Development of a 3D display

Balduin Dettling

May 13, 2016

Preface

Motivation

For as long as I can remember, I've been fascinated by all kinds of technology. It was clear to me that my final thesis would be a practical one in the field of engineering. However, I didn't have anything specific in mind for quite some time.

One day, I stumbled upon a project report on the internet about a so-called persistence of vision display. It was a simple row of LEDs, driven by a microcontroller, and mounted on an electric motor. The contraption was (and presumably still is) able to draw images into the air by precisely controlling the row of LEDs. Fascinated by this, I did some research and found that a plethora of similar projects exists, some even with detailed instructions on how to build the device in question.

I had the idea of building several rows of LEDs and arranging them on top of each other. That way, I could display several two-dimensional bitmap images above another, resulting in a three-dimensional image.

Having had no previsous experience in the field of electronics, it seemed a bit daunting to build a 3D display. I could, however, motivate myself to attempt it anyway by telling myself that the project wouldn't be much more complex than all of the 2D versions out there (which turned out to be not very far from the truth).

Credits

First and foremost, I would like to express my gratitude to Mr. Patrick Spengler, my physics teacher and the mentor of this thesis. Not only did he solve numerous malfunctions, but he also is partly responsible for my interest in electronics and engineering.

I am thankful to Ms. Ursula Schamberger, teacher of the woodworking class, where I built the wooden frame. Likewise, I would like to thank Mr. Hanspeter Rieder, who kindly granted me access to the school's electronics workshop.

Last, but most certainly not least, I am indebted to Prof. Heinz Domeisen. He mentored the advancement and refinement of the thesis in preparation for the national competition organised by "Swiss Youth in Science".

Declaration of authorship

I hereby declare that the following thesis has been authored entirely by myself, Balduin Dettling, and confirm that all external sources of information are cited correctly.

Location and Date: Signed:

Contents

1. Introduction

1.1 Current State of The Art

As discussed before, there is no shortage of projects that are similar to mine, but only display 2D images. The name "POV display", short for persistence of vision display, has become commonplace. Another popular name is "Propeller Clock" — Propeller because it rotates like one, and clock because the round shape is ideal for displaying an analog clock.

As far as I know, however, only two people have built a working version of a three-dimensional POV display so far (and have published it on the internet).[1] Neither of these is being produced on a grand scale, instead they are — like most 2D versions — side projects of makers or engineers.

1.2 Goals

The objective of this project is to build a working prototype of a 3D persistence of vision display and to document the process of doing so. The necessary steps are (roughly) as follows:

- Understand and explain how and why such a display can work in principle
- Come up with realistic technical specifications, founded on the previously outlined theory
- Select parts and build the device according to the specifications
- Write a program that makes the device display a 3D image
- Draw conclusions: Did I achieve my goals? What could have been done better? How can the project be further improved?

[1] It turned out that there was a third one: When I posted a picture of my project on the internet, a reddit user told me about the display he built years before.

2. Theory and Planning

2.1 Principle of Operation

2.1.1 Persistence of Vision

Persistence of Vision refers to the optical illusion that results whenever the image we look at quickly changes. In such situations, we don't immediately see the new image, instead we continue seeing the old image for a short timespan after it has vanished. If wikipedia is to be believed, this timespan is about $\frac{1}{25}$ of a second, but of course it is going to be different for each pair of eyes.

Although we don't normally see this effect in action, there are situations where it becomes apparent. For example, if a car drives by at nighttime, we see traces of light behind the actual car lights for a short moment (Assuming our eyes don't follow the car and stay still relative to the environment). In general, bright flashes of light followed by (relative) darkness tend to be the most pronounced manifestation of persistence of vision. It is exactly this situation that can be recreated and exploited for the successful operation of a persistence of vision display.

2.1.2 Exploiting Persistence of Vision

Normally, Persistence of Vision is nothing more than an interesting effect. However, we can use this weakness of the human eye to our benefit.

