## **ANALOG AND DIGITAL**

## **Overview**

We live in an analog world. There are an infinite amount of colors to paint an object (even if the difference is indiscernible to our eye), there are an infinite number of tones we can hear, and there are an infinite number of smells we can smell. The common theme among all of these analog signals is their **infinite** possibilities.

Digital signals and objects deal in the realm of the **discrete** or **finite**, meaning there is a limited set of values they can be. That could mean just two total possible values, 255, 4,294,967,296, or anything as long as it’s not ∞ (infinity).

[](https://cdn.sparkfun.com/assets/4/a/e/6/f/51c9c988ce395fab0e000000.png)

Real-world objects can display data, gather inputs by either analog or digital means. (From left to right): Clocks, multimeters, and joysticks can all take either form (analog above, digital below).

Working with electronics means dealing with both analog and digital signals, inputs and outputs. Our electronics projects have to interact with the real, analog world in some way, but most of our microprocessors, computers, and logic units are purely digital components. These two types of signals are like different electronic languages; some electronics components are bi-lingual, others can only understand and speak one of the two.

## **Analog Signals**

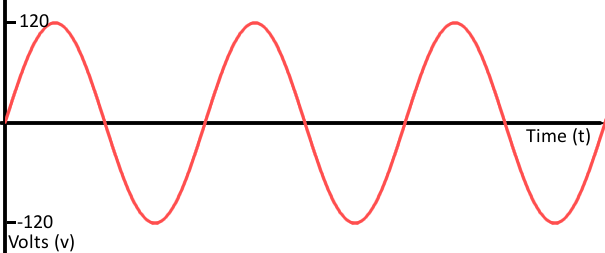
#### **Define: Signals**

Before going too much further, we should talk a bit about what a signal actually is, electronic signals specifically (as opposed to traffic signals, albums by the ultimate power trio, or a general means for communication). The signals we’re talking about are **time-varying** “quantities” which convey some sort of information. In electrical engineering the quantity that’s time-varying is usually **voltage** (if not that, then usually current). So when we talk about signals, just think of them as a voltage that’s changing over time.

Signals are passed between devices in order to send and receive information, which might be video, audio, or some sort of encoded data. Usually the signals are transmitted through wires, but they could also pass through the air via radio frequency (RF) waves. Audio signals, for example might be transferred between your computer’s audio card and speakers, while data signals might be passed through the air between a tablet and a Wi-Fi router.

### **Analog Signal Graphs**

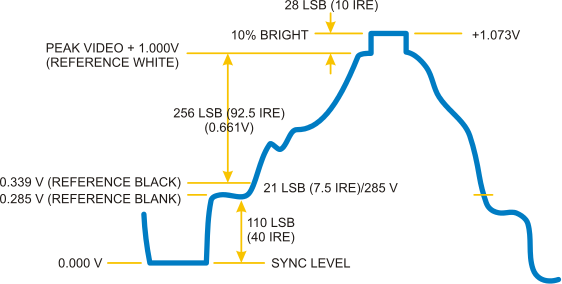
Because a signal varies over time, it’s helpful to plot it on a graph where time is plotted on the horizontal, x-axis, and voltage on the vertical, y-axis. Looking at a graph of a signal is usually the easiest way to identify if it’s analog or digital; a time-versus-voltage graph of an analog signal should be **smooth** and **continuous**.

[](https://cdn.sparkfun.com/assets/3/7/6/6/0/51c48875ce395f745a000000.png)

While these signals may be limited to a **range** of maximum and minimum values, there are still an infinite number of possible values within that range. For example, the analog voltage coming out of your wall socket might be clamped between -120V and +120V, but, as you increase the resolution more and more, you discover an infinite number of values that the signal can actually be (like 64.4V, 64.42V, 64.424V, and infinite, increasingly precise values).

### **Example Analog Signals**

Video and audio transmissions are often transferred or recorded using analog signals. The composite video coming out of an old RCA jack, for example, is a coded analog signal usually ranging between 0 and 1.073V. Tiny changes in the signal have a huge effect on the color or location of the video.

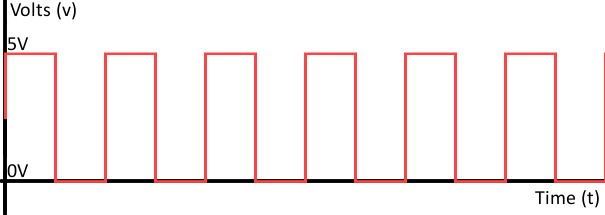
[](https://cdn.sparkfun.com/assets/5/9/3/3/8/51c48c2ece395fd35a000000.png)

An analog signal representing one line of composite video data.

