System Design Impacts on Battery Runtime of Wearable Medical Sensors

Andreas Tobola[1,2], Oliver Korpok[1], Heike Leutheuser[3], Björn Schmitz[1], Christian Hofmann[1], Matthias Struck[1], Christian Weigand[1], Björn M. Eskofier[3], Albert Heuberger[1], and Georg Fischer[2]

[1] Fraunhofer Institute for Integrated Circuits IIS, Erlangen, Germany
http://www.iis.fraunhofer.de/en/ff/med.html
[2] Institute for Electronics Engineering, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany
http://www.lte.e-technik.uni-erlangen.de
[3] Digital Sports Group, Pattern Recognition Lab, Department of Computer Science, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany
http://www5.cs.fau.de

Abstract. Building a wearable sensor from scratch is challenging work. It requires an experienced team and besides this precise requirements to save development cost. Silicon designers gave us opportunities to build ultra-low-power systems. One of the hardest decisions, is the selection and configuration of an appropriate microcontroller. We address in this work the key concepts to design wearable sensors and which common low-power concepts are not applicable. Data sheets are good for preselection, a qualified decision requires an application experiment.

Keywords: wearables, body sensor, benchmark, performance, power consumption, energy, biomedical signal processing, microcontroller, RTOS

1 Introduction

The interest for wearable technology has increased in the last few years. Characteristics of wearable sensors are their miniature design and hence, the possibility for wearing them at various body positions e.g. integrated in clothes. Areas of applications for wearable sensors range from monitoring patients in home or outdoor environments to sports science research where athletes aim for performance enhancement [6].

A majority of the wearable sensors currently on the market comprise a battery runtime that lasts at best a couple of days. Many communities regret this as an unsolved issue [2]. For longer insights, the system design of wearable sensors can be optimized to achieve longer battery runtime with satisfactory sensor data results.

A body sensor node requires energy for the sensor itself, signal conditioning, data processing and wireless transmission [4]. For instance ECG sensor circuits are available as integrated circuits at 510 μW including conditioning and can be shut down to 120 nW [7]. Furthermore special ECG circuit designs run at 2.8 μW [5]. To reduce power of the radio module the key is to optimize the transmission duty cycle [4, 9, 11, 10]. The microcontroller is the heart of a sensor node. Beside data processing, sensor nodes have to handle an increasing number of additional functional and safety requirements. This is done by the microcontroller and requires additional computing power. CoreMark [1] is state of the art to compare computing cores. The aim of this work is to evaluate the application specific power consumption of state of the art microcontrollers for wearable sensors, especially regarding biomedical signal processing.

2 Methods

2.1 Power Profiles

The power consumption of a microcontroller is a highly dynamic (from nano- to milliampere) and fast (microseconds) process. To estimate parameters from the power consumption progress we defined a measuring procedure in six steps: 1) Microcontroller setup for low-power mode. 2) Measurement of the current in low-power mode. 3) Start of the benchmark code on the microcontroller and verification of the algorithm output. 4) Recording of the supply voltage, current and events at 2.5 MSa/s. 5) Measurement of an average current using a precise multimeter. 6) Transfer of all recorded data into MATLAB for calculation of timing, power and energy parameters for each time segment.

We defined six time segments for the measured power profiles:

1. **Start-up:** Time segment when the microcontroller changes from low-power mode to run mode; lasts until the oscillator is stable.

2. **Acquisition:** Time segment while the analog-to-digital converter works.
3. **Transition:** Transition between acquisition and processing.
4. **Processing:** The microcontroller is processing the acquired data.
5. **Release:** Time segment between the command to enter low-power state and power consumption reaches low-power level.
6. **Low-power:** The microcontroller is in the low-power state again.

Fig. 1 shows an example recording with automatically detected time segments. In addition to the graphic evaluation, MATLAB calculates a set of parameters for each time segment. This allows us a more differentiated interpretation of the process behind the average power consumption.

