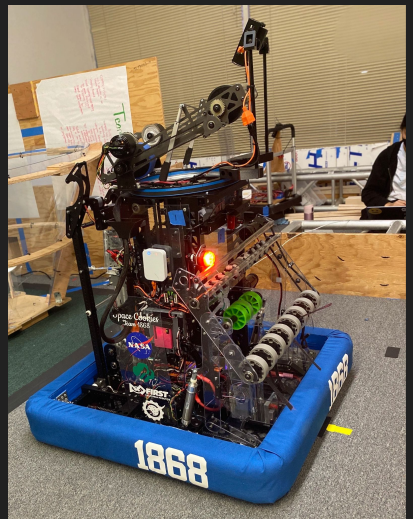
Developing an Open-Source LED Controller for FRC

By Conor Kelly Gerakos
3/7/2026
SCaLE TNG 23x

What is FRC

- First Robotics Competition
- International high-school robotics competition in which students design, build, and compete with a robot
- Robots are large, have lots of motors and moving parts, and have more EMI than most applications



What is the problem?

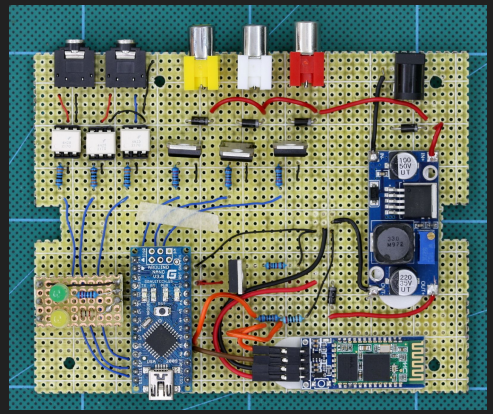
- Need for a new discrete LED controller for FRC
  - LEDs make your robot look cool, and allow for feedback to the driver
- Something that requires minimal communications bandwidth and minimal code running on the RoboRIO (main processor for the robot)
- Preferably open source and affordable with reconfigurable firmware

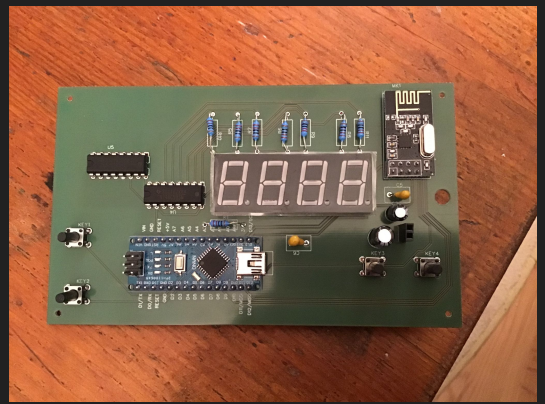
Prior LED Solutions

- Controlling off of RoboRIO DIO port
  - Takes up valuable processing time
  - More wiring complexity
- CANdle
  - Not cheap ($60), closed source, has some problems with code and firmware
- ConnectorX
  - Not cheap ($50), closed source, is relatively large, cannot use CAN

Why a PCB? Why not just an Arduino?

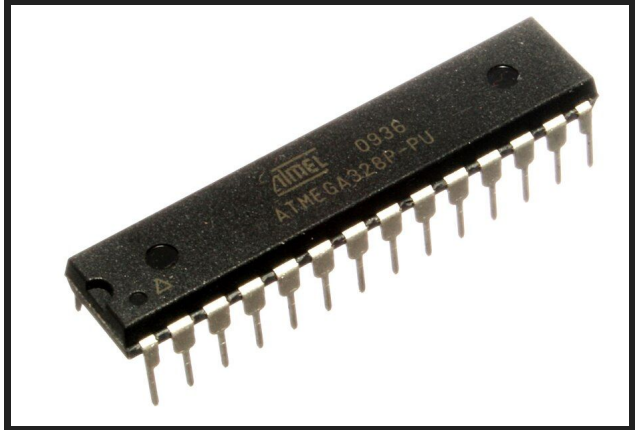
- Arduino w/ some wires and breakout boards
  - Easy, reconfigurable, quick to build, cheap
  - Wires prone to pulling out unless soldered, bulky
- Arduino on perfboard
  - Quick to build, cheap
  - Not the easiest to solder, still too large
- Arduino on a PCB
  - Easy to program, high reliability
  - Inefficient usage of space, large footprint

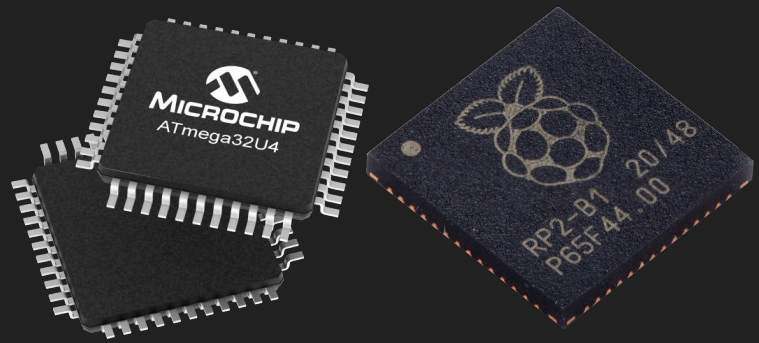




What microprocessor?

