Ethernet Fronthauling for Physical Layer Functional Splits in C-RAN: A Choice Between TCP and UDP

Venu Balaji Vinnakota, Naganithin Manne, Abhijit Mondal, Debarati Sen and
Sandip chakraborty
Indian Institute of Technology Kharagpur, Kharagpur, India 721302
Email: venuvinnakota@gmail.com, nithinmanne@gmail.com, abhimondal@iitkgp.ac.in, debarati@gssst.iitkgp.ac.in,
sandipc@cse.iitkgp.ac.in

***Abstract*—Cloud Radio Access Networks (C-RAN) have been proposed as a new paradigm shift in the *Radio Access Network* (RAN) technology as a part of the fifth generation *Long-Term Evolution Advanced* (LTE-A) networks to support better spectral and energy efficiency along with the high availability. In this paper, we discuss implementation details of a C-RAN setup with the help of USRP-based transceivers and LabView platform. To the best of our knowledge, this is the first implementation of a C-RAN architecture that functionally splits the radio resource head (RRH) and the baseband processing unit (BBU) at the physical layer and transfers the completely unprocessed raw signal elements from the RRH to the BBU pool at the cloud for signal processing and data extraction. We explore TCP and UDP as alternate protocols for fronthaul data transfer to the cloud. In order to evaluate the performance of a C-RAN fronthaul and the interplay of different performance parameters for fronthaul data transfer, we observe various metrics like the receiver goodput and the latency, and compare the performance between a C-RAN setup and a generic distributed RAN setup. We observe that TCP works better for Ethernet fronthauling compared to UDP, as it provides reliable data delivery. The analysis discussed in this paper gives insights about the implementation and performance of a C-RAN environment which is essential for designing efficient fronthauling and functional splits of a C-RAN architecture.**

***Index Terms*—Cloud RAN; 5G; Latency; Fronthaul; Delay-Goodput Trade-off**

I. Introduction

Current Radio Access Network (RAN) technologies follow a distributed architecture that faces a lot of challenges due to the huge demands of mobile data from the subscribers. In order to meet that demand, network capacity should be expanded by erecting a number of base stations (BS). This increases the *Capital Expenditure* (CAPEX) and *Operating Expenditure* (OPEX) for mobile operators [1]. Originally, heterogeneous cellular networks were proposed to increase the network capacity; but interference mitigation has become a huge challenge due to the unavailability of sufficient spectrum and frequency reuse.

Cloud based Radio Access Networks (C-RAN) evolve as a promising technology to answer the challenges raised by the existing RAN technologies. In LTE, BS is also termed as *evolved NodeB* (eNodeB), which has both *Radio Frequency* (RF) unit and *Baseband Unit* (BBU) with it. The RF unit takes care of the transceiver mechanism on the Remote radio head (RRH), whereas the BBU is responsible for handling signal processing and upper layers of the protocol stack. As a part of the architectural change in C-RAN, BBU is separated from RRH and kept at a distant location. Many such BBUs, which are separated from different eNodeBs are placed as a pool in a centralized location, called BBU pool. This functional splitting helps in addressing the problems discussed above with the help of virtualization of the BBU pool. First, energy efficiency is achieved as the RF modules become simple, and a single cloud server can host the entire BBU pool. Second, spectral efficiency can be achieved with the help of *Coordinated Multi-point Transmission* (CoMP) where the neighboring eNodeBs coordinate with each other to serve a user equipment (UE) [2]. Third, the hosting of BBU pools over a cloud provides high availability and fault tolerance.

However, there are various challenges associated with C-RAN implementation. First, a decision of logical split between the RRH and the BBU is an important aspect [3]. A functional split just after receiving the signal at the RRH is the best choice for energy efficiency, which makes the RRH very simple. However, a functional split at this stage is difficult because the entire raw signal needs to be transmitted to the cloud in the form of amplitude and phase data, known as I/Q data. As the size of the I/Q data is very large, the fronthaul that interconnects the RF with the cloud needs to be of high capacity. Although this is well known theoretically, and many fronthaul compression techniques have been proposed in the literature [4], [5], an experimental characterization is required to understand the relation between fronthaul capacity, protocols to transfer I/Q data through the fronthaul and the C-RAN performance. However, the existing implementations of C-RAN [6]–[9] have primarily focused on the different functional splitting of the RAN architecture with generic fronthaul protocols (like raw data transfer or UDP-based transfer), whereas have not analyzed the impact of the tunable parameters, like sampling rate, frequency of processing I/Q data, etc., on the performance of C-RAN in comparison with the distributed RAN architecture. Second, the protocol for fronthaul data transfer also plays a crucial role in the C-RAN performance. Existing works, like FlexRAN [9] has looked into the C-RAN performance when the I/Q data is transferred to the cloud using raw or UDP protocol. However, this incurs data loss during the transmission. A loss of I/Q data affects the signal processing significantly, and incurs a huge bit error rate [10]. TCP and UDP can work as good alternatives for

Ethernet-based fronthauling in C-RAN; however, the existing studies have not explored the benefits and drawbacks of TCP and UDP-based Ethernet fronthauling for C-RAN.

