İSTANBUL UNIVERSITY CERRAHPAŞA

RESEARCH ARTICLE

Electrica 2020; 20(1): 62-70

Power Consumption Analysis of a Wi-Fi-based IoT Device

Mehmet Erkan Yüksel

Department of Computer Engineering, Mehmet Akif Ersoy University, Faculty of Engineering and Architecture, Burdur, Turkey

Cite this article as: Yüksel ME. Power Consumption Analysis of a Wi-Fi-based IoT Device. *Electrica*, 2020; 20(1): 62-70.

ABSTRACT

The Internet of Things is a dynamic global network infrastructure consisting of interconnected smart devices with identification, sensing, data processing, and communication capabilities. It provides an intelligent ecosystem for the information society by enabling advanced services, standards, and platforms. Due to the growing market for smart devices with IP connectivity, several companies introduced low power Wi-Fi products optimized for IoT applications. Because Wi-Fi has established itself as one of the most popular wireless network technologies offering connectivity, has brought many advantages for IP enabled IoT devices (e.g., high data rate, mobility, built-in IP-network compatibility, easy integration with existing infrastructure), and has generated momentum in the IoT industry. However, since Wi-Fi was originally developed for high bandwidth applications targeting the consumer electronics market, it was not considered as a feasible technology for IoT applications. Wi-Fi-based IoT devices are typically battery-operated. Their wireless communication modules consume a relatively high amount of energy in case data needs to be sent a long distance, thus battery lifetime requirement for these devices remains a primary concern. Such small devices should transmit at high efficiency to conserve battery power, and they are required to sustain reliable operation for years on batteries even in the presence of heavy interference. Considering the limitations of battery power and long operational lifetime, the development of energy-efficient systems for these devices is an important issue. In this study, we analyzed the power consumption of a Wi-Fi IoT device deployed in field settings, where power infrastructure is inaccessible. We investigated how the device's Wi-Fi module influences the power consumption in the IoT environment. In our experimental results, we observed that Wi-Fi-based IoT devices are still power-hungry and can operate well with low power consumption by using energy-optimized power modes..

Keywords: Internet of things, wireless LAN, low -power Wi-Fi, system-on-chip, energy consumption, battery life

Introduction

The Internet of Things (IoT) is a rapidly developing technology that has a wide range of application domains, technological innovations, and social impacts. It offers an intelligent ecosystem with infinite opportunities in which embedded computing systems interact with each other and the Internet. It is a paramount technology for the information society and has led to a paradigm shift in many research areas [1].

Wireless communication technologies have attracted a great deal of attention in recent years, and therefore, the design and implementation of energy-efficient IoT systems have become more critical. IoT devices used must operate well in the IoT environment without the possibility of periodic maintenance. The replacement of these devices can be time-consuming, complicated, and expensive, even when the device itself is of low cost [1]. In addition to achieving their primary goals, they must provide extra functionalities such as energy-efficient operation, network compatibility, self-powered, self-diagnostics, and self-sustained. Moreover, they need to be extremely secure, robust, and reliable. In order to fulfill all mentioned requirements, they need to possess a certain level of intelligence along with a small form factor [1].

One of the consequences of the IoT is the creation of a wireless network that allows the interconnection of geographically distributed smart devices employed for sensing and monitoring physical phenomena to be able to connect between them and easily send their data to the related systems. The primary problem is how to obtain information from these devices for

Corresponding Author:
Mehmet Erkan Yüksel

E-mail:
erkanyuksel@mehmetakif.edu.tr

Received: 19.11.2019

Accepted: 13.01.2020

DOI: 10.5152/electrica.2020.19081

Content of this journal is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

the region in which they are deployed. Will they store the data in a memory unit, and somebody has to go and copy/transfer the data periodically to another device? How will these devices send the data? The main benefit of wireless transmission is that there is no required to install cables; hence, the devices will be able to be deployed anywhere. Wireless fidelity (Wi-Fi) is a desirable option and one of the most popular wireless network technologies for smart devices as it provides fast deployment, flexibility, and mobility support. However, being wireless also means that it has to rely on a battery to sustain itself [2, 3].

Wi-Fi is usually considered as a power-hungry wireless network technology and has not been widely adopted in the IoT era. The deployment, setup, and use of Wi-Fi-based IoT devices are also challenging tasks in IoT applications that have different requirements. The environment where IoT devices are deployed in order to monitor environmental events is highly dynamic. Therefore, power consumption influences the operation of Wi-Fi IoT devices significantly and imposes on the IoT system different restrictions. As a result of the power management policy, operation lifetime is one of the critical issues in all IoT applications.

Several IoT applications (e.g., smart cities, smart grid, home and building automation, industrial control, precision agriculture, and environmental monitoring) can make use of a constant power supply provided by energy harvesting systems in which their usage is still marginal. Although it is possible to use an autonomous power supply from alternative energy sources in some IoT environments, IoT devices tend to be battery-powered in order to keep their operations non-intrusive. However, in most cases, batteries are not expected to be recharged or replaced with new ones. Therefore, energy should be conserved. There are various methods/techniques to do so, both in hardware and software. From the hardware perspective, it is essential to select the system hardware components carefully. These components must feature low power consumption while providing the required capacity. From the software perspective, energy can be saved by controlling the data to be sent (i.e., packet size, number of packets) and the transmitting power. Radio activities (i.e., transmitting and receiving data) in IoT applications consume much energy compared with sensing and data processing activities. In this context, it is essential to control the amount of data to send and the frequency at which it is sent while maintaining the quality of service (QoS) required by the IoT application. Moreover, the further the data are sent, the more the energy needed, and the more the interferences produced. Therefore, it is also important to control the transmission range based on the destination to reach [4, 5].

