Hochschule Karlsruhe
University of
Applied Sciences

HKA

# SDR-based LTE Scanner

**Author: Abdullah Alibrahim , MN: 68532**
**Instructor: Prof. Dr. -Ing. Manfred Litzenburger**

**University of Karlsruhe**
**January 2023**

# Table of Contents

# List of Figures

# I Introduction

Long-Term Evolution (LTE) is a widely used wireless communication standard that powers mobile networks. An LTE scanner is an essential tool for analyzing and capturing LTE signals, and software defined radio (SDR) devices are a optimal choice for such scanners.

The use of SDR technology enables the decoding of broadcast channel messages, which are an important source of information about the LTE network. These messages provide details such as cell identity, frequencies, and signal strength, among others. With SDR-based LTE scanners, the software can be updated to support new features and standards, making it a versatile tool for LTE analysis.

Software Defined Radio (SDR) based LTE scanners have several advantages over traditional transceivers in the context of LTE network analysis:

1. Flexibility: SDR technology allows the radio's behavior to be defined and altered through software, making it easy to add new features and standards as they emerge. This provides a more flexible and future-proof solution compared to traditional transceivers that have fixed hardware and limited upgradeability.

2. Cost-effectiveness: SDR devices are typically less expensive than traditional transceivers, making them a cost-effective option for LTE network analysis.

3. Advanced capabilities: SDR technology enables advanced capabilities such as real-time spectrum analysis, protocol decoding, and user-defined scripts for automated analysis. These capabilities are not available in traditional transceivers, which often lack the processing power to perform these tasks.

4. Improved accuracy: SDR-based LTE scanners can be calibrated more accurately than traditional transceivers, providing more precise measurements and analysis of LTE signals.

5. Software-based approach: With SDR technology, the radio's hardware remains constant while the software can be updated to support new features and standards. This software-based approach provides a more flexible and versatile solution compared to traditional transceivers, which are limited by their hardware design.

Overall, SDR-based LTE scanners offer several advantages over traditional transceivers, making them a more flexible, cost-effective, and capable solution for LTE network analysis.

# II Description

## 1 Overview

The LTE Scanner is a software application designed to operate in conjunction with Software Defined Radio Devices, specifically the USRP B200/B210. The key functions and features of the LTE Scanner:

- Scan LTE downlink bands: The LTE Scanner has the ability to scan LTE downlink bands to detect LTE cells on all frequencies supported by the USRP B200/B210 SDR device. This provides a comprehensive view of the LTE network and enables the detection of cells.

- Decode broadcasting channel messages: The LTE Scanner is capable of decoding broadcasting channel messages, specifically SIB1 (System Information Block 1). This is a crucial source of information about the LTE network, providing details as the following:

  - PLMN (Public Land Mobile Network) identity: The PLMN identity is a unique identifier for the LTE network and consists of a Mobile Country Code (MCC) and a Mobile Network Code (MNC). The MCC identifies the country where the network is located, and the MNC identifies the specific operator within that country.

  - Tracking area code: A tracking area is a group of cells within the LTE network. The tracking area code identifies the tracking area to which the cell belongs.

  - Cell Identity: The cell identity is a unique identifier for a specific cell within the LTE network.

- Web User Interface (WebUI): The LTE Scanner provides a graphical Web User Interface (WebUI) for management and results visualization.

## 2 WebUI

This section describes the usage of the application based on the functions, that are implemented in the WebUI. After starting the application, the WebUI will be accessible on the local server on port 2250 by Default.

### 2a Navigation Bar

- /Home: The main page
- /Plot: Starts plotting server, that plots the data from Cells.json as live stream.

- /Cells.json: Returns an array of all Cells as objects, which are live detected or loaded from saved results.
- /Estimation: Returns an object, that contains the frequencies where cells are detected but the BCCH decoding failed. It is helpful after scanning a band with no results. And each frequency has an array of initial cell id's. Example: {"frequency":[cell id , cell id , ... ], "another frequency":[cell id ] }.
- /Saves: Returns all saved results.
- /IMSI: Returns pdf file contains a list of (MNC, MCC) in Germany.
- /LTE Bands: Views the LTE band allocations in Germany.

## 2b Submit form

- USRP parameters:
  - Number of antennas: Number of receive channels (1 or 2)
  - Receiver gain: Defaults between 40 and 80.

- Frequency parameters:
  - Unit in MHz
  - To scan only one frequency, set the End frequency to „–1".
  - Use the step input to determine the frequency resolution.

- Advanced parameters:
  - Timeout: the value is in seconds and it determines how long should the scanner try to find a cell on each frequency.
  - Attempts: determines how many times must the scanner try to decode the BCCH after it fails.

