# CONGRATULATIONS

Thank you for purchasing the RGB Multi-MCU base and driver board from SuperTech-IT and TheLEDCube.com

In this document, MCU means Microcontroller such as the PIC32, ATmega328P , prototype boards with such chips on them, or any other Microcontroller.

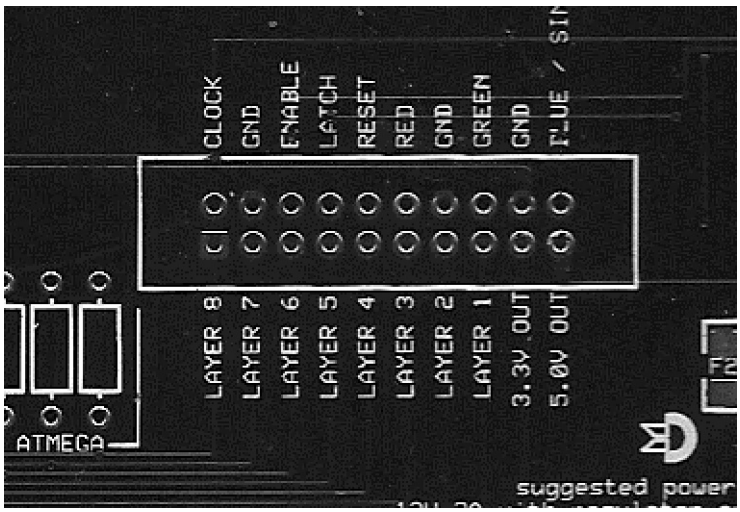
Assembly of the board itself is self-explanatory as almost all the parts are of a single common value.

All resistors are 1K, all non-electrolytic capacitors are 0.1uF and all transistors are BD136.

All other parts are marked as to their description and/or value.

The MCU board 20 pin header is 10 X 2 pins female. All other configuration or signal pins are male and are to be made by breaking off the appropriate number of pins from your 40 pin header strips.

All the signals to connect your Arduino UNO, ChipKit UNO32, Arduino MEGA2560 or other MCU / prototyping board are clearly marked.




LAYER X – Active LOW inputs to select which layer is active. These are usually only active one at a time and cycled quickly to give the illusion that they are all on when viewing the cube. Although you CAN write your code allowing more than 1 layer to be active at once, it is not recommended as this will cause a current increase that may blow the fuse on the 5V line.

Once the data you want is present on the Red, Green and Blue pins (or one of them in a single data stream such as Blue in our Arduino code) the data will be shifted into the register when the CLOCK line goes high. You must return the clock to low before you present your next bit of data.

The LATCH line transfers the data you streamed into the chips to the output latches. This data does not appear on the output pins until the ENABLE line is active though. LATCH is active high, and ENABLE is active low. This means that as long as ENABLE is high, there is no data on the outputs to the LEDs.

