Online Device-Level Energy Accounting for Wireless Sensor Nodes

André Sieber and Jörg Nolte

Distributed Systems/Operating Systems Group
Brandenburg University of Technology, Cottbus, Germany
{as,jon}@informatik.tu-cottbus.de

Abstract. Energy is the crucial factor for the lifetime of wireless sensor networks. Nonlinear battery effects and nonuniform workload distribution can lead to early node failures. This makes it necessary to manage energy consumption. But to manage energy it is essential to know how much energy is spent by the system. Additionally, for a more fine-grained management it is necessary, to know where the energy is spent. This can be a complicated task, since nodes are not identical due to device variations and the consumption can change over time.

In this paper we present an online energy accounting approach which focuses on simplicity instead on fine granularity and timing accuracy. We argue that the efficacy of an energy accounting model depends more on the input consumption data than on exact timing, especially when the real consumption varies between nodes and in time. Results show that this approach is capable of correctly accounting the energy that nodes spend in scenarios with deviating environment conditions.

Keywords: Energy, accounting, measurements, operating systems, embedded systems, wireless sensor networks.

1 Introduction

Energy is a major concern in wireless sensor networks. Sensor nodes should run for years with a limited energy budget providing a high quality of service, mostly using conventional alkaline or lithium batteries. To prevent sensor nodes from early failures due to depleted batteries, it is necessary to know the energy consumption and manage the available energy.

While the consumption can be predicted with simulators, e.g. [1], running tests on real hardware is more precise and flexible. Since it can not be assumed that all sensor network developers have access to expensive measurement equipment, a software based approach is feasible and flexible.

Within the development phase, the assumptions about the energy consumption of a node make it possible to determine its reachable lifetime. Since sensor network applications are based on modular operating systems like TinyOS [2], Contiki [3] or Reflex [4], the application developer uses the system and its drivers for the platform he/she is using. The developer has no deeper knowledge

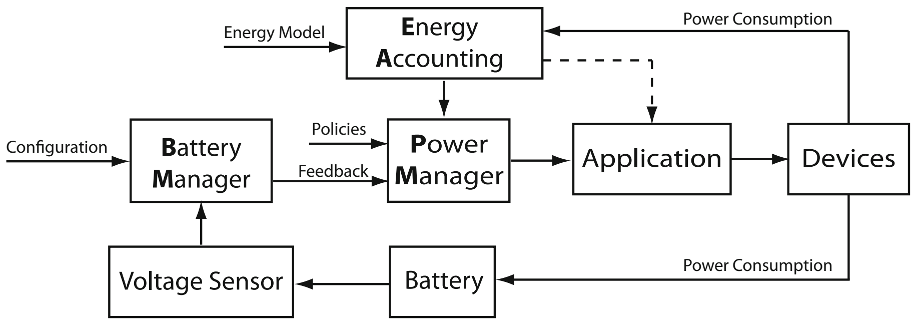


Fig. 1. Power management concept taking the current consumption and the battery state of charge into account

about the node hardware and inner behavior of the OS or the device drivers, which can lead to misleading assumptions about the sensor nodes energy consumption. Providing information about the consumption of individual devices helps the developer to find potential *energy holes* and thus increase the lifetime.

Within the actual deployment phase, online energy accounting is an instrument for the nodes power management to monitor the consumption and enforce its constraints and policies. Fig. 1 shows an approach where the consumed energy as well as information from the battery are used to control the application behavior to reach a predefined lifetime goal. Taking the battery state of charge into account makes it possible to react to the batteries non-linear effects [5] and possible inaccuracies of the consumption model. While the consumption over a certain time is interesting for energy management, the momentary consumption is interesting for the battery management which can limit the load to the battery and increase the lifetime [6].

The energy that is consumed by a sensor node depends on the devices that are active, the time these devices are active and the consumption of these devices. Which device is active is most often under control of the node software, along with the time it is active. Statements about the consumption of a device can be taken from the manufacture's datasheet. But not all nodes are equal. Besides variations of the analog and digital components, manufacturing faults can occur. While this may not render the nodes useless, it could change their power consumption footprint.

To power a node an energy source is necessary to which the node is connected directly or using a voltage regulator. Connecting the node directly exposes them to the degrading voltage level of the source, which can change the consumption foot print of the device due to voltage dependencies in the power consumption and performance of integrated circuits [7]. Additionally, the voltage can fall below the operating limit of certain devices or the entire node. Voltage regulators can avoid effects of hazardous voltage drops of batteries by delivering a constant voltage to the node and seem to make energy accounting simpler. But these regulators suffer from low efficiency [8], especially when the system drains very little energy, e.g. when in sleep mode. Furthermore, this efficiency depends on the input voltage and the power drained through the connected consumer and must be considered in an online-consumption accounting. Together with the hardware costs, the benefits may be negated and the usage must be carefully considered by the system designer [7].

An energy accounting approach must deal with such deviations. One way to face the device variations is to generate the model for each node individually. But this is expensive and error prone. Another way is to assume and use a maximum consumption for all nodes. While this much simpler, it is way too inaccurate since outliers would rule the assumption. For most nodes the real consumption would be much lower than assumed. The third way is to incorporate measurements of the nodes and usage of a statistical model to cover most but not all nodes. Facing the variation makes it necessary to implement a dynamic accounting which is capable of changing the underlying consumption values depending on the voltage level or converter efficiency.

