Sensors and Materials, Vol. 33, No. 1 (2021) 127–145
MYU Tokyo

S & M 2441

Duty-cycle Communication Protocol with Wi-Fi Direct for Wireless Sensor Networks

Shiho Tashiro[1] and Takuya Yoshihiro[2*]

[1]Graduate School of Systems Engineering, Wakayama University, 930 Sakaedani, Wakayama 640-8510, Japan
[2]Faculty of Systems Engineering, Wakayama University, 930 Sakaedani, Wakayama 640-8510, Japan

(Received July 28, 2020; accepted October 20, 2020)

Keywords***:*** sensor networks, duty cycle, Wi-Fi Direct, protocols

In order to enable environmental sensing in any place, wireless sensor networks that collect environmental sensing data through wireless multihop communication are coming into practical use. Since they may be installed without a power supply, sensor devices should be powered by batteries. For this purpose, IEEE802.15.4 has been standardized and commercialized. However, services using these sensor devices are not widespread because the sensor devices are expensive relative to the value of the provided services and the power consumption is unexpectedly large. Therefore, in this study, we proposed a new communication protocol called Duty-cycle Data-collection over Wi-Fi Direct (DDWD) for wireless sensor networks that works on cheap microchips such as ESP8266EX, which are equipped with Wi-Fi functions and are available for under 10 USD. In order to use ESP8266EX in multihop wireless sensor networks, it is necessary to transmit the sensing data to the sink node efficiently while sleeping periodically to reduce power consumption. In addition, inexpensive hardware such as ESP8266EX has low accuracy in time counting, which causes a large error in the sleep time interval and time synchronization. Therefore, the protocol must be robust against time synchronization errors. The proposed protocol adopts Wi-Fi Direct, in which a node dynamically switches its operation mode between SoftAP and STA modes according to the time-divided slot, and exchanges messages with neighboring nodes through Wi-Fi Direct. The communication path is determined in an autonomous decentralized manner. All nodes operate only in the allocated slot to reduce power consumption. By limiting the operating slot interval to about several tens of seconds, low power consumption and reliable data collection are possible at a constant time interval, even if an inexpensive Wi-Fi chip with low accuracy in time counting is used. We implemented the proposed protocol in ESP8266EX and conducted an experiment to show the reliability and low power consumption of the protocol.

1. Introduction

Wireless sensor networks (WSNs) are a promising technology for collecting measurements in a target field for environmental sensing and so forth. Since one of the critical problems in battery-powered sensor networks is power consumption, a tremendous amount of work has been

*Corresponding author: e-mail: tac@wakayama-u.ac.jp
https://doi.org/10.18494/SAM.2021.2994

ISSN 0914-4935
https://myukk.org/

performed to reduce power consumption in sensor networks. Typically, in sensor networks, duty-cycle behavior is adopted to reduce power consumption, in which each node alternates between wake-up and sleep modes. To improve the efficiency of duty-cycle devices in terms of power consumption, both the hardware and the software (i.e., protocols) have been improved. In the hardware, i.e., sensor devices and microchips, power consumption in the working (i.e., wake-up) mode as well as in the sleeping mode has been reduced. Recently, several microchips have implemented the deep-sleep mode, in which the current in the sleep mode has been greatly reduced by cutting off the power for all components except for the real-time clock (RTC). On the other hand, in the software, many techniques for duty-cycle MAC protocols have been developed to reduce transmission, reception, and listening power in communications.[1–3] As a result, the lifetime of battery-powered sensor devices has been enhanced to several months even with coin cells or AA batteries.

Unfortunately, these solutions have not been widely deployed for several reasons. In most cases, those protocols work on microchips designed for IEEE802.15.4,[4] and IEEE802.15.4 devices are still too expensive for practical deployment. Additionally, since IEEE802.15.4 expects a short transmission range, we must deploy a number of devices to cover a certain field with these protocols. From the industrial point of view, the low-cost coverage of the measured field is required to achieve (1) a relatively long range, (2) multihop networks, (3) a long lifetime by virtue of duty-cycle medium access control (MAC), and (4) a low cost due to the use of commodity devices. To the best of our knowledge, solutions to achieve these aims have not yet been devised.

Recently, a protocol called Wi-Fi direct for device-to-device communications has been standardized and implemented in commodity Wi-Fi microchips. Wi-Fi has a relatively long communication range and the devices are inexpensive since they are well populated. Recently, the ESP8266EX microchip,[5] which is a microprocessor compatible with an Arduino environment while supporting a Wi-Fi communication function, has appeared. This microchip supports both a deep-sleep mode and Wi-Fi Direct, and its price is extremely low (about 5 USD for a chip). With this microchip, we can potentially achieve practical sensor devices that satisfy the above conditions.[1–4]

In this paper, we propose a duty-cycle MAC protocol called Duty-cycle Data-collection over Wi-Fi Direct (DDWD) for inexpensive commodity microchips such as EXP8266EX towards practical solutions of using sensor networks. Different from existing duty-cycle MACs, DDWD has a long slot such as 30–60 s and collects sensed data from devices once every 30–60 min using Wi-Fi Direct. By taking advantage of deep-sleep functionality, sensor devices sleep most of the time to reduce power consumption to a very low level; we expect a lifetime of several months with coin cells or AA batteries, and furthermore we aim to achieve a positive power balance with energy-harvesting devices such as solar panels. We implemented DDWD on an EXP8266EX chip, and through evaluation, we proved that DDWD works reliably and robustly with low power consumption.

The organization of this paper is as follows: In Sect. 2, we provide a technical introduction of related technologies, i.e., EXP8266 and Wi-Fi Direct. In Sect. 3, we describe related work in the literature. In Sect. 4, we describe the proposed protocol DDWD. In Sect. 5, we present the evaluation result of the actual implementation of DDWD. We conclude the work in Sect. 6.

2. Basic Technologies

2.1 Wi-Fi Direct

Wi-Fi Direct[6] is a standard presented by Wi-Fi Alliance that directly exchanges data between end devices without access points (APs). It works with usual Wi-Fi standards such as IEEE 802.11a/b/g/n/ac/ax. In Wi-Fi Direct, a device virtually becomes an AP called SoftAP, and other devices connect to SoftAP to communicate with each other. Different from the ad hoc mode of IEEE 802.11, devices can communicate with each other only if one of the devices has SoftAP functionality. This provides seamless communication in the traditional Wi-Fi environment with traditional Wi-Fi devices. In contrast, in the ad hoc mode of Wi-Fi, all devices must be in the ad hoc mode simultaneously.

2.2 ESP8266EX

ESP8266EX is one of the microcontrollers with Wi-Fi (and Wi-Fi Direct) functions developed and sold by Espressif Systems that has an extremely low price.[5] Espressif Systems Corporation provides a series of ESP microcontrollers such as ESP32, all of which have similar functionalities with variations in minor specifications. ESP microcontrollers are programmable under the development platform Arduino,[7] and a dedicated library for ESP original functions is provided to support rich functions of ESP chips including Wi-Fi and Wi-Fi Direct. In ESP8266, we can use Wi-Fi Direct functions so that STA and SoftAP modes are dynamically switched at any time, where the STA mode provides the standard behavior for Wi-Fi end devices and the SoftAP mode provides the virtual AP behavior for Wi-Fi Direct. In addition, ESP chips allow users to log in and update the program code of devices from remote locations via Wi-Fi.