In this paper, we discuss the implementation and performance analysis of a centralized C-RAN architecture, where the RF has the minimal functionality and the entire signal processing is done at the BBU hosted over the cloud. To the best of our knowledge, this is the first work that practically implements the RRH-BBU functional split at the physical layer; the I/Q samples are captured at the RRH, and the entire unprocessed I/Q samples are forwarded to the BBU at the cloud for signal processing and data extraction. We have used USRP 2943-R for implementing the transceiver modules and a high performance workstation is used to host the cloud. We have used Ethernet fronthauling and two alternatives of transport protocols – TCP and UDP, to transfer the I/Q data from the RRH to the BBU at the cloud. Over this architecture, we measure the receiver goodput and transmission delay under various system parameters, such as sampling rate, acquisition duration (periodicity of I/Q data processing at the cloud) and so on. Our analysis shows that a C-RAN system achieves almost equal receiver goodput as of a distributed RAN setup with TCP-based Ethernet fronthauling; however, there is a significant drop in latency performance. We further look inside different latency components in this system, and observe that the latency introduced due to TCP is nominal, whereas the latency for processing the I/Q data sample dominates the overall latency. We also look into the loss characteristics of the fronthaul by comparing the performance between TCP and UDP as the transport protocols for I/Q data transfer from the RRH to the BBU pool at the cloud. This analysis gives three important insights of the system. First, TCP is a better alternate than UDP for Ethernet fronthauling in a C-RAN-based network to achieve high receiver goodput. Second, the fronthaul data is the major bottleneck for the C-RAN performance. Third, acquisition duration is another important parameter that impacts the performance of the C-RAN in terms of receiver goodput and transmission latency. The key contributions of the paper are the following, (a) we are implementing functional split at PHY - this is a major contribution, existing works have mostly focused on MAC or higher layer splits; (b) because of the PHY split, we have to deal with huge volume of data at the fronthaul - so the choice of transport protocols at the transport layer is important; (c) we do a critical analysis between TCP and UDP as the fronthaul transport protocol when functional split is done at the PHY layer.

II. Related Works

Different C-RAN architectures are proposed by the industry and the academia; out of them, *Fully centralized* and *Partially centralized* C-RAN architectures are popular ones. Fully centralized C-RAN architecture is helpful in increasing the network capacity by increasing the density of RRH, whereas huge fronthaul capacity is required to transfer the I/Q samples. In Partially centralized C-RAN architecture, less capacity fronthaul link connection is sufficient between the RRH and the BBU pool; but the network expansion is not so easy due to the increase in complexity of the RRH.

There are many platforms available to support the C-RAN setup; out of them OAI (OpenAirInterface) is the popular one [11]–[13]. OAI is originally developed by EURECOM and developed as open source by the OpenAirInterface Software Alliance. OAI is the real-time software written in C programming language and can be supported on Linux platform. 3-GPP protocol stack for Evolved Universal Terrestrial Radio Access (E-UTRAN) and Evolved Packet Core (EPC) is completely implemented in OAI.

In [14], *Next Generation Fronthaul Interface* (NGFI) -based C-RAN architecture is emulated with the help of OAI framework, commodity hardware and Ethernet fronthauling. Since *Common Public Radio Interface* (CPRI) -based C-RANs are not able to meet the scalability and performance requirements, NGFI redefines the fronthaul transport network architecture through Baseband splitting between BBU and RRH.

The performance of the Time-domain I/Q split (IF5) and frequency-domain I/Q split (IF4) are analyzed. Same work is explained in detail by the same group of authors in [9]. Both the functional splits are compared in terms of different performance metrics like fronthaul throughput, round trip delay between fronthaul and RF circuits, the CPU utilization because of BBU/RRU hardware load, data plane delay and data plane QoS.

Similar kind of work with OAI software supported Ethernet fronthaul-based C-RAN testbed is studied in [8] with the MAC layer split. RRH is kept with RF and Physical (PHY) layer processing functionalities. The authors observed that unlike CPRI, Ethernet fronthaul can not be used for transportation of high data rate I/Q sample data.

The PDCP/RLC and MAC/PHY split performance in the downlink is studied in [15] using the real-time heterogeneous testbed and OAI emulation platform. The communication between BBU and RRH is carried over 1 Gbps copper link. The throughput achieved in both the cases is studied under different modulation and coding scheme (MCS) and transport block sizes (TBS). The PDCP/RLC split is studied using stateless protocol UDP and state-full protocols TCP and SCTP. UDP achieves more throughput than the other state-full protocols TCP and SCTP.

However, the MAC/PHY layer split is studied using UDP protocol only. In [11], without using any RF card, virtualized C-RAN experiment setup has been demonstrated. The basic commands required to setup the system are discussed. In [16], the computation requirements such as CPU utilization percentage, processing time of LTE subframes and delay performance are measured for OAI-based C-RAN setup. From the observed results, a conclusion is drawn that BBU processing time is function of CPU frequency, number of physical resource blocks (PRBs) and Modulation and coding scheme (MCS) index. CPU utilization increases linearly with increment in downlink throughput.

