

Serial Interface Debug

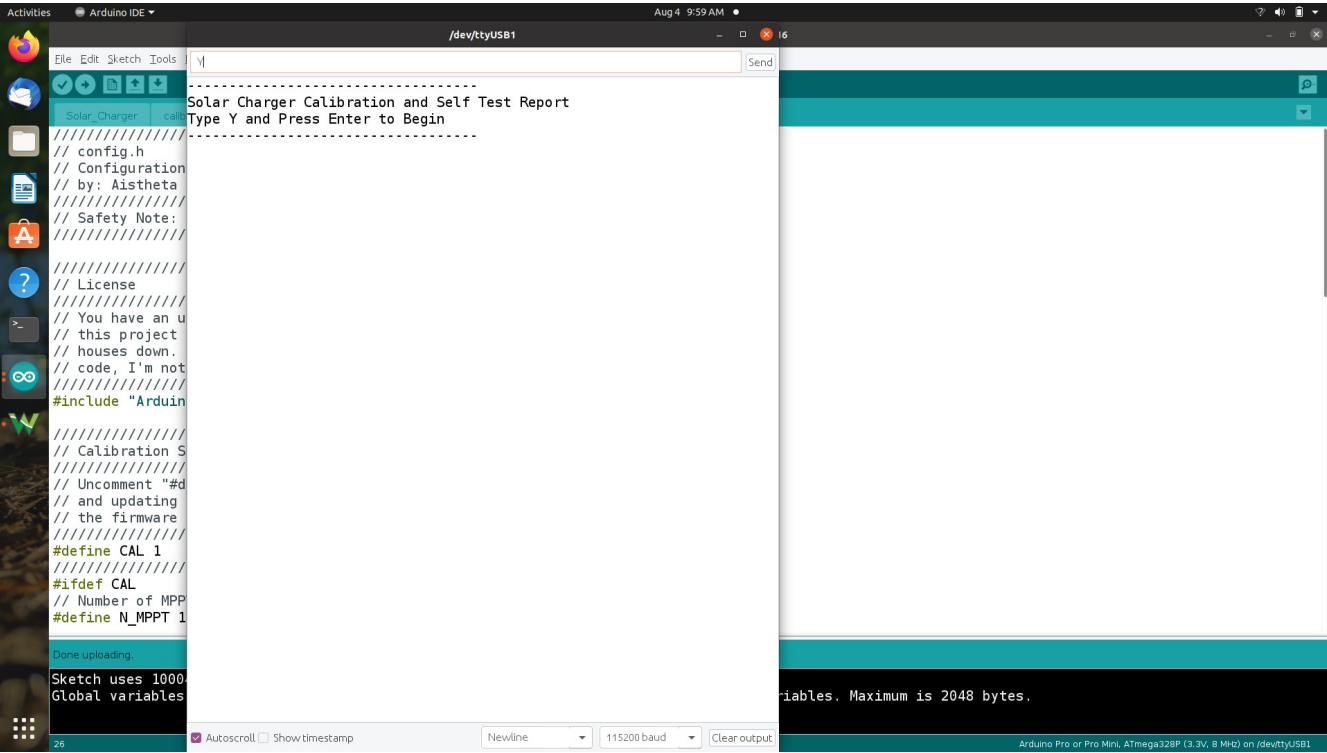
Problem: Arduino Pro Mini with FTDI adapter will not respond or read serial commands, however it can download code and transmit serial with no issues (well there was one issue, when trying 9600 baud transmit showed up as garbage on serial terminal, so I increased to 115200 and it solved).

I have confirmed the following:

