Raspberry PI 'How-To' Series

AOSONG AM2315 Temperature Sensor Implementation Guide Update

Written by: Sopwith
Revision 3.0
February 1, 2019
sopwith@ismellsmoke.net

Introduction

Here we are in February 2019, and it is time to update my AM2315 temperature sensor implementation guide. This highly accurate and popular sensor is a favorite among Raspberry Pi enthusiasts. I received an Email this week asking me why my AM2315 code did not work in Python3. This is a very good question.

My code samples for the AM2315 rely on an underlying I2C C library written by Alexander Ruedlinger. He created Python bindings for Python version 2.7. His code will not compile under Python version 3. Also, his code has not been updated since June 12, 2015. You long-time *Sopwith* followers may recall before I used the Ruedlinger code, I used a different library named *quickwire*. That project died mid-flight and required me to make a change to the Ruedlinger code five years ago.

It is time to ditch the underlying C code dependencies and change my AM2315 code to a pure Python implementation that supports both Python2.7 and Python3. This effort will, once and for all, remove the pesky C dependencies and allow makers to use their Python version of choice.

To accomplish this feat, we will use code libraries from Adafruit and SwitchDoc Labs. Adafruit will provide us their terrific Raspberry Pi Python GPIO module, and SwitchDoc Labs wrote code that uses the Adafruit module to talk to the AM2315 sensor.

The SwitchDoc Labs code may work just fine for your project. Be aware, however, their code will not work with Python3. I made slight modifications to the SwitchDoc code to make it work with both Python2.7 and Python3.

This update implementation document walks you through the steps to get your AM2315 temperature sensor working with pure Python code.

Step-by-Step

There are twelve (6) steps in the implementation of an AM2315.

1. Get your Pi up and running with *Raspian Stretch*.
2. Install required support software.
3. Wire up your AM2315.
4. Configure i2c and test the AM2315 hardware.
5. Install the *Adafruit* Python module and download the SwitchDoc Labs code.
6. Test the AM2315 software.

Step-1 - Get your Pi up and running with Raspian Stretch.

The first thing you need to do is get your Pi running. Head out to *raspberrypi.org* and download *Raspian Stretch* or *Stretch Lite.* The former has a Window GUI while the latter is a command-line only OS. Either one works.

Once the download completes, burn the image to an SD card and fire up your Pi. If you need help with this step, there is a ton of resources on the web to help you. When you are up and running and connected to the Internet, be sure to update your Pi with the latest patches.

```
$sudo apt-get update
$sudo apt-get upgrade
$sudo reboot
```

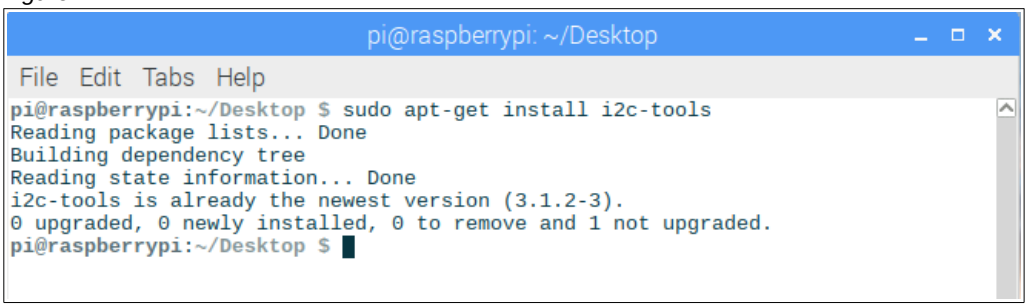
Step-2 - Install required support software.

There are three (3) software packages that must be installed for the i2c interface to work correctly. Follow the below steps to get them installed.

Open a command window and type in the command: $*sudo apt-get install i2c-tools.*

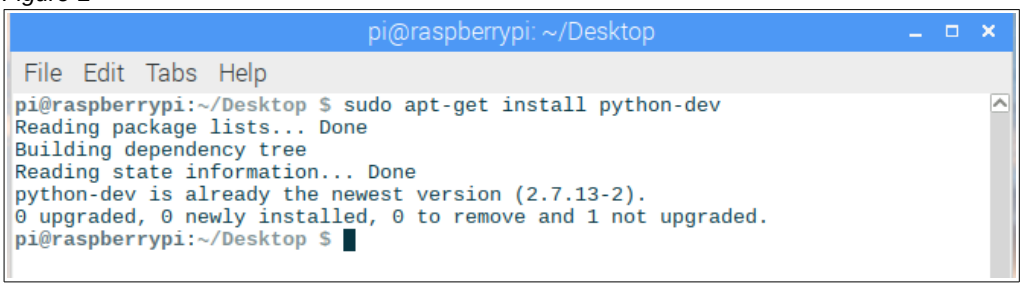
You will probably get the output shown in Figure-1 below.

