ANT: PH Sensor Testing Report

By Raymond Duong, ANT Computer Engineer

Last Updated: October 26, 2021

In this report, multiple solutions for ANT's ph sensor will be tested. All results for the tests will be recorded and compared.

Note: All tests will be conducted at room temperature so the following results are most accurate in temperature range (18°C to 25°C).

Table of Contents

Introduction

How does a PH sensor work?

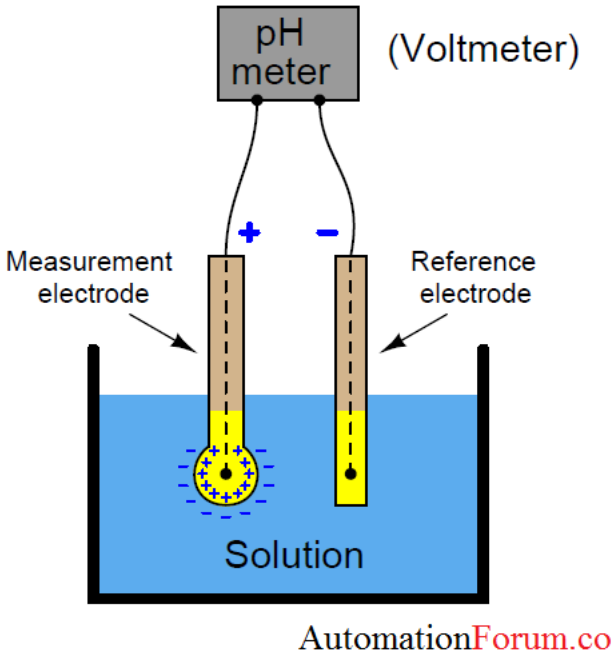


Photo Diagram of PH Sensor By Automation Forum

The working of a PH sensor is somewhat like a voltmeter. To obtain PH reading from the PH sensor, we need to read a small voltage value from the probe itself which occurs when there is voltage potential. Inner working of a glass PH probe is as follows.

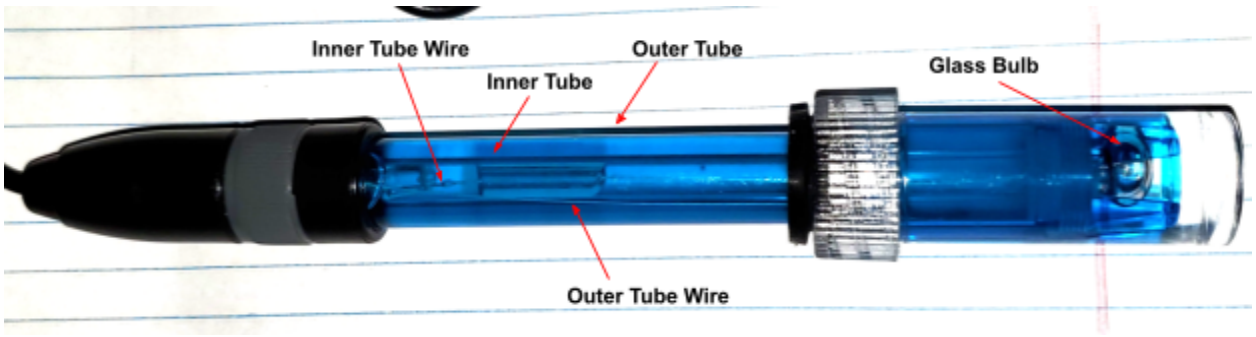


Photo Of Grove PH Sensor

A glass ph probe is composed of two enclosed glass tubes with a metal wire in each of them. The inner glass tube is the glass electrode portion of the probe with a hydrogen ion sensitive glass bulb at its end which interacts with the testing solution (ph solutions you want to measure). The outer glass tube is called the reference electrode of the probe in which it serves as the reference potential of the readings. Both tubes are filled with a neutral potassium chloride solution(KCL) buffered at PH 7.

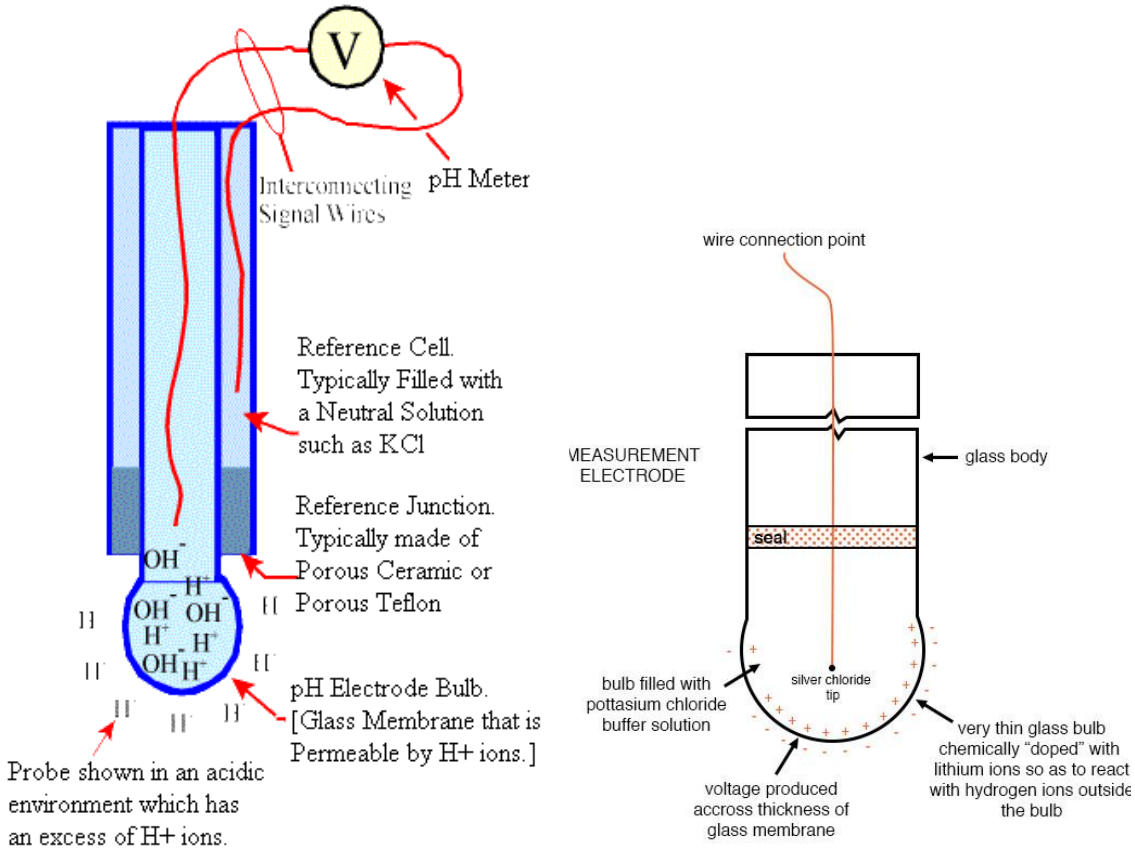


Diagram from Digital Analysis Corporation(left) and All About Circuit(right)

When the PH probe's glass bulb is submerged in an aqueous solution, the glass bulb's external surface reacts with the hydrogen ions of the external solution. Hydrogen ions penetrate the external membrane of the H+ sensitive bulb creating a layer called the gel layer. This also happens internally in the glass bulb with the neutral KCL buffered at PH 7 with a set number of hydrogen ions. This difference in hydrogen ion concentration between external and internal

membrane creates an electrical potential. Finally, the difference between the potential obtained at the glass bulb and the reference electrode gives us the voltage reading we need for ph calculations.

