Bluetooth Low Energy for Data Streaming: Application-level Analysis and Recommendation

D. Giovanelli, B. Milosevic, E. Farella
E3DA - ICT Center, Fondazione Bruno Kessler (FBK), Trento, Italy.
Email: {dgiovanelli, milosevic, efarella}@fbk.eu

***Abstract*—Wearable devices enable the unobtrusive sensing of a wide range of human activities and the development of innovative applications. Wireless communication capabilities are one of the crucial features of networked wearable devices, since they set data exchange specifications and heavily affect the power consumption of the devices. In this paper, we investigate the use of the Bluetooth Low Energy (BLE) standard for the development of wearable devices with elevated data throughput requirements such as Health-related applications. In particular, we focus on the application-level and analyze the performance of different BLE modules when interfaced with a smartphone in single- and multi-slave configurations. Data throughput and power consumption are evaluated in relation to the parameters of the protocol and we present to the reader useful guidelines for an optimal integration of BLE in devices with data streaming capabilities.**

I. Introduction

In the last years, wearable devices are receiving a lot of interest for their versatility and ease of use, with applications ranging from fitness to healthcare. The most frequent scenario is the use of wearable Body Sensor Networks (BSNs) for the monitoring and analysis of selected features of the human body. Dedicated signal processing techniques extract high level information from sensor data, both in real time and offline, providing useful insights of the user's state and performance. In particular, healthcare applications have very restrictive quality of service requirements in terms of data accuracy, latency and operation time [1].

A common BSN architecture is formed by multiple sensing nodes and one central unit, acting as data collector, processing hub and gateway towards further systems. The sensing nodes usually have a very limited dimension to be comfortably worn on the point of interest, hence they are carefully designed for wearability embedding the very needed sensing and communication technologies [2].

While in the past years there was research and development of custom wearable central units, now smartphones are the standard option in the majority of cases. This is supported by several reasons: their cheap availability and the possibility for the development of custom applications, the ease of use and the robustness of such solutions and their ever growing computational and communication capabilities [3].

Radio interfaces and communication protocols have always been a crucial part of BSNs. Their choice determines the nature of the data that the network will be able to exchange, setting the BSN specifications in term of supported number of nodes and data throughput or latency [4]. Healthcare applications have high data fidelity requirements, therefore the identification of the wireless protocol introduces additional challenges when compared to consumer applications and more general sensor networks where data can be sent more infrequently [5]. Moreover, the communication subsystem is usually the most power-consuming one, hence the need for its optimization and energy efficient power management [6, 7].

Since the beginning of the development of BSNs, several standard radio protocols have been employed, including Bluetooth, ZigBee, ANT [8], along with ad-hoc proposed solutions [9]. With the diffusion and adoption of smartphones, it has become more convenient to rely on Bluetooth as the standard protocol supported by almost all mobile phones and tablets, eliminating the need for additional hardware. This resulted in a wide adoption of Bluetooth for the wearable nodes, even with its limitations in terms of number of devices and high power consumption.

The recent development of the Bluetooth 4.0 standard (also called Bluetooth Low Energy or BLE) was directly targeted to increase the energy efficiency and device connectivity of such protocol, with positive impact for wearable devices and BSNs for both consumer and medical applications [10]. However, it is not compatible with previous versions of the standard, having different stack layers and data exchange protocols and its low energy optimized profiles result in a reduced data rate.

In this work, we perform a detailed analysis of the BLE protocol from an application point of view and we evaluate its use for a wide range of wearable devices. We target data streaming applications with high throughput and reliability requirements, such as wearable healthcare applications, and we explore the capabilities of this protocol in relation to its connection parameters. In particular, we will evaluate and compare the data throughput and latency with the associated power consumptions for three widely used BLE chips.

We consider a BSN scenario with up to 6 wearable devices interfaced with a smartphone and we focus on application-level development. While existing literature relies mainly on simulation and hardware/software customization of the used devices, we target real-life scenarios for application development, avoiding the need for hardware or stack development. The paper discusses the observed results and provides indications if BLE is suitable for an application and guidelines for efficient deployment of BLE based BSNs.

In the following Section II we introduce the related works and, for the reader's convenience, we present an overview of the BLE protocol focusing on the top layers of the stack in Section III. Power consumption is analyzed and modeled in Section IV, while Section V describes the implemented BSN. The experimental setup and results are reported in Section VI and discussed in Section VII, together with guidelines for an optimal BLE performance. Finally, the last Section VIII concludes the work.