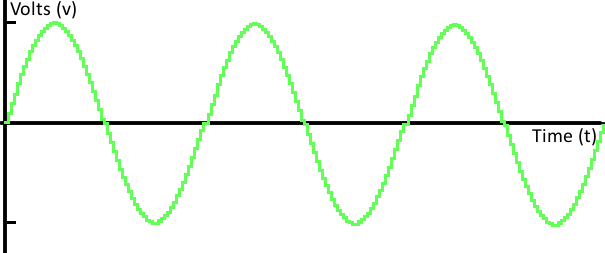
Pure audio signals are also analog. The signal that comes out of a microphone is full of analog frequencies and harmonics, which combine to make beautiful music.

## **Digital Signals**

Digital signals must have a finite set of possible values. The number of values in the set can be anywhere between two and a-very-large-number-that’s-not-infinity. Most commonly digital signals will be one of **two values** – like either 0V or 5V. Timing graphs of these signals look like **square waves**.

[](https://cdn.sparkfun.com/assets/c/8/5/b/e/51c495ebce395f1b5a000000.png)

Or a digital signal might be a discrete representation of an analog waveform. Viewed from afar, the wave function below may seem smooth and analog, but when you look closely there are tiny discrete **steps** as the signal tries to approximate values:

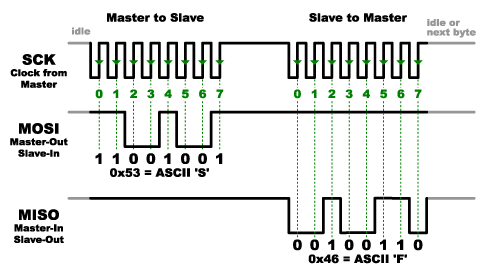
[](https://cdn.sparkfun.com/assets/0/2/8/4/6/51c85fbece395fbc03000000.png)

That’s the big difference between analog and digital waves. Analog waves are smooth and continuous, digital waves are stepping, square, and discrete.

### **Example Digital Signals**

Not all audio and video signals are analog. Standardized signals like HDMI for video (and audio) and MIDI, I2S, or AC’97 for audio are all digitally transmitted.

Most communication between integrated circuits is digital. Interfaces like serial, I2C, and SPI all transmit data via a coded sequence of square waves.

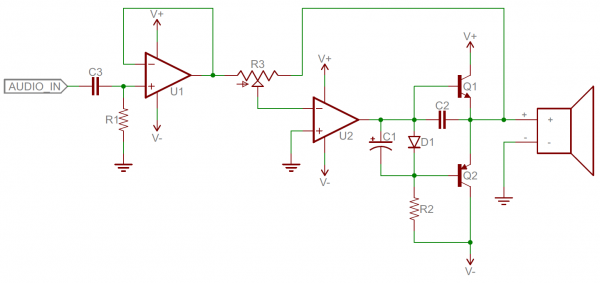
[](https://cdn.sparkfun.com/assets/e/b/3/b/4/51c4a21ece395f805a000000.png)

Serial peripheral interface (SPI) uses many digital signals to transmit data between devices.

## **Analog and Digital Circuits**

### **Analog Electronics**

Most of the fundamental electronic components – resistors, capacitors, inductors, diodes, transistors, and operational amplifiers – are all inherently analog. Circuits built with a combination of solely these components are usually analog.

[](https://cdn.sparkfun.com/assets/d/9/e/1/c/51c8bb1ece395fef60000001.png)

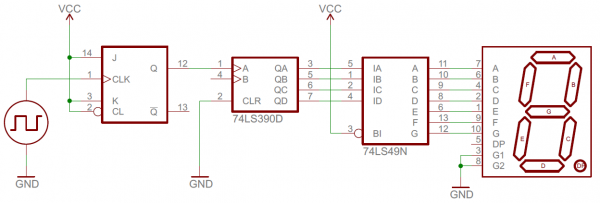
Analog circuits are usually complex combinations of op amps, resistors, caps, and other foundational electronic components. This is an example of a class B analog audio amplifier.

Analog circuits can be very elegant designs with many components, or they can be very simple, like two resistors combining to make a voltage divider. In general, though, analog circuits are much **more difficult to design** than those which accomplish the same task digitally. It takes a special kind of analog circuit wizard to design an analog radio receiver, or an analog battery charger; digital components exist to make those designs much simpler.

Analog circuits are usually much more **susceptible to noise** (small, undesired variations in voltage). Small changes in the voltage level of an analog signal may produce significant errors when being processed.

### **Digital Electronics**

Digital circuits operate using digital, discrete signals. These circuits are usually made of a combination of transistors and logic gates and, at higher levels, microcontrollers or other computing chips. Most processors, whether they’re big beefy processors in your computer, or tiny little microcontrollers, operate in the digital realm.

