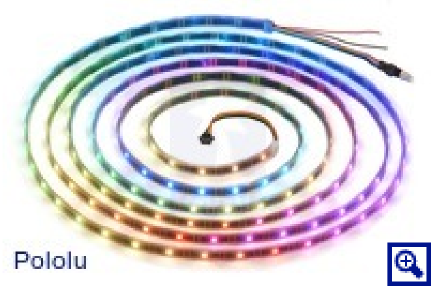
Addressable RGB 150-LED Strip, 5V, 5m (APA102C)



Pololu item #: 2554 **25** in stock

Price break	Unit price (US$)
1	82.95
10	74.65

Quantity: 1 Add to cart

backorders *allowed* Add to wish list

This LED strip has a length of **5 m** and contains **150 RGB LEDs** that can be individually addressed using an easy-to-control SPI interface, allowing you full control over the color of each RGB LED. The flexible, waterproof strip runs on 5 V and can be chained with additional APA102C strips to form longer runs or cut apart between each LED for shorter sections.

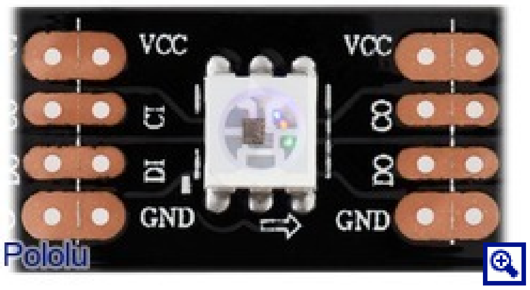
Select options: 5 m ▼ 30 per meter ▼ Go ▶

.

Description **Specs (7)** **Pictures (23)** **Resources (6)** FAQs (0) **On the blog (1)**

Overview

These flexible RGB LED strips are an easy way to add complex lighting effects to a project. Each LED has an integrated driver that allows you to control the color and brightness of each LED independently. The combined LED/driver IC on these strips is the extremely compact APA102C in a 5050 LED package, which enables higher LED densities. In the picture on the right, you can actually see the integrated driver and the bonding wires connecting it to the green, red, and blue LEDs, which are on at a very dim setting.



Close up of one segment of an APA102C-based LED strip, with the red, green, and blue LEDs on at a low brightness.

In contrast to the WS2812B used in some of our **other similar LED strips**, which uses a specialized one-wire control interface and requires strict timing, the APA102C uses a standard SPI interface for control (with separate data and clock signals) and has no specific timing requirements, making it much easier to control. Another useful feature of the APA102C is an additional 5-bit brightness control register that allows the brightness of each pixel to be adjusted independently of its color. See the bottom of this product page for a more detailed comparison of the WS2812B and APA102C.

We offer six different kinds of APA102C LED strip with different LED densities and lengths. Our strips with **30 LEDs per meter** are available in three lengths:

- **1 meter, 30 LEDs**
- **2 meters, 60 LEDs**

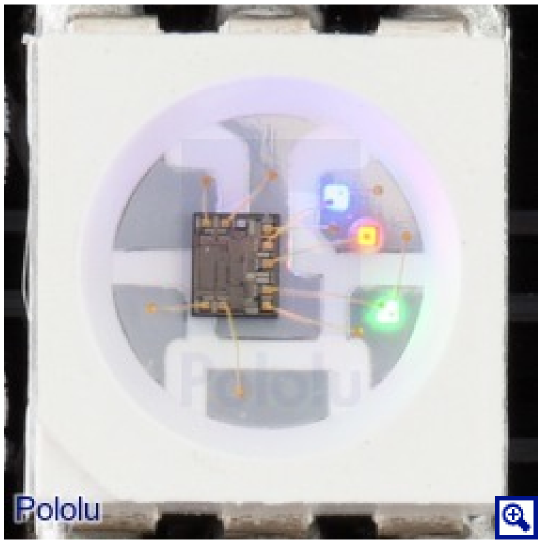
- 5 meters, 150 LEDs

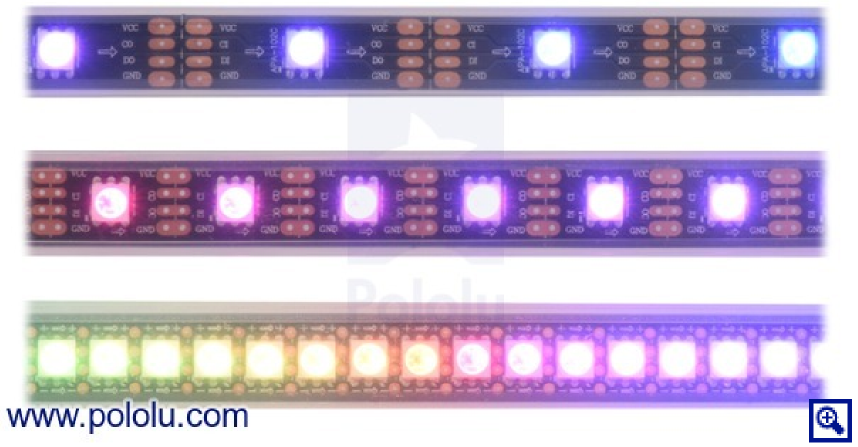
We also offer denser APA102C LED strips that have **60 LEDs per meter**:

- 1 meter, 60 LEDs
- 2 meters, 120 LEDs

Our highest-density strip has its APA102C LEDs packed together as tightly as possible, resulting in **144 LEDs per meter**:

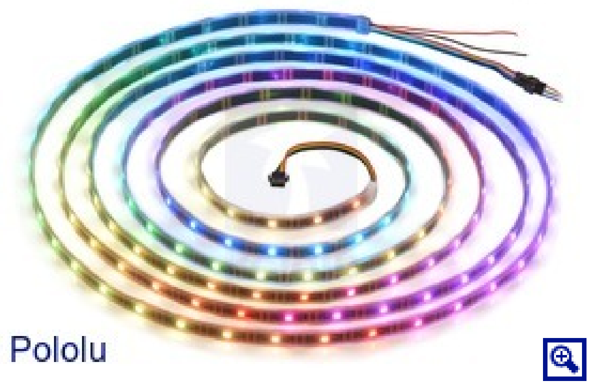
- 0.5 meters, 72 LEDs


Close up of an APA102C, with the red, green, and blue LEDs on at a low brightness.


LED side of the APA102C-based addressable LED strips, showing 30 LEDs/m (top), 60 LEDs/m (middle), and 144 LEDs/m (bottom).

Details for item #2554

This strip is **5 meters** long and has **150 LEDs** with a density of **30 LEDs per meter**. It draws about 4.1 A at full brightness (see *Current draw and voltage drop* below).



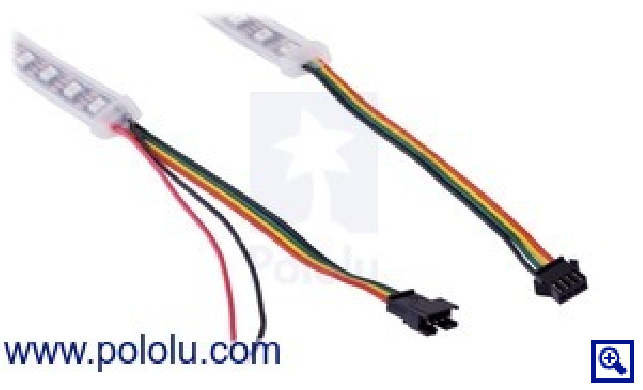
Features and specifications

- Individually addressable RGB LEDs (30, 60, or 144 LEDs per meter)
- 24-bit color control (8-bit PWM per channel); 16.8 million colors per pixel
- Additional color-independent brightness control (5-bit PWM); 32 brightness steps per pixel
- SPI control interface
- 5 V operating voltage

- Each RGB LED draws approximately 50 mA at 5 V with red, green, and blue at full brightness
- 12 mm width (30 LEDs/m) or 14 mm width (60 or 144 LEDs/m), 4 mm thickness
- Flexible, waterproof silicone rubber sheath (**IP65 protection rating**)
- Includes flexible silicone mounting brackets
- Black strip color
- Power/data connectors on both strip ends for easy chaining, and the input side includes an additional power and ground wire for alternate power connections
- Strips can be cut apart along the lines between each RGB LED segment to separate them into usable shorter sections
- Example code available for Arduino

Using the LED strip

Each LED strip has three connection points: the input connector, the auxiliary power wires, and the output connector. These can be seen in the adjacent picture, from left to right: auxiliary power wires, input connector, output connector. The strip uses 4-pin JST SM connectors.