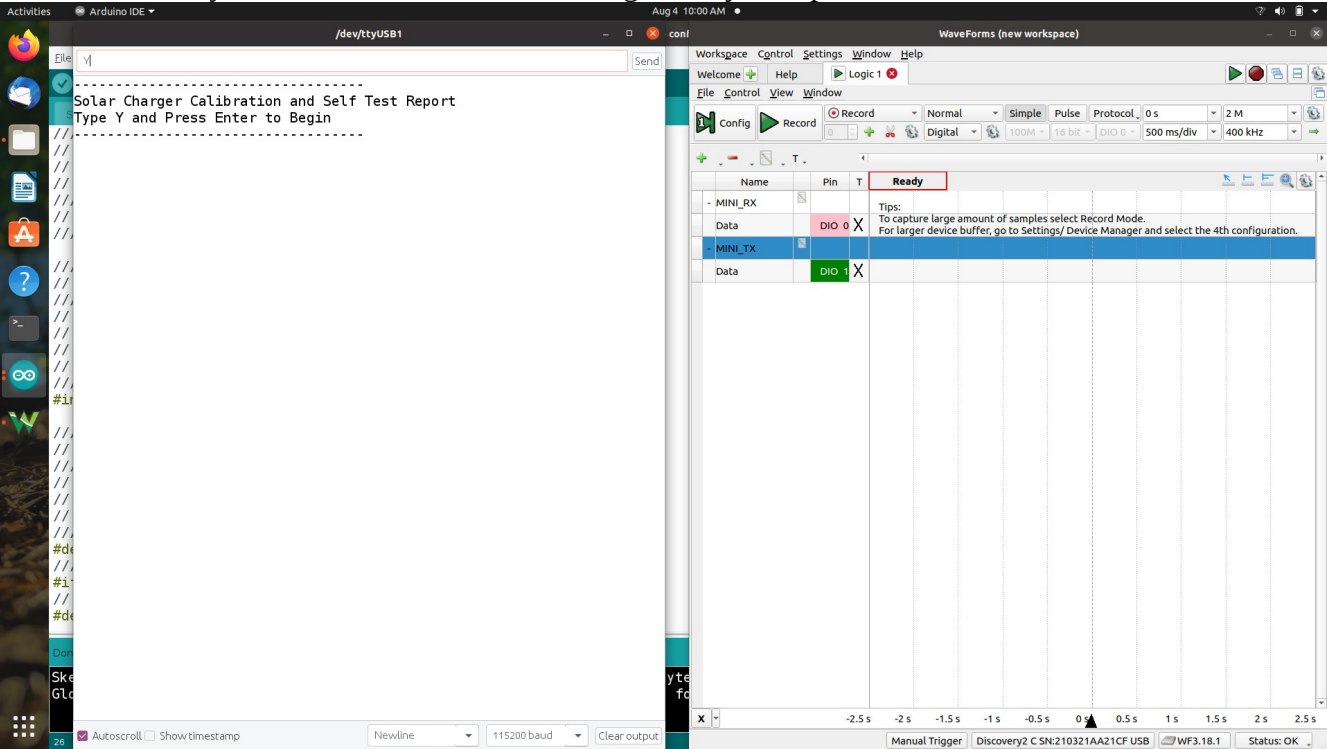
1. Arduino settings match the board (3.3V 328P 8MHz)
2. Added my user to the dialout group with “sudo usermod -a -G dialout adam”
 1. USB Serial devices are part of the dialout group, this gives Arduino run by adam permissions to the device handle
3. The serial output of the FTDI board that connects (TBD precisely) to the Pro Mini does receive proper serial
 1. Using Waveforms and Analog Discovery 2
 1. I ran a logic analyzer on the RX and TX pins and when typing Y into the Serial Monitor I can see the Y character and the line feed character being transmitted properly; logic decoded characters and it matched expected.
 2. I sniffed the RX line with the oscilloscope and captured the Y transmission and the levels are proper 0 – 3.3V.
4. I am not using the RX0 in my circuit, the pin is untouched.
5. I can download code just fine, which uses the RX pin to receive the code.

These confirmations prove that there is no issue with the Arduino Serial monitor, FTDI software, or Linux permissions as the FTDI USB → Serial is generating serial properly. The issue lies on the Arduino Pro Mini.

