

# Shinyei Dust Sensor Air Quality Monitoring

## With TI Launchpad MSP430G2

**Objective:** To know when I need to clean my room. I usually vacuum every two or three weeks, but sometimes am lazy and I need a reminder. The dust sensor will be more motivation to get me to clean, when I see the dust levels increase.

### Table of Contents:

1. Parts List
2. Work Log

### Parts List:

- Shinyei PPD42NS Air Quality Sensor
- MSP-EXP430G2553 V1.5
- 5V Adaptor (required as MSP is 3.3V only)
- SD Card adaptor
- Ethernet Adaptor
- (optional) 16x2 LCD



Figure 1: Shinyei PPD42NS Air Quality

### Work Log

Here are some things I've learned as I've worked on this project.

#### IDE

I'm using Energia from [energia.nu](http://energia.nu) which is an Arduino IDE clone for TI Launchpads. This will allow for rapid development. And ease me into the TI platform.

#### MSP-EXP430G2 V1.5

There are different version of this. My particular board is the and as explained here: [http://energia.nu/pin-maps/guide\\_msp430g2launchpad/](http://energia.nu/pin-maps/guide_msp430g2launchpad/) and [http://energia.nu/pin-maps/guide\\_msp430g2launchpad/](http://energia.nu/pin-maps/guide_msp430g2launchpad/)

You need to rotate the UART jumpers to get UART to display correctly. Oddly enough you can choose a different chip in the boards list and serial will print out right... But switch the jumpers and the correct board will work with UART. Otherwise, it does not work out of the box for serial.print. Though the blink example sketch works.

#### Dust Sensor

The code for the dust sensor is found online easily. The pinout is tricky as the colours of the wire are nonsense, but the pinout seems to be the same for all sensors, and is: PIN 1 (closest to black box) GND, PIN 3, VDD (5+V), PIN 4 output A. there is also an output B for different readings (I think size) of dust. I'm not that particular (no pun intended) about my dust so I will go with the one most people are using.

3.3V only on TI

The TI takes input of USB but only outputs 3.3V. Fail. I'll through a 5V PSU on the board. It won't be connected to USB for its use anyways.

POW function

There is a pow function (power exponent) in arduino. In TI, I changed it to powf, and included math.h.

I'm not sure if my change was correct.

EDIT: looks like there is a LED tied to pin 14. I'm moving to pin 13. I meant to use 13 earlier but accidentally used the wrong pin and kept using it. Oops.

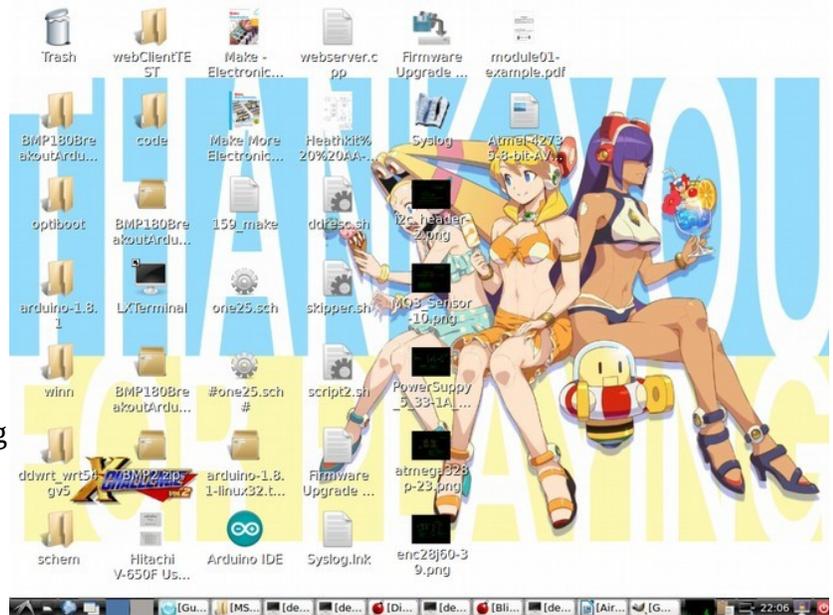
EDIT 2: Actually, I was reading from the wrong pin in software. Maybe I will leave it on the LED pin as you can see when the dust is firing off the sensor. Hm.... Neat.

EDIT 3: Yes, so I had to use powf instead of pow, and it works. The accidental incorrect pin was a bonus as it allows me to see visually how often the sensor is going off. Over time, I should have a vague grasp of the dust levels just looking at the light.