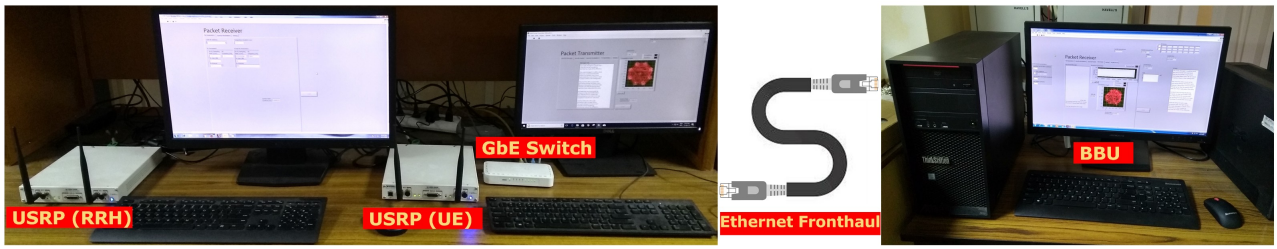

Figure 1. Snapshot of our hardware setup

III. C-RAN Implementation – Platform Details

We have realized C-RAN setup and distributed RAN setup with the help of a workstation and the USRP (Universal Software Radio Peripheral). As mentioned earlier, we have implemented the functional split between the RRH and the BBU at the physical layer, and use the fronthaul to transfer the completely unprocessed I/Q samples to the BBU for signal processing and data extraction. Fig. 1 shows the snapshot of the C-RAN setup developed at our laboratory. A software enabled workstation and USRP combination acts as a complete communication node (eNodeB), but depending upon our requirement we made it enact as either RRH or UE. Since we are focused on uplink analysis, the transmission workstation and USRP combination act as UE. In distributed RAN setup (Fig. 2), the receiving workstation and USRP combination act as LTE eNodeB. In C-RAN setup (Fig. 2), the first stage of receiving workstation and USRP combination act as RRH. Since Ethernet fronthaul is used to transfer the I/Q samples between RRH and BBU, no USRP is required at the BBU. A software enabled workstation alone can act as BBU. The details of the individual components used in the setup are discussed as follows.

A. System Configuration

In our setup, BBU and RRH are hosted at Lenovo Thinkstation P410 workstation with Intel Xeon CPU E5-1630 V4, 4 core, 8 threads at 3.7 GHz and 64 GB RAM. UE is hosted by Dell Precision Tower 3620 workstation with Intel Core i5-6500 CPU, 4 cores, 4 threads at 3.2 GHz and 4 GB RAM. We have used 1Gbps Ethernet for the fronthaul link, to connect the RRH and the cloud hosting the BBU.

B. Hardware Setup

The detailed hardware setup of the developed system is as follows.

1) TX/RX Modules: We have considered two National Instruments's USRPs 2943R types for transmission and reception. USRP 2943-R provides an integrated Labview software and hardware platform for wireless communication. Each USRP is an SDR (Software Defined Radio) platform developed over FPGA technology. As a part of the FPGA, it has one mother-board and two daughter-boards. Each daughter-board can act as either transmitter or receiver. USRP 2943R can operate from 1.2GHz – 6GHz with a bandwidth of 40MHz. Each USRP is having two ports for antenna connections. VERT 2450 antenna is used for USRP communication, which produces omni-directional radiation pattern and operates on dual band, 2.4–2.5GHz and 4.9–5.9 GHz, respectively, at 3dBi gain. PCIe 3.0 x4 cable has been used for connecting between the USRP RIO and the workstations hosting RRH and UE.

2) Cloud setup: When the USRP acts as a transmitter, it takes I/Q samples as input and transmits the RF waveform. While acting as a receiver, USRP gives the I/Q samples as output from the received waveform. In the distributed RAN's setup, the Labview program running on the receiver workstation does the whole signal processing to get the message back from the received I/Q samples. On the other hand, in C-RAN's setup, the Labview program running on the intermediate receiver (RRH) workstation transmits the received I/Q samples to the cloud server over the fronthaul link. The Labview program that implements the BBU running on the cloud workstation processes the signal to get back the original message from the received I/Q samples. The signal processing is discussed in details in system design section.

3) Fronthaul setup: Since Ethernet fronthaul is emerging as an alternative to the CPRI, we have carried the communication between RRH USRP and cloud computer hosting the BBU with the help of a 1Gbps Ethernet link. The fronthaul uses CAT 6 cable 030115 0222M DIGILINK Ethernet cable that supports up to 10Gbps, however, the cloud is connected via a NetGear Ethernet switch GS608, which provides 1Gbps switching speed, making it an effective 1Gbps fronthaul link.

4) Protocol choice for fronthaul: As mentioned, we have used Ethernet fronthauling for transfer I/Q data from the RRH to the cloud hosting BBU. Both UDP and TCP are suitable transport layer end-to-end protocol for communication over Ethernet. The existing studies [6]–[9] have explored either raw transmission (direct transfer without any transport protocol) or UDP-based transmission for data transfer over the fronthaul. Although UDP is good for latency-sensitive transmission as the overhead is, there are a number of issues with UDP. (a) UDP is an unreliable protocol; thus a loss of communication buffer is not handled by the end-to-end transmission protocol. This may result in I/Q data loss, which can be significant when the cloud is connected with the RRH via network switches. The I/Q data loss may have a significant impact on the packet error rate, and reduce the performance of the system. (b) UDP buffer management is not very robust and works like a simple store and forward buffer. This incurs out-of-order data

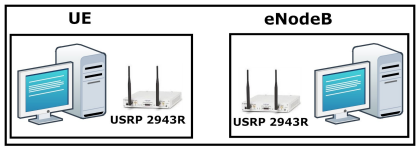

Figure 2. Distributed RAN setup

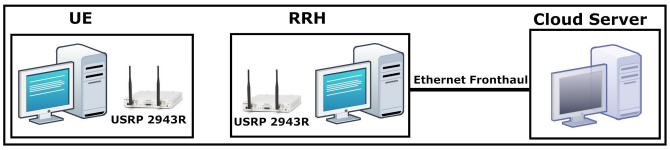

Figure 3. C-RAN setup

delivery, which may affect I/Q data processing significantly. As a consequence, in this work, we use TCP-based end-to-end data transfer for Ethernet fronthauling. TCP supports reliable, in-order data delivery; however, with the overhead of increased latency.

