Nano-Things: Pushing Sleep Current Consumption to the Limits in IoT Platforms

Giannis Kazdaridis
iokazdarid@uth.gr
Department of Electrical and Computer Engineering, University of Thessaly, Greece

Nikos Sidiropoulos
nsidirop@uth.gr
Department of Electrical and Computer Engineering, University of Thessaly, Greece

Ioannis Zografopoulos
zografop@uth.gr
Department of Electrical and Computer Engineering, University of Thessaly, Greece

Polychronis Symeonidis
posymeoni@uth.gr
Department of Electrical and Computer Engineering, University of Thessaly, Greece

Thanasis Korakis
korakis@uth.gr
Department of Electrical and Computer Engineering, University of Thessaly, Greece

ABSTRACT
In this work we illustrate a novel power management architecture towards eliminating the power draw of IoT platforms during inactive periods. Our principle suggests the employment of an off-chip Real-Time-Clock (RTC) configured to control the power supply of the under consideration mote, by enabling or disabling its power in a power-gating fashion. The selected RTC features an ultra-low power profile and it is the only module that remains powered during sleep, hence the overall mote’s consumption is substantially diminished. Additionally, we introduce an alternative topology in which the host MCU remains powered in sleep state while the power-gating scheme is applied only in the rest of the peripherals of the IoT node, in an effort to exploit the MCUs benefits such as RAM retention and ultra-fast wake-ups. The proposed principle can be adopted by any IoT mote, in order to extend the life expectancy of battery-powered applications, by pushing sleep currents an order of magnitude lower. Moreover, we demonstrate the ICARUS mote, the first sensor that draws a sleep current of only 22 nA on a 3 V supply. Direct comparison of power draw in sleep state with state-of-the-art sensors illustrates improvements of roughly 98% - 99.8%, while we demonstrate that the life expectancy of the same motes can be prolonged from 2.7 years to 19 years under specific duty-cycles.

CCS CONCEPTS
• Hardware → Circuits power issues; Sensor devices and platforms; Wireless devices; Platform power issues.

KEYWORDS
power management, sleep current, low-power design, sensor networks, IoT, RTC

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IoT ’20, October 6–9, 2020, Malmö, Sweden
© 2020 Association for Computing Machinery. ACM ISBN 978-1-4503-8758-3/20/10... $15.00
https://doi.org/10.1145/3410992.3410998

1 INTRODUCTION
Energy efficiency is a major topic of research in the community of Wireless Sensor Networks (WSNs). In most real-world applications, sensors are battery operated, facing the inherent constraint of life duration that is solely dependent on the battery’s remaining charge and the node’s power profile. Fortunately, some deployment scenarios allow for battery replacement, which is a demanding procedure nonetheless and increases the maintenance overhead. On the contrary, it is not feasible to replace the batteries of nodes that are buried under the asphalt [19] to monitor available parking slots, or ones built into houses during construction [4] to allow for smart-home monitoring. The above highlight the requirement for further improvements in the sensors’ power consumption profile to operate on a single battery charge.

A common approach for saving energy in sensor networks is the duty-cycle concept, since overhearing and idle listening is a major source of energy wastage [24]. As a matter of fact, current consumption in idle state is roughly equal to the energy required for receiving a packet through the radio. To this end, sensor nodes are configured to enter a low-power mode, the so-called sleep state, in order to save as much energy as possible during their inactive periods. The sleep state is interrupted by short, burst events, where sensors sense, process and propagate data. It might seem reasonable to neglect energy consumption in the sleep state, since more than three orders of magnitude separate current consumption in sleep and active states [8, 17]. However, given the fact that typical sensor applications operate at quite low duty-cycles ranging from 0.01 % to 1 % [8], it is expected that both states account for the systems’ power budget expenditure [14]. Notably, sleep current is usually in the order of a few μA, which suggests that substantial energy savings can be attained. For example, consider a sensor node that draws...
15 mA on average when in active state and 5 μA when in sleep. Assume also that the node features an available battery capacity of 200 mAh and it is configured to operate in a duty-cycle fashion of 0.02%. The described sensor application will last for roughly 2.4 years. Consider now that a new power mechanism is employed that drops the drawn sleep current to 50 nA. In this case the resulting lifetime will reach 6.4 years, which is a tremendous improvement.

In this paper we introduce a novel mechanism to eliminate power draw in sleep state, while we demonstrate its performance and its applicability. The key contributions are outlined:

- we present an innovative energy management architecture that eliminates the current consumption in sleep state in duty-cycling applications
- we present the ICARUS mote, the first device that features an outstanding current draw of 22 nA in sleep state
- we evaluate the proposed system in terms of power consumption and wake-up performance
- we compare the life duration of indicative motes when adopting our principle versus when using standard features, noting substantial lifetime extensions

The remainder of the paper is organized as follows. Section 2 discusses preliminary notions and power saving strategies regarding the motes’ standard operation. Section 3 reviews the related work. System components and implementation are described in section 4, while the system’s performance and evaluation in section 5. Finally, section 6 compares the performance of indicative platforms when the proposed principle is applied and section 7 concludes the paper.