- /Scan: Submits the order to the server.
- /Reset: Deletes all the results saved in Cells.json.

## 2c Save, load and delete

- Load/Save:
  - Save: It saves all the results from Cells.json under an Id of the users choice, to save just enter the „save id" and be sure the „load id" and the delete option are empty then click the button.

  - Load: It adds saved results to the Cells.json, to load just enter the load id and be sure the save id and the delete option are empty then click the button.

- Delete: It deletes the last loaded data permanently, to delete data first click on reset then load the results you want to delete then be sure that the load id and the save id are empty then click on the delete checkbox to be “True” then click the Load/Save button.

# III Requirements

## 3 Hardware requirements

- USRP B200/B210/X300/X310
- Antenna

## 4 Software dependencies

### 4a Node.js/npm

Version (v19.3.0)

Node.js is an open-source back-end JavaScript runtime server environment. It runs on the V8 JavaScript Engine and executes JavaScript code outside a web browser. NPM is the package manager for Node.js.

### 4b Openphy

OpenPHY is a LTE UE receiver implementation for real-time test, decoding, and network diagnostic purposes. Alternatively the implementation can serve as the basis for a full software UE implementation when combined with uplink and MAC/RRC layer functionality.

Link:
https://github.com/ttsou/openphy

General:

- LTE Release 8
- UE Category 3 (75 Mbps downlink)
- FDD mode

LTE specifications:

- 3GPP TS 36.211 “Physical channels and modulation”
- 3GPP TS 36.212 “Multiplexing and channel coding”
- 3GPP TS 36.213 “Physical layer procedures”

Physical Layer:

- Bandwidth: 1.4, 3, 5, 10, and 20 MHz (automatic detection)
- Channels: PSS, SSS, RS, PBCH, PCFICH, PDCCH, PDSCH
- PDCCH formats (DCI): 0, 1, 1A, 1C, 1D, 2, 2A
- Modulation: QPSK, QAM-16, QAM-64
- MIMO: 2x2 or 2x1 transmit mode 2 (diversity)

Decoding (TurboFEC):

- Turbo decoding for PDSCH (SIMD optimized)
- Convolutional decoding for PBCH and PDCCH (SIMD optimized)
- Turbo and convolutional rate matching

RF device support:

- Ettus Research USRP B200/B210
- Ettus Research USRP X300/X310

Processor Support:

Intel SSE3, SSE4, and AVX2 instruction support is automatically detected and enabled at build time if available.

Openphy dependencies:

- FFTW for computing the discrete Fourier transform (DFT) http://fftw.org
- USRP Hardware Driver for Ettus radio device support http://uhd.ettus.com

## 4c ASN1 compiler

The ASN.1 compiler for C is a standalone program that takes one or more input files, where each input file contains one or more ASN.1 modules, and generates C source code for encoding and decoding ASN.1 messages. The compiler verifies that the ASN.1 schema is valid, and generates the following:

- Diagnostic messages and optionally an output ASN.1 listing
- Easy to use C language data structures to be included in your application
- A control table for use by the space-optimized or lean encoder/decoder
- A time-optimized encoder/decoder

It is compiled and integrated in the LTE Scanner source code, and it is used to decode the LTE BCCH messages.

# IV LTE Decoding Procedure

Decoding an LTE (Long-Term Evolution) signal involves a series of processes for extracting information from the raw signal received from the LTE network. The LTE signal is composed of physical, transport, and logical channels, each with a specific purpose and set of information (Figure 1 LTE channels).

Physical channels play a vital role in the control and transfer of data within an LTE network. The following are the physical channels and their respective information:

- Physical Downlink Control Channel (PDCCH):
  - Control information, including scheduling assignments for the PDSCH
- Physical Downlink Shared Channel (PDSCH):
  - Carries the actual data being transmitted

These physical channels, PDCCH and PDSCH, work together to ensure efficient and effective data transfer in the LTE network.

The Master Information Block (MIB) and System Information Block Type 1 (SIB1), provide the mobile device with important network and system information. The MIB contains basic information about the network, such as the LTE band, downlink and uplink carrier frequencies, and cell identity. The SIB1, on the other hand, contains information about the network and parameters required for initial cell selection and cell reselection, including the Public Land Mobile Network (PLMN) identity, Tracking Area Code, Cell Identity, and RACH configuration.

These channels play an important role in ensuring proper communication and connection between the mobile device and the LTE network.

