Programmable Household LED Light Fixture

A Senior Project

presented to

the Faculty of the Electrical Engineering Department

California Polytechnic State University, San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Science

by

Jordan Locano

June, 2011

Table of Contents

List of Figures

List of Tables

Acknowledgements

I want to start off by thanking all the professors I have had over these 5 years who taught me the foundations needed for this project.

I would like to also thank Texas Instruments for offering compensation for part of the projects cost and also for the free samples that were used in this project. Their products were very useful in this project and their documentation was wonderful.

Finally, a big thanks to my boss Dustin Lanphar and co-worker Nick Wernicke who showed me the ins and outs of Altium which allowed for the design and layout of the custom PCB. Also a big thanks to Adrian Ott for programming the CNC gantry mill for the structure which made this all possible.

Abstract

This report details the development and construction of a LED light fixture to be used for household lighting. This document details background information, design ideas, project specifications, production, assembly, testing, and conclusions involved with this project. The goals of this project is to design and build a cost-effective replacement for traditional room lighting that can perform better and last longer than traditional methods.

Introduction

Home lighting is a huge unexplored field for LEDs. LEDs offer so many possibilities over traditional lights that they set to become the next big lighting technology. The trouble with using LEDs for lighting is that they can be unwieldy in large numbers. The goal of this project is to prove that control over a large number of LEDs is possible. The typical household light runs off 120 vac, so there will need to be a conversion to the LEDs <5v DC requirement. LEDs are a smart choice because they are safer than traditional lights, last longer, and are more efficient therefore saving money in the long run. This project goes over designing an LED multiplexing control board from the ground up using a popular industry software suite.

Background

While looking for a light fixture for my current house, I came to the conclusion that it would be possible to build a comparable light fixture. In building my own fixture, I was able to customize my design to allow for additional features not found in normal fixtures. The use of LEDs in light fixtures can only increase as they become cheaper to produce because of their attractive properties. LEDs consume far less energy per unit of light than any other incandescent or compact fluorescent bulb(CFL) available. They also last at least 50,000 hours, compared to 10,000 hours of traditional bulbs. There is also no harmful chemicals in LEDs unlike CFLs, which contain small amounts of mercury.[1] In order to add complex functionality to the fixture, individual control of each LED would have to be implemented. This is possible through multiplexing.

Multiplexing takes advantage of human physiology by cycling through the LEDs fast enough for the eye not to notice. This effect is also known as persistence of vision. The way the LEDs are wired allows for sets of LEDs to be driven for equal intervals while appearing to be all on. As a byproduct of multiplexing, each group of LEDs has time to cool while the other groups are being driven. This also reduces power consumption by greatly cutting the on time for every LED. Multiplexing is also very useful in allowing for shared data busses, where multiple devices need to communicate using a single data line.

Requirements

The main goal of this project is to produce a superior replacement to traditional room lighting. To achieve this the LEDs will need to provide at least the same amount of light as standard light bulbs while avoiding the intensity of LEDs. In order to be hung from any ceiling angle, the mounting hardware needs to be flexible. As with any consumer product, the interaction between user and product must be easy to learn and intuitive. Similar to a typical lamp, this fixture is powered from a standard 120vac outlet. Continuous maximum current draw tops out at 2 amps protected by a fuse, a reverse current protection diode, and a ferrite bead.

Design

LEDs

The first stage in the design was to choose the number of LEDs used in this project. Using Matlab, a visual map of the fixture shown in figure 1 was created. The original shape was going to be oval, but to simplify the construction it was changed to a circle. The program takes the design dimensions provided by the user and counts how many LEDs can fit inside the desired diameter. After tweaking the dimensions, a final configuration was chosen.

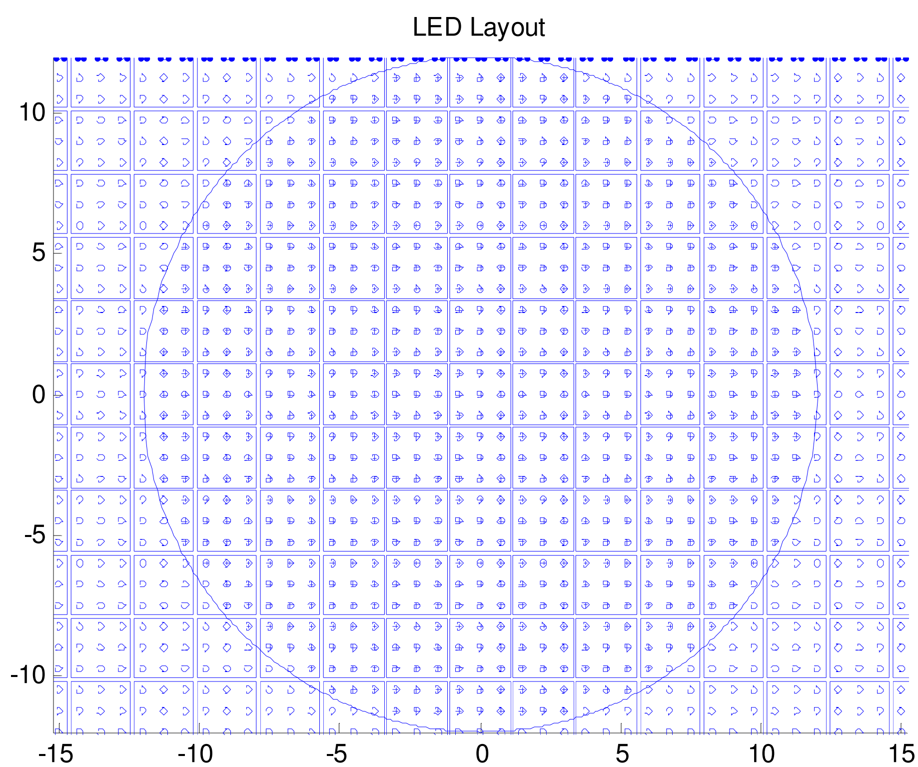


Figure 1: Matlab LED layout printout

Fixture

The physical mounting for the LEDs went through 2 main design stages. The first design was a complex structure composed of interlocking metal ribs with wood blocks in between to locate and support the LEDs. The design in figure 2 had to be abandoned due to the complexity of the design and the inability to get machine time to cut all the aluminum. The second design shown in figure 3 was adopted due to the simplicity of the parts that needed to be machined. The LEDs would be suspended by bare copper wire terminated by ring terminals.