Why do you need a PH sensor in the ANT system?

The PH sensor is an important environmental sensing device in our system, because its reading helps automate the ph balancing. Depending on the ph readings from this sensor, the ANT system determines whether to not turn on the ph up or ph down dosing pumps. Ex. If our desired PH range is between PH 5.5 and PH 7, and our pH reading from our sensor reads 5, the pump to dose PH up will be turned on until ph reaches our desired range again.

PH Sensor: Raspberry PI and Arduino USB Serial communication

Goal:

Use USB serial communication between Raspberry PI and Arduino to send Ph sensor data from Arduino to Raspberry. The Raspberry PI will take data from the Arduino via serial, process the data to get the ph reading values, and finally print the ph value on the terminal.

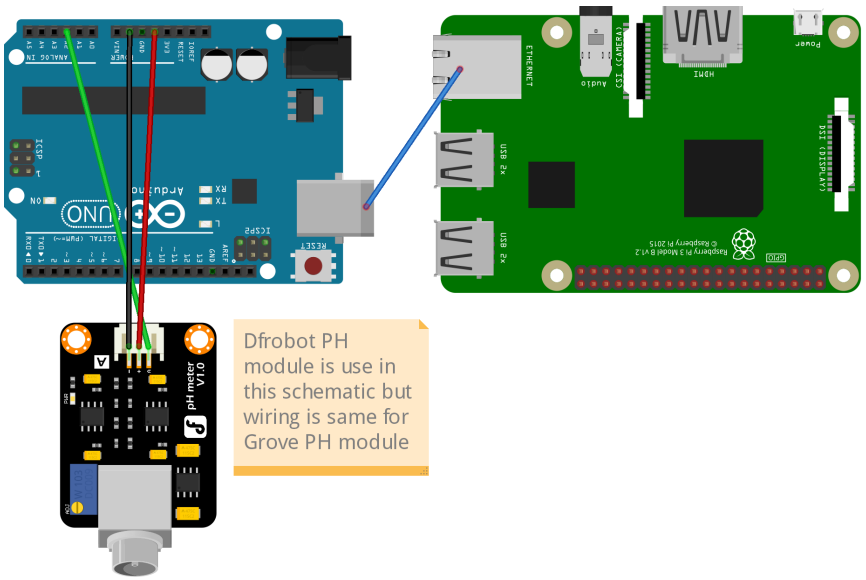
*Expected result is that a ph reading is printed and max percent error of 5% or less.

Materials:

- Raspberry PI 3B+
- Raspberry Pi Micro Usb power supply(output:5V ~ 3000mA)
- Arduino Uno
- Arduino Uno Power supply(output: 9V ~ 1000mA)
- Grove PH Sensor
- Arduino Serial to USB cable
- Jumper Wires
- PH Calibration Liquid (Preferably PH4 and PH7).

*PH calibration liquids used are PH4 and PH 7 by Atlas Scientific. Can be ordered here.

Schematic:

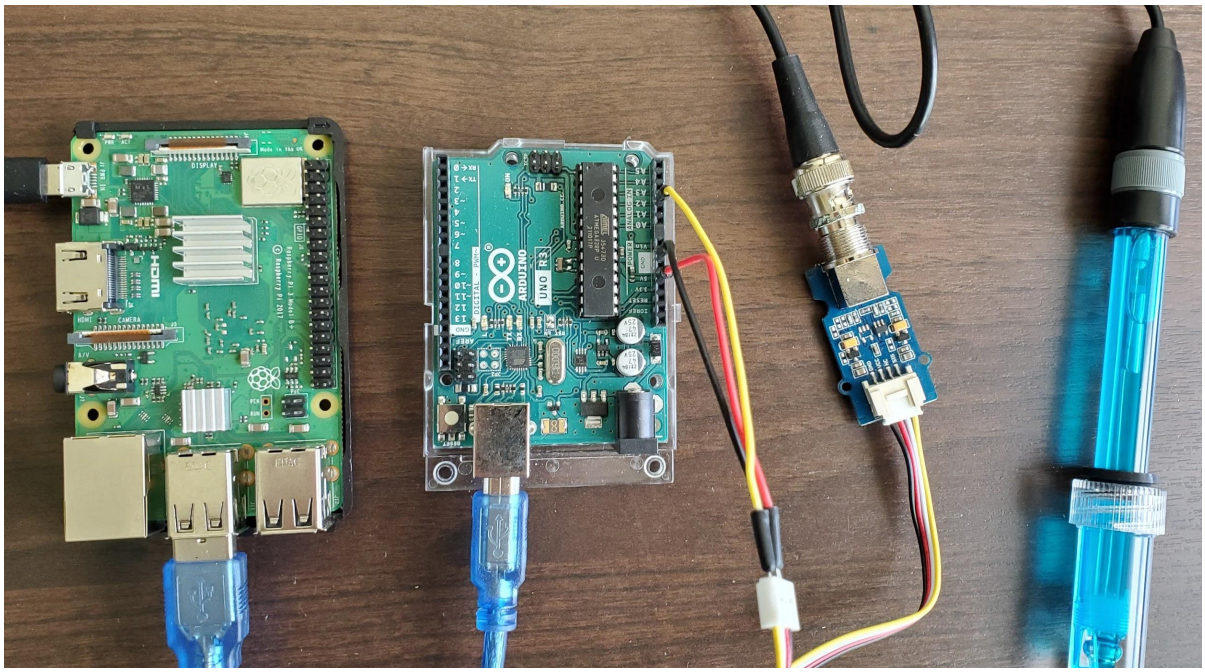


*** Red - Arduino 5V to PH module positive pin**

***Black - Arduino GND to PH module negative pin**

***Green - Arduino A2 to PH module analog signal pin**

***Blue - Arduino to Raspberry PI USB Serial cable connection**



RPI/Arduino USB Serial Connection(BLK -GND, RED - 5V, YELLOW - A2)

Procedure:

1. Upload the demo code for Grove PH sensor from Seeed's website as a sketch into the Arduino. Demo code can be found here.
2. Modify demo code. Remove or comment out "uart.println("pH meter experiment!");" and print outputs, voltage and ph, in the format: "voltage:xx ph:xx".

```
if (millis() - printTime > printInterval)   //
{
  uart.print("Voltage:");
  uart.print(voltage,2);
  uart.print(" pH:");
  uart.println(pHValue, 2);
  digitalWrite(LED, digitalRead(LED) ^ 1);
  printTime = millis();
}
```

3. Connect Raspberry PI, Arduino, and PH sensor module as shown above.

   ***Remember to enable I2C protocol on RPI.**

4. Create file ph_serial_sensor.py file in Raspberry PI and input followingRaspberry PI code:

```
import serial
import re
import sys
import os

#opening serial port
ser=serial.Serial('/dev/ttyACM0', 9600)
#Here /dev/ttyACM0 used
#Find the right usb interface for your device using 'ls /dev/tty*'

while True:
    try:
        serialdata = str(ser.readline()).split(' ') #split line read by space

        if (len(serialdata) > 1):

            pattern = re.compile(r"[\d]+[.]?[\d]+") #real expression: Find numerical value in form "xx.xx"

            try:
                voltage_value = float(pattern.search(serialdata[0]).group()) #apply real expression to serial data
                ph_value = float(pattern.search(serialdata[1]).group())
                os.system('clear')
                print(f'Voltage:{voltage_value} PH:{ph_value}')
            except AttributeError as e: #if real expression return None
                print(e)

    except KeyboardInterrupt:
        print('End of ph sensing')
        sys.exit()#exit program
```