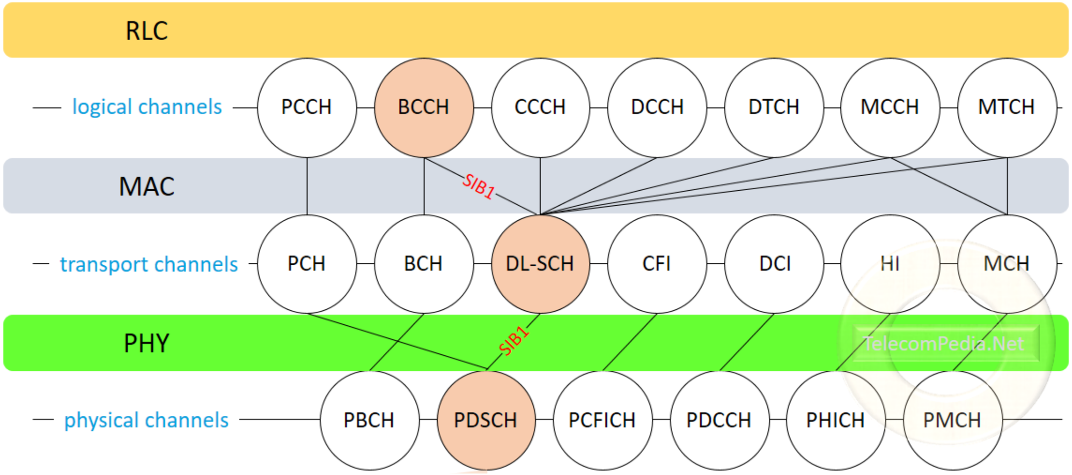


**Figure 1 LTE channels**

To extract this information from the LTE signal (Figure 2 MIB and SIB1 recovery):

1. Capture a suitable number of frames of an LTE signal using SDR hardware.
2. Determine and correct the frequency offset of the received signal.
3. Perform a blind cell search to determine the cell identity.
4. Synchronize the captured signal to the start of an LTE frame.
5. OFDM demodulate the received signal to get an LTE resource grid.
6. Perform a channel estimation for the received signal.
7. Decode the MIB for each captured frame to determine cell-wide settings.
8. Decode the CFI and PDCCH for each subframe within the captured signal.
9. Plot the reference signal measurements.
10. Based on **Information block type**, decode the MIB and PDSCH information.

    - The MIB information fields are:
        - DuplexMode - Frame structure type
        - CyclicPrefix - Cyclic prefix length
        - NDLRB - Number of downlink resource blocks
        - CellRefP - Cell-specific reference signal antenna ports
        - PHICHDuration - PHICH duration
        - Ng - HICH group multiplier
        - NFrame - Frame number

    - The PDSCH information fields are:
        - RNTI - Radio network temporary identifier value
        - PRBSet - Zero-based physical resource block (PRB) indices
        - NLayers - Number of transmission layers
        - Modulation - Modulation type
        - RV - Redundancy version
        - TxScheme - Transmission scheme
        - CRC - Cyclic Redundancy check
        - Transport Block

11. Send the Transport Block of the PDSCH to the ASN1 compiler to decode the SIB1 information (BCCH message).

    - The SIB1 information fields are:
        - PLMN Identity List: up to 6 PLMN can be identified in the list. First one is primary PLMN.
        - PLMN Identity: identifies operator global identity, combination of MCC and MNC.
        - Tracking Area Code: identifies a tracking area for paging the users.
        - Cell Identity: identifies a cell within PLMN.
        - q_RxLevMin: minimum required received RSRP level for cell selection. Actual value in dBm is obtained by multiplying by two.

Repeat all the steps in decoding procedure for each frequency in the provided range of frequencies.