978-1-4799-8981-2/15/$31.00 ©2015 IEEE

II. Related work

Since the introduction of BLE in 2010, its benefits in terms of connectivity and low power consumption have been exploited in BSNs, with several applications: motion sensing [11, 12, 13], ECG and biopotential sensing [14, 15, 16, 17], blood pressure measurement [18]. BLE is optimized for low power applications and its power consumption has been compared to competing communication protocols (Bluetooth, ZigBee and ANT) [8]. In particular, the comparison with older Bluetooth versions highlights the energy benefits of this widely adopted solution [14, 17].
The drawback of the BLE solution is the relatively low data throughput that a device can achieve. In [19], a model for throughput estimation is presented with a calculated maximum throughput of 236.712 $kbit/s$, but it does not take into account the limited hardware resources available on real BLE chips. Furthermore, the calculation is done in master-to-slave direction and a master device usually has significantly less hardware/software constraints than a BSN node. In [20], a BLE performance analysis is accomplished and compared with results obtained from other wireless transmission protocols. For BLE they report a maximum experimental link layer throughput of 122.6 $kbit/s$, but this work did not use the upper BLE stack layers and the achieved throughput considers also the packet overhead, which does not contribute to the application data throughput. In [21] similar theoretical results have been reported with the additions of experimental tests, which highlights that with a real device the application level throughput is 58.48 $kbit/s$. In this paper two identical modules are used for master and slave devices, with custom software on both ends.
To allow the connected devices to effectively employ duty cycling strategies, data exchange in BLE can only be performed sending data packets on pre-determined connection events. Given the application requirements, different choices of the period of the connection events and the number of packets to send at each event can satisfy the needed throughput, affecting in different ways the overall performance of the system. While theoretical and preliminary studies on the connection parameters optimization have been carried out [22], the real application-level performance, which is dependent on the hardware and stack employed, has not been carried out. Therefore, we investigate the performance in a realistic scenario analyzing different commonly used BLE chips for BSN nodes and using an Android smartphone with its default stack. Moreover, we also evaluated a multi-slave scenario, where up to 6 data streaming nodes were connected to a smartphone, analyzing the network performance in relation to the choice of the protocol parameters.

III. Overview of the BLE communication protocol

BLE is the last version of the protocol specification designed by the Bluetooth SIG and first released in 2010 [23]. It is targeted to low power, low throughput and low cost devices and it is not backwards compatible with previous versions of the standard. It has been designed for short range wireless communications for BSNs and wearable technologies. As for previous versions, a BLE network has a star topology, with multiple slave devices connected to one master.
The BLE stack (Fig. 1) can be divided in three main layers: Controller, Host and Application. Controller and Host layers

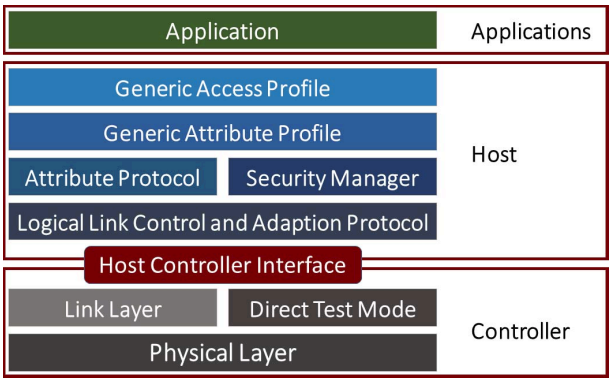

Fig. 1. The graphical representation of the BLE stack.

can reside in two separate chips or on the same chip and they communicate via the standard Host Controller Interface (HCI). Application and Host layers can also reside in separate chips but no standard exists for their interface.
In this work, we target application level optimization of BLE, hence we focus on its upper layers. In particular, the Logical Link Control and Adaptation Protocol (L2CAP) is responsible of protocol and channel multiplexing, segmentation and reassembly of packets for the lower levels, along with error and flow control. The Security Manager (SM) is responsible for device pairing and key distribution, while the Attribute Protocol (ATT) allows a device to expose a set of attributes (also called Services and Characteristics) and their associated values. The Generic Attribute Profile (GATT) defines a service framework for the interaction with attributes, and the Generic Access Profile (GAP) defines the procedures related to the discovery and link management of the device's connection. It also defines the role of the device, which can be Broadcaster, Observer, Peripheral, and Central. While the L2CAP is transparent to the user application, it interacts with the GAP and GATT for the data exchange and ATT and SM during connection initialization,