5. Run ph_serial_sensor.py using command “python3 ph_serial_sensor.py”.
6. PH sensor calibrations:

   Test two PH calibration liquids and write down the ph/voltage readings. Calculate the k and Offset values using the equations provided by Seeed.

   k= (PH2-PH1)/(V2-V1)

   Offset=[(PH2+PH1)-k*(V1+V2)]/2

   Modify the k value and Offset in the Arduino code. Reupload Arduino sketch.

```
#define Offset 42.56

pHValue = -19.185* voltage + Offset;
```

k value

Results:

K Value	Offset	PH Calibration Liquid	PH Reading	Voltage	Error %
-17.64705882	38.4117647	PH7	7.05	1.78V	0.714%
-17.64705882	38.4117647	PH4	3.98	1.95V	0.5%
-18.05882352	39.19470588	PH7	7.1	1.78V	1.4285%
-18.05882352	39.19470588	PH4	4.01	1.95V	0.25%
-17.64705882	38.4117647	Baking Soda Solution (Expected:8.3)	8.04	1.72	0.12%
-17.64705882	38.4117647	White vinegar (Expected:2.5)	2.4	2.04	4%

* PH readings may vary due to temperature

* Percent error = |(Vactual - Vexpected) / Vexpected| * 100

K and Offset Calculation:

K = (4 - 7)/(1.95 - 1.78) = -17.647058823529411764705882352941

Offset = [(4 + 7) - k(1.78 + 1.95)]/2 = 38.411764705882352941176470588235

K = (3.98 - 7.05)/(1.95 - 1.78) = -18.058823529411764705882352941176

Offset = [(3.98 + 7.05) - k(1.78 + 1.95)]/2 = 39.194705882352941176470588235293

The best K value and Offset from the table above are -17.6470588 and 38.4117647 since the pair have overall lowest percent error. Compared to the expected max error percentage, 5%, our best error percentages are well below one percent.

In terms of this ph sensor solution, Raspberry PI and Arduino serial connection have relatively accurate ph readings(<1% error). The problem with this method is that power usage by the Raspberry PI affects the value read from the Arduino, especially when with a poor power supply (ie.Low voltage error). This is also true when the Arduino is not connected to a power supply, affecting the ph value. Of course when this happens, we could modify the k and offset, but the voltage might fluctuate a lot due to lack of enough power. In conclusion, a good power source is highly recommended for this method (RPI power supply with at least 3000mA and Arduino power supply).

Arduino power supply disconnected:

K Value	Offset	PH Calibration Liquid	PH Reading	Voltage	Error %
-18.0588235 2	39.19470588	PH7	6.71~ 6.81	1.79 ~ 1.8	2.714% ~ 4.142%

PH Sensor: Raspberry PI and Grove PI Hat

Goal:

Use Grove Base Hat's built in 12 bit 8 channel Analog to Digital Converter(ADC) with Raspberry PI to read analog data from the PH sensor. Since the Raspberry PI has no built-in ADC, an external ADC is necessary for Raspberry PI to read analog data directly which the PI Hat can provide.

* Grove Base PI Hat also feature Digital/Analog/I2C/PWM/UART ports

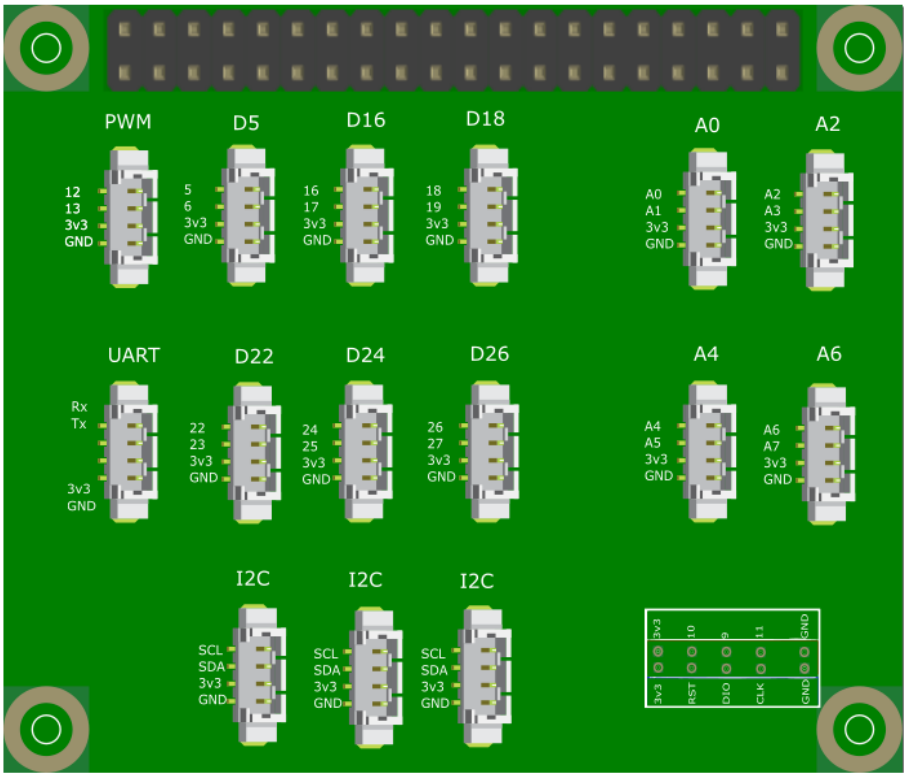
*Expected result is that a ph reading is printed and percent error 5% or less.

Materials:

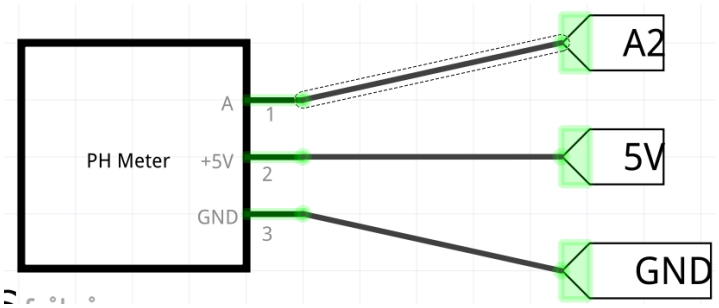
- Raspberry PI 3B+
- Grove PH Sensor
- Grove PI Hat
- Jumper Wires
- PH Calibration Liquid (Preferably PH4 and PH7).

*PH calibration liquids used are PH4 and PH 7 by Atlas Scientific. Can be ordered here.

