Report on logging SID data from an UKRAA VLF receiver using an AB electronics UK ADC Pi and a Raspberry Pi 4 Model B 2GB connected to UKRAA VLF aerial, VLF aerial tuning unit and VLF receiver

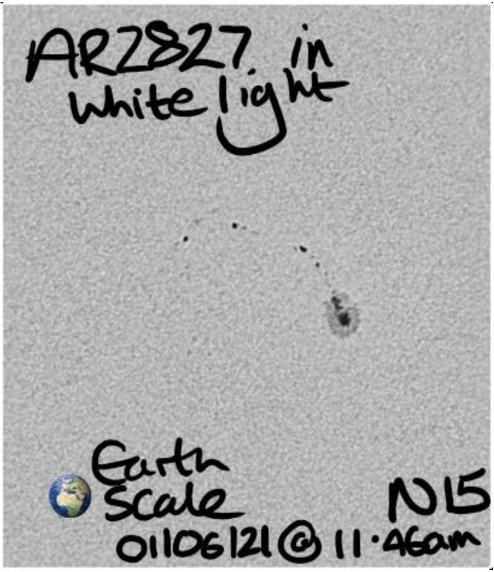
Richard Knott

January 2022

Contents

Background

I usually observe the sun in white light (with Baader Herschel wedge)



Or H-aplha (either with Quark or modified PST)

With the fickle UK weather (cloudy) I wanted to explore another wavelength of the EM spectrum. Clearly I could take the next "visible" wavelength of Calcium-K, but that was outside of my price range at the time. However reviewing some of my solar astronomy books (BAA Observing Guide to the Sun) I was aware of the potential of radio astronomy. I had always considered this to be out of my scope but the VLF monitoring for solar flares looked interesting. I also stumbled upon the Flamsteed Astronomy Society write up of their VLF receiver – they had highlighted that they were using a raspberry pi with attached ADC to record the VLF data. I was intrigued and thought that this was within my capabilitiesand made contact asking if they would share their python code – I received a reply that they were rewritting their code at that time and changing their set-up – so I thought I would give it a go myself.

I sourced the VLF aerial and receiver in kit form from UKRAA – together with prebuilt VLF ATU and VLF signal generator and proceded to buid the kit. Once finished I was thrilled at getting a reading on my voltmeter but recognised that I could not sit there all day and look at the changing number – no matter how exciting the changing values appeared appeard to be.

Reviewing the VLF receiver manual issue 1.3 November 2015 (the one available when I build my recevice kit) – UKRAA highlighted the use of a data logger (PicoLog) connected to a PC, or Radio Sky Pipe using additional Maxim MAX 186 (my board did not have this option) connected to a PC, or UKRAA Starbase application with contoller (no longer available?) connected to a PC – also available on the UKRAA website at the time was the LabJack U3-HV for connecting to Radio Sky Pipe and a PC. I am a bit of a skin flint and did not want to spend a lot on converting analogue signal to a digital system or have a full desktop PC running all the time. Having experience with raspberry pi's from previous projects that I have running 24/7 I was keen to explore the possibility of using a pi to gather, process and display the information from the VLF receiver.

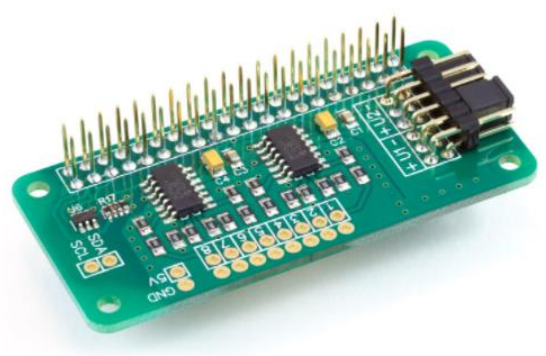
I use Pi Hut (https://thepihut.com/) when purchasing Raspberry Pi equipment – so a quick search on their website indicated a wealth of possibilites that could plug directly into the GPIO header on the pi.

Knowing that the VLF receiver has two potential outputs 

P10 0-2.5V and P7 0-5V, I was able to narrow my search down to any pi HAT ADC that could accept 0-5V (the bigger the potential difference the better) rather than the 0-2.5V (could not find an ADC for this potential difference range).

Reviewing the option available through Pi Hut I selected the AB Electronics ADC Pi as its input range was 0-5.06V rather than ADC Differential Pi that had an input of +/- 2.048V.

