IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 1, 2019, accepted December 15, 2019, date of publication December 25, 2019, date of current version January 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2962200

FPGA-Assisted Real-Time RF Wireless Data Analytics System: Design, Implementation, and Statistical Analyses

ZAHEER KHAN AND JANNE J. LEHTOMÄKI
Centre for Wireless Communications, University of Oulu, 90014 Oulu, Finland

Corresponding author: Zaheer Khan (zaheer.khan@oulu.fi)

This work was supported in part by the Infotech Oulu through the framework of digital solutions in sensing and interactions, and in part by the Academy of Finland 6Genesis Flagship under Grant 318927.

ABSTRACT A wide range of new ultra reliable low latency communication (URLLC) applications in next generation (NG) wireless systems demand real-time radio frequency (RF) data analytics of channel utilization (CU) that can help in making proactive resource allocation decisions. However, such real-time RF data analytics require processing of tens of millions of in-phase and quadrature (IQ) samples per second and sending huge quantities of samples to a resource allocating entity is not practical. We present design and implementation of an RF data analytics system which utilizes field-programmable gate arrays (FPGAs) at the network edge to process real-time streaming IQ samples from RF transceiver. FPGAs process millions of samples per second and output low-overhead descriptive statistics of wireless CU, such as mean CU values, maximum CU values, and entire histograms to obtain probability distribution of CU values, to a resource controller server where a quantile estimation based technique is used to detect congestion in CU in real-time. The FPGA-based modules are implemented on Xilinx's Zynq-7000 devices mounted with RF transceivers. We evaluate the performance of the implemented analytics system using extensive measurements, testing, and statistical analyses that are performed in both laboratory and over-the-air environments.

INDEX TERMS ARM, channel utilization, data analytics system, FPGA, histogram, measurements, prototyping, quantile, radio frequency, wireless systems, Xilinx.

I. INTRODUCTION

Next generation (NG) wireless networks are being designed to deliver not only very high data rates for enhanced mobile broadband but also to provide support for a wide range of new ultra reliable low latency communication (URLLC) applications [1]. The new URLLC applications include wireless connectivity solutions for autonomous vehicles, robotics, virtual reality (VR), and the internet of things (IoT) [2].

The creation of next generations of wireless networks that can support a variety of new applications and technologies will require transformation in a variety of ways. One such way is to use cloud technology based resource controllers that use dedicated radio frequency (RF) monitoring modules to collect/analyze data for better resource provisioning decisions [3], [4]. For example, Cisco Meraki controller utilizes monitoring data collected via access points (APs) each of which are equipped with dedicated RF modules which are used solely for data collection/processing purposes [4]. Accurate monitoring of the dynamic nature of RF environment requires real-time RF analytics. Real-time RF data analytics is the measurement, collection, and analysis of wireless data in real-time for purposes of understanding and optimizing network resource usage on the fly.

The main purpose of our work is to present design and implementation of a RF data analytics system that can process in-phase and quadrature (IQ) samples to obtain in real-time wireless channel utilization (CU) descriptive statistics. These statistics can help in making proactive resource allocation decisions at cloud technology based resource controllers. Accurate data analytics of the dynamic nature of wireless CU requires real-time processing of tens of millions of streaming IQ samples per second at a single wireless AP. For example, monitoring a single 20 MHz channel at Nyquist sampling rate

The associate editor coordinating the review of this manuscript and approving it for publication was Saad Qaisar.

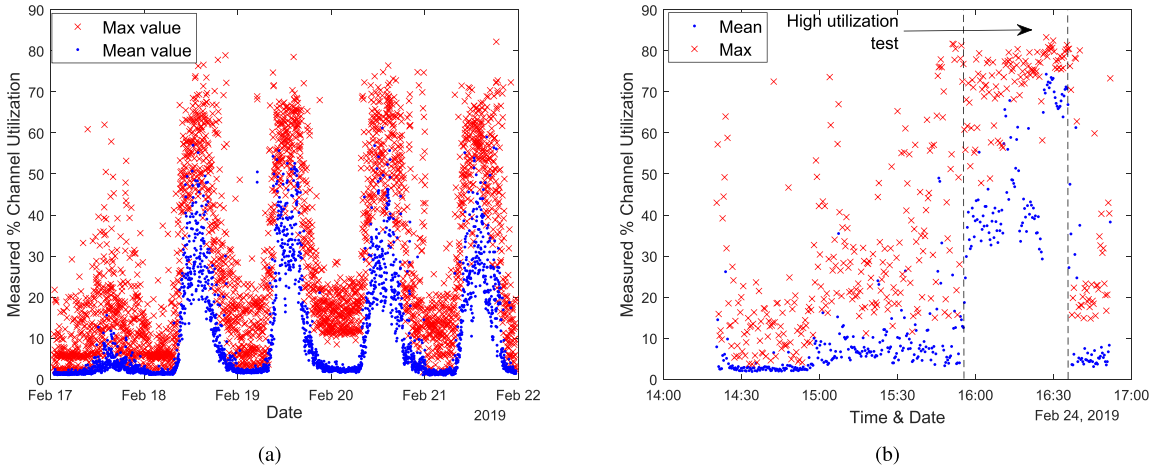


FIGURE 1. Example of over the air measurement results. a) Five days of collected mean and max %CU (percent CU) values using the implemented RA device; and b) Mean and max %CU values collected during high utilization test performed using Skype video, YouTube HD, and large file download (see Section VI).

and using a 4-byte word for each sample transfer can generate a stream of several hundred million bits per second. Sending such huge quantities of streaming data to the cloud server for analytics may be impractical due to high overhead incurred in the process of sending the data, and also high latency incurred in the process of waiting for the data to reach the cloud for analysis and the subsequent decisions to be delivered back. This factor becomes particularly critical for URLLC applications which require real-time responses. Moreover, even for use cases that may not need strict real-time decisions, sending only limited data that indicates important information relating to the estimated performance metric of interest makes more sense than sending large volumes of streaming samples of data to the controller. An RF data analytics system which uses hardware-accelerated data processing modules at the network edge can solve the problem of latency and sample transfer overhead. We use low-cost field-programmable gate arrays (FPGA) to process IQ samples to obtain CU descriptive statistics in real-time. In our work, we call such hardware-accelerated modules as RF analytic (RA) devices.

The main contributions of our work can be summarized as:

- We present a low cost real-time RF analytics system which consists of RA devices and a controller server. Each RA device acts directly in real-time on gapless streaming IQ samples from RF transceiver and processes them to output low-overhead descriptive statistics to a resource controller server. Fast real time processing of samples is achieved by offloading statistical signal processing tasks to the FPGA part of the RA device. This is different from previous works which have mostly focused on non-real time RF analytics where IQ samples are collected via off-the-shelf spectrum analyzers, or via software defined radio (SDR) platforms, such as USRP with GNU radio functions or WARP with WARPLab functions [5], [6]. In such non real-time approaches, the collected IQ samples are stored in memory buffers and then transferred to a separate module, such as a laptop or a PC server, for processing. This can lead to delay in processing of samples due to slow transfer speed between IQ data collection module and the host processing them. Moreover, limitations in memory buffer size can lead to gaps in collected data which can degrade the performance of data analytics system.
- We utilize the implemented RA devices to collect low-overhead descriptive statistics relating to real-time channel utilization (CU). The RA devices measure CU and provide meaningful insights to the server by calculating mean/maximum CU values and also histogram of CU values in real-time. This is different from approaches which tend to focus solely on mean statistics of the CU. However, mean alone cannot be a good measure as RF data exhibits very often skewed distribution. As an example, in Figs. 1a and 1b we illustrate mean/maximum statistics of wireless channel utilization (CU) collected over an unlicensed channel by us in the University of Oulu. The statistics in Figs. 1a and 1b were collected using our implemented RA device. It can be seen from the figures that at times maximum CU values alone can lead to over estimation of CU, and at times mean CU values alone can lead to under estimation of CU. We explain this observation further in Section VI.
- We present a proactive CU congestion control method which can utilize real-time CU statistics collected via implemented RA devices to determine if an AP needs more resources than allocated to it. In the presented method, at every fixed interval t, the server receives processed statistics from the RA device. As quantiles of elements in streaming data can provide more meaningful information than mean or and/or maximum values alone, we use quantile estimation techniques on collected mean CU, max CU and histogram of CU samples. We show that a quantile estimation based technique can