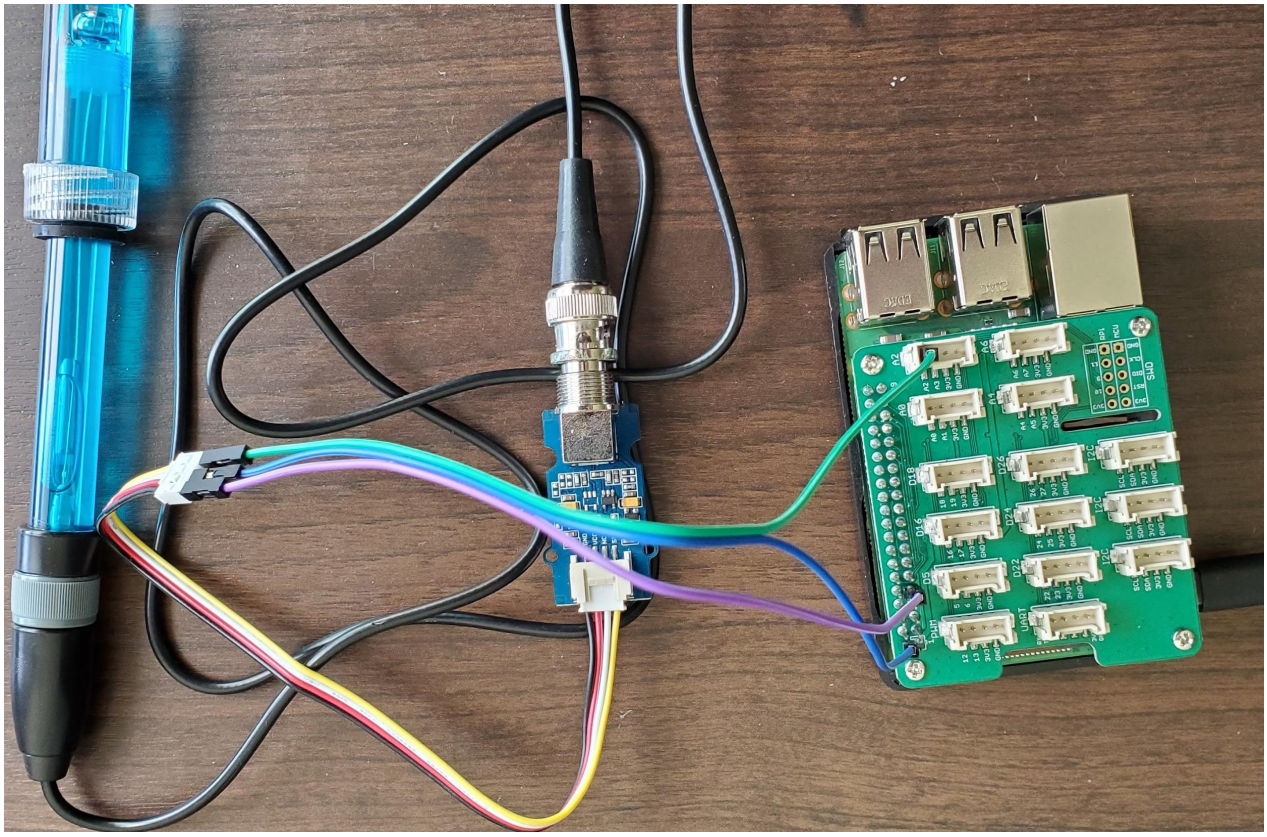
Schematic:



Grove Base HAT For Raspberry PI



Circuit Connection



Grove HAT Actual Connection Photo(BLU - 5V, PUR - GND, GRN - A2)

Procedure:

1. Install the Grove PI Base Hat on the Raspberry PI and power on the Raspberry Pi.

   Connect the Grove Sensor as shown in the schematic. Enable I2C on RPI.

   **Careful when attaching the Hat if you have a thick heatsink on Raspberry PI. Do not let the heatsink and board touch.

2. For one click install of Grove Base Hat libraries:

   Enter into the linux terminal “curl -sL https://github.com/Seeed-Studio/grove.py/raw/master/install.sh | sudo bash -s -”.

   **Not recommended since may need to modify i2c address in code

3. For step by step install of Grove Base Hat libraries:

```
git clone https://github.com/Seeed-Studio/grove.py
cd grove.py
# Python2
sudo pip install .
# Python3
sudo pip3 install .
```

4. The above steps are also available on Seeed's Wiki page here and on their Github here.

5. ***Important step* I2C address**

   Enter "sudo i2cdetect -y 1" to get the i2c address of the PI Hat

```
pi@raspberrypi:~/grove.py $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- 08 -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

   Change directory to the grove folder in grove.py. Enter "cd $HOME/grove.py/grove".

   Open adc_8chan_12bit.py and change the address to your i2c address.

```
ADC_DEFAULT_IIC_ADDR = 0X08
```

   adc.py does the same thing, so if you want to use adc module instead. Change the address in that file.

```
RPI_HAT_PID      = 0x0008
```

```
def __init__(self, address = 0x08):
```

6. Reinstall the Grove libraries after changing the addresses

```
cd grove.py
# Python2
sudo pip install .
# Python3
sudo pip3 install .
```

7. Create file ph_pihat.py and copy the code below. Comment out “print ("\33[2A")” when calibrating to see values better.

```
from grove.adc_8chan_12bit import Pi_hat_adc
import time
import sys

#ADC object
adc = Pi_hat_adc()

while True:
    try:
        voltage = (adc.get_nchan_adc_raw_data(2) * 3.3)/4096 #get voltage - ADC data * supply voltage(3.3V) / 4096 since 12bit ADC
        voltage_mv = voltage * 1000
        ph = 7 + (voltage_mv - 2415) * -0.024 #pHx = pH1 + (Ex – E1)(pH2 – pH1)/(E2-E1)
        print(f"Voltage:%.2f PH:%.2f"%(voltage,ph))
        print ("\33[2A") #Move cursor up 2 line
        time.sleep(1)

