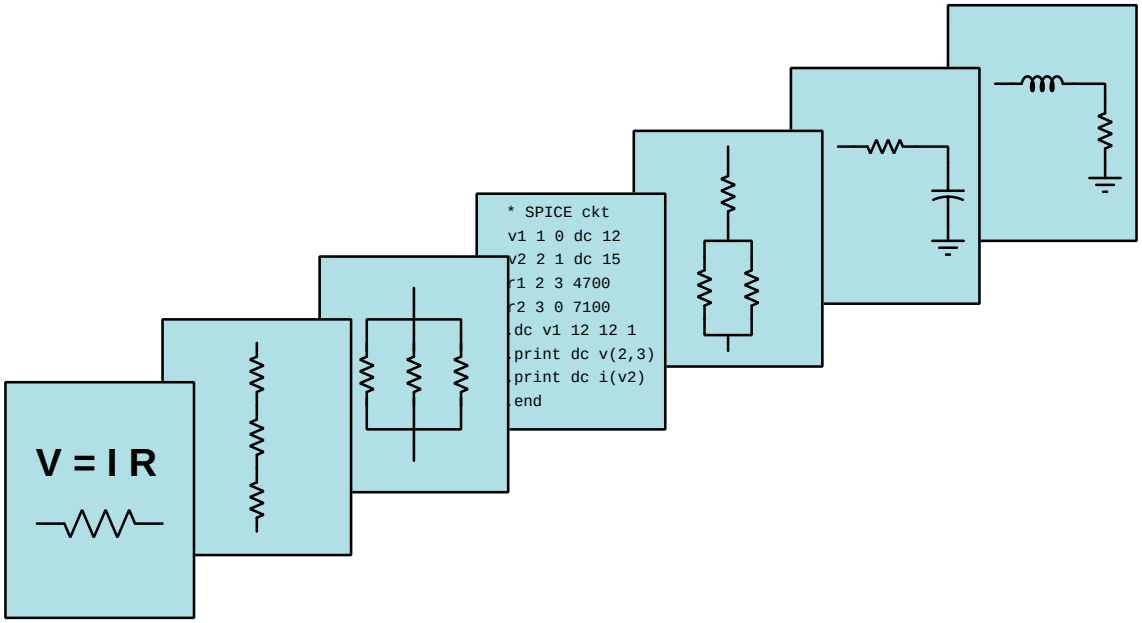
Modular Electronics Learning (ModEL) project



Pulse Width Modulation

Last update = 31 October 2024

Contents

Chapter 1

Introduction

1.1 Recommendations for students

Pulse width modulation (*PWM*) is a technique by which a load's delivered power is controlled by rapidly switching it between the states of fully-on and fully-off. Contrasted against analog power control where the flow of electrical energy is smoothly adjusted by some device such as a rheostat or transistor, pulse width modulation is far more efficient. That is to say, PWM "wastes" very little energy in the form of heat at the controlling device. This makes PWM a very attractive method for controlling load power. The ratio of "on" time to the total period of the pulse waveform is called *duty cycle*, and this simple ratio is the proportion of average power received by any load driven by a PWM source. At 100% duty cycle (i.e. full-on) the load receives full power, and at any duty cycle less than 100% the load's average power will be that proportion of its full-power rating.

PWM is used in a wide variety of applications, from light dimming to motor speed control to temperature modulation to fuel flow in modern internal-combustion engines. Its utility is limited primarily by the time span over which each on/off cycle must occur to avoid excessive "cycling" of the controlled variable and by the switching speed of the controlling element.

Important concepts related to pulse-width modulation include **Joule's Law**, **Ohm's Law**, **Conservation of Energy**, **efficiency**, **inertia**, **transistor**, **duty cycle**, **comparator**, **pulse waveforms**, **modulation** and **demodulation**, **integrator** networks, and **filter** networks.

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

- How might an experiment be designed and conducted to measure the duty cycle of a pulse waveform? What hypothesis (i.e. prediction) might you pose for that experiment, and what result(s) would either support or disprove that hypothesis?
- Why is rheostat-based control of load power inefficient?
- How does power relate to energy?
- How rapidly must we pulse power to a load in order to effectively modulate its power dissipation using PWM?

- How may we calculate duty cycle from an oscillograph display of a PWM signal?
- Where may we find PWM being used in practical applications?
- How may a comparator be used to generate a PWM signal?
- How may PWM be used to convey information (rather than modulate load power)?
- Why might we choose to use PWM to convey information?
- Is it possible to have a PWM signal with a duty cycle greater than 100% or less than 0%?
- How may we construct a *subtraction* function for PWM signals using logic gates?
- How may we construct a *low-select* function for PWM signals using logic gates?
- How may we construct a *high-select* function for PWM signals using logic gates?
- How may we construct a *multiplication* function for PWM signals using logic gates?

1.2 Challenging concepts related to pulse-width modulation

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

- **Average power transfer** – pulse-width modulation is often used as a power control technique, pulsing energy to a load and varying the average rate of power by the duty cycle of those pulses. So long as the PWM frequency is high enough that the load's own inertia does not allow for substantial variation of the final effect (e.g. an electric motor's rotor intertia maintaining an average speed, or an electric oven's thermal inertia maintaining an average temperature), on/off pulsing works quite well to modulate power to a load.

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 show how quantitative results were obtained by the author in the Tutorial chapter's examples.

- **Outcome** – Predict comparator output states given input conditions

  Assessment – Determine comparator output pulse waveform given input signals; e.g. pose problems in the form of the "Comparator output waveform" Conceptual Reasoning question.

  Assessment – Determine comparator output duty cycle given input signals; e.g. pose problems in the form of the "Duty cycle based on triangle waveform" and "Duty cycle based on sine waveform" Quantitative Reasoning questions.

- **Outcome** – Compute average load power given PWM signal characteristics

  Assessment – Calculate average load power given oscillograph images and load resistance; e.g. pose problems in the form of the "Duty cycle and power calculations" Quantitatived Reasoning question.

- **Outcome** – Independent research

  Assessment – Read and summarize in your own words reliable source documents on the subject of internal-combustion engine fuel injector control, which utilizes PWM to open and close the injector ports.

Chapter 2

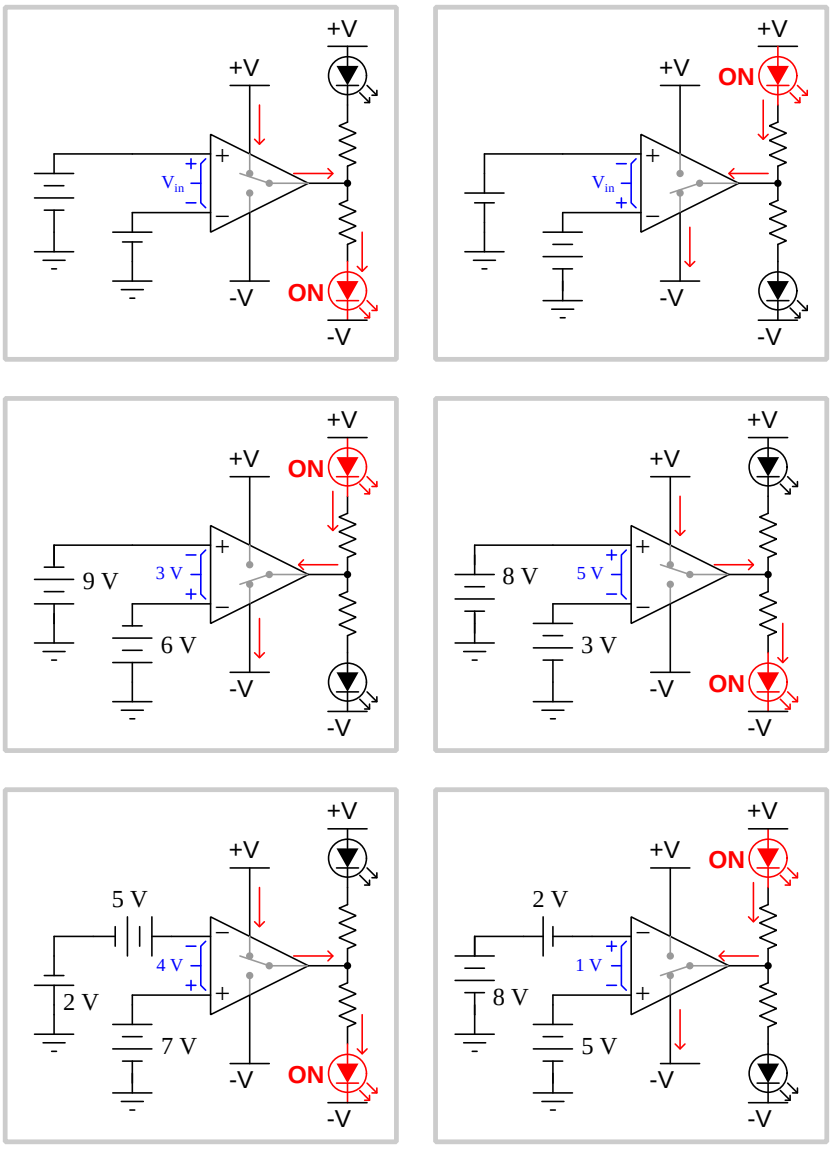
Case Tutorial

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?

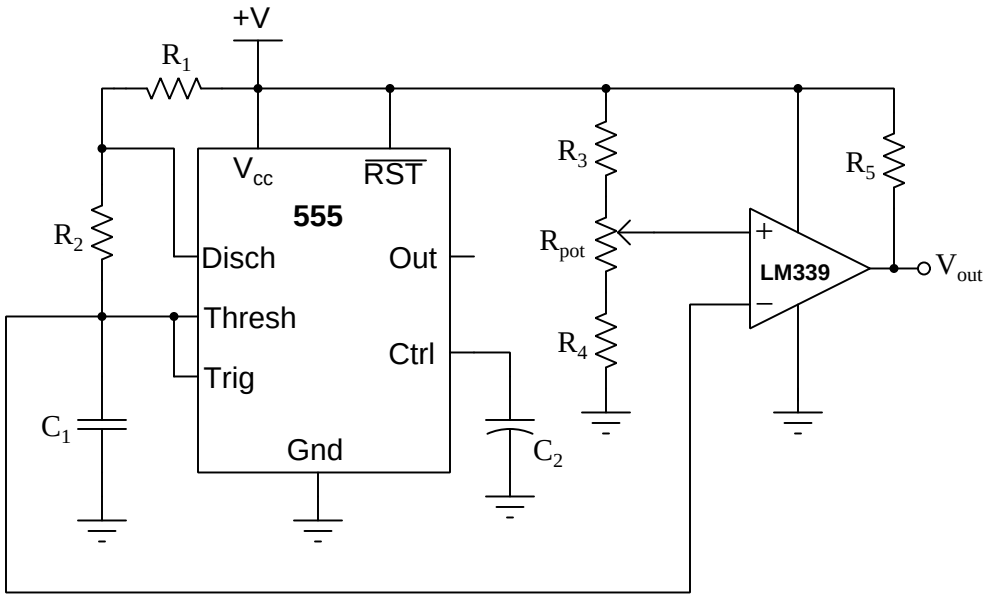
2.1 Example: comparator response to various inputs

In these illustrations, I have likened the comparator's action to that of a single-pole, double-throw switch, showing the "connection" made between power supply terminals and the output terminal:



2.2 Example: 555-based PWM signal generator

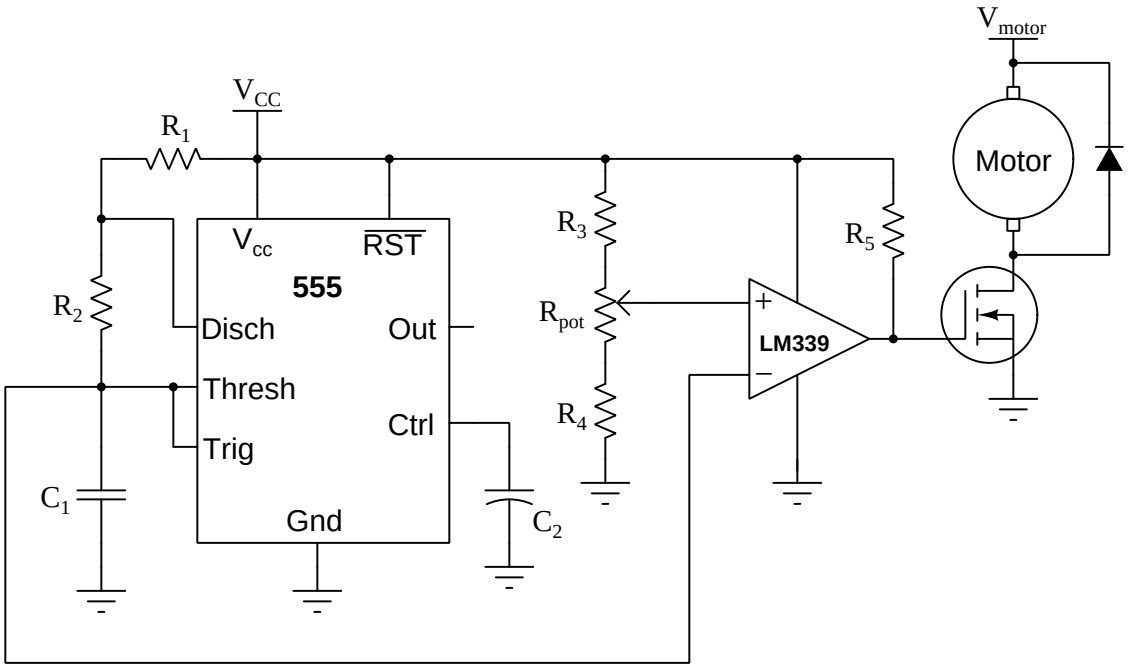
The following circuit uses a 555 timer IC as an astable multivibrator (i.e. oscillator) producing a "sawtooth" voltage signal waveform across its timing capacitor. A comparator compares this oscillating signal against a DC reference voltage set by the position of a potentiometer, to generate a PWM signal at the V_{out} terminal:



Since the 555's threshold and trigger levels are internally set by a voltage divider network to be $\frac{2}{3}$ and $\frac{1}{3}$ of the power supply voltage, respectively, these will be the upper and lower limits of the sawtooth waveform voltage. Our duty cycle adjustment potentiometer (R_{pot}) has fixed resistors of equal value above and below it to similarly limit its adjustable range to between $\frac{2}{3}$ and $\frac{1}{3}$ of the power supply voltage. Resistor R_5 is a pullup resistor, necessary in this design because we are using one of the comparators within a model LM339 quad comparator IC, each of which can only sink and cannot source output current.

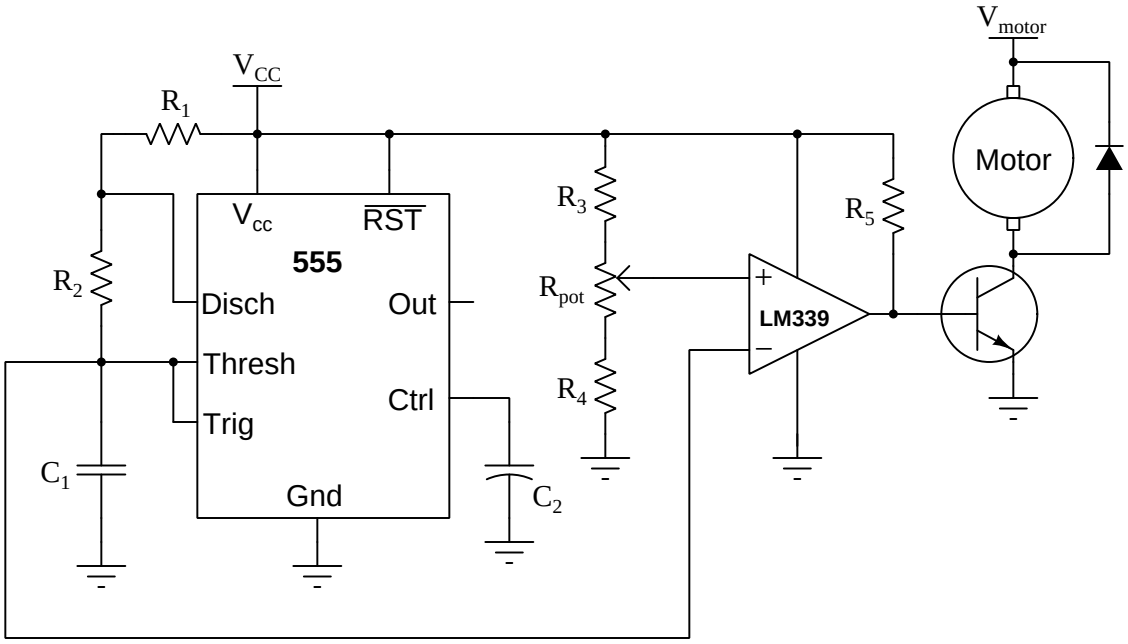
Moving the potentiometer wiper in the upwards direction will increase the PWM signal's duty cycle (i.e. "high" for more time and "low" for less time).

Next we see how a MOSFET could be connected to the output of this PWM signal generator to control power to a DC electric motor:



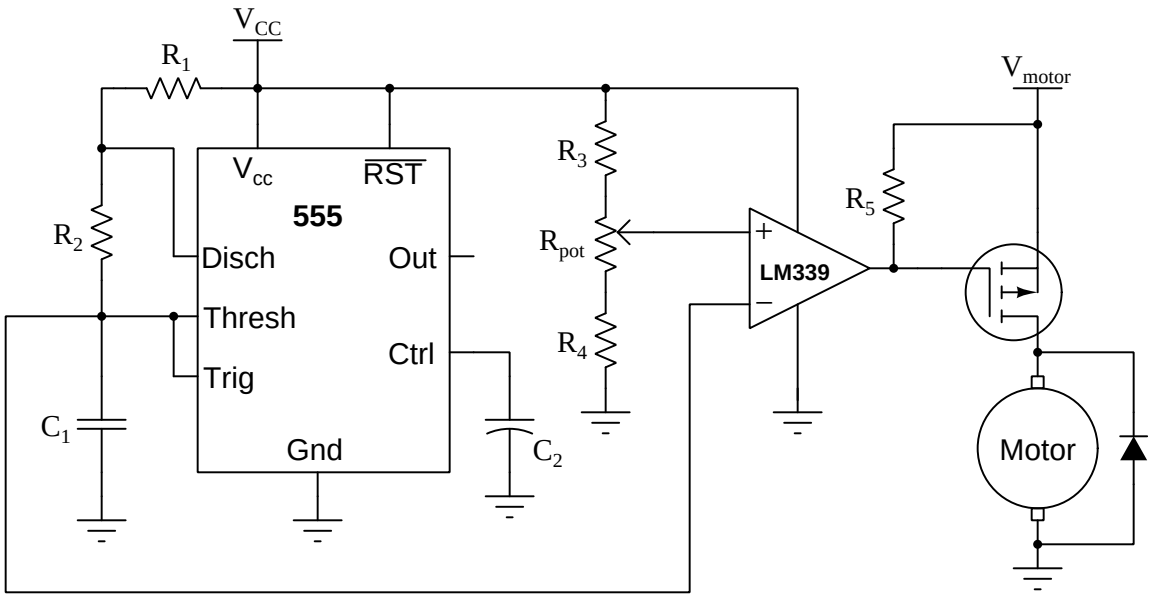
Here we use two separate power sources, one for the PWM signal generator and another for the motor. Doing so isolates much of the motor's electrical "noise" from interfering with the 555 timer. If we were to use the same DC power source for both the motor and the signal generator, the noise voltage output by the motor could very well cause unpredictable behavior from the 555 circuit, especially the LM339 comparator whose reference voltage would be affected by this noise. It is important that both DC power sources (V_{DD} and V_{motor}) share the same ground so that the PWM signal generator's output will be impressed between the MOSFET's gate and source terminals. A commutating diode is also recommended because this will spare the MOSFET from experiencing voltage transients when it turns off current to an inductive load.

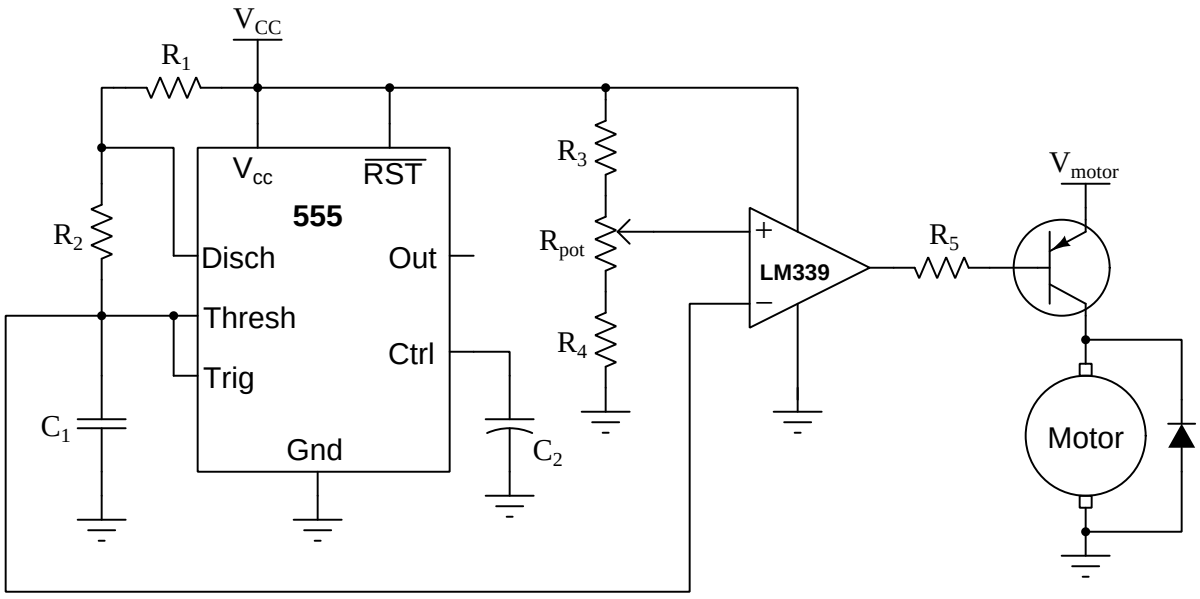
Next we a BJT version of the same motor-control circuit:



Here it is imperative to properly size resistor R_5 since it must supply adequate current to the transistor's base to saturate it in the "on" state. Too large of a value for R_5 will cause the transistor to not turn on fully, dropping excessive V_{CE} and dissipating excessive power in the form of heat; too small of a value for R_5 will force the LM339 comparator to have to sink too much current, possibly overstressing it.

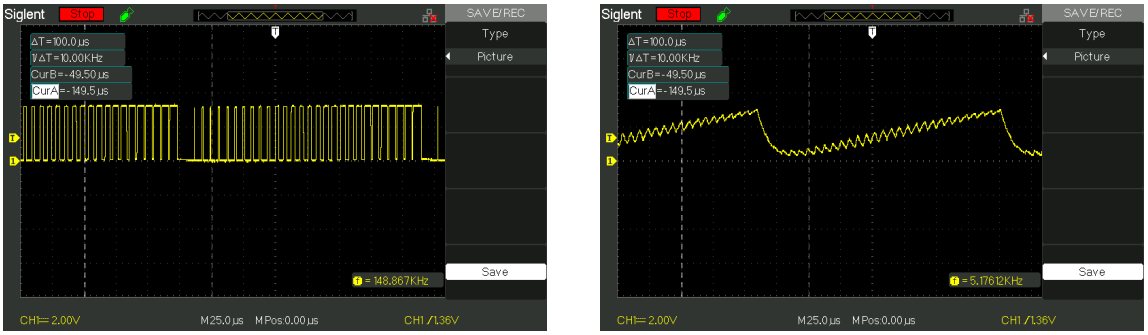
And of course high-side switching circuits are also possible, but this means the PWM signal generator's duty cycle will have a complementary effect on load (motor) power – i.e. when the duty cycle is low (output low most of the time), motor speed will be high:

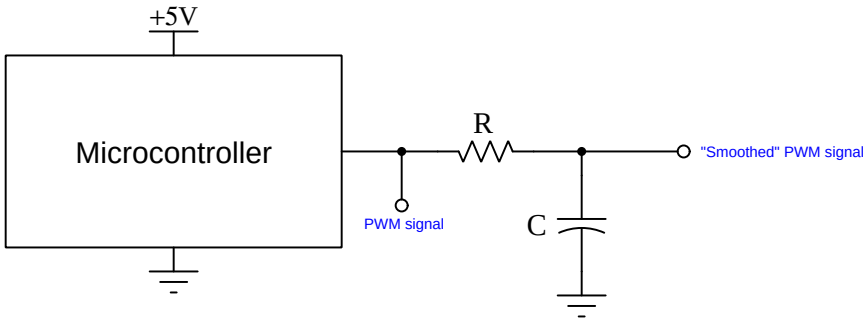




2.3 Example: PWM waveform synthesis

The following oscilloscope screenshots show a PWM signal generated by a microcontroller programmed to increment duty cycle in linear fashion and then reset that duty cycle to zero (left), and that same PWM signal "smoothed" using a low-pass RC filter network (right):

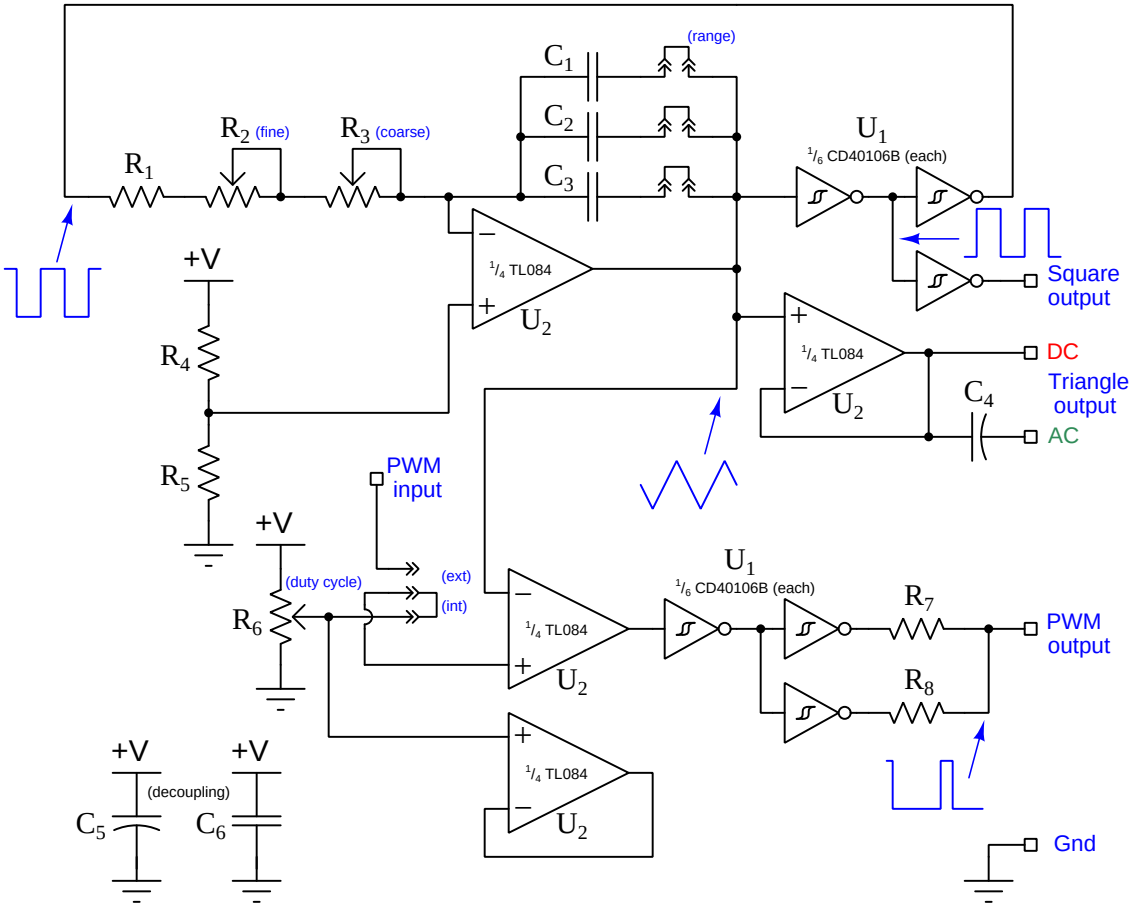




Many microcontrollers and other programmable devices have only digital (discrete high/low) output signal capability, and so if an analog output signal is desired the "smoothed" PWM technique may be a valid solution. As you can see, the filtered signal is not a perfect sawtooth waveform, but it may be close enough depending on the application.

