Design Guidelines for Ultra-low Power Gateways in Environment Monitoring Wireless Sensor Networks

Maximus Byamukama[1], Janet Nakato Nannono[1], Kabonire Ruhinda[1], Björn Pehrson[3], Mary Nsabagwa[2], Roselyn Akol[1], Robert Olsson[3], Geoffrey Bakkabulindi[1], Emmanuel Kondela[4]

[1]College of Engineering, Design, Art and Technology, Makerere University, Uganda

[2]College of Computing and Information Sciences, Makerere University, Uganda

[3]KTH Royal Institute of Technology, Sweden

[4]Departement of Computer Studies, Dar-es-Salaam Institute of Technology, Tanzania

***Abstract*—We explore techniques that can be used to reduce the power consumption of gateways in wireless sensor networks deployed in environment monitoring applications, such as Automatic Weather Stations (AWS). The challenge is the deployment of these networks in locations that are far from a consistent power source, such as a national grid. Such stations must be autonomous and power consumption must be minimized.**

We present test scenarios illustrating the impact of the suggested techniques. We explore hardware and software based methods of power reduction, assess the impact of each, the constraints to be expected and how to overcome them.

We then provide a reference implementation of a gateway in which we integrate these techniques. We show that power consumption can be reduced by 48 - 85% when comparing best and worst case scenarios. The reference implementation we provide consumes 27mW at 3V.

Keywords— Low Power; Wireless Sensor Networks; Gateway; Contiki; Automatic Weather Station; Environment Monitoring; Internet of Things

I. INTRODUCTION

Power management is one of the most important considerations in design of wireless sensor networks (WSNs). For environment monitoring in particular, low power consumption is a must-have in deployments that will be battery powered, often because they are far away from the national grid, or grid power is unreliable. Africa and many other developing countries are good examples of this case, and the penetration of grid electric power is very limited, with as low as 9% in some countries [1]. Low power design is important in these cases because it prevents early depletion of the battery, hence ensuring a higher availability of the environment-monitoring device. It is important also because it minimizes the rate at which the batteries will degrade due to repetitive charge-discharge cycles. Environment monitoring stations need to be available for as long as possible because the quantity of weather data collected has a direct impact on the quality of the subsequent weather information.

The structure of a WSN typically consists of several sensor nodes (also called motes), each including a central processing unit, a battery, a wireless communication module and an energy-harvesting unit. In an environment monitoring application, the sensors measure various environmental parameters such as temperature, solar insolation, humidity and others. The motes employ the wireless module to send out the data using a particular protocol. In our study, the IEEE 802.15.4 was used. This data is received by a designated node called the sink node. The sink node may be equipped with persistent storage, such as an SD card, or may buffer the received data in memory and transmit the data over a local network or the internet to a central repository. This transmission is made possible by an uplink device, which offers network connectivity such as a cellular modem or an Ethernet controller. The combined set-up of the sink node and uplink device is called the gateway.

Models for Internet connectivity to individual sensors in a WSN is standardized by IETF [2]. In favor of robustness and simplicity, we have rather chosen to focus on a low-complexity WSN model in which all sensor nodes are broadcasting their data to the gateway via an always-listening sink node, directly in one hop or via a relaying node just repeating the broadcast.

For all nodes, except the sink and relay nodes, a very low power-consuming design is then easy to implement. It consists of an RF-enabled microprocessor that wakes up from sleep, takes a reading, broadcasts the data and resumes the sleep state. There has been a great deal of progress in sleep current reduction over the years by many manufactures, and some devices now consume as little as 0.2µA during this state. Because of this, and the fact that in environment monitoring applications the frequency of capturing weather data is often very low, ranging from once a minute to once an hour, it is possible to design ultra-low power transmitter nodes that will last days, months or even years on a single battery.

