An Exponential Back-off Algorithm Based Interference Avoidance Strategy for Bluetooth Low Energy against Wideband Interference

Bozheng Pang
Department of Computer Science
KU Leuven
Brugge, Belgium
bozheng.pang@kuleuven.be

Tim Claeys
Department of Electrical Engineering
KU Leuven
Brugge, Belgium
tim.claeys@kuleuven.be

Hans Hallez
Department of Computer Science
KU Leuven
Brugge, Belgium
hans.hallez@kuleuven.be

Jeroen Boydens
Department of Computer Science
KU Leuven
Brugge, Belgium
jeroen.boydens@kuleuven.be

2022 International Symposium on Electromagnetic Compatibility – EMC Europe | 978-1-6654-0788-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/EMCEUROPE51680.2022.9901045

***Abstract*—Wireless connection reliability is always a top priority in Internet of Things systems, especially in a harsh electromagnetic environment. As a prominent wireless communication protocol in the 2.4 GHz frequency band, Bluetooth Low Energy always has compatibility challenges with wideband protocols working in the same frequency band such as Wi-Fi. In this paper, we present a lightweight strategy for Bluetooth Low Energy devices to cope with the compatibility issues with wideband interference. The strategy is based on the exponential back-off algorithm and is implemented mostly in the Bluetooth Low Energy application layer. The results of the experiments show that the strategy can deal with both static and dynamic wideband interference, reducing packet loss rates to around 3% and 5%, respectively.**

***Index Terms*—Bluetooth Low Energy (BLE), wideband interference, back-off algorithm, reliability**

I. Introduction

Internet of Things (IoT) technologies rely heavily on wireless communication [1]. There are various wireless protocols available for IoT applications, such as Bluetooth Low Energy (BLE), a representative of narrowband protocol, and Wi-Fi, a wideband representative [2].

BLE is a prominent wireless communication protocol that operates in the 2.4 GHz frequency band and is frequently used in IoT systems like a smart home [3]. As a result, interference is a well-known wireless communication challenge for it [4]. Due to differences in communication features, such as channel frequency range, BLE is more susceptible to wideband interference [5].

To deal with the wideband interference, BLE applies a strategy called adaptive frequency hopping (AFH) [6]. Inside the AFH strategy, two channel selection algorithms (CSAs) are defined by the BLE specification [7]. They are used to improve the BLE connection reliability by dividing the 2.4 GHz frequency band into 37 data channels and hopping pseudo-randomly inside. However, the current CSAs have been proved lack of efficiency under interference, especially wideband interference. To improve performance of the CSAs, we proposed an improved CSA in our previous work and evaluated it under both static and dynamic wideband interference [8]. The improved CSA has demonstrated its capacity to improve the reliability of a BLE connection while wideband interference is present [9]. Our improved CSA, however, requests some changes to the BLE specification and is not yet included in the BLE specification. As a result, we present an exponential back-off algorithm based strategy for BLE communications against wideband interference in this paper. Although this strategy has been found to be less efficient than the proposed improved CSA, it does not necessitate any changes to the CSAs defined by the current BLE specification. Hence, it is more compliant with the BLE specification and easier to implement, although some counters in the link layer are necessary.

The remainder of this paper is arranged as follows. The principle and details of the methodology are explained in Section II. The experimental setup used to examine the methodology under wideband interference is shown in Section III, and Wi-Fi is chosen as a representative of wideband interference. The results are discussed in Section IV. Conclusion and future work are mentioned in Section V.

II. Methodology

This methodology is designed to improved the reliability of BLE connections. Its foundations are based on the strengths of the methodology that was implemented in related work [8]. First, the needed PDR values are discussed. Secondly, the blacklisting mechanism for this methodology will be outlined in detail. Finally, the whitelisting mechanism is explained.