A. Data exchange in BLE

In a BLE connection there is a Central (or master) device and one ore more Peripheral (or slave) devices. The master initiates the connection with the Peripherals and once the desired network is established, the connected devices can expose their Services and Characteristics to the Central device (it can be also that the Central device exposes its attributes to Peripheral device). The device that exposes its attributes is called the GATT Server and the other one assumes the role of the GATT Client. The GATT role is independent form the master/slave GAP role.
Within an established network, data exchange can only be performed through the exposed Characteristics, which are data containers for 8 bit data values, arranged in arrays of up to 512 octets. Similar Characteristics can be grouped into Services and they all are stored in the attribute table in the form of a data structure. Both the GATT server and client have their own local copy of the table.
Each Characteristics has its own Properties that define how the GATT client can interact with it. The available Properties are:

- *Read*: the client can read the Characteristic value;
- *Write*: the client can write the Characteristic value;

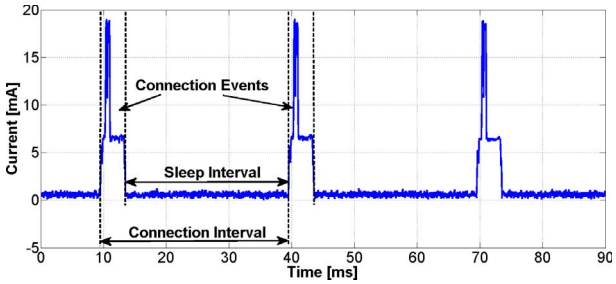

Fig. 2. Example of BLE duty cycling strategies: in this case the CI is set to 30 ms and only one PPCE is sent.

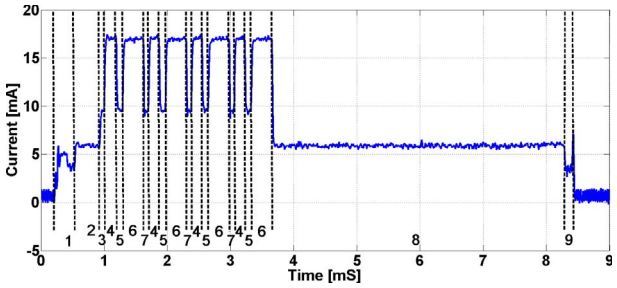

Fig. 3. Current absorption of CC2541 sending 4 packets in one CE: 1) startup, 2) pre-processing, 3) pre-RX, 4) RX, 5) RX-TX inter frame, 6) TX, 7) TX-RX inter frame, 8) post-processing, 9) pre-sleep. The phases 4 to 6 are repeated PPCE times, phase 7 is repeated PPCE-1 times, the rest is executed only once.

- *Notify*: the client can be notified when a Characteristic value has been updated by the server, without the need of a read operation;
- *Indicate*: as Notify, but with an application level acknowledgment of the notification to the server.

There are two ways for transferring data from the GATT server to the client: the first is through notifications or indications and the second is through explicit read requests. With notifications/indications, when the GATT server changes a Characteristic value in its local attribute table, the new value is automatically updated on the client's attribute table. With read requests, the update is done only when the GATT client requests the reading of that value. Transferring data from the client to the server is allowed only through write requests.
Data exchange between connected devices is always performed within Connection Events (CEs) and between two consecutive CEs the radio is in sleep mode to save energy. The time interval between two consecutive CEs is called Connection Interval (CI): the BLE standard permits CIs from 7.5 ms to 4 s (with steps of 1.25 ms). An example of typical current absorption during BLE operations is shown in Fig. 2. Another important parameter in a BLE connection is the Slave Latency (SL). This parameter represents the number of consecutive CEs that can be skipped by the Slave device if it has no data to send. In the ideal case, the best overall performance is obtained with SL set to zero [24]. The connection parameters and timings are negotiated at the connection set up and they can be changed any time during the connection. However, each device can accept or reject a timings change request based on its hardware and stack implementations.
Each CE contains at least one TX and one RX event, allowing to send/receive data Packets (i.e. one or more Characteristics values). Each Packet contains up to 20 bytes of application data and the standard defines procedures for segmenting and reassembling longer Characteristics, which will result in more than one TX/RX events to be sent. There is no imposed limit to the maximum number of Packets per CE (PPCE) that can be sent, but in practical applications limitations occur because of the timing requirements, the used stack implementation and the limited hardware resources of the devices.