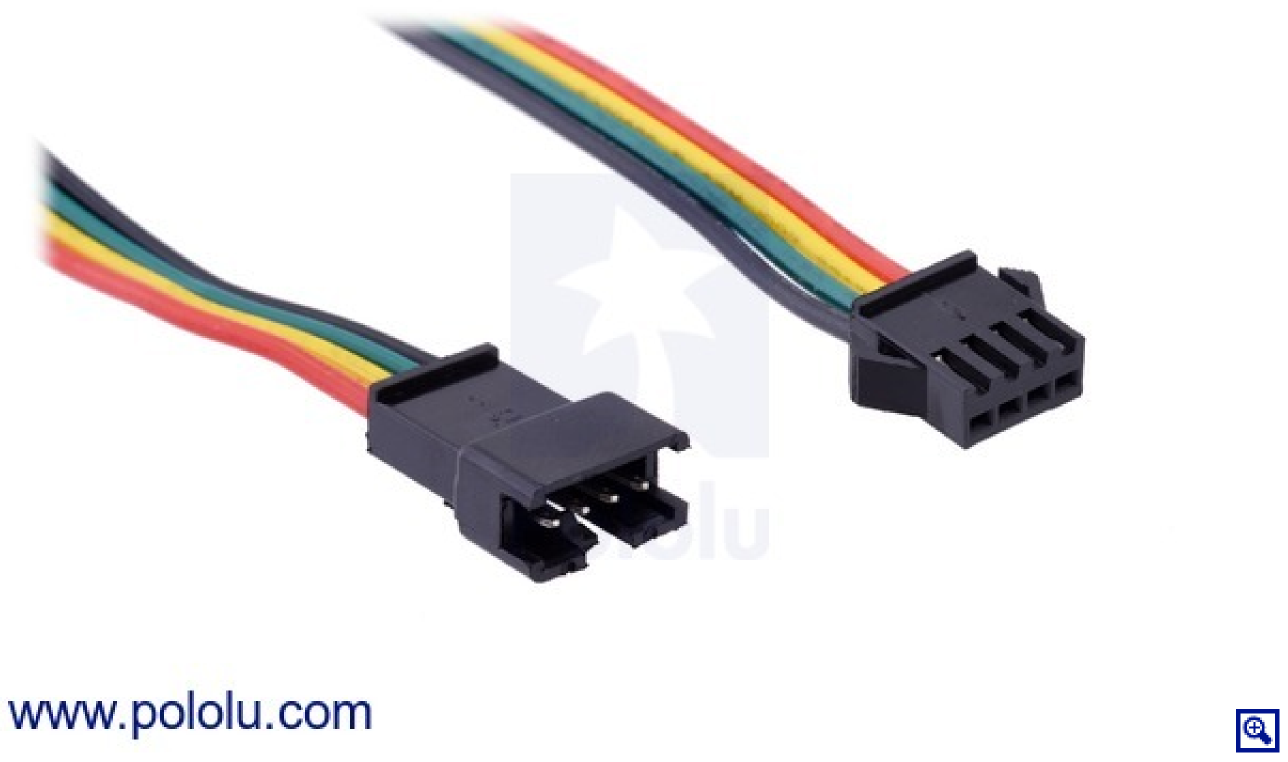
The connectors and power wires for our APA102C-based LED strips. On the left is the input end of the strip and on the right is the output end.

The **input connector** has four male pins inside of a plastic connector shroud, each separated by about 0.1″. The black wire is ground, the green wire is the data signal input, the yellow wire is the clock signal input, and the red wire is the power line.

The **auxiliary power wires** are connected to the input side of the LED strip and consist of stripped black and red wires. The black wire is ground, and the red wire is the power line. This provides an alternate (and possibly more convenient) connection point for LED strip power.

The **output connector** is on the other end of the strip and is designed to mate with the input connector of another LED strip to allow LED strips to be chained. The black wire is ground, the green wire is the data output, the yellow wire is the clock output, and the red wire is the power line.

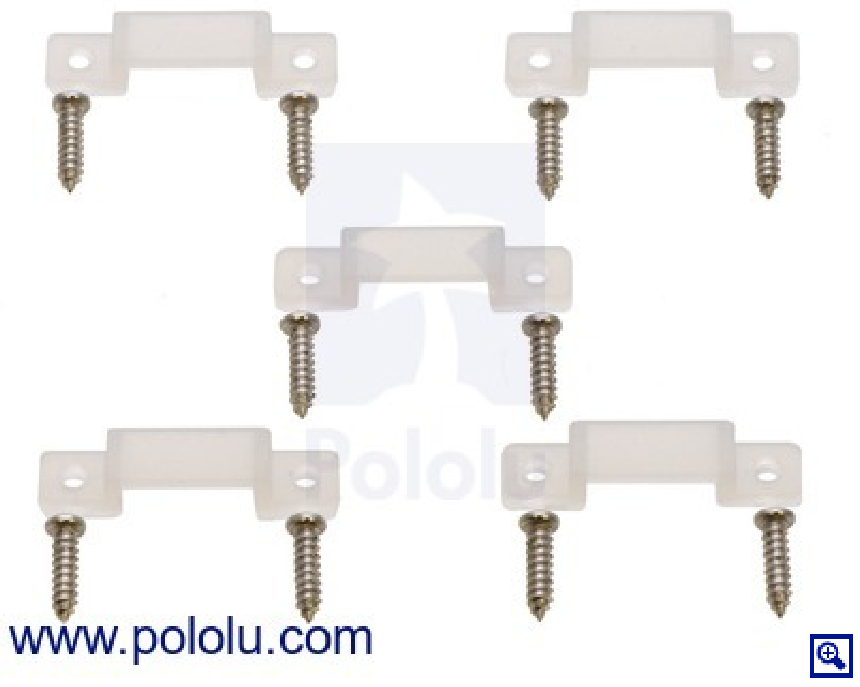
All three black ground wires are electrically connected, and all three red power wires are electrically connected.