2 PRELIMINARIES

In this section we introduce key notions and principles used in low power systems, in order to ease the presentation of the concept. Duty-cycling in sensor systems is realized by disabling as many parts as possible during inactive periods to conserve energy. Commonly, on-board modules and peripherals feature a low-power state where they consume as little power as possible, while the host MCU is responsible for managing the whole process. Of course, the host MCU also enters a sleep state to save energy. Actually, modern MCUs feature a number of low-power modes, ranging from light-sleep or standby mode to complete shut-off [14].

MCU blocks, as well as their interconnected peripherals, are sourced by different clocks which operate at variable frequencies. These clocks are progressively disabled, according to each low power mode’s constraints, in favor of better energy consumption. For instance, TI’s MSP430 family supports several lower power modes, the so-called LPMs. The clock that each module (e.g. ADC, Comparators, etc.) will be sourced from can be set, making it usable in more energy efficient LPMs albeit with a lowered performance. Evidently, the wake-up time is also affected by the LPM being used. However, the time required for an MCU to boot when power is initially supplied is referred to as cold-start time, which is always longer than any other wake-up time.

The lowest possible state in terms of power draw is the shut-off/ deep-sleep state, in which the MCU turns off completely, while only the minimum functionality required to restore the MCU back to active state from an external signal is preserved. This mode reduces the power consumption to an absolute minimum, in some microcontrollers as low as 20 nA. Apparently, this mode requires an external stimulus, thus it is not often opted for in duty-cycled systems. When considering duty-cycled schemes, the so-called standby state is employed. In this state MCUs may preserve only a time-keeping circuit active, to provide the required interrupts in the given intervals.

The typical time-keeping circuits integrated into MCUs are the Watchdog timer (WDT) and the Real-Time Clock (RTC). The WDT is a specific guard timer used to detect and recover the MCU after a malfunction occurs. Commonly, it draws more power than the RTC while it has more limitations in the supported intervals and the supported time accuracy, hence the RTC is usually preferred to provide the interrupt stimulation. However, several MCUs, such as the ATMega family, do not incorporate an RTC circuit to allow for wake-ups, hence WDT remains the only option. The RTC is a time-keeping circuit used in a wide range of systems. It is running over either a crystal oscillator or a relaxation oscillator, usually at low speed clocks, resulting in low power draw. Additionally, it offers advanced time accuracy, which is crucial when considering synchronized wake-up schemes for sensor networks. The power draw of a MCU in standby state, when the RTC remains active can be anywhere from a few hundred nA to a few μA, substantially lower compared to a WDT timer.

Another consideration in duty-cycled sensor systems is the memory retention. Most MCUs support low-power modes that retain volatile memory contents by constantly supplying power to the module. Following the latter principle, MCU manufacturers provide memory retention capabilities which preserve the state of the executed program in an effort to resume the execution from the point it was left off. Apparently, this process is crucial in intermittent computational systems [1, 7], such as sensor networks in duty-cycle schemes, providing fast recover times. On the other hand, there are state retention schemes that utilize non-volatile memories that do not require power to retain their content, such as Flash or EEPROM [3]. Non-volatile memory is, however, significantly slower and more energy hungry than RAM. A modern type of non-volatile memory, FRAM, grants much higher access speeds than other types of non-volatile memory. Although it consumes marginally more power than RAM, it still remains a better alternative, consumption-wise, to Flash or EEPROM. Notably, TI integrates FRAM technology in its MSP430FR MCU family [6].

Undoubtedly, the power profile of modern MCUs in the standby state has been been remarkably improved, however, the total consumption of sensor nodes is often marginally higher, since most sensor devices integrate a vast number of external devices, e.g., sensing modules, an RF chip and all the requisite power electronic circuitry. All the aforementioned individual elements may also feature their own standby mode, in which they draw insignificant amount of power, however, the aggregated consumption is never negligible. In the context of low-power sensor design, all power expenditures should be accounted for, and to tackle this challenge, the power-gating [20] technique is enforced. Power-gating suggests that peripherals that are not in use may be entirely disconnected from the power source, by employing a load-switch that is usually controlled by the MCU. The only sub-system that must always remain powered is the MCU that retains the time-keeping circuit. Apparently, this strategy is not often applied in the majority of sensor nodes. Even when adopted, the standby current will be roughly a few μA, since the power regulator’s quiescent current must be taken into consideration as well.
3 RELATED WORK

In this section we present the most widely adopted IoT devices along with their power characteristics, in order to provide a thorough comparison with our proposed system.