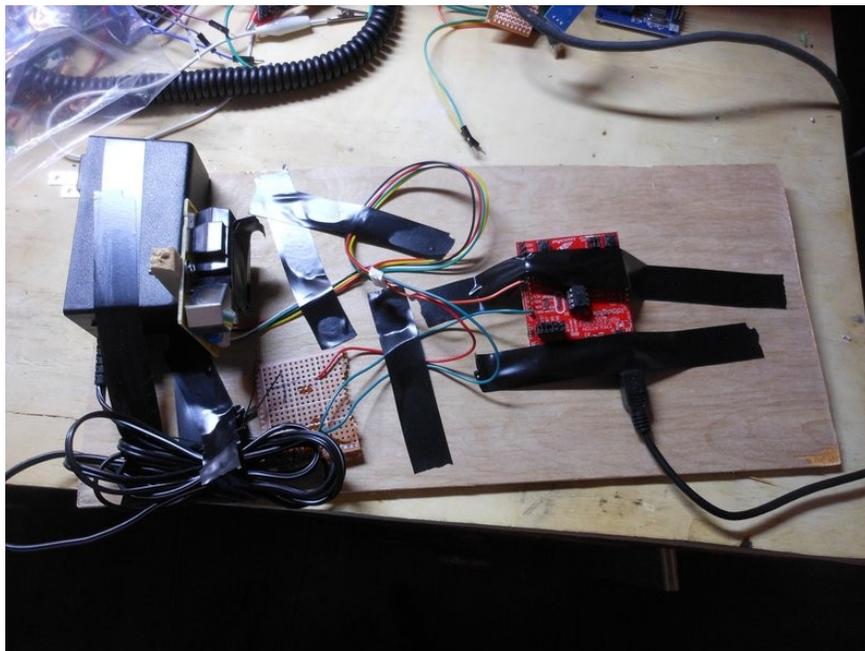
### Work Log 08/2018

I've put everything on a single piece of plywood, and need to do some more work on the code. I've decided to forgo the SD card, as I don't want to deal with reading a 2MB sd card with the buffer provided by the SRAM. I'm not quite sure how to manage that, and whether it's possible to read such large data files. I did some quick research and did not find what I wanted. It is likely possible, but let's do something simpler. Instead, I will use the EEPROM to store the last 5 minutes of data or so, and then have the server read the data every five minutes. This keeps the client simple, and puts the burden of complexity on the server. In addition, I don't want to read SD cards manually, as that is cumbersome, though possibly scripts could be made.

The PPD42 was made sure to be put vertically.

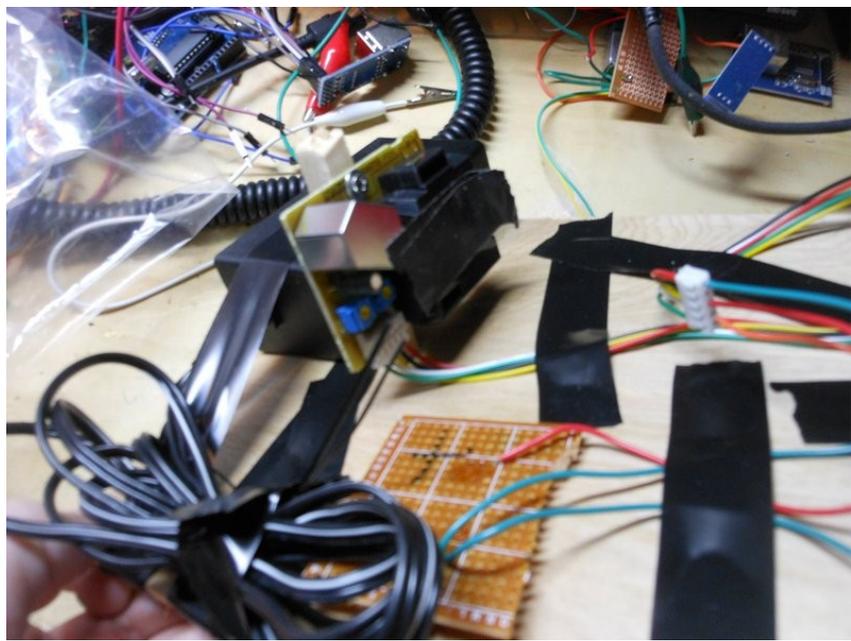


*This is an aside, but I realized today that happy pictures make me happy, so I should surround myself with pleasant things. I often keep a blank background, but I realized, that a cute picture will make me happier than a blank or default background. It's the little things. I digress.*



### EDIT: 12/2018

I have decided to change how I do this slightly. Instead of an SD card, I will connect on the LAN and use thingspeak from a locally hosted instance (and deployed with docker, possibly) or some other aggregating server to pull the data. I'll also make a shield, to simplify deployment. That might not be necessary, but I can make a shield in a few hours, and pcbs are cheap. These options such as thingspeak sometimes have graphing / plotting included.