[](https://cdn.sparkfun.com/assets/6/0/4/8/1/51c9c1f8ce395fda22000000.png)

Digital circuits make use of components like logic gates, or more complicated digital ICs (usually represented by rectangles with labeled pins extending from them).

Digital circuits usually use a binary scheme for digital signaling. These systems assign two different voltages as two different logic levels – a high voltage (usually 5V, 3.3V, or 1.8V) represents one value and a low voltage (usually 0V) represents the other.

Although digital circuits are generally easier to design, they do tend to be a bit **more expensive** than an equally tasked analog circuit.

### **Analog and Digital Combined**

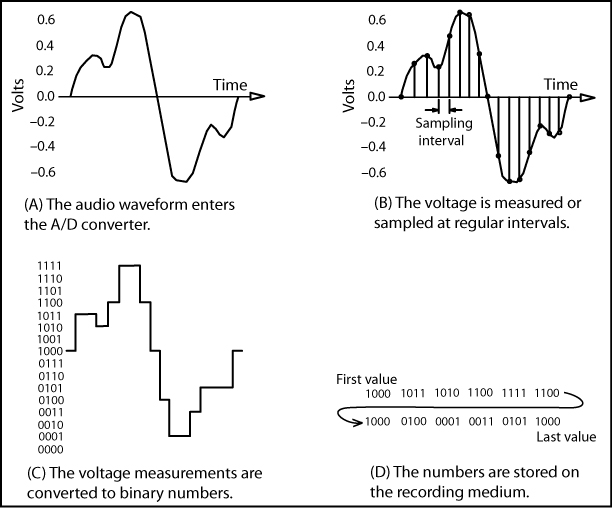
It’s not rare to see a mixture of analog and digital components in a circuit. Although microcontrollers are usually digital beasts, they often have internal circuitry which enables them to interface with analog circuitry (analog-to-digital converters, pulse-width modulation, and digital-to-analog converters. An analog-to-digital converter (ADC) allows a microcontroller to connect to an analog sensor (like photocells or temperature sensors), to read in an analog voltage. The less common digital-to-analog converter allows a microcontroller to produce analog voltages, which is handy when it needs to make sound.

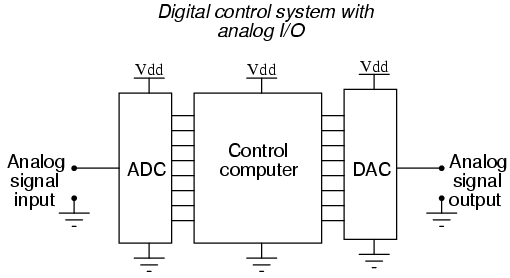
**Analog to Digital Conversion in 3 steps:**

-Sampling: Conversion from continuous-time, continuous valued signal to discrete-time, continuous-valued signal

-Quantization: Conversion from discrete-time, continuous-valued signal to discrete-time, discrete-valued signal

-Encoding: Conversion from a discrete-time, discrete-valued signal to an efficient digital data format (represented as bits)





## **What is Pulse-width Modulation?**

Pulse Width Modulation (PWM) is a fancy term for describing a type of digital signal. Pulse width modulation is used in a variety of applications including sophisticated control circuitry. A common way we use them is to **control dimming of RGB LEDs or to control the direction of a servo motor.** We can accomplish a range of results in both applications because pulse width modulation **allows us to vary how much time the signal is high in an analog fashion.** While the signal can only be high (usually 5V) or low (ground) at any time, we can change the proportion of time the signal is high compared to when it is low over a consistent time interval.

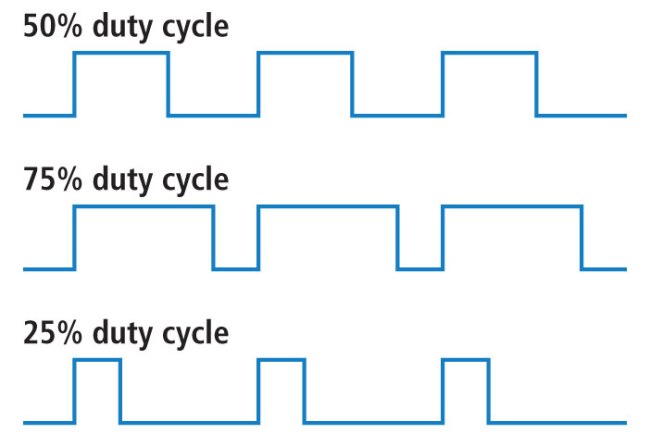
[](https://cdn.sparkfun.com/assets/8/2/c/8/3/512e85a6ce395f4d64000000.jpg)

Robotic claw controlled by a servo motor using pulse-width modulation

## **Duty Cycle**

When the signal is high, we call this “on time”. To describe the amount of “on time”, we use the concept of duty cycle. Duty cycle is measured in percentage. The percentage duty cycle specifically describes the percentage of time a digital signal is on over an interval or period of time. This period is the inverse of the frequency of the waveform.

