Date of issue: 03/2008

Energy Management Architecture for Wireless Sensor Networks

Carlos Hernández Gañán

Acknowledgements

This thesis arose in part out of years of research that has been done before I came to Philips Connectivity group. By that time, I have worked with a great number of people whose contribution in assorted ways to the research and the making of the thesis deserved special mention. It is a pleasure to convey my gratitude to them all in my humble acknowledgment.

First of all, I would like to express my gratitude to my supervisor, Thomas Falck, whose expertise, understanding, and patience, added considerably to my graduate experience. I appreciate his vast knowledge and skills in many areas, and his assistance in writing reports. Above all and the most needed, he provided me unflinching encouragement and support in various ways. His truly scientist intuition has made him as a constant oasis of ideas and passions in science, which exceptionally inspire and enrich my growth as a student, a researcher and a scientist want to be.

I am deeply indebted to Herr Gappisch. Without his guidance, support and good nature, I would never have been able to develop this thesis successfully. I benefited greatly from his ideas and insights. His involvement with his originality has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

Some debts are hard to put into words. My good friends Alfredo Carlos López, Miguel Ángel Martínez, Albert Puig and my colleagues Marc Solá, Piotr Kopajczyk, Galiya Hasanova, Marc Sillner, Leszek Seweryn and Phrabat Sarawat all know why their names are here.

My last, but not least gratitude is for my parents, it is difficult to find words to express my gratitude and thanks to both of you. I am so proud to be your son. No word to express my thanks to both of you. To my mother Julia, your love is always in my heart. I would like also thank to my brother for his meaningful supports during the years of my study. To my ‘second’ parents thank you so much for your support throughout the hard times during my studies.

I realize that not all people who contributed either directly or indirectly to my study are mentioned in this page. From the deepest of my heart, I would like to thank all of you…

Resumen

Optimizar el consumo de energía en redes de sensores inalámbricas se ha convertido recientemente en unos de los objetivos de rendimiento más importantes. El sensor inalámbrico, siendo un dispositvo microelectrónico, sólo puede estar equipado con una fuente de energía limitada. En algunos escenarios, el reabastecimiento de recursos energéticos puede ser imposible. Por otra parte, la densidad de energía de las baterías se dobla tan sólo de 5 a 20 años, dependiendo de la química particular y, además, un prologando refinamiento de cualquier tecnología produce un rendimiento decreciente. Esto implica que la gestión de energía sea tan crítica en las redes de sensores futuras como lo es ahora.

Esta tesis aborda el conato de permitir que un sistema subyacente controle la energía de manera inteligente. Con la motivación de gestionar la energía en WSN, proponemos una arquitectura constituida por tres componentes básicos: un monitor de energía, un gestor de energía y un módulo para especificar directrices de funcionamiento. Esta arquitectura permite la prioritización de requisitos del usuario en el sistema, predicción de posibles fallos, degradación grácil del sistema, y adaptación justa frente a la indisponibilidad de recursos limitados. Además, la existencia de facilidades para la gestión de energía permite al usuario dictar el tiempo de vida deseado para la red inalámbrica, una capacidad previamente inalcanzable aunque fundamental para observar cualquier fenómeno.

La arquitectura propuesta se implanta en la plataforma ya existente AquisGrain basada en IEEE 802.15.4 desarrollada por PFL Connectivity Group. Al mismo tiempo, para monitorizar el sistema y el uso de energía de cada componente se utiliza el dispositivo DS2780. Este CI implementa algoritmos intrínsecos para estimar el estado de carga de la batería teniendo en cuenta los principales atributos no lineales de ésta. Diferentes tests prueban la precisión del CI y analizan el error en la estimación del tiempo de vida.

Ulteriormente, profundizamos en el diseño y la base lógica para alcanzar el objetivo de esta tesis, así como el lenguaje de alto nivel usado para describir una política que permita al usuario especificar un conjunto de directrices que sean dinámicamente controladas y satisfechas en tiempo real por el sistema. La descripción de la política a nivel de red puede ser usada por nodos individuales para optimización local y compartida entre otros nodos para ajustar el comportamiento de toda la red. Subsecuentemente, se evalúa la viabilidad de este lenguaje y del diseño, llevando a cabo pruebas mediante el uso de un electrocardiograma digital.

El núcleo del gestor de energía implementa un algoritmo que procesa cambios en el comportamiento de la aplicación basándose en el uso pasado de energía, el estado actual de energía y la política del usuario. La arquitectura invoca este algoritmo cada vez que una nueva política es establecida. Dada una lista de reglas a satisfacer, una predicción del consumo de energía de cada regla y el total de energía disponible en el sistema, el gestor de energía identifica el máximo número de restricciones que se pueden llevar a cabo en justa competitividad. Este enfoque permite encontrar el balance óptimo, satis-

faciendo tantas reglas a la vez como sean posibles.

Finalmente, una evaluación de toda la arquitectura como conjunto es llevada a cabo demostrando que sus capacidades exceden los requerimientos de diseño. Líneas de futura investigación son propuestas para extender esta arquitectura de gestión de energía más allá del nivel de nodo.

Abstract

Optimizing the energy consumption in wireless sensor networks has recently become one of the most important performance objectives. The wireless sensor node, being a microelectronic device, can only be equipped with a limited power source. In some application scenarios, replenishment of power resources might be impossible. The energy density of batteries has only doubled every 5 to 20 years, depending on the particular chemistry, and prolonged refinement of any chemistry yields diminishing returns. This shows that power management will be as critical in future sensor networks as it is now.

This work is an effort to allow the energy to be intelligently arbitrated by the underlying system. Given the motivation for energy management in WSN, we propose an architecture consisting of three basic components: an energy monitor, an energy manager and a specification component. This architecture enables prioritization of user requirements on the system, enabling predictable failure, graceful degradation, and fair accounting in the face of scarce resource availability. Additionally, use of the energy management facilities allows a user to dictate desired network lifetime, a capability previously unachievable yet fundamental for observing certain phenomena.

The proposed architecture is set in the existing IEEE 802.15.4-based platform AquisGrain developed by PFL Connectivity Group. In addition, we select the DS2780 fuel gauge to monitor the system and component resource usage. This IC implements on-chip algorithms to estimate the battery state of charge taking into account the main non-linear attributes of chemical batteries. Different tests demonstrate the accuracy of the fuel gauge and analyze the lifetime estimation error.

Next, we delve into the design and rationale for the focus of this thesis, namely the high-level language used to describe a policy which allows a user to specify a set of directives that are dynamically checked and automatically satisfied at run-time by the system. The expression of network-level policy can be used by individual nodes for local optimization and shared among nodes to adjust the behaviour of the network as a whole. We evaluate the feasibility of this language and our design by performing experiments using an electrocardiogram application.

At the core of our energy manager engine implements an algorithm which changes the application behaviour based on past energy usage, current system energy states, and user policy. This algorithm is invoked each time a new policy is established. Given a list of rules to satisfy, a prediction of each rule's energy usage and the total amount of energy available, the energy manager identifies the maximization of a number of competing fairness constraints. This approach allows us to find the optimal balance, satisfying as many of our rules at once as optimally possible.

Finally, an evaluation of the whole architecture is carried out demonstrating that its capabilities exceed the design requirements. Future research lines are proposed to extend this energy management architecture further beyond node-level.

Executive Summary

HOW PHILIPS RESEARCH CAN IMPROVE AQUISGRAIN SOFTWARE AND HARDWARE PERFORMANCE BY USING THE ENERGY MANAGEMENT ARCHITECTURE

Designing cost-sensitive embedded products requires maximizing a platform's performance while minimizing energy use. With the recent, explosive market growth of mobile embedded devices, low power consumption has become an important design constraint. To effectively meet the energy consumption requirements of embedded systems, programmers need to understand the energy and power consumption of embedded systems as a high-priority monitoring target. Moreover, it is often a prime concern for the user to know the state of charge of the batteries in the platform. At present, the platform developed by Philips Research lacks of an energy manager.

The key to effective energy management is information and knowledge — information on what's happening and the knowledge to do something about it. More specifically, it's understanding where, when, and how much energy is being consumed and having the ability to act. Finding the hidden energy costs can be a source of substantial savings for wireless sensor networks developers if they know where to look.

Hence, we propose an architecture which allows sensornet applications writers to treat energy as a fundamental primitive. Our system will give AquisGrain platform an easy-to-use service that will be up and running with minimum energy consumption. It offers flexibility and an open design to integrate with other applications. The customer will adapt to it quickly and use it eagerly because of its simple, intelligent design and advanced features.

Addressing these challenges will have a number of positive results on AquisGrain's applications. As we document later in this proposal, the amount of energy that drop off the node lifetime will do down, meaning a higher percentage of energy is spent efficiently. This architecture will maximize the profitability of the energy, providing quality of service guarantees by using customer feedback.

Energy consumption can mushroom out of control if you don't have processes in place to manage it and a plan to guide it. Working with our architecture will enable AquisGrain's programmers to improve both control and planning. It also provides with a range of directive so you can get the most out of the software and hardware you use. For example, the system has been tested over the electrocardiogram application, achieving to reduce energy consumption to less than half. Therefore, the time that the application can be working is doubled, so the customer will reduce costs due to battery replacements.

Ours is a comprehensive solution. Every element of our elements is focused on helping the customer achieve the lowest total cost of energy consumption and supporting his own policies. We recommend adoption of this proposal now so that the new application can be energy optimized.

Table of Contents

1 Introduction

Personal-area wireless technologies have entered into certain aspects of our lives. The advent of a low-cost and extremely low-power protocol like IEEE 802.15.4 has been a major factor in redefining sensor networks [1]. As sensors and actuators research slowly matured and technology rapidly advanced, it has found itself merging into many applications, such as environment and habitat monitoring, home automations, traffic control, and more recently healthcare applications.

IEEE 802.15.4 is targeted to populate the home and office environment, offering low power solutions for toys, peripherals, control, security and monitoring type applications. IEEE 802.15.4 inception is derived from a notion of simplicity. When we compare other standards-based wireless technologies, they tend to be larger, in software and hardware implementation terms, and consume more (battery) power. Although, it is unclear at this time if IEEE 802.15.4 will actually achieve its marketed consumption statistics in addition to truly realizing a small software implementation.

This standard provides a short-range cost effective networking capability. It has been developed with the emphasis on low-cost battery powered applications, such as memory tagging, building automation, marine wireless and personal healthcare. With a tenth of the processor memory requirements of Bluetooth and a fraction of the MIPS needed for 802.11 networking devices, IEEE 802.15.4 is the best solution for loss data-rate, short-range communications.

In the healthcare field, Philips Research Aachen in the *Cableless Patient Project* has developed an ad-hoc body-centred approach for enabling continuous monitoring of vital signs without cables at the patient's body [2]. The approach of the *Cableless Patient Project* is to apply the generic concept of wireless sensor networks (WSN's) to health monitoring. Such a sensor network is a distributed micro-system consisting of cooperating nodes that comprise both processing and radio transmission in a highly miniaturized and integrated format. These characteristics make wireless sensor network technology an ideal platform for designing a distributed wireless system that allows physicians to dynamically form a body area network (BAN) tailored to the individual needs of the patient, by placing wireless nodes at the patient's body or in his immediate vicinity for acquiring, processing, storing, transmitting and displaying vital signs.

Energy efficiency is one of the major concerns in designing and implementing wireless data communication protocol for handheld devices and wireless sensor networks, IEEE 802.15.4 protocol tries to address this issue along with many other concerns in PAN. The IEEE 802.15.4 standard specifies the physical (PHY) and media access layer (MAC) for simple, low-cost radio communication networks. These networks offer low data rates and low energy consumption. The purpose of the IEEE 802.15.4 specification is to provide a standard for ultra-low complexity, ultra-low cost, ultra-low power consumption, and low data rate wireless connectivity among inexpensive devices.

A wireless sensor network consists of low-cost, low-power, and energy-

constrained sensors responsible for monitoring a physical phenomenon and reporting to a coordinator where the end-user can access the data. Since battery life directly impacts the extent and duration of mobility, one of the key considerations in the design of a wireless sensor should be to maximize the battery lifetime. Hence, the network lifetime becomes a critical concern in the design of WSNs.

1.1 Motivation

According to “Moore’s Law”, the transistor density of integrated circuits doubles about every two years [3]. As it is show in Figure 1, this law has been proven valid for the past 40 years. Researchers and designers are actively involved in reducing the size of the integrated circuits rapidly to develop atomic sized electronic devices. The biggest impediment to our technological future isn't extending Moore's law. Thanks to recent breakthroughs at the semiconductor manufacturing level, by 2010 top-tier processors should be stuffed with a billion transistors and running at more than 20 gigahertz. But, on the other hand, as the processor power doubles, the power consumption also rises.

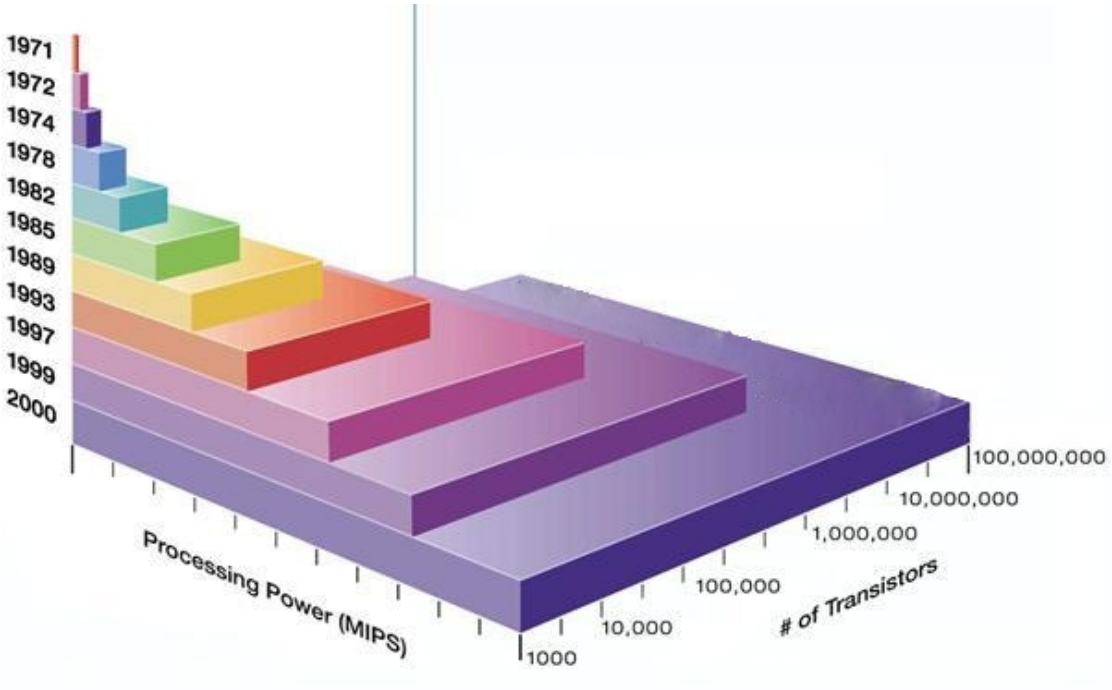


Figure 1. Growth of transistor counts during last decades and Moore's Law

During the last few decades, rechargeable batteries have made only moderate improvements in terms of higher capacity and smaller size. Compared with the vast advancements in areas such as microelectronics, the lack of progress in battery technology is apparent. Batteries, as has often been noted, do not obey Moore's Law. Therefore, managing the battery energy becomes essential.

Besides the short of battery evolution, in wireless healthcare applications the battery lifetime awareness is a major hurdle. During a patient monitoring, battery failure is critical. Hence, it is necessary to monitor the battery state of

charge of each one of the nodes that constitute the network.