2.4 Example: simple triangle/square/PWM oscillator

This simple oscillator circuit provides triangle wave output (both DC- and AC-coupled) as well as square wave output and pulse-width-modulated ("PWM") square wave output with an adjustable duty cycle. Jumpers select capacitor values for frequency range, while two potentiometers provide coarse and fine frequency adjustment:



DC power source voltage may range widely, given the advertised operating voltage ranges for the CD40106B and TL084 integrated circuits (3 to 18 Volts for the hex inverter and 10 to 30 Volts for the opamp). I have successfully tested this circuit at a supply voltage as low as 5 Volts and it seems to work well.

It is worth noting that the fairly close similarity between triangle and sine wave-shapes means students may use the triangle-wave output of this oscillator circuit as an AC voltage source to energize simple RLC networks rather than using a true sine-wave signal generator signal. The error between measured and calculated values of voltage and current are generally within the $\pm$ 5%

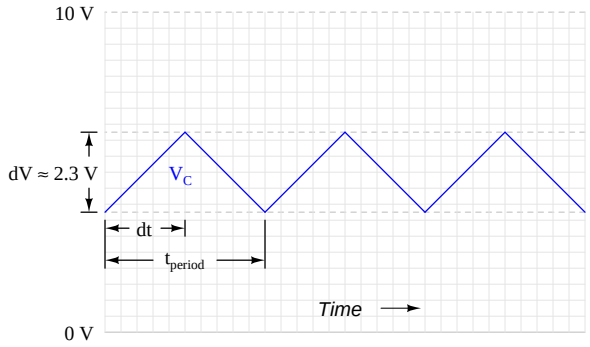
tolerance of common resistors, capacitors, and inductors when using a true-RMS multimeter to take the measurements.

Parts list:

- U_1 = CD40106B CMOS hex inverter with Schmitt trigger inputs
- U_2 = TL084 quad operational amplifier
- R_1 = 10 kΩ – *this sets the maximum frequency when both pots are at minimum resistance*
- R_2 = 10 kΩ potentiometer – *this is the "fine" frequency adjustment*
- R_3 = 100 kΩ potentiometer – *this is the "coarse" frequency adjustment*
- R_4 = 10 kΩ – *the values of R_4 and R_5 are not critical, so long as they are equal*
- R_5 = 10 kΩ
- R_6 = 10 kΩ potentiometer – *the values of R_6 is not critical*
- R_7 = 100 Ω – *R_7 and R_8 equalize load-sharing between inverters, the values being non-critical*
- R_8 = 100 Ω
- C_1 = 0.0047 μF ceramic (for oscillator timing)
- C_2 = 0.047 μF ceramic (for oscillator timing)
- C_3 = 0.47 μF ceramic (for oscillator timing)
- C_4 = 4.7 μF electrolytic (for output coupling – value not critical)
- C_5 = 47 μF electrolytic (for power supply decoupling – value not critical)
- C_6 = 1 μF ceramic (for power supply decoupling – value not critical)

The triangle wave signal begins to exhibit "ringing" at the positive and negative peaks if the total oscillator network resistance (i.e. $R_1 + R_2 + R_3$) becomes too low (approximately 20 kΩ or less). This is the major function of the fixed resistor R_1, to make sure this resistance never becomes unreasonably low.

The upper-most opamp functions as a current source for capacitors C_1 through C_3, driving a constant current through the capacitor(s) in order to produce a ramping voltage across them. The amount of current driven through the capacitor(s) is equal to the difference between the hex inverter's output voltage (square wave) and one-half supply voltage (set by divider R_4/R_5) divided by the resistance of $R_1 + R_2 + R_3$, which should be approximately $\pm\frac{V_{supply}}{2(R_1+R_2+R_3)}$. The capacitor voltage will ramp at a rate $\frac{dV}{dt}$ equal to that current divided by capacitance C. The CD40106B Schmitt-trigger inverter has a typical hysteresis value of approximately[1] 23% (e.g. 2.3 Volts for a 10-Volt V_{supply}), so the time duration of each half-cycle of the waveform is the amount of time it takes the capacitor voltage to ramp that 23% of supply voltage as opamp U_2 forces a constant current:



Here is a mathematical derivation for output frequency based on these conditions, where C represents the capacitance offered by C_1, C_2, and/or C_3 (depending on which jumpers are connected) and R represents the total series resistance of R_1, R_2, and R_3:

$$I_C = C\frac{dV}{dt} \text{ ("Ohm's Law" for any capacitor)} \qquad I_C = \frac{V_{supply}}{2R} \text{ (Current forced by opamp)}$$

$$C\frac{dV}{dt} = \frac{V_{supply}}{2R}$$

$$\frac{dV}{dt} = \frac{V_{supply}}{2RC}$$

$$dV = \frac{V_{supply}}{2RC}dt$$

$$0.23V_{supply} \approx \frac{V_{supply}}{2RC}dt$$

$$(0.23)(2RC) \approx dt$$

[1] This figure of 23% is merely a rough estimate, and should not be relied upon for accuracy. Schmitt-trigger logic gates are designed for noise tolerance, not to function as precision comparators, and so it is unwise to assume this value will be stable from IC to IC or even between different operating frequencies in this circuit!

$$t_{period} \approx (2)(0.23)(2RC)$$

$$f \approx \frac{1}{(2)(0.23)(2RC)} \approx \frac{1.09}{RC}$$

Assuming the component values shown in the parts list, with both potentiometers set to mid-position and only the middle capacitor (C_2) selected by jumper, the frequency should be approximately:

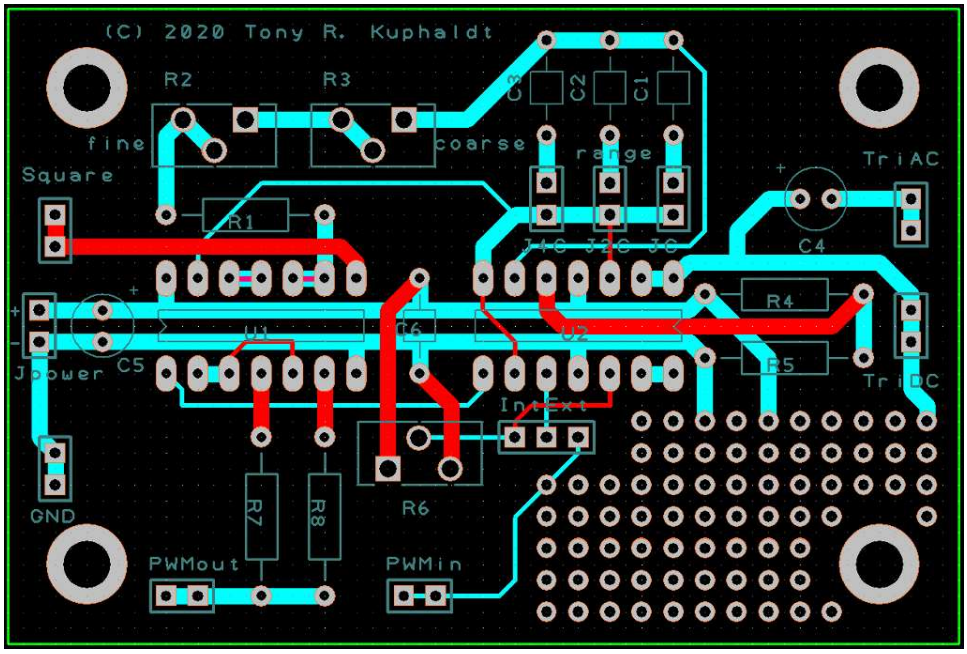
$$f \approx \frac{1.09}{(65 \times 10^3)(0.047 \times 10^{-6})} = 356 \text{ Hz}$$

I recommend choosing capacitor values of octave multiples (i.e. C, $2C$, $4C$) or decade multiples (i.e. C, $10C$, $100C$). With a large "coarse" potentiometer value (R_3), decade multiples make the most sense. Lower-valued frequency-adjust potentiometers would better suit octave-multiples of capacitor value, but then the total adjustment range of the oscillator will be more limited.

The int/ext jumper sets the signal source for the PWM signal's adjustable duty cycle. When set for "internal" the duty cycle is adjusted by potentiometer R_6; when set for "external" a signal voltage input at the "PWM input" terminal controls the duty cycle. This feature makes the circuit particularly useful for experiments requiring a custom PWM signal source: if all you need is a manually-adjustable PWM duty cycle then you can use the "internal" setting and adjust that duty cycle using the potentiometer; however, if you need the duty cycle to be controlled by another signal, use the "external" setting and then input that signal into the "PWM input" terminal to force the oscillator circuit to create a duty cycle proportional to that controlling signal.

When using the "external" PWM input signal option, the usable range of this controlling DC voltage signal falls within the peak-to-peak range of the triangle waveform. As shown previously, using a 10 Volt DC power supply we can expect the triangle wave to have a peak-to-peak amplitude of approximately 2.3 Volts, centered around 5 Volts (mid-way between the +10 Volts and Ground). Therefore, under these conditions we could expect the PWM controlling signal to have a usable range from 3.85 Volts to 6.15 Volts.

One possible PCB layout for this circuit is shown below, the dimensions of the board being 3 inches by 2 inches. Red traces are on the top copper layer of the board, while blue traces are on the bottom. DC supply power comes into the board at the left-hand edge, at the `Jpower` terminals:



Note how two pads are provided for each of the I/O terminals. This was done to facilitate a small loop of bare wire soldered between the two pads, which makes an easy connection point for "grabber" hooks or "alligator" clips. The four large pads at the board's corners are intended for nylon stand-offs, for mounting this PCB to a larger panel. These mounting pads are spaced 2.5 inches horizontally and 1.5 inches vertically from each other.

An important detail to note on this layout is the orientation of the two integrated circuits. U_1 (the CMOS hex inverter) has its pin 1 located on the left side, while U_2 (the quad opamp) is "upside-down" with its pin 1 located on the right. This is due to the locations of the DC power pins on each of these integrated circuits. In order to avoid crossing the +V and Ground power traces on the board, these two chips had to be positioned opposite-facing.

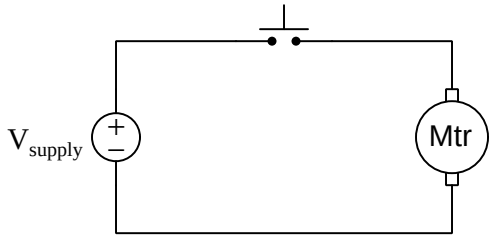
A "prototyping" area with an array of unallocated pads exists at the lower-right corner of the PCB, provided so that you may add other components to the board (e.g. power amplifier, wave-shaping networks) to enhance its capabilities.

Chapter 3

Tutorial

3.1 On-off power control

Consider the following circuit, consisting only of a DC voltage source, a pushbutton switch, and a DC electric motor:



The motor will run any time the switch is pressed, because pressing the pushbutton causes its contacts to close and complete the circuit. Releasing the switch causes its contacts to return to their "normal" (resting) state and breaks the circuit. What could be simpler than this?

When the motor runs, its power dissipation may be calculated quite simply by multiplying voltage (V) and current (I), in accordance with Joule's Law:

$$P = IV$$

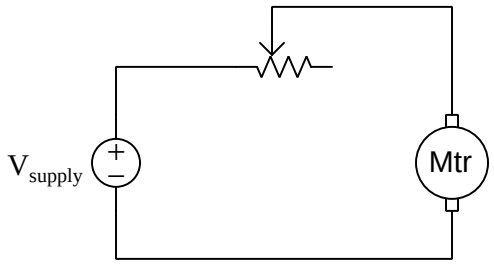
Electric motors are very efficient machines, with most[1] of this power converted into mechanical work while the remainder converts into heat and other non-useful forms.

Many applications, though, require *variable* motor power. It is simply not enough to be able to turn an electric motor on and off for applications such as electric vehicle drive motors and servo motors for robotics applications. A pushbutton switch with its discrete (on/off) nature simply does not allow continuous throttling of energy.

[1] Motors specifically designed for high efficiency often achieve over 90% conversion of electrical energy into mechanical energy.

3.2 Resistive power control

A simple means of achieving variable motor power control is to replace the pushbutton switch with a *rheostat* (variable resistor) connected in series:



The amount of electrical resistance for any series network is the total of all individual resistances. By inserting a variable resistance in series with the motor, the total amount of resistance presented to the DC source will be the motor's effective resistance plus the resistance of the rheostat. Ohm's Law predicts total current to be equal to source voltage divided by total resistance, and so by varying the rheostat we also vary total resistance, and with that we have control over total current. This, in turn controls motor power.

However, as we control motor power in this manner we will see that a much lower percentage of the electrical energy gets converted into useful mechanical work, and more of it gets converted into heat. The reason for this additional "waste" is power dissipated by the rheostat: being an electrical resistance, any current passing through the rheostat must result in energy dissipation ($P = I^2R$). Total energy put into a system must equal total energy exiting for any system that does not store energy internally, and so for any steady-state condition we can say that the total source power must equal rheostat power dissipation plus motor power (useful work plus heat dissipation):

$$P_{total} = IV_{source} = I^2R_{rheostat} + P_{mech} + I^2R_{motor}$$

Where,

P_{total} = Total power provided by the source
I = Circuit current (same for all components)
V_{source} = Voltage across the source terminals
$R_{rheostat}$ = Resistance of the rheostat
P_{mech} = Mechanical power output by the motor
R_{motor} = Wire resistance inside the motor

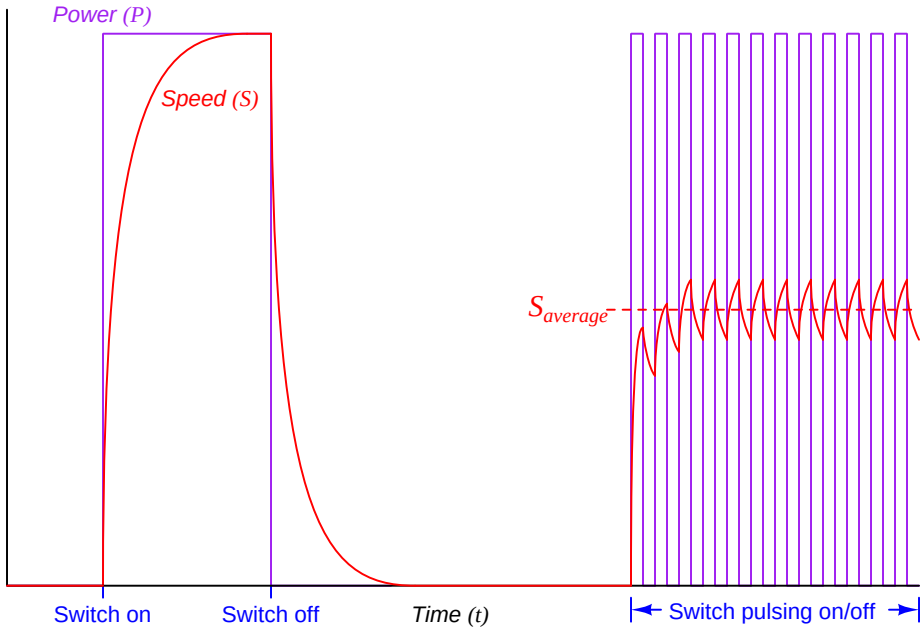
In the interest of prudent energy and resource usage, the ideal scenario is one where P_{mech} is as close to being equal with P_{total} as possible. There is little we can do to reduce I^2R_{motor} loss apart from using a better-quality motor, but the only justification for $I^2R_{rheostat}$ loss is the ability to adjust motor power.

It is worth reflecting at this point that our simple switch-based motor control circuit did not suffer this extra "loss" of energy. When the switch was open it dropped full source voltage but passed no current, so the switch dissipated no power ($P = IV = 0V = 0$). When the switch was closed it dropped no voltage while passing current, so again the switch dissipated no power

($P = IV = I0 = 0$). Our circuit was more energy-efficient, but we had no fine control over motor power.

3.3 Pulsed power control

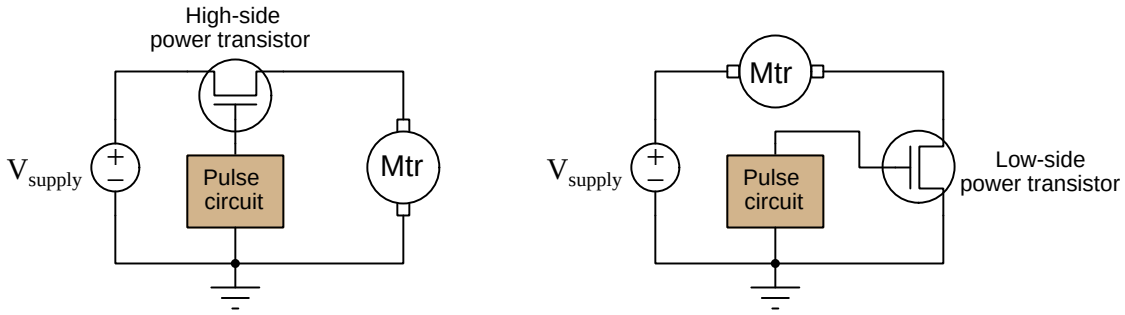
What would happen if we turned the switch on and off so fast that the motor did not have enough time to speed up or slow down very much within each phase of the switch's cycle? If we graph power (P) and motor speed (S) over time (t), we see how this might work:



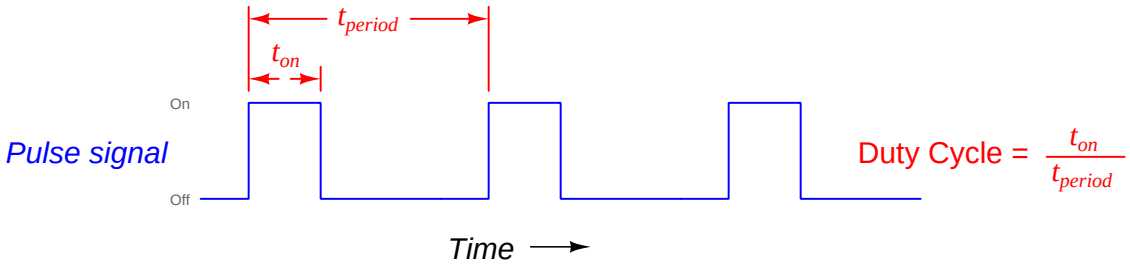
The reason the motor settles around an average speed value while the switch pulses on and off is the exact same reason the motor does not instantaneously achieve full speed when the switch turns on and the same reason it does not instantaneously stop when the switch turns off: *the motor's mechanical inertia stores kinetic energy.* Power is the rate of energy transfer, and so for any given amount of applied electric power there will be a limit to how rapidly the motor's rotor may accumulate kinetic energy. The same holds true when the switch opens: whatever is mechanically loading the motor demands power at a finite rate, which means there will be a limit to how rapidly the rotor slows down as well. Therefore, the motor's rotor speed follows a curved trajectory upward when the switch applies electrical power and follows another curved trajectory downward when the switch interrupts power. If we pulse power on and off at a rapid enough rate, the motor never has enough time to either achieve full speed or to stop, and so its speed hovers around some average value.

Pressing and releasing a pushbutton switch rapidly enough to achieve relatively smooth control of motor power may not be practical, but if we substitute a *transistor* for the pushbutton and trigger that transistor fully on and fully off with an electronic oscillator circuit pulsing at a frequency of thousands of times per second, we may achieve very smooth motor power control. Most importantly, this method of power control enjoys high energy efficiency for the same reasons as the simple pushbutton switch circuit: when the transistor is fully off it conducts zero current and therefore dissipates no heat; when fully on it drops very little voltage and therefore dissipates very little heat.

All we need to make this scheme practical is a pulse-generating circuit such as a square-wave oscillator to alternately drive the transistor fully on and fully off. Both high-side and low-side transistor switching configurations are possible:

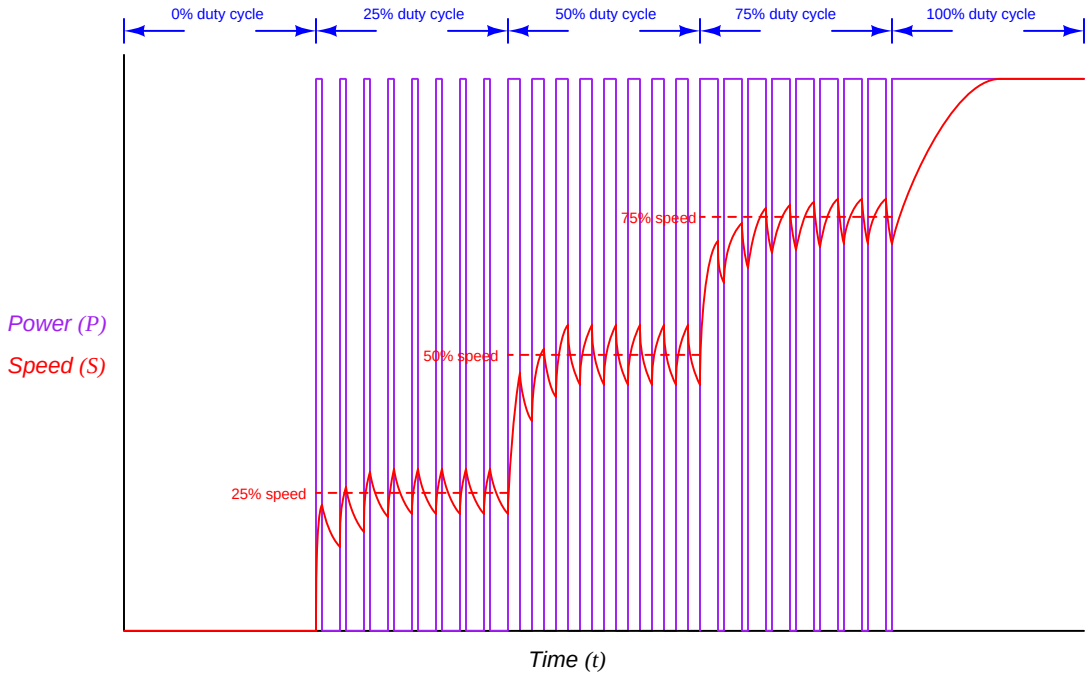


Altering the frequency of the on/off pulsing does not affect the average load power, but merely affects how much the instantaneous load power fluctuates. In other words, the more rapidly we pulse power on and off, the "smoother" our average power becomes over time, and the closer it settles to the average value. For motor speed control it is typical to use pulse frequencies in the kilo-Hertz range[2]. If we wish to influence the amount of average power, we must change the ratio of "on" time to total time (pulse period). This ratio is called *duty cycle*, and it is usually expressed as a percentage:



[2]In fact, this is why electric motors driven by such power control circuitry often emit audible *tones*. The rapid pulsing of electric power to the motor at frequencies within the human audible range results in small amounts of audible noise emanating from the motor structure. The mechanism of noise generation is a phenomenon called *magnetostriction*, where ferromagnetic materials literally expand and contract on a microscopic scale as they are magnetized and demagnetized. This vibrational motion of the ferrous pole pieces in any motor receiving pulsing electrical power induces vibrations in surrounding air molecules, hence the audible tones. This is also why *transformers* energized by AC electric power tend to "hum".

A low duty cycle results in less power applied to the motor over time because a low duty-cycle pulse is on for less time than it is off. A high duty cycle results in more power applied to the motor because the pulse is on longer than it is off. We may illustrate by showing the motor's response to five different duty cycle values: 0%, 25%, 50%, 75%, and 100%:



The average power delivered to the motor is simply the peak power delivered during each "on" period multiplied by the duty cycle. For example, if during each "on" period the motor received 35 Volts at 2 Amperes, the peak power would be 70 Watts. At 25% duty cycle, the motor's average power would be 25% of 70 Watts, or 17.5 Watts. Again, if we wished to have smoother control over the motor's power we could increase the pulsing frequency, but it is still duty cycle that controls the percentage of full power that the motor receives.

This technique is commonly referred to as *pulse width modulation*, or *PWM*. The name comes from the idea that we are modulating (controlling) power to a load of some kind by varying the ratio of the pulse's width to its period (duty cycle), and it has a great many applications. So long as the application in mind has enough energy storage or other form of inertia to average out the PWM pulses, we may achieve relatively smooth control using plain on/off elements.

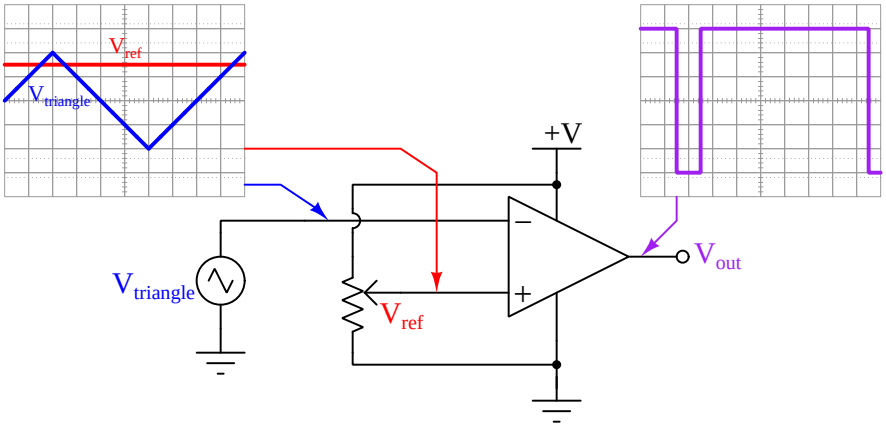
3.4 PWM applications

Just a few practical applications of PWM are listed here besides electric motor control:

- **Temperature control**: any space with a heat source may have its temperature controlled by careful PWM energization of that heat source. The pulse frequency for heating applications is often slow because temperature changes slowly with large masses. For a residential heating system the pulse period is in the order of many minutes. For a microwave oven it is in the order of seconds.
- **Fuel injector control**: the flow rate of fuel into an internal combustion engine may be controlled by simple on/off solenoid valves if those valves are rapidly pulsed on and off. The pulse frequency is in the order of hundreds of Hertz.
- **Class D amplification**: instead of gradually throttling a transistor on and off to reproduce a waveform from a steady DC source voltage, we may rapidly *pulse* the transistor on and off to achieve the same. The pulse frequency must be far greater than the waveform's frequency in order to create a faithful reproduction – for example, to synthesize a 60 Hz waveform the PWM pulse frequency must be in the order of several thousand Hertz.

