Cost Effective Alternative to Control Lighting Systems Using DMX512

Daniel Carpenter
djcarpenter1@verizon.net
Kevin Edelson
kevinedelson@gmail.com
William LaFosse
wlafosse@comcast.net
Kelly Lampayan
klampayan12@hotmail.com
Alok Saxena
alok.saxena93@yahoo.com

Abstract

Almost all lighting systems use the DMX512 protocol to simplify the use of multiple lights, but DMX control consoles are often quite expensive. We sought to create a more inexpensive DMX control console through the use of the Arduino microcontroller and the Danger Shield, a source of inputs. Using these, we successfully created a lighting system that could execute functions like color chase, color cycle, strobe, dim, and blackout. By comparing this system’s functions and monetary value with other DMX control consoles on the market, we concluded that our lighting system was a suitable substitute for higher end products given its programmable versatility.

1 Introduction

DMX512 is a method used today to control stage equipment such as intelligent lights, color changers, strobes, smoke machines, and confetti dispensers. This method was initially invented in order to interface between dimmers and consoles more efficiently; however, it has proven to be so effective that it is now the standard for controlling such equipment and is used by amateurs and professionals all over the world. Due to the high cost of stage equipment and the boards used to control lighting systems, the implementation of DMX512 can prove to be costly. Current controller boards on the market can cost thousands of dollars, so cheaper alternatives to controlling such equipment, such as the Arduino Duemilanove microcontroller, would benefit those using DMX512. Microcontrollers such as the one mentioned above can be programmed to control numerous lights in order to emulate professional DMX512 consoles. The Arduino microcontroller will be outfitted with inputs like sliders and buttons and then used like a commercial controller in this way. By using DMX512 and the Arduino, we will create a lighting system that will not only be cost effective but will also allow for a greater range of functions and customization than more expensive DMX512 controllers currently on the market.

2 Background

2.1 Lighting Systems

Intelligent or remote controlled LED lighting systems play an important role in both staged events and contribute greatly to their visual stimulation. As a result, lighting systems must be planned and installed with the proper equipment. These lighting systems utilize a large number of various LED lighting fixtures that vary greatly in capabilities such as control, special effects, shapes, and light intensity. Many LED lighting fixtures, such as the Chauvet LED Techno Strobe RGB, contain built-in “Stand Alone” programs that allow them to perform basic functions that do not require control systems. Controlling these lights using “Stand Alone” programs, however, is impractical, especially when large numbers of them are to be used at the same time. Sophisticated lighting control systems are consequently necessary to control all of these lights. For instance, do the lights need to be adjusted individually throughout their use in addition to other lighting fixtures such as moving spotlights, or are the lights only expected to change through several color selections? Since lighting systems vary greatly in complexity, control systems must utilize a standard form of communication with all of the lights. This standard is known as the DMX512 Protocol.
2.2 DMX512

DMX512 is a protocol for transferring information from the console to the stage equipment or light in the context of the project. Communicating with the light involves breaking down data into packets of information; that is, the DMX512 data consists a series of one way digital data packets that address up to 512 separate channels. DMX512 uses these packets of information to send a value in the range of 0 to 255 to each specific channel. The one condensed packet distributes the number value to the correct channel and this value is used by stage equipment to control outputs like the brightness of lights. As each channel executes an output for its own light, the outputs collectively result in a unified, simultaneous effect.

One important application of the system involves using only three channels, red, green, and blue, to create a variety of different colors. To accomplish this, the Arduino sends out a packet of data, which states that the three specified color channels have values while all the other 509 channels are zero. Each channel receives an eight digit byte, which corresponds to any base-ten integer from 0 to 255, that indicates the brightness of the light. For instance, the integers 255, 0, and 128 correspond to maximum, minimum, and intermediate brightness levels, respectively. By simultaneously outputting shades of red, green, and blue, the Arduino projects a color combination to achieve a unique color from the spectrum. One may then vary the brightness value of each channel to change the resultant color. This process manifests exactly how the DMX512 communicates digital information to lights.

2.3 The Arduino Duemilanove

2.3.1 The Arduino Board

Microcontrollers utilize small processors in order to take several inputs and give a variety of outputs. They provide a convenient way for humans to interface with technology. Their versatility, in terms of what can be connected to them, makes them perfect for controlling lights with the DMX protocol. The microcontroller we used for our project is the Arduino Duemilanove.

The Arduino board consists of an ATmega328 microcontroller, a 16 MHz crystal oscillator, a USB connection, a power source, and fourteen digital input/output pins that operate at five volts. Using inputs such as buttons and sensors, the Arduino can then control the lights in our system.

2.3.2 Why the Arduino?

There are a variety of microcontroller platforms, such as Parallax Basic Stamp, Netmedia’s BX-24, and Phidgets, available on the market. All of these microcontrollers, including the Arduino, are inexpensive and cross platform, allowing them to interface easily with all computer operating systems. We chose the Arduino, however, because of its unique open source software, which gives the user access to already developed source codes and schematic information. [1] Due to this, “DMX Simple” functions already exist within the environment, making it possible for the Arduino to communicate with the lights using DMX protocol.