In this study, we investigated the feasibility of Wi-Fi for battery-powered IoT devices. We focused on the performance concerning power consumption, interface type used, connectivity, and communication range. The remainder of the paper is organized as follows: Section 2 presents the related work. Section 3 describes the material and method for our tests and measurements. Section 4 presents an experimental study to analyze the power consumption of the Wi-Fi IoT device. Section 5 discusses the research findings. Finally, we conclude the article.

Related Work

Various studies discuss the technologies, standards, characteristics, and applications of IoT [6-10]. However, these studies did not cover the recent wireless technologies and their suitability for IoT applications. Moreover, with an ever-increasing demand for new IoT applications, energy consumption has been the primary concern for the IoT devices. With the limited energy storage, an IoT device's battery will deplete eventually, and replacing the battery manually/periodically will require great human efforts. In this context, several low power wireless technologies such as Bluetooth Low Energy (BLE), ZigBee, IEEE 802.15.4 with 6LoWPAN, IEEE 802.11ah (Wi-Fi HaLow), LoRa, and Sigfox have been developed for IoT applications. Besides, numerous approaches have been proposed in the literature in order to operate IoT systems energy-efficient and extend network lifetime [11-13]. However, since Wi-Fi technology is usually considered not to be efficient in terms of power usage, not much research has been done on Wi-Fi-based IoT devices.

In [14], Tsao and Huang presented a study including the energy consumption problems and energy-efficient technologies of the MAC protocol in Wi-Fi WLAN. In [15], Thomas et al. focused on the power usage of the most common protocols, Wi-Fi and 433MHz, and they discussed the power aspects of using each protocol in an IoT environment with experiments performed by a developed board. In [16], Thomas et al. presented an extension of their previous study [15]. They analyzed a Wi-Fi-based wireless sensor's power consumption, with and without an external microcontroller optimized for low power operation, which can be used to turn the Wi-Fi module on and off. They evaluated whether it is possible to reduce the Wi-Fi power consumption to the point where Wi-Fi can be employed instead of other wireless technologies. In [17], Feeney and Nilsson conducted several experiments by measuring the per-packet energy consumption of a Wi-Fi device working in ad hoc mode. The authors stated that energy consumption should be considered as a network layer issue, as well as MAC layer one. In [18], Muller and Rust investigated the energy consumption results of five low power Wi-Fi modules that are commonly available for embedded applications. They implemented two different use cases on all modules and evaluated them regarding energy consumption. They provided data to demonstrate the involved trade-offs and facilitate the selection of a suitable device. They also highlighted several essential aspects of low power Wi-Fi embedded system design. In [19], Kellogg et al. introduced a passive Wi-Fi system that demonstrates the device can generate Wi-Fi transmissions using backscatter communication. They designed a network stack for the passive Wi-Fi transmitters to coexist with other devices in the ISM band. They built a prototype backscatter hardware and implemented all four Wi-Fi bit rates on an FPGA platform. They designed a passive Wi-Fi IC that performs 1Mbps and 11Mbps Wi-Fi transmissions. In [20] Folea and Ghercioiu introduced the first ultra-low power Wi-Fi SoC named G2C501. They built a battery-operated, low power Wi-Fi

device to measure temperature, humidity, light, and vibration or motion. They described the device hardware and firmware components, power consumption and power management, application software, and gave a performance analysis. In [21], Lu et al. considered an alternative approach to the traditional on/off models. They explored a method that reduces the Wi-Fi chipset's power consumption across all of its operating states. Their approach uses the excess channel capacity provided by Wi-Fi networks when compared to the bandwidth demands of smartphone applications. Leveraging the inherent sparsity in direct-sequence spread-spectrum (DSSS) modulation, they proposed a transceiver design based on compressive sensing that allows Wi-Fi devices to operate their radios at lower clock rates when receiving and transmitting at low bit rates. They implemented their 802.11b-based design in a software radio platform and showed that it seamlessly interacts with existing Wi-Fi deployments. In [22], Morin et al. presented a comparison of the expected lifetime for IoT devices operating in several wireless networks: the beacon-enabled IEEE 802.15.4, time-slotted channel hopping variant of IEEE 802.15.4e, IEEE 802.11 PSM, IEEE 802.11ah, BLE, LoRa, and SIGFOX. They developed a Python-based analyzer that computes the energy consumption and lifetime of the wireless protocols based on the power required in a given state and the duration of each state. They discussed the suitability of the related wireless technologies for IoT application requirements and studied the feasibility of energy harvesting solutions. In [23], Tozlu et al. studied the feasibility of low power Wi-Fi to enable the IP connectivity of battery-operated devices by employing commercially available chips. They evaluated the power consumption of Wi-Fi devices by considering three sensor applications, investigated the interference impact, and measured the communication range performance. In [24], Trasvina-Moreno et al. investigated the energy consumption of a Wi-Fi-based wireless sensor device and its feasibility in an urban environment. They explained energy budgeting strategies and presented a theoretical analysis to evaluate the device's energy consumption in different operating modes.