- A microprocessor is necessary to control the LED strip
- Number of options available
- What does it need to be capable of?



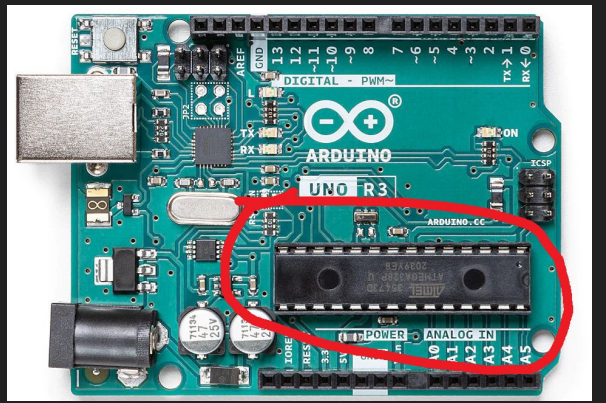


USB

- USB has a number of advantages
  - Easy to program the device
  - Easy to debug
- Hard to properly design for
  - Requires a microcontroller somewhere in the design which natively supports it (“u” series AVR)
  - Extra hardware such as 22 ohm resistors
  - Signal integrity can be a worry
- Abandoned the idea in prototyping

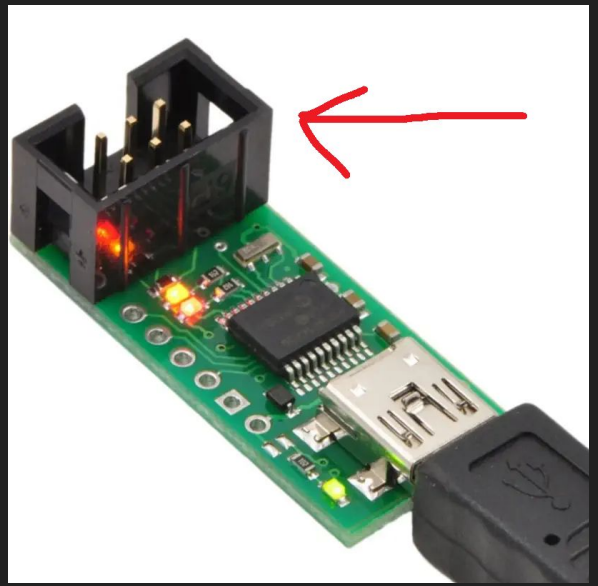
The microprocessor

- Landed on ATMega328p (best known for being the core of the Arduino UNO)
- Lots of libraries, can be programmed through Arduino Studio, powerful enough, has a DIP (through-hole) package option
- No other ICs necessary (SRAM, EEPROM, Flash all integrated into the chip)
- Drawbacks:
  - No native USB support (Arduinos use a second microprocessor)
  - Not particularly fast nor powerful (8 bit data bus)
  - Needs discrete crystal oscillator for full 16MHz clock speed
- .1 and 10 uF decoupling capacitors placed nearby



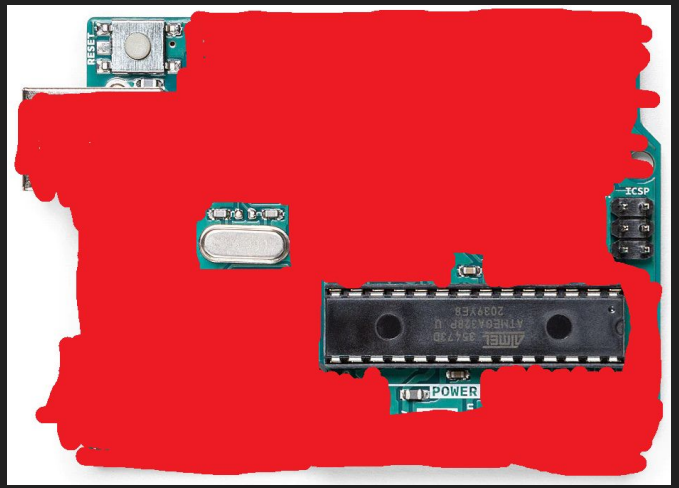
The AVR ISP Port

- AVR-architecture specific programming
- Smaller, no extra hardware needed other than an external programmer
- Extra drivers necessary, not the most reliable
- Slow, makes testing difficult