The MicaZ [15] and the TelosB [16] are considered to be [2] two of the most energy efficient platforms, that draw 15 μA and 8.8 μA respectively on a 3.3 V supply, when in sleep mode. Both boards were designed more than a decade ago, and therefore cannot compete with the latest sensor developments anymore. The Opal [9] is a prototyping platform based on an 32-bit ARM Cortex-M3 MCU that consumes roughly 8.9 μA in sleep state. Apparently, it is a good example of a board that embeds a vast number of electronic peripherals as well as an extra wireless module, without adopting any power-gating scheme, thus it features such high power draw in spite of utilizing modern ICs. The Storm [2] is also a recently developed prototyping board that utilizes a powerful 32-bit ARM Cortex-M4 MCU. Despite the fact that the authors claim the selected MCU can compete with the best-in-class IoT platforms in terms of energy efficiency, the resulted draw in sleep phase is 13 μA and 2.3 μA on 3.3 V and 1.8 V voltage rails respectively, which is not considered low. Notably, all the aforementioned sensor systems rely on the internal time-keeping functions of their host MCUs, in order to trigger wake-up signals when in sleep mode.

An alternative principle to provide wake-up interrupts is presented by the XYZ [13] and the EcoBT [22] sensor devices. These platforms support the integration of an off-chip RTC circuit, replacing the MCU’s power consuming time-keeping functions, in order to provide external interrupts to wake-up the host node. This method suggests that the MCU will enter a deep-sleep state consuming much less power. However, the attained power draw is roughly 30 μA and 2 μA respectively, since the selected ICs do not feature power efficient deep-sleep modes.

A remarkable IoT platform in terms of power efficiency is the TI’s eZ430-RF2500 [5], which draws roughly 1 μA in sleep. Its ultra-low power profile is attributed to the on-board MCU which draws just 600 nA in its LPM3, where its internal WDT remains active to re-trigger the node when required. Notably, apart from the MCU the board features only the CC2500 RF chipset, without integrating any other sensing module nor a voltage regulator, hence it is able to attain that compelling power draw. On the other hand, there is the Waspmote sensor [23], which can be fitted with a vast variety of sensors featuring only 860 nA in its lowest sleep mode with wake-up capabilities. To achieve this remarkable performance all the on-board or attached modules are power-gated with the aid of load-switches that are controlled by the host MCU. Moreover, the Waspmote is outfitted with an off-chip RTC, the DS3231, that, along with a load-switch, cuts the power of the entire node. Since, the DS3231 can only provide short interrupt signals, a latch circuit, formed by a monostatic multivibrator and a few logic gates, is employed to retain the state in order to smoothly drive the load-switch.

It is worth noting that the Waspmote is a commercial platform that doesn’t reveal the principle behind the wake-up implementation, but we reverse engineered the aforementioned circuit block to determine the components used and the architecture followed. Apparently, the majority of the devices exceed 1 μA in sleep state with just the Waspmote and the eZ430 achieving better performance. In our paper we follow a principle similar to the Waspmote’s by utilizing an extremely low-power off-chip RTC module and configuring it to manage the power of the entire sensor node, achieving a current draw of as low as 22 nA at 3 V operation. A direct comparison between the Waspmote and our system, illustrates impressive reduction in power consumption, when, at the same time, our architecture does not require the complicated array of components used by the Waspmote to maintain the state of the alarm interrupt since our selected RTC features such capabilities. Lastly, we refer to our previous work [11] presenting a small set of the proposed principle, in which we employ the TI’s TPL5111, an ultra-low power timer, instead of the proposed RTC. The TPL5111 is used to provide the external stimulus featuring 33 nA power draw. However, the time accuracy of the selected timer is utterly poor featuring 100 ppm drift, while the selected RTC supports 2 ppm accuracy, that allows the formation of synchronized wake-up events in distributed networks. Moreover, the proposed RTC can be easily re-synchronized to correct any time drift, while the aforementioned timer does not support such capabilities.

4 NANO POWER SYSTEM IMPLEMENTATION

In this section we describe the proposed architecture and we detail the technicalities of our system.

4.1 Niche Low Power Methodology

Typically, sensor nodes draw significant amounts of power when in sleep state, due to the poor power efficiency of their time-keeping circuits (regardless if they are incorporated in the MCU or not), power leakage issues and the fact that the aggregated consumption of the peripheral modules (sensors, ICs, etc.) is not negligible. In this work, we propose the employment of a on-board, off-chip RTC module with an ultra-low power profile, to manage the go-to-sleep and wake-up phases of battery powered IoT nodes. In essence, we replace the existing time-keeping circuit that consumes substantially more power, and we shut-off the entire node in order to alleviate...
any power expenditure. We propose two different topologies for utilizing an off-chip RTC, each one with different trade-offs.