C. Software Setup:

USRP 2943R is LabView[1] dependent. LabView is a data flow model-based G programming language (a graphical language), and it provides all basic data types for variables.

The G-type code is converted to machine language and gets dumped over USRP's FPGA board. Every LabView program is called a *virtual instrument* (VI). Each VI is represented in two ways, *Front-panel* and *Block diagram*. The Front-panel window shows the user interface which shows buttons like controls, indicators etc.. Block diagram includes constants, structures, terminals, subVIs, functions and wires which transfer data among other block diagram objects. The VIs provide a *modulation toolkit* (MT) for implementing standard modulation techniques and platform, which has been utilized in this implementation.

IV. C-RAN System Design: Functional Modules

Based on the basic architectural setup as shown in Fig. 2 for distributed RAN and Fig. 3 for C-RAN, we developed various functional modules for a distributed RAN setup as well as C-RAN setup, as shown in Fig. 4 and Fig. 5. In Fig. 4, the block diagram of distributed RAN is shown, where UE does packetization of the input message and M-ary PSK modulation over the packetized data. After reception of the I/Q samples, the eNodeB executes noise estimation and chopping, demodulation, validation of the packet and reconstruction of the validated data. In Fig. 5, the block diagram of C-RAN is shown, where the functionalities of UE remain same. The receiver is modified as two stage receiver. RRH is the intermediate stage of the receiver, which extracts the I/Q samples of the received RF waveform, and those I/Q samples are transmitted to BBU using TCP/UDP sender. The BBU is the second stage of receiver, which receives I/Q samples through the TCP/UDP receiver. Now, all the remaining signal processing functionalities are associated with the BBU to recover the transmitted message.

A. Packet Structure

The Packet structure has been organized as shown in Fig. 6. The input data is divided into blocks of 640 bits. Each block is made into a packet by adding 30 Guard bits, 20 Sync bits, 32 bit Packet number (I32 data type) and 640 Message bits respectively. The sync bits are *pseudo noise* (PN) sequence generated bits. A maximum length sequence of length 31 is used in the generation of PN sequence. The Packet pad sample gets added at the end of the packet. Blank frames are used for transmission, when there is no data to transmit. Cyclic pad data are the initial number of message bits that get added at the end of the 640 message bits. Suppose we have set the number of samples for symbol (n) as 8. Then the cyclic pad data are generated according to the formula:

$$\left(\frac{160}{n}\right) \times \log_2(n) = \left(\frac{160}{8}\right) \times \log_2(8) = 60 \text{ bits}$$

B. Modulator

A generic M-ary *phase shift keying* (PSK) modulator has been implemented, where we can choose our required modulation scheme for communication. The input message has been constructed into packets and M-ary PSK modulated packet transmission has been used for communication between the transmitter and the receiver. The filter coefficients are passed as inputs to the *MT generate Filter coefficients* VI in LabView to generate pulse shaping filter coefficients. These input filter coefficients, input bitstream, system parameters and symbol rate are passed as inputs to the LabView provided *MT Modulate PSK* VI, which generates the PSK modulated I/Q samples. An unbounded queue is used to receive the I/Q samples from the receiver USRP.

C. Fronthaul Data Transfer Protocol

As mentioned, we use TCP/UDP for fronthaul data transfer. As of now, only uplink communication has been implemented in our system; hence, the TCP sender transmits the I/Q samples to the cloud server. In our experiment, RRH does the TCP transmission of I/Q samples to the BBU. The TCP/UDP sender and the TCP/UDP receiver are implemented using LabView provided data communication library. There are VIs provided to perform *TCP/UDP Read* and *TCP/UDP Write* operations.

1) TCP: The I/Q samples are encoded into a stream of bytes. These streams of bytes are divided into fixed size blocks of messages, and these blocks of messages are transferred from the RRH to the BBU through the fronthaul. The RRH sends the length of the message, followed by the message, separated by a carriage return and line feed (CRLF) using *TCP Write*. The BBU uses *TCP Listen* VI to initially wait for transmission. It reads till it encounters a CRLF. It then gets the length of the data to be received, and then receives the corresponding number of bytes using *TCP Read*.