3.5 Generating PWM signals

A PWM signal is quite easy to generate from an analog "reference" voltage signal. All we need to do is use an integrated circuit called a *comparator* to compare the analog signal against a sawtooth- or triangle-wave voltage having peak-to-peak amplitude values matching the expected high and low range limits of the analog signal. Comparators[3] are hybrid analog/digital integrated circuits designed to compare two analog voltage signals against each other and generate a single digital output that is either "high" or "low" depending on which of the two analog input signals has the greater voltage:

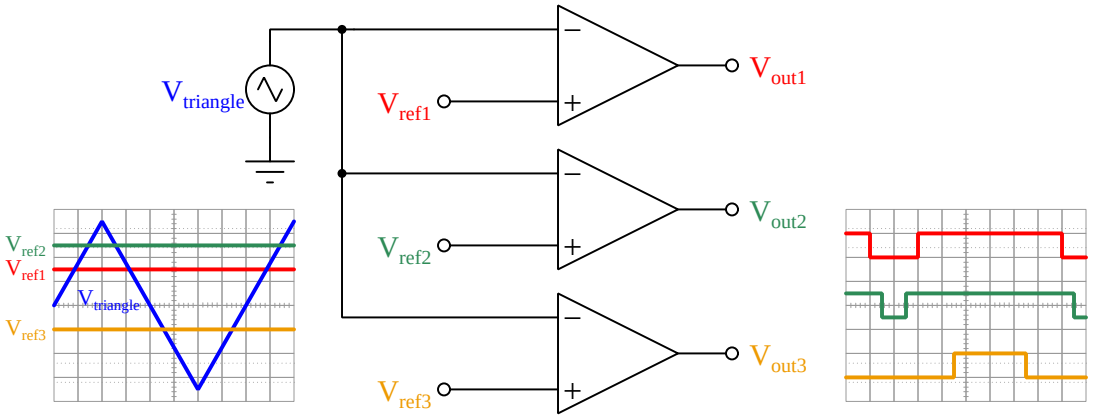


Whenever V_{ref} exceeds $V_{triangle}$, the comparator's output goes "high" (on for any ground-referenced load); whenever V_{ref} is less than $V_{triangle}$, the output goes "low" (off for any ground-referenced load). The higher V_{ref}, the longer the output remains "high" and the shorter it remains "low" during the cycle. Thus, the amplitude of V_{ref} determines the duty cycle of the pulse output, while the frequency is set by the frequency of the triangle-wave source.

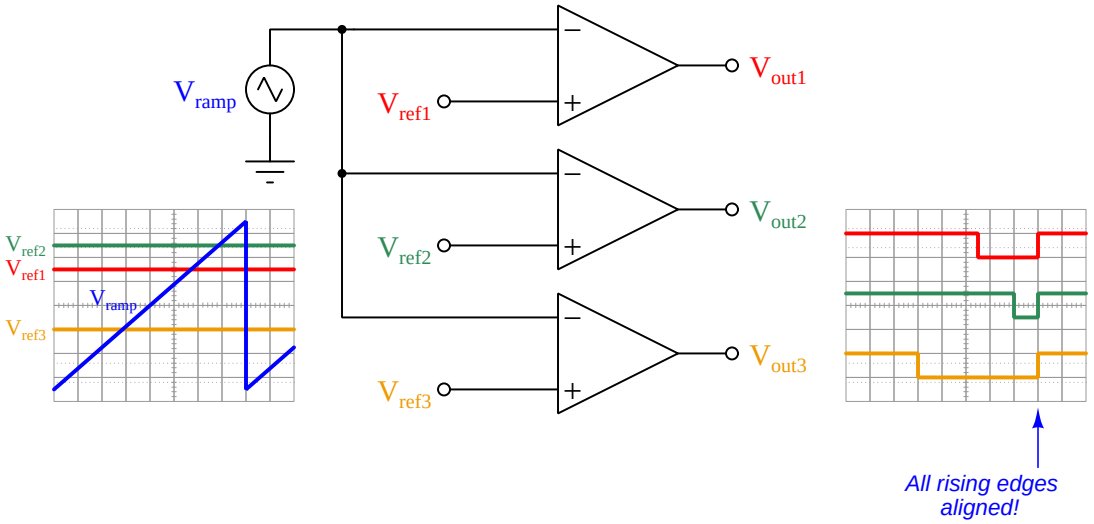
In order for this circuit to actually control power to a load, we need a transistor or other form of electrically-driven switch to handle the load's current which is nearly always greater than what the comparator is able to source or sink.

[3] To see how comparators respond to various input voltages, refer to the Case Tutorial section 2.1 beginning on page 8.

Some applications benefit from having multiple PWM signals all having precisely the same frequency. This is thankfully very easy to do with one triangle-wave oscillator and multiple comparators:



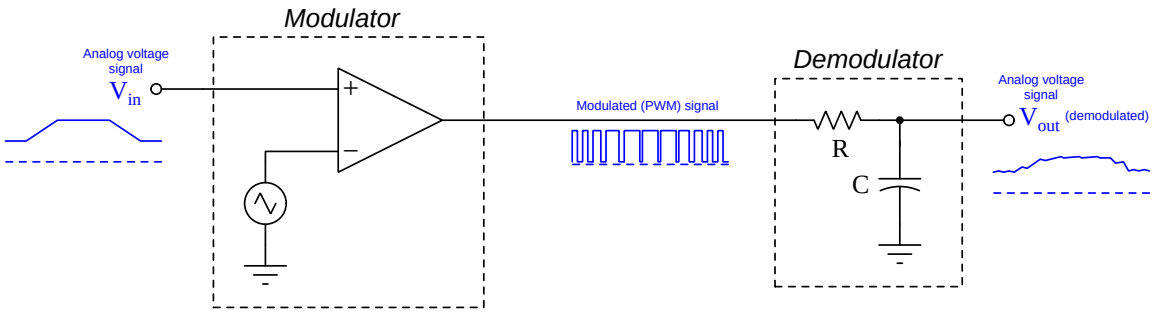
If there is a need for either the rising or the falling edges of these PWM signals to all be synchronized, simply replacing the triangle-wave signal with a ramp signal will ensure this happens:



In the example shown above, the gradual climb and rapid fall of the ramp waveform ensures alignment of the PWM pulses at their falling edges. If we wished the PWM pulses to be aligned at their rising edges instead, we could substitute a ramp waveform that rises rapidly and falls gradually.

3.6 PWM signal demodulation

If we consider pulse width modulation as a means of encoding *information* in a pulse signal, as opposed to merely a means of controlling power delivered to a load, we may consider the comparator circuit as a *signal modulator* and also apply low-pass filtering as a means of *demodulating* the pulse stream to end up with the original information:



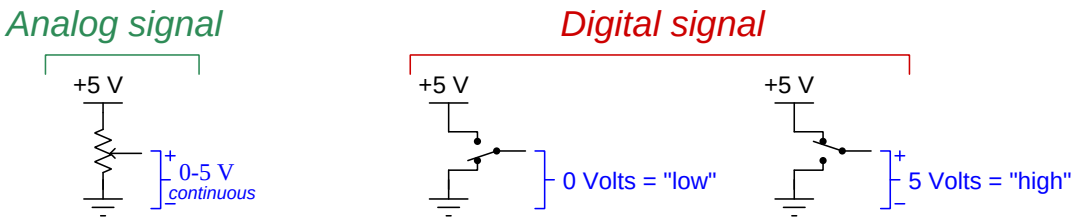
Any analog DC voltage applied to V_{in} generates a PWM signal from the comparator which will be "averaged" by the RC low-pass filter (also called a *passive integrator*) to become a "smoothed" pulsing signal resembling the original DC voltage signal. In this circuit, the time constant (τ) of the RC filter network substitutes for the mechanical inertia of the motor driven by pulse-width-modulated power, averaging the "on" and "off" states of the pulse signal into an analog voltage somewhere between full and zero based on time. The cutoff frequency of this filter must be chosen to be much lower than the PWM frequency, in order to filter out as much of the harmonics as possible and only leave the DC average value for V_{out}. At first, this process may seem useless, but it is actually very useful as a way of transmitting analog information over a communications channel that is only able to convey discrete (on/off) pulses.

A similar application for PWM is extracting an analog output signal from a *microcontroller*[4] or other digital circuit capable only of on/off states. Pulse-width modulation may be thought of as a clever way to encode analog information in any signal capable only of discrete amplitude states: even though the voltage or current is limited to "on" and "off" states, we may represent a continuously-variable signal in the form of the *ratio* between "on" time versus total "on + off" time.

[4] A "microcontroller" is a microprocessor combined with all necessary peripheral circuitry to form a functional computer on a single wafer ("chip") of semiconductor suitable for integrated circuit packaging. Many microcontrollers lack a digital-to-analog converter, but with judicious pulsing of an output pin and a passive low-pass filter network attached, the same effect may be realized: achieving analog signal capability in a device only able to switch between "high" and "low" states.

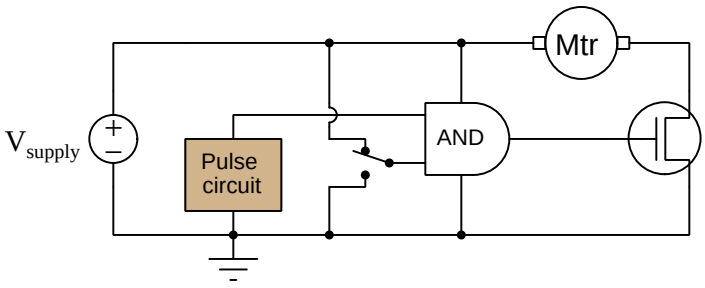
3.7 Inhibiting and overriding PWM signals

An *analog* signal is one capable of continuous variability, whereas a *digital* signal is one confined to a set of discrete states. A potentiometer producing a variable DC voltage proportional to its wiper position is a good example of an *analog voltage signal* source, while a SPDT switch producing either a "low" or "high" DC voltage is a good example of a *digital voltage signal* source:

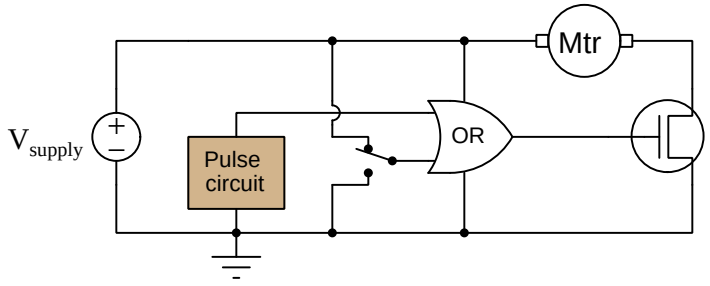


We know, however, from our study of pulse-width modulation that it is possible to synthesize an analog quantity from a discrete (low/high) signal by rapidly pulsing that signal between its two digital states. In this case the *amplitude* of the signal is still digital in nature, but the *duty cycle* of the signal is continuously variable anywhere between 0% and 100% inclusive. In other words *time* is the domain across which a digital signal still enjoys continuous variability.

The fact that PWM signal amplitudes are still digital in nature means we may pass PWM signals through *logic gates* for various effect. In the circuit below we see a PWM-based motor control where an AND gate provides an *inhibit* function for the motor: flipping the SPDT switch to the downward position overrides the pulse circuit's control of the power transistor, forcing the motor off regardless of the PWM duty cycle. In the switch's upward position, the motor runs normally:



Similarly, we could use an OR gate instead to override the PWM signal's control over the motor, in this case a "high" signal from the SPDT switch forcing the motor to run at full speed regardless of the pulse signal's duty cycle, and a "low" signal enabling the motor to run normally:

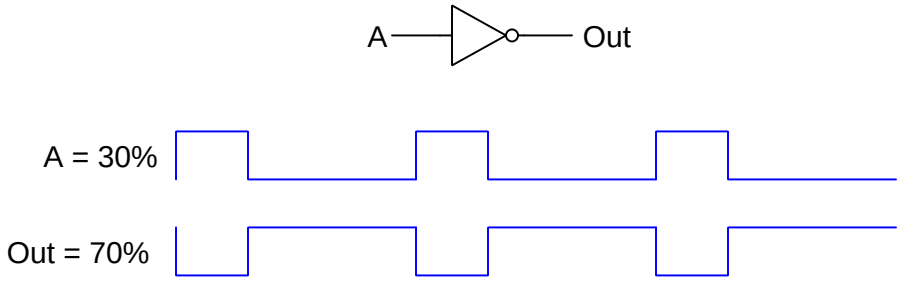


3.8 Duty cycle arithmetic

Pulse-width modulation uses digital (on/off) signals to encode analog (on/off *time*) information, and as we've already seen these signals may be passed through digital logic gates which of course are designed to handle on/off signals. Some very interesting things happen to the analog information, though, when PWM signals pass through different types of digital logic gates. We will explore some of these effects in this section.

3.8.1 PWM signal through a NOT gate

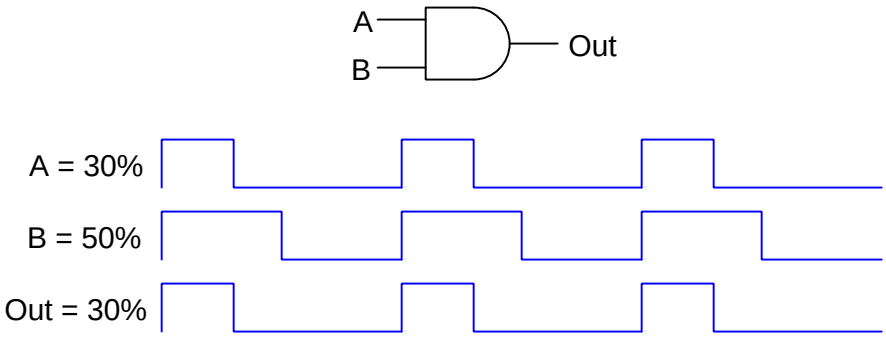
If we pass a PWM signal through an inverter (or *NOT*) gate, what we find is the input duty cycle gets translated into its *complement*:



In this example, we see an input PWM signal with a duty cycle of 30% get converted into an output PWM signal with a duty cycle of 70%. Those two percentages, being complementary to each other, when added together must equal 100%.

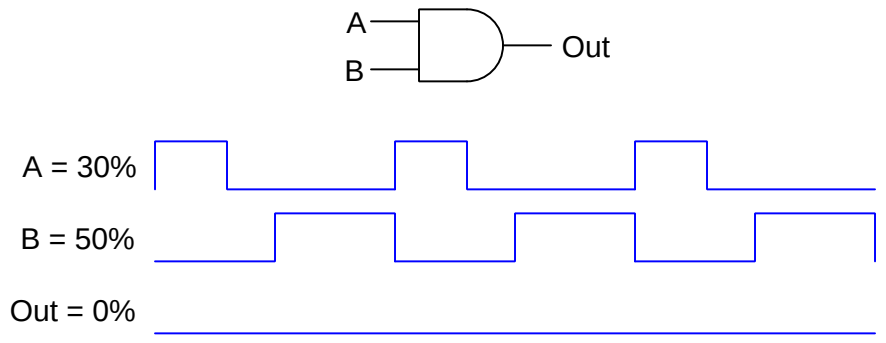
3.8.2 PWM signals through an AND gate

If we pass two PWM signals through an *AND* gate, both of those signals having the same frequency and being synchronized at their rising edges, what we find is that the output signal will have a duty cycle equal to the lesser of the inputs' duty cycle values:



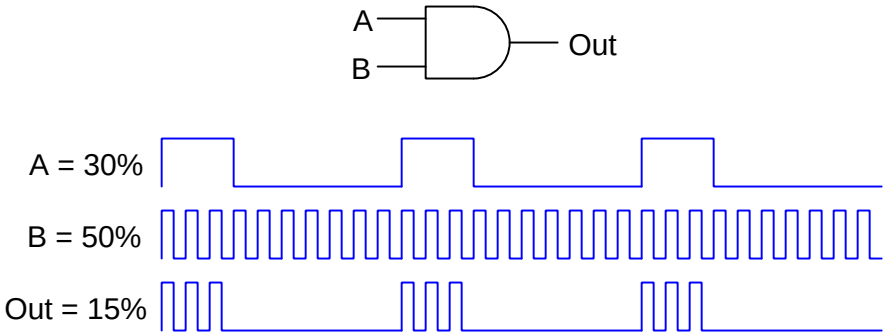
In this example signal A has a duty cycle of 30% while signal B has a higher duty cycle of 50%. Both signals have their rising edges[5] synchronized, and when this condition is met the resultant output signal's duty cycle must be the lesser of the two inputs'. This phenomenon is really the same as the *inhibit* signal example seen in the previous section using an AND gate: since all it takes to force the output of an AND gate "low" is to have any of its inputs "low" as well, whichever input signal spends more time "low" will override the other in that state. Therefore, the signal with the lowest-valued duty cycle ends up controlling the output. In control systems, this functionality is often referred to as *low-select*, the circuit outputting the signal with the lowest value from a selection of multiple inputs.

Again, it must be stressed that this low-selection functionality of an AND gate only works when the two PWM signals are synchronized with each other. Here's an example showing the same two duty cycle values resulting in a "dead" (0% duty cycle) output when that requisite synchronization is missing:



[5] This circuit will also produce the proper output duty cycle if both signals have their *falling* edges synchronized. What is important is that they must either both rise at the same time (every time) or fall at the same time (every time).

If we make one of these signals have a much greater frequency than the other, we encounter a different phenomenon. Now, the duty cycle of the higher-frequency signal subdivides the duty cycle of the lower-frequency signal:



Here, the 50% duty-cycle signal makes it so the 30% duty-cycle signal is "high" only for half the amount of time it would have been without the influence of the AND gate, thereby scaling down signal A's duty cycle of 30% to a mere 15%. What we see here is *multiplication* of duty cycle values, with the output signal's duty cycle being the product of the two input signals' duty cycles:

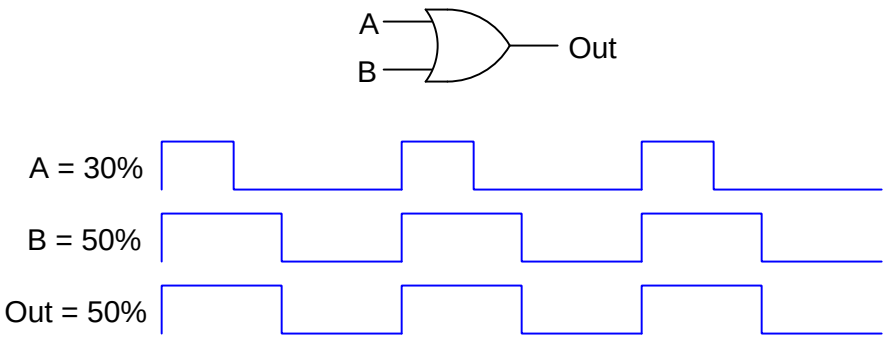
$$\text{Output} = AB$$

Where,
Output = Duty cycle of the AND gate's output pulse signal
A = Duty cycle of the AND gate's first input pulse signal
B = Duty cycle of the AND gate's second input pulse signal

Synchronization of these signals' rising or falling edges is not necessary for reliable duty cycle multiplication as it was for duty cycle low-selection, but it is important that the high-frequency signal proceeds through at least one full on/off cycle within a span of time equal to or less than the narrowest "high" time of the low-frequency signal. This ensures that each and every "high" interval of the low-frequency PWM signal experiences at least one full duty cycle of the high-frequency signal. Ideally, the high-frequency signal's frequency will be *substantially* greater than this so that every "high" interval of the low-frequency signal experiences multiple cycles of the high-frequency signal. The principle at work here is that in order for the duty cycle of one pulse signal to perfectly proportion the other, there must be a whole number of cycles of the high-frequency signal during each and every "high" interval of the low-frequency signal – having a fractional number of cycles would mean that interval gets modulated either higher than or less than the high-frequency signal's duty cycle, the amount of this error dependent on how large that fraction of a cycle is and the sign of this error dependent on where the rising and falling edges of the low-frequency signal happen to fall relative to the high-frequency signal's. The more cycles of one signal can fit into each "high" time of the other signal, the less this error will proportionately be.

3.8.3 PWM signals through an OR gate

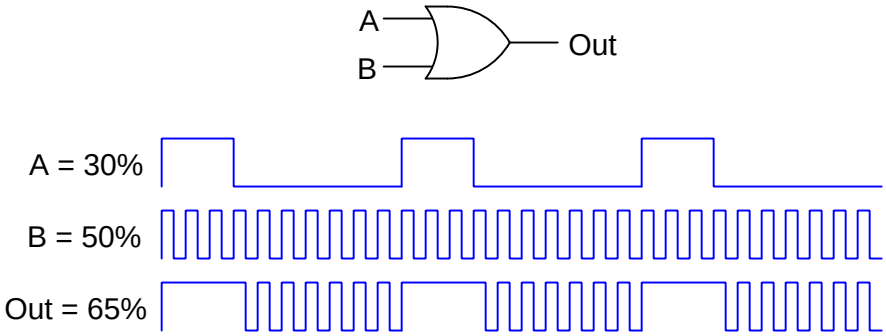
If we pass two PWM signals through an *OR* gate, both of those signals having the same frequency and being synchronized at their rising edges, what we find is that the output signal will have a duty cycle equal to the greater of the inputs' duty cycle values:



In this example signal A has a duty cycle of 30% while signal B has a higher duty cycle of 50%. Both signals have their rising edges[6] synchronized, and when this condition is met the resultant output signal's duty cycle must be the greater of the two inputs'. This phenomenon is really the same as the *override* signal example seen in the previous section using an OR gate: since all it takes to force the output of an OR gate "hight" is to have any of its inputs "hight" as well, whichever input signal spends more time "high" will override the other in that state. Therefore, the signal with the highest-valued duty cycle ends up controlling the output. In control systems, this functionality is often referred to as *high-select*, the circuit outputting the signal with the highest value from a selection of multiple inputs.

[6] This circuit will also produce the proper output duty cycle if both signals have their *falling* edges synchronized. What is important is that they must either both rise at the same time (every time) or fall at the same time (every time).

If we make one of these signals have a much greater frequency than the other, we encounter a different phenomenon. Now, the duty cycle of the higher-frequency signal forces a minimum duty cycle for the output signal:



Here, the higher-frequency signal with its 50% duty cycle controls the output when the lower-frequency signal is "low", but when the lower-frequency signal is "high" it completely overrides the other signal to force the output high for that same duration of time. In other words, the aggregate duty cycle for the OR gate's output signal will be 30% (from the A signal) *plus* 50% of the A signal's complement (50% of 70% = 35%). Generalized in mathematical form:

$$\text{Output} = A + B - AB$$

Where,
Output = Duty cycle of the OR gate's output pulse signal
A = Duty cycle of the OR gate's first input pulse signal
B = Duty cycle of the OR gate's second input pulse signal

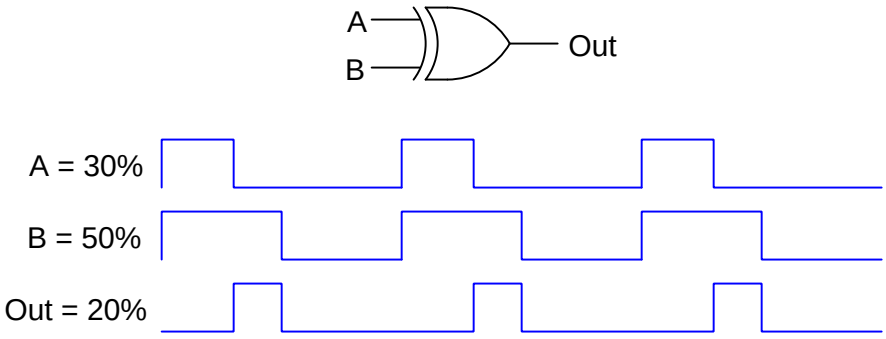
As with the AND gate scenario with two PWM signals of differing frequency, ratio of frequencies must be fairly large in order for the effect to be consistently accurate. At first this may not seem to have any practical use, but it actually bears similarity to the low/high frequency PWM usage of AND gates seen in the previous subsection of this Tutorial. In the next two paragraphs we will compare the behaviors of AND versus OR gates when receiving two PWM signals of vastly different frequency.

In the AND gate example where the output's duty cycle is the product of the two input signals' duty cycles (Output = AB), you could think of one signal as setting an upper limit for the other, and the other signal proportioning the output duty cycle anywhere from zero to that upper limit. If we imagine one signal having a duty cycle of 30% while the other signal's duty cycle gradually increases from 0% to 100%, the output's duty cycle will sweep from 0% to 30% in direct proportion with the 0-100% signal.

With this OR gate example, however, one signal establishes a lower limit for the output's duty cycle. Imagine signal A having a fixed duty cycle of 30% while signal B sweeps from 0% to 100% duty cycle: in this case, the output signal's duty cycle will sweep from 30% to 100% in direct proportion to signal B.

3.8.4 PWM signals through an XOR gate

If we pass two PWM signals through an *Exclusive-OR* (XOR) gate, both of those signals having the same frequency and being synchronized at their rising edges, what we find is that the output signal will have a duty cycle equal to the difference between the inputs' duty cycle values:



In this example signal A has a duty cycle of 30% while signal B has a higher duty cycle of 50%. Both signals have their rising edges[7] synchronized, and when this condition is met the resultant output signal's duty cycle will be the *absolute value of the difference* between the two inputs'.

$$\text{Output} = |A - B|$$

Where,

Output = Duty cycle of the XOR gate's output pulse signal
A = Duty cycle of the XOR gate's first input pulse signal
B = Duty cycle of the XOR gate's second input pulse signal

[7] This circuit will also produce the proper output duty cycle if both signals have their *falling* edges synchronized. What is important is that they must either both rise at the same time (every time) or fall at the same time (every time).

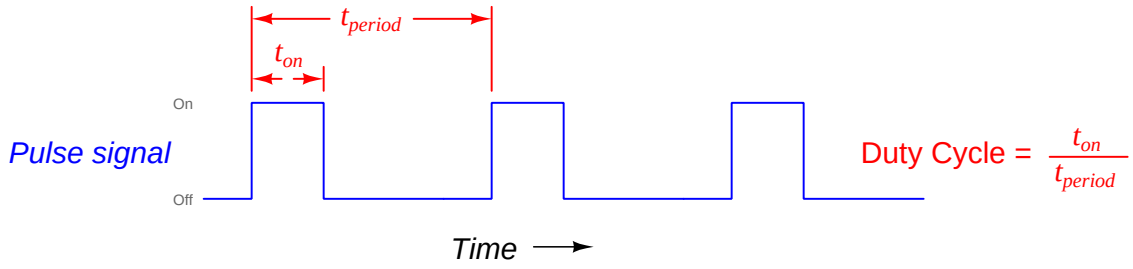
Chapter 4

Derivations and Technical References

This chapter is where you will find mathematical derivations too detailed to include in the tutorial, and/or tables and other technical reference material.

4.1 Derivation of PWM average power

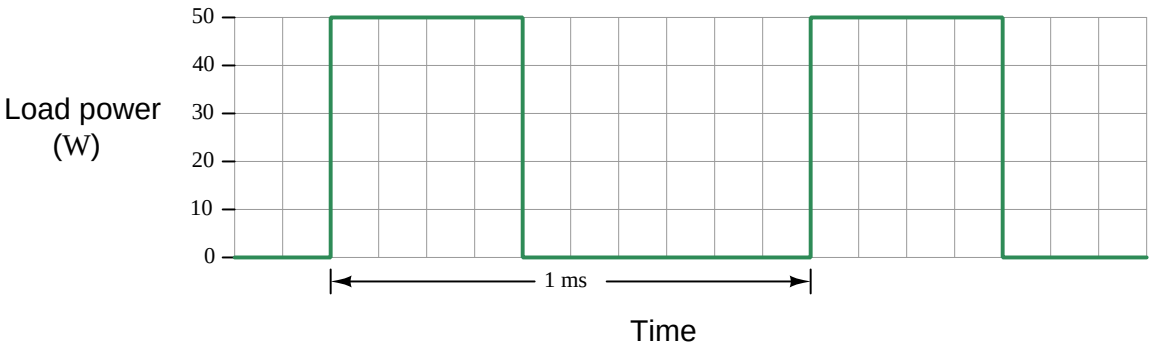
The *duty cycle* of a pulse waveform is the ratio between its "on" time versus its total period, usually expressed as a percentage:



If this waveform is the voltage across a PWM-controlled load, then calculating the average power of this load is as simple as calculating the "peak" load power (i.e. the amount of power dissipated by the load while the pulse is in its "high" state) and then multiplying that peak power by the duty cycle value. For example, if the peak voltage of our PWM waveform was 12 Volts and the load was a 5 Ohm resistor, then peak power would be $P = \frac{V^2}{R} = 28.8$ Watts. If the duty cycle happened to be 30%, then the load's average power dissipation would be 28.8 Watts $\times$ 30% = 8.64 Watts.

