Design and Implementation of a Multi-Node Optical Wireless Communication Testbed for Centralized Configuration and Adaptation of System Parameters using GNURadio's XML-RPC and ZMQ Modules

Chukwunodebem Christopher Onwuchekwa C.ONWUCHEKWA001@UMB.EDU
Maryam AminuMukhtar M.AMINUMUKHTAR001@UMB.EDU
Lenny Martinez LENNY.MARTINEZ001@UMB.EDU
Ashley D Lemus ASHLEY.LEMUS001@UMB.EDU
Victoria Planchart V.PLANCHART001@UMB.EDU
Fedor Y Semash FEDOR.SEMASH001@UMB.EDU
KC Kerby-Patel KC.KERBY-PATEL@UMB.EDU
Honggang Zhang HONGGANG.ZHANG@UMB.EDU
Micheal B Rahaim* MICHAEL.RAHAIM@UMB.EDU
University of Massachusetts Boston, 100 Morrissey Blvd., Boston, MA 02125

Abstract

Over the past two decades, Software Defined Radio (SDR) techniques have allowed the RF community to bring many wireless communications topics and educational tools from theory and simulation to experimental test systems. In our work, we have brought these capabilities to the field of optical wireless communications (OWC). In this article, we discuss key capabilities for extending our prior work to multi-cell and multi-user OWC systems with centralized configuration and control. These capabilities are enabled by GNURadio and introduced within the gr-owc out-of-tree library. We highlight how these capabilities are being applied for automated data collection and how the work enables future opportunities in systems-level research for OWC networks.

1. Introduction

Optical Wireless Communications (OWC) is an active area of interest to the research community; however, a significant amount of system-level OWC research has focused on theoretical and/or simulated analysis with limited experimental validation. In order to address this concern and support experimental OWC test systems, we have previously introduced the ***gr-owc*** out-of-tree module in GNURadio (Ahmed & Rahaim, 2021; UCaN Lab, 2021), along with a detailed description of potential hardware configurations to integrate front-end OWC hardware with USRP SDR equipment (Ahmed et al., 2022). Our initial work also demonstrated the benefits of modular design in an SDR-based OWC test system where signal processing parameters can be configured for the various characteristics of different front-end OWC hardware. This prior work demonstrated point-to-point and multi-node OWC test systems, but flowgraph parameters were either fixed at the time of execution or required manual configuration at each node.

In this article, we introduce recent improvements that build upon our previous successes with ***gr-owc***. Namely, we implement dynamic control and coordination of distributed OWC nodes from a centralized testbed controller. This is achieved through a combination of GNURadio's XML-RPC and ZMQ modules to remotely modify parameters on running flowgraphs, along with bash scripts and the Secure Shell (SSH) protocol to remotely start/stop GNURadio flowgraphs at different nodes. This results in an OWC testbed architecture that is adaptable, versatile, and scalable.

To exemplify the testbed's capabilities, we have demonstrated the application of centrally controlled subcarrier allocation for a multi-cell and multi-user OWC system implementing DC-Biased Optical OFDMA. For the purpose of performance analysis and comparison, the implementation of a centralized testbed controller has allowed for improvements in automated data collection in GNURadio. In particular, we can now implement automated packet error rate analysis for multiple scenarios (e.g., subcarrier assignments) with multiple data points for each scenario – ultimately reducing the person hours required to collect quantitative data at scale. The baseline testbed design is configured as shown in Fig. 1, but offers flexibility to scale the number of transmitter and receiver nodes.

Proceedings of the 13^{th} GNU Radio Conference, Copyright 2023 by the author(s).

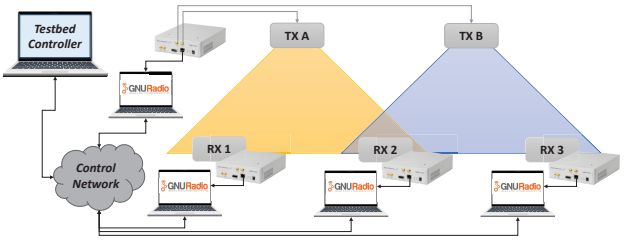

Figure 1. Testbed design for multi-cell and multi-user OWC deployments with centralized control over flowgraph parameters.