B. BLE for streaming applications

BLE is a general purpose protocol suitable for a wide range of applications, but it is mainly optimized towards low power consumption rather than high data throughput. While the low power consumption is an universal specification for battery powered BSN devices, different applications and scenarios may lead to very different data throughput and latency specifications.
Typical use cases may range from the requirement to exchange a few bytes of data per second (i.e. event notification) to consistent data rates to be delivered in real time (i.e. high resolution ECG or inertial sensing). A 9-axis inertial measurement unit (IMU) sampled at 100 Hz requires 14.4 $kbit/s$, while biopotential applications such as ECG or EMG range from 8 $kbit/s$ for a 3-lead ECG signal to 64 $kbit/s$ for 12-lead ECG signals or 8 channels of EMG sampled at 500 Hz. Moreover, advanced sensor nodes can dynamically change their requirements, adapting data sampling, compression and transmission to the context.
Given this variable scenario, a desired data rate can be achieved with different combinations of CI and PPCE, which can lead to different results in terms of power consumption and data latency. Therefore, in this work we explore the most efficient use of the BLE protocol evaluating the highest application data rate in relation to the connection parameters and the number of connected devices. We want to help readers understand if BLE is suitable for a given application and when it is, how to use it in the most efficient way.

IV. Power Consumption Model

In order to evaluate the efficiency of a BLE module, we analyzed the power consumption profiles during data exchange and we extracted a model to calculate the current consumption given the CI and the number of PPCE.
To set up the model, we identified the different phases of the BLE operations as reported by the manufacturers [25, 26] and illustrated in Figures 2 and 3. Some of the sections are fixed regardless the number of PPCE or CI duration (phases 1, 2, 3, 9), others are repeated PPCE or PPCE-1 times (phases 4 to 7), while others increase their length proportionally to PPCE (phase 8). Our model has two parameters, the CI and the PPCE. We considered a constant current absorption during each phase and we expressed all the temporal repetitions and durations in function of the two parameters. The resulting current consumption is then computed as follows:

$$I_{CEmean} = \frac{I_F t_F + PPCE\ I_P t_P + (PPCE-1) I_{P_1} t_{P_1}}{t_{CE}} \quad (1)$$

$$I_{mean} = \frac{I_{CEmean} t_{CE} + I_{sleep} t_{sleep}}{t_{ConnInt}} \quad (2)$$

In this model, I_F is the mean current and t_F is the duration of the fixed phases that do not depend on $PPCE$ (phases $1, 2, 3, 9$); I_P and t_P have the same meanings but for phases with length proportional to $PPCE$ ($4, 5, 6$ and 8); in the same way I_{P_1} and t_{P_1} are for phase 7 iterated $(PPCE - 1)$ times;

I_{sleep} is the sleep current, t_{sleep} is the Sleep Interval and t_{CE} is the total duration of the Connection Event (from phase 1 to the end of phase 9 in Fig. 3).
For each analyzed chip, each phase was characterized measuring its mean duration and absorbed current over several intervals, allowing to analytically evaluate the current consumption for different combinations of connection parameters. To validate this approach, we performed extensive current measurements in different configurations and observed an error between the computed and measured currents below 5%.

V. Implementation

To evaluate the performance of selected BLE chips, we implemented a firmware to use them as nodes of a BSN. We employed modules from three different vendors and, as much as possible, we implemented the same functionalities on all of them. In particular, to evaluate data throughput and consumption, we programmed the modules to be generators of dummy packets to be delivered to the central unit.
Before a module can transmit data over the BLE connection, it needs to configure hardware (timer, interrupts) and software (Bluetooth stack) components. Once the basic configuration is performed, the needed GATT services are added to the GATT server (which resides on the BLE module) and the BLE is set to be visible and connectible. Now the BSN node is ready and it waits to get connected by the smartphone.
To implement the data streaming between the nodes and the smartphone, we created a service composed by two characteristics: *DATA_CHARACTERSITIC* (20 Bytes with notify and read properties) and *ENABLE_CHARACTERISTIC* (1 Byte with write and read properties). The first one is for the data exchange: every time the sensor node has new data, it is packaged in 20 Bytes and it is written in the *DATA_CHARACTERSITIC*. The transfer is done with the notification mechanism, so the master device will automatically receive the new data on the next CE. The *ENABLE_CHARACTERISTIC* is used to control the data stream from the master side, enabling it when set to 1 and disabling it when 0.
The high efficiency of the BLE standard is largely due to the duty cycling between active and sleep states (see Fig. 2). The BLE radio of a sensor node is usually in sleep and wakes up at pre-defined intervals (CIs) to exchange data with the connected device. To maximize the benefits of the duty cycling policy, our BLE sensor nodes synchronize all their application tasks with the CEs and thus perform all the needed operations right before or after a CE and go to sleep in between.
The CI time is used to define the data throughput/power consumption trade-off. Given a needed application throughput, there is a degree of freedom in setting different combinations of CI and PPCE to achieve it. Since each configuration leads to different performance, this parameters are investigated to find the most convenient settings for the application requirements.

