OpenAirLink: Reproducible Wireless Channel Emulation using Software Defined Radios

Yash Deshpande, Xianglong Wang, Wolfgang Kellerer
Chair of Communication Networks, Technical University of Munich, Germany
Email: {yash.deshpande, xianglong.wang@tum.de, wolfgang.kellerer}@tum.de

***Abstract*—This paper presents OpenAirLink(OAL), an open-source channel emulator for reproducible testing of wireless scenarios. OAL is implemented on off-the-shelf software-defined radios (SDR) and presents a smaller-scale alternative to expensive commercially available channel emulators. Path loss and propagation delay are the fundamental aspects of emulating a wireless channel. OAL provides a simple method to change these aspects in real-time. The emulator is implemented using a finite impulse response (FIR) filter. The FIR filter is written in Verilog and flashed on the SDRs Field Programmable Gate Array (FPGA). Most processing transpires on the FPGA, so OAL does not require high-performance computing hardware and SDRs. We validate the performance of OAL and demonstrate the utility of such a channel emulation tool using two examples. We believe that open-source channel emulators such as OAL can make reproducible wireless experiments accessible to many researchers in the scientific community.**

***Index Terms*—wireless communication, testing, reproducibility, signal processing**

I. Introduction

Experimental validation and evaluation of wireless technologies on hardware platforms are essential for their acceptance in the industry. While simulation and theoretical modeling can provide insights and predictions, they often make abstractions or assumptions that may not reflect the complexities of actual hardware. Thus, conducting experiments on hardware helps researchers to validate their assumptions, assess the accuracy of their predictions, or uncover discrepancies in their models. Indeed, many wireless testbeds have been instrumental in advancing knowledge in the wireless domain [1]–[7].

The reproducibility of scientific experiments serves as a cornerstone for establishing the credibility and reliability of research findings. Experiments must be conducted in a controlled environment to be reproducible and for users to quickly identify the causation between a parameter and the output [8]. This reproducibility is challenging in wireless systems. Many factors outside the experimenter's control may affect the quality of a wireless channel. Multipath reflections or interference from unwanted sources affect the wireless signal and could degrade the reliability of the results [2], [5]. Moreover, many lab-based setups can't conduct wireless mobility experiments spanning longer distances. Thus, the need for a channel emulator arises. These physical devices can be configured to distort, delay, and attenuate a wireless signal per the user's need. Instead of antennas, the user places the channel emulator between the transmitter and receiver and connects them with conductive cables.

However, the cost and complexity of commercially available channel emulators make them inaccessible to institutions with modest means. They take time to set up and often require wireless networking expertise to operate. Researchers working at an application level such as robotics and control often lack this wireless networking expertise. OAL aims to bridge this gap. It aims to provide a fast-installation channel emulator where the hardware can later be repurposed once the experiment is done. On the other hand, OAL is less scalable and is meant to be used for smaller-scale scenarios. The industrial channel emulators are also "closed," meaning modifications require vendors and experts' support. Incorporating an open-source approach to the wireless channel emulator not only promotes transparency and accessibility but encourages widespread adoption and community-driven innovation [9], [10]. Moreover, it helps improve reproducibility as the same emulator can be utilized when replicating experiments [11].

This paper presents OAL, an open-source wireless channel emulator, and its design and implementation. OAL is publicly available under the GNU License[1]. The paper also presents validation and testing of the performance of OAL and outlines the methodology to validate any such channel emulator. The performance of OAL is compared with the Spirent Vertex, a state-of-the-art (SOTA) industrial channel emulator. Finally, examples of where OAL could be helpful, such as low-powered wireless networks based on IEEE 802.15.4 [12] and in an Openairinterface [13] 5G testbed, are demonstrated.

II. Related Work

The most common approach for characterizing a wireless channel's path loss, delay, and delay spread is employing a tapped delay line (TDL) model. The use of finite impulse response (FIR) filters on Digital Signal Processors (DSP) and Field Programmable Gate Array (FPGA) to emulate the TDL has been explored in many works [14]–[17]. Practical

The authors acknowledge the financial support by the Bavarian State Ministry for Economic Affairs, Regional Development and Energy (StMWi) for the Lighthouse Initiative KI.FABRIK (Phase 1: Infrastructure and the research and development program under grant no. DIK0249). The authors also acknowledge the financial support by the Federal Ministry of Education and Research of Germany (BMBF) in the program of "Souverän. Digital. Vernetzt." joint project 6G-life, project identification number 16KISK002.

