Journal of Engineering Science and Technology
Vol. 16, No. 6 (2021) 4666 - 4686
© School of Engineering, Taylor's University

ENERGY EFFICIENT SCHEME FOR WIRELESS SENSOR NETWORKS BASED ON CONTIKIMAC PROTOCOL

OMER ALI[1,2], MOHAMAD KHAIRI ISHAK[1,*]

[1]School of Electrical and Electronic Engineering, Universiti Sains Malaysia (USM)
Nibong Tebal, Pulau Pinang, 14300, Malaysia
[2]Department of Electrical Engineering, NFC Institute of Engineering & Technology (NFC IET), Multan, 60000, Pakistan
*Corresponding Author: khairiishak@usm.my

Abstract

The Internet of Things (IoT) based remote sensing applications have seen a growing trend in recent years in order to gather intelligent insights and perform real-time like decisions. The IoT applications mostly rely on the underlying Wireless Sensor Networks (WSNs) for information collection and communication. The radio operation of these WSN nodes are the most power hogging components, therefore, estimating and optimizing the power consumption on these nodes helps to improve the device lifetime. Modern low-power radios are efficient but still require optimization per application basis, therefore, MAC and cross-layer protocols help to achieve this by optimizing the device's radio duty cycles. ContikiMAC is a low-power protocol designed for WSNs that can be tuned to adjust its radio duty cycle by optimizing clear channel assessment (CCA) and channel check rates (CCR) per application basis. This article first analyses the performance of ContikiMAC in a bursts and non-uniform data environment and reports the limitations. Secondly, it investigates the ContikiMAC operation for infrequent data environments. Finally, an extension to ContikiMAC is developed for improved CCA which reduces idle-listening and false-wakeUps for improved energy consumption per node. The simulation results from Cooja Simulator thoroughly investigates these schemes and reports energy consumption improvements in contrast to ContikiMAC are reported.

Keywords: Clear channel assessment (CCA), ContikiMAC, Internet of things (IoT), Radio duty cycling (RDC), Wireless sensor networks (WSN).

1. Introduction

The proliferation of the Internet of Things (IoT) devices has seen a major growth and adoption in recent years where almost 50 billion IoT devices are currently operational [1, 2]. These IoT devices are now deployed in various scenarios ranging from residential to industrial applications, to turn sensor data into actionable insights that help make informed and adaptive decisions [3, 4]. Whether the information collection is from sensors on a vehicle, on-body sensors in a health-care environment [5] or remotely sensed data from oil fields, these IoT devices mostly depend on the underlying WSNs to sense, collect and relay the data to the enterprise networks or customer applications. This requirement puts a strong focus on the WSN nodes to be low-cost, low-power and smaller in form factor so that they can be deployed over a dense and non-uniform area. Furthermore, typically these WSN nodes rely solely on very small on-board battery or energy harvesting mechanisms to power up their operations [6, 7]. It is also very challenging in most of the cases to physically access these devices to perform maintenance or simply to install a new battery. Therefore, optimizing the power consumption of the WSN has been one of the most prominent research topics in this domain. Almost every IEEE 802.15.4 compliant radio hardware is designed to operate in low-power mode nowadays [8, 9]. Together, hardware-based solutions such as Ultra-low power wake-Up receivers (WuR) further help to reduce the energy consumption of these nodes during *wakeUp* and *idle-listening* periods [10, 11]. However, software-based approaches to further improve the energy consumption of these devices are more favourable as it allows the flexibility to tune the hardware as per application needs and without the requirement of any additional hardware. MAC and associated cross-layer protocols are mostly utilized to improve the radio duty cycle of these devices to conserve energy.

MAC protocols utilize Clear Channel Assessment (CCA) mechanism to probe the availability of radio channel before initiating communication. The CCA provides two important purposes in a meshed network. It first looks for the channel availability to avoid collisions and thereby adjusts the *idle listening* and *sleep* intervals of the nodes. Second, it records the radio energy in the medium by estimating the received signal strength indicator (RSSI) and allows communications only if it meets the threshold requirements, otherwise, it puts the node immediately to *sleep*. This CCA implementation has been a core strategy for *low-power-listening* modes-based MAC protocols for energy-efficient WSNs [12]. CCA schemes have been thoroughly studied in collision avoidance scenarios for both wired as well as wireless networks [13]. However, utilizing CCA for dynamic radio duty cycle adjustments for infrequent data network had little attention. Therefore, this research focuses on radio duty cycle adjustment based on the CCA activity of ContikiMAC protocol to reduce *idle-listening* periods and *false wakeUps* [14].

A typical radio activity can be categorized in three states, namely *sleep, wakeUp* and *idle-listening*. It is very important to consider that most of the network time at each node is consumed in *idle-listening*, whereas the *wakeUp* period is the most power-hogging of all. In a sleep mode, the radio is completely turned off and ideally, no power is wasted. The *idle-listening* period affects not only the time of radio *wakeUp*, but also its interval. A smaller CCA period may completely miss a transmission packet, whereas a larger CCA period may keep listening to channel availability, even when there is no activity. The *wakeUps* are normally associated with the timings on the radio devices. A positive *wakeUp* will enable the device to perform CCA check and proceed with communications. On the other hand, a *false WakeUp* may keep the radio on looking for transmission, when there isn't any, thus puts a strong emphasis on CCA period to be adequate to report the optimum RSSI thresholds.

WSNs have seen a lot of Operating Systems (OS) offerings that support low-power operation. In this research, we utilized Contiki OS [15], which is a lightweight WSN OS mainly for reasons as follows:

- Lightweight operating system supporting multiple hardware architectures.
- Ability to simulate the protocols using Cooja Simulator that emulates the actual device drivers.
- Open-source structure, that helps with rapid prototyping and practical implementations.

Most of the protocol implementation in Contiki OS favours immediate sleep, which helps to conserve energy between every sensing and relaying of node. In this research, Contiki-MAC protocol is used because it supports radio duty cycling for its low-power-listening mechanism. Contiki OS supports a radio duty cycling (RDC) layer to adjust the duty cycle of the radios by varying the *Channel Check Rate (CCR)* in its kernel. This allows to adjust the duty cycle per application basis.

As this research focuses on infrequent data networks, ideally, the duty cycle should be lowered. First, we measure the impact of uniform versus non-uniform data nature in a dynamic environment on ContikiMAC's performance. The performance degradation of ContikiMAC for bursty network with non-uniform packets establishes to focus on low-duty cycled infrequent data network for this research and a fixed *CCR* for the next stage. Next, with a fixed lower *CCR* (to further reduce power), we focus on MAC layers adjustments to improve the CCA time to keep the nodes for maximum amount of time in *sleep* mode and minimum in *wakeUps*. In addition, the effect of proposed scheme is investigated over multiple network topologies to report the performance and energy consumption metrics. Finally, the proposed scheme is deployed on Texas Instrument's TI CC1350 platform to observe the performance improvements.

The rest of the paper is organized as follows. Section 2 features the related work in this domain. Section 3 presents the simulation network, the network model, and its attributes. Section 4 highlights the challenges and limitations of ContikiMAC. In section 5, the proposed method is thoroughly discussed. Section 6 provides the simulations carried out to compare the newly proposed extension to ContikiMAC. Section 7 concludes our work and provides directions for future work.