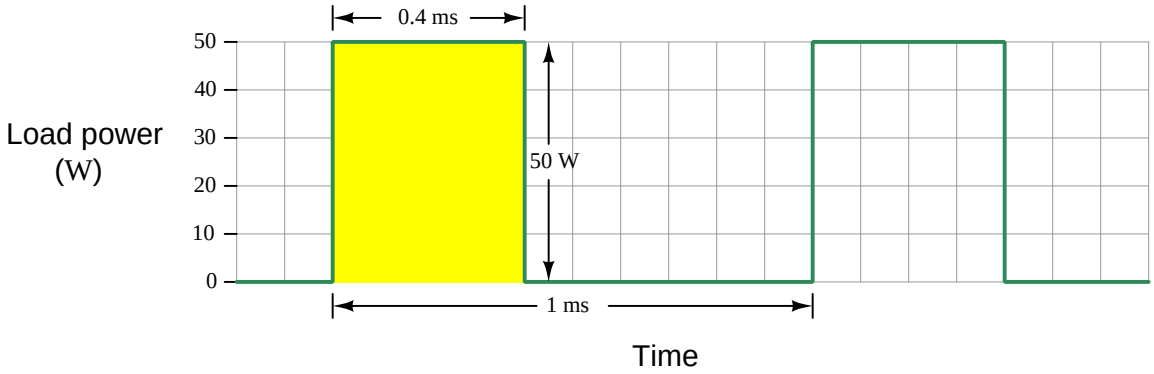
But *why* is this? Why is average power the simple product of peak power and duty cycle? The answer to this question is actually based on calculus, but a simple enough application of calculus that it will be understandable to anyone familiar with basic arithmetic and geometry.

Let's begin by plotting a PWM waveform, but do so with *load power* being the dependent variable and time being the independent variable. We will assume a scenario where the PWM frequency is 1 kHz (i.e. period is 1 ms), where the peak power is 50 Watts, and where the duty cycle is 40%:

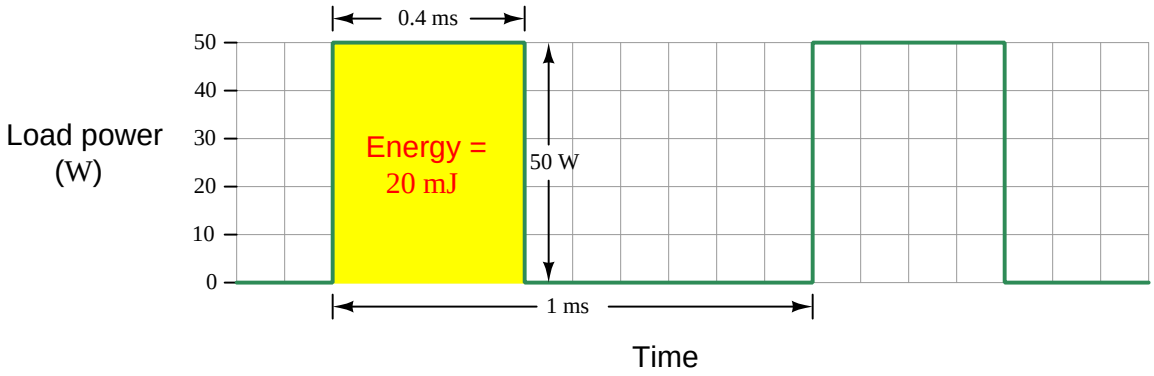


We already know that the average power in this case must be 20 Watts because the peak power is 50 Watts and the duty cycle is 40%. Let's dissect this scenario a bit more, though, to explore why this is the case. Our first step is recognizing the definition of power, which is *the rate of energy transfer over time*. In fact, 1 Watt of power is defined as 1 Joule of energy transferred from one location to another over a time of 1 second; i.e. Watts is equal to Joules per second.

If we use this graph in a creative way, we can actually calculate how many Joules of energy are delivered to the load during each of the pulse's "high" times. Knowing that Watts is equivalent to Joules per second, we can sketch the rectangular area enclosed by one of the "high" pulse times and label its height in Watts and its width in seconds:



Since we know power is energy divided by time (i.e. Watts equals Joules per second), then it must be true that energy is power multiplied by time (i.e. Joules equals Watts *times* seconds). Having labeled the height and width of one of the "high" pulse durations, we may determine the geometric area of that enclosed rectangle as the product of power and time. As we show here, 50 Watts multiplied by 0.4 milliseconds is 20 milliJoules of energy:



The value of the area enclosed by any mathematical function is known in calculus as the *integral*. Here, we have computed the integral of a 50-Watt peak pulse lasting 0.4 milliseconds as being 20 milliJoules. Our integral calculation is extremely easy because the geometric area we're computing just happens to be a rectangle, and rectangular areas are nothing more than the product of height and width. When you study integral calculus, one of the things you learn is how to compute the enclosed area for *any* given mathematical function, and this can get quite complex. Fortunately for us in this application, the computation of the function's integral is nothing more than a simple rectangular-area calculation.

Proper calculus expression of this integral is as follows – energy (E) delivered to the load from 0 milliseconds to 1 millisecond is the integral ($\int$) of load power (P) over time (dt):

$$E = \int_{t_{start}}^{t_{end}} P \, dt$$

$$E = \int_{0ms}^{1ms} (\text{50 W for 0.4 ms; 0 W for 0.6 ms}) \, dt$$

$$E = 20 \text{ mJ}$$

If the amount of energy delivered to the load during each of the pulse's "high" states is 20 milliJoules, then this is also the amount of energy delivered to the load per period since there is only one "high" state per period. In order to translate this energy-per-cycle value of 20 milliJoules into an *average rate of power* received by the load over long spans of time, we once again must recall the conceptual definition of power: *power is energy divided by time.* Since each period (cycle) of this waveform lasts 1 millisecond, and each period (cycle) delivers 20 milliJoules of energy to the load, the average power must be 20 milliJoules divided by 1 millisecond, which is an average of 20 Joules per second of time (i.e. 20 Joules of energy delivered to the load per 1000 cycles at 1 kHz), which is of course 20 Watts.

This is why average load power is peak load power multiplied by duty cycle: peak power represents the height of the energy "rectangles" enclosed by the power waveform. The duty cycle represents the width of each of those rectangles in proportion to the width of the whole cycle's period – in other words, duty cycle expresses *how wide each rectangle is* versus *how wide it could be* if the power were applied continuously. Therefore, duty cycle expresses the ratio of how much energy the load receives each cycle versus how much it could receive under full-power conditions.

Chapter 5

Animations

Some concepts are much easier to grasp when seen in *action*. A simple yet effective form of animation suitable to an electronic document such as this is a "flip-book" animation where a set of pages in the document show successive frames of a simple animation. Such "flip-book" animations are designed to be viewed by paging forward (and/or back) with the document-reading software application, watching it frame-by-frame. Unlike video which may be difficult to pause at certain moments, "flip-book" animations lend themselves very well to individual frame viewing.

5.1 Comparator generating PWM waveform

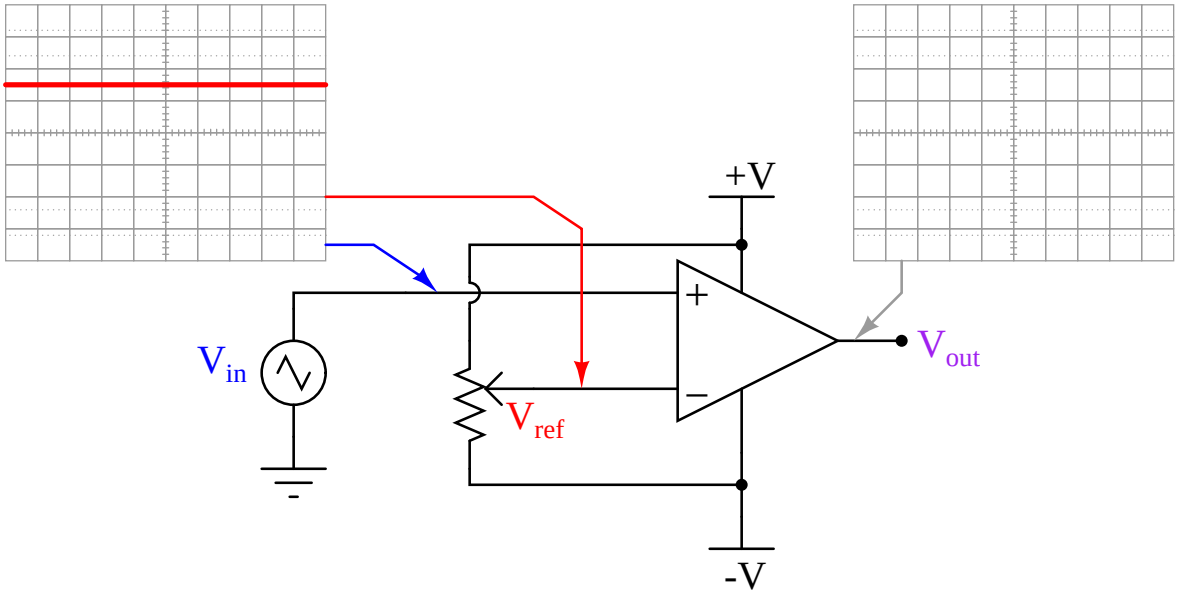
One very useful application of a comparator is to generate a *pulse-width modulation* (PWM) waveform from a DC signal, with that magnitude of the DC signal controlling the *duty cycle* (i.e. on-time versus period) of the output pulses.

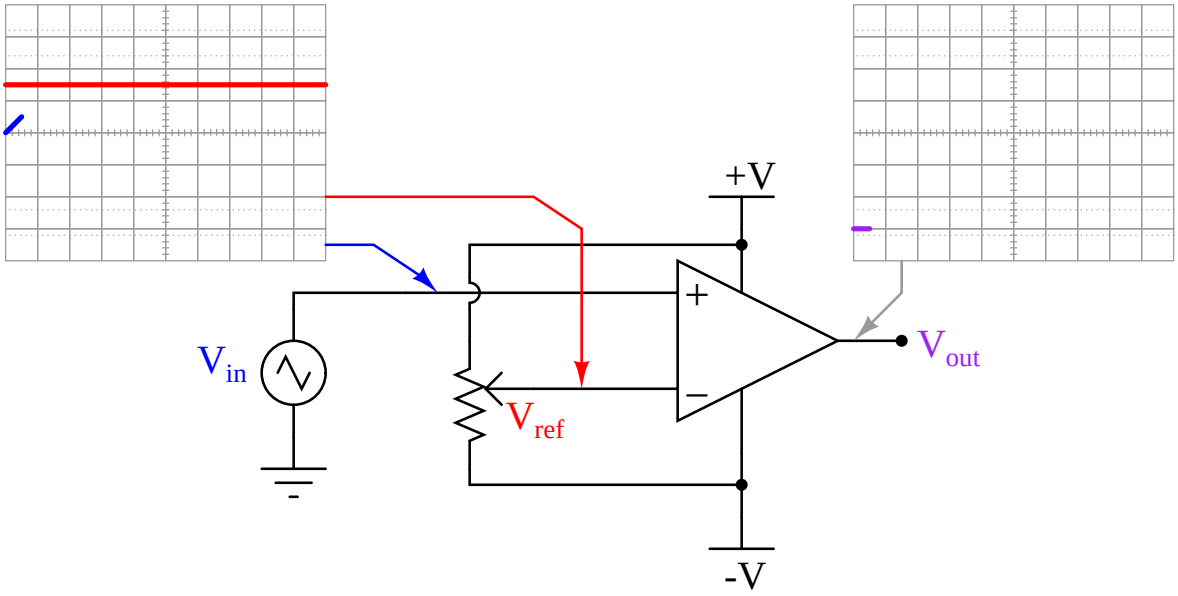
To do this, we connect the two inputs of a comparator to a varying (AC) signal source and the DC signal source, respectively. Ideally the AC source waveform should be linear: sawtooth or triangle waves work well for this purpose. The peak-to-peak magnitude of the AC waveform needs to span the expected range of the DC signal voltage.

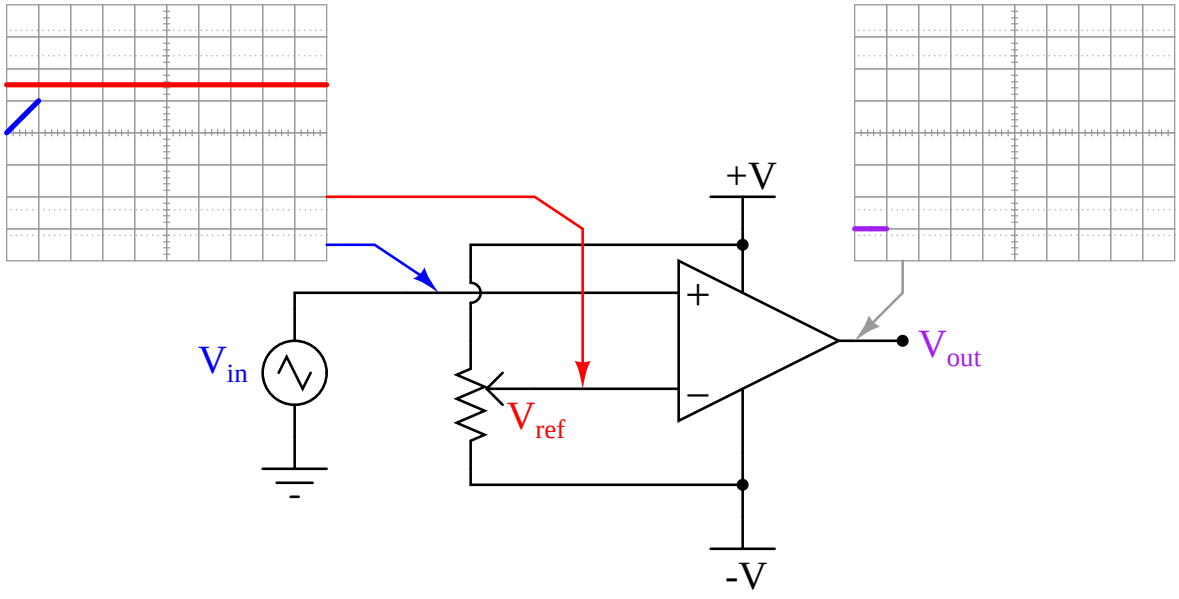
The AC source's frequency will define the frequency of the PWM output, while the DC voltage signal's magnitude defines the PWM duty cycle.

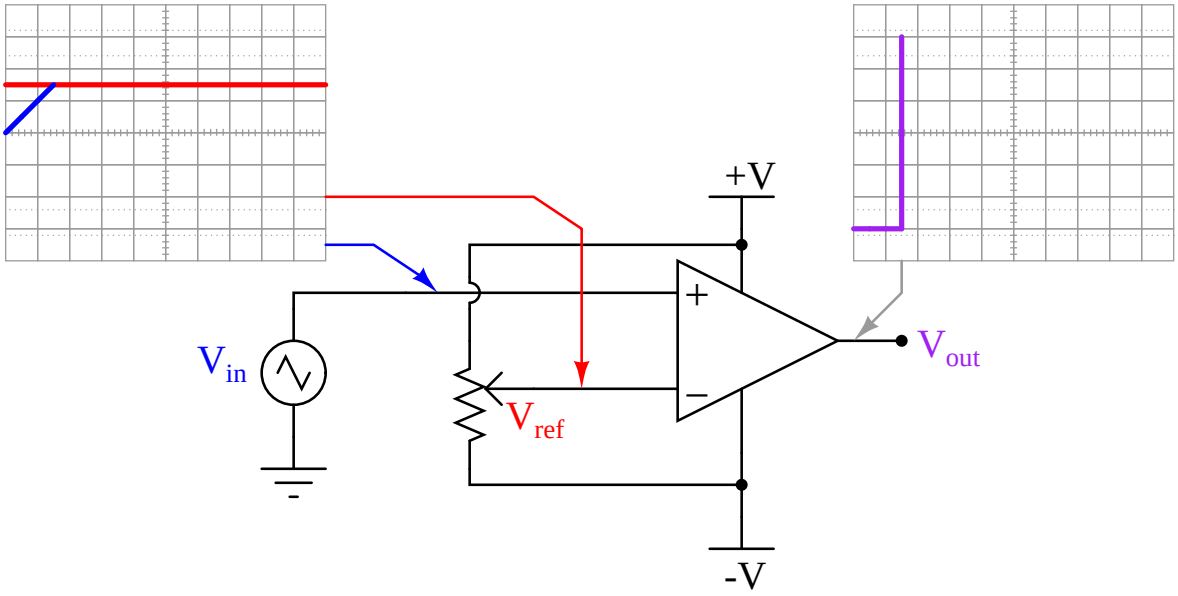
PWM pulse signals are very useful for emulating analog control using discrete-state final-control devices (e.g. transistors operated as on-off switches for minimal power dissipation).

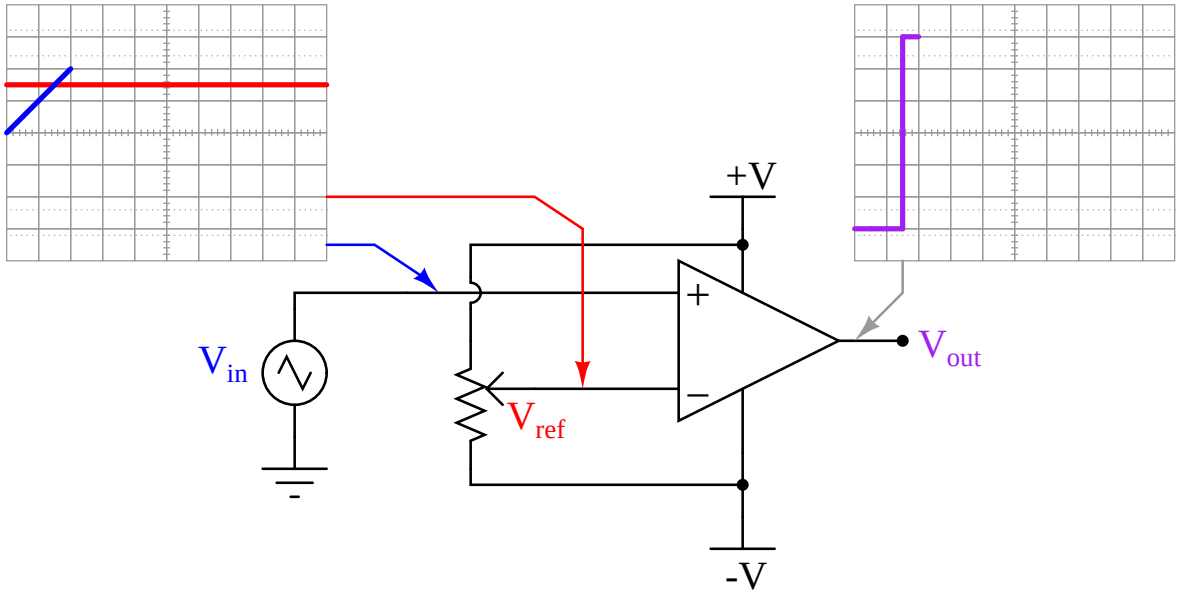
This animation shows the evolution of a PWM waveform for two different DC inputs, those input signal levels set by the wiper position on a potentiometer. The way this particular circuit is designed, a higher wiper position results in a lesser duty cycle. If the reverse characteristic is desired, simply swap + and − inputs on the comparator.