It is also worth noting that ESP chips support several low-power-consumption modes as shown in Table 1. The modem-sleep mode stops the power supply to the Wi-Fi modem to reduce power consumption, the light-sleep mode stops both the Wi-Fi modem and the CPU, both of which need hardware interrupt from the RTC or external devices. The sleep mode with the lowest power consumption is called the deep-sleep mode, in which only the RTC is working while all the other circuits are sleeping, and the chip resumes from the RTC timer. In this paper, we assume the deep-sleep mode in our protocol design to greatly reduce the power consumption of the devices.

Table 1
Low-power-consumption modes supported by ESP8266EX.

Mode	Operation Current	Description
Modem sleep	15 mA	Disables wireless communications
Light sleep	0.9 mA	Disables both wireless communications and CPU
Deep sleep	20 μA	Only RTC awakes

3. Related Work

A large number of communication protocols for low-power sensor applications have been proposed thus far. For multihop communication support in battery-driven sensor devices, IEEE 802.15.4 (ZigBee)[4] has been standardized, and the corresponding microchips have been released. However, as is known widely, the low-power performance in multihop operation is not sufficient for battery-powered sensor devices to operate in the long term. Multihop networks with Bluetooth Low Energy (BLE) have been studied in several papers.[6] However, they are also not suitable for long-term sensor operation; they are designed to operate on battery-rich devices such as smartphones. Also, the communication distance in IEEE802.15.4 and the BLE mesh is limited and too short for outdoor environmental measurements.

For lower power consumption, many low-power MAC protocols for sensor networks have been proposed. They are typically designed to provide duty-cycle operation, which alternates between wake and sleep operation modes to reduce power consumption. B-MAC supports duty-cycle behavior by providing a long preamble that covers a single duty-cycle period before signal transmission,[1] which enables devices to catch up with the preamble and prepare to receive the signal even in duty-cycle operation. RI-MAC is a receiver-initiated MAC protocol in which receivers periodically transmit beacons and the corresponding senders send frames when they receive the beacons.[2] Since senders sleep when they do not have frames to send, the energy efficiency is significantly improved. Yokotani and Yoshino proposed the Joint MAC and routing protocol for Beacon-Saving (JBS), which extends RI-MAC by combining MAC with routing functions so that nodes dynamically control forwarding paths for load balancing, resulting in far higher energy efficiency.[3] These methods can be implemented on microchips supporting IEEE 802.15.4 and enable devices with AA or coin batteries to work for up to several years. However, they need high-density device placement owing to the short communication range. In addition, the price of such devices is still considerably high because the IEEE 802.15.4 protocol is not well populated.

In contrast, Wi-Fi is the most populated communication standard; thus, commodity Wi-Fi devices are available at low prices. Also, since the transmission power is relatively large, practical long-range communication is supported. If we can achieve low-power sensor networks based on Wi-Fi, it will be useful in many practical scenes. For multihop networks on Wi-Fi, a large number of studies exist in the context of traditional Mobile Ad hoc NETworks (MANET).[9] In MANET, several routing protocols such as Ad hoc On-demand Distance Vector (AODV)[10] and Optimized Link-Start Routing (OLSR)[11] have been discussed under the assumption of the ad hoc mode in IEEE802.11 working as the underlayer protocol. However, in general, such studies have not considered power consumption, and so such sensor networks are not applicable.

Structuring networks with low-power and lossy links has been developed, and a routing protocol called Routing Protocol for Low-power and Lossy Networks (RPL) has been standardized.[12] This protocol supposes sensor networks and can work over a family of Wi-Fi standards.[13] RPL constructs a Destination Oriented Directed Acyclic Graph (DODAG) for routing paths instead of a tree, as in the traditional routing protocols. Therefore, each node may have multiple parents, i.e., multiple next-hops, and may forward packets to either of them according to the condition of links. With this property, relatively efficient packet routing is

possible even over duty-cycle MAC protocols. Although there have been several studies that combined duty-cycle MAC with RPL,[14–16] most of the RPL studies have been carried out on sensor network platforms such as ContikiOS[17] and TinyOS,[18] and only a few studies combined RPL with Wi-Fi.[19] No study have considered devices with low battery capacities equipped with AA or coin batteries, even in combination with Wi-Fi.

There have been a few attempts to reduce power consumption within the framework of Wi-Fi. First, there is a standard in the IEEE 802.11 family to reduce power consumption by taking a periodic sleep time. IEEE 802.11ah[20] has been standardized for IoT applications, in which long-range communication is possible by utilizing a sub-GHz band, and duty-cycle behavior is supported for low-battery sensor devices. IEEE 802.11ah specifies the down-clocked operation of IEEE 802.11ac so that it supports the existing commodity hardware that reduces prices of products. However, when we consider multihop behavior on the basis of these protocols, the synchronization of duty-cycle behavior is a serious problem to solve.[21,22] Also, since the synchronization needs a certain accuracy, cheap hardware with low clock precision is not acceptable. Thus, these IEEE 802.11-based methods seem to be unrealistic for building multihop sensor networks.

On the other hand, a communication standard called Wi-Fi Direct, which realizes communications among end devices without APs, has been deployed in several commercial devices. There are several academic studies that proposed multihop communications over Wi-Fi Direct standards.[23,24] However, these studies aimed at multihop communication among resource-rich devices such as smartphones, and so cheap and resource-poor devices were not supported. To the best of our knowledge, the proposed protocol DDWD is the first proposal of a communication protocol that achieves long-life sensor networks with inexpensive commodity Wi-Fi hardware equipped with a low battery budget.

Recently, a new category of communications called Low-Power Wide-Area Networks (LPWAN) is getting populated. LPWAN covers a wide area with long-range one-hop communications. However, LPWAN devices are currently expensive, and cheap deployment is not possible. Furthermore, representable LPWAN standards do not support multihop communications; thus, we need multihop forwarding methodologies if wire-connected APs cannot cover the target area. Part of the multihop protocol design proposed in this paper will be applicable to LPWAN.

4. Proposed Protocol DDWD

4.1 Assumptions

The sensor networks we consider consist of nodes and sinks. All sensor nodes periodically obtain measurement values, which are sent and relayed to one of the sinks. There are one or multiple sinks in a network, and they are supplied with power so that they are always ready to send/receive frames with nodes. We assume that each sensor node has its own node ID assigned by the administrators. We also assume that the interval of collecting measurement values is relatively long such as 30 min or 1 h. As sensor-node devices, we assume the use of low-price hardware. Specifically, we intend to use a family of ESP microchips, among which

we choose ESP8266EX for its low price with rich-enough functions. Since ESP8266EX works as a microcontroller equipped with a Wi-Fi communication function, we can build the sensor node as a one-chip device. We assume that the sensor nodes are powered by batteries with small power budget such as AA or coin batteries, such that power consumption should be extremely low, enabling them to operate for a long time. Also, in cheap microchips such as ESP series, the precision of the system clock is low such that it is generally hard to control the sleeping time accurately. The communication protocols should work even with this unreliable clock behavior.

