UVC LED Antifouling System 2020

AUGUST 27

AUGUST 27

JASMIN CASAUAY

Background

Biofouling, the accumulation of unwanted growth on submerged structures, can lead to many functional and financial issues. The most common method to prevent fouling on ship hulls is through the application of coatings.[1] The Folger Pinnacle platform has different scientific instruments, including a high definition camera. A live video feed can be seen from 23m below sea level in a rockfish conservation zone off the West Coast of Vancouver Island.[2]

Because of the depth of the instrument platform, prolific growth develops on the camera. Divers would periodically service and clean the camera and the instruments on the platform. Organism growth can sometimes obstruct the camera and prevent the on-board sensors from outputting accurate data. Biofouling accumulates quicker than we can address them. Divers or a scrub team is recruited regularly to address the serious biofouling formation on the Folger Pinnacle instrument platform. However, dive operations can be expensive.

The application of ultraviolet light (UVC) has been successful in utilizing the prevention of biofouling in seawater systems and on instruments and sensors.[3] Within the UV spectrum, the UVC wavelength (100nm to 280nm) breaks the chemical bonds between DNA and RNA polymers within microorganisms and therefore preventing the formation of biofouling. It is found that the UV light peak wavelength for antifouling was proven to be most effective at 250nm to 260 nm.

AML developed an antifouling system using ultraviolet lights. In October 2013, AML partnered with Ocean Networks Canada to deploy their UV LED Antifouling instrumentation on one of the platforms on the Neptune observatory. They tested their system in the Folger Pinnacle site because of the abundant biofouling formation due to its location at 25m depth. More information about this study is found on the Resources page.

[1] https://www.tandfonline.com/doi/full/10.1080/08927014.2019.1642334
[2] https://www.oceannetworks.ca/sights-sounds/video/live-video/folger-pinnacle-reefcam
[3] https://www.tandfonline.com/doi/full/10.1080/08927014.2019.1642334

Objective

To design a new UV Antifouling System that would run for one year, preferably two years, to be installed on the Folger Pinnacle instrument platform.

Introduction

There is already an existing UV Antifouling system currently installed on the Folger Pinnacle instrument platform. The initial idea was to upgrade and improve the existing system. However, because of insufficient documentation and information known about the existing system, Chad, Dirk, and Bjorn decided on designing a new, less complicated, and well documented UV Antifouling System.

Any documentation relating to the old UV antifouling system is found in google drive (Co-op Projects > UV LED Folger). Any documentation in this folder may not be accurate and may not necessarily provide the correct information about the existing UV antifouling system.

The newly designed UV Antifouling System is smaller and less complicated. Each enclosure has two PCBs: (1) the UV LED heatsink PCB; and (2) the main controller PCB. An average UVC LED has a limited life span of only 3000 hours. This is not enough to run for a whole year. The solution to this is installing four (4) UV LEDs in each enclosure. One LED will be on 25% at a time. By doing this, the overall life span of the UV lights in the enclosure will be 4 times longer. The duty cycle can be changed through the code by programming the microcontroller.

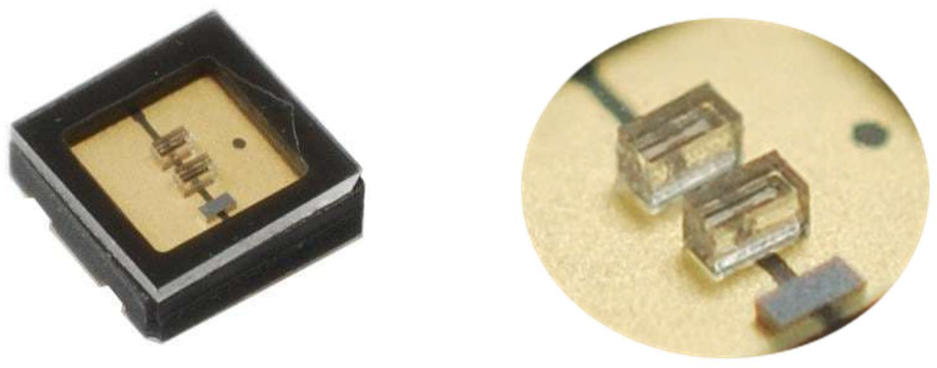


Figure 1: UV LED used in each enclosure (SML-LXF3535UVCC10)

UV LED Heatsink PCB

This PCB is designed to dissipate the heat generated by the UV LED. According to the UV LED datasheet, the absolute maximum operating temperature is $-30°C\ to + 60°C$.

