Investigation and Development of Energy Saving Techniques for Modern x86 Platforms
– With a Special Emphasis on Embedded Environments –

Author:
Sven Plaga

Supervisor:
Dr. Donal Heffernan

Submitted in Partial Fulfillment of the Requirements for the Degree "Masters of Engineering"

Submitted to
University of Limerick, Limerick, Ireland
April 2010
Abstract

Investigation and Development of Energy Saving Techniques for Modern x86 Platforms - with a Special Emphasis on Embedded Environments -

by Sven Plaga

In recent times there is a growing emphasis on energy saving solutions for processor-based systems of all types, and there is a particular emphasis on energy-saving solutions to sustain battery life in portable and remote equipment.

Some 75 percent of the world’s data processing solutions employ the x86 processor technology, in one form or another, as the underlying computing architecture. However, the ubiquitous x86 technology is characterized by high dissipation loss in relation to its computing power. Despite this significant drawback, the x86 technology is used in many applications, including embedded processor solutions.

This research project is concerned with the development of a novel technical solution which will help to make x86 based embedded systems more energy efficient. The work focuses on the widely-used PICMG COM Express Standard. The proposed solution defines an intelligent interface, referred to as the “Embedded Controller Management Interface”, which is used to optimise the energy management for the embedded system. This “Embedded Controller Management Interface” is implemented by using a dedicated microcontroller to handle the various energy management tasks. The prototype implementation consists of an adapter which includes a FPGA and an ARM-7 microcontroller to perform the various measurements. The collected data can be accessed remotely via Ethernet and is provided to the DUT (device under test) using a standardised Application Programming Interface (API) or a standard bus system, such as the I2C.

The work in this project includes background theoretical research for energy use in embedded systems, measurements of energy use in embedded systems, and the requirements specification and design for the special management interface board. The circuit implementations for some selected core modules are presented and test results for the research work to date are presented. The results of this research work show it is possible to monitor the electrical power consumption of an embedded module in order to perform instantaneous on-line energy profiling.
Declaration

This thesis is presented in fulfillment of the requirements for the Degree of Masters of Engineering.

All the work detailed in this report is completely my own and has not been submitted to any other academic authority in the university. Where use has been made of the work of other people or any information included that was taken from other sources, all such instances have been fully acknowledged and referenced.

Signature: __________________________

(Date) (Sven Plaga)
Acknowledgements

Firstly, I would like to thank my academic supervisor Dr. Donal Heffernan for his great support.

I would also like to thank my German adviser Prof. Dr.-Ing. Andreas Grzemba for his support and guidance throughout the project.

Thanks also to the great and ongoing support of the R&D department of Kontron Embedded Modules, Deggendorf.

Thanks must also go to Martin Aman, Martin Hof, Emanuel Simmel, Laurin Doerr and all other students which helped and assisted me during my research work. Without them it would not have been possible to afford all the work.

Last but not least i want to thank my life partner Nicole Pappler for her great support in reviewing the thesis.
Contents

1 General Overview 1
   1.1 Introduction ........................................................................... 1
   1.2 Rationale for the project ....................................................... 2
   1.3 Project Objectives ................................................................. 2
   1.4 Novelty of the Project ............................................................ 4
   1.5 Literature Survey ................................................................. 4
   1.6 Publication Resulting from this Work .................................... 6
   1.7 Structure of the Thesis ......................................................... 6
   1.8 Summary ............................................................................... 7

2 Background to the Embedded x86 Standards 8
   2.1 Introduction ........................................................................... 8
   2.2 Objective ............................................................................... 8
   2.3 Why the Embedded Developers Choose x86 ......................... 8
   2.4 The Future of Embedded x86 ................................................ 11
   2.5 The Backplane / Module Concept ....................................... 12
   2.6 Module Standards .................................................................. 13
      2.6.1 ETX ............................................................................... 14
      2.6.2 COM Express ................................................................. 15
      2.6.3 ESMexpress ................................................................. 17
   2.7 Summary ............................................................................... 19

3 Drawbacks of the x86 Platform 20
   3.1 Introduction ........................................................................... 20
   3.2 Objective ............................................................................... 20
   3.3 Security and Stability Issues ............................................... 20
   3.4 Energy Related Problems .................................................... 22
      3.4.1 Origins of Energy Inefficiency ....................................... 22
      3.4.2 Components of Energy Consumption .......................... 23
Contents

5.6 Summary ................................................................. 90

6 An x86 Co-Pilot Module ........................................... 91
  6.1 Introduction ......................................................... 91
  6.2 Objective .......................................................... 91
  6.3 Energy Saving ....................................................... 92
    6.3.1 Advanced Power-Down Concept ............................. 92
    6.3.2 Additional Performance Counters ......................... 96
    6.3.3 Solutions for Legacy Hardware ......................... 97
  6.4 Embedded Needs .................................................. 97
    6.4.1 Remote Debug Solution ...................................... 97
    6.4.2 Protocol Bridge ............................................... 99
  6.5 Summary .......................................................... 99

7 Implementation Considerations ................................ 100
  7.1 Introduction ......................................................... 100
  7.2 Objective .......................................................... 100
  7.3 Determination of the Essential Needs ......................... 100
  7.4 Determining a Suitable Hardware and Software Platform .... 102
  7.5 The Interconnection Scheme ..................................... 104
    7.5.1 FPGA Interconnection ....................................... 104
    7.5.2 FPGA to x86 Interconnection ................................. 106
    7.5.3 FPGA to ARM 7 Interconnection .............................. 107
  7.6 Summary .......................................................... 108

8 Conclusions and Continuation Work ............................ 109
  8.1 Review of the Achievements ..................................... 109
  8.2 Continuation Work ............................................... 109
  8.3 Knowledge Gained from this Research Project ............... 110

9 Bibliography ............................................................ 111

A IEEE Applied Electronics Conference 2010 Pilzen .................. ii

B The A20 Gate .......................................................... vii

C Overview Processor C-States ....................................... ix

D Microcontroller based Measurement ................................ xi
<table>
<thead>
<tr>
<th>Contents</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>E</strong>  COM Express Measurement Adapter</td>
<td>xiv</td>
</tr>
<tr>
<td><strong>F</strong>  Listings of LabVIEW Programs</td>
<td>xx</td>
</tr>
<tr>
<td><strong>Glossary</strong></td>
<td>xxiii</td>
</tr>
<tr>
<td><strong>List of Figures</strong></td>
<td>xxix</td>
</tr>
</tbody>
</table>
1 General Overview

1.1 Introduction

In current times there is great emphasis on energy saving solutions for processor-based systems, from an economical point of view. There is also great emphasis on energy saving solutions to sustain battery life in portable and remote equipment.

Some 75 percent of the world’s data processing solutions are employing the x86 processor architecture in one form or another. However, the ubiquitous x86 technology is characterized by high dissipation loss in relation to its computing power. Although the Intel Corporation and many other companies have optimized many parameters towards reduced power consumption, there is still much room for improvement on this platform.

This room has to be filled with new ideas, especially in the fields of embedded devices. For some years now it is quite common to use x86 technology in areas, in which small microcomputers are common. Despite the fact that these devices do not always really need the computing performance of the x86 platform, they are widely used.

The major causes for this phenomenon are the lower costs of development as well as the reduced “Total Cost of Ownership” (TCO). The developer is able to reuse code which has been originally written for personal computer operating systems. As this code is typically only available in the form of closed source libraries, which the programmer is often bound to. Therefore the developer is forced to use the x86 platform for his tasks.

The reduced TCO is often second aspect for using x86 platforms for embedded tasks. As most decision makers believe, that they can reduce servicing costs by building their embedded software on top of standard software, the embedded market is compelled to use x86 in many fields.

These aspects can be seen as drawback or as benefit at the same time. But in any case the tendency of using x86 technology in industrial appliances is constantly growing and therefore demanding new developments in many fields.
1 General Overview

1.2 Rationale for the project

To sum up the stated market observations it can be said that the optimization of the x86 technology could have a great impact on the embedded computing market sector, which is no longer dominated by small microcontrollers. Moreover, the embedded computing sector can be considered to have the greatest potential for saving energy on x86 platforms.

This estimation is based on an examination of the nature of embedded applications – these are optimized for dedicated tasks, and therefore present a realistic opportunity to create stable energy saving profiles, which will not interfere with the normal processor operation.

Furthermore, it is possible to also do a critical observation of the peripheral components. It can be assumed that most of the peripherals, which are used in the embedded sector, can be considered as so-called “Legacy Components”. These legacy components have been used in x86 embedded systems for years and have not undergone any formal critical energy review. Examples for such peripheral components are coin-changing units as well as old USB devices.

Considering the number of embedded applications that use special x86 embedded modules, it is suggested here that research work in this area is worthwhile for the future.

1.3 Project Objectives

The proposed research work is part of a larger project, which is being be carried out in cooperation with industrial partners. The overall project is titled: “Development of an Embedded Controller Management Interface”. The most important partner is Kontron Embedded Modules GmbH, see the logo in figure 1.1, which is one of the leading manufacturers of embedded x86 solutions.

![Kontron Logo]

**Figure 1.1:** Kontron company logo

Kontron Embedded Modules GmbH has one of its biggest development units in Deggendorf, which is an important contributor for the ongoing research and development work.
The specific objectives of this masters thesis research project are considered as a subset of the larger project. The longer-term goal of the full project is the development of a comprehensive design tool-set for the industrial partners, who are mainly customers of Kontron Embedded Modules GmbH.

**Figure 1.2: Principal project goals**

As illustrated in figure 1.2, the toolset consists of two parts. The first part is concerned with the energy saving related topics. These energy-saving extensions enable the OEM customer to do an efficient configuration of usage-dependent energy saving profiles. In addition to this, they can be used to match the power saving requirements.

The second part of the project is concerned with the so-called “Embedded Needs”. This aspect covers the development of typical embedded extensions and consists of the respective protocol layers, as well as the development of autonomous subsystems. The need for such embedded hardware-based extensions is based on two demands:

- The energy saving subsystems require access to the operating system and the chipset. That access is established by using the embedded extensions.
- The embedded OEM customers have a constant need for all kinds of extensions, which shortens their development cycles. Moreover, the availability of these APIs can be seen as some kind of incentive to get the OEMs to make use of the energy extensions in a second development step.
As a result of the general project objectives which were stated above, this thesis does not only cover energy saving related content. There are further concepts presented in Chapter 6 which consists of some topics which belong to the “Embedded Needs” area.

1.4 Novelty of the Project

A key novel aspect of the work is the combined focus on the hardware and operating system based approach to the power management problem. The current general research approaches on x86 embedded power management are aligned to the user behavior of the embedded appliance. However, the developers of such appliances often fear that the user might interpret an energy saving state as a product malfunction. This can be compounded by the fact that most of the current x86 based embedded appliances are running 24 hours a day and 7 days a week.

Although the ACPI specification claims that it enables new power management technologies to evolve independently of operating systems and hardware, while ensuring that they continue to work together, there is only scant published literature that discusses optimal working models, especially in embedded environments.

1.5 Literature Survey

To understand how energy management works in generic operating systems, it was necessary to find documentation dating back to the time when the research on dynamic resource scaling began. Very useful papers were written at the University of Erlangen, beginning in 2002\(^1\). This research had the goal to develop new adaptive algorithms for frequency and voltage scaling. These algorithms read the information which is provided by special hardware registers in order to find the optimal values for CPU voltage and frequency. By permanently adapting the voltage as well as the frequency, an optimal balance between energy consumption and computation power is reached. The results of the research work had a great influence on the behavior of the operating systems which are common nowadays.

Other work which was carried out at the same time did not have the same goal to develop adaptive algorithms for generic purposes. The work presented in [49] had the aim to do an

---

\(^1\) Ref. the outstanding PhD thesis of Frank Bellosa [4] and the diploma thesis of Andreas Mull [32]. Most of the papers can be obtained from here: [http://www4.informatik.uni-erlangen.de/Research/PowerManagement/](http://www4.informatik.uni-erlangen.de/Research/PowerManagement/) (last viewed: March 2010).
General Overview

An analysis of the source code in order to optimise the performance and the energy consumption. The work resulted in an improvement of the compiled code.

To understand the evolution of the energy management standards, it is necessary to have a closer look on the APM standard [15], which was the first standard introduced by the Intel Corporation and others in the year 1995. The experience with that standard led into deeper research on the ACPI standard, which is today’s virtually de-facto standard in energy saving approaches on x86 platforms. The ACPI specification is very useful if concrete problems have to be solved. For the initial introduction, the book “Building the Power Efficient PC” [27] was very helpful. This resource points to numerous other papers and specifications which have been written on energy saving.

To understand how energy saving is performed on modern x86 operating systems, the resources on www.lesswatts.org were very helpful. To get a feeling for how deep energy saving is embedded in a multi-user multi-tasking operating system, the information in the book “Linux Device Drivers” [57] is very impressive. That information must be applicable to closed source systems, too. As it is very difficult to get information from companies such as Microsoft it is hard to do comprehensive research work on the Windows Operating System. Nevertheless it was possible to get some information on Microsoft Windows ACPI through some unofficial channels which were very helpful.

During the development phases the Low Pin Count specification [9] and the I2C specification documents [44] were used. It was also necessary to make heavy use of the data books of the relevant microcontrollers (Silabs 8051 [29] and NXP LPC24xx [43]) to get the specification work done. For gathering all relevant information on the “Computer on Modules” Standard (COM Express), it was necessary to make use of the corresponding specification which is hosted by the PICMG. Fortunately, the Kontron Embedded Modules GmbH is one of the core members of PICMG so it was not a problem to get this specification.

A review of the currently published papers (IEEE, Intel Corporation and others) completed the literature research. After that it was possible to decide on the optimal direction for future development, in the context of this research project.

Non members must purchase a hard-copy version which can be ordered here: https://www.picmg.org/v2internal/specorderformsec-nonmember.htm (last viewed: March 2010)
1.6 Publication Resulting from this Work

An IEEE conference paper has been submitted to the IEEE Applied Electronics Conference in Pilzen in April 2010, as follows:


This paper describes the research work and shows it is possible to monitor the electrical power consumption of an embedded module in order to perform instant online energy profiling. The concept of implementing a back plane module solution for real-time monitoring and control is presented in the paper. Appendix A shows the full draft paper.

1.7 Structure of the Thesis

As many people do not know exactly which fields employ embedded x86 technology, Chapter 2 gives a short overview. This chapter also describes the x86 module standards which were used during this research work.

Chapter 3 introduces the drawbacks of the x86 technology. The drawback of high power consumption is very typical for the x86 platform. Therefore, this chapter also familiarizes the reader with the origins of power dissipation in general, and such fundamentals are then projected back to the x86 platform.

In order to determine the work packages for the research, it was essential to get a comprehensive overview of the state-of-the-art technologies which are currently used. Because of the huge amount of literature which has been published on energy saving, the corresponding Chapter 4 presents only a selection.

For verification purposes, it was very important to create a measurement setup which is capable of measuring a huge number of signals. The actual achievements here as well as a sketch of the development process is presented in Chapter 5.

The remaining Chapters 6 and 7 deal with the concepts which have resulted out of the findings of the state-of-the-art chapter. Whereas Chapter 6 only describes the conceptual issues, Chapter 7 is concerned with concrete implementation considerations.
1.8 Summary

This masters thesis describes the development of a hardware based system for energy saving solutions fulfilling embedded tasks. The project is still ongoing. Although not a completely finished demonstrator can be presented at this project stage, the presented work can be used as a basis for implementing energy optimized systems and this might be useful to get a first insight into the complex field of energy saving solutions on embedded x86 systems.
2 Background to the Embedded x86 Standards

2.1 Introduction

Before new concepts on energy saving can be discussed, there is a need to get an overview of the embedded x86 standards. As this project has a second aim, which involves the development of new toolsets for x86 based embedded devices, it is also necessary to identify the rationale for why developers choose the x86 platform for their embedded appliances.

2.2 Objective

This chapter aims to briefly introduce the backgrounds for the embedded x86 market. It includes a short market overview as well as an introduction to some of the most common standards.

2.3 Why the Embedded Developers Choose x86

The x86 platform is gaining more and more popularity in many fields of computing. This trend can be seen in the areas of consumer products as well as in the fields of embedded computing.

Considering consumer products, the architecture switch done by Apple is a good example of this phenomenon. On June 6, 2005 Apple Computer Inc. announced that they had decided to discontinue using the PowerPC platform for their computers in favor for the Intel x86 platform.
The same progress can be observed in the embedded world which is historically dominated by small and cost effective microcontrollers and “System on a Chip” (SoC) solutions. Since the early 90’s, this market is continuously conquered by the x86 technology. A good indicator for this is the success of the Kontron Embedded Modules GmbH, which was founded 1991 as Jumptec AG.[1]

The Jumptec AG introduced their first full integrated x86 embedded computer module in the year 1993. Since that time it has become very common to use that platform for any kind of embedded application. Examples for typical usage are:

- Kiosk information systems which provide services like general Internet access or dedicated information
- Terminal systems like cash dispenser systems or ticket vending machines
- Industrial terminals for automation and manufacturing control
- Multimedia systems like gambling machines[2]

The most important advantage for adopting the x86 platform over others is the possibility of using operating systems which are only available on that platform. This issue applies to the general consumer market as well as to the embedded market. The major operating system here is Microsoft Windows. Microsoft Windows is very prevalent in all kinds of x86 based appliances.

To fulfill the emerging demands which arise from the embedded usage of Windows, Microsoft has enriched its new Windows generations, starting with Windows Vista. Since the introduction of Windows Vista the GUI is no longer part of the operating system kernel, which enables smaller installations. Furthermore, it is possible to use special stripped down versions of Windows which can be configured for specified tasks only. This is a feature which was previously confined to systems like GNU/Linux, VxWorks or QNX for a long time.

Another advantage to using the x86 platform is the issue of code reuse. Code which has been written for personal computers can be very useful in the development of an embedded appliance. Reasons for applying code reuse could be the reduction of development time, as well as the absence of “know how” by the developer in certain application areas. A good example of such a case is the development of a kiosk system, which uses a normal Internet browser for displaying information.

---

[1] Jumptec was founded 1991 in Deggendorf and had a fast ramp up. In the year 2002 they merged with Kontron Embedded Modules GmbH in order to gain more market dominance.

[2] The mechanical “one-armed bandit” is more and more replaced by computerised successors.
At a first glance, the advantages of code reuse do not seem to be enough to bind the developers to a certain hardware platform, like the x86 platform. But many of the commercial software companies try to lock-in their customers by only providing pre-compiled libraries. As the usage of these modules is strictly bound to a specific operating system and its corresponding hardware architecture, the code reuse aspect often leads to the decision to use the Windows / x86 duo which provides a well-known development tool-chain (e.g. Microsoft Visual Studio).

A third advantage of the x86 platform is the great pool of peripheral components which are available. These devices often provide full driver support on all common operating systems, which additionally speeds up the development time.

Another motivation to implement the embedded solution on top of the x86 platform could be the great computing power of the x86 processors, which can be used for various enhancements.
2 Background to the Embedded x86 Standards

The Samsung coke vending machine which is shown in figure 2.1 is a good example for the increased computing power demands of embedded solutions. According to the information which is provided by Samsung [56], that embedded device has the ability to show all kind of options on a large touch-screen display. Moreover it offers many advanced purchasing options using credit or debit cards and cell phone payment. The vending machine is equipped with network and Wifi which can be used to monitor the stock or to update the information which is shown on the large touch-screen display. Additionally, a built-in camera and a shock sensor enables the capturing of potential vandals.

2.4 The Future of Embedded x86

This year’s (2010) Embedded World Fair which took place in Nuremberg, was full of new product examples. The most interesting prototypes, regarding the future of the x86 platform, were shown at the booth of the Intel Corporation.

