REMOTE AUDIO MONITORING SYSTEM FOR FM TRANSMITTERS WITH ADVERTISEMENT TRACKING FUNCTIONALITY

Daniel Phutinyane[1] (corresponding author)
Monageng Kgwadi[2]

[1] European Business University of Luxembourg, Chaˆteau de Wiltz, L-9516 Wiltz, G.D. of Luxembourg.

[2]University of Botswana, Private Bag UB0061, Gaborone

April 24, 2024

Abstract

Commercial radio stations generate their revenue through sales of advertisement slots and sponsorships. Normally, a log registry is generated from the studio's playback system through which the clients can confirm whether their advert played or not. However, since the terrestrial radio transmission network involves deploying multiple radio transmitters around the country to reach a wider audience, it becomes too expensive for commercial broadcasters to install and maintain those transmitters. On the other hand, if they don't reach a wider audience, they struggle to convince the customers to buy advertising space, creating a vicious cycle which needs low-cost solutions to reduce the overhead costs. Some of those costs include remote monitoring systems of the transmitters which can be achieved by using software defined radio (SDR). With SDR, most of the signal processing traditionally carried out on dedicated hardware is carried out on software. In addition, the behaviour of an SDR-based remote monitoring system can be further customised and/or configured to the needs of each particular broadcaster. This project demonstrates a low-cost remote audio monitoring system for FM transmitters which can be used to remotely monitor on air transmitters using a Nuand BladeRF 2.0 micro xA4 software defined radio. By using speech recognition, further processing was done to eventually keep count, record and update on Power BI and Web how many times each advert was played. This solution offers a low-cost solution by eliminating the requirement of multiple sensors in the transmitter sites for remote monitoring purposes.

KEYWORDS: Software Defined Radio, FM transmitters, GNU Radio, BladeRF

1 Introduction

The success of radio broadcasting to this date is attributed to the revenue generated through advertisements [1]. The clients, which may range from government to private companies and individuals normally have target and diverse markets to which they intend to advertise their products. As a result, radio stations tend to cater for these diverse markets by covering a wider geographical area, by deploying broadcast transmitters around the licensed area of operation.

Over the years, several means of broadcasting radio signal from radio station studio to the households has emerged from the traditional analogue platforms, being medium wave (MW) and frequency modulation (FM) [2] to digital platforms such as digital audio broadcast plus (DAB+) and lately, livestream [3], mobile app and smart speaker [4]. These transitions, whilst expected to compete, instead complemented each other so much that in most cases some radio stations deploy both methods. For example, in Gaborone, Botswana, Radio Botswana 1 (RB1) broadcast on 972 kHz (MW), 89.9 MHz (FM) and also via livestream, whilst in the United Kingdom (UK), Heart FM radio network broadcast across UK on terrestrial, DAB and also a livestream worldwide on www.heart.co.uk/digital.

The same applies to, say Radio Hamburg in Germany which broadcast on FM frequency of 103.6 MHz in Hamburg, Radio Hamburg app, smart speaker as well as online at www.radiohamburg.de. These different platforms provides listeners with different options through which they can listen to those radio stations, hence extending reach. In fact, in Botswana, if not other countries as well, the commercial radio stations charge for advertising slots using FM terrestrial rate card whilst livestream and mobile app are used as value add ons. This is so because the most common method of consuming radio content is still via FM terrestrial radio [5] and will still remain so in the next decades [6]. However, the same platform is susceptible to a likelihood of high downtime due to various fault sources related to the analogue electronics components used in the transmitters. Examples of those faults could be high reflected power or voltage standing wave ratio [7], rain fade effect [8], high temperatures due to a failed fan or room air conditioning system etc. As a result, to avoid prolonged downtime, it is necessary that the transmitters are always monitored. This study seeks to device a low-cost solution that can be used to remotely monitor the on and off status of FM radio transmitters as well as track advert statistics.

2 Background

2.1 FM terrestrial broadcasting

Whilst FM terrestrial broadcasting has survived the threats of the new emerging radio broadcasting platforms, its main challenge has been the associated costs related to its propagation characteristics of attenuation over distance [9], [10]. As a result, the FM terrestrial broadcasting network requires multiple broadcast transmitters to be installed at some designated transmitter sites to cover wide areas. These transmitters independently receive a common signal from the studio, which would have been uplinked via satellite, optical fibre or studio transmitter link. A typical 500W transmitter would cover a radius of 40km on average, and at the boundary of that 40km radius, another FM transmitter of the same radio station, broadcasting on a different frequency (to avoid interference) would pick up etc. Figure 1 below shows signal coverage of RB1 in the greater Gaborone area.

Figure 1:RB1 coverage map in the greater Gaborone area. Transmitters are of different power ratings hence coverage differ. For example, Gaborone is a 2KW transmitter, whilst Lobatse is 500W transmitter.

With this arrangement, the advertisers are promised that with a wider coverage (footprint) that a radio station has, they can be assured of a larger listenership which by extension is converted into customers. However, the current system, that is deployed in almost all the countries, is such that the playback

system installed in the studio, generates a registry log which is sent back to the advertisers/clients as a confirmation of a played advert. It should be noted that an advert may be played in studio whilst the transmitter is off. As a result, the listeners in that area would not have received that advert and therefore the advertiser may end up paying for a service he/she is not getting. In addition, in the absence of a remote monitoring system, the broadcasters may get to know about their off-air transmitters after several days. This implies that the listeners also missed critical information during that time. To overcome the prolonged downtime, radio stations normally install the remote telemetry monitoring system at the transmitter sites to monitor the on and off status of the transmitters, measure parameters such as forward power, reflected power, signal-to-noise ratio, etc.

2.2 Telemetry system

Telemetry is defined as "the science or process of collecting information about objects that are far away and sending the information somewhere electronically" [11]. In broadcast transmitters, such information is crucial to inform engineers and technicians about the status of the transmitters, which are most of the time in remote transmitter sites. In Botswana, the Department of Broadcasting Services (DBS) which houses RB1, RB2 and BTV uses a telemetry system from ANT GROUP (SPA) [12]. The system uses sensors to gather information about the transmitter system such as ON/OFF status of transmitter, room temperature, forward and reflected power etc. and then send them to the control room via GSM. The main challenge with this system is that it is too expensive to use as it charges every time it sends an update to control centre. In addition, when the network is too busy, it fails to make a connection. However, with most of the transmitter sites recently been connected to optical fibre network, the costs associated with using a GSM network can be avoided by exploring a telemetry solution that uses Internet connectivity instead. This can be achieved by using a software defined radio to receive audio signal at the transmitter site in the same manner that an FM radio receiver in the households receive terrestrial FM radio feed.

2.3 Software Defined Radio

Software Defined Radio (SDR) refers to reprogrammable or reconfigurable radios [13] as first used by Joe Mitola in 1991 where some radio functionalities previously performed on hardware [14] are instead performed by software. The flexibility brought about by SDR has presented new opportunities which can be exploited to address limitations of old techniques to handle different types of radio signals. "One moment your computer could be an AM radio, the next a wireless data transceiver—and then perhaps a TV set" [15].

Due to its flexibility, SDR can also be used for telemetry remote monitoring of FM transmitters and since it is pre-dominantly on software, the system has low chances of failure compared to the traditional system which is hardware based [14]. The same cannot be said about the traditional remote monitoring systems which deploy sensors (hardware) as those sensors normally malfunction because they have reached end of life or because of electric surges. Therefore, by deploying SDR based solution, the broadcasters also reduce the chances of hardware (sensor) failure. Moreover, since it is a relatively cheaper solution, the broadcasters can reduce their monitoring costs and expand their coverage areas. In addition, they can use the same solution to demonstrate to the advertisers that indeed their adverts were played.

To effectively process a radio frequency (RF) signal SDR first convert it from analogue into a digital signal using an RF front end. This RF front end consists of analogue-to-digital converter (ADC) which receives the analogue signal from the antenna and convert it into a digital format, thus fitting for processing with software [16]. As an FM radio receiver, an SDR uses its antenna to receive the electromagnetic (EM) signal and present it to the RF front-end for down conversion to the intermediary frequency (IF). The conversion to lower frequency allows for the available analogue to digital converters currently in the market to handle the speed of the incoming signal [17]. The analogue

to digital converter (ADC) then converts the signal into a digital signal which is a format required by the Field Programmable Gate Arrays (FPGA) [16]. The FPGA will then decimate and down convert the sample rate of the signal to standard requirement of the communication protocol (e.g. USB 3.0). The output is presented to the software for software processing, in this case demodulation. It is after demodulation that the audio file will be captured and stored in a computer/raspberry pi, followed by transcribing, search and count keywords (say advertiser's tagline or keyword) then send them as text to the monitoring office where an invoice of the confirmed adverts will be generated. A copy can be sent to the advertiser as well.