A. Packet Delivery Ratio

Some parameters are needed to measure the performance of the original application and the application that has been updated with the newly designed methodology to detect and avoid interference. In this manner it is possible to check if the methodology works while implementing it, and it could be constructed as a reference for debugging while implementing. Besides, these parameters are also needed for showing how the end result performs. Comparing PDR values of the whole connection for the original and the updated application, or comparing PDR values of individual channels, can show if interfered channels are indeed avoided.

First, the connection PDR for the whole connection (*CONN PDR$_{CONN}$*) is measured by incrementing the number of transmitted packets (*#TX*) by one for each packet the central sends to the peripheral. If a valid acknowledgement is received for the transmitted packet, then the number of acknowledgements (*#ACK*) is also incremented by one (see equation (1)).

$$\text{CONN PDR}_{\text{CONN}} = \frac{\#\text{ACK}}{\#\text{TX}} \tag{1}$$

Second, the channel PDRs for the whole connection (*CHAN PDR$_{CONN}$*) can be seen as an array of 37 PDR values. For each data channel individually, the *#TX* and *#ACK* are counted. *#TX[k]* is incremented by one if channel *k* is used to transmit one packet from the master to the slave. If a valid acknowledgement is received for the transmitted packet, the *#ACK[k]* is also incremented by one (see equation (2)).

$$\text{CHAN PDR}_{\text{CONN}}[\text{k}] = \frac{\#\text{ACK[k]}}{\#\text{TX[k]}} \tag{2}$$

With the two PDRs introduced before, there are two parameters to discuss, which are the moving average channel PDRs (*CHAN PDR$_{MOVAVG}$*) and the moving average connection PDR (*CONN PDR$_{MOVAVG}$*).

First, the *CHAN PDR$_{MOVAVG}$* can be represented as an array. An array with 37 elements represents the 37 data channels. Each element of this array has its own *CHAN PDR$_{MOVAVG}$*. This moving average PDR for each channel is needed to have an accurate and recent picture of the environment. Each time a channel is picked by the selection algorithm its moving average PDR will be updated. To implement this moving average, only a certain amount of the most recent PDR values are needed. This can be realized by designing a circular buffer [10]. Each channel has its own circular buffer that is as big as the window length for computing the moving average. The window length can be changed by developers/designers according to the environment, and it is set to 50 in this paper.

Suppose a packet is sent from the master to the slave. If a valid acknowledgment is received, then the PDR for this single transmission is equal to 100%. If not, the PDR is 0%. The PDR for a single transmission will be represented as either 0 (0%) or 1 (100%) in the buffers. Suppose the buffer of a certain channel *k* is full (as many elements present as the window length) and the master sends another packet via the channel *k*. The very first element that was written in this buffer is replaced by this PDR of this new single transmission. In short, if the buffer is full and a new transmission takes place, the oldest value in the buffer is replaced. Therefore, it is called a circular buffer, which is shown in Fig. 1 as an example.

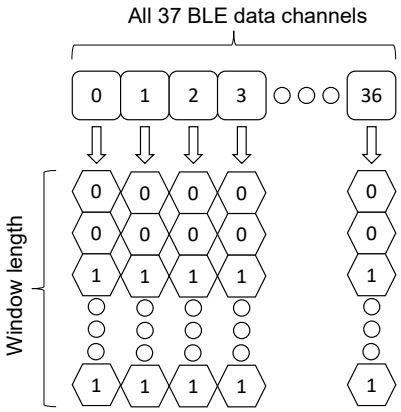

Fig. 1. An illustration of circular buffers.