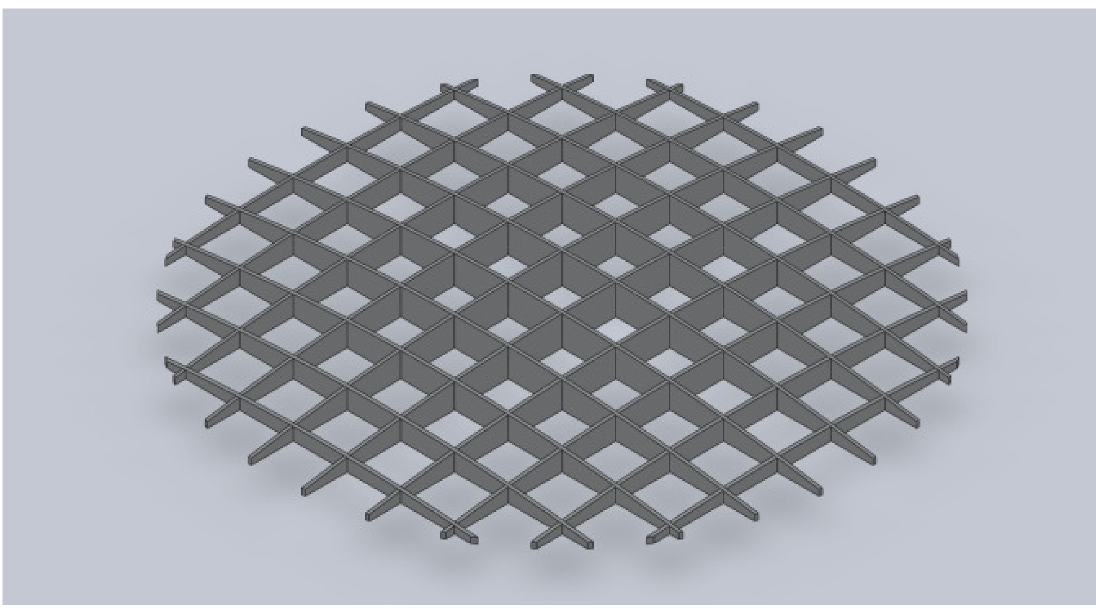


Figure 2: First desired fixture design

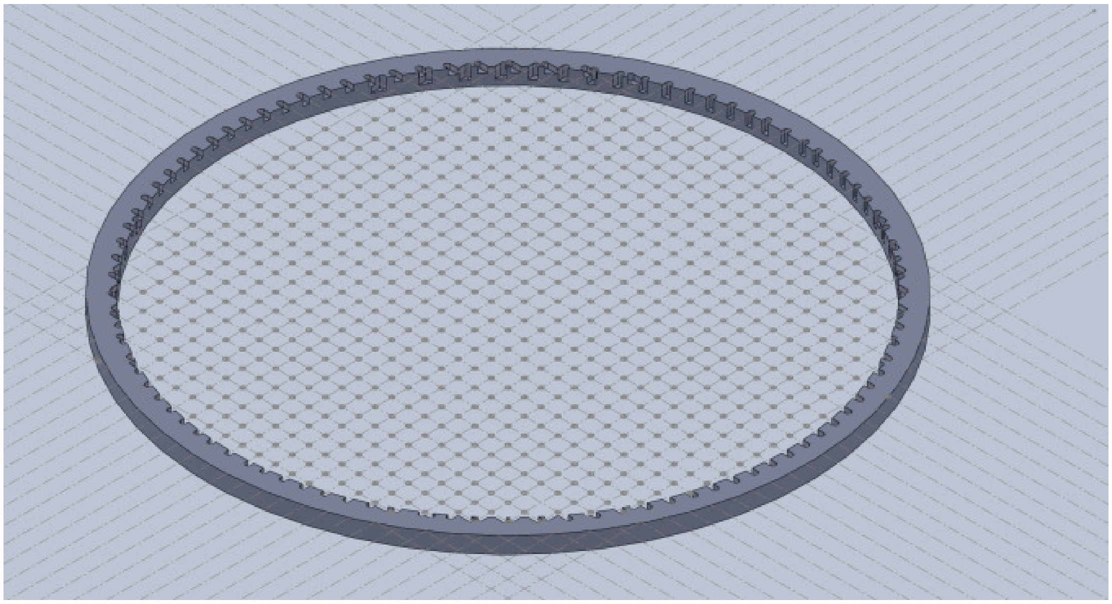


Figure 3: Final fixture design

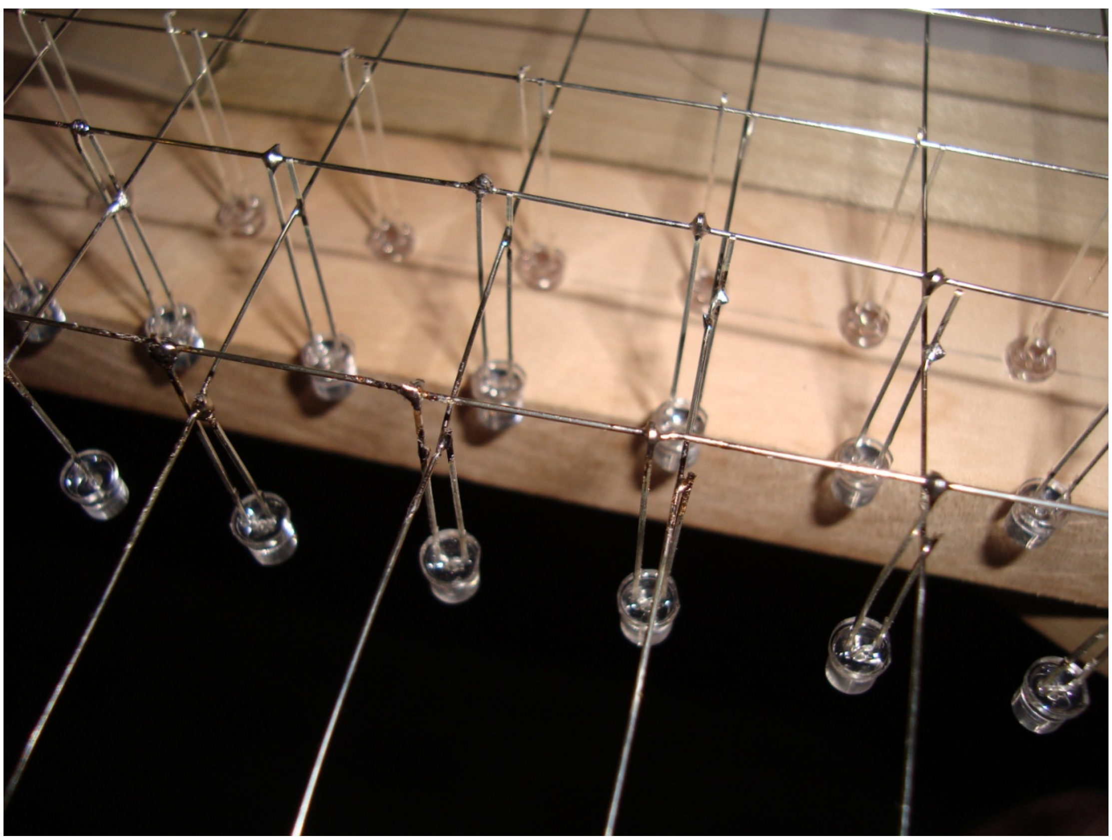


Figure 4: LED mounting on rows and columns

As shown in figure 4, the LEDs were arranged in such a way to allow multiplexing without a large mess of wires. The anode is connected to the row wires and the cathode is connected to the column wires.

PCB

For my project I decided to make an add-on board for the STK600 currently used in some CPE329 classes. I wanted to avoid making my own microcontroller board due to conflicting information on how to set up the clock and other various external components. Since the maximum number of LEDs in a row is 29, I chose a 2 amp fuse to protect the board. This allows for a maximum current of 68 milliamps per LED, much higher than the 20 milliamp continuous rating. Because the LEDs are multiplexed however, we can run the LEDs at a higher pulse current. The main idea behind this design is the ability to control so many LEDs with so few data lines. The bulk of this reduction is taken care of by the TI CD74HC154M. This demux takes 4 input lines and turns it into 16 output lines, with only 1 possible active line at a time. This gives us a hardware method of insuring only 1 row is on at a time. I then use PFETS as switches to source the LEDs. The other neat chip is the ST LED driver STP16DP05TTR. This chip uses the SPI bus on the AT mega and can be daisy-chained for an even larger number of LEDs. With only 29 LEDs we only needed to use 2 of these chips, since they can control 16 each. The TI digital potentiometer is used to set the brightness of the LED driver chip. Instead of messing with software PWM for brightness, we can change the resistance value on the fly to get different brightness levels. The final IC is the TI current shunt monitor. This allows for real time measurement of power consumption, which is useful for debugging and logging usage for statistics.