Material and Method

We employed an IoT device (FiPy, Pycom Ltd.) [25] in order to analyze only the power consumption of its Wi-Fi module. It is a small-size, low-cost, and multi-functional high-end embedded system. It includes an SoC microcontroller (MCU); a multi-sensing platform consisting of a temperature sensor, a humidity sensor, a barometric pressure sensor, an ambient light sensor, and a three-axis 12-bit accelerometer; multiple wireless communication technologies which are BT/BLE, Wi-Fi, LoRa, Sigfox, cellular LTE CAT M1/NB1; and various peripheral interfaces than can be used in different IoT applications. The IoT device is based on Linux and uses an embedded micro-operating system that supports MicroPython and FreeRTOS. It can perform as an end-device, a wireless router, a gateway, or a local data collection center in the IoT environment. Figure 1 depicts the hardware architecture of the IoT device. Besides, we used the ESP32 module, ESP32-DevKitC development board, Asus DSL-AC88U wireless router, and Arduino Nano in our tests. On the server-side, there is a Java application software running on MacBook Pro with macOS Catalina.

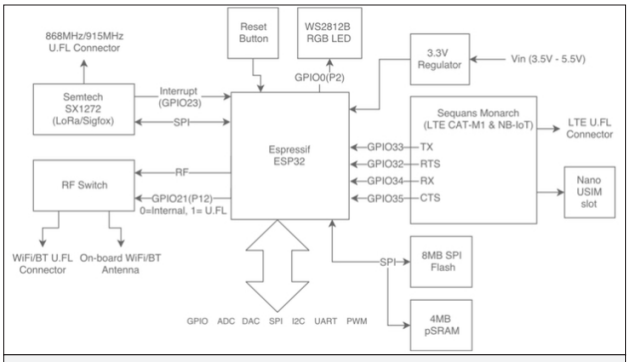


Figure 1. System block diagram of the IoT device [25]

Our IoT device has an ESP32 SoC MCU (Espressif Systems Co., Ltd.) [26] that can function as a complete standalone system or as a slave device to a host MCU, reducing communication stack overhead on the main application processor. ESP32 employs Tensilica Xtensa dual-core 32-bit LX6 MCU. It can interface with other systems to provide Wi-Fi and BLE functionality via its SPI, I2C, I2S, SDIO, or UART. It can be programmed by using AT commands and native code. Consequently, the IoT device has all the advantages required to convert it into a Wi-Fi IoT device by appropriate coding loaded into the MCU (e.g., users can utilize the ESP32 for their application programs). Figure 2 shows the ESP32 hardware architecture.

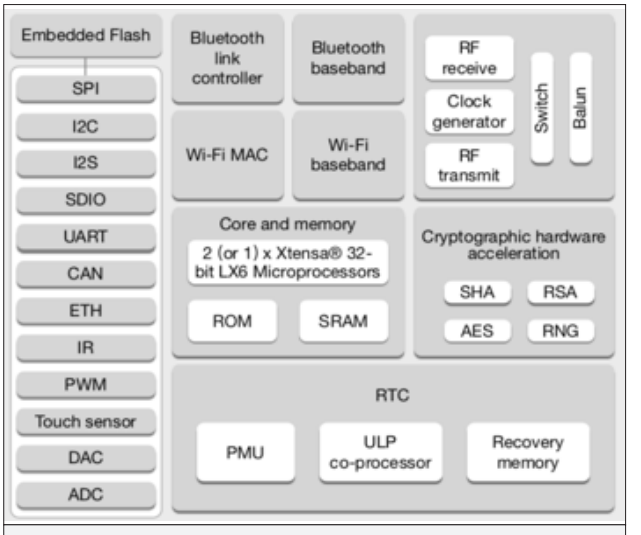


Figure 2. ESP32 hardware architecture [26]

For the Wi-Fi-based IoT devices with various hardware components, the interface type in their system architectures and the available interface speed influence the overall energy consumption significantly. Therefore, it cannot always be possible

to compare the Wi-Fi energy consumption between these devices directly [18].

We can evaluate the Wi-Fi IoT devices depending on the interface type. Wi-Fi modules can be used in a hostless mode (without any external MCU), or a hosted mode where they connect to an external MCU employing I2C, SPI, SDIO, or UART interfaces. In hostless mode, the Wi-Fi module runs the full Wi-Fi and TCP/IP stack. In hosted mode, the Wi-Fi module provides AT command library for all Wi-Fi related functions that can be easily accessed by the external/host MCU over the standard serial interface. Thus offers an easy way for users to add Wi-Fi to their embedded system designs without having to master the complexities of wireless communication protocols.

Host interface: As illustrated in Figure 3, an external MCU employs a serial interface such as SPI and UART in order to directly read/write from/to registers on the Wi-Fi module. The register interface is used to manage the Wi-Fi transceiver (e.g., sending and receiving data, configuring and controlling the Wi-Fi). Example devices are TI CC3100 [27] and Microchip ATWINC1500 [28].