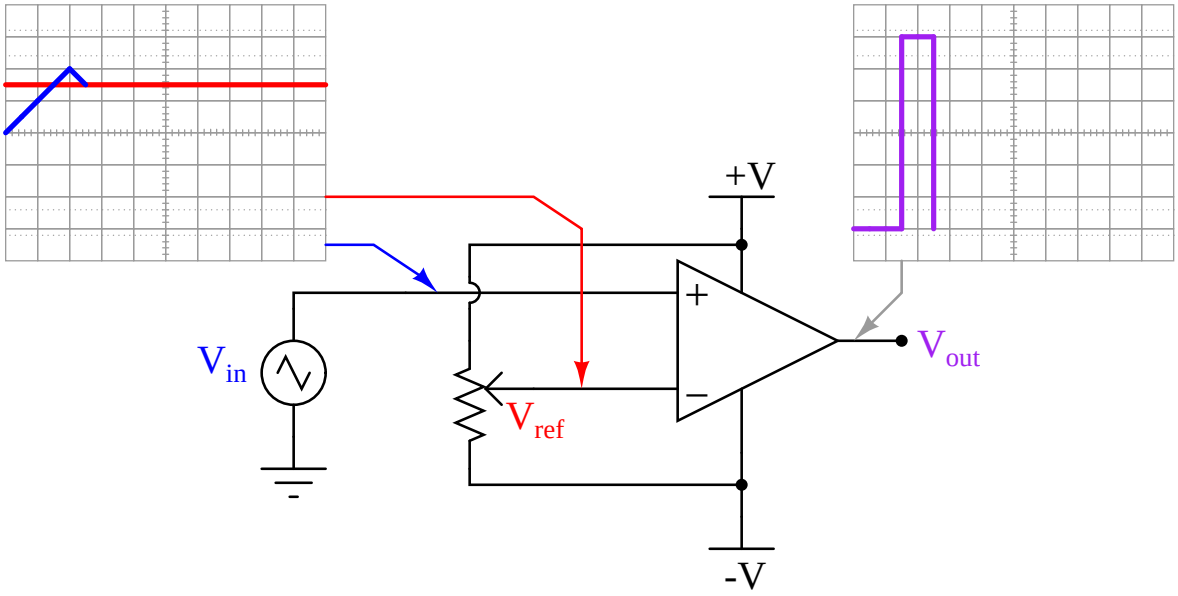

+V
V in
V ref
V out
-V

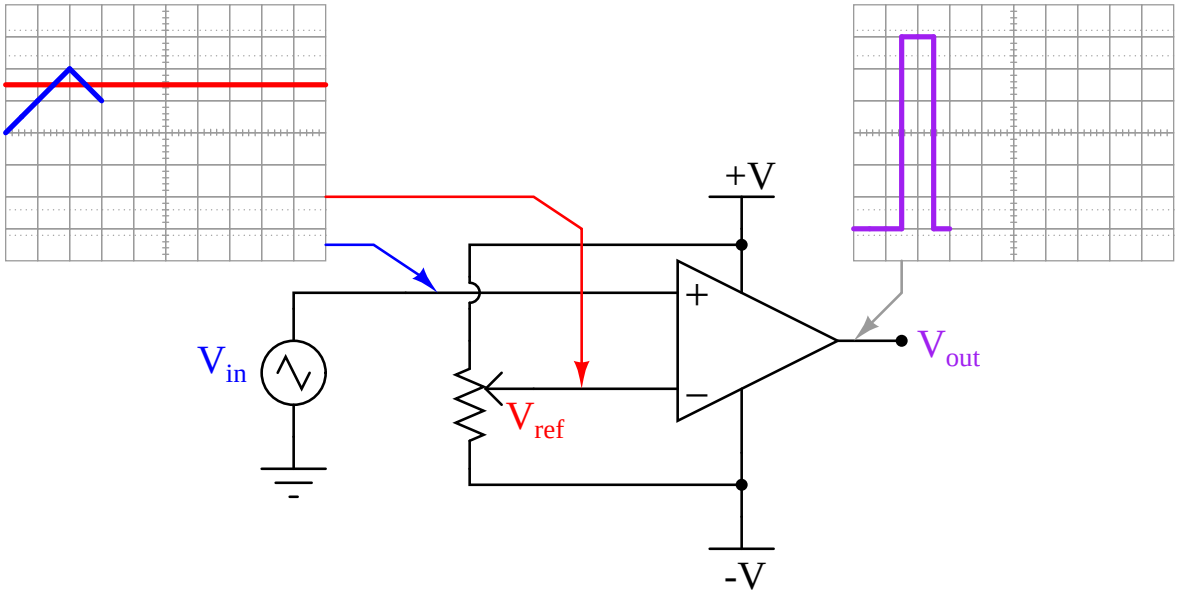

+V
V in
V ref
V out
-V

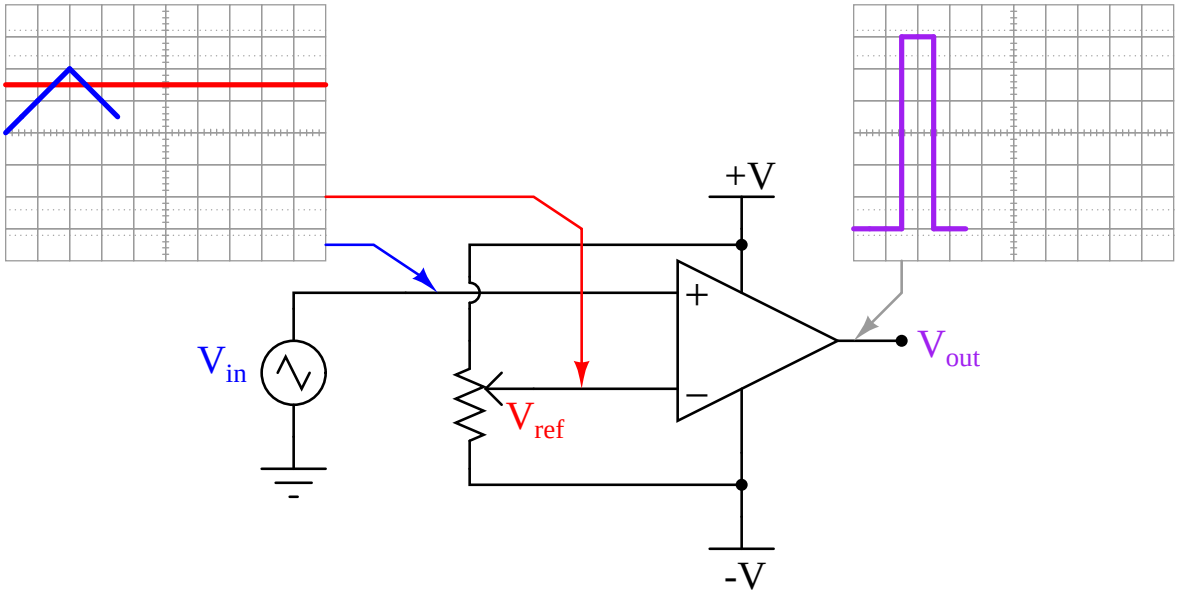

+V
V in
V ref
V out
-V

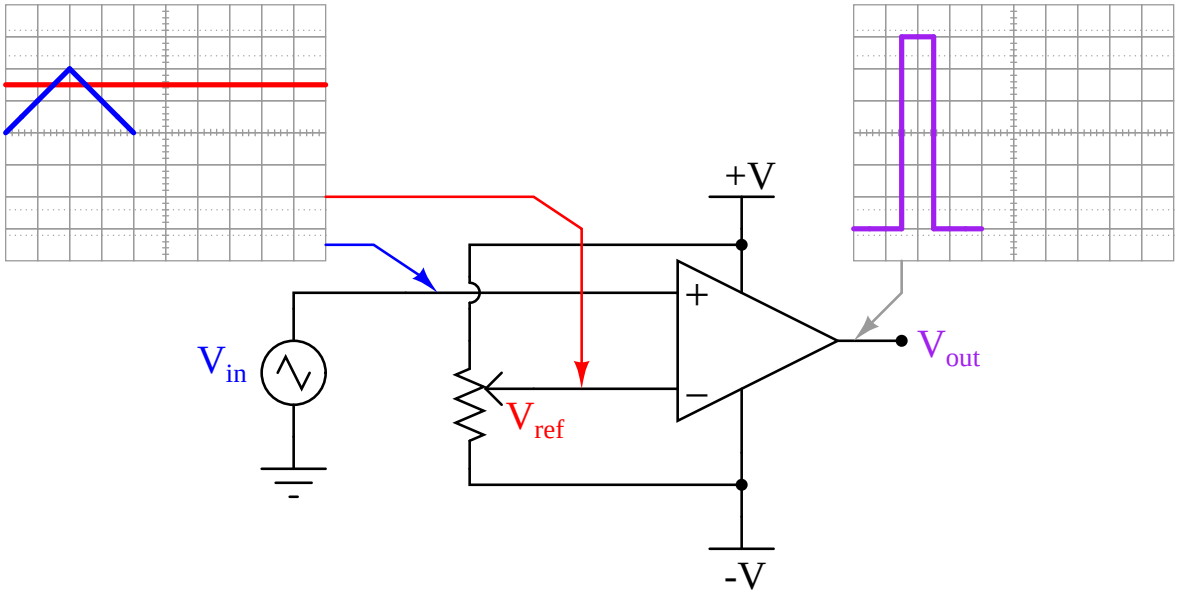

+V
V in
V ref
V out
-V

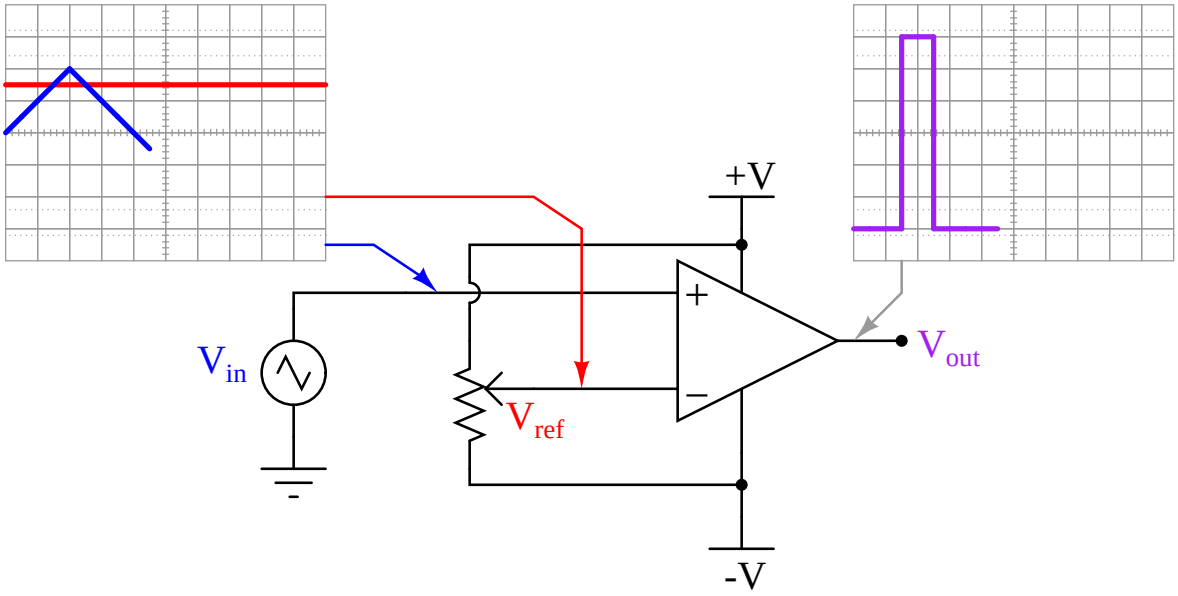

+V
+
−
V in
V ref
V out
-V

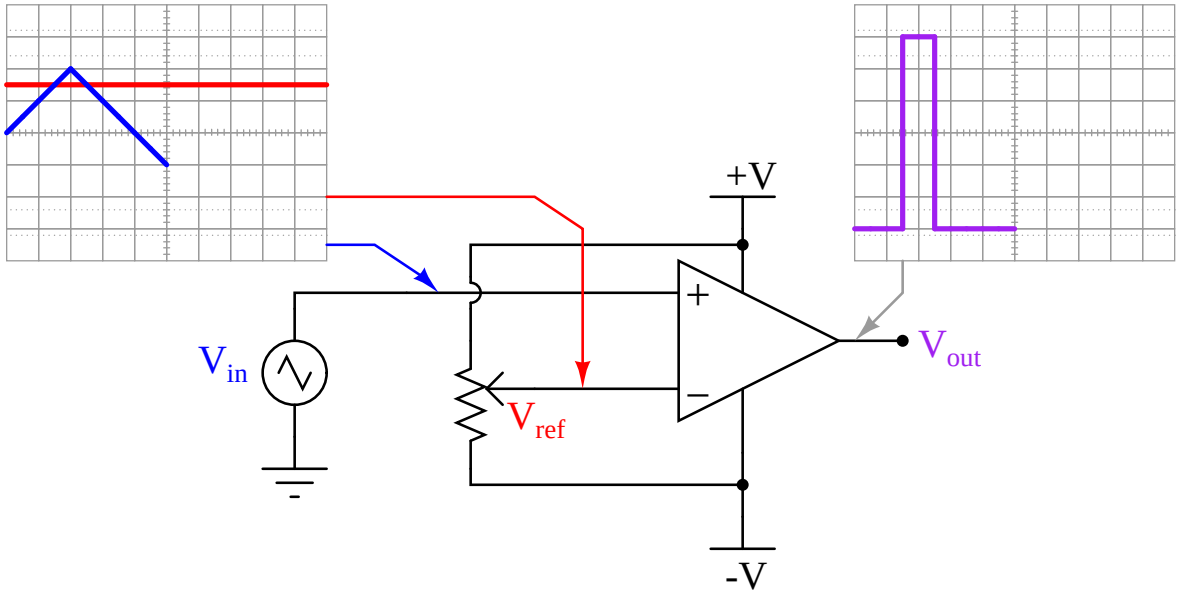

+V
V in
V ref
V out
-V

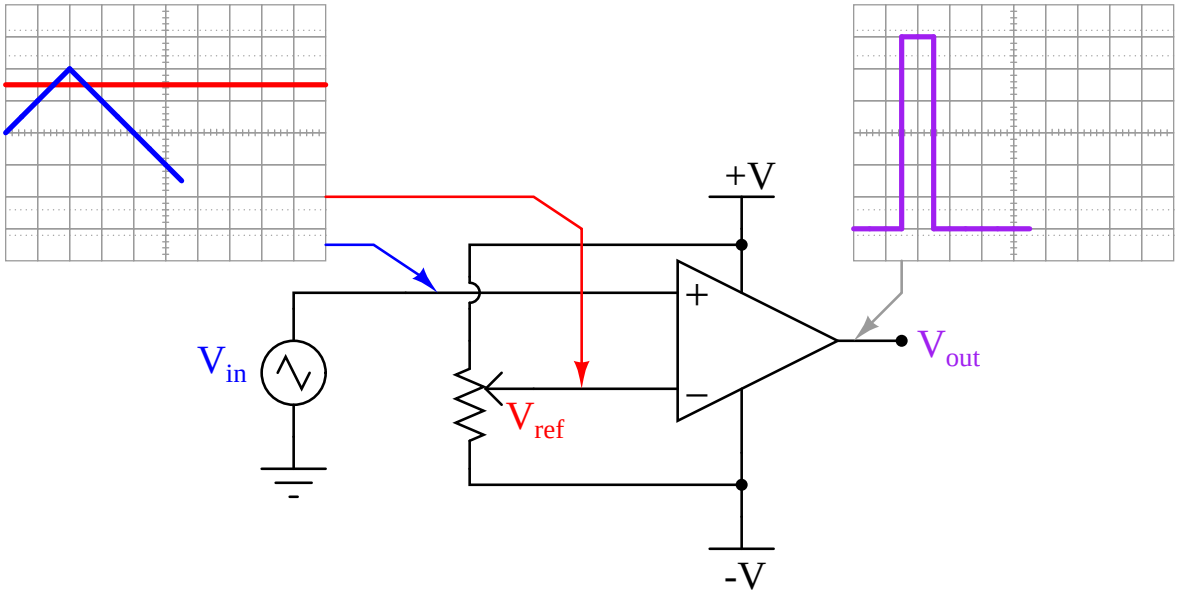

+V
V in
V ref
V out
-V

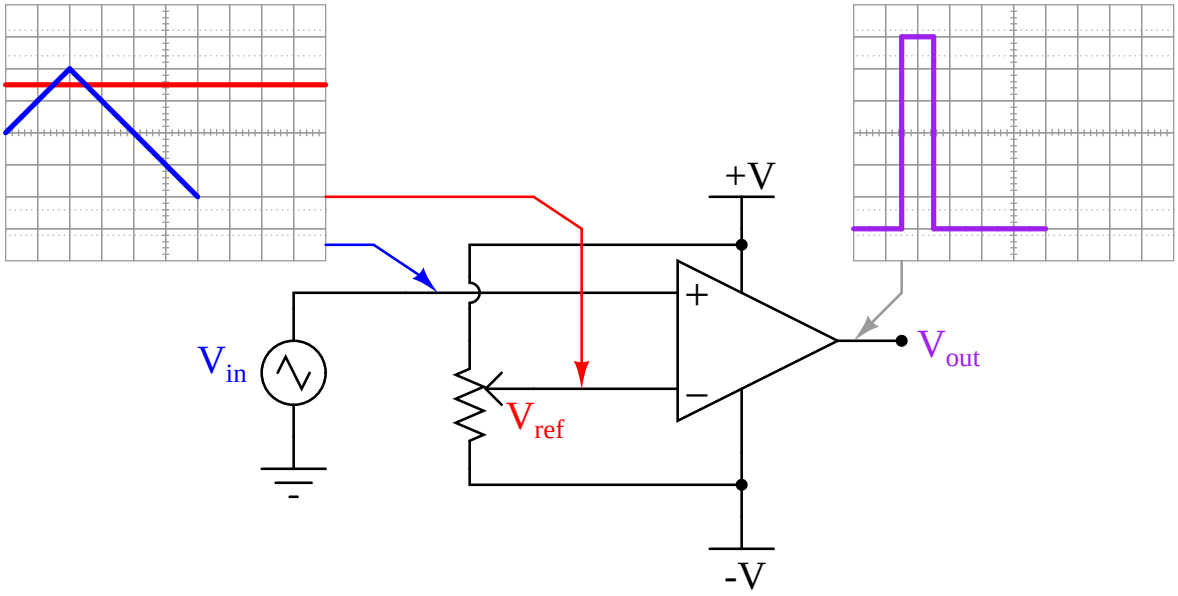

+V
V in
V ref
V out
-V

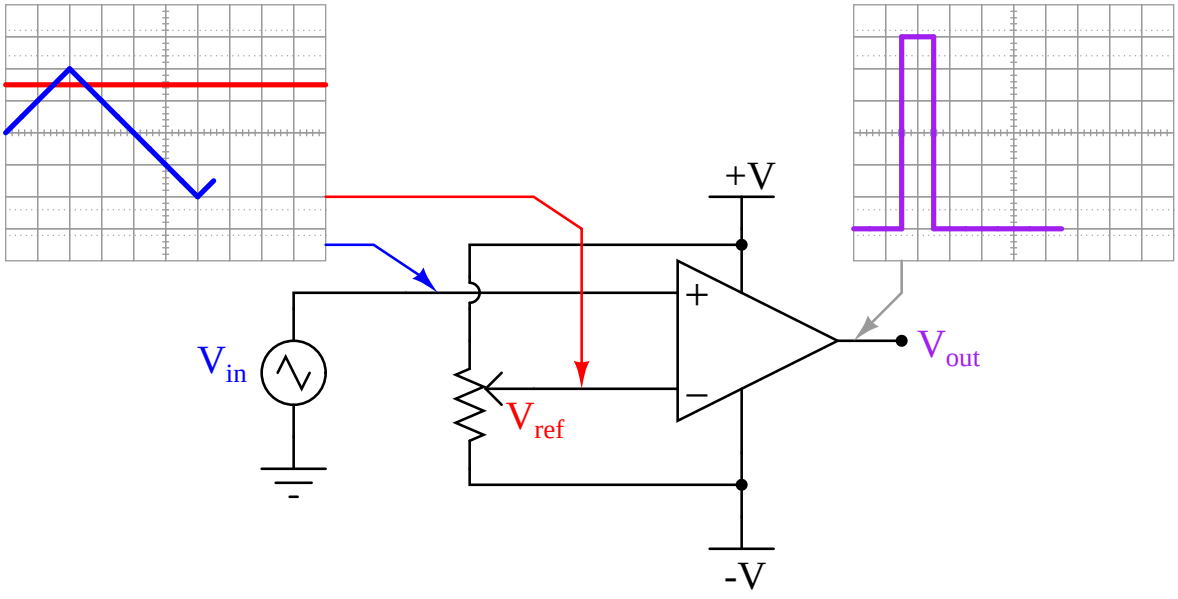

+V
V in
V ref
V out
-V

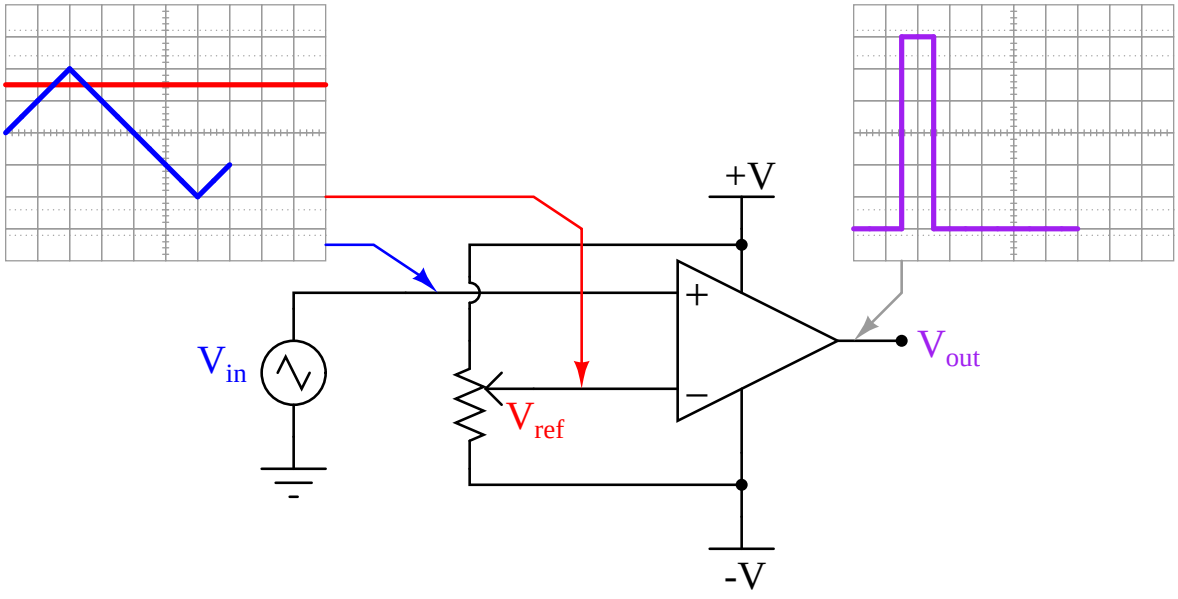

+V
V in
V ref
V out
-V

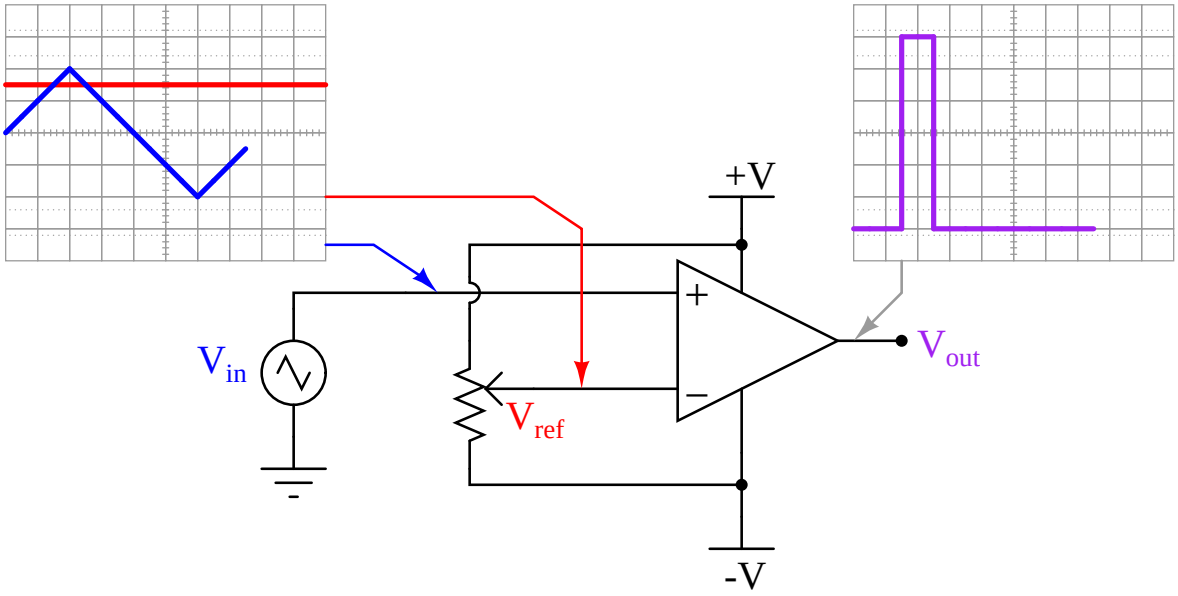

+V
V in
V ref
V out
-V

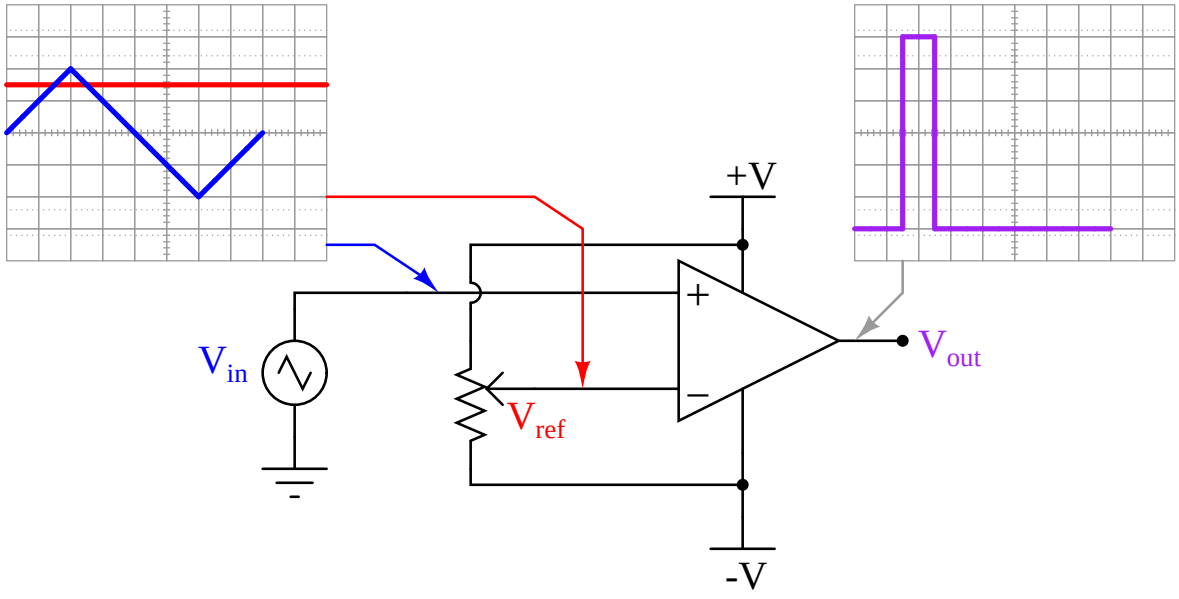

+V
V in
V ref
V out
-V

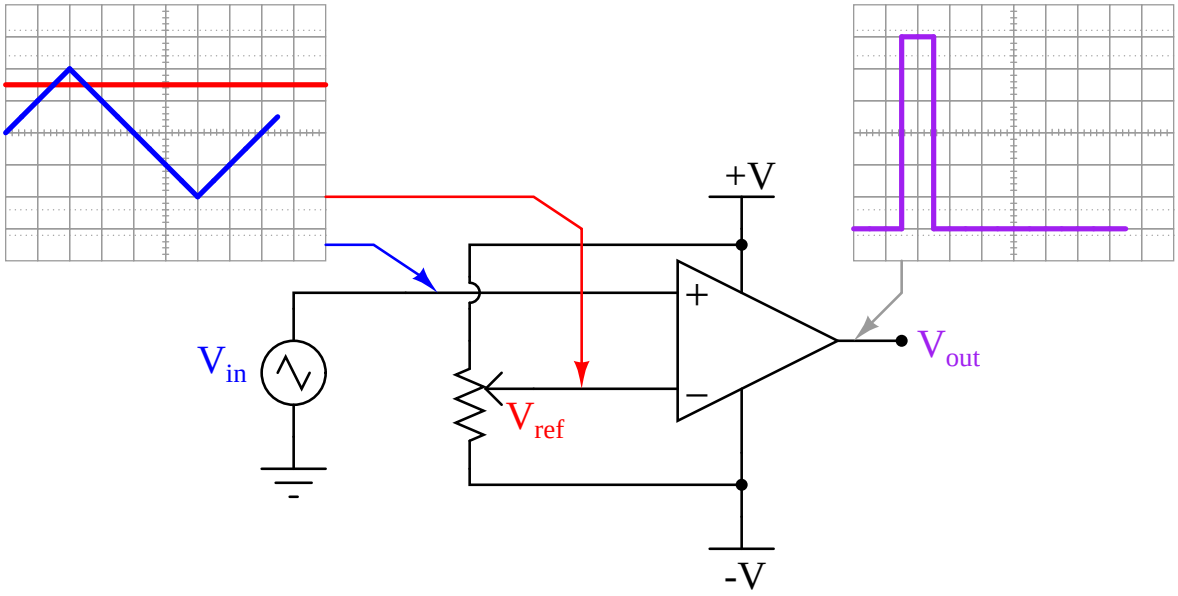

+V
V in
V ref
V out
-V

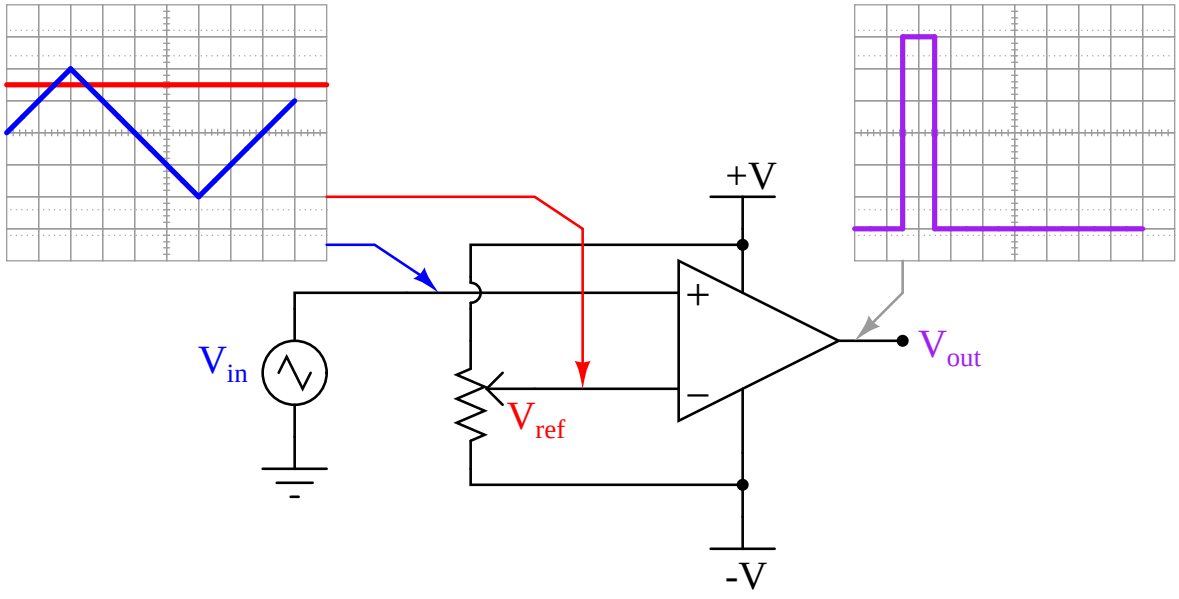

+V
V in
V ref
V out
-V

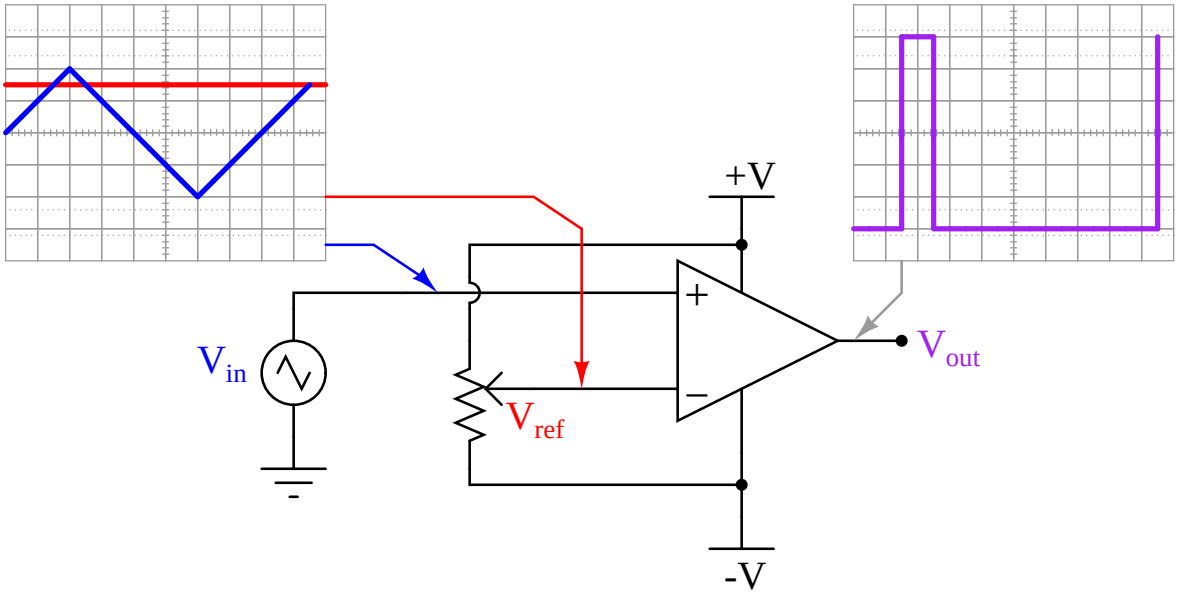

+V
V in
V ref
V out
-V

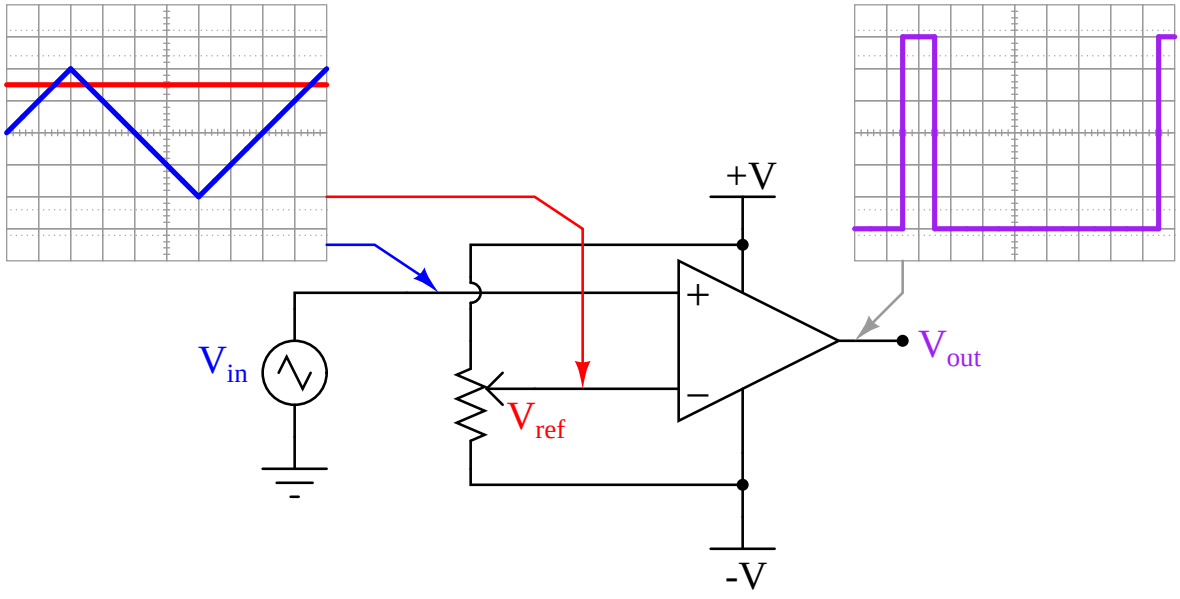

+V
V in
V ref
V out
-V

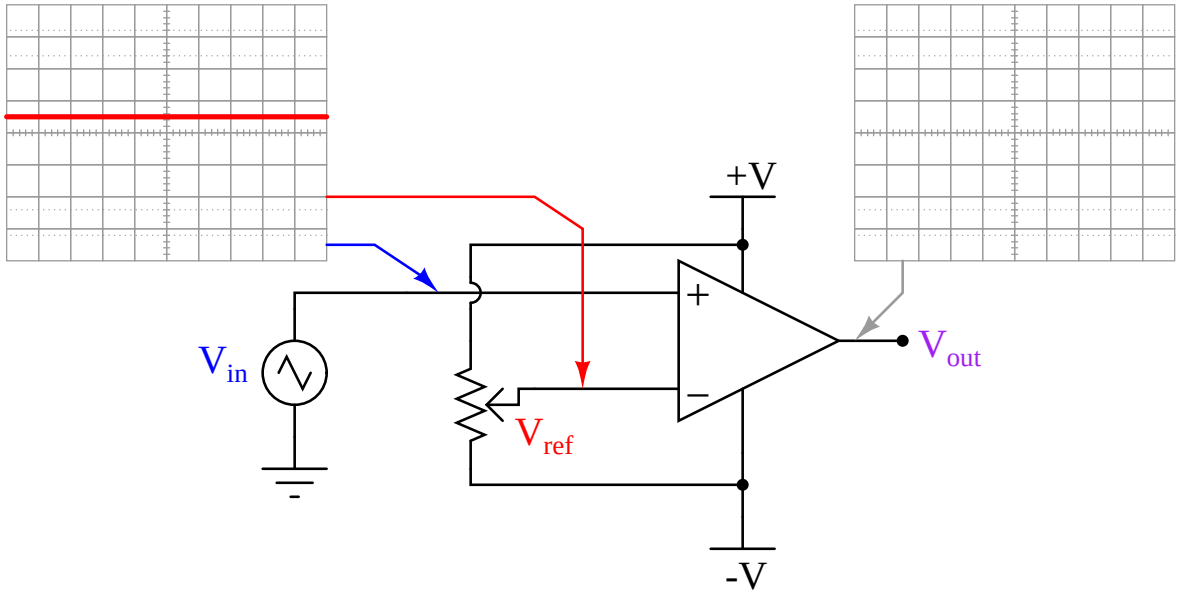

+V
V in
V ref
V out
-V

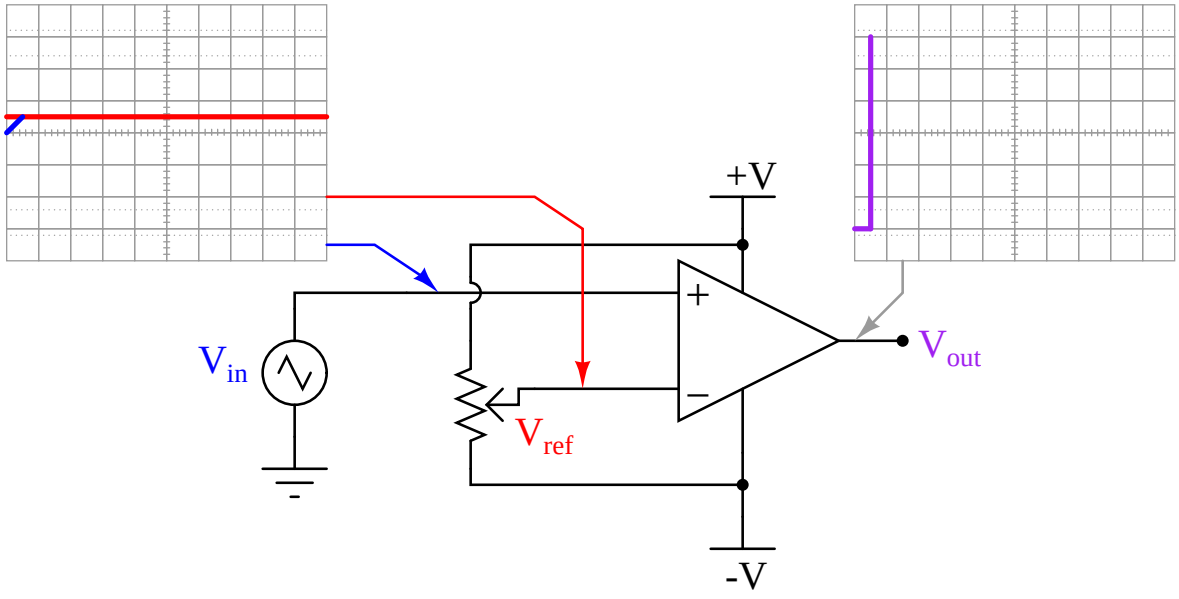

+V
V in
V ref
V out
-V

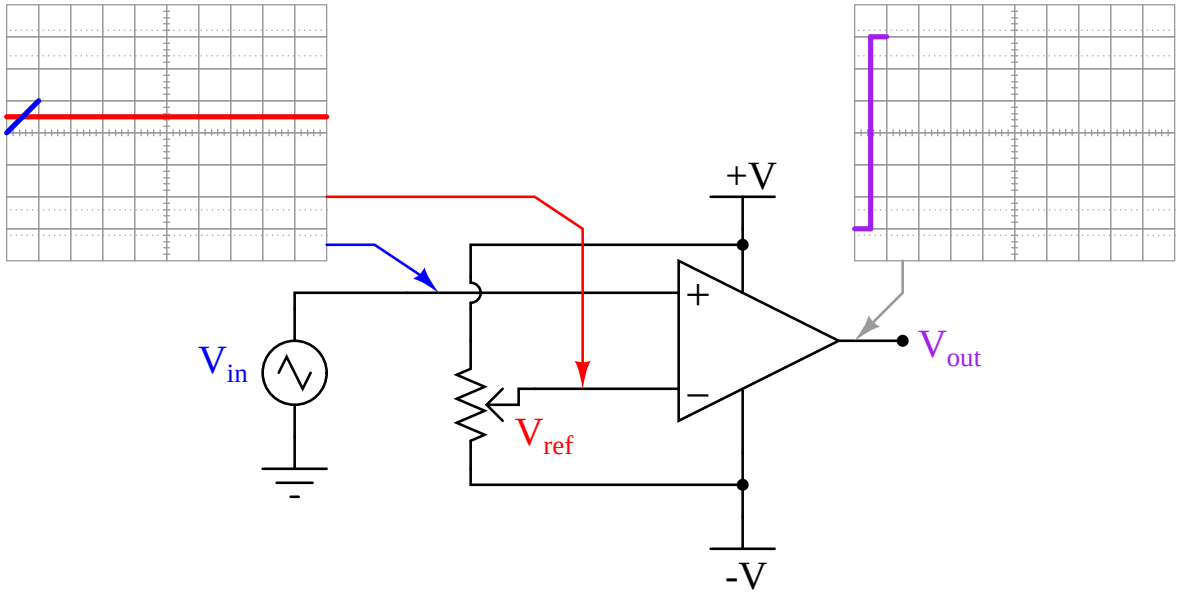