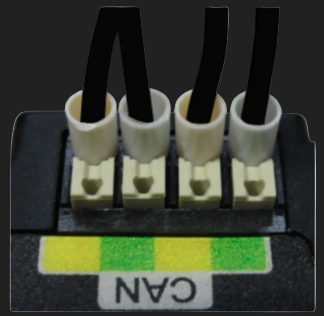
Differences from UNO

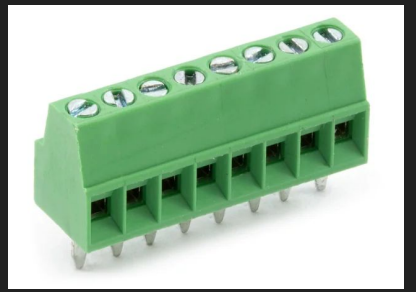
- Not everything on UNO is necessary
- No need for USB, all of the headers, etc.
  - Significant reduction in size
- ATMega328p minimal setup is outlined in documentation



Connectors

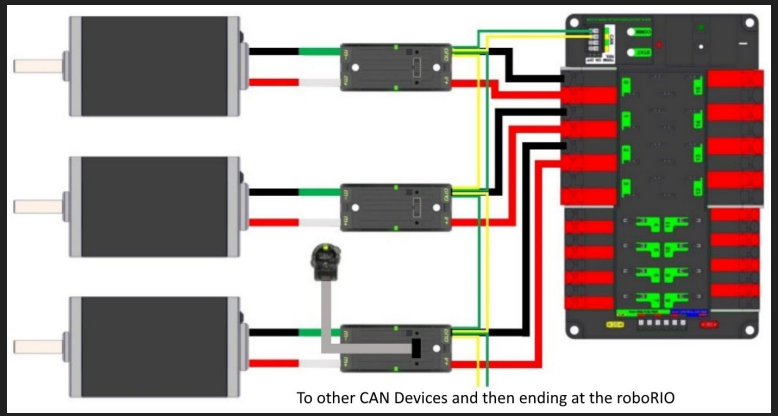
- Initial PCB had through holes which happened to be of the wrong diameter
  - Requires soldering
  - Not easily replaceable
- Went with WAGOs, which are easy to use and common in other formats in FRC
  - Little too large
- Alternatives?
  - Standard push-in Weidmuller FRC connectors
  - Screw terminals

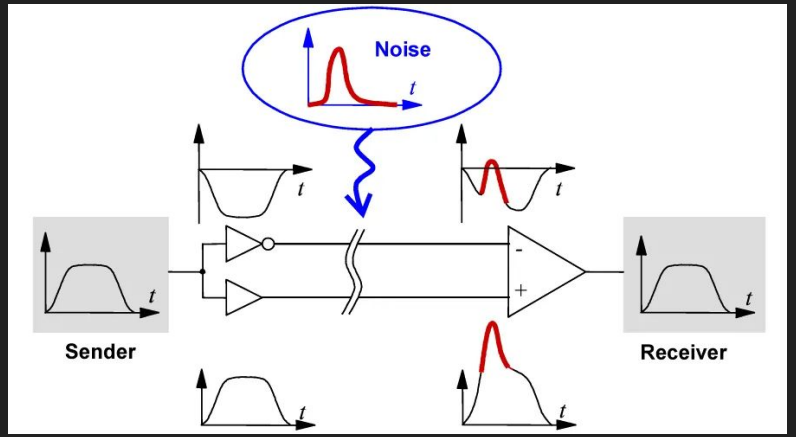




What is CAN?

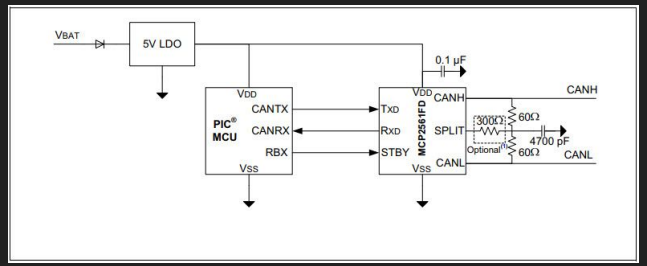
- Differential pair signaling protocol commonly used in cars
  - Extremely resilient to interference because of differential pair
- Is the most common communication protocol in FRC
- Devices are daisy-chained, so space for 2 different pairs of CAN wires necessary (unless terminating)

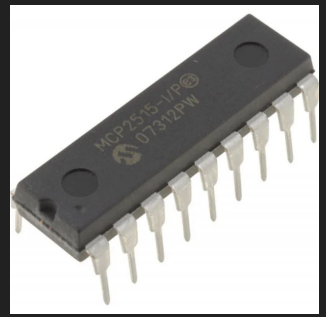




CAN ICs

- CAN is not the most common protocol
- Most common offering for the older CAN 2.0b specification is the MCP2515 CAN controller connected to some kind of transceiver
- CAN FD has the MCP2517/8, but requires another expensive closed source device to use at full speed in FRC because it uses a faster spec - CAN FD, and is only available in surface mount
- Landed on the 2515 and the MCP2561 transceiver - both are in production and are compatible





