**AUTODESK
Instructables**
(/)

Projects (/circuits/projects/)   Contests (/contest/)
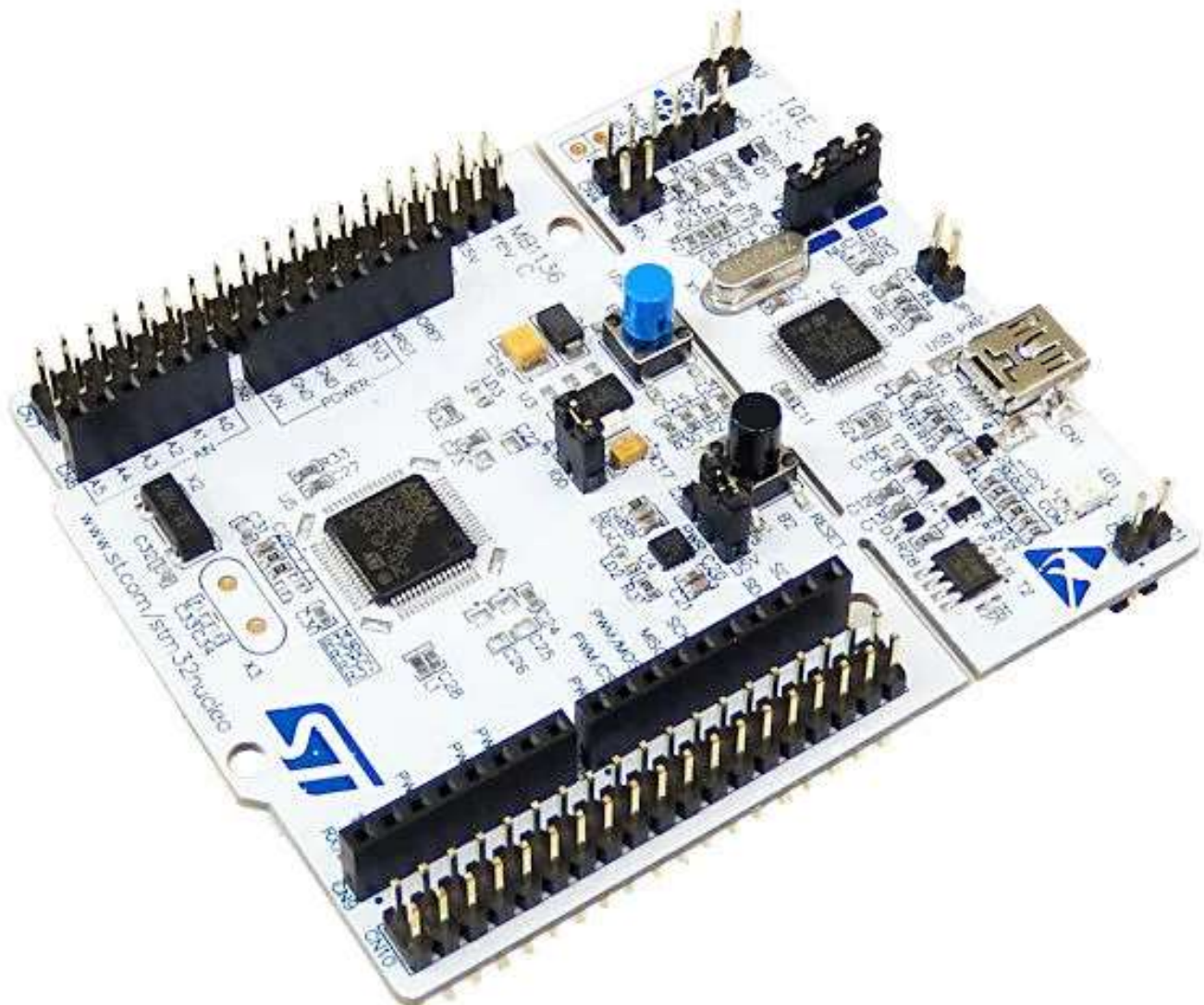
🔍 Search (/search)

STM32 CAN Interface

By UBC Solar (/member/UBC+Solar/) in Circuits (/circuits/) > Microcontrollers (/circuits/microcontrollers/projects/)

64,754   14   1

Download   Favorite

(https://www.instructables.com/member/UBC+Solar/)

The Controller Area Network bus, or **CAN bus**, is a very effective communication protocol thanks to its high speed capabilities, long range reliability, and noise immunity. For these reasons, CAN communication has become the standard in automotive technologies and high noise environments. Devices on the CAN bus are called **nodes**. All nodes on the CAN bus are

connected in parallel, meaning that each node is connected to all the other nodes on the network. A single CAN bus can have up to 115 nodes at once, depending on the rate of message transmission, but for most applications, it is recommended to have up to 32 devices. It is also recommended to keep the length between the first and last node less than 40 meters apart.

