Nigerian Research Journal of Engineering and Environmental Sciences 7(1) 2022 pp. 338-345
p ISSN: 2635-3342; e ISSN: 2635-3350

Nigerian Research Journal of Engineering and Environmental Sciences
Journal homepage: www.rjees.com

Original Research Article

Building a Cost-effective Cellular Network using OpenBTS and USRP B210

***Bello, N. and Kanu, O.A.**

Department of Electrical/Electronic Engineering, Faculty of Engineering, University of Benin, Benin City, Edo State, Nigeria.
*nosabello@uniben.edu; kanu.ogechukwu@eng.uniben.edu

http://doi.org/10.5281/zenodo.6726443

ARTICLE INFORMATION

Article history:
Received 03 Oct, 2021
Revised 09 Dec, 2021
Accepted 12 Dec, 2021
Available online 30 Jun, 2022

Keywords:
GSM
GSM network
OpenBTS
USRP B210
Software-defined radio
Software-based BTS

ABSTRACT

The paper presents the development and implementation of a global system for mobile communication (GSM) network using OpenBTS and software-defined radio (SDR) that imitates the traditional GSM architecture. Compared to the traditional GSM architecture, the use of a software-based framework is cost-effective and useful for research in man-in-the-middle attacks. The paper provides a step-by-step installation of OpenBTS software, a software-based GSM architecture running on an Ubuntu 16.04 LTS operating system. The proposed setup was deployed and tested successfully and, in the end, subscribers were registered on the test network and given phone numbers that could make and receive phone calls and send text messages, which are regular features of the traditional GSM network.

1. INTRODUCTION

As humans, communication is an integral part of our daily lives. Communication has evolved and taken different forms from cave paintings, carrier pigeons and smoke signals in ancient times, to telegraph, fax and postal services in the later times and then to digital communication in recent times (Kalamtime, 2021). The ancient times of communication suffered the drawback of the inability to convey beyond a single bit of information. The desire of humans to communicate over a distance greater than that feasible with the human voice, but with a similar scale of expediency has propelled the evolution of communication (Huurdeman, 2003). Prior systems (such as postal mail) were slow and thus, over time, mobile communication has become a necessity.

The transmission and reception of data/information have become the bed rock of institutions and individual day to day activities (Kitchin, 2014). Global System for Mobile communication (GSM) is one of the leading

mobile communication platforms and its features are constantly upgraded to provide customers with optimum satisfaction. However, the setup of mobile communication is very expensive and as such, many rural areas may never get to enjoy the benefits of mobile communication. This is owing to two distinct facts. Firstly, network service providers may not venture into setting up in rural areas, that cannot afford the services and thus run at a loss in such areas (UNCTAD, 2019). Secondly, the topography in such rural areas may not encourage the setup of, the necessary infrastructures for mobile communication (Alenoghena and Emagbetere, 2012).

In Nigeria, most rural areas are not connected to the electric grid (Elusakin et al., 2014). They encapsulate most sparsely populated areas, coupled with poor topography road infrastructure and lack of other necessary amenities. Therefore, deploying GSM networks in such areas will be expensive, as the villages are dispersed, separated by kilometers of inhabited areas (Simo-Reigadas et al., 2015). According to the Guardian newspaper, the cost of setting up a base transceiver station (BTS) in Nigeria ranges between 40 to 50 million Naira, excluding the annual maintenance cost (Guardian, 2018). With this in mind, no network provider would want to set up in any rural area, as it would be practically too expensive to spend such amount of money, just to setup a base station that will cover quite a small number of persons (UNCTAD, 2019).

This project therefore investigates the use of portable software-driven hardware components and software implementations of the traditional GSM architecture to realize exactly the same functions of the GSM network. With the key components of the network implementation being more economical than the traditional setup, the use of software as an alternative to the already existing GSM network, gives an opportunity for prototyping, future proofing and research (Ben, 2016). One can imprint the knowledge of the GSM network through prototyping, one can perform system upgrades without making rigorous and expensive physical changes, as would be the case in traditional GSM mobile network and lastly, the security of the network can be tested and improved (Kostrzewa, 2011).