The challenging design is that of the sink node and the uplink device, which together constitute the gateway. Provided low-power sensor nodes are chosen, the gateway is always the most power-hungry device in the deployment since the sink node is left awake to receive the transmitted reports and, depending on the needs of the application, the uplink device may always be on as well. Our experience with AWS in Africa has revealed

978-1-5386-2775-4/17/$31.00 ©2017 IEEE

that a number of gateways are always on. This characteristic leads to a high power consumption, unless sleep states are provided and radio duty cycling is implemented. Often, this necessitates use of large solar panels and corresponding batteries in battery-powered stations. In developing countries, such installations are prone to vandalism to acquire the solar panels and batteries and low life span due to battery wear from excessive charge-discharge cycles arising from the high consumption of the gateway and poor battery management [3] [4].

While such systems will function well in locations with an abundant power supply, an alternative design philosophy is needed for stations that will be deployed in remote areas for which post-deployment visits will be rare or even impossible.

The motivation for this work therefore comes from the need to investigate how the power consumption of gateways in environment monitoring WSNs can be minimized while retaining the core functionality.

The rest of the paper is organized as follows. Section II gives an overview of the related work in this area. In Section III, we list the general techniques to reducing power consumption in wireless embedded systems. In Section IV, we show experimental results from testing each technique and a discussion of the constraints designers should anticipate. In Section V, we discuss a reference implementation and an analysis of its power consumption. We conclude with section VI by giving conclusions and recommendations for further work.

II. RELATED WORK

Gateways for WSNs are the center of research in many academic and industrial circles. The parameters being investigated are many, and range from design for maintainability, testability and redundancy to others such as bandwidth, flexibility of uplink devices and power consumption. We reviewed literature in which the power consumption of the gateway was mentioned.

We looked at [5] on the design of flexible gateway to use with WSNs for countries with unreliable power networks. They propose that appropriate gateway devices for these scenarios must be low power and their solution is to use the ALIX 2, an embedded Linux single board computer. The board consumes up to 4W in their experiments.

In [6], the main objective is reducing the power consumption of the gateway by 50% with the Raspberry Pi as the reference hardware, by looking to exploit the sleep states of the device. No sleep states were found to be available. The board consumes about 1.2W. Their sink node is based on the ATMEGA256RFR2 microcontroller running the Contiki operating system, which is similar to what we used in our experiments.

The networked embedded systems research group at the Swedish Institute of Computer Science has done a lot of work in developing protocols to facilitate very low power operation of nodes in wireless sensor networks. In particular, the Contiki embedded operating system provides techniques for radio duty cycling (RDC) and has been ported to a number of architectures [7]. We discuss radio-duty cycling in our design strategies.

In [8], an outdoor agricultural environment monitoring system is implemented using a wireless sensor network. The requirements and installed peripherals are many, increasing the power consumption to about 16W, with a 200W solar panel. One of the installed peripherals is a CCTV camera to record incidences of vandalism, which has already been observed to be a major problem in AWS in developing countries [3].

Finally, there is also there is also work we found on designing an Ultra-Low Power gateway for WSNs in outdoor deployments that was carried out by Rajgarhia et al [9]. They agree that most WSN gateways are implemented using a high-end processors running operating systems and discuss the feasibility of using a low power micro-controller as the gateway. They implement at STM32-based gateway running an embedded operating system. Their results show that they achieve a peak power consumption of 1W.

The reviewed work indicates that ideal gateway for a given use-case is determined by the needs and requirements presented by that use-case and by the technology available at the time. Further, the test environments of the gateways in the literature do not consider the challenges of automated environment monitoring in developing countries, where many deployments need to be autonomous. As such, the definition of low-power is relative. The system in [9], for example, is a low power system if the power supply is from national grid. At 1W, it would take over 20 days to accumulate 1kWh of energy. For remote autonomous deployments however, 1W if often too high. At an operating voltage of 5V, 1W translates to a continuous current draw of 200mA. Factoring intermittent sunshine, nighttime and cloudy days, the solar panels and batteries needed to sustain this power consumption are quite large and lead to the issues mentioned in Section I—fast depletion of the battery and subsequent battery degradation from excessive charge-discharge cycles.

III. DESIGN GUIDELINES

A. Needs and Requirements

Before any optimizations, we listed first the needs and requirements of a WSN gateway in an environment monitoring application. Any optimized design must be evaluated against these functional requirements.