3 Methodology

There are numerous SDR platforms available on the market with various capabilities. Table 1 summarizes characteristics of some SDR hardware platforms along with key performance figures that have been identified as candidates for the study. Considering the overall cost paired alongside some other features gave Blade RF Micro 2.0 xA4 an advantage over the rest. HackRF One provided the cheapest option of all followed by BladeRF xA4. However, BladeRF 2.0 micro xA4 offered the most maximum bandwidth (20 MHz vs 56 MHz), which allows for higher sampling rates and hence better-quality audio. For that reason, BladeRF 2.0 micro xA4 was the preferred option for this study.

Table 1: Features of some of the common SDR hardwares [15], [16], [18], [19]

Properties	Universal Software Ra- dio Peripheral (USRP X 310)	HackR ONE F	BladeRF 2.0 micro xA4	New Horizon s (NH702 0)
Tuning Range	70MHz-6GHz	30MHz-6000MHz	70MHz-6GHz	70MHz-6GHz
Max RF band-widths	upto 160Mhz per daughter-board	20MHz	56 MHz per channel	56MHz per channel
Cost(US$)	3485	399.99	480	862
Power supply	12V, 7.5 A	5V, 0.7A (USB pow-ered)	5V, 0.7A (USB pow-ered)	5V, 0.5-2A DC

Various SDR software development platforms are available on the market such as Matlab/Simulink, Labview and GNU Radio. However, Matlab/Simulink and LabView are proprietary and come at a cost. GNU Radio was the preferred choice due to its convenience as an open-source software which offers a low-cost solution. The GNU Radio Companion (GRC) is a multi-platform free software development framework that provides signal processing functions for implementing software-defined radios [20] [18]. The GRC was used in this study to configure the SDR platform.

3.1 BladeRF 2.0 Micro xA4

The BladeRF 2.0 micro xA4 is an SDR hardware designed by Nuand characterized by 2x2 Multiple Input Multiple Output (MIMO), 70 MHz to 6GHz frequency range and a USB 3.0 communication module. The MIMO feature is driven by the two (2) separate local oscillators found in the two

channels: one oscillator for each transmitter-receiver channel. As a result, this device can broadcast two channels simultaneously through transmitter 1 and transmitter 2 as well as receiving through receiver 1 and receiver 2. The BladeRF can be configured to act as two transmitters, two receivers or MIMO system. For the purposes of this project/study, only one channel was used to receive one radio station, although, beyond the scope of this study, it can also be shown that multiple radio stations can be received using one SDR. By connecting an antenna to RX1 port of the BladeRF SDR and loading the appropriate GNU radio code it can receive a preset channel through RX1 1 port and have RX 2 port as a standby. Figure 2 shows the BladeRF 2.0 micro xA4 and its accessories.

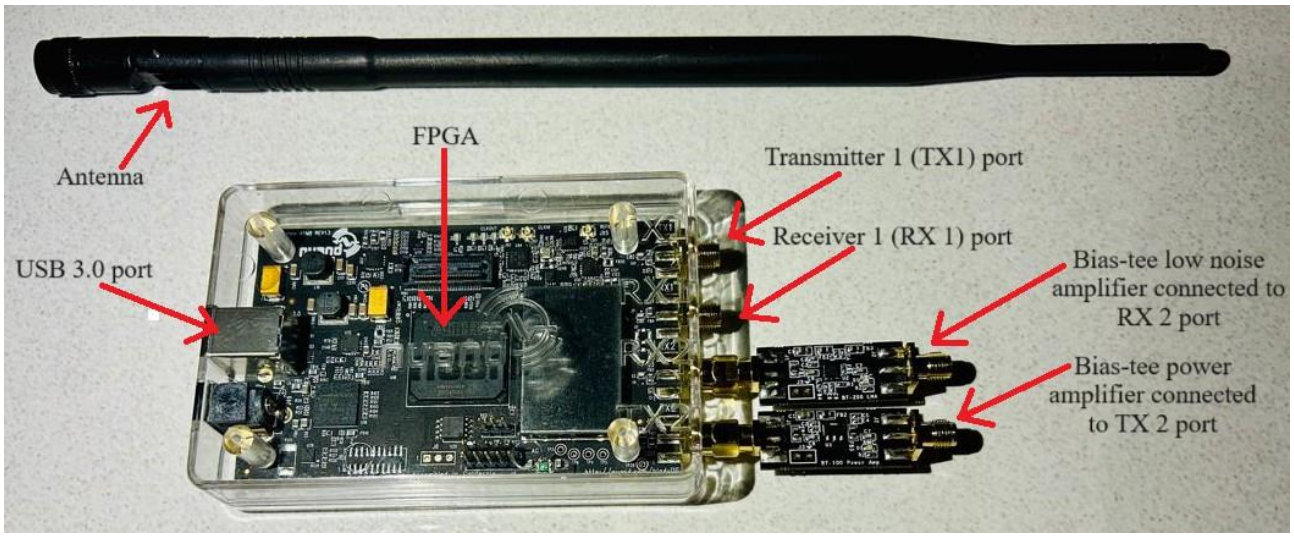

Figure 2: BladeRF hardware with its accessories [21].

The project utilized the Nuand Blade RF Micro 2.0 xA4 SDR platform, which incorporates the Cyclone V FPGA from Intel. This FPGA boasts features such as "single-cycle access embedded memory," along with dedicated DSP capabilities including hard 18×18 multipliers, and a flexible array of general logic elements for programming [22]. The SDR system also boasts an ADC/DAC resolution of 12 bits. A bias-tee power amplifier is employed to amplify the output signal before it reaches the antenna, connecting to the transmitter port of the system. This power amplifier consumes 200 mW of electrical power and delivers a maximum gain of +15 dB of radio frequency (RF) power at 200 MHz [21]. Conversely, the bias-tee low noise amplifier, rated at 300 mW, connects to the receiver port. With a typical gain of +20.2 dB at 200 MHz, its purpose is to effectively lower the noise figure and significantly enhance the dynamic range of various applications. It is linked to the antenna on its other end. The antenna utilized is a general-purpose monopole antenna with a gain of 5 dBi [23] and a voltage standing wave ratio (VSWR) of less than 1.5.

3.2 GNU Radio

GNU Radio is a freely available framework enabling users to create, simulate, and implement sophisticated real-world radio systems [24]. Its library consists of signal processing blocks coded in C++ (backend), with Python utilized for front end organization and connection of these blocks [25]. Figure 3 below shows a software organization of flow-graphs in GNU Radio.

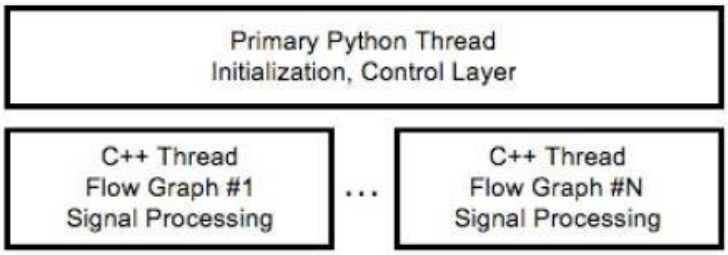

Figure 3: Software organization of flowgraphs in GNU Radio [26].

Instead of solely relying on Python code to create Python files at the frontend, users have the option to utilize a graphical tool known as GNU Radio Companion (GRC) for graphical file creation. This tool comprises five (5) main components:

- Workspace: This area holds blocks imported from the library.
- Toolbar: A graphical interface containing icons, menus, or other input/output elements.
- Library: Where all pre-existing user blocks are stored for workspace use.
- Variables: A window housing variable blocks utilized in the design, each mapping a value to a unique variable.
- Terminal: This window displays results following flowgraph compilation.

Figure 4 shows the user interface of the GNU radio companion.