Second, the *CONN PDR$_{MOVAVG}$* is the threshold on which a channel is classified as either good or bad. It is computed by adding the moving averages of all 37 data channels together and dividing the sum by 37 (see equation (3)). Important to know is that during the period a channel is blacklisted its moving average does not change, instead, it freezes. Because all 37 moving averages are used, these frozen moving averages are also involved in computing the *CONN PDR$_{MOVAVG}$*. In this manner, an accurate picture of the environment is created and thus this value will be used as a threshold for blacklisting.

$$\text{CONN PDR}_{\text{MOVAVG}} = \frac{\text{sum(CHAN PDR}_{\text{CONN}})}{37} \tag{3}$$

B. Blacklisting

Suppose a new connection event takes place. First, the CSA decides a certain channel *k* must be used for a single transmission (TX and ACK) during this upcoming connection event. Thereafter, the transmission is issued by the central. Then, the mechanism checks if it was successful (=ACK received) and updates the circular buffer for channel *k* accordingly. After the moving average PDR for this channel is calculated, it is compared to the moving average connection PDR. If *CHAN PDR$_{MOVAVG}$[k]* is smaller than *CONN PDR$_{MOVAVG}$* the channel must be blacklisted. This process is repeated for each new connection event.

With a mechanism based on the exponential backoff algorithm used in Carrier Sense Multiple Access - Collision Avoidance / Collision Detection (CSMA-CA/CD) [11], our blacklisting mechanism is designed. A channel *k* will be blacklisted for a certain number of connection intervals. To determine this number, a counter named *BL_COUNTER* records the exponent for each channel. The higher this counter becomes, the more connection events a channel must wait before being whitelisted again (see equation (4)). In this paper,

the *BL_COUNTER* is initialized to 7, and it increases by 1 each time a channel is blacklisted.

$$\#\text{CONN EVENTS[k]} = 2^{\text{BL\_COUNTER[k]}} \tag{4}$$

C. Whitelisting

A channel that is blacklisted will be whitelisted when it has gone through the certain number of connection intervals calculated by equation (4). When the channel is blacklisted, there is no way of measuring its channel quality by PDR. Therefore, the circular buffer used to store PDR information of that channel is reset. This means that all the buffer values of that channel are set to one, so *CHAN PDR$_{MOVAVG}$* is 100%. This is also to keep *CONN PDR$_{MOVAVG}$* on a high level, hence a high threshold to classify channels is achieved. After that, the mechanism is allowed to collect fresh data and thus able to accurately estimate the channel quality again.

Blacklisted channels must stay inactive for a certain amount of connection events calculated by the exponential back-off algorithm. The CSAs are called to provide a channel for each connection interval, thus they are used to count the amount of connection intervals that have taken place during the connection. If the amount of connection intervals hits zero for a blacklisted channel, the according flag in an array is set. The flag array is checked in the notify function on the central side. If one flag is set in the array, the central will update the channel map to the peripheral. The whitelisting mechanism is implemented analogously to the blacklisting mechanism. The only difference is that the whitelisting flags are now reset instead of the blacklisting ones. Also, there is no need to compute the amount of connection intervals.

Except the circular buffer, the blacklist counter also needs to be considered. Without taking care of *BL_COUNTER*, a possible issue arises. With the connection going on, the *BL_COUNTER* of a channel may become higher and higher due to the interference. Assume the interference suddenly drops and the blacklisted channel eventually is whitelisted once it has waited out its connection intervals. However, due to its high *BL_COUNTER*, the channel has to be blacklisted for a long period again even for some sporadic interference, although this channel is actually of good quality. Hence, a proper management is necessary for *BL_COUNTER*.

The management solution can be summarized as: if a channel has been interfered with for a long time but becomes interference-free for a longer period, its ”bad” history should be cleared to avoid exaggerated back-off time. The probability a channel is selected by the CSAs can be considered as nearly uniformly distributed [6, 12]. If there are 37 channels active, that probability is equal to $\frac{1}{37}$. If there are only 5 channels active, the probability a channel is selected is $\frac{1}{5}$. When a channel is whitelisted, we take the number of connection intervals x the channel was blacklisted for. Then we divide x by the number of active channels at the time the channel is whitelisted. The result of this division named z should about equal the number of times this channel could be selected in the upcoming x connection events. z is the number of successful transmissions that must happen in a continuous way on this newly whitelisted channel to have its *BL_COUNTER* reset. Thereby the ”bad” history of the channel can be reset. In this manner, the following logic is established. The longer the interference is presented for a specific channel, the more connection events this channel must stay inactive for. This then also results in a big continuous sequence of successful transmission this channel has to go through to have its *BL_COUNTER* reset. In short, the worse the environment, the longer those channels are probed before clearing its ”bad” history.