The functional requirements are:

- Receiving data from the sensors
- Appending a timestamp to data
- Buffering the data or storing on a local file system
- Transmitting the data over a network to a central repository

Thea above are all necessary and sufficient for an automated environment-monitoring device. A number of Non-Functional Requirements (NFRs) also exist, such as robustness, maintainability and manufacturer independence. A discussion of these is outside the scope of this paper, because compliance to NFRs is application-specific and their impact on power consumption is either negligible or cannot be evaluated quantitatively. The functional requirements mentioned indicate the need for RF modules for data reception, peripherals such as Real-Time Clocks (RTCs), memory devices, networking interfaces and software logic. The strategies in reducing power consumption, therefore, need interventions in all the mentioned components.

B. Strategies

The techniques for reducing power consumption of the gateway in a wireless sensor network can be categorized as either hardware or software interventions. There must also be some strategies of power reduction for the uplink device, which usually consumes more power than the sink node itself.

Hardware-level Interventions include:

i. **Selecting a low power microcontroller platform**
ii. Selecting a low power RF-module
iii. Reducing processor operating voltage
iv. Reducing processor operating frequency
v. Avoiding inefficient voltage regulators

Software-level Interventions include:

i. Implementing radio duty-cycling
    a. Use sleep/wake-up schedules in deterministic networks
    b. Use MAC level protocols in very active networks or networks with sporadic transmissions
ii. Turning off unused internal and external peripherals

Uplink interventions depend on the application. Often, power management at this level will be contingent on the frequency of data transmission over the uplink to a central repository.

IV. EXPERIMENTS

Experiments were performed for each of these strategies to confirm that power consumption actually reduces, and further to establish the magnitude of the power savings when they occur. Each experiment is presented with its results and an analysis of the results.

A. Hardware Interventions

i. Low Power Microcontroller Selection

Requirements such as hosting webservers and seamless networking support are easily achieved by using an embedded computing platform running a more or less general-purpose operating system, such as Linux. These devices are low power compared to traditional computers and achieve the same functionality. A benchmarking methodology considering several performance parameters of these single board computers is laid out in [10]. The mentioned functional requirements can, however, be achieved by most microcontroller families from a number of manufacturers. We reviewed 8-bit devices from Microchip, ST Micro and NXP Semiconductor. The most important parameter in this regard is current consumption when the microcontroller is in sleep mode. All of these have consumptions in the µA region and qualify for use in low power designs. We evaluated our design guidelines using three selected devices: ATMEGA256RFR2, ATMEGA128RFA1 and ATMEGA328. The motivation behind the first two choices were the fact that they have an in-built RF transceiver and thus provide a single chip sink-node solution and are supported by the Contiki operating system. Further, in the datasheets of all the devices we reviewed, they indicated the lowest sleep mode current consumption of less than 200nA.

Table 1: Receive current of some RF Modules used in WSNs

RF Module	Manufacturer	Current(mA)
MRF24J40	Microchip	12.5
NRF24L01+	Nordic Semiconductor	13.5
AT86RF230	ATMEL (now Microchip)	7.3 - 11.8
CC2500	Texas Instruments	13.3

ii. RF Module Selection

In the majority of the cases, sink nodes have the processor and the radio module as disjoint units. In this case, the selection of the radio module to use will have a sizeable impact. In this intervention, we were concerned mainly with the power consumption during receive mode. We reviewed a number of IEEE 802.15.4 RF modules available in the market. Table 1 shows the current consumption of some popular radio modules during receive mode only at 3.0V.

We observe that the CC2500 consumes almost twice the power consumed by the AT86RF230. The AT86RF230 is the module built into the ATMEGA256RFR2 microcontrollers and features hardware-level smart listening mode in which current consumption can be reduced by up to 50% [11]. While this table is not comprehensive, it clearly indicates that the choice of RF module can influence its contribution to power consumption significantly.

iii. Voltage Reduction

In resistive systems, a reduction of operating voltage will make the current vary linearly according to Ohm's law, $V = iR$ here V is the voltage, i is the current and R is the Resistance. In many other systems, the general principle holds except that the relationship may not be linear. Microprocessor-based systems generally consume less current when the supply voltage is reduced.