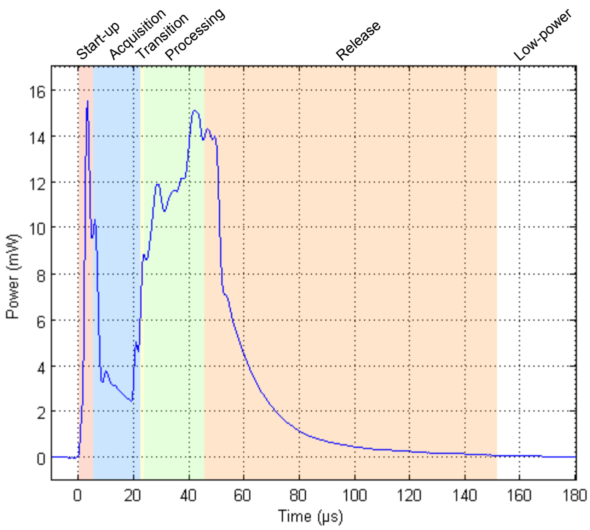

Fig. 1. Example power profile for EFM32WG990F256 at 14 MHz core clock using IAR compiler and benchmark algorithm.

The power measurements were made at a sampling rate of 2.5 MSa/s using Tektronix TDS5054B oscilloscope. As our process is periodic, we take advantage of the curve averaging function of the oscilloscope. 128 waveforms are averaged. A Tektronix ADA400 differential preamplifier has been used to measure the voltage drop over a shunt (1 Ω). Additionally two reference measurements were made using a Metrahit 26 S multimeter. The first measurement to compensate the systematic error of the differential amplifier. For a given range from 500 nA to 3 μA, the multimeter has an accuracy of $\pm$53 nA. The second reference measurement was made to verify the average current. For a given range from 15 μA to 2 mA, the multimeter has an accuracy of $\pm$2.5 μA. A principal schematic of the measurement setup is shown in Fig. 2. During the recording of the power curve the multimeter was removed from the power supply. All tests has been made at 3.0 V ($\pm$ 20 mV) power supply.

In order to get high resolution power profiles we reduced the recommended decoupling capacitors to 200 nF. To compensate voltage drops, we added a capacitor of 47 μF in front of the the shunt, keeping the shunt close to the microcontroller. To monitor the profiling process quality, the MATLAB script monitors the maximum drop of the supply voltage and the deviation to reference measurements.

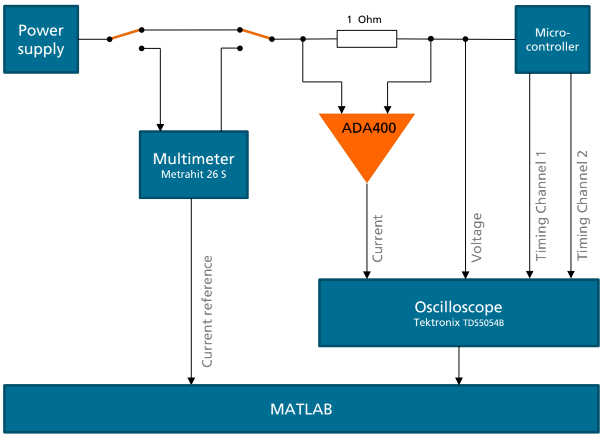

Fig. 2. Measurement setup to record microcontroller current, voltage and timing events at 2.5 MSa/s using a oscilloscope and a precision multimeter as reference.

2.2 Benchmark Algorithm

We used a benchmark algorithm from the project “Smart Sensors A”. The algorithm was designed to detect apneas in respiratory signals. The algorithm qualifies for benchmarks of biomedical signal processing due to its simplicity and diversity. The algorithm covers linear and nonlinear filters, as well as floating point and integer operations. Before the benchmark began, we have fed the algorithm with test data and verified the expected results to guarantee, that the algorithm works appropriate with the tested microcontroller, compiler and optimization level.

2.3 Sampling Method

Silicon designers provide a variety of interfaces and techniques to acquire data from the analog world. In this work the built-in analog-to-digital converters of the microcontrollers have been used. We used an interrupt driven method to sample data. For the MSP430 we used the sleep mode LPM4 during time segment “low-power”, and LPM3 during AD-conversion. On EFM32 we used EM2 during “low-power”, and EM1 during sampling. To realize fast

Table 1. Average microcontroller power consumption in μW depending on microcontroller type and compiler including individual absolute measurement error in μW.

	TI	IAR	GNU	Keil
MSP430	443.4 ± 29.2	390.4 ± 8.1	2555 ± 29.1	-
Cortex-M0+	-	100.5 ± 9.3	158.1 ± 10.1	94.2 ± 6.8
Cortex-M3	-	157.1 ± 2.1	190.4 ± 1.1	152.0 ± 1.7
Cortex-M4F	-	79.0 ± 0.6	88.7 ± 1.8	78.4 ± 2.1

wake-up times from deeper sleep modes, the internal resistor-capacitor-oscillator HFRCO has been used at EFM32 and the DCO/FFL at MSP430. To get a precise sampling timing with low jitter, an additional crystal oscillator at 32 kHz has been used to clock a low-power RTC timer. The sampling has been done at 128 Sa/s for all tests.