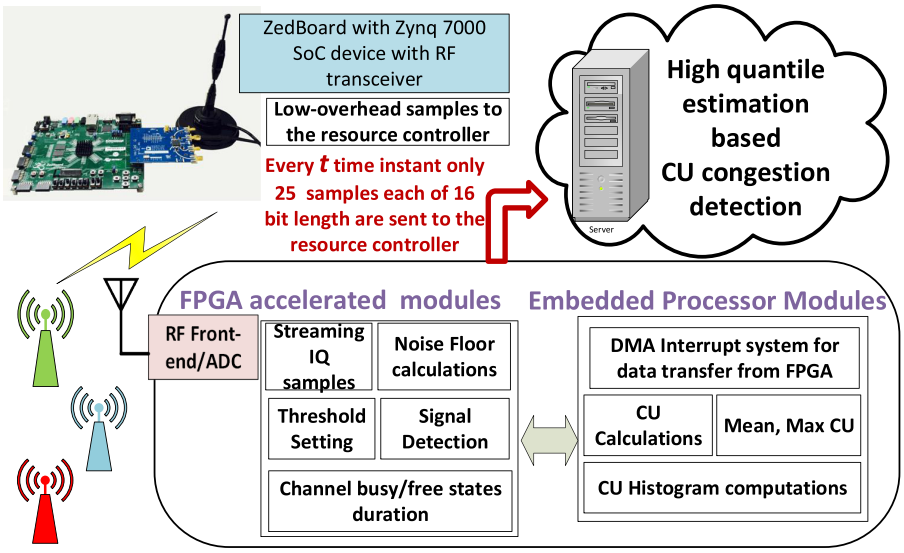


FIGURE 2. Various components of our prototyped RA device and data analytics system.

be used to design a proactive CU congestion control method at a resource controller. The method can help the controller to scale up resources when required based on real-time information guided by statistical evidence. For example, the quantile characterization of CU is used to assess the risk of getting CU above some threshold value that can cause degradation in channel usage quality for various applications, e.g, degraded quality of Skype video call.

- The RA device is realized in practical implementation by prototyping it on a low-cost ZedBoard with AD9361 RF transceiver attached to it. ZedBoard is equipped with a Xilinx Zynq-7000 system on chip (SoC). We provide extensive measurement/testing results that evaluate the real-time performance of the implemented device under three different setups: 1) laboratory measurement/testing using a signal generator which generated pulses of different duty cycles and under varied signal to noise ratios (SNRs). The laboratory testing allowed us to quantify the CU measurement performance of the prototyped RA device by testing it with known signal pulses; 2) anechoic chamber measurement/testing in which wireless signals from multiple user devices using various video/audio/real-time applications were allowed to propagate inside the room; and 3) measurements/testing in a real wireless network in which we collected data for two weeks using three ZedBoards on each of which RA device algorithms were prototyped.

In Fig. 2, we illustrate the RF analytics system presented in this paper. It shows the following components of the implemented RA device: 1) An AD9361 RF frontend attached to the RA device for streaming reception of IQ samples from multiple active APs; 2) Various FPGA accelerated modules, such as noise floor estimation, signal detection, and CU state calculations; 3) Embedded processor modules, such as sample transfer via direct memory access (DMA) from FPGA, mean CU, maximum CU, and CU histogram computations; and 4) a communication module between the RA device and the server for streaming transfer of processed CU samples. The figure also shows a server where a high quantile estimation based technique on collected CU descriptive statistics samples is used to design a proactive CU congestion control method at the server.

It is important to note that our proposed solution is not only more efficient but also less costly than approaches used in other research works which use commercial measurement devices, such as spectrum/signal analyzers or software defined radio (SDR) boards, to collect IQ samples and then use a host computer for their processing. An advanced spectrum/signal analyzer in general costs tens of thousands of dollars. An SDR board like WARP and USRP can cost from a few thousand to several thousand dollars. Our solution is less costly because the total cost of a system on chip SoC device with an RF module that can be used for prototyping our algorithms is less than a thousand dollars. Moreover, this cost can be reduced considerably further by FPGA-to-ASIC conversion and mass production.

The rest of the paper is organized as follows: In Section II, we present an overview of the related literature. We then provide a background on collected descriptive statistics and quantile estimation framework in Section III. Then in Section IV, design and architecture of the implemented RA device is presented. Validation of the device through measurements and tests is presented in Section V. In Section VI, we provide results in terms of quantile estimates of over-the-air (OTA) collected samples. Before concluding our work, in Section VII, we present a quantile estimation based method

that can help proactive CU congestion control at a resource controller.

II. RELATED LITERATURE

The importance of data driven solutions for 5G and beyond wireless systems has been recognized in several recent research works [7]–[9]. To incorporate data analytics functionality, 3rd Generation Partnership Project (3GPP) has taken a first step in this direction by introducing a network data analytics function (NWDAF) [10]. Advances in software/hardware technologies and internet of things (IoTs) allow wireless systems to collect in real-time various types of data sets from user equipments and also from wireless network elements. For example, the work in [7] provides an intelligent information forwarder solution for healthcare related big data systems using distributed wearable sensors. Moreover, use of dedicated sensors that act as RA devices for fifth generation (5G) wireless systems have been proposed in [11], [12].

To exploit useful information from analytics for a network control, configuration and management, dedicated RF measurement/data collection modules are deployed by wireless industry. The information from these dedicated modules are given to virtualized cloud-based resource controllers for efficient resource allocation. For example, Cisco System's Meraki Cloud Controller (MCC) utilizes dedicated radio module called Air Marshal in each Meraki AP [4]. The dedicated module monitors radio environment of the network. However, unlike our work, MCC collects only average CU values.

An overview for the adoption of data analytics capabilities as part of a ''next-generation'' architecture is provided in [13]. The work in [14] explores various means of integrating big data analytics with network optimization towards the objective of improving users quality of experience. Use of measurement capable devices (MCDs) and data analytics for 5G networks have been proposed in [11]. The works in [15], [16] have focused on exploiting the RF analytics using processor based modules. However, all these works perform analytics by processing off-line (stored) wireless network data. Different from these works, our solution enables an agile resource controller to exploit analytics that is derived from the knowledge of real-time gapless streaming data.

SoCs with hardware/software programmability have attracted strong interest in performing analytics and designing modules that can provide intelligence in the edge devices [12], [17]–[19]. Offloading real-time measurements and signal processing tasks relating to RF analytics on FPGA part of the SoC offers a fast flexible programmable hardware solution. Moreover, an FPGA can perform multiple processing tasks in parallel and it can also act and react in real-time on large volumes of IQ data samples. In [20], FPGAs are utilized for the implementation of an enhanced WLAN Security System targeting multimedia applications. An FPGA prototype of reconfigurable architecture of universal filtered multicarrier transmitter for 5G wireless systems is presented in [21]. The authors in [22] have highlighted the benefits of using SoC devices for real-time RF analytics, and they have also presented an FPGA implementation design for a deep learning algorithm. The authors in [23] have presented a hardware design for a spectrum sensor for next generation LTE-A wireless networks.

III. BACKGROUND TO CU, DESCRIPTIVE STATISTICS, AND QUANTILE ESTIMATION

A. CU AND DESCRIPTIVE STATISTICS

CU indicates how much any transmissions the RA device can ''hear'' on a channel, from all sources. Typically, this statistic is often given in a percentage between 0% to 100% and indicates the amount of time the RA device finds the channel busy. It includes all types of transmissions from all APs, and their clients. For the channels which are shared among multiple wireless technologies it will also include any other technology devices using the same channel.

To measure the CU, the RA device directly processes IQ samples on the FPGA in real-time (with processing speed of several million samples per second). To accurately detect signals in real-time, it is important to estimate noise level and set the detection threshold value appropriately. The threshold setting separates desired signal level versus unwanted noise level. When received $I^2 + Q^2$ value exceeds a pre-defined threshold then signal is declared to be present, otherwise, it is declared to be absent. The streaming signal detection outputs are processed sample-by-sample on FPGA using busy/free state counters which are implemented with a two-state finite state machine (FSM). The CU value in a time interval t is:

$$\alpha_t = \frac{L_F}{L_F + L_O} \tag{1}$$

where L_F represents the number of samples in time instant t in which signal is declared to be present, L_O represents the number of samples in time instant t in which signal is declared to be absent, and $L_F + L_O$ is the total number of samples in time instant t. Note that depending on the channel bandwidth, sampling rate, etc., the number of samples in time instant t may vary from several hundred thousands to a few million. A block diagram showing high level architecture of implemented CU statistics processing modules is given in Fig. 3.

Every short interval of n time units called sample block contains a sequence of CU observations $\mathcal{A}_i = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ from an unknown distribution function F. Let $M_i = \max \mathcal{A}_i$ denote the maximum value of the observations, and $m_i = \text{mean} \mathcal{A}_i$ denote the mean of the observations. The CU sample distribution can vary highly over time and it is often highly skewed. Hence, to understand behavior when CU gets closer to congestion, it may not be enough to utilize information from only mean CU sample values. In Section V, where we present our detailed RF data analytics results, we show how measuring only mean and/or max CU can be misleading in terms of handling congestion in CU. To address this problem, we also track frequency distribution of CU values in real-time by constructing histograms of measured CU values.