Figure 1 shows the current-voltage relationship for an ATMEGA256RFR2 microcontroller. We performed the experiment with the radio-on and with the microcontroller (MCU) in both active and sleep states for comparison. In each case, we observed a current reduction from an operating voltage of 3.0V to 2.4V. In the second case, we observed a reduction from 9mA to 7.3mA, about 19%.

Voltage reduction has two main implications. Firstly, the clocking frequency of the MCU is voltage dependent. Low voltages require a low operating frequency In a WSN, this would be a bottleneck for applications whose data throughput is in the range of Mbps.

Secondly, while many MCUs may operate in stand-alone applications at very low voltages, the peripherals connected to the MCU dictate to the designer what the minimum operating voltage should be.

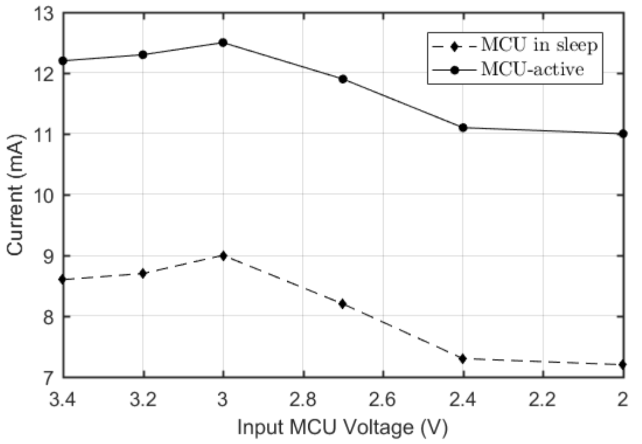

Figure 1: Current consumption of an ATMEG256RFR2 MCU at reducing voltages

For example, SD cards operate with a supply voltage of 2.7 – 3.6V [12]. In this intervention, even though we recorded the lowest power consumption at 2.4V, an application in which the received data is stored on such a card would have to operate above 2.7V.

iv. Frequency Reduction

The power consumption of a microprocessor is directly proportional to the frequency at which the processor is operated. Environment monitoring WSNs are typically low-throughput applications, with only a few hundreds of bytes being transmitted per minute**.**

As such, it is possible to operate the processors in these applications at very low frequencies. We tested this intervention with an ATMEGA328P microcontroller with a MRF24J40 radio. Figure 2 shows a comparison of running the ATMEGA328, with the radio off, at 16MHz and 4MHz. We observed a saving of up to 4.5mA while running the setup at 3.6V. With the radio on, the consumption was about 20.5mA at 16MHz indicating that the power savings from this intervention amount to about 22%.

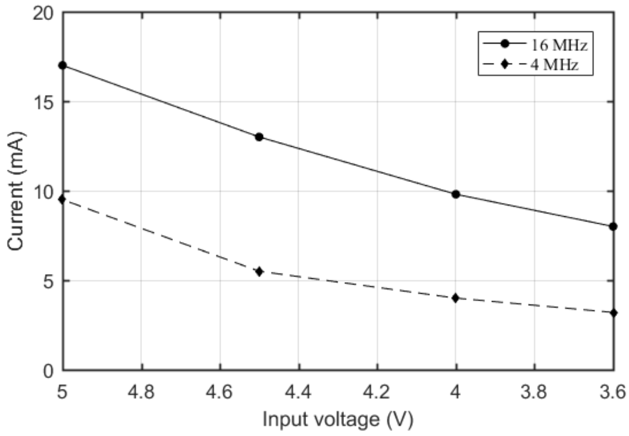

Figure 2: Current consumption of an ATMEGA328P in active mode at 16MHz and 4MHz

The downside to reduced frequency operation is loss of accuracy in timing. At 16MHz, a processor clock cycle is equal to 62.5 nanoseconds. At 4MHz, a cycle is 250ns. This will affect peripherals like the Analog to Digital Converter. Microcontroller ADCs have a minimum frequency at which they should operate. If their clock frequency is too low, the internal capacitor over which the voltage is measured will discharge too much and the measurements will be inaccurate.

v. Efficient Voltage Regulators

In many designs, the use of a voltage regulator is inevitable—either because the designer would like to implement intervention (iii) to run the system at a specific voltage, or the battery technology best suited for that given application has a high (or low) nominal voltage and cannot be connected directly to the processor.