    except KeyboardInterrupt:
        print("Ph sensing ended")
        sys.exit()
```

8. PH sensor calibrations:

   Test two PH calibration liquids and write down the voltage readings. Calculate the ph formula using the equation below. It’s recommended in ph calculation steps by Cornel Radu here.

   $$pHx = pH1 + (Ex - E1)(pH2 - pH1)/(E2-E1)$$

   *E1 and E2 are the voltage in millivolts read by the sensor while ph1 and ph2 are the PH level of the calibration liquid

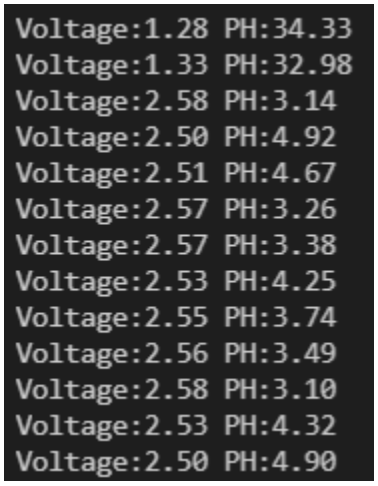
Results:

PH Formula Calculations:

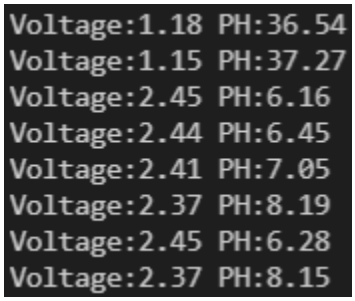
ph = 7 + (Ex - 2415) * (4 - 7)/ (2540 - 2415) = 7 + (Ex - 2415) * (-0.024)

PH Calibration Liquid	PH Reading(Most favorable)	Voltage	Percent Error
4	4.25	2.53V	6.25%
7	7.05	2.41V	0.714%

PH 4 solution:



PH7 solution:



The resulting readings using the Grove PI Base Hat were not favorable, because the voltage fluctuate a lot during testing affect the ph values by a lot as shown in the two pictures above. In the table above, ph values of the most favorable were recorded since the values can vary a lot

between each reading. Results were the same whether using adc_8chan_12bit.py or adc.py thus further testing was not needed. No concrete ph values can be recorded.

PH Sensor: Raspberry PI and Waveshare High Precision AD/DA Board

Goal:

Waveshare's high precision AD/DA board has a built-in 8 channel 24 bit ADC. With a higher bit precision of 24 bit, twice more than the Grove Base Hat, we should expect more accurate sensor reading. Also, since this board only features analog and digital signal conversion, problems should be easier to isolate.

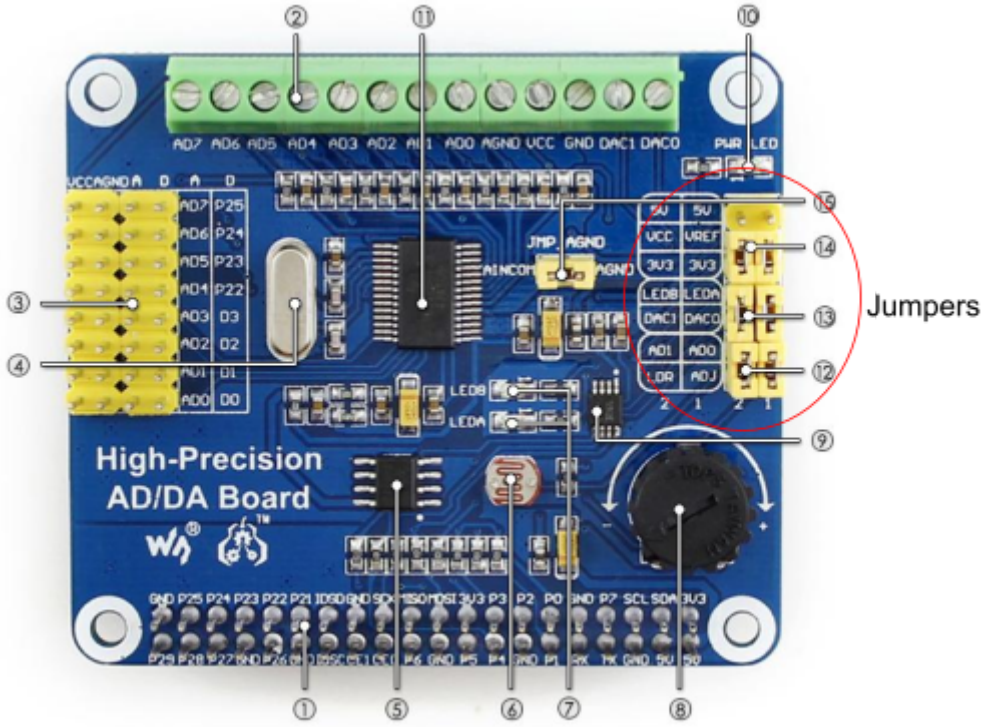
*Expected result is that a ph reading is printed and margin of ph error offset is 0.5 or less

Materials:

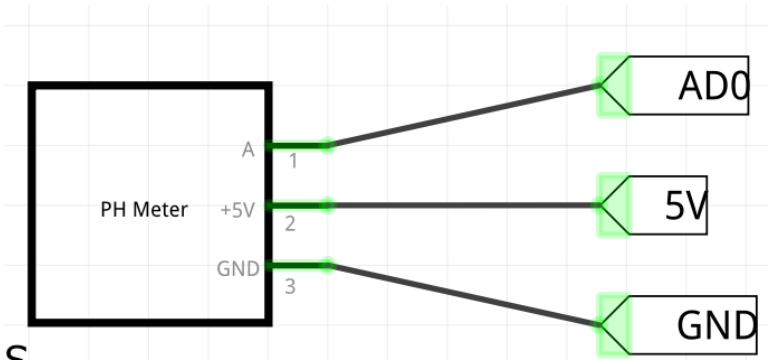
- Raspberry PI 3B+
- Waveshare High Precision AD/DA Board
- Grove PH Sensor
- Jumper Wires
- PH Calibration Liquid (Preferably PH4 and PH7).

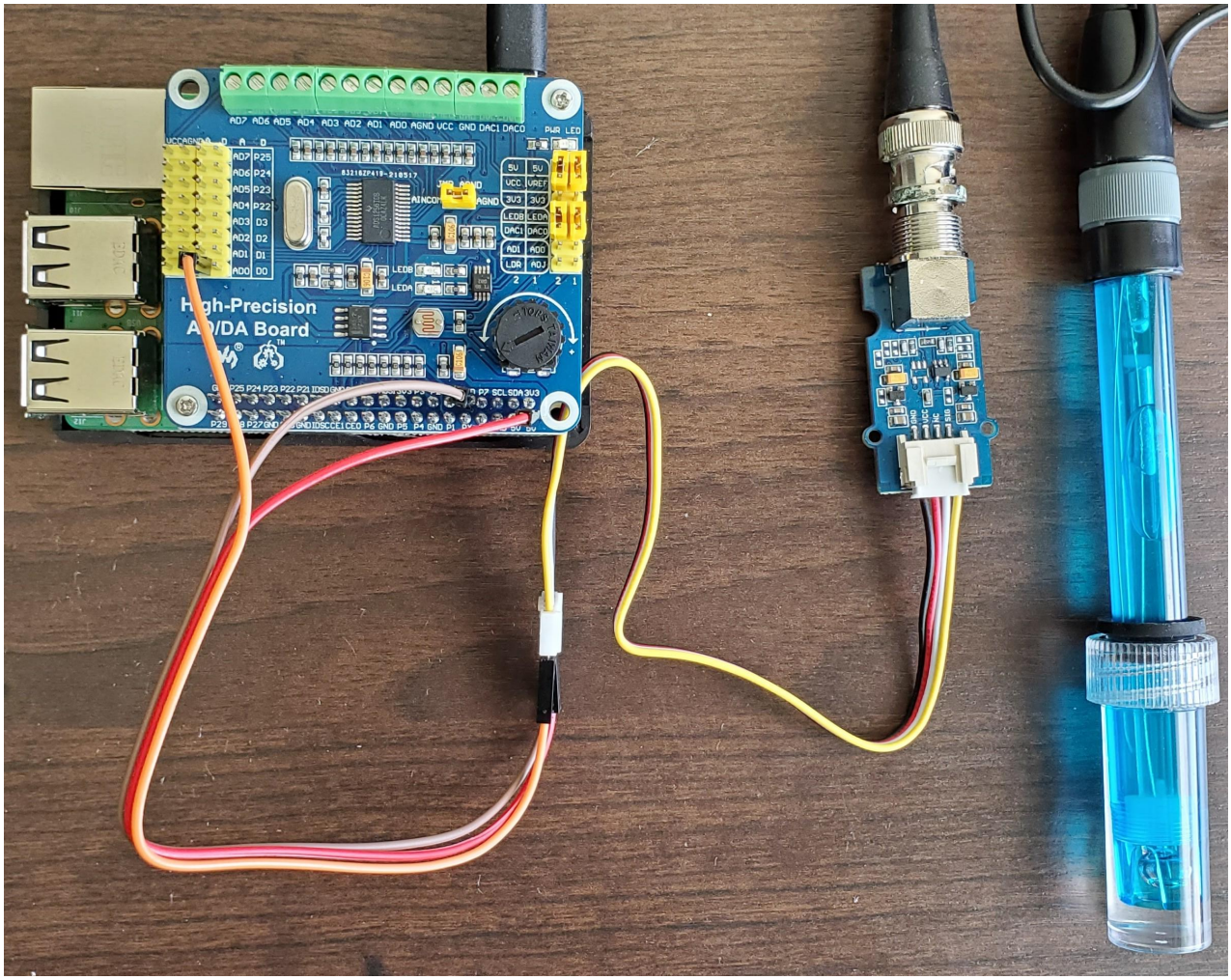
*PH calibration liquids used are PH4 and PH 7 by Atlas Scientific. Can be ordered here.

Schematic:



Waveshare High Precision AD/DA Board





Waveshare AD/DA Board Connection Photo (RED - 5V, BRN - GND, ORG - AD0)

Procedure:

1. Install the Waveshare High Precision AD/DA board on the Raspberry PI and power on the Raspberry PI. Connect the Grove Sensor as shown in the schematic
2. Jumper Configurations:

   Bridge VCC and VREF with 5V

   ***Remember to disconnect AD0 jumper with ADJ.**

***The bottom four jumpers are for testing AD0, AD1,DAC0, DAC1 respectively during demo, so remove them when you use these pins for actual use/reading with the sensor.**

3. Waveshare High Precision AD/DA board works with bcm2835-1.39 or bcm2835-1.48 libraries which can also be installed airspayce.com here.

   *Also python library spidev is needed. To install enter 'pip3 install spidev'
4. Copy Waveshare High Precision AD/DA board libraries from Waveshare's wiki page here. Unzip the folder and use a usb to move onto Raspberry PI.
5. Create a new folder folder for your code files (Ex. High_Precision_PH). Copy ADS1256.py and config.py from the folder you got from the previous step.

   *main.py can also be copied too if you want demo code for ADC
6. To get ADS1256.py and config.py enter:

   'cd $HOME/High-Precision-AD-DA-Board-Code/RaspberryPI/ADS1256'

   'cp ./ADS1256.py path/to/destination/folder'

   'cp ./config.py path/to/destination/folder'
7. Change directory to the folder for your code files. Create a new file,phsensor.py, and copy the following class code:

```python
import time
import ADS1256
import RPi.GPIO as GPIO

#ADC1256 object and initiation
adc = ADS1256.ADS1256()
adc.ADS1256_init()

#millivolt voltages from calibration
PH7_VOLTAGE_MV = 1773
PH4_VOLTAGE_MV = 1937

NUM_OF_SAMPLES = 20

class PHSensor:
    '''
    PH Sensor Class for Grove PI Sensor with Waveshare High Precision AD/DA board

    Args:
        (int) ph_min: minimum ph value ph sensor can read(default = 0)
        (int) ph_max: maximum ph value ph sensor can read(default = 14)
    '''
    def __init__(self, ph_min = 0, ph_max = 14):
        self.ph_min = ph_min
        self.ph_max = ph_max
```

```python
def read_voltage(self,pin):
    '''
    Read voltage value from ph sensor through AD/DA board

    Args:
        (int) pin: analog signal pin ph sensor module is connected on AD/DA board
    '''
    adc_value = adc.ADS1256_GetAll()
    voltage = adc_value[pin] * 5.0/ 0x7fffff #voltage - adc data * supply voltage/0x7fffff because 24bit ADC
    return voltage

def read_voltage_mv(self,pin):
    '''
    Read millivolt voltage value from ph sensor through AD/DA board

    Args:
        (int) pin: analog signal pin ph sensor module is connected on AD/DA board
    '''
    return self.read_voltage(pin) * 1000
```

```python
def read_ph(self, pin):
    '''
    Read ph value from ph sensor through AD/DA board

    Args:
        (int) pin: analog signal pin ph sensor module is connected on AD/DA board
    '''
    voltages_sum = 0
    ph = 0

    for i in range(NUM_OF_SAMPLES): #read int NUM_OF_SAMPLES of mv voltage samples from AD/DA board
        voltages_sum += self.read_voltage_mv(pin)

    voltage_mv = voltages_sum / NUM_OF_SAMPLES #average voltages
    ph = 7 + (voltage_mv - PH7_VOLTAGE_MV) * (-3 / (PH4_VOLTAGE_MV - PH7_VOLTAGE_MV)) #pHx = pH1 + (Ex – E1)(pH2 – pH1)/(E2-E1)

    return ph
```

```
def ph_calibration(self,pin):
    '''
    Returns average calibration mv voltages for ph7 and ph4

    Args:
        (int) pin: analog signal pin ph sensor module is connected on AD/DA board
    '''

    ph_solution = 0

    for i in range(2):
        if i == 0:
            ph_solution = 7
            user_input = input(f"Enter 'Y' when ph sensor is in PH {ph_solution} solution:")
        elif i == 1:
            ph_solution = 4
            user_input = input(f"Enter 'Y' when ph sensor is in PH {ph_solution} solution:")

        while user_input != 'Y':
            user_input = input(f"Enter 'Y' when ph sensor is in PH {ph_solution} solution:")

        voltages_sum = 0
        for i in range(NUM_OF_SAMPLES):#read int NUM_OF_SAMPLES of mv voltage samples from AD/DA board
            voltages_sum += self.read_voltage_mv(pin)

        print(f"The average mv voltage for PH 7 is {voltages_sum/NUM_OF_SAMPLES}")
```

```
def print_all(self,pin):
    '''
    Print updating voltage value and ph value on same line

    Args:
        (int) pin: analog signal pin ph sensor module is connected on AD/DA board
    '''
    print("PIN:A%d Voltage:%lf PH:%.2f"%(pin, self.read_voltage(pin),self.read_ph(pin)))
    print("\33[2A")
```

8. Create a main file to run the code (Ex.main2.py). Copy the following code.

```
import os
import time
import phsensor
import RPi.GPIO as GPIO

ph_sensor = phsensor.PHSensor(0,14) #phsensor object

try:
    user_calibrate = input("Calibrate ph probe?(Y/N):")
    if user_calibrate == 'Y':
        ph_sensor.ph_calibration(0)
    while True:
        ph_sensor.print_all(0) #print voltage and ph value of pin A0
        time.sleep(0.5) #update every 0.5 seconds

except KeyboardInterrupt:
    GPIO.cleanup()
    os.system("clear") # clear terminal
    print ("\r\nProgram end     ")
    exit()
```

9. Run main2.py. When asked “Calibrate ph probe?(Y/N)”, enter ‘Y’ to get calibration values. Proceed to dip the probe in the specified solution when asked and enter ‘Y’ again to proceed.

