USRP-based platform for 26/28 GHz mmWave Experimentation

Martin Danneberg, Roberto Bomfin, Ahmad Nimr, Zhongju Li, Gerhard Fettweis
Vodafone Chair Mobile Communication Systems, Technische Universität Dresden, Germany
{first name.last name}@ifn.et.tu-dresden.de

***Abstract*—This paper presents an platform for 26/28GHz millimeter wave (mmWave) experimentation. We introduce an analog mmWave multi-beam antenna array, developed at TU-Dresden, that can be interfaced with regular software-defined radio (SDR) platforms. The motivation of this setup is to enable mmWave experimentation with a relatively simple system setup. As a demonstration example, we combine the mmWave frontends with the GFDM transceiver to provide a communication link at 26GHz. Moreover, considering mobility, we implement a real-time beam tracking algorithm that switches the beam according to the user's location. The performance is evaluated in terms throughput, where we demonstrate transmissions up to 3.1Mbits/s.**

***Index Terms*—mmWave, SDR, PHY, Measurements, Evaluation.**

I. Introduction

One very important novelty of the fifth generation (5G) of mobile communication systems is the employment of millimeter wave (mmWave) bands. The advantages of this new technology are key to improve the network capacity in 5G. For instance, the new bands provide users with higher data-rates, because it allows the use of larger bandwidth than the conventional sub-6GHz spectrum. In addition, due to high path-loss in the mmWave range, it is necessary to use very directive antennas in order to deliver a signal with enough signal power at the receiver. Thus, the transmitted signal needs to have energy concentrated in the direction of a given user [1], [2]. This allows spatial signal separation with minimal interference of users that are relatively close to each other with minimal interference. As a result, the spectrum can be reused more efficiently.

On the other hand, the mmWave technology brings also major challenges for the research community and industry. In particular, the deployment of directive antennas requires beam steering to direct the signal towards users, and tracking algorithms under mobility conditions, which increases the system complexity [3]. From the experimental viewpoint, the hardware platforms for mmWave are usually sophisticated and expensive, because they are tuned to provide high data rates of $1\,\mathrm{GBits}$ and higher [4], which certainly increases the overall complexity of the transceiver systems rather than providing a basic platform for the practical evaluation of beam steering algorithms. Here, we want to highlight that for the sake of investigation of intelligent beam steering algorithms, the data rates is not decisive.

This work is supported by *National Instruments (NI)* with hardware, software and technical support. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732174 (ORCA).

Therefore, we propose a hardware setup which combines conventional Universal Software Radio Peripheral (USRP) platforms with mmWave frontends [5]. The main motivation for this setup is to allow an easier experimentation platform for mmWave bands. Since the mmWave antennas can be attached to any software-defined radio (SDR), it does not require expensive hardware. Particularly, this allows to use software frameworks such as LabVIEW Application Frameworks or OpenAirInterface in the 26/28GHz frequency range using standard SDRs and further enabling the development of mmWave beam steering algorithms in a simpler and accessible manner. As an example of how this setup can be used, we have combined our flexible real-time physical layer (PHY) based on the generalized frequency division multiplexing (GFDM) [6] with the mmWave antennas [5]. This implementation includes a real-time beam tracking algorithm based on a basic medium access control (MAC) mechanism. Then, we performed throughput measurements, in which we obtained transmission data rates of up to $3.1\,\mathrm{Mbits/s}$. Additionally, we also compared the throughput with a Windows based control PC against a Linux real-time controller. The results showed that the real-time host provides considerably more data rate than the PC-based controller.

The following sections give a detailed description of the platform. First, in Section II, the algorithms of the overall system are presented. In section III, we detail the hardware set-up. A evaluation of the experiments is given in Section IV. Finally, we conclude the paper in Section V.

II. System model

The presented signal processing blocks are implemented on the field programmable gate array (FPGA) of the SDR to allow the PHY layer to operate in real-time. The coarsely described processing blocks are based on the GFDM waveform framework [7], which can create a multitude of waveforms as mentioned in [6]. An overview for all signal processing blocks is given in Fig. 1.