Beside the traditional solutions for problems regarding automation and industrial purposes, a prototype of a novel digital signage system was presented. This concept includes a holographic glass panel which shows the signage information to the customer. Additionally the system catches the observer’s eye and adjusts the observed display to the respective eye-view.

![Figure 2.2: Demonstrator for the Intel digital signage proof of concept](image)

To accomplish this product application, the x86 platform, with an Intel i7 processor, is used to carry out tasks which traditionally belong to DSP based systems. The eye-view analytics
which is captured by two cameras is processed by using an application based on the Microsoft Windows Embedded Standard 2011[1].

To conclude it can be said that the future of the embedded x86 platform is no longer the consumer market, nor specialist examples such as normal vending machines, but the embedded x86 platform is gaining relevance in areas in which nobody has previously thought of.

2.5 The Backplane / Module Concept

As introduced earlier, it could be very interesting to consider the x86 platform for new embedded projects. For most new projects, however, it is not possible to use the form factors as they are used in customary personal computer systems. These form factors do not provide the compact size which is often demanded by embedded projects nor the facility of easy servicing or upgradeability.

A straight solution for this problem would be the development of a custom x86 mainboard, but unfortunately this is not possible for the most cases. The first barrier is the fact that most of the required documentation is only available by signing strict Non Disclosure Agreements (NDA) which are not obtainable by small companies.

Furthermore, it is very costly to buy the required licenses for the required BIOS packages from BIOS manufacturers like AMI or PHOENIX. The Development of an in-house BIOS solution is very time intensive because it requires low level debug adapters and therefore a deep knowledge in system programming is necessary[2].

To provide a solution for these problems and to bridge the development gap, companies like Kontron and others introduced the “Module Backplane Concept”. This concept reduces the so called “Full Custom Design” approach to a point were only the features which are actually needed are employed on the target development work. A small module consists of the complete x86 system including the processor and the whole chipset. All necessary bus lines and signals are provided through standardized connectors.


4On the Intel x86 platform a XDP debugger is required. Such devices can be bought from American Arium which is one of the leading manufacturers.
2 Background to the Embedded x86 Standards

Figure 2.3: Backplane with the corresponding module [2]

The Backplane is the part which is developed by the creator of the embedded appliance, who is also referred to as OEM customer of the module manufacturer. The backplane provides the sockets for the embedded module and makes use of the signals which are needed to solve the embedded problem. Depending on the respective problem, the size of the design might be varied. But, in every case, the concept ensures upgradeability, easy servicing and low development costs.

2.6 Module Standards

Every developer faces numerous standards. That is because every manufacturer of x86 modules tries to introduce his own specific standard which addresses specific needs. Within this section, the most relevant standards are briefly introduced.

Figure 2.4 illustrates a 10 year history of form-factors, as it applies for the major project partner: Kontron Embedded Modules GmbH. The DIMM PC is no longer relevant in current
times, because this form factor suffers from not being standardised. Therefore, the benefits of upgradeability and easy servicing do not comply here. The main target form factor of this project is the COM Express standard, as this one is the most highly-featured standard of Kontron Embedded Modules GmbH.

![History of form factors from the Kontron viewpoint](image)

**Figure 2.4:** History of form factors from the Kontron viewpoint [2]

### 2.6.1 ETX

The ETX standard can be considered as a generic standard which is available to all types of processors and chipsets. ETX stands for Embedded Technology Extended and is one of the first module standards which was ever specified. The ETX standard consists of information on module size and details about which socket has to be used and the corresponding pin-out. The ETX specification was developed by the Jumptec AG, which is now the Kontron Embedded Modules GmbH.

An ETX module measures 95 mm x 114 mm and has 4 connectors to connect to the backplane. The successor of the ETX standard is the COM Express standard which is maintained by the PICMG standartisation group.
2 Background to the Embedded x86 Standards

The reasons for transferring the control over the standard to an external group, are due to some bad past experiences with the self-maintained ETX standard. It turned out that the self-maintained ETX standard was incompletely specified and this let too much room for 3rd party changes which often made the modules incompatible with each other.

2.6.2 COM Express

Just like the ETX standard the COM Express standard is a generic standard for all types of processors and chipsets. Moreover the COM Express standard is the successor of the ETX standard. Therefore it was initiated by Kontron Embedded Modules, too.

As the Kontron Embedded Module GmbH features this standard, the evaluated samples and all related bottom-up work is related to COM Express. To avoid the standardisation problems, which emerged in the former ETX form factor, the standard maintenance was transferred to the non-profit standardisation group – PICMG. The standard is open to every other company which is interested but it cannot be downloaded freely.\footnote{A hard copy of the standard can be ordered here: \url{https://www.picmg.org/v2internal/specorderformsec-nonmember.htm} (last viewed: March 2010).}

As shown in figure 2.6 the COM Express standard supports more than one size. The PICMG specification in its current revision 1.0, and suggests the following sizes:

- Basic: 95 mm x 125 mm
- Extended: 110 mm x 155 mm
In contrast to the ETX specification the COM Express standard does not rely on 4 fixed connectors. The standard affords the module manufacturer the choice to adapt the number of connectors to the number of signals which should be provided. The following combinations are possible:

- **Type 1**: Single connector (220 pin), 6 PCI Express lanes, no PEG, no PCI, no IDE, 4 SATA, 1 LAN
- **Type 2**: Double connector (440 pin), 22 PCI Express lanes, PEG, PCI, 1 IDE, 4 SATA, 1 LAN
- **Type 3**: Double connector (440 pin), 22 PCI Express lanes, PEG, PCI, no IDE, 4 SATA, 3 LAN
- **Type 4**: Double connector (440 pin), 32 PCI Express lanes, PEG, no PCI, 1 IDE, 4 SATA, 1 LAN
- **Type 5**: Double connector (440 pin), 32 PCI Express lanes, PEG, no PCI, no IDE, 4 SATA, 3 LAN
2 Background to the Embedded x86 Standards

Figure 2.7: The COM Express signals and their connector [2]

2.6.3 ESMexpress

This standard can be described as a computer-on-module standard for extreme environmental conditions. The standardisation group website is: [http://www.esm-express.com](http://www.esm-express.com).

Figure 2.8: ESMexpress module [35]
This computer module standard was not specifically part of this research work so its purpose here is to act as a current example for the numerous interesting form factors which are on the market. ESMexpress modules can be obtained from MEN Micro Electronic Nuremberg GmbH which is also the initiator of this standard. The module standard contains only one size which is 95 mm x 125 mm.

The most interesting aspects regarding that form factor is the extended temperature range, the extreme shock resistance which provides stability to withstand vibration, and the electromagnetic compatibility. The extended temperature range starts at $-40 \, ^\circ C$ and ends at $+85 \, ^\circ C$.

These important characteristics are achieved by the use of low power processors like the Intel Atom and the usage of a special mechanical concept which is shown in figure 2.9. The computer board is embedded into an aluminum frame, which enables fan-less designs. Additionally it is fully compatible to the COM Express standard. This make it possible to replace an ordinary COM Express module with an ESMexpress module in order to reach the extended temperature ranges.

This form factor is another example of the constant progression of the embedded x86 technology into new fields. With this rugged form factor it is possible to create appliances for the military sector as well as for rough automation duties.
2.7 Summary

This chapter provided the necessary background information to understand the advantages of using the x86 platform for embedded solutions, and presented some reasons for the choice. The next chapter focuses some of the problems which arise, when x86 technology is chosen for an embedded appliance.
3 Drawbacks of the x86 Platform

3.1 Introduction

As shown in the last chapter, the x86 technology is gaining more and more presence in the embedded world. However the decision to use x86 for a certain application can be fraught with some distinct problems, which are specific to this platform.

3.2 Objective

This chapter covers some of the most obvious problems for the x86 technology. The topics are not strictly bound to the embedded sector, but they are more prevalent there. This chapter also has the aim to introduce some of the most important terms and formulas which are referred to in the following chapters of this thesis. It also sums up the most important areas for power dissipation from a physical point of view.

3.3 Security and Stability Issues

The decision to use the x86 platform is often associated with the decision to use a mainstream operating system. In most cases that specific operating system is not available for another hardware platform. One good example for such an OS is the widely used Microsoft Windows operating system family.

Microsoft Windows is available in three different types:

- Windows for Desktops

---

1Because of the availability on many hardware platforms, in this context GNU/Linux would be a bad example.
### 3 Drawbacks of the x86 Platform

- Windows Embedded
- Windows CE

Whereas the last item on the list (Windows CE) is not a fully featured Windows OS, the former two are full featured Windows versions. The Embedded Windows Edition enhances the capabilities of the desktop variants with special embedded functions like a layered filesystem\(^2\) which reduces the write access to solid state disks.

**Figure 3.1:** Samsung coke vending machine has a problem \[34\]

Despite of all the improvements on the embedded versions of Windows, it continues to be a standard Operating System which was originally designed for the mass market. So it is clear to see that all problems which are associated with the desktop variants also apply to the embedded version.

Besides the “Denial-Of-Service” effects, such as the “Blue Screen Of Death”, such as that shown in Figure: \[3.1\], there are numerous other security relevant problems. To get a feeling for these problems a daily visit to the most comprehensive security Internet platform [http://www.securityfocus.com](http://www.securityfocus.com) is advised.

These problems do not only affect Windows operating systems, they are significant for all types of generic systems, involving x86 GNU/Linux and others, too. All the Windows OSs

provide a huge amount of application interfaces and consist of a large amount of code. These aspects raise up the probability of errors compared to small systems with a small code-base, which are often tailored to solve only a specific problem.

3.4 Energy Related Problems

A most significant drawback when using the x86 platform, is the problem of the high energy consumption. The following sections cover this problem and introduce some terms and discuss the origins of these drawbacks.

3.4.1 Origins of Energy Inefficiency

As the x86 platform is one of the oldest CPU architectures, its design was not originally optimised for energy saving needs. Since 1978, when the first x86 processor was introduced by the Intel Corporation, the whole evolutionary process was characterized by an increase of computing power.

To advance the computing power, several techniques, like the increase of the system clock speed, were used. As the x86 technology follows the so called CISC (Complex Instruction Set Computer) philosophy, the number of instructions in the instruction sets was also constantly rising. Examples include numerous extensions like multimedia extensions³ or special hardware capabilities, which are needed for faster visualisation and virtualisation.

This progress is exactly the opposite to the RISC (Reduced Instruction Set Computer) approach, which tries to limit the complexity of the CPU and therefore the number of special hardware based commands which are available through the instruction set of the respective CPU.

As a result, the number of transistors on the x86 had to be constantly increased to implement all the new features. To continue to maintain compatibility, it is hard to remove functions which were integrated in the past⁴. As a result of this development it can be stated, that the x86 platform is suffering more than other platforms with the issue of high energy consumption.

³Every CPU manufacturer has its own name for the same or similar feature. Think of the MMX technology on Intel platforms or the 3DNow! technology on AMD systems.
⁴A good example for this is the so called A20 gate, which is explained in Appendix B.
3.4.2 Components of Energy Consumption

In a simplified manner, the modern CPUs can be seen as a collection of ordinary CMOS circuits. For an explanation a circuit is shown in Figure 3.2. That figure shows an inverter circuit which is used in digital circuits for several purposes. It can be used to invert signals as well as forming an output gate which is capable of doing tristate output for bus coupling purposes.

As this type of circuit is often used in digital circuits (and therefore in microprocessors, too) it is suitable for the explanation of the distinct factors which are causing power consumption.

![CMOS inverter circuit](image)

On CMOS circuits the energy consumption consists of the following two distinct components which also can be transferred to other FET (Field Effect Transistor) based chip-designs:

- Static power consumption
- Dynamic power consumption

---

5 Tristate means that both transistors are in their high-impedance state. As a result the output is not associated to any potential and is therefore not impairing the signals on the respective bus line.

6 Texas Instruments has published a very good technical report on that topic.
3 Drawbacks of the x86 Platform

Static Power Consumption

Looking at the CMOS inverter (Figure 3.3) again, it is clear to see that in both possible cases one of the involved Field Effect Transistors is in its OFF state. If the switching process is neglected and only the static state is examined, the quiescent current is zero. That is because there is never a direct path between VCC and GND.

In reality, however, the quiescent current is not zero. That current, which causes a static power consumption, is caused by parasitic diodes which are illustrated in Figure 3.3. These diodes are reverse biased and only their leakage current contributes to the static power consumption. The sum of all leakage currents caused by these parasitic diodes or parasitic bipolar transistors, builds the majority of the whole leakage current of the processor (ref. Equation 3.2).

![Figure 3.3: Model describing parasitic diodes present in CMOS inverter](image)

The leakage current $I_{lkg}$ can be calculated as it is shown in Equation 3.1:

$$I_{lkg} = i_s \cdot (e^{\frac{V}{T}} - 1)$$  \hspace{1cm} (3.1)

$i_s =$ reverse saturation current  
$V =$ diode voltage
3 Drawbacks of the x86 Platform

\[ k = \text{Boltzmann's Constant} \]
\[ q = \text{electronic charge} \]
\[ T = \text{temperature} \]

As a result of this, the static power consumption \( P_s \) is calculated as shown in Equation (3.2):

\[ P_s = \sum (I_{lk}) \cdot V_{cc} \quad (3.2) \]

- \( P_s \) = static power consumption
- \( I_{lk} \) = leakage current
- \( V_{cc} \) = voltage supply

In addition to the leakage current, another effect can increase the static power consumption. Considering that the switching voltage for the p-FET of case 2 in Figure 3.2 has a voltage drop which causes it not to open completely, it is clear to see that a current flows from VCC to GND. As such a unwanted current appears during a static state, this is another notable contribution to the static power consumption.

Dynamic Power Consumption

\[ \begin{align*}
C_{GD} & \quad C_{ds} \\
R_D & \quad C_{gS} \\
R_B & \quad R_S \\
G & \quad N-\text{Epitaxie} \\
\text{Source} & \quad \text{Aluminium} \\
\text{Gate} & \quad \text{SiO}_2 \\
\text{Drain} & \quad \text{Poly-Si} \\
\text{Source} & \quad \text{Aluminium} \\
\text{Gate} & \quad \text{SiO}_2 \\
\text{Drain} & \quad \text{Poly-Si} \\
\end{align*} \]

**Figure 3.4:** Model of a SIPMOS introducing the internal capacitors of a FET

Besides the parasitic diodes which are the cause for the static power consumption, there are also capacitors inside the FET. As the FET is based on an electrical field, these capacitors are
needed for the correct operation of the transistor. In addition to the wanted capacitors, there are also parasitic capacitors which arise between the FET’s or inside the FET structure. Some of the standard capacitors are shown in Figure 3.4.

These capacitors have to be charged/discharged whenever the circuit is switched. This current and the so-called “through current” (this is the “glitch current” which flows from VCC to GND when the p-FET and the n-FET from Figure 3.2 are switched at the same time) is responsible for the dynamic power consumption. All internal factors of the dynamic power consumption are called “Transient Power Consumption”.

The transient power consumption can be calculated as shown in Equation 3.3

\[
P_T = C_{pd} \cdot V_{CC}^2 \cdot f_I \cdot N_{sw}
\]

\(P_T\) = transient power consumption
\(C_{pd}\) = dynamic power dissipation capacitance (ref. Figure 3.4)
\(V_{CC}\) = voltage supply
\(f_I\) = input signal frequency
\(N_{sw}\) = number of bits switching

It is not only internal capacitors which contribute to the dynamic power consumption. In addition there is a second factor which is called “Capacitive Load Power Consumption”, referred to here as \(P_L\). This \(P_L\) is caused by capacitors which are outside of the integrated circuit and can be calculated as shown in Equation 3.4

\[
P_L = C_L \cdot V_{CC}^2 \cdot f_O \cdot N_{sw}
\]

\(P_L\) = capacitive load power consumption
\(C_L\) = external load capacitance (load per output)
\(V_{CC}\) = voltage supply
\(f_O\) = output signal frequency
\(N_{sw}\) = total number of outputs switching

The complete dynamic power consumption can be calculated by using Equation 3.5

\[
P_D = P_T + P_L
\]
Total Power Consumption

To get the total theoretical power consumption, the results of Equation 3.5 and 3.2 have to be added:

\[ P_{\text{tot}} = P_D + P_S \]  

(3.6)

\( P_{\text{tot}} \) = total power consumption  
\( P_D \) = dynamic power consumption  
\( P_S \) = static power consumption

\[ 
\begin{align*}
\text{CPU} \\
\text{(e.g. Intel Core2Duo)} \\
\text{Northbridge} \\
\text{(e.g. Intel GMCH)} \\
\text{Southbridge} \\
\text{(e.g. Intel ICH7)} \\
\end{align*}
\]

Figure 3.5: Basic components of a typical x86 Intel based system

The calculated total power consumption \( P_{\text{tot}} \) is the power consumption for one bit which is switched. Considering that a modern x86 system consists of a CPU with millions of transistors\(^7\) and additional chipset hardware components like Northbridge, Southbridge and System

\(^7\)The Intel XEON which is manufactured using the 45nm process technology, consists of about 781 million transistors.
Memory it is clear to see how the micro-consumption accumulates. A typical system layout of a current x86 Intel based system is shown in Figure 3.5.

### 3.4.3 Power Consumption of Current Systems

Due to the short product development cycles it is not easy to get comprehensive information on the power consumption of current x86 systems. The CPU manufacturers provide a so-called TDP (Thermal Design Power) value, which has the intention to provide a guideline for cooler dimensioning. As there is no general benchmark for the TDP determination, every CPU manufacturer can define its own one.

The TDP is not adequate for comparing the embedded x86 world with smaller micro-controller based solutions for two reasons:

- The TDP covers only the CPU and not the whole system which consists of the components shown in Figure 3.5.
- The CPU manufacturers often try to let the respective CPUs look “greener” than they are in reality. As there is no standard benchmark, the TDP can be “sugarcoated” by applying optimistic benchmarks instead of worst-case scenarios (like burn-in) [58].

In order to do a reliable energy consumption measurement, a benchmark has to be selected which utilizes the regions of the x86 system which are of interest. Then a measurement system has to be set-up which is constantly measuring voltage and current intensity. An example for such a system, which was designed for measurements of COM Express modules, is described in Chapter 5.5.

Because it is not possible to present a comprehensive comparison measurement within the research work, Figure 3.6 is used to provide a summary of many modern CPU cores in the context of their power consumption and computing power. The measurements were done by the German Heise publishing house which is known for their objective reporting.

The computing power was measured by using the Cinebench benchmark which can be obtained from [http://www.maxon.net](http://www.maxon.net). The measured power consumption is the consumption of the whole system, including all parts which were shown in Figure 3.5. VIA processors were not part of the measurements because they have lost market relevance. Their power consumption can be compared to the Intel Atom processors.

The x86 processor world can be divided into three major groups:

- Energy saving processors (like the Intel Atom)
3 Drawbacks of the x86 Platform

- Desktop processors
- Server processors (High computing performance and high energy consumption – e.g. Intel XEON)

![Figure 3.6: Cinebench score vs. power consumption](image)

As can be seen in Figure 3.6, the energy saving processors like Intel Atom are far away from beating out conventional micro-controller based embedded solutions. The reason for this can be related to inefficient working chipsets, as well as over optimistic marketing promises (TDP).

However, it must be stated that the improvements on the Atom platform are remarkable and this will be refined continuously. Unfortunately some of these improvements have a severe impact on computing performance (cp. Figure 3.6).
Because of the fact that Kontron Embedded Modules GmbH Deggendorf has its focus on the development of x86 modules, which are equipped with desktop processor technology\textsuperscript{8}, the conceptual considerations are targeting only this group of processors and not “green” processors like the Intel Atom.