Figure-1



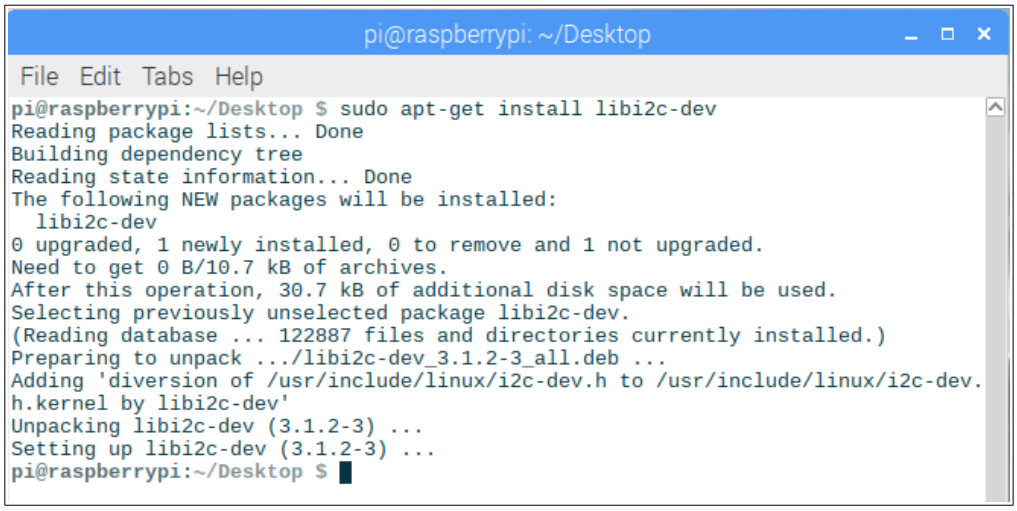
Next, run the command: $*sudo apt-get install python-dev*.
You will probably get the output shown in Figure-2 below.

Figure-2



Finally, run the command: $*sudo apt-get install libi2c-dev.*
This package is probably not installed on your Pi so the output is going to be different. The package will be installed as shown in Figure-3 below.

Figure-3



You now have all the software you need to get your AM2315 hardware wired up and running.

Step-3 - Wire up your AM2315

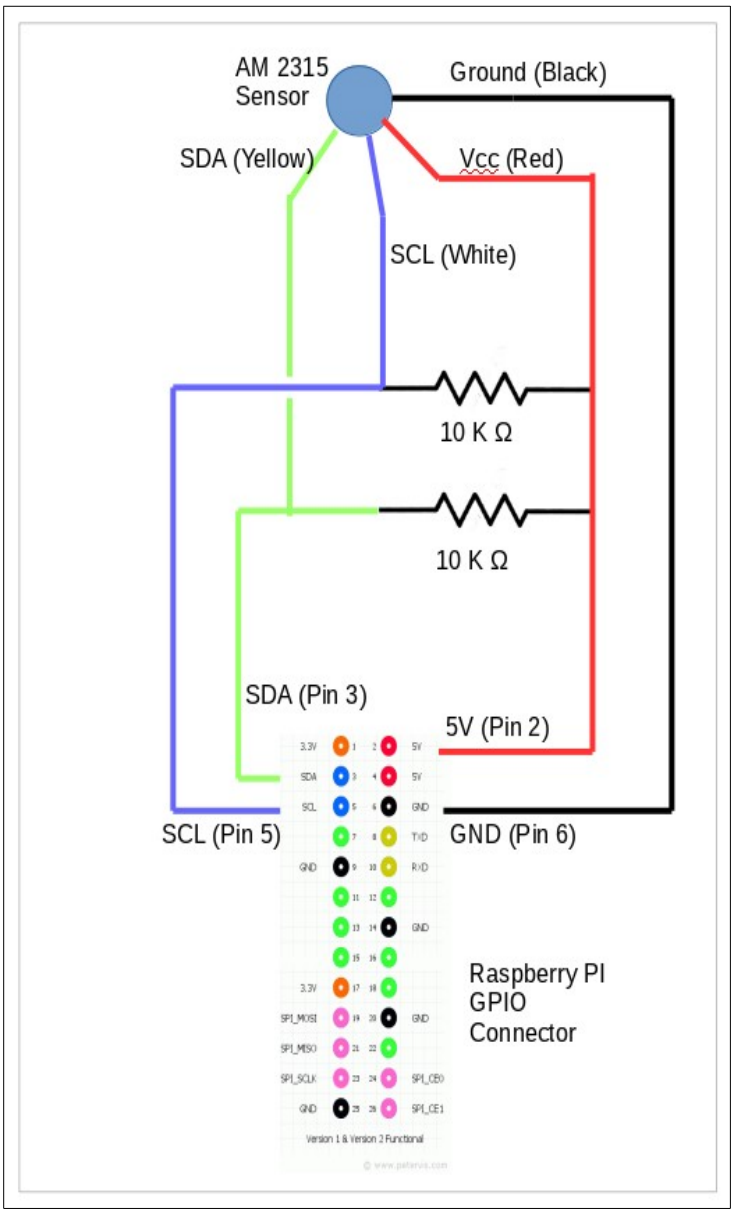
Shut down your Pi and remove the power lead while you wire up your AM2315 sensor. One of the really nice things about the Pi design is the consistency in the GPIO pins across models. The GPIO pins used by the AM2315 are the same across all Pi versions.