2.4 Tested Devices and their Configuration

Tested microcontrollers:
- MSP430F5659
- EFM32ZG222F32 (ARM Cortex-M0+)
- EFM32GG990F1024 (ARM Cortex-M3)
- EFM32WG990F256 (ARM Cortex-M4F)

ARM-C-compiler and version, followed by optimization levels:
- IAR 7.20.2.7431, High, Speed
- GNU 4.7.3, -O3
- Keil, High, Time

MSP430-C-compiler and version, followed by optimization levels:
- TI 4.3.2, -O4
- IAR 6.10.2, High, Speed
- GNU 4.8.0, -O3

Analog-to-digital converter configuration:
- MSP430: clock: 5 MHz, 2.5 V Ref., 12 Bit, sample hold time: 32 cycles
- EMF32: clock: 1/5/7 MHz (constant per experiment), 2.5 V Ref., 12 Bit, Aqu.Time: 32 cycles

3 Results

Experiment 1: We tested four microcontrollers using the benchmark algorithm. Each microcontroller was tested with selected compilers (Table 1). For each test run the estimated consumption profiles have been measured using the method described above. In this experiment all core clocks were set to 14 MHz.

Experiment 2: We used the interrupt driven sampling method to sample data using the internal AD-converter. The EMF320WG allows to select six fixed RC-oscillator frequencies for the core clock. For higher frequencies as 28 MHz an external clock source is needed. This is a crystal oscillator at 48 MHz in our case. The results are listed in Table 2.

Table 2. Energy portions to process one sample of data, start up time and transition time depending on core clock frequency and oscillator type (RC: resistor-capacitor; XO: crystal).

Core clock	Proc. time	Proc. energy	Start-up energy	Trans. energy
1.2 MHz RC	253.6 μs	376.5 nJ	171.9 nJ	60.0 nJ
6.6 MHz RC	46.4 μs	351.6 nJ	116.8 nJ	51.7 nJ
11 MHz RC	28.0 μs	320.8 nJ	103.0 nJ	43.3 nJ
14 MHz RC	21.6 μs	300.4 nJ	97.5 nJ	41.3 nJ
21 MHz RC	16.8 μs	305.1 nJ	92.5 nJ	35.3 nJ
28 MHz RC	12.8 μs	271.4 nJ	71.2 nJ	33.1 nJ
48 MHz XO	4.4 μs	163.3 nJ	987.1 nJ	$<$13.2 nJ

Table 3. Average microcontroller power consumption depending on usage of RTOS.

	Transition time	Release time	Average power consumption
Without RTOS, interrupt driven	1.2 μs	54.8 μs	76.8 μW
FreeRTOS, interrupt driven	3.6 μs	69.6 μs	579.6 μW
FreeRTOS, tickless, interrupt driven	1.2 μs	85.6 μs	132.0 μW
FreeRTOS, tickless, using semaphores	34.4 μs	133.6 μs	304.6 μW
Keil RTX, tickless, interrupt driven	3.6 μs	89.6 μs	156.1 μW

Experiment 3: We tested the impact of an real-time operating system (RTOS). The first measurement is interrupt driven sampling without RTOS. The next test is using FreeRTOS in default mode; then in tickless mode as described in [8]. We also tested an approach using semaphores to take out processing load from interrupt handler. And finally we compared Keil RTX to the best case of FreeRTOS. For evaluation (Table 3) we compared the transition and release times which have the highest impact on the average power consumption.

4 Discussion

Our aim was to create an application-dependent decision guide to select the appropriate microcontroller for wearable sensors. Several types of the MSP430 microcontroller have fulfilled our requirements in various projects for years. But higher requirements, additional features and multi-parameter sensor designs led us to search for alternatives. More powerful senors with real-time operating systems and larger SRAM are necessary. But, does this meet the requirements for low-power designs? Yes, of all twelve tested cases in Table 1 the most powerful microcontroller core (ARM Cortex-M4F) is the most efficient solution for our applications. Therefore, the compilers IAR and Keil deliver equally the best results. The second choice would be the Cortex-M0+. Comparing only the CoreMark-Power-Ratio[4], the M0+ seems to be the most efficient, but due to floating point operations, the M4 with floating-point unit consumes significantly less power in our application test. Beside this, it should be taken into account, that the M0+ does not support saturation arithmetic [3], which might be a safety issue for medical devices using fixed-point operations.