Code

I made several look-up tables in excel to aid in the programming of the device. These tables also alerted me to a problem with the initial 777 LEDs, which led to the omission of the far rows, leading to an updated LED count of 741. Table A is the output current of the LED driver corresponding to the resistance it sees. Table B is the resistance of the digital pot when the hex value is written to its inputs. Table C shows the corresponding hex values to each row.

Rext (Ω)	Output current (mA)
976	20
780	25
652	30
560	35
488	40
433	45
389	50
354	55
325	60
300	65
278	70
259	75
241	80
229	85
215	90

Table A: Output Current to Resistance

DECIMAL	FEDCBA	RF	FEDCBA	
31	00011111	1196	1	F
30	00011110	1142	1	E
29	00011101	1089	1	D
28	00011100	1036	1	C
27	00011011	984	1	B
26	00011010	933	1	A
25	00011001	883	1	9
24	00011000	835	1	8
23	00010111	787	1	7
22	00010110	742	1	6
21	00010101	697	1	5
20	00010100	655	1	4
19	00010011	614	1	3
18	00010010	575	1	2
17	00010001	537	1	1
16	00010000	502	1	0
15	00001111	468	0	F
14	00001110	436	0	E
13	00001101	406	0	D
12	00001100	377	0	C
11	00001011	351	0	B
10	00001010	325	0	A
9	00001001	302	0	9
8	00001000	280	0	8
7	00000111	259	0	7
6	00000110	239	0	6
5	00000101	221	0	5
4	00000100	205	0	4
3	00000011	189	0	3
2	00000010	174	0	2
1	00000001	161	0	1
0	00000000	148	0	0

Table B: Digital Pot Resistance Values

PORTC		Row
F	0	1
F	1	2
F	2	3
F	3	4
F	4	5
F	5	6
F	6	7
F	7	8
F	8	9
F	9	10
F	A	11
F	B	12
F	C	13
F	D	14
F	E	15
F	F	16
0	0	17
1	0	18
2	0	19
3	0	20
4	0	21
5	0	22
6	0	23
7	0	24
8	0	25
9	0	26
A	0	27
B	0	28
C	0	29
D	0	30
E	0	31
F	0	32

Table C: Row Hex Values

Development and Construction

Frame

The material for the frame was a sheet of birch plywood that we had at work. The choice to use this was easy because it was a very nice looking wood and it was not being used by anyone else. The frame was cut on a 5 axis gantry mill shown in figure 5 below. The vacuum was used to blow the dust off of the spindle to prevent a fire. The total machine time was around 2 hours.