4.2 DDWD: protocol overview

In this study, we present a communication protocol design that works on a cheap ESP8266EX microchip with an unreliable clock function. In the proposed protocol, we set a relatively long data collection interval such as 1 h, and a single data collection period is divided into 30–40 s time slots. Each node stays awake in only two slots in a period, sleeping in the rest of the slots, to sense and relay the measured values to sinks. In a single period, nodes located farther from the sink wake up earlier and sequentially relay data to the sink in a hop-by-hop manner. Thus, the working slot for each node is assigned accordingly (see Fig. 1 for an example). Data forwarding among nodes is performed sequentially in a series of slots to collect data at the sink. In this time-slot-based design, the power consumption of each node is low because only two slots in a period are awake, while simultaneously the data collection is robust against noise and a large clock drift because of the relatively long (i.e., at least several tens of seconds) time slots.

Initially, all nodes are in the initial state where they do not know the distance from the sink, and so they do not know the time slot in which they should wake up. We define the distance of a node from the sink as the number of hops required to reach the sink. In the initial state, each node is in the STA mode of Wi-Fi, in which it searches for its parent nodes (i.e., AP) to connect. Every parent found has a distance from the sink, and accordingly the node computes its own distance. Once the distance is determined, the node transits to the stable state and changes its Wi-Fi behavior to the SoftAP mode, in which it transmits beacons that include the distance encoded in the SSID string. Since the distance is defined as the hop count from the sink, the node that received the SSID computes its distance as the included distance plus 1. In this way, every node in the network sets its distance and transits to the stable state.

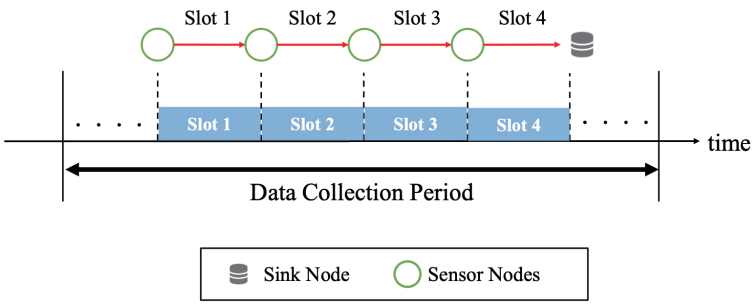

Fig. 1. (Color online) Slots and data transmissions.

In the stable state, each node calculates its working slots according to its distance. In a single data collection period, each node works on two slots; in the first slot, which we call the receiving slot, it receives data frames from its children, and in the following slot, which we call the sending slot, it sends all frames stored in the node to one of the parents. Nodes are awake in these two slots, and they sleep otherwise. Note that in the slot where nodes with distance d are sending, nodes with distance $d - 1$ are receiving. Because slots are assigned to nodes such that nodes farther from the sink are assigned with earlier slots, the data frame generated by any node can be collected at the sink within a single data collection period. Specifically, when a node with distance d is in the sending slot, it selects a parent randomly from the nodes that advertise their distance as $d - 1$ and sends all the stored data to the parent. In the next slot, the parent node does the same, and the data are passed to a node with distance $d - 1$. In this way, the data finally reach the sink. If a node cannot find any parent candidate, the node again transits to the initial state and renews its distance.

4.3 Determining distances

In the proposed protocol DDWD, when node n is powered on, it is in the initial state; in this state, the node does not know its distance from a sink. The node moves to the STA mode in Wi-Fi and scans APs working in the communication area. As a result, node n obtains a set of parent candidates, N_a. Specifically, in the scanning, node n scans APs for a certain period of time, and only the APs from which at least T beacons are received are included in N_a. These APs found are the nodes working in the SoftAP mode of Wi-Fi Direct, which advertise their distance encoded in their SSID strings via periodic beacons. Thus, from N_a, node n finds the minimum-distance node n_m (we let its distance be d_m), selects n_m as a parent of n, and sets its distance to $d_m + 1$. If there are multiple parent candidates with the same distance, node n selects one of them randomly as n_m. Once the parent is selected, node n connects to the AP n_m and sends a distance determination message that is destined for the sink. This message will be sent to the sink in the next data collection period. When n_m receives the message, the AP n_m sends back the time information to realize time synchronization between n_m and n, and finishes the connection. Note that there is a case where a beacon of the AP n_m is reachable but connection or data transmission to n_m is not possible owing to the different modulation methods or frame lengths used. In that case, when node n fails to transmit frames to the AP n_m, n removes n_m from the candidate set N_a and again selects n_m randomly from the other candidates. Once node n finishes this process, it starts working in the SoftAP mode and advertises its distance d_n by encoding it in its SSID string. Note that, in this paper, we do not care how it is encoded. After working for a certain time in the SoftAP mode, node n transits to the stable state. When all nodes run this process, all nodes determine the correct distance themselves and finally transit to the stable state.

In Fig. 2, we show an example of the node behavior in the initial state. First, only the sink node is working in the SoftAP mode and advertises its distance as zero. Nodes A–H are scanning APs and nodes A–E are within the beacon-reachable range of the sink s. Since the sink alone is the parent candidate of nodes A–E, nodes A–E set the parent candidate as $N_a = \{s\}$. Then, nodes A–C connect to the sink and determine their distance as 1. However, nodes D

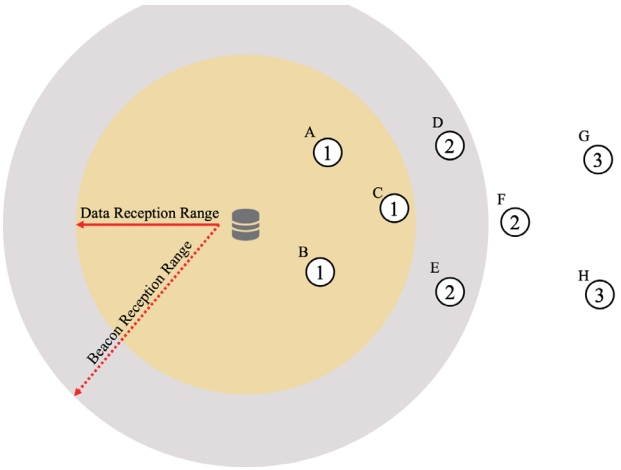

Fig. 2. (Color online) Example of determining distances.

and E fail to connect to the sink because they are not in the communication range of the sink. After that, nodes A–C move to the SoftAP mode and start advertising their distances. Nodes D and E find nodes A–C in their continuous scanning process, randomly select one of them, and determine their distance as 2 through connection to one of nodes A–C. Node F also does the same and sets its distance to 2. By repeating this process, nodes G and H determine their distance as 3.

