

# Minimizing energy consumption for handheld computers in Moby Dick

Paul J.M. Havinga, Gerard J.M. Smit  
University of Twente, department of Computer Science  
P.O. Box 217, 7500 AE Enschede, the Netherlands  
e-mail {havinga, smit}@cs.utwente.nl

## Abstract

*In this report we propose a number of techniques to reduce energy consumption for mobile computers. We use extra dedicated low-power modules to cut on processor cycles of the main CPU, i.e. hardware and software decomposition. These modules are autonomous and can be powered down individually. Furthermore, as networking consumes a large part of a mobile system's battery resources much effort is put in reducing energy consumption in this part. We use intelligent network interfaces with a power aware network protocol, and a MAC protocol that minimizes the 'on-time' of network interfaces to reduce energy. There is a direct link between QoS and energy consumption. Therefore we move power management into the QoS domain.*

## 1 Introduction

Portable and hand-held computers must be careful not to waste the scarce energy resources in their batteries. Even though battery technology is improving continuously and processors and displays are rapidly improving in terms of power consumption, battery life and battery weight are issues that will have a marked influence on how hand-held computers can be used. More extensive and continuous use of network services will only aggravate this problem since communication consumes relatively much energy. Research is needed to provide intelligent policies for careful management of the power consumption while still providing the appearance of continuous connections to system services and applications.

The Moby Dick project [8] is a joint european project (Esprit Long Term Research 20422) to develop and define the architecture of a new generation of mobile hand-held computers, that we will call the *Pocket Companion*. The design challenges lie primarily in the creation of a single architecture that allows the integration of security functions, externally offered services, personality, and communication. Research issues include: security [3], energy consumption and communication, hybrid networks, data consistency, and environment awareness.

To support multimedia functionality for the intended applications of the Pocket Companion the system needs to have real-time properties. The increasing levels of performance and integration that is required will be accompanied by increasing levels of energy consumption. Without a significant energy reduction techniques and energy saving architectures, battery life constraints will limit the capabilities of a Pocket Companion. The main theme of our approach is: enough performance for minimal energy consumption. This is in contrast to the main stream of current research in computer systems which aims at the *highest performance*, and where energy consumption is of minor concern. Even though battery technology, processors and displays are improving continuously in terms of power consumption, the every day use of the Pocket Companion will require a lot of energy; and thus battery life and battery weight remain important issues. More extensive and continuous use of network services will only aggravate this problem since communication consumes relatively much energy.

There is a direct link between QoS and energy consumption. QoS parameters, such as latency, processor speed, bandwidth and audio quality have impact on the energy consumption. We believe that a QoS framework is a sound basis for integrated management of all resources of the Pocket Companion,

including the batteries.

### **Related research**

There are various techniques for *reducing energy consumption* [2]. While low-power components and subsystems are essential building blocks for portable systems, little effort has been directed towards dedicated low-power hardware architectures by considering the system as a whole. A system wide approach is beneficial because there are dependencies between subsystems, e.g. optimization of one subsystem may have consequences for the energy consumption of other modules.

Different architectures have been proposed to address multimedia computing. These approaches are based on high-performance technology and are mostly simple extensions to current architectures. These systems fail to exploit the opportunities for energy reduction offered by multimedia. Systems like the InfoPad [11] and ParcTab [5] are designed to take advantage of high-speed wireless networking to reduce the amount of computation required on the portable. These systems are a kind of portable terminal and take advantage of the processing power of remote compute servers. This approach simplifies the design and reduces power consumption for the processing components, but significantly increases the network usage and thus also increases energy consumption. These systems also rely on the availability of the network and cannot be used when not connected. UCLA has constructed a network testbed [7] that uses a hardware architecture to localize data for both communication and video. In this way the data streams are reduced and efficiently transferred directly to their destination. Abnous and Rabaey propose an architecture for signal processing applications that is flexible and uses low power [1]. The architecture consists of a control processor surrounded by a heterogeneous array of autonomous, special purpose satellite processors. The computational demand on the control processor is minimal, its main task is to configure the system and manage the overall control flow of a given signal processing algorithm. The satellite processors perform the dominant, energy-intensive computational tasks of algorithms. The granularity of these tasks is relatively small. Some examples include address generators, multiply-accumulate processors for computing vector dot products, etc.

## **2 The Moby Dick approach to reduce energy consumption**

Research has shown that there is no single approach for reducing energy in systems like the Pocket Companion [12]. Energy reduction techniques must be applied in all levels of the system. First of all, we have to use components that use the latest developments in low power technology. Furthermore, as the most effective design decisions stem from the architectural and system level, a careful design at these levels can reduce the power consumption considerable.