2. MATERIALS AND METHODS

2.1. Materials for Software Development

A complete testbed setup for the software based cellular network consisted of a 64-bit Intel core i3 computer running on Linux operating system with 2 GB RAM, 30 GB hard disk space, USB 3.0 port, an Ettus research software defined radio (SDR), universal software radio peripheral (USRP) B210 and two GSM antennas. The OpenBTS software which implements the GSM hybrid architecture was designed to be compatible with Ettus research SDRs and the choice of the B210 was because of the relative low cost compared to other USRPs, the full duplex MIMO (2 Tx and 2 Rx) operation and fast USB 3.0 connectivity (Ettus Research, 2021). A VERT900 omnidirectional rubber duck antenna which operate between 824 to 960 MHz with a 3 dBi gain was used.

2.2. Dependencies Setup for the OpenBTS Framework

The software requirements for the setup were basically the Linux OS, the USRP hardware driver (UHD) for communication between the computer and the SDR and the successful build of the binaries of the OpenBTS software. The computer was setup with Ubuntu 16.04 LTS, after which the UHD version 3.8.5 was built from the source files before the installation of the OpenBTS software. The choice of the version of UHD was deliberate as an earlier version was incompatible with the dependencies of the OpenBTS. The steps taken to build the UHD are shown in Figure 1. It shows the commands entered in the terminal of the Linux environment. First, the repository is clone using the git software, thereafter, a build directory is created within the $uhd/host/$ directory. The desired version of UHD to be installed in the OS is selected with the $checkout$ command while $tag - l$ lists the version of UHD available. Within the build directory, the $cmake$ compiler handles the installation with the set of commands shown. Lastly, the $ldconfig$ registers the path of the installed binary file. However, it should be noted that the necessary dependencies for the build were installed which can be found on the Ettus webpage.

```
git clone https://github.com/EttusResearch/uhd.git

cd uhd/
git tag -l

git checkout release 003 008 005

cd host/
mkdir build/
cd build/

cmake ../
make
sudo make install

sudo ldconfig

export LD_LIBRARY_PATH=/usr/local/lib/
source ~/.bashrc
```

Figure 1: Building the UHD for the USRP B210 SDR (Behan et al., 2017; Aggrawal and Vachhani, 2017)

After a successful installation of the UHD, the installation of the OpenBTS software follows the same process. It was cloned and built from the range network repository on GitHub. The steps are shown in Figure 2. All the necessary dependencies needed by the OpenBTS library were installed in the host computer at the first stage of the building process shown in Figure 2. For the set of official release packages which were downloaded, it is needed to install some additional system libraries and define an additional repository source so all dependencies can be found and installed. OpenBTS uses the A5/3 shared library to support call encryption. It contains cryptographic routines that must be distributed separately from OpenBTS, hence the liba53 library was built. OpenBTS also uses the coredumper shared library to produce meaningful debugging information if OpenBTS crashes and this was also implemented.

