Network-Wide Energy Optimization for Adaptive Embedded Systems

Patrick Heinrich
Fraunhofer Institute for Communication Systems ESK
Hansastraße 32
80686 Munich, Germany
+49 89 547088-383
patrick.heinrich@esk.fraunhofer.de

Christian Prehofer
Fraunhofer Institute for Communication Systems ESK
Hansastraße 32
80686 Munich, Germany
+49 89 547088-352
christian.prehofer@esk.fraunhofer.de

ABSTRACT
This paper discusses network-wide energy optimization of embedded systems which can adapt by switching configurations. We model applications and their task chains in a network of embedded devices, including sleep modes and change of configuration, which provides a basic adaptation mechanism. We present a formal model of the energy consumption for such systems and apply it to automotive embedded systems. In particular, we develop new potential for network-wide energy savings as well as optimizations for adaptive systems. For instance, we show that non-optimal configurations may lead to a globally optimal system setup, if a system adapts regularly.

Categories and Subject Descriptors
C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based Systems – *real-time and embedded systems.*

General Terms
Design.

Keywords
Embedded systems, energy-efficiency, network-wide optimization, adaptive systems, automotive.

1. INTRODUCTION
This paper discusses energy optimization of embedded systems which can adapt and switch configurations. While there is considerable work on energy efficiency for embedded systems, we show that there is a need to model and optimize the network-wide energy consumption of such embedded systems.

We focus on automotive systems which consist of a number of electronic control units (ECUs), connected via a communication network. Applications (e.g. collision warning) consist of a chain of tasks, residing on different electronic control units. For each task chain (i.e. application) different configurations are possible. We consider the change between different applications (and their corresponding configuration) at runtime as an essential adaptation mechanism. The change between applications and their active periods are usage dependent.

In our setting, ECUs have to execute real-time tasks. Systems with hard real-time constraints must meet the given deadlines. We assume that the individual tasks are given by worst-case execution time (WCET), their deadline, cycle time and the dependency on other tasks including the necessary communication. The vehicle applications are typically decentralized, i.e. parts of application software (tasks) are executed at different ECUs. This also means that different task allocations are possible for vehicle software.

Most energy saving technologies optimize the energy consumption of one component (i.e. CPU), mostly at runtime. Examples are dynamic hardware resource management and dynamic voltage/frequency scaling. Other methods are optimizing the efficient use of the resources, such as energy-efficient task scheduling [1, 2] or trade-offs, e.g. reducing quality of service to decrease energy consumption [3, 4].

A lot of research focuses on the reduction of system-wide energy consumption, which means ECU-wide in this paper. For instance dynamic power management (DPM), which deactivate unused components of a system to save energy (e.g. [5]). [6] uses dynamic voltage scaling and DPM to reduce energy consumption system-wide and considers the energy consumption of peripherals (e.g. memory) within standby and activation/deactivation time of the processor. A software framework to use the different hardware energy-saving techniques and find a trade-off between those is presented in [7]. This optimization aims to reduce system-wide energy consumption for different applications during runtime, but the energy for reconfiguration is neglected. The possibilities of network-wide energy optimization are not used in these works. A network-wide optimization is considered within wireless sensor networks, but with the focus of energy efficient message routing.

The intention of this paper is to show the potentials and challenges of network-wide energy optimization of adaptive systems by an analytical model. In particular, we show by example that optimizations for single applications (task chains) may not result in network-wide optimal combination of several, alternating applications. We also show cases where additional hardware may reduce the energy consumption. Our model is based on the analysis of the energy consumption of the individual components and their dependencies. In this paper, we use existing data of their energy consumption. Detailed measurements for specific components and activities like networking are non-trivial and go beyond the scope of this paper.

2. NETWORK-WIDE ENERGY OPTIMIZATION
In this section, we discuss factors and challenges to model energy consumption. Besides the CPU power consumption, we consider

the energy to communicate via a network and the energy during sleep modes of components. This hardware may also have additional impact on the energy consumption of a system. An example is the calibration time of sensors, which is necessary to assure accurate results. During that time the ECU has to process a neglectable workload, but changing to sleep mode is typically not possible. In this case, an ECU-wide optimization is difficult, because the local tasks need calibrated sensors. Network-wide optimization is able to allocate additional tasks to that hardware to use the unused ECU resources during calibration time.

