Assessing the ESP8266 WiFi module for the Internet of Things

João Mesquita[‡], Diana Guimarães[*‡], Carlos Pereira[*‡], Frederico Santos[*§], Luis Almeida[*†‡]
[*]IT - Instituto de Telecomunicações
[†]CISTER - Centro de Investigação em Sistemas de Tempo-Real
[‡]FEUP - Universidade do Porto, Porto, Portugal
Email: {up201305568, dlguim, dee12014, lda}@fe.up.pt
[§] ISEC - Instituto Politécnico de Coimbra, Coimbra, Portugal
Email: fred@isec.pt

***Abstract*—The Internet of Things (IoT) is experiencing rapid growth and being adopted across multiple domains. For example, in industry it supports the connectivity needed to integrate smart machines, components and products in the ongoing Industry 4.0 trend. However, there is a myriad of communication technologies that complicate the needed integration, requiring gateways to connect to the Internet. Conversely, using IEEE 802.11 (WiFi) devices can connect to existing WiFi infrastructures directly and access the Internet with shorter communication delays and lower system cost. However, WiFi is energy consuming, impacting autonomy of the end devices. In this work we characterize a recent WiFi-enabled device, namely the ESP8266 module, that is low cost and branded as ultra-low-power, but whose performance for IoT applications is still undocumented. We explore the built-in sleep modes and we measure the impact of infrastructure parameters beacon interval and DTIM period on energy consumption, as well as packet delivery ratio and received signal strength as a function of distance and module antenna orientation to assert area coverage. The ESP8266 module showed suitability for battery powered IoT applications that allow 2-4 days recharge cycles on a 1000mAh battery with seconds-scale transmission intervals.**

I. Introduction

The number of devices with sensing and wireless communication capabilities has been constantly growing as a result of steady reductions in cost and size of computing hardware, communication interfaces and data storage. This trend is the basis for the strong growth of the *Internet of Things* (IoT) that we are witnessing nowadays across multiple domains, from Home Automation [14] to Health Monitoring [12] and Manufacturing Systems [11][6]. IoT uses Internet technology to allow devices sharing information openly and widely but also communicating directly with each other in sensing and actuation loops, i.e., the so-called Machine-to-Machine (M2M) paradigm. In industry we often find the expression *Industrial IoT* (IIoT) where *Things* refer to smart components, machines, products, etc.

The prevailing communication technology for Internet access of free moving devices is infrastructured WiFi (IEEE 802.11), particularly in urban environments. However, WiFi is typically considered a power hungry protocol and low-power devices that operate with significant autonomy rely on other technologies namely Bluetooth for personal devices, ZigBee for sensing, WirelessHART and ISA100 for industrial devices and GPRS, LTE, LoRA and Sigfox for wide-area sensing through communication operators.

In this work we revisit the use of WiFi considering recent ultra-low-power and low-cost devices, avoiding external communication operators and additional gateways, leveraging existing WiFi infrastructures and relying on WiFi and Internet security techniques. Using WiFi in manufacturing has the potential to reduce communication latency and costs and it has been recently considered either by communication solutions providers, e.g., DigitalAir, and by academic studies, e.g., on reliability [8]. Here we focus on energy consumption and area coverage, considering a use case in which smart WiFi tags are attached to relatively large products such as appliances or vehicles and allow tracking them along the production line, being removed at the end of production, recharged and re-applied at the input.

Similar studies on energy consumption can be found for both Bluetooth and IEEE 802.15.4-based protocols (ZigBee, WirelessHART and ISA100). For example, the energy consumption of the Bluetooth low energy interface in the Intel Edison module was studied in [16], standard and low energy Bluetooth were compared in [10], IEEE 802.15.4 in the TelosB motes was analysed in [13] and in the MICAz motes in [17].

Here we characterize in detail a recent WiFi-enabled device that is highly integrated and branded as ultra-low-power and low-cost, namely the ESP8266 module [1][2], which also offers a full TCP/IP stack.

This paper starts by presenting the features of the ESP8266 module and the mechanisms that allow it to lower energy consumption when performing recurrent seconds-scale communication. Then, we describe the experiments we carried out, we present the results and discuss them in the scope of the referred use case.

978-1-5386-7108-5/18/$31.00 ©2018 IEEE

II. The ESP8266 Ultra-Low-Power Module

Currently, there is a significant number of embedded platforms with WiFi connectivity. Popular ones include the Raspberry Pi, the BeagleBone, the Intel Edison module or the more recent ESP32 module. The first three aim at providing an inexpensive single-board computer and thus having relatively high computational power and consequently high power requirements. The Intel Edison module, however, has a smaller footprint and, similarly to the ESP modules, targets embedding in small devices. Unfortunately, it was discontinued in 2017. The ESP32 module [4] features a dual-core processor among other improvements over the ESP8266 module that also lead to an increased power consumption in operation. These platforms offer, beyond WiFi, short-range low-power wireless interfaces, making them suitable to implement gateways in IoT applications. For example, [6] uses a Raspberry Pi with Bluetooth low energy supporting 6LoWPAN and [9] reports the use of a Raspberry Pi combined with an Arduino board to interface to a ZigBee sensor network, while [7] focuses on reducing the power consumption of the Raspberry Pi as a gateway in a wireless sensor network application.