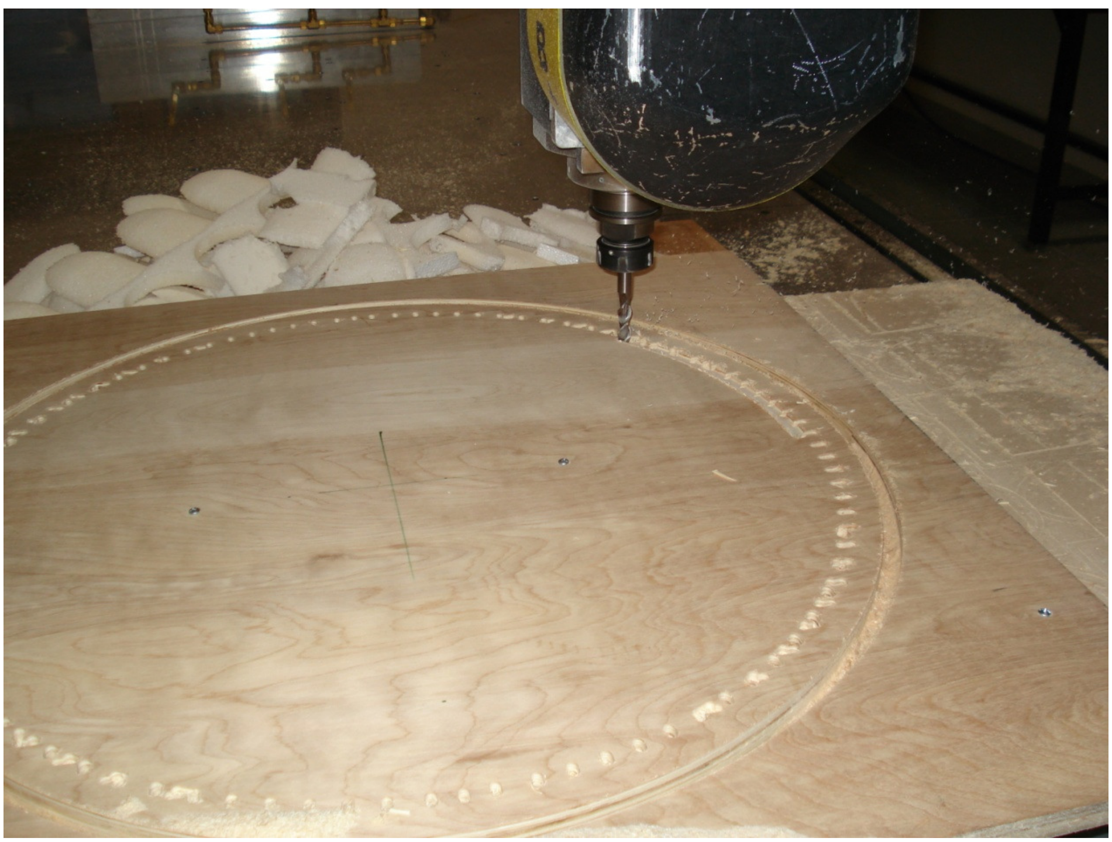


Figure 5: Light fixture Being Machined

LEDs

The LEDs are soldered to crossing copper wires that are connected to the PCB via a 34 pin molex connector. The challenge with the crossing wires is that there could be no shorts with 1482 connections. To accomplish this I hung the LEDs from the anode and soldered them to the intersections. Figure 6 shows how I prepared every LED for mounting.

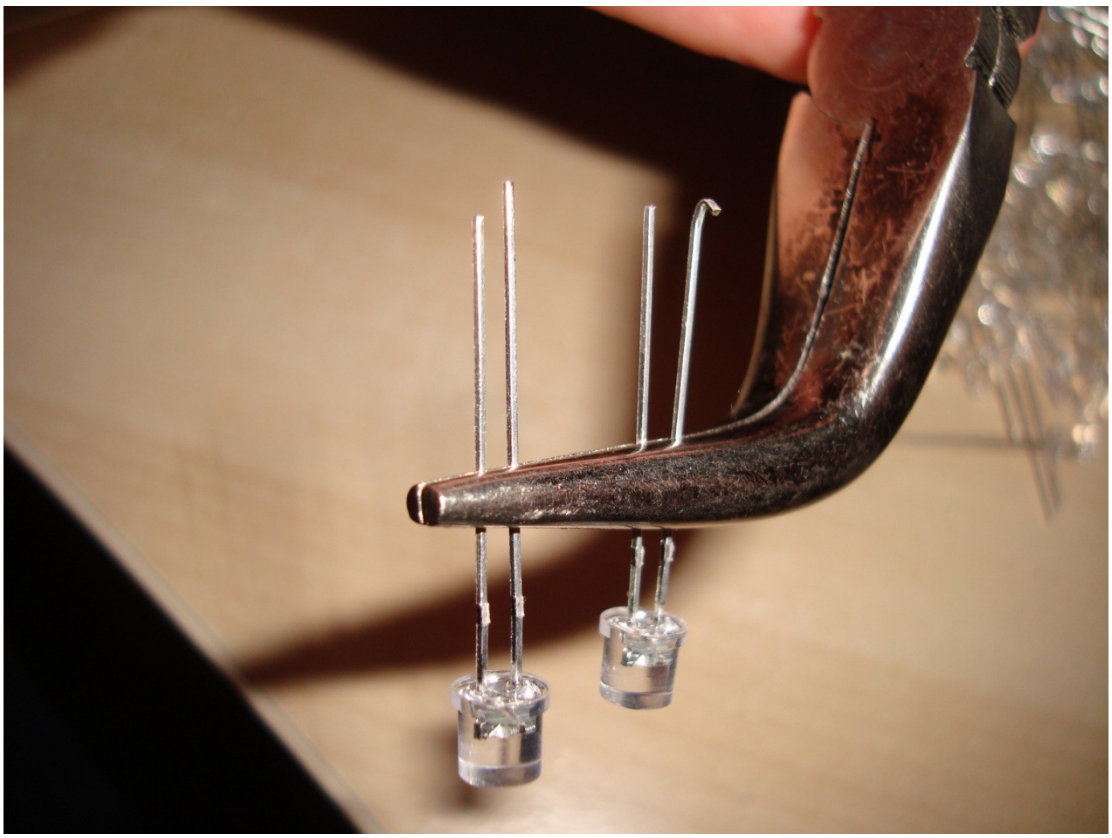


Figure 6: LED Lead Preparation

Board

The PCB that I designed in Altium was a challenge to populate because I had almost zero experience with soldering surface mount parts. Using lots of flux I was able to get all the parts soldered with no problems.