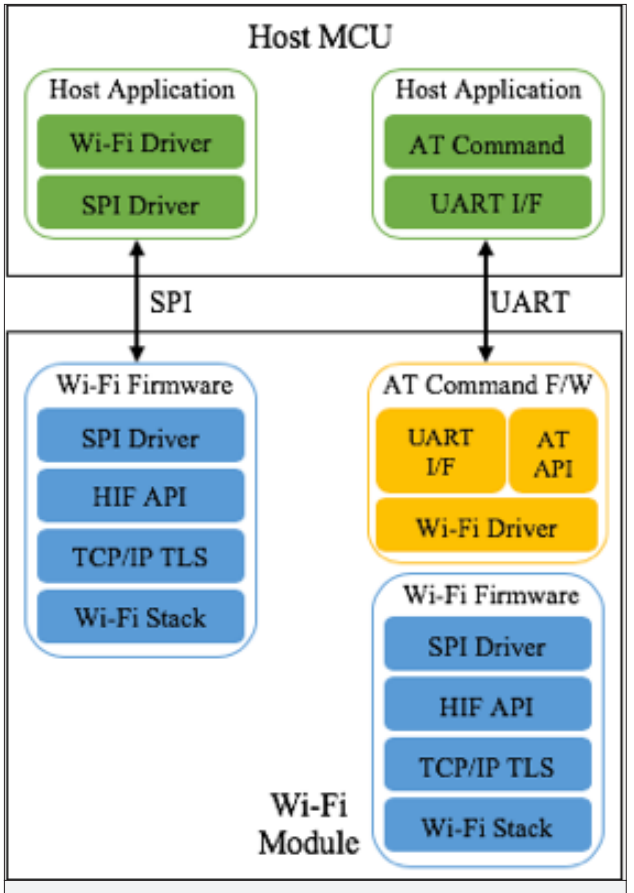


Figure 3. Host interface on external MCU (using drivers)

UART interface: The external MCU connects to the Wi-Fi module through a UART and runs AT commands to control the Wi-Fi transceiver. Figure 4 illustrates the serial communication via UART. Using AT commands via UART to send and receive data can cause overhead and longer transmission times. Representative of this interface type is Microchip RN171 [29], Telit GS2200 [30], and Espressif ESP8266 [31].

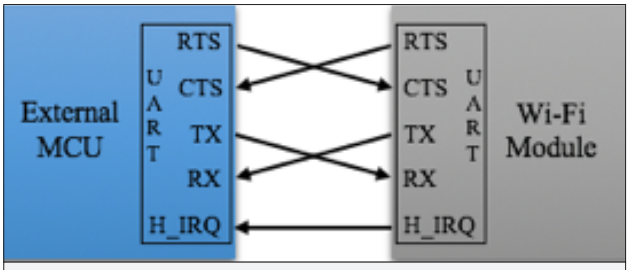


Figure 4. UART interface (AT commands via UART)

Wireless SoC (Wi-Fi MCU): It includes an embedded Wi-Fi module and allows the user to load the application program onto the internal processor that runs the Wi-Fi and TCP/IP stack (Figure 5). The user can store and run the application software in the MCU without the aid of any external processor. Thus provides to build systems where only a single CPU is required. Exemplary devices in this category are TI CC3235 [32], Telit WL865E4-P [33], and Espressif ESP32.

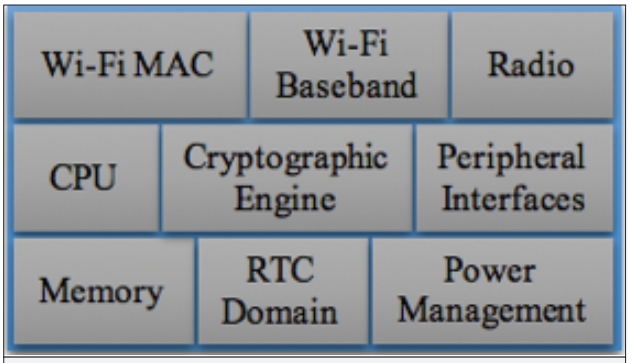


Figure 5. Wi-Fi MCU (application program on the CPU)

Experimental Study

The network we designed consists of Wi-Fi IoT devices connected to the Internet through a similar IoT device or a wireless router. Since the devices are connected to the router, we consider that the communication between them may also be possible. Figure 6 shows the device operation. In the start-up phase, after the device is powered up, it activates its Wi-Fi module and tries to associate itself with the router. It then gets its IP address through DHCP. We take into account the possibility that the device may lose its connection to the router for any reason. Therefore, we assume that the start-up process of the device occurs once in a certain period. Finally, the device sends its data to the HTTP server (a data collection center) period-

ically or when an event occurs. In these cases, an interrupt is triggered based on a timer. After the start-up phase, the device wakes up periodically to send keep-alive messages to the router to prevent disconnection.