A close-up of the JST SM connectors for our APA102C-based LED strips.

Included hardware

These LED strips ship with flexible silicone brackets and screws. Strips with lengths of 1 m or greater include five brackets and ten screws *per meter*. Our 0.5 m high-density strip ships with a total of two brackets and four screws. The brackets fit over the waterproof sheath and can be used to mount the LED strip. The LED strip also ships on a plastic reel.

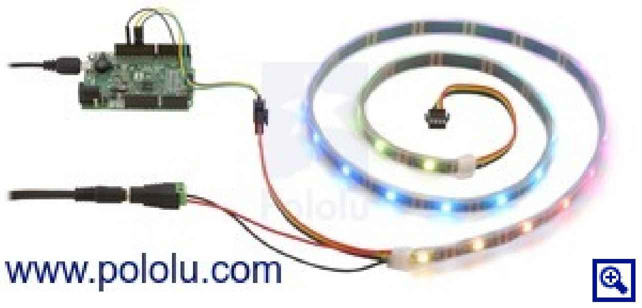


The 1m, 2m, and 5m addressable LED strips include five mounting brackets per meter; the 0.5m strip includes 2 total brackets.

Note: The strips with 30 LEDs per meter are slightly narrower than the denser strips, and the brackets included with those strips are accordingly slightly narrower than the ones included with the denser strips.

Connecting the LED strip

To control the LED strip from a microcontroller, three wires from the input connector should be connected to your microcontroller. The LED strip's ground (black) should be connected to ground on the microcontroller, and the LED strip's data input line (green) and clock input line (yellow) should each be connected to one of the microcontroller's I/O lines. The male pins inside the input connector fit the female terminations on our **premium jumper wires** and **wires with pre-crimped terminals**. If you are connecting the LED strip to a breadboard or a typical Arduino with female headers, you would want to use **male-female wires**.



Controlling an APA102C addressable RGB LED strip with an A-Star 32U4 Prime SV and powering it from a 5V wall power adapter.

We generally recommend powering the LED strip using the auxiliary power wires. Our **5 V wall power adapters** work well for powering these LED strips and a **DC Barrel Jack to 2-Pin Terminal Block Adapter** can help you make the connection between the adapter and the strip. However, you might need a **wire stripper** to strip off some more insulation from the power wires.

It is convenient that the power wires are duplicated on the input side because you can connect the auxiliary power wires to your 5 V power supply and then the power will be available on the data input connector and can be used to power the microcontroller that is controlling the LED strip. This means you can power the microcontroller and LED strip from a single supply without having to make branching power connections.

Making a custom cable

If you do not want to use our **premium jumper wires** to connect to the LED strip's input, it is possible to make a custom cable.

One option for making a custom cable is to cut off the unused output connector on the last LED strip in your chain. This can then be plugged into the input connector of the first LED strip. The wires on the output and input connectors are 20 AWG, which is too thick to easily use with our crimp pins and housings, but you could solder the wires to header pins.

Alternatively, you can get your own JST SM connectors and make a custom cable using those. The parts you would need to get are the SMP-04V-BC and the SHF-001T-0.8BS, which are described in the **SM Connector datasheet** from JST. These can be purchased from several places, including **Heilind**. You will also need some 22–28 AWG **stranded wire** and a **wire stripper**. We do not know of a great way to crimp wires onto the JST crimp pins, but we were able to successfully do it using our **narrower crimping tool** and **pliers**. (With the wider crimping tool, it is hard to avoid crimping parts of the pin that should not be crimped.) Before crimping, use pliers to bend the outer set of tabs a little bit so that they can hold on to the insulation of the wire. This makes it easier to position the crimp pin and the wire. Next, you should be able to follow the instructions on the **crimping tool product page** to crimp the wire. After that, you will probably need to squeeze the crimp pin with pliers to get it to fit into the JST plug housing. On the other end of the cable you could make a custom connector using our **crimp pins** and **crimp connector housings**, which will allow you to plug it directly into a breadboard or 0.1″ header pins.

