

6

Concluding remarks

In this chapter we will first evaluate the effectiveness of our approach in designing an energy-efficient architecture for hand-held multimedia computers. In particular we will compare the power dissipation of the test-bed of Mobile Digital Companion with a traditional architecture using a typical multimedia application. Then we give some suggestions for future research. We conclude with some general conclusions about the main issues discussed in this thesis.

6.1 Evaluation of power dissipation

The connection-centric approach with application domain specific modules gives a number of advantages like (energy) efficient processing, high performance, elimination of useless data copies, relieve of the general-purpose processor, and the possibility of an adequate energy management. We already gave some practical and theoretical background on the amount of energy that can be saved using such an architecture.

In this section we will show the effectiveness on the energy consumption of our architecture using a typical multimedia application. We will do this by comparing the power consumption of wirelessly receiving and playing MP3 music on a traditional architecture and on a Mobile Digital Companion. We will further compare the power dissipation of these architectures when idling, and waiting for an external event.

Two considerations have to be made when interpreting the resulting power dissipation:

- The architecture of a Mobile Digital Companion as described in this thesis encompasses various levels in the design cycle. The design approach is vertical oriented, which implies that all layers of the system are involved, and an optimal effect on the performance and efficiency is reached only when all layers co-operate. This thesis only covers the general system architecture, the interconnection architecture, and the wireless interface.

- In the comparison we will use the actual power consumption measured on the test-bed, and for those parts that have not been implemented, we will use the power consumption numbers from datasheets. The power consumption of the traditional architecture will be based solely on datasheets, since measuring the power consumption of a notebook computer would be too flattering and not fair.

Further note that the test-bed is designed to evaluate the *energy efficiency* of designs, and is *not* designed to be low power! The actual implementation of the test-bed is therefore primarily designed to be *flexible*, and suitable for doing experiments with various design alternatives. Because of this, the implementation test-bed used various flexible, but certainly not low-power components (i.e. we have used Xilinx FPGAs).

With these above mentioned considerations in mind, we will now compare the power consumption of wireless receiving and playing MP3 music on a traditional architecture and on a Mobile Digital Companion.

The power consumption of the various hardware modules involved are gathered from datasheets and from measurements on our testbed prototype. We have only included the main components, and have neglected the parasitic power consumption due to glue logic and the energy consumed for the actual data transfer (except for the bus). The audio module is invariant in our comparison, since both setups are assumed to use the same hardware in the same way. The wireless network interface is in both setups also the same, although a different MAC protocol is being used.

We assume that there are three basic operating modes: *active* in which the device is fully operational, *sleep/idle* in which the device is idling and ready to become active, and *off* in which a device is completely powered down. It is further assumed that the mobile system uses a dynamic power management that has powered down (off state) all other components of the system that are not in use.

We will indicate with τ_{active} the percentage of time being active and with τ_{idle} the percentage of time the part is idle. The power consumption P of each part can then be calculated using:

$$P = \tau_{active} \cdot P_{active} + \tau_{idle} \cdot P_{idle} \quad (1)$$

Because all the components that are used in the application we consider remain powered on (i.e. sleep/idle or active mode), we can state that

$$\tau_{active} = 1 - \tau_{idle} \quad (2)$$

6.1.1 Setup traditional architecture

In a traditional (CPU-centric) architecture the general-purpose processor controls the media streams of an application. The general-purpose processor in such an architecture is responsible for the communication protocol to receive the MPEG frames from the

wireless interface, it needs to decode the frames, and also must transfer the data to an audio module.

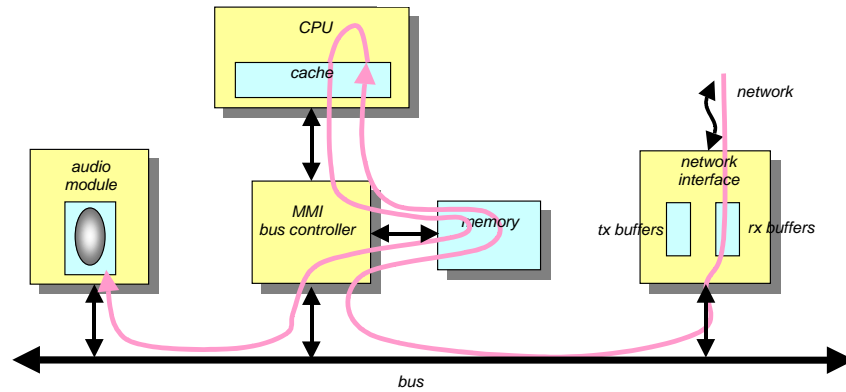


Figure 1: Data-flow through a traditional architecture.