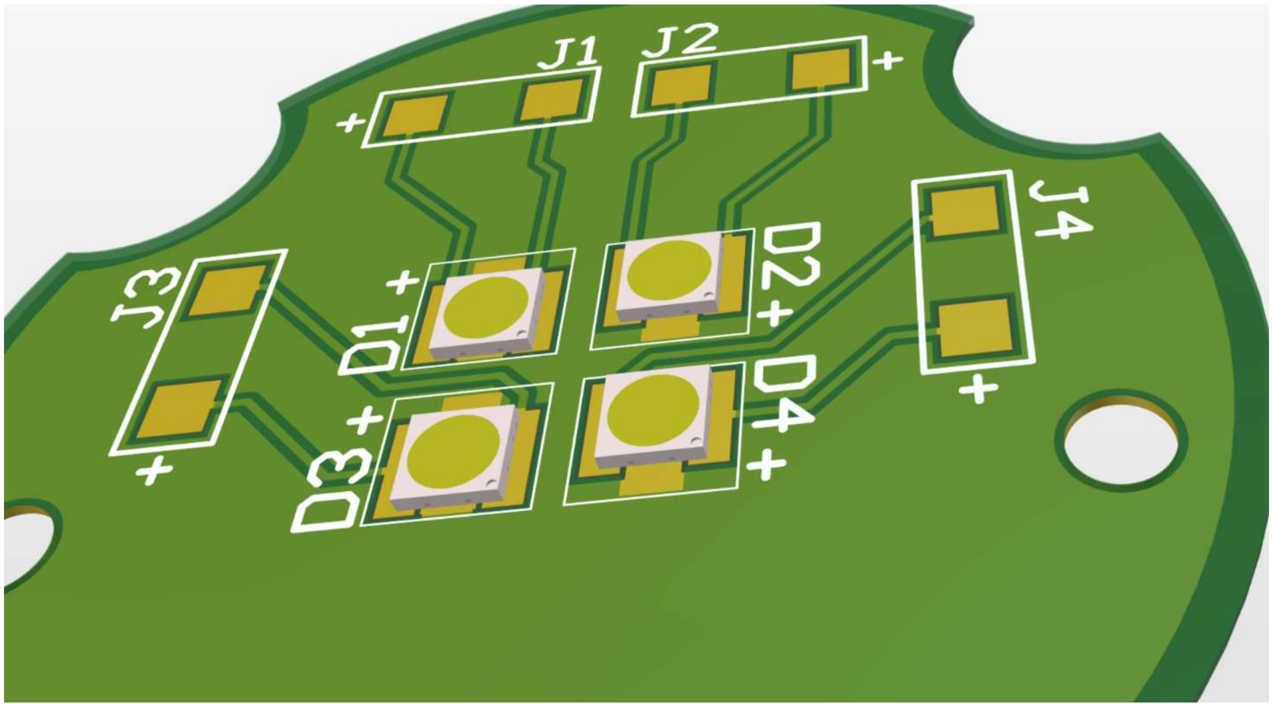


Figure 2: UV LED PCB Design

The only component on this PCB is the surface mount UV LED. Aside from the cathode and anode pads on the LED, there is also a thermal pad that will be soldered on to the PCB to help with the heat dissipation. As seen in Figure 2, the cathode and anode of an LED with the designator Dx is connected to the corresponding cathode and anode pad on Jx. The idea is to solder wires to the pads on J1, J2, J3, and J4 and have them go through the two notches on the side of the PC. This then will be attached to an 8-pin female connector (487526-7 – Digikey)

Main Controller PCB

The main PCB is $70mm\ x\ 40mm$. This board contains a microcontroller, a transceiver chip, 5V converter, 9V converter, and an LED driver for each of the four UV LED in each controller.