The remainder of the paper is structured as follows: In section 2 related work is presented. The accounting approach and how its input values were collected is presented in section 3. Implementation details are shown in section 4. In section 5 our approach is evaluated. Finally, a conclusion is given in section 6.

2 Related Work

Approaches for energy accounting can be divided into hardware based, software based and combined approaches.

The authors of [9] present a power monitoring infrastructure. While this approach is hardware based and not intended for in-field deployments, they base the need for power monitoring on device variations along with software changes that can change the power consumption as a side effect.

Coloumb counters, e.g [10], can be used to track the consumed energy online, but like other hardware based approaches they introduce an energy overhead and increase the nodes hardware cost. An in situ power observation tool based on a shunt resistor called SPOT is presented in [11]. Through iCount [12], the additional hardware costs are greatly reduced for systems featuring a voltage converter. The approach counts the switching cycles of the regulator and correlates them to the energy consumed by each switch.

Quanto [13] eliminates the disadvantage of other hardware based approaches to only measure the consumption of the whole system instead of single devices and states. Together with the information about the system consumption and the active devices within an interval, the approach uses linear regression to identify individual consumptions.

In [14] a software based approach for online energy accounting is presented. To keep track of the consumed energy, so called energy containers are introduced and refined in [15]. The approach targets the accounting of tinyDB queries. The necessary accounting infrastructure consists of finite state machines for each individual device where the states are the consuming states of the devices and the edges represent state changes. Both include information about their consumption and duration.

Another software approach is presented in [16]. For each device activation and deactivation a timestamp is taken and the difference accumulated. The management keeps track of the devices and uses them to estimate the system energy consumption.

The authors of [1] present AEON, an off-line tool for prediction and profiling the energy consumption of Mica2 nodes. To build their energy model the current consumption of three nodes was measured, resulting in a variation of approximately 5% between them but no further details where provided.

The approach presented in [17] introduces the so called passive voltage scaling. Instead of using a DC-DC converter, the node is connected directly to the battery and the approach scales the µC frequency according to the available voltage level. Through not using a converter or statically use the µC frequency available at the lowest voltage level, the approach benefits in runtime and throughput.

In [18] the impact of device variations regarding sleep modes is analyzed. For the tested µC, an Atmel SAM3U based on an ARM Cortex M3 core, a deviation of more than factor 5 between different nodes for the sleep consumption was observed. For active modes the deviation was around 10%. The effect increased with the temperature. To overcome this, the authors proposed a variability-aware duty cycle, were the node watches its temperature and adjusts its duty cycle based on a stored sleep-power vs. temperature ratio created for each node.

While the basic idea and mechanisms of our energy accounting are similar to [14] and [16], we focus on the model input instead of trying to maximize the timing accuracy. We argue that a complex model is still vulnerable to variations. A more simple model with a pessimistic enough input can cover most variations. Additionally, our approach is capable of dealing with variations generated by the energy source.

3 Device Level Energy Accounting

When designing an energy accounting infrastructure for embedded systems, the first question is how the energy is consumed. Followed by the question how that behavior can be best modeled regarding accuracy and effort. The efficiency of the model not only depends on the time but also on the consumption values fed to the model.

3.1 Device Energy Consumption

A sensor node consists of various devices with partly independent control flows. While some devices are only binary and simply activated or deactivated, other devices have multiple states. The state changes could happen under the control of the driver or happen independently as given by the device state machine. The device activation can occur as discrete and fixed events or be time depended. Devices can be context driven and thus under the control of the application or

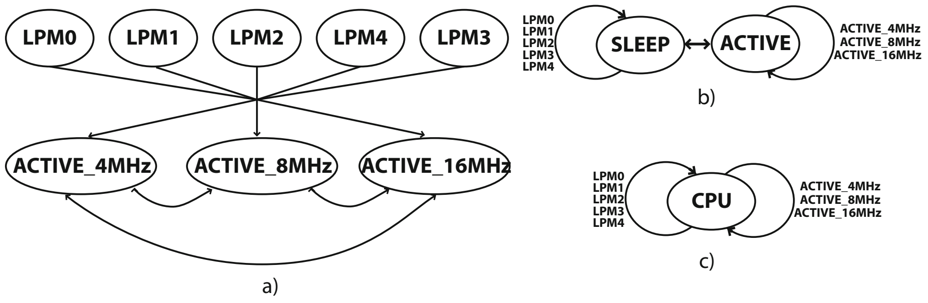


Fig. 2. Possible accounting granularity for the µC a) every sleep mode and frequency of the CPU accounted individually b) no distinction between the single sleep states and frequencies c) account all states to a single sink

demand driven when requests occur. For example, receiving over an interface is context driven while sending is demand driven. Additionally, there are different factors that can influence the device consumption. The device can have different configurations, e.g. the transmission power of the radio. The consumption can be more or less dependent on the supply voltage and/or the temperature. Some devices or device operation states have a consumption of a few µA while others consume a multiple of 10mA.