The *traditional architecture* of a mobile, shown in Figure 1, is centered around a general-purpose processor with local memory and a bus that connects peripherals to the CPU. The long arrow in the figure indicates the data stream through the system when data arrives from the network, is transferred through the receive buffers on the network interface, copied to the ‘main’ memory, and then processed by the application (i.e. MP3 decoding). After the data is processed by the application, the data will traverse via ‘main’ memory, over the bus, to the output device (the audio module). In general additional bus transfers between CPU and memory are introduced while traversing several protocol layers (e.g. for data conversion of the packets like Ethernet to IP, and subsequently IP to TCP).

The power consumption P of the various parts of concern can be found in Table 1.

Table 1: Power dissipation of various parts in a traditional architecture.

Device	Specification	P active [mW]	P sleep/idle [mW]
Processor	Mobile Pentium II/400 [10]	7500	500
Network	WaveLAN modem 2.4 GHz [11]	1800 (RX)/1825 (TX)	180
Memory	SDRAM 2x32 Mb (Micron) [9]	1188	30
Bus	Theoretical bus, similar performance as Octopus switch	1344	1344

The table shows the power dissipation of the Mobile Pentium II processor. This processor is designed for portable applications, and has special features for a low-power consumption. The wireless network interface is based on the WaveLAN I modem. This is the same module as currently being used in the Mobile Digital Companion’s network interface. The module does not include the MAC protocol implementation and bus

interface logic. The additional power consumption for these parts will be ignored. The energy required to transfer data over a bus is based on the theoretical values derived in Chapter 3. This theoretical bus has a similar performance as the Octopus switch. If we would have used for example a PCI-bus, then the required energy would be much higher. For example, the ‘low-power’ PCI9060 PCI bus master from PLX technology requires 680 mW [11].

6.1.2 Setup Mobile Digital Companion

In the architecture of the Mobile Digital Companion as discussed in Chapter 3 the general-purpose processor does not take part in the actual application. A dedicated MPEG audio decoder is being used to decode the MP3 data traffic. The general-purpose processor is only being used to initialise the connections, and setting up the Octopus switch. The processor will thus not be incorporated in the calculations of the power consumption. The network interface used is the testbed-interface board as described in Chapter 5. The medium access protocol is E²MaC. Data coming from the wireless network interface is packet by the base-station into ATM cells, with a Virtual Connection Identifier (VCI) indicating the destination of the data. The Octopus switch will use this VCI to forward received packets of that connection from the network interface to the audio module.

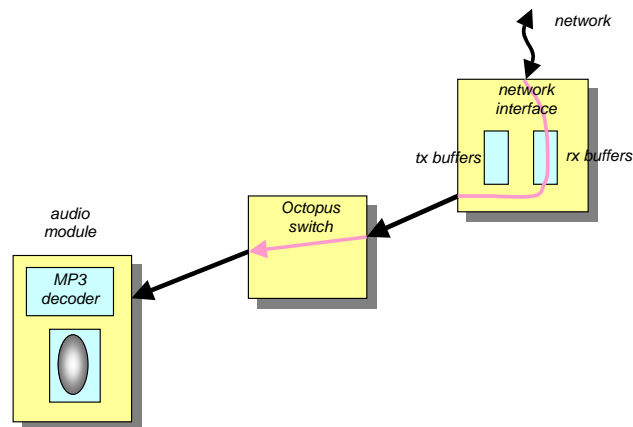


Figure 2: Data-flow through the Mobile Digital Companion.

The following table shows the power consumption P of the various parts of concern in this setup.

Table 2: Power dissipation of various parts in the Mobile Digital Companion.

