Weather Station Raspberry Pi Software Setup

By David W. Enström

This personal weather station uses the SparkFun weather sensors (Argent Data Systems), and the BC Robotics add-on hardware (HAT) to interface these sensors to the Raspberry Pi.

See: https://www.sparkfun.com/products/8942 and https://www.bc-robotics.com/tutorials/raspberry-pi-weather-station-part-1/

The weather station monitors (and optionally shares the data with Wunderground):

- **Wind Speed** – in km/h
- **Wind Direction** – in degrees
- **Rainfall** – in mm
- **rainRate** – in mm/hr
- **Temperature** – in degrees C
- **Air Pressure** – in hecto Pascals
- **Humidity** – in percent

Instructions for constructing this weather station and all the required components are in the “`Weather- Station-Raspberry-Pi-construction.pdf`” PDF file.

Raspberry Pi

The weather station is connected to a Raspberry Pi … it is actually a HAT (Hardware Attached on Top). The HAT is from BC Robotics and uses the BME280 temperature, pressure, and humidity sensor, the DS18B20 Digital temperature sensor, and the SparkFun Weather sensors from Argent Data Systems for wind and rain.

The Raspberry Pi is setup with a static IP, to connect to and manage the Pi easier, for example:

```
192.168.0.110/24
```

Do this by editing the following file:

```
etc/dhcpcd.conf
```

Command Summary

Install the `vsftpd` FTP service (in `Preferences / Add/Remove Software`) and to enable FTP write access to the Pi; edit `vsftpd.conf` and:

```
write_enable=YES
```

To edit a system file in Raspbian: `sudo leafpad` *`filename`*

To shutdown the Pi: `sudo shutdown -h now`

To start/stop WeeWX:

```
sudo /etc/init.d/weewx start
sudo /etc/init.d/weewx stop
```

Setup Raspbian for WeeWX and the BC Robotics Hardware

The following assumes that you already have Raspbian running on your Pi. See: https://www.bc-robotics.com/tutorials/raspberry-pi-weather-station-part-2/ for basic Raspbian setup instructions.

Before installing WeeWX and the BC Robotics driver, setup the required software by running the following commands (in this order) from the command prompt:

Install the Adafruit Python GPIO Library

```
sudo apt-get update

sudo apt-get install build-essential python-pip python-dev python-smbus git

git clone https://github.com/adafruit/Adafruit_Python_GPIO.git

cd Adafruit_Python_GPIO

sudo python setup.py install
```

Install the Adafruit BME280 Library

```
cd

git clone https://github.com/adafruit/Adafruit_Python_BME280.git

cd Adafruit_Python_BME280

sudo python setup.py install
```

Test the BME280 Sensor if desired

```
python Adafruit_BME280_Example.py
```

Install the ADS1x15 Library, the ADS1015 Analog to Digital chip (wind direction)

```
cd

git clone https://github.com/adafruit/Adafruit_Python_ADS1x15.git

cd Adafruit_Python_ADS1x15

sudo python setup.py install
```

Setup the DS18B20 temperature sensor and install the software library

```
cd

sudo modprobe w1-gpio

sudo modprobe w1-therm

cd /sys/bus/w1/devices

ls

cd

sudo apt-get install python-w1thermsensor
```

The “ls” command will display the contents of the devices “folder” in the window. The DS18B20 shows up as an address something like “`28-0316853d8fff`” – but each sensor has a unique ID.

You can test the sensors by running the included test app: `BCRobotics-test-app.py`

Run the Python IDE:

```
>>idle
```

Now open the “BCRobotics-test-app.py” test app from the IDE and hit 'F5' to run it. It will continuously print out the readings from the sensors, including the value read from the ADC for the wind direction. Use these values (and a test template for the wind direction) to double check the values used in the driver. Hit “<ctrl> c” to stop the program.

WeeWX Setup

The Raspberry Pi runs WeeWX, a freely available weather application written in Python. The installation of WeeWX (see: http://weewx.com/docs/debian.htm) results in the following layout:

Role	Symbolic Name	Location
WeeWX Root	WEEWX_ROOT	/
Executables	BIN_ROOT	/usr/bin/weewxd
Configuration file	CONFIG_ROOT	/etc/weewx/weewx.conf
Skins and templates	SKIN_ROOT	/etc/weewx/skins
SQLite databases	SQLITE_ROOT	/var/lib/weewx/
Web pages and images	HTML_ROOT	/var/www/html/weewx/
Documentation	DOC_ROOT	/usr/share/doc/weewx/
Examples	EXAMPLE_ROOT	/usr/share/doc/weewx/examples/
Log files		/var/log/syslog
PID File		/var/run/weewx.pid
Utilities		/usr/share/wee_*
Extensions (i.e., the driver)		/usr/share/weewx/user/

WeeWX logs many events to the system log. On Debian systems, this is `/var/log/syslog`.