Most AM2315 documentation advise the installation of two pull-up resistors for the SDA and SCL leads. I have found that this is not an absolute requirement. I have used the AM2315 without these resistors and it works just fine.

If you are having trouble getting your device to work properly, or it is inconsistent in its output, you may want to consider installing the two resistors. The resistors are placed between the AM2315 SDA and SCL leads and the power lead.

Wire your device as shown in Figure-4 below.

Figure-4



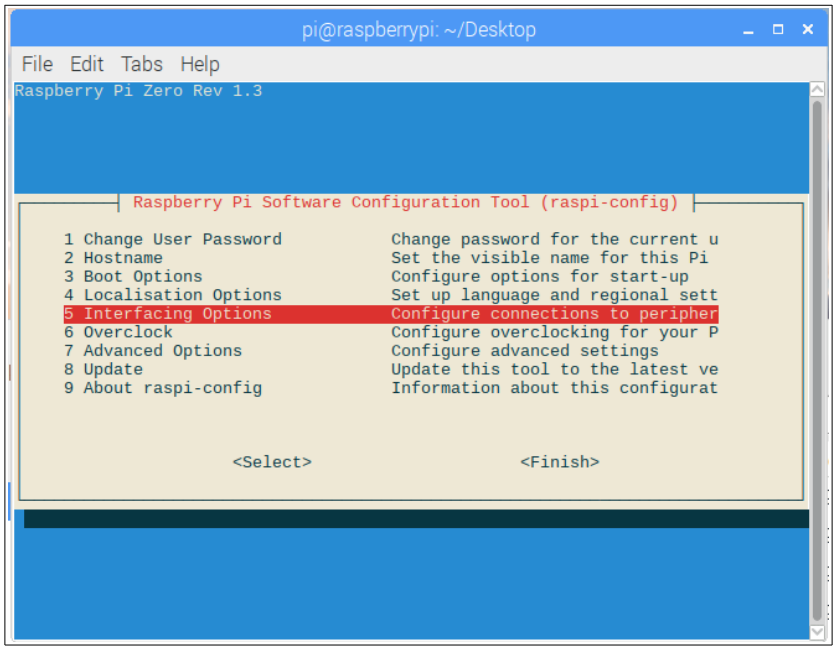
Source: http://www.adafruit.com/forums/viewtopic.php?f=45&t=48285&start=30

Step-4 - Configure i2c and test the AM2315 hardware

Before you begin this step, take a minute and look over your wiring to ensure each lead is on the right GPIO pin, the correct AM2315 lead is connected to the right GPIO pin. If you installed resistors, make sure they are in place where they belong. When everything looks good, fire up your Pi and open a command window.