VI. Experimental Results

A. Experimental Setup

The test system used in this paper is composed by three different BLE modules, which have been selected for they availability and large diffusion. The modules are: CC2541 from Texas Instruments (TI), nRF51822 from Nordic Semiconductor and BlueNRG from ST Microelectronics.
Each of them is used on a development board and with the provided proprietary BLE stack (BLE-STACK 1.4.0 for CC2541, SoftDevice S110 v8.0 for nRF51822 and BlueNRG FW v6.4 for BlueNRG). The TI and Nordic modules do not need an external microcontroller to be used and they allow to add some user application code to the microcontroller embedded in the chip (an 8051 for TI and an ARM Cortex M0 for Nordic). The BlueNRG module is different since its embedded Cortex M0 microcontroller is reserved for the BLE stack and it is not accessible to the user. In this case it is mandatory to use an external microcontroller to run the desired application and for our tests we used a development board equipped with a STM32L chip.
The BLE current consumptions and timings used as power model parameters have been measured from the boards using a low-side shunt resistor where possible and high-side shunt resistor with an amplification circuit otherwise.
The master device is a smartphone (Motorola XT1039) running Andorid 4.4.4 with a custom application that connects to the desired BLE devices, receives the data stream and logs it on a file. We used the device as is, without hardware modification and we developed the application using only standard Android APIs to evaluate the use of the smartphone as a master node for reliable BSN applications with streaming nodes. In the BlueNRG module it is not possible to synchronize the application task with the CEs, since they are not notified to higher stack levels. Given this configuration, for a close comparison of the radio chips, for the BlueNRG we report only the measured current consumption relative to the BLE module.

B. Results

Given the BLE optimization towards low data rates, we first evaluated the maximum application-level data throughput that can be achieved. The standard imposes the minimum CI to be $7.5\ ms$, hence the maximum application throughput can be achieved sending as much as possible PPCEs with the lowest CI. In a scenario where one node streams data to the smartphone, we found that, regardless of the BLE module used, the PPCE is limited by the smartphone to 3 and imposes a maximum throughput of $64\ kbit/s$.
With longer CIs, a greater number of PPCEs can be sent and if the application can tolerate some degree of data buffering, the same throughput of $64\ kbit/s$ can be again achieved. To evaluate the performance of the different devices, we thus analyzed all the configurations allowing for the maximum data throughput and compared their current consumption. The result of this analysis is shown in Fig. 4, where we plotted the mean current and the energy per bit for each module with different CI and PPCE settings used to achieve $64\ kbit/s$. We can observe how the TI module is able to send up to 15 PPCEs, while the BlueNRG and the Nordic stop at 7 and 6 respectively. The BLE standard does not provide specifications on this, hence these differences are due to the different stack implementations.
In the second experiment we analyzed application scenarios less demanding in terms of data throughput, hence we used the proposed power model to compute the node's current consumption varying the throughput from the maximum value down to zero. Since the maximum efficiency is achieved sending the maximum value of PPCEs allowed, we set the PPCE to that value for each device and decreased the CI from $7.5\ ms$ to $4\ s$. The mean current consumption for

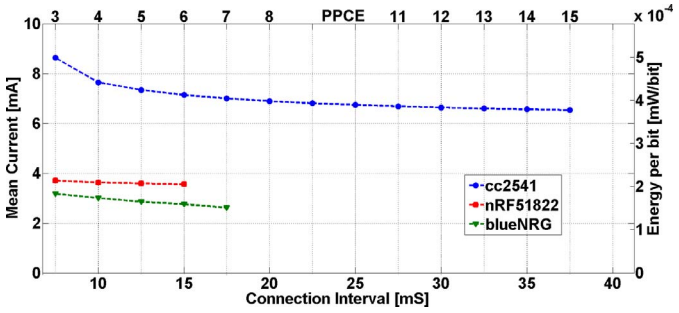

Fig. 4. Mean current and Energy per bit with fixed data throughput (64kbit/s). For each CI the relative PPCE values are reported on the top horizontal axis.

