ParkingBLE: A Wireless Solution to Parking Detection

Souradip Ghosh Kyle Qian
Department of Computer Science
Northwestern University
{sgh, kyleqian2021} @u.northwestern.edu

ABSTRACT

As congestion in cities grows, street parking for residents and visitors is becoming increasingly difficult to find, especially in the Chicago area [1]. At the same time, there have been advances in the Internet-of-Things (IoT), especially in efficient wireless networks, such as Bluetooth Low Energy (BLE), and inexpensive sensor technology capable of large-scale deployment for a wide variety of applications, including real-time parking detection. In this work, we present *ParkingBLE*, a sensor network system that addresses the growing street parking issues by providing the general public accessible information on street parking spots, derived from on-the-ground sensor data. ParkingBLE is built on top of BLE technology, and tiny, low-power, embedded devices including the Nordic nRF52840 SoC device and the TDK ICM-20948 MotionTracking device. We show that a prototype of our system is efficient, extensible, and user-friendly.

CCS CONCEPTS

• **Software and its engineering** → **Wireless and sensor networks, Embedded systems**;

KEYWORDS

Internet of Things, Networks, Bluetooth Low Energy

1 INTRODUCTION

As residents of Evanston, IL we have found ourselves regularly spending exorbitant amounts of time looking for empty parking spaces. It is a seemingly simple task, but the punishment for making the mistake of parking in a no-park zone or staying longer than an arbitrarily allotted time even once is usually a ticket that is the price of double or even triple a college student's weekly bills. Evanston itself is a fairly suburban city, but looking outwards to more urban areas like Chicago this problem only increases in inconvenience. In the past several decades, the explosion of Internet-of-Things and an emergence of cheap sensor technology has given average people greater access to resources previously out of reach. What would normally be an expensive, energy draining system can now be compressed into an embedded platform that cost less than the average parking ticket in Evanston.

It is in this environment that we saw an opportunity to apply this emerging technology to solve the growing problem of parking. ParkingBLE is intended to be an application that utilizes a network of embedded systems and sensors, communicating wirelessly over Bluetooth Low Energy, to relay parking information. Its design is intentionally cheap and energy efficient, and its interface is meant to give users instant information on what available parking spaces there are in a given area.

2 BACKGROUND

2.1 Bluetooth Low Energy (BLE)

Bluetooth low energy (BLE) was developed as a method of wireless communication similar to bluetooth classic but with a greater focus on low-energy, low-throughput interactions. At its core, it provides direct device-to-device communication with the intent of maximizing the amount of time each device can stay in sleep mode [1]. Each device acts as a server with accessible fields, and pairs of devices tend to act with central/peripheral relationships.

BLE has two main mechanisms: advertisements and connections. BLE advertisements are generally used for discovery of devices in an area around a scanner. Advertisements act as broadcast messages indicating device details, and communication from advertiser to scanner(s) is ephemeral, unidirectional and utilizes the ALOHA access control scheme. BLE connections are a bi-directional form of communication between peripheral and central nodes, and can occur after a device has been detected by advertising/scanning [1]. Connections are maintained for much longer durations than advertisements, and utilizes TDMA access control for a more stable mode of communication.

With both BLE advertisements and connections, single devices can actually act as both parts of the paradigm at the same time. Devices can be scanning and advertising, or acting as both peripheral and central nodes for connections simultaneously.

We ended up choosing BLE advertisements because of its ease of usage, and its more energy saving features. Maintaining a BLE connection requires a lot more energy between two devices, and because our peripheral devices are connected to sensors that can provide data updates in discrete increments, it made much more sense for us to focus on BLE advertisements. BLE advertisements give us the ability to send bursts of information, while powering down and sleeping between time periods of sensing.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.2 Related Systems and Protocols

