

# NITOS Energy Monitoring Framework: Real time Power Monitoring in Experimental Wireless Network Deployments

Stratos Keranidis<sup>⊕</sup>, Giannis Kazdaridis<sup>⊕</sup>, Virgilios Passas<sup>⊕</sup>,  
Thanasis Korakis<sup>⊕</sup>, Iordanis Koutsopoulos<sup>⊕</sup> and Leandros Tassioulas<sup>⊕</sup>

<sup>⊖</sup>Department of Computer and Communication Engineering, University of Thessaly, Greece

<sup>+</sup>Centre for Research and Technology Hellas, CERTH, Greece

<sup>⊓</sup>Department of Computer Science Athens University of Economics and Business, Greece  
{efkerani, iokazdarid, vipassas, korakis, leandros}@uth.gr, jordan@aueb.gr

## ABSTRACT

Development of energy-efficient protocols and algorithms requires in-depth understanding of the power consumption characteristics of real world devices. To this aim, energy efficiency analysis is performed by the research community, mainly focusing on the development of power consumption models. However, recent studies [1] have highlighted the inability of existing models to accurately estimate energy consumption even in non-composite scenarios, where the operation of a single device is analyzed. The inability of such models is further highlighted under real life scenarios, where the impact induced by the simultaneous operation of several devices renders the application of traditional models completely inappropriate. As a result, energy efficiency evaluation under complex configurations and topologies, needs to be experimentally investigated through the application of online monitoring solutions. In this work, we propose the innovative NITOS Energy consumption Monitoring Framework (EMF) able to support online monitoring of energy expenditure, along with the experiment execution. The developed framework is built on a distributed network of low-cost, but highly accurate devices and is fully integrated with the large-scale wireless NITOS testbed. The framework evaluation is performed under both low-level experiments that demonstrate the platform's high-level accuracy, as well as through high-level experiments that showcase how online and distributed monitoring can facilitate energy performance assessment of realistic testbed experiments.

## 1. INTRODUCTION

The unprecedented penetration of “smart“ mobile devices in everyday use cases, has greatly affected the trends followed by vendors developing such equipment. First, the need for offering ubiquitous network coverage, has led the indus-

try in equipping these devices with several wireless interfaces (WiFi, Bluetooth, 3G, LTE, WiMAX), to facilitate parallel network access. Second, in an effort to meet the increasing requirements generated by the use of resource-demanding applications, high-end mobile devices feature multi-core processors, high-resolution displays and support increased data rate communication technologies. Especially in the case of smartphone platforms, the energy greedy profile of the supported state-of-the-art wireless technologies may induce up to 50% of the total platform power consumption [2], under typical use case scenarios. The increased energy demands of such technologies cannot be successfully met, due to the limited energy capacity [3] that existing battery technologies are able to offer.

The overall goal, towards alleviating this unique performance discrepancy, is to reduce energy consumption wherever possible. Towards this goal, several recent research studies [4, 5, 6, 7] in the field of wireless networking have focused on reducing the total amount of energy consumed during the wireless medium access and communication operations. In this context, accurate energy consumption assessment needs to be applied by the research community, as a means of evaluating the energy efficiency of the proposed protocols and architectures. Researchers working on wireless sensor networks can base their evaluations on detailed low level specifications [8, 9] provided by developers of widely adopted sensor platforms (Tmote Sky, MICAz). However, this is not the case with vendors that develop wireless transceivers for everyday use devices, such as laptops, smartphones or tablets, where only limited information on nominal consumption is publicly provided [10]. Even in cases that accurate data sheet specifications are available, power consumption models that are based on such accurate measurements, fail to successfully calculate energy expenditure under complex configurations and topologies.

In order to enable experimenters to accurately evaluate the energy efficiency of the proposed protocols, under real world scale and settings, advanced methodologies and solutions need to be developed. In this work, we propose the innovative NITOS EMF framework that is fully integrated with the large-scale wireless NITOS testbed [11] and provides for online gathering of energy measurements, through a distributed network of low-cost, but highly accurate devices.

## 1.1 Related Work

As previously stated, a great variety of research efforts has proposed energy-efficient protocols and architectures, towards moving to “greener” solutions in telecommunications. Mechanisms proposed in recent works range from scheduling of sleep intervals and antenna configurations [12], to reduction of time spent during idle listening periods [4] and application of adaptive transmit power and physical layer (PHY) rate control [5]. The approaches above, jointly follow the methodology of first identifying key functions that exacerbate energy expenditure and subsequently attempt to control the induced impact by efficiently scheduling their operation. Other relevant works focus more on energy consumption characterisation of specific technologies, such as the power-hungry IEEE 802.11n [13], or specific platforms, such as the energy-limited smartphones [6, 7]. Trying to address the limited level of detail provided by the industry, several works [14, 2] present extensive measurements that assess the impact of low-level configurations on the overall power consumption of various platforms. In order to gather such detailed measurements, either commercial sensing hardware or custom measurement setups are employed, which result in varying levels of obtained accuracy and reliability.

In an effort to support evaluation under definite measurement setups, a limited variety of approaches have developed proper energy consumption monitoring frameworks for testbeds that specifically target wireless sensor networks [15] or data centers [16]. Similarly, the work in [17] proposes a framework developed for energy consumption monitoring of real WLAN deployments. However, this framework is restricted in characterising the consumption at the level of complete device (router). The importance of monitoring consumption of both the wireless network interface card (NIC) and the complete device as a total, in deriving hitherto unexploited tradeoffs, is highlighted in [1]. Based on comparison of energy consumed at the transceiver and the complete device level, this work proposes a novel energy model that contrasts traditional approaches, which neglect the fraction of energy consumed while each frame crosses the protocol stack.

This paper is organized as follows. In sections 2,3 and 4, we present the required platform specifications, the followed measurement methodology and detail the various framework components accordingly. In section 5, we present extensive experiments that evaluate the platform’s capabilities, while in sections 6 and 7 we discuss our future work and point out conclusions accordingly.

## 2. PLATFORM SPECIFICATIONS

The overall goal of the proposed work is to turn the rather challenging task of online energy consumption monitoring into an automated procedure that is available to wireless testbed experimenters. Below, we list the various required platform characteristics, along with the implementation choices that we followed, in order to integrate them with the developed framework:

- **Non-intrusive operation:** The normal network operation should not be affected by the monitoring procedure, in order not to result in imprecise results. Our solution runs on external hardware, which does not interfere with the measured devices.
- **Online monitoring:** Online monitoring needs to be

applied, towards energy efficiency evaluation under complex configurations and topologies. The proposed framework allows for online gathering of measurements in parallel with execution of long-term experiments.

