Identify IMPORTANT THEMES, especially GENERAL LAWS and PRINCIPLES, expounded in the text and express them in the simplest of terms as though you were teaching an intelligent child. This emphasizes connections between related topics and develops your ability to communicate complex ideas to anyone.

☑ Form YOUR OWN QUESTIONS based on the reading, and then pose them to your instructor and classmates for their consideration. Anticipate both correct and incorrect answers, the incorrect answer(s) assuming one or more plausible misconceptions. This helps you view the subject from different perspectives to grasp it more fully.

☑ Devise EXPERIMENTS to test claims presented in the reading, or to disprove misconceptions. Predict possible outcomes of these experiments, and evaluate their meanings: what result(s) would confirm, and what would constitute disproof? Running mental simulations and evaluating results is essential to scientific and diagnostic reasoning.

☑ Specifically identify any points you found CONFUSING. The reason for doing this is to help diagnose misconceptions and overcome barriers to learning.

4.1.2 Foundational concepts

Correct analysis and diagnosis of electric circuits begins with a proper understanding of some basic concepts. The following is a list of some important concepts referenced in this module's full tutorial. Define each of them in your own words, and be prepared to illustrate each of these concepts with a description of a practical example and/or a live demonstration.

Energy

Conservation of Energy

Simplification as a problem-solving strategy

Thought experiments as a problem-solving strategy

Limiting cases as a problem-solving strategy

Annotating diagrams as a problem-solving strategy

Interpreting intermediate results as a problem-solving strategy

Graphing as a problem-solving strategy

Converting a qualitative problem into a quantitative problem

Converting a quantitative problem into a qualitative problem

Working "backwards" to validate calculated results

Reductio ad absurdum

Re-drawing schematics as a problem-solving strategy

Cut-and-try problem-solving strategy

Algebraic substitution

???

4.1.3 Simple musical synthesizer design

Sketch a schematic diagram for a simple music synthesizer where one potentiometer determines the musical pitch (frequency) while another determines the volume, using the VCO of a model CD4046 IC.

Equip your synthesizer design with a switch that doubles the pitch when actuated.

Challenges

- Describe how to create a piano-like keyboard where every key is a switch initiating a different frequency.

4.1.4 CD4046B phase-locked loop applications

The Texas Instruments Application Report titled "CD4046B Phase-Locked Loop: A Versatile Building Block for Micropower Digital and Analog Applications" by David K. Morgan (TI document number SCHA002A, February 2003) is an excellent introduction to phase-locked loops in addition to a good guide on the use of the CD4046B integrated circuit (IC). A noteworthy pair of typographical errors[4] in this document revolve around the formula presented for calculating the PLL's *capture frequency* (f_C) based on the known *lock frequency* (f_L) along with the R_3 and C_2 values used in the simple version of the low-pass filter network, where $\tau_1 = R_3C_2$:

Table 4 formula:

As shown $2f_C = \dfrac{1}{\tau_1}\sqrt{\dfrac{2\pi f_L}{\tau_1}}$ **Correct** $2f_C = \dfrac{1}{\pi}\sqrt{\dfrac{2\pi f_L}{\tau_1}}$

FM demodulator, page 18:

As shown $$f_C \approx \pm\left(\frac{1}{2\pi}\right)\left(\frac{2\pi f_1}{R_3C_2}\right) = \pm 0.4 \text{ kHz}$$

Correct $$f_C \approx \pm\left(\frac{1}{2\pi}\right)\sqrt{\frac{2\pi f_L}{R_3C_2}} = \pm 0.4 \text{ kHz}$$

Reference this Application Report to answer the following questions:

- Does this PLL contain an internal low-pass filter, or must you connect your own external filter to the IC?
- Which of the phase detectors within this IC is simplest in its operation, and how does it function?
- What factors determine the output frequency of this IC's VCO?
- Which component values determine the VCO's center frequency?
- Which component values determine the VCO's frequency range?
- Describe some the useful design information found in Table 4 (page 15)
- The FM demodulation application described in section 4.1 gives component values suitable for that application. Explain how each of those values was determined.

Challenges

[4] These errors look as though someone unfamiliar with the mathematics did the typesetting from a hand-written document to a computer-based document. For example, mis-interpreting π as τ_1.

- What consequences might result from choosing a low-pass filter cutoff value that is lower than ideal?
- Why is power supply voltage one of the factors influencing VCO output frequency?

4.1.5 Tone-controlled switching system

Suppose you wish to build a remote-control system to turn on a discrete (on/off) load such as a small electric motor using radio communication. You have access to simple pre-built transmitter and receiver units capable of conveying an audio-frequency tone, and you also have a pre-built oscillator circuit designed to output a 1.5 kHz square-wave signal whenever a pushbutton switch is pressed.

The audio oscillator circuit has five terminals on it:

- +DC power
- −DC power (Ground)
- Signal output (measured between this terminal and Ground)
- One side of the switch
- The other side of the switch

Each radio unit has four terminals on it:

- +DC power
- −DC power (Ground)
- Antenna
- Signal in/out (input for the radio transmitter, output for the radio receiver)

Sketch a schematic diagram of a complete system using these radio transceivers and the audio oscillator to control a small DC load, a phase-locked loop IC at the receiving end detecting when the 1.5 kHz audio tone is detected.

Challenges

- Identify challenges for designing and testing this system.

4.2 Quantitative reasoning

These questions are designed to stimulate your computational thinking. In a Socratic discussion with your instructor, the goal is for these questions to reveal your mathematical approach(es) to problem-solving so that good technique and sound reasoning may be reinforced. Your instructor may also pose additional questions based on those assigned, in order to observe your problem-solving firsthand.

Mental arithmetic and estimations are strongly encouraged for all calculations, because without these abilities you will be unable to readily detect errors caused by calculator misuse (e.g. keystroke errors).

You will note a conspicuous lack of answers given for these quantitative questions. Unlike standard textbooks where answers to every other question are given somewhere toward the back of the book, here in these learning modules students must rely on other means to check their work. My advice is to use circuit simulation software such as SPICE to check the correctness of quantitative answers. Refer to those learning modules within this collection focusing on SPICE to see worked examples which you may use directly as practice problems for your own study, and/or as templates you may modify to run your own analyses and generate your own practice problems.

Completely worked example problems found in the Tutorial may also serve as "test cases[5]" for gaining proficiency in the use of circuit simulation software, and then once that proficiency is gained you will never need to rely[6] on an answer key!

[5] In other words, set up the circuit simulation software to analyze the same circuit examples found in the Tutorial. If the simulated results match the answers shown in the Tutorial, it confirms the simulation has properly run. If the simulated results disagree with the Tutorial's answers, something has been set up incorrectly in the simulation software. Using every Tutorial as practice in this way will quickly develop proficiency in the use of circuit simulation software.

[6] This approach is perfectly in keeping with the instructional philosophy of these learning modules: *teaching students to be self-sufficient thinkers*. Answer keys can be useful, but it is even more useful to your long-term success to have a set of tools on hand for checking your own work, because once you have left school and are on your own, there will no longer be "answer keys" available for the problems you will have to solve.