This work is an effort towards validating an energy management architecture to study the energy consumption of the wireless setup, compliant with 802.15.4, during various stages of operation. The results obtained can be used to improve the network design to achieve greater energy efficient devices. The current prototype implementation involves, as main blocks, the Philips IEEE 802.15.4-based wireless sensor network platform AquisGrain and a Dallas semiconductor MAXIM DS2780 highprecision coulomb counter [4]. The Aquisgrain 1.0 has an Atmel ATmega128L microcontroller and a Chipcon CC2420 RF transceiver integrated on one board [5][6].

The microcontroller is the logic controller for communicating with the Chipcon radio and exchanging data with DS2780 using one-wire protocol. Code to generate Radio Frequency (RF) signals compliant with IEEE 802.15.4 protocol and communicate using one-wire interface to measure current consumption is developed in C language. Code organization is done with the aim to give flexibility for future developers to customize the functionality with minimal changes. Port configuration can also be easily changed to suit different hardware configurations.

1.2 Objectives and Scope

The main target of this thesis is to provide an energy management architecture and validate it on the Philips IEEE 802.15.4-based wireless sensor network platform AquisGrain.

This architecture will provide an energy monitor which will be in charge of determining the charging condition of the battery and the component energy usage. This monitoring will be integrated in the already developed ECG application to predict the remaining operating time of the wireless ECG sensor and to assure that the user is informed about low battery condition in a timely manner.

After that, the energy management architecture will be enhanced by allowing the user to specify policy directives to be enforced depending on the current energy conditions.

The detailed work steps taken in the current master thesis have been:

- Get acquainted with the *Cableless Patient Project* and the AquisGrain platform.
- Study the different existing battery technologies, and the different battery models.
- Analyse the IEEE 802.15.4 wireless standard and the MAC layer software implemented by Chipcon and used in the Aquisgrain platform.
- Analyse the different existing approaches for estimating the lifetime of a battery and monitor the battery state of charge.
- Implement an energy saver application on the wireless platform AquisGrain running the Chipcon MAC implementation.

- Choose the best technology to monitor the current consumption of the AquisGrain platform.
- Evaluate and tune the chosen battery monitor by testing the implementation with AquisGrain sensors.
- Build an interface to allow the user to specify a policy.
- Build an energy manager to merge the user requirements with the energy status.
- Test the whole energy management architecture in the ECG application under the AquisGrain platform.

1.3 Master thesis overview

Firstly, in chapter 2 the most important techniques used for measuring and simulating the power consumption in wireless sensor networks are explained. In addition, the AquisGrain 1.0 platform main characteristics are exposed.

Next, in chapter 3 a brief overview of the existing battery technologies is given to let the reader understand the problems faced when trying to predict the battery lifetime. Besides, we also present in the same chapter different existing battery models for estimating the battery capacity performance. This theory is necessary to understand the behaviour of the battery monitor technique selected and its performance.

Chapter 4 describes the design of an accurate and efficient energy management architecture. Each one of the blocks which integrate the architecture is explained in detail.

In chapter 5 we implement the proposed solution according to the design and the requirements of the previous chapters. As in the preceding chapter, the implementation of each architecture module is explained separately.

Subsequently, chapter 6 describes the different tests which are carried out to check the proper performance of the architecture. The evaluation dwells on battery monitor behaviour as it is the hardware added to the existing AquisGrain 1.0 platform.

Finally, we close the report with the conclusions of this Master Thesis and propose some future work items.

2 State of the Art

Wireless Sensor Networks have revolutionized the design of emerging embedded systems and triggered a new set of potential applications. This particular form of distributed and ubiquitous computing raises many challenges in terms of real-time communication and coordination due to the large number of constraints that must be simultaneously satisfied, including limited power, CPU speed, storage capacity and bandwidth. These constraints trigger the need for new paradigms in terms of node/sensor design and network communication/coordination mechanisms. The design of wireless sensor networks is mainly concerned with power-efficiency issues, due to the severe limitation in terms of energy consumption [7].

Research advances in highly integrated and low power hardware, wireless communication technology, and highly embedded operating systems enable sensor networks. A sensor network may consist of several thousands of nodes, each with very limited processing, storage, sensing and communication abilities.

Sensor nodes are usually battery driven. Due to this limited energy resource, energy consumption is a crucial design factor for hardware and software developers. Although hard- and software are strongly tied together in mobile and embedded device development, energy consumption is still a minor issue for software development. To enable efficient co-design of hard- and software for such devices, a deep evaluation of applications and systems in terms of energy consumption is crucial.

Furthermore, no breakthroughs in battery technology are to be expected in the next years. As long as new technologies like fuel cells do not advance from prototype level to mass production, batteries and thus energy consumption will be the limiting factor [3]. For developers, it is crucial to evaluate the energy consumption of applications accurately since the choice of algorithms and programming styles may strongly influence energy consumption. Once nodes are deployed, it is challenging and sometimes even impossible to change batteries. As a result, erroneous lifetime prediction may cause high costs and even render a sensor network useless before its purpose is fulfilled.

On the one hand, several network simulators have been developed in order to estimate the energy consumption of nodes in different working states before system deployment. In this field, it is worth highlighting three simulators PowerTOSSIM, ATEMU and AEON. On the other hand, due to the necessity to acquire the energy state of the network it has been developed several ways to get this information in the real world. A platform called MotePlat, based on sensor nodes under TinyOS, employs a three-tier application framework and adopts an agent mechanism in the software architecture of sensor nodes and sink node. With MotePlat users can remotely acquire node status information, such as its residual energy and adjust configuration of sensor nodes. Besides MotePlat, a scalable power observation tool (SPOT) has been implemented in order to measure *in situ* the node's power and energy over a dynamic range.

This chapter introduces the main tools used to estimate or measure the power in WSN. In addition, we lay out a brief explanation of the existing IEEE

802.15.4-based platform AquisGrain 1.0. Finally, the basis of an electrocardiogram wireless application is showed.

2.1 WSNs Power Simulators

2.1.1 AEON

AEON is a novel evaluation tool to predict energy consumption of sensor nodes [8]. Based on the execution of real application and OS code and measurements of node's current draw, the model enables accurate prediction of the actual energy consumption of nodes. Thus, it prevents erroneous assumptions on device and network lifetime. Moreover, AEON gives adequate and very fine grained feedback to a software engineer on the energy efficiency of the code. Such a detailed prediction allows the comparison of different low power and energy aware approaches in terms of energy efficiency and the estimation of the overall lifetime of a sensor network.

As sensor nodes consist of several components such as microcontrollers, radios, sensors, and memory, a detailed low-level model of all these devices is necessary to enable accurate prediction of energy consumption. The application and external events influence the program execution and so the state of a node. For example, applications turn on and off components like the radio and timer interrupts change the CPU from sleep to active mode. As such state changes happen frequently and each state consumes a different amount of power, the states and the timing of the state changes need to be modelled accurately. The single states of every component form the state of the whole node. The total current draw of a node is the sum of the currents of each component in the respective states.

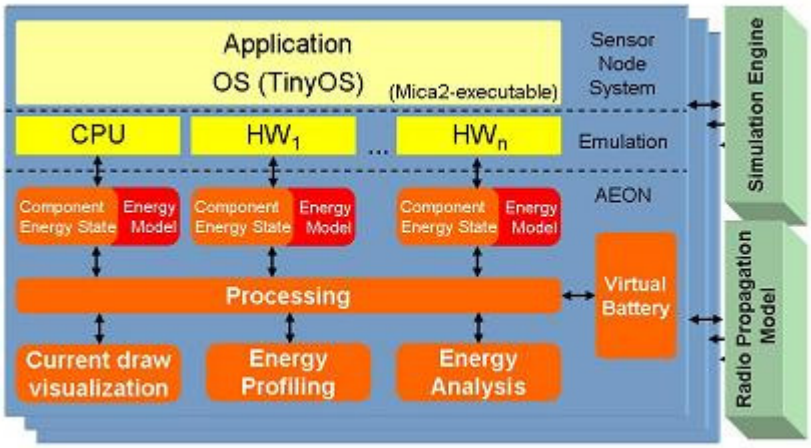


Figure 2. Block diagram of AEON's architecture.

The first challenge for the accurate prediction of energy consumption is to build a precise and detailed low-level energy model of the sensor node. To enable exact modelling of application execution and device state changes, the model is based on a sensor node emulator. The emulation of the node and the execution of real application and OS code allows to model the state of every single component at every point in time during program execution.

AEON is implemented on top of AVRORA, a highly scalable sensor node emulator. The energy model extends the implementation of each hardware component in AVRORA (see Figure 2) by monitoring its power usage during emulation. Furthermore, an energy profiling is added to enable a breakdown to individual source code routines and components. Additionally a radio propagation model provides realistic node communication. The performance analysis shows that the overhead added to AVRORA is minimal, as only the state changes of the components are monitored.

2.1.1.1 AVRORA

AVRORA simulates a network of motes with high fidelity so that an application developer can test their software on the desktop before deploying it in the real world [9]. Unique to AVRORA is the ability to support dynamically updatable code; simulation of which can help to avoid nasty surprises during deployment. Also unique is the ability to profile application code at arbitrary code locations, a feature that can help explain performance bottlenecks in the system.

AVRORA also provides a framework for program analysis, allowing static checking of embedded software and an infrastructure for future program analysis research. It is flexible, providing a Java API for developing analyses and removes the need to build a large support structure to investigate program analysis.

The provided simulator can test the programs before they are deployed onto the hardware device with cycle accurate execution times. The monitoring infrastructure allows users to add online monitoring of program behaviour for better program understanding and optimization opportunities. The energy analysis tool can analyze energy consumption and help to determine the battery life of your device.

2.1.2 PowerTOSSIM

PowerTOSSIM is a scalable simulation environment for wireless sensor networks that provides an accurate, per-node estimate of power consumption [10].

PowerTOSSIM is based on TOSSIM, an event-driven simulation environment for TinyOS applications, and derives much of its value from the tools built up around the TinyOS environment (see Figure 3). In PowerTOSSIM, TinyOS components corresponding to specific hardware peripherals (such as the radio, EEPROM, LEDs, and so forth) are instrumented to obtain a trace of each device's activity during the simulation run. To scale up to large numbers of sensor nodes, PowerTOSSIM runs applications as a native executable and does not directly simulate each node's CPU at the instruction level. Instead, PowerTOSSIM employs a code-transformation technique to estimate the number of CPU cycles executed by each node.

Finally, the trace of each node's activity is fed into a detailed model of hardware energy consumption, yielding per-node energy consumption data. This energy model can be readily modified for different hardware platforms.

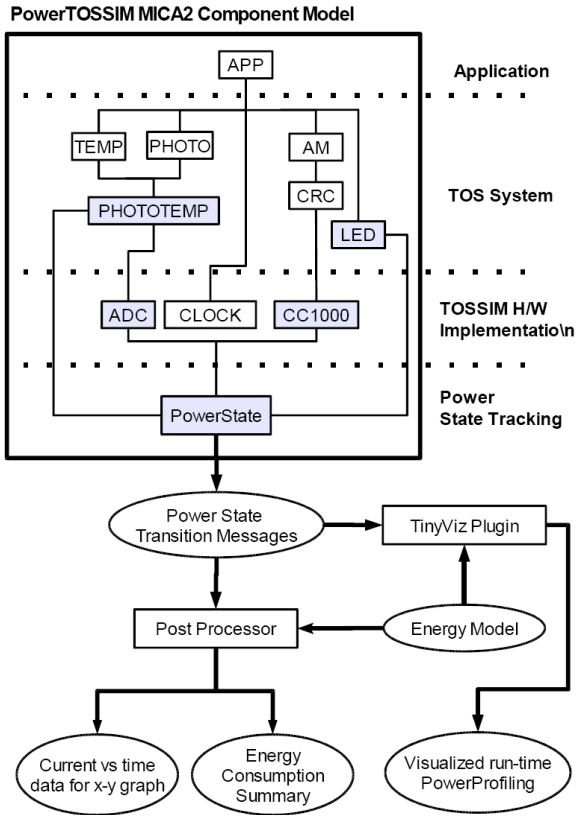


Figure 3. PowerTOSSIM architecture.

2.1.3 ATEMU

ATEMU is a software emulator for AVR processor based systems. Along with support for the AVR processor, it also includes support for other peripheral devices on the MICA2 sensor node platform such as the radio. ATEMU can be used to perform high fidelity large scale sensor network emulation studies in a controlled environment [11]. ATEMU can be directly used by developers of TinyOS related software as it is binary compatible with the MICA2 hardware. Though the current release only includes support for MICA2 hardware, it can be easily extended to include other sensor node platforms. It allows for the use of heterogeneous sensor nodes in the same sensor network. The ATEMU distribution consists of two components: the *ATEMU emulator core*, and the *xatdb graphical debugger*.

The ATEMU emulator core can simulate arbitrary numbers of nodes and can model their execution and interactions between them, such as radio communications in extremely fine detail. It offers nearly complete emulation of the MICA2 hardware platform and as a result provides results that are closer to real life operation of a distributed sensor network. The only difference between running an actual network of the MICA2 sensor nodes and emulating it in ATEMU is the operation of the "air". Currently only d^2 propagation is modelled, however this will be improved with future releases. ATEMU uses the same binary that is loaded onto the MICA2 node and uses its AVR processor emulation engine to very accurately model the execution of the code on each sensor node. Therefore, as very few details of the actual operation of a sensor node are abstracted out, it provides an excellent platform to perform unbiased com-

parisons of various sensor networking protocols and the results are significantly more realistic than other simulators.

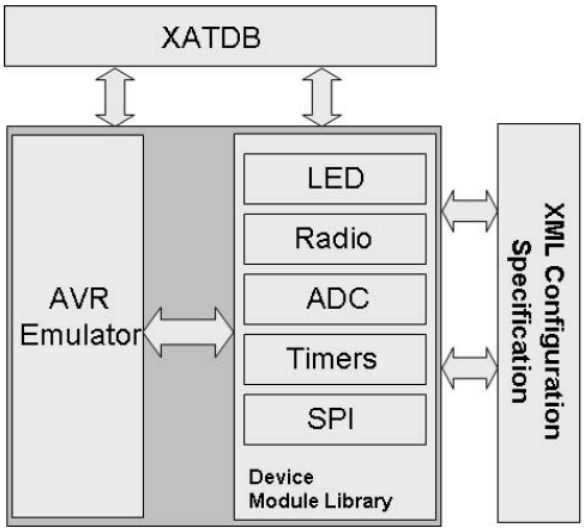


Figure 4. ATEMU architecture

Included in the ATEMU distribution is *xatdb*, Xatdb is a graphical *gdb*-like frontend to the ATEMU sensor network emulator. Xatdb provides users a complete system for debugging and monitoring the execution of their code. Using *xatdb*, users can run code built for the MICA2 platform, and debug efficiently using the ability to set breakpoints, watchpoints, as well the ability to single step through either assembly or nesC code. Xatdb is particularly powerful in its ability to provide a debugging interface to multiple nodes in a sensor network.

ATEMU differs from other existing simulators in that it has the ability to perform extremely low-level emulation of the sensor node hardware (see Figure 4). Though the current version includes support for the MICA2 sensor node, ATEMU can be easily extended to include support for other hardware platforms as well. The low level emulation capabilities of ATEMU make it an ideal complement to the existing set of simulation tools. Scenarios that require extremely high-fidelity results even at the expense of longer simulation times can benefit immensely from ATEMU. As ATEMU emulates the hardware of the MICA2 including the processor, radio interface, ADC, LEDs, and other peripheral devices, it is possible to run the same application binary image that is run on actual hardware in the emulator further eliminating any simulation artefacts that running a different binary format might introduce.