**PSW Topology**: In the PSW (Power Switch) case we employ a load-switch along with the RTC, to completely power-off the under-consideration sensor node adopting the power-gating [20] method, as illustrated in Fig. 1(b). Practically, the IoT node is powered via the load-switch which is controlled through the RTC. This approach results in as low consumption as possible, since it eliminates any power draw induced by the sensor node, as only the off-chip RTC and the load-switch remain active. Evidently, this configuration induces noteworthy delays when considering the cold-start boot time, which is relatively longer compared to the wake-up time in standard operation. Given the above trade-off, the impact of the proposed principle depends on the characteristics of each node and the duty-cycle of the application, as described in section 6. It is worth noting that the proposed topology can benefit from the FRAM technology, to instantly return to the desired program state after a sleep period, substantially reducing the comprehensive overhead.

**Interrupt Topology**: On the other hand, in the Interrupt (INT) configuration the MCU remains constantly powered even in sleep state, while the off-chip RTC is employed to provide waking signals. This strategy suggests that the host MCU exploits the deep-sleep mode, in which it consumes only a few nA, in contrast to the typical power-consuming standby mode. Moreover, we apply the power-gating principle to the remaining electronics and peripherals of the IoT node with the aid of a load-switch in an effort to eliminate their power draw when asleep. It is worth noting that in this topology we also employ an external voltage regulator featuring extremely low quiescent current in order to power the MCU. The proposed architecture is illustrated in Fig. 1(c). Despite the fact that in this topology the overall power draw is apparently higher compared to the PSW case, it still remains in the order of a few nA when considering modern MCUs. The crucial advantage of this approach is the fast wake-up time, comparable or even the same as in standby mode. Another asset of this scheme is that the host MCU can be configured at any low-power mode during sleep phase, in order to take advantage of any supported feature, such as the RAM retention capability that allows for incredibly fast wake-ups.

### 4.2 Components Selection

In this subsection we present the selected components detailing their characteristics and the exact wiring of the proposed system.

**RTC**: We reviewed several off-the-shelf RTCs and listed the most compelling ones along with their characteristics in Table 1. The most widely used in IoT devices is the DS1307, which features poor accuracy, while in some delicate applications we meet the DS3231.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Renesas</td>
<td>ESL12022</td>
<td>1 μA</td>
<td>3 ppm</td>
<td>2.5-5.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>Maxim Int.</td>
<td>DS3231</td>
<td>8 μA</td>
<td>2 ppm</td>
<td>2.3-5.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>Microchip</td>
<td>MCP79412</td>
<td>700 nA</td>
<td>10 ppm</td>
<td>1.8-5.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>Maxim Int.</td>
<td>DS1307</td>
<td>300 nA</td>
<td>23 ppm</td>
<td>4.5-5.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>ST</td>
<td>M41T62</td>
<td>350 nA</td>
<td>2 ppm</td>
<td>1.3-4.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>NXP</td>
<td>PCF8563</td>
<td>225 nA</td>
<td>29 ppm</td>
<td>1-5V</td>
<td>Int.</td>
</tr>
<tr>
<td>MicroCrystal</td>
<td>RV3028</td>
<td>45 nA</td>
<td>1 ppm</td>
<td>1.1-5.5V</td>
<td>Int.</td>
</tr>
<tr>
<td>MicroCrystal</td>
<td>RV1805</td>
<td>17/22/30 nA</td>
<td>2 ppm</td>
<td>1.1-3.0V</td>
<td>Int./PSW</td>
</tr>
<tr>
<td>Abracon</td>
<td>AM0805</td>
<td>15/22/35 nA</td>
<td>2 ppm</td>
<td>1.5-3.6V</td>
<td>Int.</td>
</tr>
</tbody>
</table>

**Table 1: Compelling RTCs and their Specifications**

The latter provides sufficient accuracy of 2 ppm, but consumes 840 nA, while other candidates exhibit better performance, consuming a few hundred nA, but are barely seen in any IoT devices. In our system we opted for the Micro Crystal RV1805-C3 [18], which achieves similar performance with the Abracon AM0805 RTC, but also features a specially designed output able to directly drive external loads.

The RV1805 supports two modes of operation, where different oscillators are activated each time. In the XTAL mode, a 32.768 kHz clock is running featuring 60 nA power draw, while the RC mode achieves worse accuracy but draws only 17 nA. To improve the RC mode performance, RV1805 enforces an auto-calibration mechanism, exploiting the XTAL crystal oscillator, in which case the average power draw is roughly 22 nA. In our application scenarios we configure the RV1805 in RC mode with auto-calibration every 512 seconds. The factory calibrated clock achieves a time accuracy of typically ± 2.0 ppm at 25 °C, while communication with the host MCU is attained over I2C.

The selected RTC features two types of output signal, an Interrupt pin (nINT) providing a short pulse to trigger external devices (lasting for 200 ms) and a stable-state output pin (PSW) that changes state when triggered and remains in this second state until otherwise instructed. When advised so, the PSW output automatically returns back to its original low-state awaiting for a second trigger pulse. In our implementation we utilize both the nINT and PSW pins to form the Interrupt and PSW schemes respectively. Notably, a stable-state signal is required to drive the load-switch in order to support the PSW topology.