Other devices could influence the usage of a device. For example, interrupts or other cpu activity can prevent a device driver from deactivating a device and thus unexpectedly increase the device active time and, as a result, the consumption.

3.2 Accounting Model

For accounting, existing driver code has to be modified. This requires understanding of the drivers functionality. For new driver code understanding is already necessary, making the overhead for the driver developer minimal.

Our approach, like e.g. [16] is based on tracking the active time of device states.

This makes it possible to model and cover a nodes energy consumption in a simple and flexible way. To account the consumed energy of a device, the active time and its consumption must be provided. For this we adopt Quanto's *energy sinks* and *power states* view to our needs. In our view a sink is a potentially independent unit that consumes energy, while a power state defines how much energy is consumed by a sink.

The energy sinks of the individual devices form a finite state machine describing the devices energy behavior. Since the accounting granularity depends on the number of sinks in our approach, it can be reasonable to switch sinks to states and vice versa. Fig. 2 shows an example of the possible accounting granularity. For development and testing purposes a detailed coverage of the energy distribution within the CPU may be useful to find *energy holes* due to wrong sleeping behavior. Within a productive real-world environment, the accounting can be much simpler since such a detailed breakdown may not be necessary for the power manager or network maintenance. When memory is very constrained it is possible to account all energy states to a single sink, providing information about the energy consumption of the device but loosing all information about its internal distribution. To support such flexibility, the proposed accounting system has to be able to switch between sinks and modify the consumption of a sink dynamically.

Another reason for modifying the consumption of a sink dynamically are changes of the supply voltage due to battery voltage decrease (or increase when connected to a harvesting system) when the node is connected directly. We propose a system that monitors the voltage and informs all involved device drivers when the voltage changes substantially. The driver developer is responsible for providing the content for a function that changes the device sink's consumption. The frequency in which the voltage level is checked may be changed depending on the load, since high loads can lead to voltage drops and thus changing the consumption of active devices. Using information about the active devices, the frequency can be increased as the load rises and decreased when little load is applied to conserve energy.

The accounted amount may be modified according to an efficiency factor induced by a voltage regulator. If a regulator is present, the supply voltage of the node is stable, but the efficiency of the regulator depends on its input voltage and the load applied. The factor must be recomputed when the regulator input (most likely a battery) voltage, or the load changes. The factor has to be applied to all accounting computations

3.3 Gathering Consumption Data

To build our energy model we ran a series of experiments where the different active modes of 90 Texas Instruments eZ430-Chronos [19] nodes where measured. While not explicitly designed as wireless sensor node, they seem ideal due to their features, size and low price compared to designated sensor nodes. They are equipped with an MSP CC430F6137 [20] which includes an embedded C1101 radio module and additionally features a pressure gauge and an accelerometer.To gain access to the ports we modified the nodes by swapping the buttons with pin connectors.

Setup. The measurement setup consisted of a PowerScale Unit [21] capable of measuring current dynamically in a range from 200nA to 500mA at a maximum of 100k samples/sec. To determine the accuracy of the measurement, the standard deviation σ of each mode for every node was calculated. Due to the design of the measurement hardware, σ rises with increased consumption. To reduce the impact of the temperature on the energy consumption the measurements where made in a room equipped with an air-conditioning system, keeping the temperature fixed at $25 \pm 1°$C.

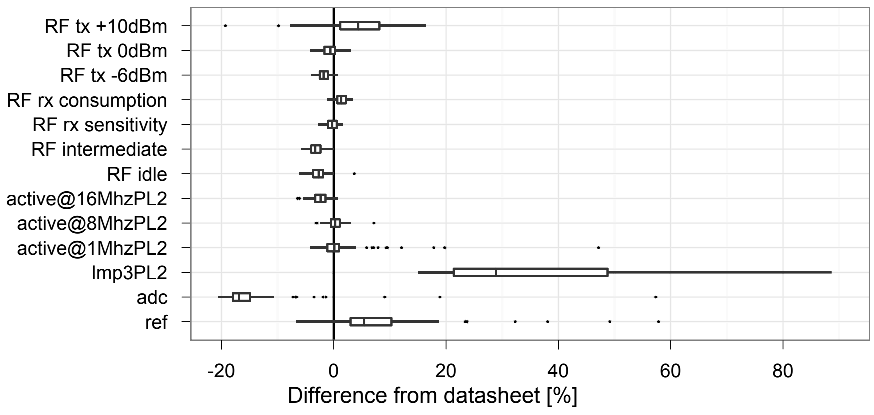


Fig. 3. Relative difference from datasheet for selected modes at 3V

The measured modes of operation include different on-chip parts of the µC. Sleep and active modes, different radio states, the consumption of the on-chip Analog-Digital-Converter (ADC) and voltage reference generator (REF) were measured. The µC features different power levels (PL0 to PL3) which determine the maximal possible frequency but depend on the voltage provided. Since PL2 is sufficient for all frequencies below 20Mhz and is enough to power all on-chip devices, it is used for building the energy accounting models. The radio modes along with the ADC and REF include the current of the µC sleep mode at PL2, which is specified by the datasheet to have a current consumption of 2.2µA at 3V.