ADC Pi



AB Electronics web site for the ADC Pi (https://www.abelectronics.co.uk/p/69/adc-pi-raspberry-pi-analogue-to-digital-converter)also had a wealth of information about the board and links to their code libraries available through GitHub (https://github.com/abelectronicsuk/).

Getting the aerial data

I connected 

P7 (0-5V analogue output) to a 3 pin GX12 socket that I had on the front panel of the enclosure (Hammond 1455N1601) I used for my VLF Receiver (gave me the option of taking 2.5V and 5V analogue outputs to front panel should I latter use the 0-2.5V analogue output)

This then connects with the 2 pin header I put into input 1 of the AB Electronics ADC Pi HAT. The AB Electronics ADC Pi HAT pugs directly into the Raspberry Pi GPIO header.

I followed the instruction from the AB Electronics web site – see appendix 7 : notes from my VLF project book.

I used the AB Electronics sample code as a basis for the GetDataAerial.py code that I wrote to record and save the VLF receiver analogue 0-5V output. The python script is run in the background and is checked by the ResetGetDataAerial.py script every 5 minutes via cron to ensure that it is running (I want to change this to a service script and this will get away from running the ResetGetDataAerial.py script every 5 minutes via cron – but it ain't broke yet…). The code is effectively an infinite loop – so runs continually in the background.

The python GetDataAerial.py code has features of the other GetDataXxxx.py code GetData – it check that the folder structure is there – if not it creates it.

The data is stored in a simple text file that can be assessed by gnuplot as data for plotting.

I have not reviewed this code for some time, so it is a mixture of CamelCase and Snake-Case syntax (bad) – but it works (for me).

A new file is created at the start of each day 00:00am UTC and data is appended each minute.

I record the potential difference from the VLF receiver every minute.

I spent some time ensuring that the potential difference was always taken at the start of each minute – this was achieved using the "time.sleep(60.0 – ((time.time()) % 60.0))" statement in the while true section of the code. Why did I do this – so that there would be correlation between GOES-16 X-ray flux timestamps and that which I recorded from the VLF receiver.

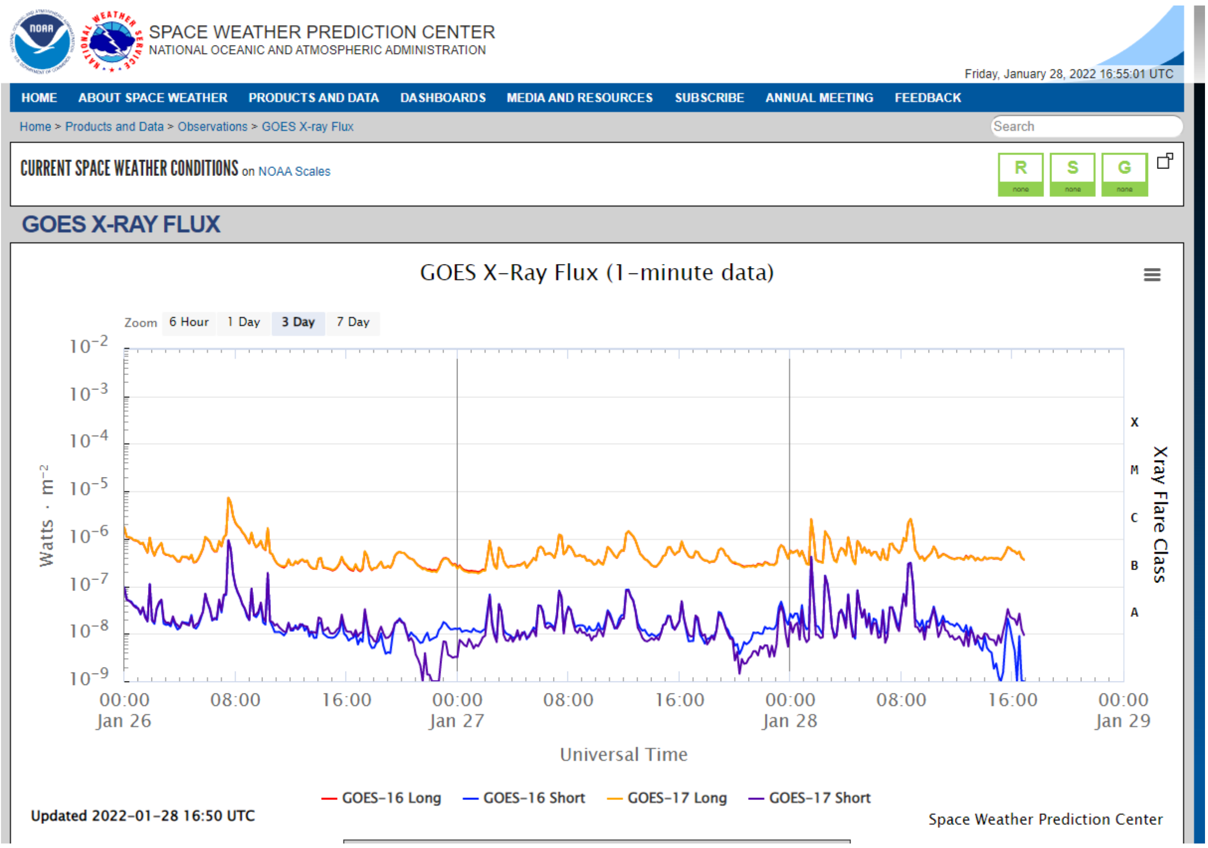
Code used for obtaining the VLF Receiver analogue output is included in Appendix 4.