+V
V in
V ref
V out
-V

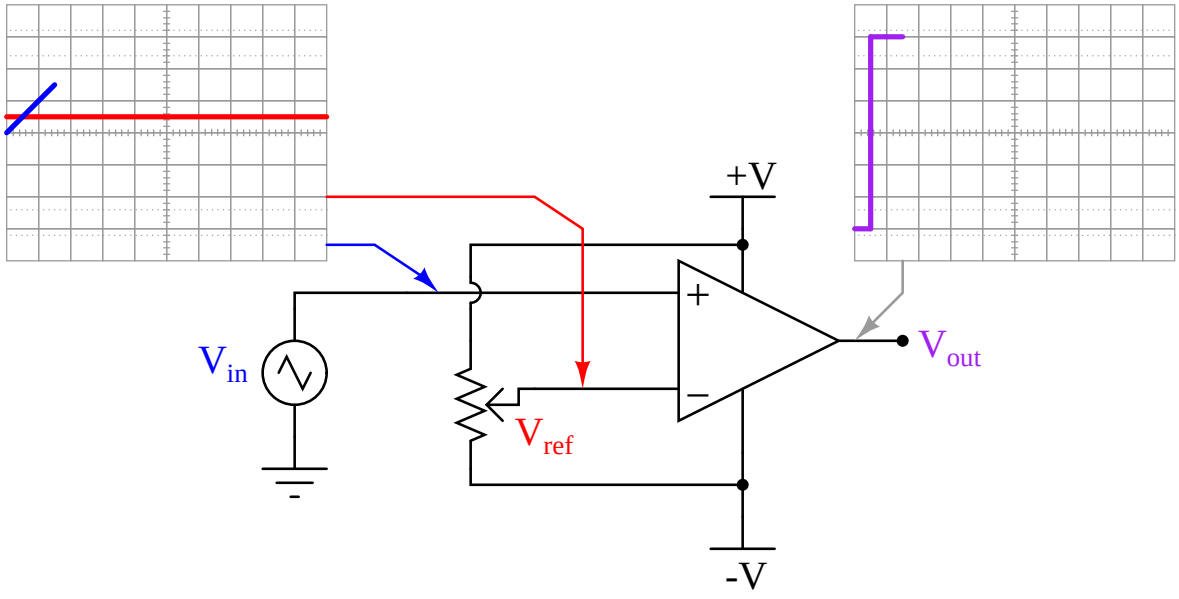

+V
V in
V ref
+
−
V out
-V

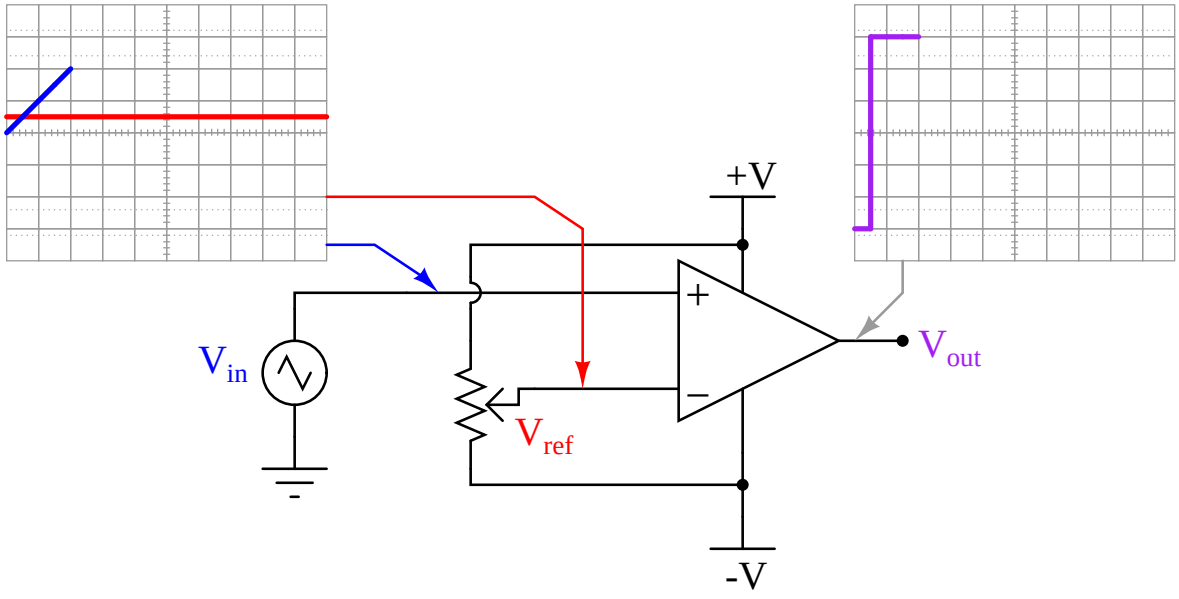

+V
V in
V ref
V out
-V

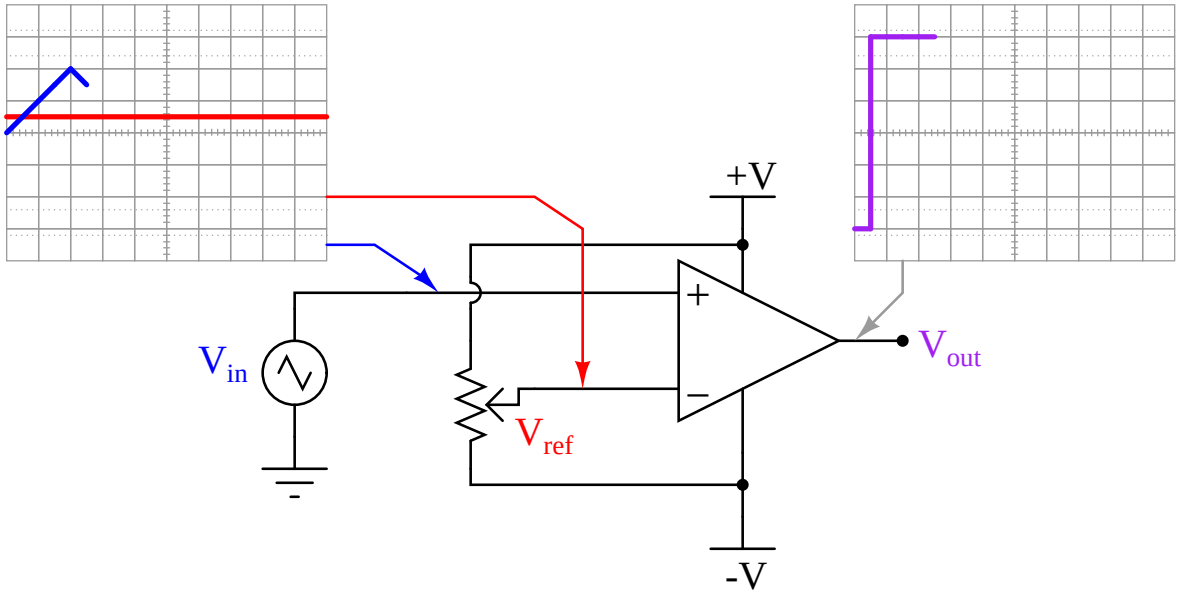

+V
V in
V ref
V out
-V

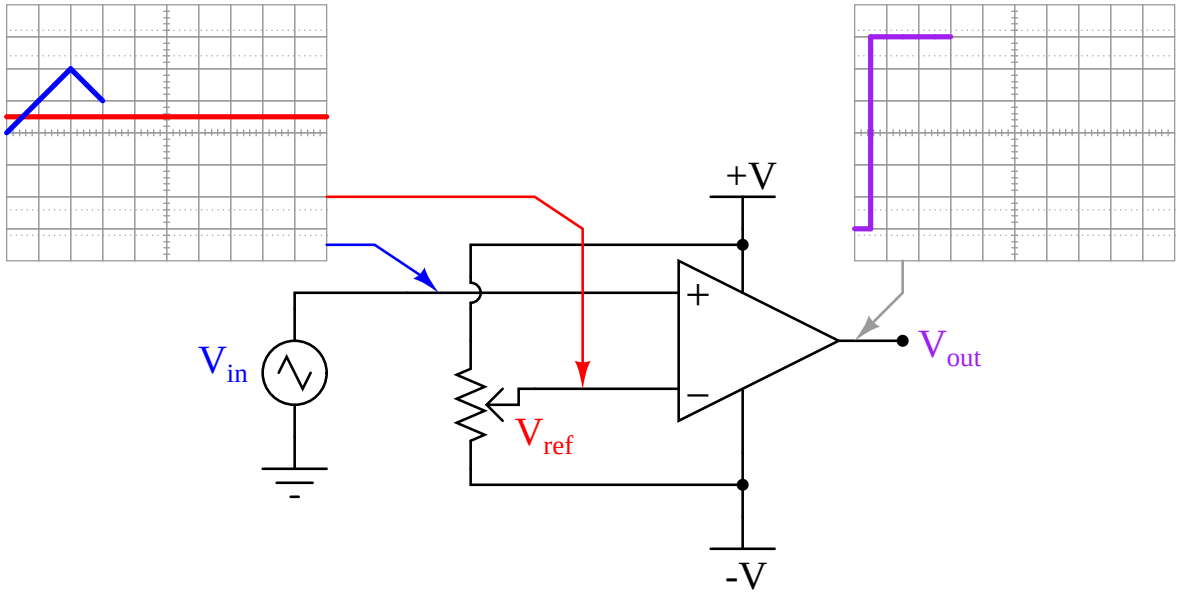

+V
V in
V ref
V out
-V

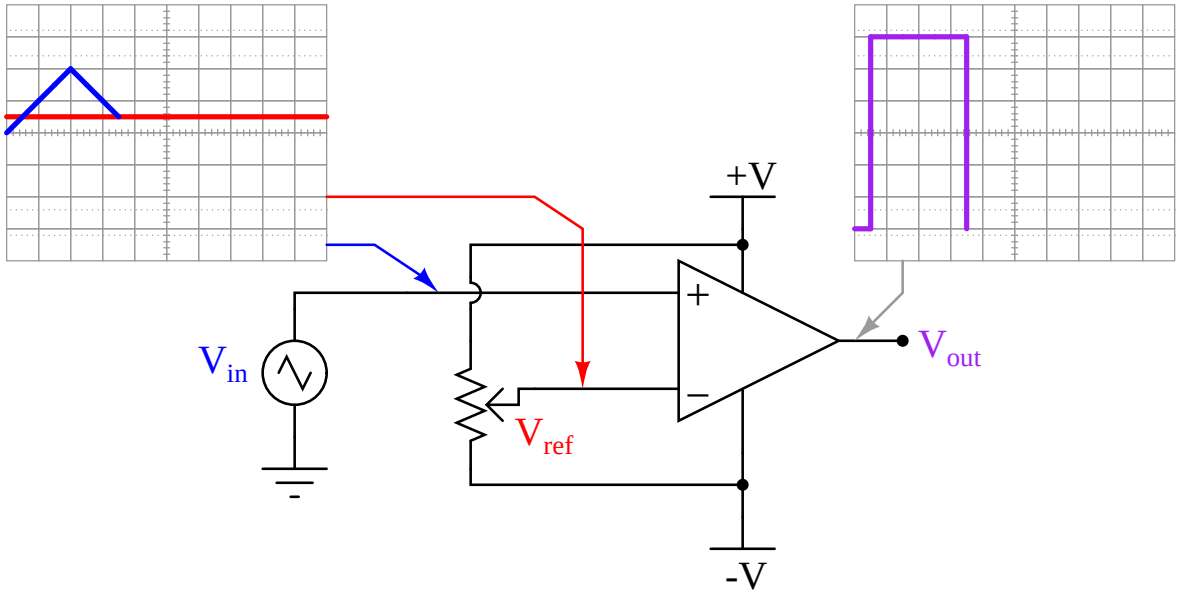

+V
V in
V ref
V out
-V

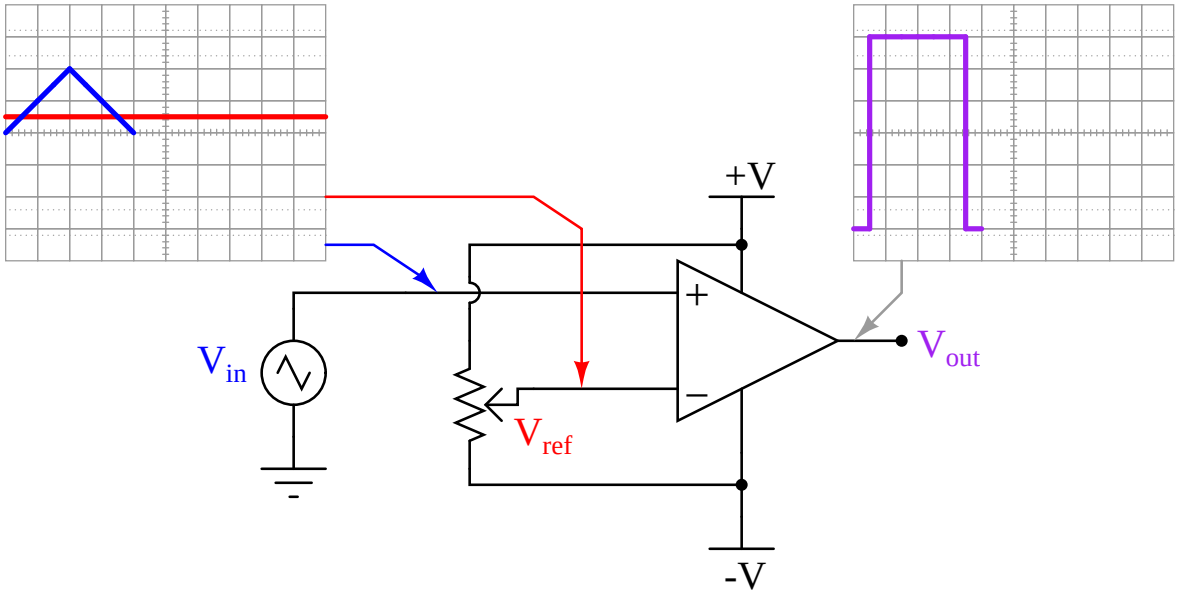

+V
V in
V ref
V out
-V

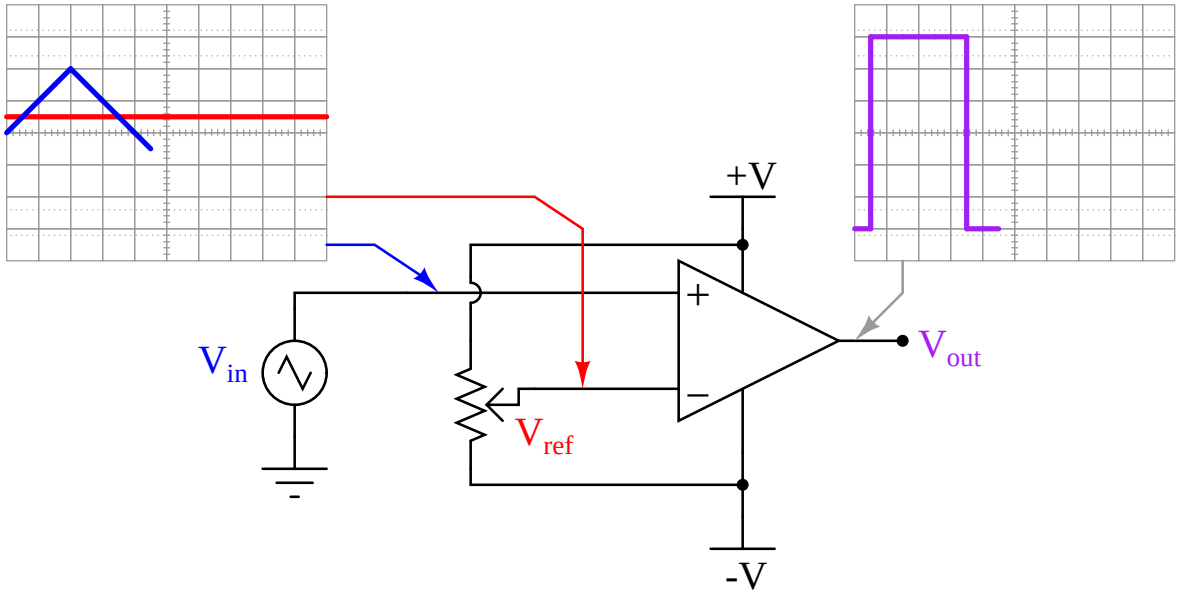

+V
V in
V ref
V out
-V

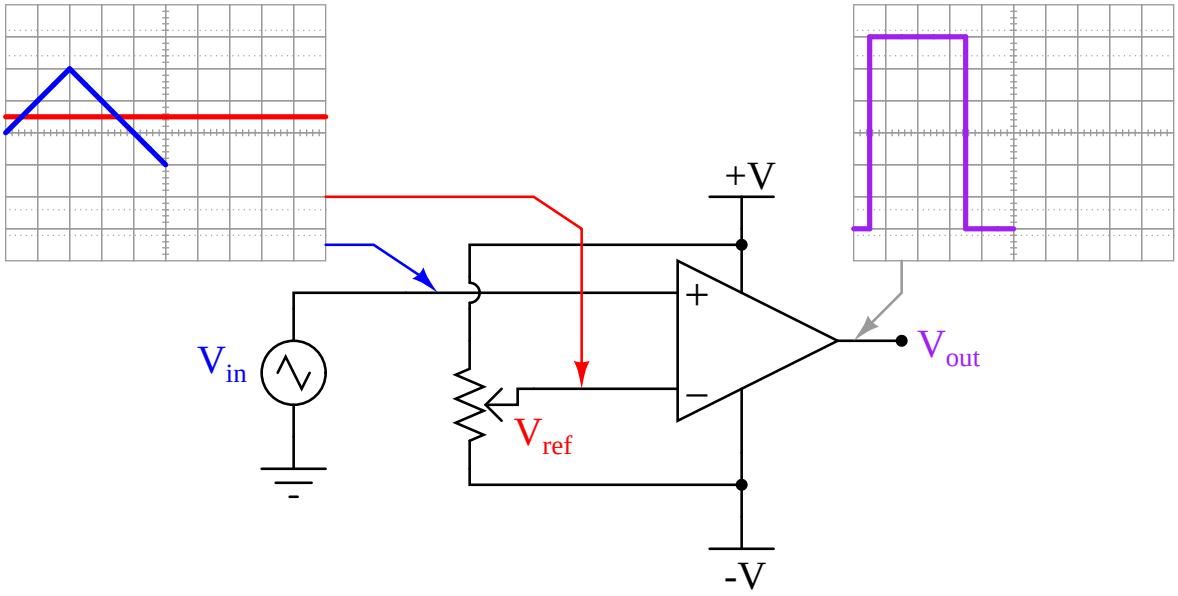

+V
V in
V ref
V out
-V

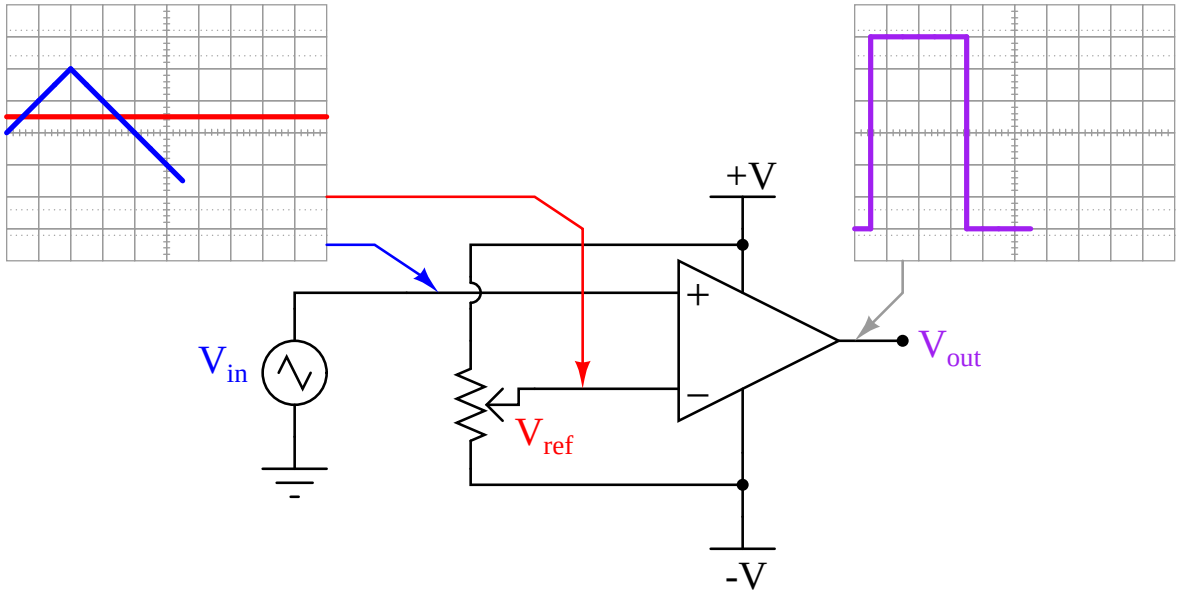

+V
V in
V ref
V out
-V

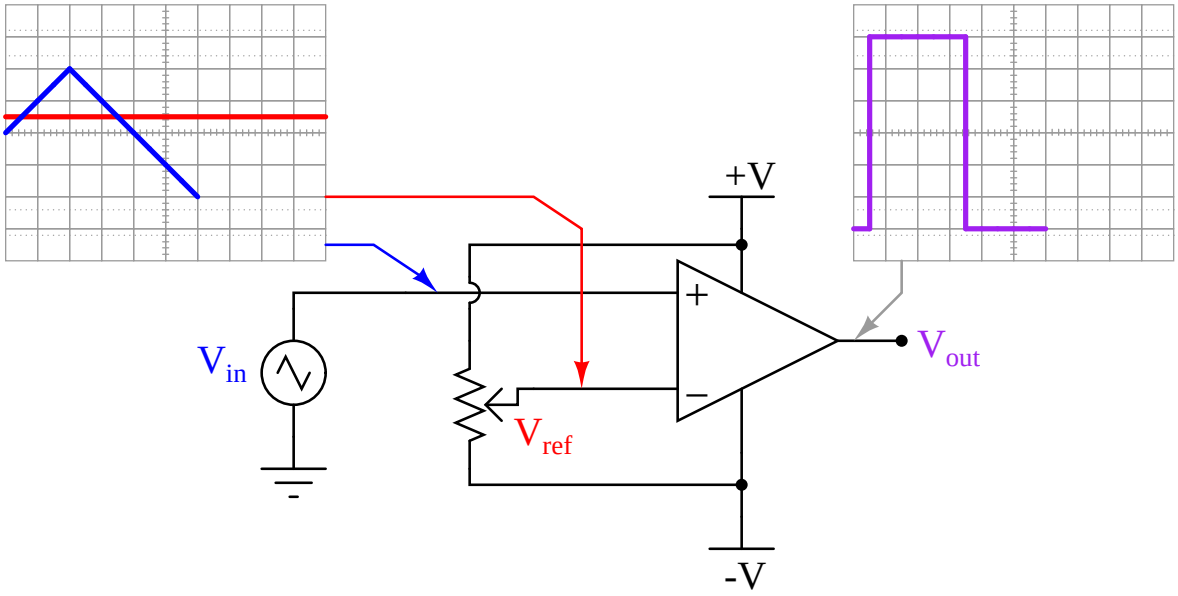

+V
V in
V ref
V out
-V

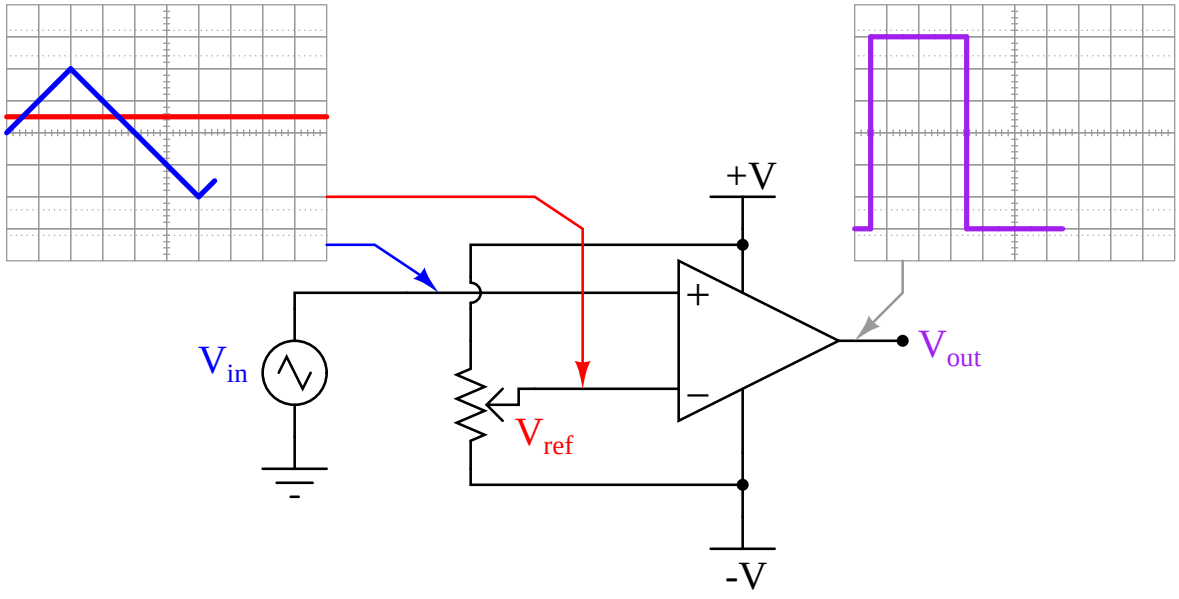

+V
V in
V ref
V out
-V

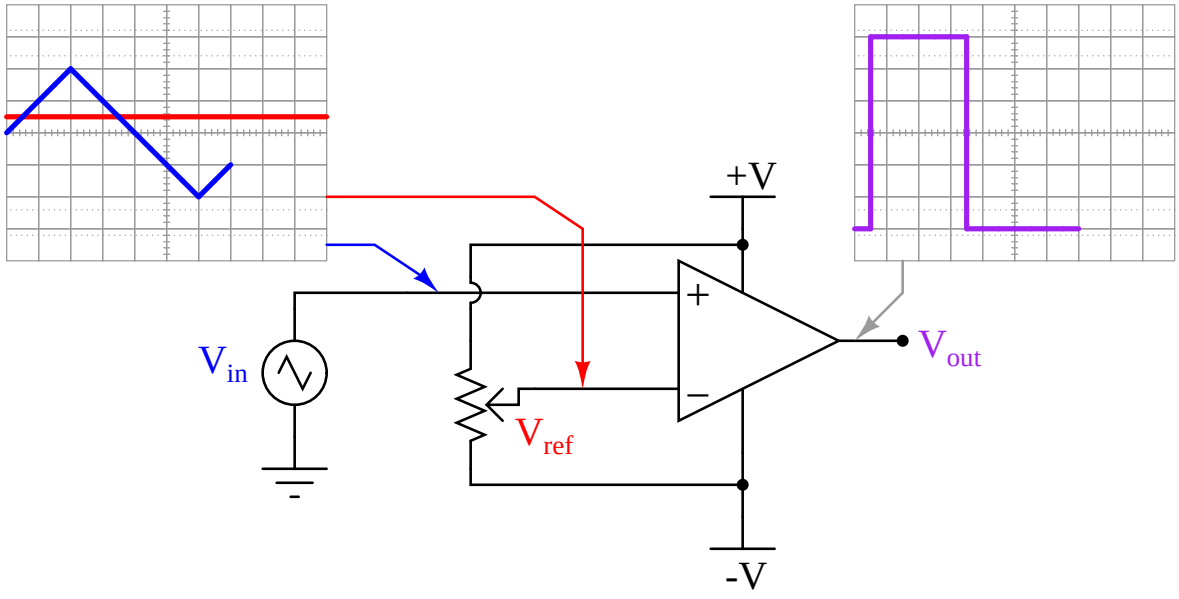

+V
+
−
V in
V ref
V out
-V

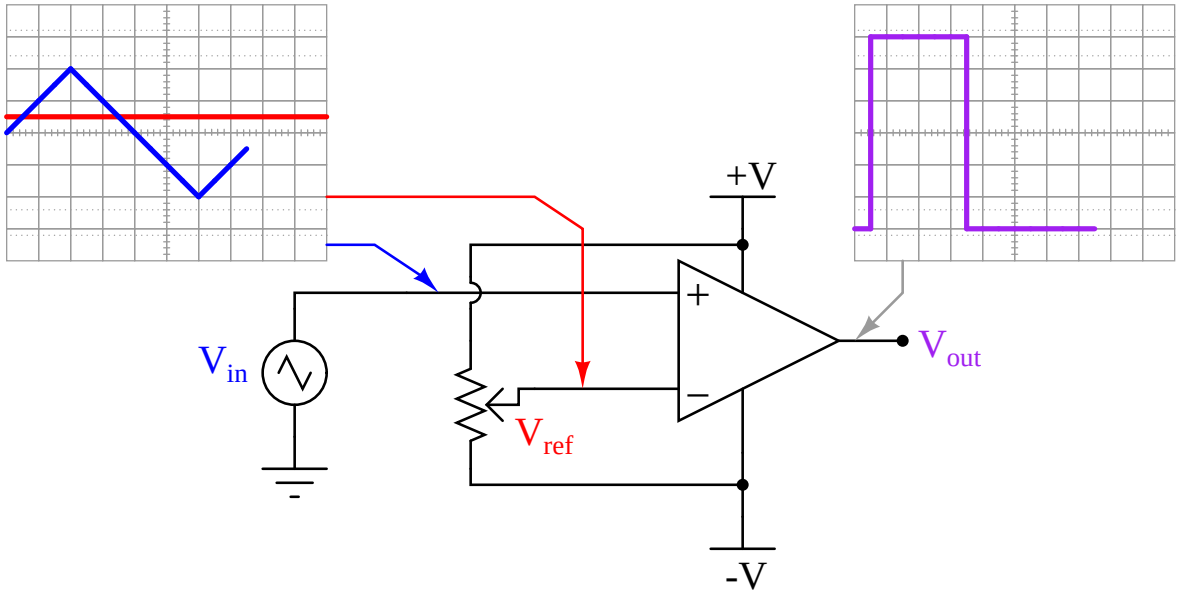

+V
V in
V ref
V out
-V

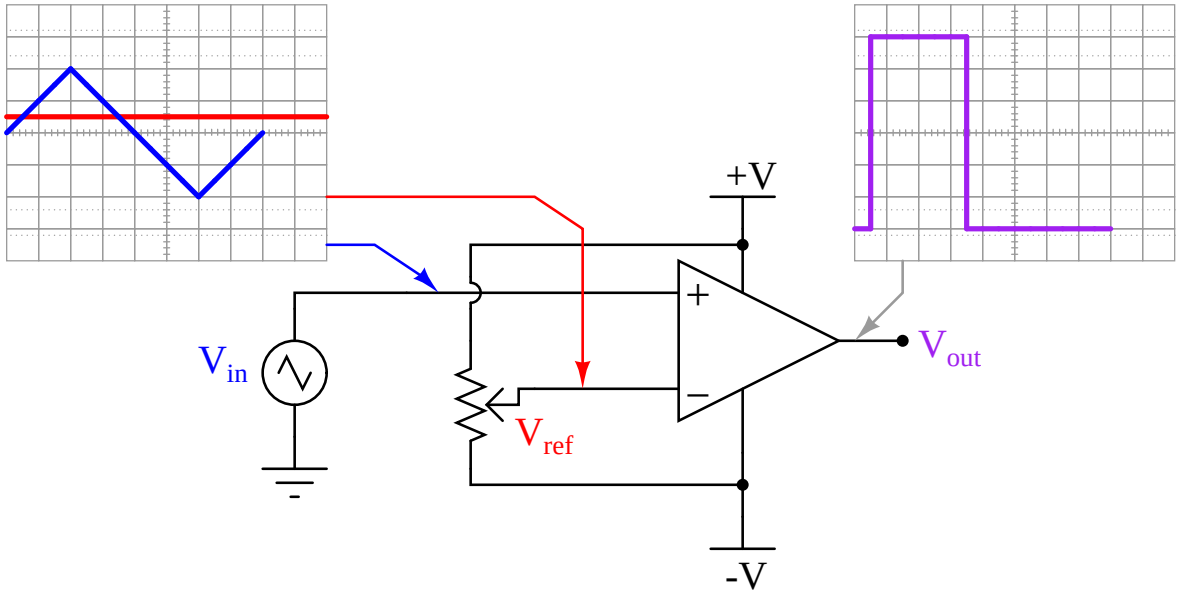

+V
V in
V ref
V out
-V

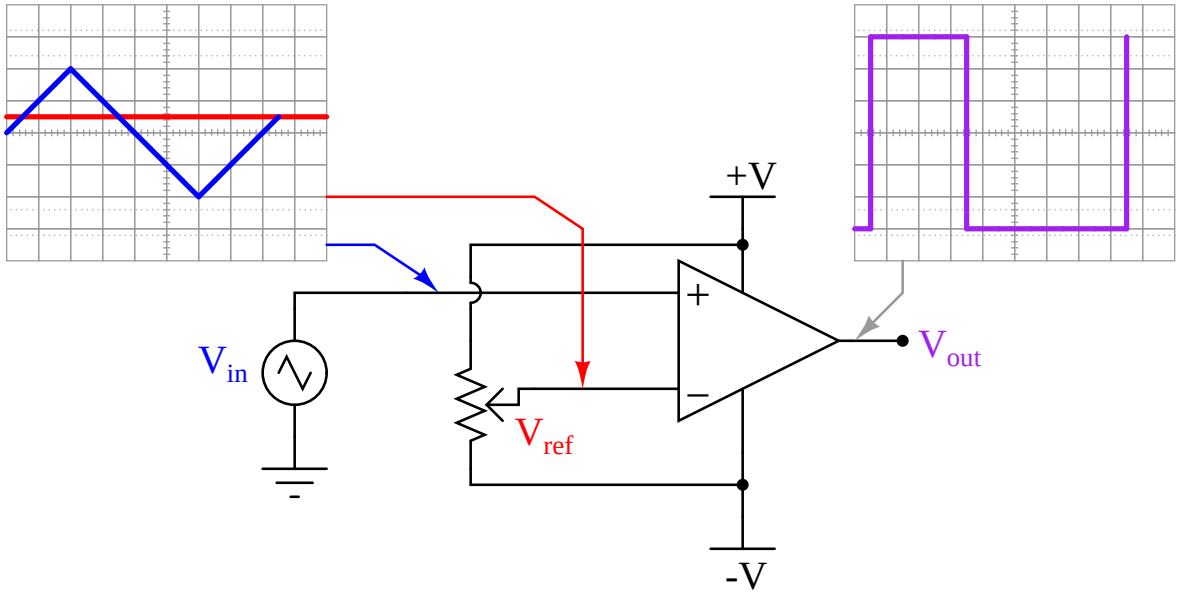

+V
V in
V ref
V out
-V

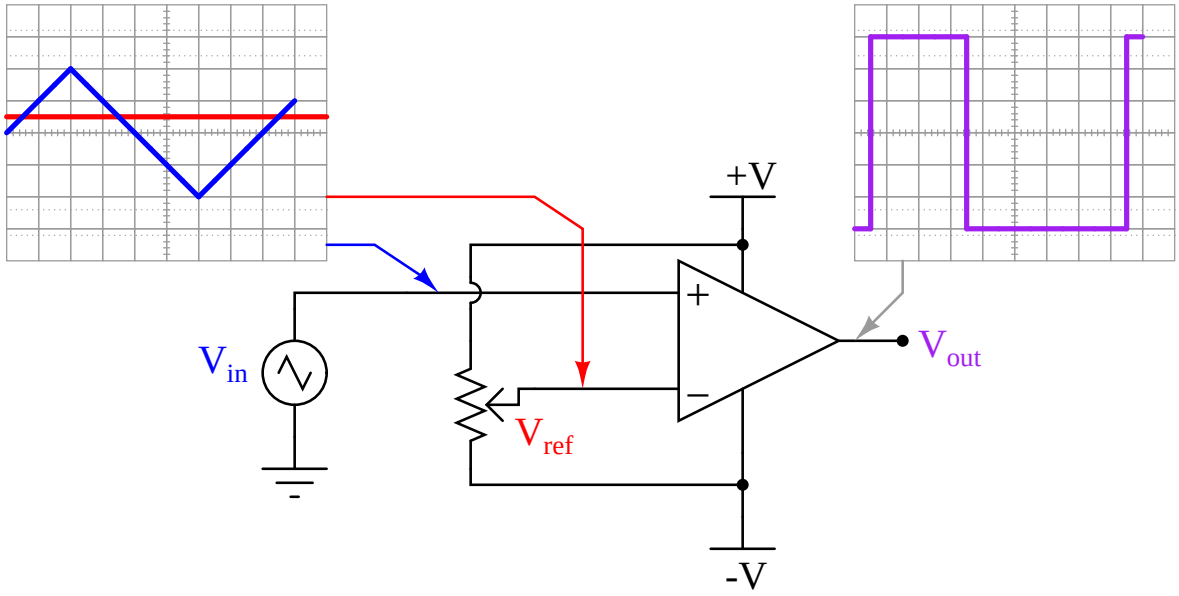

+V
V in
V ref
V out
-V

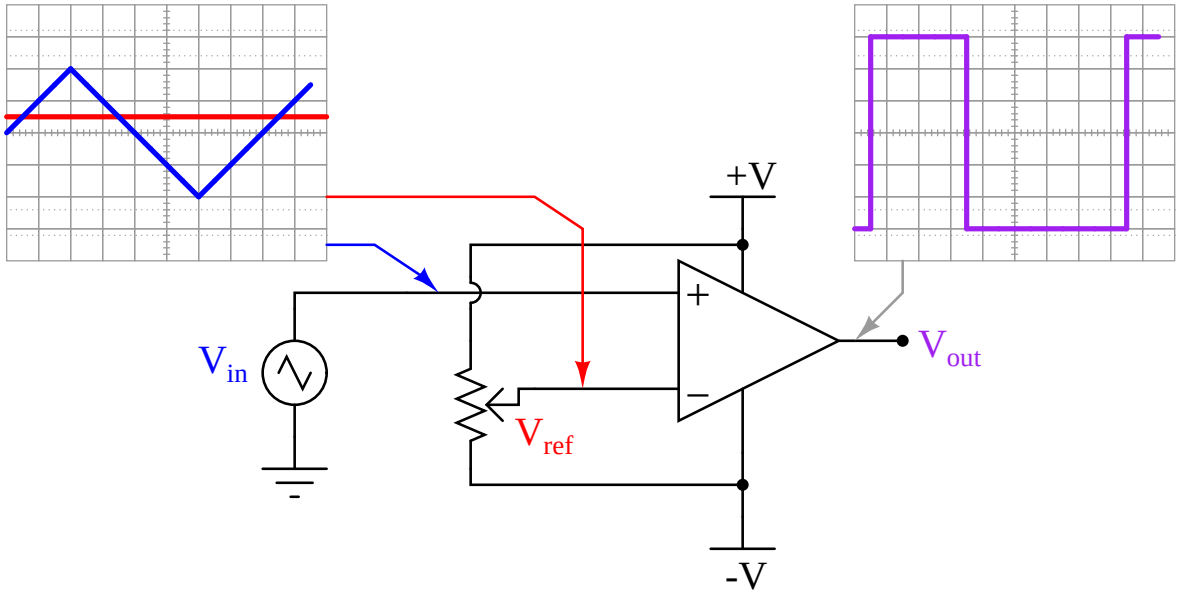

+V
V in
V ref
V out
-V

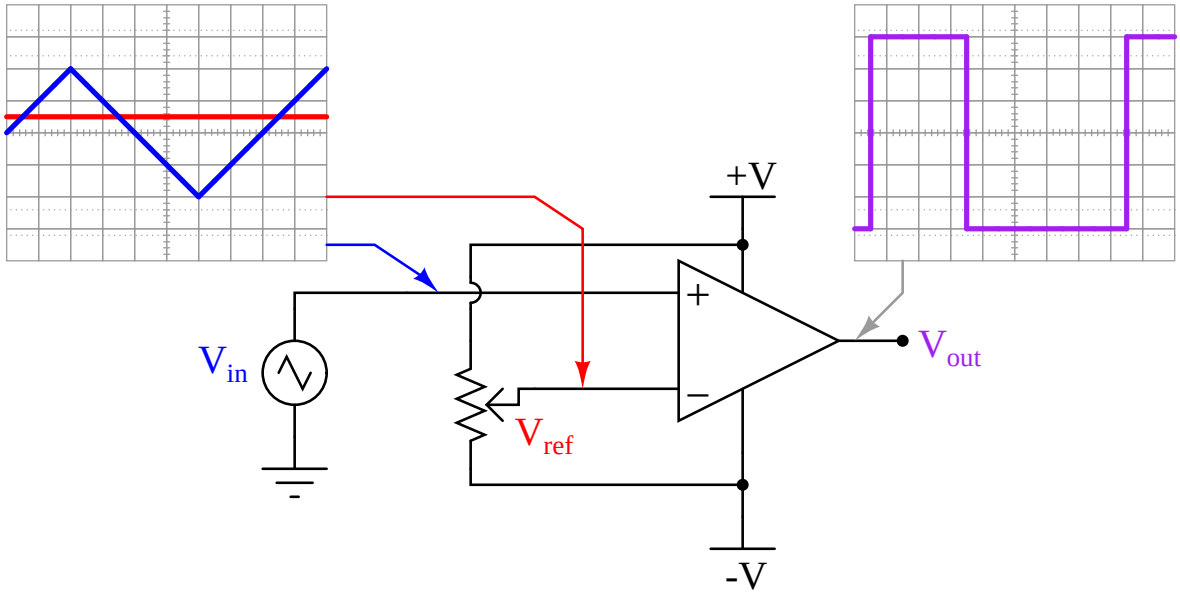

+V