2.3.3 C Programming on the Arduino

Arduino microcontrollers utilize a C-based programming language. In addition to the basic language, the Arduino compiler can also implement a wide variety of functions and libraries, both created with the compiler, and that are
open source. To accept inputs, the Arduino uses functions "analogRead(pin number)" and "digitalRead(pin number)". To give outputs, the Arduino uses the functions "analogWrite(pin number, HIGH/LOW)" and "digitalWrite(pin number, HIGH/LOW)". Being open source, there are a plethora of libraries and functions for specific attachments to the Arduino.

One such library is the "DMX Simple" library, which enables the Arduino to communicate with the light system using the DMX protocol. The function "DmxSimple.write(channel,value)" sets an analog value to a channel on the lighting system. The DMX protocol converts this into a digital signal that the lighting system can then understand and implement. Other functions include "DmxSimple.maxChannel(channels)", which sets those DMX channels to be used in the function to save space, and DmxSimple.usePin(pin number), which sets the pin to be used to output data from the Arduino to the lights. [2]

2.4 Interacting Between Hardware

2.4.1 Using the Danger Shield

The Arduino microcontroller needs to receive an input before it can carry out the functions specified in the code. Shields are boards that are designed to be connected to the Arduino board in order to maximize the board’s ability to acquire data and control different devices. The danger shield is an external board that was developed specifically for Arduino microcontrollers to provide physical inputs through sliders, basic sensors (i.e. temperature sensors), and buttons that are connected to the board’s digital and analog pins. Arduino microcontrollers can be programmed to read these inputs (Figure X of Danger Shield). For instance, when a user presses a button or adjusts a slider, the shield sends an input, such as HIGH, to the Arduino that the programming code responds to by yielding a specified output. By adjusting the sliders or pressing buttons on the Danger Shield to control input values read by the Arduino, users can physically interact with the microcontroller in a manner that would otherwise be impossible.

2.4.2 Using the DMX Protocol

The Arduino microcontroller serves as the primary control console for the lighting system utilized in this project. Once it is programmed using the DMX Simple library, the microcontroller interacts with the LED lights through the DMX Shield. The DMX Shield, like the Danger Shield, can connect directly onto the Arduino’s pins. It has two ports which allow the Arduino to be used as a DMX-Master transmitter and receiver. The DMX data must be sent in a repetitive stream of packets that includes the values for each channel. There must be a break of about 88 s-1s between each data packet. Before the packets can be sent out, it must start with a LOW byte which is followed by 8-bit information that contains a value between 0-255. High Stop bits and Release bits follow the 8-bit information and the cycle begins all over again. Each byte stores a value for a specific channel. The DMX512 protocol controls 512 channels and thus send 512 bytes of information in one stream. The sequence of the bytes determines the channel the information is being sent to. The DMX-shield transfers this data in DMX format out to DMX enabled devices. [3] [See Figure 5]

3 Capabilities of the DMX Lighting System

We have attempted to control a DMX light using the Arduino microcontroller in order to show that the Arduino is capable of implementing the same
functions in the DMX as a professional lighting system. This is where we were able to show the different functions implemented using the Danger Shield. The DMX light includes some default functions that we used for our program.

3.1 Projecting Colors

The DMX has 16 columns of 12 LEDs. Every column is divided into three channels that display a color. The colors displayed are red, green, and blue, or (RGB) which read a value ranging from (0 - 255), or off to on. The reason why the channels are only red, green and blue is because they are the three primary colors. The DMX has to be able to, when told, project other colors besides RGB. So if the color yellow were needed, then the colors Red and Blue would be set to 255 while the color green would be set to 0. If the color white were needed, then the colors RGB would be set to 255. [See Table 1]

<table>
<thead>
<tr>
<th>Red</th>
<th>Green</th>
<th>Blue</th>
<th>Resultant Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>255</td>
<td>0</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>255</td>
<td>255</td>
<td>255</td>
<td>White</td>
</tr>
<tr>
<td>255</td>
<td>255</td>
<td>0</td>
<td>Yellow</td>
</tr>
<tr>
<td>255</td>
<td>165</td>
<td>0</td>
<td>Orange</td>
</tr>
<tr>
<td>160</td>
<td>32</td>
<td>240</td>
<td>Purple</td>
</tr>
<tr>
<td>255</td>
<td>130</td>
<td>171</td>
<td>Pink</td>
</tr>
</tbody>
</table>

Table 1: Some examples of how the main colors red, blue, and green were mixed to produce other colors used in our program.

can you control what colors show in what order, but you can also control how fast and how bright the colors are displayed.

3.2 Implementation of Universal Functions

3.2.1 Slow Function

The slow function allows the user to modify the speed of functions like strobe and color chase. Using analogRead on the middle slider, we take that value and modify it. We then use it in the function "delay(delaytime)", which delays a certain number of milliseconds based on the input value. Delaying different amount of times based on the position of the slider allows the user to change the speed of any function.