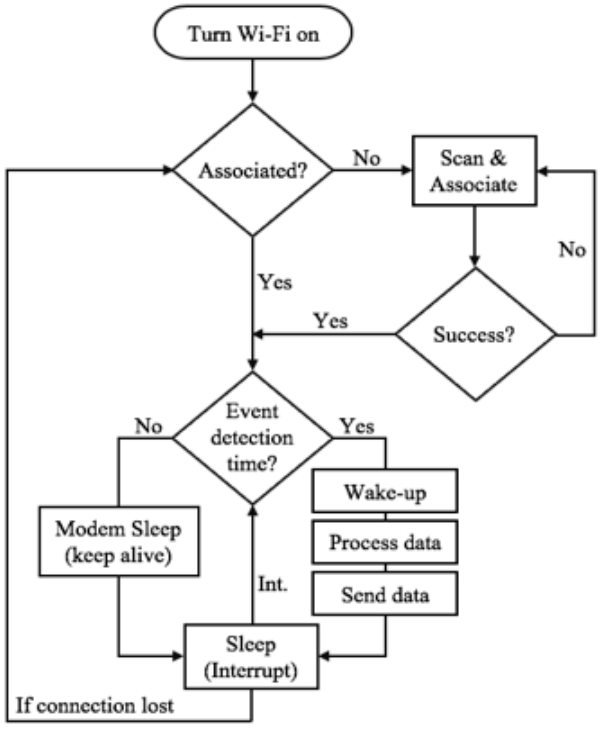


Figure 6. Operation of the Wi-Fi IoT device

The power consumption of a Wi-Fi-enabled IoT device can vary depending on the device's hardware features and IoT application requirements. Moreover, the length of the communication sequence, data-rate, packet-size (the amount of transmitted or received data), security scheme, transport layer protocol, MAC-layer retransmissions, and application program size can influence the power consumption. In this study, we defined two distinct scenarios to analyze the power consumption of the Wi-Fi-based IoT device: upload and download. Both scenarios include an end-device, a wireless router (gateway), and a server on the Internet (HTTP server). We performed our tests by making use of the techniques presented in [18] and [23]. In order to minimize the potential changes caused by external factors, the following conditions were taken into account and applied for our tests.

- Wi-Fi IoT device is the only device currently associated with a particular router.
- The device can be employed indoors or outdoors. In both cases, the distance between the device and the router is constant.
- WPA2-PSK (AES) security scheme is used for all data transmissions.

Upload scenario: Wi-Fi IoT device acts as a client and uploads sensor data to the HTTP server. Once the sensor data is available, the device turns on its Wi-Fi module, associates itself with the router, connects to the server, and then uploads the measurement data to the server. When the uploading process finishes, the device disconnects from the server and turns off its Wi-Fi module. Figure 7 illustrates the upload scenario.

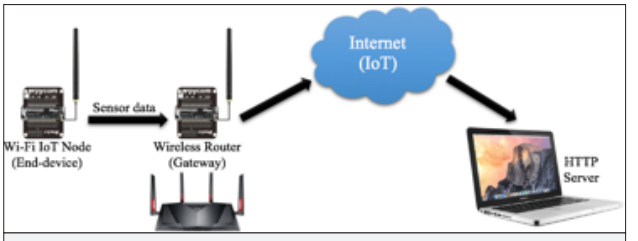


Figure 7. Uploading data to the server

Download scenario: Wi-Fi IoT device turns on the Wi-Fi module, associates itself with the wireless router, and starts the download process by transmitting its ID to the HTTP server. Then, the server sends the related data to the device. After the device receives the data, it disconnects from the server and turns off its Wi-Fi module. Unlike the upload process, in this scenario, the device can download a large amount of data from the server. Typical applications can be downloading a file, a firmware image, or current weather information to be displayed. Figure 8 illustrates the download scenario.

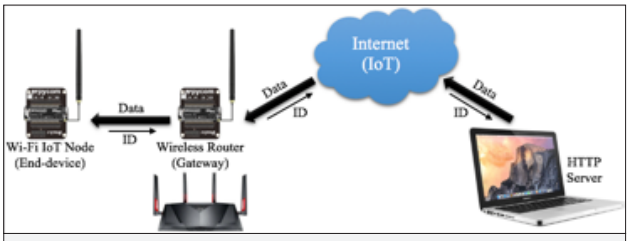


Figure 8. Downloading data from the server

We measured the current drawn at different power modes (Table 1). We applied TCP/IP and UDP/IP traffic at different data rates with various packet sizes for the transmit state measurements. The power consumption measurements are taken with a 3.3V supply at an ambient temperature of 25°C. Each measurement was repeated ten times and averaged to minimize the impacts of the factors influencing the device on the results. Although all wireless network devices such as routers, gateways, and access points are standards-based, the interaction between these devices and IoT clients may not be identical. The difference in activity may influence their power consumption profiles when Wi-Fi IoT devices are tested with different wireless network device models and settings.

Table 1. Power consumption measurements

Power mode	Power consumpiton
Active	Wi-Fi TX (+14dBm): 180mA Wi-Fi client: 130mA Wi-Fi AP: 120mA Wi-Fi RX and listening: 97mA
Modem sleep	36mA (high speed: 240MHz) 30mA (at 160MHz) 24mA (normal speed: 80MHz)
Light sleep	0.8 mA
Deep sleep	0.17mA (ULP co-processor is active) 0.12mA (ULP sensor-monitored pattern) 0.025mA (RTC timer + RTC memory)
Hibernate	5 µA
Idle (no radios)	60 mA

We considered a simple upload scenario where the Wi-Fi IoT device wakes up periodically, senses its surroundings, sends sensor data to the server, and goes back to sleep. Figure 9 shows the dynamic power consumption of the device for such an event of transmitting 260 bytes of data. In our tests, we observed that the device has a rather energy-intensive start-up phase when employing the Wi-Fi module. The peak currents measured are around 247mA at 3.3V.