Conclusion

This project was very challenging for a single person to take on. I learned so much that we were not taught in class about manufacturing an electronic product. We are mainly taught theory in our classes, and in labs we build pre-designed circuits. I was forced to research and ask around for guidance on designing a full system. In the end it worked, although there were some problems.

In order to get this project fully working I will need to spend more time debugging it. When the LEDs were being multiplexed with 50 milliamps they appeared to be dimmer than a single row being driven with 20 milliamps, so more time needs to be spent making sure everything is running as desired.

Bibliography

[1] "Comparison Chart LED Lights vs. Incandescent Light Bulbs vs. CFLs." *Design Recycle Inc.*Design Recycle Inc. Web. 11 June 2011. <http://www.designrecycleinc.com/led%20comp%20chart.html>.

[2] International Rectifier. *IRLML6401PbF*. International Rectifier. *International Rectifier*. International Rectifier. Web. 15 Feb. 2011. <http://www.irf.com/product-info/datasheets/data/irlml6401pbf.pdf>.

[3] Texas Instruments. *High-Speed CMOS Logic 4- to 16-Line Decoder/Demultiplexer*. Texas Instruments. *Texas Instruments*. Texas Instruments, 22 June 2004. Web. 15 Feb. 2011. <http://focus.ti.com/lit/ds/symlink/cd74hc154.pdf>.

[4] STMicroelectronics. *Low Voltage 16-bit Constant Current LED Sink Driver with Outputs Error Detection*. STMicroelectronics. *STMicroelectronics*. STMicroelectronics. Web. 15 Feb. 2011. <http://www.st.com/internet/com/TECHNICAL_RESOURCES/ TECHNICAL_LITERATURE/DATASHEET/CD00149163.pdf>.

[5] Texas Instruments. *TANDEM 64-TAP DIGITAL POTENTIOMETER*. Texas Instruments.*Texas Instruments*. Texas Instruments, 17 Aug. 2009. Web. 15 Feb. 2011. <http://focus.ti.com/lit/ds/symlink/tpl8002-25.pdf>.

[6] Texas Instruments. *High-Side Measurement CURRENT SHUNT MONITOR*. Texas Instruments. *Texas Instruments*. Texas Instruments, 11 Nov. 2005. Web. 15 Feb. 2011. <http://focus.ti.com/lit/ds/symlink/ina138.pdf>.

Appendix A - Bill of Materials

	Designator	Quantity	#Column Name ErrorCut Length (inches)	Comment	Description	Lead	Manufacturer	Manufacturer #	Vendor	Vendor #	Price	Ext.Price
1	C1, C2, C3, C4	4		GRM188R71H104KA93D	CAP CER .1UF 50V 10% X7 0603	0	Murata Electronics North America	GRM188R71H104K 93D	Digikey	490-1519-6-ND	0.37	1.48
2	D1	1		BAT 60A E6327	DIODE SCHOTTKY 10V 3 SOD-323	0	Infineon Technologies	BAT 60A E6327	Digikey	BAT60AE6327INC ND	0.58	0.58
3	Expand0, Expand1	2		61082-101402LF	CONN RECEPT 100POS .8M DUAL SMD	0			Digi-Key	609-1681-1-ND	6.83	13.66
4	F1	1		0437002.WR	FUSE 63V 2A FAST SMD 1206	0	Littelfuse Inc	0437002.WR	Digikey	F2885CT-ND	0.81	0.81
5	F2	1		EXC-ML20A390U	BEAD CORE 4A 100 MHZ 0805 SMD		PANASONIC	EXC-ML20A390U	Digikey	P10191CT-ND	0.21	0.21
6	J1, J2, J3	3		1935161	TERM BLOCK PCB 2POS 5.0MM GREEN	0	Phoenix Contact	1935161	Digikey	277-1667-ND	0.30	0.9
7	P1, P2	2		90130-1134	CONN HEADER 34POS .100 STR TIN		MOLEX	90130-1134	DIGIKEY	WM8244-ND		0
8	Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8, Q9, Q10, Q11, Q12, Q13, Q14, Q15, Q16, Q17, Q18, Q19, Q20, Q21, Q22, Q23, Q24, Q25, Q26, Q27, Q28, Q29, Q30, Q31, Q32	32		IRLML6401TRPBF	MOSFET P-CH 12V 4.3A SOT-23	0	International Rectifier	IRLML6401TRPBF	Digikey	IRLML6401PBFCT ND	0.51	16.32
9	R1, R2, R5, R6	4		MNR18E0AP.101	RES ARRAY 100OHM 15TERM 8RES SMD	0	Rohm	MNR18E0APJ101	Digikey	MNR18101CT-ND	0.081	0.324
10	R3, R4, R7, R8	4		741X163103JP	RES ARRAY 10KOHM 16TERM 8RES SMD	0	CTS Corporation Resistor/Electrocomponents	741X163103JP	Digikey	741X163103JPCT ND	0.158	0.632
11	R10	1		WSLK-.05CT-ND	RES .05 OHM 2W 1% 2816 SMD	0	Vishay/Dale	WSL2816R0500FEA	Digikey	WSLK-.05CT-ND	1.66	1.66
12	R11	1		CSR0603FKR200	RES 0.2 OHM 1/8W 1% 0603 SMD	0	Yageo	CSR0603FKR200	Digikey	CSR0603FKR200C ND	0.56	0.56
13	U1, U2	2		CD74HC154M	IC 4-TO-16 DECODER/DEMUX 24-SOIC	0	Texas Instruments	CD74HC154M	Digikey	296-9182-5-ND	0.82	1.64
14	U3, U4	2		STP16DP05TTR	IC LED DRIVER LINEAR 24 TSSOP		ST Microelectronics	STP16DP05TTR	Digikey	497-5976-1-ND	3.45	6.9
15	U5	1		INA138NA/250	IC CURRENT SHUNT MON SOT-23-5	0	Texas Instruments	INA138NA/250	Digikey	INA138NACT-ND	2.63	2.63
16	U6	1		TPL8002-25PWR	IC POT DGTL TANDEM 64TAP 15TSSOP	0	Texas Instruments	TPL8002-25PWR	Digikey	296-25791-1-ND	3.90	3.9

Fixture

Birch Plywood-3'x3'