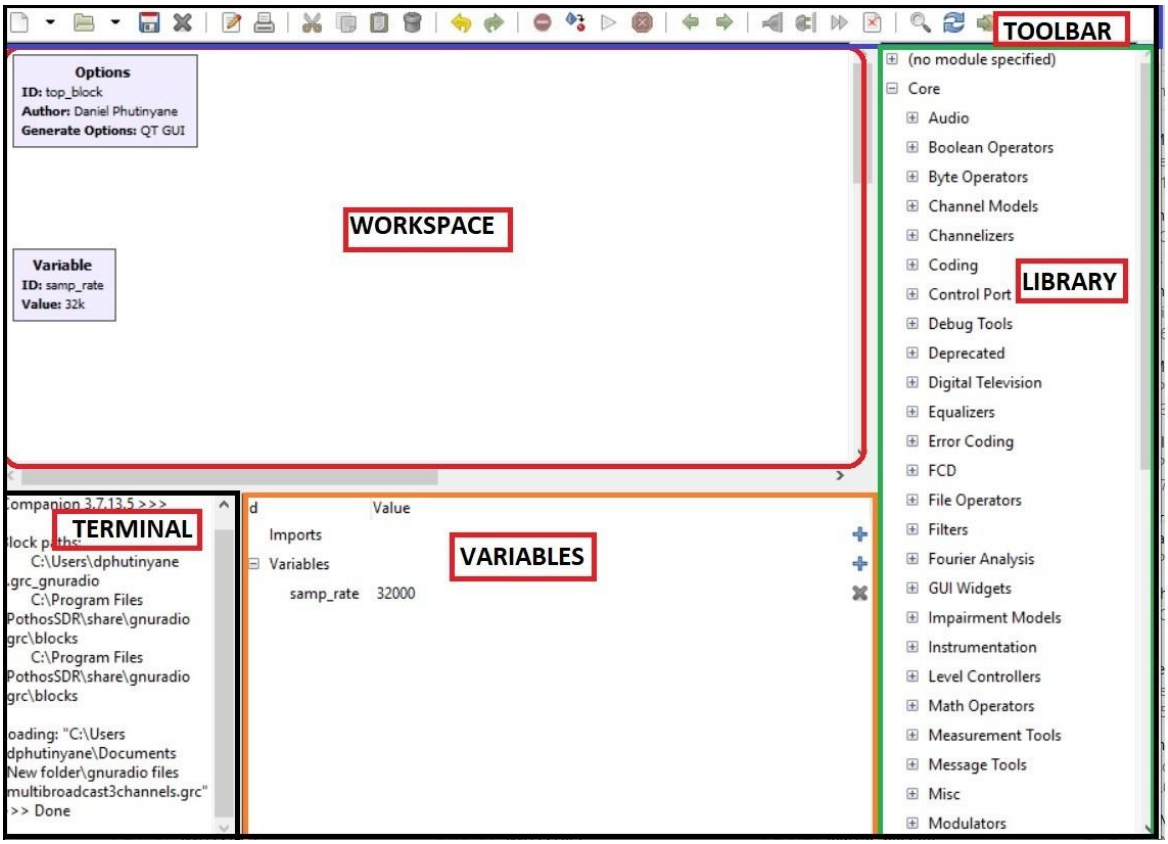

Figure 4: GRC interface showing all five parts making up a typical GNU Radio Companion.

GNU Radio is compatible with a wide range of general-purpose processors and operating systems, including Linux, macOS, and Windows. Typically, installation is facilitated through existing binary packages, as mentioned in [27], However, alternative installation methods include:

- From source (for those who want full control).
- Using PyBOMBS (for those who want it built from source and/or installed to a specific directory using a script).
- Using conda (for those who want binaries installed to managed environments).

3.3 Software Configuration

The GNU radio companion (GRC) was used to program the hardware using the inbuilt osmocom block. The USB 3.0 module provided a communication platform between host laptop and BladeRF. The GRC was run on an HP Zbook firefly 15 G7 laptop with Intel i7 CPU, 32 GB RAM and Windows 11 Pro operating system to configure the SDR hardware and visualize the signals for debugging purposes. Figure 5 shows the envisaged FM receiver system level experimental set up.

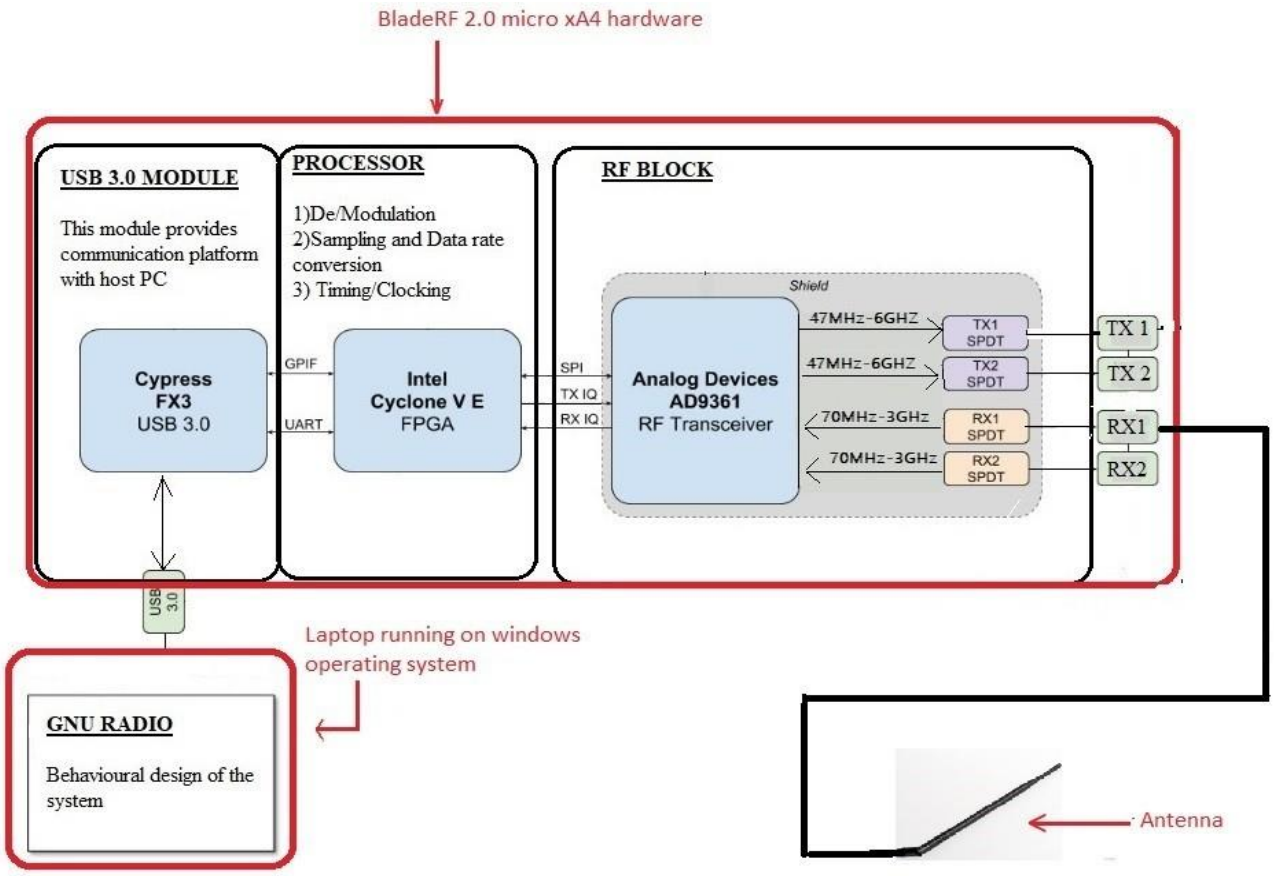

Figure 5: Hardware setup using BladeRF 2.0 micro xA4 and GNU radio [28].

3.4 Python and Jupyter Notebook

There are many different Integrated Development Environments (IDEs) out there, as much as there are many different programming languages, each with their own strength and weaknesses. For data science, Python is the most preferred language mostly for its libraries, such as Pandas, NumPy,

TensorFlow and Matplotlib etc. that make handling, processing, and presenting data much easier [29].

As for IDEs, whilst not a traditional IDE, Jupyter notebook provides an interactive computing environment for writing and running code [30], allowing programmers to analyse workflows, extract data and visualise it during the research process [31]. It has multiple kernels of which, each kernel handles a different language, such as IPython kernel which runs Python code, IRkernel for R language, Ijava for Java etc. [32]. These kernels act as the back end whilst the front-end interface consists of cells which provide a space for writing code [33]. Figure 6 below shows how Jupyter notebook interface looks like.