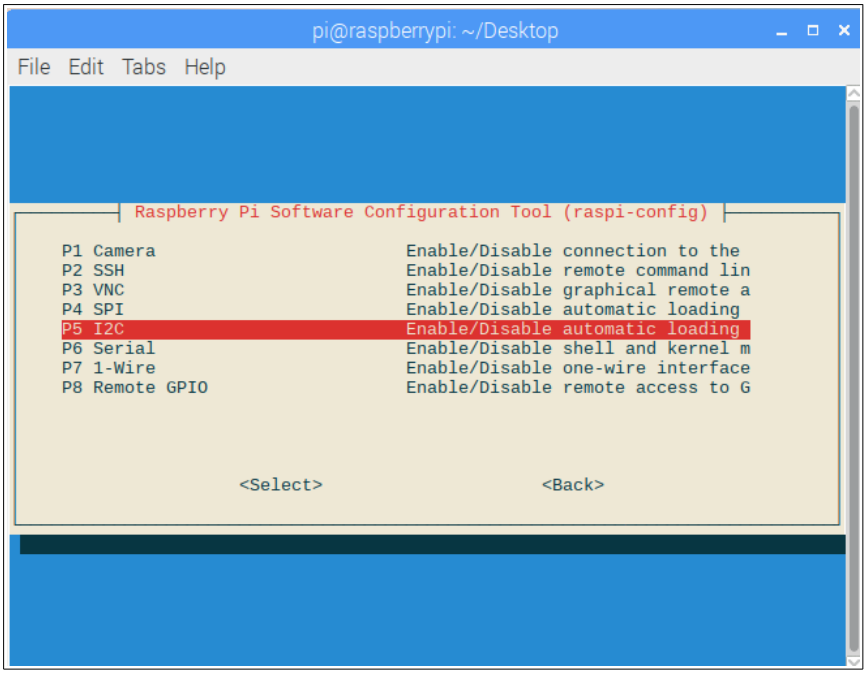
The first thing to do is enable the i2c interface on your Pi. This is done with the *raspi-config* utility. From the command window run the command: $*sudo raspi-config.* You will see the window shown in Figure-5.

Figure-5.



Press <Enter> on Line-5 and you will see the window shown in Figure-6.

Figure-6



Press <Enter> and you will see the window shown in Figure-7.

Figure-7

Select <Yes> and press <Enter> and you will see the window shown in Figure-8.

Figure-8

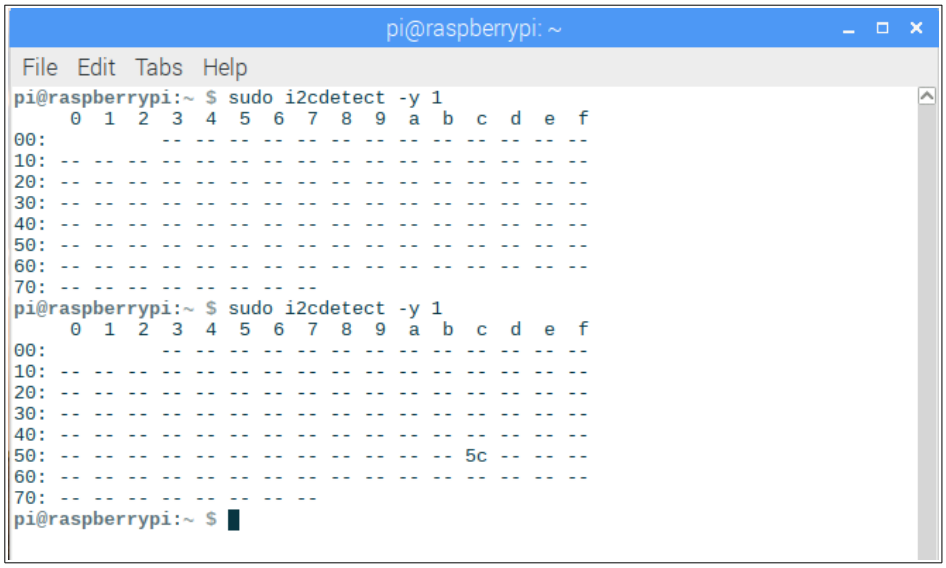
Exit *raspi-config.* Just to be safe, reboot your Pi: $*sudo reboot*.

Once you Pi is back online, let's make sure the AM2315 is wired correctly. Open a command window and type the command: $*sudo i2cdetect -y 1.*

You will need to enter this command two times about 1 second apart. This is due to the “sleep” mode of the device. You have to send it a signal to wake up, and then the command you want it to perform.