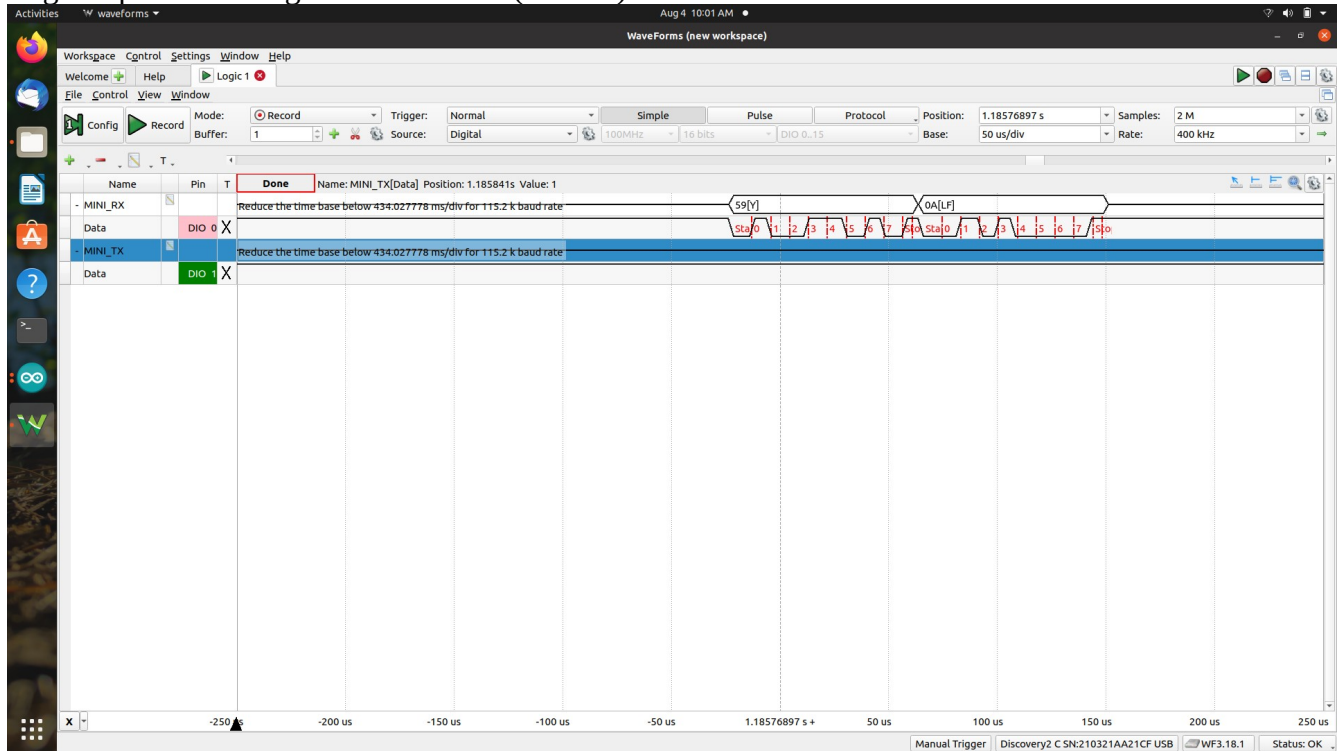
Y character loaded into Serial Monitor ready to transmit