2. Related Works

Node power management is by far the most studied topic in WSN domain, which demands not only power conservation, but also modelling and estimation to effectively predict the node lifetime. This further enhances the application specific scenarios in IoT domain, where a node can be assigned various or limited functionality based on its computational capacity and active lifetime. In some of urban and office deployment scenarios, the 802.15.4 compliant radios experience severe interference from other mediums such as WiFi, smart phones that uses radio communications on the same 2.4 and 5 GHz bands, as per IEEE 802.11 standards [16]. In such cases, the received signal strength indicator (RSSI) is always noisy due to the presence of interference on the same radio channels. Such applications, require adaptive adjustment of LPL-RSSI threshold levels to avoid false *wakeUPs* as given in [17].

Raza et al. [18] focused on dynamic power management of the WSN nodes in *idle* period to estimate the power thresholds and keep the nodes in low power states. The received power levels were then utilized in asynchronous mode to intelligently sense the power profiles, which further enabled the opportunistic *wake* and *sleep* of hardware. Sakya et al. [19] in their research focused on an energy-efficient MAC protocol for mission critical WSN applications such as environment monitoring, disaster alerts and health-care etc. where high data streams are required to be relayed with minimum latency over an estimated and larger span of device life. The proposed model takes a two-tier approach where the node with maximum message queue and maximum energy amongst its neighbours is selected as a cluster head. Later, with a robust regression analysis, nodes with adequate energy are selected to be part of this virtual cluster and participate in transmission. In [20], a low-power On-Off Keying (OOK) based transceiver is used to flood the wake-up timings across the network in asynchronous mode. The simplified radio transceiver designed helped to mitigate hardware complexities whereas the network flooding further simplified cross-layer protocol complexities.

Shamna et al. [21] proposed a cooperative communication scheme at MAC layer for multi-hop WSN to improve throughput as well as network lifetime by utilizing multi-rate capability of IEEE 802.11 networks. The proposed algorithm actively probes the channel and neighboring nodes for channel condition and disseminates the packets based on either direct-path or co-ordinated multi-hop path, whichever supports higher data-rate and energy profile for the communication. DS-MAC [22] proposes an adaptive duty cycle adjustment scheme where the duty cycling is adjusted asynchronously based on the amount of data received. This scheme is particularly important in lower-data rate or infrequent data driven WSNs. The transmitting nodes sends the wake-UP prediction time in Acknowledgement (ACK) headers, which further simplifies the need to actively monitor the channel for incoming transmission.

In [23] data aggregation techniques are implemented to eliminate the need for multiple sparse transmissions across WSN. The energy-aware algorithm at routing layer picks the best path for transmission based on the energy-profile, whereas the aggregation-MAC aggregates and assembles the packets and transmits in a stream. Data aggregation schemes perform significantly well in low-to-medium data-rate traffics where energy and latencies can be balanced. In [24], Barnawi adopts a very unique data-aggregation approach for WSN. The proposed scheme adopts Time Division Multiple Access (TDMA) based scheduling where data is aggregated in small sized packets. Zikria et.al [25] adopted a cognitive radio based scheme for legacy IEEE 802.11 MAC protocol for better channel selection and utilization.

Wu et al. [26] utilized the cognitive radio and estimated the transmit power based on the availability of available spectrum in every channel. In a cognitive radio (CR) scheme, Primary Users (PU) release the spectrum which is opportunistically utilized by Secondary Users (SU). This proposed scheme efficiently monitors the power control based on spectrum occupancy in every channel. The above-mentioned schemes utilize both IEEE 802.11 as well as IEEE 802.15.4 MAC protocols mostly because of the similar radio frequencies, latency and timing requirements and CCA estimations at the MAC layer [27].

The non-coherent energy detection in packets is another effective technique for clear channel assessments. The RSSI of received packets are measured over the prescribed symbol length, that is used to identify the availability of the channel. IEEE 802.15.4 protocol provides flexibility in symbol detection time where a maximum symbol detection time of 8 symbol periods is recommended. However, IEEE 802.15.4 based implementation observe various symbol detection-based periods due to hardware constraints. Many researchers investigated the accuracy of idle channel availability by varying the CCA symbol period. In [28], the researchers proposed an adaptive CCA algorithm for channel monitoring. The proposed algorithm adaptively adjusts the symbol period window length during low and high traffic patterns. Their results improved channel detection accuracy as well as the overall power consumption by spending less time in channel monitoring. However, it can be argued that the proposed algorithm can only function for fixed interval, fixed, and lower data-rate applications. Similarly, Amirinasab et al. [29], proposed a light-weight CCA algorithm where dynamic RSSI calculations are made to adjust the CCA symbol period length. The proposed scheme reduced the radio duty cycle that helped to improve the node power consumption. However, the impact of various data-rates and network architectures were not reported in their study, that could potentially affect the performance of their proposed algorithm.

In this article, the proposed model focused on low-to-medium data rate application for infrequent event driven network where high throughput is not required. Contiki-MAC CCA is improved to have lesser *false wakeUPs* and *idle-listening* CCAs which can further improve the network's lifetime. Cooja simulator emulates the actual hardware code, which were implemented on low-power IEEE 802.15.4 radio, including Texas Instrument TI CC 224x, CC13xx platforms [30] in a small-scale lab environment.

3. Simulation Environment

In this study we have used the Contiki OS version 3.0 as the WSN OS on a virtual machine. Contiki is an open-source operating system for constrained devices and supports multi-tasking, networking with complete TCP/IP stack, low-power routing and MAC protocols and can compile the code for multiple hardware architecture types including ARM, PIC, MSP430 and the ability to port on 8051 based systems as well. The native code can be compiled on any hardware architecture which can be then emulated in its Cooja simulator, which provides a graphical user interface (GUI) to perform various WSN based tasks. In these simulations, we have used *CollectView* and *PowerTrace* modules in Cooja to analyse different simulation parameters as well as *Energest* module for software-based estimation on node power consumption.

3.1. Node requirements

A T-mote sky platform is used in this study which is designed on Texas Instruments TI MSP430 platform. The on-board crystal oscillator on MSP430 based sky mote runs at 32 kHz frequency, which can be programmed in the increasing power of 2 (2^0, 2^1, 2^2, 2^3, 2^4, 2^5) [31]. The sky mote uses TI CC2240 low-power chip as the transceiver that operates at 2.4 GHz.

The MSP430 microcontroller controls the CC2240 transceiver through a Serial Peripheral Interface (SPI) [32].

Based on the Contiki OS driver for T-mote sky platform, the transceiver chip can be programmed to adjust for CCA as well as RSSI, where:

i. CC2420 records RSSI thresholds for a minimum of 8 symbols period (128 µs) [33].
ii. CCA utilizes the average RSSI value and validates it with the threshold, which is programmable during compile time.