BCRobotics Driver Installation

Edit the `[Station]` section in `weewx.conf` to add the following parameters, adjusted for your particular situation:

- location = short description (e.g., Honey Gables, a suburb of Ottawa, Canada)
- latitude = 45.38044
- longitude = -75.29585
- altitude = 87 meters
- station_type = **BCRobotics**
- station_url = http://www.mywebsite.com
- units = **metric**
- week_start = 0

The values in bold are mandatory definitions for the BCRobotics driver.

Once this setup is done, download the BCRobotics driver TAR file to your Raspberry Pi from here:

```
        wget -O BCRobotics.zip https://github.com/David-Enst/WeeWX-BCRobotics/archive/master.zip
```

Install the Extension Tar file (copied to `/home/pi/`) with an optional “dry run”:

```
        sudo wee_extension --install=/home/pi/BCRobotics.zip (--dry-run)
```

The `wee_device` utility can be tried, but has not been tested. It is used to configure BCRobotics stations, including updating various `weewx.conf` parameters. This can also be done manually by defining the following parameters in `weewx.conf`.

Definition of the BCRobotics driver:

```
 [BCRobotics]
    # This defines the "Spark Fun" SEN-08942 / BC Robotics weather stations.
    # See:  https://www.sparkfun.com/products/8942
    #       https://www.bc-robotics.com/tutorials/raspberry-pi-weather-station-part-1/
    # The time (in seconds) between LOOP packets.
    loop_interval = 3

    # Driver mode - tcp, udp, or serial
    mode = serial

    # If serial, specify the serial port device. (ex. /dev/ttyS0, /dev/ttyUSB0,
    # or /dev/cuaU0)
    # If TCP, specify the IP address and port number. (ex. 192.168.36.25:3000)
    port = /dev/ttyS0

    # The amount of time, in seconds, before the connection fails if
    # there is no response
    timeout = 3

    # Debug level - the level of message logging. The higher
    # the number, the more info is logged.
    debug_read = 0

    # The driver to use:
    driver = user.BCRobotics
```

Definition of `[[[Units]]]` to ensure the web pages generated display the correct units:

```
    [[StandardReport]]
        # The StandardReport uses the 'Standard' skin, which contains the
        # images, templates and plots for the report.
        skin = Standard

        [[[Units]]]
            [[[[Groups]]]]
                group_altitude = meter
                group_speed2 = km_per_hour2
                group_pressure = hPa
                group_rain = mm
                group_rainrate = mm_per_hour
                group_temperature = degree_C
                group_degree_day = degree_C_day
                group_speed = km_per_hour
```

The database and driver is setup to store values in `METRIC` by default. Therefore, the following update to the quality control parameters are required:

```
[StdQC]

    [[MinMax]]
        barometer = 800, 1110, mbar # = hPa
        pressure = 800, 1110, mbar  # = hPa
        outTemp = -50, 49, degree_C
        inTemp = -40, 49, degree_C
        outHumidity = 0, 100
        inHumidity = 0, 100
        windSpeed = 0, 300, km_per_hour
```

The `[StdConvert]` section is updated to reflect the fact that all data in the database is stored in `METRIC` by default:

```
    target_unit = METRIC    # Options are 'US', 'METRICWX', or 'METRIC'
```

Finally, the archive section is updated to force `WeeWX` to generate archive records through software generation, since the BC Robotics setup does not generate them:

```
[StdArchive]
    # If possible, new archive records are downloaded from the station
    # hardware. If the hardware does not support this, then new archive
    # records will be generated in software.
    # Set the following to "software" to force software record generation.
    record_generation = software
```

Most of these parameters should be setup during the driver installation, but they should all be verified.

Station data

The following table shows the data provided by the station hardware and those calculated by `WeeWX`.