This step-by-step tutorial will show you how to set up a CAN node using the STM32 microcontroller, including the circuit and simple C code to read and write to the CAN bus
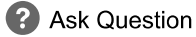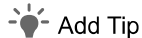
Supplies

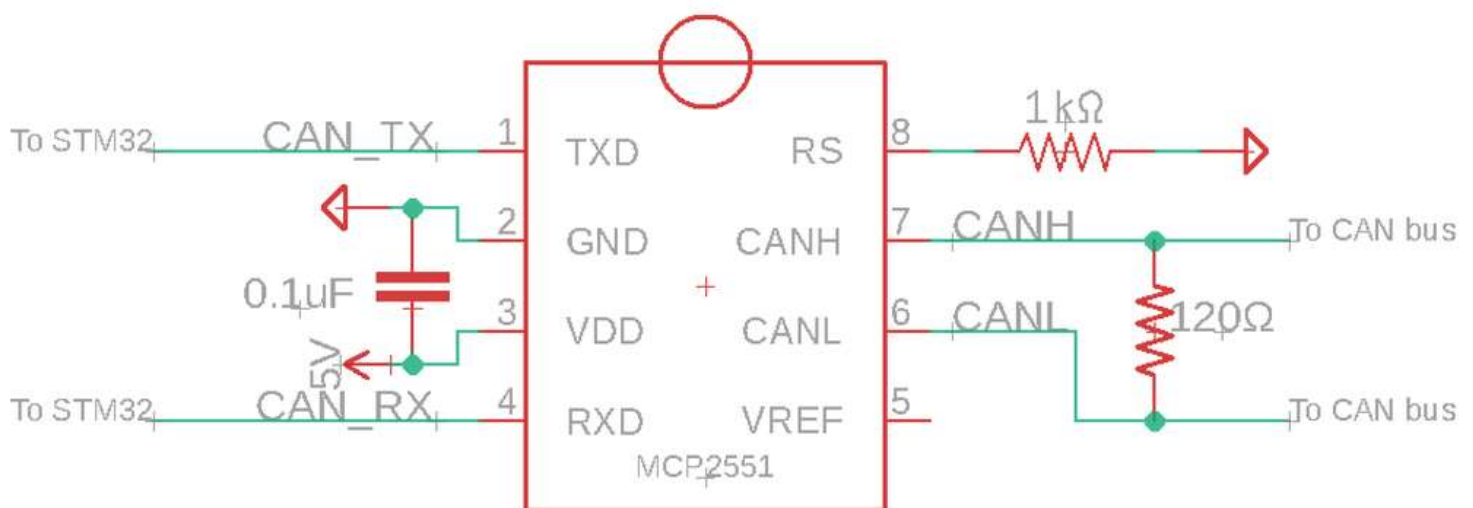For each CAN node:

- 1x STM32 breakout board (Nucleo, Blue Pill, others)
- 1x MCP2551 CAN transceiver IC
- 1x 0.1µF capacitor
- 1x 120Ω resistor
- 1x 1kΩ resistor
- 1+ readable input (button, switch, potentiometer, etc.) or output (LED, MOSFET, etc.)
- 1x Dsub9 connector

Step 1: The Transceiver Circuit

To communicate with the CAN bus, we will use the MCP2551 CAN transceiver IC. The IC acts as an intermediate transmitter/receiver pair to connect the STM32 to the CAN bus. The circuit to set up this IC is quite simple, but there are a few things to note:

- The CAN_RX (pin 4) and CAN_TX (pin 1) on the MCP2551 chip can only go to certain pins on the STM32.
  - On the STM32F1 Nucleo, connect the RX line to pin PB8 and the TX line to pin PB9.
  - On the STM32F1 blue pill, connect RX to pin PA11 and TX to pin PA12.
  - Note that these pin assignments have alternatives. Refer to the microcontroller manuals to determine which pins are capable of CAN_RD and CAN_TD
  - If using an Arduino or a board without a CAN communicator built in, the MCP2515 IC chip will be required to convert other message protocols to CAN.