The ATEMU platform relies on the use of XML based configuration specification files. Using XML, it is possible to develop a common sensor network definition specification framework that can be used to specify the various parameters of a sensor network in a simulation tool agnostic format. This specification format can then either be used directly as input into to various simulation tools, or can be converted into simulator specific configuration files.

2.2 WSNs Power Measuring Tools

2.2.1 MotePlat

Based on sensor nodes under TinyOS, MotePlat is a monitoring and control platform to facilitate the development of WSN application systems. With MotePlat, the status information of the network can be obtained remotely, and the operations of sensor nodes can be controlled as well. To debug the software modules on nodes and evaluate the network performance, some commands can be injected into sensor network Moreover, MotePlat can be used for evaluation of protocols and mechanisms for sensor networks, and facilitate to develop application systems [12].

The application framework of MotePlat platform uses a three-tier application framework, as it is shown in Figure 5. The users and the sink nodes are both clients of the management server, which connects to several sink nodes of different networks via Internet and exchanges information for them. The management server integrates the information of sensor nodes, which is transmitted through multi-hop sensor network and Internet, gets the status information of whole network, and then sends them to the user terminal.

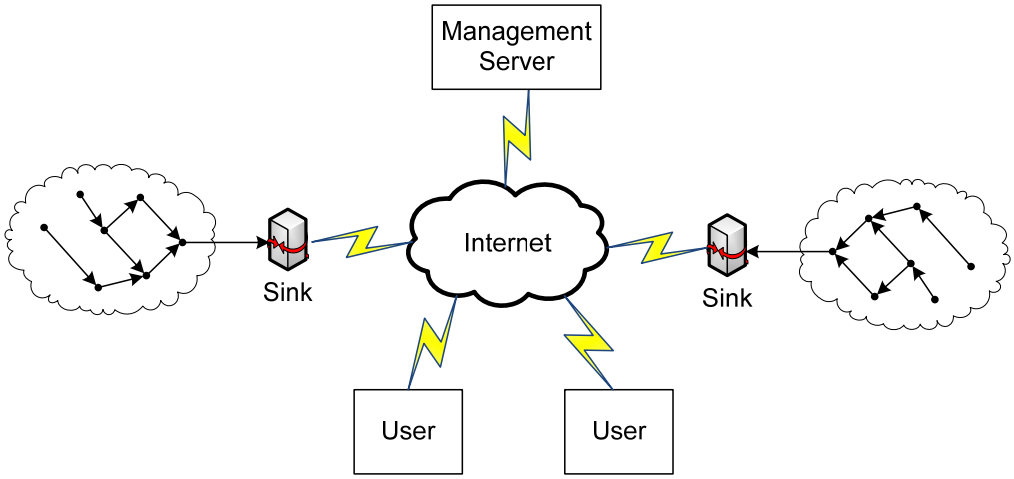


Figure 5. Application framework of MotePlat for wireless sensor networks

There are two different modes in acquiring information from sensor networks, which are time-driven mode and event-driven mode. In time-driven mode, nodes periodically read and report their status information according to preconfigured timers. In event-driven mode, nodes report the status only when important events happen. For instance, one node has to select a new parent and report the changing information when its old parent fails.

The network status information is deduced from the status information of all or most nodes in network. For example, energy distribution can be obtained from residual energy of the sensor nodes.

There are two mechanisms to distill network status information from nodes status information. One is inner-network processing, by which information is abstracted gradually during the process of data transmission. This mechanism can reduce energy consumption of the data transmission while increasing the burden of data processing and the delay of data transmission. The other is the

central processing in the management server. This mechanism reduces computing overheads of sensor nodes, but increases the amount of packets in the network.

Assuming each sensor node has the same initial energy, it can be described the residual energy of each node by using the amount of energy consumed. In other words, residual energy is the difference between the initial energy and the consumed energy.

For a given application, energy consumption is almost in proportion to times of certain operations performed by the node, and wireless communication module consumes most energy of a node, so the traffic is used to evaluate consumed energy in MotePlat. The proportion parameter between traffic and consumed energy is configurable in the system, which is applicable for different applications.

Link quality is another important aspect of nodes' status. In WSN, in order to choose an optimal route to the sink node, the routing protocol usually needs to assess the link quality between neighbour nodes. So MotePlat may get a neighbour nodes list and link quality between adjacent nodes from routing protocol. If routing protocol does not provide this information, MotePlat uses piggyback and periodical exploratory packets to obtain it.

By recording times of successful and failed data transmission, it can be evaluated communication quality between a sender node and a receiver node. In order to reduce overheads coming with link evaluation, exploratory packets are only used when normal traffic is rather low. By exchanging the information of link quality between neighbour nodes, it can be obtained the information of bidirectional link quality and neighbourhood [13].

If a user wants to monitor the network, it sends a monitoring command to the management server. The command will be forwarded to the relevant sink node which will then broadcast it to all the nodes. A command Agent within each node will continue to send the command to the Monitor Agent, which is responsible for reporting the status information to the user.

In conclusion, MotePlat presents a configurable platform for monitoring sensor networks, acquiring status information of both the nodes and the network and adjusting network behaviour.

2.2.2 SPOT

Scalable power observation tool enables in situ measurement of nodal power and energy over a dynamic range exceeding four decades or a temporal resolution of microseconds. Using SPOT, every node in a sensor network can be instrumented, providing unparalleled visibility into the dynamic power profile of applications and systems [14].

As with most sensing systems, the first stage is the sensor itself. In the case of energy monitoring, the physical quantity to measure is power, which is a product of voltage and current. It is assumed that the voltage is fixed and known a priori, or can be measured separately. Of course, current can be measured in several ways. SPOT uses a shunt resistor, which is one common method. A small resistor is placed in series with the power supply and the

voltage across this resistor, which is proportional to the current, is measured. There are several design considerations with this approach. The resistor can be placed between the mote and either the positive power supply or the negative power supply (ground).

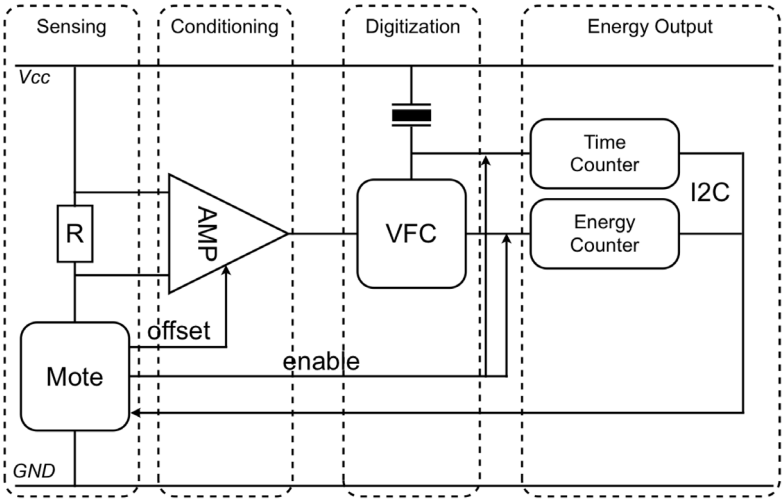


Figure 6. SPOT architecture

SPOT employs an simple architecture that uses a pure analog sampling front end to provide a large dynamic range, and a dual counter energy accumulation stage to provide high temporal resolution (see Figure 6).

These features are useful for profiling a range of systems (like pagers, PDAs, and cellphones) which exhibit highly bimodal or widely varying power profiles. SPOT will enable heretofore impossible empirical evaluations of low power designs at scale, and it will enable a new class of wireless sensor networks.

2.3 AquisGrain 1.0 Platform

Continuous monitoring of vital signs is an essential element for the treatment of patients in intensive care units to instantly detect critical events. For the physiological measurement of vital parameters sensing equipment needs to be attached to the patient's body. In intensive care units these sensors are typically wired to a bedside monitor that continuously displays the patient's waveforms, numerics and alarms. This set-up ties patients to their beds, which is inconvenient for them and complicates work for caregivers since the wires hinder access to patients.

These limitations are addressed at Philips Research in the Cableless Patient project aiming to develop a platform for enabling continuous monitoring of vital signs without cables at the patient's body. The approach of the Cableless Patient project is to apply the generic concept of sensor networks to health monitoring. A sensor network is a distributed micro-system consisting of co-operating nodes that comprise both processing and radio transmission in highly miniaturized and integrated format. Philips Research is designing a distributed wireless system that allows physicians to dynamically form a body area network (BAN) tailored to the individual needs of the patient by placing wireless nodes for acquiring, processing, storing, transmitting and displaying vital signs at the patient's body or in the immediate vicinity of the patient.

For this purpose, Philips has developed a versatile IEEE 802.15.4-based ZigBee-ready wireless sensor node platform (in the following referred to as AquisGrain) for continuous patient monitoring [2].

AquisGrain 1.0, which is the main module of the platform, contains the micro-controller unit (Atmel Atmega128L), the radio chip (ChipCon CC2420 IEEE 802.15.4), and other necessary devices [5][6].

Figure 7 and Figure 8 depict the back and the front of the AquisGrain 1.0 board, together with an indication of all of its main hardware modules.

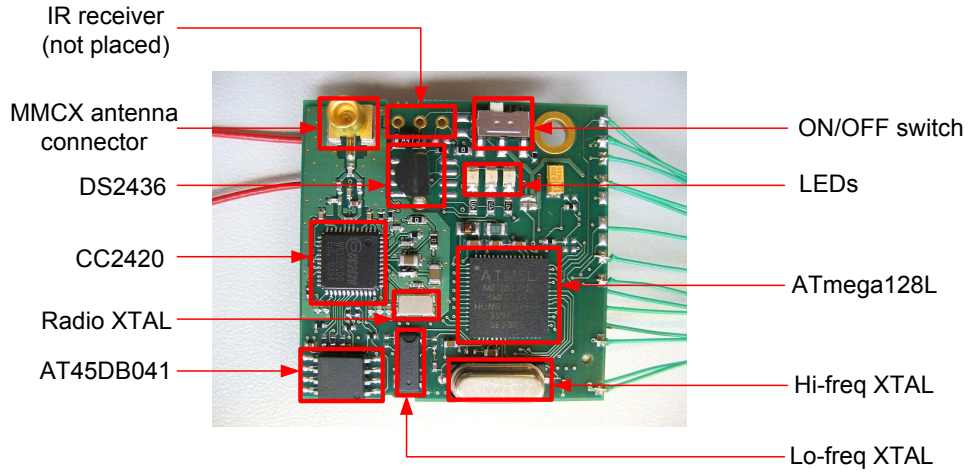


Figure 7. AquisGrain 1.0 board (front)

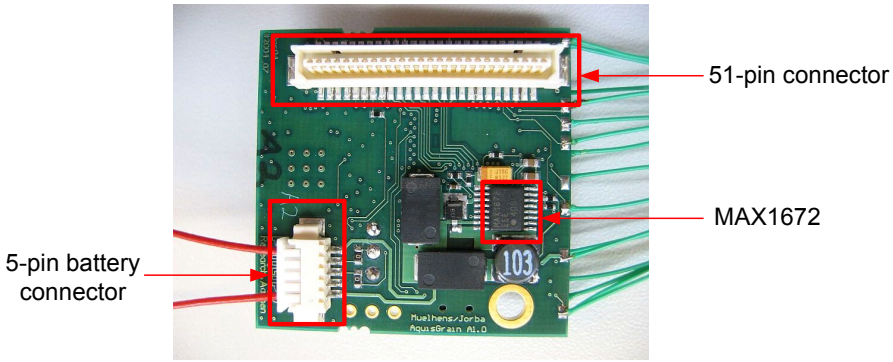


Figure 8. AquisGrain 1.0 board (back)

The 51-pin connector interfaces to the following lines:

- external interrupt signals.
- lines to perform analogical comparisons.
- 8 ADC channels.
- 11 general purpose input-outputs.
- I2C lines.
- UART interfaces.
- SPI interface.
- Battery monitor output line.

The lines for the 5-pin battery connector are the following:

- PACK+, which should be connected to the positive input power of the AquisGrain 1.0 node.
- Data, which is a bi-directional data line based in the Dallas Semi-conductor's proprietary solution one-wire interface
- A reserved line.
- Voltage charge, in which the charger power output is supposed to be connected
- PACK-, which should be connected to the negative input power of the AquisGrain 1.0 node.

Figure 9 shows the block diagram of the AquisGrain 1.0.

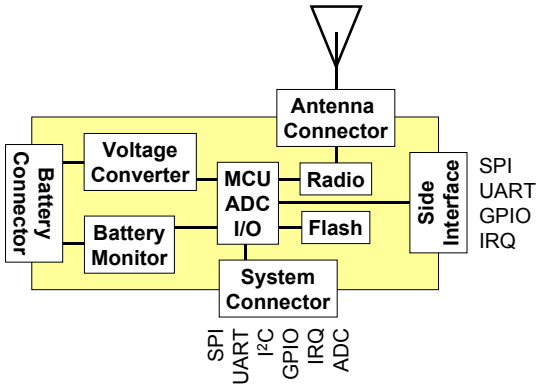


Figure 9. AquisGrain 1.0 block diagram

The main features of AquisGrain 1.0 are:

Dimensions	
Size:	35 x 36 mm^2
MCU: Atmel ATmega128L	
RAM:	4 Kbytes
Program space:	128 Kbytes
External flash:	4 Mbit
Serial i/f:	SPI, 2 UART, I2C
Other i/f:	8 PWMs
ADC:	8 10-bit chanels
Radio: ChipCon CC2420	
Frequency:	2400-2483 MHz
Channels:	16
Data rate:	250 kbit/s
Output power:	-25 to 0 dBm
RF connector:	MMCX

Power consumption	
Current (active):	31 mA
Current (sleep):	47 µA
Supply:	1.8-5.5 V

Table 1. AquisGrain 1.0 features

2.3.1 AquisGrain 1.0 Software

The software is written in C for an Atmel AVR microcontroller. ChipCon provides the basic functionality software for the CC2420 to implement the IEEE 802.15.4 MAC. The source code of the MAC software is available from ChipCon under a licensing agreement. Philips Research has enhanced this source code and has merged it with own software components.

2.3.1.1 Principles of ChipCon MAC

The ChipCon MAC works as a single thread (the main loop) interrupted by several sources (Timer, UART, ADC etc). Additional tasks can also be executed; these tasks are scheduled by TIMER1-interrupts. Always one ready task with the highest priority will be executed per timer step (interval of 320 microseconds). This means that every action by tasks is executed on interrupt level. It is like a time slice passes to a thread. For long-term operations a task must implement a finite state handler to perform its specific operations.

Four priority levels are defined; always the first run-ready task of highest priority will be executed on firing of the task scheduler (via TIMER1A). When the processing of the current task takes more than 320 microseconds the task is interrupted by the task scheduler, which can start another task of higher priority. This will interrupt the current task and cannot re-enter the same task.

Sequences of statements that may not be interrupted must be embedded between DISABLE_GLOBAL_INT and ENABLE_GLOBAL_INT functions. This disables the interrupt so that the code section cannot be interrupted. It is important to take care not to call functions within a critical section; if they also use critical sections they would enable the interrupt before the first one does it.

2.3.1.2 The application framework

The main framework (app_framework.c) is generic for all applications. The application framework calls the ChipCon MAC, the application utility and the application module and may be called by any, itself only calls the framework. By using the framework only the application part has to be implemented; thus creating the sensor-devices applications is much simpler. Figure 10 shows the module groups.