ISBN 978-3-903176-63-8 © 2024 IFIP

[1]https://github.com/n3martix/OpenAirLink

channel models at the system level simulate or emulate, at the most, three paths [18], [19]. Hence, clustering mechanisms approximate the Power-delay profile (PDP) of a large number of multipath coefficients to the few paths used for simulation/emulation [20], [21]. Thus, a sparse FIR filter is more efficient in emulating a channel with a certain path delay and at most three multipath components [17], [22], [23].

Channel emulation on FPGA and using FIR filters has been proposed by [24], [25]. However, these works are before the advent of software-defined radio (SDR)s. Hence, these works had to develop their own RF frontend and connect it to an FPGA board. The SDR hardware as shown in Fig. 1 integrates the RF frontend with digital signal processing and can retransmit the converted signal back through an output port [26]. Thus, the SDR helps build one channel emulator in a box.

A large-scale channel emulator where SDR is used as the RF frontend is proposed in [17]. However, this requires a complex FPGA array in between the SDR cards to deal with scale. This significantly pushes the cost of implementation up for such a system. Indeed, Colosseum [6] implements such a system at great cost and allows wireless researchers worldwide to connect to it. However, the idea behind OAL is for researchers to build and test smaller-scale wireless systems in their labs before testing them at scale in systems such as the Colosseum. An emulator purely based on SDRs to emulate a vehicle-to-anything (V2X) channel is presented in [23]. They also implement doppler effects and small-scale stochastic fading. However, the work evaluates only the packet error ratio (PER) of an emulated system when using the IEEE 802.11p protocol. In this paper, we also discuss some fundamental specifications in such FPGA based channel emulators, such as the dynamic range, resolution, and emulation precision, which are missing from the works mentioned above.

Finally, for completeness, we mention other ways to deal with the repeatability of wireless channels. Random unwanted effects observed on the channel can be eliminated by detecting outliers after the experiment, as shown in [2]. The experiments can also be conducted in an anechoic chamber such as [5]. One could skip the RF-level channel emulation and abstract the effects to a higher level of emulation, such as IQ-level or packet-level emulation [7]. However, this restricts the type of experiments that can be conducted and the hardware that can be used in them [27]. Finally, the abstraction from RF to IQ or packet-level emulation also misses certain scenarios in the abstraction which could be useful to capture the full effect of the test.

III. Implementation

A. Emulator Architecture

Fig. 1 shows the overall architecture of the channel emulator. The RF frontend receives and downconverts the analog signal, which is then passed to the SDRs analog to digital converter (ADC). The ADC in the SDR performs IQ sampling and quantization such that a continuous signal $X(t)$ is converted to a complex sample $\hat{X}[n] = \Delta \cdot \text{round}(\frac{X(nT_s)}{\Delta})$, where $n \in \mathbb{N}$,

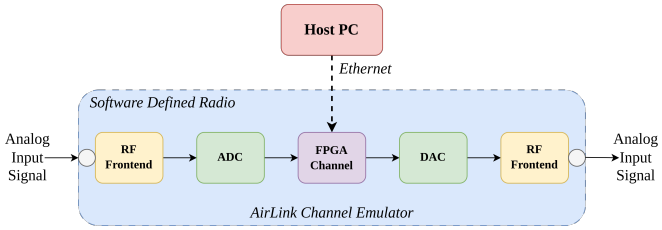

Fig. 1: Emulator Architecture: The software-defined radio (SDR) hardware is used to build the entire channel emulator, integrating the signal conversion hardware with an FPGA-based DSP. The host PC with an ethernet connection is used to control the delay and attenuation of the channel.

$\hat{X}[n] \in \mathbb{C}$, T_s is the sampling period and Δ is the quantization level. We get Δ by dividing the input signals range with the resolution of the ADC 2^R. The ADCs sampling period T_s and resolution 2^R provide a fundamental limit on the bandwidth and precision of the channel emulator. The ADC then feeds its digital output to the emulated channel on the FPGA. The channel is applied to the signal, and the digital samples output $\hat{Y}[n] \in \mathbb{C}$ are then re-converted to an analog signal $Y(t)$ by the digital to analog converter (DAC) and transmitted by the RF front end.