- The CANL pin should be connected to the other CANL pins of the other bus nodes. The same goes for the CANH pins.
- The 120Ω resistor across the CANH and CANL pins is only required if the node is a terminal node. This means that it is at the end of parallel connection wiring. In other words, the CAN bus should only have two 120Ω resistors in it, and they should be as far apart from each other as possible.
- Lastly, the 1kΩ resistor on RS (pin 8) can be exchanged for a 10kΩ resistor to control the rising/falling time of the CAN message bits. Refer to the MCP2551 chip datasheet for more details.

Step 2: Reading and Writing to the CAN Bus

```
//Battery: Faults, Battery High and Battery Low (For Dashboard Indicator)
case 0x622:

        //A10: Check if the high voltage bit is set, meaning battery is full
        if ( (CAN_rx_msg.data[6] >> 1) & 0x1)
        {
                //Reset pp;in A10, turn on LED
                GPIOA->BSRR = 0x1 << 10;
        }
```

Now that the transceiver circuit is connected to the STM32, we can start writing messages to the CAN bus. This instructable guide will not go in-depth into the STM32 code. However, be sure to check out our code for examples here (https://github.com/UBC-Solar/Firmware-v2/tree/master/Speed_Controller). Using the STM32 as a CAN node will require a CAN header file. We wrote our own, which can be found on our github here (https://github.com/UBC-Solar/Firmware-v2/tree/master/Peripherals). Here, we will give a brief overview of the read/write process.

To read from the CAN bus, we first need to know the ID of the CAN message. Every message should have a unique ID, with lower IDs having higher priority. The code snippet shown here is waiting for a CAN message with ID 0x622. In our system, if the first bit of the 6th byte is high, then we want to set pin A10 high.

When writing a CAN message, we must remember that CAN messages are multi-byte. Each written message must have an ID and length. In the second snippet of code shown, we write data to each byte, then send the message (The ID and length parameters are defined earlier in the code).
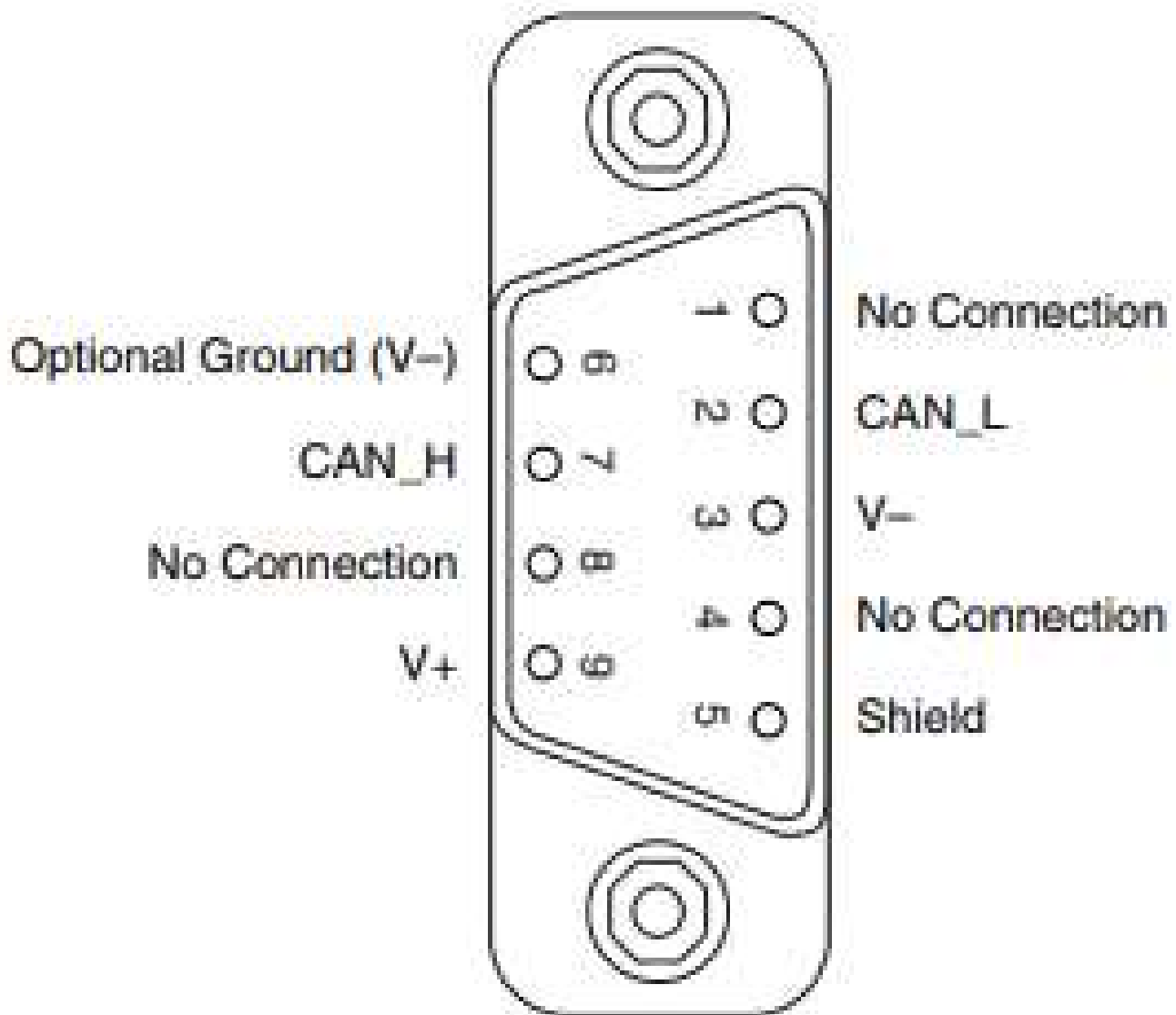
Add Tip   Ask Question   Comment   Download