The microcontroller used in this board is an ATtiny441. ATtiny841 works as well. The ATtiny441 has 12 I/O pins. Four pins are assigned to the 4 LED drivers. The other Rx and Tx pins (pin 11 and 12 respectively) are assigned to the RS485 transceiver for communication. At this point of the deployment, communication is not necessary. I haven't done any programming to test the transceiver for communication as it is not needed for the initial deployment. Sergio came up with the idea to put the RS485 transceiver chip on the board for future use. For programming, I used the Arduino IDE to program the ATtiny441.

The 5V converter converts the 48V input to 5V for the microcontroller at the R485 transceiver chip. The 9V converter steps down 48V input to 9V for the LED drivers. I will explain how the LED driver works and the problems I have encountered that led me to step down the input voltage to 9V for the LED driver. For reference, all the datasheet for all the components I used on this board is compiled in the Datasheets folder in google drive (Co-op Projects > UV LED Folger > UV Antifouling System v2 > Electrical > Datasheets).

Problems Encountered

The LED Driver I chose can take 3.5 to 60 supply voltage. The other option was to use one LED driver that has 4 channels for all 4 UV LED. The problem with that is if the LED driver malfunctions, all 4 UV LED will fail. Also, the 4 Channel LED driver that I found only takes 5 to 28V supply. I wanted to avoid an additional unnecessary voltage conversion. The datasheet for the 4 Channel LED Driver is also in google drive.

Initially, the idea was to input 48V onto the LED Drive since, according to the datasheet, it can handle up to 60V. However, the load is only dropping 6V while running on a current of 150mA. The rest of the remaining 42V at 150mA is being dropped at the LED Driver. Upon measuring the temperature, the LED Driver dissipated up to 100°C of heat with one minute of the LED being turned on.

$$6.3W\ Power\ dissipated = 42V * 150mA$$

Because of this, instead of feeding the LED Driver the entire 48V, we decided to step it down to 9V. **The LED turns off when the input voltage of the driver goes below 7.8V.** This is why we converted it down to 9V. After leaving the LED turned on for 5 minutes each, the maximum temperature measured was 79°C.