- **Distributed architecture:** Assessment of the impact induced by simultaneous operation of collocated network components, requires the development and application of distributed energy consumption monitoring solutions. The proposed framework is composed of distributed communication enabled components.
- **High sampling rate:** State-of-the-art wireless technologies are capable of high transmission rates, which generates the necessity for energy monitoring devices to feature high sampling rate components that allow monitoring of short duration events (in the order of  $\mu$ s). The developed platform achieves twice the minimum required sampling rate.
- **High sampling accuracy:** Wireless testbed infrastructure may operate in states that result in similar energy consumption, thus necessitating the application of high accuracy sampling equipment able to distinguish between the energy consumption induced in such operational states. The custom developed hardware achieves accuracy in the order of milliwatts.
- **Adaptive to heterogeneous infrastructure:** Due to the existence of several heterogeneous types of interfaces, protocols and architectures, connectivity of the developed solutions with the components under test should be easily setup in all cases. The followed measurement procedure is rather generic and allows for power consumption monitoring of any device type.
- **Low-cost hardware:** The extended scale and increasing scalability of modern experimental infrastructure require that the developed hardware components are of low cost to allow for the distributed deployment of several monitoring devices. The developed hardware solution introduces a total cost of less than €80.

## 3. MEASUREMENT METHODOLOGY

In order to estimate the instantaneous power consumption of any device, we follow a widely adopted power measurement procedure, which requires the placement of a high precision, low impedance current-shunt resistor ( $R$ ) of a known resistance value, in series with the power source and the power supply pin of the device to be measured. The exact measurement setup described above is presented in Fig. 1.

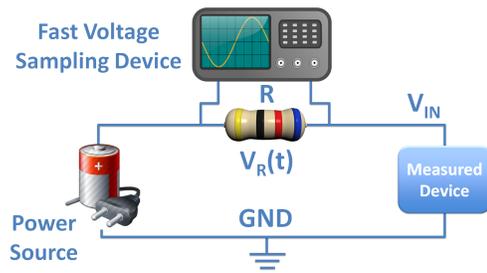


Figure 1: Representation of Power Measurement setup

By consistently measuring the voltage ( $V_R(t)$ ) across the current-shunt resistor through proper voltage metering equipment, we are able to extract the instantaneous current draw

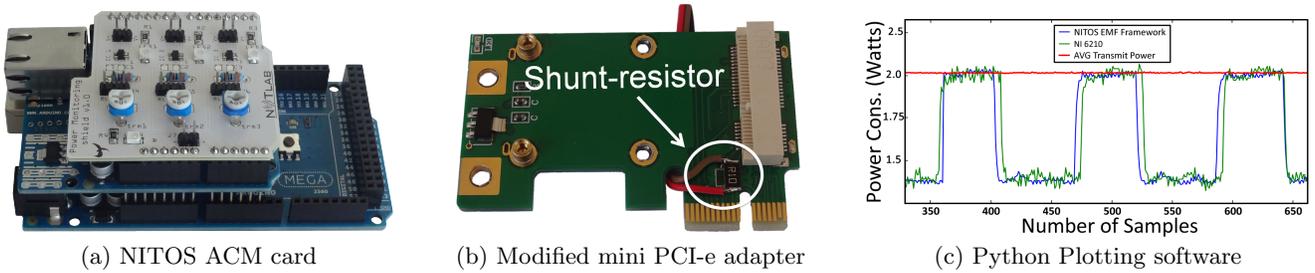


Figure 2: NITOS ACM card and accompanying hardware and software components

of the device, based on Ohm’s law. The instantaneous power consumption can be calculated as the product of the input voltage  $V_{IN}$  and the measured current draw:

$$P(t) = V_{IN} \frac{V_R(t)}{R} \quad (1)$$

In order to estimate the total energy consumption during specific events, we first need to estimate the event’s duration. Total energy consumed over an interval  $[t_0, t_1]$  is the integral of power consumption over the specified time duration ( $Dt = t_1 - t_0$ ), calculated as:

$$E(Dt) = \frac{V_{IN}}{R} \int_{t_0}^{t_1} V_R(t) dt \quad (2)$$

For instance, the duration of a single frame transmission or reception can be directly obtained as the product of the frame length and the configured PHY-layer bit rate. Subsequently, energy consumption can be obtained as the integral of the power consumption over the calculated duration. However it should be made clear that through the voltage sampling equipment, only a finite number of samples of  $V_R(\cdot)$  are acquired over  $[t_0, t_1]$  at discrete time instances.

## 4. NITOS EMF FRAMEWORK

NITOS EMF, which integrates both hardware and software components, manages to implement the aforementioned power measurement approach. Below, we detail the platform characteristics and describe how the framework has been integrated with the testbed architecture.

### 4.1 Hardware Components

The developed hardware device, is built on top of the first version of the NITOS Chassis Manager (CM) Card [18], which was initially used to control the operational status of testbed nodes. The advanced version of the card (*NITOS ACM*), which is presented in Fig. 2(a), is mainly composed of Arduino compatible open-source components, but also features custom developed hardware. The various hardware components are detailed below:

#### Arduino Mega 2560

The developed card is based on the low cost Arduino Mega 2560 board, featuring the ATmega2560 [19] 8-bit microcontroller that runs at 16 MHz and operates at 5 Volts. The ATmega2560 integrates a 16 channel Analog to Digital Converter (ADC), with a resolution of 10-bit (i.e. 1024 different values), to provide for sampling of analog signals. We use the integrated ADC to sample the voltage across the shunt resistor as presented in Fig. 1. We decided to use the Arduino Mega 2560, as it offers 256 KB of flash memory and 8 KB of SRAM, which features are required for the efficient operation of the developed software components.

#### Ethernet Shield with Micro SD card

In order to provide for distributed measurements, the card should feature network communication capabilities. Towards this goal, we decided to equip the card with the Arduino Ethernet Shield [20], which features the embedded Wiznet W5100 network controller that implements a network (IP) stack capable of both TCP and UDP communication. Another key characteristic of this shield, is the embedded Micro SD card module that provides the board with external storage capabilities, enabling for long-term data logging.

#### Custom Shield integrated with the INA139 IC

Since the integrated ADC is not able to accurately digitize the attained voltage levels on shunt resistors in cases where the monitored voltage drop is minimally varied (mV range), we equipped the developed card with the Texas Instruments INA139 [21] Integrated Circuit. INA139 is a high-side current-shunt monitor that converts a differential input voltage to an amplified value, where the amplification level is controlled through an external load resistor ( $R_L$ ) and can be set from 1 to over 100. The amplification accuracy of the INA139 Integrated Circuit (IC) is directly dependent on the selection of the current-shunt ( $R$ ) and load resistor ( $R_L$ ) values.

In order to decide about the proper value of the shunt resistor, we have to consider the average consumption of the device that will be measured. Considering that commercial wireless NICs have an average consumption of 2 Watts we select to use a shunt resistor of  $0.1 \Omega$ , which attains 60 mV of shunt voltage that is within the specified limits [21]. Considering the configuration of the  $R_L$  resistor, we decided to use a  $30 \text{ K}\Omega$  resistor, which meets the maximum output voltage requirement of 2.725 Volts that is specified in the INA139 data sheet. Having properly configured the resistor values that control the INA139, we then designed and fabricated a Printed Circuit Board (PCB), which can be directly integrated on the Arduino board. The designed PCB, which can be seen on top of the Arduino hardware in Fig. 2(a), features three individual INA139 and all the required electrical components, providing for power consumption monitoring of multiple devices.