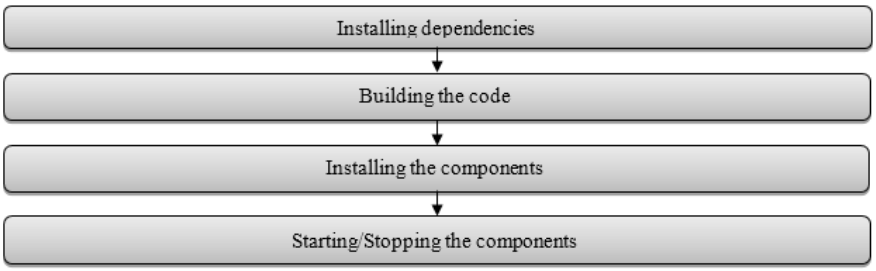


Figure 2: Building the OpenBTS software

For the next step, the build.sh script was used to compile the binary packages. It automatically installs the compiler and autoconfiguration tools as well as any required dependencies. It also controls which radio transceiver application will be built. As there are several different drivers available for the various radio types, build.sh requires an argument so it knows which hardware is being targeted (valid radio types are SDR1, USRP1, B100, B110, B200, B210, N200, and N210). This process might take close to an hour to complete. When the build script finishes, there was a new directory named “BUILDS” containing a subdirectory with the build’s timestamp which contains the deb files that is used for installation of the components. By installing all of the components on a successfully development setup, it is guaranteed that there is a functional GSM network-in-a-box. Everything needed for voice, SMS, and data will be running in a single system. The commands to execute the installation of the deb files are as shown in Figure 3 (Hatorangan and Juhana, 2014; Iedema, 2014):

```
sudo dpkg -i range-configs_5.1-master_all.deb
sudo dpkg -i range-asterisk*.deb && sudo apt-get install –f
sudo dpkg -i sipauthserve_5.0_amd64.deb && sudo apt-get install –f
sudo dpkg -i smqueue_5.0_amd64.deb && sudo apt-get install –f
sudo dpkg -i openbts_5.0_amd64.deb && sudo apt-get install –f
```

Figure 3: Installing the deb file of the various components of the OpenBTS framework

The first command installs the system config that sets the default configurations. It includes settings for the network interface, firewall rules, domain name system (DNS) configuration, logging, etc. The remaining commands install the Asterisk, SIPAuthserve, the SMQueue and the OpenBTS components. Lastly, after the complete successful installation, the last step of the building procedure which is to start and stop the components should be taken. The command for starting and conversely stopping the components are as shown in Figure 4.

```
sudo start <component/service>
sudo stop <component/service>
```

Figure 4: Commands for starting and stopping the services of the OpenBTS framework

To successfully start all the components, all other components apart from the OpenBTS component should start before it is started. In other words, start the OpenBTS component as the last instant. With this concludes the setting up of OpenBTS on Ubuntu 16.04 LTS.

2.3. Deployment of Setup

The USRP B210 device was setup by connecting the VERT900 antennas to the TX/RX and RX2 of RF A while keeping them perpendicular to each other as shown in Figure 5. If the antennas are parallel to each other, the signal can efficiently travel from the transmit to the receive antenna, but when the antennas form a 90-degree angle, the signal is being transmitted on a different plane than it is being received on.

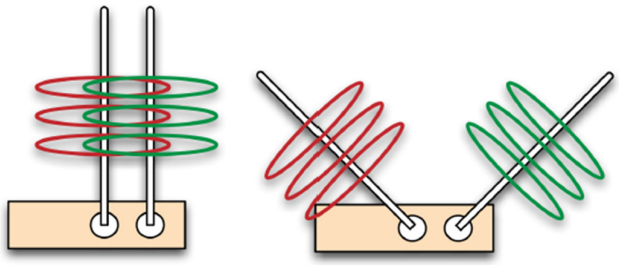


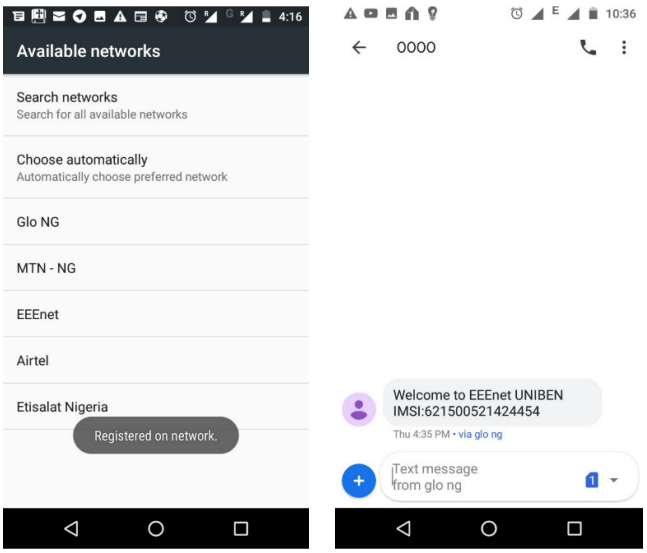
Figure 5: Antenna alignment

The USRP B210 was connected to the host computer through a 3.0 USB port and the successful connection was confirmed through the UHD check commands. This will make the OpenBTS service to automatically start an instance of the transceiver software and connect to the radio hardware. Radio samples are then exchanged between the transceiver software and OpenBTS software over a local user datagram protocol (UDP) socket. All configuration of OpenBTS was accomplished by manipulating keys stored in an SQLite3 database (Sankhe et al., 2014; Sugeng and Putri, 2018). By default, this database is stored at /etc/OpenBTS/OpenBTS.db. Each key is defined in a schema that is compiled in OpenBTS and used to validate the values being used. The easiest way to manipulate the configuration keys is via the OpenBTS command line interface (CLI). It can be accessed by the command sudo ./OpenBTS in the terminal. With the OpenBTS prompt (OpenBTS>), commands, including configuration changes, can be executed for processing by OpenBTS. After a successful setup of the test network, the messaging and voice capability were tested.