The RV1805 is essentially the only available RTC supporting a stable-state output (the PSW pin). In an effort to implement the proposed PSW configuration with alternative RTC ICs that do not support PSW output, we implemented a proof-of-concept prototype board based on the RV3028 RTC. This circuit, illustrated in Fig. 2, employs a latch IC to retain the state after triggered by a short interrupt signal. The prototype features the NC7S504 Inverter and the 74AUP1G373 D-type latch IC, which are ultra-low power and draw roughly 3nA in total, substantially lower compared to the Waspmite’s state-retention circuit that uses the SN74LVC1G123 monostable multivibrator.

**Load Switch**: Regarding the load-switch which we employ to implement the power-gating principle, there are several available off-the-shelf models. We distinguish the TPS22860 and the ADG821 featuring a quiescent current of roughly 0.3 nA, while supporting ultra-fast response times. Both switches drive loads of up to 200 mA, whereas higher load output switches can be employed when considering power-hungry IoT nodes. Alternatively, MOSFET ICs can also play the role of the load-switch.

It is worth noting that all the selected components are of low-cost (< 5€ in total), while the proposed circuit is characterized by low complexity, hence it can be easily adopted by any commercial or
4.3 Implementation Setup & Design

In this subsection we demonstrate the prototype ICARUS mote that encompasses the proposed system architecture and we explain the configuration of the system in each topology.

ICARUS Prototype Mote: The ICARUS, illustrated in Fig. 1(a), features the STM32L476RG which is an ultra-low power ARM Cortex-M4 32-bit RISC core MCU operating at a frequency of up to 80 MHz. It embeds high-speed Flash memory of 1 MB and an SRAM of 128 KB. The mote integrates an XBee-footprint socket for plugging-in wireless interfaces, such as LoRa, ZigBee, BLE, etc. Moreover, it embeds the SHT21 temperature & humidity sensor, the VEML6030 light intensity, and the MAX17048G+ battery gauge sensors, while extra sensing modules can be interfaced through the available I2C and I/O ports. In addition, an off-chip FRAM memory, the MR45V256A, is assembled on the board that can be used for memory retention schemes even when the host MCU is not powered. Of course, the ICARUS also integrates the RV1805 RTC, which, along with the ADG821 load-switch, is used to switch the power of the mote entirely on or off, or to alternatively serve as an interrupt source for waking-up the host MCU. An extra ADG821 is employed to control the power rails of the wireless interface and of the attached sensors and peripherals. Notably, each ADG821 features two internal switches. The ICARUS exhibits roughly 22 nA power draw in sleep state, exploiting the proposed mechanism.

PSW Topology Configuration: In the case of the PSW topology, when the mote receives power for the first time, it initially configures the RV1805 with the appropriate settings and the desired time interval, as illustrated in Fig. 3(a). The mote then performs its typical workload and when it has completed its tasks it issues a sleep command to the RV1805. Automatically, the RTC asserts its output signal low (the PSW pin is employed) which, as a result, cuts the power from the entire node through the load-switch. Notably, the RV1805 features a sleep state as well, in which it turns off the I2C interface and enters into a low-power state until the next cycle. After the specified by the application interval, the RV1805’s timer fires up restoring power to the host sensor through the load-switch. Apparently, in this topology, I2C communication with the host MCU is required during every active period for the instruction of the sleep command to the RTC after the completion of the workload.

Interrupt Topology Configuration: In the second scheme, the RTC is again configured by the host MCU during the initialization of the sensor, as illustrated in Fig. 3(b). However, in this approach the RV1805 is set up to provide interrupt signals directly to the host MCU without requiring the reception of a sleep command by the host node as in the first approach. Therefore, after the initial configuration, the I2C channel may be re-established only for the resynchronization of the RV1805, so as to support synchronized wake-up schemes in mesh networks. In this topology, the load-switch feature is employed to power the remaining electronics controlled by the host MCU via an I/O pin.

RV3028’s Equivalent PSW Topology Configuration: Lastly, we refer to the prototype developed based on the RV3028. The configuration of the RV3028 is realized similarly to the Interrupt topology, providing short interrupt signals. An unbuffered inverter is used to convert the generated active-low interrupt signal to an active-high which is required by the utilized latch in order to change its state from low to high and successfully drive the load-switch. The D-type latch features two input pins, the clock and the data. The clock input is connected to the inverter’s output, while the data input is asserted high, so as to switch to high-state when triggered. When the node’s workload is completed, the host MCU asserts the data pin low and instantly provides an interrupt pulse to the clock input in order to modify the state to low and cut the power off.