The FRC CAN standard

- FRC uses its own CAN standard
- Uses extended(29-bit) IDs and separates them into sections
  - All devices have a distinct 6-bit number at the end of their ID
- Every 20ms, a “heartbeat” packet is sent over CAN to give system time, status, etc
  - Its ID overlaps with other packets from the RoboRIO - packets we don’t care about
- Also requires devices to shut down when the robot is disabled

Example

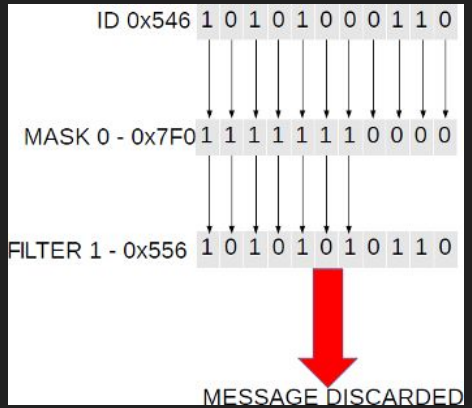
Speed Control Mode Disable from Luminary Micro Jaguar Speed Controller (dev # 4)

Field	Device Type	Manufacturer Code	API		Device Number
			API Class	Index	
Value	2	2	1	1	4

Bits	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
Bit Position	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Programming with CAN

- The 2515 library uses “masks” and “filters”
- Masks designate which bits matter, filters specify which values of those bits matter
- Takes some testing to make sure unnecessary packets aren’t clogging your receive registers on the 2515
- Had to make it disable upon receiving the shutdown broadcast message + re-enable upon it being rescinded

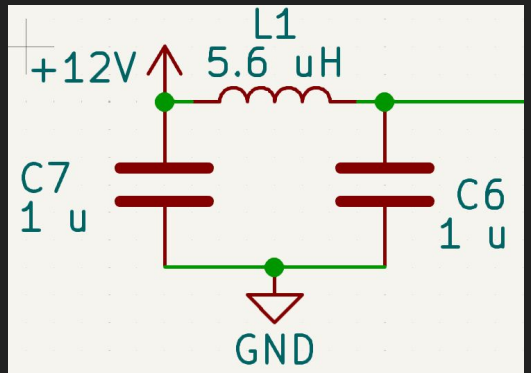


Power Supply

- Most or all power on an FRC robot is 12v
- All of the ICs used require 5v
- Conversion to 5v in FRC uses a common module - but takes up more space then necessary
- Reverse polarity protection would be nice

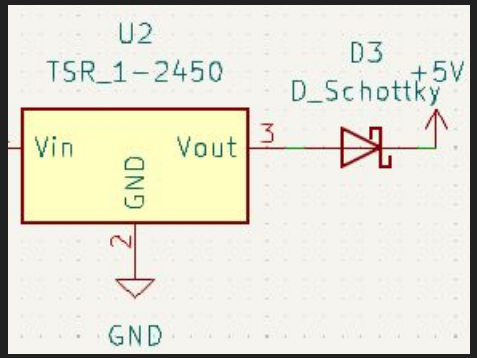
Filtering of 12v in

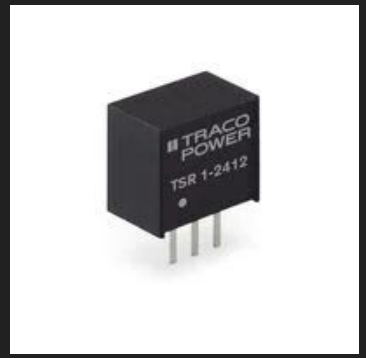
- Capacitors “pass” changes in voltage and “block” constant voltages while inductors “pass” constant voltages and “block” changes
- Capacitors placed between 12v and ground, inductor placed in series
  - Another datasheet recommendation to reduce EMI
- Downside - inductors use magnetic fields, so increased electromagnetic interference
- Slight mistake - choke was needed, not just any inductor (worse for EMI)



Power Supply IC

- Plenty of options here
- Preferably low profile with somewhere around 5 volt 1 amp output
- Needs some tolerance with the input - battery voltage in FRC can ‘droop’ all the way to 7v under high load current and is frequently >13v at full charge
- TSR 1-2450e seemed like a good idea - small (only 3 pins), high efficiency (97%) & high input tolerances (7-36v), non-isolated (common ground)
- 5v bus is connected to it through a Schottky diode for protection



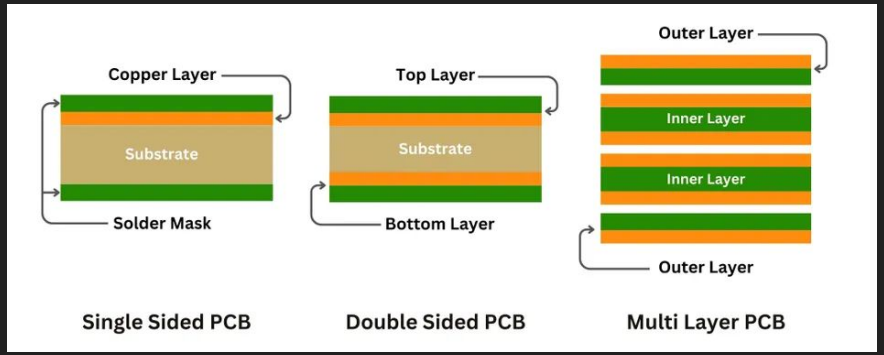


Persistent issue

- TSR likes to generate heat
- Doesn’t *seem* to hurt anything, but is definitely noticeable with a full strip of LEDs
- Could be harmful over time
- Potential solutions:
  - Higher efficiency converter
  - Passive cooling