Generally, the existing platforms differ concerning additional wireless communication interfaces, from Bluetooth to ZigBee or even UWB, but also in physical footprint, computational power, which impacts energy consumption directly, and cost. Here we investigate the features of the ESP8266 module, which claims a good compromise among all these aspects [1].

A. ESP8266 module specifications

The ESP8266 has powerful features and simultaneously a reduced footprint. The modules functional block diagram can be seen in Figure 1.

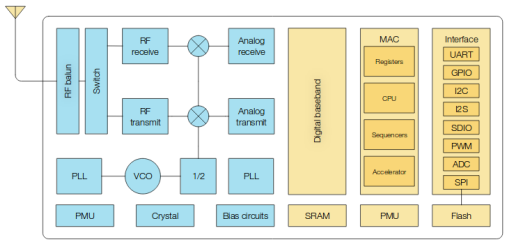

Fig. 1: ESP8266 functional block diagram [1].

The main features that make the ESP8266 particularly attractive for IoT applications are:

- standard IEEE 802.11 b/g/n (WiFi) compliance with on-board antenna;
- WPA/WPA2 security protocols;
- Tensilica L106 32-bit RISC processor with 160MHz maximum clock speed;
- 36kB internal SRAM and 4MB external SPI flash;
- several native sleep-modes;
- small form-factor (24x16 mm) making it embeddable in many devices;
- very low cost, particularly when compared to other WiFi modules.

The use of a 32-bit processor provides enough computing power to run the following features, which considerably simplify the development of IoT applications:

- a Real-Time Operating System;
- end-station, softAP or combination of both;
- a complete TCP/IP protocol stack.

Earlier experiments showed that these features and the module's computing power allow a direct integration in publisher-subscriber middlewares on which many IoT applications are being developed [12]. On the other hand, we conjecture that the whole set of features the ESP8266 module offers is barely the minimum to develop WiFi-enabled devices that are easy to program and integrate with the Internet.

B. Power-saving mechanisms in WiFi

To understand the effectiveness of the ESP8266 power-saving modes we must first revisit the power-saving mechanisms available in the WiFi protocol. A WiFi infrastructured network has two operation modes: active and power saving. The main idea behind the management of a network that contains stations in power saving mode, i.e., potentially sleeping, is that the access point (AP) will buffer the packets addressed for those stations until they wake-up [15].

For this purpose, the AP sends beacons regularly, which are management frames transmitted with a periodicity defined by configuration, namely the beacon interval, that advertise specific network information such as supported PHY rates, security protocols, supported Quality of Service and vendor specific information, plus the Traffic Indication Map (TIM). This last element informs whether the AP has unicast frames buffered for any associated station. When a station in power saving mode wakes up, it waits for the following beacon and checks the TIM to see whether it has any traffic pending in the AP and, if so, it polls the AP to retrieve it.

Broadcast/multicast frames are also buffered by the AP, which advertises them using a Delivery TIM (DTIM). This information element is also encoded in the beacon. Broadcast/multicast frames are transmitted by the AP immediately after the beacon carrying the DTIM. A DTIM is included in the beacon every configured DTIM period. All stations receiving a beacon with the DTIM indicating pending broadcast/multicast traffic should remain active to receive such traffic. Thus, the beacon interval and DTIM period determine for how long a station can sleep while interacting with the network, having a strong impact on the station power consumption.

C. ESP8266 power-saving modes

To reduce power consumption, the ESP8266 module offers three sleep modes: Modem-sleep, Light-sleep and Deep-sleep. Table I summarises the properties of these modes, which components they turn off, and the typical current consumption for the standalone module.

TABLE I: Sleep-modes - hardware level distinction and typical current consumption stated in the datasheet [1].

Item		Modem-sleep	Light-sleep	Deep-sleep
WiFi interface		OFF	OFF	OFF
AP association		Connected	Connected	Disconnec.
System clock		ON	OFF	OFF
RTC		ON	ON	ON
CPU		ON	Pending	OFF
Substrate current		15 mA	0.4 mA	∼20 uA
Aver. current	**DTIM=1**	16.2 mA	1.8 mA	-
	DTIM=3	15.4 mA	0.9 mA	-
	DTIM=10	15.2 mA	0.55 mA	-

Modem-sleep is the default sleep mode and it is recommended for applications that need real-time CPU control. In this mode, the WiFi circuit is shut-down while the other components remain on. Importantly, the WiFi association to the network Access Point (AP) is maintained, thus avoiding the need to re-connect upon waking up and the respective high latency. Light-sleep mode suits scenarios with less CPU requirements, in which the application can also suspend the CPU, but needs to communicate promptly. Thus the association to the AP is also maintained as with Modem-sleep. Moreover, Light-sleep shuts down the system clock, relying on a real-time clock (RTC) to keep track of time. Importantly, this mode suspends the CPU in DTIM periods in which there is no programmed task, only [2]. Otherwise, because the CPU will be requested in the meantime, it is kept active the whole interval, as with Modem-sleep (Figure 2).