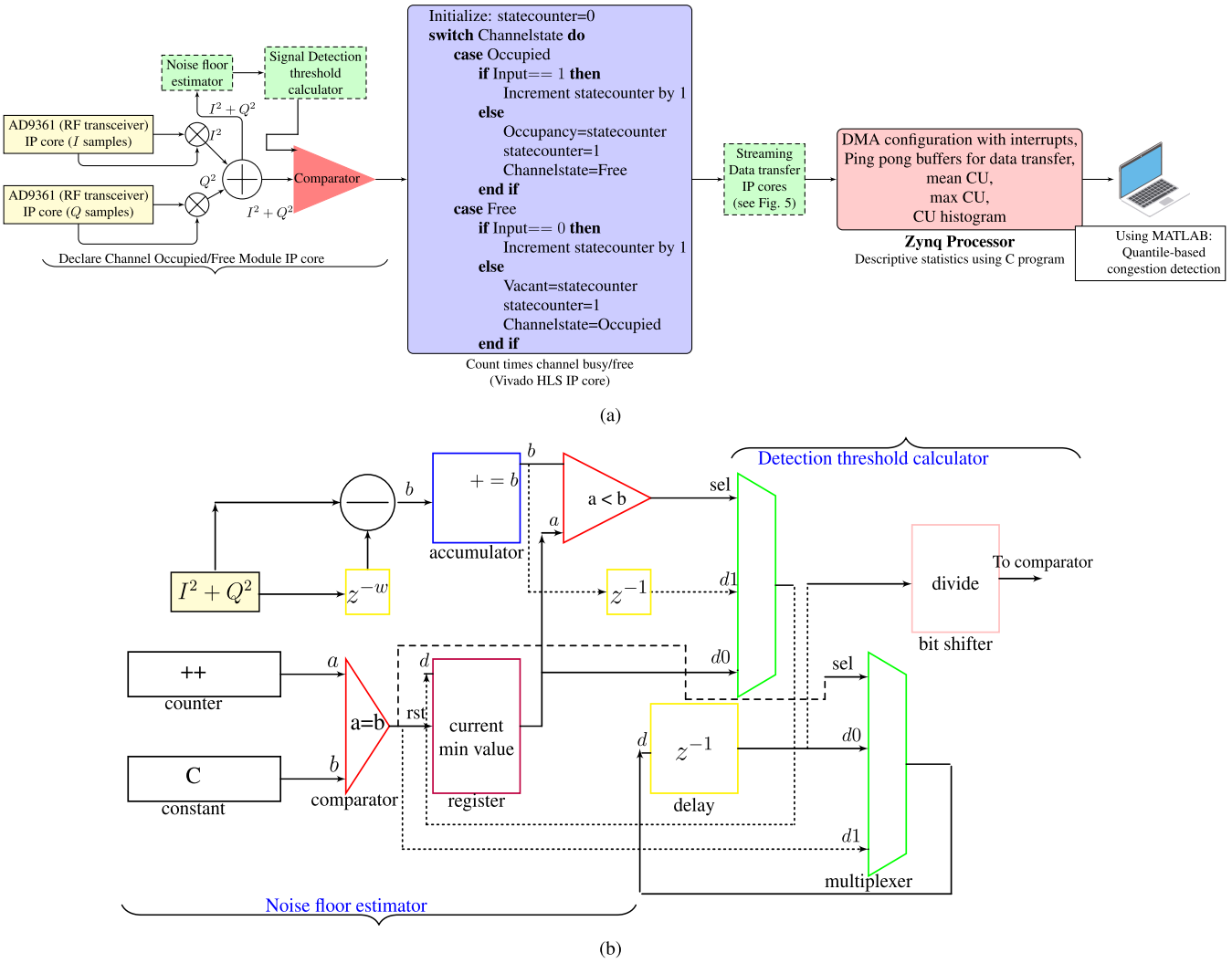


FIGURE 3. a) High level design of the implemented CU statistics processing chain; b) Circuit diagram for the noise floor estimator and the signal detection threshold calculator modules shown in the high level design.

For the short interval of n time units, a histogram $\mathcal{H}_i$ of CU observations is given by

$$\mathcal{H}_i = \{(I_1, \pi_1), (I_2, \pi_2), \cdots, (I_b, \pi_b)\} \tag{2}$$

where $I_1, I_2, \cdots, I_b$ are partitioning of CU into b contiguous intervals also known as bins. Each bin is defined as $I_j = [\underline{I}_j; \overline{I}_j)$ with $\underline{I}_j$ as the minimum value and $\overline{I}_j$ as the maximum value. When a sample of CU is within some bin I_j then the counter for that bin is incremented by one or else it remains the same. The count values for b bins are given by $\pi_1, \pi_2, \cdots, \pi_b$. The histogram is based on all CU samples in a block. Approximate CU values from a histogram can be recalculated by having a sufficiently large number of bins. The mean and max values are based on mean CU and maximum CU for each block so they carry in principle less information than the histogram.

B. QUANTILES OF CU STATISTICS

Every time interval of duration I_l, we compute quantiles of the elements in block mean, block maximum, and block CU sample vectors. Fig. 4 presents an illustrative example of continuously collected samples and samples within duration I_l. Quantiles of the sample elements in the interval I_l can be explained as follows. One can arrange the N number of sample observations in the interval I_l represented as $X_1, X_2, ...X_N$ to get the order statistics $X_{(1)} \leq X_{(2)} \leq, ..., \leq X_{(N)}$. Quantiles are order statistics and we can now define q-quantile as

Definition 1: The q-quantile of ordered sample observations in the interval I_l is a number z such that qN elements of ordered sample observations are less than or equal to z and the remaining $(1-q)N$ are greater than z.

C. QUANTILE ESTIMATES OF BLOCK MEAN SAMPLES

When we are interested in studying a series of block mean CU samples in the interval I_l then it is usual practice to approximate the distribution of samples by the normal distribution. This is due to the well known Central Limit Theorem which states that no matter what the shape of the

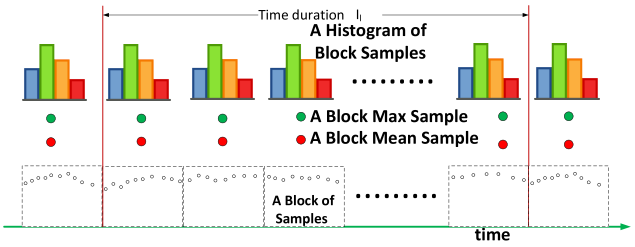


FIGURE 4. Illustration of collected CU statistics and also of time duration I_l.

population distribution is, the sampling distribution of the sample means approaches a normal distribution as the sample size gets larger. This fact holds especially true for sample sizes 30 or above. The normal cumulative distribution function (cdf) of sample means can be given as

$$p = F(x) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \tag{3}$$

where μ and $\sigma > 0$ are fixed parameters that represent expectation and standard deviation, and p is the probability that a single observation from a normal distribution with parameters μ and σ falls in the interval $(-\infty, x]$. Quantile estimates of the mean distribution can be obtained by inverting (3) and its closed-form expression is given as

$$q^* = x = \mu + \sigma\sqrt{2}\,\text{erf}^{-1}(2p-1), p \in (0,1) \tag{4}$$

so that $F(q^*) = p$.

D. QUANTILE ESTIMATOR OF BLOCK MAXIMA

When a series of block maxima samples are studied then the extremal type theorem has shown that the generalized extreme value (GEV) distributions may approximate the distribution of sample maxima [24]. The three important families of the GEVs can be combined into a single family of models having cdf

$$p = G(x) = \begin{cases} \exp\left\{-\left\{1+\xi\left(\frac{x-\bar{\mu}}{\bar{\sigma}}\right)\right\}^{-1/\xi}\right\}, & \text{with } \xi \neq 0 \\ \exp\left\{-\exp\left\{-\frac{(x-\bar{\mu})}{\bar{\sigma}}\right\}\right\}, & \text{with } \xi = 0 \end{cases} \tag{5}$$

where $\bar{\mu}$, $\bar{\sigma}$, and ξ are location, scale, and tail shape parameters, respectively, and p is the probability that a single observation from the distribution with parameters $\bar{\mu}$, $\bar{\sigma}$, and ξ falls in the interval $(-\infty, x]$. By inverting (5), quantile estimates of the max sample distribution for $\xi \neq 0$ can be given as [24]

$$q^* = x = \bar{\mu} + \frac{\bar{\sigma}}{\xi}\left\{\{-\log(p)\}^{-\xi} - 1\right\} \tag{6}$$

so that $G(q^*) = p$.