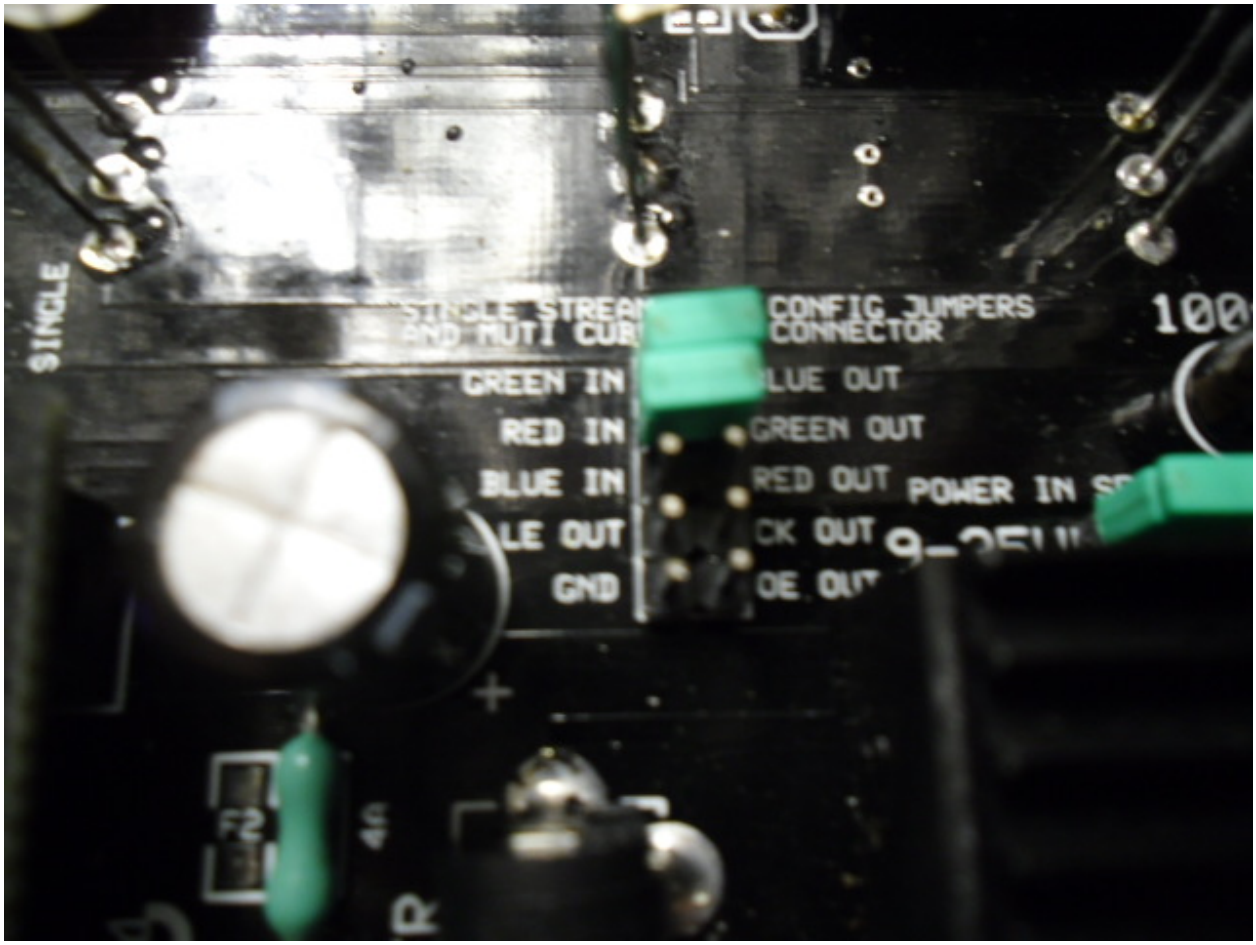
For more information on the operation of an individual DM13A shift register / constant current LED Driver IC, see http://bg.e-neon.ru/user_img/catalog_datasheets/dm13a-a.003.pdf

The 3.3V and 5V outputs are to power the connected MCU. For example, the Arduino UNO can be powered by connecting its 5V line to this output, and the ChipKit UNO can be powered from the 3.3V output. You can also use the 5V (NOT the 3.3V) as a power input link to power the board – however this is not recommended.

The Red, Green and Blue inputs are the serial data stream for each colour.

The bits are loaded into the shift registers and advanced to the next bit when a CLOCK pulse is sent.

If you wish to use a single data stream, use the pin of whatever colour you are sending the data for LAST, and link that colour's output signal to the input of the colour you send second, and link that colour's output to the input of the colour you send first in the data stream. So for instance, our Arduino code sends the red first, then the green, then the blue. This means our data goes into the blue, the blue out is linked to the green in, and the green out is linked to the red in. These signals are strategically located beside each other for easy configuration with simple jumper caps.




## *YOU MUST jumper the top 2 sets of pins as pictured above to use the Arduino UNO code in our firmware (step 7)!!!*

**Also you can see F2 here and may notice that there's 2 spots for F1 and F2, one through hole, one surface mount. This is so we can use EITHER depending on what's more readily available.**

**Simply put in which ever fuses came with your parts kit, or if you are ordering your parts locally, just use whatever is cheaper or more easily available. The additional pads also give you a great set of test points to check your fuse with a meter if you are having power issues.**

If you want to use a 9.5 V to 35V power supply instead of a 5V power supply, you will need the KIM-055L regulator module easily found on eBay. I do recommend at LEAST 10V when using the regulator though.


