Raspberry Pi NOAA Weather Satellite Receiver

by **haslettj** on January 5, 2017

Table of Contents


Author:haslettj My YouTube Channel
I'm a ham, a tinker, a maker. I like interesting things. I'm a technophile, a wood turner, and a pen maker.

Intro: Raspberry Pi NOAA Weather Satellite Receiver

A new pile of parts means a new project is in the works.

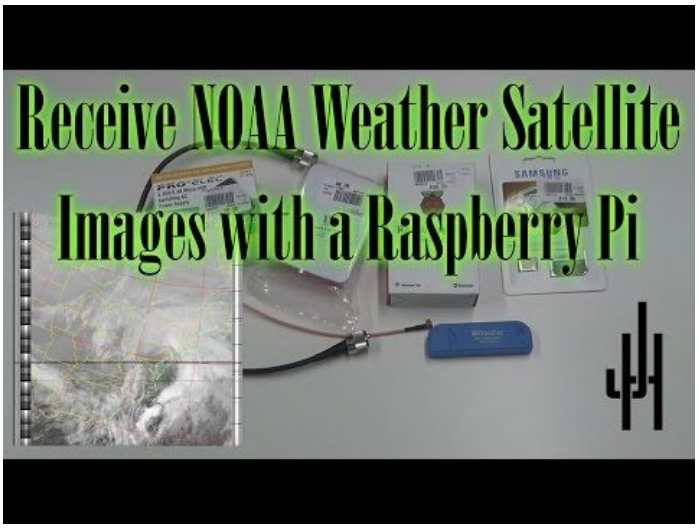
I'm building a Raspberry Pi based NOAA weather satellite receiver. I've got some experience with receiving signals from NOAA satellites, but will be a little different, as all of the work is going to be done automatically on a Raspberry Pi. This will allow me to place the receiver right at the antenna, which is mounted in my attic. With a very short piece of coax, reducing feed-line loss.

I'll be using the following list of parts:

- Raspberry Pi 3 Model B
- MicroUSB Power Supply
- Generic Raspberry Pi 3 Case
- 32 GB Micro SD Card
- NooElec SDR Dongle
- QFH Antenna

This project will require some basic Linux skills. If you are not comfortable navigating around the Linux command line, you may need to get some help if you want to follow along.

This instructable is going to focus on getting the Raspberry Pi working to receive images. Details around building a proper antenna can be found from other sources. I even made a video about it: https://youtu.be/KU75FSA6o2M



Step 1: Prepare the Raspberry Pi

The first step is getting the Raspberry Pi up and running. I'm using Raspbian (Jessie 2016-11-25).

There are plenty of tutorials on getting Raspbian set up and booting on the Raspberry Pi. You should have no difficulty in finding one:
https://www.raspberrypi.org/documentation/installation/installing-images/README.md

Once you have the Raspbian image copied to your micro SD card, boot the Raspberry Pi. You'll need to get it on the network and enable SSH. You can certainly use either a wired Ethernet connection, or wireless. I'm using wireless, as I don't have a wired connection in my attic.

Enable SSH using the Raspberry Pi Configuration utility in the Preferences menu. You may want to set it to boot to CLI while you are in there. Since this is going to be a headless setup, there is no need to waste resources on a GUI.

Next we want to make sure the Raspberry Pi is fully up-to-date. Issue the following commands:

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

Step 2: Install the Necessary Software

The first thing we need is the **USB drivers** for the RTL dongle:

```
sudo apt-get install libusb-1.0
```

Since we need to build the latest version of rtl-sdr to actually make use of the RTL dongle, we'll need to install **cmake**:

```
sudo apt-get install cmake
```

We need to make sure the Raspberry Pi doesn't load any SDR kernel modules that will interfere with the rtl-sdr sofware.