Independently of the conceptual considerations, a description of the more interesting energy saving improvements of the Intel Atom will be reviewed in Chapter 4, which is reviewing some selected state-of-the-art technologies.

### 3.5 Summary

This chapter’s purpose was to introduce the key drawbacks of the x86 processor world, introducing the most important physical fundamentals of power consumption and the corresponding formulas.

The next chapter focuses on the current state of the art techniques which can be used in order to save energy.

\textsuperscript{8}Kontron Hamburg is doing the development of the Intel Atom COM Express modules.
4 Review of State of the Art Technology

4.1 Introduction

In the previous chapter, some physical aspects of power dissipation were introduced. This chapter aims to give an overview which energy saving techniques, which are used in modern computer systems. Besides the bare technologies a discussion of the most important x86 power saving concepts is also presented done.

Because of the numerous energy saving technologies which are in use, this review cannot be totally comprehensive. However, most of the presented energy saving technologies concern the same physical effects, so the discussed technologies are representative of the state of the art in this area.

4.2 Objective

This chapter introduces some selected state-of-the-art technologies which are used for power saving. Additionally, these technologies are sorted into a suitable classification scheme. The results of this chapter are used in Chapter 6 to determine further steps for the research on embedded x86 energy saving.

4.3 Layers of Energy Saving

When talking about energy saving on x86 environments it must be stated that there is no “definite” technology in this field. The x86 energy saving, as well as the energy saving on other platforms, is a conglomerate of several technologies and concepts which can be considered to be in a layered form.
4 Review of State of the Art Technology

Figure 4.1: Layers of energy saving

A simplified illustration of the existing layers is provided by Figure 4.1 which also describes the dependency between them. Notable are the layers inside the “Management Layer” box which combine the energy saving capabilities of the preceding ones. In modern appliances, the management layer is typically implemented by the respective operating system, but there are also legacy hardware-based implementations.

The mesh work between all layers results in the x86 systems’ energy saving technology. The following sections introduce the specific layers and provide example technologies which are used in modern x86 based computers systems.

4.4 Hardware Layer

The energy saving technologies in the hardware layer can be divided into the following two groups:

- Autonomous working technologies
- Technologies which are associated with higher layers
The first group includes all technologies which can be used to save energy independently from a higher controlling instance, whereas the second group includes all technologies which are bound to layers like the management layer.

![Figure 4.2: Hardware layer – Types of technologies](image)

Several hardware parts of a x86 based computer system belong to both groups. Therefore it has been decided to group this section by hardware type and to make the group association within the explanation of the respective energy saving technology.

### 4.4.1 Power Supply

**Circuit Optimisations**

The optimisation of the power supply components is the first measure which can be taken in order to make an appliance more energy efficient. Most times an increased energy efficiency is gained by replacing old linear voltage regulators \[47\] with switched mode \[48\] power supplies.

Most of the power supplies used in the x86 world are already switched mode based. Therefore the replacement\[\textsuperscript{1}\] of old switched mode power supply controllers by smarter ones can result into more energy efficient designs. Such smart controllers are based on DSP algorithms which are capable of self-adjusting the operating parameters in order to ensure the most efficient operating point of the power supply.

\[\textsuperscript{1}\]A good example for smart switched mode power supply controllers are the products of [http://www.powervasion.com](http://www.powervasion.com)
As the optimisation of the power supply is not strictly bound to the x86 technology, this topic has not been part of the research. Although, there has been little investigation in this area, it can be stated that many x86 based embedded appliances could be more energy efficient if their corresponding power supply units would be revised.

### Segmentation of the Voltage Rails

One big conclusion of Chapter 3 is that every semiconductor device consumes energy in every operating state, regardless if this state is an idle state or a productive state. Consequently a modification to current state-of-the-art power supplies is achieved which can be described as “voltage rail segmentation” technology.

![Segmentation of voltage rails](image)

**Figure 4.3:** Segmentation of voltage rails

The basic working principle is illustrated by Figure 4.3 and simply consists of electronic switches which can be used by a higher management layer to switch different devices on or off. This technology is already used in conjunction with the power states of the ACPI standard.

However, it turns out that the standard PC main power supplies and therefore most of the embedded main power supplies, do not provide any extended capability for switching off individual connected devices.
4.4.2 Processor and Chipset

Modern x86 processors and their corresponding chipsets provide many functions which can be used to save energy consumption on the whole system. The following subsections introduce the most interesting techniques. As the Intel Atom can be considered as the most innovative low power processor, a discussion of its competitors had been omitted\(^2\).

**Intel Atom - CISC meets RISC ?**

Looking at the power consumption values of modern Intel Atom\(^3\) processors and comparing them with the consumption values of RISC processors it can be assumed that Intel is adopting parts of the RISC technology to reach these results. Although the Intel Corporation is not able to use pure RISC technology because of the CISC x86 instruction set, big changes in other areas of the processor architecture were achieved. These changes are discussed in the following section.

As introduced in Chapter 3, one of the biggest problems in big semiconductor structures are the leakage currents. The leakage current problem increases when shrinking the silicon based semiconductor structures.

\[\text{Figure 4.4: Traditional transistor vs. high-k based transistor [10]}\]

\(^2\)A discussion of the current VIA processor vs. Intel Atom can be found here: [http://www.xbitlabs.com/articles/cpu/display/intelatom-vianano_2.html](http://www.xbitlabs.com/articles/cpu/display/intelatom-vianano_2.html) (last viewed: April 2010).

\(^3\)Intel Atom is also known as LPIA (Low Power Intel Architecture)
This is because the $SiO_2$ based gate insulation gets worse as it gets smaller. Additionally it is not possible to do an unlimited shrink of traditionally generated silicon based FET structures. At a certain point the $SiO_2$ gate is no longer providing a sufficient insulation and therefore the transistor stops working.

To resolve these problems, the chip manufacturers started research programs to find replacement materials for the traditional silicon based semiconductor structures. The Intel Corporation found a solution which is illustrated in Figure 4.4.

The core idea [13] was to replace the gate insulating $SiO_2$ based material with a so called high-k material, whereas high-k stands for a material with a high dielectric constant. Therefore this material is able to provide much more capacity when the chip structure size is untouched or a sufficient capacity when the structure is shrunk. Since the new gate insulator materials were not compatible with the gate polysilicon, there was also a demand to replace the polysilicon with metal based materials.

![Processor Power Breakdown](image)

**Figure 4.5:** Intel Atom - Processor Power Breakdown [10]

The new transistor technology was introduced with the Intel Atom 45nm fabrication process. Compared to the former 65nm fabrication process, the new 45nm technology has the following advantages [10]:

- 2x increase in transistor density; either smaller size or more transistors
- 20% increase in transistor switching speed
• 5x reduction in source drain leakage power
• 10x reduction in gate oxide leakage power
• 30% reduction in transistor switching power

The new transistor technology is supplemented by new low k interconnection materials which consists of substances with a low dielectric constant. These new interconnection materials reduce unwanted parasitic capacitors and therefore reduce the dynamic power consumption of the semiconductor.

Besides the semiconductor fabrication improvements, heavy energy related improvements were achieved across the whole processor architecture. Figure 4.5 provides a chart comparing the Intel Atom to a regular Intel desktop processor, also listing all architectural measures.

![Diagram of processor architecture](image)

**Figure 4.6:** Conventional IA x86 macro-op decode (simplified)

One of the biggest decisions was the removal of the so-called Out-of-Order Execution (OOO) feature, which is integrated in every other Intel processor generation since 1995. Since that time the Intel Architecture (IA) consists of a RISC core with upstream decode units (cp. Figure 4.6).
These decode units\(^4\) are capable of decomposing complex x86 CISC macro instructions into small RISC micro instructions. The main advantage of this procedure is that the small RISC microinstructions can be re-sorted in a way where independent micro operations can be executed out of their original order. Accordingly it is possible to speed up the execution significantly. The only drawback of this technology is that there is a large amount of transistors needed, in order to implement this re-sorting feature.

\[\text{Figure 4.7: Intel Atom decoding engine}\]

Figure 4.7 shows the Intel Atom decoding engine which has an in-ordered execution instead of an out of order execution. According to the chart which is provided in Figure 4.5 this measure saves up to 26%.

With removing the out of order execution structures from the processor die, the re-coding of some of the x86 macro instructions became obsolete. Consequently the decoding process has been divided into 2 simple decoders forward selected x86 macro instructions and a complex decoder responsible for decomposing complex x86 instructions.

The forwarding of some selected x86 macro instructions is known as “macro-op fusion” and also contributes to the energy saving because complex decoding structures have been simpli-

\(^4\)As a big advantage, the x86 macro instruction code translation table is update-able. This feature is very helpful when a x86 macro instruction is malfunctioning. The update is usually done through a BIOS update.
fied. A key point of this subsection is to say that the Intel Atom processor is more CISC than any other modern x86 instruction set capable processor.

Additional information on the internal structure of modern IA processors can be found by consulting the “Intel 64 and IA-32 Architectures Optimization Reference Manual” [11] which also contains information on optimising the software for specific processor architectures. This information can be used to optimise the software for best computing performance as well as for energy efficiency.

In the future Intel plans to integrate the Atom IP core [51] in SoC solutions which integrate the whole system onto a single chip. So, Intel is able to place the Atom in product segments which were reserved for traditional SoC processors like the ARM. The remaining drawbacks of this development are:

- The Atom still keeps x86 overhead for compatibility reasons, therefore the energy efficiency is mainly reached by optimising the semiconductor
- The Atom platform still needs a BIOS because Intel can not affront the BIOS manufacturers
- The Atom will not be freely programmable as current ARM solutions, because NDA documents are needed for getting all the required low level information

**The Processor C-States**

Virtually all modern processors support special energy saving capabilities. These capabilities are bundled in so called “processor states” or “C states” and can be triggered by the management layer. Within the specific states, the following measures might be implemented:

- core voltage scaling
- frequency scaling
- clock gating - disable the clock in certain regions of the chip
- enable/disable the usage of the caches
- switch off of peripheral outputs

As not every processor architecture is made for the same purpose, the implemented measures and the number of available C states might vary. According to Equation [3.3] the scaling of the core voltage and the frequency has a big impact on energy saving. A big advantage of the scaling technology is that it can be applied during the full functional state which is called the
Typically the frequency/voltage scaling during the C0 state is applied in fixed steps. These steps are called the performance states (P-states).

The other measures, like the clock gating, are not applicable during the C0 state because they are designed to stop the internals of the processor. To apply these technologies, the processor has to enter special C-states, are called sleep states.

The sleep state can be left by sending an interrupt to the processor. The deeper the sleep is, the longer is the wakeup time. But contrary to the longer wakeup time the advantage of energy saving is enhanced with a deeper sleep.

Figure 4.8: Intel Atom processor states [10]

Figure 4.8 shows all C states which are implemented on the Intel Atom platform. Because the Atom platform is a low energy platform, this figure shows the most important C states and their corresponding energy saving measures. The displayed states are also used in current processors of other major processor manufacturers. Additionally, a comprehensive overview of processor power states is provided in Appendix C.

The C0 HFM, C0 LFM, C0 ULFM states illustrate the capability of applying multiple frequency/voltage combinations during the C0 state. The recovery time, which is needed to come up after a sleep state, consists of a hardware component as well as a software component. The software dependent recovery time component is split up into a BIOS and an Operating System component.
4 Review of State of the Art Technology

The Processor T-States

Another mechanism to save energy on the processor is the “CPU throttling” which is parted into T-states. CPU throttling means, that the management layer sends STPCLK commands to the CPU which cause it to omit clock cycles. The T states are typically given in percentage values whereas in a T-state 50% means that every second cycle is omitted.

Figure 4.9: ACPI thermal management

Figure 4.9 illustrates that the CPU throttling is part of the thermal management of the management layer. For each temperature an action can be associated – critical temperatures are associated with throttling actions or with an emergency shut-down. As omitting clock cycles has a great impact on the dynamical power consumption, it also has direct and fast influence on the system’s temperature.

Although extensive throttling saves energy, its contribution to the system’s energy saving measures is not very high. As throttling affects the system performance in a linear way, the more energy that is saved, the slower the system becomes. As a consequence of this, most of the time, throttling is not used for energy saving purposes.

\[\text{If 7 out of 8 ticks are gated-off (the maximum possible throttling), performance is reduced by roughly 7/8 or 87.5\%.}\]
Performance Counters

Originally introduced for debugging purposes, Performance Counters provide very valuable information about the current operational status to the management layer. A performance counter is a normal counter which has been embedded into the semiconductor structures of the processor. Its purpose is to count designated events inside the processor state machines.

![Performance counter basic functionality](image)

Figure 4.10: Performance counter basic functionality

The Operating System which sits on top of the management layer, can poll the performance counter in order to get its current value. In addition to the poll functionality most processor performance counters provide an interrupt driven scheme in which an interrupt is initiated when the counter has an overflow (cp. Figure 4.10).

![Performance counter - ISR based measurement](image)

Figure 4.11: Performance counter - ISR based measurement
4 Review of State of the Art Technology

For energy saving purposes, the interrupt driven approach is the most often used. As illustrated in Figure 4.11 the management layer simply measures the time between two interrupts to determine the system load. The more often an interrupt occurs the more busy the corresponding region is. Based on this information the management layer decides to activate energy saving mechanisms (e.g. in case of IDLE) or not.

According to the “Intel 64 and IA-32 Architectures Software Developer’s Manual” [14], modern Intel x86 processors provide two performance counter units. They can be configured at runtime, to watch events which are varying between certain processor generations. Examples for such events are:

- UnHalted Core Cycles
- Instruction Retired
- UnHalted Reference Cycles
- LLC Reference
- LLC Misses
- Branch Instruction Retried
- Branch Misses Retried
- and more...

The research on performance counters pointed out that an intelligent cyclic monitoring can draw a very accurate picture [21] of the current system load.

4.4.3 Miscellaneous Peripheral Components

A computer system does not only consist of a power supply, chipset and processor. Moreover it needs peripheral components like display units or sensors, in order to communicate with the environment.

This section covers some outstanding state-of-the-art peripheral technologies which are saving energy by design.

---

[4] Cp. the OProfile project which can be found here: [http://oprofile.sourceforge.net/](http://oprofile.sourceforge.net/) (last viewed: April 2010). This project consists of a GNU Linux kernel module and uses the performance counters in order to measure the efficiency of software.
E-Paper Display

A very good example for “energy saving by design” devices are displays based on the E-paper technology. Unlike conventional computer displays, a E-paper based solution does not need energy in order to display information. The energy needs only to be provided, when the displayed information should be changed.

![Microcapsules - functional principle of Electrophoretic based displays](image.png)

**Figure 4.12:** Microcapsules - functional principle of Electrophoretic based displays [18]

The E-paper consists of microcapsules which were put into a fluid. Through an applied electrical field, the microcapsules can be moved and aligned (cp. Figure 4.12). Because the microcapsules are made out of two types of particles with different types of charge and different color, shapes can be drawn on the display by altering the electrical field.

Since the viscosity of the fluid is very high, the microcapsules cannot move without a electrical field. The technology is also known as Electrophoretic and the fluids are often referred to as E-ink.

One drawback of Electrophoretic based displays is, that the displayed information vanishes after a certain amount of time. To avoid loss of information, a periodical display refresh cycle has to be performed. A big aim of the current research is to increase the time between the refresh cycles. One solution for this specific problem has been presented by NEC LCD Technologies Ltd. by introducing a Electrophoretic display which is capable of holding the information for up to one year [52].
Another drawback is that current solutions are not able to display color information. In addition to this the latency time of E-paper displays is very high\(^7\).

![NEC LCD Ltd. Din-A3 and Din-A4 paper module](image)

**Figure 4.13**: NEC LCD Ltd. Din-A3 and Din-A4 paper module \([52]\)

Advantages of the E-paper technology are:

- High contrast and high resolution
- Not angle view dependent
- Very energy efficient
- Sunlight does not impair the readability
- Information can be displayed without running computer system

To draw a conclusion, the E-paper technology is a very interesting and useful technology which can be widely used in embedded applications. Especially in applications with rarely changing displayed information, the energy saving benefits are enormous.

**Energy Autonomous Sensors**

Another good example for extraordinary energy saving devices are the so called energy autonomous sensors. These devices gain their operating voltage by “energy harvesting”. That

\(^7\)These problems are currently under heavy research. Due to a nature article Phillips has solved the latency problem by using a technology called electro-wetting: [http://www.nature.com/nature/journal/v425/n6956/full/nature01988.html](http://www.nature.com/nature/journal/v425/n6956/full/nature01988.html) (last viewed: April 2010)
means that the device is powered by energy which is harvested from the environment. The information is usually transmitted using a wireless connection. Typical examples for these types of energies are electromagnetic fields as well as vibrations.

There are usually three major energy sources which are harvested by state-of-the-art technology:

- Photovoltaics (producing electricity from ambient light - either indoors or outdoors)
- Vibration (producing electricity from vibrations of the surface the sensor is deployed on)
- Thermoelectrics (producing electricity from a temperature gradient)

Researchers at the Fraunhofer Institute for Physical Measurement Techniques (IPM) in Freiburg are currently developing a sensor system for airplanes. This system is embedded into the aircraft skin and is capable to monitor the skin’s integrity. The energy is produced by exploiting the temperature gradient.

As this technology uses energy which would otherwise be wasted, its usage is interesting for all embedded appliances with a special emphasis on energy saving. Combined with wireless communication, the system can be used in order to save weight, too.

4.5 Firmware Layer

The next layer above the hardware layer is the firmware layer. As a x86 system currently cannot be designed as a system-on a-chip which involves all required peripheral components, it deeply relies on peripheral devices which are connected by standard bus systems.

The modern peripheral devices however, are small embedded computer systems on which dedicated Operating System is running, which is called the firmware. In order not to break the chain of energy saving measures, the respective firmware has to be integrated into the whole concept. The integration usually follows three basic concepts, whereas hybrid solutions are also possible.

The first one is that the management layer is able to cut off the power supply at any time, without interfering with the firmware in any negative way. In addition to this it must be
ensured that the firmware resumes normal operation when the power supply is recovered (cp. Figure 4.14).

Contrary to simple devices, more complex peripheral devices cannot be switched off without being notified beforehand. Because of this aspect, virtually every modern bus communication standard provides a power-down mechanism for more complex devices.

That power down mechanism is typically implemented by a special command which can be sent to the respective device to shut it down properly. Optionally, the shut down procedure involves a saving of the device’s last state.
After the firmware has successfully shut down, the device remains in a deep sleep state, in which the management layer is able to send a wake up command in order to resume the device. If the last state has been saved during the shut down procedure, the last working state is restored. Depending on the respective bus system, the state restoration process can be done either by the firmware or by the management layer. Then the device resumes normal operation. The suspend and wakeup procedure of more complex peripheral devices is illustrated by Figure 4.15.

Figure 4.15: State of the Art Technology

A third method which is often combined with the previous ones, is the implementation of an energy management scheme on the peripheral device itself. As illustrated in Figure 4.16, the device checks cyclicly if its full services are needed. After a certain time of inactivity, it switches into a respective idle state. If the inactivity continues, the next idle state with an increased energy saving impact is entered. If the device detects that its full attention is needed, it switches back to normal operation state.

Some parts of the firmware layer are also microcontrollers which are supporting the management layer to implement the energy saving measures. A good example for this kind of device, is the implementation of the voltage rail example of Section 4.4.1. To perform the switch-offs, a microcontroller with a connection to the corresponding x86 Operating System is needed.