Step 3: Connecting Nodes

Optional Ground (V–) 6
CAN_H 7
No Connection 8
V+ 9

1 No Connection
2 CAN_L
3 V–
4 No Connection
5 Shield

When connecting multiple CAN nodes, careful attention should be paid to the length of cables. The two farthest nodes can be up to 40m apart from each other. Middle nodes connecting to the bus should be within 50cm of the main bus lines.

CAN connections follow an industry standard of using a Dsub9 connector with the CANL line on pin 2 and the CANH line on pin7. The option CANGND line can go on pin 3.

Add Tip   Ask Question   Comment   Download

Step 4: Make the PCB
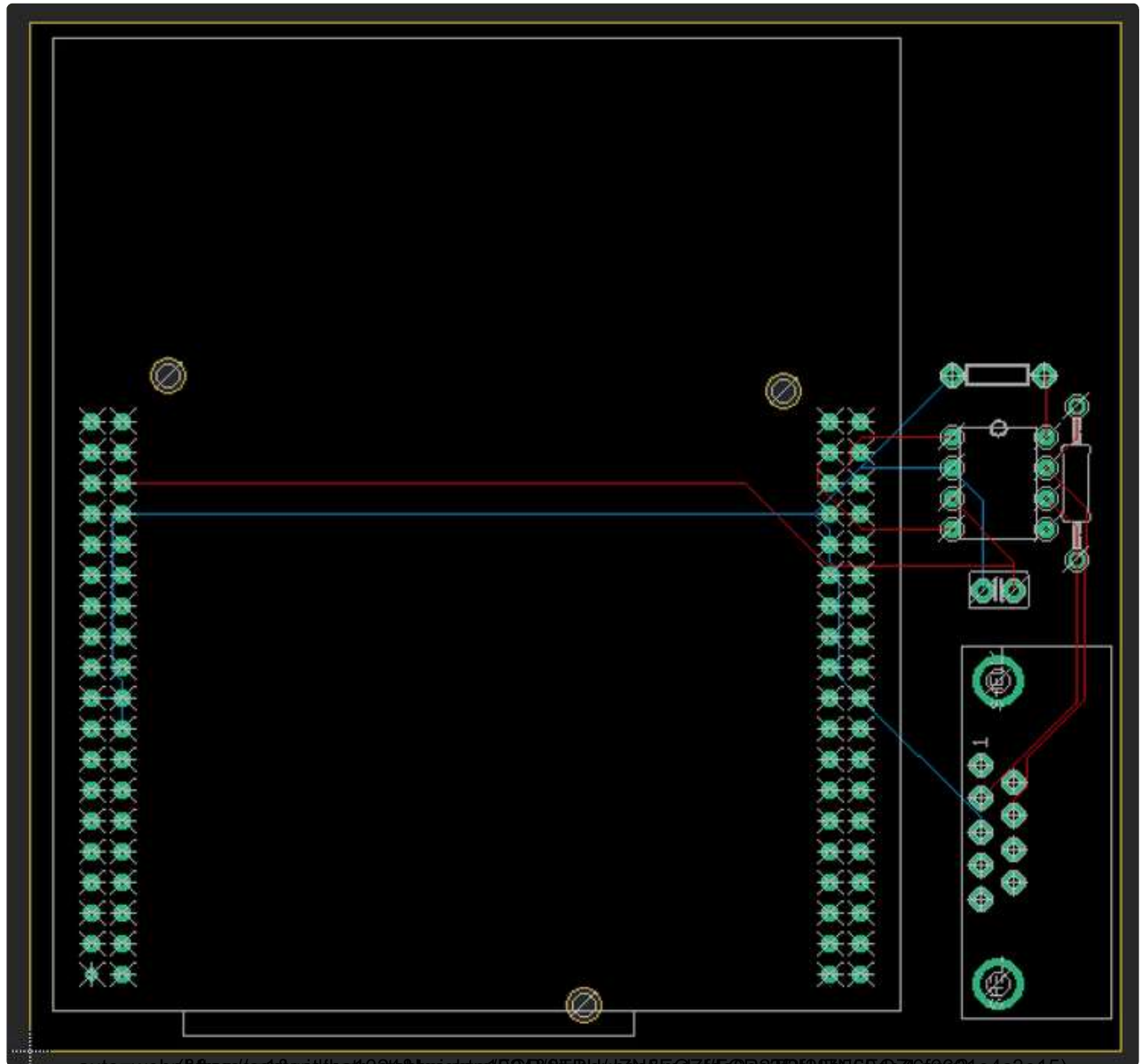
When routing CAN signals on a PCB, remember that CAN is a differential signal, and thus, routing guidelines for the CANH and CANL should be followed carefully.
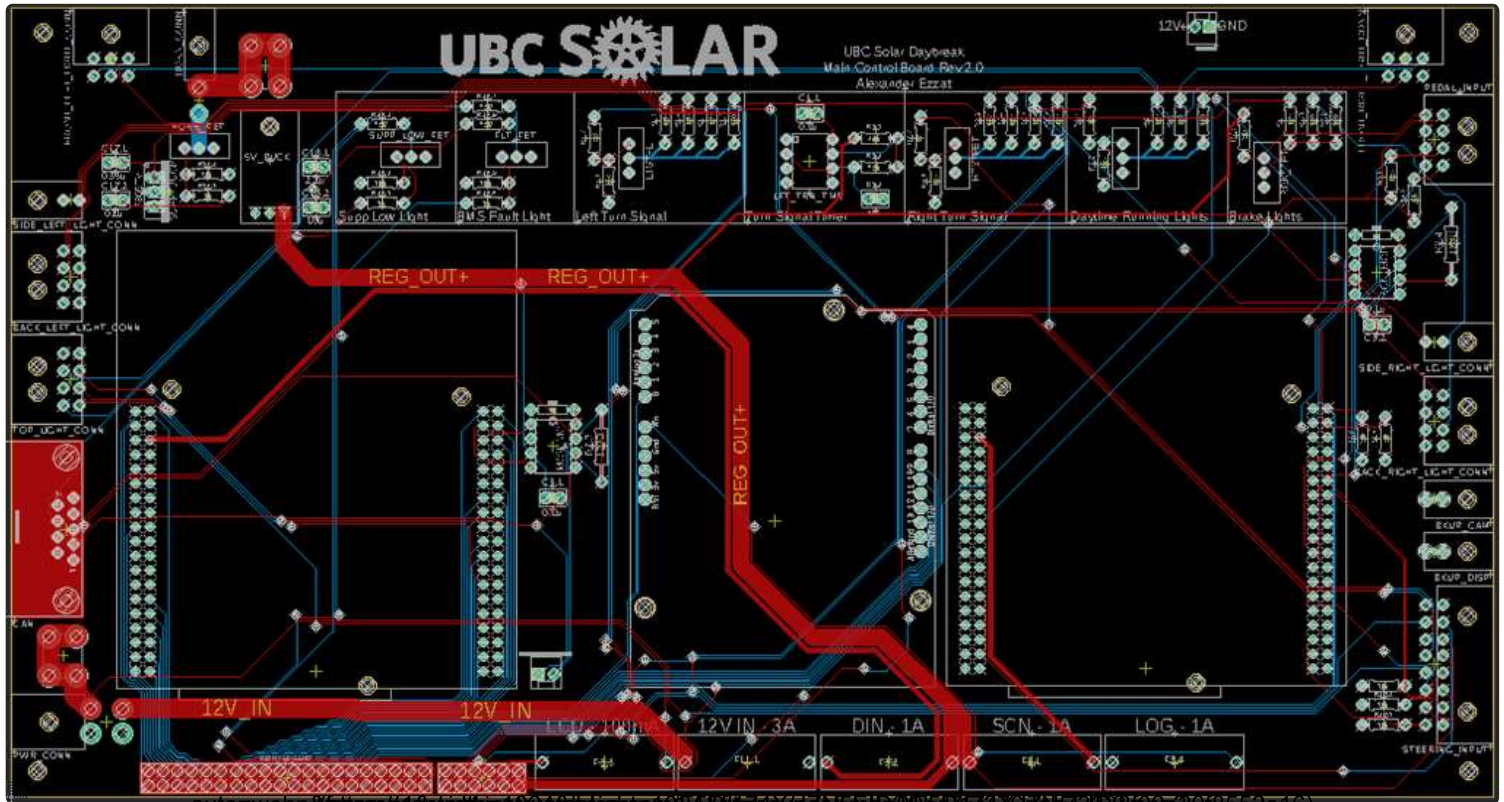
Add Tip   Ask Question   Comment   Download