The SDR connects to a host PC via Ethernet. The PC issues control commands and updates the emulated channel at run time. This architecture offers stable and low-latency pipeline processing by utilizing the SDR's FPGA and performing all signal processing onboard.

B. FPGA Channel

The FPGA channel applies the desired channel fading, including delay and power loss for signal path and multi-path effect from reflection or diffraction to the RF signal. This requires an efficient model that works in FPGA to simulate effects. The theoretical model used for the emulator is a TDL model. A TDL model consists of delay-line and tap-output components. It simulates propagation delay and scales the signal correspondingly. Mixing single paths with different path delays can produce the multipath effect. It can precisely simulate various RF fading channels by designing delay and tap scale coefficients. The TDL model is of order N and produces the output

$$\hat{Y}[n] = \sum_{i=0}^{N} b_i \cdot \hat{X}[n-i], \tag{1}$$

where $x[n-i]$ is the i-th *tap* with a coefficient b_i and $0 \leq i \leq N$.

A TDL channel can be considered a general causal FIR filter. The implementation of the emulator is powered by RF Network on Chip (RFNoC) [28], a framework used for implementing DSP in Universal Software Radio Peripheral (USRP)'s FPGA.

1) Path Delay: Block shift registers advance the digital samples through the taps at each clock cycle. Therefore, the resolution of the path delay is defined by the FPGA clock cycle frequency, $f \cdot$ Hz, giving a delay step size of $\frac{1}{f}$s. With

N number of taps, such a channel emulator can support $\frac{N-1}{f}$s of maximum delay. Considering that the RF signal propagates at the speed of light c in free space, the step size of the path distance is $\frac{c}{f} \cdot \mathrm{m}$, with a total propagation distance of $\frac{c \cdot (N-1)}{f} \cdot \mathrm{m}$. The clock cycle also quantizes the distance between multipath components in the delay spread. The minimum spread between any two components of the signal is limited to the delay step size of $\frac{1}{f}$s.

2) Attenuation: The IQ samples are scaled by the multiplier $b_i \in [-2^r, 2^r] \cap \mathbb{Z}$ inside the FIR filter, where $r+1$ is the number of bits allocated for signed integers in the FPGA. We are only interested in attenuating the signal. Therefore, the path gain for the i-th tap is,

$$G = 20log_{10}(\frac{b_i}{2^r - 1}) \tag{2}$$

where $b_i \in [1, 2^r - 1] \cap \mathbb{N}$. From Eq.2, the resolution of attenuation is given by,

$$\Delta G = 20log_{10}(\frac{b_i + 1}{b_i}). \tag{3}$$

Hence, attenuation's resolution degrades with the emulation distance between the transmitter and receiver. We have approximately a dynamic range of attenuation from the two extreme values that b_i can take, $D \approx r \cdot 20log_{10}(2)$. This dynamic range of a few hundred meters (in free space) restricts the emulation capacity. Secondly, the resolution at the tail end of this dynamic range would be 6dB, meaning a distance resolution of a hundred meters in the case of the free space path loss (FSPL) model. Hence, using a method similar to [15], we define the dynamic range given a maximum attenuation resolution ΔGm. First, we put ΔGm in Eq.3 to find the minimum value of b_i that gives us the desired resolution. Then, we can find our reduced dynamic range for the given allowed resolution by subtracting the two extreme values that b_i can take,

$$D_{\Delta Gm} = r \cdot 20log_{10}(2) + 20log_{10}(10^{\frac{Gm}{20}} - 1) \tag{4}$$

OAL employs a bit shift operation before the FIR filter that coarsely attenuates the signal for all the path components. A right shift by j bits is equivalent to dividing the sampled values by 2^j or attenuating the signal by approximately $6j$dB. If the maximum number of bit shifts is s, we get a higher dynamic range,

$$D_{\Delta Gm} = (r+s) \cdot 20log_{10}(2) + 20log_{10}(10^{\frac{Gm}{20}} - 1). \tag{5}$$