3.2.2 Strobe and Dimming Functions

A function that was added onto the program is called strobe. This function uses a button on the Danger Shield, which activates the function. This function is able to use any effect in the DMX while the lights flash on and off. No matter what function the program is running, if strobe mode is on, the lights will start blinking.

One slider on the Danger Shield activates dimming; the amount of dimming is determined by the position of the slider. Both of these functions are easy to implement. Like the DMX lights, it has channels to automate these functions just by doing "DmxSimple.write(DimChannel, dim value)" or "DmxSimple.write(StrobeChannel, strobe speed)". Without these convenient commands, these functions would be otherwise hard to code.

3.2.3 Blackout

Blackout refers to the immediate turning off of lights, no matter what function the program is
currently in. The program first checks if the blackout button is pressed. If it is, then the program enters a loop in which it continuously turns the lights off via "setcolor(0,0,0)". The program waits until a button is pressed again, at which points it exits the blackout function and continues with the program.

4 Results and Discussion

We observed that our Arduino based DMX Lighting System was able to perform the basic lighting effects: color cycle, color chase, strobe, dimming, and blackout. With its numerous sliders and buttons, the system provides a convenient interface with which users can physically manipulate lighting effects. In addition, the system proves versatile because we may add more functions to our code to increase the light’s performance. The total cost of the lighting console equipment, including the Arduino, Danger Shield, and DMX Shield, came out to $63.00.

To assess our lighting console, we compared it to others currently sold in the market. [See Table 2] Although the Arduino console seemed relatively cheap, it turned out to be more expensive than some commercially sold consoles such as the Chauvet Obey 3. Unlike these consoles, however, the Arduino console is programmable—we may add functions in the code to create more advanced lighting effects. Thus, cheap non-programmable lighting consoles such as the Obey 3 cannot perform anything beyond basic functions and prove far less versatile than the Arduino console we constructed.

Although the Arduino console consists of hobbyist electronic hardware, it can potentially achieve the same effects as more expensive consoles at a much cheaper price. The Behringer Eurolight provides advanced functions that our Arduino console does not, such as storable scenes and audio filter capabilities. However, by using the Arduino’s programming capabilities, we can write functions in our code to replicate the effects of a system that is over $100 more expensive than ours.

We observed that the Arduino console cannot fully replicate the functions of the $3000 Elation Show Designer, a programmable console that provides extensive lighting functions to the user. This, however, is expected due to the price of the product. Our system still proves effective in completing its task while using inexpensive components.

<table>
<thead>
<tr>
<th>DMX Lighting Console</th>
<th>Market Price</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chauvet Obey 3 Light Controller</td>
<td>$90.00</td>
<td>Basic Light Effects</td>
</tr>
<tr>
<td>Arduino, Dangermove, Danger Shield, and DMX Shield</td>
<td>$53.00</td>
<td>Basic Light Effects, Programmable</td>
</tr>
<tr>
<td>Behringer LC2412 Eurolight Pro 24-CH DMX Lighting Console</td>
<td>$165.00</td>
<td>24 Channel Lights, Storable Scenes, Sound to Light Effects</td>
</tr>
<tr>
<td>Elation Show Designer</td>
<td>$3000.00</td>
<td>Moving Light Fixtures, Programmable, Extensive Lighting Functions</td>
</tr>
</tbody>
</table>

Table 2: The green color represents the Arduino lighting console while the light gray color represents lighting consoles sold on the market.

5 Conclusions

We concluded that the Arduino console demonstrates a both cheap and versatile method of controlling DMX512 Lighting Systems. The results demonstrate that the Arduino console is able to accept incoming information from the user interface and convert it into the DMX512 Protocol format. Unlike commercial consoles already dedicated towards specific functions, the Arduino console requires producers to write computer programs to implement in the lighting system. Despite the difficulties that these computer programs introduce, they allow us to replicate expensive lighting consoles from cheap, hobbyist computer equipment.

Our ability to program the Arduino console opens many possibilities for future research. For example, we can use audio filters to have the lights correspond with rhythmical patterns and pitch variations in music. The console may also be programmed to control motors and movable light fixtures, thus enhancing its stage lighting capabilities. Individuals may also program the console to respond to the variety of inputs on the Danger Shield such as sound, light, and temperature sensors. There certainly is a more cost effective and flexible alternative to the store-bought norm DMX light control system.

6 Acknowledgements

We would first like to thank Alex Weiner, student of Rutgers, and Mark Sproul, Manager of Engineering Computing Services at Rutgers, for...
their time spent teaching and helping us with the research project. We would also like to thank RTA Fedja Buzancic for his guidance with the project. Thanks goes to Blase E. Ur and Levi Schmidt, along with the other counselors, for help in writing our paper and preparing our presentation. Lastly, thank you to all of the sponsors of Governor’s School who gave us this opportunity to do this research.

References