4.4 Time synchronization

Our protocol assumes operation on inexpensive devices in which the time clock is not precise. However, since nodes change their behavior depending on the time slot, they need to recognize the boundary of slots accurately to some extent. Thus, in our protocol, we synchronize the time every time nodes exchange messages with their parents in order to ensure relatively accurate time recognition.

The first time synchronization is performed when node n_s determines its distance, as described in Sect. 4.3. Node n_s connects to n_r, sends a distance determination message, and receives the time information in return. Then, n_s advertises its distance in the SoftAP mode, sleeps, and wakes up at the beginning of the appropriate slot as a stable state behavior. In the stable state, every time a node exchanges a message with other nodes, time synchronization is executed, where the parent sends its time information and the children set the time.

Note that the time information is not just the time description, but also includes the information to determine the waking-up slots for nodes. Thus, specifically, it includes the slot interval t_{slot}, the maximum number of slots, S_{MAX}, the current slot number S_{num}, and the time elapsed from the beginning of the current slot, t. Every node can determine its wake-up slots based on these four values and its own distance, as we will describe in Sect. 4.5. The meanings of these values are illustrated in Fig. 3. Here, $R > t_{slot}S_{MAX}$ holds, where R is the length of the data collection period. S_{MAX} should be a sufficiently large value to support the maximum distance of nodes because the working slots for a node are determined according to the distance of the node.

Fig. 3. (Color online) Data collection period and slots.

4.5 Determining the working slots

A node first receives the time information from the parent node when its distance is determined in the initial state, and it determines its working slots from the time information and its own distance. A node has two working slots, i.e., the receiving and sending slots, where the sending slot is located as the next slot after the receiving slot.

In order to collect all data within a single collection period, the data collection starts with the maximum-distance nodes, for example, k-hop nodes with distance k. Namely, k-hop nodes send their data to $\{k-1\}$-hop nodes in a slot, and $\{k-1\}$-hop nodes send data to $\{k-2\}$-hop nodes in the next slot, and so on, which continues until all data reach the sink. Figure 4 illustrates this operation. To do this, the receiving slot of node n is allocated as $T_{r_n} = S_{MAX} - d_n$, where T_{r_n} is the number of receiving slots and d_n is the distance of n. Similarly, the sending slot T_{s_n} is allocated as $T_{s_n} = S_{MAX} - d_n + 1$. As a result, data collection starts in the middle of a collection period, and the sink receives the data in the final slot S_{MAX} of the collection period.

4.6 Behavior of nodes in sending/receiving slots

In the stable state, nodes work only in the receiving and sending slots, and sleep in the rest. In the receiving slot, nodes move to the SoftAP mode of Wi-Fi Direct to accept frames from other nodes. In the sending slot, they send their frames and then sleeps. In the following, we explain the behavior of nodes in detail.

At the beginning of the receiving slot S_m, node n_r wakes up and moves to the SoftAP mode. In the SoftAP mode, node n_r advertises its SSID built with the network ID, node ID, and distance, where the network and node IDs should be determined by the administrators. When node n_r is connected by sender n_s and receives its data, the receiver n_r sends back the time information to realize time synchronization and finishes the communication. On the other hand, the sender node n_s randomly wakes up in the STA mode, scans parent nodes, finds parent n_r, and sends data to parent n_r. Here, the sender node wakes up after a randomly determined wait, which is similar to the random back-off in Wi-Fi. This is because the same-distance nodes are located close to node n_s with high probability, and so we wish to avoid collision among them by randomizing the transmission time. Specifically, at the beginning of each sending slot, sending node n_s randomly chooses the waiting time t_w from the range $0 < t_w < (t_{slot}/3)$. Note that, because sending data via Wi-Fi takes at least several seconds or more time in the case of collision or bad radio conditions, we set the range shorter than the full slot time. After waiting for time t_w,

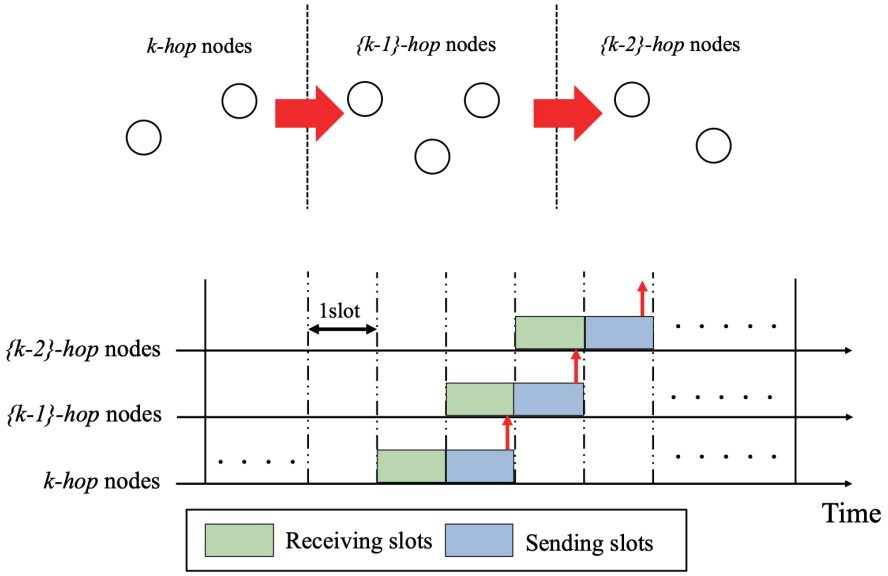

Fig. 4. (Color online) Data forwarding from k-hop nodes to $\{k-2\}$-hop nodes.

node n_s scans APs for a certain period of time, finds nodes both whose beacons were received more than T times and whose distance is $d_s - 1$, and makes a set of parent candidates, N_a. Then, node n_s selects parent n_r randomly from N_a, connects to parent n_r, sends data, receives the time information, disconnects, sets the wake-up timer for the next working slot, and sleeps. If n_s fails somewhere in this sequence, it repeats the operation again from scanning. Figure 5 illustrates the behavior of n_s and n_r in slot S_m, where sender n_s wakes up after the waiting random time t_w and sleeps after exchanging messages with n_r.

4.7 Updating distances

If all parent nodes fail, or if a fatal time synchronization error occurs, no parent node will be found in the sending slot. In this case, the node transits to the initial state again to find new parent candidates and updates the distance accordingly.

Specifically, for node n with distance d_n, neighboring nodes are those with distance $d_n - 1$, d_n, or $d_n + 1$. If all nodes with distance $d_n - 1$ disappear for some reason, node n cannot find any parent candidate in its sending slot S_m and transits to the initial state. In slot $S_m - 2$ in the next collecting interval, node n will find nodes with distance $d_n + 1$ and set its own distance to $d_n + 2$. Then, node n connects to the parent, sends a distance determination message to the sink, and continues to scan APs for a little more time. Later in slot $S_m - 1$, node n will find nodes with distance d_n, set its own distance to $d_n + 1$, and similarly send a distance determination message. In slot S_m or later, node n would not find any parent candidate so it finishes the initial state and transits to the stable state. In the next collection period, node n joins the forwarding process of the network. As a result, although node n is absent for one collection period, it returns to the collection process.