Example of successfully decoded BCCH message (ASN1):

```
<BCCH-DL-SCH-Message>
  <message>
    <c1>
      <systemInformationBlockType1>
        <cellAccessRelatedInfo>
          <plmn-IdentityList>
            <PLMN-IdentityInfo>
              <plmn-Identity>
                <mcc>
                  <MCC-MNC-Digit>2</MCC-MNC-Digit>
                  <MCC-MNC-Digit>6</MCC-MNC-Digit>
                  <MCC-MNC-Digit>2</MCC-MNC-Digit>
                </mcc>
                <mnc>
                  <MCC-MNC-Digit>0</MCC-MNC-Digit>
                  <MCC-MNC-Digit>3</MCC-MNC-Digit>
                </mnc>
              </plmn-Identity>
              <cellReservedForOperatorUse><notReserved/></cellReservedForOperatorUse>
            </PLMN-IdentityInfo>
          </plmn-IdentityList>
          <trackingAreaCode>
            1110100011101110
          </trackingAreaCode>
          <cellIdentity>
            0001011000000100110100010111
          </cellIdentity>
          <cellBarred><notBarred/></cellBarred>
          <intraFreqReselection><allowed/></intraFreqReselection>
          <csg-Indication><false/></csg-Indication>
        </cellAccessRelatedInfo>
        <cellSelectionInfo>
          <q-RxLevMin>-62</q-RxLevMin>
        </cellSelectionInfo>
        <freqBandIndicator>20</freqBandIndicator>
        <schedulingInfoList>
          <SchedulingInfo>
            <si-Periodicity><rf16/></si-Periodicity>
            <sib-MappingInfo><sibType3/>
            </sib-MappingInfo>
          </SchedulingInfo>
          <SchedulingInfo>
            <si-Periodicity><rf32/></si-Periodicity>
            <sib-MappingInfo><sibType5/>
            </sib-MappingInfo>
          </SchedulingInfo>
          <SchedulingInfo>
            <si-Periodicity><rf64/></si-Periodicity>
            <sib-MappingInfo><sibType7/>
            </sib-MappingInfo>
          </SchedulingInfo>
        </schedulingInfoList>
        <si-WindowLength><ms40/></si-WindowLength>
        <systemInfoValueTag>3</systemInfoValueTag>
        <nonCriticalExtension>
          <lateNonCriticalExtension>49 2A 03 00 C0</lateNonCriticalExtension>
          <nonCriticalExtension>
            <ims-EmergencySupport-r9><true/></ims-EmergencySupport-r9>
            <cellSelectionInfo-v920>
```

```
                                        <q-QualMin-r9>-18</q-QualMin-r9>
                                    </cellSelectionInfo-v920>
                                </nonCriticalExtension>
                            </nonCriticalExtension>
                        </systemInformationBlockType1>
                    </c1>
                </message>
            </BCCH-DL-SCH-Message>
```

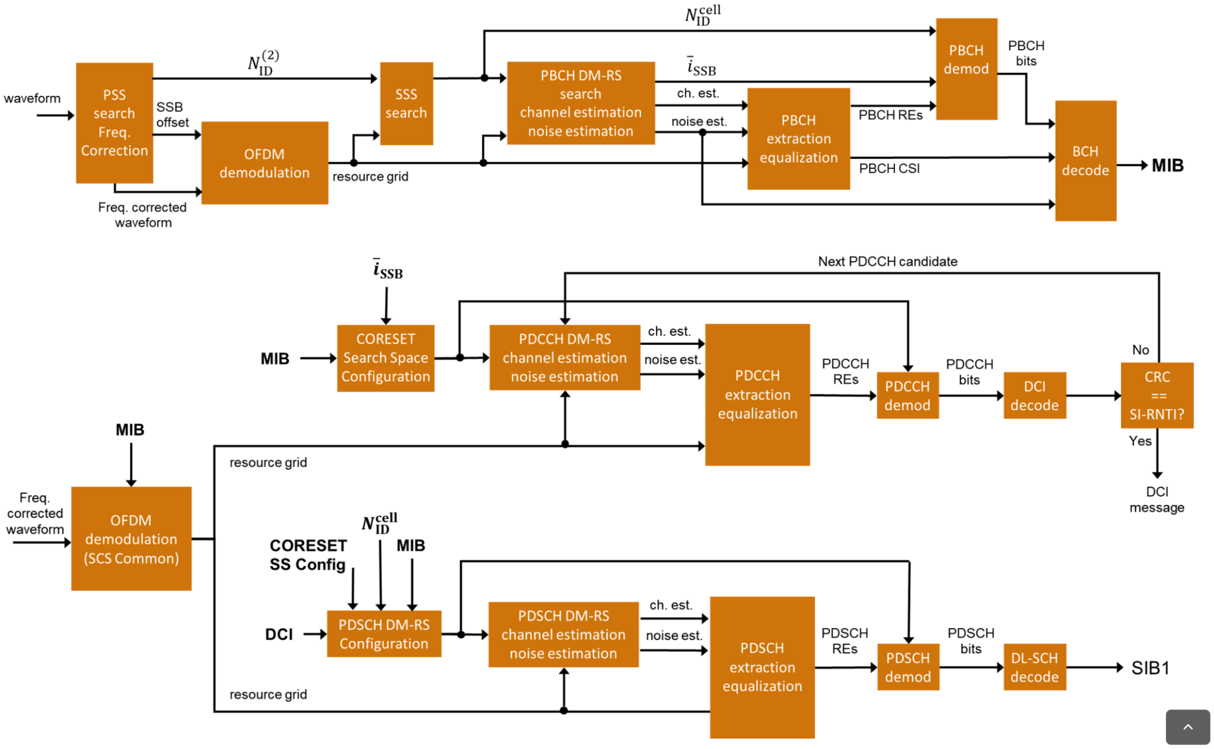


**Figure 2 MIB and SIB1 recovery**

# V Installation and setup

## 5 Operating system

Ubuntu 22.04.1 LTS (Jammy Jellyfish) is recommended to be used. It can be installed using the following link:
https://releases.ubuntu.com/22.04/

### 5a Install Ubuntu on a USB

Follow the instructions on the following website to install ubuntu on an USB stick.
https://itsfoss.com/intsall-ubuntu-on-usb/

## 6 Dependencies installation guide

Note: The order of installing the dependencies is important.