A. Physical Layer Implementation

First, the payload is transferred from a host computer to the transceiver implementation, which is running on FPGA via a minimal MAC interface. This is responsible for adding

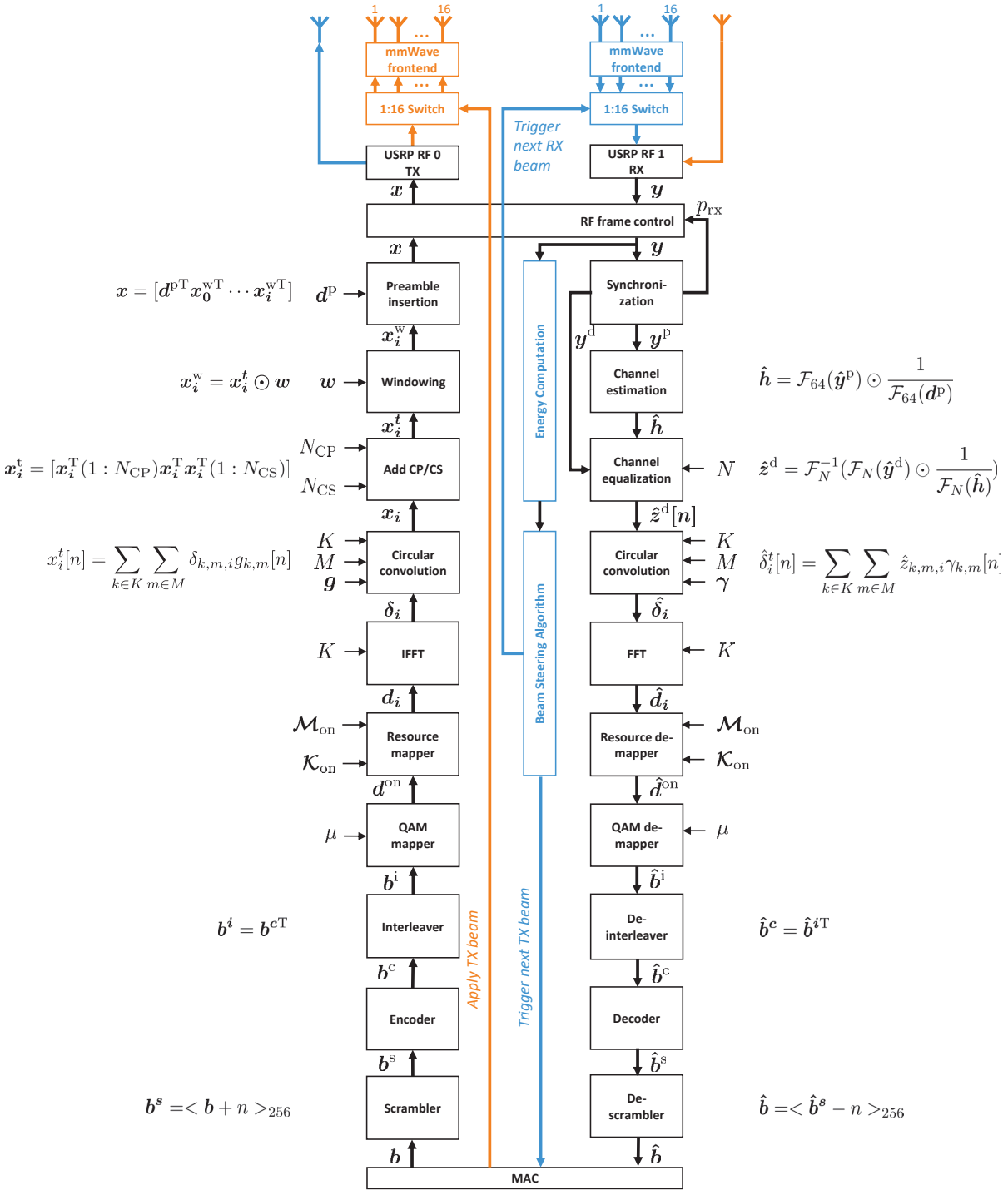

Fig. 1: The signal processing blocks implemented on the FPGA. The red color indicates the transmitter and the blue color the receiver.

a cyclic redundancy check (CRC) to the payload and adding a MAC header.