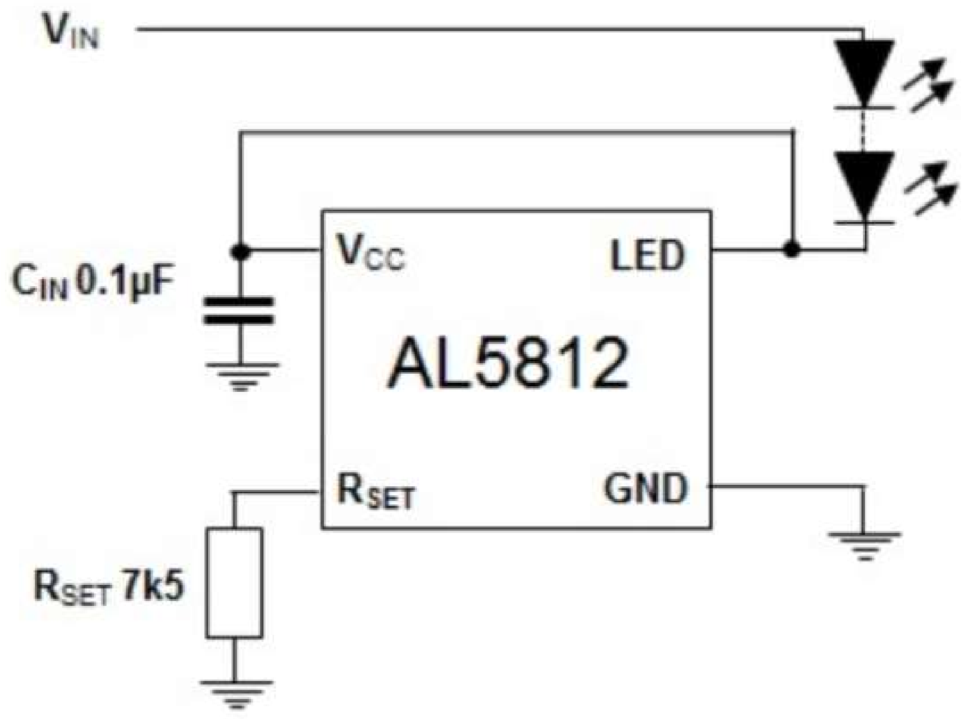


Figure 3: Typical application

The output current coming out of the LED pin can be set by choosing the appropriate value of the R_{SET}. The UV LED is rated at 6V 100mA. We need the current from the LED pin to be 150mA. The formula to find the appropriate R_{SET} value to use is found in the datasheet.

$$I_{LED} = 1500 * \frac{0.5}{R_{SET}}$$ where 1500 is the current ratio between the LED pin current and R_{SET} pin current.

The value of R_{SET} on the board is 7.5kΩ, setting I_{LED} to 100mA.

Programming

Wiring

I used and Arduino Nano and the Arduino IDE to program the ATtiny441 microcontroller. The main PCB has a 6-pin programming header to make the programming of the microcontroller easy and accessible.

1. Upload the Arduino as ISP example code to the Arduino Nano (File > Examples > Arduino as ISP)
2. Once the code is successfully uploaded, wire the Arduino Nano to the ATtiny441. Use the ICSP header on the Arduino Nano to program the ATtiny441. Wire the MISO pin on the Arduino Nano to the MOSI pin of the ATtiny441. Do the same for the rest of the pins on the ISCP header. As for the RESET pin on the ATtiny441, wire it to pin D10 on the Arduino Nano.
3. Place a $10uF$ capacitor between the RST and GND pin on the Arduino Nano. Place a $0.1uF$ capacitor in between the VCC and GND pin of the ATtiny441.

Install the ATTinyCore library from this website and follow the installation process.

https://github.com/SpenceKonde/ATTinyCore/blob/master/Installation.md

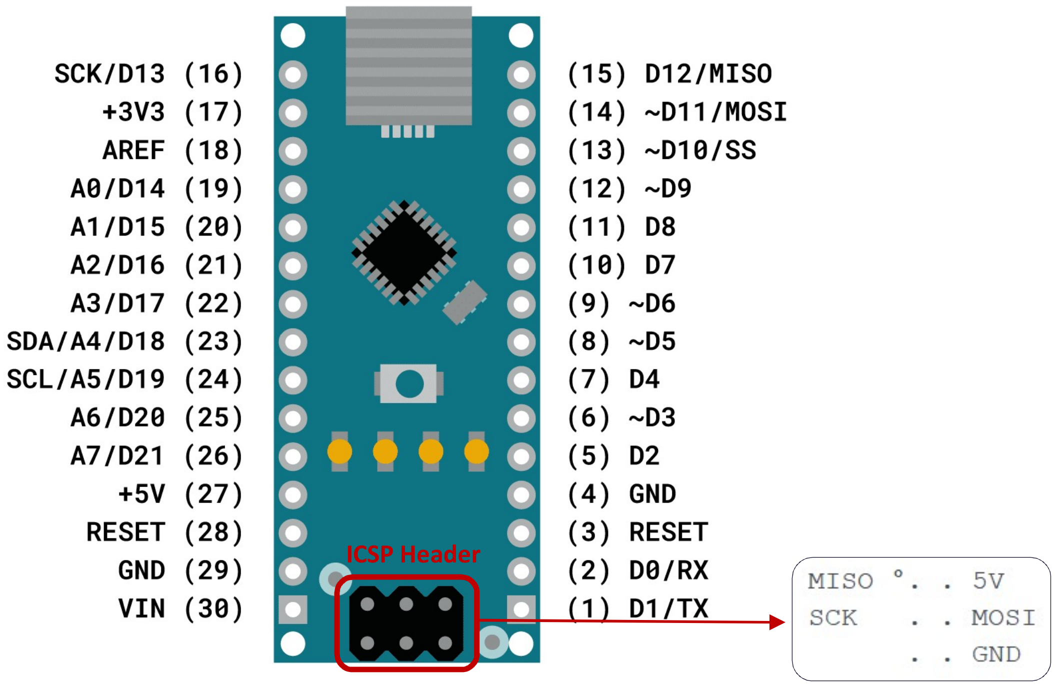
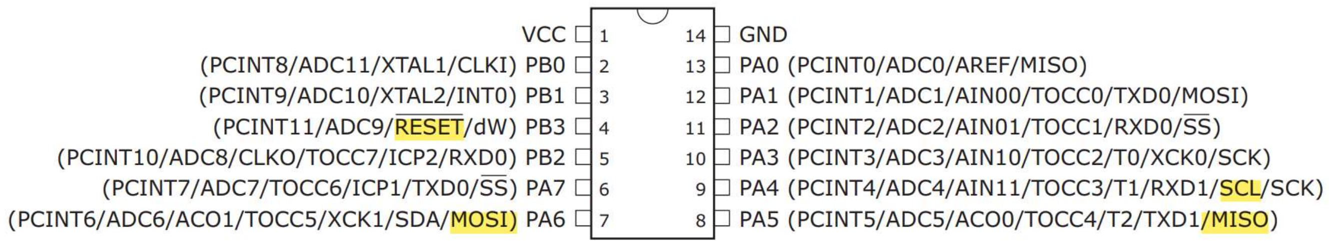