III. Experimental Setup

To examine the methodology, some experiments are set up. The experimental setup introduced in this section aims at validating the introduced methodology and investigating the effectiveness and the efficiency of it.

The experimental setup is similar to the one reported in [8]. It includes two BLE development boards (nRF52840 DK) and one Raspberry Pi 3 Model B [13, 14]. The two nRF52840 boards are used to build a BLE connection. The operating system implemented on the two BLE boards is Zephyr RTOS [15]. The Raspberry Pi is used to generate wideband interference which is Wi-Fi. To generate controllable interference, JamLab-NG is implemented, which enables the generation of repeatable and reproducible Wi-Fi interference using off-the-shelf Raspberry Pi 3 devices [16].

The experiments are conducted in a quiet office environment. The two BLE boards are placed close to one another with a distance of around 15 cm, while the Raspberry Pi is placed next to the BLE boards. The distance from the Raspberry Pi to the boards is 30 cm. The Wi-Fi interference from the Raspberry Pi is the primary source of interference in the experiments. Three types of environments are used in the experiments, which are interference-free, static-interference and dynamic-interference environments. The Raspberry Pi is off in the interference-free environment. In the static-interference one, the Raspberry Pi generates a strong continuous Wi-Fi signal throughout the whole experiment on Wi-Fi channel 1, which overlaps with BLE channels 0 to 9. During the experiment with the dynamic interference, the Raspberry Pi switches the Wi-Fi interference among Wi-Fi channels 1, 6, and 11 randomly. The JamLab-NG allows the Raspberry Pi to occupy only one Wi-Fi channel at a time. Hence, in the dynamic-interference environment, the Wi-Fi interference stays on a Wi-Fi channel for a random time period and moves to another. The Wi-Fi channel is chosen randomly by the Raspberry Pi with no predefined sequence. The time period is also random, while limited between 1 second and 10 seconds. In our experiments, only one Raspberry Pi is used. A harsher environment can be achieved by adding more Raspberry Pi devices, however, not shown in this paper.

Each experiment begins with the Wi-Fi interference generation from the Raspberry Pi. According to the three different environments defined, the Raspberry Pi generates different interference or stays silent. After that, a BLE connection starts with the establishing of a connection between the two BLE

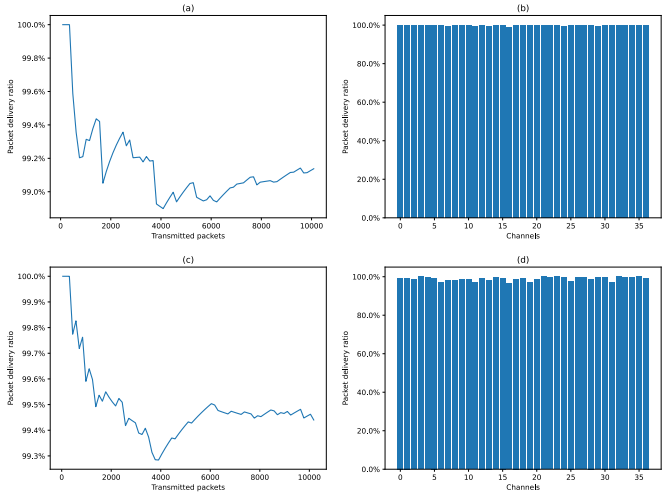

Fig. 2. Results of connection PDR and channel PDR under an interference-free environment. (a) original BLE stack connection PDR under the interference-free environment. (b) original BLE stack channel PDR under the interference-free environment. (c) modified BLE stack connection PDR under the interference-free environment. (d) modified BLE stack channel PDR under the interference-free environment.