The binary data $\boldsymbol{b}$ is fed to the PHY signal processing byte-wise. Firstly, to a scrambler to avoid repetitive patterns inside the payload, which could lead to unnecessary peaks in the transmitted signal. The channel coding follows, which is realized by a convolutional code with code rate $\frac{1}{2}$. The coded bits $\boldsymbol{b}^{\mathrm{c}}$ from the coder are interleaved by a transpose function to fully utilize the redundancy introduced by the coding. Here, the data $\boldsymbol{b}^{\mathrm{c}}$ is written row-wise into a Matrix like memory and read out column by column. Depending on the selected modulation order μ several bits $\boldsymbol{b}^{\mathrm{i}}$ are grouped into one QAM

symbol. Afterwards, the symbols $\boldsymbol{d}^{\text{on}}$ need to be mapped into the resource grid, defined by the amount of active subcarriers $\boldsymbol{\mathcal{K}}_{\text{on}}$ and subsymboles $\boldsymbol{\mathcal{M}}_{\text{on}}$. Furthermore, additional symbols such as pilots can be inserted by the resource mapper into the final symbol stream $\boldsymbol{d}_i$ of length N. According to the parameter B several data symbols indexed via i are created for one frame.

As any other orthogonal frequency division multiplexing (OFDM)-based multicarrier transmission system, the symbols stream $\boldsymbol{d}_i$ is organized in frequency domain such that each symbol $d_{i,n}$ is mapped to a certain subcarrier k. However, to transfer the data over a wireless channel the symbols need to be mapped to time-domain $\boldsymbol{\delta}_i$ which is conducted by the inverse discrete Fourier transform (IDFT) of size K. Afterwards, the GFDM specific signal processing is conducted using a circular convolution. To prepare the signals for the transmission process, a cyclic prefix (CP) with the length N_{CP} and, depending on the configuration, a cyclic suffix (CS) of length N_{CS} needs to be added for each GFDM block $\boldsymbol{x}_i$. As already indicated in the pre-standard definition of 802.11a, windowing can suppress the out-of-band (OOB) emissions severely by multiplying an window $\boldsymbol{w}$ element wise with the data $\boldsymbol{x}_{\boldsymbol{i}}^{\text{t}}$. The resulting signal $\boldsymbol{x}$ is forwarded to the radio-frequency (RF) frame control logic, which is responsible of ensuring that the transmission of the packets is restricted to a predefined time-grid.

At the receiver, a synchronization algorithm for a GFDM based system is discussed in details in [8] and introduced here briefly for reference. The GFDM preamble block is composed of $M_{\text{p}} = 2$ subsymbols and $K_{\text{p}} = 128$ subcarriers. A PN sequence $\boldsymbol{c} = (c[0], \ldots, c[K_{\text{p}} - 1])\, T$ of length K_{p} is transmitted twice inside the preamble. The receiver performs the auto-correlation with the received signal, a cross-correlation with the pre-stored preamble and finally combines both results to search for the main peak. Further, the synchronization module splits the data stream into the received data frame $\boldsymbol{y}^{\text{d}}$ for the channel equalizer and into the received preamble $\boldsymbol{y}^{\text{p}}$.

The channel estimation module calculates the channel impulse response (CIR) using the preamble $\boldsymbol{y}^{\text{p}}$ to provide the inverse channel for equalization. The received data $\boldsymbol{y}^{\text{d}}$ after removing the CP and CS is equalized using a zero-forcing (ZF) algorithm.