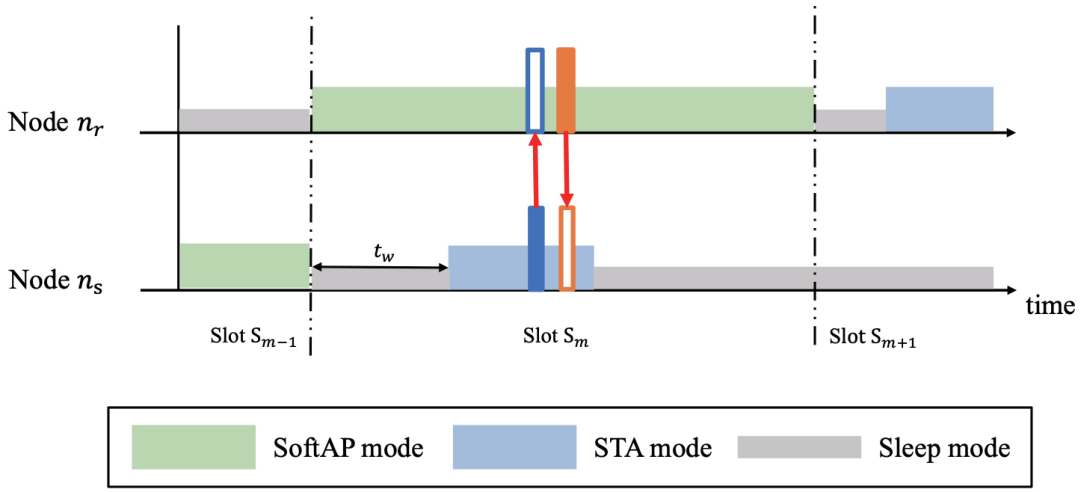

Fig. 5. (Color online) Behavior of sending/receiving nodes.

5. Evaluation

5.1 Overview

The proposed protocol in this paper is aimed at designing a practical protocol for sensor networks based on Wi-Fi Direct using inexpensive microcontrollers such as ESP families. Generally, inexpensive microcontrollers tend to have a low-precision clock; thus, we propose a protocol based on long slots to allow considerably large clock drifts. One of the focuses in our evaluation is to show that the proposed protocol works on real devices with commodity ESP-series microcontrollers. We implemented the proposed protocol with ESP8266EX, which is available for about 5 USD in 2020, and demonstrated that the protocol works in a real environment. The other focus in our evaluation is to measure the power consumption of the proposed protocol. In general, the power consumption with Wi-Fi communication is relatively high and not suitable for sensor networks. However, in the proposed protocol, we adopt a slot-based operation and each sensor basically works for 1–2 min in a single data collection period such as 60 min, which significantly reduces the power consumption. In this regard, we measure the power consumption performance of our devices and show that they have a sufficiently long lifetime under practical conditions.

5.2 Measuring accuracy in sleeping time

As a preliminary evaluation, we first measured the clock precision of the implemented device. We implemented the proposed protocol in EXP8266EX and tested the accuracy of the sleeping time, i.e., we measured the difference between the configured sleeping time and the actual sleeping time with the deep-sleep mode of EXP8266EX. Specifically, we tested five devices configured with sleeping times of 1–30 min.

We show the result in Fig. 6, in which the horizontal axis represents the configured sleeping time and the vertical axis represents the difference between the configured sleeping time and

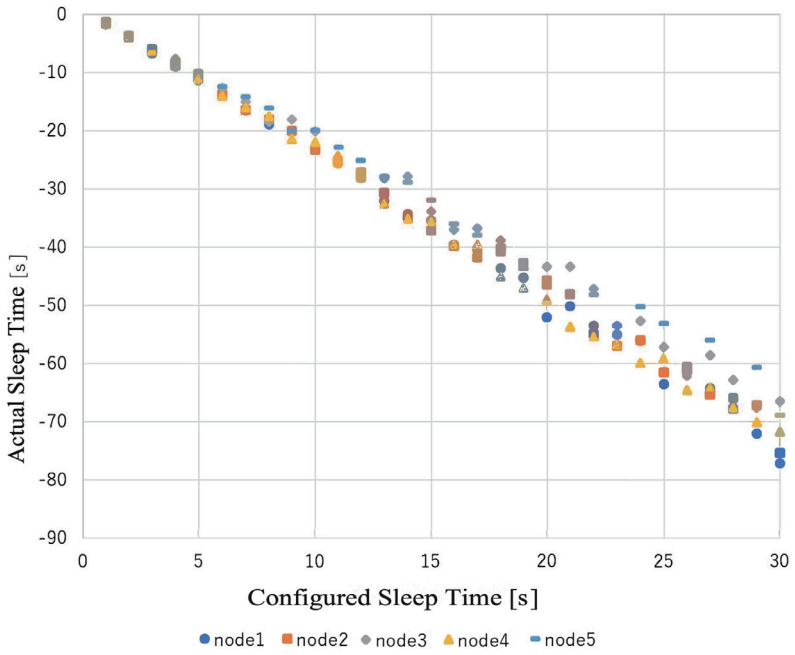

Fig. 6. (Color online) Error in sleeping time.

the actual sleeping time. The result shows that the error in the sleeping time has a strong linear correlation with the configured sleeping time, where the correlation coefficient is 0.994. Also, the variance of the error increases with the configured sleeping time. From this result, we compensate for the error by configuring the sleeping time as $1.04 \times t_{expect}$, where t_{expect} is the expected sleeping time.

5.3 Field evaluation

5.3.1 Scenario

To investigate the communication performance of the proposed method, we conducted a field evaluation with eight nodes and a sink. We searched for an evaluation field where the effect from other Wi-Fi radios is relatively small and selected a paved road in Wakayama University where the strength of other Wi-Fi radios was about −80 dBm. We set the eight devices in the layout shown in Fig. 7, i.e., we set one sink node and three nodes at 30 and 60 m locations, respectively, and two nodes at 90 m locations. The devices (both the sink and the other nodes) were set at a height of 20 cm, as shown in Fig. 8, and all devices were powered on simultaneously. As the protocol parameters, we set the slot length to 40 s and the length of the collection period to 240 s, so the number of slots, S_{MAX}, was 6, which is sufficient to afford three-hop networks. In the sending slot, nodes scanned APs 10 times, and the threshold for including the parent candidate set was $T = 7$ times, i.e., if an AP was detected fewer than seven times, it was not included in the set of parent candidates, $N_a k\ Na$. In the association process

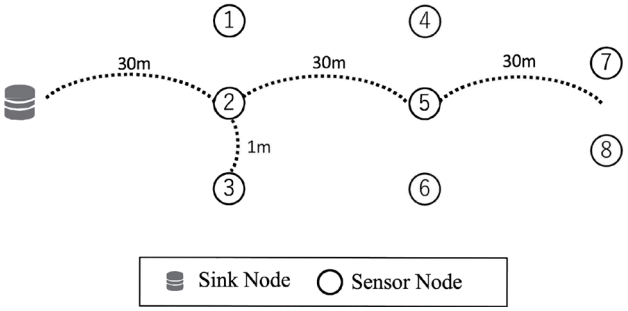

Fig. 7. Sensor locations.