I'm going to use this library:

<https://github.com/reaper7/EtherEncLib/releases>

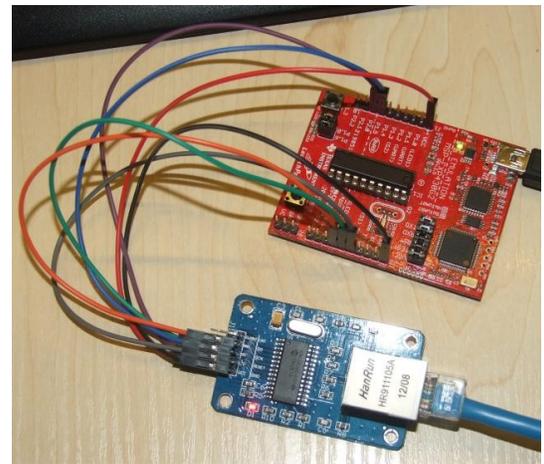
so git clone that, then

```
git tags -l
```

```
git checkout tags/v0.4.2
```

to get the latest release (or a newer one if possible).

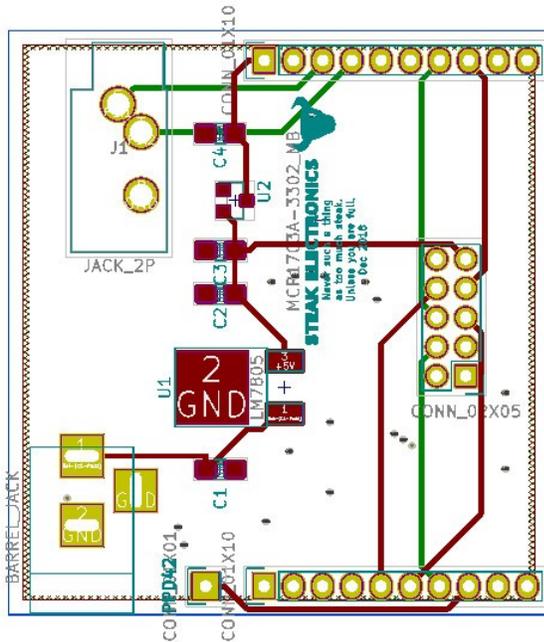
The pinout for the ENC is viewable at the figure to the right. This is from the 43oh.com forum.



*ENC and MSP430 (Note: Works with only two confirmed MSP models).*

### PCB Layout

layout is simple for the most part. I again, flipped the ENC as I had done on the Uno board, so it is inserted upside down on the board. This time I went only with the 2x5 pin enc for simplicity sake. The shield is below the MSP, so longer pin headers will be used (already have those) to give space for barrel plug. The board needs a 3.3v regulator and 5v for the PPD42. To get the sizing of the shield right, I aligned my grid with that of the design files for the msp430 dev board, and made sure the spacing between the 0.1" headers was exactly the same – easy. Ran all traces of 20 mils and made the board small as reasonably possible. Now to get them made, and actually test this.



Spacing for the headers is board perfect, by aligning my grid, with the MSP430's design files grid (eagle), and making them align. FOSS+OSH win.

*Layout + schem done in about 60-90 minutes. Easy.*

### Revision 1 Results:

I built up the PCB today. A few errors were made, but this is much better than the mess I had on the plywood originally. First off, the ENC pins are wrong, so that is not going to be able to be mounted (unless I wanted to wire pins manually). Second, the orientation of the MCP part is wrong. I'm not sure how this was done incorrectly, but I suspect that the default KiCad libraries were wrong. I have a few other changes to do, but this was overall successful, and I can respin another board quick quickly. On my todo list is:

- add gnd breakout
- add 5v breakout
- add 3v3 to enc NOT 5v
- fix enc pins
- fix orientation of main board
- add anime picture to back of board
- fix orientation of 3v3 reg
- buy more msps (need two more at least)
- buy 100 1uf 0805 caps put in main box (I'm surprised I don't keep these in stock. I do have through hole 1uf but not 0805).
- buylist + pcb : header so you can just plug in ppd. This is a sort of wish list thing, but something that will save a few seconds on build time. Although NOT necessary.