#### Custom mini-PCIe adapter

Having decided about the proper value of the current-shunt resistor, we next had to attach it in series with the power supply pins of several wireless NICs. In order to ease the interception of the power supply pins and refrain from modifying each different type of NIC, we followed a more applicable approach and inserted the current-shunt resistors on communication bus adapter cards. Fig. 2(b) illustrates a modified pci-e to mini pci-e adapter card that is attached with a high-precision current-shunt resistor of  $0.1 \Omega$ .

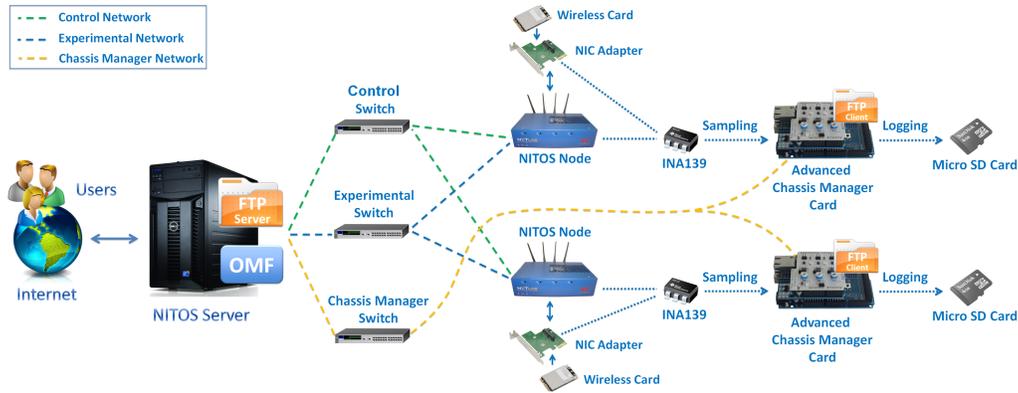


Figure 3: Integration of NITOS EMF framework with the overall testbed architecture

## 4.2 Software Components

Towards rendering the described hardware platform into a functional framework, we developed appropriate software to control the ACM cards and integrate them with the testbed.

### Arduino Software

The sampling rate of approximately 9 KHz that is supported by the default Arduino configurations, is not sufficient for sampling events that last for time intervals in the order of  $\mu$ s. We consider transmissions of typical MPDU frames of 1534 bytes length as the shortest in duration event, which requires approximately  $27\mu$ s to be transmitted at the  $TX_R$  of 450 Mbps. As a result, we need to achieve sampling rate higher than 37 KHz, in order to be able to monitor such transmissions.

To overcome this issue, we modified the default ATmega2560 ADC configurations that control the achievable sampling rate. First of all, we configured the ADC to operate in the free-running mode, which enables the ADC to continuously perform conversions without requiring proper signalling from the  $\mu$ C. Through this modification, we enable SD card logging of sampled data to take place in parallel with ADC conversions, efficiently increasing the amount of time spent in sample acquisition. Moreover, we properly modified the clock speed of the ADC prescaler from the default value of 125 KHz to 1 MHz, following the approaches in [22, 23]. Finally, we efficiently exploited the available SRAM of ATmega2560 to directly store up to 13 blocks of 512 bytes buffers into the SD card. Based on the aforementioned modifications we manage to achieve the remarkably increased sampling rate of 63 KHz, with 10-bit resolution, while only reducing the perceived accuracy by approximately 11% [24]. Additionally, we modified the default ADC voltage reference to 2.56 Volts to enable conversions of higher accuracy. The developed platform has been evaluated in comparison with the high-end NI-6210 data acquisition (DAQ) module [25] and proved of providing measurements of similar accuracy in the range under consideration.

Towards providing for remote control of the distributed cards, we developed a tiny Web Server that is based on the Arduino Ethernet Library and operates on each individual ACM card. Through the transmission of custom UDP packets, we can remotely trigger the measurement acquisition procedure. Furthermore, we developed an FTP service that provides for collection of captured data in a distributed way.

### Python Software

We also developed a set of Python scripts that enable direct

access to the collected results and moreover precise power and energy consumption calculations. Fig. 2(c) presents the implemented plotting component and also depicts a comparison between measurements gathered through the developed framework and the high-end NI-6210 device. The overall set of developed software components is publicly available for users of the NITOS testbed.

### Integration with OMF

To enable ease of use of the developed framework, we integrated its functionalities into the OMF cControl and Management Framework [26]. Based on this integration, experimenters can fully configure the operation of the ACM cards and moreover collect and access the gathered measurements through the OMF Measurement Library [27].

## 4.3 Framework Architecture

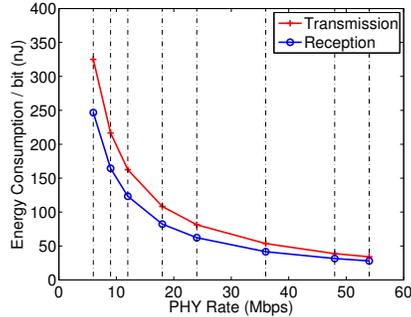
The proposed framework has been directly integrated with the underlying network architecture of the NITOS testbed.

### 4.3.1 NITOS Testbed

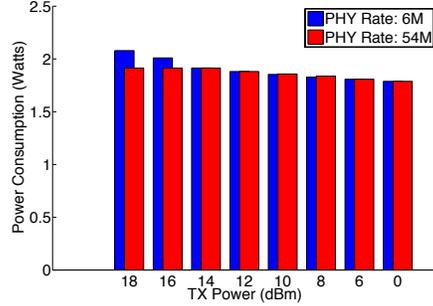
NITOS testbed currently offers 50 wireless nodes and provides open remote access to any researchers who would like to test their protocols in a real-life wireless network. The testbed architecture is illustrated in the left part of Fig. 3. Two Gigabit Ethernet switches interconnect the nodes with NITOS server, namely the *Control* switch that provide for control of experiment execution and measurement collection and the *Experimental* switch, which can be used for conducting wired experiments. A third Gigabit Ethernet, namely the *Chassis Manager* switch, is dedicated in controlling the operational status of the nodes through the transmission of custom http requests that control solid state relays on the Chassis Manager cards. NITOS nodes feature up to 3 wireless NICs, using the Atheros AR5424 and AR9380 chipsets.

### 4.3.2 Integration of NITOS EMF with the Testbed

Currently 20 of the nodes are attached with ACM cards, which together with the modified mini-PCIe adapters and node power supplies, enable for online energy consumption monitoring in a distributed way. The integration of the NITOS EMF framework with the overall testbed architecture is illustrated in Fig. 3. NITOS ACM cards are properly configured to monitor both the consumption of the NIC in an individual way, as well as the total consumption of each node. Through the *Chassis Manager* switch and the developed FTP service, measurements logged locally in the Micro SD card module of each individual ACM card are transferred in a distributed way to the NITOS server.



(a) Varying PHY-layer TX Rate



(b) Varying TX Power

Figure 4: Consumption of AR5424 NIC during frame transmission across different configurations

## 5. EXPERIMENTAL EVALUATION

Based upon NITOS testbed that implements the proposed framework, numerous experiments were conducted, and the results obtained are analyzed in this section. The first set of experiments has been designed to demonstrate the ability of the framework to accurately monitor the effect induced by low-level configurations in the overall consumption of wireless NICs. As the range of the available low-level configurations that can highly impact energy consumption is rather extended, we detail our experimental results and findings in a technical report [28] and only present a representative sample of the obtained results in this work.