The Energest module uses CPU ticks to convert the CPU seconds into the resolution of the platform which is usually in milliseconds. The programmable Energest software-module estimates the total power available on a node which is based on the following:

i. CPU Time (with radios turned off)
ii. Low-Power-Mode LPM (node sleep mode)
iii. Transmit Tx and Receive Rx mode

The overall power estimation for a node at a given time, is calculated based on the energy available on a node as given in Eq. (1).

$$E_{Total} = E_{CPU} + E_{LPM} + E_{TX} + E_{RX} \quad (1)$$

The actual T-mote sky hardware energy profile from the manufacturer is given in Table 1.

Table 1. T-mote sky platform energy profile [32].

Parameter	Node State	Value
VCC	Node Supply Voltage	3 V
E_CPU	MCU on, Radio off	1.8 mW
E_LPM	MCU in sleep, radio off	0.054 mW
E_Tx	MCU on, Tx mode	17.7 mW
E_Rx	MCU on, Rx mode	20 mW

The algorithm looks for the current state of every node based on the difference in energy levels in its previous corresponding state. This Energest code routine is then implemented throughout the simulations to get real-time node power consumption statistics. These system parameters are then compared to every next recorded node value based on the sequence timer and the current residual node energy is calculated based on the following algorithm as given below.

Algorithm 1: Algorithm to compute real-time node energy

```
Result: Node Residual Energy Calculation
initialization;
while energest_init(); do
    prev_cpu = energest_type_time(ENERGEST_TYPE_CPU);
    prev_lpm = energest_type_time(ENERGEST_TYPE_LPM);
    prev_transmit =
      energest_type_time(ENERGEST_TYPE_TRANSMIT);
    prev_listen = energest_type_time(ENERGEST_TYPE_LISTEN);
    if sequence_no $\geq$ prev_sequence then
        e_cpu = all_cpu - prev_cpu;
        e_lpm = all_lpm - prev_lpm;
        e_transmit = all_transmit - prev_transmit;
        e_listen = all_listen - prev_listen;
    else
        energest_flush();
    end
end
```

3.2. Network requirements

The behaviour of ContikiMAC for variable duty cycle rates including (low, high, and non-uniform) with varying CCRs is investigated. In a multi-hop mesh network topology with non-uniform packet rate, the channel suffers a lot of radio interference as well as timing issues. Therefore, a mesh network topology based on 20 network nodes is established that places sky motes on 1 km^2, where each node has a 125m maximum transmission range, as per the manufacturer data sheet and is given in Fig. 1.

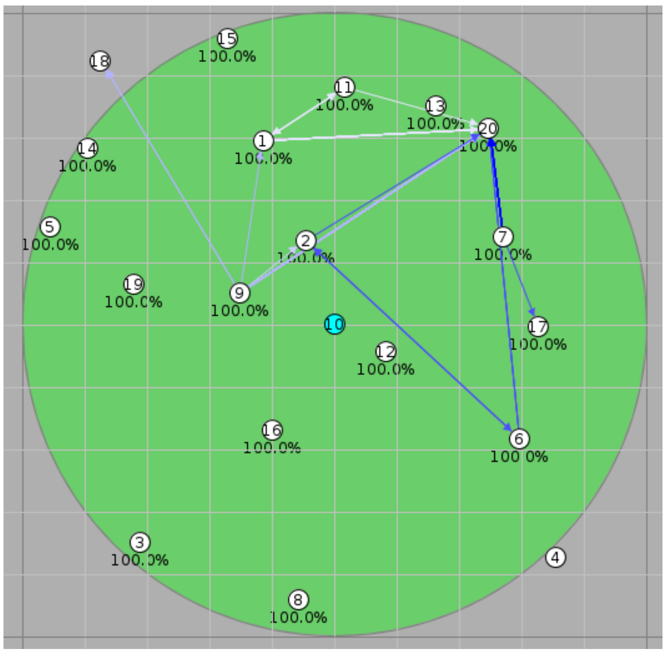


Fig. 1. Network topology for simulation scenario 1.

ContikiMAC protocol is investigated with its radio duty cycling (RDC) layer configured for multiple CCR ranging from 4 Hz to 32Hz. The network performance is measured in terms of the packet delivery ratio (PDR) for different duty cycles for all three scenarios. In this mesh topology, node 1 -19 act as random source nodes whereas node 20 is selected as the sink node. The network is fully meshed, which means that nodes relay their packets towards the sink node to form a fully converged topology. The power consumption estimates are made on every node, whereas the successful packet delivery ratio (PDR) is obtained from the sink node. The network, CCR and packet rate configurations are given in Table 2.

Table 2. Network configurations for Contiki-MAC variable packet rate test.

Parameter Description	Configuration
WSN Node Platform	T-mote Sky
Coverage Area	1 km^2
Range	125m
Protocol	Contiki-MAC with RDC

Channel Check Rate	**a). Low CCR** 4HZ **b). High CCR** 32 HZ **c). Non-Uniform** Nodes 2-6 → 4Hz Nodes 7-11 → 8Hz Nodes 12-16 → 16 Hz Nodes 17-20 → 32 Hz
Packet Rate	1/10, 1/5,1,5,10 per second
Source	Nodes 1 – 19
Sink	Node 20

The simulation environment is tested for the duty cycling impacts on a uniform as well non-uniform data on a meshed network, these results are discussed in the section that highlights some of the challenged in ContikiMAC for a non-uniform rate environment. These results will then pave way for our next set of assumptions for the proposed model that simulates ContikiMAC in a uniform environment, whereby only CCA rates and associated performance analysis in reported in Sections 5 and 6, respectively.

4. Challenges in ContikiMAC

The results divide the trends for all three scenarios from low to higher packet rates. It is observed, that as soon as the packet rates are increased, in Low-CCR based simulations, the network struggles to maintain the desired CCA frequency which results in the loss of packets and therefore drops the overall performance of the network. High-CCR scenarios perform comparatively better even when packet rates are increased due to its high CCA rates and ability to transfer the packets. However, as soon as the packet rate is increased closed to the boundary of packet intervals, the PDR performance drops significantly. On the other hand, the non-uniform CCR based distribution trends shows almost the same pattern where they perform better when the packet rate is closed to the interval period. These results are summarized in Fig. 2.

On the other hand, it is also very important to understand the power management and efficiency requirements for these resource constrained devices. In order to maintain the desired performance metric and keep the power consumption minimum the nodes must reduce their duty cycle. In these simulations, it is observed that the network, in terms of duty cycle, perform the best for low-CCR values. High and non-uniform provides reduced performance as soon as the CCR rate is increased, which requires to frequently monitor the channel for transmission as shown in Fig. 3.

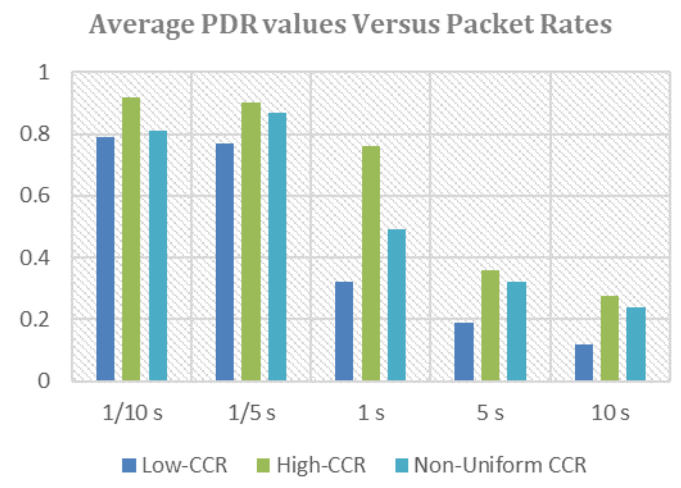


Fig. 2. Average PDR values versus packet rates.