However, energy reduction is not just a problem of the power-conscious hardware designer, but also involves careful design of the operating system and application programs. Furthermore, because the applications have direct knowledge of how the user is using the system, this knowledge must be penetrated into the power management of the system. For multimedia applications in particular, there is a substantial reduction in energy consumption possible as the computational complexity is high, have a regular and spatially local computation, and the communication between modules is significant. Improving the energy efficiency by exploiting *locality of reference* and using *efficient application-specific modules* therefore has a substantial impact on a system like the Pocket Companion.

### **2.1 Overview**

We use several supplementary approaches to reduce energy consumption in the Moby Dick architecture. In this paper we focus on the system architecture, networking, and the integration of Quality of Service and power management. Our approach is based on extensive use of power reduction techniques at all levels of system design and has a number of premises:

- An architecture with a general purpose processor surrounded by a set of heterogeneous programmable modules, each providing an energy efficient implementation of dedicated tasks.

- A reconfigurable internal communication network that exploits locality of reference and eliminates wasteful data copies.
- A system design that avoiding wasteful activity: e.g. by using *autonomous* modules that can be powered down individually and are data driven.
- A wireless communication system designed for low energy consumption by using *intelligent* network interfaces that can deal efficiently with a mobile environment, by using a power aware network protocol, and by using a MAC protocol that minimizes the ‘on-time’ of network interfaces.
- A *Quality of Service framework* for integrated management of the resources of the Pocket Companion in which each module has its own - dedicated - *local* power management. The operating system will control the power states of devices in the system and share this information with applications and users.

## 2.2 System architecture

The power consumption of a system is strongly related to a number of properties that a given system or algorithm may have. The two main themes for energy reduction at system level are to avoid wasteful activity, and to exploit locality of reference [2].

The component that contributes significantly to the total energy consumption is the *interconnect*. Experiments have demonstrated that in designs, about 10 to 40% of the total power may be dissipated in buses, multiplexers and drivers [15]. This amount can increase dramatically for systems with multiple chips due to large off-chip bus capacitance. The power consumption of the interconnect is highly dependent on algorithm and architecture-level design decisions. *Locality of reference* is therefore an important principle for reducing interconnect power consumption. Locality of reference relates to the degree to which a system or algorithm has natural isolated clusters of operation with a few interconnections between them. Partitioning the system or algorithm into spatially local clusters ensures that the majority of the data transfers take place within the clusters and relatively few between clusters. The result is shorter local buses that are more frequently used than the longer highly capacitive global buses. Localization reduces the communication overhead in processors and allows the use of reduced sized transistors, which results in a reduction of capacitance.

At the system level the locality principle can be applied to divide the functionality of the system into dedicated modules. When the system is decomposed into application-specific coprocessors the data traffic is reduced, for instance because unnecessary data copies are removed. For example, in a system where a stream of video data is to be displayed on a screen, the data can be copied directly to the screen memory, without going through the main processor.

Furthermore, general purpose processors often have to perform tasks for which they are not ideally suited. Although they can perform such tasks, dedicated implementations may take considerably shorter executing time, and might be less energy demanding. Application-specific integrated circuits (ASICs) or dedicated processors in combination with a general purpose processor offer an attractive alternative approach. A system designer uses the processor for portions of algorithm for which it is well suited (e.g. initialization), and applies an application-specific coprocessor (e.g. custom hardware) for other tasks.

Figure 1 shows an example of such a system. Each subsystem needs to be designed to balance the goals of the whole system and not just to optimize a specific subsystem. Each module must be optimized - apart for its main task - for energy consumption. It can have a separate clock, voltage control, power-down modes, and dedicated features to save energy. This is a good example of the difference between power and energy: although the application-specific coprocessor may actually consume more power than the processor, it may be able to accomplish the same task in far less time, resulting in a net energy savings. The application specific coprocessor can offload the main processor with task like JPEG and MP3 decoding, encryption, and some network protocol handling. An MPEG chip can handle video much more energy efficient than a general purpose processor. A processor needs a lot of

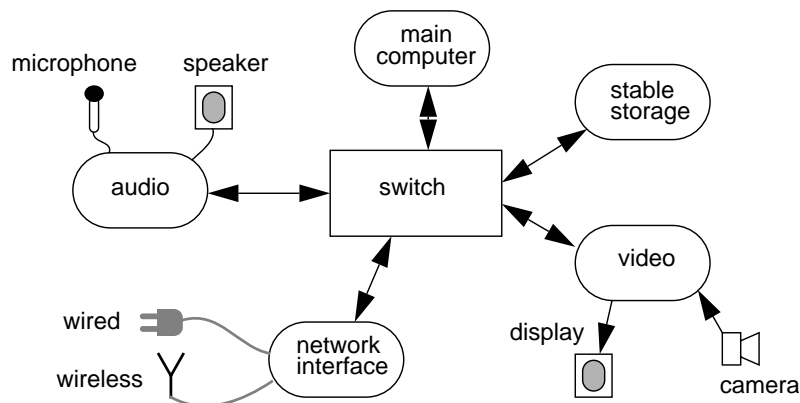


Figure 1: Pocket Companion system architecture

energy hungry bus cycles to perform the same operation, and hence requires a lot of power.