Measurement Results. Fig. 3 shows the measured current consumption of the nodes compared to the datasheet values.

Since the measurement conditions of the datasheet values could not fully be reproduced due to the connected external components, a higher consumption can be expected and is natural.

This holds especially true for the µC sleep mode, b ut the difference varies much among the nodes in this operation mode. 20% of the sleeping nodes could not reasonably be shown in Fig. 3 since they consume a multiple of the datasheet prediction. For some of the nodes the consumption while sleeping fluctuates compared to the other nodes. This fluctuation and the high consumption indicates that these nodes are defective, but not unusable. Using them in a productive - long running and thus energy conserving - sensor network can lead to unexpected node failure due to exhausted batteries.

For all other modes the discrepancy between the measured nodes and the datasheet values is considerably lower. With rising consumption, most modes show less variance and a minor number of outliers. Depending on the operation mode, some or all nodes have a lower consumption than expected, for example most radio modes consume slightly less energy. One notable exception is the

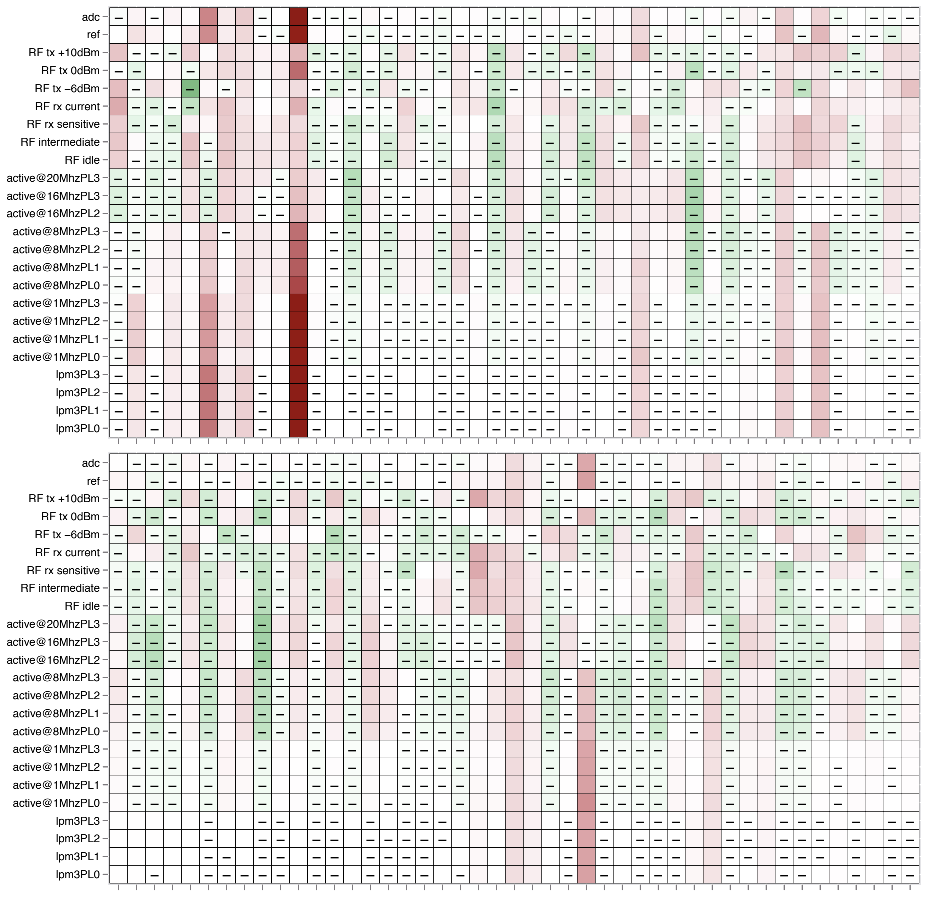


Fig. 4. Average current of each node (horizontal) in a mode (vertical). Color intensity marks the distance to the median of all nodes. "-" indicates that the mode is better than the median.

radio transmission mode with +10dB. Some of the nodes behave noticeably different than the others resulting in a wide variance. As we investigated, we identified the antenna as the problem, since some nodes have a loose antenna, resulting in a much lower consumption than expected. The nodes antenna is a metal surrounding the display, on nodes where this antenna was replaced by a simple wire based antenna, the lower consumption was also observed together with a noticeably increased range. Please note that these modified nodes are not part of the results presented and are not included in the energy model.

Differences between modules could be leveled depending on the node. For example, the REF module consumes more energy than expected on most nodes, but the ADC consumes less energy, but both modules work mostly in conjunction.