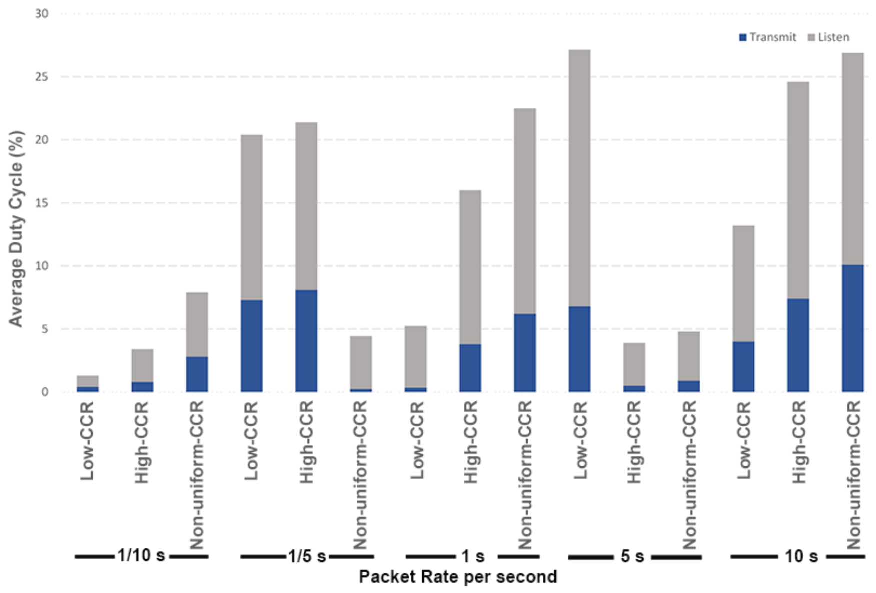


Fig. 3. Average duty cycle for variable CCR across packet rates.

Lastly, it is clear that ContikiMAC struggles to maintain its duty cycle when using non-uniform packet rates as it periodically looks for channel availability. With variations in packet rate, the protocol struggles to adjust its CCA intervals and records very high duty cycles for both *idle-listening* and *transmission* stages. Therefore, even with the ability to utilize the RDC layer on ContikiMAC, the protocol cannot adapt to the bursty nature of the channel and is most suited towards low-to-medium or infrequent data driven WSNs. This unpredictable behaviour forms the basis of our next assumption, where a simulation model is perceived for a uniform packet rate, fixed CCR ContikiMAC extension design that focuses on CCA values to improve the behaviour of the protocol.

5. Proposed Method

The proposed method investigates the effect of CCA adjustment on a low data-rate densely populated network. To this end, two network topologies were established to investigate false-wakeups and idle-listening time effects on aggregation nodes. In these experiments the emphasis was placed on selecting the number of aggregator nodes, whereas the source nodes were positioned randomly. A total of five simulations were performed for each network topology with various source nodes placement over 500 m^2 distance. The performance evaluation is then based on the average resource utilization for all network topologies. The network topology with the single aggregator node is presented in Fig. 4.

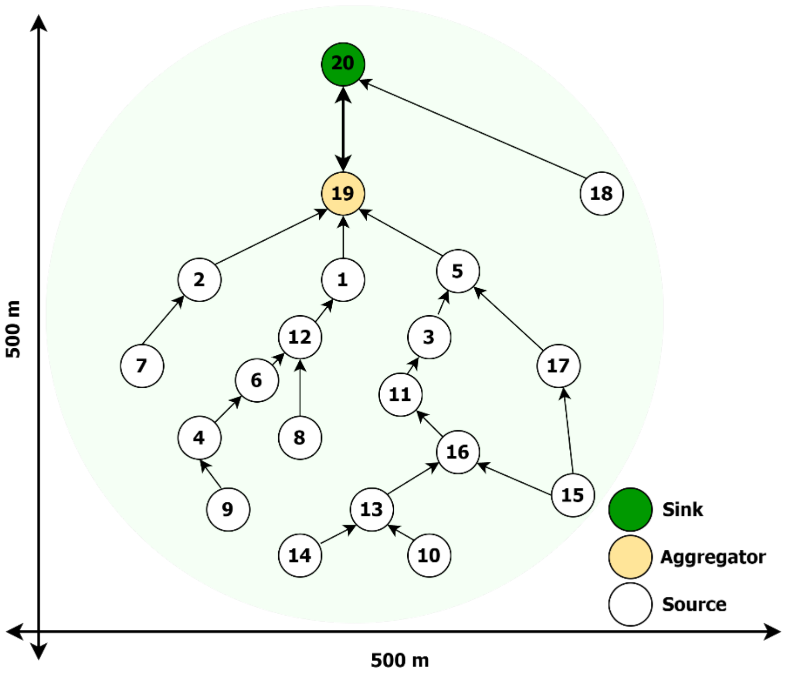


Fig. 4. Network topology with single aggregator node.

As mentioned in Section 3, the network can be programmed to have CCR up to 32 kHz, in these experiments a medium CCR value of 16 kHz is used because these rates are closed to symbol intervals and are susceptible to performance variations. Nodes 1-18 are used as *source* nodes, node 19 as aggregator node, whereas node 20 is used as a *sink* as well as RPL-border node. All the nodes have NULL_RDC configured, which would allow the nodes to stay powered on during the entire simulation period without changing its duty cycle. In the case of channel activity, the resource and power utilizations are monitored on each node. The successful packets are then relayed to node 20 which acts as a border gateway sink to complete the transaction. As node 19 appears as an aggregator that finally relays all the received packets to the *sink*, it is therefore our preferable choice to compute all energy and performance calculations on this node. In addition, nodes 1-17 were configured as *source* nodes, whereas nodes 18,19 were configured as the network aggregator nodes as based on the network topology given by Fig. 5.

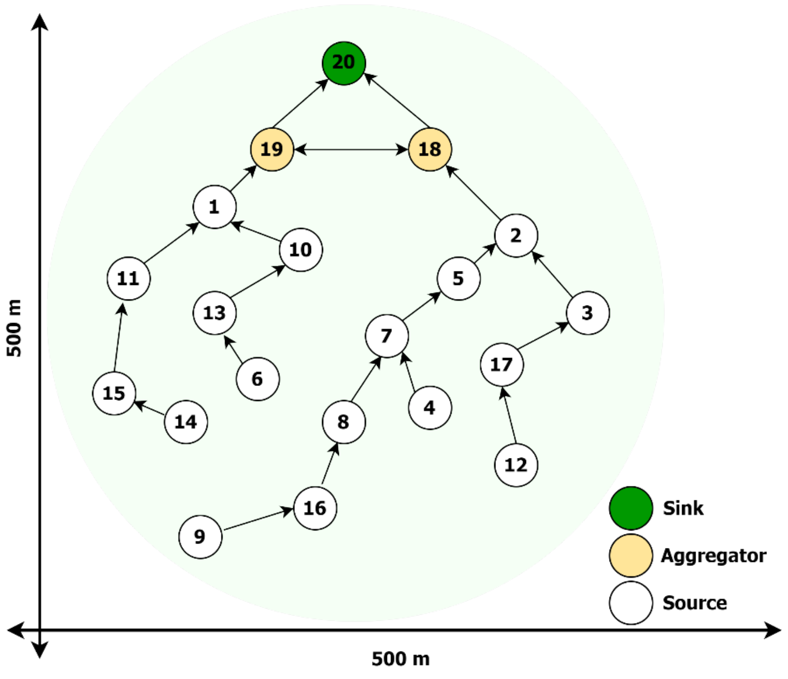


Fig. 5. Network topology with multiple aggregator nodes.