Figure 4: Arduino Nano Pinout


VCC 1
(PCINT8/ADC11/XTAL1/CLKI) PB0 2
(PCINT9/ADC10/XTAL2/INT0) PB1 3
(PCINT11/ADC9/RESET/dW) PB3 4
(PCINT10/ADC8/CLKO/TOCC7/ICP2/RXD0) PB2 5
(PCINT7/ADC7/TOCC6/ICP1/TXD0/SS) PA7 6
(PCINT6/ADC6/ACO1/TOCC5/XCK1/SDA/MOSI) PA6 7
14 GND
13 PA0 (PCINT0/ADC0/AREF/MISO)
12 PA1 (PCINT1/ADC1/AIN00/TOCC0/TXD0/MOSI)
11 PA2 (PCINT2/ADC2/AIN01/TOCC1/RXD0/SS)
10 PA3 (PCINT3/ADC3/AIN10/TOCC2/T0/XCK0/SCK)
9 PA4 (PCINT4/ADC4/AIN11/TOCC3/T1/RXD1/SCL/SCK)
8 PA5 (PCINT5/ADC5/ACO0/TOCC4/T2/TXD1/MISO)
Figure 5: ATtiny441 Pinout

I just did the simplest code for this project because essentially, this project is simply turning the LEDs ON and OFF at a set duty cycle. I used the *delay()* function to turn the LEDs ON and OFF.

I set the specific I/O pins as OUTPUT bit by bit.

Pin 10	LED 1	PA3	PORT A BIT 3
Pin 9	LED 2	PA4	PORT A BIT 4
Pin 8	LED 3	PA5	PORT A BIT 5
Pin 7	LED 4	PA6	PORT A BIT 6

The LED Driver inverts the logic level. So if the output pin is set HIGH (1) the LED turns OFF, if the output pin is LOW (0) the LED turns ON.

```
DDRA
    OUTPUT = Set to 1
    INOUT = Set to 0
PORTA
    IF set to OUTPUT,
        THEN HIGH = Set to 1; LOW = Set to 0
    IF set to INPUT,
        THEN Enable Pull Up Resistor = Set to 1; Disable Pull Up Resistor = Set to 0
```

The code can be found in the Test_v2 folder in google drive (Co-op Projects > UV LED Folger > UV Antifouling System v2 > Electrical > Code > Test_v2)