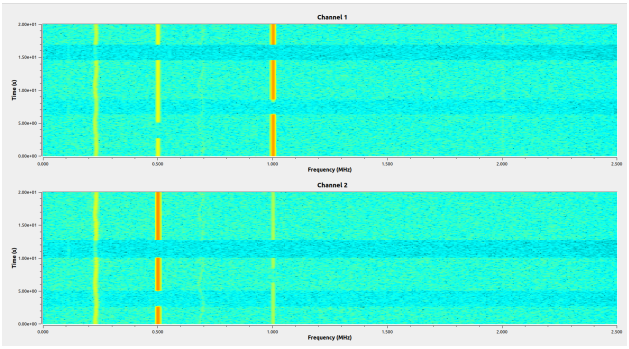

Figure 2. Waterfall plots demonstrating the impact of interference and occlusions in a scenario with two transmitters and two receivers. Both figures show the received spectrum at the two receivers when Tx 1 and Tx 2 are sending a 1MHz and 500KHz tone, respectively.

Beyond the benefits of automated data collection, this testbed architecture also enables many future investigations for dynamic parameter adaptation. In order to accommodate mobile devices with variable traffic load, practical multi-cell/multi-user OWC systems should dynamically allocate resources amongst OWC cells and users. Our testbed architecture enables future demonstration and analysis of different techniques for interference mitigation, rate adaptation based on user demand, or inter-cell handover for mobile users. Insights gained from this experimental analysis could improve network performance in practical multi-cell/multi-user OWC systems and offer experimental validation of systems-level research existing in the literature.

In summary, our open-source testbed supports the investigation of new resource allocation techniques and iterative improvements to the baseline OWC techniques that we provide within ***gr-owc***, fostering collaborative research and pushing the capabilities of future OWC systems. In this paper, we provide a detailed description of our testbed's implementation/deployment requirements and highlight the potential capabilities that our testbed can enable.

2. Background and Motivation

For multiple decades, researchers have been exploring the concept of indoor OWC for ultra-dense wireless networks (Kahn & Barry, 1997). The advancement of light emitting diode (LED) technology and signal processing capabilities has driven significant progress in point-to-point OWC link design, particularly in the area of visible light communications (VLC) (Elgala et al., 2011; Rahaim & Little, 2015). While OWC offers significant potential for improved link rates in point-to-point communications, the directionality of the optical medium provides even more potential for massive aggregate capacity in multi-cell and multi-user environments (Chi et al., 2020).

While much of the recent work in OWC networks is centered around such systems, the majority of this work is based in theory and/or simulation. Developing experimental test capabilities for these multi-node OWC systems will enable further analysis of the techniques and protocols in practical scenarios. Furthermore, a breadth of experimental observation can be endabled with joint control of system characteristics at both higher layers (e.g., resource allocation techniques) and lower layers (e.g., physical layer signal characteristics). Accordingly, we have used the foundational capabilities of GNURadio to enable multi-node signal processing for OWC systems. This prior work was used in various test scenarios for demonstrating data transmission over point-to-point OWC links (Rahaim et al., 2011) and heterogeneous RF/OWC implementations (Shao et al., 2014), and for observing signal-level characteristics of multi-node VLC systems (Little et al., 2018). An initial composition of the tools was ultimatel released as an open-source OOT library for GNURadio (Ahmed & Rahaim, 2021; Ahmed et al., 2022). The work presented in this article highlights the integration of the multi-node architecture with link implementation for real-time data transmission.

The primary motivation for systems-level OWC research is the need to accommodate highly dynamic scenarios (e.g., occlusions, mobile devices, time-varying traffic demand, etc.). In order to manage such scenarios, multi-cell OWC deployment must consider tradeoffs between overlapping coverage for reliability and isolation to maximize spatial reuse of resources. While overlapping cell coverage improves coverage, it also leads to potential interference scenarios. In Fig. 2, we show high level observations of OWC interference from the signal perspective. This motivational result is taken from a two transmitter / two receiver deployment of our testbed with receivers placed approximately 1.5m from the transmitters.