The experimental investigations were carried out on the aggregator nodes for energy and power consumption. These network topologies will be referred to as Network A and B, throughout this article. The network topologies may differ in node placement; however, the protocol configuration is consistent throughout the investigations. Table 3 describes the protocol configuration for each layer, during the study.

Table 3. Network configurations across multiple layers.

Layer	Protocol	Configuration
Application	Null	CollectView PowerTracer Custom Energest Algorithm
Transport	UDP	standard
Network	RPL/IPV6	standard
Adaptation	6LowPAN	Border router
Data Link	CSMA	standard
Radio Duty Cycling	Contiki-MAC	Null_RDC

In this simulation, the RPL protocol will automatically form a network topology by exchanging ICMP packets, whereas the 6LowPan layer will favour the exchange of IPV6 packets from the source to the sink. The ContikiMAC will perform its standard MAC layer functions by constantly monitoring the channel for activity

[34]. This simulation will utilize the node power requirements from Table 1 as well as the network configurations for Contiki as given in Table 4 below.

Table 4. Idle listening and sleep period timings from Contiki OS.

Function	CPU Time	Units (ms)
CCA_CHECK_TIME	32768/8192	0.4
CCA_SLEEP_TIME	(32768/2000) +1	1.7
MAX_NONACTIVITY_ PERIODS	10x(CCA_CHECK_TIME+CCA_SLEEP_TI ME)	21

ContikiMAC utilizes the period between CCA_CHECK_TIME and CCA_SLEEP_TIME to record the activity whereas MAX_NONACTIVITY_PERIODS estimates the time between radio activities while idly listening to the channel. It is this time interval which is later used to calculate false WakeUPs between activity periods. The CCA_CHECK_TIME is normally dependent on the hardware where this timing field is utilized to perform CCA checks. In this example, the hardware clock works on 32 kHz, so to calculate the CPU Ticks or resolution Contiki OS divides the timer by 4 to obtain ticks per second, which for MSP430 platform turns out to be 8192 ticks. Thus, it is important to estimate the idle and false listening times per node basis to estimate the original performance of ContikiMAC. The first simulation run in this scenario is carried out for 5 minutes and a timing response from several nodes is measured as given in Table 5.

Table 5. Idle-listening and false wake-up times of nodes.

Node	False WakeUp	Idle Listening
	Network A	
19	104	11,278
13	209	11,947
15	179	12,959
3	97	11,413
	Network B	
18	97	12,458
19	89	12,908
7	198	11,004
11	135	11,872
2	205	11,102

The total time spent by node 19 (Network A) in false wakeUps and idle listening is computed using Tables 4 and 5 and is reported in Eq. (2) as.

$$T_{node19} = T_{idle} + T_{falseWakeUp} T_{node\ non-active\ period} \tag{2}$$

$$T_{node19} = 11278 + (104)(21) = 13462\ ms$$

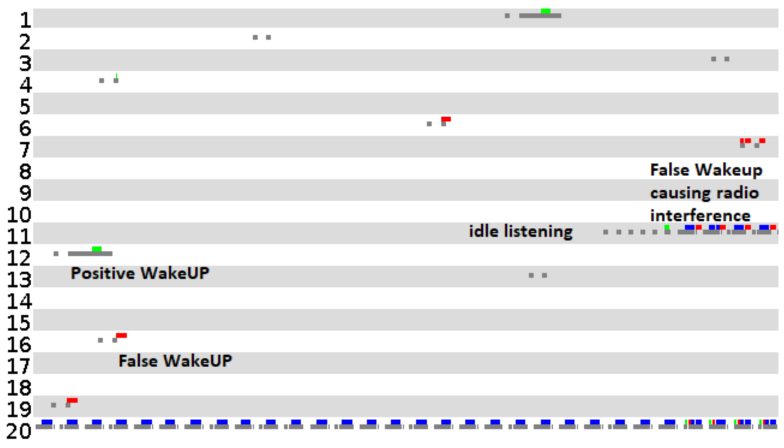
Similarly, the total time spent by aggregator nodes (18, 19) for Network B were calculated to be, 14,495 and 14,777 ms respectively.

5.1 RSSI estimation and verified time interval

The CC2420 requires 8 symbol periods to successfully make CCA calculations. However, the ContikiMAC spends more time in CCA for approximately 20 symbols period. These RSSI time check corresponds to t_RSSI = 0.128 ms (8 symbols) and T_{verify}= 0.32 ms (20 symbols) as given in [15], where,

$$T_{verify} = \frac{RTIMER\_SECOND}{10} \tag{3}$$

Therefore, a ContikiMAC implementation requires 8 symbols for positive radio activity and another 20 for RSSI verification and idle listening. Due to this extended idle-listening period, the radio actively looks for channel activity where in fact, no activity is present and mostly that is when the radio falsely wakes up based on RSSI values due to interference or neighbouring node traffic in the channel, which further increases the node idle time and wastes energy. Figure 6 is a timeline from our simulation using ContikiMAC in Cooja, where some of these periods are highlighted.



To address this long listening period, our proposed scheme adjusts the RSSI timer values in CC2420 drivers for Contiki OS, where T_{verify} are adjusted for offset and timing corrections. A brief code fragment from Contiki OS CC2420 driver (cc2420.c) is presented in Table 6.

As given in Table 6, the long symbol wait period for CCA validation is removed and an offset of one symbol period is used to avoid timing intersection.

Table 6. ContikiMAC CCA adjustment.

ContikiMAC	ContikiMAC extension
`while(!(get_status() & status_bit) && RTIMER_CLOCK_LT(RTIMER_NOW(), t0 + RTIMER_SECOND / 10);`	`while(!(get_status() & status_bit) && RTIMER_CLOCK_LT(RTIMER_NOW(), t0 + 1);`

6. Results and Discussions

Using the network parameters defined in the previous section and algorithm 1 (as given in Section 3) is invoked in Energest module to compute the CPU ticks required for the node energy consumption based on Eq. (1). Contiki utilizes power trace module to estimate the time spent in each state. The time spent in each state for aggregator nodes for all network topologies were computed and compared against the default ContikiMAC implementations. The CPU tick intervals are presented in Table 7.

Table 7. Comparison of average CPU ticks of aggregator nodes.