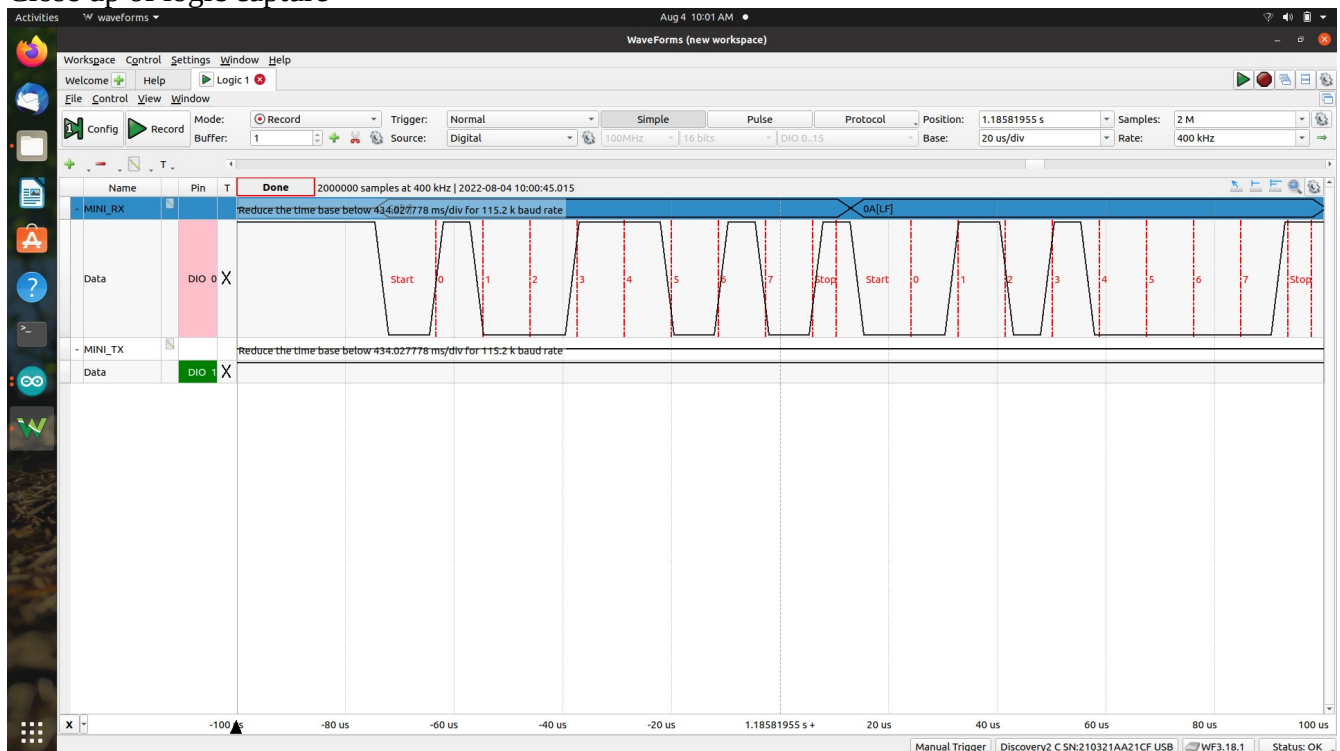
Y character ready to transmit with Waveforms logic ready to capture



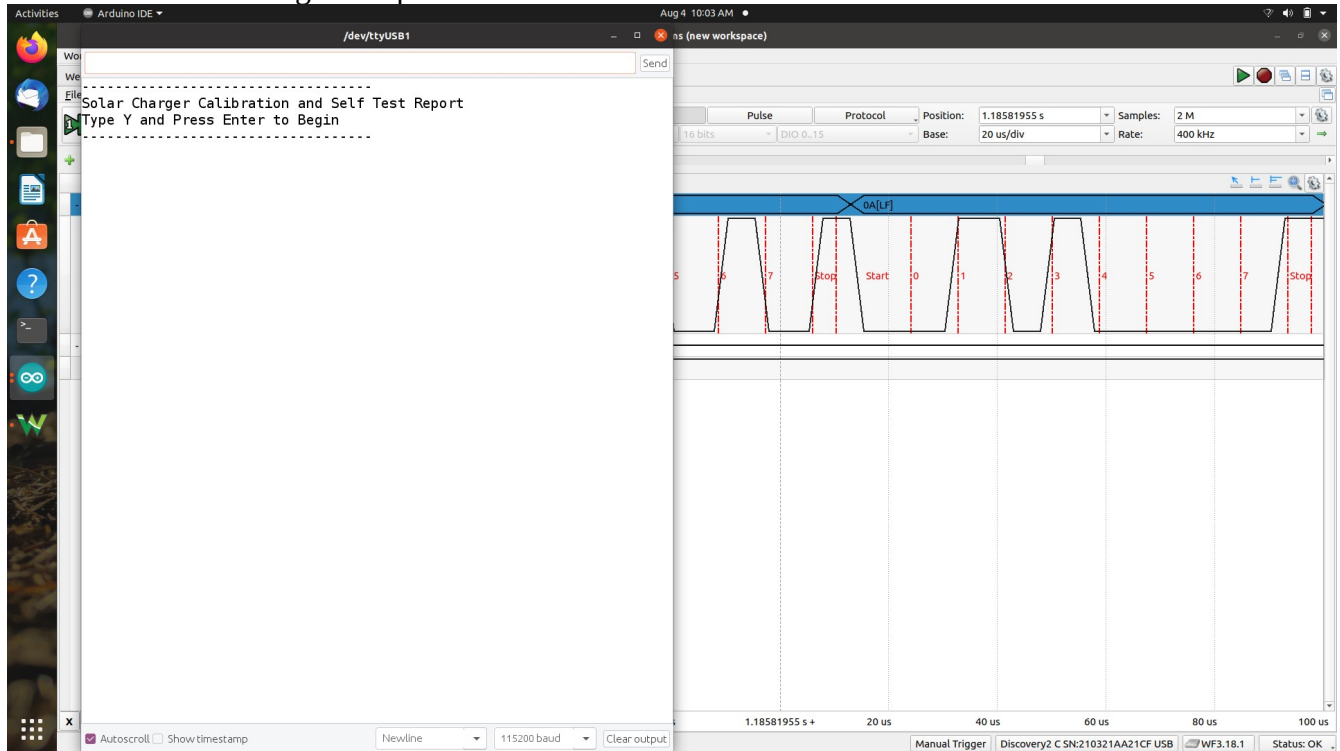
Logic capture showing decoded values (Y + LF)