Tx1 and Tx 2 are transmitting tones at 1MHz and 500KHz, respectively, and the physical channel is occluded by blocking Rx 1, Rx 2, Tx 1, and Tx2 (in that order). Note that the 120KHz signal in both scenarios is present from other overhead lights in the room. At the top of the waterfall plots, we can see that Rx 1 and Rx 2 each receive a stronger signal from their corresponding transmitter, but the tone from the

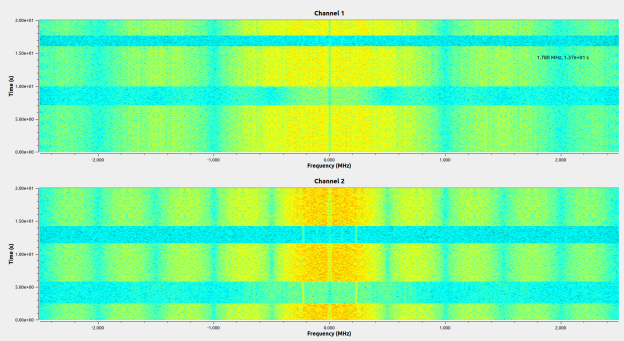

Figure 3. Recreation of scenario from Fig. 2 with 1MHz and 500KHz modulated data streams on Tx 1 and Tx 2, respectively

other transmitter is also present as interference. Following the time progression of the waterfall plot, we see that Rx 1 first loses all signal (when it is occluded) and then Rx 2 loses all signal (when it is occluded). Furthermore, the noise floor is also reduced as the DC optical power from the transmitters also leads to shot noise. Next, when Tx 1 is blocked we see a drop in the Rx 1 noise floor along with the loss of the 1MHz tone at both receivers, but the 500KHz Tx 2 tone is still present. Similarly, at Rx 2, we see the noise floor drop and the Tx 2 tone removed when Tx 2 is blocked. This process is repeated in Fig. 3, with the transmitters sending baseband modulated data waveforms at two different data rates.

While the depictions described above highlight a primary challenge of multi-cell OWC networks, the following work highlights the development of a test system to adapt to these dynamic environment scenarios. Namely, we introduce how to apply DC-Biased Optical OFDM (DCO-OFDM) to a multi-user / multi-access scenario where different subcarriers are assigned to different data streams and users. Furthermore, we highlight how this assignment can be centrally managed using GNURadios XMLRPC and ZMQ tools - enabling dynamic adaptation of the allocated resources and scripted scenarios to automate data collection for performance over a variety of scenarios.

3. Testbed Architecture

The high level architecture of our testbed is shown in Fig. 4. This is an extension of the architecture from (Ahmed et al., 2022) with the addition of a control network configuration that allows for central control over parameters in the Tx and Rx flowgraphs. As a baseline instance, we have three or more computers interconnected via Ethernet, one acting as the control station and the remaining computers running GNURadio and our OWC transmitter or receiver flowgraphs. Each GNURadio computer is connected to a Universal Software Radio Peripheral (USRP) with baseband transmission capabilities (i.e., LFTX and

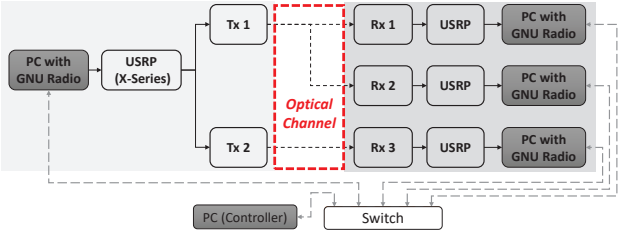

Figure 4. Multi-node OWC Testbed Architecture with central PC to control Tx and Rx parameters across the set of nodes.

LFRX daughterboards) for generation of the analog electrical signal. The OWC front end hardware (Tx and Rx blocks in Fig. 4) includes optical conversion equipment for signal transmission via intensity modulation with direct detection (IM/DD). At the control station, we've established robust control and coordination through a centralized testbed controller that interacts with distributed OWC nodes via Ethernet. GNURadio's XMLRPC and ZMQ modules, along with bash scripts, python scripts, and SSH, allows for remote control of flowgraphs parameters and design flexibility to configure future dynamic algorithms that respond to measured conditions in the network.