Table 1. Current consumption values at 3.0V used to build the models

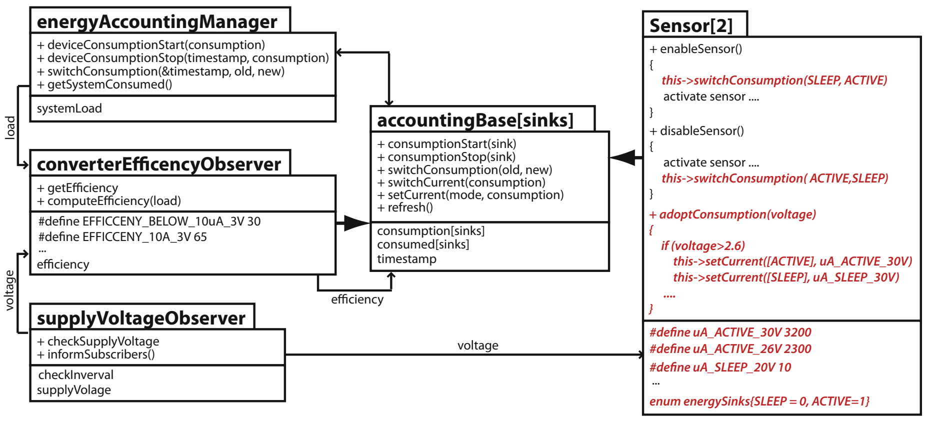
	Average current in mA		
	m-Datasheet	m-Median	m-Pessimistic
LPM3 sleep	0.0022	0.0028	0.0044
CPU@8Mhz	1.75	1.755	1.770
CPU@16Mhz	3.45	3.369	3.407
ADC	0.15	0.124	0.127
REF	0.1	0.105	0.109
Radio IDLE	1.7	1.657	1.672
Radio RX	16.0	16.214	16.371
Radio TX@0dBm	16.9	17.694	17.894
Radio TX@10dBm	33.0	34.441	35.952

Since the current consumption varies, the question arises if it is likely that nodes are "bad" or "good" in all modes of operation. Fig. 4 shows the average current consumption of each node (horizontal) in every mode (vertical) compared to that modes median. The color intensity marks the distance to the modes median (darker means more deviation), while the "-" indicates that the mode is better than the median of all nodes. As the figure shows, there are nodes that are "bad" or "good", but mostly the nodes vary.

Model Building. Table 1 shows the values used to build the energy accounting models. Apart from the datasheet values used for the *m-Datasheet* model, the median of the experiment result set is used as a second model base *m-Median*. Additionally, the 0.8 percentile of the experiment result set is used to build a pessimistic model (*m-Pessimistic*). While a pessimistic model ensures that the consumption is not underestimated which is more dangerous for power management than overestimating, using the worst nodes consumption would be much too pessimistic since it consumes a multiple of the others. The 0.8 percentile covers most of the nodes while not including a huge overestimation of better nodes consumption.

As the table shows, the differences of the modes vary between the models. Mostly the models based on the experiments differ only in a few µA. The highest impact can be expected from the sleep mode values, since it takes a substantial part of the consumption for nodes with a low duty cycle.

Since the consumption accounting is, for the sake of efficiency, not based on floating point numbers, the smallest unit for the current is one µA and thus all values are rounded up. Rounding up results in an overestimation of the sleep mode but as argued, a small overestimation is not as dangerous as an underestimation may be. Additionally, using integer numbers makes it possible to use the hardware multiplier available on many µC which reduces the computation costs.



```
+ enableSensor()
{
   this->switchConsumption(SLEEP, ACTIVE)
   activate sensor ....
}
+ disableSensor()
{
   activate sensor ....
   this->switchConsumption( ACTIVE,SLEEP)
}
+ adoptConsumption(voltage)
{
   if (voltage>2.6)
      this->setCurrent([ACTIVE], uA_ACTIVE_30V)
      this->setCurrent([SLEEP], uA_SLEEP_30V)
   ....
}
#define uA_ACTIVE_30V 3200
#define uA_ACTIVE_26V 2300
#define uA_SLEEP_20V 10
...
enum energySinks{SLEEP = 0, ACTIVE=1}
```

Fig. 5. Accounting system overview. The SENSOR shows which code must be added to existing driver code.

4 Implementation

While we implemented our approach for the event-driven operating system Reflex, it can be easily adopted to other operating systems targeting embedded systems and sensor nodes. Since Reflex is implemented in C++, it opens some opportunities to ease the integration of the accounting mechanisms into the existing device drivers. Each driver that should be accounted is derived from a base class and gains all the functions and variables necessary for the accounting. Furthermore, the device is registered at the accounting manager (*energyAccountingManager*) and, if included, by the mechanisms for the adaption of changing voltages (*supplyVoltageObserver*) and the presence of a voltage converter (*converterEfficencyObserver*).

The developer of the device driver must insert the function calls into the code parts that change the energy consumption of the device. Additionally, when the supply voltage varies and has an impact on the consumption, the developer must provide a function which changes the consumption parameters according to the input voltage.

All consumption information is stored locally at each device driver. This makes, together with the subscriber based registration of the devices by the manager and observers, the implementation flexible and changes in the granularity affect only the corresponding driver.

Fig. 5 shows the relevant parts and interfaces of the accounting system design. An example of the necessary changes to the existing driver code is also shown. The *energyAccountingManager* keeps track of the current system load. This information in needed by the *converterEfficencyObserver* to calculate the efficiency factor which is used in the calculation of the consumed energy. To track the converters base costs and adopt to the changed efficiency due to changes in the supply voltage, the *converterEfficencyObserver* is derived from the base class as other drivers.