Device	Specification	P active [mW]	P idle [mW]
MPEG decoder	Cirrus Logic EP7209 (sample frequency > 24 kHz) [3]	110	0.01
Network	WaveLAN modem 2.4 GHz [11]	1800 (RX)/1825 (TX)	180
	Buffer (SRAM HM628512) [6]	300	0.01
	Controller FPGA XC4010 (20 MHz)	120	50
switch	Octopus switch 32 Mb/s	150	60

The MPEG decoder is based on the Cirrus Logic EP7209, which is a single chip MPEG layer 2/3 audio decoder. The conversion from ATM to the required bit-stream can be done easily within the Module Interface Controller of the Octopus switch, and the power consumption involved can be neglected. The wireless network is based on the testbed network interface module that comprises basically of the WaveLAN modem, static RAM, and a controller FPGA. All these components are not low power, and can in a production implementation be replaced by dedicated low-power components.

6.1.3 Power dissipation MP3 application

To calculate the power dissipation involved in our MP3 application, we must incorporate the actual duty cycle of which the various parts are operating.

Traditional design

A MP3 audio stream with a sample frequency of 44.1 kHz, 16 bits, has a bit-rate of 128 kb/s. In the traditional architecture both the processor and the wireless interface have to be continuously in an active operating mode to be able to handle this data-stream¹.

The memory will be accessed multiple times for the processing of the communication protocols (we assume 7 times [11]) and also for decoding of the data-stream (one write and one read on an approximately 10 times larger data-size). However, the duty cycle of the memory remains low (active for approximately 10.8%) when we use a 32-bits wide memory bus, with a total access time of 50 ns. The power dissipation for memory access then becomes 216 mW. The bus interface is assumed to be active all the time.

Mobile Digital Companion

In the Mobile Digital Companion architecture, the network interface can be in idle mode for a significant time. The power cannot be turned off completely because the power-on time of the WaveLAN modem is 200 ms. We have assumed that the base-station transmits the data in bursts of 50 ATM cells (which would fill 50% of a typical frame in

¹ At least the inactivity threshold will not be reached, and thus the device will not enter an idle or sleep mode.

E²MaC with a frame rate of 50 Hz). This requires the modem to receive 7 frames per second. If we would use a larger burst-size, then we would be able to turn off the modem completely from time to time (instead of entering sleep mode) to save more energy. Incorporating the additional overhead (receiving the traffic control slot, and the transitions from changing between receiving and idle modes), the duty cycle of the modem becomes then 8% active, and 92% idle. This implies a power dissipation of 310 mW. The memory of the network interface has a much lower duty cycle (3.2% active, 96.8% idle).

The Octopus switch can handle the required data rate easily and is thus idling most of the time (99.6 % idle, 0.4% active), implying a power dissipation of 60 mW.

The MPEG decoder will be continuously active and thus dissipates 110 mW.

Comparison

Figure 3 shows the power dissipation of the two architectures with the various parts involved in the MP3 application.

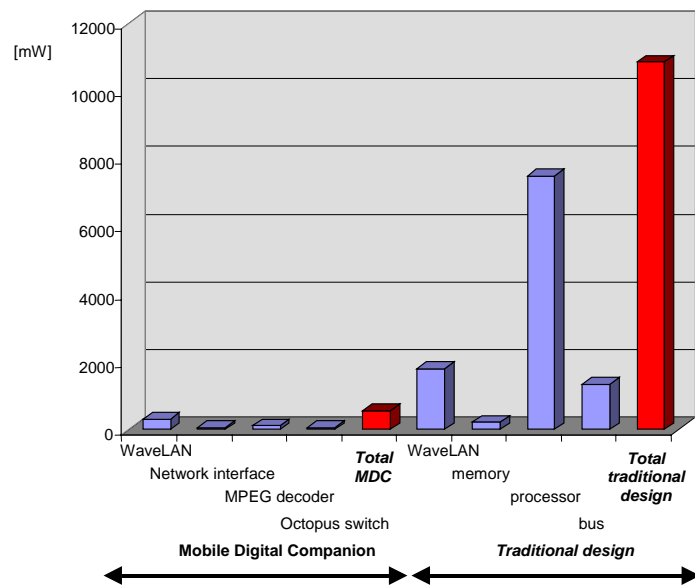


Figure 3: Power dissipation MDC and traditional design with during MP3 decoding.