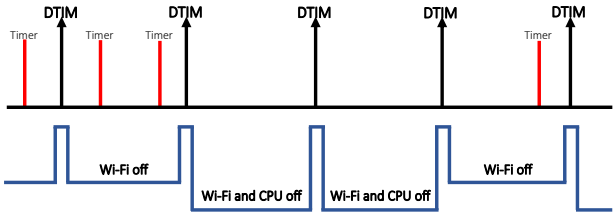

Fig. 2: Limitations of the Light-sleep mode with active tasks in the system (the blue curve below represents the current consumption variation) - adapted from [2].

Finally, Deep-sleep is used in very low-power sensor applications with data sampling / transmission periods in the minutes-scale or more. Unlike the other two modes, the module cannot get into this mode automatically and it must be forced by the user by explicit programming. In deep-sleep mode, the WiFi interface and CPU are both turned off, including the system clock, and the connection to the WiFi AP is lost and must be re-established upon waking up.

D. Communication range

Despite the importance of low power consumption, the communication is effective if within range, only, thus imposing a trade-off between communication range and transmission power. Studying this trade-off in detail is beyond the scope of this paper, but we are interested in assessing the practical communication range in low power indoor scenarios, as in a manufacturing plant.

The module features an on-board antenna which is particularly suited to be embedded in small devices. These antennas present a rather non-uniform radiation pattern, with a toroidal geometry revolving around the module and the axis perpendicular to the antenna longitudinal direction [5].

Therefore, it is also important to assess the sensitivity of the communication range with respect to the module orientation to see whether sustained communication can be achieved even when the module is moving freely.

III. Experiments

In this section we present the experiments that we carried out to perform a quantitative analysis of the ESP8266 module stand-alone performance with distinct scenarios and with different network configurations.

A. Setup

The experiments were performed indoor in a laboratory environment, with an ASUS RT-AC87U dual-band AC240030 access point, using the default IEEE 802.11 protocol. The ESP8266 module runs a FreeRTOS-based software framework [3] (version 1.2) that enables the use of any of the three reported sleep modes.

The current consumption in the different scenarios is measured with the module powered up at 3.35V directly by the Monsoon power monitor, from Monsoon Solutions Inc., with a sampling rate of 5000Hz. The current traces contain two fields, namely the current samples and the associated timestamps. Where convenient, the Y-axis in the figures is truncated for better visualization.

To better interpret the results we also perform packet captures in a monitoring computer running Ubuntu 16.04 LTS with the wireless adapter (Intel Dual Band Wireless-AC 7265) in monitor mode and we correlate these captures with the traces from the Monsoon power monitor.

In this setup we carry out the following experiments:

- Impact of the WiFi infrastructure on the module current consumption in Modem-sleep mode;

- Impact of the AP configuration (beacon interval and DTIM period) on the module current consumption in Modem-sleep and Light-sleep modes;
- Impact of data transmission in current consumption assuming a specific scenario and all sleep modes;
- Connectivity, received signal strength, packet delivery ratio and round-trip delay indoors.

B. Impact of the WiFi infrastructure

In this section we assess the current consumption of the ESP8266 module in Modem-sleep mode with the common AP configuration of 100 ms beacon interval and DTIM period of 3. This experiment was performed without transmitting any data and with no other station connected to the access point besides the module. The module was thus left in an idle state in which it would only respond to automatic infrastructure interactions via beacons. This approach allows observing the impact of the WiFi infrastructure without interference of actual traffic. Moreover, we use Modem-sleep mode, only, to maintain the AP association and the CPU switched on, eliminating variations from such sources, too.

Figure 3 shows a 5s trace of the module current consumption in Modem-sleep mode annotated with information collected from a simultaneous packet capture. As we observe, the wake-up of the module is synchronous with the DTIM message arrival and most of the times the module wakes up to receive the DTIM messages, only, and goes into sleep-mode immediately after. However, the current consumption pattern suggest that sometimes the module actually wakes up to transmit.

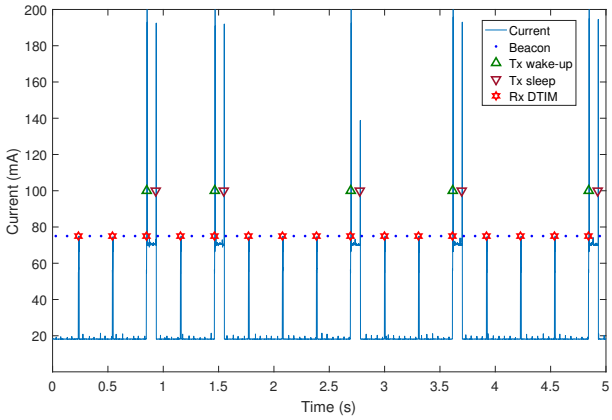

Fig. 3: Current consumption trace in Modem-sleep mode annotated with information from a simultaneous packet capture (100ms beacon interval and DTIM period of 3).