3.1. Network Architecture

The baseline design for our network architecture uses wired Ethernet connectivity to link the control PC with all of the Tx and Rx nodes. However, the control network configuration can be redesigned for wireless connectivity as desired. Currently, we use a `10.1.1.X` subnet with static IP address assignments for the controller (`10.1.1.1`) and all Tx/Rx nodes. We have also enabled passwordless SSH connections between the control PC and all remote nodes in order to allow for initiation of flowgraphs from the central controller if desired.

3.2. SDR Hardware and OWC Front-Ends

Hardware components in the testbed include (1) the USRP SDR hardware for interacting with GNURadio flowgraphs and generating the analog electrical signal, and (2) the OWC front-end transmitter and receiver equipment for intensity modulation (i.e., electrical to optical conversion) and direct detection (i.e., optical to electrical conversion). Much of this hardware is described in detail in (Ahmed et al., 2022), but we provide a brief overview of a baseline multi-node setup (Fig.5) here.

For SDR hardware in the OWC testbed, we utilize X310 and N210 USRPs. Since these USRPs allow for modular daughterboards, we can utilize the LFTX and LFRX cards to generate baseband signals directly from the USRPs. This is essential for OWC implementations since we utilize baseband electrical signals to drive the LED and re-

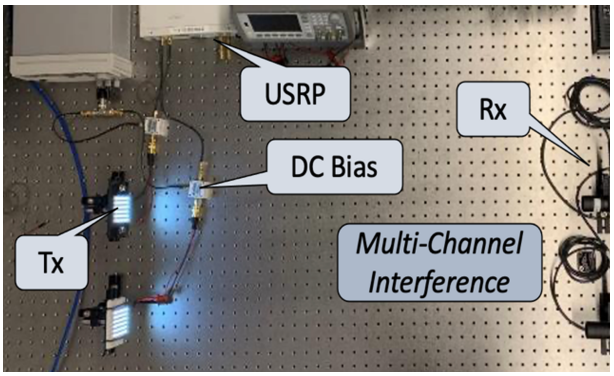

Figure 5. Front-end OWC hardware in a 2-Tx/2-Rx configuration

ceive baseband signals at the optical receiver. When implementing multi-node OWC systems, this also provides value in that a single RF chain can be used for two OWC chains. For example, at the transmitter the complex baseband signal sent from GNURadio to the USRP consists of two separate real valued signals (i.e., the real and imaginary components of the complex signal). These can be pulled out directly from the LFTX daughtercard such that two real valued signals can be generated in GNURadio and merged within a real to complex block prior to the UHD USRP Sink, allowing for transmission of the two signals on different OWC transmitters.

To implement the optical conversion for the OWC link, a variety of front-end equipment can be used with varying capabilities / price points. We detail the front-end OWC hardware design in (Ahmed et al., 2022) along with different example hardware that can be used as OWC transmitters and receivers. The main points to address for any hardware to be used with the system are as follows:

- **Ensure Linearity:** Given the voltage range output of the USRP, the electrical signal typically needs to be biased and amplified to fit within the linear range of the LEDs response curve.

- **Characterize Frequency Response:** Different hardware configurations (i.e., LEDs, photosensors, optical filters, etc.) will impact the system's frequency response. Understanding the response will allow for appropriate configuration of signal characteristics.

- **Anti-Aliasing and DC Blocking:** Given the ambient light conditions of most environments and DC component of any IM/DD OWC signal, it is helpful to block out the DC component of the electrical signal prior to sending this to the USRP. Furthermore, if a lower bandwidth signal is to be used it is important to apply appropriate analog filtering prior to sampling in order to avoid aliasing and excessive noise.

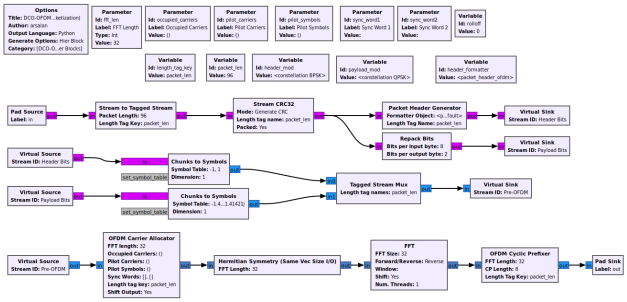

Figure 6. Hierarchical flowgraph using gr-owc Hermitian Symmetry block for generatio of a DCO-OFDM Waveform

3.3. GNURadio and Relevant Software Tools

GNURadio is the centerpiece of our testbed. It manages the underlying digital signal processing and the interaction with USRPs. Moreover, its integration with tools such as XML-RPC and ZMQ enables dynamic control and coordination from remote nodes. This interconnected design allows remote parameter modifications and intricate feedback loops, thereby driving adaptability, precision, and intelligence in the control of distributed OWC nodes.