The demodulator at the receiver side performs the inverse operations of the modulator, where a circular convolution with the receive filter $\gamma[n]$ and a discrete Fourier transform (DFT) is applied to get the received data symbol $\hat{\boldsymbol{d}}_i$. The symbols are then forwarded to the resource de-mapper, which separates the different data streams, such as pilot or control channel data from the payload $\hat{\boldsymbol{d}}^{\text{on}}$.

The QAM demapper assigns a matching bit pattern $\hat{\boldsymbol{b}}^{\text{i}}$ to the incoming symbols based on the modulation scheme μ using hard-decision. The de-interleaver performs the inverse of interleaver and provides the bits $\hat{\boldsymbol{b}}^{\text{c}}$ to the decoder. The decoder performs Viterbi algorithm to decode the received byte stream $\hat{\boldsymbol{b}}^{\text{s}}$. Finally, after the scrambler the received packet $\hat{\boldsymbol{b}}$ is created.

Parameter	Symbol	Value
Subcarriers	K	64
Active Subcarrier	$\boldsymbol{\mathcal{K}}_{\text{on}}$	4 - 28, 35 - 60
Subsymbols	M	15
Active Subsymbols	$\boldsymbol{\mathcal{M}}_{\text{on}}$	1 - 14
Number of blocks	B	2
Payload size [Byte]		160
Roll-of	α	0
Filter Type	raised-cosine	
Receive filter Type	zero-forcing	
Cyclic Prefix	N_{CP}	32
Cyclic Suffix	N_{CS}	16
QAM Mapping	QPSK	
Coding	Convolutional code with rate 1/2	
Preamble	Zadoff-Chu sequence	
Preamble length		128

TABLE I: GFDM parameters similar to the experiments in [9]

B. Frame Format

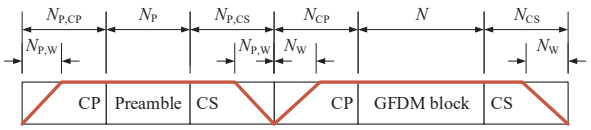

Fig. 2: Considered frame structure with one preamble and one data block. [6]

Fig. 2 depicts the general GFDM frame format, where CP and CS can be applied on both preamble and data block. Here, $N_{\text{P,CP}}$ and $N_{\text{P,CS}}$ define the preamble CP and CS lengths, respectively, whereas N_{CP} and N_{CS} denote the CP and CS lengths in the data block. It is assumed that the window, colored in red in the figure, is symmetric, thus the length of one half is given by $N_{\text{P,W}}$ and N_{W}. The block length is denoted as N and the preamble length as N_{P}. The frame consists on one preamble and B GFDM data blocks to fit the payload size.

C. Control Channel

The mobile user equipment (UE) uses the mmWave band for the uplink transmission. In Fig. 1 the UE is marked with a red color, the access point (AP) in blue. A downlink control channel is used for sending control information from the AP to the UE using a 3.75GHz link. The main objective of the control channel is to send the information to the UE about the transmitter beam. More specifically, the AP realizes the beam steering algorithm in a centralized manner, and then it informs the UE which beam it should use in the uplink. For the control channel PHY configuration, we use a regular GFDM PHY frame, which is the same for the uplink. In this work, there are 16 possible beams, which means that we need to send 4 bits of information to the UE. Still, the control channel could be further optimized to send less data.

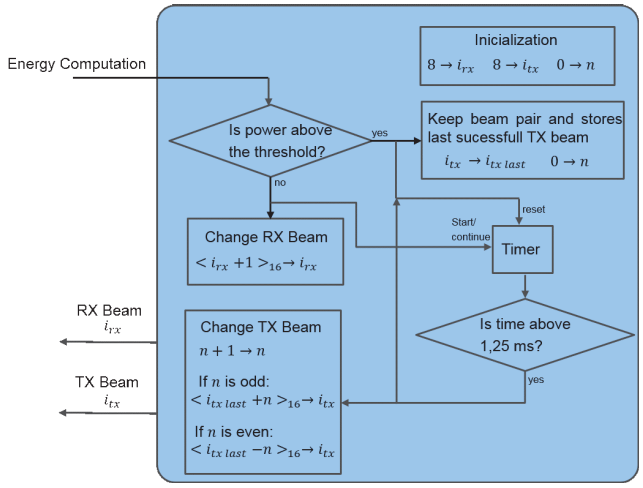

Fig. 3: Beam steering algorithm on FPGA.

