Demo Abstract: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes

Ulf Kulau*, Stephan Friedrichs † and Lars Wolf*

*Technische Universität Braunschweig
Institute of Operating Systems and Computer Networks (IBR)
Mühlenpfordtstr. 23, 38106 Braunschweig, Germany
Email: [kulau,wolf]@ibr.cs.tu-bs.de

†Max Planck Institute for Informatics
Saarbrücken, Germany
Email: sfriedri@mpi-inf.mpg.de

***Abstract*—While existing work focuses on the transceiver and the processing unit to increase the energy efficiency of wireless sensor nodes, it is missed that peripheral energy consumption may dominate that of the entire node. Related to Dynamic Voltage Scaling (DVS), even peripherals' energy efficiency benefit from a downscaled voltage level, but different peripherals require different minimum voltage levels. With this demo we combine theory and practice to present the implementation of an algorithm weighing off the benefits of a downscaled voltage level against the switching overhead, e.g. for calculating an optimal peripheral voltage schedule.**

I. Introduction

As the dynamic power consumption of CMOS gates shows a quadratic dependency on the voltage level, DVS helps to significantly improve the energy efficiency of microelectronic systems [1]. Hence, several existing DVS approaches [2], [3] lead to an increase of WSN lifetime. Nevertheless, not only MCUs but also peripherals like memory devices, sensors, or actuators benefit from a downscaled voltage level.

Each peripheral hardware device requires a minimum voltage to be properly operated. The common practice is to statically configure the lowest peripheral voltage conform to all peripheral devices' voltage requirements. This can be very inefficient, because most hardware consumes more energy when exposed to higher voltage. Hence, we seek to exploit a sensor node's mechanism to dynamically switch the peripheral voltage.
The crux is that switching the voltage does not come for free. If it would, one could simply operate every peripheral device with its minimum required voltage. But switching the voltage consumes energy as well: The additional time interfacing a scalable voltage supply prolongs the duty-cycle of a processing unit, leading to a higher energy consumption.
The concept of incorporating switching cost among power different radio modes was discussed in [4], but is focussed on the radio transceiver only. In the following we outline the algorithm introduced in [5] which will be showed in our demo. This algorithm allows for peripheral voltage scheduling that weighs off the energetic benefits of switching to a lower peripheral voltage against the switching overhead without violating the minimum voltage requirements of active hardware.
Consider a sensor node with a set S of peripheral hardware devices. In order to assess the benefits of switching to a lower peripheral voltage before using $s \in S$, we need to know how much energy is consumed when querying s using the peripheral voltage v. $e_s(v)$ depends on the time t_s necessary to query s, the peripheral voltage v, and the accumulated current $I_s(v,t)$ flowing through s as well as through the inactive peripheral hardware $S \setminus \{s\}$:

$$e_s(v) = v \int_0^{t_s} I_s(v,t)\,\mathrm{d}t \tag{1}$$

Each $s \in S$ has two attributes: 1. a minimum voltage $v_{\min}(s)$ required to properly operate s, and 2. the energy consumption $e_s(v)$ of all peripherals while only s is active, depending on the peripheral voltage v, see above. Throughout this work, we assume $e_s(v)$ to be a monotonically increasing function, i. e., that a reduction of the peripheral voltage never results in an increased energy consumption. For a constant amount C of energy, the *switching overhead*, the sensor node can adapt its peripheral voltage. The sensor node is presented a sequence of queries denoted by $[1,\dots,n]$, so that the energy consumption E of a voltage schedule is given by:

$$E = \sum_{i=1}^{n} e_{s_i}(v(i)) + \sum_{i=2}^{n} \begin{cases} C & \text{if } v(i-1) \neq v(i), \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Our goal is to minimize E, so we call a voltage schedule *optimal* if E is minimal. It follows from the monotonicity of $e_s(v)$ that an optimal schedule only uses $v(i) \in \{v_{\min}(s) \mid s \in S\} = \{V_1,\dots,V_m\}$ with $V_1 < \ldots < V_m$.

II. Algorithm

Let us, for a pair of query and voltage (i, V_j), determine the minimum amount of energy $E_{i,j}$ necessary to reach it using a feasible schedule $v(1),\dots,v(i)$ while assuming an infinite energy consumption for infeasible configurations. For the first query, we have:

$$E_{1,j} = \begin{cases} \infty & \text{if } V_j < v_{\min}(s_1), \\ e_{s_1}(V_j) & \text{otherwise.} \end{cases} \tag{3}$$

For $2 \leq i \leq n$, there is the mandatory energy consumption $e_{s_i}(V_j)$ to answer the query i itself, as well as the accumulated costs for traversing $i-1$ preceding configurations. There are two ways to reach the configuration (i, V_j) with an optimal energy consumption: Either the peripheral voltage from the previous query is kept, or it is changed. The former case yields an additional energy consumption of $E_{i-1,j}$. In the latter case we require the minimum amount of energy $\hat{E}_{i-1}$ to reach the cheapest feasible predecessor configuration and the additional costs C for switching the voltage, where $\hat{E}_{i-1} = E_{i-1,\hat{j}}$ with $\hat{j} := \arg\min_j E_{i-1,j}$. This yields, for $2 \leq i \leq n$:

$$E_{i,j} = \begin{cases} \infty & (i, V_j) \text{ is infeasible,} \\ e_{s_i}(V_j) + E_{i-1,j} & E_{i-1,j} < \hat{E}_{i-1} + C, \\ e_{s_i}(V_j) + \hat{E}_{i-1} + C & \text{otherwise.} \end{cases} \quad (4)$$

We use dynamic programming to efficiently solve the recursion by determining $E_{i,\cdot}$ before $E_{i+1,\cdot}$; the optimal overall schedule is that ending in the configuration (n, V_j), where $E_{n,j} = \hat{E}_n$ is minimal. For a detailed description of the algorithm and some extensions please refer to [5].