Current draw and voltage drop

Each RGB LED draws approximately 50 mA when it is set to full brightness and powered at 5 V. This means that for every 30 LEDs you turn on, your LED strip could be drawing as much as 1.5 A. Be sure to select a power source that can handle your strip's current requirements.

There is some resistance in the power connections between the LEDs, which means that the power voltage near the end of the strip will be less than the voltage at the start of the LED strip. As the voltage drops, RGB LEDs tend to look redder and draw less current. This voltage drop is proportional to the current through the strip, so it increases when the LEDs are set to a higher brightness.

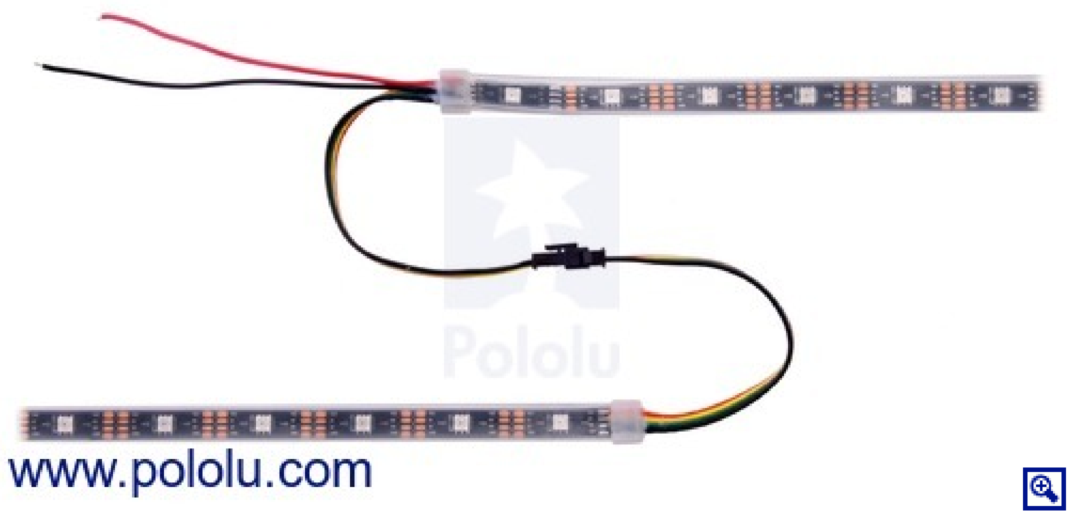
We tested the current draw and voltage drop of some LED strips by setting all the LEDs to full brightness, and these were the results:

- The 30-LED, 1-meter strip drew 1.5 A and had a voltage drop of 0.2 V.
- The 60-LED, 2-meter strip drew 2.9 A and had a voltage drop of 0.8 V.
- The 150-LED, 5-meter strip drew 4.1 A and had a voltage drop of 2.0 V.
- The 60-LED, 1-meter strip drew 3.0 A and had a voltage drop of 0.6 V.
- The 120-LED, 2-meter strip drew 4.7 A and had a voltage drop of 1.4 V.
- The 72-LED, 0.5-meter strip drew 3.3 A and had a voltage drop of 0.2 V.

The voltage drop was computed by measuring the voltage difference between ground and power on the input end of the strip, then doing the same measurement on the output end, and subtracting the two values. Note that both the current draw and voltage drop might vary from unit to unit and can also depend on details of your particular setup (such as the actual voltage across the LED strip after accounting for losses in connecting wires).

Chaining

Multiple LED strips can be chained together by connecting input connectors to output connectors. When strips are chained this way, they can be controlled and powered as one continuous strip. Please note, however, that as chains get longer, the ends will get dimmer and redder due to the voltage drop across the strip. If this becomes an issue, you can chain the data lines while separately powering shorter subsections of the chain.