Following, selected energy saving concepts of typical x86 subsystems are discussed.
4.5.1 USB

The Universal Serial Bus (USB) has been introduced in 1996. Its acronym was re-interpreted as “Useless Serial Bus” because no appliances were available for a long time. Nowadays the situation has changed completely and the USB has become one of the most used bus systems for peripheral devices.

The USB standard defines a bus-concept, consisting of one master and many slaves, whereas only the master can start a communication to the slaves. To get the respective information from the slaves, the master (in most cases the x86 Operating System) has to poll them regularly.

It is very common now, to make heavy use of the USB standard in embedded x86 areas. USB is replacing former connection standards like ISA or more complex communication standards like PCI or PCI-E. As most of the connected peripheral components in embedded x86 solutions are developed by the appliance makers, there is great potential to save energy by optimizing these USB peripheral devices.

According to the USB 2.0 specification [8] the USB standard supports the following energy saving techniques:

- Immediate bus power cut-off
- Auto-suspend
- Selective suspend by the x86 Operating System

The immediate bus power cut-off method is not mentioned as an official power saving method. However, the USB specification demands that all devices be removable at any time. Therefore, every USB hardware device has to ensure that it can be removed at any time without being damaged. Using this aspect of the USB specification, a mechanism shown in Figure 4.14 can be implemented in order to save energy. As there are devices like storage media devices, which are very sensitive to power interruption, USB supports native energy saving mechanisms.

One native energy saving technology of the USB standard is the mandatory device auto-suspend, as it is required by the specification. Device auto-suspend means that the firmware of every USB device has to monitor the bus for telegrams. If the bus is idle for a time greater than 3.0ms, the device should enter a low power mode. The transition phase from the normal mode to the low power mode should not exceed 7ms.

---

The ISA bus can also be found in modern x86 systems. To reduce the interconnection complexity, the ISA was reconstructed to save connection wires by address/data multiplexing. The multiplexed ISA bus is called “Low Pin Count” (LPC) bus and can be accessed using the same API’s as they were used for ISA. Consequently, it is very easy to get old ISA hardware run on modern x86 systems.
As the USB bus is still powered, the device can be awoken by polling it for certain information. Figure 4.17 illustrates the auto-suspend mode, in context to the other operational modes.
of a USB device. Sometimes it is required to keep all devices up, although no USB bus communication takes place. In this case, the Operating System has to ensure that the respective USB bus is provided with periodically “keep-alive” packets.

The periodic polling activities of the USB bus master have a big impact on the whole x86 power saving mechanism. As long the respective device drivers indicate a poll to their devices, the whole system is not able to enter certain energy-saving states. For example, the entering of the processor energy saving states C3/C4 requires that no bus master activity is taking place [20].

Therefore a mechanism has been introduced, which is capable of suspending individual or all devices on a USB bus. This mechanism is called “selective suspend” and enables the operating system to suspend the devices by sending a special device suspend IO Request Packet (IRP) to the respective device. By sending a corresponding wake up IRP, the device can be awoken again [17] [38].

Although the energy saving contribution of the USB energy saving measures to the overall saving is very high, many USB devices have not implemented them, or provide only scant support [50].

### 4.5.2 Storage Device Power Management

In contrary to the first hard-disk systems like MFM or RLL consisted only of the media and an amplifier. The whole intelligence was provided by the controller card.

The first hard-disk systems like MFM or RLL consisted only of the media and an amplifier. The whole intelligence was provided by the controller card.
Figure 4.18 presents the power breakdown of a notebook which has been measured by the Intel Corporation in the year 2003. Although the energy saving technology on storage devices was refined the last 7 years and new technologies like the Solid State Disk (SSD) were introduced, it is clear to see that it is important to integrate the storage media devices into the system wide energy saving concept.

Most storage devices use the Serial ATA (SATA) bus which connects them physically to the x86 host system. The x86 host system communicates with the storage system by using ATA/ATAPI commands which are transmitted over the SATA connection. Therefore there are two specifications when it comes to energy saving on modern storage systems:

- The SATA specification which is developed by the Serial ATA International Organization (SATA-IO) [45]
- The ATA8-ACS Command Set specification which is maintained by the INCITS T13 standards organization

The SATA specification is responsible for the energy saving capabilities of the physical SATA interface (PHY) which defines three basic energy saving states [37]:

- PHY Ready (PHYRDY) - the SATA PHY is ready to send/receive data
- Partial - the PHY is in a reduced power mode; exit time can be up to 10 microseconds
- Slumber - the PHY is in a reduced power mode (lower power than Partial mode); exit time up to 10 milliseconds

The sleep modes are entered in case of inactivity and can be initiated either by the host or the device. As the number of deactivated regions of the SATA interface increases with the suspend level, the recovery time increases, too. The SATA PHY is returned back into the PHYRDY mode by receiving a wake up sequence which can be sent by the host or by the device.

Besides the energy saving measures of the physical interface, the microcontroller on the storage device can implement an own energy saving scheme. The basic structure of such a scheme is defined by the ATA8-ACS specification which involves measures like the spin down of the disks in a hard drive.

Within the standard, four modes of power consumption are defined [37]:

- Active – The device is fully powered up and ready to send/receive data.
- Idle – The device is capable of responding to commands but the device may take longer to complete commands than when in the Active mode. Power consumption of the device in this state is lower than that of Active mode. If a hard drive is present, it is spun up.
• Standby – The device is capable of responding to commands but the device may take longer (up to 30 seconds) to complete commands compared to the Idle mode. Power consumption is reduced from that of Idle mode. If a hard drive is present, it is spun down.

• Sleep – This is the lowest power mode. The device interface is usually inactive and, if a hard drive is present, the drive is spun down. The device will exit the Sleep mode only after receiving a reset. Wake up time can be as long as 30 seconds.

4.6 Management Layer

The management layer sits on the top of the x86 energy management stack, takes full control over all energy saving measures which are applied during the runtime of the system. Basically there are two implementation strategies for the energy management layer:

• A hardware based approach which is called “Advanced Power Management” (APM)

• An Operating System based approach which is called “Advanced Configuration and Power Interface” (ACPI)

Because both approaches try to provide the optimal computing power / energy consumption ratio independently from the running applications, they are also called “generic energy management”.

To accomplish this, every standard defines global energy saving states, in which certain areas of the system are disabled or active. Switching between the global energy saving states is initiated by events.

4.6.1 Advanced Power Management (APM)

The APM hardware based approach was the first standard for an energy saving management layer. APM has been specified in the early 1990’s and its main philosophy was to create an energy management scheme which is completely transparent to the respective Operating System. Transparent means that the Operating System does not know what the energy management does.

This aspect was very important at this time, because the most important x86 operating systems relied on the Disk Operating System (DOS) or on the mature CP/M. As these systems do not
provide a hardware abstraction layer between the applications and the hardware, it was not possible to realize a full operating system based approach.

The APM standard defines five general energy saving states which are called “S” states. The states and their transition are shown in Figure 4.19 and have the following meaning:

- S1 “full on” – The computer is powered on, and every device is running.
- S2 “APM enabled” – APM is enabled and devices which can be resumed within a short time, might be turned off.
- S3 “APM standby” – Most devices are turned off and the CPU might be halted (cp. C1 state table C.1). A resume is possible by user activity.

Figure 4.19: APM general power states [15]
- S4 “APM suspend” – the whole content of the RAM is saved to the hard-disk and the system is turned off (APM hibernation is only a special mode of the APM suspend state). Therefore the system can quickly restore its last state.
- S5 “off” – The complete system is turned off.

![Diagram of APM operation](image)

**Figure 4.20:** Principle of the hardware based APM

As shown in Figure 4.20, the state changing events are typically sent as interrupts to the system processor. The events can be generated by various hardware components, like microcontrollers or sensors. The interrupt causes the operating system to stop its current task and to execute an interrupt associated BIOS code. After performing the energy saving measures, the BIOS code exits and the Operation System continues its operation.

The severe drawbacks of APM are:

- For every event, a hardware dependent interrupt service routine has to be provided. This routine can only be programmed by the hardware manufacturer and changes with every hardware instance.
- When an energy event occurs, the operating system is stopped for an unpredictable amount of time.
• Errors in the interrupt handler might crash the whole system.
• The overall concept is not flexible for enhancements.

4.6.2 Advanced Configuration and Power Interface (ACPI)

Introduction

In order to resolve the design related problems of APM, a consortium consisting of Intel, Microsoft, Hewlett-Packard, Toshiba and Phoenix has been working on a new x86 energy saving standard. The work on the then new specification was completed in the year 1996 and the result has been called “Advanced Configuration and Power Interface” (ACPI) \[16\]. As the major computer companies were involved in the specification process, the ACPI standard replaced the old APM standard very quickly.

The philosophy of ACPI is that the operating system has to have the full control over the power management\[11\]. As modern operating system designs demand that there is no higher instance, this issue was also very important in the mid 1990s. At that time the new operating system generation was implemented by Microsoft and IBM in the form of Windows NT and OS/2.

The ACPI standard is currently in its fourth (4.0) revision and is the definite state-of-the-art technology up to now.

Criticism on ACPI

Although ACPI resolves the most severe problems of APM, the new standard still faces a lot of criticism. Taking a look on the specification versions, the first aspect which is noticed is the huge number of pages. The first ACPI specification 1.0 had about 400 pages, excluding the book of errata. The current version 4.0, which has been released in June 2009, is about 750 pages. As it is clear to see that a correct implementation is not an easy task for every operating system manufacturer, and because of this, the ACPI standard is often designated as “bloated”.

As a result of the oversized ACPI specification and the fact that Microsoft was one of the co-authors, ACPI was also sued to be an instrument for Microsoft to gain more market dominance (cp. Figure 4.21). As a consequence of this issue, Intel decided to contribute a full featured

\[11\] This technology is known as “Operating System directed Power Management” (OSPM).
ACPI open source software stack for the free software Operating System GNU/Linux which is coordinated on: [http://www.lesswatts.org/](http://www.lesswatts.org/).

In addition to the ACPI stack, Intel also provides the following software for GNU/Linux:

- **PowerTOP**: a tool for diagnosing the efficiency of the power saving which also gives improvement suggestions.
- **ASL\(^{12}\)** validation tool: The Intel reference ASL validator tool is only provided for GNU/Linux.
- **Kernel drivers** which implement features like tick-less IDLE.

It can be assumed that the Microsoft antitrust trials were not the main cause for Intel to support the GNU/Linux platform. That aspect is confirmed by looking at the market share of GNU/Linux in the fields of server systems, which typically run 24 hours a day. As the costs for energy are constantly rising, it is very important to have an effective energy saving mechanism there. Because of these aspects, it is very important for Intel to provide good support for free software.

Additionally, many users of Intel Atom processor based appliances have decided to run GNU/Linux. This decision was made especially because of the high hardware requirements of the current

---

\(^{12}\)The ACPI Source Language is explained within the next sections – The validation tool and its source code can be downloaded here: [http://www.acpica.org/](http://www.acpica.org/) (last visited: April 2010)
Microsoft Windows generation. As Intel wants the Atom platform to be a big success, the Moblin project was founded, which is a complete GNU/Linux distribution managed by Intel. It provides special extensions like a fast “quick-boot” option as well as an Intel Atom optimized energy saving scheme and this is the Intel recommended operating system for Atom processor powered devices.

**Capabilities of ACPI**

Being the successor of the mature APM standard, ACPI provides extended capabilities for power management. Because every peripheral device has its own mechanism to save power, ACPI also provides refined configuration capabilities. Typical for operating system based Power Management (OSPM), both aspects of the ACPI are controlled by the operating system.

In detail, the supported capabilities are:

- A *system-power-management* defining mechanisms to put the computer system and its devices into a sleep mode in a safe way. ACPI also defines the corresponding wake up mechanisms.
- A *device-power-management scheme* ensuring that every peripheral component can be suspended with its own mechanisms. As no two devices are identical, ACPI defines a memory area, in which the device stores its suspend and wakeup routines in a description language called ASL. These descriptions are interpreted by the systems ASL interpreter when it comes to a suspend event. This ASL information is also often called “ACPI tables”.
- Support for the *processor-power-management* functions described in Section 4.4.2; if the operating system detects idleness, it can use these processor extensions in order to save energy.
- Support for a *Device- and processor-performance-management* enabling the user or the OS to maintain a specific computing performance level in order to save energy or decrease the temperature.
- *Configuration and Plug-and-Play* for mainboard related devices to enable the optimal energy saving mechanisms for the respective device (ACPI tables and ASL).
- Provide an *event management* to react on certain events like abrupt temperature rise.

• Provide a battery management – The APM, BIOS related battery management approach has been transferred to the OS. Therefore the OS has to be able to communicate with an intelligent battery management microcontroller. The communication is defined using special ASL instructions which are interpreted by the OS.

• Provide thermal management, to enable the user to control the maximum temperature of the system (cp. Figure 4.9).

• Specify an Embedded Controller (EC) which is a microcontroller placed somewhere on the system mainboard. The communication with the EC can be accomplished by using API’s which are implemented in every ACPI compliant system. The EC can be used to fulfill special tasks, which can vary on every system.

• Specify a SMBus Controller which enables the OS to access the SMBus by using defined API’s. The SMBus is an I2C-like bus providing access to several sensors such as temperature or acceleration sensors.

**ACPI Structure**

![ACPI Structure Diagram](image)

*Figure 4.22: Structure of ACPI [27]*
As illustrated in Figure 4.22, ACPI consists of three OS independent parts and two OS dependent parts.

The OS independent parts are the ACPI registers, which can be used to control the behavior of the OSPM by BIOS settings, the ACPI BIOS and the ACPI-tables which contain the ASL code which is interpreted by the OS in order to use the corresponding hardware correctly. The OS independent parts are passive at OS runtime and only interpreted by the OS dependent parts like the ASL interpreter.

The OS dependent parts are the OSPM system code which is typically part of the system kernel and the ACPI driver with the embedded ASL interpreter.

**ACPI States**

![ACPI States Diagram](image)

As shown in Figure 4.23, the ACPI standard defines four global system states (G0 - G3):

- **The G0 State** – In this state the system is fully functional but the OSPM can suspend certain unused devices as well as apply processor power save functions.
4 Review of State of the Art Technology

- **The G1 State** – This state is also known as sleep state in which the current state of the OS has been saved to the RAM or the Hard Disk. All components which are needed to perform a system wakeup are powered but most system areas are powerless. A wakeup event can be a mouse move or a special network broadcast (WOL = Wake On Lan). This state has sub states which are called S-states ranging from S1 to S5. The higher the S state is, the more time is needed to resume to the full functioning state G0.

- **The G2 State** – This state is also known as “Soft Off S5” state. The system can reach this state by a software power down mechanism which is virtually implemented into every current shutdown process. If the system is in this state, it is only supplied by a 5V voltage rail which ensures that a power-on can be accomplished by pressing a button. During a power up process, the OS must pass through a complete boot procedure in order to reach G0 state.

- **The G3 State** – During this state, the system is completely disconnected from the mains.

- **Legacy State** – This state is a special state and has to be used if a non ACPI capable OS is used. In that case, a BIOS setting activates the legacy state to ensure, that a power up and a power down is possible without the support of an OSPM. Additionally all power management events are handled like APM (e.g. BIOS code is executed to handle thermal events. The BIOS behavior depends on the features which were implemented by the module/mainboard manufacturer).

**G0 Energy Saving Measures**

As described above, the ACPI standard provides energy saving capabilities also in the full featured state G0. In order to do that properly, a energy saving daemon must run. This daemon performs a cyclic measurement on the performance counters (cp. Section 4.4.2) in order to determine the system’s current load status.

If the system load increases because of a program demanding more computing power, the daemon switches to a lower processor C state or increases the frequency. After the program has finished its extensive task, the daemon recognizes that and switches back to a more energy friendly state.

The technology of dynamically adapting voltage and frequency in order to meet the best ratio of required computing power to energy consumption is called “Dynamic Voltage and Frequency Scaling” (DVFS). As the determination of the right frequency/voltage combination

---

14 On GNU/Linux systems, that daemon is called the Power Governor
for a certain situation is not an easy task, the research on new improved algorithms still goes on. To get a first insight on, how these algorithms work, the research results of Frank Bellosa [4] are a very good starting point.

Another good technology which can be utilized during the G0 state, is “tick-less idle”\(^\text{15}\). It is common by some UNIX based operating systems like GNU/Linux, to implement a periodical ticking timer in order to do scheduling or load balancing. As this timer event occurs regularly even when the CPU is idle, the CPU is also forced to leave an energy saving C mode every time there is a timer event. To resolve this issue, lesswatts.org enhanced the GNU/Linux kernel so as not to do regularly timer ticks when being in the idle state.

Especially the “tick-less idle” measure serves as a very good example of the efforts which have to be accomplished by the OSPM during the G0 state:

- The measurement task must not interfere with the measurement – a very complicated task because the system has to measure itself.
- If a certain processor C state is reached, the system has to be designed to maintain this state as long no application needs resources – as the operating system can be seen as a big application too, this is also not easy to do.

• All actions triggered by the OSPM are afflicted with an unpredictable dead time. Therefore it is not possible to exploit very short idle times by the OSPM.

4.6.3 Future Development of the Energy Saving Standards

To resolve the problems of the ACPI, the processor manufacturers are performing heavy research work. As Intel is one of the biggest market players, most of the innovation work is done there. Similar to the Atom technology, there is only very scant information available on what is currently done by the Intel Research Labs.

But despite of all confidentiality, first rumors are heard since late 2007 [59], indicate that Intel is planning to move parts of the energy saving activities back from the OSPM to the hardware\[^{16}\]. According to some information from last year’s Intel Developer Forum [30], the technology enables the semiconductors of the system to do a quick decision on which areas of the silicon are currently used and which not.

The unused areas are quickly suspended by the hardware and do a so called “micro-nap”. As these napping periods are very short, the OSPM would not be able to handle them but the impact to the overall energy saving is great, as the number of micro naps is constantly summing up. Intel plans to introduce the “micro-nap” technology with the Atom SoC solution “Moorestown” which is planned to be released in fall 2010.

According to an “Intel News Fact Sheet” [12] important areas for future development are also:

• **Intelligent Control** enabling the device to determine, if it is really needed or not.

• **Super Capacitors** which can be used to bridge times of high energy demand. This technology can be used to scale down the power supply of the appliance in order to get it much more efficient.

• **Energy Harvesting** is not only a very interesting topic in the automation world – the x86 peripheral components can also benefit from this technology.

• **Low-Power Network Agent** is listening to the network traffic when the rest of the system is sleeping. In case of important telegrams, the hardware based agent wakes up the system.

\[^{16}\text{Intel calls the whole concept “Platform Power Management”}\]
4.7 Summary

This chapter had the aim to introduce the most relevant energy saving techniques which are used in current x86 computer systems. Because of the amount of information which is available it is a difficult task to give a comprehensive overview.

Nevertheless, it is possible to reduce all reviewed as well as all unreviewed technologies into the following basic approaches:

- Unused components shall be switched off – this can be applied to devices as well as to areas inside the semiconductors
- According to the information which was introduced in Section 3.4 the energy consumption can be reduced by lowering:
  - the voltage
  - the frequency
- Optimizing old structures, including the removal of legacy components (think of Intel Atom)
- Enforcement of a system-wide energy management scheme by the definition of states in conjunction with intelligent mechanisms to perform the transition between them.

The discussed state-of-the-art technologies, and their weaknesses, will be used in Chapter 6 for the introduction of a new approach which will attempt to build a bridge between the x86 Personal Computer energy saving schemes and the requirements for the embedded x86 world.
5 Measurement Setup and Configurations

5.1 Introduction

On Completion of the literature research work and associated learning, it was decided to explore the Kontron modules to see if they are capable of doing ACPI energy saving. It was also decided to evaluate the respective operating system APIs. The Microsoft Windows operating system was chosen as basis of the first evaluations as this is the most common operating system.