In an effort to test the proposed system with different MCU architectures we used the ATMega1284p, the MSP430FR5969 and the STM32L476 MCUs. To enable communication between the selected MCUs and the proposed RTCs, we utilized a slightly modified version of SparkFun’s library for the ATmega case, while we developed the corresponding libraries for the MSP430 and STM32L476.

Regarding the power supply of the RV1805, it can be powered directly from a 3 V lithium-manganese dioxide battery. However, when considering typical IoT applications, sensor nodes often feature 3.7 V cells that output roughly 3.7 to 4.2 V. In this case, a voltage regulator is required to provide lower voltage within the accepted range. However, this is not the best practice, since regulators typically feature quiescent current of a few μA. Only the TI TPS62740 and the TI TPS7A02 feature an outstanding quiescent draw of 360 nA and 25 nA respectively, which is certainly an exception not used by any known mote. In fact, most IoT nodes feature quite high sleep currents, for the sake of powering their MCUs and other peripherals with the appropriate voltage rail, commonly at 3.3 V. Another way to power the RV1805 without employing a voltage regulator is to exploit the back-up capacitor that it supports for such purposes. The RV1805 features an internal circuit able to instantly charge its back-up capacitor when power is applied, and automatically switch to this power source when power is disconnected. Practically, in our system the RV1805 charges the capacitor in every active-cycle by the sensor’s regulated rail, while it consumes zero power in sleep.

Lastly, we note that we employ the TPS7A02 regulator to power the MCU in the sleep state when the Interrupt topology is applied.
5 PERFORMANCE EVALUATION

In this section we analyze the performance of the proposed system, evaluating both the derived power draw as well as the wake-up performance of a wide set of MCUs.

5.1 Power Consumption Evaluation

In this subsection we evaluate the power consumption profiles of the RV1805 and the RV3028 under the different modes and varying voltage rails; we also discuss the instantaneous power consumption of a sensor system when our principle is applied.

In our implementation we opted for the RC mode with auto-calibration, since in XTAL mode the RV1805 draws significantly more energy (60 nA), while in RC mode (17 nA) without calibration the supported accuracy is fairly poor. The auto-calibration is performed either every 1024 or 512 seconds and lasts for roughly 50 seconds. Therefore, it is not trivial to calculate the average power draw of the RV1805. In our evaluation experiments we exploited the uCurrent meter [21], which is designed to measure ultra-low currents with a resolution of up to 1 pA. However, uCurrent focuses on monitoring currents that remain stable for substantial duration, thus, we used it only to measure the level of each phase with high accuracy. For capturing transient phases we rely on our high-fidelity monitoring tools [10, 12]. Specifically, [12] features a dynamic shunt resistor switch that alternates depending on the flowing current, which eases the process of measuring the power profile of the RV1805.

Fig. 4(a) illustrates the obtained power results at 3.4 V power supply. Notably, the red line represents the average power draw calculated as a moving mean over a window of 100 values so as to provide a more representative indication. In the illustrated experiment we configured the RV1805 to provide alarm signals with an interval of 10 seconds, while we measure its power draw. We observe that when the RV1805 is in idle state, the instantaneous power draw is roughly 19.2 nA. At 18.4 seconds the process of auto-calibration is initiated. At first the XTAL crystal powers up, stabilizes and at 20 seconds the calibration function begins, lasting for 50 seconds. The sudden power draw oscillations at 13.6 μA that occurred every 10 seconds are attributed to the interrupt signals generated as configured in this experiment. To determine the power draw of the RV1805 we calculate the average value over a single auto-calibration period, without of course considering interrupt events in this time frame. The obtained results of the RV1805, plus the RV1805 together with the ADG821 under varying voltage are presented in Fig. 4(b) along with the corresponding values of the RV3028 with and without the proposed latching circuit.

The resulted draw of a sensor system in sleep state when employing the PSW topology is presented in Fig. 4(b). Even when assuming the utilization of FRAM retention schemes, supported by the MSP430 MCU family, the power draw will remain the same, since FRAM does not require power to retain data. On the contrary, when considering the power consumption of a mote in Interrupt topology we must calculate the aggregate draw of the selected RTC and load-switch plus the power draw of the regulator and the MCU in the selected sleep state. Table 2 consolidates detailed characteristics of a wide-range of MCUs in various Low-Power Modes (LPMs), noting the draw of each state with and without a time-keeping circuit activated. Notably, the Table is separated into two main LPM categories, the obtained consumption with RAM retention enabled and without. For instance, the STM32L476 draws 1.72 μA with RTC enabled and 1.22 μA without any time-keeping circuit, while supporting RAM retention in both cases. The same MCU draws 503 nA with RTC and 88.5 nA without, when not supporting RAM retention. Now considering a sensor node that features the above MCU while configured in Interrupt topology, it will consume roughly 1.27 μA with RAM retention and 136 nA without. In the same scenario, a sensor node featuring the MSP430FR5969 will consume 547 nA, with RAM retention and only 69 nA with no retention capabilities. Fig. 4(c) summarizes the power draw of indicative MCUs in sleep state with and without RAM Retention (RR), when the Interrupt topology is applied.