The modular architecture of GNU Radio is one of its defining strengths. It supports an expansive selection of standard and custom signal processing blocks with extensive libraries and tools for community-driven development. This framework encourages the creation of custom blocks and out-of-tree (OOT) libraries, allowing researchers and developers to mold and adapt the platform to specific research needs. In particular, our ***gr-owc*** OOT module was developed within GNURadio's framework and is available through the comprehensive GNURadio Archive Network (CGRAN). In this section, we will introduce the ***gr-owc*** library, the XMLRPC and ZMQ tools along with their functionality in GNURadio, and additional software tools that we utilize at the testbed controller.

3.3.1. GR-OWC

The ***gr-owc*** module extends the GNU Radio framework specifically for OWC signal processing tools. This OOT library integrates seamlessly with GNU Radio, providing the essential tools required to design, simulate, implement, and test OWC systems. Among the custom blocks, ***gr-owc*** includes OWC channel models, an assortment of pulsed modulation schemes common to OWC, and necessary blocks to modify the core GNURadio OFDM signal chain for implementation of optical OFDM schemes. A core component used for the physcial layer implementation in our testbed is the DCO-OFDM hierarchical block, shown in Fig. 6. This block models the conventional OFDM transmitter with the addition of ***gr-owc***'s custom Hermitian Symmetry block after OFDM Carrier Allocation. This applies the complex

conjugate to the positive subcarrier values and places the updated values at the corresponding negative subcarriers in order to assure complex conjugate symmetry in the frequency domain which, accordingly, leads to a real valued time domain signal after taking the inverse FFT. Execution of this hierarchical block creates a new block in the GNURadio library capable of implementing DCO-OFDM. This block is used in the examples described in Section 4;

3.3.2. XML

In order to enable remote configuration of flowgraph parameters, we use the XMLRPC functionality supported within GNURadio. At a high level, this functionality is enabled by including an XMLRPC Server block within each GNURadio flowgraph, and specifying the IP address of the corresponding node within the testbed. This allows a remote client (e.g., the control PC) to connect with a desired server and define values for any active variable or parameter within the flowgraph. At the client node, we create a python script where the ServerProxy module is imported from the xmlrpc.client library, and a ServerProxy object is created to correspond with each of the XMLRPC Server addresses in the testbed. Within the script, we can then call the `<SP_Object>.set_<var>(<val>)` function to change the indicated variable (`var`) to the specified value (`val`) within the corresponding flowgraph.

3.3.3. ZMQ

The Zero MQ (ZMQ) library, and associated GNURadio ZMQ tools, provides the basis for remotely accessing measurements in the set of distributed nodes. We utilize the publish/subscribe mode such that the flowgraphs publish specified measurements and allow a remote subscriber (e.g., the control PC) to access the measurements as a subscriber. For implementation, the process requires each flowgraph to include a ZMQ PUB Sink block for each measurement of interest. Similar to the XMLRPC configuration, these blocks specify the IP address of the corresponding node within the testbed and utilize different ports for each desired measurement. At the control PC, the controlling python script uses the zmq library to create a socket for a ZMQ subscriber and specifies the IP address and port corresponding to the desired flowgraph and measurement.

3.3.4. Additional Software Tools

In the control PC, we have two levels of control. First off, we can utilize python scripts with the XMLRPC and ZMQ functionality, described above, to operate on running flowgraphs. Beyond this, we also look to initiate various flowgraphs at different nodes with (optionally) variable configuration settings. To implemetn this, we setup passwordless SSH connectivity between the control node and all GNURadio nodes in the testbed. With this, we can run python calls from the command line (or within bash scripts) to initiate the already generated flowgraphs at each node. When doing this, we note that the GUI options must be turned off and the options block within the flowgraph should be set for "No GUI". In addition, the flowgraphs can incorporate parameter blocks in place of variables to define values that can be configured with command line flags when executing the flowgraph via SSH.