[1]http://www.ni.com/en-in/shop/labview.html (Last accessed: April 2018)

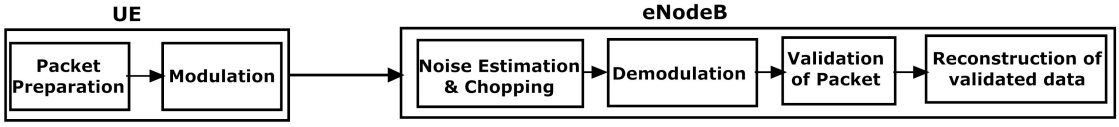

Figure 4. Distributed RAN Internal block diagram

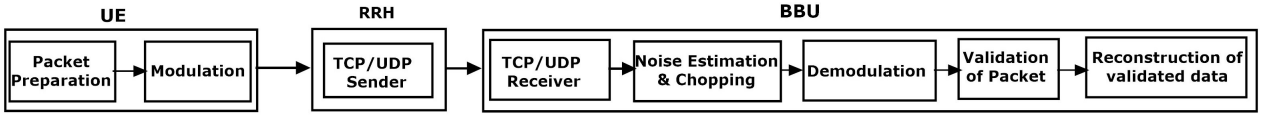

Figure 5. Cloud RAN Internal block diagram

Figure 6. Packet Structure used for communication in the Hardware setup

2) UDP: Along with TCP, UDP is also used as an alternate protocol to transfer I/Q samples between RRH and BBU. Since in UDP-based communication, I/Q samples cannot be streamed like that of in TCP-based communication, we have packetized the I/Q samples before transmitting them over the fronthaul. The I/Q samples in the packet form have been received at the BBU.

D. Noise Estimation, Chopping and Demodulator

The received I/Q samples are generally noisy in nature. The addition of noise over M-ary PSK symbols results in the increase of bit error rate (BER). Hence a noise estimator is designed to estimate the amount of noise added. By keeping a threshold value, the amount of noise added is chopped and the packet pad samples are removed from the remaining I/Q samples.

Based on the input parameters, we use *MT Resample* VI to sample the noise chopped modulated data, which is followed by *MT Demodulate PSK* VI to recover the bit stream.

E. Packet Validation and Reconstruction of Validated Data

The validation of packet is done by comparing the sync bits with that of the transmitted sync bit pattern. The packet is dropped even if one of the sync bit mismatches. Finally, the message bits are extracted from the valid packets by dropping the remaining bits.

V. Experimental Details and Results

The main objective of our experiment is to calculate the goodput and latency in both distributed RAN and C-RAN setups under various system configuration parameters. For the experiment over this system, we have generated data sequence through an application which continuously transfer text messages from the UE to the BBU in upload direction. Every experiment is repeated for multiple (at least 5 times) times, and both the average and the standard deviation are used to plot the results.

A. Experimental Methodology

In our experimental setup, the various system configuration parameters such as operating frequency, antenna gain, sampling rate, samples per symbol for both the transmitter (UE) and the receiver (RRH) are same. The whole experiment was operated at 2.45GHz with a bandwidth of 40MHz. The antennas connected to the USRP are operated at a gain of 20dB. Both the USRPs are separated by a distance of 6 ft. The text message is entered as input from the transmitter side, and the entered text message is encoded into ASCII. These message bits are grouped in the form of packets as shown in Fig. 6.

In order to perform pulse shaping, filter parameters are entered as an input at the transmitter side. We choose *Root Raised Cosine filter* as our filter with a roll-off factor of 0.5 and filter length 6. 8-PSK modulation is applied over packetized data at the transmitter. The transmission rate varies with varying the input sampling rate of the USRP hardware. Since we have to calculate the receiver goodput at different transmission rates, we varied input sampling rates from 5 Mega-samples per sec (MSPS) to 35 MSPS at a gap of 5 MSPS. The generated I/Q samples are needed to be grouped in the form of symbols. We have chosen 8 samples at the transmitter to represent one symbol. The same number of samples per symbol is kept in the receiver.

On the receiver side, acquisition duration is used as a configurable parameter. The receiver fetches I/Q samples for every acquisition duration window and process them as a batch. The larger the acquisition duration window, the larger will be the I/Q sample size received in one processing batch.

B. Performance Metrics

We observe two metrics as the primary performance measurement from the developed system – (a) *Receiver Goodput* and (b) *Uplink Transmission Latency*. The Goodput is computed as the effective rate of data reception at the RRH avoiding the signaling overhead (header bits, TCP overhead etc.). The uplink transmission latency has multiple components. (a) *First I/Q Sample Reception Time:* This includes the wireless delay between the two USRPs and the time to receive the first I/Q sample from the RF waveform. (b) *TCP Delay:* This includes the average delay for fronthaul data transmission. (c)

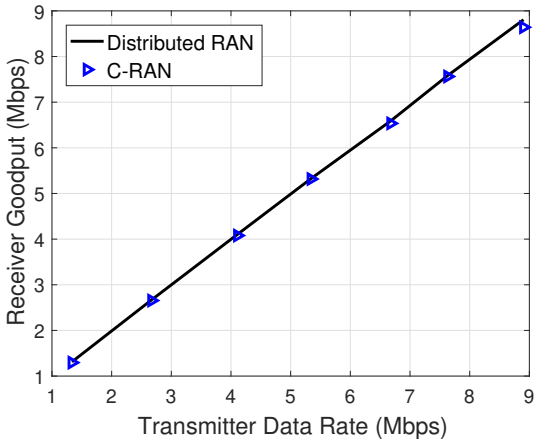

Figure 7. Comparison of transmitted data rate and receiver goodput at a constant acquisition duration of 40 ms in the case of TCP communication