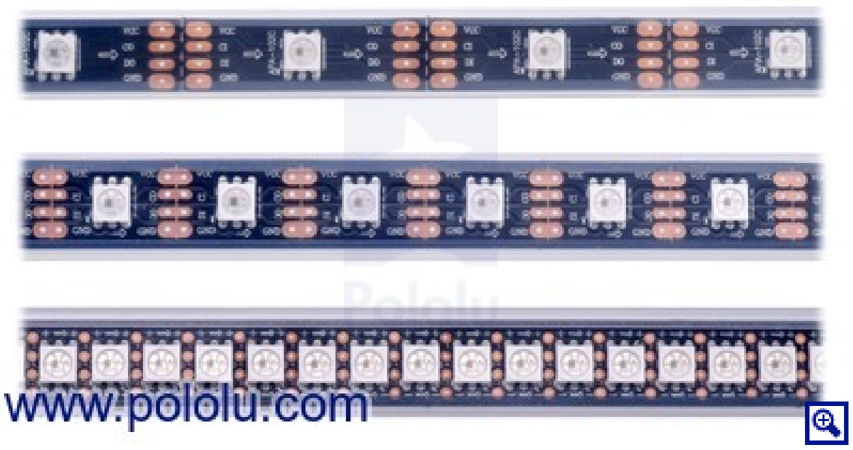


Two APA102C-based addressable RGB LED strips connected.

We recommend chains of LEDs powered from a single supply not exceed 180 total RGB LEDs. It is fine to make longer chains with connected data lines, but you should power each 180-LED section separately. If you are powering each section from a different power supply, you should cut the power wires between the sections so you do not short the output of two different power supplies together.

Cutting

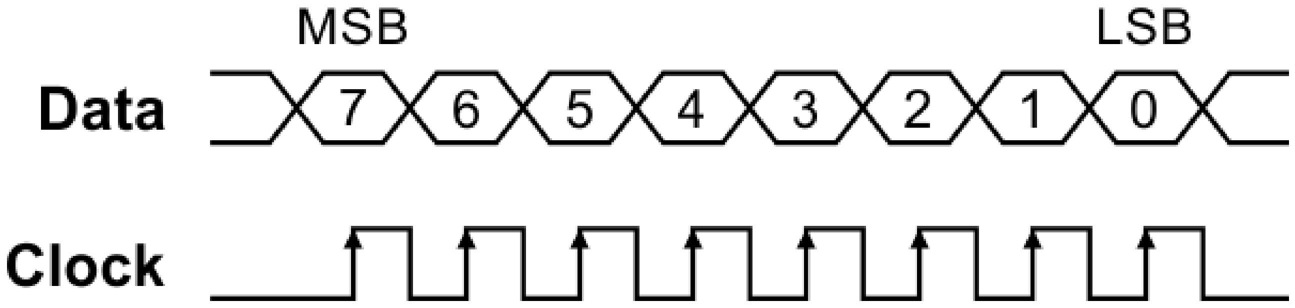
The LED strip is divided into segments, with each segment containing one RGB LED. The strip can be cut apart on the lines between each segment to separate it into usable shorter sections. The data connection is labeled **DO**, **SDO**, **DI**, or **SDI**, the clock connection is labeled **CO**, **CLKO**, **CI**, or **CLKI**, the positive power connection is labeled **VCC** or **5V**, and the ground connection is labeled **GND**. Each LED in the picture below is at the center of its own segment; there are lines printed on the PCB silkscreen where the segments can be cut.



Protocol

These LED strips are controlled through an SPI protocol on the data and clock input lines. The protocol is documented in the **APA102C datasheet** (1MB pdf), but we describe it below with some modifications that we have found to work better.

The default, idle state of the clock signal line is low, and the data signal is read on each rising edge of the clock. To update the LED colors, you need to toggle the clock line while driving the data line with the value of each bit to send; this can be done through software (bit-banging), or it can be handled by a hardware SPI peripheral in a microcontroller. There is no minimum clock frequency, although using a lower frequency means that it will take longer to update the entire sequence of LEDs (especially when controlling a long strip or many strips are chained together), so you will probably want to use the fastest practical clock speed to get the best update rate.