The two different wake-up current consumption patterns can be explained by different values in the TIM bitmap control. Figure 4 shows the situation in which the multicast bit is false indicating that the AP has no buffered multicast traffic for delivery. Thus, the module wakes up to receive and process the beacon frame, only,

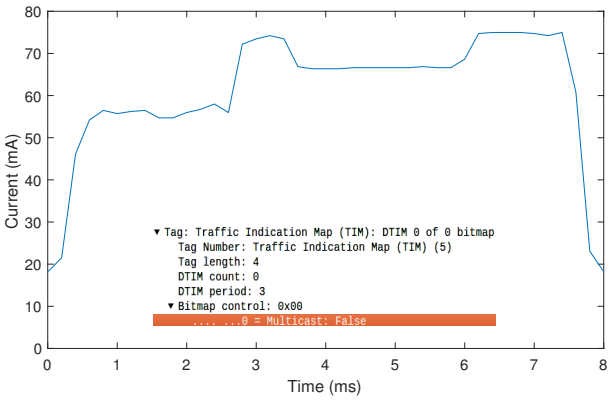

Fig. 4: Current consumption when there are no broadcast/multicast buffered frames at the AP.

and goes into sleep-mode immediately after. This process takes about 8ms and has an average current consumption of 61.4mA.

On the other hand, Figure 5 shows the case in which the multicast bit is true. In this case, the module has to wake-up to transmit a Null data frame with the power management bit set to 0 to inform the AP that it is awake and ready to receive the buffered frames. After that, the module transmits another Null data frame with the power management bit set to 1 to inform the AP that it is going to enter in sleep-mode again. The whole process takes approximately 90ms (1025% increase w.r.t DTIM listen only) and has an average current consumption of 71.4mA (16.3% increase w.r.t DTIM listen only).

These results show a potentially significant impact of the network load (traffic to be received) on the power consumption of the modules. In scenarios based on multicast transmission, the modules will be waking up frequently to receive those packets, reducing the efficiency of the sleep mode and increasing power consumption significantly.

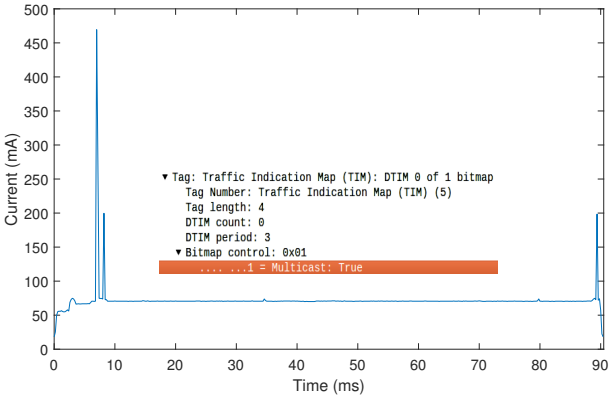

Fig. 5: Current consumption when there are broadcast/multicast buffered frames at the AP.

C. Impact of the AP configuration

Since the stations in power-saving mode associated to an AP wake up every x beacons to receive the TIM (DTIM) and see whether there is any traffic for them, including broadcast/multicast traffic, pending in the AP and retrieve it, the beacon interval and the DTIM period (x) impact the stations power consumption. These parameters are configured in the AP and thus we carried out a set of experiments to assess this impact on the ESP8266 modules power consumption. As in the previous experiment, to evaluate the impact of the beacon interval and DTIM period configurations, only, on the module power consumption, we measured the current consumption without any transmission of data for two modes of operation, namely Modem-sleep and Light-sleep. We did not consider normal active mode because it keeps the WiFi interface on all the time, neither Deep-sleep mode because it does not maintain wireless connectivity and thus AP beacons are ignored. In particular, we consider two different experiments for each of the referred modes:

- Beacon interval variation from 100 to 1000ms, in steps of 100ms, while maintaining the DTIM interval fixed at 3 (Fig. 6);
- DTIM period variation from 1 to 10, in steps of 1, while maintaining the beacon interval fixed at 100ms (Fig. 7).

Both experiments showed that, in Modem-sleep mode, the module can indeed reduce current consumption by setting the beacon interval or the DTIM period to larger values, taking advantage of the longer available time to maintain the WiFi interface off. Curiously, a different behaviour was observed in Light-sleep mode, with the current consumption increasing with larger beacon intervals or DTIM periods. Figure 6 also shows that, at a beacon interval of 700ms and higher the current consumption in Light-sleep converges to that of Modem-sleep. This effect was already explained in Section II, due to tasks that are activated during the DTIM period. In this case, the CPU is not suspended. Figure 8 shows current traces for the cases of 100ms (left) and 400ms (right) beacon interval, with DTIM period of 3 in Light-sleep mode. Both cases show the typical spikes at the DTIM occurrences, in some of which the module remains active for approximately 90ms to retrieve pending traffic, with an average current consumption of 71.4mA, and then there is a plateau at 18.2mA corresponding to switching off the WiFi interface but not the CPU (similar to Modem-sleep) and another plateau at 0.85mA corresponding to switching off both the WiFi interface and the CPU (95.3% reduction w.r.t Modem-sleep). However, longer intervals between DTIM indications increase the chance that some task, including operating system services, requires activation in that interval, preventing the CPU suspension. It is also more likely, with such longer intervals, that there will be pending traffic in the AP to be retrieved by the module when a DTIM comes.