If a light source is spun quickly enough in a circle, we don't see a spinning point anymore, but instead we see one solid circle. This is because the rotational period is shorter than the time it takes for the light to fade away on our retinas. The rotational period at which we stop seeing a flickering circle and start seeing a solid, still circle is called the flicker fusion threshold.

By controlling this light source quickly and accurately, we can make it so it is always turned on at certain points in the circle, and always turned off at others. By doing so, we are effectively multiplexing a zero-dimensional display (a point-like light source, e.g. an LED) so that it displays a one-dimensional image. By rotating a whole row of LEDs (a one-dimensional display), we can display two-dimensional images. Finally, if there are multiple rows of LEDs that together form a two-dimensional display, it is possible to display a three-dimensional image.

2.2 Specifications

- Resolution of 16 * 100 * 10 pixels (radius * circumference * height)
- Each pixel consists of an RGB LED
- Colour depth of 3 bit per pixel, or one bit per LED
- Rotational frequency of at least 30 Hz, or as fast as it needs to be in order to not flicker

2.3 Components

2.3.1 Control

For controlling the LEDs, I have bought a Teensy 3.1 microcontroller development board. It has a small footprint — ideal for the fast rotation — and still packs quite some power: The processor runs at 96 MHz at a register width of 32 bit. There are 34 configurable I/O pins.

The controller has to have some kind of reference point, so that it can calculate the current speed and angular position of the device. For this, I used a simple hall sensor on the rotating side, which passes by a stationary magnet after each rotation.

2.3.2 Data Transmission

34 I/O pins are quite a lot, but not quite enough to directly control 480 LEDs. In order to still access all those LEDs individually, I had to use LED drivers. These come in the form of integrated circuits and receive some amount of data that they use to control a corresponding amount of LEDs.

Initially, I had planned to implement a colour depth of 8 bits, allowing for $2^{24} = 16777216$ different colours. However, there were several difficulties that I would have had to deal with. First of all, all of the available LED drivers had one or both of the following shortcomings:

- Controlled using I^2C instead of SPI

  Because I^2C uses pullup resistors instead of a push-pull drive, it takes a little moment for the logic level to return to its normal state. In order to ensure correct data transmission, it is necessary to set a relatively slow clock speed. The two I^2C interfaces of the Teensy only support clock speeds of 2.4 MHz — and because I^2C requires some overhead data to send out the slave's addresses, the actual data rate will be even lower.

  However, the minimum data rate required to control the LEDs at an 8-bit colour depth is as follows:

  $$24 \frac{\text{bit}}{\text{pixel}} * 16000 \frac{\text{pixels}}{\text{rotation}} * 30 \frac{\text{rotations}}{\text{second}} = 11.52\,\text{Mbps}$$

  This means that I'd need at least 5 parallel I^2C lines to control all the LEDs with 8 bit colour depth. Because each Teensy only has two of them, I'd need three Teensy boards, which would be possible but very cumbersome.

- Insufficient PWM frequency

  If the colour depth is more than one bit, the LEDs will have to be dimmed using PWM. Normally, a rather low frequency of a few hundred to a few thousand Hz is sufficient as a PWM frequency, and most LED drivers modulate at frequencies within that range. In my case, however, the LEDs are moving into a new pixel 3000 times a second (assuming rotationat 30 Hz). This means that a PWM frequency under 3000 Hz will involve losing information, and one slightly above 3000 Hz still won't look good. There would have to be a significant number of PWM cycles within each pixel, for example 10, which would require a PWM frequency of 30 KHz.

Because of that, I ditched the idea of 8-bit colour depth and instead went for the simpler approach of one-bit colour depth. Both problems are solved by switching to 1 bit: The data rate doesn't have to be as high (and there are lots of chips that are controlled by SPI), and PWM is not necessary anymore.

2.3.3 LED drivers

After quite some searching around, I decided to use the TLC5927 by Texas Instruments. This is essentially a shift register with some extra features. Like every shift register, it has a serial data input (SDI), a serial data output (SDO), and a clock pin (CLK). Additionally, there is a latch pin (LE), which serves the purpose of only refreshing the state of the LEDs once all data has been sent.