Step 5: Expanding the Board

Throw some more nodes together, add some inputs/outputs, and connect all their CANH and CANL pins. Note that each STM32 or other microcontroller requires its own MCP2551 chip; they cannot be shared.

With that said, try to keep your PCBs smaller than the one shown here

Add Tip   Ask Question   Comment   Download

Step 6: Order Your PCBs From JLCPCB

JLCPCB offers fast, high quality service at very reasonable prices. Get 5 boards, any colour with tons of customization, for only $2! And if it's your first order, get 10 boards for the same price!

Just upload your gerbers and get an instant quote! Submit your order and your boards will be reviewed for production within the hour. Once you pay, you can expect your high quality boards in as soon as three days!

Check it out here (https://jlcpcb.com/quote)
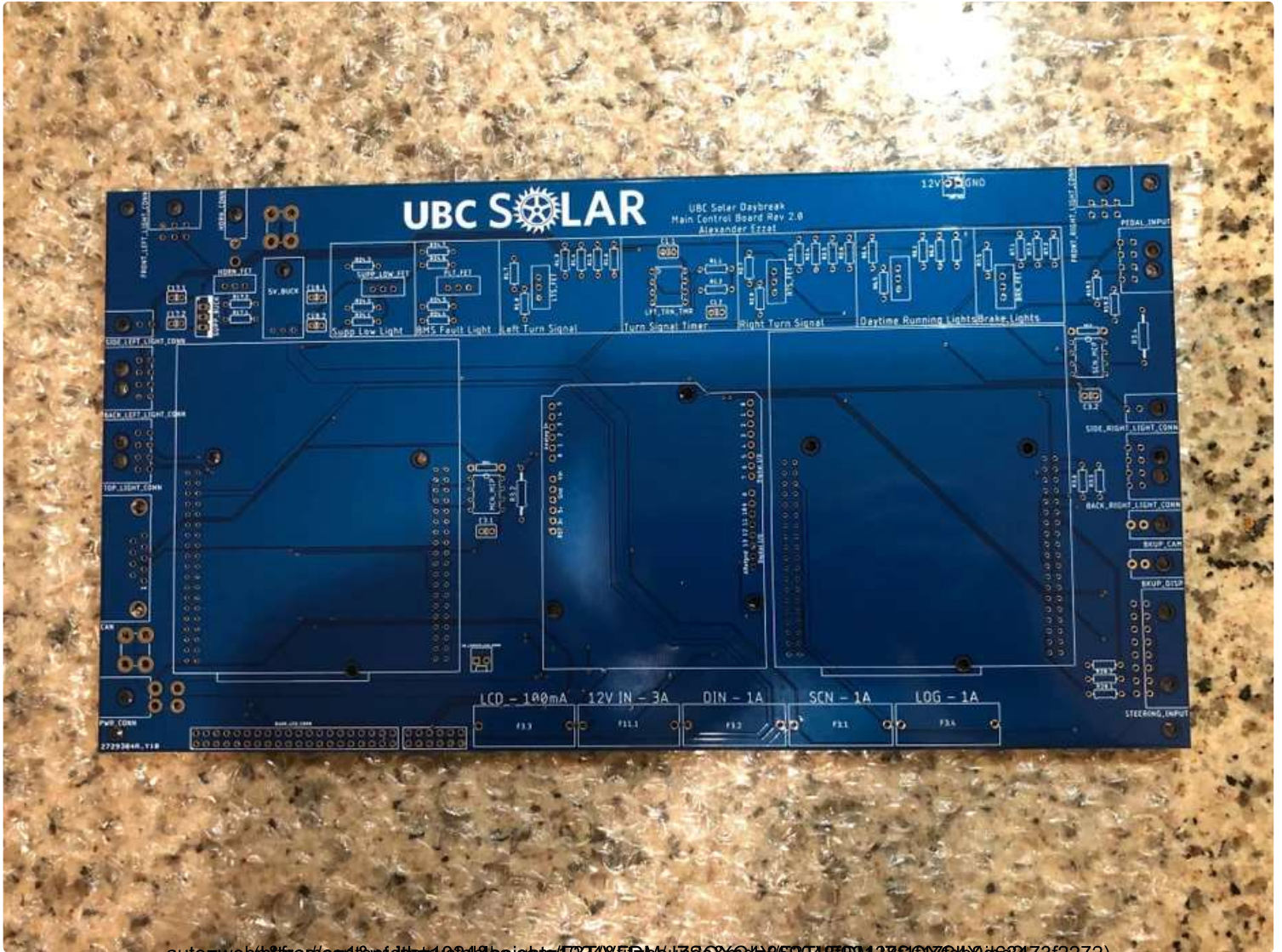
💡 Add Tip   ❓ Ask Question   💬 Comment   Download

Step 7: Get Your Boards!

Huge shout out to JLCPCB for sponsoring this project. JLCPCB (ShenzhenJLC Electronics Co., Ltd.), is the largest PCB prototype enterprise in China and a high-tech manufacturer specializing in quick PCB prototyping and small-batch PCB production. They were kind enough to provide UBC Solar (http://ubcsolar.com) with our new PCBs for our solar-powered race car. We placed our order on Friday and got the boards on Wednesday!