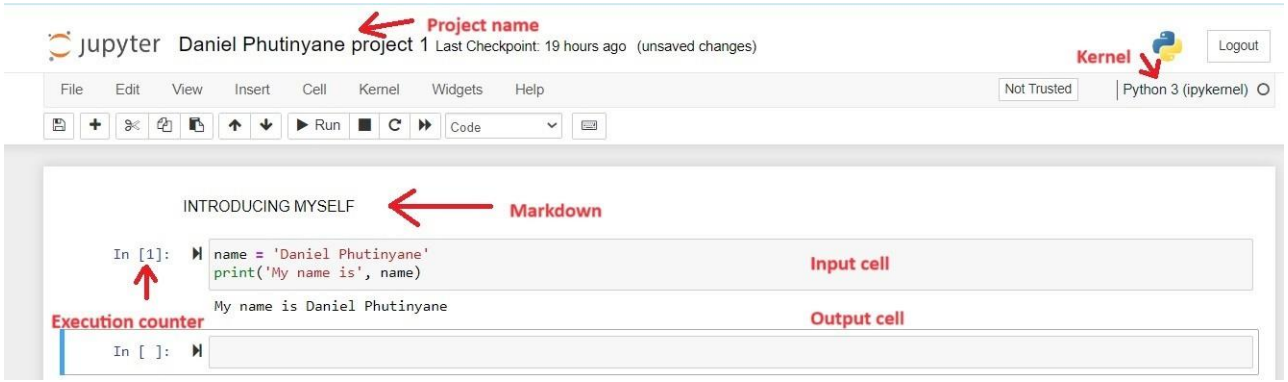

Figure 6:An illustration of a Jupyter notebook interface with a sample code to show input and output cell.

Compared to other IDEs, Jupyter notebook stands out as the most preferred as Figure 7 below shows. Some of the reasons why this is so because of its strength in interactive data exploration, visualization capabilities, data cleaning and transformation as well as integration with big data tools [34].

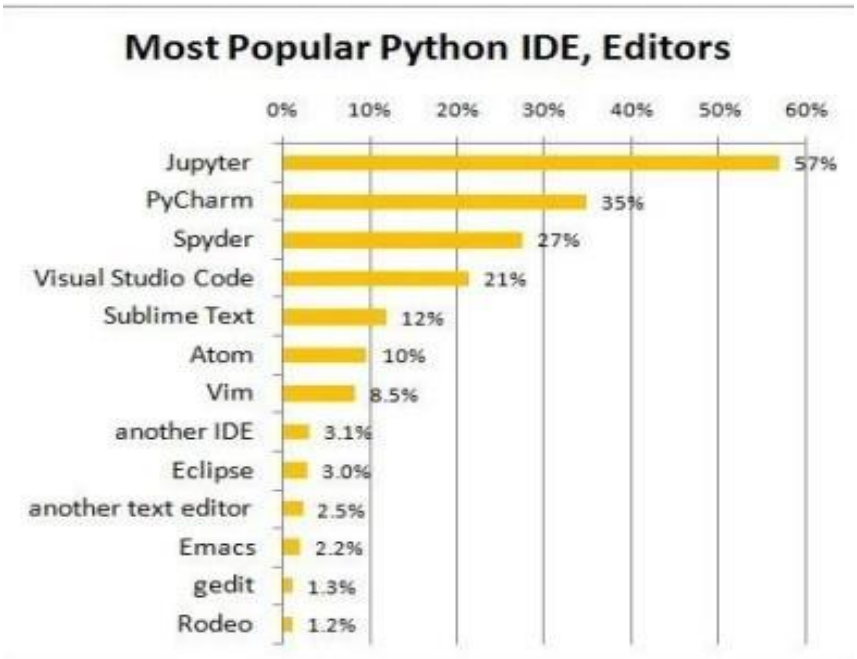

Figure 7:Popular IDEs [29].

3.5 Power BI Desktop

Whilst Jupyter notebook have visualization capabilities, there are other tools specifically designed for visualization of data. Some of those are Microsoft Power BI desktop and Tableau desktop and they are almost equal in strength. The most notable difference however, between the two, is that Microsoft Power BI desktop is free to use whilst Tableau desktop requires a commercial license [35]. For that reason, Power BI desktop was used in this study.

4 Experimental Results and Analysis

Data collection was done by running the GRC flowchart in Figure 8 below which shows a block diagram representing the software implementation of an FM receiver configuration on the GRC software.

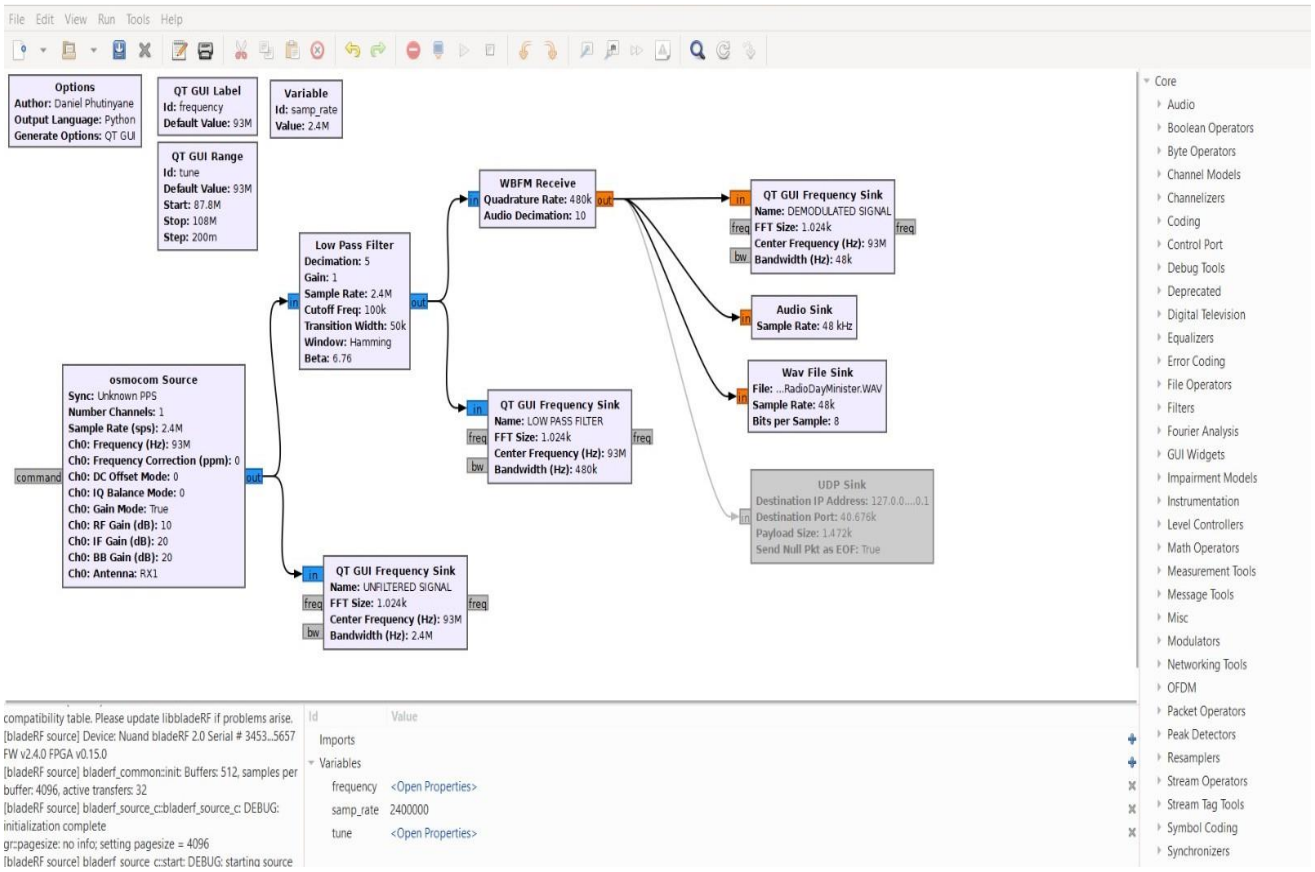

Figure 8:A GNU Radio Companion (GRC) setup of FM receiver showing how GRC blocks are connected to receive FM signal of 93.0 MHz.

Each of these blocks served a distinct role towards realization of this FM radio receiver as outlined below.

- OSMOCOM SOURCE: This block takes in an FM signal received via the antenna and the RX 1 port. It then samples the signal at a rate of 2,400 MHz. Variables such as gain are set as and when needed, to receive a clearer signal. The CH0 antenna takes the argument that corresponds to the physical port where the antenna has been connected.
- LOW PASS FILTER: By decimating the incoming signal (sampled at 2,400 MHz) from osmocom source block by a factor of 5, this block filtered the unwanted signal and passed the output which was set to 480 KHz for demodulation at the next stage.
- WBFM RECEIVE: A wide-band frequency demodulating block whose function is to demodulate a frequency modulated signal into audio signal. It took an RF signal from low pass filter block and demodulated it into audio signal, by decimating the RF signal with a factor of 10. The factor was calculated by considering the quadrature rate of 480 KHz as well as the sample rate of 48

KHz in the audio sink.