E. QUANTILE ESTIMATOR OF VALUES OVER THRESHOLD

Using only block mean and/or block maxima CU samples is not enough for CU analysis. The use of hardware accelerated data collection modules allow the possibility of obtaining large number of streaming samples. This means that we can look at beyond mean and max CU values. We use another extreme value analysis that is more efficient as it views all those observations as extreme observations that surpass a defined threshold level τ. Hence, now we are interested in all excess CU values over the threshold τ. To achieve this the observations in collected histograms during the interval I_l are utilized to study extreme values that exceed τ. We can write the empirical cumulative distributive function (ecdf) of the excess CU values above τ

$$\begin{aligned} F_\tau(a) &= P\{\alpha - \tau \leq a \mid \alpha > \tau\} \\ &= \frac{F(a+\tau) - F(\tau)}{1 - F(\tau)} \end{aligned} \tag{7}$$

for $0 \leq a < a_f - \tau$. The finite right endpoint of the distribution F is denoted by a_f. $F_\tau(a)$ is thus the probability that a CU sample exceeds the threshold τ by no more than an amount a, given that the threshold is exceeded. Generalized Pareto (GP) distribution may approximate the distribution of sample values exceeding the threshold τ [24]. The cdf can be written as

$$p = \tilde{G}(x) = \begin{cases} 1 - \left(1+\xi\left(\frac{x-\breve{\mu}}{\breve{\sigma}}\right)\right)^{-1/\xi}, & \text{with } \xi \neq 0 \\ 1 - \exp\{-\left(\frac{x-\breve{\mu}}{\breve{\sigma}}\right)\}, & \text{with } \xi = 0 \end{cases} \tag{8}$$

where $\breve{\mu}$, $\breve{\sigma}$, and ξ are location, scale, and shape parameters, respectively, p is the probability that a single observation from the distribution with parameters $\breve{\mu}$, $\breve{\sigma}$, and ξ falls in the interval $(-\infty, x]$. By inverting (8), quantile estimates of the sample distribution for $\xi \neq 0$ can be given as [24]

$$q^* = x = \breve{\mu} + \frac{\breve{\sigma}}{\xi}\left\{(1-p)^{-\xi} - 1\right\} \tag{9}$$

so that $\tilde{G}(q^*) = p$.

IV. RF ANALYTICS SYSTEM AND IMPLEMENTED RA DEVICE DESIGN

RF IQ data acquisition in a wireless system with MHz channel bandwidths can produce several millions of samples per second. Signal detection, channel utilization and descriptive statistics modules implemented on a device close to the network edge reduces overhead as transfer of several million raw IQ samples is no longer required. It also increases measurement speed, accuracy, and performance. For example, as gaps in IQ data acquisition can have adverse effect on valid inference in real-time CU analysis, hardware acceleration of signal processing modules allows gapless acquisition and processing of streaming raw samples in parallel. Gapless streaming also provides increase in the number of descriptive statistics samples that can be used to estimate quantiles.

A. TOOLS FOR HDL DESIGN OF THE RA DEVICE

The RA device solution presented in this work is implemented on a low-cost Zedboard which adopt's SoC technology based on Xilinx's Zynq architecture [25]. Zedboard provides in the same chip a multi-core ARM processor tightly coupled to an FPGA. Moreover, it also allows to connect to it with an agile RF transceiver via its FPGA Mezzanine Card (FMC) connector. We have used AD9361 which is a high performance, highly integrated agile RF transceiver. The data processing modules which output various CU related statistics are implemented by incorporating new IP cores in the programmable logic part of the FMCOMMS2 HDL reference design of Analog Devices [26], and also by modifying their No-OS software for the processor unit. For the FPGA part, the processing modules in the form of IP cores are developed and incorporated in the reference design by using three different tools: 1) Vivado which is a software produced by Xilinx for synthesis and analysis of HDL designs; 2) Xilinx system generator (XSG) for digital signal processing (DSP) which can be used to design, test and implement high-performance DSP algorithms on Xilinx FPGA devices. XSG allows to package the implemented DSP design as an IP core that can be added to the Vivado IP catalog for use in another HDL design. Another advantage of using XSG is that it provides the possibility of simulation even before the compilation of the model; and 3) Vivado high level synthesis (HLS) which enables processing modules written in C/C++ code to be targeted into Xilinx devices without the need to manually create register-transfer level (RTL) design. This greatly reduces the implementation time. For the embedded processor, the C programming software is created using the Xilinx's Software Development Kit.

B. MAIN HARDWARE/SOFTWARE COMPONENTS OF THE RA DEVICE

Fig. 5 shows high level data flow between various IP cores of our implemented RA device on Zynq-7000 based Zedboard with AD9361 RF transceiver. The figure also includes the IP cores of the original FMCOMMS2 reference design [26]. It can be seen from the figure that the AXI AD9361 IP core interfaces to the RF transceiver device. The main function of this core is to handle all the low level signaling, which is defined by the device's digital data interface, and to forward the received IQ data to a more simple first in, first out (FIFO) interface. Our implemented IP cores take streaming IQ samples and perform CU related processing. As shown in Fig. 5, the ADC PACK core collects samples from our implemented IP cores and passes them to the DMA core. The DMA core writes CU data to the main memory.

The use of an FPGA and an embedded processing unit to obtain various CU related statistics allows partitioning of time critical signal processing tasks to be done on FPGA, letting the processor do less critical processing. In Fig. 3, we illustrate this partitioning of tasks between the FPGA and the embedded processor. Various hardware modules

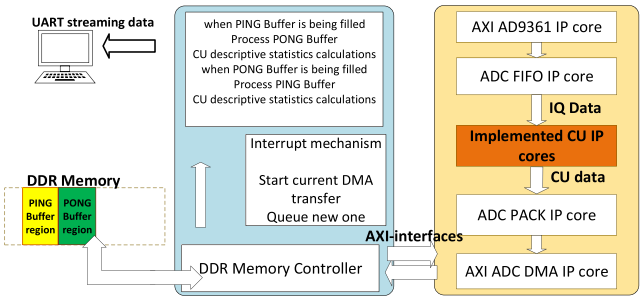


FIGURE 5. CU statistics processing chain (including IP cores from the FMCOMMS2 reference design [26]) showing data flow among IP cores.

implemented on the FPGA of Zedboard are explained in the next subsections.

1) ADAPTIVE NOISE FLOOR ESTIMATOR AND DETECTION THRESHOLD CALCULATOR

To measure CU on a channel in real-time, it is important to distinguish between desired wireless signals present in the channel and unwanted noise. To achieve this typically requires estimation of noise level, and setting an appropriate signal detection threshold value. Our implemented adaptive noise floor estimation (NFE) module operates in real-time directly on the streaming samples which are an unknown combination of signal samples and noise samples (see Fig. 3b for the design details). The NFE module is based on statistical extreme value theory. The module exploits the wireless signal samples property that mostly they are not randomly distributed among all the received samples but are clustered together, for example in the form of a data packet. Also, samples corresponding to the noise only represent that no wireless transmissions are present and they are clustered together in time domain. We use minimum operation to find the energy detector output with noise-only samples, and use it to estimate noise floor. As noise floor can change with time, this is taken into account by updating the estimated noise floor value after every fixed interval of time. We implement this in the hardware by using a counter. When the counter reaches the value representing fixed interval the comparator circuit is triggered and the estimated noise floor value is updated. The estimated noise floor is used to calculate/update the detection threshold τ_d which is typically a value γ dB above the estimated noise floor level. The threshold setting separates desired signal level versus unwanted noise level. Typically, the value γ is selected after lab testing the RF transceiver with signals of various signal-to-noise ratios (SNRs).

2) CU BUSY/FREE STATE AND ITS DWELL TIME CALCULATOR

The high speed and parallel-processing capabilities of an FPGA in Zedboards allows us to use CU state calculator module in parallel with NFE/detection modules. The sample level CU state can be either busy (meaning desired wireless signal is declared to be present) or free (meaning desired wireless signal is declared to be absent) state. When the

I^2+Q^2 value exceeds the detection threshold τ_d then wireless signal is declared to be present, otherwise, it is declared to be absent. A comparator is used to compare the streaming I^2+Q^2 samples against τ_d. It compares them and gives as output either 1 (signal present) or 0 (signal absent) values.

The streaming output of the comparator is given as input to the channel state dwell time calculator IP core. The pseudocode of its implementation on FPGA using Vivado HLS is given in Fig. 3. The IP core has two outputs: i) length of occupied (busy) state; and ii) length of free (vacant) state. For the first output, when the channel is observed busy it starts calculating for how long (in terms of number of samples) the channel remains in the busy state. It outputs the length of busy state (denoted as l_f) when the state changes from the busy to vacant. For the second output, when the channel is observed vacant it starts calculating for how long the channel remains in the vacant state. It outputs the length of vacant state l_o when the state changes from vacant to busy. Following (1), the streaming outputs of lengths of vacant/busy channel state can be used to calculate CU within some time interval as

$$\alpha_t = \frac{L_F}{L_F + L_O} = \frac{\sum_{j=1}^{n} l_{f,j}}{\sum_{j=1}^{n} l_{f,j} + \sum_{k=1}^{m} l_{o,k}} \tag{10}$$

where n and m are the number of busy/vacant periods, respectively.