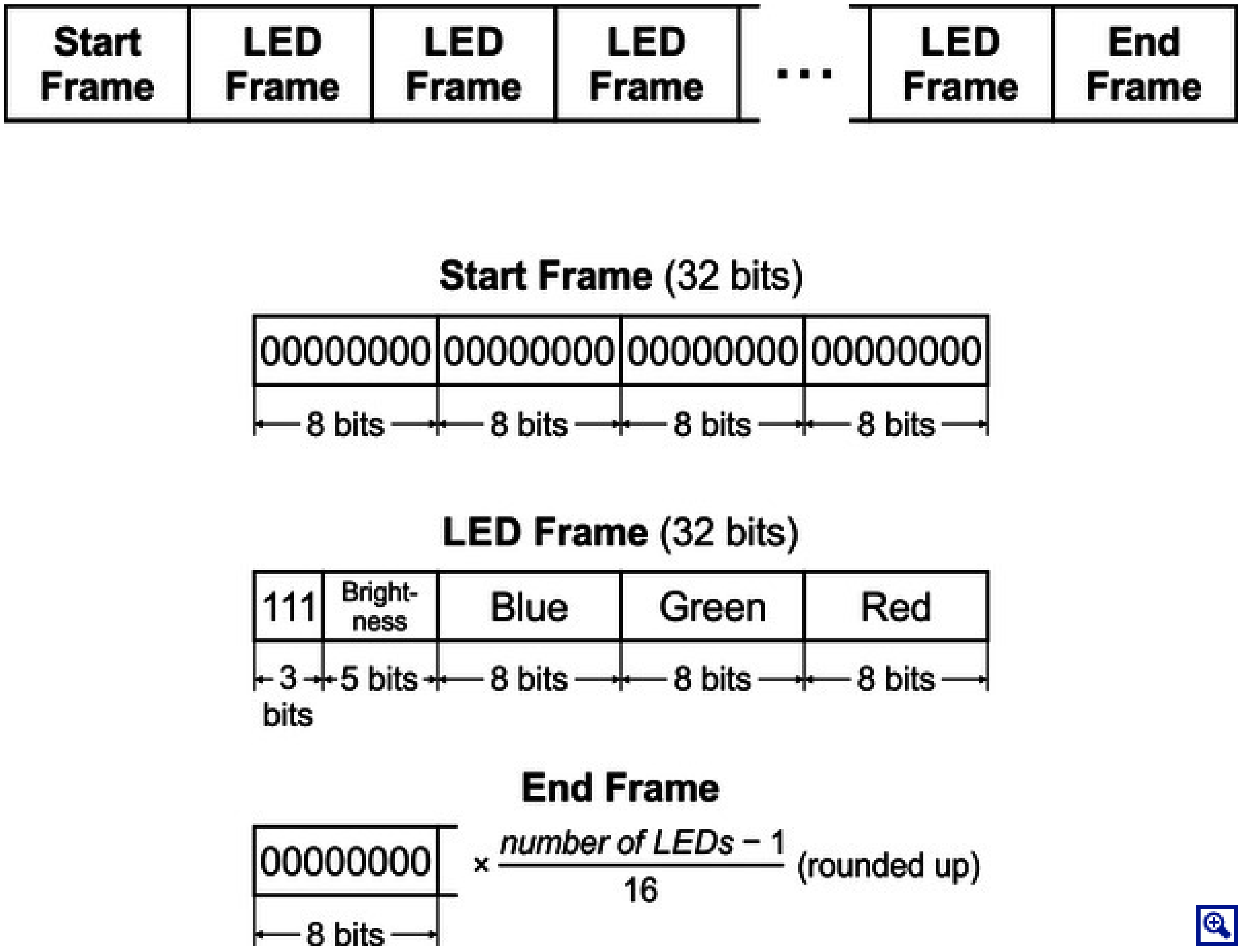


APA102C control signal timing diagram.

The data for each LED is encoded as a sequence of 32 bits (4 bytes) called an LED frame. The first three bits of the LED frame should be ‘1’. The next 5 bits are a “global”, color-independent brightness value (0–31) that is applied equally to all three color channels. The remaining 24 bits are the color values, in BGR (blue-green-red) order. Each color value uses 8 bits (0–255). The most significant bit of each value is transmitted first. The first LED frame transmitted applies to the LED that is closest to the data input connector, while the second color transmitted applies to the next LED in the strip, and so on.

To update all the LEDs in the strip, you should send a “start frame” of 32 ‘0’ bits, then a 32-bit “LED frame” for each LED, and finally an “end frame”. If you send fewer LED frames than the number of LEDs on the strip, then some LEDs near the end of the strip will not be updated.

The APA102 datasheet recommends that the end frame be composed of 32 ‘1’ bits, but we have found this does not work reliably in certain situations and can sometimes lead to glitches. This can be avoided by using an end frame that consists of at least $(n-1)$ extra clock edges, where n is the number of LEDs, with ‘0’ on the data line. It is often easiest to round up to a multiple of 16 clock edges so that you are counting bytes instead (there are 2 clock edges in a bit and 8 bits in a byte); you would therefore send $((n-1)/16)$ bytes (rounded up to the next whole number). For a more detailed explanation, see the comments in the source code of our APA102 Arduino library, discussed below.



APA102C data format.

