Poster Abstract: Multi-Connection Scheduling based on Connection Subrating for Fair Resource Allocation in Bluetooth Low Energy Networks

Moonbeom Kim
Chung-Ang University
School of Computer Science and Engineering
Republic of Korea

Jeongyeup Paek
Chung-Ang University
School of Computer Science and Engineering
Republic of Korea

ABSTRACT

Bluetooth Low Energy (BLE) is a representative wireless technology for Internet of Things (IoT) that enables concurrent communication with multiple devices at low power. However, the connection establishment mechanism of BLE is susceptible to *resource overlap problem* among connected peripherals, leading to resource unfairness. To address problem and improve resource utilization, we propose a *"Subrating-based Connection Scheduling (SCS)"*. It schedules and periodically manages connections by considering the service requirements of the connected devices. Preliminary experiments demonstrate that *SCS* achieves higher throughput and fairness compared to popular commercial BLE stacks while satisfying the requirements of peripherals.

CCS CONCEPTS

• **Networks** → **Network protocol design**; **Link-layer protocols**.

KEYWORDS

Bluetooth, BLE, Multi-Connection Scheduling, Resource Fairness.

1 INTRODUCTION

Bluetooth Low Energy (BLE) has become an integral technology in various daily devices including smartphones, smartwatches, desktops, and tablets. It provides diverse features, including multi-connection, allowing users of various wireless system and application (e.g. audio/video, environment monitoring/control, and in-vehicular) to experience convenient high-quality services. In multi-connectivity networks, resource scheduling is an essential mechanism to ensure quality of service (QoS) by satisfying the requirements of each device. However, most commercial BLE devices independently schedule the connection and manage the resources for peripherals without any consideration of their requirements or status. Eventually, these schemes cause a *resource overlapping problem*, where a newly connected peripheral invades the resources allocated to previously connected ones. Bluetooth specification does not specify how a central device should manage the resource sharing of multiple connections [1].

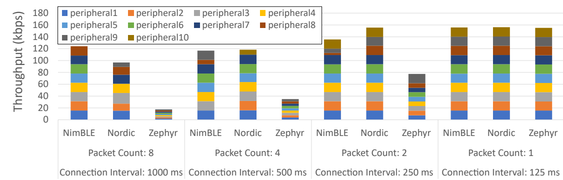


Figure 1: Resource Overlapping (Max. Payload Size: 244 Bytes).

Fig. 1 shows the consequences of the resource overlap problem in popular BLE stacks (i.e. Nimble[1], Nordic[2], and Zephyr OS[3]) from our preliminary experiments. These BLE stacks typically adopt a very simple connection anchor point decision mechanism and priority policy to quickly establish new connections without taking into account the status and requirements of the peripherals. For example, they set the *transmit window offset (txWinOffset)* to zero and the *transmit window size (txWinSize)* to 1.25 or 2.5 ms. These enable rapid transmission of the first packet while not affecting synchronization. For this reason, their BLE central device will block the current *connection event (connEv)* to start the next one, which will cause the length of the blocked connEv to be shorter than time needed for a data exchange. This ultimately results in not only performance degradation, but also disconnection between central and peripheral.

To address this problem, we propose *"Subrating-based Connection Scheduling (SCS)"* using the *connection subrating* feature in the latest Bluetooth specification version 5.3 [1]. *SCS* manages the independently handled connEv of peripherals on a synchronized schedule and determines anchor points for appropriate resource allocation while considering the resources required for each connection. In this poster, we implement *SCS* on an nRF52840 DK[4], and evaluate the performance by comparing it to the NimBLE, Nordic, and Zephyr OS BLE stacks on a real testbed.

2 *SCS* DESIGN

Connection subrating allows fast connection updates with minimal delay while maintaining the power-saving properties of low-duty

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A4A5034130 & No. 2021R1A2C1008840), and also by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2023-RS-2022-00156353) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

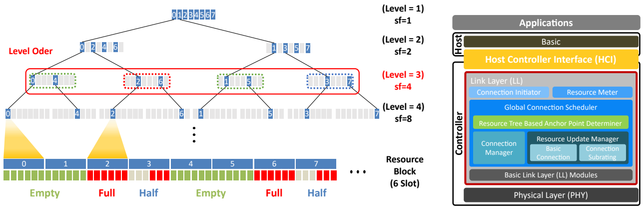
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) 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.

SenSys '23, November 12–17, 2023, Istanbul, Turkiye

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0414-7/23/11…$15.00
https://doi.org/10.1145/3625687.3628411

[1] Apache Mynewt-NimBLE, https://mynewt.apache.org/latest/network/index.html
[2] Nordic Semiconductor, https://www.nordicsemi.com/
[3] Zephyr Project, https://www.zephyrproject.org/
[4] nRF52840 Development Kit, https://www.nordicsemi.com/Products/Development-hardware/nrf52840-dk



(a) Resource Management Tree. (b) Central's BLE Stack.

Figure 2: *SCS* Overview

cycle connections [1]. It also allows fast transition from low-duty cycle to high-duty cycle (and vice versa), efficiently handling dynamic packet rates or bursty traffic. To address the resource overlapping problem and achieve connection fairness, we leverage these features of the connection subrating in the design of *SCS*.