Through the second set of experiments, we aim at showcasing how the online and distributed sensing capabilities of the proposed framework can aid towards realistic evaluation of wireless protocols in terms of energy efficiency. In this context, we conduct two experiments that implement an everyday life scenario of simultaneous file uploading by multiple peers, under varying configurations and channel conditions. In parallel with the experiment execution, we monitor the power consumption of each node’s NIC and thus we manage to get a high level perception of the impact that the varying conditions and the simultaneous operation of multiple nodes induce in the overall energy consumption. We present these two sets of experiments in the following section and organise them in two different groups, namely the low-level and the high-level ones.

### 5.1 Low level Experiments

The experimental setup in this first set of experiments consists of just two communicating nodes that operate on the vacant Ch.36 of the 5 GHz band. In this setup, we adjust specific PHY-layer configurations at the transmitter node, while we constantly monitor the energy consumption of the individual wireless NICs at both the transmitter and receiver side. Based on off-line processing of the collected results, we isolate specific events, such as frame transmission or reception and average multiple of them, in order to characterize their instantaneous power consumption, under each specific PHY-layer configuration. In the following experiments, we use two different Atheros chipsets, namely the AR5424 and AR9380 that implement different versions of the IEEE 802.11 protocol. The AR5424 was released in 2005 as the first complete single-chip of Atheros that supported the IEEE 802.11 a/g protocol, while the AR9380 chipset is currently the state-of-the-art IEEE 802.11n compatible Atheros chipset, featuring 3 RF chains and supporting up to three transmit and receive spatial streams (SS).

Chipset	AR5424		AR9380	
Antennas	1x1	1x1	2x2	3x3
Mode	Power Consumption (Watts)			
Sleep	-	-	0.12	-
Idle	1.47	0.49	0.56	0.69
Receive	1.52	0.62	0.74	0.85
Transmit	1.97	0.98	1.75	2.45

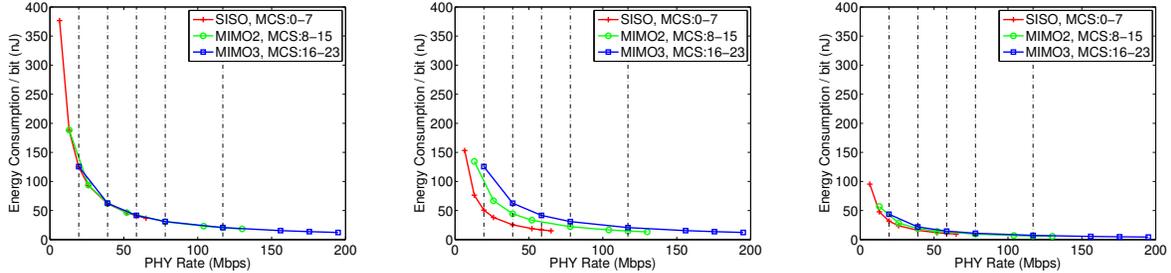
Table 1: Power consumption of AR5424 and AR9380 NICs across different operational modes

We start by characterising the power consumption profile of the two different wireless chipsets, across various operational modes, and present the collected results in Table 1. Based on the collected results, we observe that the AR9380 chipset consumes much less power than the AR5424 chipset across all operational modes. This observation comes in contrast with the increased power consumption that multiple-input and multiple-output (MIMO) enabled chipsets are characterised by, due to the use of multiple radio chains and the complicated baseband processing that are required to enable for MIMO communication. However, the later manufactured MIMO-enabled chipsets feature advanced IC technology, which is the key feature that results in their significantly optimized energy efficiency in all operational states. Regarding the sleep mode power consumption of the AR5424, we remark that the generic IEEE 802.11 *Power Saving Mode (PSM)* mode is not supported by the open-source *Mad-WiFi* and *ath5k* drivers that control its operation and thus this value is not presented in Table 1. On the other hand, we were able to configure the PSM mode for the AR9380 NIC through the *ath9k* driver, which set the NIC in low-power state (0.12 W) by disabling most of its circuitry.

In the following experiments, we characterize the power consumption characteristics during frame transmission and reception under various settings, for both of the IEEE 802.11a/g and IEEE 802.11n compatible wireless NICs and present the obtained results in sections 5.1.1 and 5.1.2 accordingly. We also investigate the impact of the PSM mechanism on the consumption of wireless NICs and delay induced to the generated flows, through specifically designed experiments that are presented in section 5.1.3.

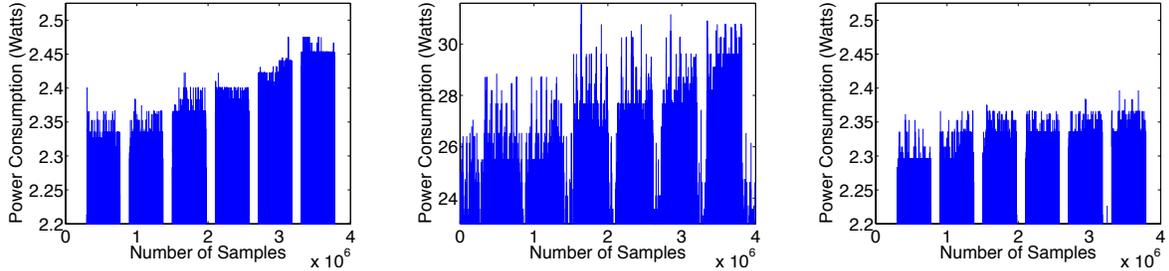
#### 5.1.1 Experimentation with IEEE 802.11a/g NICs

In this experiment, we start by characterizing the instantaneous power consumption of the IEEE 802.11a/g compatible Atheros AR5424 chipset, during frame transmission and reception events. Frames are transmitted under fixed  $TX_R$  values between the 6 Mbps and 54 Mbps that are supported by the IEEE 802.11a/g standards. We also fix the Trans-



(a) Transmission - All RF chains constantly enabled (b) Transmission - Only required number of RF chains enabled (c) Reception - Only required number of RF chains enabled

Figure 5: Energy Consumption per bit of AR9380 NIC during transmission across different MCS and Antenna settings



(a) Enabled A-MPDU aggregation NIC Power Consumption (b) Enabled A-MPDU aggregation Atom Node Power Consumption (c) Disabled A-MPDU aggregation NIC Power Consumption

Figure 6: Power Consumption of AR9380 NIC and Atom node during transmission across varying Application-Layer Traffic

mission Power ( $TX_P$ ) of the transmitter node at the maximum value of 18 dBm. Based on the collected results, we observe that the consumed power, when transmitting or receiving frames under different  $TX_R$ , does not vary significantly. As duration of frame transmission is monotonically related to the configured  $TX_R$ , it is important to quantify energy efficiency in terms of energy consumption per transmitted bit of information ( $E_B$ ). We calculate  $E_B$ , expressed in Joules/bit, as the division of the power consumption values collected for each different  $TX_R$  by the exact  $TX_R$  value expressed in bits/sec. Although  $E_B$  is a useful metric, it does not include the amount of energy that is spent for retransmission of undelivered frames, as it does not consider the frame delivery rate at the receiver, which significantly differentiates between different  $TX_R$  in realistic environments. In Fig. 4(a), we plot the obtained  $E_B$  across the available IEEE 802.11a/g compatible  $TX_R$  configurations, considering both the transmission and reception events. We notice that higher  $TX_R$  settings always result in lower  $E_B$ , which is mainly due to the decreased duration of the transmission or reception duration.