Thread is another wireless protocol, it is built on top of 15.4 and is similar to BLE but adds things like IP communication, and addressing and mesh maintenance. It is meant to be easy to install and operate, very energy efficient, scalable to hundreds of devices, and allows for mesh networking without single points of failure. It communicates through the unslotted CSMA/CA access control, and can be created with a combination of star of mesh topologies, where end devices are each connected to one outer, which communicates with other routers to forward packets. The issue with thread, and the ultimate reason we did not use it is because of its unnecessary complications. The data we are sending is so simple that IPv6 seems like overkill, as it is not as if we are sending this data over WiFi, as the central node in our system will be permanently connected to a LAN configuration. Not only this, but thread would require much more setup compared to the ease of use of BLE advertisements, and the energy usage by BLE advertisements still ends up being more attractive than the energy usage of 15.4 based protocols [2].

LPWANs such as LoRaWAN or Sigfox were other wireless protocols we considered. The purpose of LPWANs is to provide communication at the region/city scale, with emphasis on a wide area of coverage, low power, and low throughput. They are quite common options for doing smart-city technologies, as well as smart-metering, and smart-homes, as LPWAN options give huge ranges of coverage, up to 40 Km for rural areas and 10 Km for urban areas and excel at sending tiny messages [3]. This option may have been viable for this system, but due to its complex setup, and our budget for supplies and space for this project, we decided to stick with BLE advertisements, but future developments especially at an expanded range could definitely see use from these types of wireless protocols.

3 DESIGN

In the ParkingBLE system, we have designed the networking functionality of the sensor network around a simple protocol built around BLE. In particular, we chose to employ BLE advertisements over BLE connections and other protocol designs to obtain the best combination of flexibility, energy efficiency, and usability (on behalf of both the designers and programmers). Moreover, advertisments are ideal for sending small, controlled packets of information, which is also a key component that lets the ParkingBLE system stay simple and extensible.

3.1 Minimum Viable Product

ParkingBLE's design at its simplest, i.e. the *minimum viable product* (Figure 1), is a combination of a *sensor device* and a *central device*. Sensor devices are deployed at parking spots for a targeted street, and are responsible for determining the status of respective parking spots (i.e. occupied or available), typically at a specified time interval. Sensor devices must also communicate with devices in the system that are responsible for gathering parking status information, and are typically battery powered. At a high level, the central device is designed to perform this gathering and organization of parking status information, make system-wide decisions, and act as the intermediary between the on-the-ground system and the user

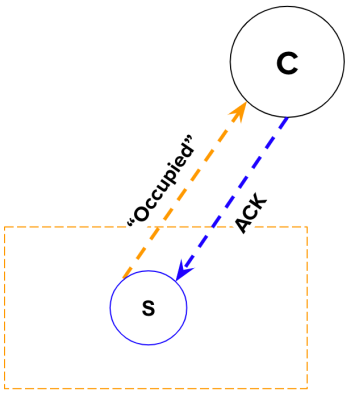

Figure 1: Minimum viable product design for ParkingBLE.

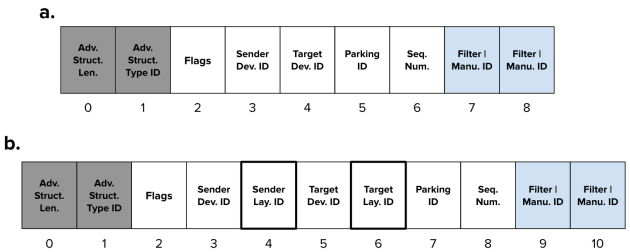

Figure 2: (a) The BLE advertisement payload layout for the minimum viable product design. (b) Layout when integrating layers into ParkingBLE, bolded fields are extensions for layers.

interface. The central device, unlike sensor devices will be assumed to have a constant power supply at all times. In the minimum viable product design, the sensor device will therefore communicate directly with the central device.

This communication is accomplished through BLE advertisements. Specifically, the sensor device will wrap parking status information into an advertisement payload targeted for the central device. The advertisement payload (Figure 2a) of the mimimum viable product design includes information categorized into the "sender"'s signature and the "target"'s signature, where each device also has a unique device ID. In the ParkingBLE protocol, these features allow typical target devices, such as the central device, to scan for payloads designated for itself, while typical sender or end devices such as sensors, will sign the appropriate fields in the advertisement payload, and set a bit in the flags field with its parking status. These attributes are necessary for streamlined communication between devices and simplifies the gather of parking status information from all sensor devices.