Using your favorite text editor, create a new file named /etc/modprobe.d/no-rtl.conf and put the following text in the file. You need to run that text editor as sudo (i.e. 'sudo vi' or 'sudo nano' etc) to write to the modprobe.d directory:

```
blacklist dvb_usb_rtl28xxu
blacklist rtl2832
blacklist rtl2830
```

Install the most recent build of **rtl-sdr**:

```
cd ~
git clone https://github.com/keenerd/rtl-sdr.git
cd rtl-sdr/
mkdir build
cd build
cmake ../ -DINSTALL_UDEV_RULES=ON
make
sudo make install
sudo ldconfig
cd ~
sudo cp ./rtl-sdr/rtl-sdr.rules /etc/udev/rules.d/
sudo reboot
```

We'll need the **sox** audio toolkit in order to manipulate the received audio stream:

```
sudo apt-get install sox
```

We need a way to schedule the captures to happen as the satellites pass overhead. Install the **at** scheduler:

```
sudo apt-get install at
```

We need a way to know when the satellites will pass overhead. We'll use an application called **predict**:

```
sudo apt-get install predict
```

Finally, we'll need **wxtoimg** to convert the captured audio stream to an actual image:

```
cd ~
wget http://www.wxtoimg.com/beta/wxtoimg-armhf-2.11.2-beta.deb
sudo dpkg -i wxtoimg-armhf-2.11.2-beta.deb
```

Step 3: Testing Things Out

Once all the software is installed, make sure your USB dongle is connected and run the following command:

```
sudo rtl_test
```

You should see the following output:

```
Found 1 device(s):
 0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4 43.9
Sampling at 2048000 S/s.

Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.

Reading samples in async mode...
```

If you receive any error messages, you'll need to troubleshoot them before continuing. If your RTL Dongle isn't working at this point, it won't do any good to continue.

You need to run predict one time to set your ground station location. To do that you'll need your latitude and longitude. You can get your latitude and longitude from google maps by searching for your address, the right clicking on the pointer and selecting "What's Here". One thing to note is that Google displays North as positive and East as positive numbers. Predict uses North as positive but **WEST** as positive. Make sure to adjust accordingly, or your predictions will be no good.

Run predict from the command line, and select option 'G'. Enter your ground station information and exit the program.:

```
predict


                      --== PREDICT  v2.2.3 ==--
                 Released by John A. Magliacane, KD2BD
                               May 2006


                         --==[ Main Menu ]==--

[P]: Predict Satellite Passes          [I]: Program Information
[V]: Predict Visible Passes            [G]: Edit Ground Station Information
[S]: Solar Illumination Predictions    [D]: Display Satellite Orbital Data
[L]: Lunar Predictions                 [U]: Update Sat Elements From File
```

```
[O]: Solar Predictions                [E]: Manually Edit Orbital Elements
[T]: Single Satellite Tracking Mode   [B]: Edit Transponder Database
[M]: Multi-Satellite Tracking Mode    [Q]: Exit PREDICT

               *  Ground Station Location Editing Utility  *


                     Station Callsign  : W1AW
                     Station Latitude  : 41.7169 [DegN]
                     Station Longitude : 72.7271 [DegW]
                     Station Altitude  : 25 [m]


          Enter the callsign or identifier of your ground station
```

You need to run **wxtoimg** once to accept the terms and conditions.

```
wxtoimg
```

You need to tell wxtoimg where your base station is at, so it can properly generate an overlay map. You do so by creating a file in your home directory named ~/.wxtoimgrc. In this file, North is positive, as with predict, however **EAST** is positive, which is the opposite of what predict uses. Make sure to adjust your values appropriately.

Using your favorite text editor, create a new file named ~/.wxtoimgrc and put the following text in the file, substituting your values:

```
Latitude: 41.7169
Longitude: -72.7271
Altitude: 25
```

Step 4: The Scripts

Now that you've got a working RTL dongle on your Raspberry Pi, it's time to actually make it receive some weather maps. Make sure your antenna is connected and located as best as possible. I've got mine mounted in my attic and it works fairly well from there. If you can mount it outside, that would be even better. Get it as high as possible.