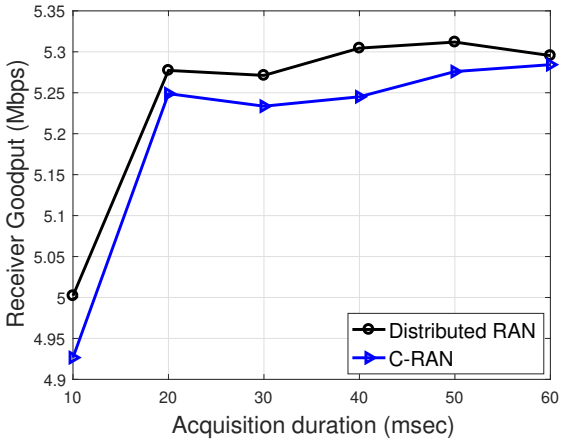

Figure 8. comparison of different RRH receiver acquisition durations and receiver goodput at a constant transmission rate of 20 MSPS in the case of TCP communication

I/Q Sample Processing Delay: This delay component corresponds to the processing delay of I/Q data. The performance metrics for UDP-based ethernet fronthauling are discussed separately under packetdrop bottleneck subsection.

C. Goodput Analysis

The experiment has been carried with 8-PSK modulation of data. Since in 8-PSK, 3 bits have translated into 1 symbol; 640 bits are mapped into $\frac{640}{\log_2(8)}$ Symbols. Further 8 Samples are mapped into one symbol; so the entire payload will give $\frac{640\times8}{log_2(8)}$ number of samples. The total preamble of $(30+20+32+60) = 142$ bits translate into $\frac{142\times8}{\log_2(8)}$ number of samples. The packet pad of 200 samples is also added at the end of the whole packet. Hence a total of $\frac{640\times8}{\log_2(8)} + \frac{142\times8}{\log_2(8)} + 200 \approx 2285$ samples are needed to transmit 640 message bits. Based on the I/Q sampling rate, we can compute the effective transmission rate in Kbps. For example, if the I/Q sampling rate is set as 5 MSPS, effective transmission rate would become approximately 1367 Kbps.

Based on the relation between the transmission rate in MSPS and the same in Kbps, the receiver goodput is compared with distributed RAN and C-RAN, as shown in Fig. V-B. We observe that there is a marginal drop in goodput between the distributed RAN and the C-RAN setup. This indicates that TCP can work good for fronthaul data transfer to ensure no loss in I/Q samples; therefore keeps the receiver goodput for C-RAN same as the receiver goodput for distributed RAN.

Next, we observe the receiver goodput for the various acquisition duration, as shown in Fig. 8. We observe that as we increase acquisition duration from 10 ms to 20 ms, the goodput increases; after that, the godput gets almost saturated. From this observation, we understand that acquisition duration has an impact on goodput. As we increase the acquisition duration, more amount of I/Q data is processed in batch; hence, goodput increases. However, after a threshold, the amount of I/Q data that is transferred to the BBU gets saturated due to fronthaul capacity, and therefore, the goodput also gets saturated.

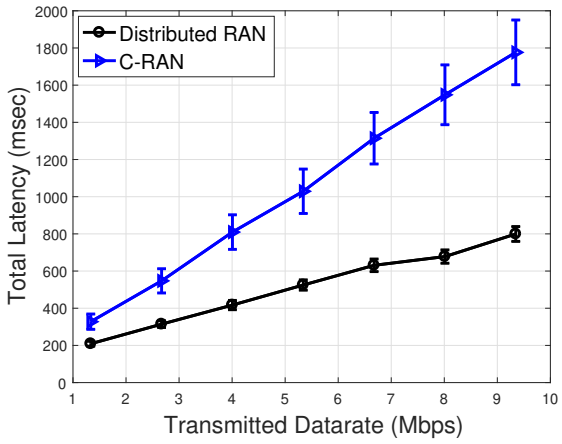

Figure 9. Comparison of transmitted data rate and total latency at RRH receiver acquisition duration of 40 ms in the case of TCP communication

D. Latency Analysis

Next, we analyze the transmission latency for the two scenarios – distributed RAN and C-RAN. In Fig. 9 (with respect to various transmission rates) and Fig. 10 (with respect to various acquisition duration) the total average latency required for a uplink transmission from a UE to the BBU is compared between distributed RAN architecture and C-RAN architecture. In Fig. 9, the total average latency is plotted for various transmitted data rates by fixing receiver acquisition duration at 40 ms. We observe that the latency increases as we increase the transmission rate. Further, the delay for C-RAN is higher compared to the delay for distributed RAN. In Fig. 10, the total average latency is plotted for various receiver acquisition times by fixing transmission data rate of 20 MSPS. We observe that the total latency increases as we increase the acquisition duration. Therefore, one important observation is that acquisition duration has an inverse effect on the transmission latency, although it does not have any impact

on the receiver goodput.

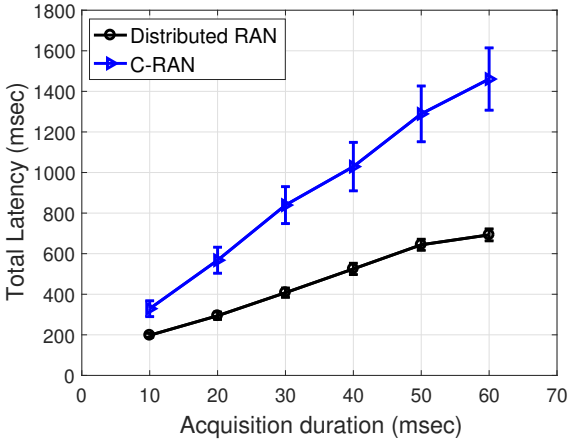

Figure 10. Comparison of RRH receiver acquisition Duration and total latency at a transmitted data rate of 20 MSPS in the case of TCP communication