An important consideration was the organization of a measuring setup that would be capable of verifying the energy saving impact of defined technologies and configurations.

The evaluation process has not yet completed but results for the work to date will be presented in this thesis, and a review of selected peripheral components for further planned work will be defined. As Kontron is the major project partner, the research work focuses more on those modules.

5.2 Objective

This chapter explains the configuration and the respective constraints of two measurement setups. First of all a microcontroller-based setup was created, so as to get an overview assessment of the behavior of the energy management scheme for Microsoft Windows XP. As this scheme tries to optimise the behavior in a generic fashion, it can be referred to as “generic power management”.

In order to make longer-term measurements, possible a second LabVIEW based setup was created.
5 Measurement Setup and Configurations

5.3 Devices Under Test

5.3.1 Kontron C690 AMD Powered Module

The Kontron C690 module was the first Device Under Test (DUT) which has been reviewed. See Figure 5.1. It has special capabilities as an on-board 8-Bit Controller which can be used for several tests. A big benefit was the access to all schematics, kindly granted by Kontron Embedded Modules GmbH. This enabled the measurement of special voltages, such as processor-core voltage and made new types of special measurements possible.

![C690 - Device under test](image)

Figure 5.1: C690 - Device under test

The C690 module has the following features:

- CPU: AMD Turion 64 X2 TL-52 with 1,6 GHz dual core, socket CPU
- Graphics controller: Integrated ATI Rade 3D graphics core X1200, shared memory up to 1024 MB
- RAM: Up to 4 GB DDR2 (667/800) dual channel SODIMM memory
- Power consumption: typ. idle 12,5W @ AMD Turion 64 X2 TL-52
- Dimensions (H x W): 125 mm x 95 mm
5 Measurement Setup and Configurations

5.4 Pilot Survey - A Microcontroller based Setup

After the beginning of this research project, it was decided to review the technologies which were introduced in Chapter 4. The focus of the first efforts was to understand the relationship between the Microsoft Windows XP energy management APIs and the corresponding hardware. These APIs are capable of triggering voltage and frequency scaling as well as initiating the suspension of certain hardware regions of the system.

5.4.1 Measurement Setup

In order to get a good first insight on the API and hardware relations it was decided to take the Kontron C690 as a base for the key measurements. This module has some special features which can be utilized for the required measuring. It provides an on-board MCU (Micro Controller Unit), a 8-Bit 8051 from Silicon Laboratories. The default modules are not equipped with this particular MCU whose purpose is the realisation of special OEM customer features.

![Capabilities of the Silicon Laboratories MCU](image)

**Figure 5.2:** Capabilities of the Silicon Laboratories MCU [29]
The general capabilities of the MCU are shown in Figure 5.2. According to the Silicon Laboratories manual for the C8051F340 it provides: GPIOs, several state machines for protocols like RS232 or USB, and finally Analogue/Digital Converters (A/D Converter). Referring to the data-sheet [29] the A/D converters can perform at a sampling rate of 200 kbps (kilo samples per second).

The on-board MCU itself is connected to several bus systems like I2C enabling it to be used for several tasks such as debugging. Also connected to the MCU is the amplification circuit transferring the voltage drop of the 12V rail shunt resistor to a measurable value. This voltage drop represents the amount of current which is delivered by the 12V power supply\(^1\). To measure that current, one of the A/D converters of the MCU is used. The whole setup is illustrated in Figure 5.3.

To estimate if the sampling rate of 200kps is sufficient for the desired measurements, a short literature research was carried out. This research pointed out that the IBM Research Division made a similar measurement some years ago [42]. The assumption of the IBM Research Division was that the \( f^{-1} \) of the signals on the 12V power rail ranges from \( 10^1 \) s to \( 10^{-2} \) s.

As this assumption has not been rejected at the end of the paper it was decided to adopt it. Accordingly, the sampling rate of 200kps was accepted as satisfactory.

---

\(^1\) On x86 embedded platforms, the 12V rail is usually the only provided voltage.
5 Measurement Setup and Configurations

5.4.2 Created Software

The created software for this project was split up to form a microcontroller part and a PC part for data collection.

Microcontroller Software

The software running on the microcontroller converts the amplified shunt resistor voltage to its digital representation. Additionally, the software provides a mechanism to transmit the measured values through the USB interface to the data collection node (ref. Figure 5.4).

![Figure 5.4: Principle of the USB communication](image)

To reduce implementation effort on microcontroller and host side, it was decided to use the USB “Human Interface Device Class” (HID) for the transfer. Because of this decision it was possible to shorten the implementation time again as Silicon Laboratories provided a sample application for the microcontroller side and a Microsoft Windows driver class for the host side.

---

2Sadly an implementation error in the Silicon Laboratories glue class complicated the development process. This class can be used to build own applications which make use of the USB features of the respective MCU’s. Fortunately, the bugs were found and the patch, which is listened in Appendix D, was sent to Silicon Laboratories.
5 Measurement Setup and Configurations

Data Collector Node Software

The second software part which was developed is concerned with the various requests, and it processes the data measured by the microcontroller. As introduced in the previous section, the HID class is used for USB Communication. As this class was created for so-called Human-Interfaces like mice and keyboards, it is available on virtually every operating system.

Figure 5.5: Tasks of the data collector node

Once started, the data-collection application requests the measured data from the microcontroller and saves it into a file. Because the microcontroller lacks a RTC, the gathered data cannot be time-stamped when measured. Consequently the timestamp is generated on the data-collection-node. This timestamp however, is necessarily shifted and therefore measurements made with this setup are not completely accurate. The measurement principle is shown in Figure 5.5.

After each measurement the data collector had to be stopped because of some undefined communication problems. After some debug sessions it was suggested that these communication errors have their cause in performance related problems on the 8-Bit MCU.

5.4.3 Performed Measurements

The first measurements had the goal to manipulate the frequency registers and processor-core voltage-registers of the AMD CPU and to measure the impact using the microcontroller.
These first measurements were not time critical and had the purpose to check the sample rate assumptions made in Section 5.4.1.

One aim of the first measurements was to set values for the frequency and the processor-core voltage which were not approved by the CPU manufacturer (AMD). It was crucial to find out why only some distinct frequency/core-voltage combinations are allowed by the CPU manufacturer.

In order to manipulate the register values from Microsoft Windows XP, running on the DUT, the CrystalCPU ID tool was used. The CrystalCPU ID program is available as binary code as well as source code and it can be downloaded from: [http://crystalmark.info/download/index-e.html](http://crystalmark.info/download/index-e.html). Using the CrystalCPU ID toolset had the advantage that no dedicated Microsoft Windows XP driver had to be implemented.

The results of the test were:

- Wrong combinations of frequency and processor core voltage lead to stability problems
- Only designated frequency/core voltage combinations are stable on all processors from the respective processor manufacturers
- Although a sample rate of 200kbs was expected, a measurement duration of 30 minutes only resulted in 3172 samples.

### 5.4.4 Limitations and Conclusions

Following the first measurements it was possible to draw an early conclusion. Taking the first assumptions of Section 5.4.1 into account, the 3172 samples within 30 minutes do not meet the Nyquist-Shannon sampling theorem. Reviewing all components, the 8-bit MCU turned out to be inapt to deliver the sampled values through its communication state machines.

As a consequence the 8-Bit MCU cannot be used to analyse the frequency scaling behavior of the operating system. For that task and for longer-term measurements, another measurement setup was created described in Section 5.5.

Concerning the future use of microcontrollers within the research, a second conclusion was drawn: if the MCU is to be used to transfer data using more complex protocols, like USB or Ethernet, it has to be more powerful.

---

3 That topic was part of the work of Bin Lin et al. In [31] a new method is proposed, which makes it possible to use the prohibited areas for energy saving, too. For getting this done, a CPU profiling technique is used to find the allowed values for each individual CPU. The profiling process must be repeated on every new CPU.
5.5 LabVIEW based Setup

The second measuring setup had aimed to perform long-term measurements with the capability of monitoring a wide range of voltages. The measurements of the current intensity are performed by measuring a voltage which is directly proportional to the current.

Because the long term measurements would generate a great amount of data it was decided to use the well proven National Instruments LabVIEW software suite. LabVIEW performs all measurements, data collection and data analysis. As the data is available in ASCII code, it can be passed into the Matlab software in order to make further analysis possible\(^4\).

5.5.1 Data Acquisition Hardware

LabVIEW works best with original National Instruments (NI) components. Unfortunately the author was required to use a LeCroy Waverunner in combination with LabVIEW. The goal was to use the LeCroy analogue measuring inputs in conjunction with other LabVIEW components to enhance the number of available inputs. This attempt failed because the LeCroy Waverunner, which claims to be LabVIEW compatible, did not properly send out raw real numeric values.

So it was necessary to limit the employed hardware to the original NI hardware. For the first measurements it was decided not to buy any new hardware. Two NI measurement devices were selected for the first approach. As both devices combine their channels for the complete setup there was a need to create an interconnection between them. The interconnection ensures that all measured values, regardless of which device captured them, are aligned to the same timebase. This enabled the setup to accommodate the additional channels, as required.

The selected components are described in the following sections.

NI PXI-1042

The NI PXI-1042 is a general-purpose 8-slot chassis for PXI cards. PXI stands for “PCI eXtensions for Instrumentation”. The used device is equipped with a PXI 6070E measurement card providing the following capabilities \(^{24}\):

\(^{4}\)Refering the PAPI project which is doing the same but focus the analysis of the Performance Counters \(^{19}\). The PAPI project is located here: [http://icl.cs.utk.edu/papi/](http://icl.cs.utk.edu/papi/)
5 Measurement Setup and Configurations

Figure 5.6: NI PXI-1024

- 16 analogue inputs
- 12 bit input resolution
- 1.25 Ms/s sampling rate
- 8 digital input/outputs (IO)
- input range from ± 0.05V to ± 10V
- 2 analogue outputs
- analog/digital triggers

NI PCI-6115 PCI Measurement Card

Figure 5.7: NI PCI-6115 measurement card [23]
This PCI measurement card can be used with a normal PC.

- 4 analogue inputs
- 12 bit input resolution
- 12 MS/s sampling rate
- input range from ±0.2V to ±42V
- 4 analogue outputs
- 8 digital IO
- analog/digital triggers

**Connection Blocks**

To connect the analogue signals to the respective measurement units, so-called connection blocks are needed. The proposed setup uses the following connection blocks:

- CB-68LP
- BNC 2110

**5.5.2 Current Measurement**

Because power is a product of voltage and current, accurate measurement of the current intensity is demanded. Fortunately the work of Johann Bretzendorfer [6], which describes different measurement approaches, helped to speed up the decision process.

Four measurement methods were compared:

- discrete linear-hall-effect-based sensor A1321EUA-T manufactured by Allegro
- full integrated linear-hall-effect-based sensor ACS712 manufactured by Allegro
- shunt resistor
- using the DC resistance of the inductors of switch mode power supplies
The later method is not usable. The other measurement methods were reviewed. The results are presented in table 5.1.

<table>
<thead>
<tr>
<th></th>
<th>A1321EUA-T</th>
<th>ACS712</th>
<th>shunt resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>el. complexity</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>calibration efforts</td>
<td>5</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>positioning</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>measuring error</td>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>required space</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>summary</td>
<td>27</td>
<td>14</td>
<td>14</td>
</tr>
</tbody>
</table>

**Table 5.1:** Decision matrix - 1 equals best, 6 equals worst

During the review process aspects like financial cost, electrical complexity, calibration efforts, positioning, measuring error, and required space on the PCB were compared. The aspect of electrical complexity includes problems arising if the measurement device is integrated in the whole circuit.

**Figure 5.8:** Hall-effect based sensors in several mounting positions
As the discrete linear-hall-effect based sensor A1321EUA-T needs a special position to operate correctly, the aspect of its positioning was important. This issue was confirmed by some tests (ref. Figure 5.8). But it cannot be guaranteed, that the sensor always is mounted in its most optimal position. Additionally, it emerged distortions in switch-mode power supplies can heavily impact on the results.

Hence, towards an integrated design, the second hall-effect based sensor ACS712 did not show such a dependency. Therefore it has been decided to abandon the usage of discrete hall-effect-based sensors. The two remaining measurement methods were almost identical in their parameters.

Furthermore, it was important to consider that other modules should also be measured. As various boards have unknown current consumptions, the impact of the shunt resistor’s voltage drop can also cause unforeseen effects.

Taking all this into account it was decided to use the integrated hall-effect-based sensor ACS712 for the measurements. Another advantage of the integrated hall-effect-based sensor ACS712 is that it does not need any amplifying circuits and is therefore ready to use.

The chosen integrated hall-effect-based current sensor has a sensitivity of 66mV/Gauss at a maximum current of 30A. The proportional voltage can be directly passed to an A/D converter which has to process voltages from 0 to 4095mV.

The maximum measurement error is calculated at the sensitivity error ($\pm 3\%$), the linearity error ($\pm 1.5\%$) and the symmetry error ($\pm 2\%$). Consequently the overall measurement error is $\pm 6.5\%$ [22].

### 5.5.3 COM Express Measurement Adapter

Some “computer on module” PCBs provide an opportunity to measure the current consumed by the module, e.g. by providing a shunt resistor. Unfortunately such benefits can only be used for ones own measurements, if the schematics are known and modifications on the module are allowed.

In most practical cases this is not possible, because details of other modules are unknown. Therefore, an adapter 5 meeting the COM Express standards 6 has been developed, as shown in Figure 5.9.

5The schematics are provided in the Appendix E
6To accomplish this, the “COM Express Carrier Design Guide” [39] was very helpful.
5 Measurement Setup and Configurations

**Figure 5.9:** COM Express measurement adapter during calibration

This adapter uses integrated silicon hall-effect-based sensors (Type: ACS712) to measure the currents on the 12V and the 5V rail. Figure 5.10 illustrates a typical application of the chosen integrated hall-effect based sensors where the measured current is measured through the integrated circuit.

![Typical application of the ACS712](image)

**Figure 5.10:** Typical application of the ACS712

The 12V rail is the main power supply for the module and is provided by the backplane. The 5V rail is also provided by the backplane and is optional. If the 5V rail is available, the x86 module provides some “wake-on” capabilities during a G-state that is higher than 0.
Therefore, the 5V rail does not have to deliver much power. If it is decided to use the x86 module “exception-less” inside the G0, only the 12V rail has to be provided. An illustration of this ratio is shown in Figure 5.11.

![Figure 5.11: Voltage rails in context to the adapter](image)

To avoid measurement errors, the hall sensors on the adapter can be supplied with external voltage. As the power consumption of the hall sensors can be considered as extremely low, the 12V rail can also be used for power supply. This configuration can be done by a jumper.

### 5.5.4 The Measurement Setup

As the hardware measurement apparatus is completely based on National Instruments components, the software is developed by using the NI LabVIEW software suite. The programs created using LabVIEW use graphical blocks to represent the real measurement hardware as well as special functions. The resulting program modules are called “Virtual Instruments” (VI).

For the measurement on the x86 modules it was necessary to measure several voltages, such as the CPU-core-voltage or the power supply voltages. The measured currents are represented by special voltages which are proportional to the current.

To get as many analog channels as possible the measurement cards described in Section 5.5.1 were networked by using normal TCP/IP based components. For the first measurements it was sufficient to join the PXI-1042 (16 analog inputs) and the NI PCI-6115 PCI (4 analog inputs) together. The PCI measurement card is needed because its 4 channels provide a measuring capability up to 42V whereas the PXI channels only support up to 10V, which is too low for some signals.
To simplify the setup, the PCI-6115 card was mounted into the master computer which is also collecting the data of the PXI over the network (ref. Figure: 5.12). As stated in section 5.5.1 the PXI device is a specialized measuring computer which also can be used as a standalone device (equipped with Microsoft Windows XP Embedded OS). To enable the PXI for real-time network measurement, a special NI real-time OS must be booted using an USB stick.

**Figure 5.12:** Interconnection of the used LabVIEW devices

For the measurement process three VIs have been created. The first one is the “host VI” running on the master computer.

The second VI is a “Slave VI” which is running on the slave measurement devices. Every slave measurement device needs its own customised slave VI.

The last VI is the “calculation VI” which is able to calculate all needed values from the measured data log-files.
Figure 5.13 illustrates the measurement procedure. More details can be revealed by looking at Appendix F which contains the self-explanatory original source files.

![Flowchart of a measurement procedure](image)

**Figure 5.13:** Flowchart of a measurement procedure

All the VIs are used in conjunction with a special LabVIEW software module called “DataAQu-sition” (DAQ) providing a wizard for easier adding of the available data channels. As the captured datafiles can grow up to hundreds of gigabytes, a special tool named DiaDEM is
used to visualize the measured values. The DiaDEM tool can also compare multiple results by plotting them into the same diagram.

5.5.5 Calibration Procedure

After the LabVIEW measurement devices were successfully networked, it was necessary to check the accuracy of the measurements.

Zero Point Calibration

For the required “zero-point-calibration”, the COM Express measurement adapter was operated with 0 Ampere. The expected result was to get a plot of 0 Ampere in the corresponding DiaDEM chart. The actual result can be seen in Figure 5.14.

![First measurement – Current intensity was 0Amps](image)

Obviously, the result does not meet the expected one. The measurement shows extraneous values which might be the result of distortion phenomena and induced voltage. In addition to this, the whole measurement seems to have a slight offset. The offset was compensated for by determining the average of the first measurement. This value was used as a correction factor for all future measurements. The noise measurement was averaged by replacing every value with the the average of the 100 successor values.
After the configuration of the corresponding LabVIEW modules, and improving the cable shielding, the measurement was repeated.

**Figure 5.15:** Zero-Point calibrated measurement – Current intensity was 0Amps

The result of these efforts is shown in Figure 5.15 showing the improvements where the uncompensated measurement is drawn in red.

**Scale Factor Calibration**

As the voltage produced by the integrated hall-effect based sensor is proportional to the current, it is necessary to determine the corresponding scale factor.

\[
ScalingFactor = \frac{realCurrent}{measuredVoltage}
\]  

(5.1)

Because of component tolerance issues this scale factor has to be found for every attached hall sensor separately. Measurements pointed out that the typical factor ranges from 5 to 6 and after some measurements it was obvious that that factor depended on the direction of the current flow.

The first step in finding the scale factor was to supply the measurement adapter with a known DC current. Then the factor was calculated using the equation 5.1. Figures 5.16 and 5.17 show the final measurements after calibration.
5 Measurement Setup and Configurations

Figure 5.16: Final calibrated 5V rail with ±3A and ±1.5A measured

Figure 5.17: Final calibrated 12V rail with ±3A and ±1.5A measured

Aggregation of all Calibration Results

Final considerations on the accuracy resulted in the following theoretical assumptions:

- offset error: about 0.02A
• noise (amplitude): about 0.01A
• discrepancy between measured current and defined current: about 1% to 3%

After applying all corrections the current can be calculated using the equation 5.2:

\[ I_{\text{real}} = \left( U_{\text{measured}} - \frac{U_{Vcc}}{2} - U_{\text{OffsetCorrection}} \right) \cdot \text{ScalingFactor} \]  \hspace{1cm} (5.2)

### 5.5.6 Remaining Constraints and Possible Improvements

The measurement configurations and setups described above still give room for further improvements.

One aspect is that the signal lines shielding is still not sufficient. For this, the next version of the measurement adapter will need mini BNC sockets instead of the present pin-based connectors.

Up to now \( U_{Vcc} \) has been taken as constant. But it is part of Equation [5.2] and so its value influences the calculated results for \( I_{\text{real}} \). Therefore \( U_{Vcc} \) has to be measured to enhance the quality of the result.