```
Enter 'Y' when ph sensor is in PH 7 solution:Y
The average mv voltage for PH 7 is 1810.6155169743918
Enter 'Y' when ph sensor is in PH 4 solution:Y
The average mv voltage for PH 7 is 1809.5008503795677
```

Example calibration step. Values above are for demo only

10. If you enter ‘Y’ in the above step, write down the average mv values for ph 7 and ph 4. Modify ‘PH7_Voltage_MV’ and ‘PH7_Voltage_MV’ in phsensor.py with the written values. Repeated above steps through 9 for necessary recalibration.
11. If you did not enter ‘Y’ in step 9, the program will proceed to print the voltage and ph reading from the ph sensor.

12. Calibrate ph sensor with ph7 and ph4 calibration liquid.
13. PH sensor calibrations:

    Test two PH calibration liquids and write down the voltage readings. Calculate the ph formula using the equation below. It's recommended in ph calculation steps by Cornel Radu here.

    $$pHx = pH1 + (Ex - E1)(pH2 - pH1)/(E2-E1)$$

    *E1 and E2 are the millivolts voltage read by the sensor while ph1 and ph2 are the PH level of the calibration liquid

Results:

PH Formula Calculation:

ph = 7 + (Ex - 1780) * (4 - 7)/ (1940 - 1780) = 7 + (Ex - 1780) * (-0.01875)

PH Calibration Liquid	PH Reading	Voltage	Error %
PH7	7 ~ 7.25	1.7V	0% to 3.57%
PH4	3.91 ~ 4.12	1.932V ~1.944V	0% to 3%
Baking Soda Solution (Expected:8.3)	8.00 ~ 8.16	1.717V ~ 1.724V	1.68% to 3.6%
White vinegar (Expected:2.5)	2.33 ~ 2.46	2.021V ~ 2.028	4% to 13.2%
Fresh Lemon Juice (Expected:2)	2.21~2.33	2.029V ~ 2.035V	10.5% to 16.5%
Dawn Soapy Water (Expected:8.7 ~ 9.3)	9.04~9.17	1.664V ~ 1.6718V	Within expected range

*Soapy water pH varies by concentration. In this case, I added a lot of Dawn dish soap. Dawn ph level.

The ph readings with Wave High Precision AD/DA board aren't as consistent as the results from Arduino and RPI USB serial connection, because Waveshare's board has a 24 bit precision ADC thus the sensor values are more sensitive (each ADC channel max value is 7FFFFF or 8388607). Also with higher bit precision, the readings can be more precise like to the hundredth thousandth place.

Compared to the Grove PI Hat method, the pH values are much more stable without a big spike/drop of voltage affecting the actual pH value. This makes the Waveshare AD/DA board much more accurate in readings.

Conclusion

In conclusion, out of all the three solutions, USB serial between RPI/Arduino and Waveshare's AD/DA board provides the most stable while accurate readings. Even though USB serial connection between RPI/Arduino has the most stable readings, this method has to watch out for power usage and multiple power supplies. Waveshare's AD/DA board on the other hand is more easy to set up and has only one function AD/DA conversion which makes it easier to debug than PI Hat. It also has a very good bit precision for accurate readings. Finally, Grove Base PI Hat is proven to not be applicable for PH sensor reading at this time since it has the worst fluctuation in voltage and ph values. Although some values read through Grove PI Hat are close to the actual pH, the readings are too inconsistent and hard to debug due to the variety of ports/functionalities of this board.

USB Serial:
Pro:
-Very Stable Accurate Readings
-Working Demo code by ph sensor manufacturer
Con:
-Reading depends on adequate power supply/source

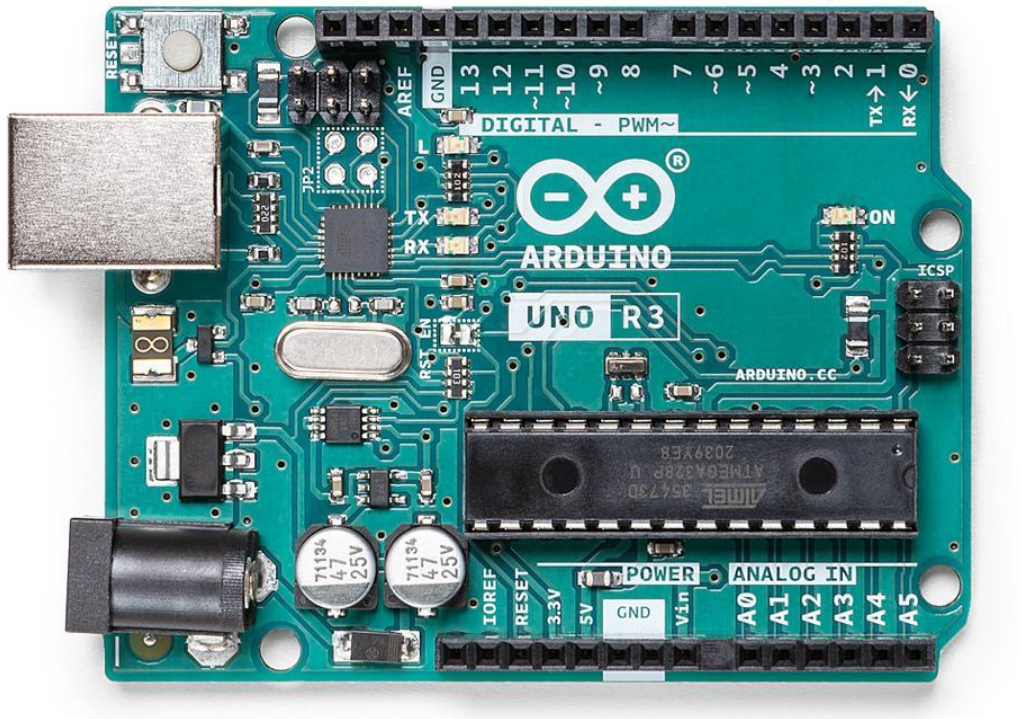
Waveshare High Precision AD/DA Board:
Pro:
-High bit precision readings
-Easy Setup RPI Hat
-Well Written Code Libraries
-One Main Function AD/DA conversion
-Built in demo test(LED and potentiometer)
Con:
-Sensitive Reading(slight fluctuation in reading)

Grove Base Hat:
*Not applicable for PH sensor reading purpose

Hardware

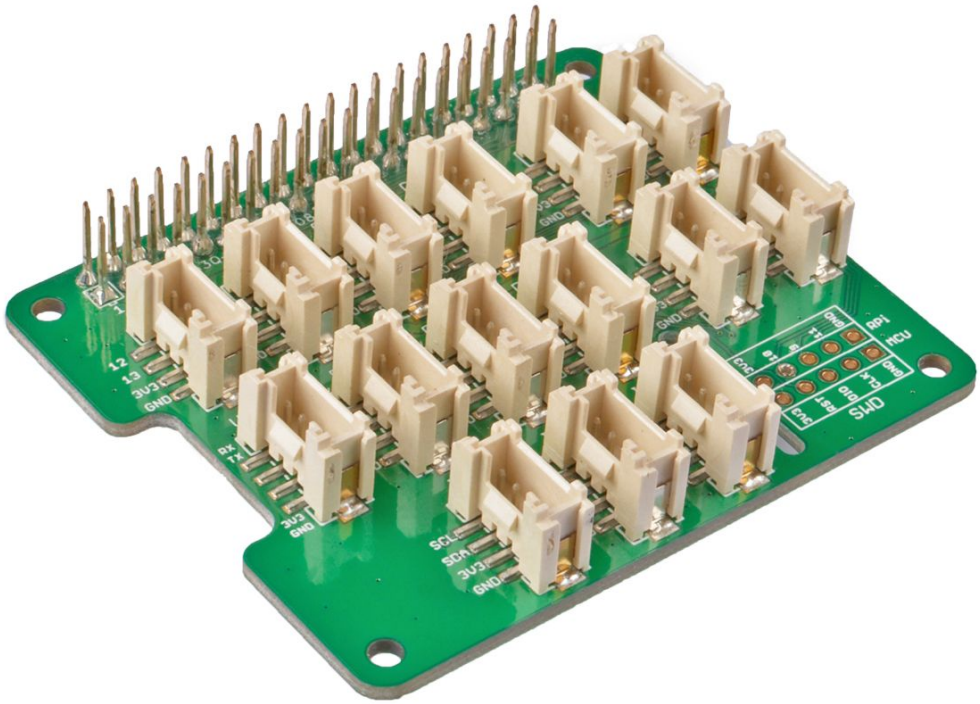
- **Arduino UNO:**

  Arduinos are a popular series of cheap open source development boards that are simple to use. Like Raspberry PI, it can be used with a variety of electronics devices, but Arduino UNO has analog ports great for use with analog sensors/devices. Unlike Raspberry PI though, it does not have a computer interface and can only run one sketch continuously at a time.