Clearly shown is that the traditional architecture has a much higher power dissipation than the MDC architecture (i.e. 20.6 times higher).

Power dissipation breakdown

Figure 4 shows the resulting power dissipation of the various parts for the MP3 application in the MDC architecture.

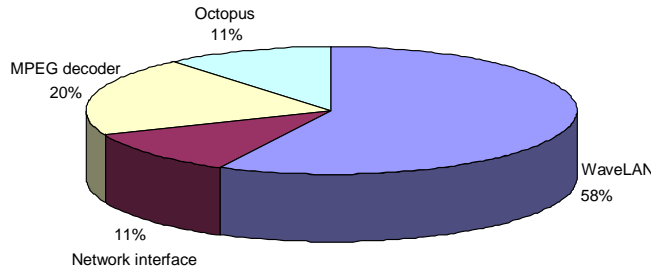


Figure 4: Power dissipation breakdown MDC when decoding MP3.

The energy required for the wireless communication takes a significant part (58% + 11%) of the total energy consumption of the MDC. Note that the WaveLAN modem is not optimised for hand-held devices.

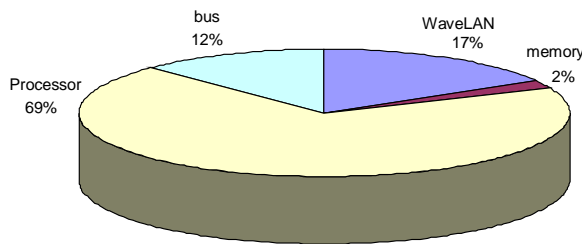


Figure 5: Power dissipation breakdown traditional architecture when decoding MP3.

In the traditional architecture, of which the power dissipation breakdown is shown in Figure 5, the wireless communication even has a much higher power dissipation (i.e. 1800 mW, instead of 430 mW), but still is not the most dominant. Even the power dissipation due to memory access is relatively low. Most of the power dissipation is due to the processor handling the communication protocols (IP, TCP, etc.) and decoding the MP3 data-stream.

6.1.4 Power dissipation when idling

It is expected that the hand-held will be idling for a significant time during the day. Most of the time the system will be waiting for an external event (like an incoming data packet from the wireless interface or the user pressing a button). The power dissipation during these idle periods will thus be an important factor to determine the lifetime of the batteries.

Traditional design

In a traditional design the network module must be powered on all the time because the protocol is usually based on a broadcast mechanism. Measurements on our WaveLAN system indicates that there are many (short) broadcasts (with a frequency of approximately 10 Hz). The power dissipation will thus be 1800 mW because the receiver has to be 'on' all the time. This broadcast mechanism also implies that the processor will be interrupted frequently. We will assume that this processing will take 0.1% of its time, thus $\tau_{active} = 0.1\%$. The resulting power dissipation is then approx. 507 mW.

The bus is not allowed to sleep. The memory will be idling most of the time, which results in a power dissipation of 30 mW.

Mobile Digital Companion

The network protocol is based on E²MaC which implies that the network interface has to be turned on for a short time to receive the Traffic Control Slot (TCS). The frequency of which the TCS has to be received depends on the application. Here we will assume an interactive application (like waiting for an incoming phone call) that needs to receive the TCS twice per second. The active time is then approximately 0.1%, which implies a power dissipation of 182 mW. Since we are also capable of powering down the whole modem, we can even further reduce the power consumption to approx. 73 mW (incorporating the power-up latency of 200 ms). This once again shows that the power-up latency is an important factor in the power dissipation. In future modern designs this will be an important design issue. The network interface contributes with approximately 50 mW (idling).

The Octopus switch consumes 60 mW when idling. All other parts of the system can be turned off.

Comparison

Figure 6 shows the power dissipation of the two architectures when idling.