The accounting system was inserted into the device drivers for the Texas Instruments eZ430-Chronos nodes. As timer base for the timestamp generation we used the 32kHz timer of the MSP430. This set the resolution for time at 1/32ms. Since nodes should run for a long period of time, the timestamps needed to use 64-bit values. The consumption of the different devices and modes spread over several orders of magnitude. To cover this we used 16-bit datatypes, resulting in a minimal current consumption at 1µA to a maximal current consumption of 65.53mA. The consumed energy is stored in 64-bit values to cover the entire lifetime of a sensor node.

5 Evaluation

To evaluate the approach several experiments using typical sensor node applications were performed. If not noted otherwise, each application was executed with the three accounting models and then the actual consumption of each application is measured on two different nodes.

The first application, *app1*, let the node sleep for a long period of time. Every two minutes the node woke up to sample a sensor (the internal voltage) using the ADC. The results and the accounting data were then sent to a base station. Additionally, the node activated the radio receive mode for 50ms during each interval.

The second application, *app2*, used a low power listening scheme to sample the radio channel every 250ms. Every 10 seconds the node sent the accounting data to the base station. Additionally, every 5 seconds the supply voltage was checked.

To test the supply voltage adaption the third application *app3* was used. For this application the node was equipped with an LED which was toggled every 500ms. The node checked the supply voltage every second and sent the accounting data every 10 seconds. *app3* was only evaluated with one node and the *m-Median* model, since no statistical data about the LEDs power consumption was available and the current consumption of the used LED was measured. The *m-Median* model was compared to the actual consumption for 3.0V and 2.2V. Additionally, an experiment was performed where the voltage was decreased from 3.0 to 2.2V in steps of 0.2V.

Application 4, *app4*, utilized the radio module and the CPU active state to test a first implementation of the converter efficiency adaption. The node stayed in the 8Mhz active state for 500ms and afterwards listened for messages for another 100ms. Additionally, the accounting data was sent with the maximum transmission power. An ON Semiconductor NCP1400 [23] was used for this experiment. Fig. 6 shows its measured efficiency at various supply voltage levels. For the experiment a supply voltage of 1.5V was used. The measured efficiency was translated to 6 steps (<100µA, <1mA, <3mA, <10mA, <20mA and >20mA) for the online adaption.

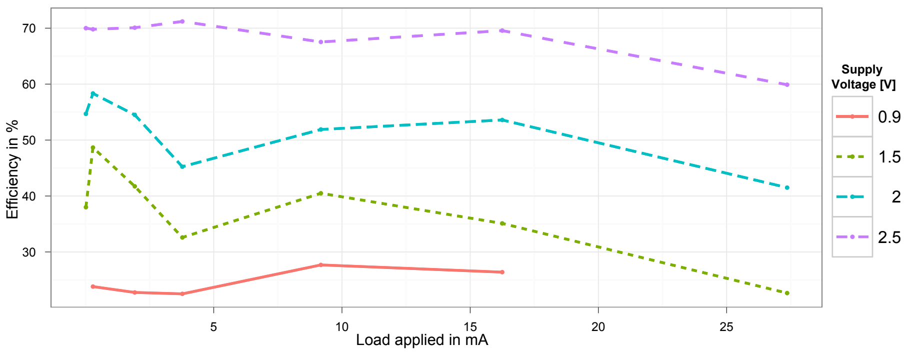


Fig. 6. Measured efficiency of step-up voltage converter NCP1400A33T1

To show how the accounting can be used by the application, *app5* monitores the consumed energy during its active phase. This information is used to dynamically calculate the sampling frequency based on available energy and desired runtime. With each invokation the node activated the radio receive mode for 50ms, sent a message to a base station and lit up an LED for 200ms.

In Table 2 the results for *app1* are shown. From the accounted consumption and runtime, the average consumption was calculated. The consumption predicted by the models is almost identical, except for the *m-Pessimistic*. It predicts a 20% higher consumption due to its parameters.

In Fig. 7 the predicted consumption is described. In all models the radio receive mode represents the biggest part of the consumption, followed by the low power sleep mode. The pessimistic model accounts for a bigger part to the sleep modes than the others. This behavior confirms to the expectations, since *app1* does little except sleeping. In comparison with the actual consumption measured for two devices, the *m-Pessimistic* model is better suited to the increased consumption of node B. Most likely this is due to a higher sleep consumption.

The results of application *app2* are shown in Table 2. As the results show, the difference between the three models is small. *m-Datasheet* and *m-Median* differ from each other by around 2%, while the difference between *m-Datasheet* and *m-Pessimistic* is 3%.