To build stronger guarantees of information flow in the ParkingBLE system, the minimum viable prouduct asserts that all advertisements within the system be acknowledged by its targeted recipient. Acknowledgements (acks) follow specific invariants; the sender and target signatures are flipped in ack payloads, and a specified bit in the flags field is set to designate the packet as an ack.

Acks are also "pseudo-atomic," in the sense that a recipeint receiving a new advertisement payload must process its information and *immediately* send an ack designated for the sender. These are motivated by design choices that will be described in Section 3.3. In the minimum viable product, this extra communication will occur between the sensor and the central device. Consequently, the functionality of both devices must be augmented to handle not only the parking status information but also acks. When a sensor device sends an advertisement with parking status information, it simultaneously begins to scan for acknowledgements. Moreover, the sensor device does not stop advertising until it receives an ack, at which time it stops both advertising and scanning immediately and waits for the next interval to send data. The central device is constantly scanning for ParkingBLE advertisement payloads. When receiving an advertisement from the sensor device, it formulates an ack and begins to advertise the ack indefinitely until it needs to send an ack to another device. Note that the central device also performs scanning and advertising simultaneously.

Of course, the minimum viable product design must scale to many devices across a large region, especially when considering modern street parking designs. This design must also be energy efficient and flexible such that devices running on battery power will not need frequent attention and replacement. We explore some of these extensions in the following subsections.

3.2 Extensions: Scalability

Although BLE advertisement and connection ranges have massively improved over several iterations of the protocol, its maximum range can only stretch up to tens of meters [4]. Additionally, given that a ParkingBLE deployment can also need to handle tens, if not hundreds of parking spots, one central device is not enough to process all parking status information. To address these scalability concerns, we expand upon the minimum viable product design by adding hierarchy to the system. In particular, ParkingBLE supports what we call *layers*, a tree-like topology that can route information and extend the range of the ParkingBLE system (Figure 3). A layer consists of a set of *relayer* devices, which have the sole purpose of "relaying" or routing data from the sensor devices to the central device. A ParkingBLE system can support an arbitrary number of layers, n, where the 0th layer consists of only the central device, and the nth layer consists only all sensor devices. For example, the minimum viable product would only have two layers. This idea of "ordering" amongst layers creates a tree-like hierarchy, where information is routed from the bottom layer (nth) to the top layer (0th). In contrast to traditional tree toplogies, the hierarchy in ParkingBLE does not consist of strict parent-child relations between nodes. Instead, information is routed at a "layer granularity," where devices in a lower layer must simply route information to the next, upper layer.

With respect to functionality, the advertisement payload and acks must change to complement a layered hierarchy and support sender's and target's layer IDs. Additionally, the idea of target information when sending data *up* the layers can be loosened because the target will always be the central device. With these design changes

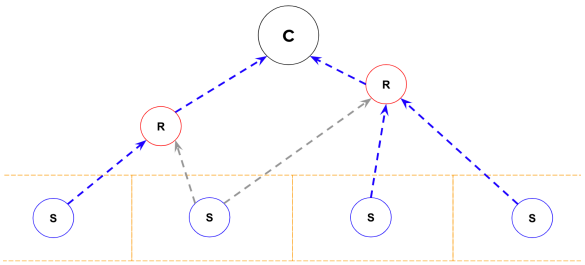

Figure 3: An example deployment of ParkingBLE built with three layers, the top layer consisting of the central device, the middle layer consisting of two relayer devices, and the bottom layer consisting of four sensor devices. Note that arrows indicate information flow, and the gray arrows demonstrate the lack of strict parent-child relations and propogation at "layer granularity".