5 Eye hooks

1 Carabiner

1/16" Braided steel cable-6'

1/4"x3/4" Aluminum angle stock-5'

Double sided tape

Tinned copper wire-200'

Bare ring terminals #4 stud-200

3/8” Wood screws-200

User Interface

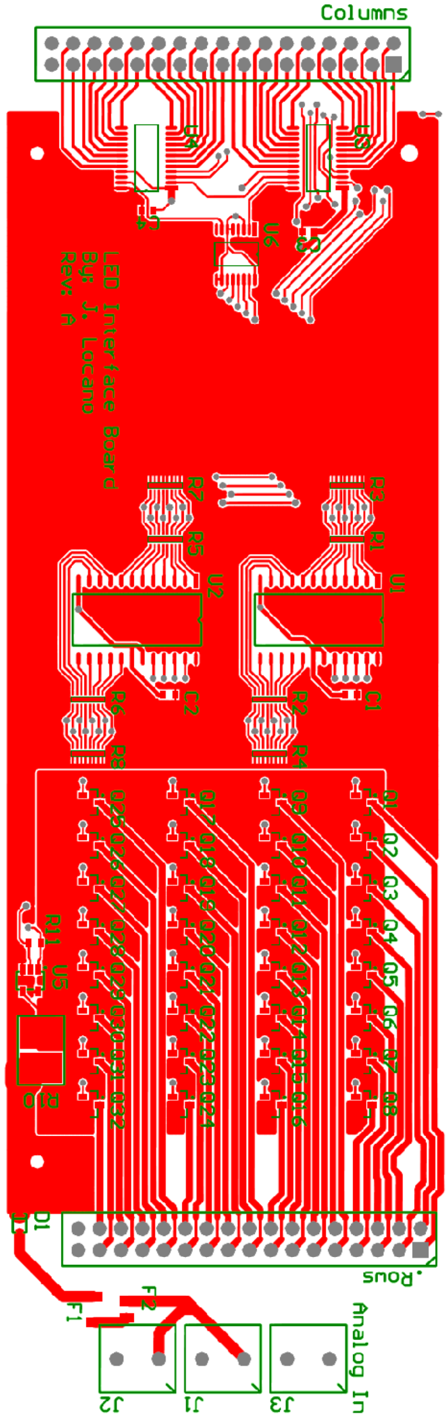
7 Conductor sprinkler wire-25'