Fig. 8. (Color online) Setup of sensor devices.

of Wi-Fi with a parent AP, if it took more than 15 s, we judged it as a connection failure and restarted the process from scanning APs. The collected value was a dummy 12-byte string. We ran the system for 11 collection periods and evaluated the data collection performance. A summary of the configuration is shown in Table 2.

5.3.2 Results

In Table 3, we show the number of data values generated and collected at each node. Because the number of generated data values at each node and the number of received data values at the sink are the same, we consider that all the generated data values are received by the sink without loss. The result shows that the proposed protocol DDWD reliably worked in a real environment. Note that the number of generated data values on node 4 is 9, 2 less than 11, which is because node 4 transits to the initial state once during the experimental period as explained later. Note also that the number of generated data values on nodes 7 and 8 is 10; this is because the initial state of these distant nodes finished later than that of the other nodes such that they entered the stable state from the second data collection period.

In Fig. 9, we show the data collection paths of each data collection period to show the detailed behavior of the network. In the initial state, the distance of nodes 2, 3, and 4 is set to 1, and this distance is kept until the end of period 11. As a result, the distance of node 1 is 2 throughout the experiment. Since the parent node is randomly selected from the candidates, the data collection paths are always different in each period. In particular, in the third and seventh periods, node 4 failed to connect to the sink, and the data held by node 4 at that time reached the sink in the next period. As a result, 11 data values out of 84 took two periods to reach the sink as shown in Table 4. From the result, we observe that sometimes nodes select parents of 60 m distance, which causes network instability, and may invoke a delay or loss of data. Next, in Fig. 10, we show a histogram of the wake-up times of nodes in their sending slots. We observe that in most cases, a short wake-up time of less than 17 s is sufficient to send data to the next hop, but sometimes a longer time is needed. Note that these long-time cases are the 60 m cases. This also shows that we should avoid unstable long connections in routing to achieve the stable

Table 2
Field evaluation parameters.

Category	Variable	Value
Node layout	Distance between nodes	30 m
	Node height	0.2 m
Parameter values	Slot length	40 s
	Data collection interval	240 s
	Connection threshold	7 out of 10 trials
	Association time limit	15 s
	Collected data size	12 bytes
	Running time	11 data collection periods

Table 3
Numbers of generated and collected packets.

Nodes	Sink	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
Numbers of generated packets	—	11	11	11	9	11	11	10	10
Numbers of received packets	84	1	14	11	28	13	6	0	0
Numbers of sent packets	—	12	25	22	37	24	17	10	10

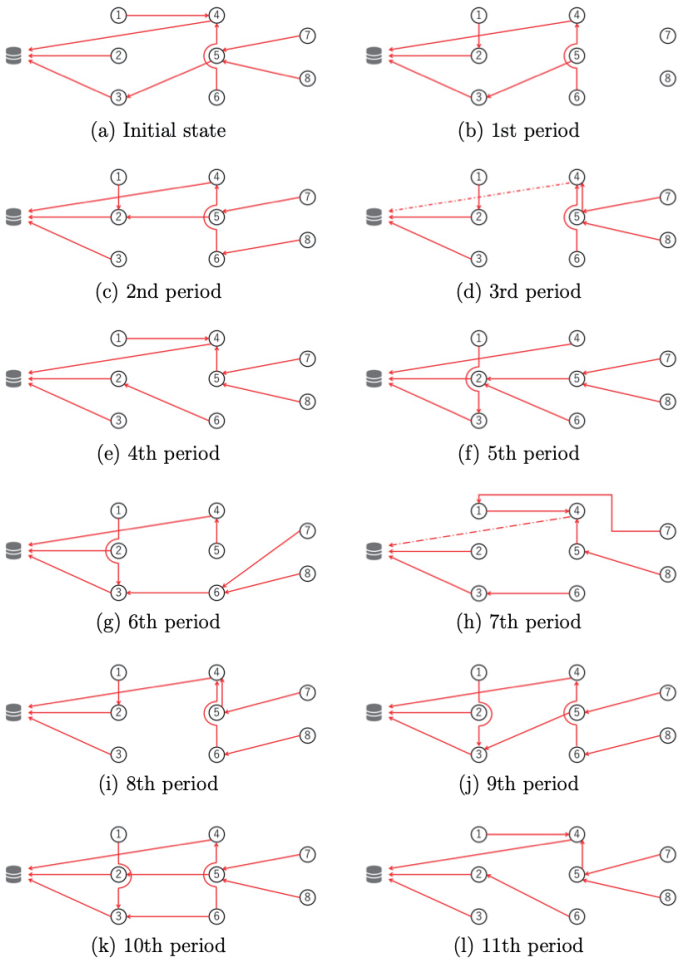

Fig. 9. (Color online) Routing paths for 11 periods.

Table 4
Delay of packets in reaching sink.

Numbers of data reaching sink within one period	73
Numbers of data reaching sink in two periods	11
Numbers of generated data	84

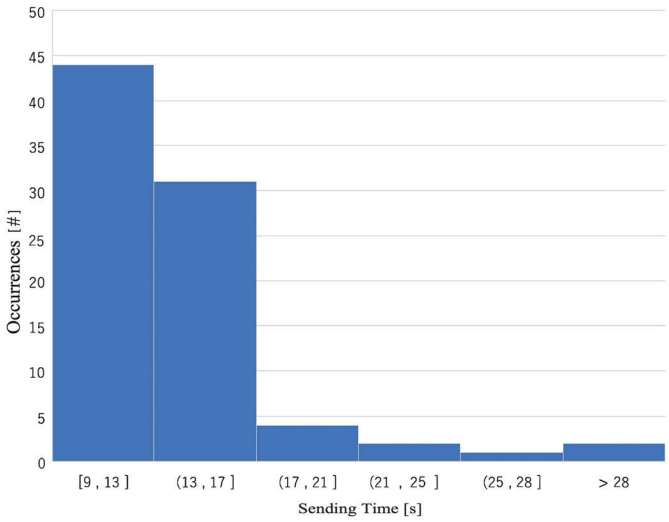

Fig. 10. (Color online) Wake-up time of nodes in sending slots.

collection of sensed data. Note that the slot length should be appropriately determined from the wake-up time distribution to prevent failure in packet forwarding. If node density or the number of packets to be forwarded is too large for the slot length, the data collection ratio will be unsatisfactorily low. When we cannot measure the wake-up time distribution directly, we can measure the data collection ratio at the sink. When we set up a new network, we should adjust the slot length t_{slot} such that the data collection ratio is satisfactorily high.

We thus confirmed that the proposed protocol stably works on an inexpensive microchip, EXP8266EX, even though its clock precision is not high. However, the protocol sometimes selects an unstable link to forward data, which causes a delay in data delivery. To achieve stable data collection, we need to introduce a mechanism to prefer entirely select stable links instead of randomly select links (i.e., parents) from the candidates.