Next, we fix the  $TX_R$  value and characterize the power consumption of frame transmissions under varying  $TX_P$  settings, by configuring the  $TX_P$  of the transmitter node at the 8 available settings between the values of 18dBm and 0 dBm. Fig. 4(b) presents the instantaneous power consumption across the 8 available  $TX_P$  configurations and under the minimum (6Mbps) and maximum (54 Mbps) available  $TX_R$  settings. We clearly notice that power consumption decreases monotonically with the decrease of the  $TX_P$  and that the decrease rate varies across different  $TX_R$  configurations. Moreover, we observe that the maximum achievable energy saving of 14% can be obtained in the case that the  $TX_P$  is reduced from the default value of 18 dBm to the minimum of 0dBm, in the case that  $TX_R$  is set to 6 Mbps.

### 5.1.2 Experimentation with IEEE 802.11n NICs

In this second experiment, we investigate the impact of low-level configurations on the energy consumption of the IEEE 802.11n /MIMO compatible Atheros AR9380 chipset. AR9380 can be configured in 3 different SS configurations, namely SISO, MIMO2 and MIMO3, where one, two, or three RF-chains are used accordingly. Moreover, each different spatial stream configuration offers 8 different modulation and coding schemes (MCS), resulting in up to 24 different MCS settings for the MIMO3 case. In this experiment, we transmit frames, under fixed MCS indexes among the ones offered in each SS configuration and calculate the instantaneous power consumption in each setting. More specifically, we use channel bandwidth of 20 MHz and guard interval of 800 ns, resulting in  $TX_R$  settings that range from 6 Mbps in the MCS0 case to 195Mbps in the MCS23 case.

In order to calculate  $E_B$  for the various MCS settings, we follow the procedure described in the previous section. In Fig. 5(a), we plot the obtained  $E_B$  across the 23 available  $TX_R$  configurations, in the case that all RF-chains are constantly enabled, even in cases that the configured SS setting does not use the excess antennas. Based on the collected results, we notice that MCS configurations significantly impact power consumption, as imposed by the calculated  $E_B$ , which ranges from 376.6 nJ/bit (MCS0) to 12.2 nJ/bit (MCS23). This finding suggests that there exists a huge potential for minimisation of energy expenditure (up to 97%), through proper adaptation of MCS configurations. Figures 5(b) and 5(c) present  $E_B$  measurements for transmission and reception accordingly, in the case that only the required number of RF-chains are enabled for each configured SS setting. We notice that proper activation of the required number of RF-chains (SISO, MIMO2) can significantly increase energy savings up to 60% for transmission (27% for reception), as for the MCS0 case, where  $E_B$  re-

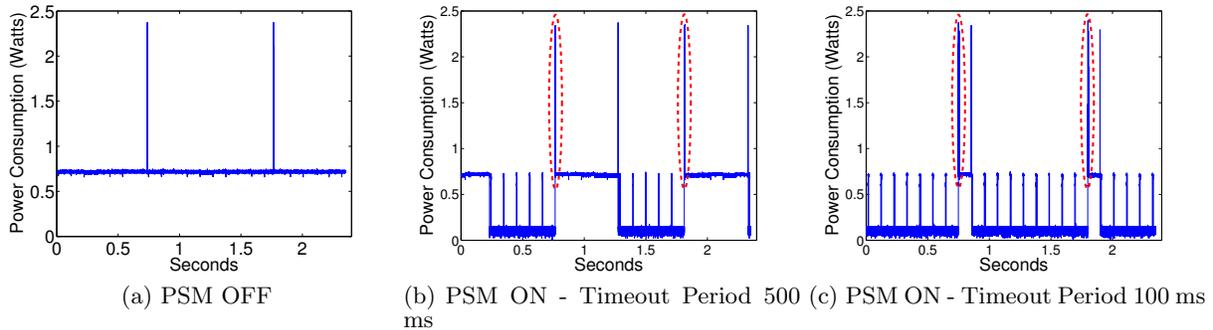


Figure 7: Instantaneous Power Consumption of AR9380 NIC and impact of PSM

duces to 153.2 nJ/bit (95.38 nJ/bit). More specifically, we also observe that MCS indexes within the same SS configuration do not remarkably impact power consumption, while indexes of different SS settings result in highly diverse  $E_B$  values, due to the power consumption values that significantly vary between different antenna settings (especially during transmission, as presented in Table 1).

The last part of this experiment has been designed to assess the impact of Application-layer Traffic Rate ( $TR_R$ ) on the consumption of wireless NICs. In this experiment, we fix the MCS index to 23, resulting in the PHY-layer  $TX_R$  value of 195 Mbps. Under this fixed configuration, we run an experiment that varies the  $TR_R$  at the transmitter node in 6 steps, among the values of 10, 20, 50, 100, 200, 300 Mbps. The whole experiment runs for 60 seconds and approximately  $4 \times 10^6$  voltage samples are collected.

Fig. 6(a) illustrates the power consumption of the NIC under the various configured  $TR_R$  values. We clearly observe that increment of Application-layer Traffic Rate results in increased power consumption, where the highest monitored increase of 0.13W, is observed between the 10 and 300Mbps  $TR_R$  values. We also notice that this observation holds even when  $TR_R$  reaches the 300 Mbps value and increases above the capacity limit. However, it is unclear whether the increased energy expenditure is solely related to the increased amount of bytes or also related to the number of frames delivered to the driver. The work in [1], which considered energy consumption on the total node level, revealed and quantified that a substantial proportion of energy is consumed during the packet processing through the Operational System (OS) protocol stack. Moreover, this work suggests that the monitored energy expenditure on the total node level, is primarily associated to the frame processing itself, rather than to the amount of bytes handled. We managed to verify the findings of the work in [1], by measuring the power consumption of an Atom-based node equipped with the AR9380 NIC and by configuring the same  $TR_R$  settings. As illustrated in 6(b), increasing application layer Traffic Rate values result in increased power consumption, where the highest monitored increase of approximately 3.5W, is observed between the 10 and 300Mbps  $TR_R$  values.

Furthermore, in order to assess the impact of PHY-layer A-MPDU frame aggregation on power consumption, we disable frame aggregation and repeat the same experiment. As demonstrated in Fig. 6(c), A-MPDU aggregation slightly increases the resulting power consumption, as monitored for the lowest  $TR_R$  value. However, the monitored trend of increasing power consumption across increasing  $TR_R$  values, is not identified in this case. These observations are rather

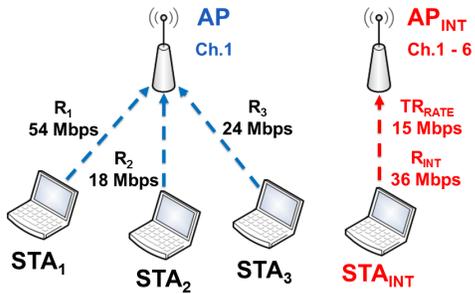
pioneering and yield interesting insights regarding the impact of traffic load, number of frames delivered to the driver and A-MPDU aggregation on energy consumed by the NIC.