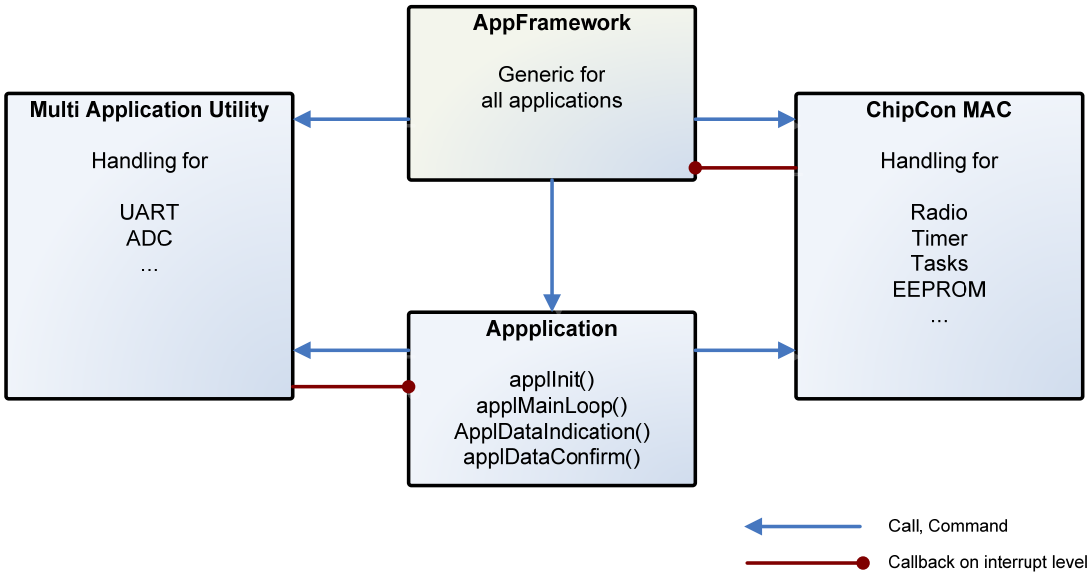


Figure 10. AquisGrain 1.0 software architecture

A sensor application module must implement the functions described in the application template (app_template.c). The application has access to global variables of the framework and all global variables of the ChipCon MAC. The application module implements an application description structure that describes all application specific information (name, version, type, coordinator y/n, etc…).

2.3.1.3 Programming boards

AquisGrain 1.0 board can be programmed using following boards:

- Atmel STK 500 for downloads. (Using serial communication with the AquisGrain 1.0 requires a special cable).
- MIB510 programming board (can be used both for downloading and serial communication).

2.4 Electrocardiogram Overview

The ECG is a non-invasive technique, meaning that this signal can be measured without entering the body at all. ECG monitoring and interpretation have always been tasks conventionally assigned to trained medical care personals. Although being more comprehensive in the related knowledge, the constraints to manpower are also very obvious. Fatigue factors and overwhelming workloads are both possible causes to delayed emergency response that may have reduced the chances for patients' survival. By automating this process, the system frees the medical professionals of the tedious tasks to center their attentions on something much more demanding.

Measurement of characteristic electrocardiogram (ECG) signal time intervals, such as QRS-complex width, P–Q and Q–T intervals, requires accurate detection of the QRS onset and offset points (see Figure 11). This is a "classical" problem of quantitative electrocardiography. It has been approached in many

different ways, especially after the establishment of computerised electrocardiography and the introduction of “intelligent” electrocardiographs

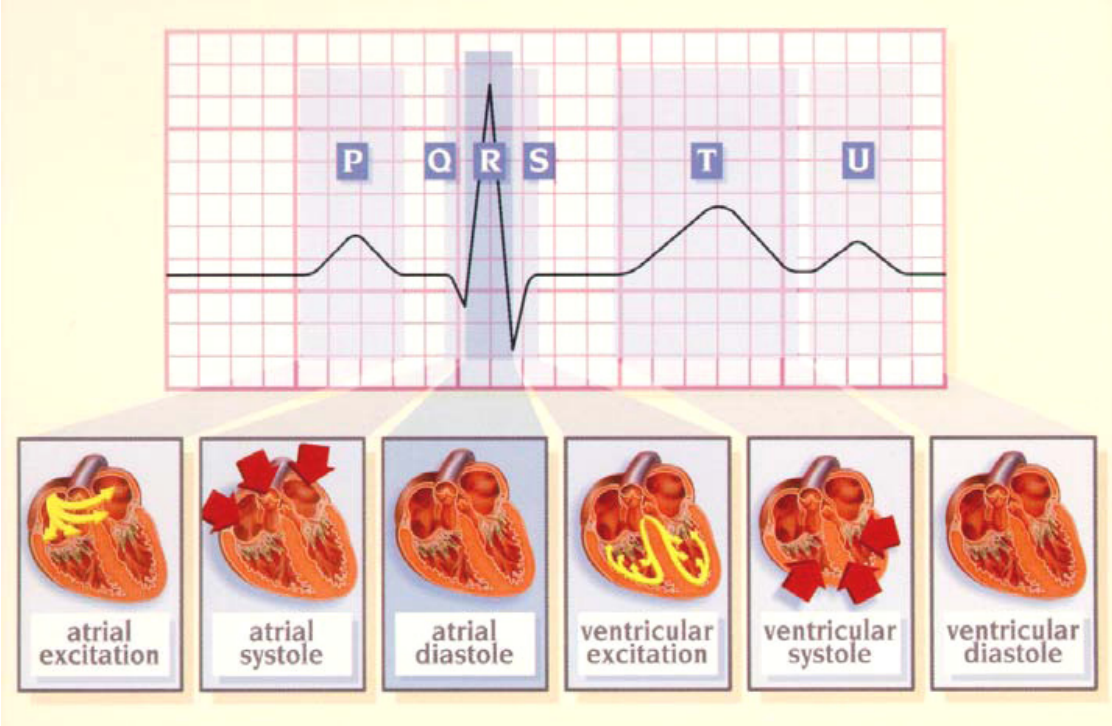


Figure 11. Characteristic ECG pattern of a healthy heart beating

Electrodes are used for sensing bio-electric potentials as caused by muscle and nerve cells. ECG electrodes are generally of the direct-contact type. They work as transducers converting ionic flow from the body through an electrolyte into electron current and consequentially an electric potential measurable by the front end of the ECG system.

By eliminating the long cables between nodes, the patient is comfortably able to move around without the hassle of wires, while also being able to place the electrodes on themselves without being impeded by leads. Similarly it provides the doctor or nurse with a trouble-free approach to the patient’s ECG signal. In addition, software could allow ECG signals to be saved and sent possibly by email to other parts of the world. In fact, for patients in rural and regional areas an ECG report could be sent via email to a doctor for examination.

Philips Research has developed an application which runs under AquisGrain 1.0 platform and captures samples from the leadwires and processes the signal to display the ECG waveform and the heart rate of the patient.

3 Battery Theory Background

Research has brought about a variety of battery chemistries, each offering distinct advantages, but none providing a fully satisfactory solution. With today's increased selection, however, better choices can be applied to suit specific user applications.

The consumer market, for example, demands high energy densities and small sizes. This has to be done to maintain adequate runtime of portable devices that are becoming increasingly more powerful and power hungry. Relentless downsizing of the portable equipment has pressured manufacturers to invent smaller batteries.

3.1 Overview of battery technologies

There are five major types of secondary rechargeable batteries [15]:

- **Lead-acid** battery is the most widely used and its main application is in the automotive field. Its advantages are low cost, high voltage per cell and good capacity life. Its disadvantages are that it is relatively heavy, it has poor low-temperature characteristics and it cannot be left in the discharged state for too long without being damaged. Charging of a lead-acid battery may be carried out with a simple charger under controlled-time conditions. The rated capacity is usually given at the C rate, i.e. the capacity obtained when a fully charged battery is discharged to bring it to an end-voltage of 1.75 V/cell in 20 hours is the most economical for larger power applications where weight is of little concern. The lead acid battery is the preferred choice for hospital equipment, wheelchairs, emergency lighting and UPS systems.
- **Nickel-cadmium (NiCd)** battery is mechanically rugged and long-lived. In addition it has excellent low-temperature characteristics and can be hermetically sealed. Its cost however is higher than either lead-acid or the nickel-zinc battery. The choice between NiCd and lead-acid depends very much on the particular application and on the performance characteristics required. This chemistry is mature and well understood but relatively low in energy density. The NiCd is used where long life, high discharge rate and economical price are important. Main applications are two-way radios, biomedical equipment, professional video cameras and power tools. The NiCd contains toxic metals and is not environmentally friendly.
- **Nickel-Metal Hydride (NiMH)** has a higher energy density compared to the NiCd at the expense of reduced cycle life. NiMH contains no toxic metals. Applications include mobile phones and laptop computers.
- **Lithium Ion** (Li-ion) is the fastest growing battery system. Li-ion is used where high-energy density and light weight is of prime importance. The Li-ion is more expensive than other systems and must follow strict guidelines to assure safety. Applications include notebook computers and cellular phones.

- **Lithium Ion Polymer** (Li-ion polymer) — a potentially lower cost version of the Li-ion. This chemistry is similar to the Li-ion in terms of energy density. It enables very slim geometry and allows simplified packaging. Main applications are mobile phones.
- **Reusable Alkaline** — replaces disposable household batteries; suitable for low-power applications. Its limited cycle life is compensated by low self-discharge, making this battery ideal for portable entertainment devices and flashlights.

Table 2 summarizes some of the common types of secondary batteries.

	Volts per cell (V)	Energy density (Wh/kg)	Battery discharge rate	Battery life	Cost per Cycle (€)	Battery uses
Ni-Cad	1.25	45	Moderate to high	Up to 1500 charge/discharge cycles	0.03	Low-end: cordless phones, PDAs, handheld computers, portable hand tools, tape players and toys
NiMH	1.5	70	Moderate	Up to 500 charge/discharge cycles	0.09	Midlevel: cordless phones, cell phones, laptops, PDAs, handheld computers and other portable devices
Li-Ion	1.0-3.6	135	Low	Up to 1000 charge/discharge cycles	0.11	High-end: cordless phones, cell phones, laptops, PDAs, handheld computers and other portable devices
Li-Ion polymer	1.0-3.6	135	Low	Up to 1000 charge/discharge cycles	0.22	High-end: cell phones, laptops, PDAs, handheld computers and other portable devices
Reusable Alkaline	1.5V	80 (initial)	Low	Up to 50 charge/discharge cycles	0.1	Potential for backup power systems, electric vehicles, cordless power tools and mobile electronic devices

Table 2. Characteristics of commonly used rechargeable batteries.

3.2 Battery Packs

Downsizing required smaller and more compact cell design. The button cell, which gained popularity in the 1980s, was a first attempt to achieve a reasonably flat geometry, or obtain higher voltages in a compact profile by stacking. The early 1990s brought the prismatic cell, which was followed by the modern pouch cell [16].

Nowadays, the most extended types of battery packs are:

- **Cylindrical cell:** is the most widely used packaging style. The advantages are ease of manufacture and good mechanical stability. The cyl-

inder has the ability to withstand high internal pressures. The cylindrical cell is moderately priced and offers high energy density. Typical applications are wireless communication, mobile computing, biomedical instruments, power tools and other uses that do not demand ultra-small size. The drawback of the cylindrical cell is less than maximum use of space. When stacking the cells, air cavities are formed. Because of fixed cell size, the pack must be designed around the available cell size.

Almost all cylindrical cells are equipped with a venting mechanism (see Table 12) to expel excess gases in an orderly manner. Whereas nickel-based batteries feature a resealable vent, many cylindrical Li-ion contain a membrane seal that ruptures if the pressure exceeds 3448 kPa (500 psi). There is usually some serious swelling of the cell before the seal breaks. Venting only occurs under extreme conditions.

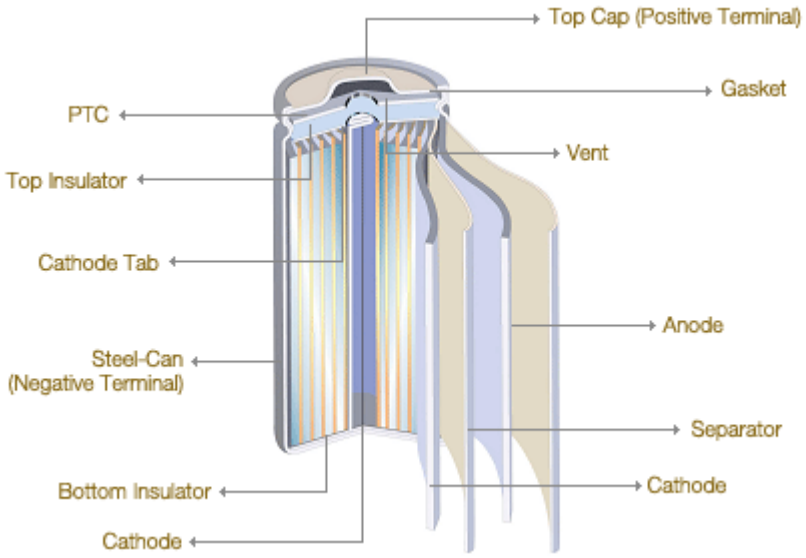


Figure 12. Cross-section of a classic NiCd cell

- **Button cell**: its construction (non rechargeable) was primarily a size issue. The button cell was designed to low power consumption application such as watches, calculators, hearing aids etc. The rechargeable button cell is more complicated because there are no safety vents in a button cell. The lack of safety vents (see Figure 13) results in a prolonged charging time –10+ hours– to minimize the heat increase and subsequent cell swelling.

  Rechargeable lithium button cells are available at the market with typical weights of 0.3 to 6 g and with capacities of 0.010 to about 100 mAh. These very small accumulators are produced for applications with very shallow cycles, where a deeper discharge is possible from time to time but only as an exception. These button cells are for the support of memories and timers.

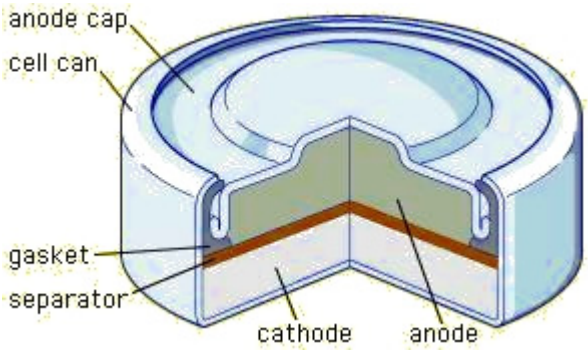


Figure 13. Cross-section of a button cell

- **Prismatic cell:** is specially suited for applications that require a tin battery; Mobil phone, PDA etc. The most common technology used for prismatic cells is Lithium. The shape of the prismatic cell (see Figure 14)gives it a bit lower energy density and stability than the cylindrical shape. The battery is also more expensive to manufacture than the cylindrical cell.

  The disadvantage of the prismatic cell is slightly lower energy densities compared to the cylindrical equivalent. In addition, the prismatic cell is more expensive to manufacture and does not provide the same mechanical stability enjoyed by the cylindrical cell. To prevent bulging when pressure builds up, heavier gauge metal is used for the container. The manufacturer allows some degree of bulging when designing the pack. The prismatic cell is offered in limited sizes and chemistries and runs from about 400 mAh to 2000 mAh and higher.

  Because of the very large quantities required for mobile phones, special prismatic cells are built to fit certain models. Most prismatic cells do not have a venting system. In case of pressure build-up, the cell starts to bulge. When correctly used and properly charged, no swelling should occur.