III. Implementation

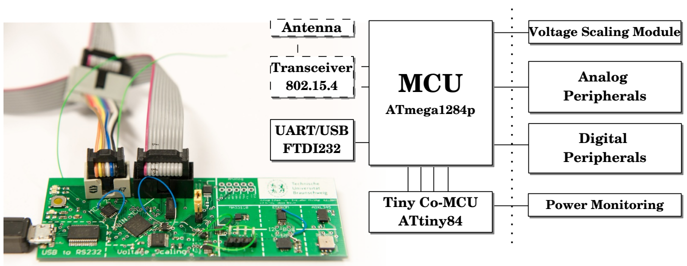

Figure 1. Block diagram and a picture of the actual prototype implementation.

Related to the INGA sensor node [6], we use an 8-bit Atmel ATmega1284p MCU as processing unit. Thus, the low computational capabilities of this MCU demonstrates that our approach is sufficiently lightweight for WSN requirements.
Figure 1 shows a picture and a block diagram of the prototype. Compared to ordinary sensor nodes a voltage scaling module is connected to the processing unit via I2C-bus. This module provides a voltage level of $1.8\,\text{V} \leq v \leq 3.3\,\text{V}$ with an 8-bit resolution to the peripherals. In this case, the overhead of switching to an arbitrary voltage level is $C \approx 7.76\,\mu\text{J}$. This includes the increased active time of the MCU and the voltage scaling module's static power dissipation, refer to [3], [5] for details.
Our prototype's sensing unit is divided into an analog and a digital section. The analog section offers the ability of connecting fully analog sensors to the ADC channels of the ATmega1284p, while the digital section includes the devices of Table I. All of them are connected via I2C bus. In order to calculate an optimal schedule, we need information describing the overall peripheral energy consumption.For this reason, we added a tiny co-MCU to the prototype, which is able to concurrently sample the current consumption of the peripherals (a shunt is used in connection with current sense amplifiers) and to measure the time (the co-MCU can be triggered by the ATmega1284p via digital GPIOs). Hence, $e_s(v)$ can be measured for any given values of s and v.

Table I. Equipped peripherals for demonstration.

Peripheral s	Device	Description	$v_{\min}(s)$ [V]
A	ADXL345	Accelerometer	2.000
E	AT24C08C	EEPROM	1.800
P	BMP085	Pressure Sensor	1.800
G	L3G4200D	Gyroscope	2.400
M	MAG3110	Magnetometer	1.950

IV. Evaluation

Although the demonstration will give the user already the chance to optimize custom schedules, the following table depicts some exemplary schedules to show the general benefit of our approach. The energy savings are compared against three classical strategies. ConstDefault is what happens when a sensor node has no mechanism to adapt the peripheral voltage. A constant peripheral voltage of 3.3 V is kept. ConstMaxMin is the trivial strategy that uses the maximum minimum voltage, i. e., $\max_{s \in S} v_{\min}(s)$, for all queries. AlwaysSwitch always switches the voltage to its minimum requirement. It ignores the switching overhead.

Table II. Sample schedules to show the benefit of peripherals' voltage scheduling compared to naive approaches.

	Energy saved by Scheduled compared to		
Query Sequence	ConstDefault	ConstMaxMin	AlwaysSwitch
AEPGMAEPGM	45.80 %	17.13 %	0.97 %
GAMGAMGAM	46.15 %	17.04 %	0.49 %
GAMPE	46.91 %	18.52 %	1.40 %
GPGPGPGPGP	31.54 %	0.00 %	20.29 %
PAMPE	47.90 %	20.41 %	2.53 %

V. Demonstration

With a GUI a custom query of peripherals (cf. Table I) can be created. This query is transferred to the prototype board via USB. As the implementation follows a fully self-optimizing approach, the prototype board firstly self-parametrizes the energy functions $e_s(v)$ of involved peripherals. Afterwards, the board executes the optimization algorithm to get the optimal voltage schedule for the given query. Finally the query is processed while the optimal schedule is compared against the trivial voltage strategies as described in the previous section. For this purpose, the second tiny MCU samples the current consumption of ConstDefault, ConstMaxMin, AlwaysSwitch and of course Scheduled. The results are send back to the PC, where the GUI displays an oscilloscope of the current consumptions as well as an analysis of the saved energy.

References

[1] U. Tietze and C. Schenk, *Electronic Circuits: Handbook for Design and Application*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[2] L.B. Hoermann, P.M. Glatz, C.Steger and R.Weiss, "Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling," *Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European*, pp. 1 –8, april 2011.

[3] U. Kulau, F. Büsching, and L. C. Wolf, "A node's life: Increasing WSN lifetime by dynamic voltage scaling," in *The 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013 (IEEE DCoSS 2013)*, Cambridge, USA, May 2013.

[4] R. Jurdak, A. G. Ruzzelli, and G. M. P. O'Hare, "Radio sleep mode optimization in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, 2010.

[5] S.Friedrichs, U.Kulau and L.Wolf, "Energy-efficient voltage scheduling of peripheral components on wireless sensor nodes," in *Communications Workshops (ICC), 2014 IEEE International Conference on*, June 2014, pp. 860–865.

[6] F. Büsching, U. Kulau, and L. Wolf, "Architecture and Evaluation of INGA - An Inexpensive Node for General Applications," in *Sensors, 2012 IEEE*. Taipei, Taiwan: IEEE, oct. 2012, pp. 842–845.