### 6a Update and upgrade ubuntu repositories

After installing the OS the default package manager of ubuntu apt will be used to install the dependencies. Update and upgrade its repositories by executing the following commands on the terminal:

```
sudo apt update
sudo apt upgrade
```

### 6b USRB Hard Drive

Execute the following command to install the USRB driver:

```
#install UHD packages
sudo apt-get install libuhd-dev uhd-host

#download UHD images
sudo /usr/lib/uhd/utils/uhd_images_downloader.py
```

### 6c FFTW

```
#install build packages
sudo apt install build-essential

#download source code, then extract it and enter the repository
wget http://www.fftw.org/fftw-3.3.10.tar.gz
tar -xf fftw-3.3.10.tar.gz
```

```
cd fftw-3.3.10/

#configure, build and install
./configure
make
sudo make install
```

## 6d OpenPHY

```
#install packages and libraries
sudo apt install git
sudo apt install autoconf
sudo apt install libtool

#clone source code and enter the repo
git clone https://github.com/ttsou/openphy.git
cd openphy/

#configure, build and install
autoreconf -i
./configure
make
sudo make install
```

## 6e Node.JS

```
#install packages
sudo apt install nodejs npm
sudo npm install -g n
sudo n latest
```

Note: Reboot the system then continue.
Recommended: install chrome browser then set it as default browser and from chrome settings, allow pop-ups and redirects.

## 6f LTE Scanner

```
#set git config and clone source code
git config --global user.name "elpmiS"
git config --global user.email "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@github.com"
git clone https://github.com/e1pmiS/LTE_Scanner.git

#enter the repository and install Nodejs dependencies
cd LTE_Scanner/
npm install

#To start the API.
node main.js
```

At this point the WebUI will be locally accessible on the following link: http://localhost:2250

# VI OS Credentials and start guide

## 7 OS Credentials

- Username: hka
- Password: 123

## 8 start guide

To start the application, open the Terminal and execute the following:

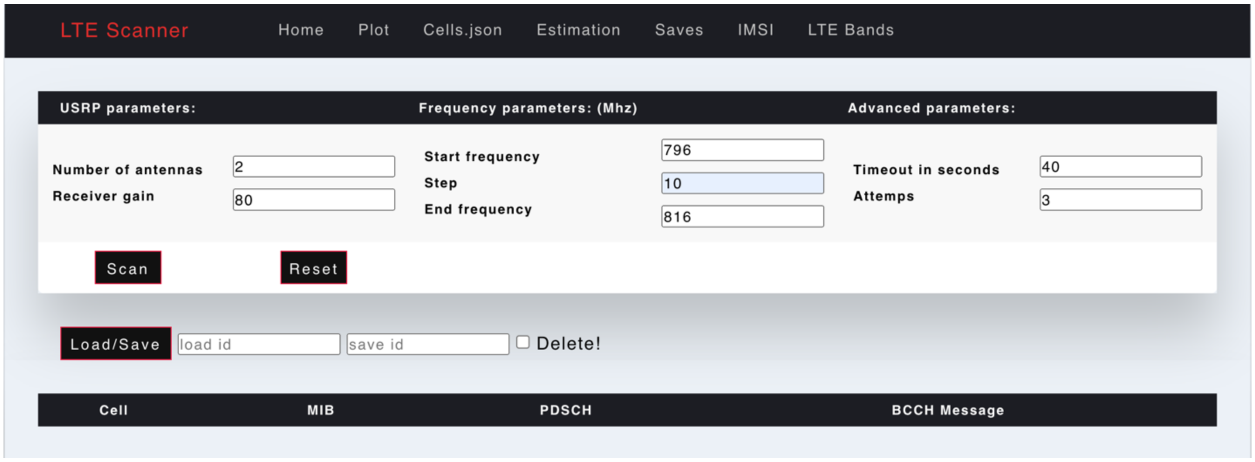
```
#enter the repository

cd LTE_Scanner/

#To start the API.

node main.js
```

# VII Test Cases

## 9 LTE band 800

In this test, the frequencies at which LTE cells can be detected were determined to be 796 MHz, 806 MHz, and 816 MHz. The WebUI was used to initiate a scan by specifying the start frequency, step value, and end frequency (796 MHz, 10, and 816 MHz, respectively), while keeping other parameters at their default values. The scan was initiated by clicking on the scan button, and the results were automatically live streamed as shown in Figure 4 WebUI - Results,  Figure 5 Node plot - Results.