In order to perform zero point calibration, it is necessary to short the measurement pins to ground. In the current version of the PCB that is not possible without soldering.

To get rid of the distortions, an average computation was performed. That calculation results in a slight time shift - which should not have a huge impact but should be kept in mind.

Currently, the master VI saves the values into plain ASCII files. That could be improved by saving the data into the DiaDEM format which has some time-saving advantage when processing it in later steps.

Because of the networking structure, the setup is bound to a maximum sample frequency of 10 kHz. Higher sample frequencies result in time shifts between channels which are located on different devices.
5.5.7 Measurements

In this section some interesting LabVIEW measurements are presented. For the first measurements it was important to have full control over the Device Under Test (DUT). For that reason it was decided to take a Kontron C690 module for the first measurements (for technical details ref. Section 5.3.1).

As this module is the same one as the one that was used for the pilot survey (ref. Section 5.4) it was possible to change the processor frequency and the core voltage manually. Because of the availability of the complete circuit schematics, it was also possible to measure interesting voltages, like the processor-core voltage, directly. This enabled a direct control of all software triggered events. Therefore it was possible to prove that the ACPI energy saving scheme was working on this module.

C690 / MS Windows XP – CPU Ramp Utilization

For this measurement the AMD CPU utilization tool\(^7\) was used. This tool produces a constantly rising CPU utilization whose time is programmable. For the presented measurements a rising time of 120s was chosen. At the end of the rising time the tool tries to maintain a processor load of 100%.

This tool was released by AMD to allow developers to test of the behavior of the processor driver. A graphical interface is also provided which displays the P-state of each processor. It also paints a graph of the current processor load.

<table>
<thead>
<tr>
<th>possible processor configurations</th>
<th>corresponding color in Figure 5.18 and 5.19</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynamic by power management</td>
<td>green</td>
</tr>
<tr>
<td>2100MHz @ 1.125V</td>
<td>blue</td>
</tr>
<tr>
<td>2000MHz @ 1.1V</td>
<td>n.a.</td>
</tr>
<tr>
<td>1800MHz @ 1.075V</td>
<td>n.a.</td>
</tr>
<tr>
<td>1600MHz @ 1.05V</td>
<td>n.a.</td>
</tr>
<tr>
<td>800MHz @ 0.8V</td>
<td>pink</td>
</tr>
</tbody>
</table>

**Table 5.2:** Conditions ramp measurement

---

\(^7\)The Tool is only available through official channels as it has been put under a NDA. Version 2.02 was used for the presented diagrams.
Figure 5.18: Ramp function measurement - Currents on the 12V rail

Figure 5.19: Ramp function measurement - Processor core voltage

Figure 5.18 shows the currents on the 12V rail for the respective measurement conditions. Figure 5.19 shows the respective processor-core voltage. As all measurements were done in
5 Measurement Setup and Configurations

G0 state, the current on the 5V rail shows no significant change. Therefore this has been excluded from the presented plots.

The first aim for the measurements was to prove that ACPI is working on the Kontron boards. The first DUT which was selected was an AMD C690 module running Microsoft Windows XP without an installed processor driver.

By using the AMD PST utility\(^8\), providing a mechanism to choose a custom CPU frequency and a custom processor-core voltage the maximum manufacturer allowed settings were chosen (ref. Table 5.2 – 2100MHz@1.125V). As expected, the applied core voltage never changes in a significant way. The current on the 12V rail rises with the intervals in which the CPU load has been increased by the running AMD CPU utilization tool. As the CPU voltage is never been altered, this behavior can be considered normal for a CPU which is not operated by an OSPM. The rising current can be explained by the rising number of CPU transistors which are utilized when the system load is increased.

The same result has been achieved for the minimum allowed settings of 800MHz@0.8V and running the same utilization ramp using the AMD CPU utilization tool (ref. table 5.2).

For analysis of the OSPM actions the processor driver for the AMD C690 was installed. This enables the operating system to access the power saving capabilities and enables it to perform CPU load dependent DVFS. For the measured results shown in green color (ref. Figure 5.18 and 5.19) the AMD CPU utilization tool was used again to produce a constant rising load ramp.

As Microsoft Windows XP was in idle state before starting the constant load increase, the CPU voltage and frequency start at its lowest rate. There is no possibility to show the current frequency value synchronized with the LabVIEW measurements\(^9\), so the processor-core voltage serves as an indicator for the operating systems actions.

In the middle of the chart it can be seen that the operating system reacts to the rise in system load (detected by querying the performance counters) by increasing the CPU core voltage. This increase is typically performed in combination with an increase of the CPU frequency. The flutter of the core-voltage is caused by the algorithm which is implemented in Microsoft Windows XP. Until the AMD CPU utilization tool is exited, the CPU load is maintained at a high level.

\(^8\)PST Utility is also a NDA secured tool which purpose is to change the CPU frequency and core voltage
\(^9\)If this feature is needed in the future, it is possible to make measurements on the chipset directly. As the specifications of the chipset are NDA secured and such a measurement can cause malfunctions it was decided to skip this for the first time.
The results of the measurements discussed in this section can be summarized as follows:

Kontron modules are capable of performing DVFS using ACPI compliant operating system extensions – to enable the DVFS:

- the BIOS must provide ACPI support
- a corresponding CPU driver has to be installed onto the respective operating system

C690 / MS Windows XP – Benchmark Utilization

The aim of the following measurements are to get a look at the typical current patterns, showing when the system is repeating the same tasks. The following reviews, how the current pattern changes when energy saving measures like DVFS are applied.

This aspect is also interesting in terms of security, where these current patterns enable the so-called “side-channel-attacks”.

<table>
<thead>
<tr>
<th>Possible processor configurations</th>
<th>corresponding color in Figure 5.20 and 5.21</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynamic by power management</td>
<td>red</td>
</tr>
<tr>
<td>2100MHz @ 1.125V</td>
<td>green</td>
</tr>
<tr>
<td>2000MHz @ 1.1V</td>
<td>n.a.</td>
</tr>
<tr>
<td>1800MHz @ 1.075V</td>
<td>n.a.</td>
</tr>
<tr>
<td>1600MHz @ 1.05V</td>
<td>n.a.</td>
</tr>
<tr>
<td>800MHz @ 0.8V</td>
<td>blue</td>
</tr>
</tbody>
</table>

Table 5.3: Conditions benchmark measurement

---

10 On a side-note: If a MS Windows system with an installed Intel driver is booting on an AMD hardware platform, it comes to a blue screen. That is because of the Intel driver – it checks for the competitor CPU and (over-)reacts by throwing the blue screen. This fact was very disturbing while performing the measurements on different modules.

11 This aspect has been analyzed in context to performance counter but not in the context to the distortions which are caused by the energy management which is based on the performance counters. Moreover the noise which is caused by the energy management (immediate load change etc.) can be exploited.
To generate repeating tasks, the freely available benchmark tool PC Mark 2002 was used. For the first measurements the CPU was run at its lowest frequency / voltage configuration of
800MHz@0.8V (Results ref. table 5.3). As the power management was disabled there was no change of the CPU core-voltage. The current shows a characteristic pattern which depends on the task.

For the next run of the benchmark, the CPU was put to its highest possible configuration values of 2100MHz@1.125V (Results ref. table 5.3). Because of the deactivated energy management the CPU core-voltage does not change again. The pattern of the current is similar to the previous pattern. Due to the hand-driven setup, the plots are shifted. But the two measurements prove that the same actions produce the same current patterns. The reason for this lies in the fact that the same tasks utilize the same hardware regions.

The next step was to enable the energy management by installing the processor driver. After the benchmark had been started again, the red plot was measured (ref. Figures 5.20 and 5.21). The shape of the resulting current is similar to the previous ones. The notable difference between them is that the DVFS influenced the patterns to be more distinctive. This can be explained in noting that the operating system is detecting idle times of the CPU. During idleness the corresponding power governor daemon reduces frequency and processor-core voltage which results in an immediate decline of the current.

Therefore the CPU power management has an impact on energy saving as well as on other areas like security or the silence of the system.

### 5.6 Summary

This chapter introduced the state-of-the-art power saving techniques and presented all measurement setups and some of the more interesting results. Based on work which was presented in earlier chapters, a concept for “energy saving” and “embedded needs” will be developed in the following chapters.

---

12On Lenovo Thinkpad notebooks a noise phenomena is known which is caused by the Intel C4 state changes.
6 An x86 Co-Pilot Module

6.1 Introduction

Because of the industrial focus of this project, it was necessary to develop concepts for practical use. These concepts involve two aspects which have been explained in the introductory Chapter 1.3 which are “Energy Saving” and “Embedded Needs”. A concept will be presented here to fulfill both of these aspects.

Hence, this chapter does not only present “Energy Saving” concepts. Moreover it provides a full overview of the whole concept; incorporating solutions for the so-called “Embedded Needs”. These solutions have the aim to form a toolset which will have the potential to make the development of x86 embedded appliances more streamlined.

6.2 Objective

This chapter focuses on the concepts which have been deduced from the material which has been presented so far. This involves energy saving concepts as well as implementation specifications for the embedded toolsets. As the implementation of the various concepts, based on the results specifications, can be realized autonomously, for some tasks at least, the author refers to the solution as the “Co-Pilot for the x86 platform”.

This chapter on the Co-Pilot is divided into two parts. And builds up the theoretical framework. Based on this framework, the following chapter will introduce the implementation considerations.
6.3 Energy Saving

During the literature research process, discussed in Chapter 4, the basic techniques to save energy were identified as:

- lowering the voltage and the frequency
- optimising the semiconductor structures
- switching off unused components
- all measures should be triggered by an intelligent algorithm

Based on conclusions several new concepts were developed, whereas the most promisingly ones are presented here. What all of the presented concepts have in common, is that they can be combined with the existing OSPM-based energy scheme, and do not need any modifications inside the semiconductor structures. They are also applicable to every x86 processor platform, including Intel as well as AMD or VIA.

6.3.1 Advanced Power-Down Concept

The first presented concept is an advanced power-down concept. The central element of this concept is a central low-power microcontroller device, which is placed inside the “chipset zone”.

Figure 6.1: Principle of the advanced power-down concept
The term “chipset zone” simply means that the microcontroller is interacting with the most interesting buses and has the capability of controlling the x86 module. As the module provides all relevant buses to the backplane, the chipset zone is also extended to the backplane.

As illustrated by Figure 6.1, the low-power microcontroller device is connected to selected peripheral bus systems. It also has the capability to power-on the x86 module as well as the ability of determining the present operational state of the x86 module.

![Figure 6.2: Co-Pilot state in context to the conventional ACPI states](image)

When the x86 module is within the ACPI G0 state, the module has full control over all bus connections and therefore control over all associated peripheral components. If the OSPM decides to change into a G-state, which is greater than G1, then the module is completely powered down and the x86 Co-Pilot state is entered. The changing of the G-state can be detected by the low-power microcontroller device by monitoring the respective signals.

Inside the Co-Pilot state, the low-power microcontroller takes over the control and is able to control selected peripheral components in order to:

- collect data
• wait for a special wake-up event
• forge activity (e.g. to let the users of a vending machine think that it is in a full operational mode)

The data collection aspect is very valuable for telemetry appliances, which only need to process the collected data in certain intervals. If the data collection buffer is full, a wakeup signal is generated and the x86 module is woken up. After the data has been processed by the respective application, the OSPM recognises idleness, and powers down to the Co-Pilot state (cp. state diagram in Figure [6.2]).

Although the ACPI defines special wake up scenarios, like wake-on-LAN, there are no customisable wake-up scenarios. As the Co-Pilot is running when the x86 module has been powered down, such wake-up scenarios can be implemented using the Co-Pilot.

The activity forgery feature is interesting to the engineers of vending machines. The greatest problem of a deep sleeping vending machine, is that possible customers could misinterpret the sleep as a malfunction. To avoid this, the x86 Co-Pilot can be used in conjunction with an E-Paper solution (cp. Section [4.4.3]).

Within the G0 state, the x86 module makes normal use of the E-Paper display, whereas the Co-Pilot just forges activity during the Co-Pilot state. When the Co-Pilot detects user activity, it is able to cache the user input and trigger the x86 module wakeup. Figure [6.2] shows the Co-Pilot state, in contrast to the standard ACPI states.

A sample application scenario is shown by Figure [6.3]. If no x86 module presence is needed, the OSPM performs a shutdown to a higher G-state and the low-power microcontroller takes over control the PS/2 keyboard. During the Co-Pilot state, the microcontroller collects the input data and resumes the x86-module if its presence is needed again.

After a successful resume, the microcontroller transmits the collected data to the x86-module by using the USB Human Interface Device class. Instead of the PS/2 keyboard, a telemetry station is also a possible peripheral device.
Figure 6.3: Co-Pilot scenario using a PS/2 keyboard as an example peripheral device
6.3.2 Additional Performance Counters

Another promising concept is the enhancement of the performance counters. The current performance counters, which are provided by the x86 hardware, are only covering the internals of the x86 systems. As it is very interesting for an embedded appliance to have a window to the real world, this concept involves the introduction of performance counters, which can be used to draw a picture of the environment of an embedded appliance.

The gathered information can be used by the OSPM power governor to determine the next energy saving state by taking environmental information into account. Figure 6.4 illustrates the implementation of such an additional performance counter, on a “human load” counter counting the number of people within a certain range.

The events are counted by a microcontroller device and counter overflows are signaled by interrupts to the respective OSPM power governor daemon. As the number of embedded appliances is very high in certain areas, the gathered data of all appliances could also be valuable for other purposes. For instance, a heating control of a public building could make use of the data, which has been measured by several embedded appliances.

These additional performance counters work exactly like the introduced performance counters of Section 4.4.2. But by performing real-world measurements, they are enabling a new dimension for the OSPM power governor.
6.3.3 Solutions for Legacy Hardware

As most of the current embedded x86 appliances make use of old legacy peripheral components, it is also very important to find a solution there. A possible concept for the treatment of the legacy devices could be an approach as was described in Section 4.4.1.

The segmentation of voltage rails in combination with an OSPM controlled low-power microcontroller might be a sufficient solution for mature devices. As most legacy devices do not demand much initialization efforts, a simple power-on and power-off scheme might be applicable in the most cases.

6.4 Embedded Needs

In order to provide extra value for the developers of x86 appliances, it has been decided to implement an embedded toolset. Here, only a subset of the complete toolset is discussed. As the embedded toolset also requires a low-power microcontroller, it can be combined with the energy saving concepts.

6.4.1 Remote Debug Solution

An important goal for the embedded toolset is the development of remote debug features with the following capabilities:

- provide the possibility of remote debugging on an x86 module whose OS does not respond
- all (hardware) events leading up to a crash should be logged
- an access using the widely used Internet protocol TCP/IP should be possible

![Figure 6.5: Principle of the x86 module remote debug](image)
The decision process of choosing the signals of interest proved to be complex. In the end it was decided to limit the first monitoring efforts to the Low Pin Count (LPC) bus \[9\], the System Management Bus (SMBus), and the power supply. Later on it could be advantageous to also consider the PCI or the PCI-E bus.

**LPC-Port-80 Debug**

The reasons for including the mature Low Pin Count Bus into the observations are related to the so-called BIOS Port 0x80, 0x84 Post Codes. After completing a routine at boot time, the BIOS sends a manufacturer dependent code to LPC Port 0x80.

If the boot process sticks at a certain point, the history of the messages could be very informative for tracking down the error. Additional to the boot phase, the operating system occasionally sends such codes to Port 0x80.

Considering that failures will likely occur outside of the laboratory, for field testing the temporary monitoring and recording of this information could be considered to be advantageous.

**SM Bus Access**

The System Management Bus (SM-Bus)\[2\] is a bus system similar to I2C which connects low-level devices such as temperature sensors or battery controllers to the x86 Chipset. If the respective operating system is equipped with drivers for the corresponding chipset it can query the sensors in order to get the desired data. Besides the operating system, the BIOS makes extensive use of the SM-Bus at boot time to get certain things done.

As the devices connected by the SM-Bus provide very interesting low-level data, there is a good chance that failure during the boot process, or during runtime, can be deduced from the data recorded by sniffing the SM-Bus. A good example on this is thermal errors which can be found by looking for the thermal data transmitted over the SM-Bus by thermal sensors. Because the employed SM-Bus hardware is different on every module, the captured data can only be used if there is a deep knowledge of the module used.

6.4.2 Protocol Bridge

Another very valuable part of the embedded toolset is the ability to provide a bridge to different protocols. The peripheral devices, which are used in combination with the embedded x86 modules, often use protocols which are not typical for the x86 world.

![Diagram of protocol bridge]

Figure 6.6: Principle of the protocol bridge

Examples for such protocols are the CAN bus as well as the I2C bus or the LIN bus. As shown in Figure 6.6, the protocol bridge consists of a microcontroller which translates the data stream from a typical x86 bus system to a bus system which is needed for the respective embedded appliance. A typical x86 bus system would be the Low Pin Count (LPC) bus as it is very easy to implement this on the x86 OS side.

6.5 Summary

This chapter described some new energy saving concepts as well as well as giving an introduction to parts of the embedded x86 toolset, which is a key part of the overall research project.

The following chapter provides a short discussion on the implementation considerations.
7 Implementation Considerations

7.1 Introduction

This chapter aims to examine and determine the next steps for the implementation phase. First all basic needs were identified and formulated as distinct “basic services”. These basic services are the basis for the scenarios which were stated in the previous chapter. Additionally, a suitable hardware platform and an appropriate software framework were selected. Finally, it is proposed that the various individual components are be interconnected using a self-defined interconnection standard.

7.2 Objective

This chapter describes all the necessary “basic services” and some of the corresponding APIs. Furthermore, an introduction into the selected software and hardware frameworks is given. A detailed description of the interconnection schemes completes this chapter.

As the implementation process is still going on, this chapter cannot provide a complete project overview. The detailed implementation explanations are part of further research work and associated documents which will be provided during the project.

7.3 Determination of the Essential Needs

At the beginning of the implementation consideration process, the microcontroller-based concepts, which have already been introduced, were analyzed and subdivided into three layers. These Layers are illustrated in Figure 7.1. They represent all the requirements of the concepts discussed in Chapter 6 as well as of some further ideas.
The layers in detail are:

- **Layer 0: Essential subsystems** – This layer provides basic access to the hardware bus systems. As described in the specification chapter, the access to peripheral components is considered to be very important. Therefore, the essential subsystem layer of the microcontroller has to provide at least support for the most used bus systems like: I2C, USB, Ethernet, RS232 and SPI.
  Additionally it has to provide capabilities such as A/D converters and freely configurable GPIO’s. A PS/2 support for implementing the Co-Pilot keyboard scenario is considered to be optional.
  In order to achieve an effective data throughput rate, all hardware protocols should be provided by hardware state machines.

- **Layer 1: Protocol** – This layer provides software-based protocols built up on the hardware protocols provided by the essential subsystems. For instance, these software components perform the communication using Ethernet-based protocols. Another example is a battery management scheme, requiring communication through the RS-232 port us-
7 Implementation Considerations

7.4 Determining a Suitable Hardware and Software Platform

Have identified the essential requirements, it was then necessary to determine suitable hardware to realize the identified needs. For the microcontroller decision process, the experiences with the first 8-bit controller based measurement setup proved to be very helpful (cp. Section 5.4).

As a result of the poor performance results of the 8-bit microcontroller, and because, the CoPilot has to perform more than one task at one time, it was decided to select a controller with a great computing power reserve.

As the controller should also be a low-power device, the number of possible candidates is very small. The following controllers were shortlisted:

- ARM 9 architecture based controller
- ARM 7 architecture based controller
- Motorola Coldfire

As it was important not to be dependent of a certain microcontroller manufacturer, the Motorola Coldfire was dismissed. The main difference between the remaining ARM microcontroller types is that the ARM 9 has a MMU\(^1\) whereas the ARM 7 does not have a MMU.