Voltage regulators consume a quiescent current even when not actively supplying power to the load. Compared to typical consumption of the microcontrollers we reviewed, some quiescent currents are unacceptable for low power designs.

For example, we measured the current consumption of two SD card modules—one with an on-board AMS3117 regulator to convert from 5V to 3.3V for the actual SD card and another was a bare-minimum module without any active electronics; just direct pin connections. Table 2 shows the currents measured in the absence of any write or read operation, and the contribution to the total current consumed. If a regulator is inevitable, designers should choose switching regulators or efficient low-dropout linear regulators (LDOs).

To show the impact of using LDOs, another experiment on this technique was performed with the Radio Sensors Model-S2 board [13] that features the ATMEGA256RFR2 MCU. The board uses the LP2950 [14] LDO to achieve 3.0V to supply to the MCU and the rest of the peripherals. This LDO consumes a very low current when not active. We measured a quiescent current of 89µA. This is very low compared to the ~5mA of the AMS3117 used on popular platform boards such as Arduino. The designer can also choose to eliminate regulators. The constraint here is that the battery technology must be chosen appropriately. Many microcontrollers operate in the 1.8-5.5V range. There are a number of battery technologies that have nominal voltages in this range but this parameter alone is not enough. The battery must also have a fairly flat discharge curve, such that the voltage doesn't change much as charge is drained by the load. Lithium-ion and Lithium iron phosphate ($LiFePO_4$) batteries are good contenders when using this approach.

Table 2: Comparison of current consumed by two SD card modules with and without a regulator

COMPONENT	CURRENT (mA)	MCU CURRENT (mA)	CONTRIBUTION
SD Module with AMS3117 Regulator	4.98	8.5	37%
SD Module - Bare	0.198	8.5	2.2%

B. Software Interventions

The interventions discussed so far focus on power reduction outside the microprocessor unit. There are software-level interventions that can be implemented in firmware to reduce the power consumption even further, especially in WSNs.

i. Radio duty-cycling

The radio transceiver in a sink node typically consumes more power than the microcontroller unit. In environment monitoring, transmitter nodes are in the sleep state most of the time. As such, an ever-listening sink node will waste power. To realize a long lifetime in battery-powered deployments, the radio in the sink node must be switched off as much as possible. However, when the radio is off, the sink node is not able to receive any messages. As such, it must managed in a manner that allows it to receive messages while being kept in the off-state when no message is available from the transmitter nodes. Radio duty-cycling refers to the process by which the sink nodes radio is switched off when a transmission is not expected [7].

In our research work, we have an automatic weather station measuring nine parameters with three transmitter nodes and one sink node. All data is contained in five reports, each transmitted once a minute. The duration of a transmission from an ATMEGA128RFA1 radio transceiver is about 20ms [15] so the sink node is only actively receiving data for about 100ms every minute and is thus inactive for over 99% of the time.

In an application in which there are a number of messages being sent out in a unit of time, the transmission times are either known or unknown. In the case where the times are known, the sink node can establish and maintain schedule of these arrival times and turn on the radio just before an expected transmission and go to sleep after the message has been captured successfully. In applications where the messages on the network may be sent out randomly, the transmitter and receiver must communicate to establish a rendezvous point. Embedded operating systems such as Contiki provide several mechanisms of achieving this. These are often called MAC level protocols. For example, in the ContikiMAC protocol [7], the sink node turns on its radio several times a second to poll the medium for activity. If activity is detected, the radio is kept on longer to receive the full frame.

We did not find any reliable implementation of a MAC level protocol for the platforms we chose and we implemented this intervention using a system-level schedule based on deterministic transmission times to illustrate its impact.