### 2.3 Communication networks

The wireless network interface consumes a significant fraction of the total power [14]. Therefore the network and medium access protocols of a wireless system should be designed for low energy consumption.

#### MAC protocol

We use a TDMA protocol to coordinate delivery of data to receivers. The basic objective is that the protocol tries to minimize all actions of the network interface, i.e. minimize 'on-time' of the transmitter as well as the receiver. A base station is responsible for traffic scheduling. Mobiles with scheduled traffic are indicated in a list, which allows mobiles without traffic to reduce power rapidly. As switching between states (i.e. off, idle, receiving, transmitting) consumes time and energy, the number of state-transitions have to be minimized. By scheduling bulk data transfers, an inactive terminal is allowed to doze and power off the receiver.

A base station dictates a frame structure within its range. A frame consists of a number of data-cells and a traffic control cell. The frame structure influences the latency. This is an example of a trade-off between energy consumption and QoS. The traffic control is transmitted by a base station and contains the information about the subsequent data-cells, including when the next traffic control cell will be transmitted. The overhead and energy consumption involved in the frame mechanism can be reduced when the frame size is adapted to the situation e.g. when only one mobile communicates, the frame size is increased.

A multicast or broadcast mechanism also reduces energy consumption (and increase performance as well) since the information is sent only once, and the base station - with plenty of energy - disseminates the information.

#### Hybrid networks

Over short distances, typically of up to five metres, high-speed, low-energy communication is possible. Private houses, office buildings and public buildings can be fitted with 'micro-cellular' networks with a small antenna in every room at regular intervals, so that a Pocket Companion never has to communicate over a great distance - thus saving energy - and so that the bandwidth available in the aether does not have to be shared with large numbers of other devices - thus providing high bandwidth density (Mbps/m<sup>2</sup>) [9]. Over larger distances (kilometres rather than metres), the Pocket Companion can use the standard infrastructures for digital telephony (such as GSM).

#### Error control

Wireless networks have a much higher error rate than the normal wired networks. In the presence of

a high packet error rate, some network protocols (such as TCP) overreact to packet losses, mistaking them for congestion. This leads to backing off to a lower transfer rate which increases the energy consumption because it leads to a longer transfer time. Forward Error Correction (FEC) can be used to improve the performance and to reduce energy consumption, not only at the data link level, but also in higher levels of the protocol stack [10]. In particular, FEC can be beneficial for reliable multicast communication, because it reduces the number of acknowledgements.

### **Decomposition**

In normal systems much of the LAN protocol stack is implemented on the main processor. Thus, the network interface and the main processor must always be 'on' for the LAN to be active. In these systems performance and energy consumption is a significant problem. Decomposition of the network protocol stack and a careful analysis the data flow in the system can reduce the energy consumption.

1. In a system constructed out of independent components like depicted in figure 1, the network module can do basic protocol processing and can move the data directly to its destination. Hence unnecessary data copies are eliminated.
2. A dedicated processor of the network module can handle most of the lower levels of the protocol stack, thereby allowing the main processor to sleep for extended periods of time without affecting system performance or functionality. The network module is a small and efficient system, it for example does not need wide buses and a large memory.
3. Part of the network protocol can be handled on another machine, e.g. the base station. For example, the mobile units can use a private lightweight energy aware protocol rather than a protocol like TCP/IP. This private protocol can be dedicated to the wireless network. In addition to that, when data is buffered on a base station, retransmissions of data that are caused by errors in the wireless network can remain local thus not involving the higher network protocol layers.

## **2.4 Helper programs**

In Moby Dick we introduced the concept of helper programs [13]. A helper program is in some sense related to and resembles so called mobile agents. In general a mobile agent is an autonomous program that can migrate under its own control from machine to machine in a heterogeneous network. By migrating to the location of a resource, the agent can access the resources more (energy) efficiently.

There are a good number of reasons for using helpers on a Pocket Companion. It is convenient, provides environment awareness and allows asynchronous interaction. Moreover, helpers can be used to save processing power or reduce communication, and hence save battery power. A helper has the ability to perform information retrieval and filtering, while returning to the client only the relevant information, thus saving communication bandwidth. For example, if a Pocket Companion runs a helper with a large user interface, no communication resources are used for the user-interaction because the user-interface is executed locally on the Pocket Companion. On the other hand, if a Pocket Companion has a relatively high computational intensive task, it can send a helper to a remote server to perform this task on his behalf, thus saving processing power for the Pocket Companion.