❖ **Grove Base Hat for Raspberry PI**

An attachable Hat for Raspberry PI made by Seeed. It features Digital/Analog/I2C/PWM/UART ports and most importantly a built-in 8 channel 12bit ADC since a RPI has no analog conversion. More info at this link.



❖ **Grove PH Sensor (E-201C-Blue):**

Seeed's Grove PH module with a blue ph probe sensor. Since this ph sensor operates at 3.3V and 5V, Seeed claims that it works with Arduino and theoretically with Raspberry with additional ADC conversion with a Hat. No current demo code/instructions are provided at Seeed's site for Raspberry PI. More details at their site.

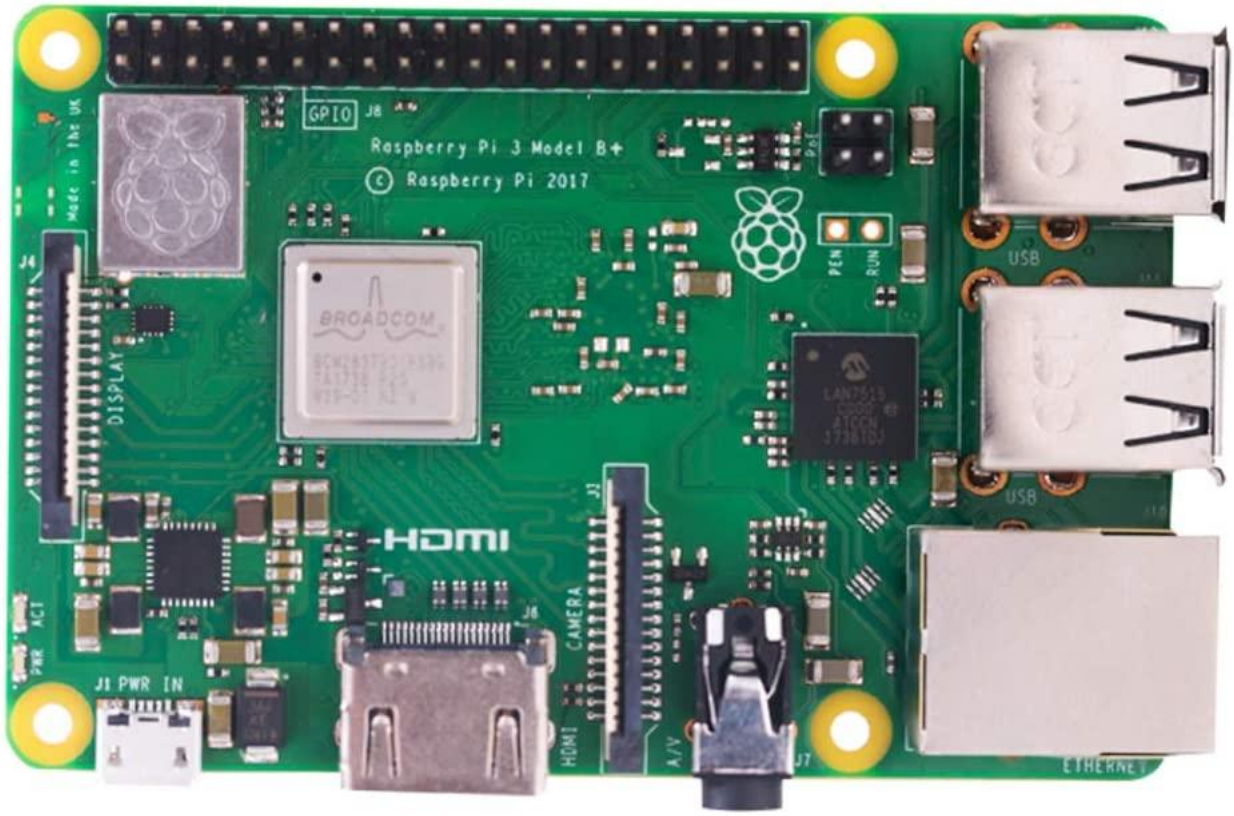
Features:

Operating voltage: 3.3V/5V

Resolution:±0.15PH（STP）

❖ **Raspberry PI 3B+:**

Raspberry PI is a popular mini computer board great for developers. It sports GPIO pins to connect variety of circuit and electronic devices (ie. sensor, motors,..etc). Also, with a more powerful processor than Arduino, RPI can run more computation and work like a computer.



RP3B+ features:

CPU:Broadcom BCM2837B0 quad-core A53 (ARMv8) 64-bit @ 1.4GHz

Network:Gigabit Ethernet (via USB channel), 2.4GHz and 5GHz 802.11b/g/n/ac Wi-Fi

Bluetooth: Bluetooth 4.2

Storage: Microsd card slot

Peripheral: USB, HDMI, Camera Serial Interface (CSI), Display Serial Interface (DSI)

❖ **Waveshare High Precision AD/DA Board**

Raspberry Hat board provided by Waveshare with the primary function of ADC/DAC conversion for the Raspberry PI board. It sports an impressive ADS1256 8 channel 24 bit precision ADC which is great for our application. Available here.

Features:

GPIO: 40 Pin GPIO

ADC:ADS1256, 8ch 24bit high-precision ADC (4ch differential input), 30ksps sampling rate

DAC:DAC8552, 2ch 16bit high-precision DAC

References

- Bergas, Izara. “What Is the PH of Dawn Dish Detergent?” *What Is the Ph of Dawn Dish Detergent?*, FINDANYANSWER, 2 Apr. 2020, https://findanyanswer.com/what-is-the-ph-of-dawn-dish-detergent.
- Digital Analysis Corporation. “PH Probe Architecture.” *Ph Probe Architecture*, Digital Analysis Corporation, http://www.phadjustment.com/TArticles/pH_Probe.html .
- Endress+Hauser. “Potentiometric Ph Measurement - Youtube.” *Potentiometric Ph Measurement*, Youtube, 26 Apr. 2013, https://www.youtube.com/watch?v=P1wRXTl2L3I.
- “High-Precision AD/Da Board.” *High-Precision AD/DA Board - Waveshare Wiki*, Waveshare, https://www.waveshare.com/wiki/High-Precision_AD/DA_Board.
- Kuphaldt, Tony R. “Chapter 9 - Electrical Instrumentation Signals:PH Measurement.” Edited by All About Circuits, *Lessons in Electric Circuits*, All About Circuits, https://www.allaboutcircuits.com/textbook/direct-current/chpt-9/ph-measurement/.
- MetrohmTV. *Webinar Recording: PH Measurements Made Easy – Basics of Potentiometric PH Measurements*, YouTube, 18 Sept. 2020, https://www.youtube.com/?gl=DE .
- “Ph Electrodes.” *Chemistry LibreTexts*, Libretexts, 9 June 2020, https://chem.libretexts.org/Bookshelves/Analytical_Chemistry/Supplemental_Modules_(Analytical_Chemistry)/Analytical_Sciences_Digital_Library/JASDL/Courseware/Analytical_Electrochemistry%3A_Potentiometry/05_pH_Electrodes.
- “Ph Meter.” *Wikipedia*, Wikimedia Foundation, 22 July 2021, https://en.wikipedia.org/wiki/PH_meter.
- Shubham. “Send Sensor Data from Arduino to Raspberry Pi in 5 Easy Steps!” *Yantraas*, Yantraas, 9 Apr. 2021, https://yantraas.com/send-sensor-data-from-arduino-to-raspberry-pi/.

- Sivaranjith. “PH Measurement : Glass Electrode Method.” *PH Measurement : Glass Electrode Method*, AutomationForum.co, 3 Apr. 2018, https://automationforum.co/ph-measurement-glass-electrode-method/.
- Vturesource. “Glass Electrode - Youtube.” *Glass Electrode*, Youtube, 18 Sept. 2016, https://www.youtube.com/watch?v=pn8SA_DzvdI.
- Zuo, Baozhu. *Grove Base Hat for Raspberry Pi*, Seeed, https://wiki.seeedstudio.com/Grove_Base_Hat_for_Raspberry_Pi/.
- Zuo, Baozhu. “Grove - PH Sensor Kit (E-201C-Blue).” *PH Sensor Kit (E-201C-Blue)*, Seeed, https://wiki.seeedstudio.com/Grove-PH-Sensor-kit/.