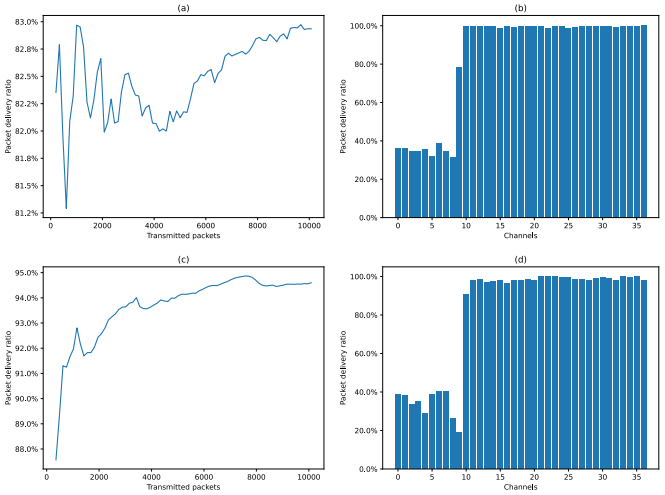

Fig. 3. Results of connection PDR and channel PDR under a static interference environment. (a) original BLE stack connection PDR under the static interference environment. (b) original BLE stack channel PDR under the static interference environment. (c) modified BLE stack connection PDR under the static interference environment. (d) modified BLE stack channel PDR under the static interference environment.

boards and ends when the number of connection intervals reaches 10000. Note that the BLE connection can only be established after the Wi-Fi interference is generated. In this way, the BLE connection is made sure to work fully under the Wi-Fi interference. In the experiments, the circular buffers are set as 50 elements big for testing purpose. The moving averages are thus calculated with the 50 elements in the buffer. The following are the BLE connection parameters that are used in the experiment. By default, both the central and peripheral transmission powers are set to 0 dB. 2M PHY is the physical mode employed. The connection interval is set to 7.5 ms, which is the minimum value in the BLE standard, to speed up the experiments.

IV. Results and Discussion

To show the effectiveness and efficiency of the introduced methodology, there are three experiments needed. First, the experiment is conducted in the interference-free environment, and the results are treated as a baseline of all the other experiments. Second, the methodology is implemented and tested under static wideband interference generated by the Raspberry Pi. Last, the wideband interference is changed into a dynamic one to further examine the methodology. After the explanation for the three experiments, we further discuss our proposed methodology for its possible usage, advantages, limitations, etc.

A. Interference-free Environment

As a baseline, the interference-free environment is firstly used, and the results are shown in Fig. 2. For both the original BLE stack and the modified one, Of the total of around 10000 packets that are sent by the master, only around 1% of them are lost. That results in a connection PDR of 99%. Figs. 2(a) and (c) show that the PDR of the overall connection never dropped below 98.8%. The individual channel PDRs in Figs. 2(b) and (d) demonstrate the excellent quality of all channels because of the interference-free environment. This data also provides a reference to compare all the following results with.

In addition, these results reveal the randomness of the CSAs (CSA #2 in the test) used by BLE. In an interference-free environment, the CSA #2 provides a quite similar probability for each channel to be chosen, thus the channel usage distribution is even in the results.

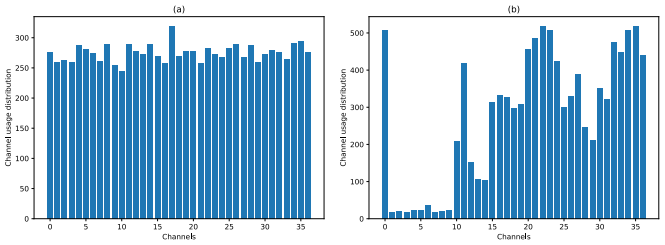

Fig. 4. Channel usage distribution under a static interference environment. (a) channel usage under the static interference environment using the original BLE stack. (b) channel usage under the static interference environment using the modified BLE stack.