in mind, we designed relayers to always scan for ParkingBLE advertisement payloads and handle payloads *only* if they are arriving from the relayer's adajacent lower layer. Once such a payload is received, the relayer must ack immediately, as in the minimum viable product. A key difference in a layer-based design is that data needs to be sent upwards following the ack. Here, we apply an optimization to prevent relayers from receiving, acknowledging, then advertising the data upwards, and waiting for that repsective ack. Our advertisement payload allows the ack to the lower layer to serve a dual purpose. Because the ack will be processed only by checking the ack flag and target device and layer IDs, the rest of the fields can be repurposed to serve as the relayed payload to the adjacent upper layer. The sender device and layer ID can be set to the relayer's information, which will be picked up and processed by an upper-layer device. Once the upper layer device sends an ack to the relayer, the relayer stops advertising the payload, effectively ending communication with both adjacent layers. To prevent lost acks to lower layers, relayer devices reference count the number of acks from the upper layer to a predefined number, so that there is enough time to communicate the necessary data to both adjacent layers. In this work, our reference count limit for relayers is 4.

3.3 Extensions: Efficiency

Since the ParkingBLE system consists of small, embedded devices, some of which are designed to run on battery power, we also augment the system with a couple of features that promotes energy efficiency. In order to choose the best efficient features, it is important to note that the most important set of devices is the sensor device layer. These devices not only run solely on battery power, but are also the most populous layer and the most difficult to maintain; malfunctions would be tricky to replace at the right time, since this would require a parking spot to be out of commision for a period of time. With this in mind, our work focuses on energy efficiency of sensor devices in particular.

Sensor devices are optimized to send data only if there has been a change in the parking status. Since parking spots are often occupied for extended periods of time, this optimization prevents frequent advertisements, which are the most expensive operation that sensor devices perform. Less frequent advertising also reduces

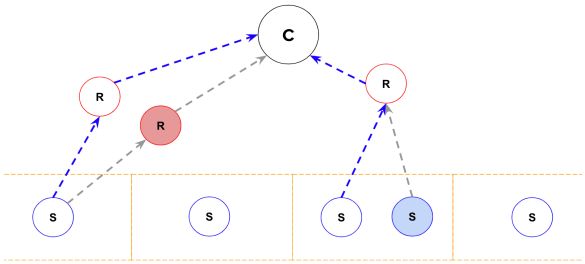

Figure 4: An example of some of the flexibility and efficiency features of the ParkingBLE system. The highlighted relayer device (in red) can be replaced with a new relayer without interruption, and the highlighted sensor node (in blue) can act as a redundant node for its respective parking spot.

congestion across the entire system and prevents longer wait times for acknowledgements. It is also important to note that relayer devices, which are also assumed to be battery powered, are fewer in number, do not have sensors attached, and does not require any contact with street parking. Consequently, these are simpler to update or replace. We do not apply any optimizations for relayer devices.

Another feature we introduce is a mechanism for the central device to make a system-wide decision to throttle down sensor devices' interval frequency. For example, at night time or on weekends, it is unlikely that parking will be in high demand or changing frequently. As a result, it is more optimal for sensor devices to synthesize parking statuses at a longer interval (e.g. every 30 minutes instead of every 5 minutes). This boosts efficiency for the same reasons already described. The decision is "slowly" propagated as a set bit in the flags field of the advertisement payload. The central device will set this flag in its acks, and all other devices will dynamically take the flag value from acks coming from its adjacent upper layer. Over time, this will ensure that sensor devices' interval frequency changes.

3.4 Extensions: Flexibilty

As briefly described in Section 3.2, one of the concerns for the ParkingBLE system is malfunctioning or dead devices. Consequently, flexibility, the ability to add and replace devices, is an important component of this system. Luckily, the layer-based design allows for seamless replacement since devices communicate at a "layer granularity" instead of predetermined devices. For example, a relayer device replacement can be deployed first in the same area as a malfunctioning relayer, which can be subsequently taken out.