### 9a WebUI



USRP parameters:

Number of antennas: 2

Receiver gain: 80

Frequency parameters: (Mhz)

Start frequency: 796

Step: 10

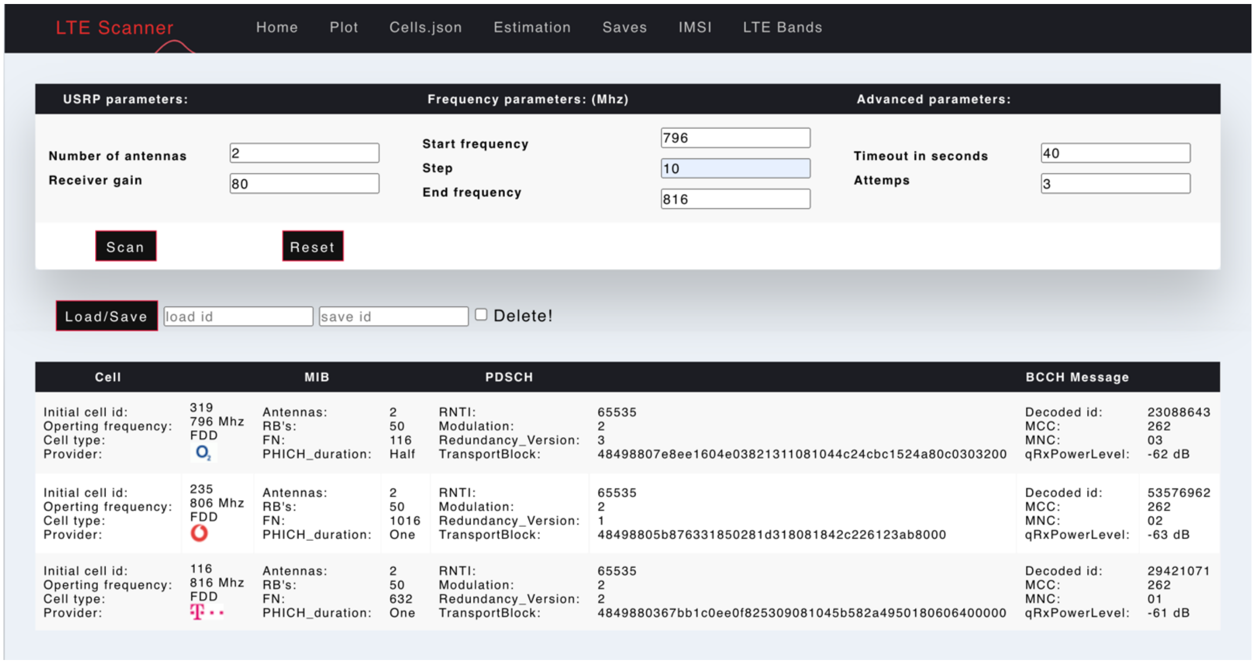
End frequency: 816

Advanced parameters:

Timeout in seconds: 40

Attemps: 3

**Figure 3 WebUI - Submit parameters**



| USRP parameters: | | Frequency parameters: (Mhz) | | Advanced parameters: | |
|---|---|---|---|---|---|
| Number of antennas | 2 | Start frequency | 796 | Timeout in seconds | 40 |
| Receiver gain | 80 | Step | 10 | Attemps | 3 |
| | | End frequency | 816 | | |