D. Beam tracking algorithm

The top part of Fig. 1, presents the diagram of the real-time system implemented on the FPGA. For the beam tracking algorithm, we firstly perform an energy computation of a given amount of samples. This step is performed in the block *Energy Computation* in Fig. 1. For the experiments presented in this paper, we configured the system to average 2500 samples, which corresponds to $62.5\,\mu$s at $40\,$MHz clock rate. This is the necessary time to probe one beam combination. For simplicity, we perform an exhaustive search approach for the receiver part of the algorithm. This can be observed in Fig. 3 in the sub-block *Change RX Beam*, where the RX beam i_{rx} is changed sequentially, when the power/energy is not above the threshold. That is, if a give channel provides energy above a certain threshold, then this beam pair is kept until the channel changes. For the RX part of the algorithm, we compute that it takes $16\times\ 62.5\,\mu\text{s} = 1\,\text{ms}$ to cover all beams. In order to keep a reasonable time margin, the algorithm waits for $1.25\,\text{ms}$ to change the TX beam via the control channel. To avoid the time consuming exhaustive search approach for the TX, the algorithm probes the neighboring beams of the last successful TX beam. This process is also shown in Fig. 3. For instance, when all RX beams do not provide a reasonable channel, the TX beam is changed to $< t_{\text{tx,last}} + n >_{16}$ or $< t_{\text{tx,last}} - n >_{16}$, if the iteration step n is odd or even, respectively. Therefore, considering that the mobile UE moves slow enough such that the next successful TX beam is a neighbor beam of the last successful beam, the overall searching time of the algorithm is $2\times\ 1.25\,\text{ms} = 2.5\,\text{ms}$.

III. Experimental Setup

The presented PHY and MAC layer implementation is implemented using the *National Instruments (NI)* LabVIEW Communications System Design Suite and the entire project can be downloaded [10]. Therefore, the widely known USRP-RIO series from *NI* is chosen as SDR. Like the *Ettus* USRP X310, a *Xilinx* KINTEX 7 FPGA is connected to two independent, broadband radios. A control PC can be connected by means of PCI Express extension. A new generation under the brand name USRP 2974 provided by *NI* includes an embedded PC attached to a USRP X310 in a standalone device. In the experiment we will compare both versions with each other. Fig. 4 drafts the used hardware set-up for the evaluation.

In the experiment, test UDP packets are created by an application running on the left PC. A $1\,$GBit network switch routes the data to the first USRP. Here, the UDP payload is encapsulated to data format used by the transmitter. After the digital signal processing, the baseband signal is shifted to the intermediate frequency of $2.4\,$GHz by the USRP analog frontend. The mmWave frontend, shown in Fig. 5, has 16 inputs which are corresponding to 16 independent beams. Since only a single beam in a single direction is going to be radiated, a 1 to 16 RF switch connects the USRP TX output to the input of the mmWave frontend. The FPGA then selects the respective port and therefore the beam via a separate TTL signal.

As described in [11], the $2.4\,$GHz signal is up-converted to $26\,$GHz before entering the passive Butler matrix beam-forming network. The Butler matrix routes the inputs similar to a Butterfly graph for a fast Fourier transform (FFT) algorithm through several passive delays and attenuators to form the individual beams. The power amplifiers (PAs) amplify the signal, before the quasi-yagi antennas radiate it. Although the Butler matrix can operate bidirectional the amplifiers fix the frontend either to transmitter or receiver. Thus, the mmWave receiver frontend first amplifies the signal on each antenna via low noise amplifiers (LNAs), and feed them into the Butler matrix and down-converts the output to $2.4\,$GHz. Fig. 5 illustrates the developed frontend. The operating frequency ranges from $26\,$GHz to $30\,$GHz. This covers the 5G NR frequencies for Europe ($26\,$GHz) as well as for US and Asia ($28\,$GHz).