BCRobotics Station Data			
Database Field	**Observation**	**Loop**	**Archive**
dateTime	-	D	
usUnits	-	D	
interval	-	D	
barometer		S	
pressure	pressure	H	
altimeter		S	
inTemp (Case)	case_temp	H	
outTemp	outTemp	H	
outHumidity	outHumidity	H	
windSpeed	windSpeed	H	
windDir	windDir	H	
rain	rain	D	
rainRate	rainRate	D	
rainTotal	-	S	
dewpoint	-	S	
windchill	-	S	
heatindex	-	S	

H indicates data provided by **H**ardware
D indicates data calculated by the **D**river
S indicates data calculated by the StdWXCalculate **S**ervice

Some references:

https://www.bc-robotics.com/tutorials/raspberry-pi-weather-station-part-2/

http://www.weewx.com/docs/usersguide.htm#about

https://docs.python.org/2.7/contents.html

https://www.lenntech.com/calculators/humidity/relative-humidity.htm

https://sourceforge.net/p/raspberry-gpio-python/wiki/Inputs/

https://github.com/David-Enst/WeeWX-BCRobotics

Weather Formulæ

Wind Direction

The wind direction sensor outputs a different number read by an ADC (analog-to-digital converter) based upon the direction—but this number will change a small amount based on temperature, power fluctuations, and the actual resistors in the sensor. so you can't just check if the ADC output matches a number in a list. Instead, they fall within a range as shown in the table.

Actual readings from my sensor, used to fine-tune the Python code in the driver, are shown below. Note that in the `BCRobotics.py` driver the following selections are used to try to cover *all* possible values. This eliminates an out-of-range error, and therefore a faulty wind direction (or program crash!). It is quite likely that the driver will work properly with the range selection values shown, but it doesn't hurt to do a little test to see the actual values from your device. The numbers are found by manually moving the wind direction sensor to each of the 16 directions and reading the output. Write these numbers down. A compass rose template is provided at the end of this document for your convenience.

Direction (Degrees)	Resistance (Ohms)	Voltage (V=5v, R=10k)	ADC Reading	Range Selection
N (0)	33k	3.84v	20352	19600-20999
NNE (22.5)	6.57k	1.98v	10512	9000-10799
NE (45)	8.2k	2.25v	11952	10800-13999
ENE (67.5)	891	0.41v	2176	2000-2299
E (90)	1k	0.45v	2416	2300-2999
ESE (112.5)	688	0.32v	1712	1000-1999
SE (135)	2.2k	0.90v	4768	4000-4999
SSE (157.5)	1.41k	0.62v	3264	3000-3999
S (180)	3.9k	1.40v	7440	6600-8999
SSW (202.5)	3.14k	1.19v	6336	5000-6599
SW (225)	16k	3.08v	16336	15900-16999
WSW (247.5)	14.12k	2.93v	15536	14000-15899
W (270)	120k	4.62v	24464	24000-24999
WNW (292.5)	42.12k	4.04v	21440	21000-21999
NW (315)	64.9k	4.78v	22992	22000-23999
NNW (337.5)	21.88k	3.43v	18208	17000-19599

Wind Speed

Wind speed is the rate of horizontally moving air past a fixed point, and measured by an anemometer. Some definitions for the weather app (on the Raspberry Pi):

Wind Speed – Current wind speed is the average wind speed recorded over a defined period, 3 seconds for example (i.e., the “LOOP” period).

Average Speed – An average of all wind speed-readings (i.e., LOOP readings) collected over a defined period, 5 minutes for example (i.e., the ARCHIVE period).

Peak Speed – A rolling value showing the highest wind speed recorded over a given interval (usually the ARCHIVE interval).

Wind Chill and Feels-Like Temperature

Wind Chill

Wind chill temperature only works for temperatures at or below 10°C (50°F) and wind speeds above 4.8kph (3.0 mph). The standard Wind Chill formula from Environment Canada (using °C and km/h; but the same formula converted to °F and MPH is used in the USA) is:

$$T_{wc} = 13.12 + 0.6215T_{\mathrm{a}} - 11.37v^{+0.16} + 0.3965\mathrm{T}_{\mathrm{a}}v^{+0.16}$$

Where:

$\mathbf{T_{wc}}$ is the wind chill index, based on the Celsius temperature scale;

$\mathbf{T_a}$ is the air temperature in degrees Celsius; and

$\boldsymbol{v}$ is the wind speed at 10m (33 ft.) standard anemometer height, in kilometres per hour.