Make sure you jumper the board to the 9-35V setting (marked 10-35V ob V3 Rev4 and above boards) if using more than 5V OR YOU WILL DAMAGE THE BOARD AND /OR MCU. (The new boards are marked 10-35 rather than 9-35 because tests proved that the board would malfunction at 9V)


When putting in the transistors, please ensure the base is AWAY from the cube (closest to the edge of the board). **For the transistors we use, this means the BACK of the transistors face towards the reset button.**

NOTE: if you want the onboard reset button to work when not using an UNO eliminator, you need to connect the RESET on the 20 pin header to the RESET line of your UNO board. Please be aware that when you do this, you may not be able to program the board while it's connected.

If you are using a BRIDGE BOARD and you want the onboard reset button to reset your uno/mega/chipkit then you must jumper the RESET LINK on the bridge board.

If you find your board is failing to program when you go to upload code to it, disconnect the reset line until you are done programming.

If you connect the 5V line to your Arduino or Chipkit, and you are using a desktop computer, you should actually be able to run the cube powered by the USB power from your prototyping board.

The reverse is always true in that you can power your proto boards from the 5V out on the base / driver board if you power the board from the power jack (recommended) .
The ChipKit UNO32 may also be powered by interconnecting the 3.3V on the proto-board and the base/driver header.

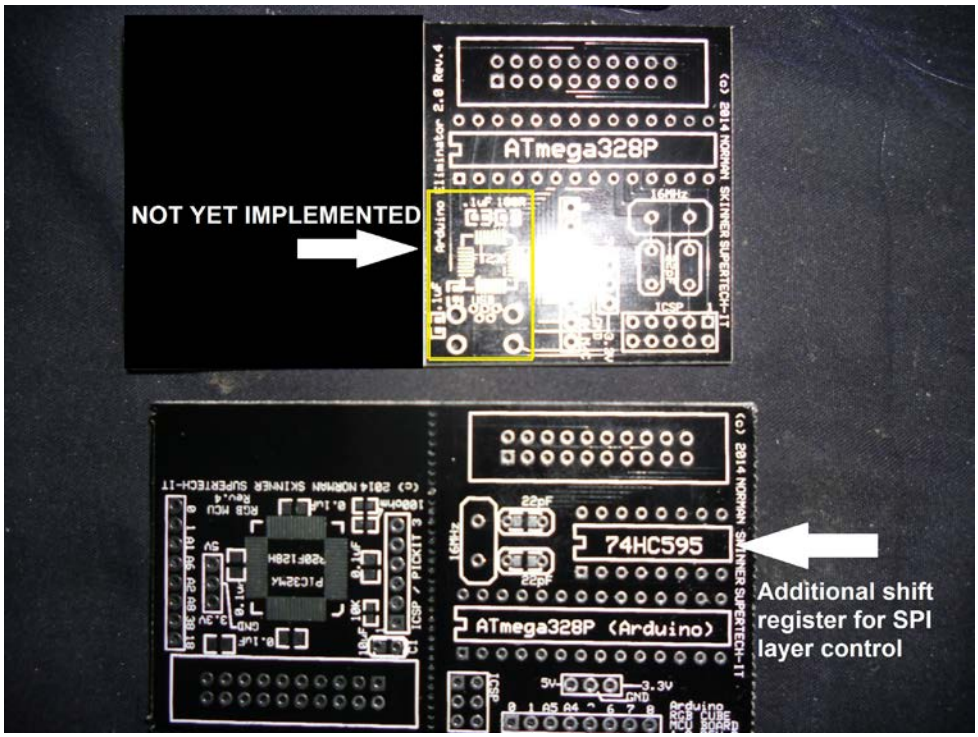
If the 3.3V LED does not light, but the 5V LED does:

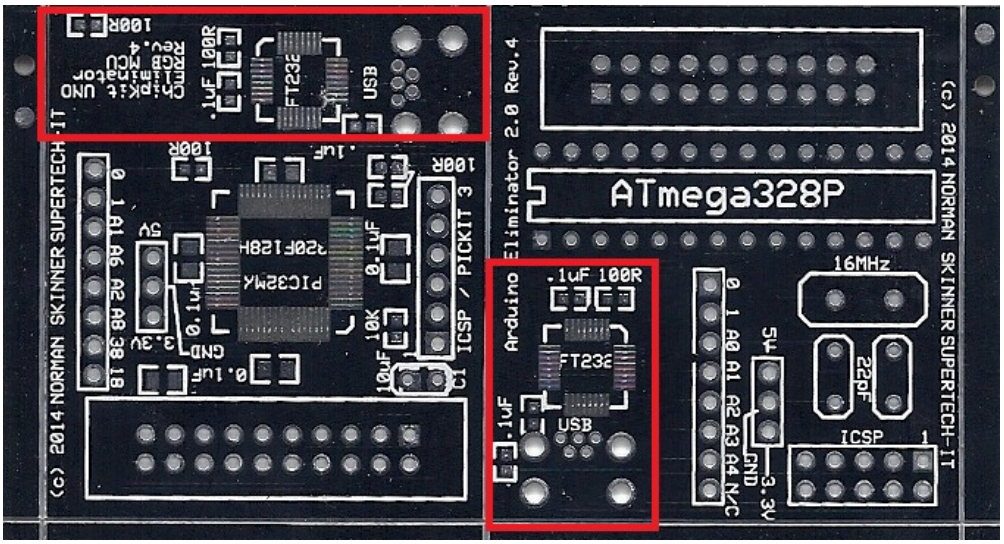
Check that the 3.3V fuse is not blown. This is the 1/2A fuse by measuring the voltage at both ends of the 3.3V fuse (F1). If there is 3.3V at one end, but not the other, replace the fuse. If there is no 3.3V at either end, but the 5V LED is on, check the 3.3V 1117 regulator. If there is no 3.3V and the 5V LED is not lit, make sure you have a jumper on the LED EN pins near the LEDs (V4 R3.12 and up). Make sure your voltage select jumper is in place and making good contact. No power flows anywhere without this jumper being in place. Also check the 5V fuse (F2), although if this fuse blows, it will not affect the 3.3V. If all the voltages look good, but one or both of the LEDs are not lit make sure the LED(s) orientation is correct (not in backwards).

Please note that the beta USB function on the non-SPI Arduino UNO eliminator has not yet been implemented. Do not install the parts in the yellow area at all. The eliminator still works as a non-SPI layer eliminator. This means if you are using the Arduino standalone code, you can transfer the chip after it's programmed in your UNO R3 directly to the eliminator board for use. You can also program the chip on the board directly using an ICSP programmer like the USBTiny or USBasp. If you want to use the slightly faster SPI layer code then use the Arduino eliminator with the additional shift register.
With the ChipKit UNO eliminator, you will need to be able to solder the processor to the board which has a 0.5mm pin pitch. You will require a PicKit3 or compatible programmer to transfer code to the board. One method I use is to use the MPIDE working with my chipkit attached by wires to the cube until I have my code the way I want it, and then I will use MPLAB to read my ChipKit board, and then write the code I just read to the ChipKit UNO32 Eliminator board to run the cube without wires – freeing up my ChipKit UNO32 for other projects. You can get these processors as free samples from Microchip.com You will want the PIC32MX320F128H-801PT.

MicroChip allows up to 3 samples of 2 different types. Since the eliminators work with this project with the 128K and the 64K chips (PIC32MX320F64H-801PT), I usually get 3 of each when I place my sample orders.

The 10uF capacitor on this board **<u>MUST BE CERAMIC</u>** or the microprocessor might not reset right on power up.
These are readily available in SMT 0603 and 0805 sizes, and either will fit the eliminator.






For the newer separate snap apart eliminators that you may have ordered in addition to your V4 or higher boards, please do not install any of the components in the red area. Because of a problem with the FTDI chips, this function is not implemented. Also please note you MUST use the non-SPI layer code, as the additional shift register has been removed from the Arduino eliminator.

If you require more information, please contact me at SuperTech@TheLEDCube.com

Your input is appreciated, and often the information you request becomes part of this document in the future.