Code

```
int timeON = 60000;   // ON time in minutes
// Time between switching OFF one light and switching ON the next light in ms (milliseconds)
int timeOFF = 50;

/* Note 1
  LED Driver inverts the pwm signal.
  HIGH (1) = OFF
  LOW (0) = ON
  UV LED requires atlease 7.8V input in the LED Driver to light up the LED
*/

void setup(){
  // Set I/O pins A3, A4, A5, and A6 as OUTPUT
  DDRA = B01111000;
  // Set OUTPUT pins as LOW by setting it to 1 (See Note 1)
  PORTA = B01111000;
}
void loop(){

  // TURN LED 1 ON
  PORTA = B01110000;
  delay(timeON);
  // TURN LED 1 OFF
  PORTA = B01111000;
  delay(timeOFF);

  // TURN LED 2 ON
  PORTA = B01101000;
  delay(timeON);
  // TURN LED 2 OFF
  PORTA = B01111000;
  delay(timeOFF);

  // TURN LED 3 ON
  PORTA = B01011000;
  delay(timeON);
  // TURN LED 3 OFF
  PORTA = B01111000;
  delay(timeOFF);

  // TURN LED 4 ON
  PORTA = B00111000;
  delay(timeON);
  // TURN LED 4 OFF
  PORTA = B01111000;
  delay(timeOFF);
}
```