PCB Layers

- PCBs are separated into layers - amount of layers varies
- 2 main options - 2 or 4 layer
- 4 layer is more expensive
- 2 layer can be harder to route and forgoes some advantages of 4 layer
    - Namely a separate power/ground plane for signal integrity
    - Easier separation of data lines

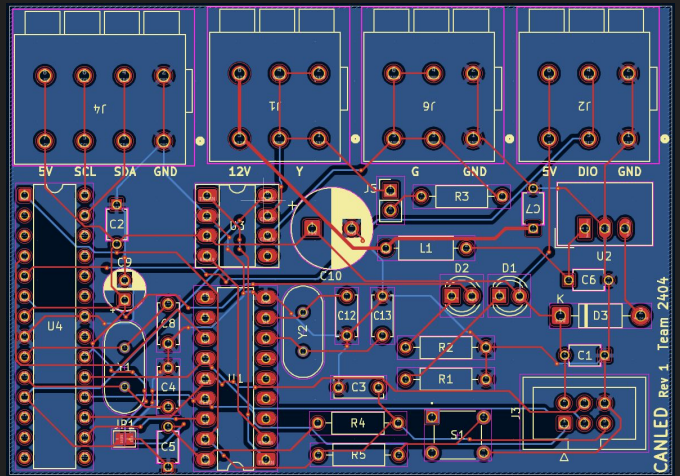


Ground Plane

- Big advantage of 4 layer PCBs - one whole layer can be dedicated to ground
  - Improves signal integrity and structural strength of board
  - Takes full advantage of your copper pour
- In the first iteration, ground was just a trace which ran throughout the board
- In the revision, I managed to make the second layer of the design (mostly) ground plane

Don’t be like me

- Gaps in the ground plane are really bad for EMI, especially if you run any high speed signals over them
- I ran the highest speed signal over one… ATMega 16MHz crystal trace
- Along with having another really big gap
- Hasn’t had a noticeable effect on performance, but its not great for signal integrity

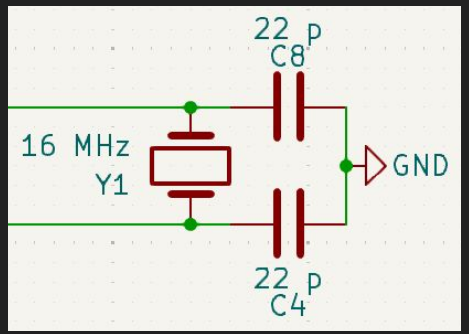


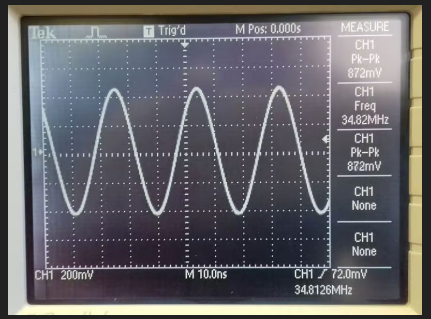
Controlling & Powering the LED strip

- Pretty simple - communication is just a single digital pin
- Lots of libraries on the software side
- Maintaining signal integrity is less of a worry than it seems - digital is pretty resilient & it’s traveling over ground plane
- Datasheet wants a 1000 uF capacitor between power and ground near it

Crystal oscillators

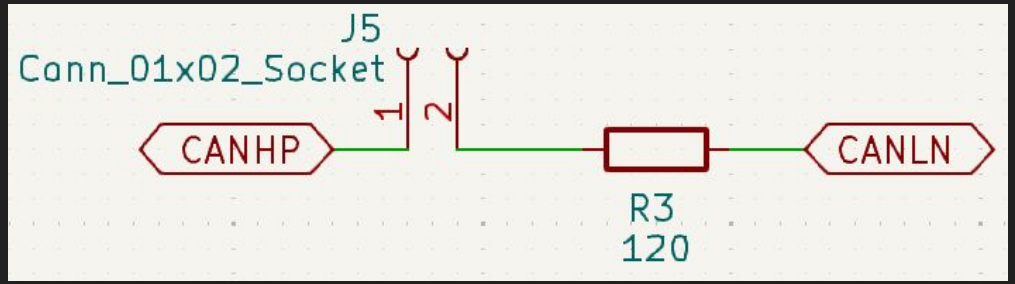
- Component which can create a sine wave with a pair of bypass capacitors and feedback
- 2 necessary - one for the 2515 and one for the 328p
- Both are placed away from 12v and over ground plane for signal integrity