4. Results and Analysis

As a demonstration of the testbed's remote control and automation capabilities, we will introduce a set of flowgraphs for performance analysis at the signal level and at the level of packetized data transmission. We first introduce the flowgraphs that can be run on the GNURadio compute nodes with manual control, then introduce the automation scripts that utilize XMLRPC and ZMQ for automated scenario configuration and data collection. Finally, we introduce the implementation of bash scripts to remotely initiate flowgraphs with a range of configuration parameters while also managing flowgraph parameters via python scripts.

4.1. Test Scenarios / Demo Flowgraphs

4.1.1. Multi-Node Tone and Signal Flowgraphs

The first set of flowgraphs, shown in Fig. 7, highlight the USRP configuration for transmitting and receiving multiple OWC signals from a single RF channel on the USRP. In the transmit flowgraph, we have two transmit signals that can be individually enabled or disabled, and configured as BPSK/OOK modulated waveforms or tones. At the receive, each channel can be configured for direct observation or observation after passing through a low pass filter. In both flowgraphs, XMLRPC server blocks are included to enable remote control over parameters if desired. Furthermore, the receiver flowgraph is enabled with two ZMQ Pub Sink blocks to observe the received power at each receiver.

4.1.2. Multi-User OWC with DCO-OFDMA

The second set of flowgraphs, shown in Fig. 8, demonstrates the adaptation of the the previous flowgraphs for data transmission via DCO-OFDM. In the transmit link, we include three separate data inputs (i.e., audio, pre-defined data pattern, or data file) that can be selected by enabling the desire blocks. In the signal path, we utilize three instances of the previously defined hierarchical block for DCO-OFDM - each applying a different subcarrier assignment (i.e., low subcarriers, high subcarriers, and a mixed option). The second image is a subset of the modified flowgraph showing the same procedure for signal path with the addition of a second signal path having pre-defined subcar-

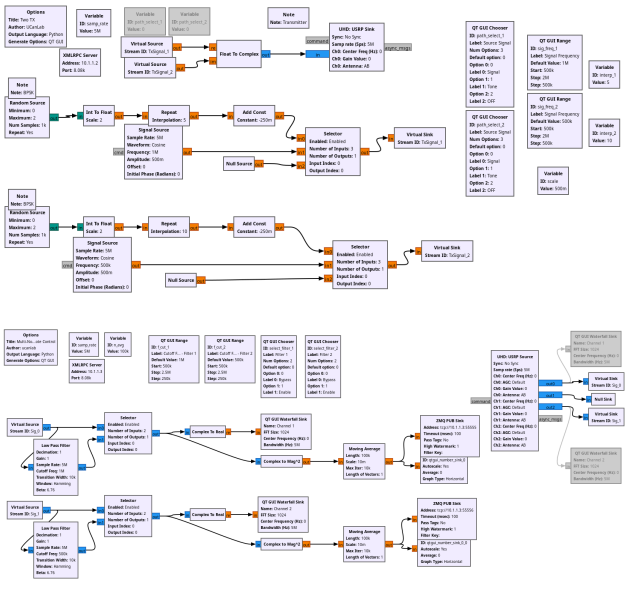

Figure 7. Two-channel basic transmitter (*top*) and receiver (*bottom*) with remote configuration via XMLRPC and access to power measurements via ZMQ.

rier allocation. The third image depicts the receive flowgraph with the 3 OFDM receivers corresponding to the specied subcarrier options within the three DCO-OFDM blocks. Note that this works with the conventional OFDM receiver implementation as long as the subcarrier selection only observes the positive (or negative) allocated carriers in isolation. When running these flowgraphs, the user can manually configure the primary signal subcarriers at the transmitter and receiver to observe the importance of synchronization in carrier selection. Furthermore, interference can be observed by enabling/disabling the second transmit channel and observing the limitations when the two channels utilize the same subcarrier selection (depending on the environment setup a relative location of the two transmitters and receiver) and related performance improvement when the primary and secondary transmissions are set to different subcarrier selections. Note that the receiver flowgraph can be run on multiple Rx nodes with different carrier allocation at each node.