4.2.1 Miscellaneous physical constants

Note: constants shown in **bold** type are *exact*, not approximations. Values inside of parentheses show one standard deviation (σ) of uncertainty in the final digits: for example, the magnetic permeability of free space value given as $1.25663706212(19) \times 10^{-6}$ H/m represents a center value (i.e. the location parameter) of $1.25663706212 \times 10^{-6}$ Henrys per meter with one standard deviation of uncertainty equal to $0.0000000000019 \times 10^{-6}$ Henrys per meter.

Avogadro's number (N_A) = **6.02214076** $\times 10^{23}$ **per mole** (mol^{-1})

Boltzmann's constant (k) = **1.380649** $\times 10^{-23}$ **Joules per Kelvin** (J/K)

Electronic charge (e) = **1.602176634** $\times 10^{-19}$ **Coulomb** (C)

Faraday constant (F) = **96,485.33212...** $\times 10^{4}$ **Coulombs per mole** (C/mol)

Magnetic permeability of free space (μ_0) = $1.25663706212(19) \times 10^{-6}$ Henrys per meter (H/m)

Electric permittivity of free space (ϵ_0) = $8.8541878128(13) \times 10^{-12}$ Farads per meter (F/m)

Characteristic impedance of free space (Z_0) = 376.730313668(57) Ohms (Ω)

Gravitational constant (G) = $6.67430(15) \times 10^{-11}$ cubic meters per kilogram-seconds squared ($m^3/kg\text{-}s^2$)

Molar gas constant (R) = **8.314462618... Joules per mole-Kelvin** (J/mol-K) = 0.08205746(14) liters-atmospheres per mole-Kelvin

Planck constant (h) = **6.62607015** $\times 10^{-34}$ **joule-seconds** (J-s)

Stefan-Boltzmann constant (σ) = **5.670374419...** $\times 10^{-8}$ **Watts per square meter-Kelvin**^{4} ($W/m^2 \cdot K^4$)

Speed of light in a vacuum (c) = **299,792,458 meters per second** (m/s) = 186282.4 miles per second (mi/s)

Note: All constants taken from NIST data "Fundamental Physical Constants – Complete Listing", from `http://physics.nist.gov/constants`, National Institute of Standards and Technology (NIST), 2018 CODATA Adjustment.

4.2.2 Introduction to spreadsheets

A powerful computational tool you are encouraged to use in your work is a *spreadsheet*. Available on most personal computers (e.g. Microsoft Excel), *spreadsheet* software performs numerical calculations based on number values and formulae entered into cells of a grid. This grid is typically arranged as lettered columns and numbered rows, with each cell of the grid identified by its column/row coordinates (e.g. cell `B3`, cell `A8`). Each cell may contain a string of text, a number value, or a mathematical formula. The spreadsheet automatically updates the results of all mathematical formulae whenever the entered number values are changed. This means it is possible to set up a spreadsheet to perform a series of calculations on entered data, and those calculations will be re-done by the computer any time the data points are edited in any way.

For example, the following spreadsheet calculates average speed based on entered values of distance traveled and time elapsed:

	A	B	C	D
1	Distance traveled	46.9	Kilometers	
2	Time elapsed	1.18	Hours	
3	Average speed	= B1 / B2	km/h	
4				
5				

Text labels contained in cells `A1` through `A3` and cells `C1` through `C3` exist solely for readability and are not involved in any calculations. Cell `B1` contains a sample distance value while cell `B2` contains a sample time value. The formula for computing speed is contained in cell `B3`. Note how this formula begins with an "equals" symbol (`=`), references the values for distance and speed by lettered column and numbered row coordinates (`B1` and `B2`), and uses a forward slash symbol for division (`/`). The coordinates `B1` and `B2` function as *variables*[7] would in an algebraic formula.

When this spreadsheet is executed, the numerical value `39.74576` will appear in cell `B3` rather than the formula `= B1 / B2`, because 39.74576 is the computed speed value given 46.9 kilometers traveled over a period of 1.18 hours. If a different numerical value for distance is entered into cell `B1` or a different value for time is entered into cell `B2`, cell `B3`'s value will automatically update. All you need to do is set up the given values and any formulae into the spreadsheet, and the computer will do all the calculations for you.

Cell `B3` may be referenced by other formulae in the spreadsheet if desired, since it is a variable just like the given values contained in `B1` and `B2`. This means it is possible to set up an entire chain of calculations, one dependent on the result of another, in order to arrive at a final value. The arrangement of the given data and formulae need not follow any pattern on the grid, which means you may place them anywhere.

[7] Spreadsheets may also provide means to attach text labels to cells for use as variable names (Microsoft Excel simply calls these labels "names"), but for simple spreadsheets such as those shown here it's usually easier just to use the standard coordinate naming for each cell.

Common[8] arithmetic operations available for your use in a spreadsheet include the following:

- Addition (`+`)
- Subtraction (`-`)
- Multiplication (`*`)
- Division (`/`)
- Powers (`^`)
- Square roots (`sqrt()`)
- Logarithms (`ln()` , `log10()`)

Parentheses may be used to ensure[9] proper order of operations within a complex formula. Consider this example of a spreadsheet implementing the *quadratic formula*, used to solve for roots of a polynomial expression in the form of $ax^2 + bx + c$:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

	A	B
1	x_1	= (-B4 + sqrt((B4^2) - (4*B3*B5))) / (2*B3)
2	x_2	= (-B4 - sqrt((B4^2) - (4*B3*B5))) / (2*B3)
3	a =	9
4	b =	5
5	c =	-2

This example is configured to compute roots[10] of the polynomial $9x^2 + 5x - 2$ because the values of 9, 5, and -2 have been inserted into cells `B3`, `B4`, and `B5`, respectively. Once this spreadsheet has been built, though, it may be used to calculate the roots of *any* second-degree polynomial expression simply by entering the new a, b, and c coefficients into cells `B3` through `B5`. The numerical values appearing in cells `B1` and `B2` will be automatically updated by the computer immediately following any changes made to the coefficients.

[8] Modern spreadsheet software offers a bewildering array of mathematical functions you may use in your computations. I recommend you consult the documentation for your particular spreadsheet for information on operations other than those listed here.

[9] Spreadsheet programs, like text-based programming languages, are designed to follow standard order of operations by default. However, my personal preference is to use parentheses even where strictly unnecessary just to make it clear to any other person viewing the formula what the intended order of operations is.

[10] Reviewing some algebra here, a *root* is a value for x that yields an overall value of zero for the polynomial. For this polynomial ($9x^2 + 5x - 2$) the two roots happen to be $x = 0.269381$ and $x = -0.82494$, with these values displayed in cells `B1` and `B2`, respectively upon execution of the spreadsheet.

Alternatively, one could break up the long quadratic formula into smaller pieces like this:

$$y = \sqrt{b^2 - 4ac} \qquad z = 2a$$

$$x = \frac{-b \pm y}{z}$$

	A	B	C
1	x_1	= (-B4 + C1) / C2	= sqrt((B4^2) - (4*B3*B5))
2	x_2	= (-B4 - C1) / C2	= 2*B3
3	a =	9	
4	b =	5	
5	c =	-2	

Note how the square-root term (y) is calculated in cell C1, and the denominator term (z) in cell C2. This makes the two final formulae (in cells B1 and B2) simpler to interpret. The positioning of all these cells on the grid is completely arbitrary[11] – all that matters is that they properly reference each other in the formulae.