Thus, we get the worst-case resolution for our system by plugging $D_{\Delta Gm} = 6$dB in Eq. 4 and solving for Gm. Putting it all together, we get attenuation resolution for when bit-shift is employed,

$$Gbs \approx 20log_{10}(\frac{10^{\frac{3}{10}}}{2^r} + 1). \tag{6}$$

The effect of the bit-shift operation is shown in Fig. 2 for $r = 15$. The resolution from Eq 3 is plotted against the desired attenuation G. The dynamic range without bit shifting is restricted to 80 dB, and the resolution worsens as we reach this value. The bit-shifting value increases the dynamic range well beyond 80 dB and the worst-case resolution Gbs to 0.00058 dB. The zoomed portion of the plot shows how the resolution is bounded at Gbs by the blue dashed line. Finally, we mention that the IQ output values in the FPGA FIR filters $\hat{Y}[n]$ can only be integers. Due to truncation, a small phase distortion of maximum value 0.5*LSB is introduced by the FPGA channel for both the I and Q parts of the sample.

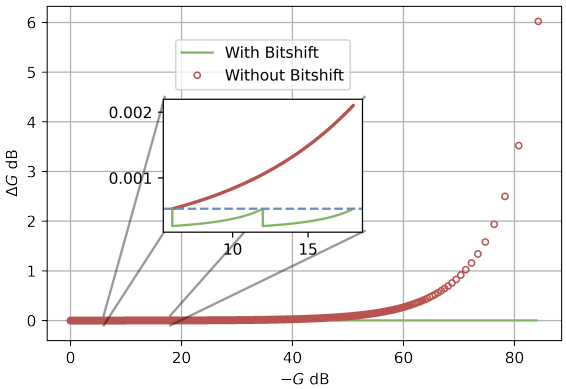

Fig. 2: Attenuation Resolution: As the attention increases, the resolution of the emulator degrades. For a $r = 15$ bit division of the digital sample, a maximum attenuation of roughly 82 dB is possible. However, the resolution at this attenuation level is almost 6 dB. The bit-shifting operation can greatly improve this resolution. The worst-case resolution when bit shift is employed for the same value of r is 0.000528 dB shown in the dashed blue line.

C. Channel Update

Section III-A mentions that the emulator connects to the host PC through Ethernet. A program in the host PC controls the emulator and updates the b_i, where $i \in \{1, 2, ..., N\}$ coefficients on the FPGA in runtime. Similar to Coloseum [6], the program that updates these coefficients reads a pre-prepared CSV file. Suppose the channel can be updated u times per second, then the maximum velocity of the channel emulator at resolution $\frac{c}{f} \cdot \mathrm{m}$ is $\frac{c \cdot u}{f} \cdot \mathrm{m/s}$. Thus, we can increase the maximum velocity of the nodes if we decrease the resolution.

D. Calibration

The last step of implementation involves hardware calibration. For USRPs, the RF frontend TX and RX gains can be adjusted via the UHD library tools. The emulator is connected and set in a pass-through mode, i.e., no attenuation or delay is applied. Then, a specific transmit power is set on the transmitter, and the received power must equal the sum of transmit power, the transmitter's TX gain, and the receiver's RX gain. If that's not the case, the RX gain of the RF frontend of OAL should be adjusted accordingly. Due to different values of the peak-to-average power ratio (PAPR) and inherent losses in wireless hardware, this calibration must be done before every test setup. The USRP also produces a tone with the

Parameter	Formula	Value
Maximum Bandwidth	-	120 MHz
Frequency Range	-	400MHz to 4.4GHz
Delay Resolution	$1/fs$	5 ns
Maximum Delay	$(N-1)/fs$	205 ns
Attenuation Resolution	Eq.6	0.000528 dB
Maximum Attenuation	Eq.5	≈144 dB
Maximum Velocity	$(c \cdot u)/f$	1500 m/s

TABLE I: OAL Specifications: The achievable bandwidth, delay, attenuation resolution, and velocity.

oscillator frequency, which is caused by the DC offset due to the local oscillator power leakage when up/down-converting RF signals. A self-calibration program by the UHD library can minimize but not eliminate it.

IV. Specifications

The SDR chosen for OAL is the USRP X310. This device is widely used and supported by many software frameworks, such as GNU Radio and MATLAB. The USRP is equipped with a Kintex 7 FPGA that runs the emulated channel. A signed integer on the Kintex 7 is 16 bits, giving the value of r to be 15. The ADC has a maximum sample rate $\frac{1}{T_s}$ of 200 MSample/s and a resolution R of 14-bit/sample ADC, while the DAC resolution is 16-bit/sample. The SBX-120 daughterboard is the RF front-end, with 120MHz analog bandwidth and independent TX and RX modules, simultaneously emulating uplink and downlink channels by a single USRP. A 2x1 multiplexer/demultiplexer can be realized on the same hardware instead of simultaneous uplink and downlink emulation. Providing this option to OAL users is a part of our future work. The daughterboard of the selected USRP supports a frequency range of 400MHz to 4.4GHz.

