Tips and tricks for driving WS2811 LED strips - Part 1: Taming the wily WS2811

Lee Goldberg - March 03, 2015

Thanks to their versatility, commodity-level pricing and the eye-popping visual effects they create, programmable LED strips have become a staple ingredient for both Maker-style projects and more conventional LED lighting applications. But as I quickly discovered when using them as a “Mood LED” for my Smart Hat project³, the serial protocol bus used to address and control the LEDs can be somewhat complicated and tricky to use. I’ve summarized some of the key design issues and located a few resources to help shorten your climb up the learning curve when you need to include LED strips in one of your projects.

There are several types of LED strips but this article will focus on driving LED arrays which use WS2811⁴ and WS2812⁴ driver chips. This is in part because products like Adafruit’s NeoPixel strips which use them tend to be less expensive than similar products which use LPD8806 drivers. The cost was only one of the reasons I used Adafruit’s NeoPixel strips (they are based on the ’2811) for my Smart Hat project. I also like NeoPixels because they have consistently high quality and come in a wide variety of pixel densities and offer several form factors besides the traditional strip such as...
rings, matrixes, and several different types of individual pixels.

On the other hand, commanding a WS2811 device via its serial control bus is not a straightforward task. Part of this is because it uses a single combined clock and data line to ripple the instructions for each LED down the serial daisy chain bus which runs between each pixel (Fig. 3). This means that that pixels can't be addressed individually so any change requires refreshing the instructions loaded into all the pixels upstream of the one you want to update.

To give you a sense of what's involved, here's a quick overview of the sequence your MCU has to go through when it talks to a '2811-based LED strip:

After resetting the LED chain by holding the input low for 50usec, the MCU or whatever's controlling the string begins sending a stream of 24-bit color codes (8 bits each for the R, G & B LEDs) down the line. The first set of 24-bit RGB values is latched and displayed by the first LED pixel which allows the remainder of the bit stream to pass down the chain. The second LED pixel grabs RGB sequence #2, displays it, and so on until the end of the chain is reached (Fig.4).
Fig. 4: Programming WS2811 LED drivers requires the 24-bit color code for each pixel to be cascaded down the serial control bus until it reaches its intended location. Courtesy of World Semiconductor.

To complicate matters further, the receive clock the LED pixels use to decide when to latch the data is derived from the serial data stream. To make generating the clock easier, the data line must toggle during each bit period (sort of like an NRZ code) so the only difference between a "1" and a "0" is how long the data line spends in each state. Don’t panic if you’re not familiar with pulse width-modulated (PWM) bit streams, Figure 5 provides a visual explanation of how it works. (Note: If you’re still shaky on how, there’s a post on the "Do it Yourself Christmas" forum which provides a more detailed explanation).

Fig. 5: Generating logic "1", "0" and "reset" states on the WS2811 control bus. Courtesy of World Semiconductor.

Even if you are still struggling with some aspect of PWM-based data streams, it’s easy to see that generating instructions for a string of WS2811 LED drivers requires some very specific and complicated timing that can make the driver code a tad tricky to write from scratch. Fortunately, much of the hard work has already been done and there are libraries of LED strip drivers and applications available for Arduino, Raspberry Pi, LaunchPad and other popular platforms. There are also assembly language driver routines for WS2811-style products available for a number of MCUs, including Atmel’s popular 8-bit machines, TI’s TMS430 series, and Microchip’s PIC16F15xx devices.

We'll take a closer look at implementing LED strips on some of these platforms in subsequent installments, as well as a few of the more unique applications people have created using LED strip
technologies. But if you are too excited to wait for me to post my next article, you can get a great head start by dropping over to the tutorials section of the AdaFruit site and going through their “NeoPixel Uberguide”. In addition to some helpful info on programming the LED strips, Adafruit's tutorial provides a lot of practical introductory material on the electrical and mechanical aspects of using LED strips I won't be able to include in this series.

References

1. "WS2811 Signal line 256 Gray level 3 channel constant current LED drive IC" data sheet and application notes from World Semiconductor (Note: the title is spelled as it appears on the data sheet).
2. "WS2811 pixel compatibility" a thread on Do it Yourself Christmas forum
4. "WS2811 & WS2813 - Single-wire three-channel constant current LED driver IC with PWM" a data sheet from World Semiconductor
5. NeoPixel Uberguide – Adafruit

Related articles:

- Controlling Multiple WS2812B-Based LED Strips
- The skinny on LED strips