To analyze the latency components further, we explore different latency components. The individual latencies involved in distributed RAN setup and C-RAN setup is shown in Table I (for various transmission rates) and Table II for the various acquisition duration. As earlier, the acquisition duration is kept fixed at 40 ms for the first case, and the transmission rate is kept constant at 20 MSPS for the second case. From the table, we observe that the first I/Q sample reception time is comparatively low compared to other latency components. As the transmission rate increases, the total I/Q sample size also increases, which impacts the I/Q processing time and the TCP latency. Further from Table II, we observe that I/Q sample size per block also increases as we increase the acquisition duration; as a result the latency for transferring data over the fronthaul gets increased. This impacts the total latency for data transfer over the C-RAN architecture.

E. Packet Drop at the Fronthaul Bottleneck

As there are no retransmission of packets in UDP-based communication, some of the packets are lost. Reconstruction of the message at the cloud server is not possible because of the packet drop. The packet drop and the corresponding goodput are calculated for different transmission rates by keeping a constant RRH receiver acquisition duration of 40msec. The packet drop and goodput in UDP-based Ethernet fronthauling is compared with TCP-based Ethernet fronthauling in Fig. 11 and Fig. 12 respectively. From Fig. 11, it is observed that the packet drop in TCP-based communication is almost negligible, whereas the packet drop in UDP-based communication increases with the increase in transmission rates. Hence, we are not able to achieve high goodput in UDP-based communication. In Fig. 12, it can be observed that the goodput in UDP-based communication decreases with the increase in transmission rates, whereas TCP-based Ethernet fronthauling supports high goodputs.

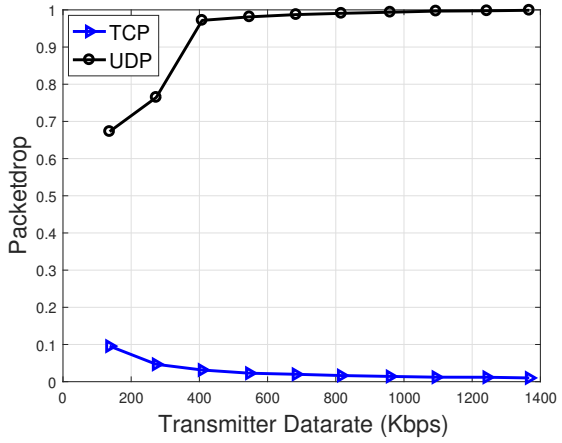

Figure 11. Comparison of Packet drop at the BBU pool for different transmission rates using TCP and UDP Ethernet fronthauling and at a constant RRH receiver acquisition duration of 40msec

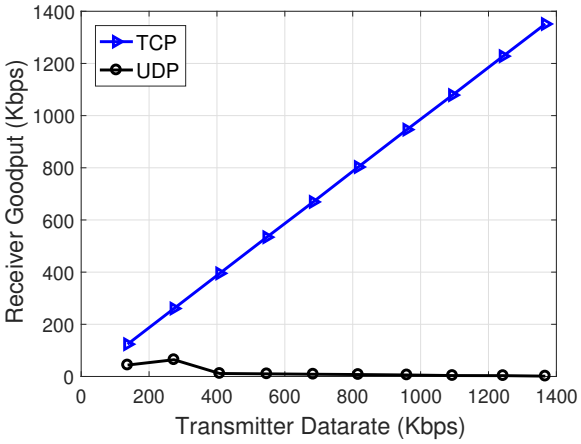

Figure 12. Comparison of C-RAN receiver goodput for different transmission rates using TCP and UDP Ethernet fronthauling and at a constant RRH receiver acquisition duration of 40 msec

VI. Conclusion

This paper discussed about the development of a C-RAN architecture with USRP and LabView platform, where TCP-based Ethernet fronthauling is used for communication between the RRH and the cloud hosting the BBU. We have explored TCP and UDP as the possible fronthaul protocols for transporting I/Q signals. We observed the impact of various system parameters on the latency-goodput performance for uplink data transmission, and compared the performance between a distributed RAN architecture and a C-RAN architecture. This paper has two major contributions. First, it provides an alternate implementation strategy for C-RAN setup without using the OAI protocol stack, which is comparatively simple, but can help emulating C-RAN behavior. Second, we observe that TCP can work as an alternate for reliable fronthaul communication, which can support goodput for C-RAN similar as a distributed RAN; however with the cost

Table I
COMPARISON OF TRANSMISSION LATENCY (IN MS) IN THE CASE OF TCP COMMUNICATION AND AT A CONSTANT RRH RECEIVER ACQUISITION DURATION OF 40 MS

S.No	Tx Rate (Mbps)	IQ Sample size (KB)	First IQ sample reception time (ms)	Processing latency - Distributed RAN (ms)	Total latency - Distributed RAN (ms)	Processing latency - C-RAN (ms)	TCP latency (ms)	Total latency - C-RAN (ms)
1	1.34	2×10^5	66.6	142.42	209.02	155.23	105.85	327.68
2	2.67	4×10^5	64	250.44	314.44	275.65	207.65	547.3
3	4.11	6×10^5	64.4	352.69	417.09	440.56	304.65	809.61
4	5.34	8×10^5	64.2	460.74	524.94	575.72	389.24	1029.16
5	6.68	1×10^6	70	560.87	630.87	747.85	496.63	1314.48
6	7.63	1.2×10^6	77.4	600.49	677.89	876.43	594.32	1548.15
7	8.9	1.4×10^6	84.6	714.85	799.45	991.32	700.46	1776.38