Furthermore, there is a pin called "output enable" (OE). If it is at GND, all LEDs are turned off, and it's at V_{DD}, they are controlled by the data that the driver chip has received.

The most important feature, however, is the fact that each output is controlled by a constant current driver. Without these, it would've been necessary to solder a resistor in series with each individual LED, for a total of 480 resistors, 960 additional solder joints and a whole lot of wasted board space. With the constant current drivers, however, I could use a single resistor to set the current for all 16 LEDs attached to one chip.

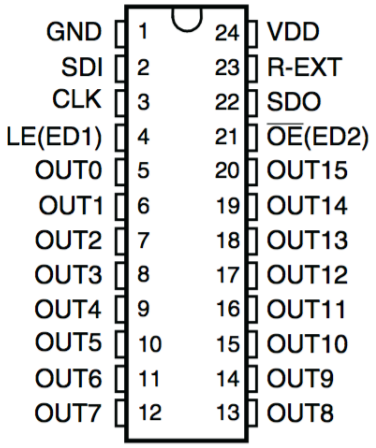

Figure 1: Pin layout of the TLC5927 LED driver

Unlike most simple shift registers, the drivers' outputs are current sinks instead of current sources. This will be important when selecting the LEDs: In order to control each colour separately in an RGB LED, the cathodes have to be separated, while the anodes can be connected together.

2.3.4 LEDs

Like all the other parts, I ordered the LEDs from Digikey. Their website contains an excellent tool, enabling the user to restrict the selection based on different criteria. My requirements were the following:

- Colours: Red, green and blue
- Current: 20 mA or more. Because the LEDs are within one pixel only $\frac{1}{100}$ of the time, they are essentially pulse width modulated, reducing their effective brightness. In order to still see the image clearly without darkening the room, they have to be bright enough.
- Mounting type: Surface mount, so that I can route traces past the LEDs on the other side of the PCB
- Size: At most 5 * 5 mm, preferably smaller, but still large enough to be able to solder them by hand
- Diffused lens, so the colours mix well even when viewing the LED directly instead of using it for illumination.

These requirements reduced the category "LED Indication - Discrete" from 18940 to 130 products. From those remaining products, I chose the one that was cheapest in a quantity of 160. This led me to an LED with the beautiful name CLVBA-FKA-CAEDH8BBB7A363.

2.3.5 Capacitors

A digital circuit like an LED driver can change its current consumption significantly within a short amount of time. Because the connections to the power supply are relatively long, they have a substantial parasitic inductivity. So when the IC suddenly needs more current because all its LEDs have just been switched on, this current is not available instantly. Worse yet, if the LEDs are on and then are turned off, the current can't immediately stop flowing and will induce dangerous voltage spikes.

In order to prevent those two situations from happening, I used capacitors. Namely, each LED driver has its own 1 µF ceramic capacitor.

2.3.6 Motor

In order for the whole device to rotate, I needed a motor. Luckily, I could salvage an appropriate part from an old RC motor boat. It's a big and heavy brushed motor. The fact that it has a built-in fan and was water-cooled in the boat leads me to think that its power output is much higher than what I need.

Because of its high torque, I could attach the motor directly to the rotating shaft instead of using gears or a belt drive. That way there are less moving parts and the motor can turn at a lower rotational speed, both resulting in less wear and noise.

2.3.7 Power Transmission

To transmit power from the stationary power supply to the rotating part, I used slip rings. I made a relatively simple version, consisting of two round pieces of copper sheet on the rotating end, and two long pieces of copper sheet on the stationary part.

The slip rings must be able to transmit the current necessary to light all LEDs up at once. Assuming that each LED needs 20 mA, this current is about 9.6 A. Additionally, there is a small current draw by the microcontroller, but it will be negligible in comparison. Testing has shown that the slip rings don't heat up noticeably when conducting 10 A for several minutes.

In practice, however, this current is not reached since the vast majority of pixels are switched off when displaying a typical 3D image.