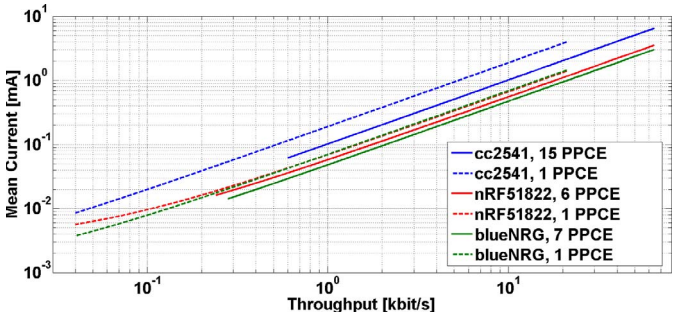

Fig. 5. Current consumption as function of Throughput. This is computed using maximum number of PPCE allowed on each module and for minimum number of PPCE (i.e. one PPCE).

each device is shown in Fig. 5. As expected, the current consumption decreases dramatically with the decrease of the data throughput.

The last experiment was targeted to evaluate applications needing more than one sensor device connected to the smartphone. This multislave configuration has been tested with up to 6 slave devices and each node has been set up with the same configuration (same CI and PPCE). For each number of nodes, we wanted to find the maximum achievable throughput. For this purpose, for each configuration, we recorded 5 minutes of streaming data for 5 times and the result of this test is summarized in Table I. The throughput is calculated knowing the number of PPCE, the payload (20 bytes) and the CI; F_s is the corresponding sensor sampling frequency if one sample is sent each CE. The data logged during each test has been analyzed to verify its integrity and that the streaming was consistent with the settings. When the throughput was more than 3 % lower than the expected one or the lost packets were more than 3 %, the connection has been considered not reliable. For each number of connected nodes, we lowered the CI until we found a reliable connection. Moreover, in this multislave case we could establish a reliable data stream only with 1 PPCE, even with longer CIs up to 20 ms

VII. Discussion

The reported results highlight that in a single streaming device scenario an optimized choice of the connection parameters improves the system efficiency while maintaining the desired throughput. Buffering data to transmit it less frequently can give up to 47 % of energy saving. This saving is affected by the maximum value of PPCE the module supports, the throughput needed and by the overall current profile of the selected device. In some situations the proposed optimizations have a limited effect due to the hardware and stack implementations which limit the usable configurations (see the red trace in Fig. 4).

As a guideline for optimal BLE connection parameters, one may consider the equation to compute the application throughput in the ideal case (no packet loss):

$$TH = \frac{PL \times PPCE}{CI} \tag{3}$$

where, PL is the application level payload, whose maximum allowed value is 20 bytes, CI is the Connection Interval and TH is the calculated throughput. This relation can be rewritten as:

$$CI = \frac{PL \times PPCE}{TH}. \tag{4}$$

Knowing the throughput needed for the application and the maximum value of PPCE allowed (this is affected by the chosen module, its BLE stack and also by the smartphone and its operating system), it is possible to calculate the CI that gives the best energy performance.

As example, if the application uses data from a 9 axis, 16 bit motion sensor sampled at 100 Hz, each BLE packet can contain an entire sample of the sensor. In fact: 16 bits each of the 9 axis gives 144 bits or 18 bytes each sample, two remaining bytes can be used as counter. The needed throughput is 160 $bits \times 100\ Hz = 16\ kbit/s$. Suppose to use cc2541 module, taking some margins a possible choice for parameters is 10 PPCE and, appling eq.4, 100 ms for CI. If a latency of 100 ms is too much PPCE and consequently CI can be reduced.

It has to be remarked that the proposed optimization is an energy/latency trade-off, since the data needs to be buffered between CEs and therefore a delay is introduced. The introduced latency must be evaluated for each application considering how much data has to be sent, the tolerated error rate, CI and SL parameters. In an ideal case SL should be zero to avoid energy waste on the master side, but if there are packet transmission errors, using a non zero SL and a shorter CI will improve the overall performance.