Table II
COMPARISON OF TRANSMISSION LATENCY (IN MS) IN THE CASE OF TCP COMMUNICATION AND AT A CONSTANT TRANSMISSION RATE OF 20 MSPS

S.No	Acquisition duration (ms)	IQ Sample size (KB)	First IQ sample reception time (ms)	Processing latency - Distributed RAN (ms)	Total latency - Distributed RAN (ms)	Processing latency - C-RAN (ms)	TCP latency (ms)	Total latency - C-RAN (ms)
1	10	2×10^5	64.2	133.63	197.83	157.66	107.01	328.87
2	20	4×10^5	64.2	229.85	294.05	294.05	209.4	567.65
3	30	6×10^5	64.2	343.55	407.75	467.24	308.01	839.45
4	40	8×10^5	64.2	460.74	524.94	575.72	389.24	1029.16
5	50	1×10^6	64.2	579.64	643.84	739.54	485.36	1289.1
6	60	1.2×10^6	64.2	628.5	692.7	884.75	511.77	1460.72

of increased latency. Such analysis provides a first step for a C-RAN prototyping platform using LabView and usage of TCP for Ethernet fronthauling, which can be explored further in the development of a robust cost-effective next generation communication solution. Since the reconstruction of the message at the cloud server was not possible because of the packet drop in UDP-based Ethernet fronthauling, goodput and latency calculations are not carried for high data-rates in the case of UDP-based communication. In future, the packet retransmission-based UDP communication will be explored to measure the goodput and the latency.

REFERENCES

[1] K. Chen and R. Duan, "C-ran the road towards green ran," *China Mobile Research Institute, white paper*, vol. 2, 2011.

[2] A. Checko, H. L. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. S. Berger, and L. Dittmann, "Cloud ran for mobile networks—a technology overview," *IEEE Communications surveys & tutorials*, vol. 17, no. 1, pp. 405–426, 2015.

[3] J. Liu, S. Zhou, J. Gong, Z. Niu, and S. Xu, "Graph-based framework for flexible baseband function splitting and placement in C-RAN," in *IEEE ICC*. IEEE, 2015, pp. 1958–1963.

[4] I. A. Alimi, A. L. Teixeira, and P. P. Monteiro, "Toward an efficient c-ran optical fronthaul for the future networks: A tutorial on technologies, requirements, challenges, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 708–769, 2017.

[5] J. Liu, A. Liu, and V. K. Lau, "Compressive interference mitigation and data recovery in cloud radio access networks with limited fronthaul," *IEEE Transactions on Signal Processing*, vol. 65, no. 6, pp. 1437–1446, 2017.

[6] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, 2014.

[7] Z. Kong, J. Gong, C.-Z. Xu, K. Wang, and J. Rao, "ebase: A baseband unit cluster testbed to improve energy-efficiency for cloud radio access network," in *IEEE ICC*, 2013, pp. 4222–4227.

[8] G. Mountaser, M. L. Rosas, T. Mahmoodi, and M. Dohler, "On the feasibility of MAC and PHY split in cloud RAN," in *IEEE WCNC*, 2017, pp. 1–6.

[9] C.-Y. Chang, N. Nikaein, R. Knopp, T. Spyropoulos, and S. S. Kumar, "FlexCRAN: A flexible functional split framework over ethernet fronthaul in Cloud-RAN," in *IEEE ICC*, 2017, pp. 1–7.

[10] C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Impact of packetization and scheduling on C-RAN fronthaul performance," in *IEEE Globecom*, 2016, pp. 1–7.

[11] O. Neji, N. Chendeb, O. Chabbouh, N. Agoulmine, and S. B. Rejeb, "Experience deploying a 5G C-RAN virtualized experimental setup using OpenAirInterface," in *IEEE 17th ICUWB*. IEEE, 2017, pp. 1–5.

[12] C. Y. Yeoh, M. H. Mokhtar, A. A. A. Rahman, and A. K. Samingan, "Performance study of LTE experimental testbed using openairinterface," in *18th IEEE ICACT*, 2016, pp. 617–622.

[13] A. Virdis, N. Iardella, G. Stea, and D. Sabella, "Performance analysis of OpenAirInterface system emulation," in *3rd IEEE FiCloud*, 2015, pp. 662–669.

[14] S. S. Kumar, R. Knopp, N. Nikaein, D. Mishra, B. R. Tamma, A. A. Franklin, K. Kuchi, and R. Gupta, "FLEXCRAN: Cloud radio access network prototype using OpenAirInterface," in *9th COMSNETS*, 2017, pp. 421–422.

[15] N. Makris, P. Basaras, T. Korakis, N. Nikaein, and L. Tassiulas, "Experimental evaluation of functional splits for 5G Cloud-RANs," in *IEEE ICC*, 2017, pp. 1–6.

[16] T. X. Tran, A. Younis, and D. Pompili, "Understanding the computational requirements of virtualized baseband units using a programmable cloud radio access network testbed," in *IEEE ICAC*, 2017, pp. 221–226.