ParkingBLE also supports redundancy, where there can be up to two sensor devices per parking spot. Redundancy can be important when replacement is very difficult, sensor data may be flaky, or when parking spots are large (i.e. rest areas, etc.).

4 MECHANICS AND IMPLEMENTATION

We implemented the ParkingBLE system using the Nordic nRF52840 SoC device, which runs an ARM Cortex M4 CPU and supports Bluetooth 5 and Bluetooth Low Energy. Additionally, we used the TDK

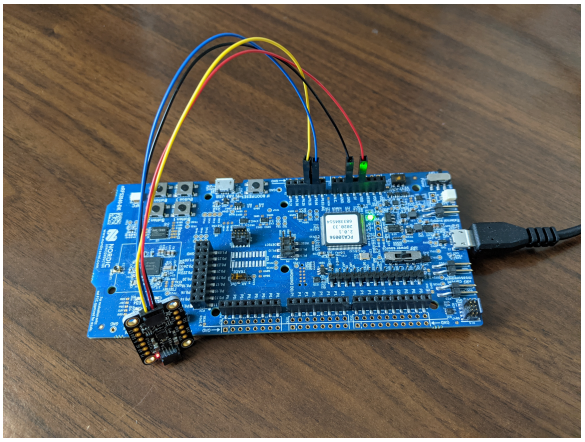

Figure 5: An example of a prototyped sensor device. An ICM-20948 is connected to the nRF52840.

ICM-20948 MotionTracking device, a low-power sensor equipped with a magnetometer, accelerometer, and other capabilities. This device also supports an I2C interface, which allows for seamless integration with the nRF52840 device. Most code written for our prototype utilizes the nRF52840 SDK. All device information, which includes device and layer IDs, sensor configurations, and BLE setup is built in at compile-time. In order to set new device information, the device must be taken out of commission and new code must be compiled and flashed onto the device.

This section describes implementation details for each components of the ParkingBLE system. However, it is up to a deployment team who is using ParkingBLE to determine appropriate locations to place sensor and relayer devices such that the system runs effectively.

4.1 Sensor Devices

Sensor devices are built on top of the nRF52840 device with the ICM-20948 attached directly to the board. They query the ICM-20948 at a time interval set at compile. The query is triggered by a timer interrupt, whose handler queries the ICM-20948's state via a device register. The sensor device then synthesizes the parking status by determining if the data has passed a set threshold. In this prototype, we support querying both magnetometer data (a large reading signifies an "occupied" status) and accelerometer data (a large y-axis reading also signifies an "occupied" status). Note that even though these devices represent the edge of the system, we elected to perform the computation on these devices since the computation itself is quite small and advertising the result (1 bit) instead of the raw data (10 or more bits) is more efficient. Once the sensor devices has a new status to advertise up the layers, it begins to simultaneously scan and advertise. The advertisement interval in our implementation is 500 ms, and stops scanning and advertising immediately upon receiving the first ack, regardless of how many subsequent acks it could receive later. There are no strict timeouts supported by sensor devices, so if no ack is received it will continue to advertise until the next interval.

If the central device decides to propagate a "throttle down" decision (Section 3.3), the sensor device changes its query time interval once it receives the decision by stopping and restarting the timer with the new interrupt frequency. In our prototype, the sensor device queries the ICM-20948 every 5 minutes, and queries every 30 minutes when "throttled down."

Redudancy (Section 3.4) is supported on sensor devices by incrementing the sequence number field of the advertisement payloads by 2. Doing so allows another device to occupy the same parking spot, where the starting sequence number across the two devices can simply be staggered.

4.2 Relayer Devices

Relayer devices are also built using the nRF52840 device. In contrast to the sensor device, relayers are constantly scanning for advertisement payloads from their respective, adjacent lower layers by checking payloads' manufacturer ID and layer ID (See Figure 1). Upon the receiving an advertisement payload, the relayer device begins to relay the data via the mechanism described in Section 3.1. However, during this advertisement period, the relayer device does not accept new advertisement payloads from its adjacent lower layer. Only after the relayer has received the proper number of acks (determined by its reference count limit), does it stop advertising and start accepting new advertisements. As with the sensor devices, the advertising interval is 500 ms. As with sensor devices, relayers also do not support timeouts for acks.