Various software modules implemented on the embedded processing system of the Zedboard are explained in the next subsections.

3) DMA CONFIGURATION USING INTERRUPTS FOR DATA TRANSFER

Fast streaming channel state samples are required to be moved to the Zynq embedded (processor system) PS where the samples are further processed to calculate descriptive statistics in real-time, such as mean, maximum and CU histogram values. The challenge in this case is the efficient delivery of samples from the FPGA of Zynq to the main memory of the system without the intervention of the PS so that the computational power of the PS can be used for the fast real time processing of samples. To achieve this we have used AXI DMA configuration with interrupts. DMA allows direct access to the main memory and is used here to quickly transfer data between the memory and the FPGA without the intervention of the PS. Note that without DMA with interrupts typically data transfers which involve the PS can incur a lot of overhead during each transfer which can slow down the system and can lead to gaps in the collected streaming samples.

The transfer flow involves FPGA using the DMA controller (DMAC) to write to the memory and the PS reads from the memory for further processing. The implemented interrupt-based control of the AXI DMA controller allows the PS to initiate the transfer only, then it does processing of samples while the transfer is in progress, and it finally receives an interrupt from the DMAC when the submitted transfer is completed. As the FPGA sends streaming channel state samples in real time, the DMAC will almost continuously write to the main memory. For the DMAC to write and the PS to read without any collisions, ping pong buffers based scheme is utilized.

TABLE 1. Resource utilization of the prototyped RA device.

Resource	Utilization	Available	Utilization %
LUT	11601	53200	21.81
LUTRAM	694	17400	3.99
FF	20697	106400	19.45
BRAM	6.50	140	4.64
DSP	69	220	31.36
IO	124	200	62

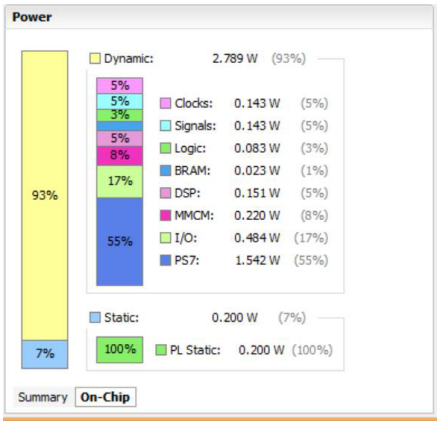


FIGURE 6. Power consumption of the prototyped RA device.

4) PING PONG BUFFERS BASED PROCESSING

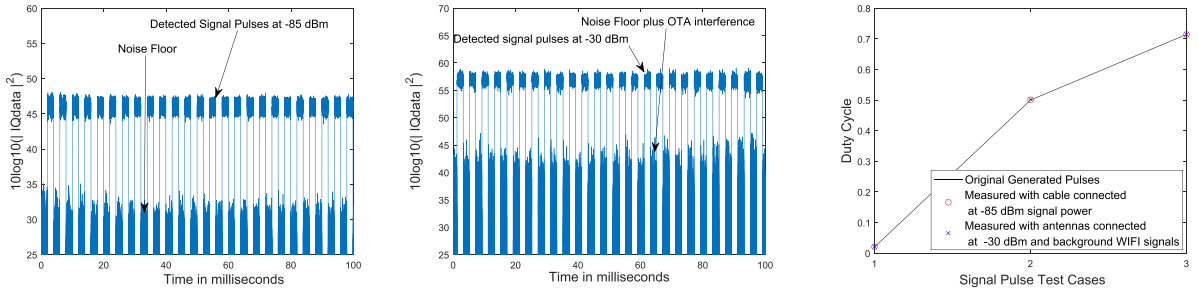
Our implementation of ping pong buffers can be explained as follows. We divide part of the main memory into two buffers which we name as ping pong buffers. The DMAC writes to only the parts of the memory allocated to the two buffers. When the DMAC writes to the ping buffer part the PS processes samples written to the pong buffer, and when the DMAC writes to the pong buffer the PS processes samples written to the ping buffer. The PS keeps moving between the two buffers and performs stream processing of the CU samples. Fig. 5 shows the use of ping pong buffers.

5) BLOCK MEAN, MAX, AND HISTOGRAM CALCULATION

The PS obtains block mean m_i and block maximum M_i CU values from the streaming CU samples by simply calculating mean and maximum of values for every n samples, where n is the block size that should be sufficiently large. It also constructs an entire histogram $\mathcal{H}_i$ (see (2)) for every n samples.

V. VALIDATION OF THE PROPOSED DEVICE THROUGH MEASUREMENTS AND TESTS

In Table 1, we show the resource utilization of the implemented design on FPGA. It can be seen from Table 1 that most resource utilization occurs in terms of lookup tables (LUTs), digital signal processing (DSP) blocks, flip-flops (FFs) and input/output (IO) blocks. In Fig. 6, we show power consumption of the implemented design. It can be seen from



(a) Wired connection measurements

(b) OTA measurements in the presence of background WiFi interference

(c) Wired and OTA measurements

FIGURE 7. Measurement/testing results of the RA device using signal generator under various scenarios.

the figure that out of a total power consumption of 2.9 watts 55% is consumed by the embedded processor (PS7) block.

Before the implemented SoC-based RA device is deployed, it is also important to evaluate its measurement performance under various controlled laboratory testing scenarios. In our work, we evaluate the measurement performance of the implemented device under different scenarios: 1) Testing with various signals using a signal generator connected via wire to the antenna connector of the RF front end of the device; and 2) OTA testing with various signals using a signal generator connected with an antenna and the device RF front end also connected with an antenna. In all the testing setups, the Zedboard board with RF front end was connected to a laptop computer via UART serial port. The results outputs on the laptop computer were generated using the MATLAB software.

A. TESTING WITH WIRED SIGNAL GENERATOR

In our measurements/testing laboratory, we tested the signal detection performance of the implemented device with 2.4 GHz pulse signals of various duty cycles (the fraction of one period in which signal is ON) and various absolute power levels. For example, the device is tested with very short signals having duty cycle of only 2%. In wireless networks such short pulse signals can be beacon signals which are used for transmitting critical information about the network. The device is also tested with pulses having 50% and more than 50% duty cycles. In wireless networks, such medium to high duty cycle pulse signals can represent high utilization of channel by multiple users. We varied the signal power levels from -90dBm to -75dBm. The signals were generated using an Agilent E4438C vector signal generator which was connected via wire to the antenna connector of the Zedboard board's RF front end running the implemented algorithms. The result outputs of the tests were recorded on a laptop computer via serial port connection between the Zedboard and the laptop. The signal generator was used to repetitively generate a group of streaming pulse signals in real-time. In Fig. 7(a), our device testing results show that even at -85dBm the implemented device can detect streaming signal pulses accurately. Fig. 7(a) shows results for pulse with 50% duty cycle.

B. SIGNAL GENERATOR TESTING WITH ANTENNA ATTACHED

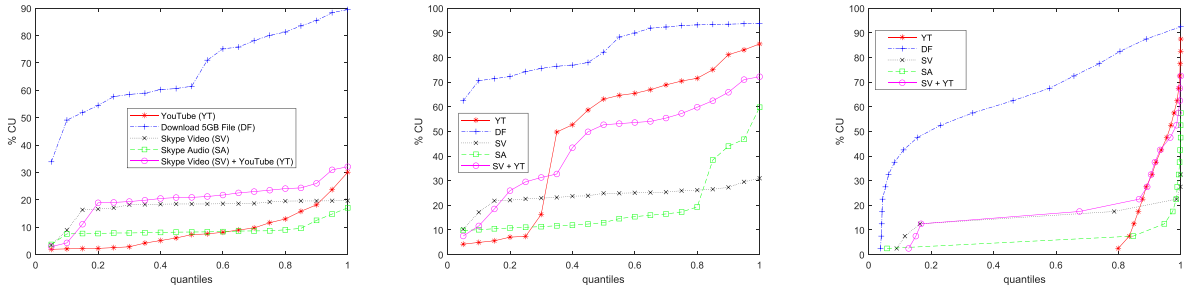
We also performed OTA testing using the signal generator in our laboratory. The signal detection was tested with pulse signals of various duty cycles and various absolute power levels. The signals were transmitted using antenna connected to signal generator and were received at Zedboard board also connected with an antenna. In Fig. 7(b), we present example results of OTA device testing showing that even at -30dBm the implemented device can detect streaming OTA signal pulses accurately. Streaming pulses with 50% duty cycle are used in the results of Fig. 7(b). The accuracy results relating to measuring the duty cycle for short (2% duty cycle, test case 1), medium (50% duty cycle, test case 2), and high (72% duty cycle, test case 3) pulses for both OTA and wired signal generator transmissions are presented in Fig. 7(c). It can be seen from the figure that under both OTA and wired scenarios the implemented device measured the duty cycle correctly.