Rotary switch with knob

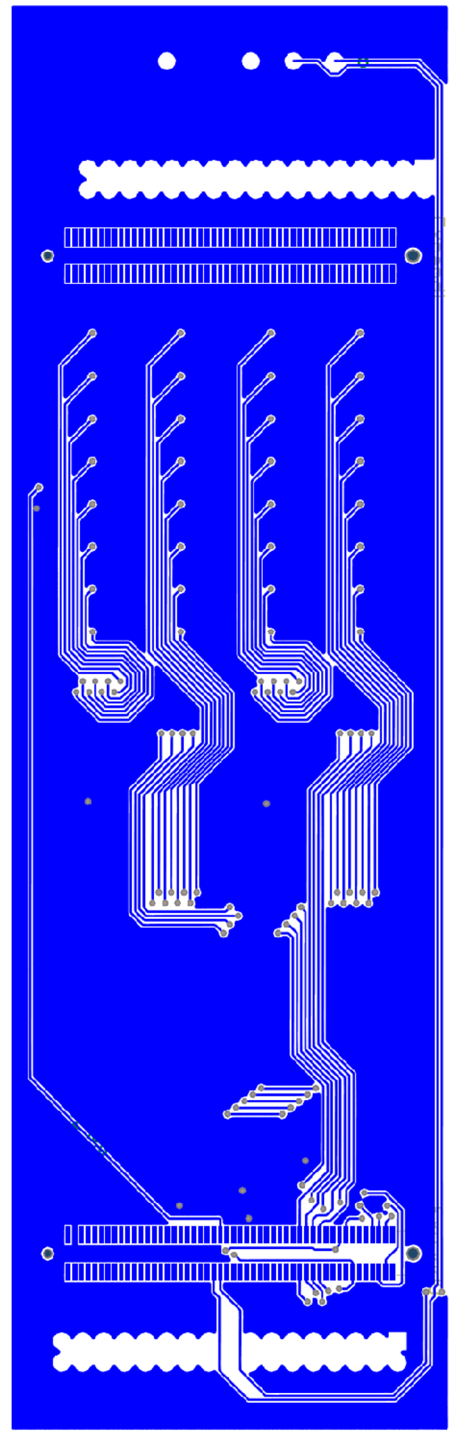
Linear potentiometer with knob

5v 2A AC/DC wall adaptor

Appendix B - PCB Layout

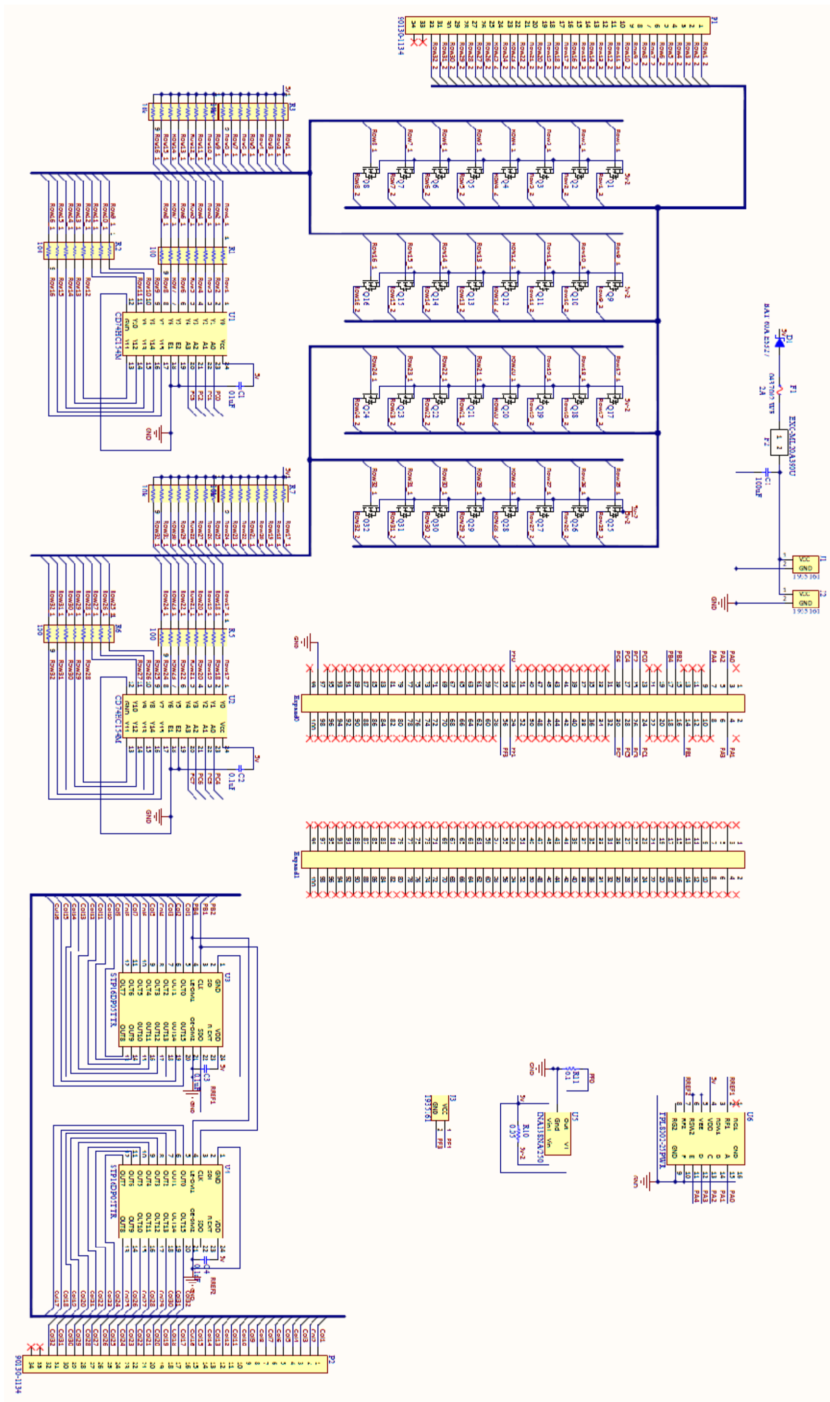


Top



Bottom

Appendix C - Schematic



Appendix D - Matlab Code

```
%Author: Jordan Locano
%Purpose: Visual aid for LED layout and automated counting
r = 1/8;
distbw = .75;%.75 Distance between LEDs
cubewid = 2.125;%2.125 Width of cube(not important for final config)
hcubewid = cubewid/2;%Not important for final config
len = 12;%13/12  Radius in horizontal dir(was used for oval)
wid = 12;%9/12  Radius in vertical dir(was used for oval)
count = 0;

for m=-22.5:distbw:22.5
    for n=-18:distbw:18
        x=[(m-hcubewid) (m+hcubewid)];
        y=[(n-hcubewid) (n+hcubewid)];
        if m<5000
            hold on;
        end
        if (((abs(m)+1/8+1/16)/len)^2+((abs(n)+1/8+1/16)/wid)^2)<=1
            count = count + 1;
            line([m m],[n-0.1 n+0.1])
            line([(m+0.1) (m-0.1)],[n n])
        end
        if mod(n,distbw*3)==0
            if mod(m,distbw*3)==0
            line([(m-hcubewid) (m-hcubewid)],y)
            line([(m+hcubewid) (m+hcubewid)],y)
            line(x,[(n-hcubewid) (n-hcubewid)])
            line(x,[(n+hcubewid) (n+hcubewid)])
            end
        end
        x = m-(1/8):0.01:m+(1/8);
        a=m;
        b=n;
        y1= b + (a + r - x).^(1/2).*(r - a + x).^(1/2);
        y2= b - (a + r - x).^(1/2).*(r - a + x).^(1/2);
        plot(x,y1);
        plot(x,y2);
    end
end
axis([-12 12 -12 12])
axis equal

count=count+1-1

x = -len:0.1:len;
y1= (wid*(len + x).^(1/2).*(len - x).^(1/2))/len;
y2= -(wid*(len + x).^(1/2).*(len - x).^(1/2))/len;
plot(x,y1);
plot(x,y2);
```