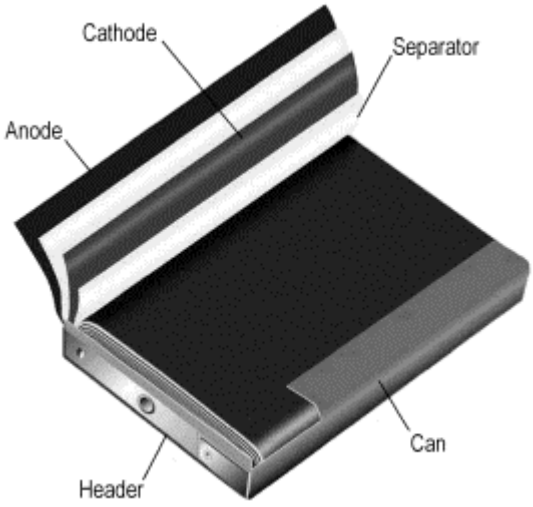


Figure 14. Cross-section of a prismatic cell

- **Pouch cell:** are typically used for Lithium Polymer cells with solid electrolytes, providing a low cost "flexible" construction. The electrodes and

the solid electrolyte are usually stacked in layers or laminations and enclosed in a foil envelope, as shown in Figure 15. The solid electrolyte permits safer, leak-proof cells. The foil construction allows very thin and light weight cell designs suitable for high power applications but because of the lack of rigidity of the casing the cells are prone to swelling as the cell temperature rises. Allowance must be made for the possibility of swelling when choosing cells to fit a particular cavity specified for the battery compartment. The cells are also vulnerable to external mechanical damage and battery pack designs should be designed to prevent such possibilities.

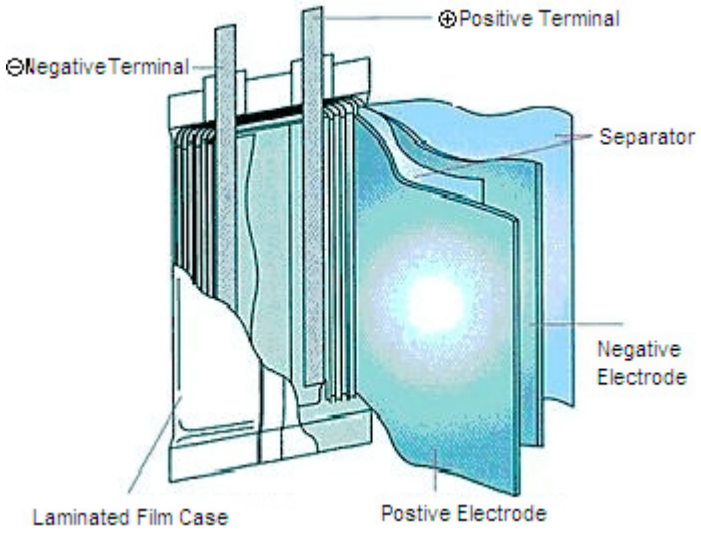


Figure 15. Cross-section of a pouch cell

3.3 Battery Protection Circuit

The purpose of cell protection is to provide the necessary monitoring and control to protect the cells from out of tolerance ambient or operating conditions and to protect the user from the consequences of battery failures. Cell protection can be external to the battery and this is one of the prime functions of the Battery Management System [17].

In general cell protection should address the following undesirable events or conditions:

- Excessive current during charging or discharging.
- Short circuit
- Over voltage - Overcharging
- Under voltage - Exceeding preset depth of discharge (DOD) limits
- High ambient temperature
- Overheating - Exceeding the cell temperature limit
- Pressure build up inside the cell
- System isolation in case of an accident
- Abuse

A modern battery is a delicate storage device that requires protection to safeguard against damage. Excessive temperatures will cause all cells to fail eventually. Most protection circuits therefore incorporate a thermal fuse which will permanently shut down the battery if its temperature exceeds a predetermined limit.

The most basic protection is a fuse that opens on excess current. Some fuses disengage permanently and render the battery useless once the filament is broken; other safety devices are resettable. A resettable fuse provides on-battery over-current protection. It has a similar function to a thermal fuse but after opening it will reset once the fault conditions have been removed and after it has cooled down again to its normal state. It requires no manual resetting or replacement and so is very convenient for the user who may not even be aware of its operation.

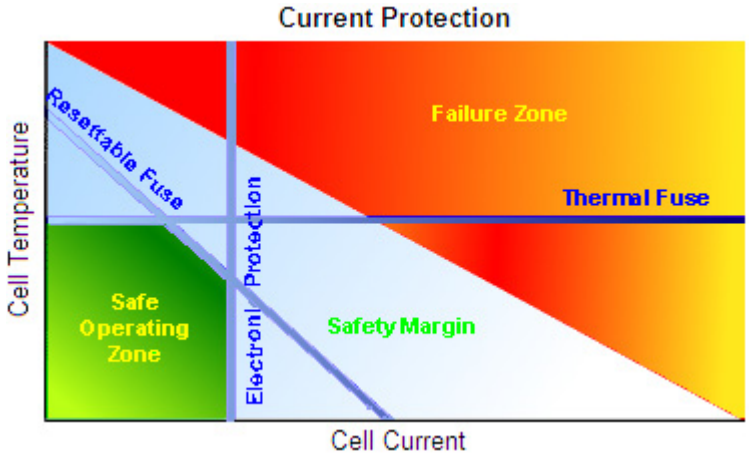


Figure 16. Protection schemes providing two levels of protection from both over-current and over temperature.

Figure 16 shows three protection schemes providing two levels of protection from both over-current and over temperature. If one fails the other one is there as a safety net.

The fuse is triggered when a particular temperature is reached. The temperature rise can be caused by the resistive self heating of the thermistor due to the current passing through it, or by conduction or convection from the ambient environment. Thus it can be used to protect against both over- current and over-temperature.

Batteries used in hazardous areas must be intrinsically safe. Hazardous areas include oil refineries, mines, grain elevators and fuel handling at airports. These areas are typically serviced with two-way radios and computing devices. Intrinsically safe batteries prevent excessive heat buildup and the danger of an electric spark on equipment failure. Because of tight approval standards, intrinsically safe batteries carry twice to three-times the price tag of regular packs.

Over-current protection is normally provided by a current sensing device which detects when the upper current limit of the battery has been reached and interrupts the circuit. Since current is difficult to measure the usual method of current sensing is by measuring the voltage across a low ohmic value, high precision, series, sense resistor in the current path. When the

specified current limit has been reached the sensing circuit will trigger a switch which will break the current path. The switch may be a semiconductor device or even a relay. Relays are inexpensive, they can switch very high currents and provide very good isolation in case of a fault but they are very slow to operate. Field Effect Transistor (FET) switches are normally used to provide fast acting protection but they are limited in their current carrying capability and very costly for high power applications [17].

Once the fault conditions have been removed, the circuit would normally be reconnected automatically, however there are particular circumstances when the circuit would be latched open. This could be to protect an unsuspecting service engineer investigating why a high voltage battery had tripped out.

Another battery that contains high-level protection is lithium-ion. Commercial Li-ion packs contain one of the most exact protection circuits in the battery industry. These circuits assure safety under all circumstances when in the hands of the public. Typically, a FET opens if the charge voltage of any cell reaches 4.30 V and a fuse activates if the cell temperature approaches 90 °C. In addition, a disconnect switch in each cell permanently interrupts the charge current if a safe pressure threshold of 1034 kPa is exceeded. To prevent the battery from over-discharging, the control circuit cuts off the current path at low voltage, which is typically 2.50 V/cell. Prolonged storage at voltages of 1.5 V/cell and lower damages the lithium-ion, causing safety problems if attempted to recharge.

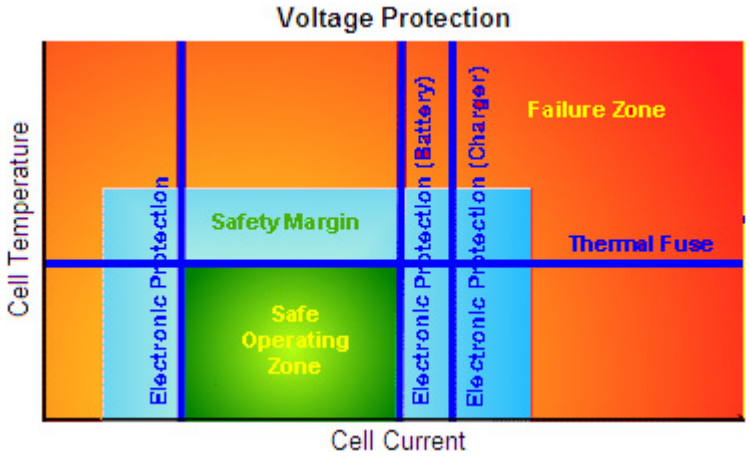


Figure 17. Scheme for over and under-voltage, as well as temperature protection

Figure 17 shows a scheme for over and under-voltage, as well as temperature protection. In this case it also shows interaction with the charger. Batteries can be damaged both by over-voltage which can occur during charging and by under-voltage due to excessive discharging. This scheme allows voltage limits to be set for both charging and discharging. Batteries can be particularly vulnerable to overcharging. By providing the charger with inputs from voltage and temperature sensors in the battery, the charger can be cut off when the battery reaches predetermined control limits. The diagram above only shows a single voltage cut off from the charger, however multiple protection circuits can be implemented to provide a comprehensive protection scheme involving the charger as well as the protection built into the battery.

It should be noted that each protection device added into the main current

path will increase the effective internal impedance of the battery, as much as doubling it in the case of single cell batteries. This adversely affects the battery's capability of delivering peak power.

Figure 18 shows an example of a battery failure when no protection circuit is placed. The battery pack was replaced in the defibrillator, and when the unit was turned on, an explosion or battery rupture occurred.

Figure 18. Automated external defibrillator explosion due to lithium battery rupture.

3.4 Battery Performance Characteristics

The battery is often considered by engineers as a constant voltage source that does not require any second thought. However, battery is in fact a little portable chemical factory with limited available resources.

In order to choose the optimum battery for an application, it is necessary to know the parameters which are used to characterise cell performance. Energy cells have been developed for a wide range of applications using a variety of different technologies, resulting in a wide range of available performance characteristics [15]. In the next points, factors an applications engineer should take into account when specifying a battery to match the performance requirements of the end product are explained.

3.4.1 Cell Chemistry

The nominal voltage of a galvanic cell is fixed by the electrochemical characteristics of the active chemicals used in the cell, the so called cell chemistry. The actual voltage appearing at the terminals at any particular time, as with any cell, depends on the load current and the internal impedance of the cell and this varies with temperature, the state of charge and with the age of the cell.

Figure 19 shows typical discharge discharge curves for cells using a range of cell chemistries when discharged at 0.2 C rate. Note that each cell chemistry has its own characteristic nominal voltage and discharge curve (see Figure 19). Some chemistries such as Lithium Ion have a fairly flat discharge curve while others such as Lead acid have a pronounced slope.

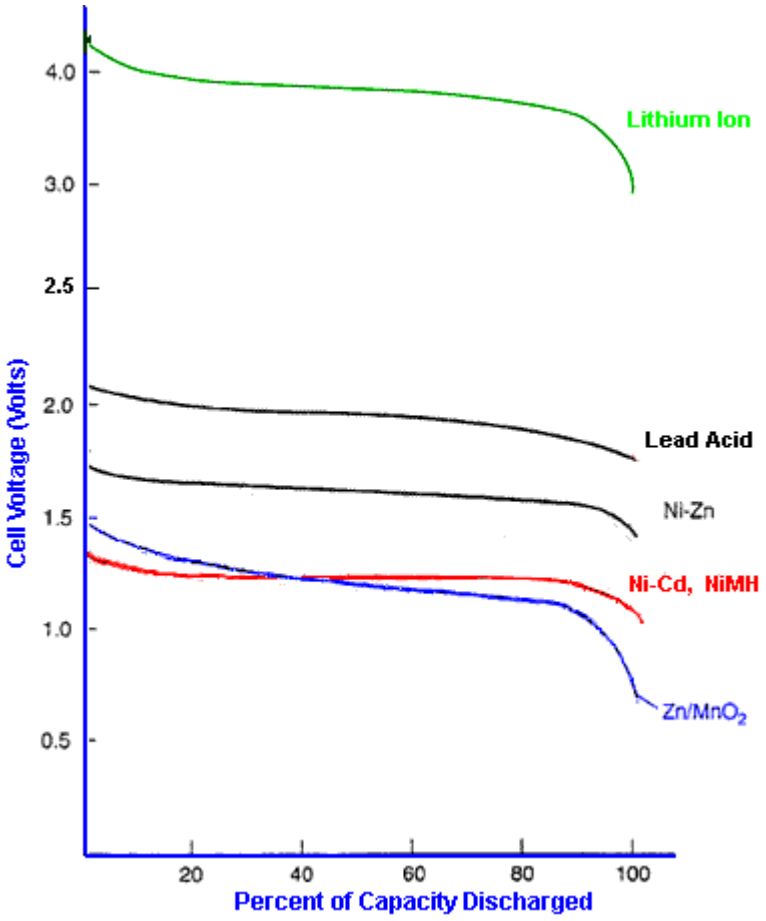


Figure 19. Discharge curves for different cell chemistries

The power delivered by cells with a sloping discharge curve falls progressively throughout the discharge cycle. This could give rise to problems for high power applications towards the end of the cycle. For low power applications which need a stable supply voltage, it may be necessary to incorporate a voltage regulator if the slope is too steep. This is not usually an option for high power applications since the losses in the regulator would rob even more power from the battery.

A flat discharge curve simplifies the design of the application in which the battery is used since the supply voltage stays reasonably constant throughout the discharge cycle. A sloping curve facilitates the estimation of the state of charge of the battery since the cell voltage can be used as a measure of the remaining charge in the cell. Modern Lithium Ion cells have a very flat discharge curve and other methods must be used to determine the state of charge

3.4.2 Thermal Characteristics

The effect of ambient temperature on battery efficiency depends strongly on the specific battery chemistry being considered (see Figure 20). Most batteries perform well at room temperature [16]. Higher temperatures allow for increased mobility of the electrolyte materials, which result in lower internal resistance. This has the effect of increasing the effective capacity of the battery. However, continuous exposure to elevated temperatures have other

undesirable effects, such as shortened cycle life (the number of times the battery can be charged/discharged), and an increased rate of self-discharge (the irreversible loss of charge that occurs when no current is drawn from the battery).

Temperatures of less than 15 °C cause capacity to drop and at low temperatures under -10 °C primary cells on a zinc basis cannot be efficiently applied, except when a fraction of the nominal capacity is sufficient. Lithium batteries still show good load capability at -20 °C. The accumulator's capacity in general deteriorates but some designs like the NiCd series with sintered electrodes are still applicable to -45 °C. Rechargeability is at these temperatures very poor except if the charging device is sophisticated enough to compensate this.

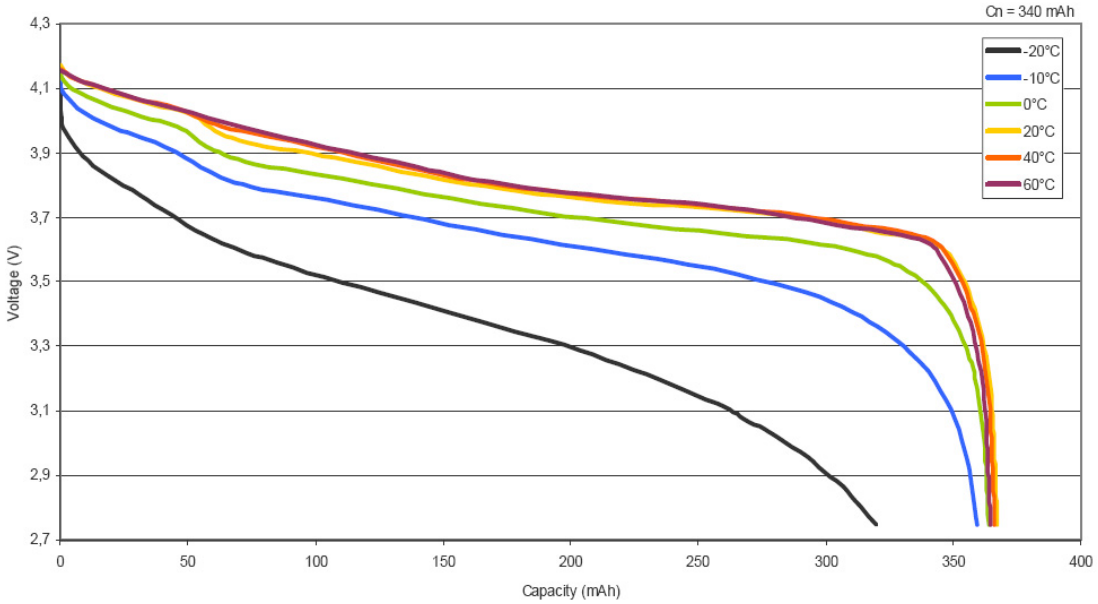


Figure 20. Performance of Lithium Ion batteries as the operating temperature changes

3.4.3 Self Discharge Characteristics

Batteries do not retain charge forever. There are many ways charge can be lost, depending on battery chemistry and design [15].