B. Static Interference

After the tests in the interference-free environment, tests under the static interference are performed. The results of both the original BLE stack and the modified one are illustrated in Fig. 3. As can be seen in the figures, the connection PDRs between the original BLE stack and the modified one are no longer identical. The original BLE stack can only achieve a maximum PDR of 83% with static interference, whereas the modified BLE stack can reach a value of 95%. When compared

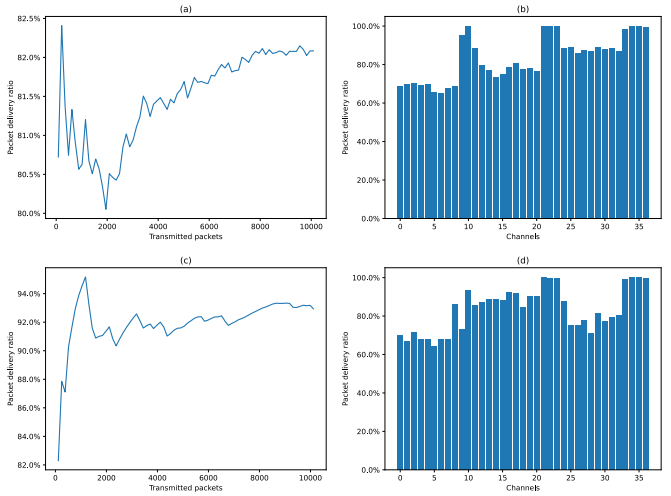

Fig. 5. Results of connection PDR and channel PDR under a dynamic interference environment. (a) original BLE stack connection PDR under the dynamic interference environment. (b) original BLE stack channel PDR under the dynamic interference environment. (c) modified BLE stack connection PDR under the dynamic interference environment. (d) modified BLE stack channel PDR under the dynamic interference environment.

to interference-free values, the original BLE stack performs 16% worse, and the modified version only performs 4% worse. Comparing the PDR curves in Fig. 3, the curve (c) tends to rise and is smoother, which illustrates the stability of using the proposed methodology.

When it comes to the channel PDR, both versions of the BLE stack have a similar feature as shown in Figs. 3(b) and (d). The PDRs on BLE channels 0 to 8 are close to 40% in both figures. However, they produce different connection PDRs of 83% and 95%, respectively. The reason for this is that while using the modified BLE stack, fewer packets are transferred on BLE channels 0 to 8. For instance, the 40% of using the original BLE stack is calculated by $\frac{95}{262}$, while the one from the modified BLE stack is from $\frac{8}{21}$. As an example, the channel usage of both versions of BLE stack is shown in Fig. 4.

C. Dynamic Interference

Following the static interference, a dynamic interference environment is employed to further assess the implemented methodology's performance limits. The experimental results in the dynamic interference environment are shown in Fig. 5. The connection PDRs are 82% and 93%, respectively, when employing the original BLE stack and the modified one. When compared to values obtained in the interference-free environment, the original BLE stack loses 17% of its reliability, while the modified BLE stack loses only 6%.

In Fig. 5(b) and (d), the BLE data channels overlapped with Wi-Fi channels 1, 6, and 11 exhibit a low channel PDR. The original BLE stack and the modified one produce similar channel PDRs. They results in different connection PDRs due to the channel usage difference, similar to the results under static interference.

D. Discussion

Our experiments and results provide BLE users and developers an idea about the BLE connection reliability under interference, especially Wi-Fi interference. Since the BLE specification does not specify a standard methodology to detect interference and distinguish channel quality, the developers/vendors may implement our methodology or a part of it into their products based on the application and environment.

The measured performance certainly depends on the used interference signal and on the traffic load for the current measurement setup. Nevertheless, the findings are interesting and solid work. Besides, reliability is only one aspect of BLE communications. Other aspects of the communication performance, such as latency and throughput, may be impacted by the proposed methodology. They are not measured in the experiments, but are theoretically discussed here to give some insights. Due to the increase of the reliability, less packets are corrupted/lost and retransmitted. As a result, the throughput is supposed to be improved and closer to the maximum value. Same to latency, the average latency should be decreased since less packets are lost or retransmitted. However, the time used for pre-processing and post-processing in a BLE connection is expected to be extended because of the extra computation introduced by the methodology.