Protocol	CPU_Time	LPM_Time	Tx_Time	Rx_Time
Network A – Node 19				
Contiki-MAC	3898.12	31545.15	162.48	479.39
Contiki-MAC extension	4439.47	42746.72	144.63	498.27
Network B – Node 18				
Contiki-MAC Extension	3001.68	29127.03	127.98	163.87
Network B – Node 19				
Contiki-MAC extension	3612.48	30804.3	139.98	198.25

The linear power estimation model [35, 36] is used to calculate the power consumed in each state by computing the current consumption over time in a certain state. The overall power consumption can be computed using Eq. (1). The following code fragment function (“Contiki/collect/sensorData.java”) from Contiki OS is invoked to compute the power consumption in a certain radio state.

```
Public double getPower( )
{
Return (values [state_Time x E_state) / (values
[Total_Time]);
}
```

where,

State_Time is the time spent in each state (such as CPU_Time, LPM_Time, Rx_Time and Tx_Time)

E_state is the current energy consumption reported from algorithm 1 in each state (such as E_CPU, E_LPM, E_Rx, E_Tx)

Total_Time is the CPU time spent in ON and idle CPU states ($T_{Total\_Time} = T_{CPU\_Time} + T_{LPM\_Time}$)

These estimates are mathematically given in Eqs. (4) to (7) and are used to compute average power consumption in each state, which is later used in eq.1 to compute the total power drawn for the node during the simulation period.

$$E_{\_CPU} = \frac{CPU\_Time \times VCC \times E\_CPU}{CPU\_Time + LPM\_Time} \tag{4}$$

$$E_{\_LPM} = \frac{LPM\_Time \times VCC \times E\_LPM}{CPU\_Time + LPM\_Time} \tag{5}$$

$$E_{\_TX} = \frac{Tx\_Time \times VCC \times E\_Tx}{CPU\_Time + LPM\_Time} \tag{6}$$

$$E_{\_Rx} = \frac{Rx\_Time \times VCC \times E\_Rx}{CPU\_Time + LPM\_Time} \tag{7}$$

The above results show that for low-to-medium rates on the simulated networks, there is a slight increase in CPU as well as LPM_time. However, because the radios are in low power mode and only consume a fraction of the power as compared to transmission states, the overall effect on power consumption is significantly low. However, a significant improvement in Rx_Time is observed which will improve the overall power consumption. Texas Instrument (TI) development boards CC1352R were configured with the proposed algorithm to monitor the node power consumption. TI Code Composer Studio with EnergyTrace module was used to observe the packet exchange between source, aggregator and sink nodes for both network topologies. The experimental setup is presented in Fig. 7.

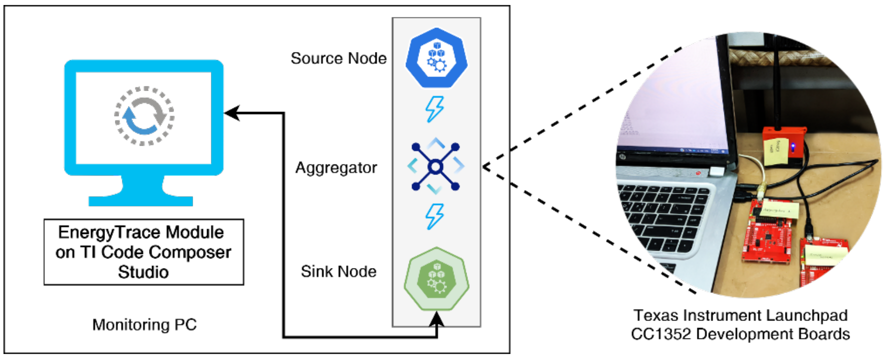

Source Node
Aggregator
Sink Node
EnergyTrace Module on TI Code Composer Studio
Monitoring PC
Texas Instrument Launchpad CC1352 Development Boards

Fig. 7. Experimental setup to measure energy consumption on IEEE 802.15.4 based TI CC1352 boards.

The network simulations were carried out for a period of 8 hours each, where, the average power consumption, node wakeup and idle-listening times were record for each network topology. These power consumption estimations utilized Eq. (1) and Eqs. (4) to (7), which are given in Table 8 and Fig. 8.

Table 8. Average power consumption (mW) of aggregator nodes.

Protocol	E_CPU	E_LPM	E_Tx	E_Rx	E$_{\text{Total}}$	PDR (%)
Contiki-MAC	0.59	0.145	0.243	0.811	1.789	99
Network A – Node 19						
Contiki-MAC extension	0.59	0.178	0.213	0.609	1.59	99
Network B – Average of Node 18 and 19						
Contiki-MAC extension	0.534	0.148	0.212	0.325	1.219	99

The proposed method with a single aggregator node reports slight improvement over ContikiMAC in terms of Rx and Tx (mW) power consumption of the nodes. However, as the number of aggregator nodes was increased, a significant overall power reduction was observed. It is important to mention that in real-world deployment scenario there is a limited control over source node placements, therefore, its impact cannot be accurately investigated. However, the positioning, and number of aggregator nodes is important. The results from Tables 7 and 8, clearly indicate that by the addition of aggregator node, the network is load-balanced resulting in lesser collisions, thus, reduced power consumption. It is also noteworthy, that the effect of CCA adjustments with multiple aggregator nodes reduces false wakeups, thus resulting in increased network lifetime. Finally, with the addition of multiple aggregator nodes, the network can be logically segmented that could help in provided multiple data routing paths for improved network convergence and reliability.

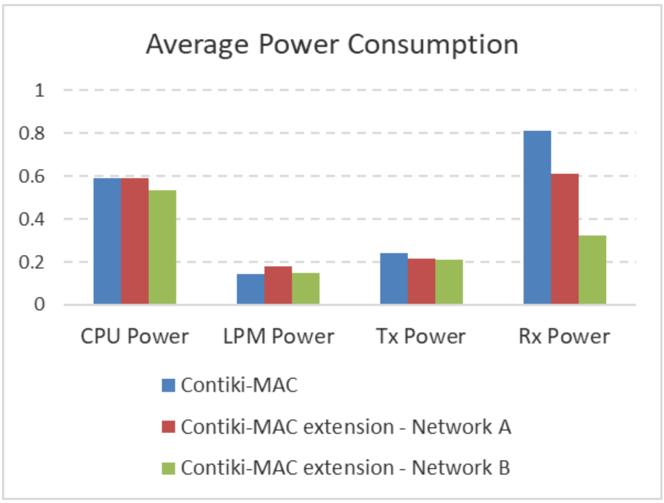


Fig. 8. Average power consumption (mW) of aggregator nodes.

The performance metrics using PDR suggests that for uniform infrequent packet rates, the proposed scheme performs well and can maintain a very high delivery ratio. The effect of CCA estimation, backoffs effects the PDR and in case of poor packet delivery performance, a re-transmission occurs which causes an additional overhead, thus consuming additional power. The linear power estimation is a simple and straight forward technique to estimate the available node power. In a simulation environment this technique can be utilized to estimate the power consumption in each state with little to no effect from the node available voltages. However, with the actual hardware, the linear estimation model suffers from the variance in voltage levels over time as the trend is not linear. It is also important to consider that the type of batteries also affect the power drawn and hence changes the power estimation [35, 37]. However, in this research as the battery voltages are simulated using Cooja simulator, the linear estimation model performs well.