We are going to need to create a few scripts to automate everything. This is where that Linux familiarity is going to come in.

First we'll create a couple directories to hold our files:

```
cd ~
mkdir weather
cd weather
mkdir predict
cd predict
```

Next we'll make the two scripts that kick off the scheduling. The first is '**schedule_all.sh**'. This script will be called nightly at midnight. It downloads satellite pass information from celestrak and creates a TLE file for predict to use. Then it removes all AT jobs from the system, so no passes get double scheduled. Finally, it calls the second script '**schedule_satellite.sh**' for each satellite we are interested in.

Using your favorite text editor, create a new file in ~/weather/predict named **schedule_all.sh** and put the following code in the file:

```
#!/bin/bash

# Update Satellite Information

wget -qr https://www.celestrak.com/NORAD/elements/weather.txt -O /home/pi/weather/predict/weather.txt
grep "NOAA 15" /home/pi/weather/predict/weather.txt -A 2 > /home/pi/weather/predict/weather.tle
grep "NOAA 18" /home/pi/weather/predict/weather.txt -A 2 >> /home/pi/weather/predict/weather.tle
grep "NOAA 19" /home/pi/weather/predict/weather.txt -A 2 >> /home/pi/weather/predict/weather.tle
grep "METEOR-M 2" /home/pi/weather/predict/weather.txt -A 2 >> /home/pi/weather/predict/weather.tle



#Remove all AT jobs

for i in `atq | awk '{print $1}'`;do atrm $i;done


#Schedule Satellite Passes:

/home/pi/weather/predict/schedule_satellite.sh "NOAA 19" 137.1000
/home/pi/weather/predict/schedule_satellite.sh "NOAA 18" 137.9125
/home/pi/weather/predict/schedule_satellite.sh "NOAA 15" 137.6200
```

The second script, '**schedule_satellite.sh**', recurses through each pass of the given satellite for the current day. It determines if the maximum elevation is 20 degrees or greater. If it is, then it calculates the length of the pass, and schedules the recording and processing of the pass. If the maximum elevation is less than 20 degrees, the pass is ignored, as it generally won't produce a decent image.

Using your favorite text editor, create a new file in ~/weather/predict named **schedule_satellite.sh** and put the following code in the file:

```
#!/bin/bash
PREDICTION_START=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}" | head -1`
PREDICTION_END=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}" | tail -1`


var2=`echo $PREDICTION_END | cut -d " " -f 1`

MAXELEV=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}" | awk -v max=0 '{if($5>max){max=$5}}END{print max}'`

while [ `date --date="TZ=\"UTC\" @${var2}" +%D` == `date +%D` ]; do

START_TIME=`echo $PREDICTION_START | cut -d " " -f 3-4`
var1=`echo $PREDICTION_START | cut -d " " -f 1`

var3=`echo $START_TIME | cut -d " " -f 2 | cut -d ":" -f 3`
```

```
TIMER=`expr $var2 - $var1 + $var3`

OUTDATE=`date --date="TZ=\"UTC\" $START_TIME" +%Y%m%d-%H%M%S`

if [ $MAXELEV -gt 19 ]
 then
   echo ${1//" "}${OUTDATE} $MAXELEV

   echo "/home/pi/weather/predict/receive_and_process_satellite.sh \"${1}\" $2 /home/pi/weather/${1//" "}${OUTDATE} /home/pi/weather/predict/weather.t

fi

nextpredict=`expr $var2 + 60`

PREDICTION_START=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}" $nextpredict | head -1`
PREDICTION_END=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}"  $nextpredict | tail -1`

MAXELEV=`/usr/bin/predict -t /home/pi/weather/predict/weather.tle -p "${1}" $nextpredict | awk -v max=0 '{if($5>max){max=$5}}END{print max}'`

var2=`echo $PREDICTION_END | cut -d " " -f 1`

done
```

When the time comes for a give pass to be recorded and processed, at kicks off the final script '**receive_and_process_satellite.sh**'. When this script is called, it uses rtl_fm to receive the audio from the satellite pass, and sends that audio to sox for processing. Sox saves the audio to a file. Once the pass is complete, wxmap is called to generate an overlay map for the image. Finally, wxtoimg is called to generate the actual image and place the overlay map on it.