VI. OTA DATA COLLECTION, AND QUANTILES

A. DATA COLLECTION IN AN ANACHOIC CHAMBER

We use the implemented device to calculate real-time descriptive statistics of OTA wireless transmissions so that we have some idea what the OTA data looks like and how spread out it is in terms of block mean CU values, block max CU values, and CU values in block histograms. The implemented device was first used in the anechoic chamber of the University of Oulu where also a WiFi access point was deployed for measurement purposes. An anechoic chamber is a room that almost completely blocks outside RF signals. This allowed RF signals from only one WiFi access point to propagate inside the room. In the same room, one HP laptop and two smart phones were connected to the access point using WiFi.

To see how OTA CU data behaves we ran on the devices applications like high definition (HD) YouTube (YT) which is an example of streaming application, Skype audio (SA), Skype Video (SV) which are examples of real-time applications, Skype video plus YouTube (real-time and streaming applications), and downloading large file (DF). In Figs. 8a, 8b and 8c, we show quantiles for the block mean



(a) Block Mean CU (b) Block maximum CU (c) CU values from histograms

FIGURE 8. Anachoic chamber results for various applications showing quantiles for the cumulative probabilities p in the interval (0,1].

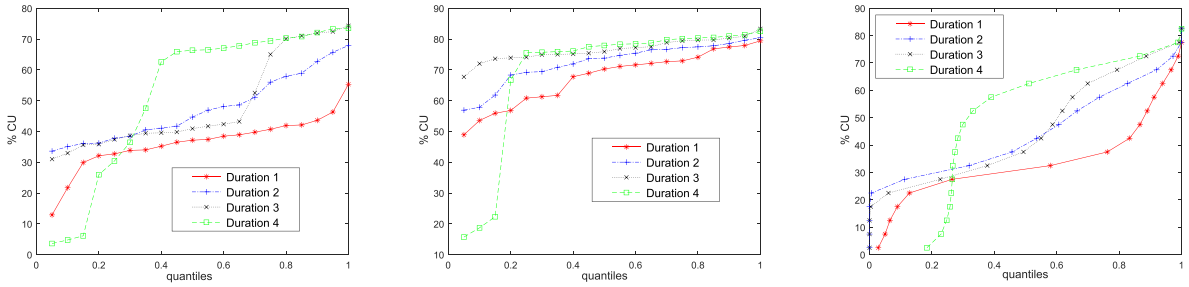
CU samples, block max CU samples, and CU samples from histograms, respectively. The three figures show quantiles for 20 evenly spaced cumulative probabilities from 0.05 to 1. The results in the three figures tell us that both block mean and block max sample data can be sometimes misleading. For example, block mean CU samples data for a streaming application like HD YouTube videos shows that the quantile for 0.8 probability is 15% or less, block max CU sample data shows that the quantile for 0.8 probability is 70% or less, and the block CU sample data shows that the quantile for 0.8 probability is only 3% or less. This means that if only block max sample data is used for CU analysis then one can be mislead to believe that the channel is highly utilized. However, in reality channel is mostly not used as shown by the quantile of block CU sample data. On the other hand, block mean CU sample data for the same streaming applications also show that quantile for 1 probability is 30% or less. However, both block max CU sample and block CU sample data shows that the quantile for the same probability is 88% or less. This means that if only block mean sample data is used for CU analysis then one can be mislead to believe that the channel is lightly utilized all the time. However, in reality although channel is mostly not used but when it is used it is utilized very highly. Moreover, for the streaming application case, quantiles in the three figures tell us that up to 0.8 probability, mean CU data is close to CU data and between 0.8 to 1 probabilities it is max CU data which is closer to CU data.

Our results in the three figures show that use of streaming application alone or its use with other applications (such as YouTube plus Skype) can lead to greater differences among mean CU, max CU and CU values for the same medium and/or high quantiles. However, interestingly this difference is significantly reduced for real-time applications. For example, the three figures show that for real-time applications (such as Skype audio plus video) the quantile for 0.8 probability values are 20% or less for the block mean, 25% or less for the block max and 21% for the block CU.

Overall, the results show that for high CU estimation mean CU measurements alone can be misleading because of uneven spread in CU values, and maximum measurements alone can be misleading when CU traffic is low but bursty. The CU samples obtained from collected histograms provide better accuracy in terms of distribution of values and hence can be more useful for estimation of high CU.

B. OTA HIGH CU MEASUREMENTS

We also performed a high CU stress test in which three laptops and four smart phones were used. The purpose of the test was to increase the number of simultaneous high CU applications (such as Skype video, HD YouTube, and File Download) on multiple devices to a point where application performance is degraded. This test allowed us to evaluate how the CU statistics behave under congested utilisation of a channel. The laptops and smart phones were connected using a 2.4 GHz channel to a WiFi AP. The implemented device was set to collect statistics in the same channel. A standard speed test software showed that the WiFi link had 54 Mbps upload/download speeds. The stress test measurements were performed for 40 minutes. In the first 30 minutes no degradation in performance was observed when all 7 devices were using skype video call session and three laptops were also running HD YouTube vidoes in parallel. In the last 10 minutes, when one laptop computer also started downloading a large file this started some degradation in the Skype video quality. In Figs. 9a, 9b and 9c, we present results that quantify CU behavior across various quantiles. In the three figures, we show quantiles of the block mean CU samples, block max CU samples, and block CU samples obtained from the histogram, respectively. The quantiles are computed for collected samples during every $I_l = 10$ minutes of time duration. Duration 1 samples show quantiles for the first ten minutes of the stress test, Duration 2 samples show quantiles for the second ten minutes of the stress test, and so on. It can be seen from the two figures Fig. 9a and Fig. 9c that up to the quantile for 0.9 probability, the mean CU values estimate accurately the CU, however, between the quantile for 0.9 probability and the quantile for 1 probability the mean CU values underestimate the CU. On the other hand, Fig. 9b and Fig. 9c show that until the quantile for 0.9 probability the max values can overestimate the CU, however, between the quantile for 0.9 probability and the quantile for 1 probability the max values estimate more accurately the CU as compared to the mean values. For the Duration 4, i.e., the last 10 minutes of the stress test in which service quality degraded



(a) Block mean (b) Block maximum (c) CU values from Histograms

FIGURE 9. High CU measurement results showing quantiles for the cumulative probabilities p in the interval (0,1].

(as discussed earlier), we can see from Fig. 9c that even the quantile for 0.4 probability can be as high as 62% CU.

VII. QUANTILE BASED CONGESTION DETECTION

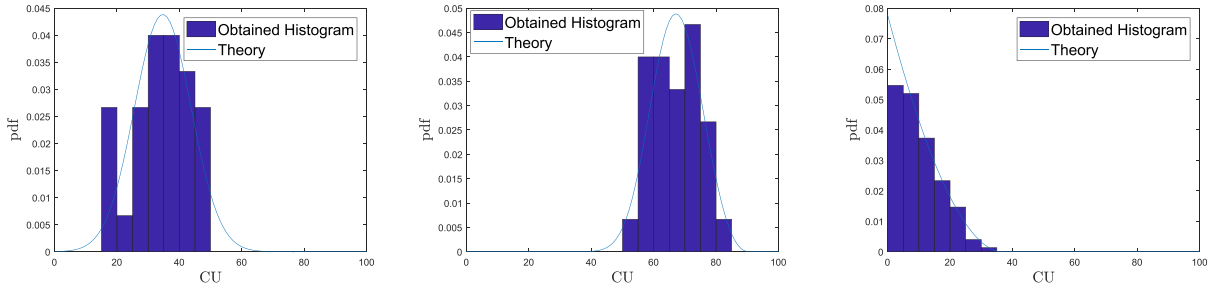
Our experiments presented in the previous sections allowed us to collect and analyse CU data under various scenarios. By understanding how CU behaves under various real/non-real time applications and different network conditions, we next present a quantile estimation based CU congestion detection method. The method utilizes the CU data from the implemented RA device. By detecting the signs of congestion in CU, a resource controller can avoid the wireless users being effected by it by adapting its resource allocation among multiple APs.