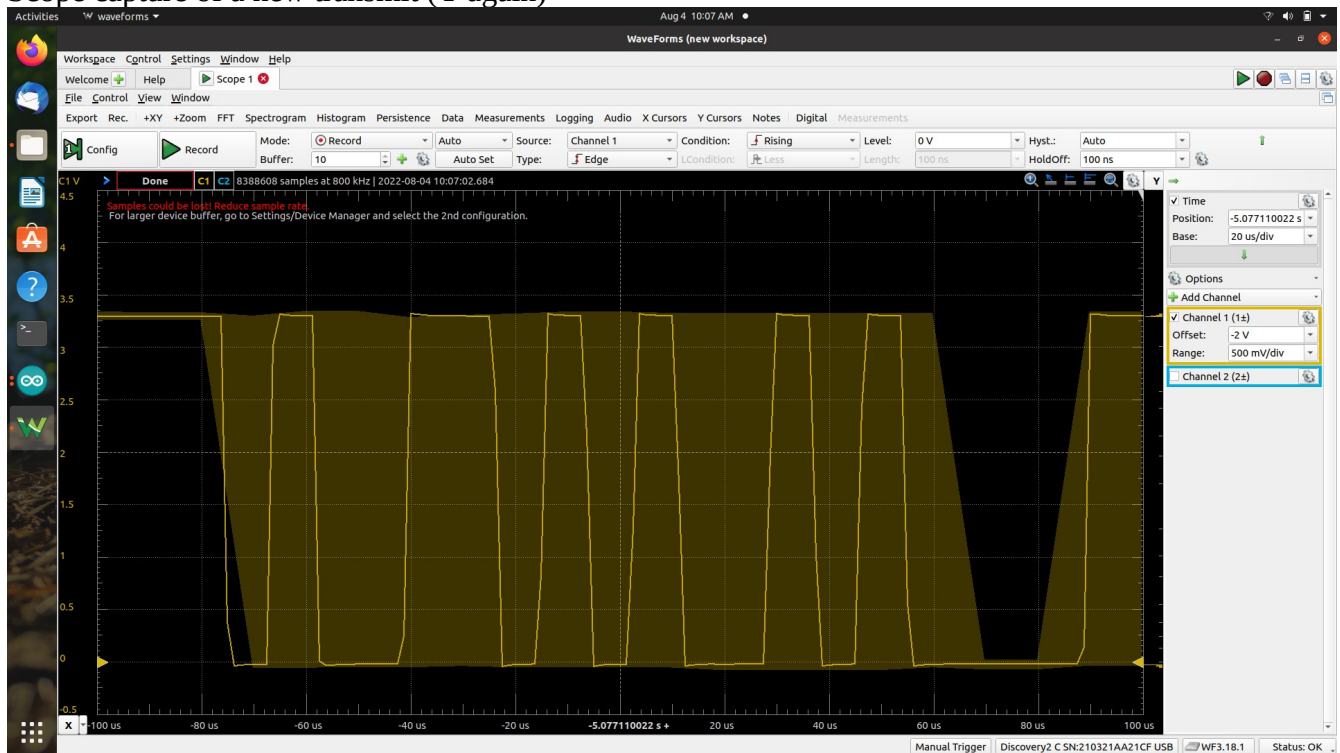
Close up of logic capture



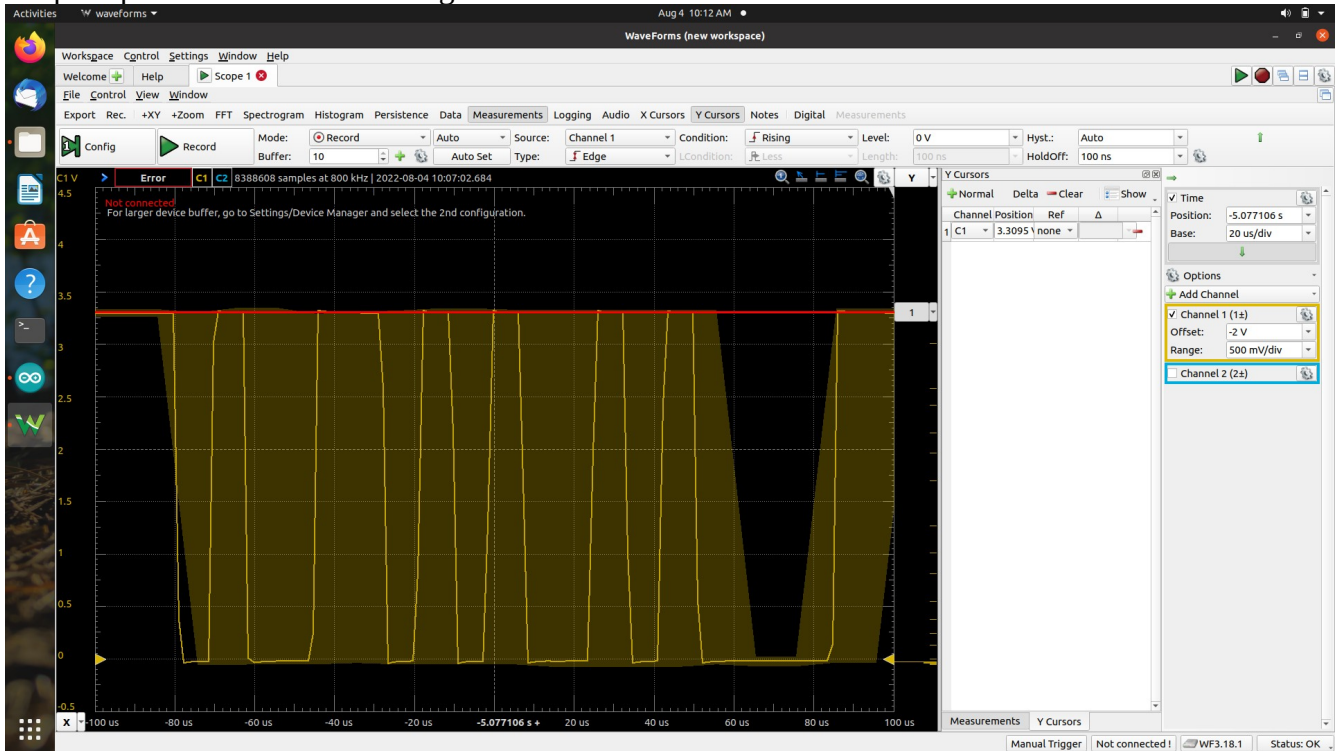
Serial Monitor showing no response



Scope capture of a new transmit (Y again)



Scope capture with cursor showing 3.3V level



Snippet of code highlighting print line that would get printed if Serial was ready (received something). “Serial Monitor showing no response” above shows that nothing gets printed, hence Serial.available is never > 0 (nothing received).

```
File Edit Sketch Tools Help
Solar_Charger calibration.cpp calibration.h config.h mppt.cpp mppt.h

inbyte i = 0;
// Set calibrating to 1
calibrating = 1;
// Init n_mppt to 0
n_mppt = 0;
}

// calibration_state_machine() function
// State machine for calibration
void calibration_state_machine() {
  switch (cur_cal_state) {
    // INIT_CAL State
    // waits for user to press enter to transition to MEAS state
    case INIT_CAL:
      // Wait for enter key (new line)
      if (Serial.available() > 0) {
        Serial.println("Serial Available");
        inbyte = Serial.read();
        // echo character
        Serial.print(inbyte);
        // 0x0A = \n 0x59 = Y
        if (inbyte == 0x0A || inbyte == 0x59) {
          cur_cal_state = MEAS;
          break;
        }
      }
    }
  }
}

Done uploading.
Sketch uses 10004 bytes (32%) of program storage space. Maximum is 30720 bytes.
Global variables use 1491 bytes (72%) of dynamic memory, leaving 557 bytes for local variables. Maximum is 2048 bytes.
99 - 100
Arduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz) on (dev/ttyUSB1)
```