Because two pieces of copper sheet rubbing against each other aren't the most reliable electrical connection, I used a 2200 µF electrolytic capacitor to smooth out the supply voltage. Even with it, I found that the pressure between the two copper conductors had to be quite high for them to form a good contact.

2.4 Circuit Diagram

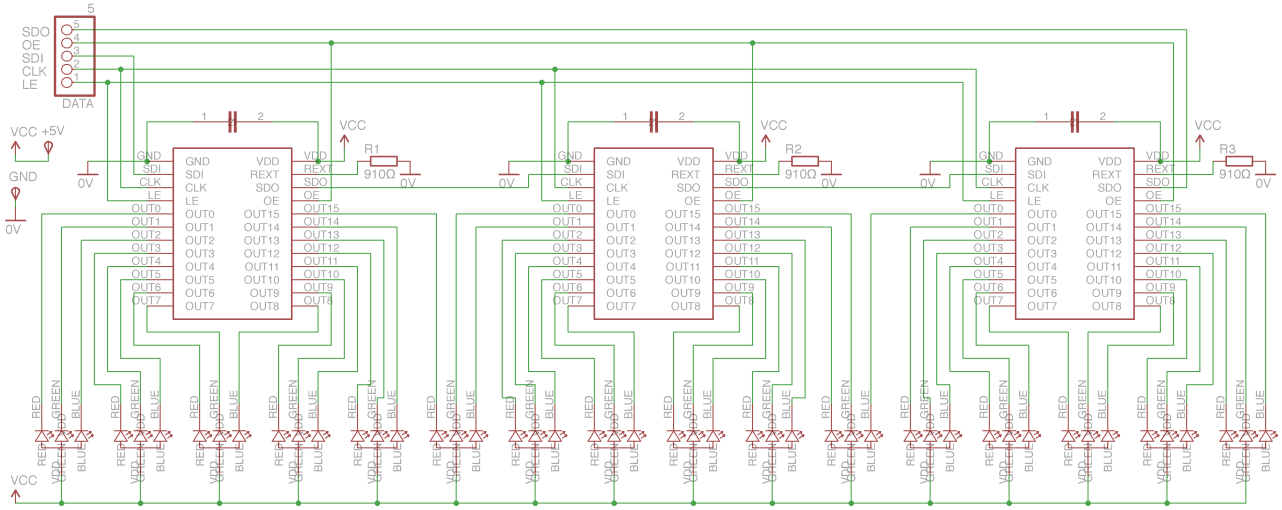

Figure 2: The finished circuit diagram

The circuit diagram — as well as the PCB layout — were created in Eagle. Its free version is limited to two-layer boards of sizes up to 10 cm by 8 cm, but that's more than enough for my purposes.

In figure 2, the final version of the schematic is pictured. For the sake of simplicity and readibility, the layout corresponds roughly to how the board will look.

I had to configure the footprints for the LED drivers and the LEDs by myself, which I did by reading about the physical dimensions in the respective data sheet. For the resistors and capacitors, Eagle had pre-built footprints which I could use.

2.4.1 Data Interface

For transmitting data to the LED drivers, I used a simple row of 5 pins. Two of those pins are the data lines: The input is connected to the first chip's input, and the output comes from the last chip's output. The rest of the pins (clock, output enable and latch enable) are control lines, which go to all three chips simultaneously. The order of these connections was determined when making the PCB layout.

2.4.2 LED Drivers

The LED drivers are arranged in a so-called daisy chain configuration, where the serial data output of each chip is connected to the serial data input of the next chip. This is not only the case within one board, but also between the different boards: The data output of one board is connected to the next board's input. That way, all of the 30 LED drivers can be controlled by a single SPI bus without anything like a chip select signal. To send data to the chips, the microcontroller can simply send out 480 bit without interruption.