The first time you run the command, you will see output shown in the top ½ of Figure-6. No device was found. Running it a second time you should see the device address (0x5c) in the output. This is shown in the bottom ½ of Figure-9.

Figure-9



You will have to experiment with the delay between the two *i2cdetect* commands. It the delay is too long or too short you will not detect the device. On my Pi, a delay of about 1 second works. *Hint*: You can use the up-arrow key to quickly repeat commands entered in a console window without retyping.

If you cannot detect the device (0x5c never appears) then you must go back and review your wiring. The most common problem is the mis-wiring of the resistors is you installed them. The second most common is using the wrong GPIO pins on the Pi. Do not continue with any more steps until you get this working. Do not blame the device. They are quite hardy and seldom fail unless you smoke them with too much voltage.

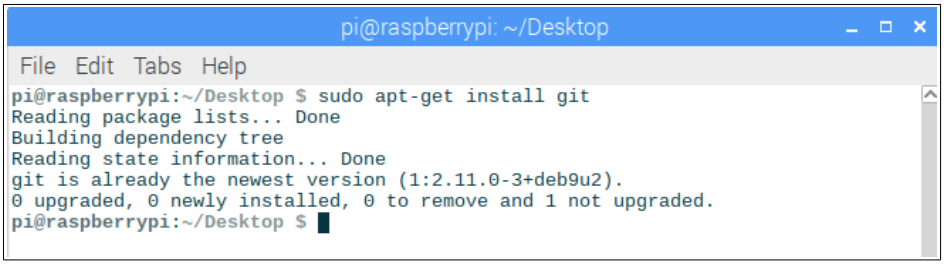
Step-5 - Install the Adafruit Python GPIO module

The Adafruit Python GPIO takes care of all the low-level communication with the Raspberry Pi GPIO ports. Installing it is a snap.

Open a command window and move to the *Download* folder: $cd *Download*.

Next, make sure *git* is installed on your Pi. Enter the command: $*sudo apt-get install git.*
You should see the output shown below in Figure-10.

Figure-10

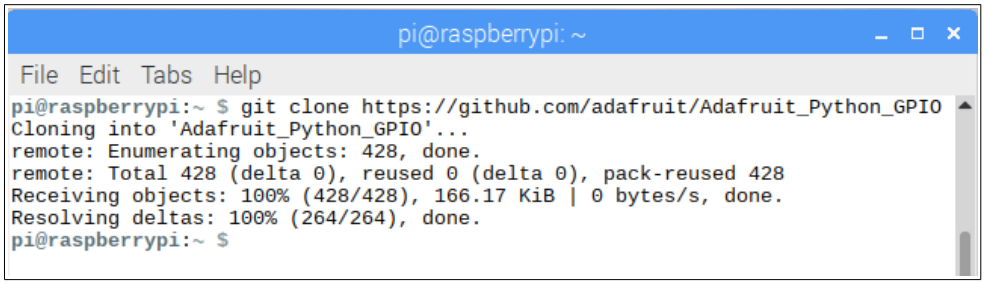


If *git* is not installed on your Pi, the above command will install it.

Next, enter the command: $*git clone https://github.com/adafruit/Adafruit_Python_GPIO*.
This will download the source code for the Adafruit GPIO module.

You should see the output shown in Figure-11.