The specifications of the OAL are shown in TableI. Theoretically, the bandwidth of the channel emulator depends on the Nyquist criterion based on the sampling rate of the ADC. This value for the USRP X310 is 200MHz since the sampling rate is 200 MSamples/s and the sampling method is IQ. However, OAL's bandwidth is limited by the bandwidth of the RF frontend SBX-120 daughterboard, which is 120 MHz. The daughterboard also supports a frequency range of 400MHz - 4.4GHz. This value is large enough to cover most commercially known wireless systems, except for wideband systems.

The FPGA clock cycle frequency f is 200 MHz. This gives us a delay resolution of 5 ns. In free space, that corresponds to a distance resolution of 1.5 m. The number of taps in the FIR filter N is 42, and hence we get a maximum delay of 205 ns or a maximum distance of emulation between transmitter and receiver in free space of 60 m. Further extending the number of taps N and determining the limits of the filter size is a part of our future work. The attenuation resolution of 0.000528 dB and dynamic range of 144 dB are obtained from $r = 15$ and $s = 8$. These limits are placed due to the size of integers on the Kintex 7 and the truncation caused by integer values on the FPGA. Finally, we tested on a 4-core Intel i7 PC with 10G ethernet card, the effect of the channel update rate. At an update rate u of 1000 updates per second, the maximum CPU utilization was just 54 percent. Thus we can control multiple OAL SDRs from a single host PC. At the best distance resolution and update rate, the maximum velocity that AirLnk can support is 1500 m/s. This speed is already much more than what is needed for most mobility applications. It must be noted that OAL does not emulate doppler effects, which would play an essential role in higher velocity emulation. This is also a part of our future work.

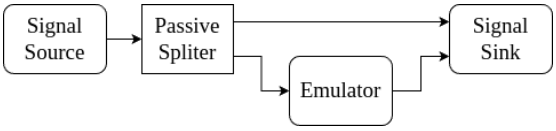

Fig. 3: Verification Setup: A signal is split into two copies using a passive RF splitter. One copy goes directly to the signal sink and the other to the sink via the channel emulator. The comparison between the two copies allows us to test the latency of the OAL and verify the delay operation.

V. Verification and Performance Tests

This section aims to verify OAL's operation, test its performance, and demonstrate that it can reliably reproduce a wireless channel. The tests in this section utilize other USRPs and inexpensive RF modules and can be reproduced for testing the performance of any such wireless channel emulator.

The test setup is organized as shown in Fig. 3. A test signal generated by the signal source is split into two identical copies using a passive RF splitter. One copy is sent to the emulator for processing, while the other is used as a reference. Both copies are recorded and compared at the signal sink. The signal source and sink are the same SDRs, namely the USRP X310. All three SDRs clocks are disciplined via an external PPS and 10MHz source using the NI Octoclock time distribution device to mitigate the effects of clock drifts in the latency measurements.

A. Processing Latency and Delay

The latency of the emulator refers to the time required for the RF signal to pass through the USRP and apply channel effects. The additional time OAL adds is measured by analyzing the difference between the test signal's time arrival from the two paths shown in Fig. 3. The delay is derived from the negative of the lag at the point where the normalized cross-correlation between the two signals has the highest absolute value. To determine each OAL component's latency, we first measured the processing latency through pass-through, where the received signal is immediately retransmitted without applying the FPGA channel. We then enabled each component of our FPGA channel and measured the increased latency to demonstrate the extra time introduced by the channel components. Around 52 percent of the latency is introduced by the RF frontend and ADC/DAC components, which have the tasks of signal trans-receive, up/down-conversion, and sampling. The FPGA channel is implemented using RFNoC [28], which constrains its processing latency performance due to the Module-NoCCore-Module structure

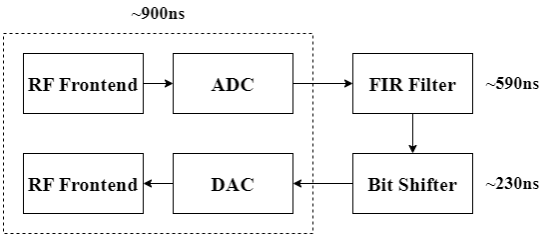

Fig. 4: OAL Processing Latency: The processing latency of OAL. Each component's variance in processing latency was maximum $\pm$1ns. This certainty in the latency is helpful for the reproducibility of the wireless channel.

with FIFO buffers in between to ensure pipeline processing and prevent sample loss due to buffer overflows.