4.2. Python Scripts for Automation

The flowgraphs described above can each be run with manual configuration via the GUI interface. However, the additional functionality for central control adds capabilities to automate data collection over a breadth of scenarios and enables future options for implementing adaptive algorithms for resource allocation and link reconfiguration.

In the baseline automation script, the core functionality is structured around two nested loops that cycle through various experimental settings to modify the configuration and collect a set of power measurements. By placing the desired settings in arrays, the script offers the flexibility to experiment with a wide range of configurations without altering the underlying logic. The script uses the arrays to define a set of flowgraph configurations. For each configuration, the script modifies the remote flowgraph parameters via XMLRPC to remote servers (i.e., STA2_control and STA3_control). The script then enters an inner loop where it records multiple power measurements acquired via ZMQ sockets. The measurements are processed to compute the average power, which is then printed to the console for real-time monitoring, providing a comprehensive sweep across various configurations.

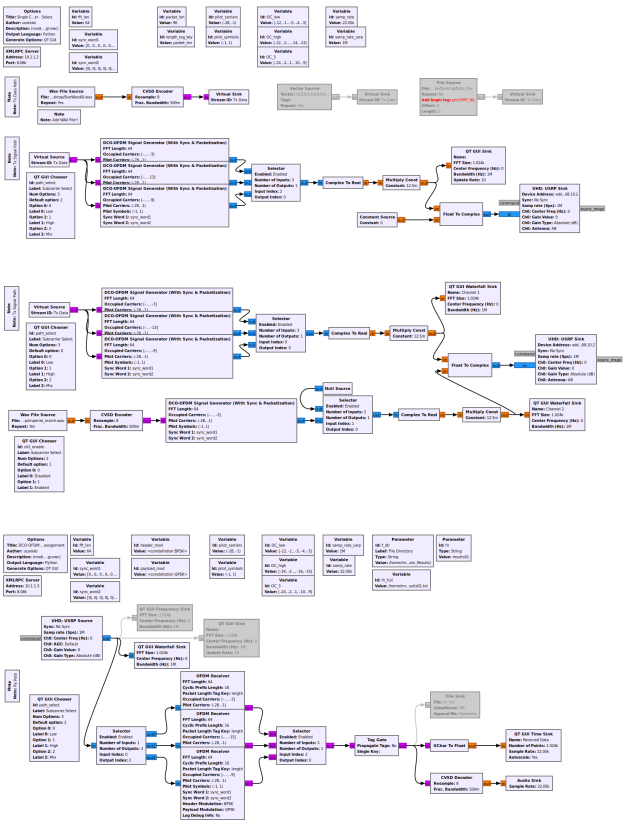

Figure 8. DCO-OFDMA Flowgraphs highlighting a single transmitter design with user-specified subcarrier subset (*top*), an adaptation of the first design to add a second transmission link with option to enable or disable the transmission (*middle*), and a single receiver design with subcarrier subset selection (*bottom*).

4.3. PER Testing with Central Control

The final demonstration highlights automated data collection for observing packet error rate (PER) in different interference settings with various subcarrier assignments. The premise of the PER test is the transmission of a large multi-

line text file where each line consists of the same number of characters as the bytes specified in the OFDM packetization process. After transmission and storing the results to a file at the Rx node, the file can be observed to compare the total number of lines in the received file with the lines in the transmitted file. While this can be run manually, it becomes incredibly time consuming when reconfiguring the receiver file names and restarting flowgraphs.

To simplify the process, we created a bash script that centrally initiates the receiving DCO-OFDM GNURadio flowgraph on a computer, and then starts the transmit DCO-OFDM GNURadio python script. When the transmit flowgraph is set to run with "No GUI" and configured to run to completion, the bash script can repeatedly initiate the flowgraph via SSH, wait until it completes, pause, and then repeat the process so that multiple instances of the file are transmitted. Additionally, the subcarrier subset is defined as a parameter in the flowgraph so that the Tx subcarrier allocation can be included in command line call each time the flowgraph is initiated. At the receiver, the flowgraph can not be set to run to completion since the USRP source is continuously sourcing samples (even once the file has been transferred). To address this issue, we set the file name as a variable/parameter on the receiving flowgraph so that we can dynamically change this value from the command-line interface via XMLRPC. At the completion of the bash script, the set of received files can be observed at the Rx node to determine PER for each defined scenario.