| Cell | | MIB | | PDSCH | | BCCH Message | |
|---|---|---|---|---|---|---|---|
| Initial cell id:<br>Operting frequency:<br>Cell type:<br>Provider: | 319<br>796 Mhz<br>FDD<br>O2 | Antennas:<br>RB's:<br>FN:<br>PHICH_duration: | 2<br>50<br>116<br>Half | RNTI:<br>Modulation:<br>Redundancy_Version:<br>TransportBlock: | 65535<br>2<br>3<br>48498807e8ee1604e03821311081044c24cbc1524a80c0303200 | Decoded id:<br>MCC:<br>MNC:<br>qRxPowerLevel: | 23088643<br>262<br>03<br>-62 dB |
| Initial cell id:<br>Operting frequency:<br>Cell type:<br>Provider: | 235<br>806 Mhz<br>FDD | Antennas:<br>RB's:<br>FN:<br>PHICH_duration: | 2<br>50<br>1016<br>One | RNTI:<br>Modulation:<br>Redundancy_Version:<br>TransportBlock: | 65535<br>2<br>1<br>48498805b876331850281d318081842c226123ab8000 | Decoded id:<br>MCC:<br>MNC:<br>qRxPowerLevel: | 53576962<br>262<br>02<br>-63 dB |
| Initial cell id:<br>Operting frequency:<br>Cell type:<br>Provider: | 116<br>816 Mhz<br>FDD<br>T · · | Antennas:<br>RB's:<br>FN:<br>PHICH_duration: | 2<br>50<br>632<br>One | RNTI:<br>Modulation:<br>Redundancy_Version:<br>TransportBlock: | 65535<br>2<br>2<br>4849880367bb1c0ee0f825309081045b582a4950180606400000 | Decoded id:<br>MCC:<br>MNC:<br>qRxPowerLevel: | 29421071<br>262<br>01<br>-61 dB |

**Figure 4 WebUI - Results**

## 9b Node Plot

The LTE scanner application is designed with a built-in plotting server that displays all detected cells on a plot. The plot features two axes, with one representing the operating frequency and the other representing the minimum received signal level (Rx min level). As the pointer on the plot encounters a cell, information about that particular cell is displayed. This feature allows the user to easily visualize the detected cells and gain insight into their operating frequencies and signal strengths.

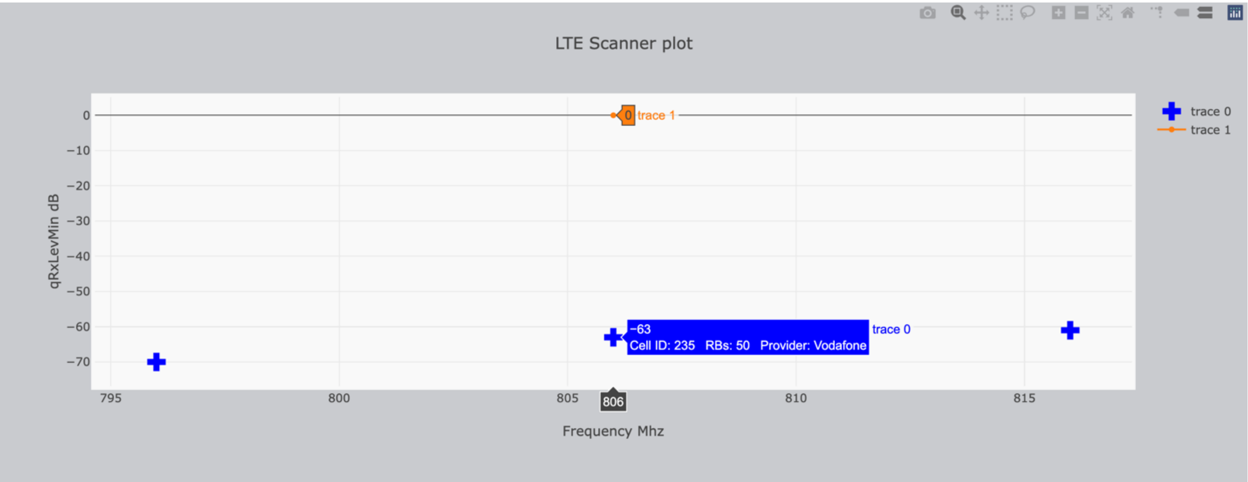
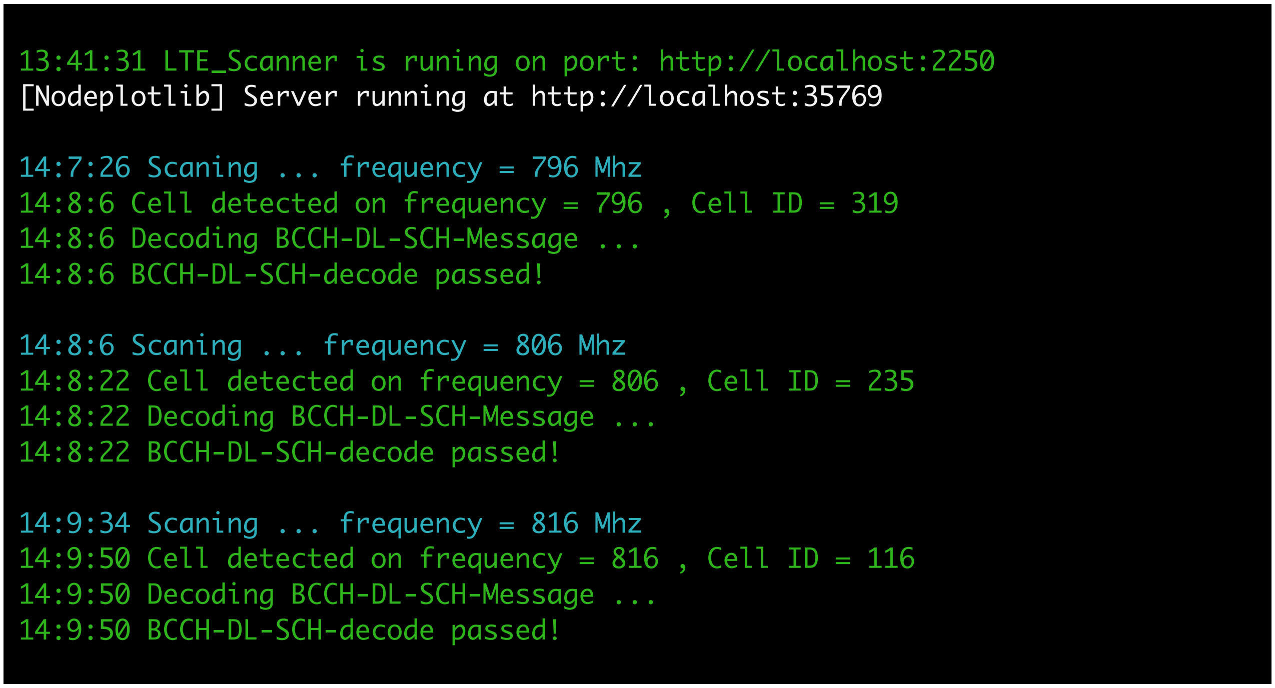