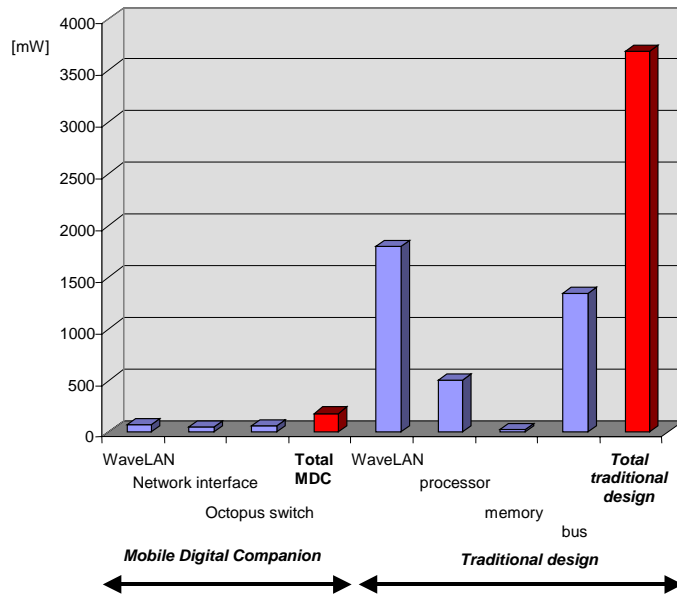


Figure 6: Power dissipation when idling.

Note that the power consumption of the MDC is constituting mainly of idling components. Because in our test-bed we have not used low-power components, the resulting power dissipation is much higher than could be expected when using components that are designed for a low idle power consumption.

6.2 Future research

A thesis is never finished. Although in this thesis we have presented solutions to a number of problems in the field of mobile multimedia computing, many others have remained unsolved or received only minor attention. This section attempts to give a few suggestions for future research.

Having an energy efficient architecture that is capable to handle adaptability and flexibility in a mobile multimedia environment requires more than just a suitable hardware platform. Therefore, 1) more research is required in the operating system architecture that needs to deal with the hardware platform and the adaptability and flexibility of its devices, 2) we need to provide support for reconfigurable computing, and 3) we need to have a model of all resources in the mobile computer so that we can design a proper energy management system.

Research in these items will be continued in the Moby Dick project. The *Chameleon* project [12] will in particular perform research in reconfigurable computing for these systems.

6.2.1 *Operating system architecture*

Our design of the architecture is geared toward achieving low energy consumption, while achieving good application performance like high throughput and low end-to-end latency. In meeting these requirements we have to place constraints on the operating system and communication protocols. The operating system for the *Mobile Digital Companion* has to deal with the flexibility and adaptability of its devices. Since operating system support is out of the scope of this thesis, we will only mention the main relevant topics. The operating system being used for the experiments is derived from Lucent Technologies Bell Labs' *Inferno* [4] and must be adapted in several ways:

- The operating system must be adapted to the hardware architecture: it must be able to manage *connections* between modules rather than DMA transfers, and it must provide a suitable interface to change and manage the functionality of the programmable modules. The operating system has to provide the service such that applications can make effective and efficient use of programmable hardware.
- Applications can exhibit different behaviour depending on its QoS-level and operating conditions (energy resources, current network conditions, etc.). The operating system must define and manage a global energy policy, it must manage the required *adaptation* and, of course, inform applications of changes in their environment. Energy state transitions should never compromise functionality.
- It must provide provisions for *distributed processing* in which part of the application or service will be (possibly even dynamically) reallocated and performed on a remote server when this is more efficient.
- Since it is not feasible to try to adapt the current *communication protocols*, and adapt all current systems and applications, our target will be to support existing and widely used protocols. However, in our connection-centric model in which data streams flow directly from source to sink without interference from a CPU, we believe that it is neither feasible nor efficient that all functional modules implement the whole communication protocol stack of several protocols. Furthermore, most existing protocols are not designed to operate over a wireless link, and do not provide any, or not efficient support for mobile computers. The effect of these protocols is that they waste energy and often have a poor performance and a poor quality. Therefore, we should adopt a distributed processing style in which parts of the communication protocol will be handled by a remote server (most probably the base station to which the mobile is connected to at that moment). The mobile will be relieved from the task of handling 'standard' protocols and will use an internal dedicated and efficient protocol to communicate with the base-station. The resulting architecture can be viewed as a packet routing system that supports data transfer between modules and the wireless link without microprocessor intervention. In MOBY DICK both base stations and mobiles use the operating system *Inferno*. The

operating system allows application and system functions to be split and migrated easily. The base station will handle the TCP/IP protocol in lieu of the mobiles, and use an internal dedicated protocol with the mobile to transfer the packets [1].