Spreadsheets are particularly useful for situations where the same set of calculations representing a circuit or other system must be repeated for different initial conditions. The power of a spreadsheet is that it automates what would otherwise be a tedious set of calculations. One specific application of this is to simulate the effects of various components within a circuit failing with abnormal values (e.g. a shorted resistor simulated by making its value nearly zero; an open resistor simulated by making its value extremely large). Another application is analyzing the behavior of a circuit design given new components that are out of specification, and/or aging components experiencing drift over time.

[11] My personal preference is to locate all the "given" data in the upper-left cells of the spreadsheet grid (each data point flanked by a sensible name in the cell to the left and units of measurement in the cell to the right as illustrated in the first distance/time spreadsheet example), sometimes coloring them in order to clearly distinguish which cells contain entered data versus which cells contain computed results from formulae. I like to place all formulae in cells below the given data, and try to arrange them in logical order so that anyone examining my spreadsheet will be able to figure out *how* I constructed a solution. This is a general principle I believe all computer programmers should follow: *document and arrange your code to make it easy for other people to learn from it.*

4.3 Diagnostic reasoning

These questions are designed to stimulate your deductive and inductive thinking, where you must apply general principles to specific scenarios (deductive) and also derive conclusions about the failed circuit from specific details (inductive). In a Socratic discussion with your instructor, the goal is for these questions to reinforce your recall and use of general circuit principles and also challenge your ability to integrate multiple symptoms into a sensible explanation of what's wrong in a circuit. Your instructor may also pose additional questions based on those assigned, in order to further challenge and sharpen your diagnostic abilities.

As always, your goal is to fully *explain* your analysis of each problem. Simply obtaining a correct answer is not good enough – you must also demonstrate sound reasoning in order to successfully complete the assignment. Your instructor's responsibility is to probe and challenge your understanding of the relevant principles and analytical processes in order to ensure you have a strong foundation upon which to build further understanding.

You will note a conspicuous lack of answers given for these diagnostic questions. Unlike standard textbooks where answers to every other question are given somewhere toward the back of the book, here in these learning modules students must rely on other means to check their work. The best way by far is to debate the answers with fellow students and also with the instructor during the Socratic dialogue sessions intended to be used with these learning modules. Reasoning through challenging questions with other people is an excellent tool for developing strong reasoning skills.

Another means of checking your diagnostic answers, where applicable, is to use circuit simulation software to explore the effects of faults placed in circuits. For example, if one of these diagnostic questions requires that you predict the effect of an open or a short in a circuit, you may check the validity of your work by simulating that same fault (substituting a very high resistance in place of that component for an open, and substituting a very low resistance for a short) within software and seeing if the results agree.

Appendix A

Problem-Solving Strategies

The ability to solve complex problems is arguably one of the most valuable skills one can possess, and this skill is particularly important in any science-based discipline.

- Study principles, not procedures. Don't be satisfied with merely knowing how to compute solutions – learn *why* those solutions work.
- Identify what it is you need to solve, identify all relevant data, identify all units of measurement, identify any general principles or formulae linking the given information to the solution, and then identify any "missing pieces" to a solution. Annotate all diagrams with this data.
- Sketch a diagram to help visualize the problem. When building a real system, always devise a plan for that system and analyze its function *before* constructing it.
- Follow the units of measurement and meaning of every calculation. If you are ever performing mathematical calculations as part of a problem-solving procedure, and you find yourself unable to apply each and every intermediate result to some aspect of the problem, it means you don't understand what you are doing. Properly done, every mathematical result should have practical meaning for the problem, and not just be an abstract number. You should be able to identify the proper units of measurement for each and every calculated result, and show where that result fits into the problem.
- Perform "thought experiments" to explore the effects of different conditions for theoretical problems. When troubleshooting real systems, perform *diagnostic tests* rather than visually inspecting for faults, the best diagnostic test being the one giving you the most information about the nature and/or location of the fault with the fewest steps.
- Simplify the problem until the solution becomes obvious, and then use that obvious case as a model to follow in solving the more complex version of the problem.
- Check for exceptions to see if your solution is incorrect or incomplete. A good solution will work for *all* known conditions and criteria. A good example of this is the process of testing scientific hypotheses: the task of a scientist is not to find support for a new idea, but rather to *challenge* that new idea to see if it holds up under a battery of tests. The philosophical

principle of *reductio ad absurdum* (i.e. disproving a general idea by finding a specific case where it fails) is useful here.

- Work "backward" from a hypothetical solution to a new set of given conditions.
- Add quantities to problems that are qualitative in nature, because sometimes a little math helps illuminate the scenario.
- Sketch graphs illustrating how variables relate to each other. These may be quantitative (i.e. with realistic number values) or qualitative (i.e. simply showing increases and decreases).
- Treat quantitative problems as qualitative in order to discern the relative magnitudes and/or directions of change of the relevant variables. For example, try determining what happens if a certain variable were to increase or decrease before attempting to precisely calculate quantities: how will each of the dependent variables respond, by increasing, decreasing, or remaining the same as before?
- Consider limiting cases. This works especially well for qualitative problems where you need to determine which direction a variable will change. Take the given condition and magnify that condition to an extreme degree as a way of simplifying the direction of the system's response.
- Check your work. This means regularly testing your conclusions to see if they make sense. This does *not* mean repeating the same steps originally used to obtain the conclusion(s), but rather to use some other means to check validity. Simply repeating procedures often leads to *repeating the same errors* if any were made, which is why alternative paths are better.

Appendix B

Instructional philosophy

B.1 First principles of learning

- **Anyone can learn anything** given appropriate time, effort, resources, challenges, encouragement, and expectations. Dedicating time and investing effort are the student's responsibility; providing resources, challenges, and encouragement are the teacher's responsibility; high expectations are a responsibility shared by both student and teacher.

- **Transfer is not automatic.** The human mind has a natural tendency to compartmentalize information, which means the process of taking knowledge learned in one context and applying it to another usually does not come easy and therefore should never be taken for granted.

- **Learning is iterative.** The human mind rarely learns anything perfectly on the first attempt. Anticipate mistakes and plan for multiple tries to achieve full understanding, using the lessons of those mistakes as feedback to guide future attempts.

- **Information is absorbed, but understanding is created.** Facts and procedures may be memorized easily enough by repeated exposure, but the ability to reliably apply principles to novel scenarios only comes through intense personal effort. This effort is fundamentally creative in nature: casting new concepts in one's own words, running mental experiments to perceive patterns and trends, etc. These acts of making knowledge "one's own" need not be perfect in order to be effective, as the value lies in the activity and not necessarily the finished product. Depth of understanding and permanence of memory are both directly related to mental effort during the learning process – i.e. *easy learning is shallow and fleeting.*

- **Education trumps training.** There is no such thing as an entirely isolated subject, as all fields of knowledge are connected. Training is narrowly-focused and task-oriented. Education is broad-based and principle-oriented. When preparing for a life-long technical career, education beats training every time.

- **Character matters.** Poor habits are more destructive than deficits of knowledge or skill. This is especially true in collective endeavors, where a team's ability to function depends on trust between its members. Simply put, no one wants an untrustworthy person on their team. An essential component of education then, is character development.