Fig. 4 displays the measured processing latency, with a total time of approximately 1.72us. The measured latency demonstrated high stability with the maximum variance at each component bounded by $\pm$1ns. We performed the same overall processing latency test for the Spirent Vertex channel emulator. The results showed similar stability, but the magnitude of the processing latency was $\approx$ 3.19us. The datasheet of another SOTA industrial channel emulator, namely the Keysight F8800A Propsim F64, states the value of processing latency to be 2.6us. Hence, OAL stands out from other channel emulators regarding latency, although it lacks the cross-channel mixing usually available in industrial ones. Adding a Doppler block for IQ multiplication will increase the processing latency by at least 200 ns.

Path delay refers to the emulator's capability to delay a signal to emulate wireless propagation delay. We set the FIR filter coefficients to emulate a given propagation delay and measure it using the same method used to measure the processing latency. As mentioned in Section IV, the resolution for path delay for OAL is 5ns. We tested the path delay performance of OAL across the entire dynamic range of the emulator. Once again, the results demonstrated the same stability as in the case of processing latency and perfect accuracy.

B. Attenuation

The emulator's other core operation is to scale the RF signal with the desired attenuation. We verify whether the received power at the signal sink matches the expected values after attenuation. At the same time, we conduct a conformance test to check if the signal is valid at reception after the emulation operation. Hence, instead of measuring and comparing the received signal power, we use IEEE 802.15.4 hardware transceivers as signal source and sink. The attenuation test is conducted by Zolertia Re-Mote (Mote) [29] and Contiki OS [30]. The Mote carries a cc2538 system-on-chip microcontroller and runs an IEEE 802.15.4 radio with a build-in input received signal-strength indication (RSSI) measurements. From the data sheet [31], the actual input power is given by $P = \text{RSSI} - \text{offset}$, where offset is the front-end gain set during system design and varies for microcontrollers, the value for the cc2538 is 73dB.

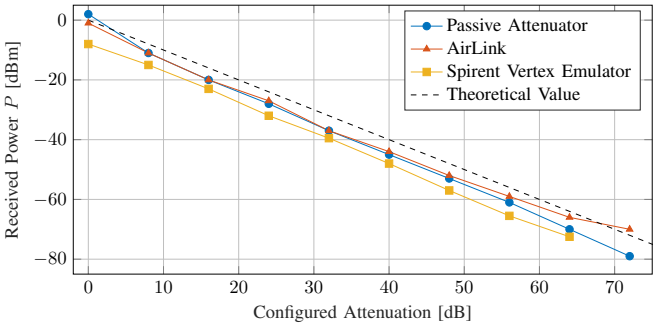

Fig. 5: Attenuation Operation: The attenuation of OAL is compared with a passive variable attenuator and the Spirent Vertex. All the measured values are averages over 1000 measurements. The received power values are lower than the theoretical value due to unaccounted losses in the conducive cables.

Fig. 5 compares OAL with a passive attenuator and the Spirent Vertex. The figure plots the average received input power at the Mote from the reported RSSI over 1000 measurements at each configured attenuation level. The maximum variance for all three attenuation methods was $\pm$3dB.

All measured values of P are lower than the expected theoretical value due to unaccounted losses in the conductor cables and contact. The root mean square (RMS) between the measured value and the expected theoretical value for OAL was 10.44dBm, which is lower than the 15.17dBm of the passive attenuation and 24.29dBm for the Spirent Vertex.