3. RESULTS AND DISCUSSION

3.1. Registration

The registration of a mobile phone on the test network, was done by manually scanning for available GSM networks within the area, by changing mobile network in cellular setting of the mobile phone, from automatic to manual and selecting the test network as shown in Figure 6a. Only SIM-cards with their IMSI that are already registered on the network will be authenticated on the network. The mobile phone receives a custom welcome message from the network, that contains the welcome message and the IMSI of the mobile SIM as shown in Figure 6b. At this stage, the user has been successfully subscribed to the network and thus can SMS and call other users of the network.



(a) (b)

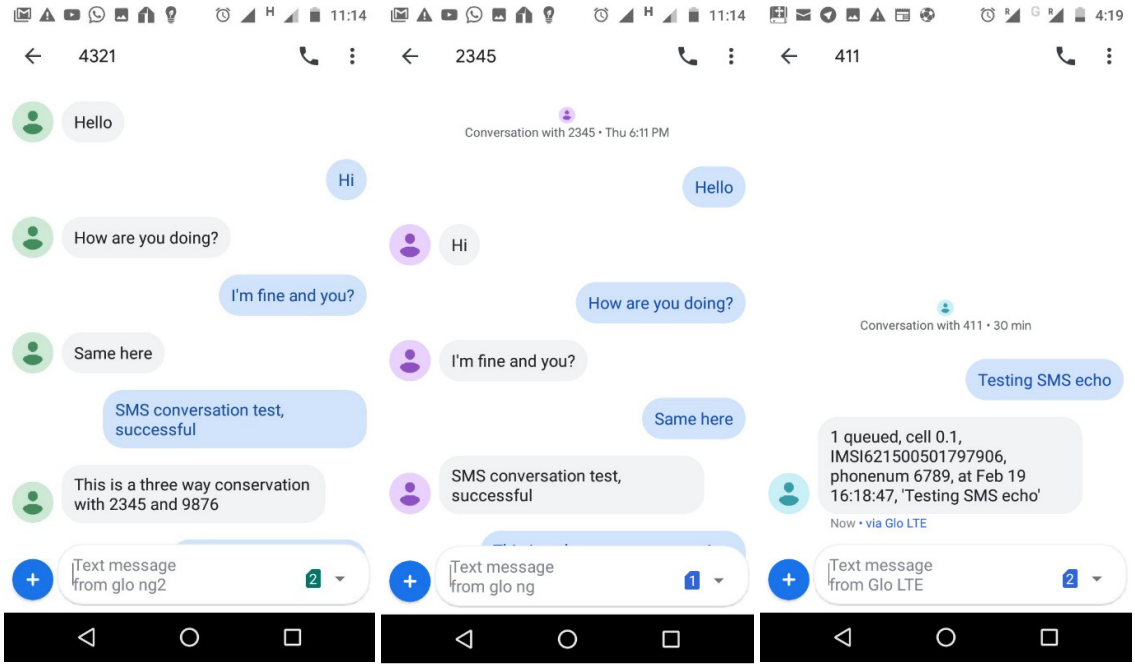
Figure 6: Subscriber registration and authentication (a) registration (b) welcome message

3.2. Sending SMS and Making Calls

To test the connectivity of the mobile phone with the network, a test message was composed and sent to 411, from the registered phone on the test network. Figures 7a and 7b shows the SMS conversation between the two phones. 411 is a short-code handler in SMQueue and will just echo back the message along with some information about the subscriber and this is shown in Figure 7c. Instead of an echo message, an SMS conversation can be initiated between two subscribers of the network and that was tested with two mobile phones registered with the network. The successful test showed the two-party SMS conversation capability of the test network and for the number of SMS messages tested within the coverage of the network the SMS arrived almost immediately.