D. Impact of data transmission

This set of experiments aimed at assessing the power consumption performance of the ESP8266 module in an IIoT use case as that referred in Section I, in which the module (smart tag) had to recurrently transmit a product data record. We consider a simplified record with 85B of data containing information about the product, timestamps and state of relevant production steps and other operational information. Then, we measured the current consumption for all four possible modes of operation: Active, Modem-sleep, Light-sleep and Deep-sleep modes. We do not consider any attached sensors, actuators or signalling devices to remove this common factor from the measurements. To assess the impact of the transmission frequency on the power consumption, we use two different cases, namely the transmission of 1 record (85B) every 1s and the transmission of 10 records together (850B) every 10s. These two cases are repeated for all execution modes with the exception of

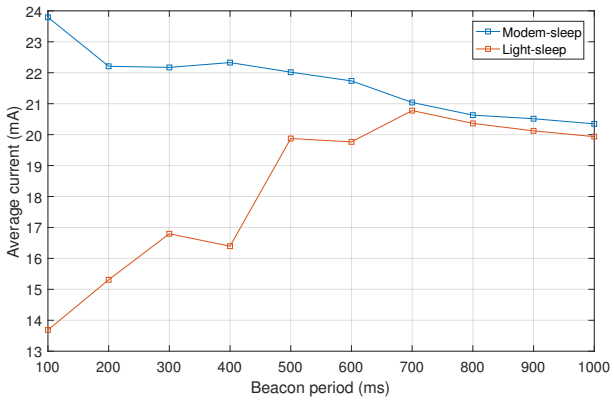

Fig. 6: Average current consumption varying the beacon period between 100ms and 1000ms, DTIM fixed at 3.

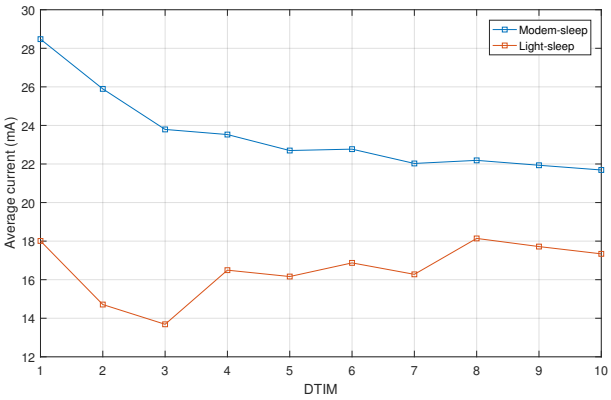

Fig. 7: Average current consumption varying the DTIM period from 1 to 10, beacon period fixed at 100 ms.

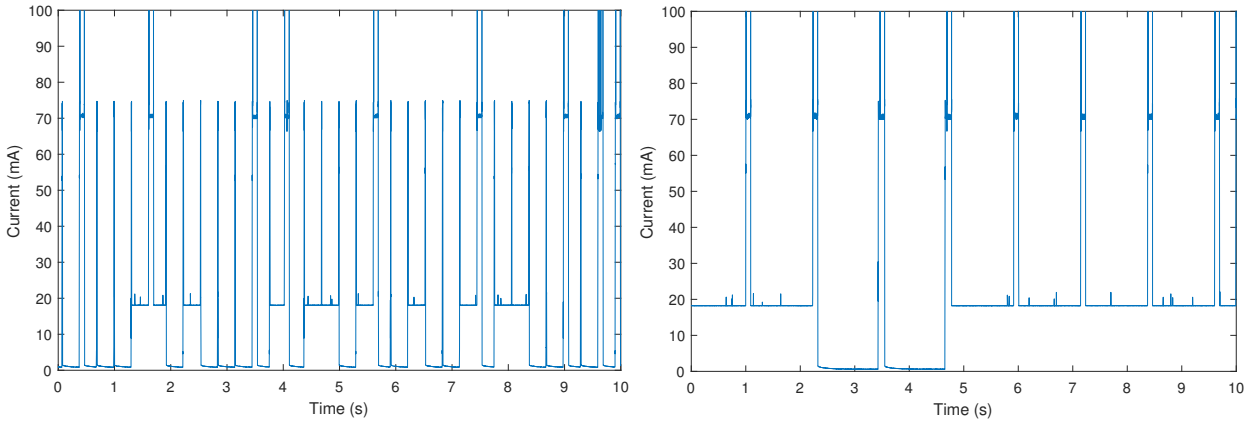

Fig. 8: Current trace with beacon interval of 100 ms and DTIM 3 (left) and beacon interval of 400 ms and DTIM 3 (right), in Light-sleep mode.

the Deep-sleep mode for which the 1s period could not be implemented due to the time required to re-establish the WiFi connection.

