How to Guide - Water Sensor via A2D

CMPT 433 D100 - Beagle-Gotchi

This manual guides you through the following

1. Determine & Understand your Sensor
    1.1. Reading SMD Resistors
2. Connect to your Sensor
3. Interpret & Calibrate your Sensor
4. Troubleshooting

Required Materials

- BeagleBone Green (BBG)
    - an unused ground, 3.3V power (min), and analog input (AIN) pin
- 3 Jumper cables
- A simple Water Sensor (check Adafruit or HAOYU Electronics)

1. Determine & Understand Your Sensor

This is a very simple device composed of a single transistor, LED, and some resistors. When water comes in contact with the metal bars & connects two of them, it causes a controlled short which we can detect via the BBG's A2D sensors. The output voltage varies a little with respect to the amount of water, but it is hard to detect given the noise.

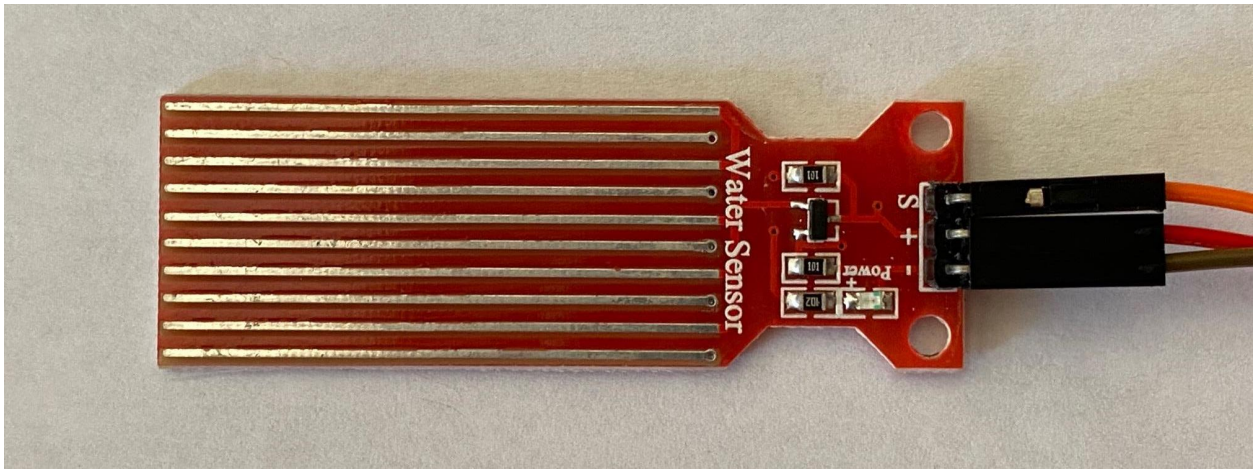

Fig 1. Front of device

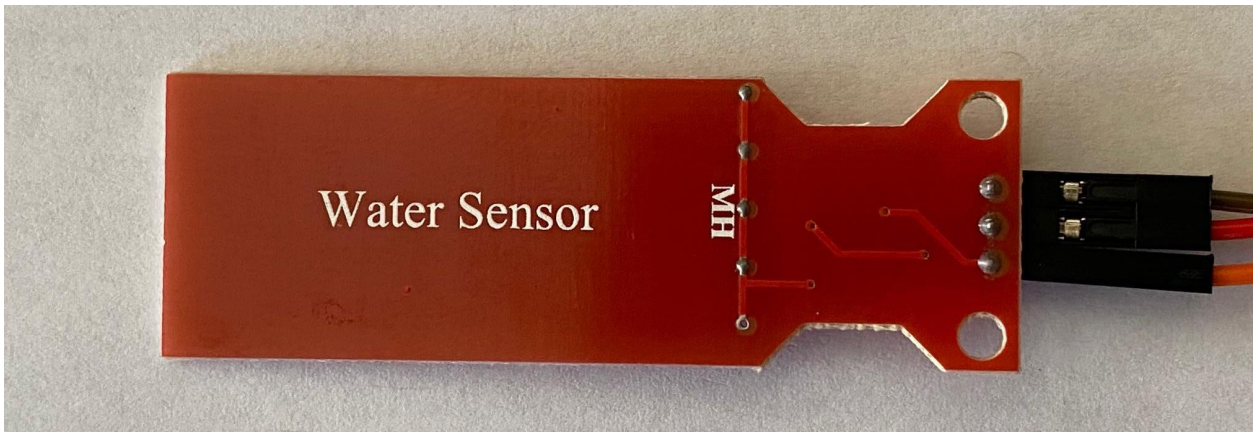

Fig 2. Back of device

The orange wire connecting to [S] is the pin we'll read the analog voltage from, while the red and black wires connecting to [+] and [-] are for power and ground, respectively. This device is rated for between 3V and 5V, so it's compatible with the BBG.