Before the experiment is conducted, a *Rohde & Schwarz (R&S)* signal generator is connected at the transmitter instead of the USRP and the receiver is a *R&S* $40\,$GHz spectrum analyzer with an attached horn antenna. This way the transmit power is calibrated, because two different software implementations are used for the evaluation. Besides the described PHY layer, a channel sounder presented in [12] is used to measure the equivalent channel seen by the transceiver implementation.

IV. Evaluation

The presented set-up is evaluated in several aspects. First, we conduct a power calibration to ensure that the power levels of a *R&S* signal generator, our channel sounder and the GFDM transceiver are aligned with the help of a *R&S* $40\,$GHz spectrum analyzer. The maximum allowed input power for the mmWave frontend is $0\,$dBm. To leave enough head room, we select the input power at the input of the switch to be $0\,$dBm, resulting in a power level of $-6\,$dBm at the input of the front-end, due to the losses inside the RF switch.

In the first set of experiments, the distance between transmitter and receiver is kept at $2.5\,$m. Here we evaluate the CIRs, depicted in Fig. 6. It is observed that although the mmWave

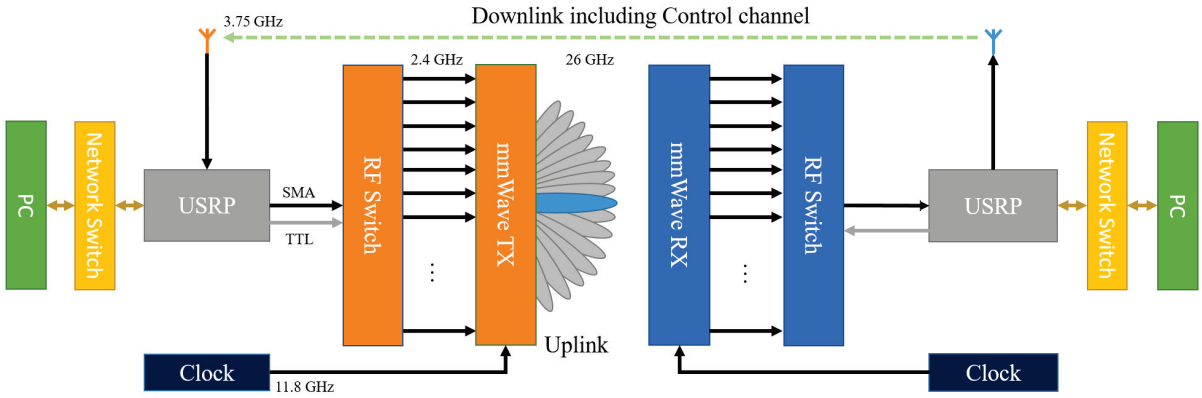

Fig. 4: Experimental set-up.

Fig. 5: Picture of the used mmWave frontend.

frontend is attached, the resulting channel seen by the SDR platform is reasonably flat.