### 5.1.3 Experimentation with IEEE 802.11 PSM

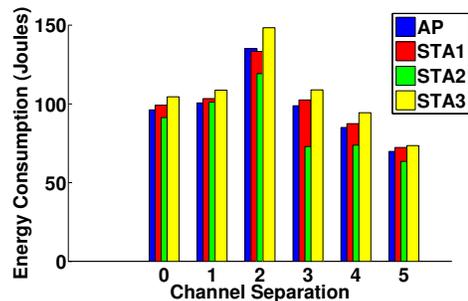
Through this experiment, we aim at quantifying the potential energy savings of the IEEE 802.11 PSM that is designed to set the wireless NICs of stations (STAs) in a low-power state during periods of inactivity and periodically activate them to fetch cached data from the access point (AP). PSM is controlled through *Beacon* frames that are transmitted by the AP at the constant *Beacon Interval* and include the *Delivery Traffic Indication Message* (DTIM), which is used to notify sleeping STAs about incoming data cached at the AP that are waiting to be delivered. Once the sleeping STA is being informed through the DTIM that frames are waiting to be delivered, it returns to active mode and transmits the *PS-Poll* frame to inform the AP that it is ready to receive the cached data. The DTIM may be reported every  $n$  *Beacon Intervals* and in this case the STA only needs to awaken at the DTIM interval to check for cached data. However, as shown in [29], most commercial cards are configured to wake up and receive all transmitted *Beacon* frames. The last determines that the maximum period a WLAN NIC can remain in sleep mode is indicated by the *Beacon Interval*.

In this experiment, we use the AR9380 chipset, which supports the PSM mechanism and configure the default *Beacon Interval* of 100 ms. We use the ping application to transmit ICMP [30] packets every 1 second, from the AP to the STA and measure the delay induced during NIC sleep intervals. In parallel with the described experiment, we monitor the power consumption of the STA's NIC. First, we disable the PSM mechanism and as depicted in Fig. 7(a), we observe that the STA's NIC consumes between 0.69 W and 0.74 W in idle and receive mode accordingly, while it consumes 2.45 W to transmit the *ICMP Echo Reply* packets, resulting in the total energy consumption of 1.6923 J within the 2.35 seconds of observation.

In the second phase, we activate the PSM mode, so that the STA's NIC constantly operates in sleep mode and is activated to receive *Beacon* frames every 100 ms and get informed about cached data. In Fig. 7(b), we present the instantaneous power consumption of the STA's NIC and highlight in the red ellipses the instances during which the NIC transmits the *PS-Poll* frame and subsequently receives the *ICMP Echo* and transmits the *ICMP Echo Reply* packets. We also observe that the NIC does not directly sleep, but remains in idle mode for 500 ms, which is the default *Timeout Interval*. In this phase, the NIC consumes approximately



(a) Experimental Topology



(b) Energy Consumption/ NIC on each channel

Figure 8: 1st high-level experiment using 802.11a/g compatible NICs

1.0613 J, resulting in an overall reduction of 37%, in comparison with the first phase. In Fig. 7(c), we plot the consumption during the third phase, where we set the *Timeout Interval* equal to 100 ms and measure that the NIC consumes just 0.4635 J (73% reduction) during the observation interval. Regarding the ping application delay, we remark that it equals 1.3 ms in the first phase, while in both of the next two phases it approximates the 150 ms value, as the configuration of a shorter *Timeout Interval* does not impact the frequency of *ICMP Echo Reply* packet transmissions. Based on the obtained results, we remark that significant energy savings can be attained, through proper scheduling of sleep intervals, while maintaining delay within acceptable limits.

Having executed several low-level experiments, we managed to identify that significant amount of energy can be saved, through proper configuration of MAC protocol parameters. However, further investigation is required in order to arrive at safe conclusions and we aim at examining these initial observations in detail as part of our future work. Nevertheless, the extended set of presented low-level experiments have clearly demonstrated the advanced energy monitoring capabilities of the NITOS EMF framework.

## 5.2 High level Experiments

The experimental scenario in the following two experiments includes 3 STAs that are associated with the same AP and are simultaneously uploading a file of fixed size. The same experiment is repeated under varying channel conditions and different PHY-layer configurations for each wireless NIC, while energy consumption is being constantly monitored. In the first experiment, we use only IEEE 802.11a/g compatible NICs, while in the second one IEEE 802.11n compatible NICs are employed.

### 5.2.1 First Experiment based on IEEE 802.11a/g

In this experiment, the 3 STAs are simultaneously uploading a file with size of 25MBs, while being associated with the AP that is operating on Ch.1 of the 2.4 GHz band. Another pair of collocated nodes is generating interference, with the station node ( $STA_{INT}$ ) transmitting on uplink at the Application-layer traffic load of 15Mbps. Fig. 8(a) illustrates the experimental topology along with the PHY-layer rate settings of each specific NIC. The experiment is repeated 6 times, where in each different run we configure the  $AP_{INT}$  to operate on a different channel between Ch.1 and Ch.6 of the 2.4 GHz band. The experimental setup is inspired by our previous work on Dynamic Frequency Selection Algorithms [31].

During each different run, we monitor the energy consumption of each NIC and plot the collected results in Fig. 8(b). While  $AP_{INT}$  moves from Ch.1 to Ch.2 and subsequently to Ch.3, we notice that the total energy consumption of all NICs is increased. Due to adjacent channel interference, transmissions of  $STA_{INT}$  are not always detected by the three STAs, which results in frame collisions and subsequent frame retransmissions. The overall effect is that the file transmission durations are increased for each individual node and thus impact the overall energy expenditure. As  $AP_{INT}$  moves from Ch.3 to Ch.5, we notice that the energy consumption tends to decrease for all NICs, resulting in the lowest monitored values in the case that the interfering link operates on Ch.6, as it no more interferes with the 3 STAs.

A particular observation in all cases is related to the energy performance of STA2. While STA2 uses the lowest PHY-layer rate of 18 Mbps, compared to the rates of STA1 (54Mbps) and STA3 (24Mbps), it manages to result in the lowest energy expenditure in all cases. This comes in contrast with the higher  $E_B$  values that the lower PHY-layer rates correspond to. However, this uniquely identified performance results due to the fact that STA2 completes the file uploading sooner than the rest nodes, in all cases. Based on the experimental log files, we observe that STA2 always achieves the highest throughput. The increased throughput performance of STA2 can only be associated with the "Capture Effect" phenomenon [32], due to which certain topology configurations result in unfair throughput distribution for specific links.