Getting Goes 16 X-ray flux data

Recording data from the UKRAA VLF receiver is great with the AB Electronics Pi ADC Hat.

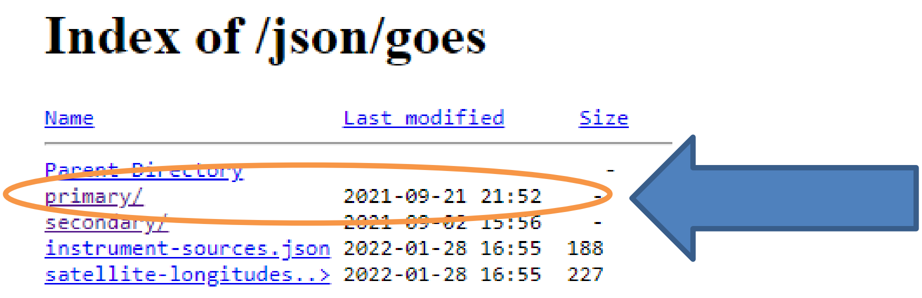
But one wants to compare this to actual X-ray flux data from our Sun to see if there is correlation between SID events recorded via the UKRAA VLF receiver and those recorded elsewhere.

Real time X-ray flux data is presented for the GOES 16 satellite through the Space Weather Prediction Centre (swpc) web site (URL: https://www.swpc.noaa.gov/products/goes-x-ray-flux)



Historic numerical data is available through the swpc web portal (URL: https://services.swpc.noaa.gov/json/goes/)

It is the primary data that I was interested in – this is from GOES-16 (secondary data is from GOES-17).



Index of /json/goes

Name	Last modified	Size
Parent Directory		-
primary/	2021-09-21 21:52	-
secondary/	2021-09-02 15:56	-
instrument-sources.json	2022-01-28 16:55	188
satellite-longitudes..>	2022-01-28 16:55	227

Reviewing the primary data – identified “xrays-3-day.json” as data required.

Index of /json/goes/primary

Name	Last modified	Size
Parent Directory		-
suvi/	2021-09-21 21:52	-
differential-electro..>	2022-01-28 16:56	284K
differential-electro..>	2022-01-28 16:56	838K
differential-electro..>	2022-01-28 16:56	71K
differential-electro..>	2022-01-28 16:57	2.0M
differential-protons..>	2022-01-28 16:56	527K
differential-protons..>	2022-01-28 16:57	1.6M
differential-protons..>	2022-01-28 16:56	131K
differential-protons..>	2022-01-28 16:57	3.7M
integral-electron-fl..>	2022-01-28 16:57	801
integral-electrons-1..>	2022-01-28 16:56	27K
integral-electrons-3..>	2022-01-28 16:56	80K
integral-electrons-6..>	2022-01-28 16:56	7.0K
integral-electrons-7..>	2022-01-28 16:57	194K
integral-proton-flue..>	2022-01-28 16:57	1.5K
integral-protons-1-d..>	2022-01-28 16:56	236K
integral-protons-3-d..>	2022-01-28 16:56	709K
integral-protons-6-h..>	2022-01-28 16:56	58K
integral-protons-7-d..>	2022-01-28 16:56	1.6M
integral-protons-plo..>	2022-01-28 16:56	119K
integral-protons-plo..>	2022-01-28 16:56	357K
integral-protons-plo..>	2022-01-28 16:56	29K
integral-protons-plo..>	2022-01-28 16:56	833K
magnetometers-1-day...>	2022-01-28 16:57	256K
magnetometers-3-day...>	2022-01-28 16:57	767K
magnetometers-6-hour..>	2022-01-28 16:56	64K
magnetometers-7-day...>	2022-01-28 16:57	1.7M
xray-background-7-da..>	2022-01-28 16:57	552
xray-flares-7-day.json	2022-01-28 16:56	30K
xray-flares-latest.json	2022-01-28 16:56	453
xrays-1-day.json	2022-01-28 16:57	641K
xrays-3-day.json	2022-01-28 16:57	1.9M
xrays-6-hour.json	2022-01-28 16:57	159K
xrays-7-day.json	2022-01-28 16:57	4.4M

