Experimental Validation of a Bluetooth-Based Speech Audio Broadcasting Model

Mathias Baert*, Jowan Pittevils†, Bart Moons* and Jeroen Hoebeke*
*imec-IDLab-UGent, Ghent, Belgium
†Cochlear Technology Centre, Mechelen, Belgium
Email: mathias.baert@ugent.be

***Abstract*—The concept of broadcasting audio using Bluetooth has been standardized as Auracast. It enables a variety of use cases, both for personal use as well as in public venues. Due to its novelty, available research work about the technology and its performance, is limited. This paper provides a performance evaluation of the listening experience on Bluetooth-based broadcast audio streams, by performing measurements on a testbed that can set up an audio link between a broadcaster and receiver on Bluetooth-capable hardware. The testbed can introduce variable attenuation on the audio link and inject nearby interference from a Bluetooth Low Energy (BLE) connection. The measurements are compared against simulation results from a model that was designed and implemented as part of previous work. A comparable performance between the testbed and model is shown to be achieved for different attenuation settings. Next to this, the work in this paper presents an extension to the model that enables it to provide accurate simulation results for a broadcast audio stream operating in proximity to multiple BLE connections.**

***Index Terms*—Bluetooth, LE Audio, Auracast, LC3, simulation, testbed**

I. Introduction

One of the interesting exhibitions at CES 2024 was the Auracast Experience from Bluetooth SIG, which allowed users to experience the added benefit of being able to synchronize to broadcast audio streams present in a public venue or provided by a private device [1]. It enables a variety of use cases: flight-specific broadcast information in an airport, assistive listening systems in an auditorium or theatre, etc. Auracast is part of the recent extension to the Bluetooth ecosystem that allows wireless audio streaming on top of Bluetooth Low Energy (BLE). This paradigm, denoted as LE audio, provides standardization for synchronized rendering of multiple audio streams at different receivers (e.g. left and right earbud) and extends the available use cases by providing support for both connection-based and broadcast-based (Auracast) audio streaming. Transmitting on top of BLE is made possible via the development of the new Low Complexity Communication Codec (LC3) that can provide sufficient audio quality at much lower bit rates compared to the legacy Low Complexity Subband Codec (SBC) that is used in Classic Bluetooth audio [2]. The encoding procedure of LC3 is built upon the Modified Discrete Cosine Transform (MDCT) mechanism, that converts N audio input time domain samples, from a 7.5 or 10 ms frame, into N spectral coefficients. Subsequently, the MDCT data is transmitted over the air and the decoding procedure transforms the spectral coefficients back to a time domain signal [3]. The codec is optimized for 10 ms encoded frames and only provides support for 7.5 ms to cope with legacy intervals from Bluetooth Classic [4].

Auracast transmitters can broadcast one or more audio stream(s) towards an unlimited number of Auracast receivers. The underlying mechanism that enables an Auracast transmitter to broadcast audio, is a Broadcast Isochronous Group (BIG). It can consist of one or more Broadcast Isochronous Streams (BISes), to group related audio streams together. Each BIS represents a separate audio stream. Grouping multiple streams within a single BIG enables synchronized playback of audio broadcasted on all the streams in the group (e.g. two streams for a left and right earbud). Our previous work in [5] presents an extensive study of the audio quality and intelligibility that can be expected from received encoded audio that is broadcasted on top of a BIG. To achieve this, a model was implemented to simulate how an on-air broadcast stream could be impacted by network conditions and interference. The model is able to simulate the packet schedule associated with encoded audio frames being transmitted by a BIG broadcaster. It applies packet loss scenarios on the on-air packet stream based on a pre-defined Packet Error Rate for each BLE frequency channel. Afterwards, it can mimic a potential BIG receiver attempting to reconstruct the encoded audio frame stream (with one or more potential gaps due to frame losses) by performing decoding and, where needed, Packet Loss Concealment (PLC). The model was employed to test out different sets of LC3 parameters and BIG configurations on 20 distinct audio files. The work also provides an executable of the implementation of the model. Researchers can employ it to, for instance, evaluate the potential improvement on the listening experience from novel PLC techniques without requiring real BLE-capable hardware. Apart from [5], the availability of previous research work on LE audio and Auracast is limited. The work in [6] presents an alternative retransmission mechanism for a BIG. Next to this, a technical overview of LE audio and its associated use cases and challenges is provided in [7]. Hence, it is clear that usage of this technology will keep increasing in the years to come

The paper results from the imec.icon project BLE2AV, financed by imec and Flanders Innovation & Entrepreneurship. The authors would like to thank Stijn Kindt for his assistance during the evaluation and writing process and the anonymous reviewers for their insightful comments and suggestions.