- **People learn to be responsible by bearing responsibility.** An irresponsible person is someone who has never *had* to be responsible for anything that mattered enough to them. Just as anyone can learn anything, anyone can become responsible if the personal cost of irresponsibility becomes high enough.

- **What gets measured, gets done.** Accurate and relevant assessment of learning is key to ensuring all students learn. Therefore, it is imperative to measure what matters.

- **Failure is nothing to fear.** Every human being fails, and fails in multiple ways at multiple times. Eventual success only happens when we don't stop trying.

B.2 Proven strategies for instructors

- Assume every student is capable of learning anything they desire given the proper conditions. Treat them as capable adults by granting real responsibility and avoiding artificial incentives such as merit or demerit points.
- Create a consistent culture of high expectations across the entire program of study. Demonstrate and encourage patience, persistence, and a healthy sense of self-skepticism. Anticipate and de-stigmatize error. Teach respect for the capabilities of others as well as respect for one's own fallibility.
- Replace lecture with "inverted" instruction, where students first encounter new concepts through reading and then spend class time in Socratic dialogue with the instructor exploring those concepts and solving problems individually. There is a world of difference between observing someone solve a problem versus actually solving a problem yourself, and so the point of this form of instruction is to place students in a position where they *cannot* passively observe.
- Require students to read extensively, write about what they learn, and dialogue with you and their peers to sharpen their understanding. Apply Francis Bacon's advice that "reading maketh a full man; conference a ready man; and writing an exact man". These are complementary activities helping students expand their confidence and abilities.
- Use artificial intelligence (AI) to challenge student understanding rather than merely provide information. Find productive ways for AI to critique students' clarity of thought and of expression, for example by employing AI as a Socratic-style interlocutor or as a reviewer of students' journals. Properly applied, AI has the ability to expand student access to critical review well outside the bounds of their instructor's reach.
- Build frequent and rapid feedback into the learning process so that students know at all times how well they are learning, to identify problems early and fix them before they grow. Model the intellectual habit of self-assessing and self-correcting your own understanding (i.e. a cognitive *feedback loop*), encouraging students to do the same.
- Use "mastery" as the standard for every assessment, which means the exam or experiment or project must be done with 100% competence in order to pass. Provide students with multiple opportunity for re-tries (different versions of the assessment every time).
- Require students to devise their own hypotheses and procedures on all experiments, so that the process is truly a scientific one. Have students assess their proposed experimental procedures for risk and devise mitigations for those risks. Let nothing be pre-designed about students' experiments other than a stated task (i.e. what principle the experiment shall test) at the start and a set of demonstrable knowledge and skill objectives at the end.
- Have students build as much of their lab equipment as possible: building power sources, building test assemblies[1], and building complete working systems (no kits!). In order to provide

[1] In the program I teach, every student builds their own "Development Board" consisting of a metal chassis with DIN rail, terminal blocks, and an AC-DC power supply of their own making which functions as a portable lab environment they can use at school as well as take home.

this same "ground-up" experience for every new student, this means either previous students take their creations with them, or the systems get disassembled in preparation for the new students, or the systems grow and evolve with each new student group.

- Incorporate external accountability for you and for your students, continuously improving the curriculum and your instructional methods based on proven results. Have students regularly network with active professionals through participation in advisory committee meetings, service projects, tours, jobshadows, internships, etc. Practical suggestions include requiring students to design and build projects for external clients (e.g. community groups, businesses, different departments within the institution), and also requiring students attend all technical advisory committee meetings and to dialogue with the industry representatives at those meetings.
- Repeatedly explore difficult-to-learn concepts across multiple courses, so that students have multiple opportunities to build their understanding.
- Relate all new concepts, whenever possible, to previous concepts and to relevant physical laws. Challenge each and every student, every day, to *reason* from concept to concept and to explain the logical connections between. Challenge students to verify their conclusions by multiple approaches (e.g. double-checking their work using different methods). Ask *"Why?"* often.
- Maintain detailed records on each student's performance and share these records privately with them. These records should include academic performance as well as professionally relevant behavioral tendencies.
- Hold mandatory "check-in" meetings between all program faculty and each new student during their first term. Offer these to all other students as an option, except for any students continuing to manifest unprofessional behaviors, poor academic performance, or who have some other need for a face-to-face meeting with faculty.
- Address problems while they are small, before they grow larger. This is equally true for tutoring technical concepts as it is for helping students build professional habits.
- Build rigorous quality control into the curriculum to ensure every student masters every important concept, and that the mastery is retained over time. This includes (1) review questions added to every exam to re-assess knowledge taught in previous terms, (2) cumulative exams at the end of every term to re-assess all important concepts back to the very beginning of the program, and (3) review assessments in practical (hands-on) coursework to ensure critically-important skills were indeed taught and are still retained. What you will find by doing this is that it actually boosts retention of students by ensuring that important knowledge gets taught and is retained over long spans of time. In the absence of such quality control, student learning and retention tends to be spotty and this contributes to drop-out and failure rates later in their education.
- Finally, *never rush learning.* Education is not a race. Give your students ample time to digest complex ideas, as you continually remind yourself of just how long it took you to achieve mastery! Long-term retention and the consistently correct application of concepts are always the result of *focused effort over long periods of time* which means there are no shortcuts to learning.

B.3 Proven strategies for students

The single most important piece of advice I have for any student of any subject is to take responsibility for your own development in all areas of life including mental development. Expecting others in your life to entirely guide your own development is a recipe for disappointment. This is just as true for students enrolled in formal learning institutions as it is for auto-didacts pursuing learning entirely on their own. Learning to think in new ways is key to being able to gainfully use information, to make informed decisions about your life, and to best serve those you care about. With this in mind, I offer the following advice to students:

- **Approach all learning as valuable**. No matter what course you take, no matter who you learn from, no matter the subject, there is something useful in every learning experience. If you don't see the value of every new experience, you are not looking closely enough!
- **Continually challenge yourself**. Let other people take shortcuts and find easy answers to easy problems. The purpose of education is to stretch your mind, in order to shape it into a more powerful tool. This doesn't come by taking the path of least resistance. An excellent analogy for an empowering education is productive physical exercise: becoming stronger, more flexible, and more persistent only comes through intense personal effort.
- **Master the use of language**. This includes reading extensively, writing every day, listening closely, and speaking articulately. To a great extent language channels and empowers thought, so the better you are at wielding language the better you will be at grasping abstract concepts and articulating them not only for your benefit but for others as well.
- **Do not limit yourself to the resources given to you**. Read books that are not on the reading list. Run experiments that aren't assigned to you. Form study groups outside of class. Take an entrepreneurial approach to your own education, as though it were a business you were building for your future benefit.
- **Express and share what you learn**. Take every opportunity to teach what you have learned to others, as this will not only help them but will also strengthen your own understanding[2].
- Realize that **no one can give you understanding**, just as no one can give you physical fitness. These both must be *built*.
- **Above all, recognize that learning is hard work, and that a certain level of frustration is unavoidable**. There are times when you will struggle to grasp some of these concepts, and that struggle is a natural thing. Take heart that it will yield with persistent and varied[3] effort, and never give up! That concepts don't immediately come to you is not a sign of something wrong, but rather of something right: that you have found a worthy challenge!