5. Summary and Conclusions

We have introduced a multi-node OWC Testbed architecture that employs distributed GNURadio nodes and a central controller with the ability for flowgraph parameter updates using XMLRPC and for data retrieval via ZMQ. We have introduced these improvements as flowgraph examples within the ***gr-owc*** OOT module, including examples for dynamic control of distributed OWC nodes from a central controller. We have also demonstrated centralized control for multi-cell, multi-user OWC systems, implementing DC-Biased Optical OFDMA, and described our process for automated data collection of PER analysis across various scenarios, significantly reducing data collection time.

The testbed architecture enables future investigations into dynamic parameter adaptation for mobile devices and resource allocation across OWC cells and users. It facilitates exploration of interference mitigation, rate adaptation, and inter-cell handover, potentially enhancing network performance in practical OWC systems and providing experimental validation. The open-source testbed fosters collaborative research in OWC, supporting new resource allocation techniques and improvements to existing OWC methods.

Acknowledgements

The authors thank the Air Force Research Laboratory and Wright Brother's Institute for supporting this project during the 2022/23 Beyond 5G SDR University Challenge.

References

Ahmed, Arsalan and Rahaim, Michael B. gr-owc: An open source gnuradio-based toolkit for optical wireless communications. In *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 48–53, 2021. doi: 10.1109/WiMob52687.2021.9606271.

Ahmed, Arsalan, Dzhezyan, Gregory, Aboutahoun, Hussam, Chu, Victor, DiViccaro, Jordan, Ohanian, Vahae, Rezaeiboroujerdi, Sharareh, Saheb, Ilyas, Urban, Evan, Wu, James, et al. SDR beyond radio: An OOT GNU Radio library for simulation and deployment of multi-cell/multi-user optical wireless communication systems. In *Proc. of the GNU Radio Conference*, volume 7, 2022.

Chi, Nan, Zhou, Yingjun, Wei, Yiran, and Hu, Fangchen. Visible light communication in 6g: Advances, challenges, and prospects. *IEEE Vehicular Technology Magazine*, 15(4):93–102, 2020. doi: 10.1109/MVT.2020.3017153.

Elgala, Hany, Mesleh, Raed, and Haas, Harald. Indoor optical wireless communication: potential and state-of-the-art. *IEEE Communications magazine*, 49(9):56–62, 2011.

Kahn, J.M. and Barry, J.R. Wireless infrared communications. *Proceedings of the IEEE*, 85(2):265–298, 1997. doi: 10.1109/5.554222.

Little, Thomas, Rahaim, Michael, Abdalla, Iman, Lam, Emily, Mcallister, Richard, and Vegni, Anna Maria. A multi-cell lighting testbed for vlc and vlp. In *2018 Global LIFI Congress (GLC)*, pp. 1–6, 2018. doi: 10.23919/GLC.2018.8319111.

Rahaim, M, Miravakili, A, Borogovac, T, Little, T, and Joyner, V. Demonstration of a software defined visible light communication system. In *Proc. 17th Annu. Int. Conf. Mobicom*, pp. 1–4, 2011.

Rahaim, M. B. and Little, T.D.C. Toward practical integration of dual-use vlc within 5g networks. *IEEE Wireless Communications*, 22(4):97–103, 2015. doi: 10.1109/MWC.2015.7224733.

Shao, Sihua, Khreishah, Abdallah, Rahaim, Michael B., Elgala, Hany, Ayyash, Moussa, Little, Thomas D.C., and Wu, Jie. An indoor hybrid wifi-vlc internet access system. In *2014 IEEE 11th International Conference on*

Mobile Ad Hoc and Sensor Systems, pp. 569–574, 2014. doi: 10.1109/MASS.2014.76.

UCaN Lab, Ubiquitous Communications and Networking Lab. UCaNLabUMB/gr-owc: Initial release. Github, www.github.com/UCaNLabUMB/gr-owc, 2021. URL https://zenodo.org/record/5237300.