The external resistor which is connected to each LED driver sets the output current. The following equation, taken from page 15 in the datasheet, describes the relation between the output current (I_{OUT}) and the resistor's value (R_{EXT}):

$$I_{OUT} = 15 * \frac{1.25\,V}{R_{EXT}}$$

Let's find out the resistance necessary for the output current to be 20 mA:

$$R_{EXT} = 15 * \frac{1.25\,V}{20\,mA} = 937.5\,\Omega$$

The next smaller value within the E24 series is 910 Ω. By choosing this resistance, the output current is 20.6 mA, which is close enough to the desired 20 mA. I ordered 40 through-hole resistors of this value, because at the time I had in mind to etch my own boards at home.

2.4.3 LEDs

The LEDs have their common anodes connected to V_{DD}. The cathodes are connected to a driver output each, and are arranged red - green - blue (from left to right). Series resistors aren't needed because the LED drivers handle current regulation.

2.5 PCB Design

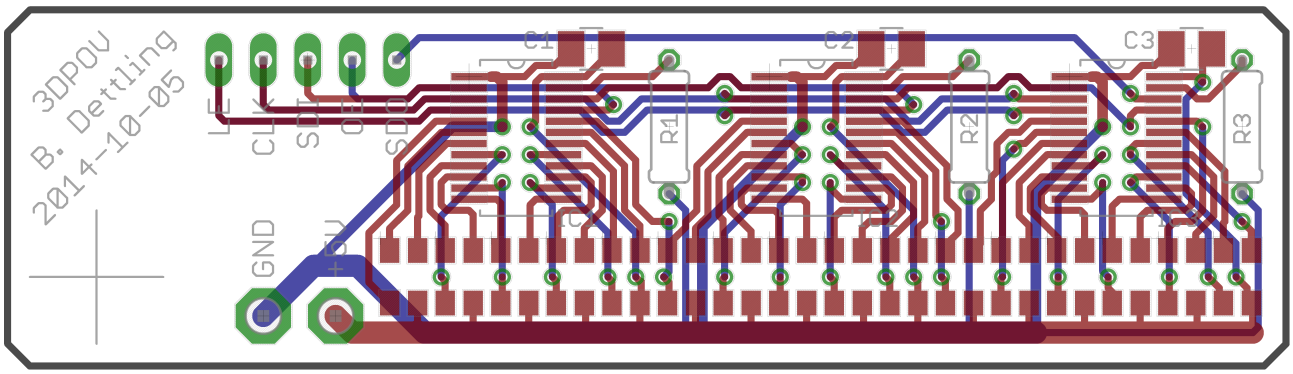

Figure 3: The PCB layout in Eagle. Red = upper copper layer, blue = lower copper layer, green = vias, grey = silkscreen.

2.5.1 General Remarks

The defining factor for the board layout was the pin layout of the LED drivers. The outputs are all at the bottom of the chip, while the data interface is at the top. This led me to the layout I have now: the LEDs are in a row at the bottom of the board where they can be directly connected to the driver outputs on the upper copper layer. The control lines are is routed on the back side of the board behind the chips, with vias going to the front side to connect to each chip. The data lines go straight from one chip to the next one on the upper copper layer, with the last output going back to the board's data interface on the back side. The resistors are placed on the right side of their respective driver IC, jumping across the data traces.

The supply lines are placed beneath the row of LEDs. Had I placed the supply on the upper edge of the board, I still would've had to connect all the 16 anodes to V_{DD}. The driver chips, on the other hand, only need one connection to $V_{DD}$and GND each, so by placing the supply lines at the bottom of the board I have less traces running across the board vertically.

From a layout perspective, it would've made sense to put the GND line at the top of the board, since the chips need one connection to that trace each, while the LEDs don't. However, I opted to put both supply lines right above each other to reduce electromagnetic interference. If all the LEDs are switched on suddenly and a lot of current starts to flow, a magnetic field builds up quickly around the supply traces. This could induce bothering or even dangerous currents in the surrounding electronics. By having the two supply lines close to each other, their magnetic fields largely cancel each other out. Another advantage of having the supply at the bottom is that the distance to the delicate chips are a bit farther away.

2.5.2 Space-saving measures

In order for the production to be as cheap as possible as well as to minimise the rotating mass, I tried to make the boards as small as possible. Ultimately, I managed to bring the size down to 73 mm 73 mm