When the temperature is −20 °C (−4 °F) and the wind speed is 5 km/h (3.1 mph), the wind chill index is −24. If the temperature remains at −20 °C and the wind speed increases to 30 km/h (19 mph), the wind chill index falls to −33.

Feels-Like Temperature

A heat index value is valid for temperatures at or over 27° Celsius.

There are many formulae devised to approximate the original tables by Steadman. Anderson et al (2013), NWS (2011), Jonson and Long (2004), and Schoen (2005) have lesser residuals in this order. The former two are a set of polynomials, but the third one is by a single formula with exponential functions.

The formula below approximates the heat index in degrees Fahrenheit, to within ±1.3 °F (0.7 °C). It is the result of a multivariate fit to a model of the human body. The temperature must be equal to or greater than 80 °F (27 °C) and relative humidity equal to or greater than 40%. This equation reproduces the above NOAA National Weather Service table (except the values at 90 °F (32 °C) and 45%-70% relative humidity vary unrounded by less than ±1, respectively). *(WeeWX uses this one)*

$$\mathrm{HI} = c_1 + c_2T + c_3R + c_4\,TR + c_5T^2 + c_6R^2 + c_7T^2R + c_8TR^2 + c_9T^2R^2$$

Where:

HI = heat index (in degrees Fahrenheit)
T = ambient dry-bulb temperature (in degrees Fahrenheit)
R = relative humidity (percentage value between 0 and 100)

$c_1 = -42.379$	$c_2 = 2.04901523$	$c_3 = 10.14333127$
$c_4 = -0.22475541$	$c_5 = -6.83783 \times 10^{-3}$	$c_6 = -5.481717 \times 10^{-2}$
$c_7 = 1.228\ 74 \times 10^{-3}$	$c_8 = 8.5282 \times 10^{-4}$	$c_9 = -1.99 \times 10^{-6}$

Temperature

The DS18B20 temperature sensor shows up on the Raspberry Pi with an address something like `28-0517c13642ff` as noted in the setup instructions above. The sensor provides a value in centigrade.

The simple formula to convert between Fahrenheit (°F) and degrees Celsius (°C):

$T_c = {}^{9}/_{5} \times (T_f - 32)$
$T_f = ({}^{5}/_{9}) \times T_c + 32$

Where:

T_c is temperature in Celsius
T_f is temperature in Fahrenheit

WeeWX looks after all these conversions … see the documentation on the Standard web report.

Humidity

The humidity is measured inside the case containing the Pi, therefore this relative humidity (RH) reading is relative to the temperature inside the case (i.e., `case_temp`) and must be converted to a RH for the temperature outside (i.e., the `outTemp`). The following formulæ are used:

See: https://www.lenntech.com/calculators/humidity/relative-humidity.htm

```
# This humidity is measured inside the case,
# which is warmer than the ambient air. Therefore
# it is converted to external humidity based upon
# the case_temp and outTemp. First calculate the
# absolute moisture level in the air.

absMoisture = (humidity * 0.42 * math.exp(case_temp * 0.06235398)/10)

# Now calculate the humidity reading to the outside temperature
humidity = (absMoisture * 10 / (0.42 * math.exp(outTemp *  0.06235398)))
```

Pressure

The current pressure is displayed in hectopascals (hPa). A trend of the changing pressure shows what has happened over the last 3 hours.

Rainfall

The rainfall amount accumulated over the LOOP period is tracked. Weather Underground displays:

- **Rain** – the accumulation of rain in the past 1 hr and the past 24 hrs
- **Rate** – the current rate of rainfall in mm/hr for the LOOP period

There are certain situations where a single random rain "tick" may occur:

- Certain gusts of strong wind could cause a random tick—if the "bucket" is not exactly level and when at rest on the high side a strong gust could knock it back;
- Melting frost or dew may accumulate enough to cause a random tick over time;
- An almost full "bucket" and a sudden gust of wind can also cause it to tick over.

The WeeWX driver therefore has a random tick detector to catch single ticks. The downside is that the first tick when it actually starts to rain will be ignored.