979-8-3503-6224-4/24/$31.00 ©2024 IEEE

thus it is relevant to assess its capability of being able to cope with various realistic attenuation and interference scenarios.

One of the main contributions of the presented work in this paper is to provide a validation of results provided by the BIG model against measurements performed on real BLE-capable hardware, in order to further demonstrate the reliability of insights from the model. The measurements are performed on a controllable testbed, connecting BIG broadcaster and BIG receiver hardware to each other via several shielded SMA components. The testbed can introduce variable attenuation on the link between broadcaster and receiver, and has the capability to inject BLE interference from a single BLE connection in the testbed. The measurements performed while BLE interference is present, expose that the model is unable to fully mimic the behavior of a BIG under those network conditions and thus a second contribution of this paper is to provide an extension to the BIG model that is able to cope with specific behavior associated with interference from one or more BLE connection(s). The evaluation results and updated executable of the model are provided in [8]. The remainder of the paper is organized as follows. Section II describes the BIG model and the internal scheduling of BLE connections, limited to a description of the functionality that is relevant to the evaluation performed in this paper. The controllable testbed and the LE audio software used in this paper, are described in Section III. Section IV describes the results for the variable attenuation and BLE interference scenarios. The extensions made to the BIG model are illustrated in Section V and Section VI concludes the paper and provides a brief outlook on future work.

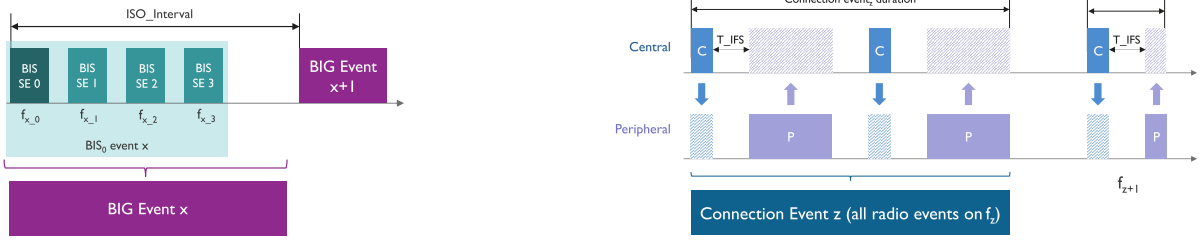
II. Primer on BLE Connections and the BIG Model

The BIG model can mimic a BIG broadcaster packetizing LC3 encoded audio frames into BIS Protocol Data Units (PDUs) to be sent on air as part of a BIG. These PDUs can be transmitted in periodically occurring BIG events which are always a pre-configured ISO_Interval apart [9]. The evaluated BIGs in this paper are limited to a single BIS per BIG. The number of subevents (NSE) parameter associated with a BIG, indicates the number of transmission opportunities (i.e. subevents) that are present in each BIG event (per BIS). The Burst Number (BN) indicates how many of those subevents are original transmissions. The evaluated BIGs in this paper use a BN of 1 so each audio frame can always fit in a single PDU (no fragmentation) and only one audio frame is transmitted per BIG event. The subevents can be divided into $\frac{\text{NSE}}{\text{BN}}$ groups or thus, in case of BN equal to 1, NSE groups. The remaining $\frac{\text{NSE}}{\text{BN}}$ - 1 groups can contain either retransmissions of the preceding BN original transmission(s) or pre-transmissions of PDUs that will be sent as original transmissions in a future BIG event. The Immediate Repetition Count (IRC) parameter indicates how many groups contain retransmissions of the BN original transmission(s). The pre-transmission functionality is out of scope so all evaluated BIGs in this paper use an IRC parameter equal to $\frac{\text{NSE}}{\text{BN}}$. Figure 1a depicts the contents of a single BIG event. The associated BIG has a BN of 1, an NSE parameter of 4 and an IRC of 4. Each subevent has an associated frequency channel provided by the Channel Selection Algorithm (CSA) #2 [9]. The channel selection set can be limited by blacklisting one or more channels via a 37-bit channel map, that contains at least two non-blacklisted channels. Per BIS event in a BIG event, the frequency channels of two subsequent subevents will always be different. The BIG model includes the functionality of the CSA #2 in order to provide a frequency channel to each subevent. The model is pre-configured with a Packet Error Rate (PER) for each frequency channel. For each subevent, the model takes a sample from a uniform distribution between 0 and 1, and compares it to the threshold defined by the PER of the current frequency channel, to determine whether the associated PDU is lost or not. This way, the model is able to mimic a BIG receiver attempting to receive incoming subevents under the current network conditions. The outcome stream of frames, containing one or more potential gaps, can be provided to LC3 in order to perform decoding of correctly received frames and potential PLC for lost frames. The PLC technique provided as part of LC3 is based on repeating the MDCT data associated with the last correctly received frame and performing pseudo-random sign scrambling on the spectral coefficients [3].