The challenge of network-wide optimization is to model the energy consumers and their dependencies correctly. Assuming a specific set of task chains, one for each application, we aim to find the right assignment of tasks to ECUs and the timing of sleep modes to ensure lowest energy consumption. This is a challenge because of the large number of parameters and aspects. A vehicle may have hundreds of functions and even a single hardware has different energy modes. As an example, [8] has five different energy modes.

We consider adaptive systems where the active applications, expressed as task chains, may vary over time. For each set of active task chains, we have several possible configurations and one has to be selected. Alternating task chains may then switch between different active task chains or just between different configurations of the same task chains.

Changing task chains means activating and deactivating tasks and, if necessary, changing the network configuration, which both consume energy.

Furthermore, the energy consumption depends on the usage of the system. If a configuration is used for a very short time, energy-efficiency is less critical w.r.t. the overall system.

3. MODELLING ENERGY CONSUMPTION

In this section, we present a model for adaptive, network-wide energy consumption of embedded systems. A simple, running example shall show the energy consumption of different configurations and the advantages of network-wide and usage behavior dependent optimization. The illustrated system is part of a vehicle application which changes its set of task chains (i.e. the applications) depending on external conditions. To simplify the example, the system just has one active task chain at any time. Tasks are executable at every node, except they need local resources (i.e. sensors). For simplicity, we assume that the execution time only depends on CPU frequency and it is assumed that a valid schedule can be found.

In our case study, the first application is object identification using radar sensors, which is used for adaptive cruise control (ACC). ACC uses intelligent object detection (e.g. for obstacles). The ACC task chain uses two radar sensor-ECUs and two ECUs which execute the necessary tasks (T) and communicate using a network. Figure 1 shows a configuration (C1a) and its hardware and the tasks with its dependencies. Task T_0 captures the radar data and preprocess these. Task T_1 identifies objects recorded by the radar sensor and task T_2 compares the results of the object identification. Tasks T_1 and T_2 are used in two instances at the two sensors in the example. Task T_3 is part of an application specific function, e.g. ACC, which is out of scope of this case study.

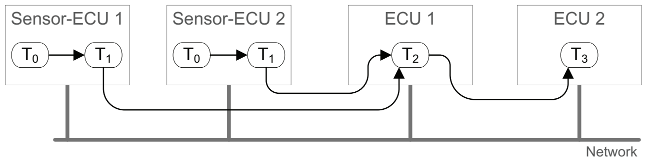

Figure 1: Task Chain ACC - Configuration C1a

Assuming ACC is not usable with a vehicle speed below 30 km/h the radar sensor is used for collision warning in this case. Thus we alternate between ACC and the second application, collision warning, in this example. The collision warning application which is less safety critical uses just one radar sensor. This task assembled as configuration (C2a) is shown in Figure 2.

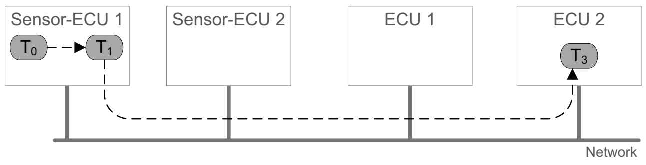

Figure 2: Task Chain Collision Warn. - Configuration C2

The tasks and the communication between these have specific parameters. We assume the worst-case execution time at a specific frequency is convertable to the task execution time t_e at frequency f_{te}. The cycle time t_c and the necessary communication per cycle c_t between tasks are also shown in Table 1.

Table 1: Task Properties

Task	T_0	T_1	T_2	T_3
Worst-Case Execution Time	10 ms @ 82 MHz	15 ms @ 82 MHz	10 ms @ 132 MHz	n/a
Cycle Time	100 ms	100 ms	100 ms	n/a
Communication per Cycle	$T_0 \rightarrow T_1$: 192 kBit	$T_1 \rightarrow T_2$: 32 kBit	$T_2 \rightarrow T_3$: 16 kBit	

3.1 Modeling Energy-Consumers and System Assumptions

In this section, we detail the energy consumers of our case study. Hardware has a processing speed f_{hw} and dependent power consumption P_{hw}. Highly energy efficient processors exist such as [8], which consumes 180 µA/MHz. Other embedded hardware without energy-awareness such as [9] consumes about 5 mA/MHz. Assuming a medium energy-efficient hardware the sensor hardware, which is used here, consumes 800 µA/MHz. The sensor-ECUs work with 82 MHz and a supply voltage of 1.65 V, which result in a power consumption of 108 mW. We assume a more efficient ECU hardware. This needs 500 µA/MHz and consumes 109 mW at 132 MHz and 1.65 V supply voltage. Both components are able to change to a sleep mode, during which the hardware consumes power P_s of 82.5 µW (50 µA). The change to the sleep mode itself and back to normal consumes time t_{cs}, which is supposed to be 5 ms with the power P_{cs} of 108 mW respectively the energy E_{cs} of 0.54 mWs. The number of sleep n_s depends on the tasks' cycle time.