C. Conformance Test with 5G NR

The attenuation verification conducted in Section V-B also demonstrates that OAL can be used for IEEE 802.15.4 signals. OAL's design objective is to accommodate a spectrum of RF signals and protocols. To demonstrate the versatility of OAL, we measured the uplink Block error rate (BLER) in an OpenAirInterface [13] testbed. A USRP B210 is employed as the gNB, while a USRP B210 mini is used at the user equipment (UE). In 5G, the devices adapt their Modulation Coding Scheme (MCS) depending on the channel quality, which leads to the BLER fluctuating due to this adaptation. Hence, we fix the MCS to 6 for this test and apply different attenuation values to the uplink signal. OpenAirInterface gNB reports the BLER every second, and we measure 200 such measurement reports for one test. Fig. 6 shows an increase in the BLER concurrent with increments in path loss. These results substantiate OAL's adeptness at simulating 5G NR signal propagation, thus endorsing its utility in a broad spectrum of applications.

VI. Conclusion

This paper presents OAL an open-source channel emulator realized with SDRs. Controlling the wireless channel quality in various experiments is crucial for the reproducibility of scientific work. Thus, wireless channel emulators are used to have a controlled RF environment and to emulate larger distances in laboratory rooms. OAL aims to provide a small-scale channel emulator to emulate point-to-point wireless links. The paper presents the FIR implementation of OAL on the SDRs

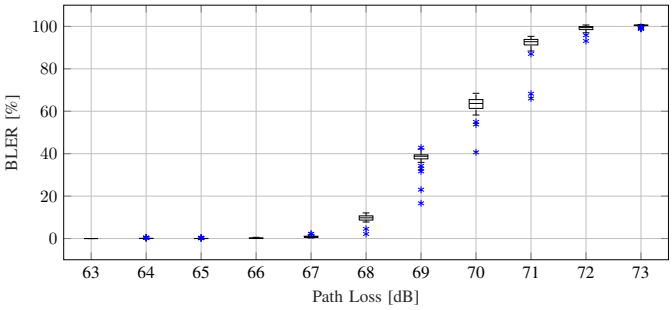

Fig. 6: 5G NR Signal Conformance: The Block Error Rate increases with higher attenuation, culminating in a link failure. The MCS was kept constant throughout the measurements to 6. The NR gNB reports the BLER every second, and 200 such measurements were conducted for each path loss setting.

FPGA and OAL's specifications. The test and validation results show that OALs performance is comparable to or better than SOTA industrial wireless channel emulators. The variance in emulating path delay and attenuation with OAL is minimal, demonstrating its utility in reliably reproducing a wireless channel. The paper demonstrates that OAL can preserve the symbol quality from various wireless protocols, proving its versatility. To the best of our knowledge, we find that OAL offers at least a 10x cost reduction while providing the same quality and reproducibility of wireless channels as compared to SOTA industrial channel emulators.

REFERENCES

[1] I. Broustis, J. Eriksson, S. V. Krishnamurthy, and M. Faloutsos, "A blueprint for a manageable and affordable wireless testbed: Design, pitfalls and lessons learned," in *2007 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities*, 2007.

[2] E. Nordstrom, P. Gunningberg, and H. Lundgren, "A testbed and methodology for experimental evaluation of wireless mobile ad hoc networks," in *First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, 2005.

[3] J. Breen, A. Buffmire, J. Duerig, K. Dutt, E. Eide, M. Hibler, D. Johnson, S. K. Kasera, E. Lewis, D. Maas, A. Orange, N. Patwari, D. Reading, R. Ricci, D. Schurig, L. B. Stoller, J. Van der Merwe, K. Webb, and G. Wong, "Powder: Platform for open wireless data-driven experimental research," in *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*. New York, NY, USA: Association for Computing Machinery, 2020.

[4] G. Georgis, M. Filo, A. Thanos, C. Husmann, J. C. De Luna Ducoing, R. Tafazolli, and K. Nikitopoulos, "Sword: Towards a soft and open radio design for rapid development, profiling, validation and testing," *IEEE Access*, vol. 7, 2019.

[5] N. H. Vaidya, J. Bernhard, V. V. Veeravalli, P. R. Kumar, and R. K. Iyer, "Illinois wireless wind tunnel: a testbed for experimental evaluation of wireless networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Experimental Approaches to Wireless Network Design and Analysis*. New York, NY, USA: Association for Computing Machinery, 2005.

[6] T. Melodia, S. Basagni, K. R. Chowdhury, A. Gosain, M. Polese, P. Johari, and L. Bonati, "Colosseum, the world's largest wireless network emulator," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 2021.