BLE connections can be used to set up data transfer between a central and peripheral. For instance, a smartphone (central) can set up a low-throughput BLE connection with a smartwatch (peripheral) to inform the user of incoming text messages or calls. In case of a firmware update or larger amounts of sensor data, a (temporary) high-throughput connection could even be required between both entities. Figure 1b contains an example illustration of two subsequent connection events that are a single, pre-configured, connection interval apart. Each connection event consists of one or more packet exchanges and every packet exchange consists of the central transmitting a PDU to the peripheral and vice versa. Each subsequent PDU transmission in either direction should be T_IFS (150 us) apart and each connection event contains at least a single packet exchange between central and peripheral. In case both entities have no application data to transmit, two empty PDUs are transmitted that act solely as keep-alive messages for the connection in both directions (assuming slave latency is 0 [9]). The header overhead for each BLE packet transmitted on the 1 Mbps Uncoded (1M) physical data rate (PHY) is 10 bytes and the payload field can be up to 251 bytes long. A connection event that solely contains a single packet exchange with empty keep-alive messages, requires only 310 us (2 * 80 us on-air time + a single T_IFS). All PDUs in the same connection event are transmitted on the same frequency channel, selected by the CSA #2. The channel selection probability can be considered uniform over all non-blacklisted channels [10].

III. Testbed Framework

A schematic and picture of the testbed can be found in [8]. The broadcaster and receiver are implemented by using two custom boards, each consisting of a dual-core nRF5340



(a) BIG event with 4 subevents (SEs) for a single audio frame.

(b) Illustration of two subsequent connection events in a BLE connection.

Fig. 1: Packet schedule examples for a BIG broadcaster and a BLE connection.

system-on-chip (SoC) module and an external antenna connector mounted on it. The boards are interconnected via several shielded SMA components in between. The link between both nodes is attenuated with 74 dB using various attenuator components and the broadcaster and receiver are enclosed in metal cases in order to limit the potential impact by undesired outside interference as much as possible. The variable attenuator connected to the testbed can introduce an additional attenuation up to 30 dB on the link. BLE interference can be introduced on the link via a splitter, connected to the BLE central and peripheral that are in an active BLE connection (two u-blox BMD-345 evaluation kits, based on the nRF52840 SoC). The Ellisys Bluetooth Vanguard sniffer is used to be able to validate correctness of ongoing BIG and BLE traffic on the testbed. The BLE stack can be divided into two parts: the Controller (containing the physical and link layers) and the Host (containing all upper layers and an application on top). The Host Controller Interface (HCI) is used to enable communication between both parts. For the BLE interference boards, the Host and Controller are implemented on the same nRF52840 thus the HCI is implemented on the microcontroller firmware. For the BIG broadcaster, the Host and Controller are each implemented on one of the cores on the nRF5340 (app or net core) and the HCI transport is provided via the RPMsg Messaging Protocol [11]. The broadcaster uses a custom C application (implemented on top of the Host implementation from Zephyr [12]) on the app core and a closed Controller .hex file from Nordic Semiconductor [13] on the net core. This provided the most stable broadcaster implementation for all BIG configurations that are evaluated in this paper. The BIG receiver uses the native_posix board as a compilation target [14] to implement a custom application on top of the Host as a Linux executable on a PC. The Controller is implemented on the net core of the nRF5340 and the app core contains a bridging application between the Host application on the PC and the Controller on the net core. HCI transport is provided in two steps: a physical UART connection between the PC & the nRF5340 and the RPMsg Messaging Protocol between app & net core. On the net core, the receiver employs either the open-source Zephyr Controller implementation or the closed Nordic Controller. The Zephyr Controller is shown to only provide a stable receiver implementation in case a 1M PHY is used and IRC is set to 1. However, the advantage of Zephyr is that it provides an open-source implementation that can be extended with additional functionality if needed. In order to perform validation of the results against the BIG model, a PER per frequency channel is required to be able to probabilistically assess the occurrence of losses on a packet schedule of a BIG. Therefore, the Zephyr stack is extended to be able to continuously maintain, as a BIG receiver, both the current total amount of Rx attempts on each frequency channel and the amount of times an attempt failed. In the context of Nordic nRF53 devices, this is done mainly by altering the *nordic/lll/lll_sync_iso.c* file in Zephyr [12]. Failed reception and bad CRC are both considered as the same kind of failed Rx attempt. Post-processing of this monitoring information can provide a PER for each frequency channel, to be employed by the BIG model. For IRCs >1, the closed Nordic Controller is used on the net core. The associated BIG model validation uses the PERs per frequency channel that were gathered for IRC == 1 (all measurements were performed in immediate succession).