### 5.2.2 Second Experiment based on IEEE 802.11n

In this second experiment, the 3 STAs follow the same file uploading process, while the AP is operating on Ch. 36 of the 5 GHz band, using a channel bandwidth of 40 MHz. However, in this scenario MIMO enabled NICs are used and the size of the file to be uploaded is 1 GB. We statically fix the single (SS) stream configuration for STA1, while STA2 and STA3 are configured with the double (DS) and triple (TS) stream setting accordingly. We also enable the default MCS adaptation mechanism of the driver, which results in the assignments of MCS7 for STA1, MCS15 for STA2 and MCS21 for STA3, as presented in Fig. 9(a). The experiment is executed in two phases, where in the first phase each node completes the file uploading, using the medium in an *individual* way, as the rest two STAs are disabled. In the second phase, we configure the 3 STAs to transmit simultaneously and monitor energy performance, during the execution of the *combined* experiment.

Fig. 9(b) depicts the total energy consumed by each in-

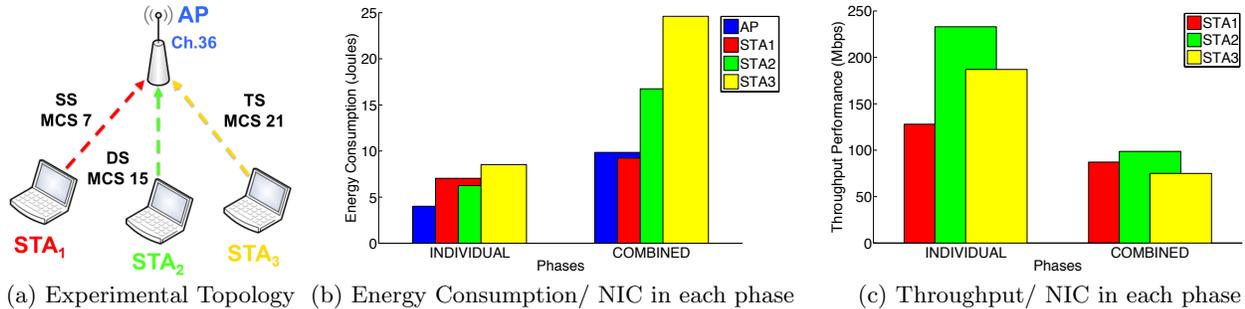


Figure 9: 2nd high-level experiment using 802.11n compatible NICs

dividual wireless NIC, as monitored during both phases of the experiment. In the first phase, we notice that the AP results in the lowest energy expenditure, which comes from the fact that it operates in reception mode, in which case the energy consumption is significantly less compared with the consumption during transmission. We also notice that STA3 using the TS mode consumes the highest amount of energy (8.53J), while STA1(7.04J) and STA2 (6.25J) follow accordingly. In this case, the energy performance of all nodes is directly related with the  $E_B$  that each different MCS index is characterized by, which values are 7.705nJ/bit, 7.535nJ/bit and 6.66nJ/bit for MCS21, MCS7 and MCS15 accordingly.

In the second phase, we observe that average energy consumption is significantly increased for all nodes, resulting due to the decrease in channel access opportunities and the corresponding increase of idle listening periods. More specifically, STA3 again consumes the highest amount of energy (24.61J), while STA2(16.75J) and STA1(9.22J) follow accordingly. In this case, energy performance cannot be associated with the  $E_B$  parameter, as it does not consider the amount of time spent during idle listening periods. Based on the throughput performance evaluation, which is illustrated in Fig. 9(c), we notice that in the combined experiment all nodes achieve nearly equal performance, which results in approximately equal time spent for the completion of the file uploading process. In this case, NICs that enable higher number of RF-chains and result in higher instantaneous power consumption, eventually induce higher total energy consumption within the same experiment duration.

As it is clearly demonstrated through the two high level experiments, important factors, such as topology and interference conditions and interaction among simultaneously transmitting nodes, can greatly impact energy expenditure. Due to the inherent inability of power consumption models to accurately analyze energy consumption in such complex scenarios, online energy consumption monitoring needs to be applied, in order to arrive at concrete conclusions.

## 6. DISCUSSION AND FUTURE WORK

NITOS EMF is designed to monitor the consumption of wireless testbed infrastructure at the level of the wireless NIC, as well as at the total node level. The developed solutions consist an ideal environment for conducting research related to the development of energy-efficient protocols and architectures. However, the main target of energy-aware protocols is to reduce the consumption of mobile devices, such as smartphones, tablets and laptops, which operate on batteries of limited capacity and are characterised by restricted duration of operation. Currently, NITOS EMF is able to monitor the consumption of mobile devices, however

execution of realistic remote and mobile scenarios cannot be supported, due to the framework’s mobility limitations.

In an effort to provide for energy efficiency evaluation of realistic experiments that include mobility scenarios as well, we designed the mobile version of the NITOS ACM card and integrated it with mobile phones. The Mobile Phone Power Monitoring solution follows a similar to the NITOS ACM design, as it is also based on the INA139 current-shunt monitor, while a Micro SD card module is also used for local storing of measurements. However, the Arduino Mega board has been replaced by the compact Arduino-like Pro Micro board [33] that operates at 3.3 V and features the ATmega32u4 [34] running at 8 MHz, while the Bluetooth interface RN-42-N [35] is used instead of the Ethernet interface for wireless remote control and transferring of measurements. Moreover, the mobile solution does not require external power supply, as it is properly powered by the mobile phone’s battery, without affecting the power consumption measurement procedure. Fig. 10(a) presents the fully operational prototype solution and its various components integrated with a Samsung Galaxy Nexus mobile phone.

The maximum sampling rate of the Pro Micro has been increased from the default value of 4.33 KHz to 17 KHz through the application of techniques similar to the ones described in section 4.2. The low power consumption profile of the various selected components result in a total consumption of 0.043 W and 0.092 W, in idle and monitoring mode accordingly. Considering an experimental scenario that lasts for one hour, we estimate that the mobile solution will require approximately 30 mAH for power monitoring and additionally 8.2 mAH to transfer the 67.5 MBs generated file through the Bluetooth interface. As a result, the proposed measurement approach does not limit the operation of recent smartphone models, which feature batteries that exceed the capacity of 2000 mAH. Fig. 10(b) presents an indicative experiment, in which we monitor the power consumption of the Samsung Galaxy Nexus mobile phone. We highlight three phases, where during the first phase the phone’s display is activated, then in the second phase and approximately at 2.8 s we power it off and observe that it takes approximately 4.2 s till the device falls in sleep mode in the third phase.