Communication also needs energy for the data transport. Within safety critical systems often time-triggered communication protocols such as FlexRay are used, which need synchronized nodes and a common known communication schedule. This means bandwidth is reserved and a reconfiguration is necessary to reallocate bandwidth. In the specific case of time-triggered communication not only sender and receiver need to be reconfigured, but all nodes within the network need to be reconfigured. We consider here a FlexRay communication controller [10] and the corresponding transceiver [11], for which we can estimate energy consumption as follows. The FlexRay controller has a maximum power consumption of 165 mW and the FlexRay transceiver 175 mW (normal mode). Using the maximum bandwidth of 10 MBit/s a bit consumes 34 nWs/Bit per node, which is a very rough estimation, because of static energy consumers such as for synchronization. Due to this and for simplicity we estimate a fixed (not per node) energy consumption of 30 nWs/Bit in total within our case study.

In section 3.3 the energy for changing configuration is considered. We assume an activation/deactivation of one task consumes 1 ms and with that a task activation/deactivation energy E_{tc} of 0.108 mWs, which includes the loading of software into memory and ECU-internal configurations. If the whole ECU changes to sleep mode, just the energy for changing to sleep mode and back is considered. The reconfiguration of the network consumes energy at every node's communication controller as discussed above. [12] measured the switch of a communication schedule including acknowledgement of the nodes which is 80 ms. The energy to change the network configuration E_{nc} within this case study (2 sensor-ECUs and 2 ECUs) is then 108.8 mWs per change. The number of task changes n_{tc}, network changes n_{nc} and ECU mode changes n_{mc} can easily be obtained from the configurations. A further aspect is the different active times of the applications C1 and C2, here C1 is 10% and C2 90% of time activated. This could be the case for a vehicle, which is commonly used within cities, e.g. taxis. The number of configuration changes is assumed to be 60 per hour, e.g. every minute.

The calibration time of the radar sensor (as discussed at section 2) is assumed to be 5 ms. This calibration is necessary after every sleep mode and means a restart 5 ms before the first (valid) calculation can be done.

To simplify the calculation, the costs for communication within an ECU are neglected. An ECU just has one sleep mode and a fixed cost for mode switching. The storing and buffering of data is not considered, that means memory costs are neglected. The cost for communication is assumed as fixed energy consumption per transferred bit; no overhead costs (e.g. error checks, sync, keep-alive messages, routing, etc) are analyzed in detail. Sensors are allowed for deactivation, but after activation a specific time period (calibration time) is necessary before a sensor is usable. It is considered that sensor-ECUs and ECUs change to sleep mode after task execution.

3.2 Finding Optimal Configurations

In this section we present the model to estimate the energy consumption of an application and calculate the energy consumption of different configurations, but without considering the change of configurations. The energy consumption for a specific configuration is modeled with the summation of the energy consumptions of task executions, sleep periods with the associated mode changes and communications. The energy consumption is calculated over a common multiple of all task periods t_{mult}, which also determines the number of sleeps n_s. The equations to calculate the total energy consumption of a configuration are shown at equation (1-4).

$$E_{config} = \sum_{j=0}^{\substack{No.\ of\\ ECUs}} (E_{tasks,j} + E_{sleeps,j}) + E_{com} \quad (1)$$

with

$$E_{tasks} = t_{tasks} \cdot P_{hw} = \left(\sum_{i=0}^{\substack{Tasks\ at\\ hardware}} \frac{t_{e,i} \cdot f_{te,i}}{t_{c,i}} \right) \frac{t_{mult}}{f_{hw}} \cdot P_{hw} \quad (2)$$

and

$$E_{sleeps} = ((t_{mult} - t_{tasks} - n_s \cdot t_{cs}) \cdot P_s) + n_s \cdot E_{cs} \quad (3)$$

and

$$E_{com} = \sum_{l=0}^{\substack{No.of\ network\\ communication}} (c_t \cdot E_c) \quad (4)$$