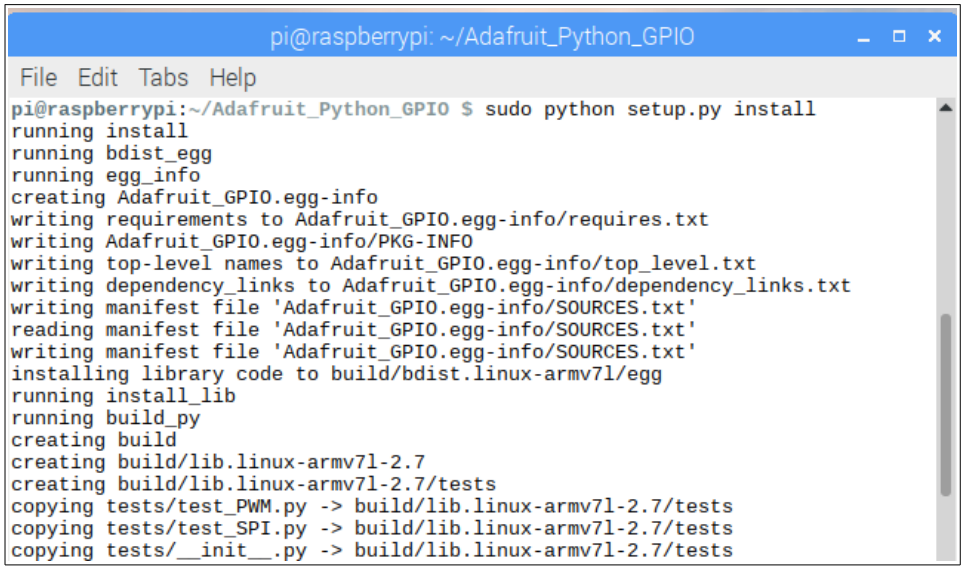
Figure-11



The above command will create a folder named *Adafruit_Python_GPIO.* Move into that folder using the command: $*cd Adafruit_Python_GPIO.*

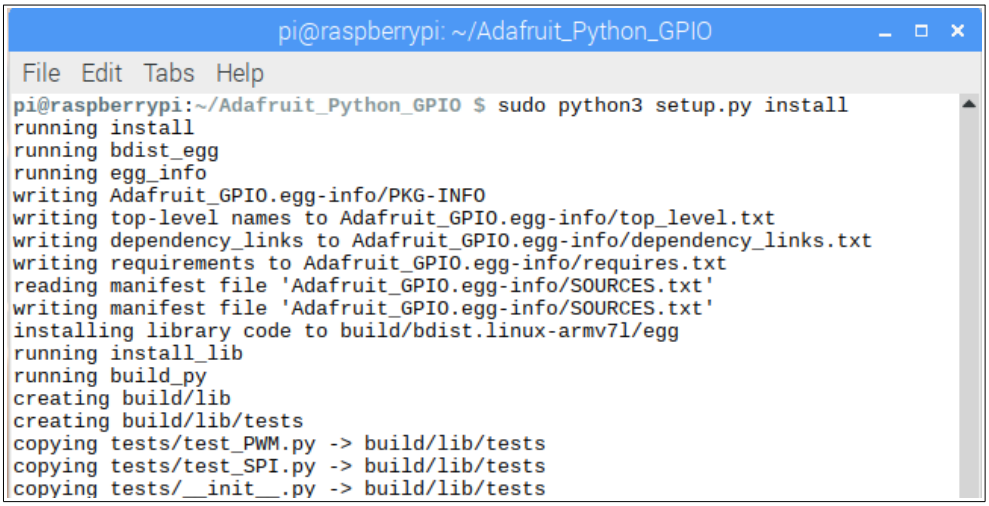
Next, build and install the library using the command: $*sudo python setup.py install*.
The build will start as shown in Figure-12.

Figure-12



We need the Adafruit module to work with Python3, so repeat the above step using python3 instead of python (Figure-13).

Figure-13



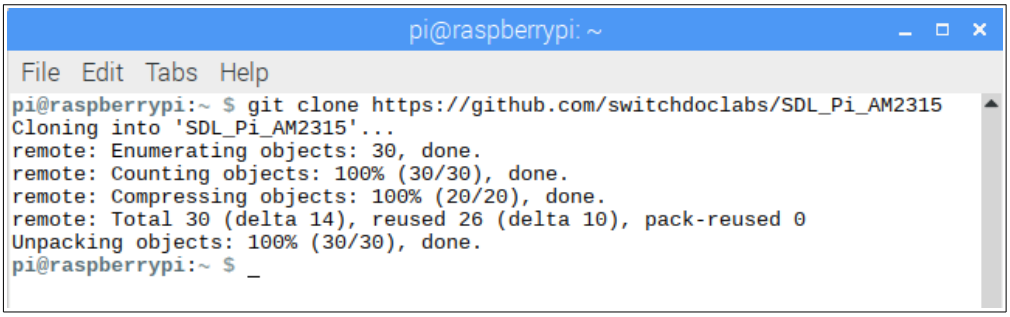
Once this step completes, download the SwitchDoc Labs code.

Change back to your home folder: $cd ~.

Next, enter the command: $*git clone https://github.com/switchdoclabs/SDL_Pi_AM2315*.
This will download the source code for the SwitchDoc Labs AM2315 module.

You should see the output shown in Figure-14.