Figure 9 shows the distribution of the current measurements for each referred mode. The average current consumed throughout the entire experiment with the module in Active mode was 70.9mA for transmissions every second and 70.8mA for transmissions every 10s. In Modem-sleep mode, the average current consumption is considerably reduced when compared to the Active mode, showing 27.0mA (61.9% reduction w.r.t. Active) for transmissions every 1s and 23.6mA (66.7% reduction w.r.t. Active) for transmission every 10s. Further reductions were achieved enabling the Light-sleep mode, reducing the average current to 22.1mA (18.2% reduction w.r.t. Modem-sleep) when transmitting every 1s, and to 15.3mA (35.2% reduction w.r.t. Modem-sleep) when transmitting every 10s. These results also show that the improvements in energy efficiency were higher for the slower transmission frequency (once every 10s) and that the Light-sleep mode presents stronger variations caused by the occasional suspension of the CPU.

Table II summarises the average current consumption values for all execution modes and shows the respective estimated battery lifetime defined as the time to reach 10% of battery capacity, assuming the module was powered by a battery with a capacity of 1000mAh and the battery depletion was linear.

E. Connectivity in an indoor deployment

In this section we address the effective connectivity in an indoor scenario using the on-board antenna. We carried out two experiments, one to assess the RSSI that we can expect from the module's transmissions in indoor deployments and another one to assess the boundaries of connectivity, measuring packet delivery ratio (PDR) and round-trip delays (RTD).

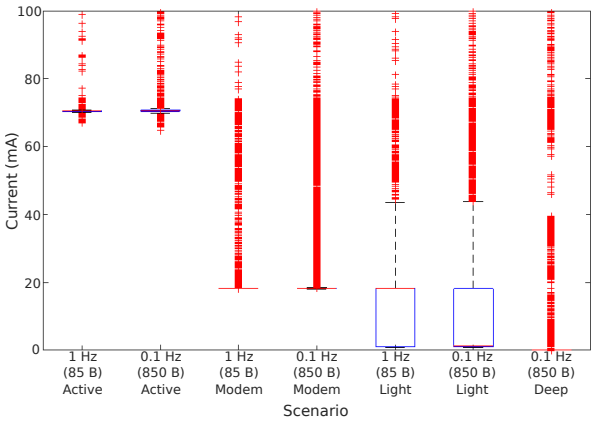

Fig. 9: Current consumption in an IIoT use case with seconds-scale transmission intervals using different execution modes.

1) RSSI of module's transmissions: The RSSI is often used has a quick assessment of how good a wireless link is. In this experiment the module was programmed to send continuously 85B data packets every second at

TABLE II: Average/median current and estimated battery lifetime for different execution modes.

Scenario		Average / median current (mA)	Battery lifetime (h)
1 Hz (85 B)	**Active**	70.9 / 70.5	12.7
	Modem-sleep	27.0 / 18.3	33.4
	Light-sleep	22.1 / 18.2	40.7
	Deep-sleep	-	-
0.1 Hz (850 B)	**Active**	70.8 / 70.4	12.7
	Modem-sleep	23.6 / 18.2	38.2
	Light-sleep	15.3 / 1.4	58.7
	Deep-sleep	9.8 / 0.03	92.3

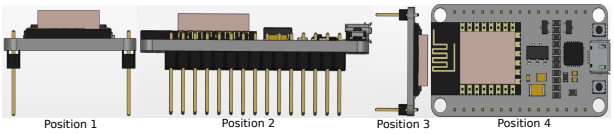

Fig. 10: Positions of the module's built-in antenna with respect to the receiver located in the viewer's position.

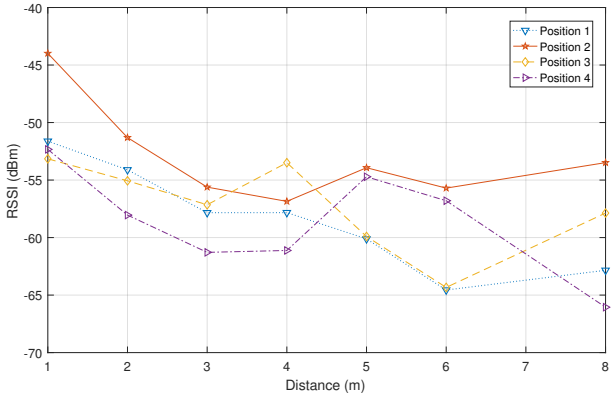

Fig. 11: RSSI measurements for different module positions and distances to the receiver.

distances varying from 1 to 8m from a receiver (our monitoring station) in line of sight. Moreover, given the expected non-uniform radiation pattern of the module built-in antenna (see Section II-D), for each distance point we measured the RSSI with four different module positions as shown in Figure 10.