We are currently in the process of developing an integrated solution that will combine the various components on a single PCB and result in a very compact design able to fit in most mobile phone cases with minor modifications. Fig. 10(c) presents the PCB design that is under fabrication. As soon as the PCB based solution is available, we will integrate it with mobile phones of volunteers, such as lab members and University students, and start collecting measurements that correspond to the energy that is consumed during long-term

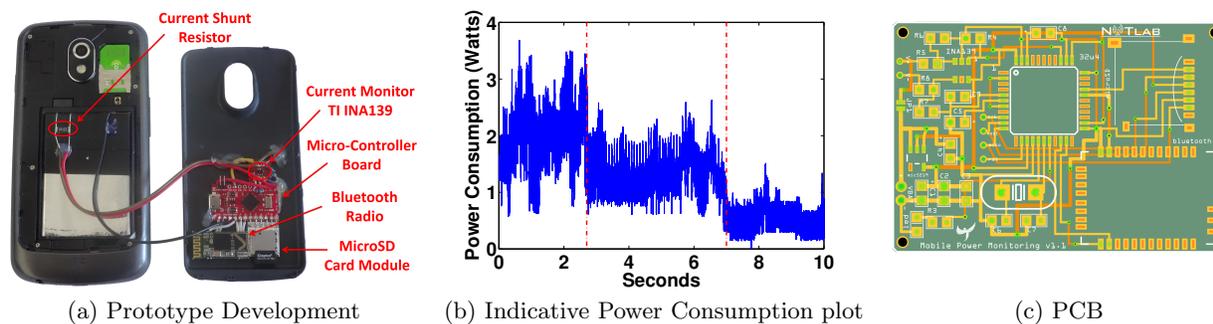


Figure 10: Mobile Phone Power Monitoring Solution

execution of everyday life scenarios. We are also developing an Android application that will be used to monitor the running processes and activated modules during the power monitoring period, in order to derive important conclusions regarding the energy that is consumed by each individual task. As part of our future work, we also plan on extending the range of collected measurements, by connecting the framework with the available software defined radio and sensor hardware of NITOS, as proposed in our recent work [36].

## 7. CONCLUSIONS

In this work, we introduced the novel NITOS EMF framework that is able to characterise the consumption of wireless testbed infrastructure in an online way. The proposed framework is built on a distributed network of low-cost, but highly accurate devices and is fully integrated with the large-scale wireless NITOS testbed. Through extensive experiments, we demonstrated the advanced platform capabilities that can aid towards energy performance assessment of realistic testbed experiments.

## 8. ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreements n. 287581 (OpenLab IP project), n. 265496 (STAMINA STREP project) and n. 258301 (CREW IP project). Moreover, the authors would like to thank Akis Chanos for his valuable help during the initial hardware modifications and testing.

## 9. REFERENCES

- [1] A. Garcia-Saavedra, P. Serrano, A. Banchs, and G. Bianchi. "Energy consumption anatomy of 802.11 devices and its implication on modeling and design". In *Proceedings of CoNEXT*, 2012.
- [2] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. "energy consumption in mobile phones: A measurement study and implications for network applications". In *Proceedings of IMC*, 2009.
- [3] J. Tarascon. "Key challenges in future Li-battery research". *Philos Trans A Math Phys Eng Sci*, pages 3227–4, 2010.
- [4] X. Zhang and K. Shin. "EMiLi: energy-minimizing idle listening in wireless networks". In *Proceedings of Mobicom*, 2011.
- [5] Kishore R., Ravi K., Honghai Z., and Marco G. "Symphony: Synchronous Two-phase Rate and Power Control in 802.11 WLANs". In *Proceedings of Mobisys*, 2008.
- [6] J. Manweiler and R. R. Choudhury. "Avoiding the rush hours: WiFi energy management via traffic isolation". In *Proceedings of MobiSys*, 2011.
- [7] M. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. "Energy-delay tradeoffs in Smartphone Applications". In *Proceedings of MobiSys*, 2010.
- [8] "Tmote sky Specifications", <http://goo.gl/Tc9qR>.
- [9] "MICAz Specifications", <http://goo.gl/rLYHU>.
- [10] "Atheros White Paper - Power Consumption and Energy Efficiency Comparisons".
- [11] "NITOS Wireless Testbed", <http://nitlab.inf.uth.gr>.
- [12] K. Jang, S. Hao, A. Sheth, and R. Govindan. "Snooze: energy management in 802.11n WLANs". In *Proceedings of CoNEXT*, 2011.
- [13] C. Li, C. Peng, S. Lu, and X. Wang. "Energy-based rate adaptation for 802.11n". In *Proceedings of Mobicom*, 2012.
- [14] D. Halperin, B. Greenstein, A. Sheth, and D. Wetherall. "Demystifying 802.11n power consumption". In *Proceedings of SIGOPS HotPower*, 2010.
- [15] A. Hergenroder, J. Horneber, and J. Wilke. "SANDbed: A WSN Testbed for Network Management and Energy Monitoring". In *GIITG KuVS Sensornetze*, 2009.
- [16] A. Kipp, J. Liu, T. Jiang, J. Bucholz, L. Schubert, M. Berge, and W. Christmann. "Testbed architecture for generic, energy-aware evaluations and optimisations". In *Infocomp*, 2011.
- [17] K. Gomez, R. Riggio, T. Rashed, D. Miorandi, and F. Granelli. "Energino: Hardware and Software Solution for Energy Consumption Monitoring". In *Proceedings of WiOpt*, 2012.
- [18] G. Kazdaridis, S. Keranidis, H. Niavis, T. Korakis, I. Koutsopoulos, and L. Tassioulas. "An Integrated Chassis Manager Card Platform Featuring Multiple Sensor Modules". In *Proceedings of Tridentcom*, 2012.
- [19] "ATmega 2560 micro-controller", <http://goo.gl/IFHwq>.
- [20] "Arduino Ethernet Shield", <http://goo.gl/LXs1oG>.
- [21] "Texas Instruments INA139", <http://goo.gl/rPQLB>.
- [22] "Advanced Arduino ADC", <http://goo.gl/AwQ95>.
- [23] "Enhancing Arduino's ADC", <http://goo.gl/BRXCX>.
- [24] "Atmega ADC accuracy vs clock speed", <http://goo.gl/qTlhx>.

- [25] "NI-6210 DAQ module", <http://goo.gl/oFSJw>.
- [26] "OMF-control & Management Framework", <http://omf.mytestbed.net/>.
- [27] "OML Measurement Library", <http://mytestbed.net/projects/oml/wiki/>.
- [28] "Energy Characteristics of NITOS NICs", <http://nitlab.inf.uth.gr/NITlab/papers/EnergyTR.pdf>.
- [29] M. Tauber and S.N. Bhatti. "The Effect of the 802.11 Power Save Mechanism (PSM) on Energy Efficiency and Performance during System Activity". In *Proceedings of GreenCom*, 2012.
- [30] "Internet Control Message Protocol", <http://goo.gl/bjQCyr>.
- [31] G. Kazdaridis, S. Keranidis, A. Fiamegkos, T. Korakis, I. Koutsopoulos, and L. Tassiulas. "Novel metrics and experimentation insights for dynamic frequency selection in wireless LANs". In *Proceedings of ACM WinTECH*, 2011.
- [32] J. Lee, W. Kim, S. Lee, D. Jo, J. Ryu, T. Kwon, and Y. Choi. "An experimental study on the capture effect in 802.11a networks". In *ACM WinTECH*, 2007.
- [33] "Pro Micro Board", <http://goo.gl/9QdRs8>.
- [34] "ATmega 32u4 micro-controller", <http://goo.gl/gln5Fp>.
- [35] "RN-42N Bluetooth Radio", <http://goo.gl/6MRgiH>.
- [36] V. Passas, K. Chounos, S. Keranidis, W. Liu, L. Hollevoet, T. Korakis, I. Koutsopoulos, I. Moerman, and L. Tassiulas. "Online Evaluation of Sensing Characteristics for Radio Platforms in the CREW Federated Testbed". In *Proceedings of ACM Mobicom*, 2013.