Self-discharge is the electrical capacity that is lost when the cell simply sits on the shelf. Self-discharge is caused by electrochemical processes within the cell and is equivalent to the application of a small external load. Li-ion cells typically lose 8% of their capacity for the first month and then 2% for subsequent months (see Figure 21). In many cases the rate of loss may decline after a time due, for instance, to the build-up of a passivation film on lithium anodes. In order to reduce self-discharge, it is recommended to store cells and batteries at lower temperatures.

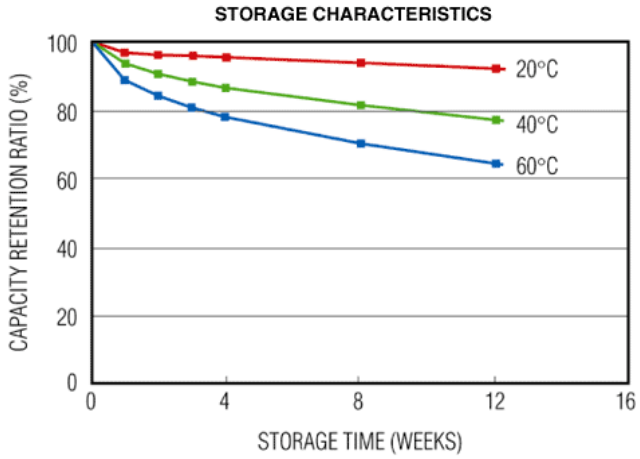


Figure 21. Typical self discharge rates for a Lithium Ion battery

3.4.4 Internal Resistance

Internal resistance in a chemical cell is due mainly to the resistance of the electrolyte between electrodes. Any current in the battery must flow through the internal resistance. The internal resistance is in series with the voltage of the battery, causing an internal voltage drop.

One of the requirements of a battery for digital applications is low internal resistance. Measured in milliohms, the internal resistance is the gatekeeper that, to a large extent, determines the runtime. The lower the resistance, the less restriction the battery encounters in delivering the needed power spikes. A high mW reading can trigger an early 'low battery' indication on a seemingly good battery because the available energy cannot be delivered in the required manner and remains in the battery.

3.4.5 Discharge Rates

The charge and discharge current of a battery is measured in C-rate. Most portable batteries, with the exception of the lead acid, are rated at 1 C. If the discharge takes place over a long period of several hours as with some high rate applications such as electric vehicles, the effective capacity of the battery can be as much as double the specified capacity at the C rate.

The discrepancy in capacity readings with different C-rates largely depends on the internal resistance of the battery (see Figure 22). To compensate for the different readings at various discharge currents, manufacturers offer a capacity offset. Applying the capacity offset does not improve battery performance; it merely adjusts the capacity calculation if discharged at a higher or lower C-rate than specified.

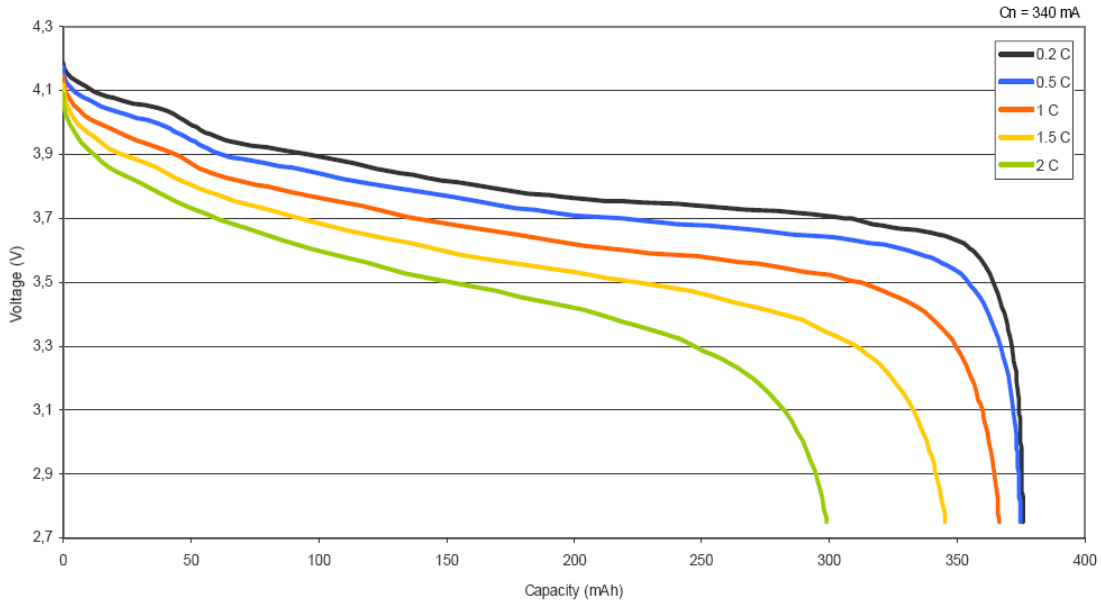


Figure 22. Typical discharge curves for a Lithium Ion battery during variable discharging rates.

3.4.5.1 Peukert Equation

Peukert's Law expresses the capacity of a battery in terms of the rate at which it is discharged. As the rate increases, the battery's capacity decreases, although its actual capacity tends to remain fairly constant. The basic form of the equation is:

$$C = I^n \cdot T$$

Where:

I = the discharge current in amps

T = the time in hours

C = the capacity of the battery in amp hours

n = Peukert's exponent for that particular battery type

The advantage of Peukert's equation is that it allows us to perform a logarithmic interpolation of the logarithmic battery discharge curve. This enables us to calculate the manufacturer's rated discharge current for the rated time at any point along the curve.

3.4.5.2 Ragone Plot

For short discharge times and if weight is the critical factor, specific power can be more important than specific energy. A battery that can only be discharged relatively slowly will be inappropriate for a rapid discharge, of the order of a few minutes. The Ragone Plot compares the discharge performance of some reserve batteries with generator systems, for specific energy and specific power (see Figure 23). There are clearly large differences between achievable power densities. Thermal batteries can deliver up to 5 kW/kg of power, whilst silver-zinc can manage only 1 kW/kg and lithium about 0.1 kW/kg if cooling is

used. Lead-acid batteries used for this purpose can only achieve about 0.45 kW/kg.

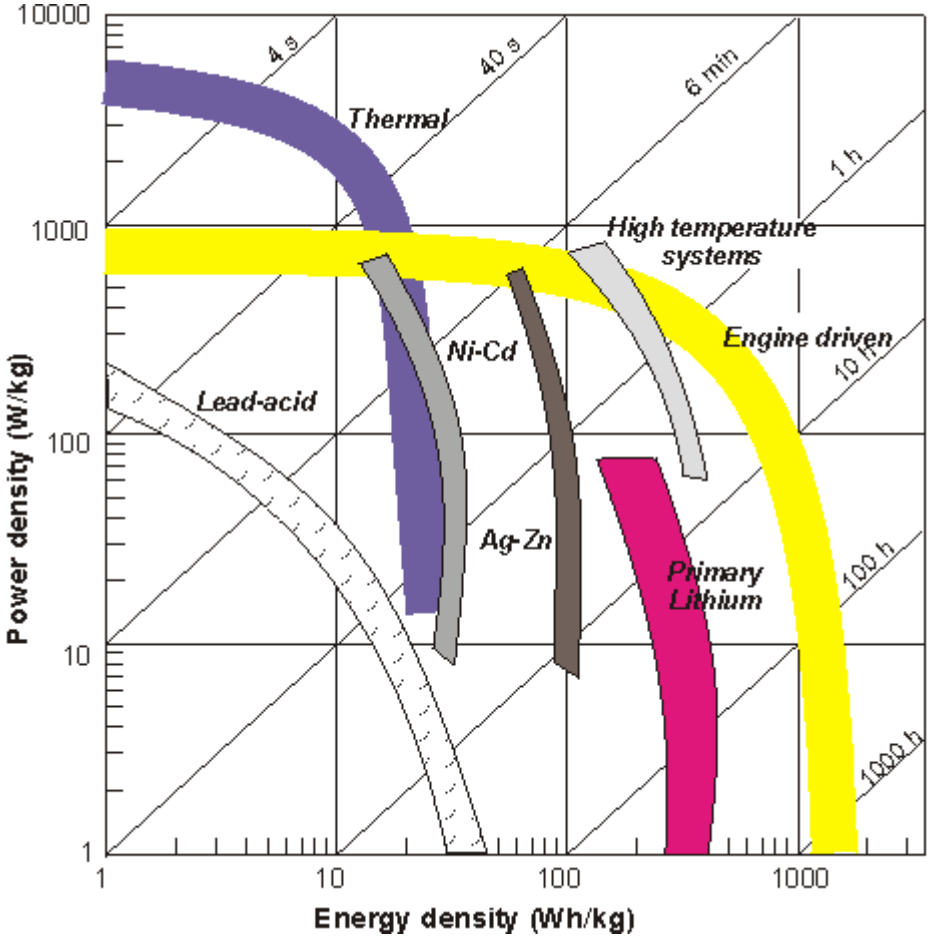


Figure 23. Ragone plot illustrating the energy-power characteristics of various electro-chemical power sources.

3.4.6 Durability/Cycle Life

Both the capacity and rate characteristics of new batteries changed as it ages (see Figure 24). Changes can be summarized as *loss of chemical capacity* and *impedance increase*. Capacity loss is caused for example because crystalline structure of rechargeable material starts to change [16]. Particles can break up, became disconnected from bulk material and therefore can no longer participate in energy storage.

The cycle life is defined as the number of cycles a cell can perform before its capacity drops to 80% of its initial specified capacity. Every time a battery is charged and discharged, it uses one cycle.

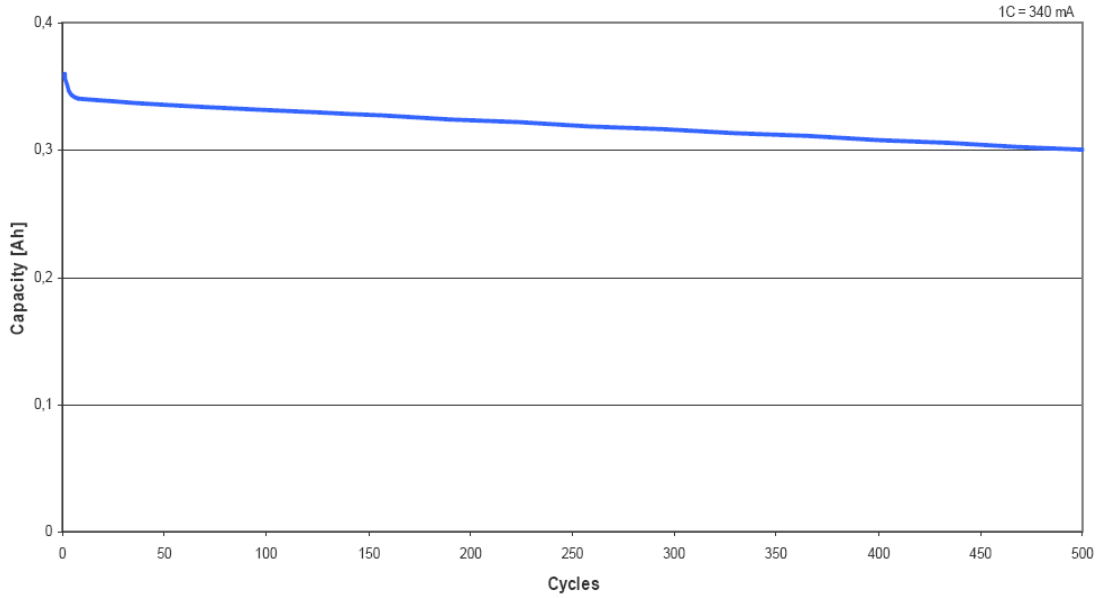


Figure 24. Battery capacity evolution during 500 cycles

Cycle life decreases with increased Depth of Discharge[1] (DOD) and many cell chemistries will not tolerate deep discharge and cells may be permanently damaged if fully discharged. Special cell constructions and chemical mixes are required to maximise the potential DOD of deep cycle batteries.

3.4.7 Shelf Life

Batteries degrade not only during cycling, but also during storage. This makes *cycle number* a rather uncertain estimate of effective battery *age* in view of its performance. Degradation with storage is highly dependent on storage conditions, mostly voltage and temperature.

3.5 Battery Models

Researchers have developed numerous computationally feasible mathematical models that capture battery behaviour in sufficient detail.

3.5.1 Physical Models

Physical models are the most accurate and have great utility for battery designers as a tool to optimize a battery's physical parameters [18]. However, they are also the slowest to produce predictions and the hardest to configure, providing limited analytical insight for system designers.

These models take into account the temperature effect and support for Arrhenius temperature dependence and cycle aging, but they have a high computational complexity.

[1] Discharging a battery to 20 percent or less of its full charge capacity.

3.5.2 Empirical Models

Empirical models are the easiest to configure, and they quickly produce predictions, but they generally are the least accurate. Although they work well in certain special cases, the constants used have no physical significance, which seriously limits their analytical insight.

At present, three different empirical models are used:

- **Peaukert's law:** provides a simple way to account for rate dependence. Though easy to configure and use, Peukert's law does not account for time-varying loads. Most batteries in portable devices experience widely varying loads.
- **Battery efficiency model:** models the battery *efficiency*—the ratio of actual capacity to theoretical capacity—as a linear-quadratic function of the load current [19]. This model accounts for rate dependence and can handle variable loads.
- **Weibull fit model:** use statistical methods to model the discharge behaviour of lithium-oxyhalide cells. With three coefficients fit a Weibull model to express voltage as a function of *delivered capacity,* or charge lost.

3.5.3 Abstract models

Instead of modelling discharge behaviour either by describing the electrochemical processes in the cell or by empirical approximation, abstract models attempt to provide an equivalent representation of a battery. Although the number of parameters is not large, such models also employ lookup tables that require considerable effort to configure. In addition, despite acceptable accuracy and computational complexity, these models have limited utility for automated design space exploration because they lack analytical expressions for many variables of interest.

The most widespread abstract models can be grouped in three different kinds:

- **Electrical-circuit models:** PSpice circuits consisting of linear passive elements, voltage sources, and lookup tables to model nickel-metalhydride and lithium-ion batteries, respectively. Electrical-circuit models are inherently continuous-time and, while their simulation times are faster than those of physical models, they are still time consuming.
- **Discrete-time model:** Using VHDL, Luca Benini and colleagues [20] approximated the continuous-time model shown in Figure 25 to a discrete-time model. This approach incorporates battery voltage dependence on first-order effects—charge state, discharge rate, and discharge frequency—and the second-order effects of temperature and internal resistance.