The broadcaster can contain the contents of multiple LC3 encoded binaries as hard coded strings. Next, it pre-configures the IRC of the BIG and can be instructed to perform experiments for one or more binaries (with a hard coded handling order). For each binary, the same pre-configured amount of experiments is performed. One experiment consists of transmitting all encoded audio frames of the binary on the BIG. The BIG always uses 1M PHY for the experiments presented in this paper and a single batch of experiments performed for all pre-configured binaries, is referred to as an evaluation run. At the start of an evaluation run, dummy data is transmitted on the BIG for a pre-defined initiation time period. The receiver should synchronize to the BIG within that time period in order to start receiving incoming experiment data. After the initiation time period, the broadcaster starts transmitting LC3 frames from the hard coded string of the first binary in the sequence. Both incoming and lost frames on the receiver are written to an output LC3 binary file by means of either the full frame or an empty frame indication. Between experiments for a single binary, the broadcaster transmits dummy data on the BIG for a pre-defined period, to allow the receiver to detect that an experiment has ended and that a new one has started. Upon finishing the last experiment, the broadcaster terminates the current BIG and sets up a new BIG to broadcast the

experiments for the next binary on. This process is repeated until the required amount of experiments is performed for each binary. Using the known sequence of binaries, the receiver can match each received set of experiments to the correct binary. Moreover, additional post-processing is performed on the output binary of each experiment, to add missed leading lost frames that represent experiment data and prune trailing lost frames that actually represent dummy data. The receiver will only detect the start of an experiment upon successful reception of a non-dummy data frame so in case one or more of the first frames of an experiment are lost, these lost frames are not considered as part of an experiment by the receiver. For each experiment, the first 3 correctly received frames (with potential gaps in between) of the output binary are compared against the original binary to find the first occurrence of this sequence. Upon detection of the sequence in the original binary, the number of frames before that sequence are added as additional leading lost frames in the output binary. In case one or more dummy data frames are lost after the experiment ended, the receiver will consider these lost frames as part of the experiment and thus add them to the output binary. Therefore, after adding potential missing lost frames in front of the binary, the number of frames in the output binary is compared against the number of frames in the original binary, to find out the whether one or more of the trailing lost frames should be pruned from the output binary. The employed software implementation and its associated post-processing procedure enables efficient evaluation of a BIG on the testbed, despite the unidirectional nature of the technology.

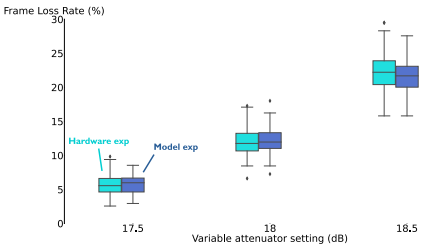
IV. Testbed Evaluation

The evaluation is performed using two distinct LC3 binaries based on encoded audio files (one male and one female) from the PTDB-TUG dataset [15] (both were also part of the 20 distinct audio files used in [5]). To better represent mandatory speech audio use cases, both audio files are downsampled to 16 kHz before encoding. The LC3 encoding parameters are set to a 10 ms frame interval and an encoding bit rate of 32 kbps, which leads to frames of 40 bytes. These are default settings that each BIG observer should support [4], in order for a public BIG broadcaster to know that every listener will at least support these parameters. A single evaluation run performs 50 experiments per binary and each evaluation run is validated on the BIG model as well. The evaluation uses the frame loss rate and instrumental Perceptual Evaluation of Speech Quality (PESQ) metric [16] on the decoded audio of the experiments during the analysis. PESQ can analytically predict the Mean Opinion Score (MOS) of an audio signal similar to how this would be done by a group of listeners (typically between 1.0 and 4.5). The work in [5] illustrates the applicability of the metric to assess the difference in audio quality for various BIG experiments. It includes a sample page [17] that exposes three zones to categorize the PESQ metric in, which can be related to the listening experience in the context of LC3 encoded audio with frame losses: A ($>$ 3.0), B (between 3.0 and 2.0) and C ($<$ 2.0). Decoded speech audio (using LC3 PLC) that is categorized in Zone C, indicates an audible addition of vocoding artefacts, detrimental to the listening experience. In Zone B, the impairment becomes less audible but the drop in audio quality remains annoying. From Zone A, the drop in audio quality becomes less annoying. Further analysis in [5] indicates that the frame loss rate of 2% is shown to be an upper (integer) boundary for almost no PESQ scores of the 20 evaluated audio files being in Zone C.