To optimize the energy consumption of configuration C1a (see Figure 1), the task allocation is changed. One possible configuration C1b is shown in Figure 3, which reduces the energy consumption by 2.80%. This is a result of the lower energy consumption of ECU 1 compared to the sensor-ECUs. Figure 4 shows configuration C1c, which reduces the energy consumption further by 2.98%. This energy saving is a result of taking into account the calibration times of the sensors. Comparing configuration C1a and C1c an energy saving of 5.78% is reached. This shows the potential of network-wide optimization.

The application "collision warning" has the configuration C2 (Figure 2). The calculated energy consumptions of all the configurations are shown in detail in Table 2.

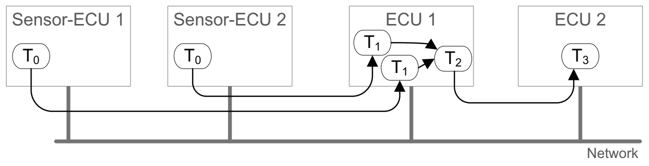

Figure 3: Task chain ACC - Configuration C1b

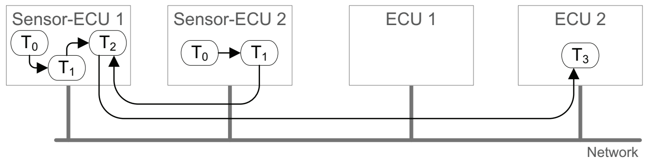

Figure 4: Task chain ACC - Configuration C1c

Table 2: Energy consumption [mWs] of the configurations

Configuration	C1a	C1b	C1c	C2
Task Execution	75.83	63.66	76.96	32.47
Sleep Periods + Mode Change	0.017 + 16.20	0.019 + 16.20	0.018 + 10.80	0.022 + 5.40
Data Transport	2.40	12.00	1.44	0.96
Total	**94.45**	**91.88**	**89.21**	**38.85**

3.3 Optimal Configurations for Alternating Configurations

In this section we show that the combination of C1a and C2 results in a more energy efficient system instead of the combination of the most energy efficient configurations C1c and C2.

The vehicle may change task chains to execute different functions or to be more energy efficient. Nevertheless changing ECU and network configuration also consumes energy as discussed in Section 3.1. Another aspect is the length of stay within a configuration and the number of changes n_{cc} during a time period. Equations (5) and (6) show the calculation of the total energy consumption including energy for configuration changes E_{cc}. Factors α and β represent the percentage within the configurations.

$$E_{total} = \left(E_{config,a} \cdot \alpha + E_{config,b} \cdot \beta\right) + n_{cc} \cdot E_{cc} \quad (5)$$

with

$$E_{cc} = E_{tc} \cdot n_{tc} + E_{nc} \cdot n_{nc} + E_{cs} \cdot n_{mc} \quad (6)$$

Combining the C1c and C2, which are individually the energy efficient options, results in a set of alternating configurations as shown in Figure 5. (Note that the two alternating task chains are shown in one figure, even though not executed simultaneously.)

However, the set of configurations C1a and C2 as shown at Figure 6 is more energy efficient (2.89%), because of the bigger similarity of the configurations. In detail, the set "C1c+C2" has to (de)activate one task, change mode of sensor-ECU 1 and reconfigure the network. (Note that communication slot of task T2 is not usable of task T1, because of different size/bandwidth.) Configuration set "C1a+C2" has to change sensor-ECU 2 and ECU 1 to sleep mode and back, but no task (de)activation and no network reconfiguration. Table 3 shows the energy consumptions of these two sets. In this particular case, changing the network dominates the additional energy consumption. The small number of tasks has no real influence to the energy consumption which is not the case for larger systems.

This shows that a combination of most energy efficient configurations may lead to sub-optimal energy efficiency. The reasons are the energy necessary to change configurations and the usage profile.