For each distance and orientation we gathered 60 RSSI data points. Figure 11 plots the respective average RSSI values. For distances of 4m and larger, we can observe the typical RSSI large variations in indoor environments caused by multipath and other interferences. According to the theoretical radiation pattern of the built-in antenna [5], without multipath interference we would expect position 2 to be the most favourable, then 4, followed at similar levels by 1 and 3. Position 2 does perform better in our measurements, and positions 1 and 3 perform similarly. However, position 4 performs worse than expected, which can be due to multipath interference, which impacts the four positions differently. Most importantly, the RSSI values in all cases are within typical ranges for good reception, varying from an overall average of -52.9dBm for position 2 to -57.3dBm, -57.3dBm and -58.4dBm for positions 1, 3 and 4, respectively.

2) Connectivity limits: In order to assess the practical range of operation of the module in an indoor scenario we measured the PDR and the RTD in several places in a building, some in line of sight and other with concrete and brick walls in between (Figure 12).

A monitoring station was placed in the position represented by the wireless tower symbol and the ESP8266

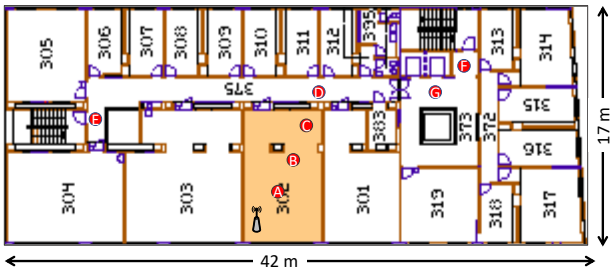

Fig. 12: Places in the building where we tested the limits of connectivity with the ESP8266 module.

positioned consecutively on the landmark represented by the letters. Landmark 'A', 'B' and 'C' are inside the room and are equidistants (3m apart). The remaining landmarks explore situations of interest or extremes: 'D' is separate of the source by a wall, 'E' to 'G' are in maximum distance (in multiple directions) and 'G2' is the same place as G, but on the floor below.

These experiments were carried out using the *ping* command, with 100 packets per point and 85B data payload, issued by a monitoring station running Ubuntu 16.04 LTS and connected to the AP via an Ethernet 100Mbit/s link, having the ESP8266 module as destination. This way, there are only two transmissions in the wireless medium, namely the *ping* request by the AP and the *ping* response by the module.

Figure 13 shows the distribution of the RTD measurements, with letters A, B, C, D, E, F, and G corresponding to the places marked in Figure 12. As expected, the measurements at places A, B, C and D show similar RTD results with a mean (and median) of 6.80 (4.07), 11.54 (4.10), 9.01 (4.06) and 9.76 (4.34) ms, respectively. On the other hand, as the distance from the access point and the number of traversed walls increases, the number of Wi-Fi retransmissions also increases leading to a clear extension of the RTD. At places E, F, G and G.2 we get an RTD mean (and median) of 11.87 (4.8), 14.08 (5.36), 15.95 (6.00) and 30.87 (15.3) ms, respectively. Moreover, with the default configurations of the module, the PDR was 100% at positions A, B, C, D, E and F, since all losses could be recovered with retransmissions. At places G and G.2 this was not always the case, with the PDR decreasing to 99% and 79%, respectively.

IV. Conclusion

Given the prevalence of WiFi at the Internet borders, the use of WiFi-enabled devices can bring a leap forward in the development of the Internet of Things in spaces where WiFi infrastructures exist, including in manufacturing. However, size, cost and autonomy issues, among other idiosyncrasies, have prevented developing such WiFi-enabled devices. Recently, technological advances in processor design have opened the way to modules that integrate WiFi interfaces with mid-range processors and

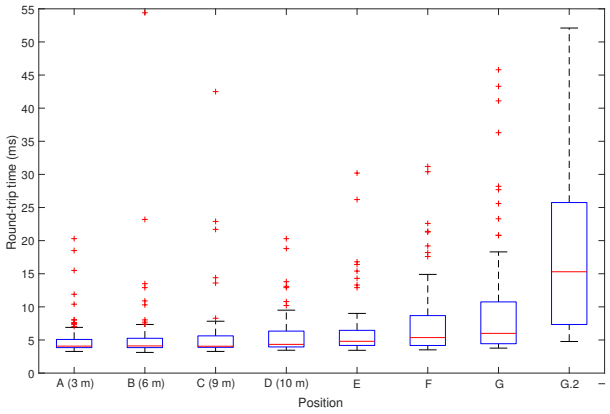

Fig. 13: RTD measurements distribution.

which offer effective sleep modes to enable low duty-cycle operation. This is the case of the ESP8266 module from Espressif Systems available for very low prices (few USD). Nevertheless, a detailed characterization of this module energy consumption and communication range was not available until now, to the best of the authors knowledge.