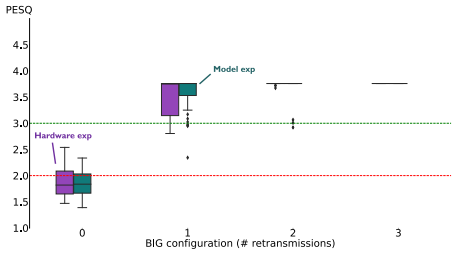
TABLE I: Average frame loss rates for variable attenuation.

Attenuation	IRC	HW Mean (StDev)	Model Mean (StDev)
17.5 dB	1	5.68 (1.55)%	5.69 (1.39)%
	2	0.37 (0.38)%	0.27 (0.34)%
	3	0.02 (0.09)%	0.02 (0.09)%
	4	0.00 (0.00)%	0.00 (0.04)%
18 dB	1	12.05 (2.12)%	12.21 (1.86)%
	2	1.34 (0.69)%	1.42 (0.61)%
	3	0.20 (0.27)%	0.17 (0.27)%
	4	0.03 (0.12)%	0.02 (0.10)%
18.5 dB	1	22.11 (2.80)%	21.71 (2.38)%
	2	5.67 (1.61)%	4.76 (1.29)%
	3	1.38 (0.66)%	1.02 (0.58)%
	4	0.40 (0.44)%	0.27 (0.35)%

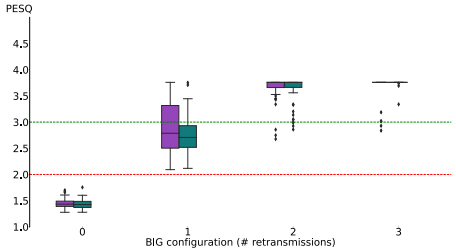
The evaluation runs with variable attenuation use the variable attenuator to change the RF attenuations at 2.4GHz between broadcaster and receiver. This way, the model can be validated for a variety of frame loss scenarios on real hardware. Multiple evaluation runs were performed for 3 distinct variable attenuator configurations (17.5 dB, 18 dB and 18.5 dB), which are added on top of the 74 dB attenuation that is already present. For each configuration, four evaluation runs were performed (IRC 1 to 4). Table I illustrates a clear comparability between the average frame loss rates of the experiments performed on real hardware and the results gathered from the model. It is clear that a higher attenuation requires more frame retransmissions to ensure that the average frame loss rate remains sufficiently below the 2% threshold for PESQ scores not being in Zone C. Figure 2a provides a more detailed overview of the comparability between the frame loss rate distribution pairs for the various variable attenuator configurations that employed IRC 1. Figure 2b, 2c and 2d illustrate the PESQ distributions of the female binary experiments, for the different evaluation runs of 17.5 dB, 18 dB and 18.5 dB configuration respectively. The PESQ distribution pairs are shown to be comparable, especially for higher frame loss rates. It should be noted that only 50 experiments are performed per binary so even more comparable distributions could be achieved by performing additional experiments. Lastly, by adding additional attenuation on the BIG (within an otherwise enclosed testbed), it was expected that the reported PER per frequency channel by the receiver (as described in Section III) would be comparable between all channels. However, the results indicated that some frequency channels experienced a larger PER compared to others [8]. Further investigation is needed to find out whether this is related to undesired



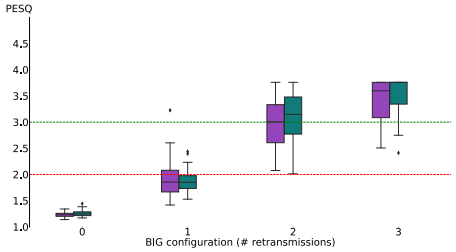
(a) Frame loss rate distribution for no retransmissions.



(b) PESQ distribution for 17.5 db attenuation (female binary).



(c) PESQ distribution for 18 db attenuation (female binary).



(d) PESQ distribution for 18.5 db attenuation (female binary).

Fig. 2: Evaluation of the variable attenuation experiments

TABLE II: Average frame loss rates for BLE interference scenario.

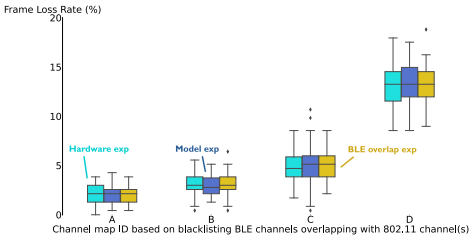
ChM	HW Mean (StD)	Model Mean (StD)	Overlap Mean (StD)
A	2.08 (0.85)%	2.12 (0.87)%	2.08 (0.92)%
B	3.10 (1.05)%	2.94 (0.95)%	3.19 (1.04)%
C	4.97 (1.44)%	4.98 (1.65)%	5.21 (1.42)%
D	13.17 (2.09)%	13.39 (2.03)%	13.29 (2.06)%

interference on the enclosed testbed or it is a hardware-specific issue of the broadcaster and receiver. Regardless, all reported PERs were used as input to the BIG model and provided the comparable results as described above.