[7] M. L. Sichitiu, I. Guvenc, R. Dutta, V. Marojevic, and B. Floyd, "Aerpaw emulation overview," in *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*. New York, NY, USA: Association for Computing Machinery, 2020.

[8] V. Bajpai, A. Brunstrom, A. Feldmann, W. Kellerer, A. Pras, H. Schulzrinne, G. Smaragdakis, M. Wählisch, and K. Wehrle, "The dagstuhl beginners guide to reproducibility for experimental networking research," *SIGCOMM Comput. Commun. Rev.*, vol. 49, no. 1, feb 2019.

[9] M.-W. Wu and Y.-D. Lin, "Open source software development: an overview," *Computer*, vol. 34, 2001.

[10] L. Morgan, J. Feller, and P. Finnegan, "Exploring value networks: theorising the creation and capture of value with open source software," *European journal of information systems*, vol. 22, 2013.

[11] R. W. Bowman, "Improving instrument reproducibility with open source hardware," *Nature Reviews Methods Primers*, vol. 3, no. 1, 2023.

[12] "Ieee standard for low-rate wireless networks," *IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015)*, 2020.

[13] F. Kaltenberger, G. d. Souza, R. Knopp, and H. Wang, "The openairinterface 5g new radio implementation: Current status and roadmap," in *WSA 2019; 23rd International ITG Workshop on Smart Antennas*, 2019.

[14] T. Laakso, V. Valimaki, M. Karjalainen, and U. Laine, "Splitting the unit delay [fir/all pass filters design]," *IEEE Signal Processing Magazine*, vol. 13, no. 1, 1996.

[15] K. C. Borries, G. Judd, D. D. Stancil, and P. Steenkiste, "Fpga-based channel simulator for a wireless network emulator," in *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, 2009.

[16] H. Eslami, S. V. Tran, and A. M. Eltawil, "Design and implementation of a scalable channel emulator for wideband mimo systems," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, 2009.

[17] A. Chaudhari and M. Braun, "A scalable fpga architecture for flexible, large-scale, real-time rf channel emulation," in *2018 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, 2018.

[18] M. Narandzic, C. Schneider, R. Thoma, T. Jamsa, P. Kyosti, and X. Zhao, "Comparison of scm, scme, and winner channel models," in *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, 2007.

[19] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials," *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 6, 2014.

[20] R. He, B. Ai, A. F. Molisch, G. L. Stuber, Q. Li, Z. Zhong, and J. Yu, "Clustering enabled wireless channel modeling using big data algorithms," *IEEE Communications Magazine*, vol. 56, no. 5, 2018.

[21] M. Tehrani-Moayyed, L. Bonati, P. Johari, T. Melodia, and S. Basagni, "Creating rf scenarios for large-scale, real-time wireless channel emulators," in *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2021.

[22] D. Mattera, F. Palmierl, and S. Haykin, "Efficient sparse fir filter design," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 2002.

[23] G. Ghiaasi, T. Blazek, M. Ashury, R. R. Santos, C. Mecklenbräuker *et al.*, "Real-time emulation of nonstationary channels in safety-relevant vehicular scenarios," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[24] G. Judd and P. Steenkiste, "Repeatable and realistic wireless experimentation through physical emulation," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, jan 2004.

[25] ——, "Using emulation to understand and improve wireless networks and applications," in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2*. USA: USENIX Association, 2005.

[26] M. Dillinger, K. Madani, and N. Alonistioti, *Software defined radio: Architectures, systems and functions*. John Wiley & Sons, 2005.

[27] A. Panicker, O. Ozdemir, M. L. Sichitiu, I. Guvenc, R. Dutta, V. Marojevic, and B. Floyd, "Aerpaw emulation overview and preliminary performance evaluation," *Computer Networks*, vol. 194, 2021.

[28] M. Braun, J. Pendlum, and M. Ettus, "Rfnoc: Rf network-on-chip," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016.

[29] "Zolertia Remote: Lightweight Internet of Things hardware development platform," https://zolertia.io/product/re-mote/.

[30] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *IEEE International Conference on Local Computer Networks*, 2004.

[31] *CC2538 Powerful Wireless Microcontroller System-on-Chip for 2.4-GHz IEEE 802.15.4-2006 and ZigBee Applications*, Texas Instruments, 2023, accessed on: April 2, 2024. [Online]. Available: https://www.ti.com/lit/gpn/cc2538