The experiment set-up involves a receiver node supplied with an RTC module to keep track of time. In this type of scheduling, when the node is first powered on, it will remain on for a given amount of time until all transmitter nodes have sent in a message. For example, if the transmitter nodes transmit once per minute, the sink node remains on for one minute the first time it is powered. The arrival times of all messages in the network are captured. Since the inter-transmission times are known, a schedule can be established with this information only.

We programmed the transmitter to send out messages every 10 seconds and MCU to wake up 1 second before the anticipated arrival. In 60 seconds, this came to approximately 10s of activity. The current consumed during active time was 12.5mA and 638µA with the radio switched off.

The average weighted current in this case is given by:

$$12.5 \times \frac{10}{60} + 0.638 \times \frac{50}{60} = 2.6\, mA \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

We chose to wake up one second for proper visualization of the currents with human eyes and as proof of concept. However, the transceiver wake up time is of the order of microseconds. Hence, the total time the MCU is awake can be reduced to sub-millisecond values, in which case the savings would be much higher.

There are two constraints we observed with this technique.

One, a poor clock will throw the system off schedule and some transmissions will be missed. RTCs like the DS1307 [16] have an accuracy of ±1 minute per year but we have lost up to 5 minutes in a month in outdoor deployments. We have observed much better accuracy with the DS3231 [17]. Moreover, external RTCs have a resolution of only 1 second. Internal clocks provided by the microcontroller have sub-millisecond resolutions but would need to be given a separate power source, since they would reset if the main power source runs out.

Two, the transmitter nodes themselves can be thrown off schedule by a number of reasons, such as loss of power or timing jitter. The only way to overcome this is to re-establish a schedule with the neighboring nodes after an arbitrary amount of time.

It would seem that implementing any of the MAC level protocols, such as those provided by the Contiki operating system is the better solution, as this enables the transmitter and receiver to always establish a rendezvous point even with the constraints mentioned above. In addition, in very active or dense WSNs, establishing rendezvous points by clock synchronizations consumes more power than MAC level protocols [18] [19]. The major drawback of using these protocols is that the power consumption of the transmitter nodes is increased, since they have to poll the medium frequently to establish contact with the sink node.

ii. Switching off Unused peripherals

Most manufacturer datasheets provide power consumption values for their MCUs with no peripherals active. The additional current drawn by the peripherals must obviously be considered since all MCUs have peripherals and their share of the total power drain is quite significant [20]. If the MCU is going to be put to sleep as recommended, the internal peripherals will be switched off by default. External peripherals, such as external flash memory, active sensors and even uplink devices can be powered down when not in use through a MOSFET and powered up when needed.

In our experiments, the reward for using this intervention was much smaller than the effort involved. The RTC and SD card modules already consume less than 600µA combined. The uplink device, which can be turned on and off in this manner, has been given its own section.

C. *Uplink Interventions*

The other important part of the gateway, the uplink, is the most challenging. This is because, on its own, it typically consumes several magnitudes more power than the sink node. Power management for the uplink device will depend mainly on the availability of low power operation modes and on the needs and requirements of the environment monitoring system being designed.

In many cases, weather data is not immediately consumed by its intended recipients. Such a requirement may provide a window of opportunity to save power in the gateway, by putting the uplink device to sleep or turning it off completely. Compared to microcontrollers, the sleep-mode power consumption of most uplink devices is quite high. We looked at several uplink options available for use in WSNs and their typical power consumptions. We investigated cellular modems, Ethernet and WiFi modules and a custom uplink device, in which a linux-based embedded computer, the Raspberry Pi Zero, was connected to a 3G USB modem. Table 3 summarizes our findings.

In many developing countries, deploying stations with Ethernet or Wi-Fi in remote areas is very difficult. Our experiment thus concentrated on cellular uplinks.

The power consumption is affected by the data upload frequency and the boot time of the uplink device.All the cellular options we tested showed an average boot time of ~15s, in which there was a variable current consumption of 80-150mA. If the device is to be turned on once a minute, 15 seconds will account for 25% of the total time in which power is consumed, yet no actual data is being uploaded. If the upload interval is once an hour, the boot time accounts for only 0.4%.