- AUDIO SINK: To confirm that the audio was set to the rate that can be handled by the sound card, this block was used. It allows the listener to listen to the FM receiver radio through the speaker system of the laptop. Sample rate here was set to match the sample rate of the WBFM receive output.
- WAV FILE SINK: This block saves the output of the FM receiver to the laptop which acts as the server in this project. The file path location was set through the "file" parameter, and the .wav extension was appended to ensure that the saved file can easily be accessed as a wav file. By copying the file path this file was loaded into the Jupyter notebook for further processing using the audio path variable. The advantage of using this block is that the saved file also acts as a backup should there be any need to access it again in the future.
- UDP SINK: As an alternative to WAV FILE SINK block, the final output can be sent directly to Jupyter notebook by using this block. It takes the destination IP and port as arguments and by using the UDP socket in Jupyter notebook, the output of this block can be availed for further processing. The setback here is that there is no back up which may be needed in the future. For this project, this block was disabled as WAV file sink was preferred.
- QT QUI FREQUENCY SINK: Finally, the graphical representation of the output of the SDR was plotted using FFT sink blocks. When compiling the GRC code in Figure 8 this block visualizes the output of the demodulated and the filtered signal as shown in Figure 9.

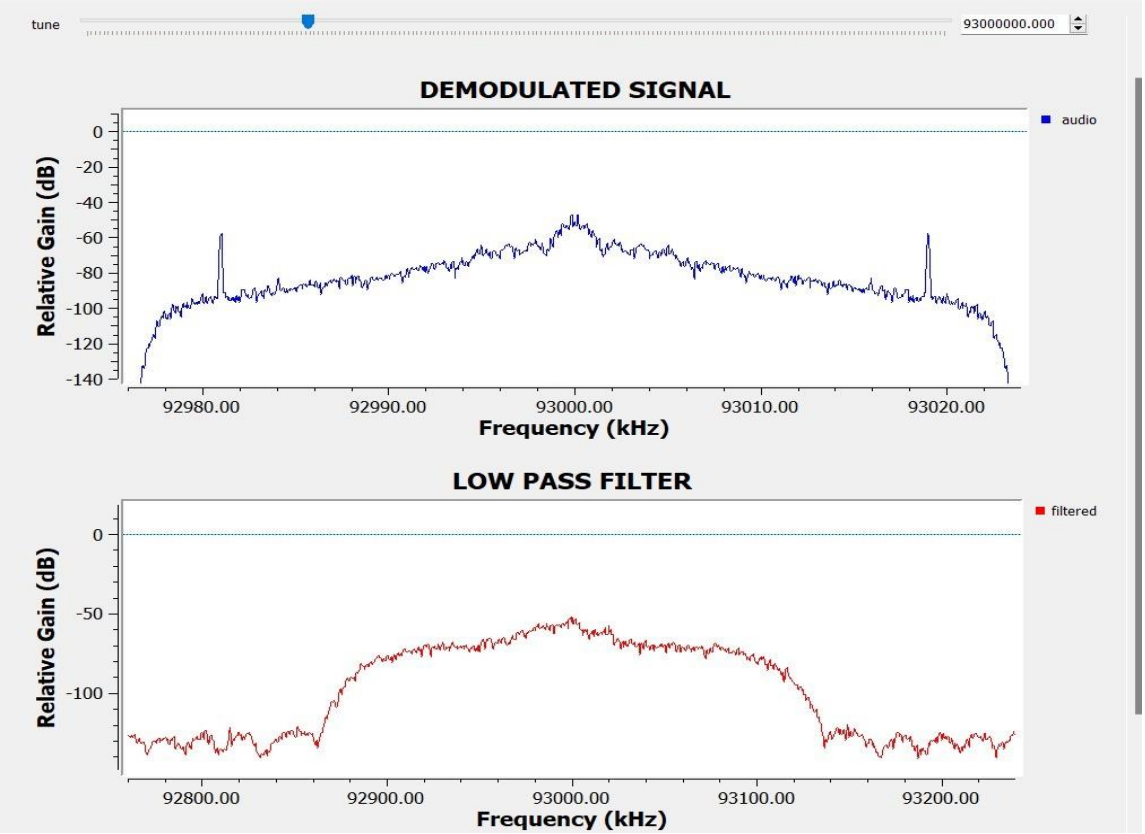

Figure 9:The QT GUI Frequency sink output showing the demodulated and the filtered signal.

The file recorded using WAV FILE SINK in GRC of Figure 8 was then loaded into Jupyter notebook and then passed through speech recognition. Figure 10 below is an extract of the python code used. The keyword counts is a variable that takes the arguments of string type. In this instance, the arguments "radio", "paris" etc. were initialised to zero and by passing them through the speech recogniser, they were counted every time they are encountered.

Thereafter, the audio was transcribed, keeping count of the keywords and sending them for visualization on

```
[3]: # Initialize the speech recognition recognizer
     recognizer = sr.Recognizer()

[4]: # Initialize keyword counts
     keyword_counts = {"radio": 0, "paris": 0,"olympic": 0, "botswana": 0, "ebi": 0}
```

Figure 10:File path for a recorded audio file.

Power BI desktop and web browsing through URL. Figure 11 below shows the transcribed text from the audio using speech recognition.

```
    except sr.UnknownValueError:
        print("Speech Recognition could not understand audio")
        if energy < energy_threshold:
            print("It's likely music.")
    except sr.RequestError as e:
        print(f"Could not request results from Google Speech Recognition␣
↪service; {e}")
```

```
English Transcription: present senior government officials here present the
business community members of the media invited guests A very warm welcome and
good morning to you all I'm here to deliver to you today a message from our
director general excellency means Audrey Azalea on the occasion of world radio
day 13th februari 2024 on this world radio day we celebrate not only the history
of radio but it's central role in our societies now and in the years to come the
Year 2024 mocks and milestones in the history of the media it is the year we
celebrate the centenary of the first live radio broadcast of the Olympic games
on the eve of the next games that will be held in Paris this must only reminds
us that since it's creation at the end of the 19th century Radio has been with
us bringing us together around powerful moments and shared emotions and so for
over a century it has been informing us entertaining us and also educating us as
this year's theme underlines it is all the more true today despite the growing
influence of the internet and social networks radio continues to be a prime
source of information and entertainment it is estimated that over 4 billion
people listen to It radio is the medium that reaches places others do not while
almost the third of the population did not have a decent internet connection in
2023 abortion that Rises to half of the population in rural areas radio is more
inclusive and accessible particularly in Crisis situations for example in
Afghanistan during the decision which UNESCO immediately and firmly Condemned to
deprive Afghan girls and women of their fundamental rights to learn and teach
the organisation has put in place what is an affect education over the airwaves
supporting radio Begum in particular this radio station run by Afghan women for
Afghan women provides literacy courses and gives them boys radio can also be the
voice of the voiceless Enabling all individuals and communities to express
themselves and to bring the diversity of their culture to life that is why you
no supports and encourages Community Radio All Over the World as We See It radio
is more than a technical means of broadcasting it embodies a certain idea of
information cultural diversity and the education for all we could go as far as
to say that radio can and must be a humanist medium T13 februarie once again
acknowledge the road travelled by radio and the power of its Airways to build to
broadcast the possibility of a better world that is the message from our
director from the director general of UNESCO but I think in the true Spirit of
inclusivity and making sure that we do not leave anybody behind directors of
ceremony allow me please to give a brief summary of the message in setswana 1320
2400 years so Romano has since 35 billion the fatum 1000 Rebellion
Setswana Transcription: setaelo directors here president senior government
official member and good morning to you today on the case of world radio
february 2020 ruthe we celebrate the sentenari of the first games on the eve of
the stone reminds that since radio us and us this guest information le motshega
```

Figure 11:Transcription of audio into text using speech recognition.

Since the idea was to deploy this system in all transmitter sites where monitoring is needed, it was also of importance to consider a mechanism that could automatically pick the location of where the audio

recording is done, and this was done by using the laptop as a server and then reading its IP address. In real life deployment, a laptop/server could be replaced with a raspberry pi. Figure 12 below shows how this was done.

```
[8]: # Get the local IP address of the server using gethostbyname_ex
try:
    server_ip = socket.gethostbyname_ex(socket.gethostname())[2][0]
except socket.error as e:
    print(f"Error getting server IP address: {e}")
    server_ip = "Unknown"
```

Figure 12:Obtaining server ip address of the server/laptop at the transmitter site using socket.gethostname.