It is worth noting that some MCUs feature quite high power draw when their internal time-keeping circuit is activated but achieve extremely low consumption when it is not. For example, the STM32G473 if no retention is supported, consumes 672 nA with the RTC enabled and only 65 nA with the RTC disabled. Similarly the MSP430F2274, attains 600 nA with the RTC and 100 nA without, when RAM retention is supported. Apparently, in the latter scenarios it is deemed efficient to adopt the Interrupt topology, while for other MCUs, such as the ATMega1284p, that present extreme power draw (850 nA), the PSW scheme seems a better option.

Lastly, we demonstrate the percentage reduction of the sleep current, when comparing existing sensor systems with our principle. When considering the eZ430 we illustrate a reduction of 97.8 %, while in the case of Storm, we highlight a substantial reduction of 99.83 %, assuming that the PSW topology is applied in both cases.
### 5.2 Wake-Up Time Evaluation

In this subsection we characterize the wake-up time required for different MCUs to recover to an active state from the various low-power modes, and how this affects the performance of the proposed system. Apparently, the application of the proposed PSW topology comes at the cost of additional time required for the sensor to enter an active state, since in this topology the node disconnects completely from power when asleep. Table 2 presents the wake-up times of indicative MCUs under various low-power modes. Evidently, the cold-start time is always slower compared to the standby mode, while the modes that support RAM retention capabilities are exceptionally fast. For example, the STM32L476 features 7 ms when waking-up from a RAM retention state and 256 ms when no RAM retention is supported, while it requires 3.9 ms for cold-start. The wake-up times in different modes must be taken into consideration when selecting between the PSW and the Interrupt topology. For instance, a MCU with fast cold-start, such as the MSP430FR5969, can exploit the PSW topology while for slower MCUs, such as the STM32L073, it is preferable that they are configured in Interrupt topology.

Undoubtedly, wake-up times add to the overall active time of a node operating in duty-cycle. While potentially a drawback for our implementations, it should be noted that this is only the case when these times comprise a substantial portion of the cycle’s duration in a duty-cycled scheme. It is worth noting that a typical node requires roughly 200 ms [8] in active period when required to propagate a frame, while much longer active cycles are required when considering more sophisticated applications or processes, such as the re-establishment of a mesh network. As a result, the overhead of 1 ms cold-start in the case of the MSP430, might be an acceptable energy cost to pay in order to minimize the power draw in sleep state. However, the 7.52 ms cold-start of the STM32L073 or an even longer cold-start duration might add a notable energy loss in the sensor’s power budget. Consequently, MCUs featuring fast wake-up times are less likely to affect the overall active energy, but when considering slower families, it is preferable to take the overall energy trade-off into account before adopting the proposed principle.

<table>
<thead>
<tr>
<th>MCU</th>
<th>Core Architecture</th>
<th>LPM w/ full RAM retention</th>
<th>LPM w/ o RAM retention</th>
<th>Cold-start</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>cons. w/ RTC</td>
<td>cons. w/o RTC</td>
<td>wake-up</td>
</tr>
<tr>
<td>Atmega1284p</td>
<td>8-bit AVR</td>
<td>4.5 μA (WDT)</td>
<td>850 nA</td>
<td>110.4 μs/4 ms/65 ms</td>
</tr>
<tr>
<td>STM32G473RE</td>
<td>32-bit Cortex-M4</td>
<td>81.5 μA</td>
<td>80.5 μA</td>
<td>9.5 μs</td>
</tr>
<tr>
<td>STM32L476RG</td>
<td>32-bit Cortex-M4</td>
<td>1.72 μA</td>
<td>1.22 μA</td>
<td>7 μs</td>
</tr>
<tr>
<td>STM32L432KC</td>
<td>32-bit Cortex-M4</td>
<td>1.63 μA</td>
<td>1.15 μA</td>
<td>8.2 μs</td>
</tr>
<tr>
<td>STM32L073RZ</td>
<td>32-bit Cortex-M0+</td>
<td>860 nA</td>
<td>430 nA</td>
<td>3.5 μs</td>
</tr>
<tr>
<td>MSP430FR5969</td>
<td>16-bit MSP430</td>
<td>700 nA</td>
<td>500 nA</td>
<td>7 μs</td>
</tr>
<tr>
<td>MSP430F2274</td>
<td>16-bit MSP430</td>
<td>600 nA (WDT)</td>
<td>100 nA</td>
<td>1 μs</td>
</tr>
<tr>
<td>MSP430F401R</td>
<td>32-bit Cortex-M4F</td>
<td>860 nA</td>
<td>700 nA</td>
<td>9 μs</td>
</tr>
<tr>
<td>ATSAM4LC8C</td>
<td>32-bit Cortex-M4</td>
<td>3.4 μA</td>
<td>2.3 μA</td>
<td>1.5 μs</td>
</tr>
</tbody>
</table>