[2] On a personal note, I was surprised to learn just how much my own understanding of electronics and related subjects was strengthened by becoming a teacher. When you are tasked every day with helping other people grasp complex topics, it catalyzes your own learning by giving you powerful incentives to study, to articulate your thoughts, and to reflect deeply on the process of learning.

[3] As the old saying goes, "Insanity is trying the same thing over and over again, expecting different results." If you find yourself stumped by something in the text, you should attempt a different approach. Alter the thought experiment, change the mathematical parameters, do whatever you can to see the problem in a slightly different light, and then the solution will often present itself more readily.

B.4 Design of these learning modules

"The unexamined circuit is not worth energizing" – Socrates (if he had taught electricity)

These learning modules, although useful for self-study, were designed to be used in a formal learning environment where a subject-matter expert challenges students to digest the content and exercise their critical thinking abilities in the answering of questions and in the construction and testing of working circuits. Every effort has been made to embed the following instructional and assessment philosophies within:

- The first goal of education is to enhance clear and independent thought, in order that every student reach their fullest potential in a highly complex and inter-dependent world. Robust reasoning is *always* more important than particulars of any subject matter, because its application is universal.
- Literacy is fundamental to independent learning and thought because text continues to be the most efficient way to communicate complex ideas over space and time. Those who cannot read with ease are limited in their ability to acquire knowledge and perspective.
- Articulate communication is fundamental to work that is complex and interdisciplinary.
- Faulty assumptions and poor reasoning are best corrected through challenge, not presentation. The rhetorical technique of *reductio ad absurdum* (disproving an assertion by exposing an absurdity) works well to discipline student's minds, not only to correct the problem at hand but also to learn how to detect and correct future errors.
- Important principles should be repeatedly explored and widely applied throughout a course of study, not only to reinforce their importance and help ensure their mastery, but also to showcase the interconnectedness and utility of knowledge.

These learning modules were expressly designed to be used in an "inverted" teaching environment[4] where students first read the introductory and tutorial chapters on their own, then individually attempt to answer the questions and construct working circuits according to the experiment and project guidelines. The instructor never lectures, but instead meets regularly with each individual student to review their progress, answer questions, identify misconceptions, and challenge the student to new depths of understanding through further questioning. Regular meetings between instructor and student should resemble a Socratic[5] dialogue, where questions serve as scalpels to dissect topics and expose assumptions. The student passes each module only after consistently demonstrating their ability to logically analyze and correctly apply all major concepts in each question or project/experiment. The instructor must be vigilant in probing each student's understanding to ensure they are truly *reasoning* and not just *memorizing*. This is why "Challenge" points appear throughout, as prompts for students to think deeper about topics and as starting points for instructor queries. Sometimes these challenge points require additional knowledge that hasn't been covered in the series to answer in full. This is okay, as the major purpose of the Challenges is to stimulate analysis and synthesis on the part of each student.

The instructor must possess enough mastery of the subject matter and awareness of students' reasoning to generate their own follow-up questions to practically any student response. Even completely correct answers given by the student should be challenged by the instructor for the purpose of having students practice articulating their thoughts and defending their reasoning. Conceptual errors committed by the student should be exposed and corrected not by direct instruction, but rather by reducing the errors to an absurdity[6] through well-chosen questions and thought experiments posed by the instructor. Becoming proficient at this style of instruction requires time and dedication, but the positive effects on critical thinking for both student and instructor are spectacular.

An inspection of these learning modules reveals certain unique characteristics. One of these is a bias toward thorough explanations in the tutorial chapters. Without a live instructor to explain concepts and applications to students, the text itself must fulfill this role. This philosophy results in lengthier explanations than what you might typically find in a textbook, each step of the reasoning process fully explained, including footnotes addressing common questions and concerns students raise while learning these concepts. Each tutorial seeks to not only explain each major concept in sufficient detail, but also to explain the logic of each concept and how each may be developed

[4] In a traditional teaching environment, students first encounter new information via *lecture* from an expert, and then independently apply that information via *homework*. In an "inverted" course of study, students first encounter new information via *homework*, and then independently apply that information under the scrutiny of an expert. The expert's role in lecture is to simply *explain*, but the expert's role in an inverted session is to *challenge*, *critique*, and if necessary *explain* where gaps in understanding still exist.

[5] Socrates is a figure in ancient Greek philosophy famous for his unflinching style of questioning. Although he authored no texts, he appears as a character in Plato's many writings. The essence of Socratic philosophy is to leave no question unexamined and no point of view unchallenged. While purists may argue a topic such as electric circuits is too narrow for a true Socratic-style dialogue, I would argue that the essential thought processes involved with scientific reasoning on *any* topic are not far removed from the Socratic ideal, and that students of electricity and electronics would do very well to challenge assumptions, pose thought experiments, identify fallacies, and otherwise employ the arsenal of critical thinking skills modeled by Socrates.

[6] This rhetorical technique is known by the Latin phrase *reductio ad absurdum*. The concept is to expose errors by counter-example, since only one solid counter-example is necessary to disprove a universal claim. As an example of this, consider the common misconception among beginning students of electricity that voltage cannot exist without current. One way to apply *reductio ad absurdum* to this statement is to ask how much current passes through a fully-charged battery connected to nothing (i.e. a clear example of voltage existing without current).

from "first principles". Again, this reflects the goal of developing clear and independent thought in students' minds, by showing how clear and logical thought was used to forge each concept. Students benefit from witnessing a model of clear thinking in action, and these tutorials strive to be just that.

Another feature of these learning modules is that they do not treat topics in isolation. Rather, important concepts are introduced early in the series, and appear repeatedly as stepping-stones toward other concepts in subsequent modules. This helps to avoid the "compartmentalization" of knowledge, demonstrating the inter-connectedness of concepts and simultaneously reinforcing them. Each module is fairly complete in itself, reserving the beginning of its tutorial to a review of foundational concepts.

To high standards of education,

Tony R. Kuphaldt

Appendix C

Tools used

I am indebted to the developers of many open-source software applications in the creation of these learning modules. The following is a list of these applications with some commentary on each.

You will notice a theme common to many of these applications: a bias toward *code*. Although I am by no means an expert programmer in any computer language, I understand and appreciate the flexibility offered by code-based applications where the user (you) enters commands into a plain ASCII text file, which the software then reads and processes to create the final output. Code-based computer applications are by their very nature *extensible*, while WYSIWYG (What You See Is What You Get) applications are generally limited to whatever user interface the developer makes for you.

The `GNU/Linux` computer operating system

> There is so much to be said about Linus Torvalds' `Linux` and Richard Stallman's `GNU` project. First, to credit just these two individuals is to fail to do justice to the *mob* of passionate volunteers who contributed to make this amazing software a reality. I first learned of `Linux` back in 1996, and have been using this operating system on my personal computers almost exclusively since then. It is *free*, it is completely *configurable*, and it permits the continued use of highly efficient `Unix` applications and scripting languages (e.g. shell scripts, Makefiles, `sed`, `awk`) developed over many decades. `Linux` not only provided me with a powerful computing platform, but its open design served to inspire my life's work of creating open-source educational resources.