**Figure 5 Node plot - Results**

## 9c Console log

On the Terminal side the console logger is designed to show more specific information about the scanning process as shown below.



```
13:41:31 LTE_Scanner is runing on port: http://localhost:2250
[Nodeplotlib] Server running at http://localhost:35769

14:7:26 Scaning ... frequency = 796 Mhz
14:8:6 Cell detected on frequency = 796 , Cell ID = 319
14:8:6 Decoding BCCH-DL-SCH-Message ...
14:8:6 BCCH-DL-SCH-decode passed!

14:8:6 Scaning ... frequency = 806 Mhz
14:8:22 Cell detected on frequency = 806 , Cell ID = 235
14:8:22 Decoding BCCH-DL-SCH-Message ...
14:8:22 BCCH-DL-SCH-decode passed!

14:9:34 Scaning ... frequency = 816 Mhz
14:9:50 Cell detected on frequency = 816 , Cell ID = 116
14:9:50 Decoding BCCH-DL-SCH-Message ...
14:9:50 BCCH-DL-SCH-decode passed!
```

## VIII Limitaions

- The Scanner works fine only on LTE bands lower than 1 Ghz, and to get better results on higher bands use GPSDO module or external frequency reference.

- Only the following SDR devices are supported:
    - Ettus Research USRP B200/B210
    - Ettus Research USRP X300/X310

## IX Conclusion

In conclusion, SDR-based LTE scanners are a powerful tool for capturing and analyzing LTE signals in mobile networks for several reasons.

Firstly, SDR technology enables the decoding of broadcast channel messages, which provide important information about the LTE network, including cell identity, frequencies, and signal strength, among others. This information can be used to analyze and optimize the performance of the network.

Secondly, SDR devices are more flexible than traditional transceivers, as their behavior can be defined and altered through software. This allows for the addition of new features and standards as they emerge, making SDR-based LTE scanners a versatile and future-proof solution.

Thirdly, SDR technology is cost-effective compared to traditional transceivers, making it a more accessible option for LTE network analysis.

Fourthly, SDR-based LTE scanners offer advanced capabilities such as real-time spectrum analysis, protocol decoding, and user-defined scripts for automated analysis. These capabilities are not available in traditional transceivers, which often lack the processing power to perform these tasks.

Finally, SDR-based LTE scanners can be calibrated more accurately than traditional transceivers, providing more precise measurements and analysis of LTE signals. This accuracy enables a deeper understanding of the network and its performance.

# X References

(openphy), URL: https://github.com/ttsou/openphy
(FFTW), URL: http://fftw.org
(Ettus-UHD), URL: http://uhd.ettus.com
(Nodejs), URL: https://nodejs.org/en/docs/
(Telekompedia), URL: https://telecompedia.net/sib1-in-lte/
(Mathworks), URL: https://de.mathworks.com/help/5g/ug/nr-cell-search-and-mib-and-sib1-recovery.html