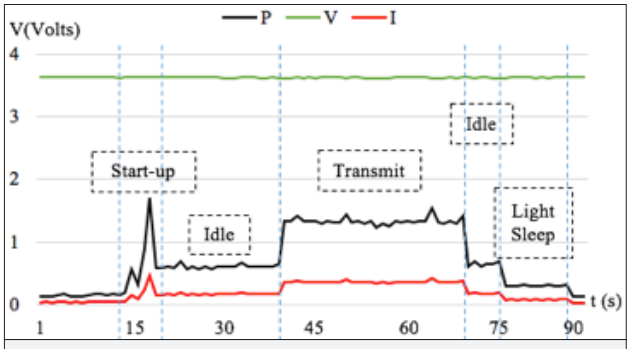


Figure 9. Power consumption in upload process

Figure 10 depicts the dynamic power consumption of the device for the download scenario.

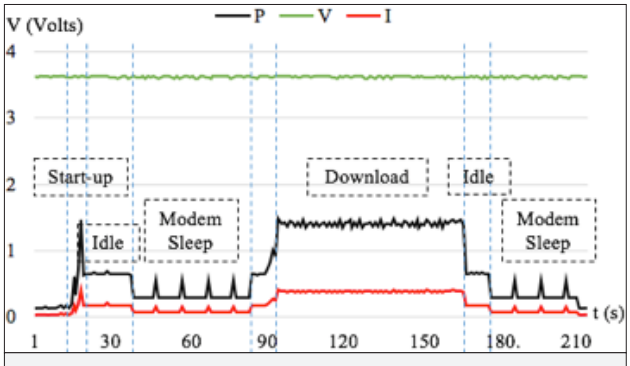


Figure 10. Power consumption in download process

Figure 11 presents the energy consumption status of the device for different packet sizes and data rates in the upload scenario. Depending on the application requirements, the device puts itself into different sleep modes between periodic short operations. It spends a small amount of time to transmit or receive data. During the wake-up process, it activates the hardware components, stabilizes the regulators, initializes the operating system, and loads the program from flash memory into RAM. Therefore, the majority of time and energy goes to the wake-up process and application software operations.

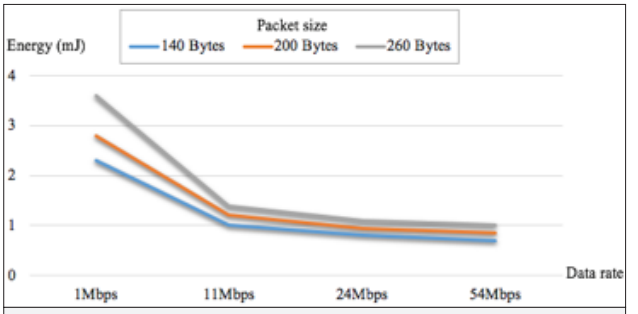


Figure 11. Effect of packet size and data rate on energy consumption

Findings

Many Wi-Fi-enabled IoT devices spend most of their time in different low power modes (sleep states) in order to achieve energy-efficient operations. These devices only need to transmit data at specific time intervals. The efficiency in turning off the device's Wi-Fi module and reconnecting when data needs to be transmitted, or the efficiency in remaining associated and connected in adjusted low power mode depends on the time interval between two transmissions. In our scenarios, we observed that the association of a Wi-Fi IoT device with a wireless router requires a relatively high amount of energy. However, our device has a variety of low power modes where it can stay associated with the router and connected to the network (keeps its assigned IP address). For example, the ESP32 SoC module in the IoT device provides five configurable low power modes: active, modem sleep, light sleep, deep sleep, and hibernation. Table 2 presents the measurement results obtained from the device in the upload scenario. We can state that although the energy for each connection is independent of the transmission interval, the energy to stay connected directly depends on the transmission interval.

We can say that the type of programming interface employed has little impact on energy consumption [18]. In order to evaluate this claim, the two scenarios (upload and download) were both used in two different ways on the Wi-Fi module: using native code on the internal CPU (Wi-Fi SoC) and using AT commands on the external MCU. Table 3 shows the measurement results. Sending AT commands from a host MCU via a serial interface takes a little more time compared to the native code. The native code consumes 8.5% and 4.3% less energy for the

Table 2. Energy consumption of the Wi-Fi IoT device

Transmission Period (mins)	Connecting and transmitting data (100 bytes)	Remaining connected and transmitting data (100 bytes)
1	1685 mJ	893 mJ
3	1685 mJ	1956 mJ
5	1685 mJ	2437 mJ
10	1685 mJ	5184 mJ
15	1685 mJ	6270 mJ

upload and download scenarios, respectively. Therefore, the use of native code on the Wi-Fi SoC allows a small reduction in energy consumption. On the other hand, in the case of using AT commands, the energy consumption of the host MCU also has to be taken into account (it leads to additional energy consumption). As a result, we can state that the native code solution is more favorable than AT commands.

Table 3. AT commands vs. native code

	AT Commands		Native Code	
Scenario	energy (mJ)	time (s)	energy (mJ)	time (s)
Upload (100 bytes)	1842	10.87	1685	10.32
Download (128 KB)	2456	13.45	2350	12.75

Using UDP protocol instead of TCP protocol helps to reduce energy consumption. Because of its design features, TCP (connection-oriented) has a higher overhead than the UDP (connectionless) in order to establish the connection [18]. Table 4 presents the measurement results for the upload scenario. In this scenario, our Wi-Fi IoT device wakes up periodically and sends 200 bytes of data. We assume all data packets are transmitted successfully without any need for retransmissions. WPA2 is used as a security mechanism. UDP reduces the overall energy consumption by 14% in our IoT device compared to TCP. Although UDP significantly reduces energy consumption, it can not provide an extremely effective solution. Our IoT device has rather small receive buffer sizes compared to the high throughput capacity on the server. Since UDP does not provide flow control, it can be possible that the server overloads the low power IoT device. Therefore, data packets may be lost.