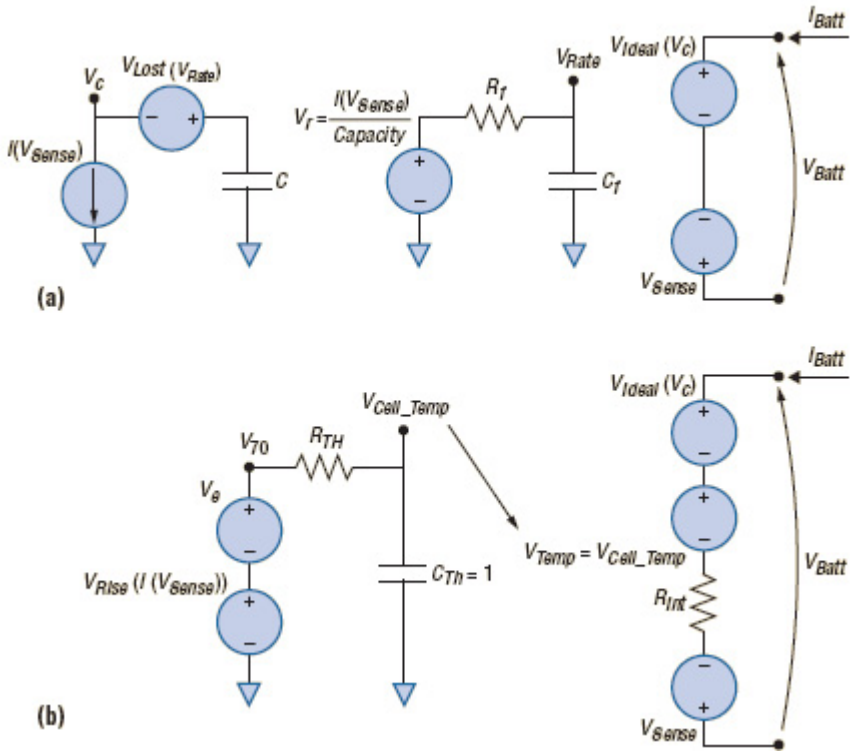


Figure 25. Continuous time model. The model incorporates battery voltage dependence on (a) first-order effects and (b) second-order effects

- **Stochastic Model:** Carla-Fabiana Chiasserini and Ramesh R. Rao [21] developed a battery model, shown in Figure 26, that represents charge recovery as a decreasing exponential function of the state of charge and discharged capacity. Assuming each cell's load to be a pulsed discharge, this model represents discharge and recovery as a transient stochastic process.

  Each discharge demand of *i* units causes a transition to *i* states lower, while rest periods cause state transitions to successively higher states. Capacity gain is expressed as $G = Acu/N$, where *Acu* represents the average number of charge units and *N* is the *nominal capacity*—the charge extractable by a constant load. This model is useful for representing pulsed discharge, as it can obtain capacity gain for different types of stochastic loads analytically without simulation; Chiasserini and Rao [21] reported several analytical results related to distributing the load between two cells of a battery package. However, because it concentrates only on charge recovery, their model does not account for other battery

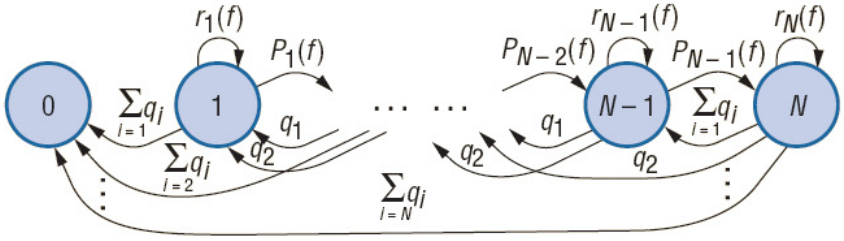


Figure 26. Stochastic Cell Model

3.5.4 Mixed Models

Some models combine a high-level representation of a battery for which experimental data determines the parameters with analytical expressions based on physical laws [22].

For example, it has been developed a high-level analytical model that characterizes a battery using two constants, α and β, derived from the lifetime values for a series of constant load tests. The α parameter is a measure of the battery's theoretical capacity, while β models the rate at which the active charge carriers are replenished at the electrode surface.

Starting with Faraday's law for electrochemical reaction and Fick's laws for concentration behaviour during one-dimensional diffusion in an electrochemical cell, these researchers obtained the following expression relating the load *i*, battery lifetime *L*, and battery parameters:

$$\alpha = \int_0^L i(\tau) \cdot d\tau + \lim_{\varepsilon \to 0^+} 2 \sum_{m=1}^{\infty} \int_0^{L-\varepsilon} i(\tau) \cdot e^{-\beta^2 m^2 L}$$

The first term represents the charge the load consumed over the period [0, *L*), while the second term represents the charge that was "unavailable" at the electrode surface at the time of failure *L*. The unavailable charge models the effect of the concentration gradient that builds up as the flow of active species through the electrolyte falls behind the rate at which they discharge at the electrode surface.

Battery lifetime predictions using this model closely match both simulation results and experimental measurements. The simulation time is moderate, and the authors point out that it is possible to trade off accuracy with speed by reducing the number of terms in the summation and approximating the continuous-time load waveform *i*(*t*) to an *N*-step staircase (in the extreme case of a constant load approximation, *N* = 1). However, the model does not account for the effect of temperature and capacity fading on the discharge characteristics. Compared to the stochastic model, it has higher computational complexity but requires less configuration effort and offers more analytical insight.

4 Energy Management Architecture Design

Currently, most wireless sensor networks are designed for low-power operation, although the knowledge about the actual current consumption is limited. At present, a standard architecture to manage the sensornets energy does not exist [23]. In order to settle this matter once and for all, an architecture which places energy as the foremost design consideration is necessary. This architecture must arbitrate intelligently the usage of the energy, allowing accounting of energy usage and predictable failure despite a fluctuating environment. Moreover, this architecture must allow the user to control the network quality-of-service and the location of energy.

Taking into account these requirements, an architecture consisting of three basic components (see Figure 27) is proposed:

- Energy Monitor: to supervise the current state of charge of the node's energy sources.
- Policy Specification: to allow the user to dictate sensoring rates, desired lifetime or quality-of-service, despite dynamic changing environment conditions.
- Energy Manager: to perform admission control on activities based on the energy information received from the energy monitor to enforce user policies.

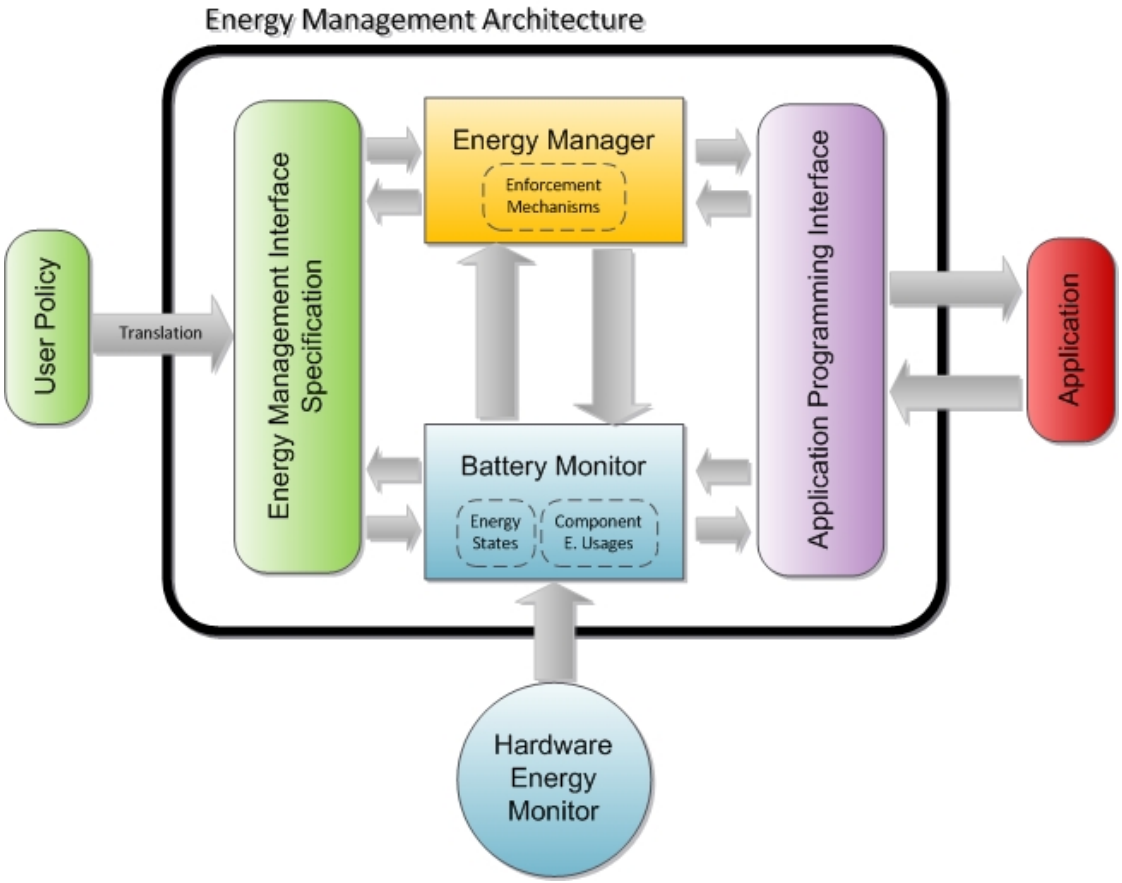


Figure 27. Energy Management Architecture Scheme

In this chapter the main points that have been considered in order to achieve the requirements of the proposed architecture are explained.

4.1 Energy Monitor

In order to have a real-time view of the energy states and, therefore, of the nodes energy consumptions which form the network a component is necessary which can provide all this information with great accuracy. Once our architecture has at its disposal this accurate information, it could apply energy-related optimization at different levels: application, routing protocol, MAC protocol, etc.

According to the type of energy source that the nodes are using, the monitor should be able to get the system energy levels. In sensor networks, the most used energy source is a chemical battery, although recent researches are considering other kind of energy sources such as super-capacitors and photovoltaic-cells. Hence, the monitor gives the component energy usage and the energy levels according to the battery type and the specific application.

Periodically, the monitor reports the component usage, such as microprocessor states or radio usage. These reports provide the user a fine-grained view of energy usage and make it possible to apply a control of the different activities by means of a policy specification. At this point, the correlation between different activities becomes a problem, as it is not clear who is responsible of the energy consumption. There have been several research efforts that attempt to provide some notion of grouping. They may be adapted to providing energy accounting in different ways, varying on flexibility, code reusability, and complexity. The suitable choice depends on the application and programming paradigm.

Besides, the actual energy in the energy system should also be reported. The monitor should provide information about the state of charge of the battery. In order to do that, the monitor must be aware of the battery characteristics. Thus, all the data that the battery manufacturer could provide should be known by the monitor in order to estimate the remaining battery energy.

4.1.1 Battery Selection

The steps for selecting a type of battery to use it as power supply require a meticulous study not only of the application or the wireless sensor node but also of the environment where the node is supposed to be working.

According to the specifications of the AquisGrain 1.0 platform the characteristics that the battery should achieve are:

- Voltage Range: 1.8-5.5 V
- Intermittent load or pulse load: 5 mA - 31 mA

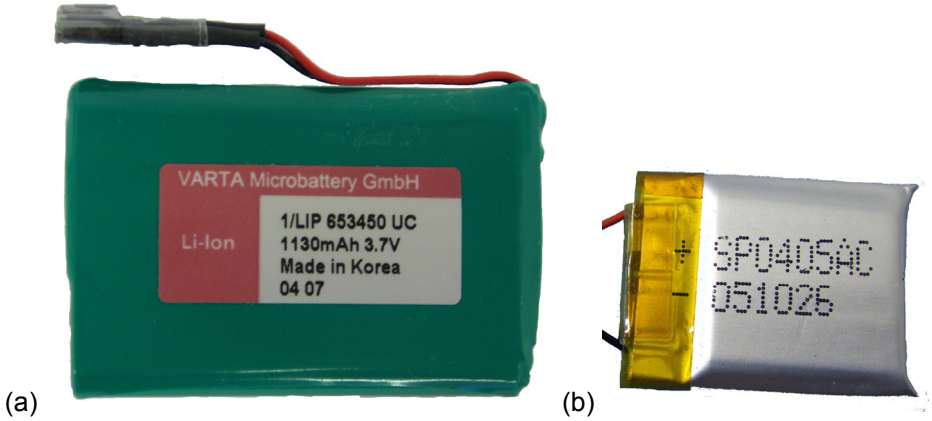
In order to facilitate the testing, a secondary battery is used, so additional requirements would be a rapid charging at ambient temperature and normal atmosphere conditions. Besides, as the ECG application is supposed to be running in the Intensive Care Unit (ICU) in a hospital, the working temperature is an average of 24 ºC [24].

The lifetime of the battery should be as long as possible, although according to the ECG application as the average length of stay (LOS) in the ICU is 71 hours, so three days is the minimum lifetime [24]. On the other hand, the dimensions, the weight and the shape of the battery, like in other wireless sensor applications, are an important constraint. The ideal battery is as small and light as possible, so a compromise between capacity and dimension should be achieved. Nevertheless, the preferred battery pack is the prismatic cell as it could be attached to the AquisGrain 1.0 platform easily.

Rechargeable batteries for portable devices are available in two main chemistries (see chapter 3 for other unusual chemistries): Nickel-cadmium, and lithium-ion. Each has its advantages and disadvantages: Nickel-cadmium is the traditional battery of choice for the medical industry because it is rechargeable, safe, reliable, inexpensive and has a constant discharge voltage. Lithium ion and nickel metal hydride are relatively new to this field and their benefits include: light-weight, rechargeable, higher cell voltage, greater capacity, and better environmentally.

On the positive side, in lithium ion batteries there is more capacity and voltage for less weight. For instance, a nickel metal hydride a battery weighs around 68 g and has a capacity of 270 mAhr. The Li-ion replacement weighs only 51.4 g but has a 560 mAhr capacity. For the user, this represents a battery with more than twice the capacity for only 76% of the weight. On the negative side for the Li-ion chemistry is the variable voltage 4.2 V at full charge, only 2.5 V at full discharge and the need for a discharge protection circuit.

The fact that the voltage does not remain constant during the lifetime of the battery will help to provide an algorithm to estimate the state of charge of the battery. For these reasons, lithium ion batteries are the best option for our application.



(a) (b)

Figure 28. (a) Selected Varta Battery (b) Selected Lishen Battery

For our testing application two different types of Li-Ion batteries were selected:

1. **Lishen SP0405AC:** is a Lithium Polymer battery and it is really small (Size is 4x20x25 mm), it has 140 mAh capacity and its weight is about 3 g (see Figure 28.b). Therefore, it is a good battery for this testing the application because of its small capacity. Lishen SP0405AC's output power is 1 W so that is good enough to power up the AquisGrain 1.0 platform.

2. **Varta 653450UC:** is a Lithium Ion battery whose dimensions are bigger (Size is 6.6x34x49.8 mm), it has 1130 mAh capacity and its weight is about 24.5 g (see Figure 28.a). With these characteristics, it is ideal for attaching it to the node, to run for more than a week with the ECG application.

4.1.2 Battery monitoring technology & selected method

According to the type of battery, there are several different methods to monitor the different elements which can be useful in order to determine the state of charge of a battery. For instance, deep discharge, wet and gel electrolyte, lead-acid batteries are the standard power source for electric powered bicycles. Most power indicators for this application are simple voltmeters. They are widely used because they are inexpensive, robust and compact. However, voltmeter based monitors are at best inaccurate and at worst misleading indicators of remaining battery capacity.