SMS can also be sent from the test network with the “sendsms” command in the OpenBTS CLI. It is done by specifying the destination IMSI and a source number as seen in Figure 8. Besides the SMS test, there is the voice test and OpenBTS allows one to perform a tone test to a registered user. To perform the tone test, a test call was placed to the number 2602 and as expected, the tone test just plays back a constant tone and is typically used to check that the asterisk component is running as well as the proper configuration and functionality of the call routing and downlink audio (Iedema, 2014). Still in the voice test, a call was placed to 2600 from the test mobile handset. The number 2600 is a short-code handler of asterisk that provide echo service just like 411 of the SMQueue. Basically, all audio information that asterisk receives will immediately be echoed back to the caller. In addition to confirming the items listed for the test tone call, the echo call will

reveal any delay or uplink quality issues present in your network. A final call was placed from one of the mobile phones to another as shown in Figure 9. This test verifies that the test network performs voice calls between two registered users.



(a) (b) (c)

Figure 7: SMS test

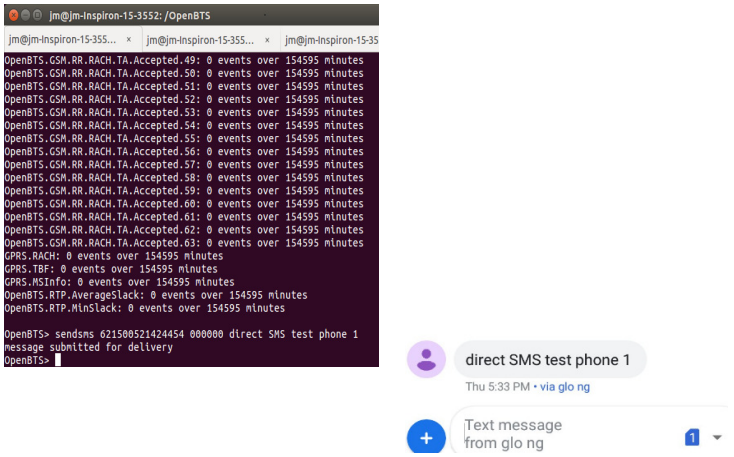


Figure 8: Direct SMS test

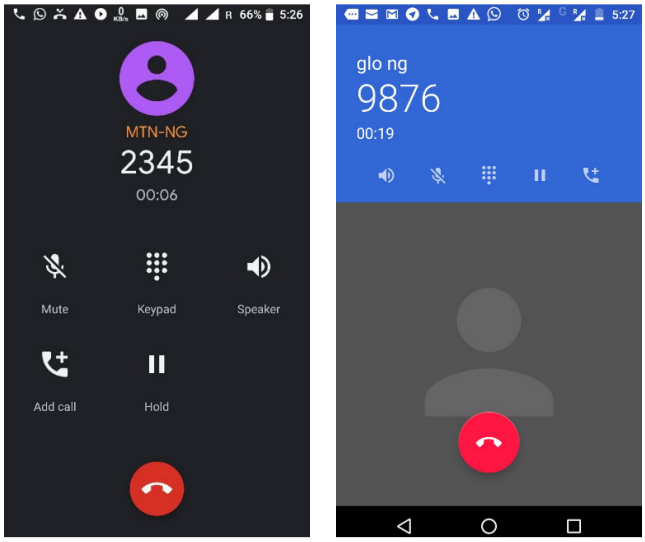


Figure 9: Voice test

This model was setup as a small-scale network as it had approximately a 30-meter radius coverage in a non-line of sight (NLOS) environment which depends on the activity of the day in the GSM band used. This model can be scaled up into a much bigger network, which can be used in the event of natural disasters and rural areas where there is no GSM communication. In a field deployment, the USRP B210 cannot be used alone due to its limitation, in terms of networks coverage area as a result of its, low transmit power at the RF front end. To solve this drawback a RF power amplifier can be connected to the front end of the B210, making the signal high enough to provide a microcell or, another solution would be to purchase a SDR with high transmit power (Manickam, 2018).