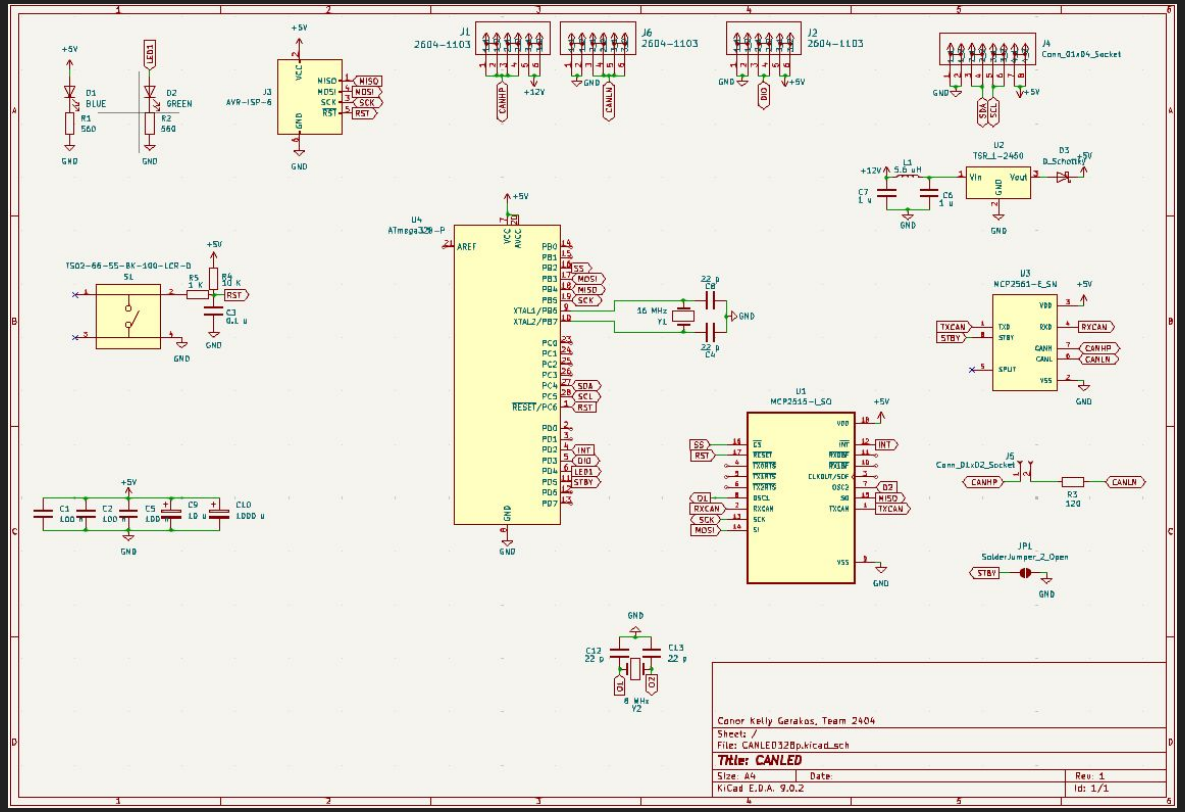


Extra features

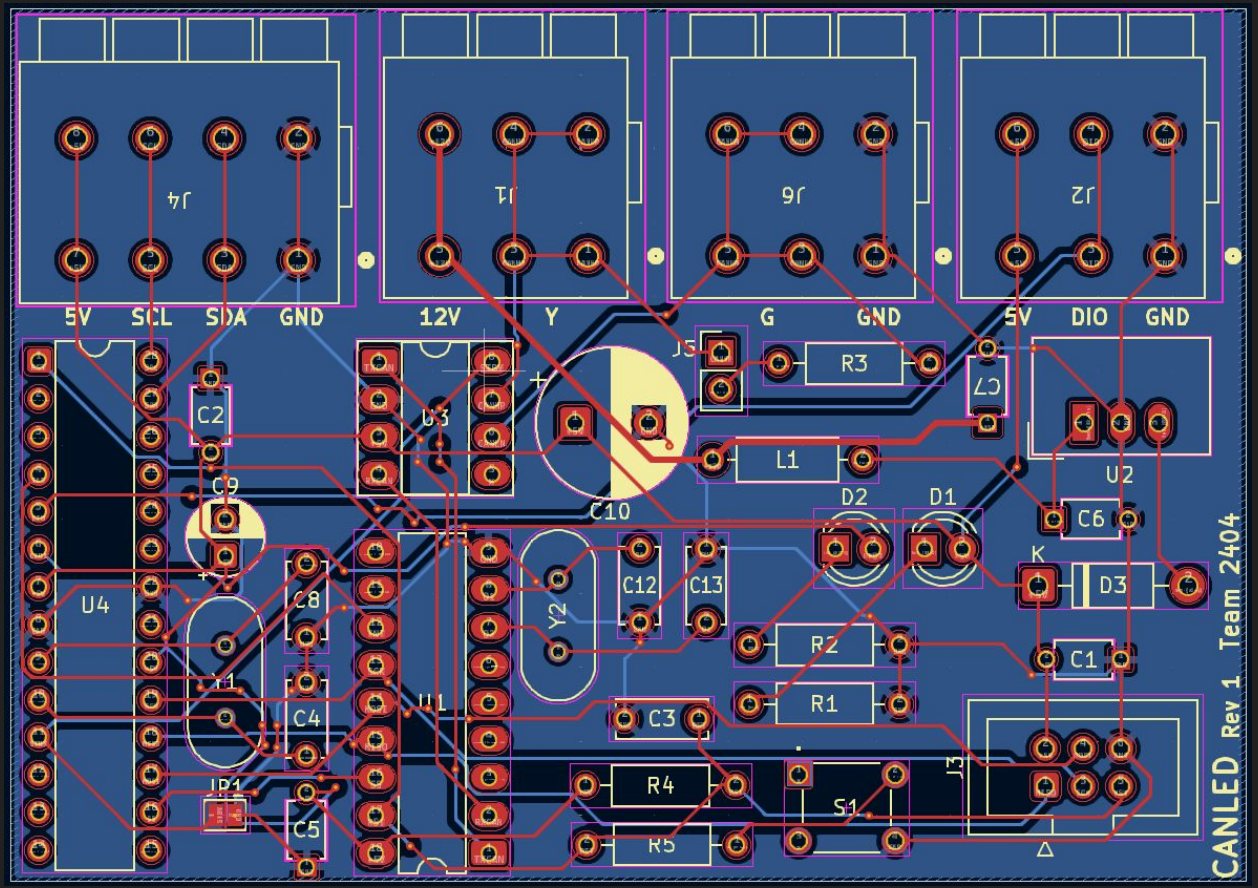
- CAN termination optional with 2-pin header which can be shorted together to connect CANH and CANL together and to ground through a 120-ohm resistor
- I2C port for any number of devices (time of flight, IMU, etc)