Various companies make this exact same sensor, but with slightly different parts. I obtained mine from Adafruit via https://www.adafruit.com/product/4965. However, one might also come across a separate version from HAOYU Electronics via https://www.hotmcu.com/water-level-sensor-liquid-water-droplet-depth-detection-p-113.html that will likely behave exactly the same due to how basic the components are.

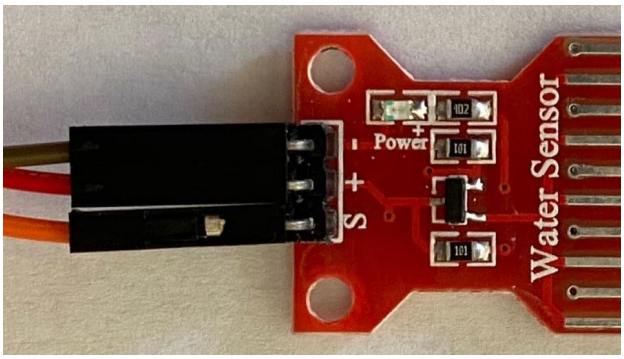

Fig 3. Zoomed in front

Lets dive a little deeper into the components this is made up of. If we zoom into the sensor, we'll notice that the individual resistors actually have 3 small numbers printed on them. These are SMD (Surface Mounted Device) resistors and we read them differently than the coloured bands on traditional through-hole resistors. It's actually somewhat similar to IEEE floating point numbers.

1.1 Reading SMD Resistors

When reading resistors of the form [ABC] where A, B, and C are digits we use the following rules:

- The first two digits to the left represent the significant digits of the resistance.
- The third number represents how many times to multiply 10 to the significant digits.

Put more simply, we can represent the resistance of an SMD resistor with 3 digits as the following function:

$$resistance(A,B,C) = AB * 10^{C}$$

Ex: In the image above [102] and [101] correspond to 1k Ohms and 100 Ohms respectively.

There are more complicated SMD Resistor patterns that consist of certain letters or more digits, but we are not interested in those for now. See https://www.hobby-hour.com/electronics/smdcalc.php for more details.

2. Connect to your Sensor

This sensor has 3 pins. These are from left to right when looking at the front of the device:

1. [S] Analog Voltage. We will wire this to p9_35 (AIN6), but you may change this as necessary to avoid collisions. See the A2DGuide on the course website for more info on valid pins.
2. [+] Power. We will wire this to p9_32.
3. [-] Ground. We will wire this to p9_1, but you can connect it to any open ground.

Since this is a simple analog device, we can follow the guide from course notes here https://youtu.be/uZNgaTA38w0 in order to wire the device to A2D. However, there is one small difference, since we already have resistors in the chip we don't need to hook up a resistor on the ground wire; we can simply connect each pin directly into the BBG. Much nicer!

3. Interpret & Calibrate your Sensor

For sensing we can follow the A2D guide from the course notes here https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt433/guides/files/A2DGuide.pdf to get setup & read a few sample voltages.

For example, you should be able to test the device by running the following command on the BBG:

$ cat /sys/bus/iio/devices/iio:device0/in_voltage6

Similar to with the light sensor, the sensor will not report zero even when outside of water (due the minimal conductivity of the air). For this reason it's important to determine a lower bound so that you don't accidentally get false positives. I found that 1024 works pretty well in this case.

One thing to keep in mind is that even once all the water is removed from the sensor, it will still produce high values for several seconds. This is likely due to residual moisture in the air that the sensor is too delicate to ignore.

The following is a short C function that will return true if the water sensor is currently touching water:

```
#include <stdbool.h>
bool isSubmerged() {
   // replace N in in_voltageN to the number of the AIN pin you're using
   FILE* fp = fopen("/sys/bus/iio/devices/iio:device0/in_voltage6", "r");
   if (fp == NULL) { printf("Error reading file\n"); exit(-1); }

   const int MAX_LINE_LEN = 32;
   char fileBuf[MAX_LINE_LEN];
   fgets(fileBuf, MAX_LINE_LEN, fp);
   fclose(fp);

   const int WETNESS_LIMIT = 4096/4; // lower bound voltage for wetness
   return atoi(fileBuf) > WETNESS_LIMIT;
}
```

4. Troubleshooting

4.1 Unable to read values from the BBG's A2D

Once the device is attached to power, the red LED in the top right (near the text that says power) should turn on. If this is not happening, double check that you've plugged the wire from [+] into the correct port on the BBG. You can find more info about cape expansion headers in the following link https://beagleboard.org/support/bone101

In addition, make sure that the device's [-] pin is attached to ground. In the pin header diagrams ground pins are notated with DGND.

4.2 Why do non-liquid substances set off the sensor?

As a side-effect of how the sensor is implemented, any object that conducts electricity may set off the sensor. However, objects such as paper do a poor job at conducting and can sometimes

4.3 I'm getting false positives!

Make sure that you have defined a sufficiently large lower bound for the voltage, or have some other form of noise control. Taking the geometric average, or the average of the last n samples can help in these cases.

Debouncing is not a large problem with this sensor, since it takes a while for water to be removed, and typically evaporation produces a clean curve.