In the multislave configuration, we tried to send as many PPCEs as possible, with relatively low values of CI and as many nodes connected as possible. Since the master device can serve only one connection at the time (time multiplexing is used for connecting multiple slaves), the number of connected nodes is dependent on the CI duration and the consequent ability to serve all the desired devices within one interval. Concurrently analyzing the operations and current consumption of a connected network of devices, we found that the distribution of CE slots and their order in one CI are randomly set by the master, they are managed by the Android stack and are not controllable by the user. As general result, when more slaves are connected to the smartphone, we obtain that a slave can

TABLE I. Result of multislave connection test

n	CI $[ms]$	PPCE	TH $[kbit/s]$	Equivalent freq.$[Hz]$
2	7.5	1	21.3	133
3	7.5	1	16	133
4	10	1	12.8	100
5	13.75	1	10.7	73
6	16.25	1	8	62

reliably send only one PPCE. With longer CIs (i.e. $100\ ms$) these restrictions expire and more PPCE can be reliably sent. We have also documented some situations where two nodes are overlapped each other sharing the same CE window. In this case, the connection remains active but data is transferred only from one of the two nodes at the time and they switch their role about every second. Overall optimizations discussed above are valid in one slave configuration or multislave configuration with long CIs (i.e. longer than $100\ ms$). With multiple slaves and CI in the order of $20\ ms$ (and shorter) only one PPCE can be sent, therefore the only optimization is to properly set the CI using Eq.4. For a comparison, the current Apple stack implementation for iOS limits the minimum CI to $20\ ms$ with one PPCE.

In all cases the sensors reading and/or data elaboration in the node should be performed in synchrony with CEs to allow an effective duty cycling and prolonged sleep states. Unfortunately, this is not always possible, since the BLE standard does not define an unified notification of the CE between stack layers. Some of the stack implementations add this feature, which results very useful but at this time it can not be considered part of the standard.

VIII. Conclusion

In this work, we analyzed the BLE standard and its use for BSN applications, with particular interest for healthcare for devices with high data throughput and reliability requirements. We introduced the protocol and its features from an application-level perspective, evaluating its characteristics and how they are implemented in available devices and stacks. We employed a smartphone as the network master device, taking the advantage of the availability of such devices and their ability to handle the needed communication and processing requirements without the need for additional hardware.

An evaluation of three widely adopted BLE modules was performed and maximum application level throughput of $64\ kbit/s$ has been obtained. This limit has been demonstrated to be imposed by the smartphone and its software instead of BSN node. The effect of connection parameters optimization has been evaluated for the BLE chips and their stack implementations. The multislave connection has also been tested, but since in this configuration there are many variables that play a role in network performances and lot of them are out of user control, evaluation has been done in the most critical conditions and with a maximum of 6 active connections.

Acknowledgment

This work has been supported by CoRehab S.r.l. and by the EU EIT project "A generic Platform for Movement Training".

References

[1] Y.-L. Zheng, X.-R. Ding, C. Poon, B. Lo, H. Zhang, X.-L. Zhou, G.-Z. Yang, N. Zhao, and Y.-T. Zhang, "Unobtrusive sensing and wearable devices for health informatics," *IEEE Trans. on Biomedical Engineering*, vol. 61, no. 5, pp. 1538–1554, May 2014.

[2] B. Calhoun, J. Lach, J. Stankovic, D. Wentzloff, K. Whitehouse, A. Barth, J. Brown, Q. Li, S. Oh, N. Roberts, and Y. Zhang, "Body sensor networks: A holistic approach from silicon to users," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 91–106, Jan 2012.

[3] O. Amft and P. Lukowicz, "From backpacks to smartphones: Past, present, and future of wearable computers," *IEEE Pervasive Computing*, vol. 8, no. 3, pp. 8–13, 2009.

[4] R. Cavallari, F. Martelli, R. Rosini, C. Buratti, and R. Verdone, "A Survey on Wireless Body Area Networks: Technologies and Design Challenges," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1635–1657, 2014.

[5] B. Milosevic, C. Caione, E. Farella, B. D., and L. Benini, "Sub-sampling frameworks comparison for low-power data gathering: a comparative analysis," *Sensors*, vol. 15, no. 3, pp. 5058–5080, 2015.

[6] M. Benocci, E. Farella, L. Benini, and L. Vanzago, "Optimizing ZigBee for data streaming in body-area bio-feedback applications," in *Intl. Workshop on Advances in Sensors and Interfaces (IWASI)*, 2009, pp. 150–155.

[7] M. Magno, L. Benini, L. Gaggero, J. La Torre Aro, and E. Popovici, "A versatile biomedical wireless sensor node with novel drysurface sensors and energy efficient power management," in *Intl. Workshop on Advances in Sensors and Interfaces (IWASI)*, 2013, pp. 217–222.