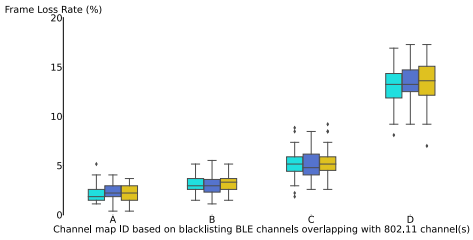
The next evaluation runs will make use of the capability to introduce external BLE interference into the testbed. The Throughput example from Nordic Semiconductor [18] is employed to establish a single BLE connection that is transmitting dummy data from the central to the peripheral. The original example can be instructed (via serial communication) to perform a pre-defined number of transmissions of an application payload of a pre-defined length. This is slightly altered to ensure that the transmissions of the application payload are continuously repeated, in order to achieve continuous interference during the experiments. Using the default configuration of the example, one application payload transmission leads to fragmentation into two BLE packets of 251 bytes payload, per application payload of 495 bytes. The PHY is set to 1 Mbps (to match the BIG) and the connection interval is set to 12.5 ms. The latter is not an integer multiple of the 10 ms ISO_Interval, to avoid experiments with no overlapping connection and BIG events. Based on these configurations, each connection event consists of four on-air packets of 251 bytes payload. Eight evaluation runs are performed, using an IRC of 1 or 2 and a channel map chosen from four distinct options, based on the BLE channels respectively overlapping with no, one (1), two (1 and 11) or all three (1, 6 and 11) non-overlapping 802.11 channel(s). The associated channel maps are respectively: [0xFF, 0xFF, 0xFF, 0xFF, 0x1F], [0x00, 0xFC, 0xFF, 0xFF, 0x1F], [0x00, 0xFC, 0x7F, 0x00, 0x1C] and [0x00, 0x04, 0x60, 0x00, 0x1C], and are referred to as channel map A to D for the remainder of the paper. The variable attenuator is set to 0 dB so no frame loss occurs due to too high attenuation of the BIG link. Table II illustrates the average frame loss rates for the four evaluation runs that employed IRC 1. The results are shown to be comparable for each channel map. Figure 3 illustrates the comparability between frame loss rate and PESQ distribution pairs for the male and female binary separately. The frame loss rate distributions for each binary are shown to be comparable. But Figure 3d illustrates that even for channel map C, the male binary indicates almost no PESQ scores in Zone C, even though Table II indicates an average frame loss rate of 4.97% (for HW). It should be noted that the (integer) frame loss rate threshold of 2% for zone C in [5], is based on a much larger experiment set and on 20 different audio files. Figure 3c and 3d illustrate that, for similar frame loss rate distributions, two distinct binaries experience different behavior regarding PESQ score distribution, so the 2% threshold is shown to be solely applicable as a generic upper boundary for almost no PESQ scores being in zone C.

V. BLE Overlap Model

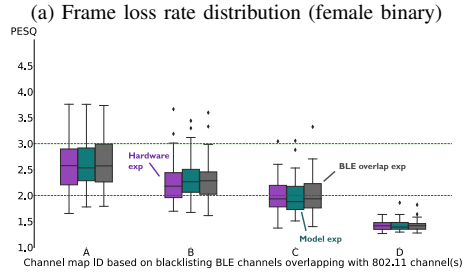
The measurements performed on real hardware for the overlapping BLE connection and using IRC 2, indicate a frame loss rate of 0% for each channel map [8]. In case a connection event overlaps with a BIG event containing two subsequent subevents for the same BIS, there is a 0% probability of both subevents being lost. Overlap in time between the event and both subevents is possible but the CSA#2 ensures that two subsequent subevents from a single BIS event will never use the same frequency channel. The current implementation of



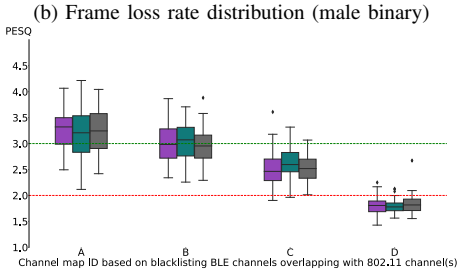
(a) Frame loss rate distribution (female binary)



(b) Frame loss rate distribution (male binary)



(c) PESQ distribution (female binary)



(d) PESQ distribution (male binary)