A. COLLECTED DATA AND VALIDITY OF DISTRIBUTION MODELS

Before we present the CU congestion detection method, it is important to present results which verify that quantiles of CU data of a real wireless network can be estimated well by using the statistical distribution models introduced in Section III. To achieve this, the implemented RA device was used to collect the OTA CU descriptive statistics over a period of two weeks in one of the busiest part of the University of Oulu. The device collected mean CU, maximum CU and CU values from histograms in a 2.4GHz WLAN channel where three APs were operating. The RA device was configured to output every 20 seconds a block mean, a block max value, and a CU histogram. Fig. 1 shows collected block mean and block maximum CU data values in percentage for six days. It can be seen from the figure that there is a daily pattern for weekdays (Feb 19-22) where there is high CU from 8am till 6 pm and there is less traffic usage in other times during the same week days. In general, there is less traffic utilization for all the times in weekend (Feb 17-18). The collected data was used to verify that its block mean CU values are well-modeled by a normal distribution, its block max CU values are well-modeled by a GEV distribution, and its CU values over threshold are well modeled by a GP distribution. In Fig. 10, we compare probability distribution functions (pdfs) approximated by sample histograms with theoretical pdfs. Fig. 10a, 10b and 10c, show the obtained histograms for the block mean, block max, and values over threshold data, respectively. The figures also show the overlay of the normal, GEV and GP distribution functions, respectively, computed for the same data from MATLAB. It can be seen from Figs. 10a and 10b that the normalized mean and max sample data histograms approximate the normal and GEV distributions, respectively, even when a small number of $10 \times 3 = 30$ samples are used. However, Fig. 10c shows that the normalized histogram approximates the GP distribution much better. This is due to the reason that the sample values over threshold histogram in Fig. 10c is constructed from a larger number of CU samples obtained from $10 \times 3 = 30$ CU histograms (with each histogram based on large number of samples). In the figures, as the area of pdf is summing to one so that the normalized histogram's integral over the range is 1. We have chosen in the figure bin width $= 5$. So sum of values of bars needs to be multiplied by 5 to get sum of the values of bars equal to 1. This scaling is necessary to be able to compare theoretical pdf with histogram.

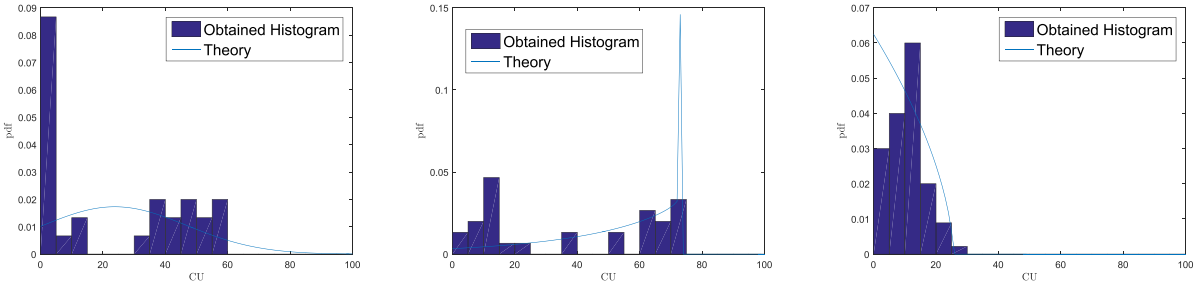
To provide a comparison, we have considered WARPLab framework of WARP which is a widely used SDR platform [5]. It has been widely used by research works (see [6]) to obtain various wireless network statistics, such as RSSI and CSI values. In WARPLab, on the FPGA part a baseband module buffers incoming IQ samples from an attached radio interface. The buffered samples are then transferred to the host computer via Ethernet, or USB communication interface where the IQ samples can be processed to obtain CU statistics. However, using buffers to store, transfer and then process on a host computer leads to large gaps in the collected samples. The size of these gaps depend on the limited size of buffer and also on the interface used between SDR and the computer for transferring the samples.

In Figs. 10 and 11, we compare the performance in terms of obtained data quality via WARP SDR platform with our implemented solution. As discussed in details in the previous paragraphs, Figs. 10a, 10b, and 10c show that the CU statistical data sets obtained via our implemented solution are well-modeled by theoretical distributions. For the same time duration, however, Figs. 11a, 11b, and 11c show that the CU statistical data collected using the WARP cannot be modeled by the distribution models. This is due to the presence of



(a) Block mean CU sample histogram and normal distribution (b) Block maximum CU sample histogram and GEV distribution (c) Block CU values over threshold histogram and GP distribution

FIGURE 10. Comparing sample histograms collected via our implemented solution with theoretical pdf.

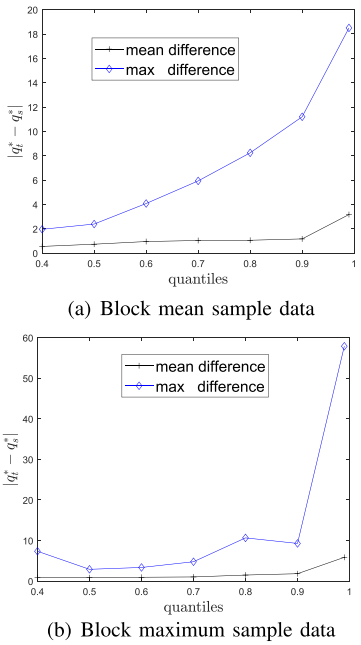


(a) Block mean CU sample histogram and normal distribution (b) Block maximum CU sample histogram and GEV distribution (c) Block CU values over threshold histogram and GP distribution

FIGURE 11. Comparing sample histograms collected via WARP platform with theoretical pdf.

gaps in the collected IQ data via WARP which leads to lower data quality. This means that due to the lower data quality obtained via WARP, the quantiles of CU data of a wireless network cannot be estimated well by using the statistical distribution models. Hence, unlike our implemented solution, typical SDR-based approaches cannot be utilized for reliable real time RF data analytics.

In Figs. 12a, and 12b, we present absolute difference values between quantiles estimated using sample data q_s^* and quantiles estimated using closed form expressions q_t^* given in Section III, (see (4), and (6)). The results show that the mean absolute difference $|q_t^* - q_s^*|$ is less than 4 for block mean data and it is no more than 5 for block max data. Moreover, as expected, the maximum absolute difference results show that for higher values there can be more difference between estimates obtained via sample quantiles and the closed form expressions. This is a well known statistical feature due to the reason that sample data can be sparser at the higher end of the distribution, so modelling extreme quantiles will have lower associated precision. However, the quantile estimates obtained via closed form equations can be used to provide more precise estimates than the sample data quantiles.



(a) Block mean sample data

(b) Block maximum sample data

FIGURE 12. Absolute difference values between quantiles estimated using sample data and quantiles estimated using theory (Equations (4), and (6), respectively).

B. PROPOSED METHOD

The crux of the proposed method is to use the CU values obtained from the most recent $|\mathcal{H}|$ histograms sent by the RA device. First, the median quantile estimate q_τ^* of the CU data values is calculated and compared against a predefined threshold CU value α_t. Second, when $q_\tau^* > \alpha_t$ then the values over threshold technique (presented in Section III)

is used to estimate high quantile q_h^* of CU data. If the estimated high quantile q_h^* exceeds certain predefined high CU value α_t^h then the resource allocation optimization should be triggered. Steps involved in the method are presented in Algorithm 1.

Algorithm 1 Quantile Estimation Based Congestion Detection

Initialize: Threshold quantile q_τ, high quantile q_h^*, threshold CU value α_t, and high CU α_t^h
Input: $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, \cdots, \mathcal{H}_N\}$ histograms collected in time duration I_l
Obtain CU value samples $\{x_1, x_2, \cdots\}$ from collected $|\mathcal{H}|$ histograms.
if $q_\tau^* \leq \alpha_t$ **then**
  Go to **Input**
else
  Fit the GP distribution to the CU samples that exceed the α_t and estimate parameters ξ and $\breve{\mu}$.
  Estimate the high quantile q_h^* using (9)
  if $q_h^* > \alpha_t^h$ **then**
    Channel likely to be under congestion
    Optimize resource allocation
    Go to Input
  else
    Channel likely not under congestion
    Go to Input
  end if
end if

We observe from our extensive measurements (see for example results in Fig. 9c) that the value of median quantile to be 50 percent or more is a first good indicator of very high CU. Hence, we suggest to use these values, i.e., median quantile and $\alpha_t = 50$, as a trigger to initiate the value over threshold technique to estimate high quantile q_h^*. From our measurements with real applications, such as Skype video, YouTube HD, and large file downloads, we also observe that when the high quantile, such as 0.7 quantile, is greater than 65 then the likelihood of CU getting congested increases and the resource controller should optimize its allocation to avoid this. Note that the suggested use of 0.7 quantile and $\alpha_t^h = 65$ are just an example. In our work, we set the time duration $I_l = 10$ minutes. This allowed us to use large number of CU samples from $10 \times 3 = 30$ histograms for the quantiles shown in Fig. 9c. In general, the choice of I_l and also the choice of a high quantile to perform optimization in resource allocation is a design parameter.

VIII. CONCLUDING REMARKS

We present design and implementation of a radio frequency (RF) analytics system which enables collection/processing of radio environment statistics in real-time. The analytics system consists of hardware accelerated data collection/processing modules and a resource controller server. The hardware accelerated modules are implemented on ZedBoards which are system on chip (SoC) devices, where each device is equipped with a Xilinx Zynq-7000 Field Programmable Gate Array (FPGA) and dual core embedded ARM processor. Each device acts directly in real-time on streaming IQ samples and processes them to obtain low-overhead descriptive statistics, such as mean, maximum, and histogram of wireless channel utilization (CU) values. The collected statistics are sent as low-overhead data (16-bit values) to a resource controller server. The resource controller uses CU statistics, their quantile estimates, and a quantile based congestion detection method to proactively control congestion in CU.