After obtaining the IP address, together with the counted keywords, the output was displayed on Power BI using a pie chart, heat map, bar chart and donut chart. Figure 13 below shows the output as viewed on Power BI desktop. From the figure, it can be seen that three keywords Radio, Olympic and Paris were mentioned during the seven (7) minutes when the speech was made. These keywords were used as the advert names which would have been pre-configured to be monitored. In addition, by obtaining the IP address, it has been shown that different transmitter sites can be identified by their respective IP address to monitor which transmitters are on air. In this case, the transmitter accessed was located at 192.168.56.1 transmitter site.

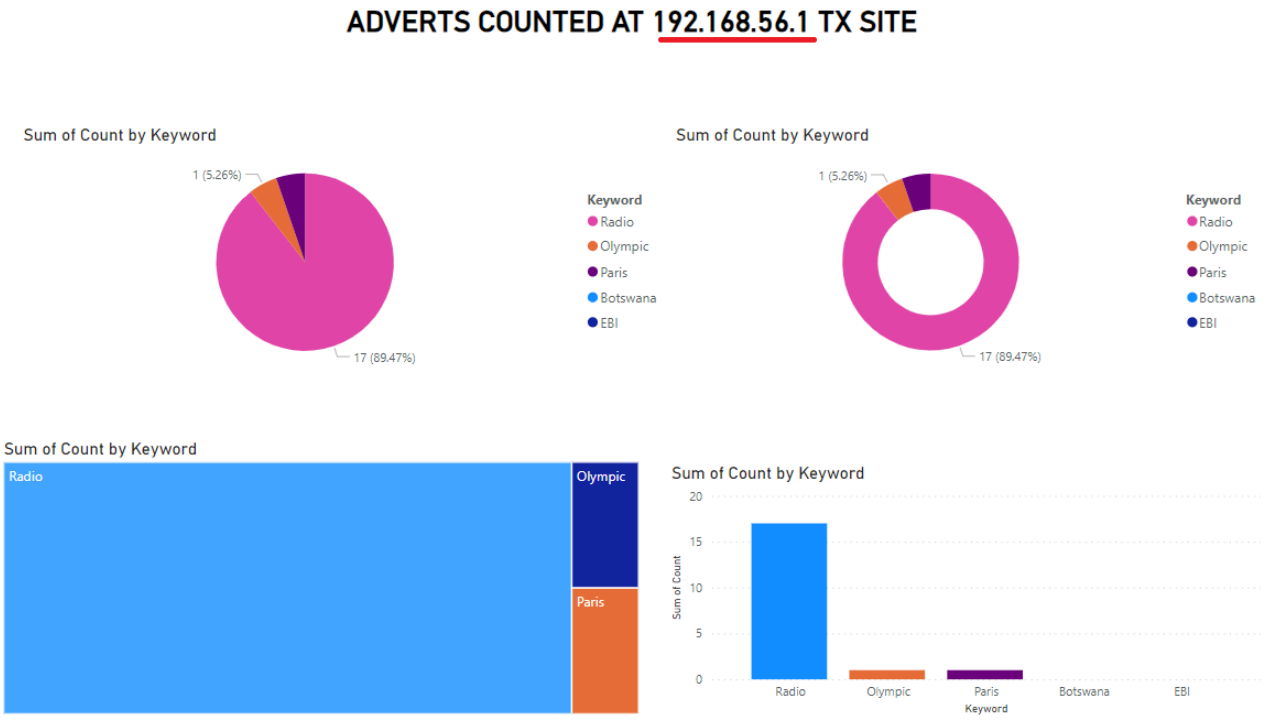

Figure 13:A visual display of the counted keywords using Microsoft Power BI desktop.

There were three (3) ways used in displaying the output data. Data presented in Figure 13 above was actually obtained from a generated CSV file by using the CSV module's csv.writer class. The other two were obtained using URL in web browser and the web functionality in Power BI. Figure 14 below shows the two put side-by-side for easy illustration from which it can be seen that the data was obtained from IP address 192.168.56.1 as shown in Figure 13. In addition, the displayed text shows the total number of occurrences for the respective words, as well as the date and time on which the count was done.

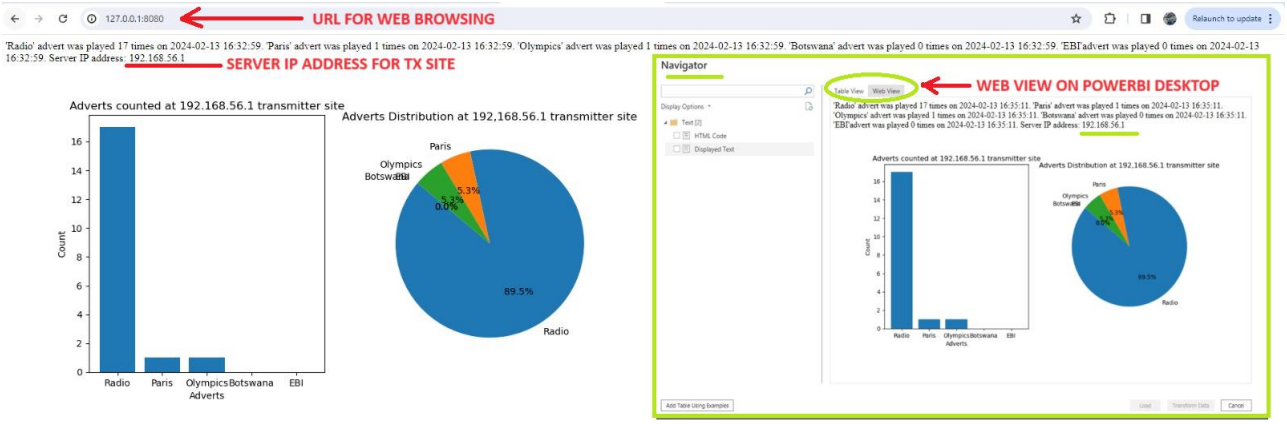

Figure 14: A visual display of the counted keywords using Microsoft Power BI desktop. The charts on the left shows data displayed on the web browser using http://127.0.0.1:8080. Similarly, the charts on the right were obtained using Power BI desktop web view functionality.

For monitoring purposes of where users have logged in, it can be observed in the local host where other users are logged from. Figure 15 shows that there is a user logging on using local host IP 127.0.0.1 and three other users logging on one IP address being 192.168.70.58. Those three users are the one previously mentioned in Figure 13 and Figure 14 above.

```
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080, threaded=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:8080
 * Running on http://192.168.70.58:8080
Press CTRL+C to quit
127.0.0.1 - - [13/Feb/2024 16:32:59] "GET / HTTP/1.1" 200 -
192.168.70.58 - - [13/Feb/2024 16:35:11] "GET / HTTP/1.1" 200 -
192.168.70.58 - - [13/Feb/2024 16:35:11] "GET /favicon.ico HTTP/1.1" 404 -
192.168.70.58 - - [13/Feb/2024 16:35:11] "GET / HTTP/1.1" 200 -
```

```
[ ]:
```

Figure 15: Jupyter notebook output showing users currently accessing the output data.

The previous results shown were observed using the same laptop which was running the code. This did not confirm the remote monitoring capability. To confirm whether it can be viewed remotely, an iPad was used, and the output was consistent with the expectations as shown in Figure 16.