These adaptations are necessary and worthwhile because they lead to significantly lower energy consumption while the performance is high. However, a secondary goal is that when our system will be used with current applications, protocols and operating systems with no modification, then our system should not perform worse than the existing systems. For these applications, and to be able to develop and test new applications, the general-purpose processor can play an important role.

Another important issue related to the operating system is that applications have a clean and simple interface. Only when application programmers are willing to use the possibilities that our system provides, and can easily use the programmers interface, then the system can show its real value.

6.2.2 Reconfigurable computing

Reconfigurable computing systems combine programmable hardware with programmable processors to capitalise on the strengths of hardware and software [8]. The earliest configurable computing machine was proposed, designed, and implemented by Professor Gerald Estrin at UCLA in the early 1960s [5]. Estrin proposed the “Fixed plus variable structure computer”, where some fixed hardware was dedicated to an inflexible abstraction of a programmable processor and a flexible component implemented digital logic. Today, the most common devices used for reconfigurable computing are *Field Programmable Gate Arrays* (FPGA). FPGAs present the abstraction of gate arrays, allowing developers to manipulate flip-flops, small amounts of memory, and logic gates.

Currently, many reconfigurable computing systems are based on FPGAs. However, these systems have a number of limitations [7]:

- *Limited functionality* – Not all computations can be implemented efficiently with today’s FPGAs: they are well suited to algorithms composed of bit-level operations, but they are ill suited to numeric operations, such as high-precision multiplication or floating point calculations. General-purpose processors (including digital signal processors) use optimised function units that operate in bit-parallel fashion on long data words. Compared with general-purpose processors, FPGAs are inefficient in performing ordinary arithmetic operations.
- *Gate capacity* – Available FPGAs provide an equivalent of 10K to 1000K gates. These devices are often large enough to experiment with the basic strategies, but limit the scope of the designs. Future FPGAs will be much larger and will have much broader application, including highly complex communications and signal-processing algorithms.
- *Configuration speed* – Most existing FPGAs use relatively slow paths for device configuration, and few have the ability to reconfigure only selective parts of the device. The configuration speed determines the characteristics of the computation

model: it should change frequently enough to take advantage of programmability, but slowly enough to mask hardware configuration time.

- *Memory structures and interface* – FPGAs currently provide little on-chip memory for storage of intermediate results in computation; thus many reconfigurable computing applications require large external memories. The transfer of data to and from the FPGA increases energy consumption and may slow down the computations.

Although less energy efficient than application specific integrated circuits, reconfigurable devices, such as field programmable gate and function arrays can be used for implementing customised circuits. These technologies allow circuits to be created and removed on demand: instead of including all the customised circuits in a system, only those components required for a particular algorithmic stage need to be present. When finished they can be replaced by other customised hardware following a hardware reconfiguration at run time. This capability is especially relevant for wireless networks, where operating conditions are often unpredictable and protocol standards may vary from one network to another.

The *Mobile Digital Companion* has a hierarchical-granularity architecture. The programmability granularity of the modules of the *Companion* is coarse, and the modules themselves can be programmed more fine-grained. Most of the modules of the *Companion* include programmable hardware that can be used to implement the various circuits. Currently there is a large gap between the hardware devices and the application. Research that is needed in this area includes design languages, development methods, as well as compile-time and runtime environments and operating system support for such systems.

6.2.3 *Modelling energy management*

Applications that users run on a mobile need several functional resources of the system, such as processor, memory, wireless network interface, compression/decompression logic etc. In the *Companion's* architecture we assume that such modules are programmable and can adapt to the demands of the applications and to the state of the environment, e.g. available bandwidth, bit error rate, available energy, etc. In general these modules are not independent and choices for the setting of one module may influence other modules. For example: when video has to be transmitted it can be compressed, which reduces the required bandwidth on the wireless network. However, more compression requires not only more processing power, it also needs better error-control. All these functional modules often have contradictory effects on the resources needed, and a trade-off has to be made to find an optimal solution. Not only the parameters can be changed, it might as well be profitable to migrate complete function from one module to another, possibly even to another machine. A complicating factor is that a wireless environment is very dynamic, so it is not feasible to search for *the* optimal solution.