We provide extensive measurement results that evaluate the real-time performance of the implemented measurement/analytics algorithms on the ZedBoards under three different setups: 1) laboratory measurements/testing; 2) over the air (OTA) measurements/testing in an anechoic chamber using various video/audio/real-time applications running on wireless devices; and 3) measurements/test in a real wireless network in which we collected measurements data for two weeks. Our results show that the implemented RF analytics system can help fulfill the 5G and beyond vision of real-time analytics driven proactive resource allocation for wireless networks. Moreover, proactive detection of congestion point in wireless CU can also help in ensuring better co-existence of multiple wireless systems operating in a same spectrum band.

ACKNOWLEDGMENT

The authors would like to thank Xilinx University Program (XUP) for providing us a donation of four Zedboards which were used in this work for implementing the proposed RA device.

REFERENCES

[1] "The status of open source for 5G," 5G Americas, Bellevue, WA, USA, White Paper, Feb. 2019. [Online]. Available: http://www.5gamericas.org/files/6915/5070/2509/5G_Americas_White_Paper_The_Status_of_Open_Source_for_5G_Feb_2018.pdf
[2] G. Pocovi, H. Shariatmadari, G. Berardinelli, K. Pedersen, J. Steiner, and Z. Li, "Achieving ultra–reliable low–latency communications: Challenges and envisioned system enhancements," *IEEE Netw.*, vol. 32, no. 2, pp. 8–15, Mar. 2018.
[3] T. Cooklev, J. Darabi, C. Mcintosh, and M. Mosaheb, "A cloud-based approach to spectrum monitoring," *IEEE Instrum. Meas. Mag.*, vol. 18, no. 2, pp. 33–37, Apr. 2015.
[4] Cisco, "Meraki cloud controller," Cisco Syst., San Jose, CA, USA, Tech. Rep., Mar. 2017. [Online]. Available: https://documentation.meraki.com/Architectures_and_Best_Practices/Cisco_Meraki_Best_Practice_Design/Meraki_Cloud_Architecture
[5] *Rice University WARP project*. [Online]. Available: https://warpproject.org/trac/wiki/PapersandPresentations
[6] J. Tadrous and A. Sabharwal, "Interactive app traffic: An action-based model and data-driven analysis," in *Proc. 14th Int. Symp. Modeling Optim. Mobile, Ad Hoc, Wireless Networks (WiOpt)*, May 2016, pp. 1–8.
[7] P. Jiang, J. Winkley, C. Zhao, R. Munnoch, G. Min, and L. T. Yang, "An intelligent information forwarder for healthcare big data systems with distributed wearable sensors," *IEEE Syst. J.*, vol. 10, no. 3, pp. 1147–1159, Sep. 2016.
[8] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in *Proc. IEEE INFOCOM IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.

[9] L.-C. Wang and S.-H. Cheng, "Data–driven resource management for ultra–dense small cells: An affinity propagation clustering approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 267–279, Jul. 2019.
[10] "Network data analytics services," Eur. Telecommun. Standards Inst., Sophia Antipolis, France, Tech. Rep. ETSI TS 129 520, 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/129500_129599/129520/15.00.00_60/ts_129520v150000p.pdf
[11] J. Perez-Romero, A. Zalonis, L. Boukhatem, A. Kliks, K. Koutlia, N. Dimitriou, and R. Kurda, "On the use of radio environment maps for interference management in heterogeneous networks," *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 184–191, Aug. 2015.
[12] Z. Khan, J. J. Lehtomaki, E. Hossain, M. Latva-Aho, and A. Marshall, "An FPGA–based implementation of a multifunction environment sensing device for shared access with rotating radars," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 11, pp. 2561–2578, Nov. 2018.
[13] A. Kaloxylos, "Application of data mining in the 5G network architecture," in *Proc. 13th Int. Conf. Digit. Telecommun. (ICDT)*, Mar. 2018, pp. 1–6.
[14] D. F. Rueda, D. Vergara, and D. Reniz, "Big data streaming analytics for QoE monitoring in mobile networks: A practical approach," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1992–1997.
[15] M. Kulin, T. Kazaz, I. Moerman, and E. De Poorter, "End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications," *IEEE Access*, vol. 6, pp. 18484–18501, 2018.
[16] J. Shen, J. Cao, X. Liu, and C. Zhang, "DMAD: Data–driven measuring of Wi–Fi access point deployment in urban spaces," *TISTACM Trans. Intell. Syst. Technol.*, vol. 9, no. 1, pp. 1–29, Aug. 2017, doi: 10.1145/3065949.
[17] A. Hernandez, E. Garcia, D. Gualda, J. M. Villadangos, F. Nombela, and J. Urena, "FPGA–based architecture for managing ultrasonic beacons in a local positioning system," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 8, pp. 1954–1964, Aug. 2017.
[18] M. Bassoli, V. Bianchi, I. De Munari, and P. Ciampolini, "An IoT approach for an AAL Wi-Fi-based monitoring system," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 12, pp. 3200–3209, Dec. 2017.
[19] C. Khona, "Key attributes of an intelligent IIoT edge platform," Xilinx, San Jose, CA, USA, White Paper, Sep. 2017. [Online]. Available: https://www.xilinx.com/support/documentation/white_papers/wp493-iiot-edge-platforms.pdf
[20] T. Hayajneh, S. Ullah, B. J. Mohd, and K. S. Balagani, "An enhanced WLAN security system with FPGA implementation for multimedia applications," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2536–2545, Dec. 2017.
[21] V. Kumar, M. Mukherjee, and J. Lloret, "Reconfigurable architecture of UFMC transmitter for 5G and its FPGA prototype," *IEEE Syst. J.*, to be published.
[22] F. Restuccia and T. Melodia, "Big data goes small: Real-time spectrum-driven embedded wireless networking through deep learning in the RF loop," *CoRR*, vol. abs/1903.05460, 2019. [Online]. Available: http://arxiv.org/abs/1903.05460
[23] M. S. Murty and R. Shrestha, "Hardware implementation and VLSI design of spectrum sensor for next-generation LTE—A cognitive-radio wireless network," *IET Circuits, Devices Syst.*, vol. 12, no. 5, pp. 542–550, Sep. 2018.
[24] S. Coles, J. Bawa, L. Trenner, and P. Dorazio, *An Introduction to Statistical Modeling of Extreme Values* (Lecture Notes in Control and Information Sciences). Springer, 2001. [Online]. Available: https://books.google.fi/books?id=2nugUEaKqFEC
[25] "Zynq-7000 SoC technical reference manual," Xilinx, San Jose, CA, USA, Tech. Rep. UG585 (v1.12.2), Jul. 2018. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf
[26] Analog Devices. (2018). *FMCOMMS2 No-OS Project, GitHub Repository*. [Online]. Available: https://github.com/analogdevicesinc/no-OS/tree/master/ad9361

ZAHEER KHAN received the Dr.Sc. degree in electrical engineering from the University of Oulu, Finland, and the M.Sc. degree in electrical engineering from University College Borås, Sweden, in 2011 and 2007, respectively. He is currently an Adjunct Professor with the University of Oulu, Finland. He was a Tenure Track Lecturer with the University of Liverpool, U.K., from 2016 to 2017, and a Research Fellow/Principal Investigator with the University of Oulu, from 2011 to 2016. He was a recipient of the Marie Curie Fellowship, from 2007 to 2008. His research interests include the implementation of advanced signal processing and wireless communications algorithms on Xilinx FPGAs and Zynq system-on-chip (SoC) boards, applications of game theory to model distributed wireless networks, prototyping access protocols for wireless networks, the IoT location tracking systems, cognitive and cooperative communications, and wireless signal design.

JANNE J. LEHTOMÄKI received the Ph.D. degree from the University of Oulu, Finland, in 2005. He is currently an Adjunct Professor with the Centre for Wireless Communications, University of Oulu. He spent the fall 2013 semester with Georgia Tech, Atlanta, USA, as a Visiting Scholar. He is currently focusing on spectrum measurements and terahertz band wireless communication. He was the General Co-Chair of the IEEE WCNC 2017 International Workshop on Smart Spectrum, the TPC Co-Chair of the IEEE WCNC 2015 and 2016 International Workshop on Smart Spectrum, and the Publicity/Publications Co-Chair of ACM NANOCOM, from 2015 to 2017. He has served as a Guest Associate Editor for the *IEICE Transactions on Communications* Special Section, from February 2014 to July 2017, and the Managing Guest Editor for *Nano Communication Networks* Special Issue, in June 2016. He has coauthored the paper receiving the Best Paper Award from the IEEE WCNC 2012. He is an Editorial Board Member of *Physical Communication*.