V in
V ref
V out
-V


+V
V in
V ref
V out
-V


+V
V in
V ref
V out
-V


+V
V in
V ref
V out
-V


+V
V in
V ref
V out
-V


+V
V in
V ref
V out
-V

Chapter 6

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?

6.1 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.

6.1.1 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.

6.1.2 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

Normal state of a switch

Ohm's Law

Efficiency

Joule's Law

Properties of series circuits

Kinetic energy

Transistor

Duty cycle

Pulse-Width Modulation (PWM)

Comparator

Modulator

Demodulator

Filter

Harmonic frequency

Cutoff

6.1.3 Transistor states

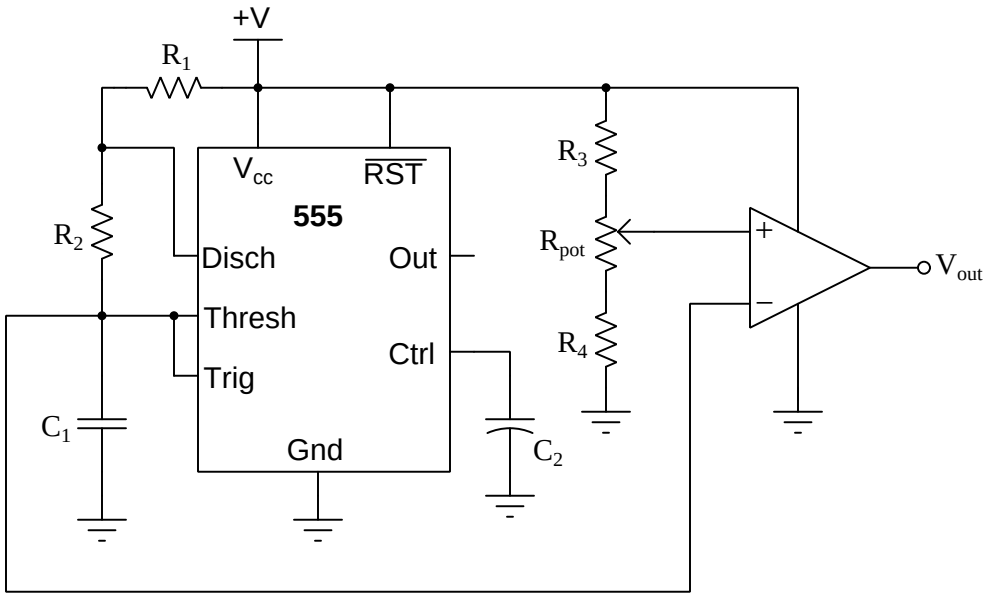
Explain why it is important for the final power transistor(s) in a PWM power control circuit to operate at full cutoff and full saturation, and not in the linear (active) mode in between those two extremes. What might happen if the power transistor(s) were to be less than cut-off or less than saturated when carrying load current.

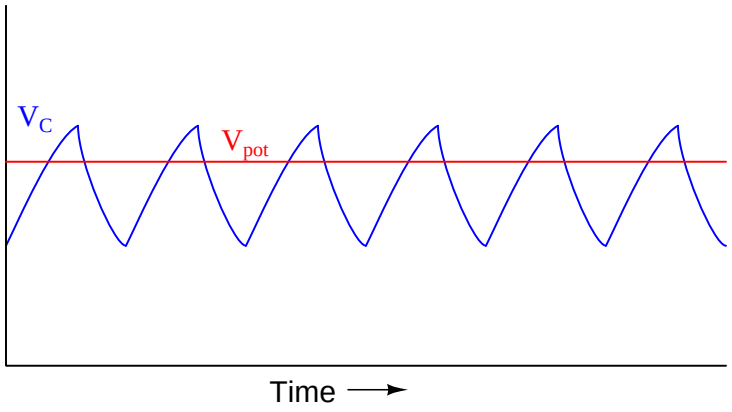
Challenges

- How much voltage does a BJT drop (V_{CE}) when fully turned on?
- How much voltage does an FET drop (V_{DS}) when fully turned on?