Similarly, by varying the data rates (from low to high) the proposed scheme was implemented in the same simulation environment to observe the power consumption trends as given in Fig. 9. The transmission data rate was increased in the powers of 2^n, where n holds the positive integer values. The proposed scheme performs better as compared to the ContikiMAC for lower data transmission rates. It is observed that from low to medium data rates, the scheme provides marginal improvements over the ContikiMAC, however, for higher data transmission rates the average energy consumption increases slightly due to overlapping CCAs, backoffs and retransmission time intervals.

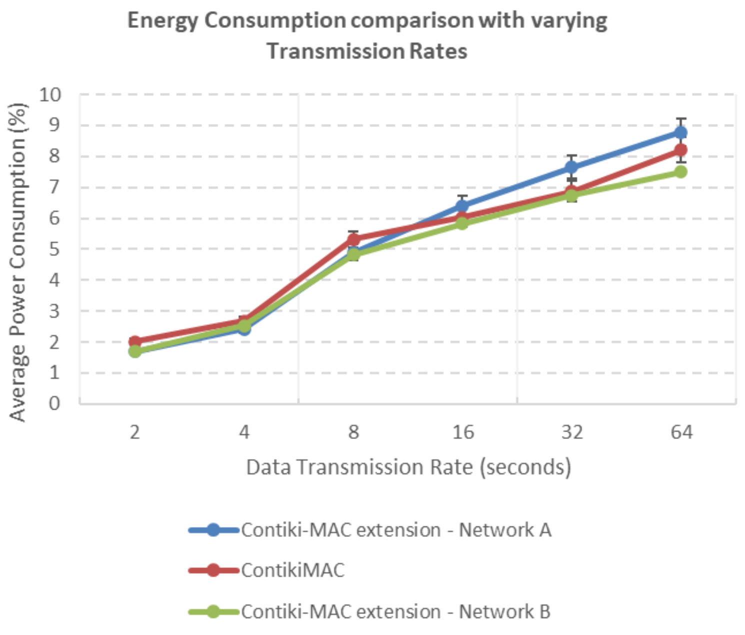


Fig. 9. Average energy consumption over variable data transmission rates.

7. Conclusion and Future Work

Modern IoT devices are being rapidly deployed in almost every sector these days. One of the most fundamental requirements is close to zero maintenance, improved lifespan and consistent data gathering. Most of these requirements require a low-power hardware operation so that these devices can last longer. IEEE 802.15.4 compliant radios are already power efficient devices but in order to improve the lifespan, these devices need to be optimized and tuned per application basis. These requirements are normally fulfilled at MAC and associated cross-layer protocols by adjusting the radio duty cycle of these radios. In this research we focused on ContikiMAC protocol for WSN using Contiki OS which is an open-source operating system for constrained devices. To optimize the power performance of ContikiMAC we first established a scenario where this protocol might not work at its best. It was investigated that ContikiMAC protocol struggles to deliver efficient results in a non-uniform packet rate environment. The simulation results clearly outlined that at higher packet rates, the protocol demands higher channel check rates which inherently requires to increase its radio duty cycling and thus making it less power efficient.

These assumptions then helped to build a WSN network model with uniform packet rate among all nodes. Sky motes were simulated in Cooja environment. The proposed scheme involved adjusting the CCA timings in ContikiMAC to avoid unnecessary symbol delays to validate the receive RSSI during channel inactive periods. Cooja simulator's Energest module was re-written to report the power changes per node basis during the simulation. Later, using Cooja's CollectView, node parameters were recorded. The proposed scheme provided slight improvements in power conservation as compared to ContikiMAC. The proposed scheme also helped to reduce false wakeUPs due to which the nodes returned to sleep immediately and therefore with next positive wakeUP period, the results showed a reduction in Rx times. The proposed scheme can be further extended in future work for higher data rates, different node density and network composition as well as adaptive CCA timing adjustments for any data rate and network payload.

Nomenclatures

E_{CPU}	Energy consumed in the nominal state
E_{LPM}	Energy consumed in the low power listening state
E_{Total}	Total Energy consumed by the nodes
E_{TX}	Energy consumed in the radio transmission state
E_{RX}	Energy consumed in the radio receive state
T_{node}	Total time spent by the node in a particular state
T_{verify}	Time verification interval for RSSI measurement
VCC	Node nominal supply voltage

Abbreviations

ACK	Acknowledgement
ARM	Advanced RISC Machines
CR	Cognitive Radio
CCA	Clear Channel Assessment
CCR	Channel Check Rate
CPU	Central Processing Unit
GUI	Graphical User Interface
IEEE	The Institute of Electrical and Electronics Engineers
ICMP	Internet Control Message Protocol
IoT	Internet of Things
MAC	Medium Access Control
MSP	Mixed Signals Processors
OOK	On-Off Keying
OS	Operating System
PDR	Packet Delivery Ratio
PIC	Peripheral Interface Controller
PU	Primary User
RDC	Radio Duty Cycling
RPL	Routing Protocol for Low-Power and Lossy Networks
RSSI	Received Signal Strength Indicator
SU	Secondary User
TI	Texas Instruments
TDMA	Time-division Multiple Access
WSN	Wireless Sensor Network

Author Contributions

Conceptualization, Omer Ali and Mohamad Khairi Ishak; Data curation, Omer Ali; Formal analysis, Omer Ali; Funding acquisition, Mohamad Khairi Ishak; Investigation, Mohamad Khairi Ishak; Methodology, Omer Ali; Project administration, Mohamad Khairi Ishak; Resources, Mohamad Khairi Ishak; Software, Omer Ali; Supervision, Mohamad Khairi Ishak; Validation, Omer Ali and Mohamad Khairi Ishak; Visualization, Omer Ali; Writing – original draft, Omer Ali; Writing – review and editing, Mohamad Khairi Ishak.

Funding

The authors would like to thank Universiti Sains Malaysia (USM) for the Research Grant (RUI: 1001/PELECT/8014049) to carry out this research.

References

1. Statista Research. (2018). Internet of things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions). Retrieved February 12, 2020, from https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.
2. Lee, I. (2019). The internet of things for enterprises: an ecosystem, architecture, and IoT service business model. *Internet of Things*, (7).
3. Jell, T.; Baumgartner, C.; Broring, A.; and Mitic, J. (2019). BIG IoT – interconnecting IoT platforms from different domains - final results. *Proceedings of the International Conference on Engineering, Technology and Innovation*. Valbonne Sophia-Antipolis, France, 1-4.
4. Rodriquez, J.M.; and Stammati, L. (2018). The economic impact of IoT: putting numbers on a revolutionary technology. Retrieved February 12, 2020, from https://www.frontier-economics.com/media/1167/201803_the-economic-impact-of-iot_frontier.pdf.
5. Fotopoulou, K.; and Flynn, B.W. (2006). Wireless powering of implanted sensors using RF inductive coupling. *SENSORS*, 765-768.
6. Dini, M.; Romani, A.; Filippi, M.; Bottarel, V.; Ricotti, G.; and Tartagni, M. (2015). A nanocurrent power management IC for multiple heterogeneous energy harvesting sources. *IEEE Transactions on Power Electronics,* 30(10), 5665-5680.
7. Dutta, P.; and Dunkels, A. (2012). Operating systems and network protocols for wireless sensor networks. *Philosophical Transactions of the Royal Society A,* 370, 68-84.
8. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; and Ayyash, M. (2015). Internet of things: a survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys and Tutorials*, 17(4), 2347-2376.
9. Ray, P.P. (2018). A survey on internet of things architectures. *Journal of King Saud University - Computer and Information Sciences*, 30(3), 291-319.
10. Ghribi, M.; and Meddeb, A. (2020). Survey and taxonomy of MAC, routing and cross layer protocols using wake-up radio. *Journal of Network and Computer Applications*, 149.
11. Oller, J.; Demirkol, I.; Casademont, J.; Paradells, J.; Gamm, G.U.; and Reindl, L. (2016). Has time come to switch from duty-cycled MAC protocols to wake-up radio for wireless sensor networks?. *IEEE/ACM Transactions on Networking*, 24(2), 674-687.