[8] A. Dementyev, S. Hodges, S. Taylor, and J. Smith, "Power consumption analysis of Bluetooth Low Energy, ZigBee and ANT sensor nodes in a cyclic sleep scenario," in *IEEE Int. Wireless Symposium (IWS)*, April 2013, pp. 1–4.

[9] C. Buratti, R. D'Errico, M. Maman, F. Martelli, R. Rosini, and R. Verdone, "Design of a Body Area Network for Medical Applications: The WiserBAN Project," in *Intl. Symp. on Applied Sciences in Biomedical and Comm. Technologies*, ser. ISABEL '11, 2011, pp. 164:1–164:5.

[10] M. Altini, S. Polito, J. Penders, H. Kim, N. Van Helleputte, S. Kim, and F. Yazicioglu, "An ECG Patch Combining a Customized Ultra-low-power ECG SoC with Bluetooth Low Energy for Long Term Ambulatory Monitoring," in *Conf. on Wireless Health*, 2011.

[11] E. Mackensen, M. Lai, and T. Wendt, "Bluetooth Low Energy (BLE) based wireless sensors," in *IEEE Sensors*, Oct 2012, pp. 1–4.

[12] M. Zhang, W. Xia, and L. Shen, "Bluetooth low energy based motion sensing system," in *Int. Conf. on Wireless Communications and Signal Processing (WCSP)*, Oct 2014, pp. 1–5.

[13] T. K. Lai, A. Wang, C.-M. Chang, H.-M. Tseng, K. Huang, J.-P. Li, W.-C. Shih, and P. Chou, "An 8x8 mm2 bluetooth low energy wireless motion-sensing platform," in *Int. Symp. on Information Processing in Sensor Networks*, April 2014, pp. 341–342.

[14] B. Zhou, X. Chen, X. Hu, R. Ren, X. Tan, Z. Fang, and S. Xia, "A bluetooth low energy approach for monitoring electrocardiography and respiration," in *Intl. Conf. on e-Health Networking, Applications Services (Healthcom)*, Oct 2013, pp. 130–134.

[15] A. Jara, D. Fern'ndez, P. Lopez, M. Zamora, B. Ubeda, and A. Skarmeta, "Evaluation of bluetooth low energy capabilities for continuous data transmission from a wearable electrocardiogram," in *Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, July 2012, pp. 912–917.

[16] H. Strey, P. Richman, R. Rozensky, S. Smith, and L. Endee, "Bluetooth low energy technologies for applications in health care: proximity and physiological signals monitors," in *Int. Conf. on Emerging Technologies for a Smarter World (CEWIT)*, Oct 2013, pp. 1–4.

[17] D. Craven, M. Glavin, L. Kilmartin, and E. Jones, "Potential for extended battery life in mobile healthcare with bluetooth low energy and signal compression," in *Signals and Systems Conference (ISSC 2012), IET Irish*, June 2012, pp. 1–6.

[18] Z.-M. Lin, C.-H. Chang, N.-K. Chou, and Y.-H. Lin, "Bluetooth Low Energy based blood pressure monitoring system," in *Intelligent Green Building and Smart Grid (IGBSG)*, April 2014, pp. 1–4.

[19] C. Gomez, I. Demirkol, and J. Paradells, "Modeling the maximum throughput of bluetooth low energy in an error-prone link," *IEEE Communications Letters*, vol. 15, no. 11, pp. 1187–1189, 2011.

[20] K. Mikhaylov, N. Plevritakis, and J. Tervonen, "Performance Analysis and Comparison of Bluetooth Low Energy with IEEE 802.15.4 and SimpliciTI," *Journal of Sensor and Actuator Networks*, vol. 2, no. 3, pp. 589–613, 2013.

[21] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11 734–11 753, 2012.

[22] M. Siekkinen, M. Hiienkari, J. Nurminen, and J. Nieminen, "How low energy is bluetooth low energy? comparative measurements with ZigBee/802.15.4," in *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, April 2012, pp. 232–237.

[23] Bluetooth SIG, "Specification of the bluetooth system v4.0," 2010.

[24] R. D. S. C. Philipp Kindt, Daniel Yunge, "Precise energy modeling for the bluetooth low energy protocol," in *CoRR*, 2014.

[25] S. Kamath, "Measuring Bluetooth Low Energy Power Consumption," in *Application Note AN092 Texas Instruments*, 2010.

[26] Nordic Semiconductor, "S110 nRF51, SoftDevice Specification v2.0," 2014.