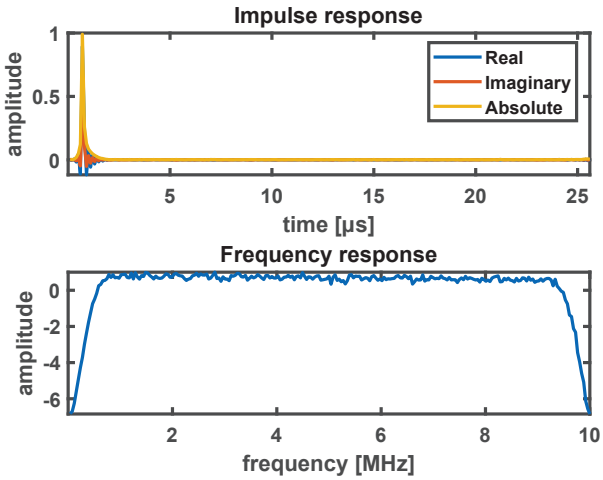

Fig. 6: Channel impulse and frequency response

Afterwards, we moved the transmitter parallel to the receiver to investigate the narrowness of a single beam. Fig. 7 shows that the half-power beam-width given in [5] of 7° matches approximately the results. The signal to noise ratio (SNR) and measured power at the spectrum analyzer is decreasing by 4 dB at 15 cm distance from the starting point. This coarsely corresponds to the beam width in 2.5 m distance, where $\arctan(\frac{15}{250}) \approx 3.5\,°$.

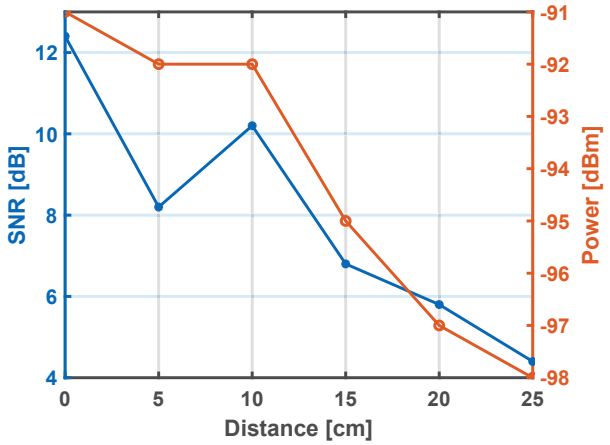

Fig. 7: Evaluated beamwidth: The SNR is measured using the channel sounder, the power is measured using a spectrum analyzer.

The last experiment in this paper explores the capabilities of the beam-steering algorithm under two different walking-speeds and using a X310 based set-up with a control PCs running Windows against the USRP 2974 running the *NI* Real-Time Linux. The PHY parameters for the GFDM transceiver implementation are given in Table I. The data-rate is measured with UDP packets using a UDP payload size of 160 Bytes. Fig. 8 shows the achieved data rates with the set-up. The UE moves parallel to the receiver, i.e. the angle changes, with moving speeds around 1.1 km/h and 2.2 km/h starting from a 0° angle until the receiver cannot detect any data. The experiment shows, that compared to the static case at 0° the data rate drops with the increasing speed. However, although most of signal processing is moved to the FPGA, the operating system has a influence on the performance of the system. The control PC running Windows drops many UDP packets, such

that the performance affected by the moving transmitter is less significantly than for the Linux Real-Time system. Although, the control PC running Windows has not the same CPU as the USRP 2974, Windows and the running LabVIEW environment are perturbing severely the experiment.

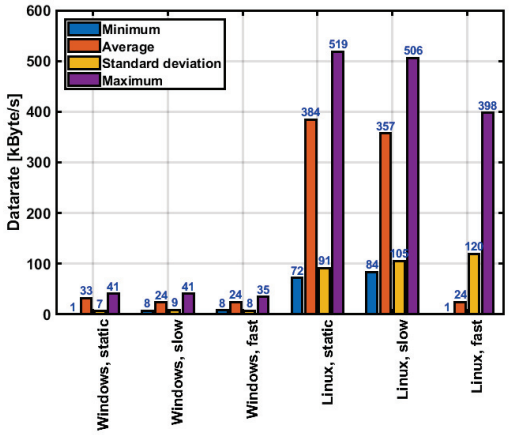

Fig. 8: Achieved data-rate

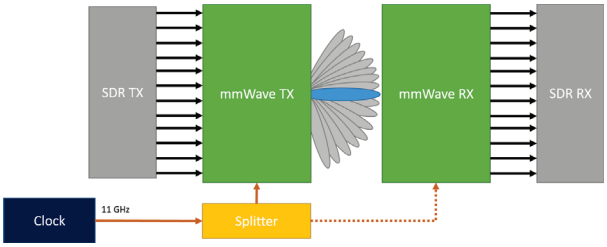

Fig. 9: Hardware set-up using the *NI* Massive MIMO Framework. The clock signal for the mmWave Frontend can either be shared as in the figure or separated like in the experiment.