Adaptability and flexibility are two recurring items when we mention energy efficiency and performance on mobile multimedia computers. The architecture of the *Mobile*

Digital Companion has many ways in which adaptation can be applied. This leads to a key problem of *policy optimisation*, which must be the central issue in any energy management system. The policy is the algorithm that decides what measures have to be taken to minimise the energy consumption. Traditional power management schemes only decide how and when to activate or shut down system resources to minimise the energy consumption, depending on usage patterns and performance constraints.

Currently several system developers and vendors are pursuing a long-term, wide scope strategy (ACPI and OnNow) to greatly simplify the task of large and complex power-managed systems. However, both ACPI and OnNow assume a CPU and operating system centric system, where the activities of the system are managed by a single entity. Furthermore, ACPI and OnNow are developed to support the implementation of power managed computer systems, and are too detailed to effectively support design exploration [2].

In the early phases of the design of any part of the system, either hardware or software, the designer needs to experiment with alternative designs. However, energy efficiency is not only a one-time problem that needs to be solved during the design phase. When the system is operational, frequent adaptations to the system are required to obtain an energy efficient system that can fulfil the QoS requirements imposed. Finding the energy management policy that minimises energy consumption without compromising performance beyond acceptable levels is already a complex problem. If the resources are also flexible, and can adapt their functionality, this problem becomes even bigger.

To be able to make valid decomposition that satisfies many requirements and provides an efficient solution, more research is needed to 1) provide support for early system level architectural exploration of energy-managed systems, and 2) to provide a model that can be used to manage adaptable entities at various levels of the architecture.

6.3 Conclusion

In this thesis we considered the problem of designing an architecture for a mobile multimedia computer. The requirement of portability of hand-held multimedia computers and portable devices places severe restrictions on size and energy consumption. In its most abstract form, a mobile computer system has two sources of energy drain during operation: communication and computation. Broadly speaking, minimising energy consumption is a task that will require minimising the contributions of communication and computation, making the appropriate trade-offs between the two.

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. These devices often require real-time processing capabilities, and thus demand high throughput. 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 these machines. More extensive and continuous use of network services will only aggravate this problem since communication consumes relatively much energy.

As the mobiles must remain usable in a wide variety of environments, they must be flexible enough to accommodate a variety of multimedia services and communication capabilities and adapt to various operating conditions in an (energy) efficient way.

We have shown that it is not sufficient to simply continue advancing our chip architectures and technologies as just more of the same: building microprocessors and devices that are simply more complicated versions of the kind built today. We use the technology and the abundant logic gates to build an architecture that is capable of processing multimedia applications that operate in the dynamic mobile environment. Key issues in this are *energy efficiency* and *Quality of Service*.

Main principles – We have found that there are two main principles that can be used when designing mobile multimedia systems.

1. *System-wide layer integration/co-operation.* Co-operation or integration of the various layers significantly improves energy efficiency of the system because it reduces waste and data streams retain a high locality of reference.

The art of low-power design used to be a narrow speciality in analog-circuit design. As the issue of energy efficiency becomes even more pervasive, the battle to use the bare minimum of energy will be fought on multiple fronts: semiconductor technology, circuit design, design automation tools, system architecture, operating system, and application design. We have shown that there is a vital relationship between hardware architecture, operating system architecture and applications architecture, where each benefits from the others. In our architecture we have applied several supplementary energy-reduction techniques on all levels of the system. Achieving high energy efficiency requires first of all *the elimination of the waste* that typically dominates the energy consumption in general-purpose processors. The second main principle used is to have a *high locality of reference*. The philosophy is that all operations that are required on the data should be done at the place where it is the most efficient, thereby also minimising the transport of data through the system.