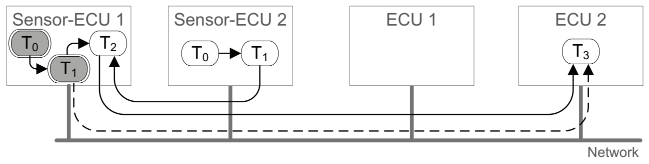

Figure 5: Alternating configurations (C1c+C2) (not active at the same time)

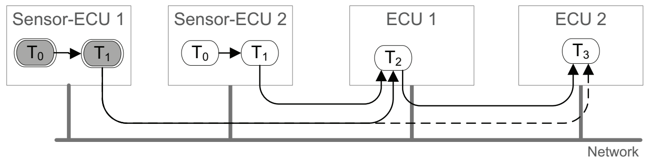

Figure 6: Optimal set of configuration (C1a+C2), which alternate (not active at the same time)

Table 3: Total energy consumption [mWs] of the sets of configurations

Configuration	Without energy for configuration changes	With energy for configuration changes
C1c + C2	43.89	45.71
C1a + C2	44.41	44.43

4. CONCLUSION AND FURTHER STEPS

This paper has discussed and analyzed the energy consumption of decentralized adaptive systems. Based on examples the potentials of network-wide optimization and the effects of configuration to the total energy consumption were shown. The goal was to present the potentials and challenges of network-wide energy optimization of adaptive systems. We have shown that different configurations of a single task chain may have considerable differences in energy consumption of 5.78%. In case of alternating task chains, locally optimal configurations do not result in globally optimal configuration. In our model the energy consumption difference due to this is 2.89%. The reasons are energy consumption for configuration changes and the usage profile of the system. Similarly, adding additional hardware may improve energy efficiency. This results in further research challenges and also new energy saving possibilities.

Our model is based on existing data on energy consumption. Validating our model by measurements would require highly detailed measurements (in terms of time and functional isolation) and is not covered in this paper.

Our approach and first results allow one to design the configuration depending on the use of the car to be more energy efficient. E.g. the configuration for taxis, which are most time in cities and below 60 km/h, may differ from normal cars. Another possibility is the calculation of different valid configurations, which are only energy-efficient for a specific kind of usage. The decision, which set of configuration is used, is done during vehicle runtime, e.g. based on the route of the car, which is known due to the navigation system.

5. REFERENCES

[1] Dong-In Kang, S. Crago, and Jinwoo Suh, "A fast resource synthesis technique for energy-efficient real-time systems," in Proceedings of the 23rd IEEE Real-Time Systems Symposium RTSS 2002, IEEE, Ed, 2002.

[2] Jingcao Hu and R. Marculescu, "Energy-aware communication and task scheduling for network-on-chip architectures under real-time constraints," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'04), IEEE, Ed, 2004, pp. 234–239.

[3] C. A. Rusu, R. Melhem, and D. Mosse, "Maximizing the system value while satisfying time and energy constraints," IBM J. Res. & Dev, vol. 47, no. 5, pp. 689–702, 2003.

[4] M. Baker, Topics in power and performance optimization of embedded systems, Dissertation, Arizona State University, 2011, Available: http://hdl.handle.net/2286/jq8f23vj0yh

[5] L. Benini, A. Bogliolo, and G. de Micheli, "A survey of design techniques for system-level dynamic power management," in Transactions on Very Large Scale Integration (VLSI) Systems, IEEE, Ed, 2000.

[6] R. Jejurikar and R. Gupta, "Dynamic voltage scaling for systemwide energy minimization in real-time embedded systems," in Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'04), 2004, p. 78.

[7] G. Zeng, H. Tomiyama, H. Takada, and T. Ishihara, "A Generalized Framework for System-Wide Energy Savings in Hard Real-Time Embedded Systems," in Proceedings of the 5th International Conference on Embedded and Ubiquitous Computing EUC 2008, 2008, pp. 206–213.

[8] Energy Micro, EFM32G Reference Manual. Available: http://cdn.energymicro.com/dl/devices/pdf/d0001_efm32g_reference_manual.pdf.

[9] Freescale Semiconductor, "MPC5554-Microcontroller Data Sheet," 2008.

[10] Freescale Semiconductor, "MFR4310 Reference Manual: FlexRay Communication Controllers," 2008.

[11] NXP Semiconductors, "TJA1080A FlexRay transceiver - Product data sheet," 2011.

[12] P. Heinrich, D. Eilers, R. Knorr, M. Königer, and B. Niehoff, "Autonomous Parameter and Schedule Configuration for TDMA-Based Communication Protocols Such as FlexRay," in Proceedings of Int. Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11, IEEE, Ed, 2012