12. Zheng, X.; Cao, Z.; Wang, J.; He, Y.; and Liu, Y. (2017). Interference resilient duty cycling for sensor networks under co-existing environments. *IEEE Transactions on Communications*, 65(7), 2971-2984.
13. A. King, A.; and Roedig, U. (2018). Differentiating clear channel assessment using transmit power variation. *ACM Transactions on Sensor Networks*, 14(2), 1-28.
14. Chien, T.V. (2019). A comparative study of network performance between ContikiMAC and XMAC protocols in data collection application with ContikiRPL. *International Journal of Computer Networks and Information Security*, 11(8), 32-37.
15. Duquennoy, S.; Finne, N.; and Dunkels, A. (2012). The Contiki open source OS for the internet of things. Retrieved February 12, 2020, from https://github.com/contiki-os/contiki/blob/master/dev/cc2420/cc2420.c.
16. Yick, J.; Mukherjee, B.; and Ghosal, D. (2008). Wireless sensor network survey. *Computer* Networks, 52(12), 2292-2330.
17. King, A.; Brown, J.; Vidler, J.; and Roedig, U. (2015). Estimating node lifetime in interference environments. *Proceedings of the 40th Local Computer Networks Conference Workshops*. Clearwater Beach, FL, USA, 796-803.
18. Raza, U.; Bogliolo, A.; Freschi, V.; Lattanzi, E.; and Murphy, A.L. (2016). A two-prong approach to energy-efficient WSNs: wake-up receivers plus dedicated, model-based sensing. *Ad Hoc Networks*, 45, 1-12.
19. Sakya, G.; and Sharma, V. (2019). ADMC-MAC: Energy efficient adaptive MAC protocol for mission critical applications in WSN. *Sustainable Computing: Informatics and Systems,* 23, 21-28.
20. Sutton, F.; Buchli, B.; Beutel, J.; and Thiele, L. (2015). Zippy: on-demand network flooding. *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, 45-58.
21. Shamna, H.R.; and Lillykutty, J. (2017). An energy and throughput efficient distributed cooperative MAC protocol for multihop wireless networks. *Computer Networks,* 126, 15-30.
22. Wang, G.; Yu, J.; Yu, D.; Yu, H.; Feng, L.; and Liu, P. (2015). DS-MAC: An energy efficient demand sleep MAC protocol with low latency for wireless sensor networks. *Journal of Network and Computer Applications*, 58, 155-164.
23. Kim, D.; Kim, J.; and Park, K.H. (2011). An event-aware MAC scheduling for energy efficient aggregation in wireless sensor networks. *Computer Networks*, 55(1), 225-240.
24. Barnawi, A.Y. (2015). Aggregation for adaptive and energy-efficient MAC in wireless sensor networks. *Journal of Systems and Software*, 109, 161-176.
25. Zikria, Y.B.; Ishmanov, F.; Afzal, M.K.; Kim, S.W.; Nam, S.Y.; and Yu, H. (2018). Opportunistic channel selection MAC protocol for cognitive radio ad hoc sensor networks in the internet of things. *Sustainable Computing: Informatics and Systems*, 18, 112-120.
26. Wu, C.-M.; and Lo, C.-P. (2017). Distributed MAC protocol for multichannel cognitive radio ad hoc networks based on power control. *Computer Communications*, 104, 145-158.
27. Garg, V. (2010). *Wireless communications and networking*. California: Morgan Kaufmann.

28. Ali, O.; Ishak, M.K.; and Bhatti, M.K.L. (2021). Adaptive clear channel assessment (A-CCA): Energy efficient method to improve wireless sensor networks (WSNs) operations. *AEU - International Journal of Electronics and Communications*, 131.
29. Nasab, M.A.; Shamshirband, S.; Chronopoulos, A.T.; Mosavi, A.; and Nabipour, N. (2020). Energy-efficient method for wireless sensor networks low-power radio operation in internet of things. *Electronics*, 9(2).
30. Ko, J.G.; Tsiftes, N.; Dunkels, A.; and Terzis, A. (2012). Pragmatic low-power interoperability: ContikiMAC vs TinyOS LPL. *Proceedings of the 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*. Seoul, Korea, 94-96.
31. Nasseri, M.; Al-Olimat, H.; Alam, M.; Kim, J.; Green, R.; and Cheng, W. (2015). Contiki Cooja simulation for time bounded localization in wireless sensor network. *Proceedings of the 18th Symposium on Communications and Networking*. Alexandria, USA, 1-7.
32. Texas, I. (2007). 2.4 GHz IEEE 802.15.4/ZigBee-ready RF transceiver. Retrieved February 12, 2020 from https://www.ti.com/lit/ds/symlink/cc2420.pdf?ts=1617425858670.
33. Uwase, M.-P.; Bezunartea, M.; Tiberghien, J.; Dricot, J.-M.; and Steenhaut, K.J. (2017). Experimental comparison of radio duty cycling protocols for wireless sensor networks. *IEEE Sensors Journal*, 17(19), 6474-6482.
34. Dunkels, A.; Eriksson, J.; Finne, N.; Österlind, F.; Tsiftes, N.; Abeillé, J.; and Durvy, M. (2012). Low-power IPv6 for the internet of things. *Proceedings of the Ninth International Conference on Networked Sensing*. Antwerp, Belgium, 1-6.
35. Kim, J.U.; Kang, M.J.; Yi, J.M.; and Noh, D.K. (2015). A simple but accurate estimation of residual energy for reliable WSN applications. *International Journal of Distributed Sensor Networks*, 11(8).
36. Li, J.; Zhou, H.Y.; Zuo, D.C.; Hou, K.M.; Xie, H.P.; and Zhou, P. (2014). Energy consumption evaluation for wireless sensor network nodes based on queuing petri net. *International Journal of Distributed Sensor Networks*, 10(4).
37. Kerasiotis, F.; Prayati, A.; Antonopoulos, C.; Koulamas, C.; and Papadopoulos, G. (2010). Battery lifetime prediction model for a WSN platform. *Proceedings of the Fourth International Conference on Sensor Technologies and Applications*. Venice, Italy, 525-530.