We also implemented a key extension to the relayer device that allows it to ignore duplicate advertisements. Duplicates can propagate in the ParkingBLE system if a sensor device sends trailing advertisements (for a particular parking spot and sequence number) right as the relayer sends an ack. To handle duplicates, each new advertisement payload is cached, and all advertisements are queried in this cache before being handled. If there is a cache hit, the advertisement need not be handled again, and the relayer device ignores it. We implement this cache as a ring buffer containing a pair of information (the parking spot ID and sequence number).

We do acknowledge that it is possible for duplicate information to propagate at higher layers, where two relayers propagate the same information to another relayer higher up. This scenario can occur if relayers are placed too close to each other, and can cause relayers to wait for acks indefinitely if a device in the adjacent upper layer has already received the same data. Consequently, it is important that deployment of the ParkingBLE system is carried out well.

4.3 Central Device

In our implementation, the central device is built exactly as a relayer device with extra functionality. In particular, the central device simply has to handle and propagate system-wide decisions. In our prototype, the central device only supports the throttle down decision (see Section 3.3), which we implemented with a button interrupt that sets the appropriate flag in the device's ack payload (the flags field). Additionally, since the central device is also the highest-ranking device in the system, it only stops advertising acks once there is another ack to deliver.

4.4 User Interface

We implement the user interface as a web server running Node.js 10.16.0, which reads data outputted by the central device in real time. The central device is connected to a Real Time Transfer (RTT) terminal, which is a simple `telnet` client supported by the nRF52840 device. The web server formats this data and presents it on a user-friendly web interface (Figure X).

5 EVALUATION

5.1 Current Prototype and Limitations

We evalauted the functionality and performance of a small number of configurations of the ParkingBLE system. Our current prototype consists of up to three nRF52840 devices and one ICM-20948 device. Consequently, the configurations we tested have at most 3 layers, one for the sensor device, one for a relayer device, and one for the central device. We also tested the functionality of the minimum viable product design, consisting of one central device and one sensor device.

From initial testing, we discovered that the ICM-20948's magnetometer functionality, the primary sensor necessary for the ParkingBLE system, is not well suited for the nRF52840. In particular, queries of its registers provided little usable data. To address this issue, we opted to use two approaches: the accelerometer sensor in order to build a proof of concept where the sensor device can query the ICM-20948 via its I2C interface, and simulated data in order to showcase functionality and performance. In future iterations of this system, we intend to use a different magnetometer sensor connected to the sensor device.

5.2 Results

To test the functionality of our prototype, we ran the minimum viable product implementation in debug mode. For testing purposes, we opted to use smaller intervals for the sensor device: 15 seconds for standard interval, and 30 seconds for "throttled down" intervals. Figures 6 and 7 shows example output from running this implementation in conjunction with the functioning web server. We also successfully tested the interval frequency change the sensor device via button interrupt-based "throttle down" feature in the central device. Our output demonstrated in the figures includes the advertisement buffers received and sent for both the sensor and central device, as well as output designated for the web server to parse and display in the user interface shown in Figure 7.

We were also interested in the effectiveness of the ParkingBLE protocols on the efficiency of the sensor device, in particular, in the context of the minimum viable product implementation. However, since energy profiling is difficult for the nRF52840, we opted to measure a proxy instead: time. By utilizing the nRF52840's clock cycle counter in debugging mode, we measured the number of clock cycles spent that the sensor device spends performing tasks specific to the ParkingBLE system, including event handling, advertising, and scanning. On average, the total time spent in interrupt handlers