Using your favorite text editor, create a new file in ~/weather/predict named **receive_and_process_satellite.sh** and put the following code in the file:

```
#!/bin/bash

# $1 = Satellite Name
# $2 = Frequency
# $3 = FileName base
# $4 = TLE File
# $5 = EPOC start time
# $6 = Time to capture

sudo timeout $6 rtl_fm -f ${2}M -s 60k -g 45 -p 55 -E wav -E deemp -F 9 - | sox -t wav - $3.wav rate 11025

PassStart=`expr $5 + 90`

if [ -e $3.wav ]
 then
   /usr/local/bin/wxmap -T "${1}" -H $4 -p 0 -l 0 -o $PassStart ${3}-map.png

   /usr/local/bin/wxtoimg -m ${3}-map.png -e ZA $3.wav $3.png
fi
```

Once all three scripts have been created, we need to make the executable, by issuing the following commands:

```
chmod +x schedule_all.sh
chmod +x schedule_satellite.sh
chmod +x receive_and_process_satellite.sh
```

Now, the last step in automating the reception is to schedule the first script to run just after midnight. Run the following command:

```
crontab -e
```

It will open a text editor of your choice to edit the cron file. Simply add the following line to the bottom of this file and save:

```
1 0 * * * /home/pi/weather/predict/schedule_all.sh
```

Now, at 00:01 the schedule will start. (if you don't want to wait, simply run the following command once to kick the process off:)

```
/home/pi/weather/predict/schedule_all.sh
```

Step 5: Check things out

To see upcoming satellite passes that are scheduled to be processed, run the following command:

```
atq
```

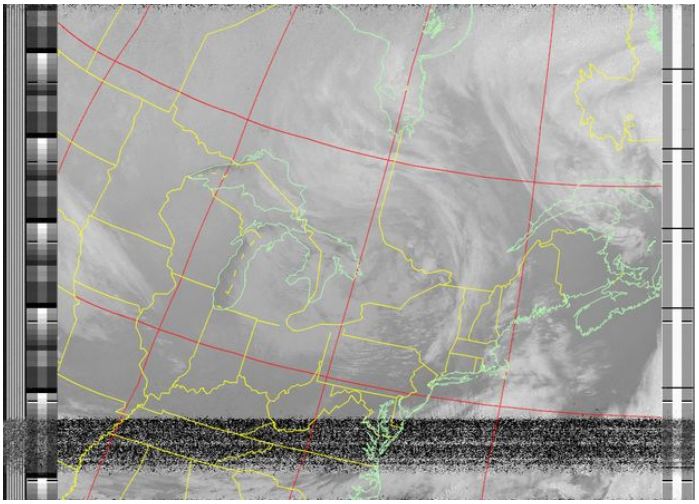
Check the ~/weather directory for your results. Each pass will generate three files:

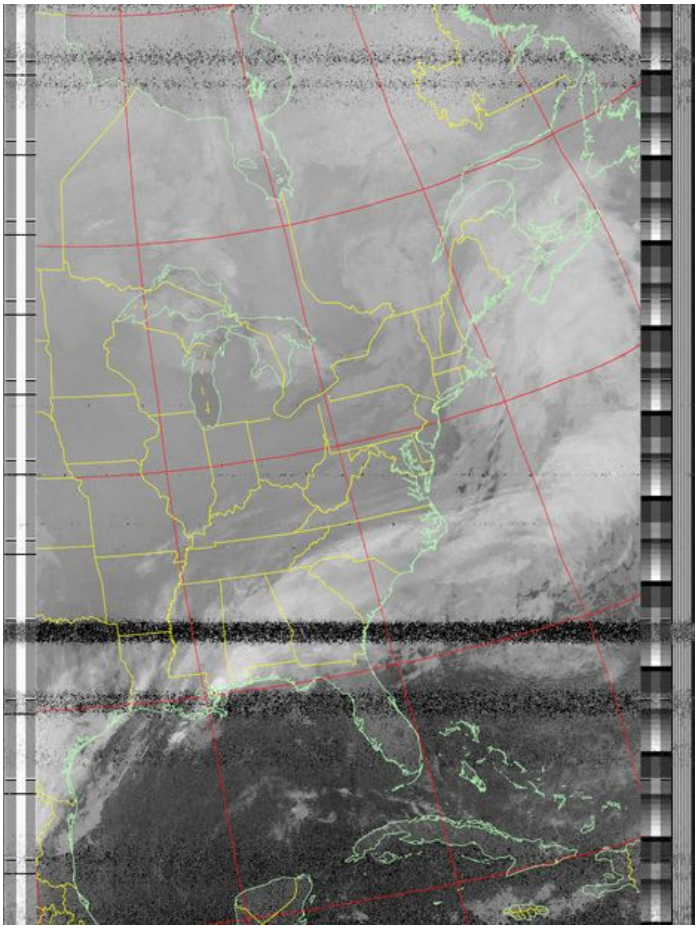
```
NOAA1820170105-181147-map.png  <-- Map overlay file
NOAA1820170105-181147.png      <-- The final image file
NOAA1820170105-181147.wav      <-- The raw audio file
```

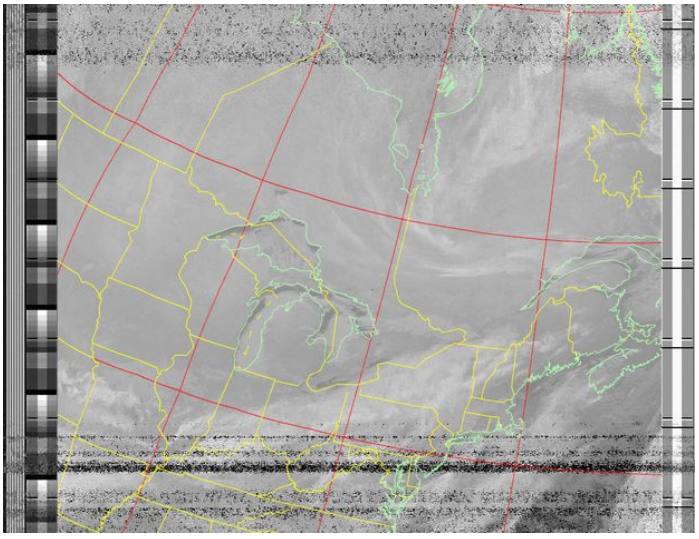
The format of the file name is the satellite name, followed by the date (YYYYMMDD) and time (HHMMSS) of the start of the pass. There are several filters that wxtoimg can use when generating the image file. I've had different luck with different satellites and passes on each filter. Feel free to play around with wxtoimg on the original map and audio files and see what you come up with.

One thing to note, the receiver can only receive one satellite at a time. If two satellites are passing overhead at the same time, the first one to start recording will "win" and be recorded. The second recording will error out and stop.

Step 6: Some of my results







Related Instructables

rtl-sdr on Ubuntu by braingram

RTL-SDR FM radio receiver with GNU Radio Companion by v3l0c1r4pt0r

SDR on pcDuino by jingfeng

Walk-in Talkin' Closet (Closet that talks to you and helps you!) by ErikZ12

Downloading NOAA satellite images cheaply and easily by sdobbie

Track Airplanes with RTL SDR and ADS-B! by rickosgood

Comments