V. Conclusion

In conclusion, this research introduces a methodology for improving BLE connection reliability in the presence of wideband interference. The results show that incorporating the exponential back-off algorithm into the BLE stack makes the connection more reliable. BLE connection reliability improves by around 11% to 12% under both static and dynamic wideband interference. In comparison to some previous studies, such as [8], the methodology does not need changing the channel selection algorithms from BLE standard, making it more compatible with the current BLE specification.

As future work, various elements in the proposed methodology could be investigated further to discover how they affect efficiency and efficacy of the methodology. Narrowband interference, such as BLE and Zigbee, can be examined as well.

Acknowledgment

The authors would like to thank Lowie Lameire for his help in developing and implementing the exponential back-off algorithm based interference avoidance strategy.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128610001568

[2] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of Things (IoT) communication protocols: Review," in *2017 8th International Conference on Information Technology (ICIT)*, May 2017, pp. 685–690.

[3] M. Collotta and G. Pau, "A Solution Based on Bluetooth Low Energy for Smart Home Energy Management," *Energies*, vol. 8, no. 10, pp. 11 916–11 938, Oct. 2015, number: 10 Publisher:

Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1996-1073/8/10/11916

[4] B. Pang, T. Claeys, D. Pissoort, H. Hallez, and J. Boydens, "A Study on the Impact of the Number of Devices on Communication Interference in Bluetooth Low Energy," in *2020 XXIX International Scientific Conference Electronics (ET)*, Sep. 2020, pp. 1–4.

[5] J. Wyffels, J.-P. Goemaere, B. Nauwelaers, and L. De Strycker, "Influence of Bluetooth Low Energy on WIFI Communications and Vice Versa," in *ECUMICT 2014*, ser. Lecture Notes in Electrical Engineering, L. De Strycker, Ed. Cham: Springer International Publishing, 2014, pp. 205–216.

[6] 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)*, Sep. 2019, pp. 1–4.

[7] "Bluetooth Core Specification v5.3," p. 3085.

[8] B. Pang, K. T'Jonck, T. Claeys, D. Pissoort, H. Hallez, and J. Boydens, "Bluetooth Low Energy Interference Awareness Scheme and Improved Channel Selection Algorithm for Connection Robustness," *Sensors*, vol. 21, no. 7, p. 2257, Jan. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/7/2257

[9] B. Pang, T. Claeys, J. Vankeirsbilck, K. T'Jonck, H. Hallez, and J. Boydens, "A Probability-based Channel Selection Algorithm for Bluetooth Low Energy: A Preliminary Analysis," in *2021 XXX International Scientific Conference Electronics (ET)*, Sep. 2021, pp. 1–4.

[10] "Circular buffer," Dec. 2021, page Version ID: 1062264742. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Circular_buffer&oldid=1062264742

[11] M. M. Rahaman, K. Ashrafuzzaman, M. S. Chowdhury, and M. O. Rahman, "Performance measurement of different backoff algorithms in IEEE 802.15.4," in *2016 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, Oct. 2016, pp. 1–4.

[12] M. O. Al Kalaa and H. H. Refai, "Selection probability of data channels in Bluetooth Low Energy," in *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2015, pp. 148–152, iSSN: 2376-6506.

[13] "nRF52840 DK." [Online]. Available: https://www.nordicsemi.com/Products/Development-hardware/nRF52840-DK

[14] R. P. T. Ltd, "Buy a Raspberry Pi 3 Model B," publication Title: Raspberry Pi. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-3-model-b/

[15] "Zephyr Project," publication Title: Zephyr Project. [Online]. Available: https://www.zephyrproject.org/

[16] "JamLab-NG," Jun. 2021. [Online]. Available: https://github.com/TuGraz-ITI/JamLab-NG