Table 4. TCP vs. UDP

	TCP		UDP	
Scenario	energy (mJ)	time (s)	energy (mJ)	Time (s)
Upload	2476	10.58	2130	9.46

In our study, the energy measurements were carried out by employing both a wireless router and another Wi-Fi IoT device without any load. Since our device was the only client connected through the router, similar measurement results occur. In real-world applications, the load on the router and the server can have a significant impact on the connection time and the required energy. In this context, we performed a simple case study using the Wi-Fi IoT device in an indoor environment with a typical network load. The device connects to the HTTP server, sends its ID, and receives data from the server. In this situation, we evaluated data transfers and observed that the network load significantly influences the connected time and required energy.

The use of different low power modes (sleep states) in a WiFi IoT device may not always reduce power consumption effectively. For example, wake-up time from the deep sleep mode or hibernation mode for our device can slow down data transmission, influence the connection time, and increase energy consumption, depending on the application requirements. Therefore, sometimes, it may be the right solution not to use low power modes for WiFi IoT devices.

There are two different methods for obtaining an IP address in IP-based systems: dynamic IP address allocation and static IP address allocation. The dynamic IP address allocation employs a DHCP server that automatically assigns an IP address every time a device accesses the network. It can consume a significant amount of energy due to the energy attributed to the DHCP exchange involved. On the other hand, the static IP address allocation is achieved by manually configuring the IP address that the system should use statically/permanently. It is relevant for systems where the control over the IP network topology is possible. It is favorable from the energy consumption perspective.

Wi-Fi networks usually use the dynamic IP address allocation for ease of management (because DHCP is a very flexible protocol) [16]. In this context, we analyzed the power consumption in the case of using static and dynamic IP address allocation on the Wi-Fi IoT device. In our scenarios, the device's Wi-Fi module is turned off to save power after the data transmission is completed, so an IP address must be allocated whenever the Wi-Fi module is turned on. In our experiments, we observed that there is a considerable amount of reduction in power consumption when using the static IP address allocation compared to DHCP, and the DHCP exchange influences battery life when data transmissions are short. However, it can be possible to reduce the IoT device's power consumption by using long DHCP leases.

Conclusion

Ever advancement of wireless communications, digital electronics, integrated circuits, sensors, micro-electro-mechanical systems, embedded computing systems, and information technology has made the fast proliferation of smart devices into our daily life. Over the past decade, these devices have become mobile terminals of the Internet, and their ever-increasing capabilities have enabled the realization of various IoT applications based on the collaborative effort of a large num-

ber of smart devices. In the future, everything will be able to connect to the Internet for intelligent use and management. Therefore, both academy and industry address challenges facing the IoT era.

In this article, we analyzed the power consumption of a Wi-Fi IoT device used in the field of environmental monitoring. Two different scenarios for the device were performed and then investigated how its SoC MCU with integrated Wi-Fi consumes energy. The obtained results show involved trade-offs and facilitate the selection of most appropriate wireless network technology for IoT applications. Besides, we highlighted several essential aspects that have to be kept in mind during the design and implementation of a low power Wi-Fi IoT device. Reducing the power consumption of a Wi-Fi IoT device by leveraging the different sleep modes supported by the device can be the right solution. Using sleep modes is a valuable strategy for dramatically extending the battery life and operational lifetime of the Wi-Fi IoT device that does not need to be active all the time. Consequently, although Wi-Fi is still relatively power-hungry compared to other wireless network technologies in IoT applications such as BLE, ZigBee, Z-Wave, LoRa, and NB-IoT it can provide exciting opportunities for specific IoT applications.

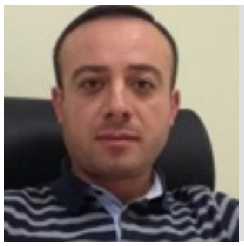
Peer-review: Externally peer-reviewed.

Conflict of Interest: The author have no conflicts of interest to declare.

Financial Disclosure: The author declared that the study has received no financial support.