Fig. 3: Evaluation of BLE interference experiments

the model only employs the PER per frequency channel to simulate network conditions and does not take the context of the overlapping schedules from interference into account. One of the contributions of this paper is to present and evaluate an extension to the model that takes the context of overlapping BLE connections into account. It simulates the packet exchange schedule of consecutive connection events in one or more BLE connection(s), based on a provided scenario. A simulated BLE connection schedule consists of a singular *busy* connection event containing application data, separated by a pre-defined idle period of connection events containing a keep-alive packet exchange in between. Multiple idle periods can be defined for a connection and the simulation switches between them in sequence order after each occurrence of a busy connection event. The model uses microsecond accuracy. Upon startup, the model generates a random starting point of the first connection event, per BLE connection, within a starting window between *-duration of a busy connection event* and *ISO_Interval of the BIG - duration of a busy connection event*, to ensure that the first connection event can also start before the first BIG event that takes place at the simulated time of 0 us. It also generates the context of the first connection event based on random selection of all possible connection events in the first idle period (first one is busy and the rest is idle). In case no idle periods are defined, each connection event can be considered busy. Next, the model assesses potential overlap between the BLE connection(s) and subsequent simulated BIG events. Per BIG event, this process consists of three steps. The first step is to move each BLE connection forward in time until its current connection event either overlaps with the current BIG event or surpasses it. The model uses the defined idle period(s) to match the current connection event to the associated type of connection event (idle or busy). Next, the connection events of the remaining BLE connections that overlap with the current BIG event (in time) are compared to each other to assess whether there is an overlap in time and frequency between radio events. The specification indicates that a connection event is prematurely closed in case the receiving entity (central or peripheral) fails to receive a packet [9]. Hence, overlapping radio events between two connection events cause the remaining radio events after the overlapping radio events to be pruned from both connection events, and will thus not be used when assessing overlap between the connection events and the BIG event. The final step is to actually assess overlap between the radio events of the connection events and the subevents in the BIG event. Subevents overlapping in time and frequency with a radio event, are considered to be lost and the connection event associated with the radio event is prematurely closed. The three steps are repeated until the current connection event of each BLE connection surpasses the current BIG event. The remainder of this section describes two different scenarios that were evaluated on the BIG model, by using the BLE overlap extension. Similarly to before, each evaluation run for a specific scenario and BIG configuration consists of 50 experiments for each binary. The connection employed in the

TABLE III: Average frame loss rates of BLE overlap scenarios.

Scenario	# conn	IRC	Overlap Mean (StDev)
Throughput	5	1	9.90 (1.98)%
		2	0.87 (0.58)%
	10	1	18.52 (2.56)%
		2	3.36 (1.11)%
		3	0.55 (0.49)%
Mice & keyboards	2x15	1	7.44 (2.70)%
		2	0.58 (0.61)%
	2x20	1	10.36 (2.81)%
		2	1.23 (0.74)%
		3	0.13 (0.23)%

first scenario is based on the BLE interference used in Section IV and consists of consecutive busy connection events without any idle period in between. Each connection event contains eight radio events, alternating between 2088 us (containing part of application data) and 80 us (empty packet). Table II illustrates that the BLE overlap model extension with a single connection, is able to replicate the average frame loss rates indicated by the experiments performed on real hardware. Furthermore, the results from the BLE overlap model for IRC 2 are shown to provide a frame loss of 0% for all channel maps, illustrating that this behavior can now also be accurately mimicked compared to the expected behavior on real hardware [8]. Table III indicates BLE overlap results for evaluation runs performed for multiple connections, using channel map A (no blacklisting) for the connections and BIG. For 5 overlapping connections, it is shown that a single retransmission is still sufficient to achieve an acceptable frame loss rate. Going up to 10 overlapping connections requires 2 retransmissions to get an acceptable frame loss rate. The second scenario evaluates multiple wireless BLE mice and keyboards being active at the same time. The mouse and keystroke data transmitted on the connections, is based on the data formats defined in [19]. For a keyboard and mouse, this is respectively 8 bytes (7 keys + modifier) and 3 bytes (X, Y and buttons). Both connections use a connection interval of 7.5 ms. The mouse is simulated as a continuously busy connection (no idle period) and each connection event consists of two radio events (80 and 160 us) with a total duration of 390 us. The keyboard uses two idle periods based on the time between pressing a key down & releasing it (A) and the time between releasing a key and pressing the next one (B). Continuous typing is considered, based on an inter-keystroke interval of 135 ms [20], between subsequent key presses (i.e. A + B). The evaluation assumes that A = B / 2 thus the idle periods are set to 45 and 90 ms respectively. The busy connection event contains 2 radio events (80 and 200 us) with a total duration of 430 us. Table III illustrates that, for up to 15 overlapping keyboards and mice (30 in total), a single retransmission is sufficient to achieve an acceptable frame loss threshold. For (2x) 20 overlapping connections, two retransmissions are required. In general, using 1 or 2 retransmission(s) is shown to be sufficient for various realistic use cases (excluding other interference or network conditions). Additional experiments can be performed to find out the min. # connections that require 2 retransmissions for both scenarios and what the additional impact is of more restrictive channel maps.