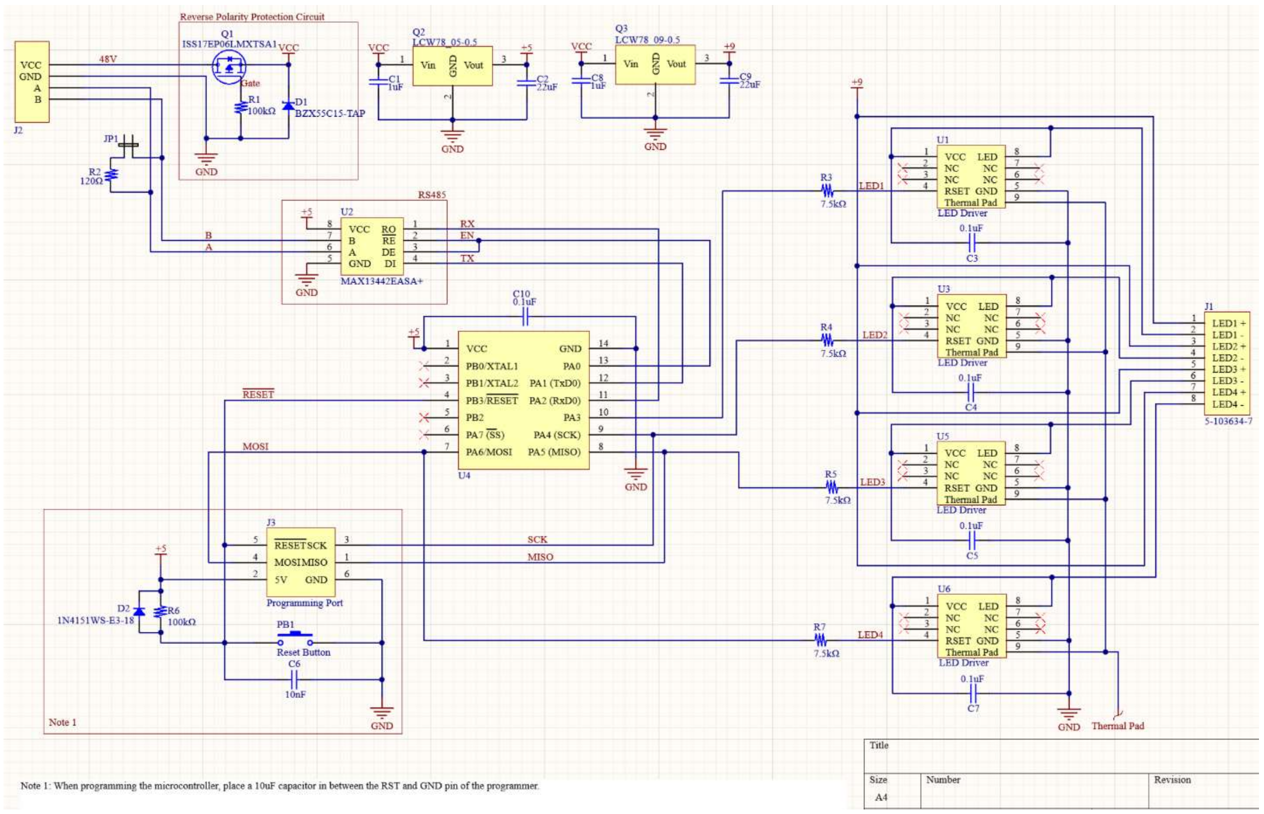


Figure 6: Schematic Diagram for main PCB

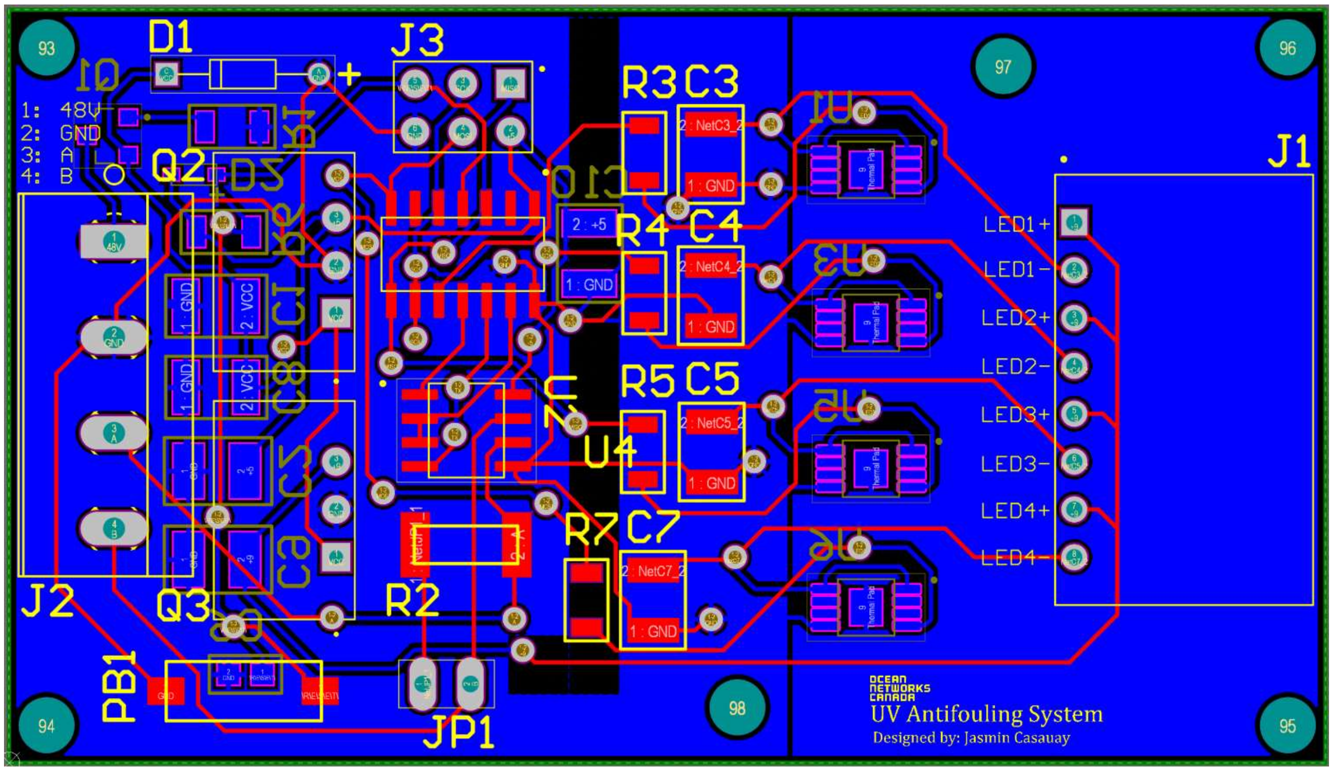


Figure 7: PCB Design for main PCB

Resources:

AML Study

https://amloceanographic.com/case-studies/foul-free-folger-pinnacle/

ONC Folger Pinnacle Site

https://www.oceannetworks.ca/aml-oceanographic-harnesses-uv-light-biofouling-control-0

Programming the ATtiny441

https://www.youtube.com/watch?v=gXXdoeu7yWw