5.4 Evaluation on power consumption

5.4.1 Scenario

To evaluate the power consumption of DDWD with our ESP8266EX implementation, we measured the lifetime of nodes in a typical scenario of sensor networks. We supposed environmental sensing to make measurements in a certain area of a field, so we assumed the

scenario shown in Fig. 11. In this scenario, we covered the field with 90 nodes, in which nodes with distance 1 (from the sink) have the heaviest load, i.e., each node receives seven packets from each of the children and sends 22 packets (7 × 3, plus 1 generated by itself) to the parent in each data collection period. In order to emulate the node, we set a device with three children and one parent, and received seven packets from each child and sent 22 packets to the parent in each period.

We show the evaluation parameters in Table 5. We used the same devices as those used in the field evaluation described in Sect. 5.3, where the devices were equipped with an NCR1850B[25] battery of 3400 mAh capacity, and minimal parameters were set to t_{slot}, t_R, S_{MAX}, and so forth.

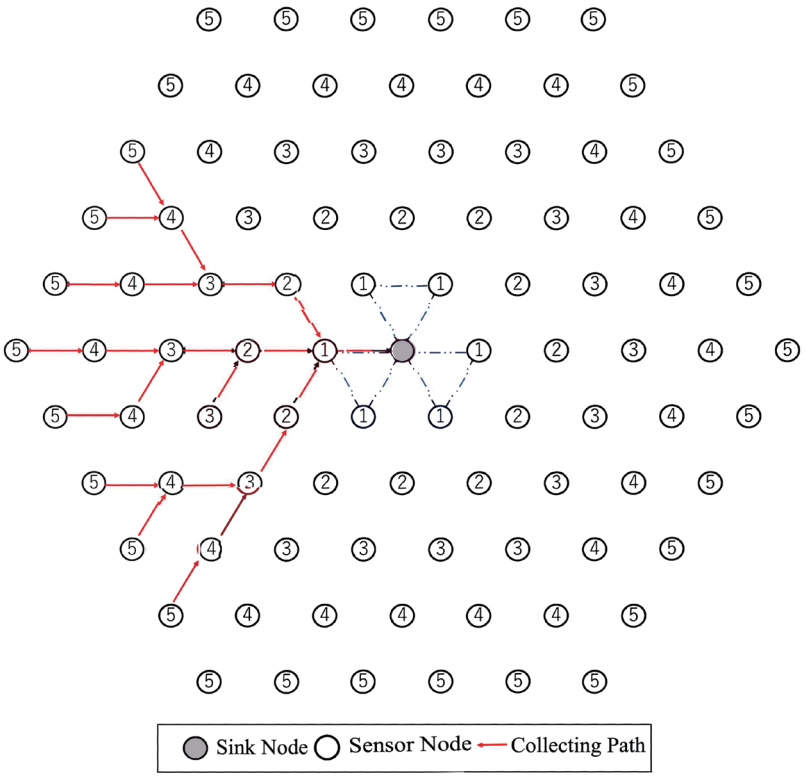

Fig. 11. (Color online) Scenario for power consumption evaluation.

Table 5
Power consumption evaluation parameters.

Parameter	Variable	Value
Battery capacity	$C_{battery}$	3400 mAh
Operation voltage	$V_{battery}$	3.6 V
Slot length	t_{slot}	40 s
Period length	t_R	160 s
Number of slots	S_{MAX}	3
Wake-up time in receiving slot	t_{recv}	40 s
Wake-up time in sending slot	t_{send}	13 s
Sleeping time in a period	t_{sleep}	107 s

Note that we assumed that the average wake-up time in a sending slot was 13 s, which was the average wake-up time in the field evaluation shown in Sect. 5.3.

5.4.2 Results and analysis

The devices were found to have a lifetime R of 2600 periods; thus, the total wake-up time T_{active} was 38.28 h and the total sleep time T_{sleep} was 77.28 h. From the results, we calculated the lifetime for a data collection period of 1 h in the scenario shown in Fig. 11. From the current in the deep-sleep mode, which was 20×10^{-3} mA, the total energy consumed in the sleep mode was computed as $C_{sleep} = T_{sleep} \times 20 \times 10^{-3} = 1.55$ mAh. Then, the energy consumed in the working slots was computed as $C_{active} = C_{battery} - C_{sleep} = 3398.45$ mAh. Note that the energy consumption in working slots is about $C_{active} / C_{sleep} = 99.96\%$ even when the data collection period is 160 s. The energy consumption per period in the sleep mode and that in the working slots are $C_{sleep}^{p=160s} = C_{sleep} / R$ and $C_{active}^{p=160s} = C_{active} / R$, respectively. If a period is assumed to be 1 h, since the wake-up time per period is the same, the values are $C_{sleep}^{p=1h} = C_{sleep}^{p=160s} \times [3600 - (T_{active} / R)] / [160 - (T_{active} / R)] = 0.02$ mAh and $C_{active}^{p=1h} = C_{active}^{p=160s} = 1.31$ mAh, respectively. The lifetime of the device was computed as $R_{lifetime} = C_{battery} / \left(C_{sleep}^{p=1h} + C_{active}^{p=1h} \right) = 106.78$ days. The computational results are summarized in Table 6. The results show that the device will have a lifetime of about 106 days if a data collection period is 1 h.

5.4.3 Discussion on deployment with solar panels

We show that the energy required by the device can be supplied with small solar panels. Generally, the estimated power generated by a solar panel is represented as $E_p = P \times H \times K$ where P is the capacity of the solar panel in watts, H is the average solar radiation per day, and K is the loss factor. We assumed a very small 1 W solar panel so that $P = 1$. According to a database on solar radiation records,[26] H is 3.32 kWh/m^2 in Tokyo if we set the solar panel horizontally. We used $K = 0.7$ as a generally used value for K. The amount of energy charged by the solar panel is $C_{solar} = E_p / V_{solar}$, where V_{solar} is the voltage of the solar panel. If we use $V_{solar} = 5.05$ V as a typical value for a 1 W solar panel, the estimated amount of power generated per day is computed as 422.55 mAh. Then, if we assume the voltage of the battery as $V_{battery} = 3.6$ V as shown in Table 5, the estimated amount of power charged per day is

Table 6
Power consumption evaluation results.

Item	Variable	Value
Total lifetime	R	2600 periods
Total sleeping time	T_{sleep}	77.28 h
Total wake-up time	T_{active}	38.28 h
Energy consumption in sleep mode	C_{sleep}	1.55 mAh
Energy consumption in working slots	C_{active}	3398.45 mAh
Energy consumption in sleep mode per period if period is 1 h	$C_{sleep}^{p=1h}$	0.02 mAh
Energy consumption in working slots per period if period is 1 h	$C_{active}^{p=1h}$	1.31 mAh
Lifetime if period is 1 h	$R_{lifetime}$	106.78 days

Table 7
Solar panel simulation results.