Appendix E - AVR Code

```
#include <stdint.h>
#include <avr/io.h>
#include <stdlib.h>
#include <avr/interrupt.h>
#define F_CPU 16006912
//#define F_CPU 3686400
#include <util/delay.h>

#define INPUT 0x00
#define OUTPUT 0xFF
#define ALLON 0x00
#define ALLOFF 0xFF

//#define SPI_SCK    7
//#define SPI_MOSI   5
#define SPI_SCK    1
#define SPI_MOSI   2
#define SPI_SS     4

#define DIG_POT_A 0
#define DIG_POT_B 1
#define DIG_POT_C 2
#define DIG_POT_D 3
#define DIG_POT_E 4

#define I_MEAS 0
#define UI_1 1
#define UI_2 3

void startup(void);
void RES_Write_val(uint8_t val);
void SPI_MasterTransmit(uint8_t data);
void SPI_xmit_long_word(uint8_t one,uint8_t two,uint8_t
three,uint8_t four);

static uint8_t shift[] =
{0xFF,0x7F,0x3F,0x1F,0x8F,0xC7,0xE3,0xF1,0xF8,0xFC,0xFE,0xFF};
static uint8_t row[] =
{0xF1,0xF2,0xF3,0xF4,0xF5,0xF6,0xF7,0xF8,0xF9,0xFA,0xFB,
0xFC,0xFD,0xFE, 0xFF,
                                    0x00, 0x10, 0x20, 0x30, 0x40,
0x50, 0x60, 0x70, 0x80,0x90, 0xA0, 0xB0, 0xC0, 0xD0};
static uint8_t letters[26][7] =
{{0x07,0x09,0x11,0x11,0x1F,0x11,0x11},//A0
{0x1E,0x11,0x11,0x1E,0x11,0x11,0x1E},//B1
{0x0E,0x11,0x10,0x10,0x10,0x11,0x0E},//C2
{0x1E,0x11,0x11,0x11,0x11,0x11,0x1E},//D3
{0x1F,0x10,0x10,0x1E,0x10,0x10,0x1F},//E4
{0x1F,0x10,0x10,0x1E,0x10,0x10,0x10},//F5
{0x0E,0x11,0x10,0x10,0x13,0x11,0x0E},//G6
```

```
{0x11,0x11,0x11,0x1F,0x11,0x11,0x11},//H7
{0x1F,0x04,0x04,0x04,0x04,0x04,0x1F},//I8
{0x0F,0x01,0x01,0x01,0x01,0x11,0x0E},//J9
{0x11,0x12,0x14,0x18,0x14,0x12,0x11},//K10
{0x10,0x10,0x10,0x10,0x10,0x10,0x1F},//L11
{0x11,0x1B,0x1F,0x15,0x15,0x11,0x11},//M12
{0x11,0x11,0x19,0x15,0x13,0x11,0x11},//N13
{0x0E,0x11,0x11,0x11,0x11,0x11,0x0E},//O14
{0x1E,0x11,0x11,0x1E,0x10,0x10,0x10},//P15
{0x0E,0x11,0x11,0x11,0x11,0x0E,0x01},//Q16
{0x1E,0x11,0x11,0x1E,0x14,0x12,0x11},//R17
{0x0E,0x11,0x10,0x0E,0x01,0x11,0x0E},//S18
{0x1F,0x04,0x04,0x04,0x04,0x04,0x04},//T19
{0x11,0x11,0x11,0x11,0x11,0x11,0x0E},//U20
{0x11,0x11,0x11,0x11,0x11,0x0C,0x04},//V21
{0x11,0x11,0x11,0x11,0x15,0x15,0x0A},//W22
{0x11,0x11,0x0A,0x04,0x0A,0x11,0x11},//X23
{0x11,0x11,0x11,0x0F,0x01,0x11,0x0E},//Y24
{0x1F,0x01,0x02,0x04,0x08,0x10,0x1F}};//Z25
static uint8_t numbers[10][7] =
{{0x04,0x0C,0x04,0x04,0x04,0x04,0x0E},//1
{0x0E,0x11,0x01,0x02,0x04,0x05,0x1F},//2
{0x0E,0x11,0x01,0x07,0x01,0x11,0x0E},//3
{0x11,0x11,0x11,0x1F,0x01,0x01,0x01},//4
{0x1F,0x10,0x10,0x1F,0x01,0x01,0x1F},//5
{0x0E,0x11,0x10,0x1E,0x11,0x11,0x0E},//6
{0x0F,0x11,0x01,0x02,0x04,0x04,0x04},//7
{0x0E,0x11,0x11,0x1F,0x11,0x11,0x0E},//8
{0x0E,0x11,0x11,0x0F,0x01,0x11,0x0E},//9
{0x0E,0x11,0x11,0x11,0x11,0x11,0x0E}};//0

uint32_t frame;

int main(void)
{
      startup();

    uint32_t refreshNum = 500;
    uint8_t i=0;
      uint8_t j=0;
      uint32_t k=0;

      for(;;)
    {
            for(j=0;j<33;j++)
        {
                  for(i=0;i<30;i++)
                  {
                  PORTC=row[i];

                  k = 0;
            while (k != refreshNum)
            {
                k++;
```

```
                if(j<5)
                {
                    SPI_xmit_long_word(shift[j+3],0xFF,0xFF,0xFF);
                }
                else if(j>4&&j<9)
                {
                    SPI_xmit_long_word(shift[j+3],shift[j-
5],0xFF,0xFF);
                }
                else if(j>8&&j<13)
                {
                    SPI_xmit_long_word(0xFF,shift[j-5],0xFF,0xFF);
                }
                else if(j>12&&j<17)
                {
                    SPI_xmit_long_word(0xFF,shift[j-5],shift[j-
13],0xFF);
                }
                        else if(j>16&&j<21)
                {
                    SPI_xmit_long_word(0xFF,0xFF,shift[j-13],0xFF);
                }
                else if(j>20&&j<25)
                {
                    SPI_xmit_long_word(0xFF,0xFF,shift[j-
13],shift[j-21]);
                }
                else if(j>24&&j<33)
                {
                    SPI_xmit_long_word(0xFF,0xFF,0xFF,shift[j-21]);
                }
            }
        }
            }
    }
      return 0;
}


void startup(void)
{
      DDRA = 1<<DIG_POT_A | 1<<DIG_POT_B | 1<<DIG_POT_C |
1<<DIG_POT_D | 1<<DIG_POT_E;
      RES_Write_val(0x0C); //1196ohm

      DDRB = 1<<SPI_MOSI | 1<<SPI_SCK | 1<<SPI_SS;
      SPCR = 0x51;
      SPSR = 0x00;
    SPI_xmit_long_word(0x00,0x00,0x00,0x00);

      DDRC = OUTPUT;
    PORTC = 0xFF;

      DDRF = ~_BV(I_MEAS) | ~_BV(UI_1) | ~_BV(UI_2);
```

```
}

void RES_Write_val(uint8_t val)
{
      PORTA = val;
}

void SPI_MasterTransmit(uint8_t data)
{
    SPDR = ~data;
    while (!(SPSR & (1<<SPIF)));
}

void SPI_xmit_long_word(uint8_t one,uint8_t two,uint8_t
three,uint8_t four)
{
    SPI_MasterTransmit(one);
    SPI_MasterTransmit(two);
    SPI_MasterTransmit(three);
    SPI_MasterTransmit(four);
    PORTB |= _BV(SPI_SS);
      PORTB &= ~(_BV(SPI_SS));
}
```

Appendix F - Row Control FET Datasheet[2]

Datasheet Available at Referenced Website

Appendix G - Row Demultiplexer Datasheet[3]

Datasheet Available at Referenced Website

Appendix H - LED Sink Driver Datasheet[4]

Datasheet Available at Referenced Website

Appendix I - Digital Potentiometer Datasheet[5]

Datasheet Available at Referenced Website

Appendix J - Current Shunt Monitor Datasheet[6]

Datasheet Available at Referenced Website