These design decisions are application-specific and must be made following the needs and requirements. It is evident though that waiting longer hours to transmit saves more power.

V. RESULTS

We did an integration of the hardware-level interventions we have discussed and provided an implementation to assess the power consumption. The implementation is based on the ATMEGA256RFR2 sink node called the RS2 Mote from Radio Sensors AB in Sweden [13]. The application was designed to achieve the functional requirements—receive wireless frames, append a timestamp from a Real-Time Clock (RTC) and save to persistent memory (SD Card).

Table 3: Active and Sleep mode current consumption of some common uplink options.

Device	**Interface**	**Current –Active (mA)**	**Current-Sleep (mA)**
SIM800L	Cellular-2G	80	5
SIM5320E	Cellular -3G	100	12.9
ENC28J60	Ethernet	120-160	1.2
ESP8266	WiFi	85	0.077
Huawei E173S + Raspberry Pi Zero	Cellular - 3G	150	N/A*

**Sleep mode is not available on the Raspberry Pi.*

The mote runs the Contiki embedded operating system. We used a DS3231 RTC to provide system time and a bare-minimum SD card module for persistent storage. The sink node features a 16MHz external oscillator and a low drop-out regulator that provides 3.0V to the whole board.

Table 4 shows the power consumption improvement we achieved by following only the hardware-level strategies in the worst and best case scenarios. In the best case scenario, we supply 3.0V at 16MHz and take advantage of the hardware-provided smart-listening feature. We achieved a current of 9.0mA at 3.0V (27mW), compared to 17.4mA without optimizations, which corresponds to a saving of 48%.

We excluded the uplink in this test because the data upload interval, in practice, can be chosen such that the consumption is negligible in comparison to the sink node. For example, in our field test, five sensor nodes transmit 200 bytes each once a minute. This equals about 1.4 MB per day. We established a transmission time of 17ms for one 200-byte report. This gives a transmission time of about 122s if we upload data once a day. Using an average consumption of 150mA during upload, and 9mA consumed by the sink node in the remaining ~777,000 seconds of the day, this would account for only 2%.

We powered the set-up using was a standard 18650 Li-ion battery with 2200mAh rating without a recharge source. We achieved a battery lifetime of about 228 hours (9 days and 12 hours).

At the time of writing, we did not have field data for the radio duty cycling implementation. However, we can interpolate the laboratory results to get an estimation. From equation (1) showing the average current consumption with the radio duty cycling implementation, we see that the total power consumption reduction is increased to

$$\frac{17.4 - 2.6}{17.4} = 85\%$$

Further, the same battery should power the system for

$$\frac{9}{2.6} \times 228 = 789\ hrs \approx 33\ days$$

Table 4: Power consumption improvement using hardware interventions only

Current (mA)	MCU	RTC	SD	TOTAL
No optimization	12.5	0.4	4.98	17.4
With optimization	8.5	0.4	0.198	9.0
Reduction				**8.4**

VI. CONCLUSIONS AND FURTHER WORK

We have listed guidelines that need to be followed in practice to reduce power consumption of the gateway device in a Wireless Sensor Network for environment monitoring. We have shown that, compared to designs with no intervention at all, at least 48% power saving can be achieved with hardware interventions alone and >85% using radio duty cycling. The impact of how often data is uploaded by the uplink device has also been demonstrated. Given any needs and requirements, designers can implement the strategies that matter more given their requirements. We see numerous research directions to follow to reduce the gateway power consumption even more in WSN based automatic weather stations.

The way forward in this regard to explore the impact of radio duty-cycling is to implement the ContikiMAC RDC protocol for the ATMEGA256RFR2, which has already been proven to be attain very low power consumption with the hardware interventions alone.

Alternative means for waking the sinknode using ultrasonic signals have been shown to achieve sub-µA current consumption [21]. The ultrasonic sensors consume very little current and can be always left powered on instead of the radio. This is an interesting intervention and what needs to be investigated is how effective it would be when transmitter nodes are far away from the sink node.

The authors are currently involved in a number of field tests studying these issues.