Solar_Charger - calibration.cpp | Arduino 1.8.16

```

inbyte_i = 0;
// Set calibrating to 1
calibrating = 1;
// Init n_mppt to 0
n_mppt = 0;
}

////////////////////////////////////
// calibration_state_machine() function
// State machine for calibration
////////////////////////////////////
void calibration_state_machine() {
    switch (cur_cal_state) {
        //////////////////////////////////
        // INIT_CAL State
        // waits for user to press enter to transition to MEAS state
        //////////////////////////////////
        case INIT_CAL:
            // Wait for enter key (new line)
            Serial.println("Test if INIT_CAL");
            if (Serial.available() > 0) {
                Serial.println("Serial Available");
                inbyte = Serial.read();
                // echo character
                Serial.print(inbyte);
                // 0x0A = \n 0x59 = Y
                if (inbyte == 0x0A || inbyte == 0x59) {
                    cur_cal_state = MEAS;
                    break;
                }
            }
    }
}

```

Done Saving.

Sketch uses 4044 bytes (13%) of program storage space. Maximum is 30720 bytes.
Global variables use 63 bytes (3%) of dynamic memory, leaving 1985 bytes for local variables. Maximum is 2048 bytes.

98 Arduino Pro or Pro Mini, ATmega328P (3.3V, 8 Mhz) on (devttyUSB1)

The screenshot shows the Arduino IDE interface. The main editor window displays a C++ program titled "Solar Charger Calibration and Self Test Report". The program includes a header file "SolarCharger.h" and a "void setup()" function that prints a series of "Test if INIT_CAL" messages. The serial monitor window on the right shows the output of the program, which includes the same "Test if INIT_CAL" messages. The status bar at the bottom of the IDE indicates that the board is "Arduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz) on /dev/ttyUSB1".

The screenshot shows a web browser window displaying a circuit diagram for an FT232RLSSOP module. The browser's address bar shows the URL: <https://cdn.sparkfun.com/datasheets/BreakoutBoards/FTDI Basic-v22-3.3V.pdf>. The diagram illustrates the electrical connections for the module, including a USB connector, capacitors (C1, C2, C5), and a 3.3V/5V power supply. The FT232RLSSOP chip is connected to a TX0, RXI, CTS, DTR, TXLED, and RXLED. A 3.3V/5V power supply is connected to the module via a JP1 connector.

Firefox Web Browser

Aug 4 11:07 AM

Writing a Library for Arduino - x Writing a Library for Arduino - x How to Reset Arduino Pro Mini - x FTDI Basic-v22-3.3Vsch - x sparkfun arduino pro mini - x Arduino-Pro-Mini-sch - x screen com port ubuntu - x

https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/Arduino-Pro-Mini-v14.pdf

1 of 1

Power isolation jumper (for low power applications)

UCC = 5V or 3.3V Output
Max Voltage Inputs 16VDC
Max Current Outputs 150mA

Board is marked with combination of resonator frequency and regulator voltage.

Optional Pullups for I/O lines

ATMEGA328P

Pinout table:

Pin	Function
1	PC0(ADC0)
2	PC1(ADC1)
3	PC2(ADC2)
4	PC3(ADC3)
5	PC4(ADC4/SDA)
6	PC5(ADC5/SCL)
7	AD06
8	AD07
9	PD0(RXD)
10	PD1(TXD)
11	PD2(INT1)
12	PD3(INT1)
13	PD4(AOKT0)
14	PD5(T1)
15	PD6(AIN0)
16	PD7(AIN1)
17	PB0(VCC)
18	PB1(OC1A)
19	PB2(SS/OC1B)
20	PB3(MOSI/OC2)
21	PB4(MISO)
22	PB5(SCK)

Header pinout:

Pin	Function
1	GND
2	CTB
3	VCC
4	TXD
5	RXD
6	DTB

FTDI Basic

Off Grid Breakout headers

JP2

JP3

open hardware

Released under the Creative Commons Attribution Share-Alike 3.0 License
http://creativecommons.org/licenses/by-sa/3.0

TITLE: Arduino-Pro-Mini

SEP