Description	Variable	Value
Average solar radiation per day	H	3.32 kWh/m^2
Loss factor	K	0.700
Solar panel capacity	P_{solar}	1.00 W
Generated power per day	E_p	2324 mWh (= 422.55 mAh)
Estimated charged power per day	C_{solar}	645.56 mAh
Energy consumption per day	$C_{(1\ day)}^{total}$	31.84 mAh

computed as 422.55 × (V_{solar} / $V_{battery}$) = 645.56 mAh as shown in Table 7. Since the power consumption of our implementation computed from the evaluation above is 31.84 mAh/day, even a 1 W small solar panel is sufficient to operate the system.

6. Conclusion

In this paper, we proposed a new duty-cycle communication protocol DDWD for sensor networks that works on Wi-Fi Direct. For practical reasons, we assumed an inexpensive commodity microcontroller with low-precision clocks such as the ESP family. To collect measured values periodically in a relatively large interval such as 1 h, the proposed protocol works on the basis of relatively large slots of 30–40 s to achieve robust data collection. To the best of our knowledge, DDWD is the first duty-cycle communication protocol for sensor networks that works on Wi-Fi Direct. We implemented the proposed protocol on a microchip, ESP8266EX. Through field evaluation, we confirmed that our protocol worked robustly to collect measured data from every sensor device. We also evaluated the performance in terms of energy consumption and found that the required energy for our protocol is sufficiently small to operate with a small solar panel. In conclusion, the proposed protocol robustly worked on inexpensive ESP-family microchips with sufficiently small power consumption. As future work, we aim to develop a better next hop selection method to achieve more stable data collection.

Acknowledgments

The authors sincerely appreciate the support of Professors Akiyama and Yamaguchi in Wakayama University on the implementation and evaluation of our protocol in a real environment.

References

1 J. Polastre, J. Hill, and D. Culler: Proc. 2004 ACM 2nd Int. Conf. Embedded Networked Sensor Systems (ACM 2014) 95–107.
2 Y. Sum, O. Gurewits, and D. B. Johnson: Proc. 2008 ACM 2nd Int. Conf. Embedded networked sensor systems (ACM 2014) 1–14.
3 M. Yokotani and T. Yoshihiro: 2018 IEEE 14th Int. Conf. Wireless and Mobile Computing, Networking and Communications (IEEE 2018). https://doi.org/10.1109/WiMOB.2018.8589180
4 IEEE Computer Society: IEEE Std 802.15.4-2003, IEEE Standard for Information technology: Telecommunications and Information Exchange Between Systems: Local and Metropolitan Area Networks Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer. https://ieeexplore.ieee.org/document/4621343

5 Espressif Systems ESP8266 WROOM Series: https://www.espressif.com/en/products/hardware/esp-wroom-02/overview (accessed July 2020).
6 Wi-Fi Alliance: Wi-Fi Direct, https://www.wi-fi.org/discover-wi-fi/wi-fi-direct (accessed July 2020).
7 Arduino: https://www.arduino.cc/ (accessed July 2020).
8 N. Todtenberg and R. Kraemerb: Ad Hoc Networks **93** (2019). https://doi.org/10.1016/j.adhoc.2019.101922.
9 A. Boukerche, B. Turgut, N. Aydin, M. Z. Ahmad, L. Boloni, and D. Turgutd: Comput. Networks **55** (2011) 3032. https://doi.org/10.1016/j.comnet.2011.05.010
10 C. Perkins, E. Belding-Royer, and S. Das: IETF RFC3561 (2003), https://tools.ietf.org/html/rfc3561 (accessed August 2020).
11 T. Clausen and P. Jacquet: IETF RFC3626 (2003), https://tools.ietf.org/html/rfc3626 (accessed August 2020).
12 T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander: IETF RFC6550 (2012), https://tools.ietf.org/html/rfc6550 (accessed August 2020).
13 J. Yi, T. Clausen, and Y. Igarashi: Proc. 2013 Conf. Wireless Sensor (2013). https://doi.org/10.1109/ICWISE.2013.6728773
14 A. Y. Barnawi, G.A. Mohsen, and E.Q. Shahra: Procedia Comput. Sci. **151** (2019). https://doi.org/10.1016/j.procs.2019.04.028
15 M. Michel, S. Duquennoy, B. Wuoitin, and T. Voigt: Proc. 2015 IEEE Int. Conf. Distributed Computing in Sensor Systems. https://doi.org/10.1109/DCOSS.2015.10
16 M. Bezunartea, M. Gamallo, J. Tiberghien, and K. Steenhaut: Proc. 2016 ACM 12th Int. Symp. QoS and Security for Wireless and Mobile Networks (2016) 135. https://doi.org/10.1145/2988272.2988279
17 A. Dunkels, B. Gronvall, and T. Voigt: Proc. 2004 IEEE 29th Annual Int. Conf. Local Computer Networks (IEEE 2004), https://doi.org/10.1109/LCN.2004.38
18 P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler: Ambient Intelligence, W. Weber, J. M. Rabaey, and E. Aarts Eds. (Springer, Berlin, 2005) Chap. 6. https://doi.org/10.1007/3-540-27139-2_7
19 H.S. Kim, J. Ko, and D.E. Culler: IEEE Commun. Surv. Tutorials **19** (2017). https://doi.org/10.1109/COMST.2017.2751617
20 W. Sun, M. Choi, and S. Choi: J. ICT Standardization **1** (2013) 83. https://doi.org/10.13052/jicts2245-800X.115
21 J. Elson, L. Girod, and D. Estrin: ACM SIGOPS Operating Syst. Rev. **36** (2002). https://doi.org/10.1145/844128.844143
22 S. Ganeriwal, R. Kumar, and M. B. Srivastava: Proc. 2003 1st ACM Conf. Embedded Network Sensor Systems (ACM 2002). https://doi.org/10.1145/958491.958508
23 N. Adam, C. Tapparello, and W. Heinzelman: Proc. 2019 Int. Conf. Computing, Networking and Communications. https://doi.org/10.1109/ICCNC.2019.8685663
24 C. Funai, C. Tapparello, and W. Heinzelman: Proc. 2019 Int. Conf. Computing, Networking and Communications. https://doi.org/10.1109/ICCNC.2017.7876178
25 Panasonic NCR18650B: https://www.batterijservice.nl/en/panasonic-ncr18650b-3400mah-unprotected.html (accessed July 2020).
26 New Energy and Industrial Technology Development Organization (NEDO) Solar Radiation Database. http://app0.infoc.nedo.go.jp (accessed July 2020) (in Japanese).

About the Authors

Shiho Tashiro received her B.E and M.E from Wakayama University, Japan, in 2017 and 2020, respectively. She is currently working with Kyocera Communication Systems Co., Ltd. She is interested in wireless networking and communications as well as sensor networking.

Takuya Yoshihiro received his B.E., M.I., and Ph.D. degrees from Kyoto University in 1998, 2000, and 2003, respectively. He was an assistant professor at Wakayama University from 2003 to 2009. He has been an associate professor at Wakayama University since 2009. He is currently interested in graph theory, distributed algorithms, computer networks, wireless networks, medical applications, bioinformatics, and so forth. He is a member of IEEE, ACM, IEICE, and IPSJ. (tac@wakayama-u.ac.jp)