If a digital signal spends half of the time on and the other half off, we would say the digital signal has a duty cycle of 50% and resembles an ideal square wave. If the percentage is higher than 50%, the digital signal spends more time in the high state than the low state and vice versa if the duty cycle is less than 50%. Here is a graph that illustrates these three scenarios:

[](https://cdn.sparkfun.com/assets/f/9/c/8/a/512e869bce395fbc64000002.JPG)

50%, 75%, and 25% Duty Cycle Examples

100% duty cycle would be the same as setting the voltage to 5 Volts (high). 0% duty cycle would be the same as grounding the signal.

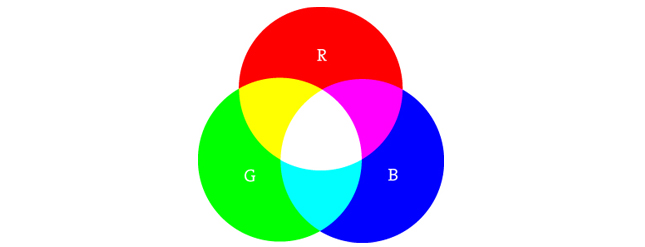
## **Examples**

You can control the brightness of an LED by adjusting the duty cycle.

[](https://cdn.sparkfun.com/assets/4/a/f/c/e/512e8754ce395fac64000000.jpg)

PWM used to control LED brightness

With an RGB (red, green, blue) LED, you can control how much of each of the three colors you want in the mix of color by dimming them with various amounts.

[](https://cdn.sparkfun.com/assets/c/f/a/7/e/51479890ce395f1f2f000000.jpg)

Basics of color mixing

If all three are on in equal amounts, the result will be white light of varying brightness. Blue equally mixed with green will get teal. As slightly more complex example, try turning red fully on, and green 50% duty cycle and blue fully off to get an orange color.

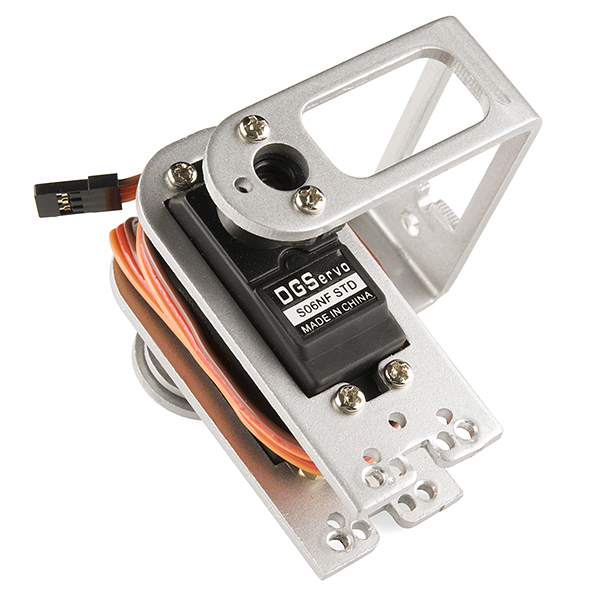
[](https://cdn.sparkfun.com/assets/b/9/f/f/8/512e8756ce395f8e60000000.jpg)

PWM can be used to mix RGB color

The frequency of the square wave does need to be sufficiently high enough when controlling LEDs to get the proper dimming effect. A 20% duty cycle wave at 1 Hz will be obvious that it’s turning on and off to your eyes meanwhile, 20% duty cycle at 100 Hz or above will just look dimmer than fully on. Essentially, the period can not be too large if you’re aiming for a dimming effect with the LEDs.

You can also use pulse width modulation to control the angle of a servo motor attached to something mechanical like a robot arm. Servos have a shaft that turns to specific position based on its control line. Our servo motors have a range of about 180 degrees.

Frequency/period are specific to controlling a specific servo. A typical servo motor expects to be updated every 20 ms with a pulse between 1 ms and 2 ms, or in other words, between a 5 and 10% duty cycle on a 50 Hz waveform. With a 1.5 ms pulse, the servo motor will be at the natural 90 degree position. With a 1 ms pulse, the servo will be at the 0 degree position, and with a 2 ms pulse, the servo will be at 180 degrees. You can obtain the full range of motion by updating the servo with an value in between.

[](https://cdn.sparkfun.com/assets/a/6/7/4/e/512e8755ce395fc264000000.jpg)

PWM used to hold a servo motor at 90 degrees relative to its bracket