Compared with the actual measurements the *m-Datasheet* model performs quite well. It overestimates the actual consumption of node A by 2% and of node B by 1%. With 5% for node A, the *m-Pessimistic* model has the

Table 2. Average current of *app1* and *app2* compared with the accounted consumption

	Average current in mA				
	m-Datasheet	*m-Median*	*m-Pessimistic*	A	B
app1	0.01002	0.01014	0.01222	0.0098	0.0115
app2	0.32869	0.33556	0.33841	0.3221	0.3251

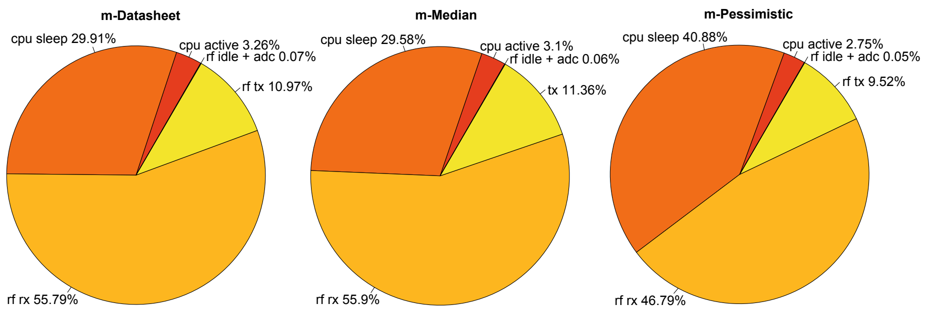


Fig. 7. Contribution of each device to the total consumption for each model with *app1*

highest overestimation. In Fig. 8 the predicted consumption is further described. The biggest part in the consumption is accounted for the radio receiving mode. Around 92% of the energy is spend to maintain the listening scheme of *app2*. The variation between the models is low, the ratio between the different consumers is almost the same. The pessimistic model accounts a little more for the sleep mode than the others, which reflects in the ratio.

As we further investigated the actual consumption, we noticed that radio receive mode behaves than expected. It seems that the actual transition to the receive state within the radio module takes almost 1ms in which the consumption is lower than the radio idle state, resulting in an overestimation of the radio consumption. We reduced this overestimation by waiting 500ns before starting the accounting for the receive mode. Additionally, sometimes and nondeterministic, the radio does not go into the receive mode after being commanded to, although the internal radio state machine thinks it is. We are currently searching for a solution to these problems to further improve the accounting mechanism.

Table 3 shows the results of the measurements of *app3*. For all three evaluated voltage scenarios, the difference between the accounted consumption and the

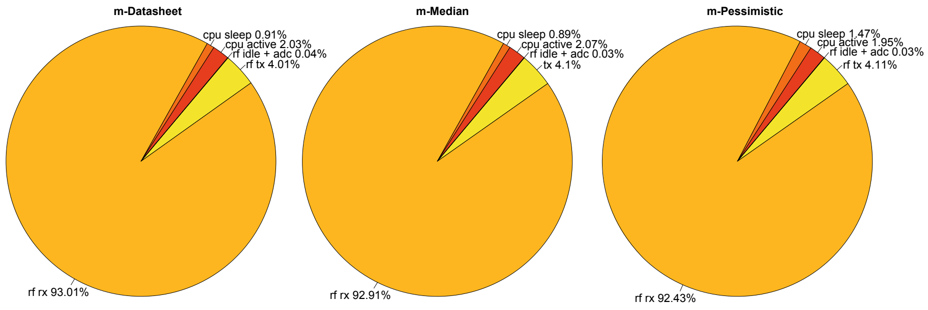


Fig. 8. Contribution of each device to the total consumption for each model with *app2*

Table 3. Current consumption of *app3* compared with the accounted consumption under different voltages using node A

	Average current in mA	
	m-Median	measured
3.0V	1.809	1.804
2.2V	0.750	0.743
3V down to 2.2V	1.280	1.272

actual measured consumption is below 1%. This is mainly motivated due to knowledge of the exact LED consumption and the huge influence of the LED on the total consumption, but shows that the *voltageObserver* works as intended.

Table 4. Current consumption of *app4* compared with the accounted consumption under different efficiency models

Average current in mA for node A			
measured	efficiency adoption	fixed worst efficiency	fixed best efficiency
5.139	5.301	8.707	4.198

In Table 4 the results of application *app4* are shown. The base consumption of the voltage converter is included in the average current. The difference between the actual consumption and the online accounting model with converter efficiency adaption is 3%. This value is considerably better than assuming a fixed efficiency. Using the best measured efficiency for 1.5V supply voltage, the consumption is underestimated by 22%. When using the worst efficiency the average current is assumed to be 68% higher than measured. While this experiment is simple and should be expanded in the future, the results show that an adaption to the converter efficiency is better suited to model the consumption than using a static model.

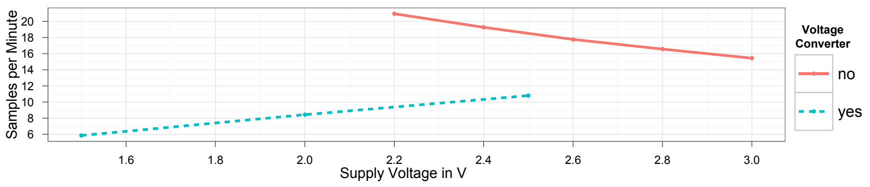


Fig. 9. Adaptive calculated sampling rate unter different voltages for *app2*

Fig. 9 shows how *app5* sets its frequency under differed supply voltages with and without a voltage converter. Without a converter the frequency increases with lower voltages since the LED consumption reduces. Although using a voltage converter makes it possible to use lower voltages, application frequency is lower in the beginning and decreases with lower voltages due to the reduced efficiency of the converter and the stable supply voltage to the LED.