Wunderground

The WeeWX software can share its reading with your own Weather Underground web page. Register your Personal Weather Station (PWS) at wunderground.com, where you will receive a station ID along with a password. Enter these into the `weewx.conf` file as directed here:

http://weewx.com/docs/usersguide.htm#[StdRESTful]

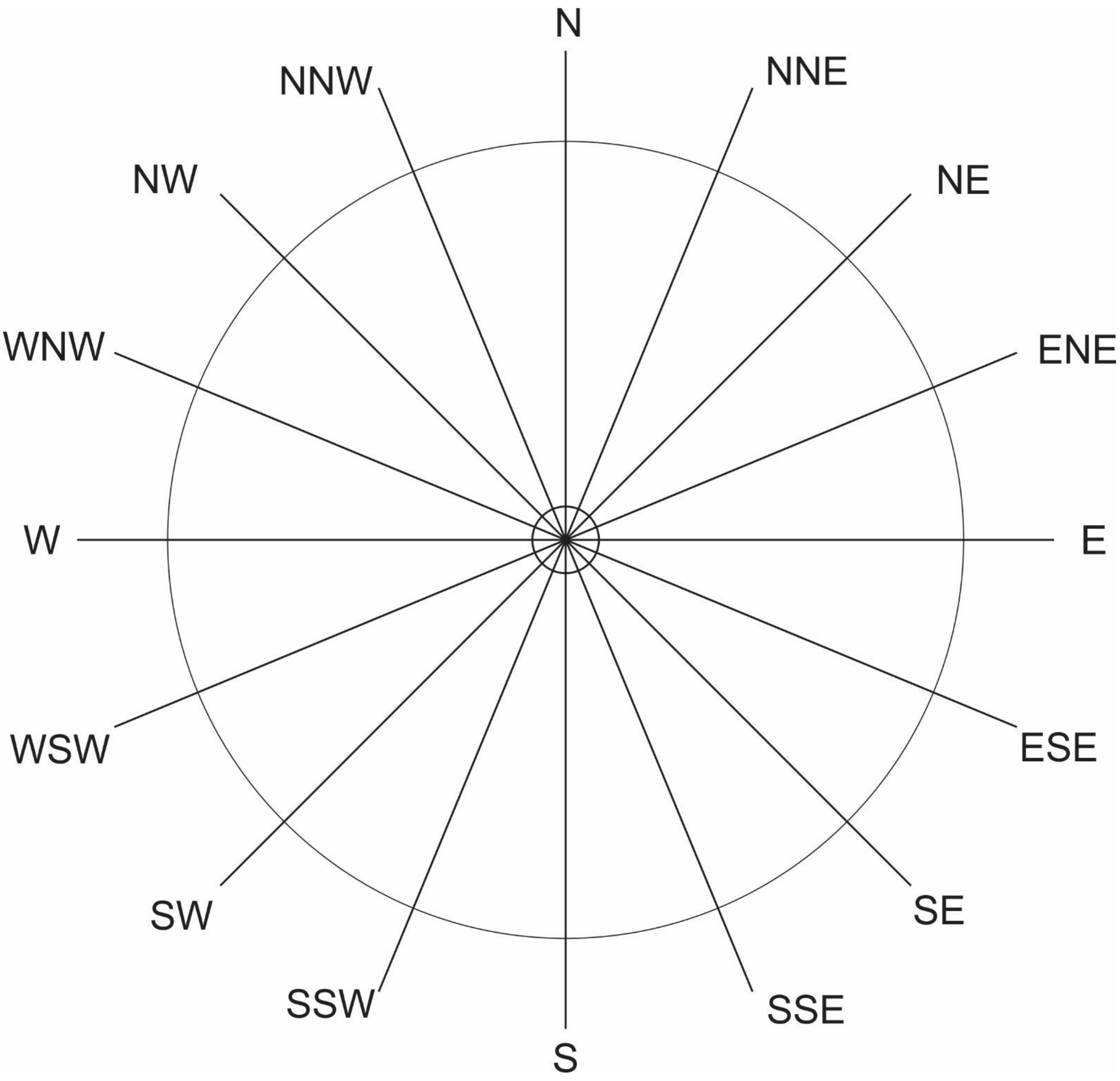
Wunderground registration detail example:

Your Station ID:	**ABCDEFGxx**
Your Station Key/Password:	**abxycdij**

Example WU PWS URL:

https://www.wunderground.com/personal-weather-station/dashboard?ID=IOTTAWA98

Once your PWS is setup you can use this WU URL in your `[Station]` section's `station_url` setting.


N
NNE
NE
ENE
E
ESE
SE
SSE
S
SSW
SW
WSW
W
WNW
NW
NNW