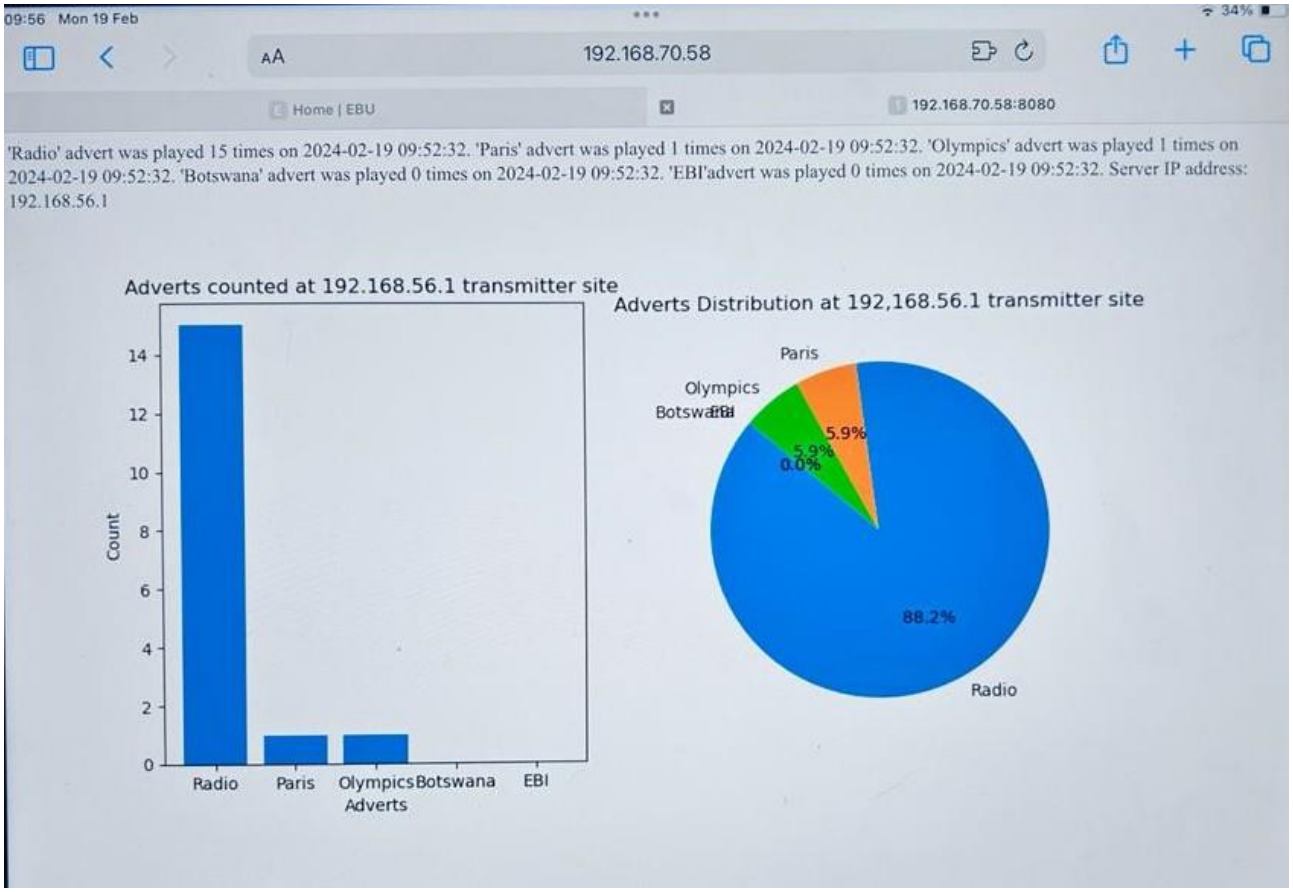

Figure 16: Accessing counted adverts using an iPad.

Normally, commercial radio stations play their adverts every twenty (20) minutes. Bearing that in mind, the code was written such that a background task would run every 20 minutes to fetch new data. Figure 17 shows how this was achieved.

```
[9]: # Schedule the background task every 20 minutes
     schedule.every(20).minutes.do(background_task)

[9]: Every 20 minutes do background_task() (last run: [never], next run: 2024-02-13
     16:52:44)
```

Figure 17: Background task scheduled every 20 minutes using schedule () module.

4.1 Experimental Setup

The Blade RF Micro 2.0 SDR platform was programmed using GNU Radio Companion (GRC) to receive the FM Radio signal on 93.0 MHz. Communication between the SDR and the laptop was via the USB 3.0 cable which sent the code to BladeRF SDR for execution of the GRC commands and returned the output as audio to the laptop. The audio was then saved as a WAV file in the local disk of the laptop, where it was then accessed using Jupyter notebook for further processing. Figure 18 shows BladeRF 2.0 micro xA4 SDR connected to a tri-band antenna for signal reception, whilst the other end is connected to the laptop using a USB 3.0 cable. The iPad was used for remote monitoring purposes as shown in Figure 16.

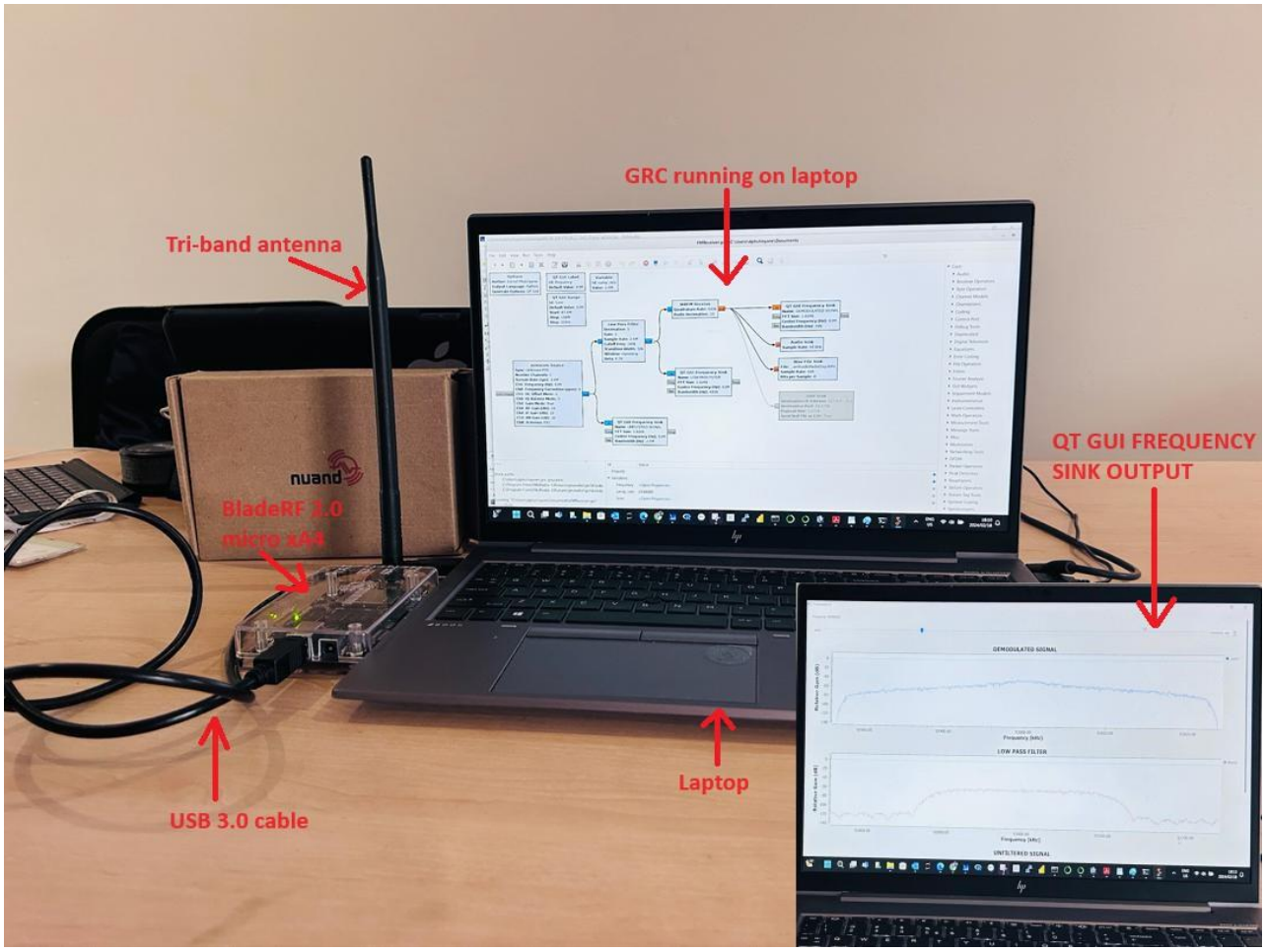

Figure 18: A setup showing how FM Radio signal was received using BladeRF 2.0 micro xA4 and GRC running on laptop.

5 Conclusions and Future work

5.1 Conclusions

In this study it has been shown that using a GNU Radio code running on BladeRF 2.0 micro xA4, an FM radio receiver can be implemented to collect data. In this case, the source of data was an FM radio. Since transmitters located in adjacent transmitter sites have different frequencies, the receiver was set to the respective FM frequency (93.0 MHz) which was expected to be transmitting in that area. The received data was then sent via internet to a remote monitoring device, being an iPad and displayed through URL and Microsoft Power BI desktop. By virtue of receiving data from that channel, it automatically implies that the transmitter is on, and therefore meeting the objective of this project.