In this paper we showed how the automatic interaction with the WiFi infrastructure and its configuration in terms of beacon interval and DTIM period affects the power consumption and the effectiveness of the sleep modes. Curiously, the typical AP configuration of 100ms beacon interval and DTIM period of 3 yields the lowest average current consumption for the WiFi infrastructure (14.71mA) using the Light-sleep mode, which keeps the AP association. We also saw how data transmissions and sleep modes affect power consumption in the scope of an example IIoT application. When transmitting in the seconds-scale, the module can operate consecutively for 2 to 4 days from a small 1000mAh battery. Finally, we verified that the ESP8266 module offers good connectivity within a common building deployment, showing packet delivery ratios of 99% or higher in the same floor, but still usable across floors.

Future work will assess the module performance during handovers as well as with Direct WiFi configuration, in softAP and station/softAP combined modes, which are required for non-infrastructured use cases as in process control.

ACKNOWLEDGMENT

This work was funded by Instituto de Telecomunicaes, ref. UID/EEA/50008/2013, and by project NanoSTIMA – Macro-to-Nano Human Sensing: Towards Integrated Multimodal Health Monitoring and Analytics, NORTE-01-0145-FEDER-000016, supported by Norte Portugal Regional Operational Programme (NORTE 2020), through Portugal 2020 and the European Regional Development Fund.

REFERENCES

[1] Espressif Systems, *ESP8266 WiFi module*, https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf.

[2] Espressif Systems, *ESP8266 Low Power Solutions*, https://www.espressif.com/sites/default/files/9b-esp8266-low_power_solutions_en_0.pdf.

[3] Espressif Systems, *ESP8266 RTOS SDK*, https://github.com/espressif/ESP8266_RTOS_SDK.

[4] Espressif Systems, *ESP32 Datasheet*, https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[5] STMicroelectronics, *Low cost PCB antenna for 2.4GHz radio: Meander design*, AN3359 Application note, March 2011.

[6] R. Amornpornwiwat, P. Piyachat, V. Chawathaworncharoen, V. Visoottiviseth and R. Takano, "MATEMA6: Machine Tele-Monitoring Assistance with 6LoWPAN," 5th ICT International Student Project Conference (ICT-ISPC), pp. 49-52, Nakhon Pathom, Thailand, 2016.

[7] F. Astudillo-Salinas, D. Barrera-Salamea, A. Vzquez-Rodas and L. Solano-Quinde, "Minimizing the power consumption in Raspberry Pi to use as a remote WSN gateway," 8th IEEE Latin-American Conference on Communications (LATINCOM), Medellin, Colombia, 2016.

[8] Gianluca Cena, Stefano Scanzio and Adriano Valenzano, "Experimental Characterization of Redundant Channels in Industrial Wi-Fi Networks", World Conference on Factory Communication Systems (WFCS), Aveiro, Portugal, 2016.

[9] A. D. Deshmukh and U. B. Shinde, "A low cost environment monitoring system using raspberry Pi and arduino with Zigbee," International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2016.

[10] N. Kajikawa, Y. Minami, E. Kohno and Y. Kakuda, "On Availability and Energy Consumption of the Fast Connection Establishment Method by Using Bluetooth Classic and Bluetooth Low Energy," 4th International Symposium on Computing and Networking (CANDAR), pp. 286-290, Hiroshima, Japan, 2016.

[11] N. B. Long, H. Tran-Dang and D. S. Kim, "Energy-aware Real-time Routing for Large-scale Industrial Internet of Things," in IEEE Internet of Things Journal, 16 April 2018.

[12] C. Pereira, D. Guimarães, J. Mesquita, F. Santos, L. Almeida and A. Aguiar, "Feasibility of Gateway-less IoT E-health Applications," 27th European Conference on Networks and Communications (EuCNC), Ljubljana Slovenia, June 2018.

[13] R. A. Rashid, M. R. B. Resat, M. A. Sarijari, N. H. Mahalin, M. S. Abdullah and A. H. F. A. Hamid, "Performance investigations of frequency agile enabled TelosB testbed in Home area network," 2nd IEEE International Symposium on Telecommunication Technologies (ISTT), pp. 335-340, Langkawi, Malaysia, 2014.
ach Shelby and Carsten Bormann, "6LoWPAN: The wireless embedded Internet - Part 1: Why 6LoWPAN?" EE Times, 23 May 2011.

[14] P. Spachos, Liang Song and S. Gregori, "Power consumption of prototyping boards for smart room temperature monitoring," 22nd IEEE International Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD), Lund, Sweden, 2017.

[15] C. S. Stangaciu, M. V. Micea and V. I. Cretu, "Energy efficiency in real-time systems: A brief overview," 8th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), pp. 275-280, Timisoara, Romaina, 2013.

[16] D. M. Tung, N. Van Toan and J. G. Lee, "Exploring the current consumption of an Intel Edison module for IoT applications," IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Turin, Italy, 2017.

[17] Y. M. Yusof, A. K. M. M. Islam and S. Baharun, "An experimental study of WSN transmission power optimisation using MICAz motes," International Conference on Advances in Electrical Engineering (ICAEE), pp. 182-185, Dhaka, Bangladesh, 2015.