6 Conclusion and Future Work

In this paper we have shown how the online accounting of energy can be made more robust against device variations. A high number of devices were measured and the results were used to build more realistic models and thus cover more nodes by introducing only little overestimation. Our approach is designed to be flexible to fit the level of accounting to the differed needs within the development phase and the real-world deployment. In contrast to other software based accounting approaches, our system is able to adopt to the changing consumption of a device. We proposed a mechanism for taking consumption changes due to decreased voltage into account. Additionally, an approach to overcome the variable efficiency of voltage converters due to dependencies on load applied and supply voltage was proposed. This makes it possible to use accounting not only for development but also for dynamic online power management in real-world deployments.

In the future we want to expand our accounting model to other devices and nodes, especially designated sensor nodes. Furthermore, we plan to use the information about the consumption the accounting gathers for fine-grained online energy management. Additionally, the approach will be evaluated in a real sensor network project for development as well as energy monitoring and management in the deployment.

References

1. Landsiedel, O., Wehrle, K., Götz, S.: Accurate prediction of power consumption in sensor networks. In: EmNets 2005 Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors (2005)
2. Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., Culler, D.: TinyOS: An operating system for sensor networks. In: Ambient Intelligence (2004)
3. Dunkels, A., Gronvall, B., Voigt, T.: Contiki - a lightweight and fexible operating system for tiny networked sensors. In: Proceedings of the First IEEE Workshop on Embedded Networked Sensors (2004)
4. Walther, K., Nolte, J.: A Flexible Scheduling Framework for Deeply Embedded Systems. In: Proc. of 4th IEEE International Symposium on Embedded Computing (2007)
5. Linden, D.: Handbook of batteries, 2nd edn. McGraw-Hill Companies (1995)
6. Sieber, A., Nolte, J.: Device Management for Limiting the Load Applied to Batteries, 11. GI/ITG KuVS Fachgesprch Sensornetze (2012)
7. Park, C., Lahiri, K., Raghunathan, A.: Batterydischarge characteristics of wireless sensor nodes: An experimental analysis. In: Proceedings of the IEEE Conf. on Sensor and Ad-hoc Communications and Networks (SECON), Santa Clara, pp. 430–440 (2005)

8. Park, S., Saviddes, A., Srivastava, M.B.: Battery Capacity Measurement and Analysis Using Lithium Coin Cell Battery. In: Proc. Int. Symp. Low Power Electronics & Design (2001)
9. Woehrle, M., Beutel, J., Lim, R., Yuecel, M., Thiele, L.: Power monitoring and testing in wireless sensor network development. In: Workshop on Energy in Wireless Sensor Networks (2008)
10. Texas Instruments, bq26231 Low Cost Battery Coulomb Counter For Embedded Portable Applications, Webpage http://www.ti.com
11. Jiang, X., Dutta, P., Culler, D., Stoica, I.: Micro power meter for energy monitoring of wireless sensor networks at scale. In: Proceedings of the 6th International Conference on Information Processing in Sensor Networks (2007)
12. Dutta, P., Feldmeier, M., Paradiso, J., Culler, D.: Energy Metering for Free: Augmenting Switching Regulators for Real-Time Monitoring. In: Proceedings of the 7th International Conference on Information Processing in Sensor Networks (2008)
13. Fonseca, R., Dutta, P., Levis, P., Stoica, I.: Quanto: tracking energy in networked embedded systems. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (2008)
14. Kellner, S., Bellosa, F.: Energy accounting support in TinyOS. 2. GI/ITG KuVS Fachgesprch Systemsoftware und Energiebewusste Systeme (2007)
15. Kellner, S.: Flexible Online Energy Accounting in TinyOS. In: Marron, P.J., Voigt, T., Corke, P., Mottola, L. (eds.) REALWSN 2010. LNCS, vol. 6511, pp. 62–73. Springer, Heidelberg (2010)
16. Dunkels, A., Osterlind, F., Tsiftes, N., He, Z.: Software-based on-line energy estimation for sensor nodes. In: Proceedings of the 4th Workshop on Embedded Networked Sensors (2007)
17. Cho, Y., Kim, Y., Chang, N.: PVS: passive voltage scaling for wireless sensor networks. In: Proceedings of the Symposium on Low Power Electronics and Design (2007)
18. Wanner, L., Apte, C., Balani, R., Gupta, P., Srivastava, M.: A case for opportunistic embedded sensing in presence of hardware power variability. In: Proceedings of the 2010 International Conference on Power Aware Computing and Systems (2010)
19. Texas Instruments, eZ430-Chronos Development Tool Datasheet, Webpage http://www.ti.com
20. Texas Instruments, CC430F61xx 16-Bit Ultra-Low-Power MCU Datasheet, Webpage http://www.ti.com
21. Hitex Development Tools GmbH, PowerScale Datasheet, Webpage http://www.hitex.com
22. HAMEG Instruments GmbH, HM8143 Datasheet, Webpage http://www.hameg.com
23. ON Semiconductor, NCP1400A Datasheet, Webpage http://onsemi.com