4. CONCLUSION

In this study, a low-cost localized GSM mobile system that is scalable was presented which can be implemented within a short period and thus can facilitate quick restoration of wireless connectivity in disaster hit areas and rural areas. This model was implemented in a laboratory setting and analyzed with realistic deployment network scenarios. The system solves an important problem of providing localized SMS and voice connectivity, in area where there is no such infrastructure, due to the high cost of setting up traditional GSM mobile communication network.

5. ACKNOWLEDGEMENT

The authors would like to appreciate the contribution and support from the staff of the Department of Electrical/Electronic Engineering, University of Benin, Benin City, towards the success of this research. Also, the efforts of Ugbala Valentino are acknowledged by the authors.

6. CONFLICT OF INTEREST

There was no conflict of interest regarding this research work.

REFERENCES

Aggrawal, K. and Vachhani, K. (2017). Reconfigurable cellular GSM network using USRP B200 and OpenBTS for disaster-hit regions. IEEE 13th Malaysia International Conference on Communications (MICC), pp. 141–146.

Alenoghena, C.O. and Emagbetere, J.O. (2012). Base station placement challenges in cellular networks: The Nigerian experience. IEEE 4th International Conference on Adaptive Science & Technology (ICAST), pp. 7–11.

Ben, H. (2016) *Welcome to GNU Radio*. Available: https://www.youtube.com/watch?v=4zW7rB15LrY [30 Nov 2020]

Behan, L., Orcik, L., Rezac, F., Baronak, I. and Lin, J.C.W. (2017) *'Prepaid voice services based on openbts platform'*, Proceedings of the 3rd Czech-China Scientific Conference 2017.

Elusakin, J. E., Olufemi, A.O. and Chuks, D.J. (2014). Challenges of sustaining off-grid power generation in Nigeria rural communities. *African Journal of Engineering Research*, 2, pp. 51–57.

Ettus Research (2021) *USRP B210 (Board Only)*, [Online], Available at: https://www.ettus.com/all-products/ub210-kit/ [13 Jun 2021].

Guardian (2018) Telecoms operators spend 40 million naira to site base transceiver stations, [Online], Available at: https://guardian.ng/technology/telecoms-operators-spend-40m-to-site-basetransceiver-stations/.

Hatorangan, E. and Juhana, T. (2014). Mobile phone auto registration to openbts-based cellular network in disaster situation. 8th International Conference on Telecommunication Systems Services and Applications (TSSA), pp. 1–3.

Huurdeman, A.A. (2003) *The worldwide history of telecommunications*, John Wiley & Sons.

Iedema, M. (2014) Getting started with OpenBTS: build open source mobile networks. O'Reilly Media, Inc., p. 1–3.

Kalamtime (2021), *'Evolution of Communication from Ancient to Modern Times'* [Online], Available: [30 May 2021]

Kitchin, R. (2014) *The data revolution: Big data, open data, data infrastructures and their consequences*, Sage.

Kostrzewa, A. (2011). Development of a man in the middle attack on the GSM Um-Interface. *Technische Universitt Berlin Fakultt IV, Institut f*

Manickam, S. (2018). Design Concepts of Low-Noise Amplifier for Radio Frequency Receivers. In: *RF Systems, Circuits and Components*, IntechOpen.

Sankhe, K., Pradhan, C., Kumar, S. and Murthy, G.R. (2014). Cost effective restoration of wireless connectivity in disaster hit areas using OpenBTS. 2014 Annual IEEE India Conference (INDICON), pp. 1–6.

Simo-Reigadas, J., Municio, E., Morgado, E., Castro, E.M., Martinez, A., Solorzano, L.F. and Prieto-Egido, I. (2015). Sharing low-cost wireless infrastructures with telecommunications operators to bring 3G services to rural communities. *Computer Networks*, 93, pp. 245–259.

Sugeng, W. and Putri, T.D. (2018). Communication System Design of Remote Areas using Openbts. *International Journal of Advanced Computer Science and Applications*, 9, pp. 224–229.

United Nations Conference on Trade and Development (UNCTAD) (2019) *Digital Economy Report 2019: Value creation and capture: implications for developing countries*, United Nations Publications New York, NY.