In Table 2 we tested for the most efficient core frequency. Comparing only the processing energy and time, the maximum available core frequency would be the best choice. But the energy cost for start-up using a crystal oscillator is much higher as all time segments together. Thus, the highest clock using a RC-oscillator is the most efficient setup.

In the last experiment we tested impacts of an operating system on power consumption. The tested operating systems extend the transition and release time segments. Beside this, taking an RTOS in default mode would lead into a waste of energy. Thus, an RTOS with tickless mode is highly recommended for low-power sensor nodes. Semaphores can be used, but should be taken with care. Especially in this case a processing task should not run for each sample. Finally, in our test cases Keil RTX and FreeRTOS did not deliver measurable differences.

We defined six time segments to analyze the process behind the average power consumption. The key of low-power systems is to keep the microcontroller as long as possible [3] in time segment "Low-power". During this time the deepest low energy mode possible should be used. As wearable sensors have low data rates, common low-power methods like direct memory access (DMA) or the peripheral reflex systems (PRS) require more power as the interrupt driven sampling technique. DMA is recommended for sampling rates higher than 5 kSa/s [12]. Common biomedical signal sensors have sampling rates up to 1 kSa/s [4]. Thus, we decided to use a interrupt driven method as best choice for biomedical signal processing. This allows us to use the next deeper sleep mode, which results in 45 times less average power consumption as driven by PRS or DMA.

[4] CoreMark-Power-Ratio defined as CoreMark/MHz normalized by mW/MHz, which gives CoreMark/mW as core power efficiency scale.

5 Conclusion

The best solution for our low-power biomedical signal processing applications is the ARM Cortex-M4F clocked at the highest available RC-oscillator for the core and a 32 kHz crystal oscillator for precise interrupt-driven sampling. The use of an RTOS is necessary to handle all functional and safety requirements. Following the discussed aspects above an RTOS takes a acceptable amount of energy.

However, the major impact on power consumption is given by non-technical requirements (e.g. transmission duty cycle). Extended battery runtime of wearable biomedical sensors is less a question of technology, it is a question of use case decisions, made in cooperation with stakeholders (scientists, physicians, customers, product managers, users) and engineers [13].

References

1. The Embedded Microprocessor Benchmark Consortium, `http://www.eembc.org`, seen Oct. 2014.
2. Bouchaud, J.: MEMS and sensors for wearable electronics, GSA European Executive Forum (2014)
3. Martin, T.: The designer's guide to the Cortex-M processor family: A tutorial approach (2013)
4. Yang, GZ.: Body sensor networks. SpringerLink (2014)
5. Sarpeshkar, R.: Ultra low power bioelectronics: Fundamentals, biomedical applications, and bio-inspired systems. Cambridge University Press (2010)
6. Leutheuser, H., Schuldhaus. D., Eskofier, BM.: Hierarchical, multi-sensor based classification of daily life activities: Comparison with state-of-the-art algorithms using a benchmark dataset. PLos ONE, vol. 8, no. 10, pp. e75196, (2013)
7. Analog Devices: AD8232: Single-lead, heart rate monitor front end. Rev. A. (2013)
8. Tverdal, M.: Operating system directed power reduction on EFM32, masters thesis (2010)
9. Kindt, P., Yunge, D., Diemer, R., Chakraborty, S.: Precise energy modeling for the Bluetooth Low Energy Protocol. archiveX.org (2014)
10. Schwarzmeier, A., Reuter, L., Mena-Carrillo, J. , Weber, W., Fischer, G., Weigel, R., Kissinger, D.: A Low-power embedded communication platform for medical applications. Biomed Tech (2012)
11. Reuter L., Schwarzmeier A., Weigel R., Fischer G.: Energy budget in mobile health assistance systems. Biomed Tech (2012)
12. Silicon Labs: EMF32 analog to digital converter. Application note AN0021, Rev. 1.09 (2013)
13. Tobola, A., Hofmann, C., Weigand, C.: Requirement engineering in health care and telemedicine. Proceedings BMT2012, ISSN (Online) 1862-278X, ISSN (Print) 0013-5585 (2012)