Bram Moolenaar's `Vim` text editor

> Writing code for any code-based computer application requires a *text editor*, which may be thought of as a word processor strictly limited to outputting plain-ASCII text files. Many good text editors exist, and one's choice of text editor seems to be a deeply personal matter within the programming world. I prefer `Vim` because it operates very similarly to `vi` which is ubiquitous on `Unix`/`Linux` operating systems, and because it may be entirely operated via keyboard (i.e. no mouse required) which makes it fast to use.

Donald Knuth's TEX typesetting system

Developed in the late 1970's and early 1980's by computer scientist extraordinaire Donald Knuth to typeset his multi-volume magnum opus *The Art of Computer Programming*, this software allows the production of formatted text for screen-viewing or paper printing, all by writing plain-text code to describe how the formatted text is supposed to appear. TEX is not just a markup language for documents, but it is also a Turing-complete programming language in and of itself, allowing useful algorithms to be created to control the production of documents. Simply put, TEX *is a programmer's approach to word processing*. Since TEX is controlled by code written in a plain-text file, this means anyone may read that plain-text file to see exactly how the document was created. This openness afforded by the code-based nature of TEX makes it relatively easy to learn how other people have created their own TEX documents. By contrast, examining a beautiful document created in a conventional WYSIWYG word processor such as Microsoft `Word` suggests nothing to the reader about *how* that document was created, or what the user might do to create something similar. As Mr. Knuth himself once quipped, conventional word processing applications should be called WYSIAYG (What You See Is *All* You Get).

Leslie Lamport's LATEX extensions to TEX

Like all true programming languages, TEX is inherently extensible. So, years after the release of TEX to the public, Leslie Lamport decided to create a massive extension allowing easier compilation of book-length documents. The result was LATEX, which is the markup language used to create all ModEL module documents. You could say that TEX is to LATEX as `C` is to `C++`. This means it is permissible to use any and all TEX commands within LATEX source code, and it all still works. Some of the features offered by LATEX that would be challenging to implement in TEX include automatic index and table-of-content creation.

Tim Edwards' `Xcircuit` drafting program

This wonderful program is what I use to create all the schematic diagrams and illustrations (but not photographic images or mathematical plots) throughout the ModEL project. It natively outputs PostScript format which is a true vector graphic format (this is why the images do not pixellate when you zoom in for a closer view), and it is so simple to use that I have never had to read the manual! Object libraries are easy to create for `Xcircuit`, being plain-text files using PostScript programming conventions. Over the years I have collected a large set of object libraries useful for drawing electrical and electronic schematics, pictorial diagrams, and other technical illustrations.

`Gimp` graphic image manipulation program

Essentially an open-source clone of Adobe's `PhotoShop`, I use `Gimp` to resize, crop, and convert file formats for all of the photographic images appearing in the ModEL modules. Although `Gimp` does offer its own scripting language (called `Script-Fu`), I have never had occasion to use it. Thus, my utilization of `Gimp` to merely crop, resize, and convert graphic images is akin to using a sword to slice bread.

`SPICE` circuit simulation program

`SPICE` is to circuit analysis as TEX is to document creation: it is a form of markup language designed to describe a certain object to be processed in plain-ASCII text. When the plain-text "source file" is compiled by the software, it outputs the final result. More modern circuit analysis tools certainly exist, but I prefer `SPICE` for the following reasons: it is *free*, it is *fast*, it is *reliable*, and it is a fantastic tool for *teaching* students of electricity and electronics how to write simple code. I happen to use rather old versions of `SPICE`, version 2g6 being my "go to" application when I only require text-based output. `NGSPICE` (version 26), which is based on Berkeley `SPICE` version 3f5, is used when I require graphical output for such things as time-domain waveforms and Bode plots. In all `SPICE` example netlists I strive to use coding conventions compatible with all `SPICE` versions.

Andrew D. Hwang's `ePiX` mathematical visualization programming library

This amazing project is a `C++` library you may link to any `C/C++` code for the purpose of generating PostScript graphic images of mathematical functions. As a completely free and open-source project, it does all the plotting I would otherwise use a Computer Algebra System (CAS) such as `Mathematica` or `Maple` to do. It should be said that `ePiX` is *not* a Computer Algebra System like `Mathematica` or `Maple`, but merely a mathematical *visualization* tool. In other words, it won't determine integrals for you (you'll have to implement that in your own `C/C++` code!), but it can graph the results, and it does so beautifully. What I really admire about `ePiX` is that it is a `C++` programming library, which means it builds on the existing power and toolset available with that programming language. Mr. Hwang could have probably developed his own stand-alone application for mathematical plotting, but by creating a `C++` library to do the same thing he accomplished something much greater.

`gnuplot` mathematical visualization software

Another open-source tool for mathematical visualization is `gnuplot`. Interestingly, this tool is *not* part of Richard Stallman's GNU project, its name being a coincidence. For this reason the authors prefer "gnu" *not* be capitalized at all to avoid confusion. This is a much "lighter-weight" alternative to a spreadsheet for plotting tabular data, and the fact that it easily outputs directly to an X11 console or a file in a number of different graphical formats (including PostScript) is very helpful. I typically set my `gnuplot` output format to default (X11 on my Linux PC) for quick viewing while I'm developing a visualization, then switch to PostScript file export once the visual is ready to include in the document(s) I'm writing. As with my use of `Gimp` to do rudimentary image editing, my use of `gnuplot` only scratches the surface of its capabilities, but the important points are that it's *free* and that it *works well*.

`Python` programming language

Both Python and C++ find extensive use in these modules as instructional aids and exercises, but I'm listing Python here as a *tool* for myself because I use it almost daily as a *calculator*. If you open a Python interpreter console and type `from math import *` you can type mathematical expressions and have it return results just as you would on a hand calculator. Complex-number (i.e. *phasor*) arithmetic is similarly supported if you include the complex-math library (`from cmath import *`). Examples of this are shown in the Programming References chapter (if included) in each module. Of course, being a fully-featured programming language, Python also supports conditionals, loops, and other structures useful for calculation of quantities. Also, running in a console environment where all entries and returned values show as text in a chronologically-ordered list makes it easy to copy-and-paste those calculations to document exactly how they were performed.

Appendix D

Creative Commons License

Creative Commons Attribution 4.0 International Public License

By exercising the Licensed Rights (defined below), You accept and agree to be bound by the terms and conditions of this Creative Commons Attribution 4.0 International Public License ("Public License"). To the extent this Public License may be interpreted as a contract, You are granted the Licensed Rights in consideration of Your acceptance of these terms and conditions, and the Licensor grants You such rights in consideration of benefits the Licensor receives from making the Licensed Material available under these terms and conditions.