Table 2: Compelling MCUs Characteristics in Different Low-Power States

As already mentioned, FRAM functionality can be exploited in PSW topology to allow for state retention. However, there is a minor overhead time required to store all data before the shut down and a similar overhead to recover the data from the FRAM. We measured this time exploiting TI’s libraries for the MSP430FR5969 and obtained an overhead of roughly 1 ms when restoring the application context and 100 B of data from FRAM to RAM. This overhead must also be taken under consideration when assuming state retention via TI’s FRAM support.

### 6 COMPARISON

In this experiment we evaluate the lifetime of two state-of-the-art platforms under varying duty-cycles comparing their performance when adopting our principle versus the standard features, in order to understand the improvement induced by the proposed system. Indicatively, we consider the platforms eZ430-RR2500 equipped with the MSP430FR5969 and the Storm which encompasses the ATSAM4LC8C. Their power draw when asleep is 5.53 μA and 13 μA respectively, while the resulting power draw when adopting our technique is discussed in section 5.1. Moreover, Table 2 details the wake-up times for each MCU. Notably, the MSP430F2274 supports only a WDT timer which is unable to provide alarm intervals greater of 8 s, thus we configure the mote to perform intermediate wake-ups in order to extend the maximum supported interval, which results in an average draw of 5.53 μA. Regarding their consumption in active state, we consider the average values of 12 mA for the eZ430 and of 8.6 mA for the Storm, calculated by their instantaneous power profiles under different tasks when active. In addition, we assume an active period of 200 ms [8] and that both motes feature batteries with an available capacity of 360 mAh while the battery’s self discharge is ignored.

Fig. 5(a) and 5(b) depict the calculated life expectancy of the aforementioned motes under varying duty-cycles. In each case, we also plot configurations that feature RAM retention (RR) when available. Apparently, when considering a 0.025 % duty-cycle, we observe a substantial increase of 183 % and 603 % for the eZ430 and the Storm respectively, considering the PSW configuration for both motes. Specifically, we note that the eZ430 extends its life from 4.8 years to 13.6 and the Storm to 19 years all the way from just 2.7 years. Notably, under this duty-cycle, the Storm presents a worse lifetime with the standard features compared to the eZ430, while it presents higher life duration increase when our method is applied. This comes as a result of its poor performance in sleep (13 μA) compared
to the eZ430 (5.53 µA), while featuring lower consumption in its active state. The Interrupt configuration yields less pronounced but similar results. In the case of the Storm, there is a differentiation among the PSW, Interrupt and Interrupt RR topologies in low duty-cycles, which stems from the significant variation of the power draw obtained between the corresponding states, 22 nA, 944 nA and 2.3 µA respectively. On the contrary, the eZ430 features nearly equal consumption on these states, hence the resulting lifetime is roughly similar. Higher duty-cycles present lower lifetime increase for both devices. However, even on 0.2 % and 0.3 % duty-cycles the Storm presents an increase of 75 % and 50 % respectively, when using the PSW topology. At the same time, the effectiveness of our approach for the eZ430 taper off (less than 10 % increase) for duty-cycles over 0.45 %, while for the Storm, for duty-cycles over 1.5 %.

Similarly, we plot the calculated life expectancy of the Icarus mote, in different topologies in Fig. 5(c). To this aim, we assume a consumption of 7 µA in sleep state when in standard operation and an average consumption of 15 mA in the active phase. We observe that the PSW and Interrupt topologies achieve nearly equal behavior, since the obtained power draw in these states are quite similar, 22 nA and 132 nA respectively. On the other hand, when RAM retention (1.26 µA draw) is enabled the overall lifetime is diminished, but still substantially improved over the standard operation.

Concluding, our principle greatly benefits IoT platforms with low energy-efficiency in sleep state when in low duty-cycles. We also note that the duration of the active period, does not affect the life expectancy of an IoT platform, unless it is extremely low, comparable with the wake-up times of the various MCUs.

7 CONCLUSIONS

In this work we employed an external RTC module to control the go-to-sleep and wake-up functions of any IoT mote. The suggested principle is set-up with low-cost off-the-shelf components, while it remains minimally invasive to the host node. By adopting our strategy the power consumption of an IoT mote in sleep state can drop as low as 22 nA, which is at worst a reduction of 98 %, compared to the most power efficient platforms available. Remarkable lifetime extensions can be achieved in low duty-cycled scenarios, while in higher duty-cycles the benefit is minimal. Finally, we foresee that IC manufacturers will add similar functionalities to the ones proposed in this work into their MCU’s within the next years in order to reduce the power profile of their products.