Data can be accessed via a GET request from the Raspberry Pi.

Data is transferred to json dictionary and then processed to obtain short (0.05 to 0.4nm) x-ray flux and long (0.0 to 0.8nm) x-ray flux values at 60 second intervals from 00:00am to 24:00pm for yesterday and then stored in a text file in the x-ray data folder.

Code used for obtaining the GOES-16 x-ray flux data is included in Appendix 3.

Getting Sunrise and Sunset data

Since no solar flares are detectable by the VLF receiver at night-time, it seems sensible to identify two datum points for inclusion on the VLF plot – sunrise and sunset.

There are several web based sites that offer this information on a daily basis – however as this was to be an autonomous system then a scrapped solution was sort.

Googling solutions identified a simple API solution that could be accessed via a GET request to https://api.sunrise-sunset.org/json.

URL: https://sunrise-sunset.org/api

The web page has sample code and instruction for obtaining the required data.

Sunset and sunrise times API

We offer a **free API** that provides **sunset and sunrise times** for a given **latitude and longitude.**

Please note that **attribution is required** if you use our API. Check "Usage limits and attribution" section below for more information.

API documentation

Ours is a very simple REST api, you only have to do a GET request to **https://api.sunrise-sunset.org/json**.

Parameters

- **lat** (float): Latitude in decimal degrees. Required.
- **lng** (float): Longitude in decimal degrees. Required.
- **date** (string): Date in YYYY-MM-DD format. Also accepts other date formats and even relative date formats. If not present, date defaults to current date. Optional.
- **callback** (string): Callback function name for JSONP response. Optional.
- **formatted** (integer): 0 or 1 (1 is default). Time values in response will be expressed following ISO 8601 and day_length will be expressed in seconds. Optional.

Note : all times are UTC with no adjustment for summer light saving – this is good as data from GOES 16 X-ray flux is in UTC.

Code used for obtaining the sunrise and sunset times each day is included in Appendix 2.

Plotting the data

All well and good to have the aerial data and corresponding GOES 16 X-ray flux data in .txt format. However the data needs to be visualised to appreciate if there has been a SID event. Consideration of plotting programs was considered for the Raspberry Pi and gnuplot was selected as a suitable application.

gnuplot

A portable, multi-platform, command-line driven graphing utility

Brought to you by: broeker, cgaylord, lhecking, sfeam

Install gnuplot

To install gnuplot on the Raspberry Pi use the following command :

```
sudo apt-get install gnuplot-x11
```

You may have to answer "Y" if prompted.

Concept to have two plots on one graph:

Plot 1 : Y1 axis and X axis – where Y1 axis is linear scale of potential difference from the AB Electronics Pi ADC hat on top of the GPIO raspberry Pi header (VLF signal strength from 0-5V output of VLF receiver) range 0v to 5V and X axis is time based starting at 00:00 hours to 24:00 hours the previous day from stored aerial data folder for that date.