- Solderable jumper to short STBY on the 2561 to ground



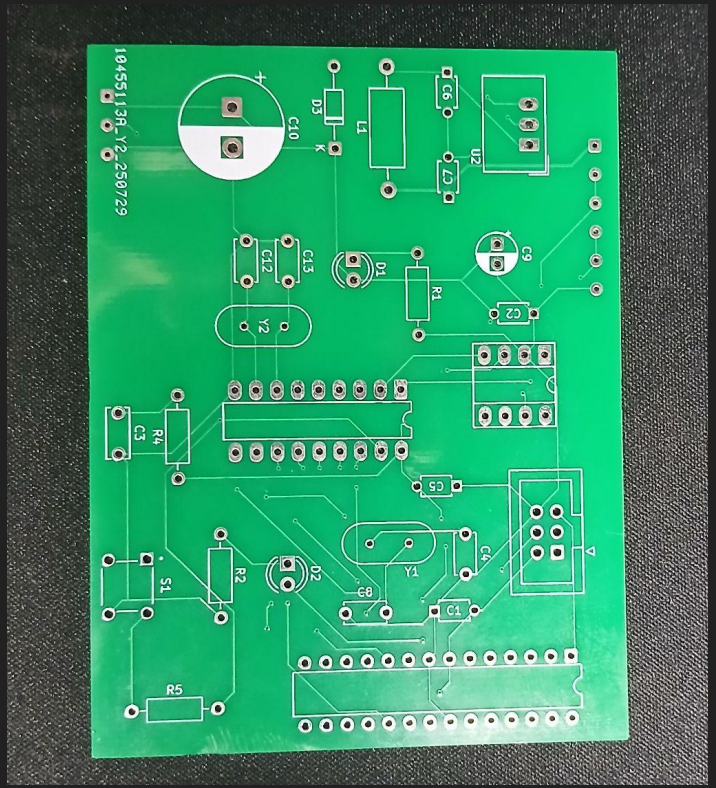
Schematic

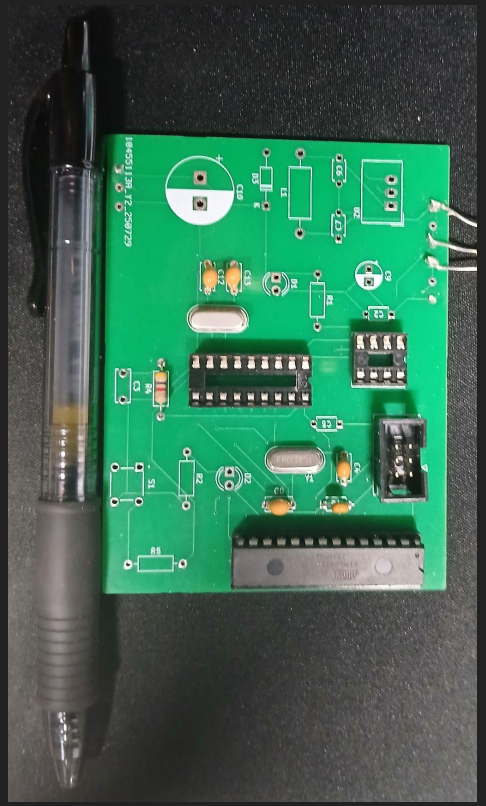

Conor Kelly Gerakos, Team 2404
Sheet: /
File: CANLED328p.kicad_sch
Title: CANLED
Size: A4
Date:
Rev: 1
KiCad E.D.A. 9.0.2
Id: 1/1