For example, to update all 30 LEDs on a 1-meter strip, you should send a 32-bit start frame, thirty 32-bit LED frames, and a 16-bit end frame, for a total of 1008 bits (126 bytes). If multiple strips are chained together with their data connectors, they can be treated as one longer strip and updated the same way (two chained 1-meter strips behave the same as one 2-meter strip).

Each RGB LED receives data on its data input line and passes data on to the next LED using its data output line. The update rate is generally limited only by the speed of the controller; our Arduino library below can update 60 LEDs in about 1.43 milliseconds, so it is possible to update nearly 700 LEDs at 60 Hz. However, constant updates are not necessary; the LED strip can hold its state indefinitely as long as power remains connected.

Note: The minimum logic high threshold for the data and clock signals is 3.5 V, so you should use level-shifters if you want to control these strips from 3.3 V systems. It might be possible to control them with 3.3 V signals directly, but using the strip out of spec like this could lead to unexpected problems.

Sample code

To help you get started quickly, we provide an **APA102 Arduino library** (it also works with our Arduino-compatible **A-Star modules**).

Additionally, the **DotStar Arduino library** and **Raspberry Pi Python module** from Adafruit should work with these strips since the DotStars are based on the APA102. The **FastLED Arduino library** is another option that focuses on performance and provides advanced functionality like color correction.

Comparison with WS2812B LED Strips

Like the WS2812B used in **some of our older LED strips**, the APA102C combines an RGB LED and driver into a single 5050-size package, allowing them to be packed as densely as 144 LEDs per meter. However, while the WS2812B uses a one-wire control interface with strict timing requirements (timing requirements so strict that it is typically impractical to have interrupt-based events running on the controlling microcontroller while it is updating the WS2812B LEDs), the APA102C uses a standard SPI interface, with separate data and clock signals, that lets it accept a wide range of communication rates; the trade-off is that two I/O lines are required to control it instead of just one.

The APA102C provides a 5-bit color-independent brightness control that is not available on the WS2812B. This feature can be used to vary the intensity of each pixel without changing its color, and it enables much subtler variations at the low end of the LEDs' brightness range.

In addition, the APA102C uses a much higher PWM (pulse-width modulation) frequency for controlling each color channel—about 20 kHz, compared to around 400 Hz on the WS2812B. As a result, APA102C LEDs can be less prone to flickering when recorded with a camera and are more suited to applications like persistence-of-vision (POV) displays. (The color-independent brightness is modulated separately at about 600 Hz).

For further comparison of the ICs, see the **WS2812B datasheet** (266k pdf) and **APA102C datasheet** (1MB pdf).

While APA102C strips and WS2812B strips are physically very similar, they are **not** functionally compatible with each other. The easiest way to tell them apart is to look at the strips' end connectors and the connections between each LED segment: WS2812B strips have three connections (power, data, and ground), while APA102C strips have four (power, clock, data, and ground). On strips with 30 LEDs/m, you can also check whether "WS2812B" or "APA-102C" is printed next to each LED.

Related products

Addressable RGB 30-LED Strip, 5V, 1m (APA102C)

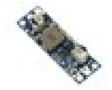
Addressable RGB 60-LED Strip, 5V, 2m (APA102C)

Addressable RGB 60-LED Strip, 5V, 1m (APA102C)

Addressable RGB 120-LED Strip, 5V, 2m (APA102C)

Addressable High-Density RGB 72-LED Strip, 5V, 0.5m (APA102C)

Pololu 5V, 9A Step-Down Voltage Regulator D24V90F5

 **Pololu 5V Step-Up Voltage Regulator U3V50F5**

Pololu 5V Step-Up/Step-Down Voltage Regulator S18V20F5

Wall Power Adapter: 5VDC, 3A, 5.5×2.1mm Barrel Jack, Center-Positive

Wall Power Adapter: 5VDC, 5A, 5.5×2.1mm Barrel Jack, Center-Positive

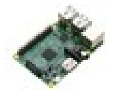
DC Barrel Jack to 2-Pin Terminal Block Adapter

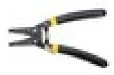
A-Star 32U4 Micro

A-Star 32U4 Mini ULV

A-Star 32U4 Prime LV

Arduino Uno R3

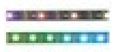

Raspberry Pi Model B+

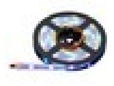

Wire Stripper 10-20 AWG Solid (12-22 AWG Stranded)

Related categories

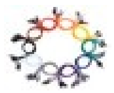
APA102C-Based LED Strips

APA102-Based Addressable RGB LEDs


WS2812B-Based LED Strips


LED Strips

Programmable Controllers


Premium Jumper Wires

Wires with Pre-crimped Terminals

Regulators and Power Supplies