\(^1\)A Memory Management Unit is a logic embedded inside the semiconductor structures of a processor/microcontroller. By providing a management of the system memory it enables security features like memory protection.
7 Implementation Considerations

Because of the missing MMU unit, it can be assumed that the ARM 7 based microcontrollers are more energy efficient than ARM 9 based microcontrollers\(^2\). Also, as it is intended to shrink down the prototype size following the experimental implementation of all planned features, the energy efficiency of the microcontroller is a very important issue. Consequently, the ARM 7 architecture was preferred over the ARM 9 architecture.

For the implementation of the first prototype, the ARM 7 based NXP LPC-2468 [43] microcontroller was selected. This microcontroller is supports a wide range of hardware communication protocols using built-in state machines and the product is also very well documented.

Despite of the large amount of supported hardware communication protocols, the NXP microcontroller still lacks the support for typical x86 communication standards such as LPC or PCI.

\(^2\)cp. the energy saving impact of the “semiconductor structures removal”, which was exemplary discussed on the Intel Atom processor: Section 4.4.2

\[[43]\]
In order to bridge this gap and to make the prototype more versatile, it was decided to also integrate a FPGA into the concept. As the FPGA provides reconfigurable logic, all required state machines can be implemented. The whole hardware configuration is shown in Figure 7.2.

As it was not possible to implement all of the required features from scratch, it was decided to use a software framework to implement all necessary features. For the ARM 7 platform the following frameworks were shortlisted:

- µC Linux
- OpenRTOS
- FreeRTOS

In the end the µC Linux was picked for the application implementation. µC Linux is a special version of the Linux kernel which can be used where no MMU support is needed. Because of the missing memory protection, a malfunctioning user-space program could crash the whole system. Since this is not usual behavior on embedded appliances, this issue was not considered as a severe drawback.

One advantage of µC Linux is that the kernel-driver structure is the same as it is on the “normal” Linux kernel. As it is also necessary to implement some Linux kernel drivers on the x86 some existing drivers could be adapted and reused.

### 7.5 The Interconnection Scheme

A final aspect of the first implementation consideration was the development of an interconnection scheme, connecting the FPGA, the ARM7 and the x86. As it is shown in Figure 7.2 the FPGA plays a central role as it is required to implement additional state machines which are not provided natively by the ARM 7.

#### 7.5.1 FPGA Interconnection

To allow for the possibility of core reuse inside the FPGA and to get the whole system scalable, it was decided to use a standardized interconnection bus system inside of the FPGA. During the selection process several interconnection bus standards were considered [55]. As
7 Implementation Considerations

A key issue for the current project was the free license issue, and the realisation of an easy implementation, the “Wishbone” standard was taken for the FPGA interconnection.

![Wishbone shared bus topology](image)

**Figure 7.3:** Wishbone shared bus topology [36]

The Wishbone standard [36] defines two types of IP cores: masters and slaves – where the master always initiates the communication. The standard allows a point-to-point connection as well as a shared bus topology (cp. Figure 7.3) in order to allow reuse of existing cores.

![Layers of interconnection](image)

**Figure 7.4:** Layers of interconnection – incorporating the x86 and the ARM7

Figure 7.4 illustrates the bus systems and connection methods which are planned to be used for the interconnection. For each communication partner, Wishbone modules have to be provided.
7.5.2 FPGA to x86 Interconnection

For the first implementation steps it was decided to use Low Pin Count (LPC) \[9\] for connecting the x86 to the FPGA. Nevertheless, it is also possible to use other x86 bus systems like PCI for the connection. But for now the LPC bus provides the following capabilities, which are very advantageous for future steps of the project:

- easy to read open bus specification
- bus is only clocked at 33.3 Mhz
- as the LPC is the ISA bus successor, every operating system API for accessing the ISA bus can be used for LPC
- the ACPI specification defines a microcontroller (Embedded Controller; short: EC) device which assists the OSPM and connects via the LPC bus

![Figure 7.5: ACPI embedded controller registers in the context to the ARM 7](image)

As virtually every ACPI compliant operating system has built-in support for the EC, especially the point above is considered to be very important. In order to establish a communication with an EC, the ACPI specification defines two LPC registers:

- data register for information register on LPC address 0x62
- command register on LPC address 0x66

These registers can be used to exchange data with the EC. As the FPGA is between the ARM 7 and the LPC bus, these registers are generated inside the FPGA. Therefore, they have to be
provided to the internal Wishbone. By incorporating another Wishbone module, the microcontroller is able to receive this information. The relationship of the registers in the context of the ARM 7 is illustrated by Figure 7.5.

### 7.5.3 FPGA to ARM 7 Interconnection

![Diagram of FPGA to ARM 7 interconnections]

**Figure 7.6: FPGA to ARM 7 interconnections**

The final interconnection scheme which had to be developed, was the connection between the FPGA and the ARM7. As it can be seen in Figure 7.6 the ARM 7 is connected by using two bus types.

First of all a SPI based small bandwidth connection was introduced. This connection serves as a notification channel, which can be used to send information from the FPGA to the ARM7.
The second connection type is a high bandwidth shared memory connection, enabling the ARM 7 to quickly access the registers provided by the FPGA. This issue is not relevant when using low-end bus systems, like the LPC, but for the future usage of a PCI, or a PCI-E connection, this is considered as an absolute “must”.

### 7.6 Summary

This chapter had the aim to present the results of the implementation consideration process. It described the hardware decisions as well the selected software frameworks. Additionally a more detailed system interconnection scheme was presented.

---

**Figure 7.7:** Final context of the presented microcontroller based concept

If the ARM 7 concept as discussed is compared to the layers of energy saving, as presented in section [4.3](#), it becomes clear to see that every energy-saving layer can be affected or measured by the proposed embedded x86 Co-Pilot (cp. Figure [7.7](#)).
8 Conclusions and Continuation Work

When asking manufacturers of embedded devices about their strategies in energy saving, there is often a vagueness, or even disinterest, in the replies. Currently, energy saving measurements are often considered as an unnecessary complexity that can give rise to malfunctions and thus they are often ignored.

However, in the near future, all this could change very quickly. Such changes in people’s thinking will be driven by increased energy costs as well by new European Union regulations. A good example of such new energy regulation is demonstrated by the recent ban energy inefficient light bulbs.

The work presented in this thesis, as part of a wider research project, aims to develop pre-engineered which manufacturers can be apply to an existing product appliance.

8.1 Review of the Achievements

The presented research work consists of a review of the most important state-of-the-art energy saving techniques for the x86 based architectures, as well as a proposed implementation concept for an embedded controller management scheme which can be used to implement various energy-saving techniques.

8.2 Continuation Work

The proposed concept seems has been accepted by the wider research group and the concept is in its implementation phase now. Once a first successful demonstrator has been completed the concept will be evaluated in a real product environment.

There are some drawbacks. Unfortunately, not all ARM 7 devices are supported by $\mu$C Linux. Because of this some of the ARM 7 device drivers are currently being rewritten. It is estimated...
that the implementation work for the first demonstrator will be finished during September 
2010 and it is planned to present this work the IEEE Applied Electronics Conference in Pilzen 
[40].

Independently from the implementation work, new concepts and scenarios will be developed 
using the Embedded Controller Management Interface framework as defined within this the-
sis.

### 8.3 Knowledge Gained from this Research Project

After initially spending time investigating the research on the state-of-the-art technologies and 
generic hardware-based concepts on x86 power saving, a richer understanding has been gained 
into the world of embedded x86 systems, as well as the basic technologies which can be used 
to save energy there.

Although, on starting the project work, the author already has a basic comprehensive under-
standing of the discipline, much more depth and appreciation was gained during the review 
phase of the current concepts, which lead to the conception of the new concepts.

A completely new and unknown experience for the author was that of leading a project team, 
which consisted of three research students. During this project, weekly meetings had to be 
arranged to review the progress, synchronize results and to discuss further processes.

Additionally, there was the need to build up a new infrastructure in order to preserve the 
knowledge of the findings. The new infrastructure involves several Wikis as well as a central 
project server for all files. The Wikis’ purpose is to act like a meta layer which refers to the 
distinct files on the project server.
9 Bibliography


[28] Silicon Laboratories. Silicon laboratories an249 - human interface device tutorial. Good starting point when USB support on Silicon Laboratories MCU’s is needed.


A IEEE Applied Electronics Conference
2010 Pilzen

An IEEE conference paper has been submitted to the IEEE Applied Electronics Conference in Pilzen in April 2010, as follows:


This paper describes the research work and shows it is possible to monitor the electrical power consumption of an embedded module in order to perform instant online energy profiling. The concept of implementing a back plane module solution for real-time monitoring and control is presented in the paper.

The full draft paper is provided on the following pages.
A Proposal for a COM Express Adapter for Energy Profiling and Long-Time Failure Analysis

Dipl.-Ing. (FH) Sven Plaga  
University of Applied Sciences Deggendorf  
Deggendorf, Germany  
sven.plaga@fh-deggendorf.de

Prof. Dr.-Ing. Andreas Grzemba  
University of Applied Sciences Deggendorf  
Deggendorf, Germany  
andreas.grzemba@fh-deggendorf.de

Dr. Donal Heffernan  
University of Limerick  
Limerick, Ireland  
donal.heffernan@ul.ie

Abstract—Since it is possible to use x86 based modules in embedded projects, the complexity of the Embedded World has increased dramatically. This paper describes the results of our current research work, which is aimed at helping the OEM customer of embedded modules, as well as the developers, to track down errors. The paper shows it is possible to monitor the electric power consumption of an embedded module in order to perform instant online energy profiling. The proposed concept consists of an adapter which includes a FPGA and an ARM 7 microcontroller to perform the measurements as well as the data collection. This adapter is put between the x86 “Computer on Module” (which is the Device Under Test (DUT)) and the backplane where it has access to all important buses and signals. The collected data can be accessed remotely via Ethernet and is provided to the DUT using the standardized EC Application Programming Interface (API) or standard bus systems, such as the I2C. The presented work focuses on the widely-used PICMG COM Express Standard [1] but the work can be adapted to other module standards, too.

I. INTRODUCTION

For a long time now, it is common to use the x86 technology for embedded solutions, where previously solutions used small and customized microcontroller based designs. The x86 technology can be found in the industrial sector as well as in applications such as information terminals (Kiosk Systems) or vending machines.

A. Embedded x86

In contrast to the traditional approaches the use of the x86 technology offers numerous advantages. One of these is the possibility to reuse programs which were originally written for the Personal Computer, in order to shorten the development time (think of a Kiosk System which is based up on an ordinary internet browser). Because of the dominance of the x86 platform and Microsoft Windows, most such application programs are only available in the form of a precompiled x86 binary.

However, the decision to use the x86 platform can have many advantages, beyond the support of closed source software modules. Existing x86 chipsets offer much more computing power than traditional embedded solutions. Additionally the possibility of re-using well known Personal Computer tool chains, with simplified handling, can be seen as another distinct advantage of the x86 world.

Because of the complexity of the x86 chipset a BIOS and other detailed information from the CPU manufacturers is needed. There are a number of companies on the market which support the appliance developers in developing their x86 hardware based solutions. Such companies are offering small devices which are often referred to as “Computer on Modules”.

Figure 1 illustrates the basic concept behind the x86 modules. A x86 module, which is equipped with standardized connectors is put onto the backplane which in turn provides the counterpart circuits. The x86 module itself is a complete x86 Personal Computer which consists of everything that is needed to Operate, except for the Mass Storage and the Operating System.

The x86 module provides all relevant signals to the backplane. Figure 2 shows the signals for the COM Express specification, that support many common interface standards. As the backplane is designed by the appliance developer, the x86 module specifications attempt to provide as many signals as possible to the backplane.

The x86 module is powered by the backplane through the standardized connector. Normally there is only one Voltage rail to be provided - for the COM Express this is a voltage of 12V.
B. The Problem of Complexity

It's clear that design complexities are increasing for both the x86 developer and the OEM product user. On one side, there is the developer of the x86 Module who is trying to track down errors on his x86 module. On the other side, there is the OEM user of the x86 module whose only intention is to build up a running system on top of the module.

Both sides are facing some of the same problems such as thermal dependent instability, power supply failures, or erroneous peripheral components. All of these error types result in unpredictable crashes or even more subtle problem behavior of the overall system. In our current research investigations [9] on energy saving on embedded x86 modules, we were confronted with the same issues. Such problems led to the realization of a demand for a tool which is capable of monitoring the main power supply in order to be able to create an energy consumption profile of the corresponding x86 module. In order to resolve such problems, the idea of an intelligent adapter was created.

II. THE BASIC CONCEPT

As it can be seen in Figure 3 the proposed adapter consists of module and backplane sockets. The monitored signals are processed by the measurement units, whereas the others are just connected straight through. Taking into account the amount of the signals involved, this can be seen as a very complicated task. Moreover, it is important to maintain the signal quality of high-performance bus systems such as the PCI-E bus.

The "smart" parts of the proposed adapter are formed by a FPGA (Field Programmable Gate Array) and an ARM 7-based processor.

The FPGA serves as a low-level protocol converter and enables the creation of new logical Finite State Machines (FSM) which are not provided by the ARM 7.

The ARM 7 is the central part of the concept. To shorten development time and to make the Unit more predictable, it was decided to run a derivative of Linux on the ARM 7. Because the ARM 7 is lacking MMU (Memory Management Unit) support, the OS needed to be Micro-Linux which is capable of running on such devices. Possible alternatives to Micro-Linux [8] are Operating Systems like FreeRTOS or its commercial derivative OpenRTOS [7].
the gathered data to the flash memory or to a removable media like a USB Stick or a SD-Card.

As it is supposed that there is the need to integrate many cores into the FPGA, it was decided to use an SoC (System on a Chip) bus system for connecting the cores to each other. For the purposes of simplicity it was decided to use the license free open standard Wishbone [4]. As Wishbone uses a Master-Slave principle and is capable of communicating with multiple slaves using addressing, there is also the advantage of core reuse.

III. MONITORED SIGNALS

The decision process of choosing the signals of interest was complex. Finally it was decided to limit the first monitoring efforts to the Low Pin Count (LPC) bus [11], the System Management Bus and the Power Supply. Later on it could be advantageous to take a look on the PCI or PCI-E Bus too, but at this time the efforts are not exceeding the potential benefits.

A. LPC

The reasons for including the mature Low Pin Count Bus into our observations are related to the so-called BIOS Port 0x80, 0x84 Post Codes. After completing a routine at boot time, the BIOS sends a manufacturer dependent code to LPC Port 0x80. If the boot process sticks at a certain point, the history of the messages could be very informative for tracking down the error. Additional to the boot phase, the Operating System occasionally sends such codes to Port 0x80.

Considering that failures will likely occur outside of the laboratory, for field testing the temporary monitoring and recording of this information could be considered as very advantageous.

B. SM-Bus Sniffer

The System Management Bus (SM-Bus) [5] is a bus system similar to I²C which connects low-level devices such as temperature sensors or battery controllers to the x86 Chipset. If the respective Operating System is equipped with drivers for the corresponding chipset it can query the sensors in order to get the desired data. Besides the Operating System, the BIOS\(^1\) makes extensive use of the SM-Bus at boot time to get certain things done.

As the devices which the SM-Bus connects together provide very interesting low-level data there is a good chance that a failure during the boot process or during the runtime can be deduced from the data which were recorded by sniffing the SM-Bus. A good example on this would be thermal errors which can be found by looking for the thermal data which had been transmitted over the SM-Bus by the thermal sensors.

Because of the fact that the employed SM-Bus hardware is different on every module, the captured data can only be exploited if there is a deep knowledge of the module which is used.

\(^1\)Because of security reasons, the access to the SM-Bus at boot time is very limited on some modules. If this feature is demanded by an OEM customer, this can only be realised by the support of the respective module manufacturer.

C. Power Supply

For the realisation of the energy profiling feature it is important to have the ability to monitor the voltage and the intensity of the current. As mentioned before, the COM Express boards have to be supplied with a voltage of 12V, thus this voltage line needs to be monitored. Additional to this voltage, the voltage of 5V also generated on the Backplane has to be observed. This Voltage is generated for peripheral components in order to allow them to run during module sleep states. Figure 5 illustrates the monitored voltages in relation to the proposed measuring adapter.

![Fig. 5. Monitored Voltages](image-url)

In order to measure the current intensity several methods can be used. During our research we compared the shunt resistor based measurement setup (ref. Figure 6) with a concept based on an integrated hall sensor (ref. Figure 7).

![Fig. 6. Shunt Resistor Based Measurement Setup](image-url)

It turned out that the shunt resistor based measurement setup has several disadvantages such as a necessary amplification circuit and the voltage drop which can cause distortions on the DUT.

![Fig. 7. Integrated Hall Effect Based Sensor Based Measurement Setup](image-url)

Therefore an integrated silicon hall effect based sensor (Type: ACS712 [6]) was used. The advantages are:

- much easier handling as only one component has to be placed on the Printed Circuit Board and
- no voltage drop on the main power supply line.
The intensity of the current is linear with the proportional voltage which is connected to the A/D converter of the ARM 7.

IV. ACCESS FROM THE DUT

In order to complete the energy profiling capabilities a connection to the DUT is needed. This connection should be enabled to provide the respective Operating System with the measured values for current intensity and voltage. These values might be useful e.g. while developing new DVFS (Dynamic Voltage and Frequency Scaling) algorithms.

We provide two communication channels to the DUT:

- I2C based Communication
- LPC based Communication

The I2C based communication is possible by using the I2C Interface which is provided by the ARM7 in conjunction with the I2C Interface which is available on many x86 COM Express Modules.

A second communication can be offered by using the Embedded Controller Interface which is based on LPC. Because it is part of the Advanced Configuration and Power Interface (ACPI) [12] [10] standard this interface is implemented on virtually every modern Operating System.

V. PRELIMINARY WORK

Currently a passive COM Express measurement adapter has been designed and tested. This adapter is based on the introduced integrated hall effect based sensors and was used in conjunction with a LabVIEW driven setup. The Adapter is working as expected and none of the connected-through signals were impaired.

VI. CURRENT STATE OF THE WORK

Currently (March 2010) the proposed adapter is under heavy development, as it is required for the on-going research which is targeting new energy saving concepts based on the x86 platform. The first functional PCB is expected during August 2010.

VII. CONCLUSION

As stated above, the proposed concept is valuable to every embedded x86 developer including the designers of x86 modules. Thanks to its flexible concept which includes a FPGA, new Finite State Machines can be created and the adapter might be usable for new research areas beyond energy profiling or the error tracking, in the future.

ACKNOWLEDGMENT

The authors would like to thank their project partner Kontron Embedded Modules GmbH for their excellent support in the course of the research work to date.

REFERENCES

B The A20 Gate

In the early years, the 8086 had the capability to address 1 MB of space which was considered to be enough for all further times. Unfortunately it was only possible to address a space of 64 kb with the available 16 bit CPU registers. Fortunately there were two 16 bit registers which could be used to address the memory - so it was decided to do a memory segmentation. The addressing of a byte is done by <segment>:<offset>.

To get the number of segments there was done by using an easy calculation:

$$16_{\text{Segments}} = \frac{1024_{\text{kbOverallMemory}}}{64_{\text{kbLinearMemory}}}$$ (B.1)

The size of 64 kb for the linear memory within a segment was taken because of the fact that the most .com programs which were made at that time had no need for more memory. As the resulting segments (\(\log_2(16_{\text{Segments}}) = 4_{\text{Bit}}\)) only need 4 bits of the corresponding segment register it was decided to create \(2^{16}\) segments and let them overlap.