Layout

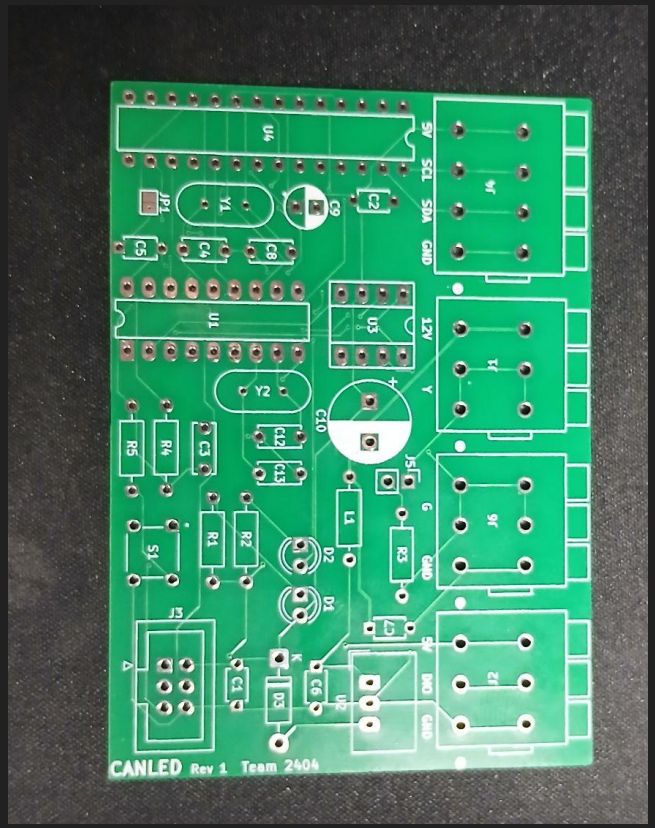

5V
SCL
SDA
GND
12V
Y
G
GND
5V
DIO
GND
J4
J1
J6
J2
U4
U1
U2
C2
C9
C8
C4
C5
C10
C12
C13
C3
C7
C6
C1
R3
L1
D2
D1
D3
R2
R1
R4
R5
S1
J3
Y1
Y2
IP1
K
CANLED Rev 1 Team 2404

First PCB


104551138_Y2_250729
C10
D3
K
L1
C6
C7
U2
C12
C13
D1
R1
C9
C2
Y2
R4
C3
C5
C4
Y1
S1
R2
D2
C8
C1
R5



Revision

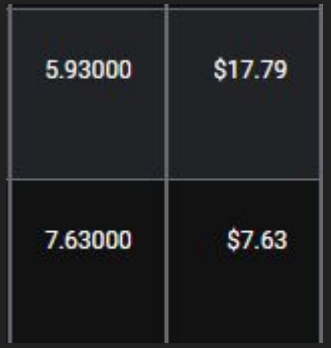


Shortcomings

- Current draw of LED strip is higher than expected - 2A converter would be much better for larger strips
- Board is overall too large (WAGOs)
- Mounting holes would be useful
- Ground plane gaps… Board should have been 4 layer

Pricing

- Using codes, PCB manufacturing cost for 5 boards from JLCPCB was $3.20
- Most expensive components were… the WAGO connectors
- All four of them increased the price by $25…
- Total price of $50.06 without shipping, so half the price was connectors
- Overall not horrible, priced similarly to the cheapest closed source competitor
- Tip: Check component prices before you design!



Potential Upgrades

- MCP2515 and 2551 could be combined into 25651 surface-mounted IC
- New controller for FRC has native support for CAN Flexible Data rate, so the 2561/2515 combo will need to be replaced with 2562/2518 for full speed performance
- WAGO connectors are too big, could be replaced with more common connectors
- Could move from through-hole to surface-mount components to further reduce size
- ATMega328p isn’t super powerful - could upgrade to use more complex LED patterns

Conclusion

- All files available on github - github.com/Team-2404/CANLED
- Project was overall successful - it can control a decently sized strip of addressable LEDs
- We plan to use it on our competition robot this year
- FRC is plagued by copious amounts of expensive closed-source hardware, despite most software being FOSS
- FRC could use more electronics design instead of just wiring - hopefully this inspires other teams to design their own custom circuits

Thank You!

Q&A