## **2.5 QoS and power management**

In the Moby Dick architecture, Quality of Service (QoS) is a framework to model integration and integrated management of all the system services and applications in the Pocket Companion. The consumption of resources by one application might affect other applications, and as resources run out, all applications are affected. Since communication bandwidth, energy consumption and application behaviour are closely linked, we believe that a QoS framework is a sound basis for integrated management of the resources of the Pocket Companion.

In order to integrate power awareness in the QoS framework, changes must be made to hardware, drivers, firmware, operating system, and applications. The operating system will control the power

states of devices in the system and share this information with applications and users.

### 3 Current status and future directions

Currently we are designing the system architecture of the Pocket Companion. Application specific coprocessors will be built with standard components, like microcontrollers and FPGAs. Because the system is decomposed of dedicated application specific subsystems that are connected with each other via the switch, energy consumption is reduced and - although not a primary goal - performance is increased. A testbed is being build to verify our statements and evaluate the results.

In cooperation with Nedap N.V. we have built a wireless link based on near-field radio [9]. We have designed, analysed and simulated a real-time, energy efficient MAC protocol for this link [6]. We are able to implement, test and evaluate several mechanisms for low power MAC protocols. Several measurements will be made e.g. concerning energy consumption, error rates and mode switching delays. The flexibility of our network interface allows experimenting with different system decompositions and network protocols.

As hardware architecture and system software are related, we also experiment with system software. The operating system *Inferno* from Lucent Technologies [4] is quite well suited for this purpose. In *Inferno* applications and system may be split easily - even dynamically - between client and server. Certain functions of the system can be omitted, or migrated from the portable system to a remote server. The remote server handles those functions that can not be handled efficiently on the portable machine. Using *Inferno* and the hardware architecture of the Pocket Companion gives us a system that is modular and can be adapted to performance and power requirements. We are working on real time scheduling and a QoS manager in *Inferno*. A prototype of a security system for helper programs (*Harpoon*) in *Inferno* has been built and tested.

### 4 References

- [1] Abnous A, Rabaey J.: "Ultra-Low-Power Domain-Specific Multimedia Processors," Proceedings of the IEEE VLSI Signal Processing Workshop, San Francisco, October 1996.
- [2] Havinga P.J.M., Smit G.J.M.: "Low power system design techniques for mobile computers", internal report University of Twente, 1997
- [3] Helme A., Mullender S.J.: "The Architecture of Trust", internal report University of Twente, February 1997.
- [4] "Inferno reference manual", Lucent Technologies 1997, document id: TM01FR10, see also <http://cruel.com/vanni/>
- [5] C. Kantarjiev et al.: "Experiences with X in a wireless environment", Mobile and location-independent computing symposium, Cambridge MA, August 1993.
- [6] Linnenbank, G.R.J. et al.: "A request-TDMA multiple-access scheme for wireless multimedia networks", Proceedings Third Workshop on Mobile Multimedia Communications (MoMuC-3), 1996.
- [7] W. Mangione-Smith, et al.: "A low power architecture for wireless multimedia systems: lessons learned from building a power hog", Proceedings of the international symposium on low power electronics and design, August 1996
- [8] Mullender S.J., Corsini P., Hartvigsen G. "Moby Dick - The Mobile Digital Companion", LTR 20422, Annex I - Project Programme, December 1995 (see also <http://www.cs.utwente.nl/~havinga/pp.html>).
- [9] van Opzeeland, M., Poortinga, R.: "Design and realization of a wireless ATM-network", Msc. thesis University of Twente, Enschede, May 1997.
- [10] Rizzo L.: "On the feasibility of software FEC", internal report University of Pisa, 1997, available at: <http://www.iet.unipi.it/~luigi/fec.html>.
- [11] Sheng S., Chandrakasan A., Brodersen R.W.: "A Portable Multimedia Terminal", IEEE Communications Magazine, pp. 64-75, vol. 30, no. 12, Dec., 1992.
- [12] Smit G.J.M., Havinga P.J.M.: "A survey of energy saving techniques for mobile computers", internal report University of Twente, 1997.
- [13] Smit, G.J.M. Havinga, P.J.M. van Os, D.: "The Harpoon security system for helper programs on a Pocket Companion", to appear in Proceedings Euromicro 97, September 1997.
- [14] M. Stemm, et al. "Reducing power consumption of network interfaces in hand-held devices", Proceedings Third Workshop on Mobile Multimedia Communications (MoMuC-3), 1996.
- [15] Mehra R., Rabaey J.: "Exploiting regularity for low-power design", proceedings of the international Conference on computer-aided design, 1996.