Figure-14



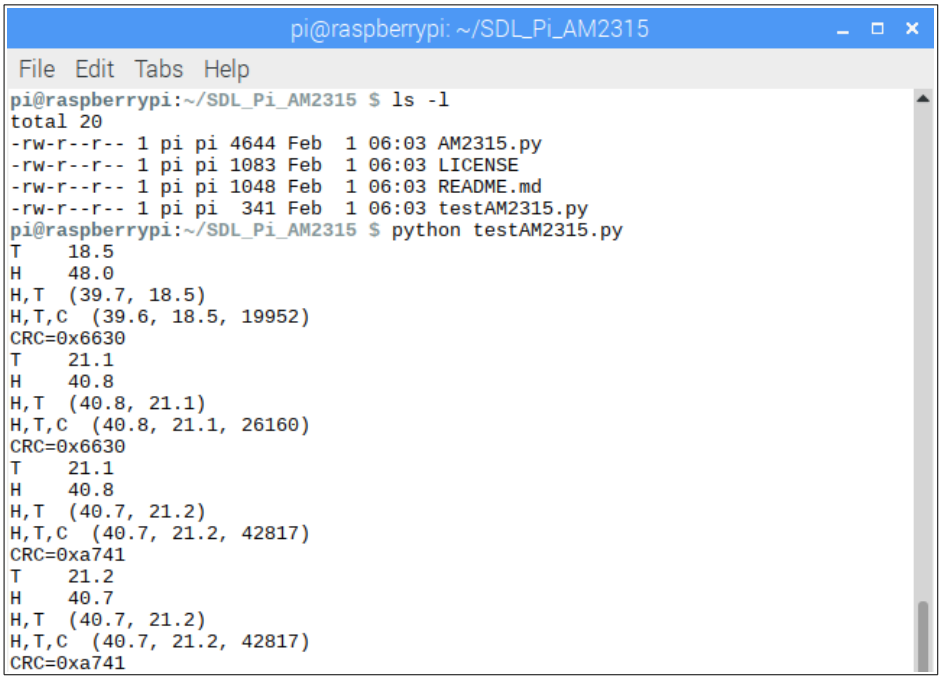
Step-6 - Test the AM2315 software

Let's run a quick test using the SwitchDoc Labs test program. Move into the SwitchDoc labs folder.
$cd SDL_Pi_AM2315

Run the test script.
$python testAM2315

The script will run in a loop until you kill it with <Ctrl><c>. If everything is working correctly you should see the output shown in Figure-15.

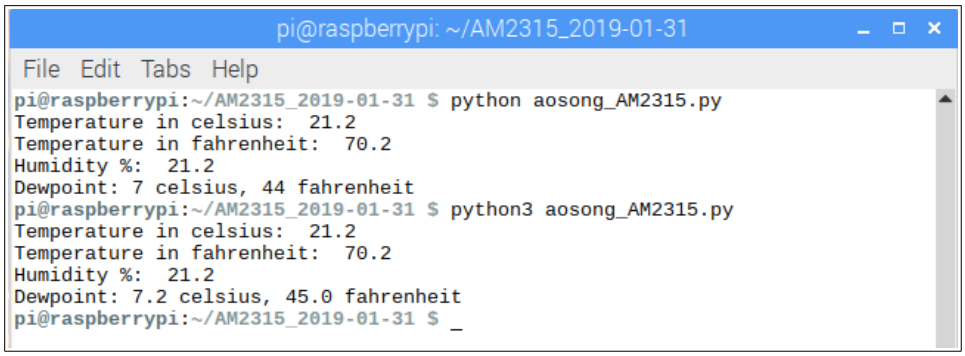
Figure-15



NOTE: The SwitchDoc labs code will not work in Python3.

I modified the well-written SwitchDoc labs Python code so that it works with Python3. I also added the ability to obtain temperatures in Fahrenheit, and added my dewpoint calculation function. My script is named aosong_AM2315.py. You can see the output of the script in Figure-16.

Figure-16



Study the source code and tweak it to fit your needs. If you are new to Python, the script is simple enough to show you how easy Python is to learn.

Summary

The implementation of the AM2315 temperature sensor code no longer requires any C library dependencies. With this new code, Python2.7 and Python3 are supported. The *Raspian Stretch* OS runs on every Pi ever built. This means you can connect your AM2315 to any Pi you want.

Contact me if you have any issues getting you sensor working. Bug reports and enhancement requests are also welcome. Send them to sopwith@ismellsmoke.net.

Sopwith - February 1, 2019