Whilst this data received could have been any other data, the preferred option was to pay attention to the adverts which will also act as a confirmation to the customer that indeed their advert was received by the listeners across the country. This was achieved by using advert names as the keywords, and in this case, there were five (5) keywords used, Radio, Paris, Olympics, Botswana and EBI. Since in practical application these keywords would be used to track the advert, then a phrase will be used instead of just one word. That approach will ensure that the keyword is only counted when the entire phrase has been detected, so as to avoid a confusion that will arise as a result of mentioning that word in a different context. As per results shown in Figure 14, Radio advert played 17 times, Paris and Olympics played 1 time, whilst Botswana and EBI adverts never played. The dates and times also show, as well as the IP address which corresponds to the physical transmitter site.

5.2 Future work

In the next phase, there will be a database of IP addresses which corresponds to the transmitter sites such that the code will return the name of the transmitter site instead of IP address. Furthermore, the system will run parallel by obtaining data directly from the GRC flowchart into Jupyter notebook using UDP SINK, and then using WAV FILE SINK for backup purposes. Lastly, a raspberry pi will be used instead of laptop. This way the cost of the system will be significantly lower compared to when using the laptop.

6 acknowledgements

This research would have not been possible without the skills I got from the course instructors at European Business University of Luxembourg. Special thanks also go to the university for giving me this valuable scholarship without which I wouldn't have managed to enrol in this course.

Declarations

- Funding: This research would have not been possible without the scholarship I got from the European Business University of Luxembourg.
- Conflict of interest/Competing interests: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.
- Ethics approval: N/A
- Consent to participate: N/A
- Consent for publication: I, the undersigned, give my consent for the publication of identifiable details, which can include photograph(s) and/or videos and/or case history and/or details within the text ("Material") to be published in the above Journal and Article.
- Availability of data and materials: N/A
- Code availability: Available on request.
- Authors' contributions: D. Phutinyane: Conceptualization, Funding acquisition, Software, Investigation, Testing and Validation, writing (original draft). M. Kgwadi: Supervision, writing (review and editing).

References

[1] M. C. Keith, *The radio station: Broadcast, satellite and internet*. Routledge, 2012.

[2] G. L. Frost, *Early FM radio: Incremental technology in twentieth-century America*. JHU Press, 2010.

[3] P. Gilski and J. Stefan´ski, "Can the digital surpass the analog: Dab+ possibilities, limitations and user expectations",
International Journal of Electronics and Telecommunications, vol. 62, no. 4, 2016.

[4] P. LORIN and L. THORSAGER, "Designing navigation in smart speakers: Guidelines for designing a voice user interface for navigating radio content using a smart speaker", 2019.

[5] T. Hussain, T. Mujahid, M. Abdullah, S. Saeed, and S. A. Alharbi, "Sddrm: Software defined digital radio mondiale", in *2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, IEEE, 2019, pp. 1–5.

[6] P. Gilski, "Dab vs dab+ radio broadcasting: A subjective comparative study", *Archives of Acoustics*, 2017.

[7] W. Shan, G. Chenjiang, Z. Yanqiu, and D. Jun, "A vector method for determining voltage standing wave ratio of radio communication", in *2015 12th IEEE International Conference on Electronic Measurement Instruments (ICEMI)*, vol. 02, 2015, pp. 656–660. DOI: 10.1109/ICEMI.2015.7494303.

[8] M. E. Sanyaolu, O. Dairo, and L. Kolawole, "Rainfade analysis at c, ku and ka bands in nigeria", *A Dissertation Submitted in the Department of Physical Sciences to the School of Postgraduate Studies, Redeemer's University Ede, Osun State, Nigeria in Partial Fulfilment of the Requirements for The Award of the Degree of Masters of Science (M. Sc) in Communication Physics*, vol. 103, 2016.

[9] A. Popleteev, "Improving ambient fm indoor localization using multipath-induced amplitude modulation effect: A year-long experiment", *Pervasive and Mobile Computing*, vol. 58, p. 101 022, 2019.

[10] M. G. Domingo, P. K. E. Estebal, G. A. Tongco, and G. P. Mappatao, "Development of a reliable path-loss model for fm broadcast reception in office locations", *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 4, pp. 1654–1661, 2020.

[11] *Telemetry — english meaning - cambridge dictionary*, https://dictionary.cambridge.org/dictionary/ english/telemetry, (Accessed on 02/20/2024).

[12] *Ant group - antgroup.it*, https://www.antgroup.it/en/mission.html, (Accessed on 02/20/2024).

[13] J. H. Reed, *Software radio: a modern approach to radio engineering*. Prentice Hall Professional, 2002.

[14] E. Buracchini, "The software radio concept", *IEEE Communications Magazine*, vol. 38, no. 9, pp. 138–143, 2000.

[15] S. Cass, "Tools & toys: Hardware for your software radio", *IEEE Spectrum*, vol. 43, no. 10, pp. 51–54, 2006.

[16] I. Martoyo, P. Setiasabda, H. Y. Kanalebe, H. P. Uranus, and M. Pardede, "Software defined radio for education: Spectrum analyzer, fm receiver/transmitter and gsm sniffer with hackrf one", in *2018 2nd Borneo International Conference on Applied Mathematics and Engineering (BICAME)*, IEEE, 2018, pp. 188–192.

[17] T. Hentschel, G. Fettweis, and M. Bronzel, "Channelization and sample rate adaptation in software radio terminals", in *3rd ACTS Mobile Commun. Summit*, 1998, pp. 121–26.

[18] H. A. Haldren III, "Studies in software-defined radio system implementation", 2014.

[19] D. C. Tucker and G. A. Tagliarini, "Prototyping with gnu radio and the usrp-where to begin", in *IEEE Southeastcon 2009*, IEEE, 2009, pp. 50–54.

[20] *Gnu radio - wikipedia*, https://en.wikipedia.org/wiki/GNU_Radio, (Accessed on 06/21/2021).

[21] *Bladerf 2.0 micro - nuand*, https://www.nuand.com/bladerf-2-0-micro/, (Accessed on 09/18/2021).

[22] *Bladerf 2.0 micro xa4 - nuand*, https://www.nuand.com/product/bladerf-xa4/, (Accessed on 10/01/2021).

[23] *Tri-band antenna - nuand*, https://www.nuand.com/product/tri-band-antenna/, (Accessed on 09/23/2021).

[24] *Guided tutorial introduction - gnu radio*, https://wiki.gnuradio.org/index.php/Guided_Tutorial_ Introduction, (Accessed on 09/19/2021).

[25] M. S. Gandhi and P. Kumar, *Gnu radio+ usrp2 implementation of a single-carrier zero-correlation-zone cdma system*, 2013.

[26] W. Song, "Configure cognitive radio using gnu radio and usrp", in *2009 3rd IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications*, 2009, pp. 1123–1126. DOI: 10.1109/MAPE.2009.5355934.

[27] *Installinggr - gnu radio*, https://wiki.gnuradio.org/index.php/InstallingGR, (Accessed on 09/19/2021).

[28] *Bladerf-micro.pdf*, https://www.nuand.com/bladeRF-micro.pdf, (Accessed on 06/21/2021).

[29] S. Saabith, T. Vinothraj, and M. Fareez, "A review on python libraries and ides for data science",

Int. J. Res. Eng. Sci, vol. 9, no. 11, pp. 36–53, 2021.

[30] K. M. Mendez, L. Pritchard, S. N. Reinke, and D. I. Broadhurst, "Toward collaborative open data science in metabolomics using jupyter notebooks and cloud computing", *Metabolomics*, vol. 15, pp. 1–16, 2019.

[31] B. M. Randles, I. V. Pasquetto, M. S. Golshan, and C. L. Borgman, "Using the jupyter notebook as a tool for open science: An empirical study", in *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, 2017, pp. 1–2. DOI: 10.1109/JCDL.2017.7991618.

[32] *Jupyter notebook documentation — jupyter notebook 7.1.0 documentation*, https://jupyter-notebook.readthedocs.io/en/latest/, (Accessed on 02/16/2024).

[33] *Introducing jupyter notebook — codecademy*, https://www.codecademy.com/article/introducing-jupyter-notebook, (Accessed on 02/16/2024).

[34] B. E. Granger and F. Pe´rez, "Jupyter: Thinking and storytelling with code and data", *Computing in Science Engineering*, vol. 23, no. 2, pp. 7–14, 2021. DOI: 10.1109/MCSE.2021.3059263.

[35] F. Skender and V. Manevska, "Data visualization tools-preview and comparison", *Journal of Emerging Computer Technologies*, vol. 2, no. 1, pp. 30–35, 2022.