Section 1 – Definitions.

a. **Adapted Material** means material subject to Copyright and Similar Rights that is derived from or based upon the Licensed Material and in which the Licensed Material is translated, altered, arranged, transformed, or otherwise modified in a manner requiring permission under the Copyright and Similar Rights held by the Licensor. For purposes of this Public License, where the Licensed Material is a musical work, performance, or sound recording, Adapted Material is always produced where the Licensed Material is synched in timed relation with a moving image.

b. **Adapter's License** means the license You apply to Your Copyright and Similar Rights in Your contributions to Adapted Material in accordance with the terms and conditions of this Public License.

c. **Copyright and Similar Rights** means copyright and/or similar rights closely related to copyright including, without limitation, performance, broadcast, sound recording, and Sui Generis Database Rights, without regard to how the rights are labeled or categorized. For purposes of this Public License, the rights specified in Section 2(b)(1)-(2) are not Copyright and Similar Rights.

d. **Effective Technological Measures** means those measures that, in the absence of proper authority, may not be circumvented under laws fulfilling obligations under Article 11 of the WIPO Copyright Treaty adopted on December 20, 1996, and/or similar international agreements.

e. **Exceptions and Limitations** means fair use, fair dealing, and/or any other exception or

limitation to Copyright and Similar Rights that applies to Your use of the Licensed Material.

f. **Licensed Material** means the artistic or literary work, database, or other material to which the Licensor applied this Public License.

g. **Licensed Rights** means the rights granted to You subject to the terms and conditions of this Public License, which are limited to all Copyright and Similar Rights that apply to Your use of the Licensed Material and that the Licensor has authority to license.

h. **Licensor** means the individual(s) or entity(ies) granting rights under this Public License.

i. **Share** means to provide material to the public by any means or process that requires permission under the Licensed Rights, such as reproduction, public display, public performance, distribution, dissemination, communication, or importation, and to make material available to the public including in ways that members of the public may access the material from a place and at a time individually chosen by them.

j. **Sui Generis Database Rights** means rights other than copyright resulting from Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, as amended and/or succeeded, as well as other essentially equivalent rights anywhere in the world.

k. **You** means the individual or entity exercising the Licensed Rights under this Public License. **Your** has a corresponding meaning.

Section 2 – Scope.

a. License grant.

1. Subject to the terms and conditions of this Public License, the Licensor hereby grants You a worldwide, royalty-free, non-sublicensable, non-exclusive, irrevocable license to exercise the Licensed Rights in the Licensed Material to:

A. reproduce and Share the Licensed Material, in whole or in part; and

B. produce, reproduce, and Share Adapted Material.

2. Exceptions and Limitations. For the avoidance of doubt, where Exceptions and Limitations apply to Your use, this Public License does not apply, and You do not need to comply with its terms and conditions.

3. Term. The term of this Public License is specified in Section 6(a).

4. Media and formats; technical modifications allowed. The Licensor authorizes You to exercise the Licensed Rights in all media and formats whether now known or hereafter created, and to make technical modifications necessary to do so. The Licensor waives and/or agrees not to assert any right or authority to forbid You from making technical modifications necessary to exercise the Licensed Rights, including technical modifications necessary to circumvent Effective Technological Measures.

For purposes of this Public License, simply making modifications authorized by this Section 2(a)(4) never produces Adapted Material.

5. Downstream recipients.

A. Offer from the Licensor – Licensed Material. Every recipient of the Licensed Material automatically receives an offer from the Licensor to exercise the Licensed Rights under the terms and conditions of this Public License.

B. No downstream restrictions. You may not offer or impose any additional or different terms or conditions on, or apply any Effective Technological Measures to, the Licensed Material if doing so restricts exercise of the Licensed Rights by any recipient of the Licensed Material.

6. No endorsement. Nothing in this Public License constitutes or may be construed as permission to assert or imply that You are, or that Your use of the Licensed Material is, connected with, or sponsored, endorsed, or granted official status by, the Licensor or others designated to receive attribution as provided in Section 3(a)(1)(A)(i).

b. Other rights.

1. Moral rights, such as the right of integrity, are not licensed under this Public License, nor are publicity, privacy, and/or other similar personality rights; however, to the extent possible, the Licensor waives and/or agrees not to assert any such rights held by the Licensor to the limited extent necessary to allow You to exercise the Licensed Rights, but not otherwise.

2. Patent and trademark rights are not licensed under this Public License.

3. To the extent possible, the Licensor waives any right to collect royalties from You for the exercise of the Licensed Rights, whether directly or through a collecting society under any voluntary or waivable statutory or compulsory licensing scheme. In all other cases the Licensor expressly reserves any right to collect such royalties.

Section 3 – License Conditions.

Your exercise of the Licensed Rights is expressly made subject to the following conditions.

a. Attribution.

1. If You Share the Licensed Material (including in modified form), You must:

A. retain the following if it is supplied by the Licensor with the Licensed Material:

i. identification of the creator(s) of the Licensed Material and any others designated to receive attribution, in any reasonable manner requested by the Licensor (including by pseudonym if designated);

ii. a copyright notice;

iii. a notice that refers to this Public License;

iv. a notice that refers to the disclaimer of warranties;

v. a URI or hyperlink to the Licensed Material to the extent reasonably practicable;

B. indicate if You modified the Licensed Material and retain an indication of any previous modifications; and

C. indicate the Licensed Material is licensed under this Public License, and include the text of, or the URI or hyperlink to, this Public License.

2. You may satisfy the conditions in Section 3(a)(1) in any reasonable manner based on the medium, means, and context in which You Share the Licensed Material. For example, it may be reasonable to satisfy the conditions by providing a URI or hyperlink to a resource that includes the required information.

3. If requested by the Licensor, You must remove any of the information required by Section 3(a)(1)(A) to the extent reasonably practicable.

4. If You Share Adapted Material You produce, the Adapter's License You apply must not prevent recipients of the Adapted Material from complying with this Public License.

Section 4 – Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed Material:

a. for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, and Share all or a substantial portion of the contents of the database;

b. if You include all or a substantial portion of the database contents in a database in which You have Sui Generis Database Rights, then the database in which You have Sui Generis Database Rights (but not its individual contents) is Adapted Material; and

c. You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of the contents of the database.

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under this Public License where the Licensed Rights include other Copyright and Similar Rights.

Section 5 – Disclaimer of Warranties and Limitation of Liability.

a. Unless otherwise separately undertaken by the Licensor, to the extent possible, the Licensor offers the Licensed Material as-is and as-available, and makes no representations or warranties of any kind concerning the Licensed Material, whether express, implied, statutory, or other. This includes, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of latent or other defects, accuracy, or the presence or absence of errors,

whether or not known or discoverable. Where disclaimers of warranties are not allowed in full or in part, this disclaimer may not apply to You.

b. To the extent possible, in no event will the Licensor be liable to You on any legal theory (including, without limitation, negligence) or otherwise for any direct, special, indirect, incidental, consequential, punitive, exemplary, or other losses, costs, expenses, or damages arising out of this Public License or use of the Licensed Material, even if the Licensor has been advised of the possibility of such losses, costs, expenses, or damages. Where a limitation of liability is not allowed in full or in part, this limitation may not apply to You.

c. The disclaimer of warranties and limitation of liability provided above shall be interpreted in a manner that, to the extent possible, most closely approximates an absolute disclaimer and waiver of all liability.

Section 6 – Term and Termination.

a. This Public License applies for the term of the Copyright and Similar Rights licensed here. However, if You fail to comply with this Public License, then Your rights under this Public License terminate automatically.

b. Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:

1. automatically as of the date the violation is cured, provided it is cured within 30 days of Your discovery of the violation; or

2. upon express reinstatement by the Licensor.