References

1. M. E. Yuksel, "The design and implementation of a batteryless wireless embedded system for IoT applications", Electrica, vol. 19, no. 1, pp. 1-11, 2019. [CrossRef]
2. P. Silva, N. T. Almeida, R. Campos, "A comprehensive study on enterprise Wi-Fi access points power consumption", IEEE Access, vol. 7, pp. 96841-67, 2009. [CrossRef]
3. M. Bassoli, V. Bianchi, I. De Munari, P. Ciampolini, "An IoT approach for an AAL Wi-Fi-based monitoring system", IEEE Transactions on Instrumentation and Measurement, vol. 66, no. 12, pp. 3200-9, 2017. [CrossRef]
4. W. Wang, Y. Chen, L. Wang, Q. Zhang, "Sampleless Wi-Fi: bringing low power to Wi-Fi communications", IEEE/ACM Transactions on Networking, vol. 25, no. 3, pp. 1663-72, 2017. [CrossRef]
5. K. S. Low, M. C. R. Talampas, "Wireless sensor networks for intelligent transportation applications: A survey", In Industrial Wireless Sensor Networks: Applications, Protocols, and Standards, Boca Raton, FL, USA, CRC Press, 2013, pp. 47-77.
6. V. Tsiatsis, S. Karnouskos, J. Holler, D. Boyle, C. Mulligan, "Internet of Things: Technologies and Applications for a New Age of Intelligence", Academic Press, London, UK, 2019.
7. S. Greengard, "The Internet of Things", MIT Press, Cambridge, MA, USA, 2015. [CrossRef]
8. M. Alioto, "Enabling the Internet of Things: From Integrated Circuits to Integrated Systems", Springer, Cham, Switzerland, 2017. [CrossRef]
9. S. Li, L. Da Xu, S. Zhao, "The internet of things: a survey", Information Systems Frontiers, vol. 17, no. 2, pp. 243-59, 2015. [CrossRef]
10. P. Patel, D. Cassou, "Enabling high-level application development for the Internet of Things", Journal of Systems and Software, vol. 103, pp. 62-84, 2015. [CrossRef]
11. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, "Internet of things: a survey on enabling technologies, protocols, and applications", IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-76, 2015. [CrossRef]
12. Z. Abbas, W. Yoon, "A survey on energy conserving mechanisms for the internet of things: Wireless networking aspects", Sensors, vol. 15, no. 10, pp. 24818-47, 2015. [CrossRef]
13. K. Mekki, E. Bajic, F. Chaxel, F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment", ICT Express, vol. 5, no.1, pp. 1-7, 2019. [CrossRef]
14. S. L. Tsao, C.-H. Huang, "A survey of energy efficient MAC protocols for IEEE 802.11 WLAN", Computer Communications, vol. 34, no. 1, pp. 54-67, 2011. [CrossRef]
15. D. Thomas, R. McPherson, J. Irvine, "Power analysis of local transmission Technologies", In 12th Conference on Ph.D. Research in Microelectronics and Electronics, Lisbon, Portugual, 2016. [CrossRef]
16. D. Thomas, R. McPherson, G. Paul, J. Irvine, "Optimizing power consumption of Wi-Fi inbuild IoT device: An MSP430 processor and an ESP-03 chip provide a power-efficient solution", IEEE Consumer Electronics Magazine, vol. 5, no. 4, pp. 92-100, 2016. [CrossRef]
17. L. M. Feeney, M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment", In 20th Annual Joint Conference of the IEEE Computer and Communications Society (IEEE INFOCOM), Alaska, USA, 2001, pp. 1548-57.
18. A. D. Muller, A. Rust, "Low-power wireless: what is possible with WiFi?", In Wireless Congress: Systems & Applications, Munich, Germany, 2015.
19. B. Kellogg, V. Talla, S. Gollakota, J. R. Smith, "Passive Wi-Fi: bringing low power to Wi-Fi transmissions", In 13th Usenix Conference on Networked Systems Design and Implementation, Santa Clara, CA, USA, 2016, pp. 151-64.
20. S. Folea, M. Ghercioiu, "Ultra-low power Wi-Fi tag for wireless sensing", In IEEE International Conference on Automation, Quality and Testing, Robotics, Cluj-Napoca, Romania, 2008, pp. 247-52. [CrossRef]
21. F. Lu, G. M. Voelker, A. C. Snoeren, "SloMo: Downclocking WiFi communication", In 10th USENIX Symposium on Networked Systems Design and Implementation, Lombard, IL, USA, 2013, pp. 255-8.
22. E. Morin, M. Maman, R. Guizzetti, A. Duda, "Comparison of the device lifetime in wireless networks for the Internet of Things", IEEE Access, vol. 5, pp.7097-114, 2017. [CrossRef]
23. S. Tozlu, M. Senel, W. Mao, A. Keshavarzian, "Wi-Fi enabled sensors for internet of things: a practical approach", IEEE Communications Magazine, vol. 50, no. 6, pp. 134-43, 2012. [CrossRef]
24. C. A. Trasvina-Moreno, A. Asensio, R. Casas, R. Blasco, A. Marco, "WiFi sensor networks: a study of energy consumption", In IEEE 11th International Multi-Conference on Systems, Signals & Devices, Barcelona, Spain, 2014, pp. 1-6. [CrossRef]
25. FiPy, Available from: URL: https://pycom.io/product/fipy/.
26. Espressif ESP32 SoC, Available from: URL: https://www.espressif.com.
27. TI CC3100, Available from: URL: http://www.ti.com.
28. Microchip ATWINC1500, Available from: URL: https://www.microchip.com.
29. Microchip RN171, Available from: URL: https://www.microchip.com.
30. Telit GS2200, Available from: URL: https://www.telit.com.
31. Espressif ESP8266, Available from: URL: https://www.espressif.com.
32. TI CC3235, Available from: URL: http://www.ti.com.
33. Telit WL865E4-P, Available from: URL: https://www.telit.com.



Mehmet Erkan Yuksel received his B.S. degree in Computer Engineering from Firat University in 2007, his MSc and Ph.D. degrees in Computer Engineering from Istanbul University in 2010 and 2014, respectively. In 2015, he joined the Department of Computer Engineering at Burdur Mehmet Akif Ersoy University, where he is now an Assistant Professor. His research interest includes embedded systems, wireless network technologies, wireless sensor networks, software-defined networks, RFID, agent-based modeling and simulation, artificial intelligence, machine learning, and data mining.