V. Conclusion

This work presented an experimental mmWave platform using the popular USRP SDR to simplify research on smart spectrum usage in mmWave bands, because unnecessary computational overhead due to the typical large bandwidths can be omitted. The hardware platform has been integrated with a real-time end-to-end PHY implementation including beam tracking, which enables the evaluation of algorithms for beam-steering and thus spatial reuse for different users. As results, we showed that the real-time PHY successfully works with the beam tracking, where a throughput of $2.9\,\mathrm{Mbits/s}$ is achieved with mobility.

As future work, many extensions are targeted. For instance, we plan to extend the single-user setup with a multiple antenna system by attaching multiple USRPs instead of a single one and the 1 to 16 RF switch as shown in Fig. 9. Because in theory, using a passive Butler matrix, each input and the respective beam are independent from each other, such that multi-user spatial multiplexing can be explored. Here, the *NI* Massive MIMO software could serve as a starting point. Additionally, the evaluation and optimization of more efficient beam tracking algorithms are also planned to be evaluated.

References

[1] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter Wave Mobile Communications for 5G Cellular: It Will Work!" *IEEE Access*, vol. 1, pp. 335–349, 2013.

[2] Z. Pi and F. Khan, "An introduction to millimeter-wave mobile broadband systems," *IEEE Commun. Mag.*, vol. 49, no. 6, pp. 101–107, June 2011.

[3] T. Obara, T. Okuyama, Y. Aoki, S. Suyama, J. Lee, and Y. Okumura, "Indoor and outdoor experimental trials in 28-GHz band for 5G wireless communication systems," in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Aug 2015, pp. 846–850.

[4] T. Kadur, H. Chiang, and G. Fettweis, "Experimental Validation of Robust Beam Tracking in a NLoS Indoor Environment," in *2018 25th International Conference on Telecommunications (ICT)*, June 2018, pp. 644–648.

[5] X. Wang, M. Laabs, D. Plettemeier, K. Kosaka, and Y. Matsunaga, "28 ghz multi-beam antenna array based on wideband high-dimension 16x16 butler matrix," in *2019 13th European Conference on Antennas and Propagation (EuCAP)*, March 2019, pp. 1–4.

[6] M. Danneberg *et al.*, "Universal waveforms processor," in *EuCNC 2018,*, pp. 357–362.

[7] N. Michailow *et al.*, "Generalized frequency division multiplexing for 5th generation cellular networks," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3045–3061, Sep 2014.

[8] I. S. Gaspar, L. L. Mendes, N. Michailow, and G. Fettweis, "A synchronization technique for generalized frequency division multiplexing," *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 1, p. 67, May 2014. [Online]. Available: http://dx.doi.org/10.1186/1687-6180-2014-67

[9] M. Danneberg, Z. Li, P. KÃ¼hne, A. Nimr, S. Ehsanfar, M. Chafii, and G. Fettweis, "Real-time waveform prototyping," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, July 2019, pp. 1–5.

[10] M. Danneberg, A. Nimr, M. Matthe, A. C. Ferreira, Z. Lin, P. Kühne, S. Ehsanfar, N. Michailow, A.-B. Martinez, D. Zhang, and G. Fettweis, "Flexible transceiver implementation." [Online]. Available: http://owl.ifn.et.tu-dresden.de/GFDM/

[11] X. Wang, M. Laabs, D. Plettermeier, K. Kosaka, and Y. Matsunaga, "Mimo antenna array system with integrated 16x16 butler matrix and power amplifiers for 28ghz wireless communication," in *2019 12th German Microwave Conference (GeMiC)*, March 2019, pp. 127–130.

[12] M. Danneberg, R. Bomfin, S. Ehsanfar, A. Nimr, Z. Lin, M. Chafii, and G. Fettweis, "Online wireless lab testbed," in *2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*, April 2019, pp. 1–5.