2. *Use a Quality of Service framework.* We have demonstrated in our research and in particular in the design of a system architecture, a switching network, and the wireless network design, that Quality of Service is not only important to provide an adequate level of service for a user, but can also be used as a tool to achieve an energy-efficient system. Users and applications request a certain QoS level. The system then operates in such a way that it will try to satisfy these requirements, but never gives more quality than required and necessary. Adaptability is the basic mechanism to achieve this.

Of particular importance to the system architecture is the interconnection structure that connects the application domain specific modules. The system architecture of the *Mobile Digital Companion* is connection centric, which means that the media type of the traffic drives the data flow in the system using *connections*. In our infrastructure all

connections are identified with a connection identifier which is used to identify the type of data, and the module destination address. This identifier provides the mechanism to support lightweight protocols that provide data-specific transport services that are associated with a certain QoS. This approach not only eliminates the need to transfer a large number of address bits per access, it also gives the system the possibility to control the QoS of a task down to the communication infrastructure.

The wireless network is another important aspect of a mobile multimedia system. We have shown that energy-awareness must be applied in almost all layers of the network protocol stack. To achieve maximal performance and energy efficiency, *adaptability* is important, as wireless networks are dynamic in nature. Furthermore, if the application layer is provided with feedback on the communication, advantage can be taken from the differences in data streams over the wireless link. To allow this, feedback is needed from many layers: the physical layer provides information on link quality, the medium access layer on effectiveness of its error correction, and the data link layer on buffer usage and error control.

Although our testbed currently consists of various small printed circuit boards containing Field Programmable Gate Arrays (FPGAs), microcontrollers, and memory, the complexity of these designs is low. This low complexity will make it possible to transfer the architecture to a (large) custom IC.

The lessons learned from the design of this architecture serve as a first step towards a system-level design of an energy-efficient mobile multimedia computer.

References

- [1] Balakrishnan H., Padmanabhan V.N., Seshan S., Katz R.H.: A comparison of mechanisms for improving TCP performance over wireless links", *ACM SIGCOMM '96*, Stanford, August 1996.
- [2] Benini L., De Micheli G.: "Dynamic Power Management, design techniques and CAD tools", *Kluwer Academic Publishers*, ISBN 0-7923-8086-X, 1998.
- [3] Cirrus Logic EP7209, Ultra-low-power audio decoder system-on-chip, <http://www.cirrus.com>.
- [4] Dorward S., Pike R., Presotto D., Ritchie D., Trickey H., Winterbottom P.: "Inferno", *Proceedings COMPCON Spring '97*, 42nd IEEE International Computer Conference, 1997, URL: <http://www.lucent.com/inferno>.
- [5] Estrin G. "Organization of Computer Systems: The Fixed-plus Variable Structure Computer", *Proceedings of the Western Joint Computer Conference*, pp. 33-40, 1960.
- [6] Hitachi, "HM628512 Series, 524288-word ´ 8-bit High Speed CMOS Static RAM", 1995.
- [7] Mangione-Smith W.H., et al.: "Seeking solutions in configurable computing", *IEEE Computer*, pp. 38-43, December 1997.
- [8] Mangione-Smith W.H., Hutchings B.L.: "Configurable computing: the road ahead", *1997 reconfigurable architectures workshop*, 1997.
- [9] "16 MEG x 32 SDRAM DIMM", <http://www.micron.com>. Micron Technology Inc., 1999.
- [10] Mobile Pentium II, <http://www.intel.com/mobile/pentiumII>.
- [11] PLX technology: "PCI9060, PCI Bus master interface chip for adapters and embedded systems", datasheet, 1995, <http://www.plxtech.com/download/9060/datasheets/9060-12.pdf>.
- [12] Gerard J.M. Smit, Martinus Bos, Paul J.M. Havinga, Sape J. Mullender, Jaap Smit: "Chameleon - reconfigurability in hand-held multimedia computers", *proceedings First International Symposium on Handheld and Ubiquitous Computing*, HUC'99, September 1999.
- [13] Steenkiste P.: "Design, implementation and evaluation of a single-copy protocol stack", *Software – practice and experience*, January 1998.
- [14] WaveMODEM 2.4 GHz Data Manual, Release 2, AT&T 1995.