The remaining problem was that only 16 Byte of the last segment are associated to real memory. If an offset greater than 16 would be provided the result would cause an unpredictable situation.

To avoid unpredictable situations the 8086 performed a wrap around if a program tried to address memory beyond the 1024 kb boundary. This simple protective behavior was not intended to be exploited by any kind of software. No “normal” software would address the first byte of the memory by writing “give me the byte which can be addressed by 1 MB + 1B” instead of “give me the first byte”.

Unfortunately Microsoft used that “feature” in Microsoft DOS in form of the interrupt \(30_{h}\) which was intended to be a compatibility function to the old mature CP/M.

\[1^*640K\text{ ought to be enough for anybody.”} - Bill Gates, 1981]
As the 80286 was introduced there were no longer only 20 address lines - 24 address lines were introduced in order to support up to 16MB of memory. For the x86 platform the compatibility to old programs is very important. So a solution has to be found for this problem so that the interrupt function \( 30_h \) was able to be used on future versions of the CPU.

The solution to this problem is very easy and is illustrated in Figure B.1. The 20th address line is simply gated through an “AND gate”. That AND gate is connected to the keyboard controller which had another unused GPIO (General Purpose Input Output). Using this mechanism it is possible to simulate mature conditions when they are needed for compatibility reasons.

One big problem is that the condition of the A20 Gate is not readable to any software or the operating system. So every program has to check the condition of the A20 Gate by writing an unique pattern to the first byte and to the 1MB + 1B and comparing them.

The modern processors introduced complex internal caching mechanisms. This issue caused problems because the external A20 Gate was not able to do its job reliably when the modern internal cache structures are not capable of limiting the whole memory to 1MB.

On that certain point the CPU manufacturers had two decision opportunities:

- Cut the support for old programs and clean up the processor core
- Continue by following the x86 dogma which is to stay compatible with old software

Unfortunately they decided to maintain the A20 Gate “feature” and it found its way into the processor die and is alive up to now!
## Overview Processor C-States

<table>
<thead>
<tr>
<th>Mode</th>
<th>Name</th>
<th>Measures</th>
<th>CPU’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>operating state</td>
<td>CPU fully turned on</td>
<td>All Processors</td>
</tr>
<tr>
<td>C1</td>
<td>halt</td>
<td>Stops CPU main internal clocks via software; bus interface unit and APIC are kept running at full speed.</td>
<td>since 486DX4</td>
</tr>
<tr>
<td>C1E</td>
<td>enhanced halt</td>
<td>Stops CPU main internal clocks via software and reduces CPU voltage; bus interface unit and APIC are kept running at full speed.</td>
<td>Socket 775 CPUs</td>
</tr>
<tr>
<td>C1E</td>
<td>–</td>
<td>Stops all CPU internal clocks</td>
<td>Turion 64, 65-nm Athlon X2</td>
</tr>
<tr>
<td>C2</td>
<td>stop grant</td>
<td>Stops CPU main internal clocks via hardware; bus interface unit and APIC are kept running at full speed.</td>
<td>since 486DX4</td>
</tr>
<tr>
<td>C2E</td>
<td>extended stop grant</td>
<td>Stops CPU main internal clocks via hardware and reduces CPU voltage; bus interface unit and APIC are kept running at full speed.</td>
<td>Core 2 Duo and above (Intel only)</td>
</tr>
<tr>
<td>C3</td>
<td>sleep</td>
<td>Stops all CPU internal clocks</td>
<td>Pentium II, Athlon and above, but not on Core 2 Duo E4000 and E6000</td>
</tr>
</tbody>
</table>

*Table C.1: Overview of all Processor C-States [53]*
### Overview Processor C-States

Table C.2: Overview of all Processor C-States

<table>
<thead>
<tr>
<th>Mode</th>
<th>Name</th>
<th>Measures</th>
<th>CPU’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3</td>
<td>deep sleep</td>
<td>Stops all CPU internal and external clocks</td>
<td>Pentium II and above, but not on Core 2 Duo E4000 and E6000; Turion 64</td>
</tr>
<tr>
<td>C3</td>
<td>AltVID</td>
<td>Stops all CPU internal clocks and reduces CPU voltage</td>
<td>AMD Turion 64</td>
</tr>
<tr>
<td>C4</td>
<td>deeper sleep</td>
<td>Reduces CPU voltage</td>
<td>Pentium M and above, but not on Core 2 Duo E4000 and E6000 series; AMD Turion 64</td>
</tr>
<tr>
<td>C4E/C5</td>
<td>enhanced deeper sleep</td>
<td>Reduces CPU voltage even more and turns off the memory cache</td>
<td>Core Solo, Core Duo and 45-nm mobile Core 2 Duo only</td>
</tr>
<tr>
<td>C6</td>
<td>deep power down</td>
<td>Reduces the CPU internal voltage to any value, including 0 V</td>
<td>45-nm mobile Core 2 Duo only</td>
</tr>
</tbody>
</table>
D Microcontroller based Measurement

Patch for Silicon Labs. USB Class

In this section the patch for the Silicon Laboratories Windows USB HID wrapper class is presented. This patch includes corrections for three flaws including a handle-leak patch removing the cyclic crashes.

Unfortunately the handle-leak is still present in current application notes.

Listing D.1: Patch which is correcting some silicon Labs. programming errors

```
diff -u -r orig/HIDDevice.cpp working/HIDDevice.cpp
--- orig/HIDDevice.cpp 2007-04-04 10:15:44.000000000 +0200
+++ working/HIDDevice.cpp 2010-03-22 19:52:25.896287312 +0100
@@ -1,4 +1,8 @@
-/*! HIDDevice.cpp: implementation of the CHIDDevice class.
+/*! File: HIDDevice.cpp:
+// Purpose: implementation of the CHIDDevice class
+// created: by silabs BUT this is the BUGFREE VERSION !!!
+// -> original one has handle leaks!
+/**
+ /////////////////////////////////////////////////////////////////////////////////////////////
@@ -161,7 +165,7 @@
 } }
 } }
-
+ SetupDiDestroyDeviceInfoList(hHidDeviceInfo); // Patch by Plaga -> Correction of the Handle Leak!
 return status;
 } }
```

The patch was created using GNU patch.
if ((m_MaxReportRequest > 0) && (numInputBuffers != DEFAULT_REPORT_INPUT_BUFFERS))
{
  // Ensure that we are setting the input buffers to a valid setting
  if (numInputBuffers > m_MaxReportRequest) m_MaxReportRequest = numInputBuffers;
  if (numInputBuffers > m_MaxReportRequest) {
    m_MaxReportRequest = numInputBuffers;
    HidD_SetNumInputBuffers(hHidDeviceHandle, numInputBuffers);
  }
  else
  {
    // bug made by silicon labs! corrected by:
    // Pull the current settings if default is selected
    // HidD_GetNumInputBuffers(hHidDeviceHandle, (PULONG)& m_MaxReportRequest));
    //Check revised codes below...but still got a warning...slm,
    ULONG ulNumber;
    // Pull the current settings if default is selected
    HidD_GetNumInputBuffers(hHidDeviceHandle, (PULONG)(&m_MaxReportRequest));
    HidD_GetNumInputBuffers(hHidDeviceHandle, &ulNumber);//(PULONG)(&
    m_MaxReportRequest));
    m_MaxReportRequest = (DWORD)ulNumber;
  }

  // Cleanup the prepared data
  HidD_FreePreparsedData(preparsedData);
}

// Increment the device number
return status;
BYTE CHIDDevice::GetReport_Control(BYTE* buffer, DWORD bufferSize)
{
    /* All stanzas with m_InputReportBufferLength were
       m_OutputReportBufferLength before but that is apparently wrong.
    */
    BYTE status = HID_DEVICE_SUCCESS;
    unsigned char reportID = buffer[0];
    if (bufferSize >= m_OutputReportBufferLength)
        if (bufferSize >= m_InputReportBufferLength)
            // Clear out the report buffer, and set the head to the report ID
            memset(buffer, 0, bufferSize);
    if (IsOpened())
        if (!HidD_GetInputReport(m_Handle, buffer, m_OutputReportBufferLength))
            if (!HidD_GetInputReport(m_Handle, buffer, m_InputReportBufferLength))
                status = HID_DEVICE_TRANSFER_FAILED;
}
E  COM Express Measurement Adapter

The COM Express measurement adapter is a small PCB which is put between the x86 module and the backplane. That device was introduced in Chapter 5.5.3. This appendix provides the schematics and the layouts.

Layouts

![COM Express Measurement Adapter Layout](image)

**Figure E.1:** COM Express measurement adapter top view
Figure E.2: COM Express measurement adapter 1st inner layer

Figure E.3: COM Express measurement adapter 2nd inner layer
Figure E.4: COM Express measurement adapter bottom view
Schematics

Figure E.5: COM Express measurement adapter - schematic 1
Figure E.6: COM Express measurement adapter - schematic 2
Figure E.7: COM Express measurement adapter - schematic 3
This appendix contains the graphical listings of the created LabVIEW programs. The LabVIEW software is explained in Chapter 5.

The printed version of this thesis might not be suitable for reading the schematics. As they have been embedded in a vectorized format, they are viewable in the PDF version.

**TCP Slave Program**

This VI is downloaded to the PXI-1042 slave-unit.

![Diagram of the TCP Slave Program](image)

**Figure F.1:** Program running on the TCP slave system
Host Program

This VI is executed on the host. The host consists of a NI PCI-6115 PCI measurement card for measurement purposes and also performs data collection from the slave units.

Figure F.2: Program running on the host system
Calculation Program

This VI calculates all needed values out of the measured data.

Figure F.3: Program used to calculate values out of the measured data
Glossary

A/D Converter
Analogue/Digital Converter.

ACPI
Advanced Configuration and Power Interface

AMI
American Megatrends Incorporated a major provider of BIOS solutions.

APIC
Advanced Programmable Interrupt Controller

APM
Advanced Power Management

ASCII
American Standard Code for Information Interchange

ATA
Advanced Technology Attachment

BIOS
Basic Input Output System – The first program executed on a x86 system.

Cache Miss
When a needed information cannot be found inside the processor caches it has to be obtained from the fixed memory. These events causing poor performance and are called Cache Misses.

CISC
Complex Instruction Set Computer
Glossary

CMOS
Complementary Metal Oxide Semiconductor

Co-Pilot
Is the term introduced for the proposed concept.

COM Express
The Computer On Module standard

DOS
Disk Operating System

DUT
Device Under Test

DVFS
Dynamic Voltage and Frequency Scaling.

ETX
Embedded Technology Extended Standard

FET
Field Effect Transistor

FPGA
Field Programmable Gate Array

free software
Mentioned in the GNU context: “free” as in “free speech” not in “free beer” (Richard Stallman)

GNU
GNU is Not Unix – GNU software generally provided as “free software”

GUI
Graphical User Interface

HID
Human Interface Device – a USB hardware class
Glossary

IA
IA-32, IA-64 or IA: Intel Architecture

IDE
Integrated Drive Electronics – An interface standard for the connection of storage devices.

IO
Input Output

IP
Intellectual Property – referring hardware implementations

IRP
IO Request Packet (a USB standard term)

ISA
Industry Standard Architecture

ISR
Interrupt Service Routine

Kernel
The central component of an operating system

ksps
kilo samples per second

LAN
Local Area Network

Legacy Peripheral
Old mature devices used because of wide their spread – These devices often connected using old legacy protocols.

LLC
Last Level Cache – typically includes L1 and L2 Caches

lock-in
Vendor lock-in also known as proprietary lock-in – a strategy aimed to make the consumer dependent on a vendor.
Glossary

LPC
Low Pin Count

LVDS
Low Voltage Differential Signaling

MCU
Micro Controller Unit (microcontroller)

MMU
Memory Management Unit

NDA
Non Disclosure Agreement

NI
National Instruments – manufacturer of LabVIEW

OEM
Original-Equipment-Manufacturer

OOO
Out of Order Execution

OS
Operating System

OSPM
Operating System directed Power Management

PC
Personal Computer

PCB
Printed Circuit Board

PCI
Peripheral Component Interconnect
Glossary

PEG
PCI Express for Graphics

PHOENIX
Phoenix Technologies Ltd a major provider of BIOS solutions

PICMG
PCI Industrial Computer Manufacturers Group – non-profit standartisation taking care of standards.

Power PC
A RISC architecture created by an Apple-IBM-Motorola alliance known as AIM (1991).

PXI
PCI eXtensions for Instrumentation (a National Instruments term)

RISC
Reduced Instruction Set Computer

SATA
Serial ATA

SMBus
System Management Bus

SoC
System on a Chip

Solid State Disk
Memory consisting of semiconductor based memory elements

SSD
Solid State Disk

TCO
Total Cost of Ownership

TDP
Thermal Design Power
Glossary

USB
Universal Serial Bus

VHDL
Very High Speed Integrated Circuit Hardware Description Language

VI
Virtual Instrument (a National Instruments term)

Wifi
Wireless lan interface based on the IEEE 802.11 standards.

x86
The family of processors originally invented by the Intel Corporation

XDP
Intel’s eXtended Debug Port
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Kontron company logo</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>Principal project goals</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>Samsung coke vending machine with special capabilities</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Demonstrator for the Intel digital signage proof of concept</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Backplane with the corresponding module</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>History of form factors from the Kontron viewpoint</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>A typical ETX module</td>
<td>15</td>
</tr>
<tr>
<td>2.6</td>
<td>The COM Express modules and their different sizes</td>
<td>16</td>
</tr>
<tr>
<td>2.7</td>
<td>The COM Express signals and their connector</td>
<td>17</td>
</tr>
<tr>
<td>2.8</td>
<td>ESMexpress module</td>
<td>17</td>
</tr>
<tr>
<td>2.9</td>
<td>ESMexpress mechanical concept</td>
<td>18</td>
</tr>
<tr>
<td>3.1</td>
<td>Samsung coke vending machine has a problem</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td>CMOS inverter circuit</td>
<td>23</td>
</tr>
<tr>
<td>3.3</td>
<td>Model describing parasitic diodes present in CMOS inverter</td>
<td>24</td>
</tr>
<tr>
<td>3.4</td>
<td>Model of a SIPMOS introducing the internal capacitors of a FET</td>
<td>25</td>
</tr>
<tr>
<td>3.5</td>
<td>Basic components of a typical x86 Intel based system</td>
<td>27</td>
</tr>
<tr>
<td>3.6</td>
<td>Cinebench score vs. power consumption</td>
<td>29</td>
</tr>
<tr>
<td>4.1</td>
<td>Layers of energy saving</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>Hardware layer – Types of technologies</td>
<td>33</td>
</tr>
<tr>
<td>4.3</td>
<td>Segmentation of voltage rails</td>
<td>34</td>
</tr>
<tr>
<td>4.4</td>
<td>Traditional transistor vs. high-k based transistor</td>
<td>35</td>
</tr>
<tr>
<td>4.5</td>
<td>Intel Atom - Processor Power Breakdown</td>
<td>36</td>
</tr>
<tr>
<td>4.6</td>
<td>Conventional IA x86 macro-op decode (simplified)</td>
<td>37</td>
</tr>
<tr>
<td>4.7</td>
<td>Intel Atom decoding engine</td>
<td>38</td>
</tr>
<tr>
<td>4.8</td>
<td>Intel Atom processor states</td>
<td>40</td>
</tr>
<tr>
<td>4.9</td>
<td>ACPI thermal management</td>
<td>41</td>
</tr>
<tr>
<td>4.10</td>
<td>Performance counter basic functionality</td>
<td>42</td>
</tr>
</tbody>
</table>
List of Figures

4.11 Performance counter - ISR based measurement ............................................ 42
4.12 Microcapsules - functional principle of Electrophoretic based displays [18] .... 44
4.13 NEC LCD Ltd. Din-A3 and Din-A4 paper module [52] ................................. 45
4.14 Cut-off power device suspend procedure .................................................... 47
4.15 Complex peripheral device suspend procedure ............................................ 47
4.16 Energy management on the peripheral device .............................................. 48
4.17 USB 2.0 suspend state diagram [8] .............................................................. 50
4.18 Notebook power breakdown [20] ................................................................. 51
4.19 APM general power states [15] ................................................................. 54
4.20 Principle of the hardware based APM ......................................................... 55
4.21 Exhibit State of Iowa vs. Microsoft Corp. [3] ............................................... 57
4.22 Structure of ACPI [27] .............................................................................. 59
4.23 ACPI States [27] ...................................................................................... 60
4.24 Actions of the OSPM power governor ....................................................... 62

5.1 C690 - Device under test ........................................................................ 66
5.2 Capabilities of the Silicon Laboratories MCU [29] ...................................... 67
5.3 Setup of the microcontroller driven measurements ....................................... 68
5.4 Principle of the USB communication .......................................................... 69
5.5 Tasks of the data collector node ................................................................ 70
5.6 NI PXI-1024 .......................................................................................... 73
5.7 NI PCI-6115 measurement card [23] .......................................................... 73
5.8 Hall-effect based sensors in several mounting positions ............................... 75
5.9 COM Express measurement adapter during calibration .............................. 77
5.10 Typical application of the ACS712 [22] .................................................... 77
5.11 Voltage rails in context to the adapter ....................................................... 78
5.12 Interconnection of the used LabVIEW devices ......................................... 79
5.13 Flowchart of a measurement procedure ................................................... 80
5.14 First measurement – Current intensity was 0Amps .................................... 81
5.15 Zero-Point calibrated measurement – Current intensity was 0Amps ......... 82
5.16 Final calibrated 5V rail with ± 3A and ±1.5A measured ............................ 83
5.17 Final calibrated 12V rail with ± 3A and ±1.5A measured ............................ 83
5.18 Ramp function measurement - Currents on the 12V rail ......................... 86
5.19 Ramp function measurement - Processor core voltage ............................. 86
5.20 Benchmark measurement – Currents on the 12V rail .............................. 89
5.21 Benchmark measurement – Processor core voltage .................................. 89

6.1 Principle of the advanced power-down concept ......................................... 92

xxx
List of Figures

6.2 Co-Pilot state in context to the conventional ACPI states ........................................ 93
6.3 Co-Pilot scenario using a PS/2 keyboard as an example peripheral device .................. 95
6.4 Example for a new performance counter – the “human load” counter .......................... 96
6.5 Principle of the x86 module remote debug ................................................................. 97
6.6 Principle of the protocol bridge .................................................................................. 99

7.1 Layers of implementation from a microcontroller point of view .................................. 101
7.2 All components of the x86 embedded controller management interface ..................... 103
7.3 Wishbone shared bus topology [36] ............................................................................ 105
7.4 Layers of interconnection – incorporating the x86 and the ARM7 ............................... 105
7.5 ACPI embedded controller registers in the context to the ARM7 ............................... 106
7.6 FPGA to ARM 7 inteconnections ............................................................................. 107
7.7 Final context of the presented microcontroller based concept .................................... 108

B.1 A20 Gate workaround [7] ........................................................................................... viii

E.1 COM Express measurement adapter top view ............................................................. xiv
E.2 COM Express measurement adapter 1st inner layer ................................................... xv
E.3 COM Express measurement adapter 2nd inner layer ................................................... xv
E.4 COM Express measurement adapter bottom view ...................................................... xvi
E.5 COM Express measurement adapter - schematic 1 ..................................................... xvii
E.6 COM Express measurement adapter - schematic 2 ..................................................... xviii
E.7 COM Express measurement adapter - schematic 3 .................................................... xix

F.1 Program running on the TCP slave system ................................................................. xx
F.2 Program running on the host system .......................................................................... xxi
F.3 Program used to calculate values out of the measured data ....................................... xxii