Plot 2 : Y2 axis and X axis – where Y2 axis is base 10 logarithmic scale of GOES 16 X-ray flux for long X-rays (0.1nm to 0.8nm) and X axis is time based starting at 00:00 hours to 24:00 hours the previous day from stored x-ray data folder for that date

Additional information to be added to graph:

Label 1 : down time of Ramsloh transmitter – 07:00am to 08:00am UTC each day

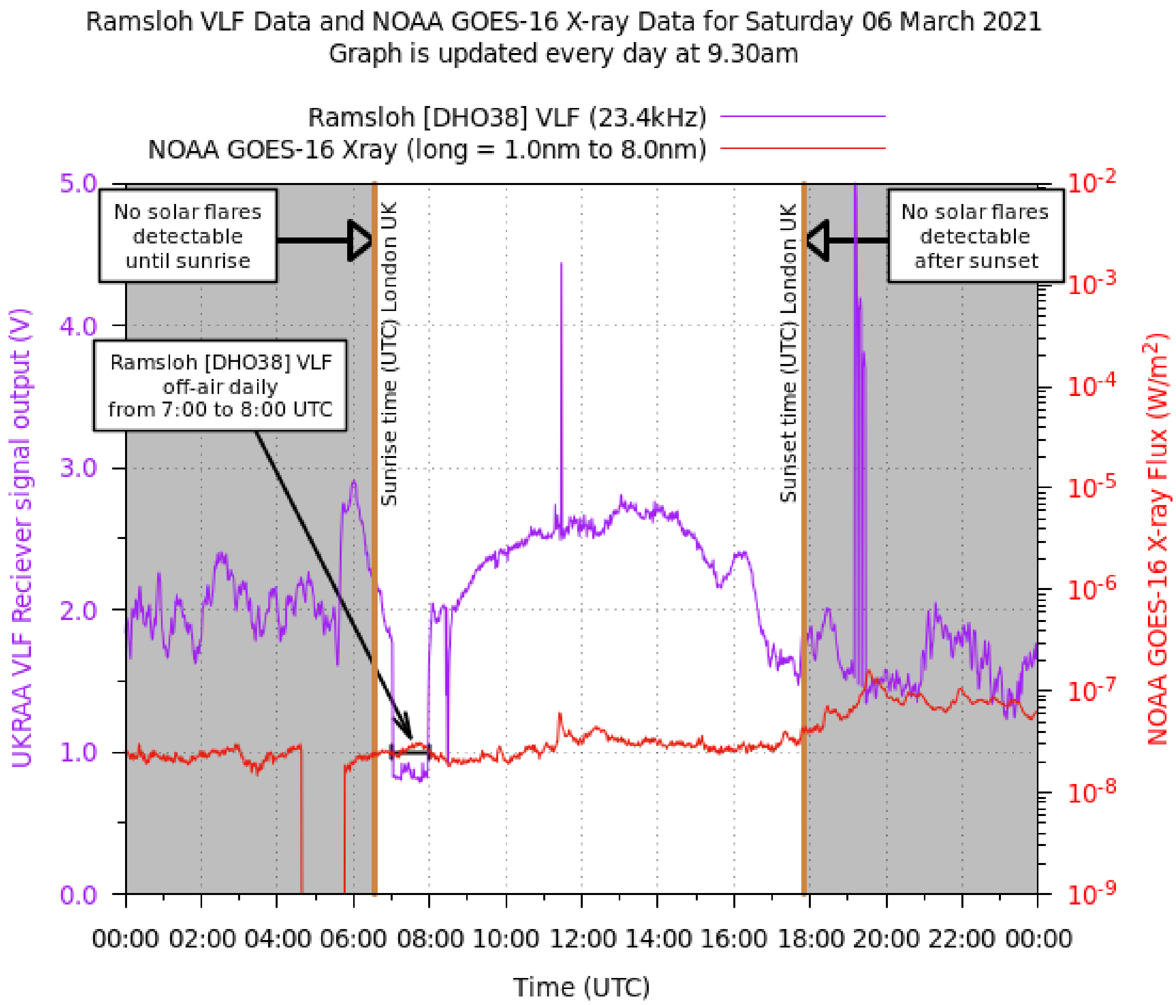
Label 2 : Sunrise time – vertical line on x-axis to identify when SID event detectable by VLF receiver.

Label 3 : Sunset time – vertical line on x-axis to identify when SID event detectable by VLF receiver.

Grey area 1 : from 00:00 to sunrise – to emphasise no SID detectable before sunrise

Grey area 2 : from sunset to 24:00 - to emphasise no SID detectable after sunset

Example plot:



Code used for plotting yesterday's VLF and GOES 16 X-ray flux is included in Appendix 5.

PYTHON LIBRARY AND DEMOS

https://github.com/abelectronicsuk/ABElectronics_Python_Libraries

See web site for how to download and install libraries.

Going through AB ELECTRONICS I2C INFO

$ sudo i2cdetect -y -1

Shows attached i2c devices.

I can see 0x68 ✓ HAPPY

0x69 ✓