6.1.4 Comparator output waveform

Sketch the comparator's output signal in this PWM circuit, overlaying that voltage waveform on the 555 timer's capacitor voltage waveform:





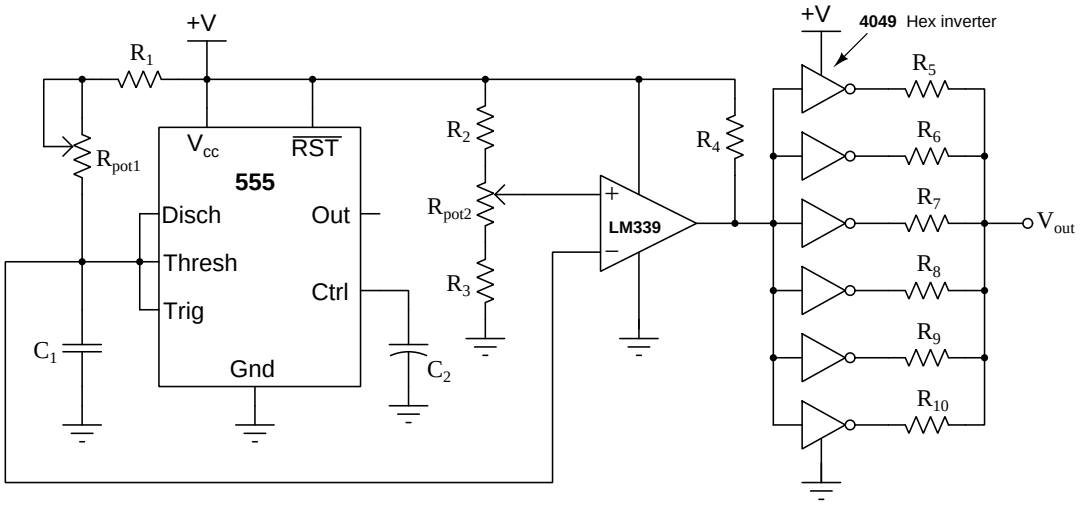
Challenges

- Estimate the duty cycle of the output PWM waveform.

- If the potentiometer wiper is moved in the upward direction, what will happen to the value of the PWM signal's duty cycle?

6.1.5 CMOS output drive

The following PWM circuit is equipped with a set of six CMOS inverter gates (part number 4049) ganged together in parallel:



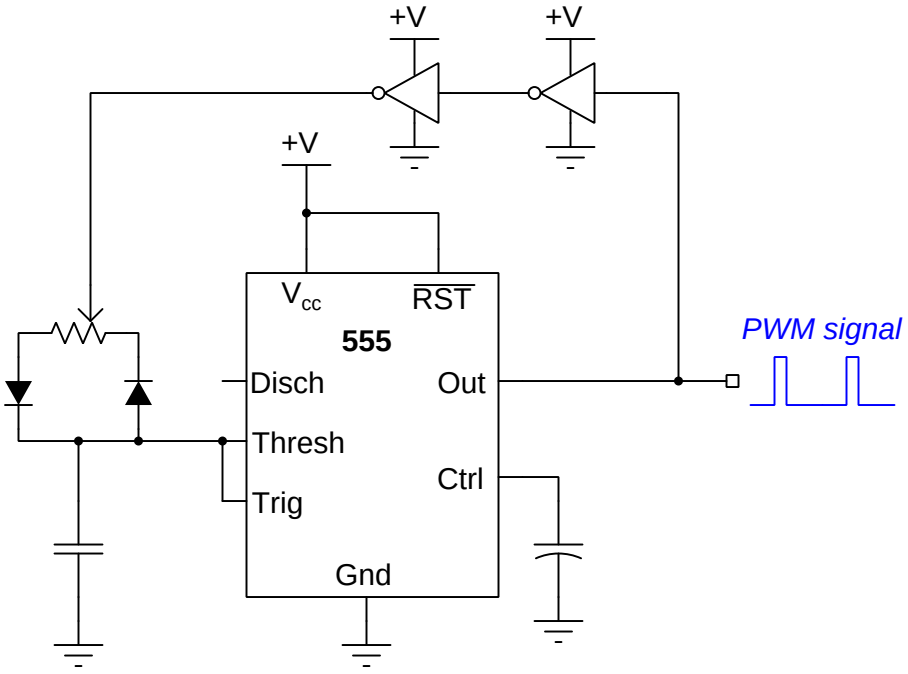
What do you suppose the purpose of this design is, using all those logic gates?

Challenges

- Why not just connect all the inverter gates directly in parallel, rather than have each inverter's output go through a resistor?
- What resistor value would you suggest using for R_5 through R_{10}?
- Adding inverter gates to the output of this circuit has an interesting effect on the duty cycle control. The potentiometer will now act backward from the way it was before (decreasing duty cycle when it formerly increased duty cycle, and visa-versa). Explain why.

6.1.6 PWM circuit using a 555 timer

Explain how the following 555 timer IC circuit works to produce an output waveform with a duty cycle adjustable using the potentiometer:



Which way should you move the potentiometer's wiper to increase duty cycle?

Challenges

- ???

6.2 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[4]" for gaining proficiency in the use of circuit simulation software, and then once that proficiency is gained you will never need to rely[5] on an answer key!

[4] 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.

[5] 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.

6.2.1 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.

6.2.2 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*[6] 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.

[6] 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[7] 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[8] 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[9] 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.

[7] 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.

[8] 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.

[9] 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	

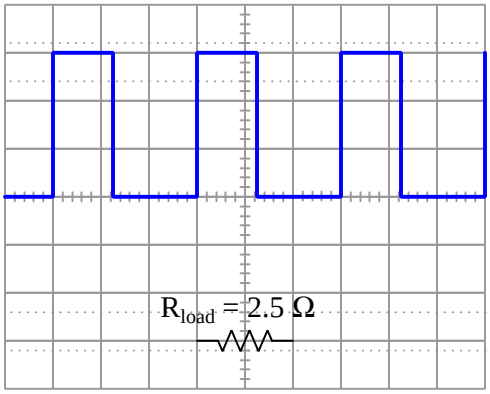
Note how the square-root term (y) is calculated in cell `C1`, and the denominator term (z) in cell `C2`. This makes the two final formulae (in cells `B1` and `B2`) simpler to interpret. The positioning of all these cells on the grid is completely arbitrary[10] – all that matters is that they properly 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.

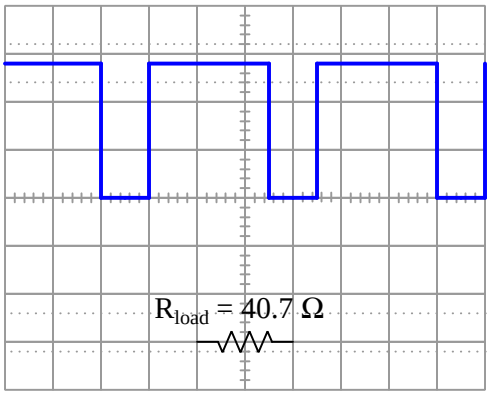
[10] My personal preference is to locate all the "given" data in the upper-left cells of the spreadsheet grid (each data point flanked by a sensible name in the cell to the left and units of measurement in the cell to the right as illustrated in the first distance/time spreadsheet example), sometimes 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.*

6.2.3 Duty cycle and power calculations

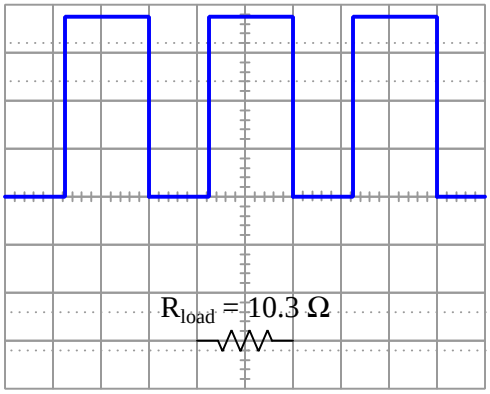
Calculate the duty cycles of the following waveforms, and also the average power dissipated by the load assuming the specified load resistance values:



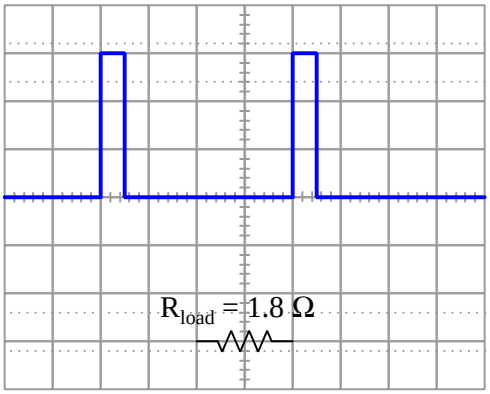
Vertical sensitivity = 1 Volt/div
Probe ratio = 1:1
Coupling = DC
Timebase = 0.5 ms/div



Vertical sensitivity = 5 Volts/div
Probe ratio = 10:1
Coupling = DC
Timebase = 0.2 ms/div



Vertical sensitivity = 1 Volt/div
Probe ratio = 10:1
Coupling = DC
Timebase = 20 µs/div



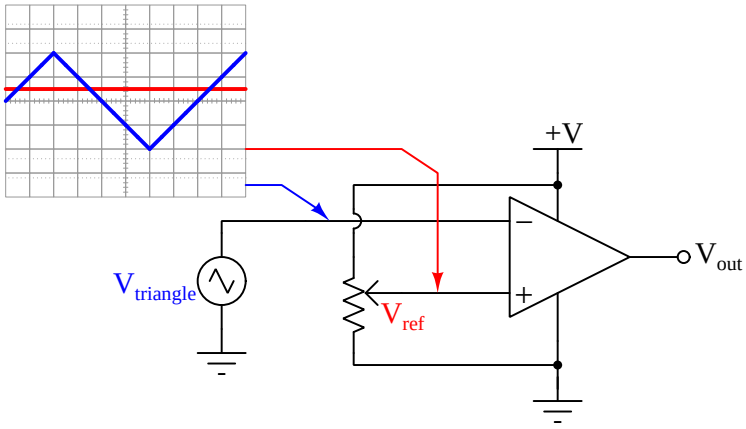
Vertical sensitivity = 20 Volts/div
Probe ratio = 1:1
Coupling = DC
Timebase = 0.1 ms/div

Challenges

- Which oscilloscope setup parameters (vertical sensitivity, probe ratio, coupling, and timebase) are necessary for performing these calculations? Which parameters are unnecessary, and why?

6.2.4 Duty cycle based on triangle waveform

Calculate the duty cycle of this comparator circuit's output signal based on the two input voltages shown by the oscilloscope:



Furthermore, calculate the duty cycle for each of these input conditions for the same comparator circuit:

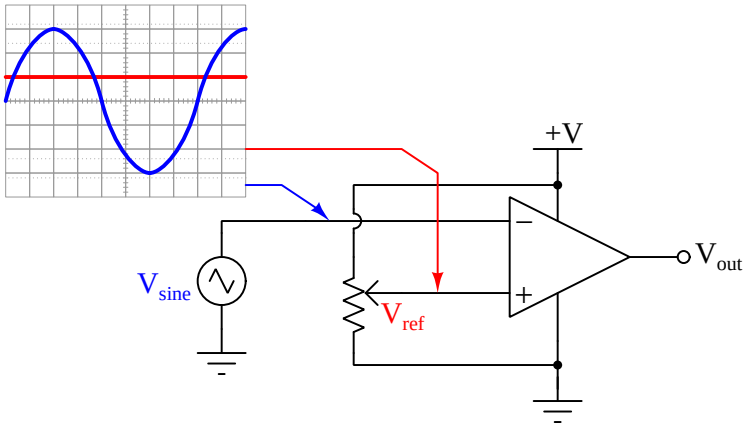
- $V_{triangle}$ peak values +3 Volts and −3 Volts ; V_{ref} = 0 Volts
- $V_{triangle}$ peak values +6 Volts and −6 Volts ; V_{ref} = −1.8 Volts
- $V_{triangle}$ peak values +10 Volts and −5 Volts ; V_{ref} = 3.9 Volts

Challenges

- Re-calculate what each of these duty cycle values if the input terminals on the comparator were swapped (i.e. V_{ref} to the inverting and V_{sine} to the non-inverting) given the same input signals.

6.2.5 Duty cycle based on sine waveform

Calculate the duty cycle of this comparator circuit's output signal based on the two input voltages shown by the oscilloscope:



Assuming the same 3 Volt peak amplitude for the sine wave, calculate the necessary V_{ref} to generate the following duty cycle values:

- Duty cycle = 75%
- Duty cycle = 50%
- Duty cycle = 10%

Challenges

- Re-calculate what each of these duty cycle values if the input terminals on the comparator were swapped (i.e. V_{ref} to the inverting and V_{sine} to the non-inverting) given the same input signals.

6.3 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.

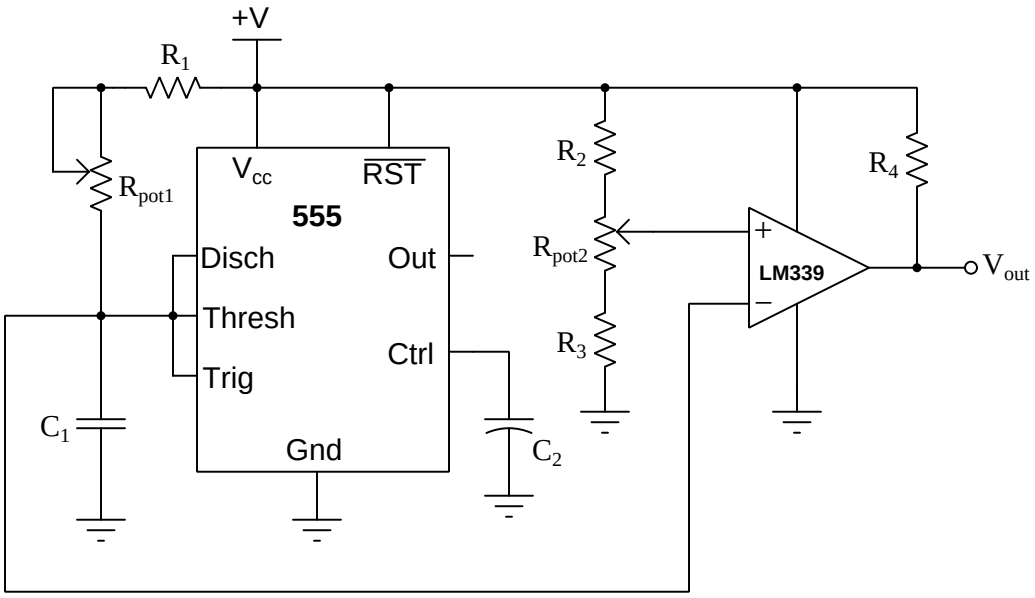
As always, your goal is to fully *explain* your analysis of each problem. Simply obtaining a correct answer is not good enough – you must also demonstrate sound reasoning in order to successfully complete the assignment. Your instructor's responsibility is to probe and challenge your understanding of 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.

6.3.1 Effects of faults in a PWM circuit

Predict the effects resulting from each of the following faults in this circuit:



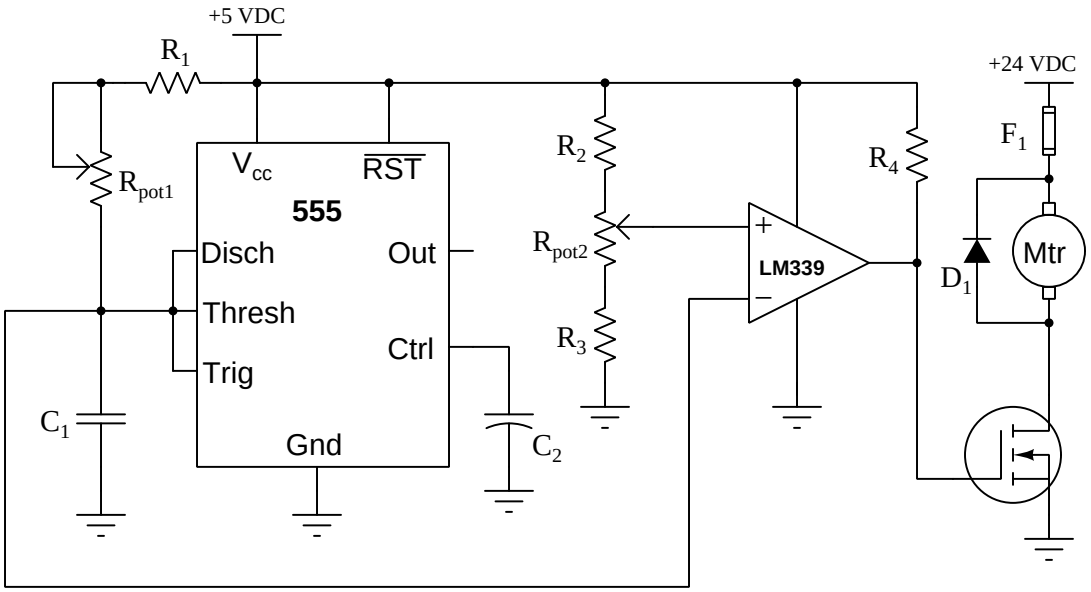
- Resistor R_1 failed open:
- Resistor R_1 failed shorted:
- Capacitor C_1 failed open:
- Capacitor C_1 failed shorted:
- Resistor R_2 failed open:
- Resistor R_2 failed shorted:
- Resistor R_3 failed open:
- Resistor R_3 failed shorted:
- Resistor R_4 failed open:
- Resistor R_4 failed shorted:

Challenges

- This circuit is not perfectly linear. That is to say, when R_{pot2} is adjusted, the wiper position will not *precisely* correspond to duty cycle. Explain why.
- What purpose do resistors R_2 and R_3 serve?

6.3.2 Identifying possible faults in a motor speed controller

Identify which of the listed faults are sufficient in themselves to prevent the motor from running regardless of either potentiometer's wiper position:



- Resistor R_1 failed open
- Resistor R_2 failed open
- Resistor R_3 failed open
- Resistor R_4 failed open
- Capacitor C_1 failed shorted
- Diode D_1 failed open
- Diode D_1 failed shorted
- Comparator output failed high (open)
- Comparator output failed low
- 5 VDC source dead
- 24 VDC source dead

Challenges

- What purpose is served by diode D_1, and must it be present any any type of PWM-controlled load?
- Add a logic gate to this circuit providing on/off control based on the logic level of an input line, where a "low" state on this line turns the motor off and a "high" state turns it on.

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

"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.

The following principles inform the instructional and assessment philosophies embodied in these learning modules:

- 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[1] 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[2] 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[3] 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

[1] 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.

[2] 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.

[3] 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 characteristic of these learning modules is a lack of step-by-step instructions in the Project and Experiment chapters. Unlike many modern workbooks and laboratory guides where step-by-step instructions are prescribed for each experiment, these modules take the approach that students must learn to closely read the tutorials and apply their own reasoning to identify the appropriate experimental steps. Sometimes these steps are plainly declared in the text, just not as a set of enumerated points. At other times certain steps are implied, an example being assumed competence in test equipment use where the student should not need to be told *again* how to use their multimeter because that was thoroughly explained in previous lessons. In some circumstances no steps are given at all, leaving the entire procedure up to the student.

This lack of prescription is not a flaw, but rather a feature. Close reading and clear thinking are foundational principles of this learning series, and in keeping with this philosophy all activities are designed to *require* those behaviors. Some students may find the lack of prescription frustrating, because it demands more from them than what their previous educational experiences required. This frustration should be interpreted as an unfamiliarity with autonomous thinking, a problem which must be corrected if the student is ever to become a self-directed learner and effective problem-solver. Ultimately, the need for students to read closely and think clearly is more important both in the near-term and far-term than any specific facet of the subject matter at hand. If a student takes longer than expected to complete a module because they are forced to outline, digest, and reason on their own, so be it. The future gains enjoyed by developing this mental discipline will be well worth the additional effort and delay.

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.

This methodology of assigning text-based modules to students for digestion and then using Socratic dialogue to assess progress and hone students' thinking was developed over a period of several years by the author with his Electronics and Instrumentation students at the two-year college level. While decidedly unconventional and sometimes even unsettling for students accustomed to a more passive lecture environment, this instructional philosophy has proven its ability to convey conceptual mastery, foster careful analysis, and enhance employability so much better than lecture that the author refuses to ever teach by lecture again.

Problems which often go undiagnosed in a lecture environment are laid bare in this "inverted" format where students must articulate and logically defend their reasoning. This, too, may be unsettling for students accustomed to lecture sessions where the instructor cannot tell for sure who comprehends and who does not, and this vulnerability necessitates sensitivity on the part of the "inverted" session instructor in order that students never feel discouraged by having their errors exposed. *Everyone* makes mistakes from time to time, and learning is a lifelong process! Part of the instructor's job is to build a culture of learning among the students where errors are not seen as shameful, but rather as opportunities for progress.

To this end, instructors managing courses based on these modules should adhere to the following principles:

- Student questions are always welcome and demand thorough, honest answers. The only type of question an instructor should refuse to answer is one the student should be able to easily answer on their own. Remember, *the fundamental goal of education is for each student to learn to think clearly and independently.* This requires hard work on the part of the student, which no instructor should ever circumvent. Anything done to bypass the student's responsibility to do that hard work ultimately limits that student's potential and thereby does real harm.

- It is not only permissible, but encouraged, to answer a student's question by asking questions in return, these follow-up questions designed to guide the student to reach a correct answer through their own reasoning.

- All student answers demand to be challenged by the instructor and/or by other students. This includes both correct and incorrect answers – the goal is to practice the articulation and defense of one's own reasoning.

- No reading assignment is deemed complete unless and until the student demonstrates their ability to accurately summarize the major points in their own terms. Recitation of the original text is unacceptable. This is why every module contains an "Outline and reflections" question as well as a "Foundational concepts" question in the Conceptual reasoning section, to prompt reflective reading.

- No assigned question is deemed answered unless and until the student demonstrates their ability to consistently and correctly apply the concepts to *variations* of that question. This is why module questions typically contain multiple "Challenges" suggesting different applications of the concept(s) as well as variations on the same theme(s). Instructors are encouraged to devise as many of their own "Challenges" as they are able, in order to have a multitude of ways ready to probe students' understanding.

- No assigned experiment or project is deemed complete unless and until the student demonstrates the task in action. If this cannot be done "live" before the instructor, video-recordings showing the demonstration are acceptable. All relevant safety precautions must be followed, all test equipment must be used correctly, and the student must be able to properly explain all results. The student must also successfully answer all Challenges presented by the instructor for that experiment or project.

Students learning from these modules would do well to abide by the following principles:

- No text should be considered fully and adequately read unless and until you can express every idea *in your own words, using your own examples.*
- You should always articulate your thoughts as you read the text, noting points of agreement, confusion, and epiphanies. Feel free to print the text on paper and then write your notes in the margins. Alternatively, keep a journal for your own reflections as you read. This is truly a helpful tool when digesting complicated concepts.
- Never take the easy path of highlighting or underlining important text. Instead, *summarize* and/or *comment* on the text using your own words. This actively engages your mind, allowing you to more clearly perceive points of confusion or misunderstanding on your own.
- A very helpful strategy when learning new concepts is to place yourself in the role of a teacher, if only as a mental exercise. Either explain what you have recently learned to someone else, or at least *imagine* yourself explaining what you have learned to someone else. The simple act of having to articulate new knowledge and skill forces you to take on a different perspective, and will help reveal weaknesses in your understanding.
- Perform each and every mathematical calculation and thought experiment shown in the text on your own, referring back to the text to see that your results agree. This may seem trivial and unnecessary, but it is critically important to ensuring you actually understand what is presented, especially when the concepts at hand are complicated and easy to misunderstand. Apply this same strategy to become proficient in the use of *circuit simulation software*, checking to see if your simulated results agree with the results shown in the text.
- 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[4] effort, and never give up!

Students interested in using these modules for self-study will also find them beneficial, although the onus of responsibility for thoroughly reading and answering questions will of course lie with that individual alone. If a qualified instructor is not available to challenge students, a workable alternative is for students to form study groups where they challenge[5] one another.

To high standards of education,

Tony R. Kuphaldt

[4] 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.

[5] Avoid the temptation to simply share answers with study partners, as this is really counter-productive to learning. Always bear in mind that the answer to any question is far less important in the long run than the method(s) used to obtain that answer. The goal of education is to empower one's life through the improvement of clear and independent thought, literacy, expression, and various practical skills.

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

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.

31 October 2024 – minor edits to the Tutorial, including a typographical error correction where I used the word "to" instead of "two". Also edited images 4002 through 4005 to contain the load resistance values in the graphic rather than in the body of the LaTeX text. Also made two typographical error corrections courtesy of Daniel Renshaw.

25 October 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 image_6594 and image_6595 to show the three output PWM signals offset from each other for better clarity of their duty cycles.

7 October 2024 – added a new Conceptual Reasoning question on an alternative way to generate PWM signals using a 555 timer.

2 November 2023 – minor page-formatting edits to the Tutorial, and added new questions to the Introduction chapter.

29 June 2023 – added more explanatory text on comparator function.

21 April 2023 – changed title of this module from "Pulse Width Modulation Power Control" to simply "Pulse Width Modulation".

19-20 April 2023 – began writing a new section discussing mathematical relationships between duty cycle values for PWM signals passing through logic gates. Also added more commentary on generating multiple PWM signals using one oscillator, in order to have those PWM signals exhibit the exact same frequency.

5 April 2023 – wrote a Derivations and Technical References section showing why average PWM

power is the duty cycle multiplied by the peak power.

28 November 2022 – placed questions at the top of the itemized list in the Introduction chapter prompting students to devise experiments related to the tutorial content.

20 July 2022 – corrected an error in image_2841 by swapping polarities of the 2 Volt, 5 Volt, and 7 Volt sources in the lowest-left example.

23 January 2022 – added a new section to the Tutorial discussing the use of logic gates to control PWM signals.

19 January 2022 – edited one diagram in the Tutorial to show both high-side and low-side transistor switching configurations for a PWM power control system. Also added a new Diagnostic Reasoning question showing a full 555-based motor speed control circuit.

30 November 2021 – expanded the "555-based PWM signal generator" Case Tutorial section to include a power transistor driving a load, and also divided the Tutorial chapter into sections.

4 November 2021 – added a Case Tutorial chapter showing a triangle/square/PWM signal generator design. Also made some minor edits to certain illustrations.

27 August 2021 – added a new Case Tutorial chapter showing a 555 timer and LM339 comparator used as a PWM signal generator.

31 May 2021 – modified image_2841 to have more complex examples.

9 May 2021 – commented out or deleted empty chapters.

9 April 2021 – added a Case Tutorial section showing PWM signals generated by a microcontroller then filtered to produce analog waveshapes.

31 March 2021 – minor additions to the Tutorial, including a pulse diagram showing the definition of duty cycle.

13 January 2021 – added more explanatory text on comparators, in case this is the reader's first exposure to these ICs.

7 November 2020 – added some Quantitative questions.

4 November 2020 – added content to Tutorial explaining the relationship between duty cycle and average power. Also added more comments on the instructor notes for "Effects of faults in a PWM circuit" question. Also added a Case Tutorial chapter showing comparator responses to different input conditions, as review.

8 October 2020 – significantly edited the Introduction chapter to make it more suitable as a pre-study guide and to provide cues useful to instructors leading "inverted" teaching sessions.

9 July 2020 – added a small amount of content to the Tutorial, showing the pushbutton switch

being replaced by a transistor.

1 July 2020 – added content to the Tutorial as well as some questions.

30 June 2020 – document first created.

Index