Add Tip   Ask Question   Comment   Download

Be the First to Share

Did you make this project? Share it with us!

I Made It!

Recommendations

(/Make-Your-Own-Flashing-Lights-Freeform-Sculpture-W/)

Make Your Own Flashing Lights Freeform Sculpture With a 555 Timer (/Make-Your-Own-Flashing-Lights-Freeform-

❤ 14 👁 1.7K

(/SPI-Nixie-Display/)

Nixie Display Module With SPI Interface (/SPI-Nixie-Display/) by CaptainBrainFart (/member/CaptainBrainFart/) in Microcontrollers

🏆 (/)   ❤ 123 👁 17K

(/contest/anythinggoes/)

💡   Add Tip

❓   Ask Question

💬   Post Comment

We have a **be nice** policy.
Please be positive and constructive.

Add Images   Post

Comments

(/member/sanfour2011/)   sanfour2011 (/member/sanfour2011/) 2 years ago on Step 7

Reply   ▲ Upvote

where is the second Code Snippet for Writing? I think you want just Advertising for JLCPCB

Post Comment

Categories

🖥 Circuits
(/circuits/)

🔧 Workshop
(/workshop/)

✂ Craft
(/craft/)

🍴 Cooking
(/cooking/)

🏠 Living
(/living/)

🚲 Outside
(/outside/)

📘 Teachers
(/teachers/)

Find Us

About Us

Who We Are (/about/)
Why Publish?
(/create/)

Resources

Sitemap (/sitemap/)
Help (/how-to-write-a-great-instructable/)
Contact (/contact/)