VI. Conclusion

The insights gained from the BIG model are shown to mimic the behavior that is observed when performing measurements on a controllable testbed. For variable attenuation settings, the PESQ and frame loss rate distributions are shown to be comparable, while using the original BIG model [5]. For BLE interference from a single connection, a similar conclusion can be made for IRC 1 and various channel maps. The results for IRC >1 expose a limitation of the original model [5] when comparing against the specific interference schedule of a single BLE connection. The proposed BLE overlap model extension is shown to be able to address this gap. Moreover, it can also be used to provide insights into BIG behavior when multiple interfering BLE connections are present. Future work could extend the BLE overlap model with additional functionality that is currently left out of scope and the BIG model could be further validated for other configurations that, for instance, employ fragmentation or use other PLC techniques.

References

[1] Engadget, "Bluetooth Auracast is everywhere at CES 2024. So, what is it?" https://www.engadget.com/bluetooth-auracast-is-everywhere-at-ces-2024-so-what-is-it-224509414.html?, Jan 2024. [Online].

[2] M. Afaneh, "A Technical Overview of LC3," https://www.bluetooth.com/blog/a-technical-overview-of-lc3/, Nov 2020. [Online].

[3] Bluetooth SIG, "Low Complexity Communication Codec," Sep 2020. [Online].

[4] N. Hunn, "Introducing Bluetooth LE Audio," Independently published, Jan 2022.

[5] M. Baert, B. Moons, J. Pittevils, Y. Song, N. Madhu, and J. Hoebeke, "Evaluation of BLE-based audio broadcasting under probabilistic interference," *Computer Communications*, 2024.

[6] J. Lee, "Broadcast Audio Transmission for Bluetooth LE on an Interfered ISM Band," *IEEE Internet of Things Journal*, 2019.

[7] L. K. Baghel, R. Raina, and S. Kumar, "Evolution of bluetooth classic audio towards bluetooth le audio: Challenges and road ahead," in *2023 IEEE 17th International Conference on Industrial and Information Systems (ICIIS)*, 2023.

[8] M. Baert and J. Pittevils, "Dataset," https://gitlab.ilabt.imec.be/publicble2avrepos/datasetbigmodelvalidation.

[9] Bluetooth SIG, "Core Specification 5.2," Dec 2019. [Online].

[10] B.-Z. Pang, T. Claeys, D. Pissoort, H. Hallez, and J. Boydens, "Comparative study on afh techniques in different interference environments," in *2019 IEEE XXVIII International Scientific Conference Electronics (ET)*.

[11] "Developing with nRF5340 DK," https://developer.nordicsemi.com/nRF_Connect_SDK/doc/2.5.1/nrf/device_guides/working_with_nrf/nrf53/nrf5340.html#developing-with-nrf5340-dk, accessed: 2024-02-21.

[12] "Zephyr RTOS (commit 4433ff71)," https://github.com/zephyrproject-rtos/zephyr/tree/4433ff7154ffcc0342412ae52546346e210d75b9, accessed: 2024-02-07.

[13] "LE Audio controller for nRF5340 (version 3349)," https://developer.nordicsemi.com/nRF_Connect_SDK/doc/2.5.1/nrf/libraries/bin/bt_ll_acs_nrf53/index.html, accessed: 2024-02-07.

[14] "Bluetooth tools: Running on qemu and native posix," https://docs.zephyrproject.org/2.7.5/guides/bluetooth/bluetooth-tools.html#bluetooth-qemu-posix, accessed: 2024-02-21.

[15] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "A Pitch Tracking Corpus with Evaluation on Multipitch Tracking Scenario." in *Interspeech - International Conference on Spoken Language Processing*, 2011.

[16] International Telecommunication Union, "ITU-T Rec. P. 862. perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs." Feb 2001. [Online].

[17] M. Baert, "Audio Samples," https://imec-idlab.github.io/MultTransSamplePage/.

[18] "Bluetooth: Throughput (nRF Connect SDK 2.5.1 documentation)," https://developer.nordicsemi.com/nRF_Connect_SDK/doc/2.5.1/nrf/samples/bluetooth/throughput/README.html, accessed: 2024-02-01.

[19] "Device class definition for hid 1.11," https://www.usb.org/document-library/device-class-definition-hid-111, accessed: 2024-01-19.

[20] J. Duprez, M. Stokkermans, L. Drijvers, and M. Cohen, "Synchronization between keyboard typing and neural oscillations," *bioRxiv*, 2020.