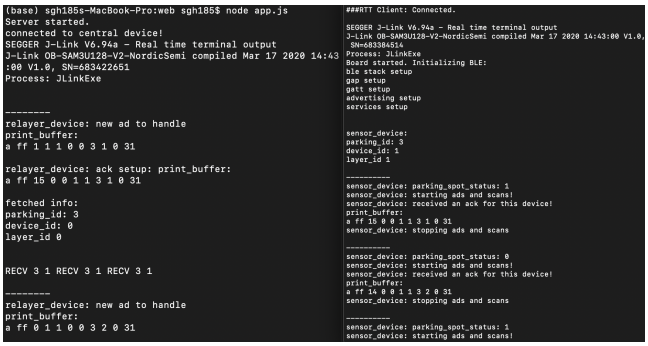

Figure 6: An example of the minimum viable product implementation in debug mode, in real time, and with simulated data. The webserver is running on the left terminal, printing out the central device's output that it parses. The right side terminal shows the output of the sensor device generating the simulated data.

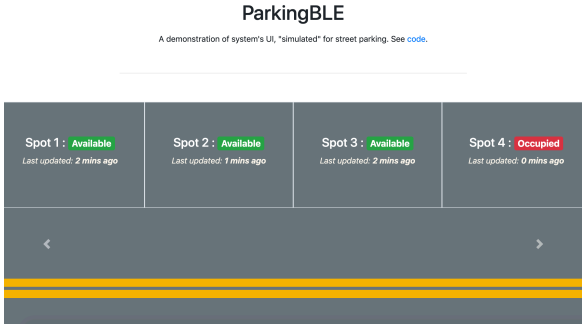

Figure 7: A demonstration of the user-interface for ParkingBLE that corresponds to the output from the figure above.

and advertisement event handlers averages 28,353 clock cycles (with a standard deviations of 1,442 clock cycles), which represents 0.000029% of the interval time. Interrupt handling is more expensive, amounting on average 21,649 clock cycles, while advertisement event handling takes 6,704 clock cycles on average. Additionally, we also measured execution with a different reference point: the time spent advertising and scanning. We started measuring from the point advertising and scanning began until the point when an ack was received. On average, 29,625 clock cycles pass after starting advertising and scanning before receiving an ack (with a standard deviation of 3,437 clock cycles). Generally speaking, these results are promising indications of the effectiveness of the ParkingBLE system design.

6 CONCLUSIONS

From this project we can see the viability of applying wireless Internet-of-Things technology to try and solve the problem of parking. ParkingBLE's utilization of the many advantages of BLE advertisements demonstrates the wireless protocols ability to solve these sorts of everyday problems. Energy efficiency, a wide range of coverage, and a cheap platform lie at the heart of Internet-of-Things, and its relevancy and convenience to the every day user is ever expanding as the technology behind Internet-of-Things expands.

7 ACKNOWLEDGEMENTS

This project is a part of the course CS397/497: "Wireless Protocols for the Internet of Things," Winter '21, Northwestern University. Many thanks goes out to Branden Ghena for his help and guidance through the quarter and our classmates for their feedback.

8 ARTIFACT

All of our code is publicly available on GitHub (at https://github.com/sgh185/nu-wirelessiot-base/tree/project). To use and deploy our code, please follow the build instructions. Note that the most basic system requires multiple nRF52840 devices and the Segger J-Link toolchain.

9 REFERENCES

[1] Kindt, P., Yunge, D., Diemer, R., and Chakraborty, S. Energy Modeling for the Bluetooth Low Energy Protocol, 2020. ACM Trans. on Embedded Computing Systems (January 2020). Pages 2-4.

[2] Siekkinen, M., Hiienkari, M., and Nurminen, J. K. How Low Energy is Bluetooth Low Energy? Comparative Measurements with ZigBee/802.15.4. WCNC 2012 Workshop on Internet of Things Enabling Technologies, Embracing Machine-to-Machine Communications and Beyond. 1-6

[3] Mekki, K., Bajic, E., Chaxel, F., Meyer, F. Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT. (2018) IEEE Conference - Second International Workshop on Mobile and Pervasive Internet of Things.

[4] Gomez, C., Oller, J., Paradells, J. Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. ISSN 1424-8220. Page 20.