SCS schedules the connEv of peripherals and allocates their resources into a *resource block list* which consists of 512 blocks and has a period of 3.84 sec. A block includes six *resource slots*, and each slot represents a time resource of 1.25 ms. On the resource block list, each connEv is synchronized with the clock of the BLE central. However, when each *connection interval (connItvl)* of the peripherals is mutually prime, the size of the list and scheduling complexity may increase exponentially. To solve this problem, we leverage the approach in BLEX [4]. When a connection is established, *SCS* sets connItvl to 7.5 ms and calculates *subrating factor (sf)* as $7.5 * 2^n$ ms ($0 \leq n \leq 9$) to mimic the original connItvl.

Resource management tree: We adopt a *resource management tree* for resource search and update, as shown in Fig. 2a. It is a perfect binary tree based on bitwise operations to reduce the memory usage of the tree and the block list in resource-constrained embedded devices. All node sizes are 8 bits, and each leaf node represents the status of six resource slots (i.e. one block) with 6 bits. The root and internal nodes indicate whether resources, which repeat with specific *sf*, can be allocated to peripherals. By doing this, the total memory size required for resource management is 5119 bytes (resource list 4096 bytes and resource tree 1023 bytes).

BLE Central stack of *SCS* is depicted in Fig. 2b. We only modify the link layer of the controller on the central side. *SCS* contains a total of 6 modules, and the operation procedure is as follows:

(1) When a central establishes a connection with a peripheral, *connection initiator* sets the connItvl as 7.5 *ms* and calculates the *sf* as defined earlier.
(2) *Anchor point determiner* searches for available repeated resources based on *sf* in the resource management tree using level order traversal. It then determines the *subrate base event (sb_ev)* for scheduling the connection. To avoid overlap between peripherals with different connItvl, if the available slots is more than twice the necessary resources, it determines the anchor point as $index = \lfloor \frac{(lastIndex - firstIndex)}{necessaryResources * 2} \rfloor * necessaryResources$. Otherwise, *firstIndex* is set as the anchor point.
(3) *Connection manager* calculates the parameters *txWinOffset* and *sb_ev* depending on the index of resource block list found in the resource management tree.
(4) To adapt to necessary resource changes of each peripheral, *resource meter* uses an exponentially weighted moving average to periodically measure changes in the connEv period.

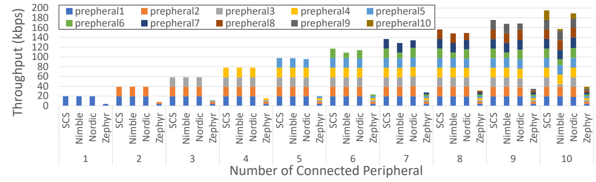


Figure 3: Throughput based on the Number of Peripherals.

(5) If the measured value is lower or higher than the threshold[5], *resource update manager* updates the resources for the peripheral by using the default or connection subrating procedure.

3 EVALUATION

We evaluate the performance of *SCS* on a real testbed under stress scenario and compare it against the NimBLE, Nordic, and Zephyr OS BLE stacks. The connItvl of all peripherals is set to 500 ms, and each node transmits 5 packets consecutively (max. payload size: 244 bytes) during a connItvl. Under this setup, the maximum reachable throughput of each peripheral is 19.52 kbps.

Fig. 3 plots the throughput and resource occupancy of each peripheral depending on the number of peripherals connected to the central. Nordic has an avg. throughput of $\approx$ 18.91 kbps, which is $\approx$ 97 % of the expected throughput. For Nimble and Zephyr OS, the avg. throughput are measured as $\approx$ 15.71 ($\approx$ 80 %) and $\approx$ 3.88 kbps ($\approx$ 20 %), respectively. Moreover, all three other stacks do not achieve fair throughput among peripherals. In particular, Zephyr OS suffers significantly from the resource overlapping problem. This is because the central quickly establishes a new connection by preempting previously scheduled connection without considering any countermeasures against resource overlaps. On the other hand, *SCS* achieves $\approx$ 19.52 kbps, 100% of the expected max. achievable fair throughput, for each and every peripheral, thus improving not only the throughput but also the resource allocation fairness.

4 SUMMARY

We have demonstrated the resource overlap problem of popular BLE stacks, and proposed a *Subrating-based Connection Scheduling (SCS)* scheme to address the problem. *SCS* manages the synchronized connection schedule with consideration of each peripheral on a single timeline, and fairly allocates resources while guaranteeing QoS. *SCS* achieves throughput upto $\sim$ 20 % (Nimble), $\sim$ 3 % (Nordic), and $\sim$ 80 % (Zephyr OS) higher compared to other commercial BLE stacks. In our future work, we plan to enhance *SCS* for dynamic resource rescheduling in various scenarios and conduct comparative experiments against prior studies [2–4].

REFERENCES

[1] 2023. Bluetooth Core Specification Version 5.3. https://www.bluetooth.com/specifications/specs/core-specification-5-3/ [last accessed on Sept 2023].
[2] F John Dian, Amirhossein Yousefi, and Sungjoon Lim. 2018. Time scheduling of central BLE for connection events. In *IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 763–767.
[3] Yeming Li, Jiamei Lv, Borui Li, and Wei Dong. 2023. RT-BLE: Real-time Multi-Connection Scheduling for Bluetooth Low Energy. In *IEEE INFOCOM International Conference on Computer Communications*. 1–10.
[4] Eunjeong Park, Hyung-Sin Kim, and Saewoong Bahk. 2021. BLEX: Flexible Multi-Connection Scheduling for Bluetooth Low Energy. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week)*. 268–282.

[5] We set threshold to 2.5 ms which is the time needed to exchange the max size packet.