For the avoidance of doubt, this Section 6(b) does not affect any right the Licensor may have to seek remedies for Your violations of this Public License.

c. For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate terms or conditions or stop distributing the Licensed Material at any time; however, doing so will not terminate this Public License.

d. Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

Section 7 – Other Terms and Conditions.

a. The Licensor shall not be bound by any additional or different terms or conditions communicated by You unless expressly agreed.

b. Any arrangements, understandings, or agreements regarding the Licensed Material not stated herein are separate from and independent of the terms and conditions of this Public License.

Section 8 – Interpretation.

a. For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduce, limit, restrict, or impose conditions on any use of the Licensed Material that could lawfully

be made without permission under this Public License.

b. To the extent possible, if any provision of this Public License is deemed unenforceable, it shall be automatically reformed to the minimum extent necessary to make it enforceable. If the provision cannot be reformed, it shall be severed from this Public License without affecting the enforceability of the remaining terms and conditions.

c. No term or condition of this Public License will be waived and no failure to comply consented to unless expressly agreed to by the Licensor.

d. Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver of, any privileges and immunities that apply to the Licensor or You, including from the legal processes of any jurisdiction or authority.

Creative Commons is not a party to its public licenses. Notwithstanding, Creative Commons may elect to apply one of its public licenses to material it publishes and in those instances will be considered the "Licensor." Except for the limited purpose of indicating that material is shared under a Creative Commons public license or as otherwise permitted by the Creative Commons policies published at `creativecommons.org/policies`, Creative Commons does not authorize the use of the trademark "Creative Commons" or any other trademark or logo of Creative Commons without its prior written consent including, without limitation, in connection with any unauthorized modifications to any of its public licenses or any other arrangements, understandings, or agreements concerning use of licensed material. For the avoidance of doubt, this paragraph does not form part of the public licenses.

Creative Commons may be contacted at `creativecommons.org`.

Appendix E

References

Banerjee, Dean, "PLL Performance, Simulation, and Design", 5th edition, Texas Instruments Incorporated, May 2017.

Best, Roland E., *Phase-Locked Loops – Design, Simulation, and Applications*, fourth edition, The McGraw-Hill Companies Incorporated, New York, NY, 1999.

"KSZ8051MNL/RNL 10BASE-T/100BASE-TX Automotive Physical Layer Transceiver", document DS00002310B, Microchip Technology Incorporated, 2018.

Palermo, Sam, "Lecture 4: Phase Detector Circuits", ECEN620 Network Theory Broadband Circuit Design lecture notes, Texas A&M University, Analog & Mixed-Signal Center, Fall 2023.

Morgan, David K., "CD4046B Phase-Locked Loop: A Versatile Building Block for Micropower Digital and Analog Applications", Application Report SCHA002A, Texas Instruments Incorporated, February 2003.

Appendix F

Version history

This is a list showing all significant additions, corrections, and other edits made to this learning module. Each entry is referenced by calendar date in reverse chronological order (newest version first), which appears on the front cover of every learning module for easy reference. Any contributors to this open-source document are listed here as well.

20 November 2025 – moved a paragraph in the "Example: PLL filter cutoff experiment" Case Tutorial section.

23 April 2025 – replaced "phase comparator" with "phase detector" in most places within this document to avoid confusion. This includes image_5923, image_5924, and image_5926. Also changed a section title from "Phase detectors" to "Phase detector types".

22 April 2025 – added a comment to the "Flip-flop phase detectors" subsection of the Tutorial chapter mentioning how this class of phase detector is more susceptible to noise than the XOR-based type. Without this caveat it may have been misunderstood that flip-flop phase detectors were superior in all regards to XOR phase detectors, when in fact they have weaknesses just like any other technology.

5 April 2025 – modified the title of the Case Tutorial section "Example: CD4046 experiment" to be "Example: PLL phase detector type experiment"

21 November 2024 – divided the Introduction chapter into sections, one with recommendations for students, one with a listing of challenging concepts, and one with recommendations for instructors. Also edited some of the instructor notes.

21-24 April 2024 – added content to the Introduction and Tutorial chapters, and corrected a typographical error.

23 December 2023 – minor wording edit to the Tutorial regarding the synthesis of harmonic frequencies using a phase-locked loop.

6 February 2022 – added a Conceptual Reasoning question asking how to build a crude electronic

music synthesizer using the VCO of a model CD4046B phase-locked loop IC.

31 January 2022 – added a Conceptual Reasoning question referencing a Texas Instruments document on the CD4046B phase-locked loop.

30 January 2022 – document first created.

Index

B, back-fed signal, 17
H, feedback function, 17
K, compensator function, 17
R, reference signal, 17
Y, output signal, 17

Adding quantities to a qualitative problem, 48
Annotating diagrams, 47

Block diagram, 17

Capture range, 21
Cardinal rule of negative feedback, 17
Checking for exceptions, 48
Checking your work, 48
Code, computer, 57
Cutoff, 20
Cycle slip, 23, 27

Degenerative feedback, 17
Dimensional analysis, 47

Edwards, Tim, 58

Feedback, 17
Feedback, negative, 23
Feedback, positive, 23
Filter cutoff, 20
FPGA, 29
Frequency, 16
Frequency synthesizer, 29

Gain, 17, 29
Graph values to solve a problem, 48
Greenleaf, Cynthia, 31

Harmonic frequency, 29
Hertz, 16
How to teach with these modules, 55
Hwang, Andrew D., 59

Identify given data, 47
Identify relevant principles, 47
Intermediate results, 47
Inverted instruction, 55

Knuth, Donald, 58

Lamport, Leslie, 58
Limiting cases, 48
Lock range, 21

Metacognition, 36
Microprocessor, 29
Mixer circuit, 27
Moolenaar, Bram, 57
Murphy, Lynn, 31

Negative feedback, 17, 23

Open-source, 57

Positive feedback, 17, 23
Problem-solving: annotate diagrams, 47
Problem-solving: check for exceptions, 48
Problem-solving: checking work, 48
Problem-solving: dimensional analysis, 47
Problem-solving: graph values, 48
Problem-solving: identify given data, 47
Problem-solving: identify relevant principles, 47
Problem-solving: interpret intermediate results, 47
Problem-solving: limiting cases, 48
Problem-solving: qualitative to quantitative, 48
Problem-solving: quantitative to qualitative, 48
Problem-solving: reductio ad absurdum, 48

Problem-solving: simplify the system, 47
Problem-solving: thought experiment, 47
Problem-solving: track units of measurement, 47
Problem-solving: visually represent the system, 47
Problem-solving: work in reverse, 48

Qualitatively approaching a quantitative problem, 48

Range, capture, 21
Range, lock, 21
Reading Apprenticeship, 31
Reductio ad absurdum, 48, 54, 55
Regenerative feedback, 17
RF mixer, 27

Schoenbach, Ruth, 31
Scientific method, 36
Simplifying a system, 47
Socrates, 54
Socratic dialogue, 55
SPICE, 31
Stallman, Richard, 57
Synthesizer, harmonic frequency, 29

Thought experiment, 47
Torvalds, Linus, 57

Units of measurement, 47

Visualizing a system, 47

Work in reverse to solve a problem, 48
WYSIWYG, 57, 58