REFERENCES

[1] International Energy Agency, "Africa Energy Outlook," International Energy Agency, Paris, 2014.

[2] Internet Engineering Task Force, September 2007. [Online]. Available: https://tools.ietf.org/html/rfc4944.

[3] M. Nsabagwa, M. Byamukama, J. Sansa-Otim and R. Okou, "Network Densification Strategies for Automatic Weather Stations: Challenges and Opportunities for Uganda," in *2016 IST-Africa Week Conference*, Durban, 2016.

[4] D. Rogers and V. Tsikunov, ". Weather and Climate Resilience: Effective Preparedness through National Meteorological and Hydrological Services," World Bank Publications, 2013, p. 135.

[5] M. Zennaro, H. Ntareme and A. Bagula, "On the design of a flexible gateway for wireless sensor networks," in *1st Workshop on Wireless Broadband Access for Communities and Rural Developing Regions*, Karlstad, 2008.

[6] R. Olsson, "Green Gateway Project," [Online]. Available: https://www.kth.se/social/files/55e602dff2765422f115a3c7/robert-csd-project.pdf . [Accessed 10 June 2016].

[7] Swedish Institute of Computer Science, "Radio duty cycling," [Online]. Available: https://github.com/contiki-os/contiki/wiki/Radio-duty-cycling. [Accessed 1 Feb 2017].

[8] C. S. H. Y. Jeonghwan Hwang, "Study on an Agricultural Environment Monitoring Server System using Wireless Sensor Networks," *MDPI Sensors,* vol. 10, no. 12, pp. 11189-11211, 2010.

[9] P. Rajgarhia, P. James, V. Nainwal, P. Sowjanya and S. S. Koshy, "Development of an Ultra-low power WSN gateway for outdoor deployments," in *ICICES*, Chennai, 2014.

[10] A. Nungu, R. Olsson, J. Guo and B. Pehrson, "Towards single Watt and nJoule/bit routing," in *UbuntunetConnect2014*, Lusaka, 2014.

[11] ATMEL Corporation, [Online]. Available: http://www.atmel.com/images/atmel-8393-mcu_wireless-atmega256rfr2-atmega128rfr2-atmega64rfr2_datasheet.pdf. [Accessed 20 Feb 2017].

[12] SD Group, 22 Janury 2013. [Online]. Available: https://members.sdcard.org/downloads/pls/simplified_specs/part1_410.pdf. [Accessed 22 April 2017].

[13] Radio Sensors AB, [Online]. Available: www.radio-sensors.com. [Accessed 1 August 2016].

[14] On Semiconductor, "LP2950 DataSheet," [Online]. Available: https://www.onsemi.com/pub/Collateral/LP2950-D.PDF. [Accessed 5 Feb 2017].

[15] R. Olsson and B. Pehrson, "Powering Devices Using Ultra-capacitor batteries," in *AFRICON2015*, Addis Ababa, 2015.

[16] Maxim Integrated, [Online]. Available: https://datasheets.maximintegrated.com/en/ds/DS1307.pdf. [Accessed 1 March 2017].

[17] Maxim Integrated, [Online]. Available: https://datasheets.maximintegrated.com/en/ds/DS3231.pdf. [Accessed 1 3 2017].

[18] V. Jelicic, M. Magno, D. Brunelli, V. Bilas and L. Benini, "Analytic comparison of wake-up receivers for WSNs and benefits over the wake-on radio scheme," in *7th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*, New York, 2012.

[19] E. Y. A. Lin, J. M. Rabaey and A. Wolisz, "Power-efficient Rendez-vous schemes for dense wireless sensor networks," in *IEEE International Conference on Communications*, Paris, 2004.

[20] ATMEL Corporation, "Innovative Techniques for Extremely Low Power Consumption," [Online]. Available: http://www.atmel.com/images/doc7903.pdf. [Accessed 1 2 2017].

[21] E. Lattanzi, M. Dromedari, V. Freschi and A. Bogliolo, "A Sub-A Ultrasonic Wake-Up Trigger with Addressing Capability for Wireless Sensor Nodes," *ISRN Sensor Networks,* vol. 2013, 2013.