High-end battery monitoring equipment produces instantaneous variations in state-of-charge readings, in response to loads placed on the battery during typical daily use. For example, the computer industry uses a *SMBus* for monitoring a battery's status. In this case, both the power monitoring and charging systems are paired with the battery technology. The power monitor tracks the flow of current into and out of the battery. The battery's full charge level is also tracked, so the power monitor can assess battery degradation over time. This monitoring/charging system has the additional feature of being programmable, so that key parameters can be entered and the charging system accurately matched to battery performance specifications.

For our purpose, the aim is quite similar to this one. The monitoring for the ECG application should accurately measure the battery voltage and the total charge flowing out of the battery.

4.1.2.1 Current Measurement

At present, engineers and scientists use different methods to measure the current consumptions under a specific load. There are three technologies that are typically used for measuring current: sense resistors, current transformers and Hall effect sensors.

4.1.2.1.1 Sense Resistors

Sense resistor is simply a resistor placed in series with the load. By Ohm's law, the voltage drop across the device is proportional to the current. For low currents, this provides very accurate measurements given the resistance value has a tight tolerance.

Although sense resistors with high performance thermal packages have been developed for larger currents, they still result in insertion loss. In addition, they do not provide a measurement isolated from transient voltage potentials on the load. Sense resistors also require other circuitry such as instrumentation amplifiers to generate a distinguishable signal.

That simple current sensing technique serves practical applications:

- High-Side Current-Sense Amplifiers: Resistor-based current sensing is simple, easy to use, low cost, extremely linear and requires no calibration. The statement that voltage across a resistor is directly proportional to the current through is commonly known as Ohm's Law: V=I·R.

  As a caveat, note that all resistors dissipate power when current passes through them. Because the dissipation produces heat that, in turn, affects the resistance, power dissipation in a sensing resistor must be carefully assessed. A larger sense-resistor value yields better accuracy, but dissipates more power: $P = I^2R$, where *I* is the measured current and *R* the sensing resistance.

  The magnitude of the measured current is application-specific rather than a design parameter, so the sensing-resistor value should be as low as possible to minimize "joule heating". Choosing a small-valued sensing resistor yields a low-level sensing voltage across it, requiring an amplifier to boost that voltage to a level suitable for interface to a comparator, analog-to-digital converter or other external circuit.

  However, low sensing voltages are vulnerable to the measurement error induced by the inherent bias current and input-offset voltage of amplifiers. For example, a practical full-scale sensing voltage might range from 50 mV to 200 mV. If the amplifier's maximum input-offset voltage is ±5 mV, the measurement error is ±10% at 50 mV (full scale) and even worse at lower currents.

  The current-sensing amplifier must have low input-offset voltage and low input-bias current. A dedicated high-side current-sense amplifier (see Figure 29) places a current-sensing resistor between the voltage source (a battery, for instance) and the load.

  By avoiding extraneous resistance in the ground plane, that arrangement greatly simplifies the PCB layout and generally improves the overall circuit performance. Current passing through the sense resistor (R_{SENSE}) develops a voltage drop, which is sensed by the op amp and drives the MOSFET transistor to sink current through R. The voltage drop across R equals the voltage across the sensing resistor:

  $$K \cdot I_{SENSE} \cdot R_{SENSE} = I_0 \cdot R \qquad \text{or} \qquad I_o = K \cdot I_{SENSE} \cdot \left(\frac{R_{SENSE}}{R} \right).$$

  Thus, $V_o = K \cdot I_{SENSE} \cdot \left(\frac{R_{SENSE}}{R} \right) \cdot R_o$

  The sensor output current is proportional to the load current. Typically, a current mirror is included to increase the output current by a factor of K. If you require a voltage output, convert the current to voltage by placing an output resistor (R_O) between the current output and ground. Resistors R and R_O can easily be factory trimmed to achieve a current-sensing accuracy of 1% or better.

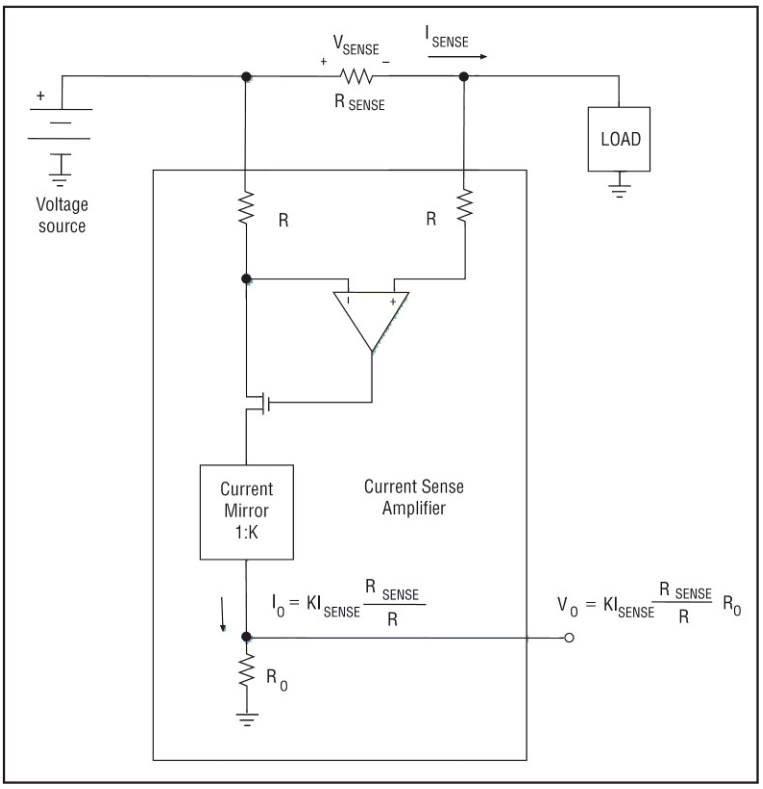


Figure 29. High-side current sensor for current measurements using Ohm's Law

- Current Monitor and Protection: High power-supply circuits often incorporate shortcircuit or overload protection (Figure 30.a). The IC shown integrates a reference, comparator and latch. R1 and R2 set the trip current. The comparator compares the current-sensor output voltage with the reference voltage. When load current reaches the allowed maximum, the comparator output turns off the p-channel MOSFET switch by latching to logic high. No current flows to the load. The p-MOSFET remains off until a reset is applied or the power supply is toggled.

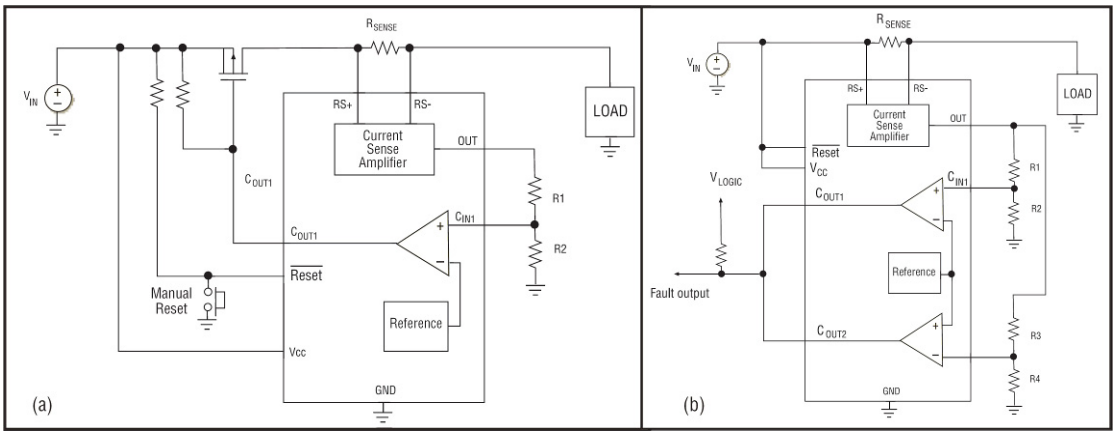


Figure 30. Schematics protection circuit with (a) a p-channel MOSFET or (b) A current-window, circuit comprising R1-R4, the comparators and reference.

Battery chargers and other applications must guard against overcurrent due to short circuits and undercurrent due to open circuits. For this purpose, the current-window detector circuit (Figure 30.b) but includes

a second comparator for monitoring undercurrent. The two comparator outputs are open-drain, and therefore can be “wire-OR’d” together or can remain separate outputs. When the monitored current falls outside of the window, the IC alerts the system by asserting a fault condition.

- Fuel Gauges and Battery Management: The current-sensing amplifier shown before is a relatively simple and general-purpose device. Specific applications such as fuel gauging and battery management, on the other hand, require additional functions and features to be integrated on-chip.

  Fuel gauging is important for battery applications, in which the battery capacity is monitored with precision to optimize the system performance and prolong battery life. For example, the battery pack in a laptop computer often integrates a smart fuel gauge for monitoring and supervising the charging and discharging.

  As it is shown in Figure 31, such gauging devices usually have a digital coulomb counter that tracks the accumulated charge and discharge actions. Thus, a given battery is fully charged when it accepts a certain amount of charge (in coulombs, C). Similarly, a battery is empty (discharged) when a given amount of charge is taken from it.

  Thus, the time integral of current equals the total charge. A current-sensing amplifier measures battery current, and the coulomb counter acts as a time integrator that accounts for the total charge flow during a charge or discharge cycle.

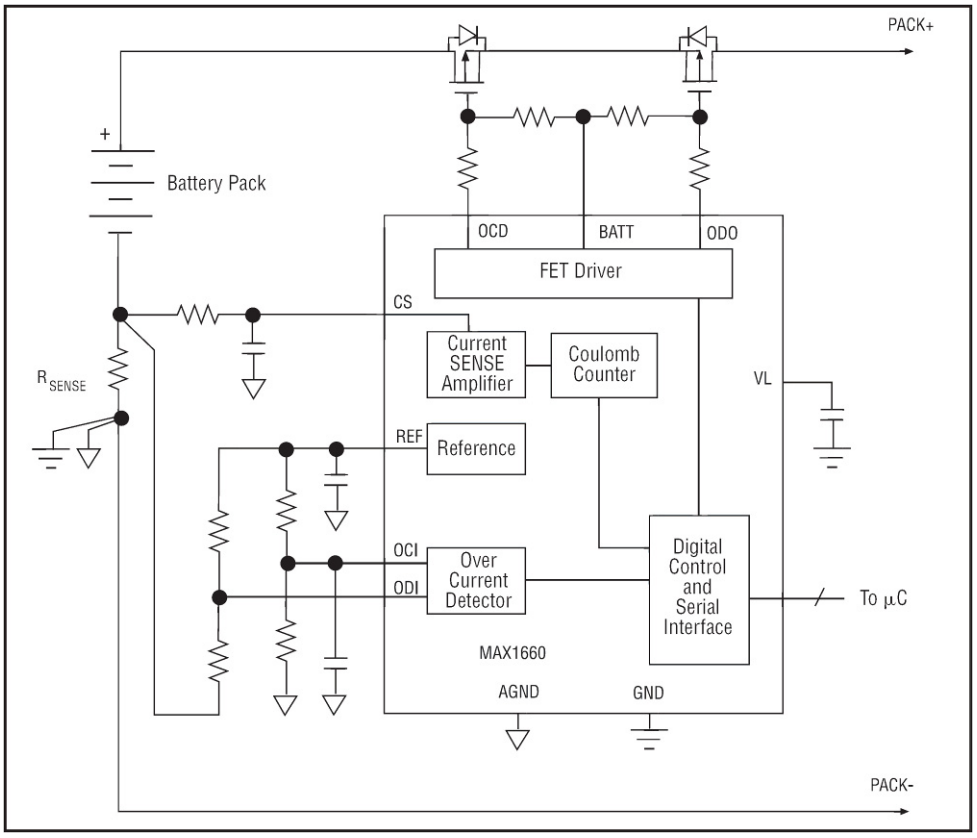


Figure 31. Fuel-gauging device that tracks charge/discharge currents

The current sensor for fuel-gauging applications requires a bidirectional current-measurement capability. When charging a given battery pack, the maximum charge is set by the user. When the coulomb counter reaches the set value, it alerts the microcontroller to stop charging because the battery is fully charged.

Similarly, during discharge due to normal batter y use, the gauge functions as a fuel gage that informs the user how much battery capacity remains. When the coulomb counter reaches a set minimum limit, it prevents over-discharge by alerting the microcontroller that the battery is empty. Thus, the coulomb counter prolongs battery life by preventing excessive charging or discharging.

The current sensor also provides overload and short-circuit protection by continuously monitoring current flow. By shutting off the MOSFETs in response to a short circuit, the current-sense amplifier disconnects the battery to protect it against shortcircuit faults.

4.1.2.1.2 Current Transformers

Central to all of the AC power transducers is the measurement of current. This is accomplished using a current transformer (CT), a "donut" shaped device through which is threaded the wire whose current is to be measured (see Figure 32).

A current transformer is a type of "instrument transformer" that is designed to provide a current in its secondary which is accurately proportional to the current flowing in its primary.

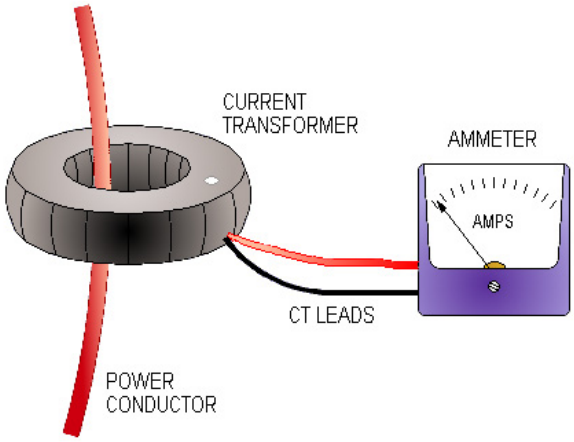


Figure 32. Simple current transformer circuit

Current transformers are designed to produce either an alternating current or alternating voltage proportional to the current being measured. The current transformers used with the Wattnode transducers produce a 333 mV alternating voltage when the rated current is measured (either 30 A, or 50 A). The OSI power transducers employ CT's that produce 5 V output at rated value.

Current transformers measure power flow and provide electrical inputs to power transformers and instruments. Current transformers produce either an alternating current or alternating voltage that is proportional to the measured

current. There are two basic types of current transformers: wound and toroidal. Wound current transformers consist of an integral primary winding that is inserted in series with the conductor that carries the measured current. Toroidal or donut-shaped current transformers do not contain a primary winding. Instead, the wire that carries the current is threaded through a window in the toroidal transformer.

Current transformers have many performance specifications, including primary current, secondary current, insulation voltage, accuracy, and burden. Primary current, the load of the current transformer, is the measured current. Secondary current is the range of current outputs. Insulation voltage represents the maximum insulation that current transformers provide when connected to a power source. Accuracy is the degree of certainty with which the measured current agrees with the ideal value. Burden is the maximum load that devices can support while operating within their accuracy ratings. Typically, burden is expressed in volt-amperes (VA), the product of the voltage applied to a circuit and the current.

There are a variety of applications for current transformers. Some devices are used to measure current in electronics equipment or motors. Others are used in street lighting. Current transformers with small footprints mount on printed circuit boards (PCBs) and are used to sense current overloads, detect ground faults, and isolate current feedback signals.

Larger devices are used in many three-phase systems to measure current or voltage. Commercial class current transformers that monitor low-power currents are also available. Some current transformers are weatherproof or are rated for outdoor use. Others meet MIL-SPEC, ANSI C-12, or IEC 1036 standards. Generally, ANSI class devices are intended for power monitoring applications where high accuracy and minimum phase angle are required.

4.1.2.1.3 Hall Effect Sensors

When a current-carrying conductor is placed into a magnetic field, a voltage will be generated perpendicular to both the current and the field. This principle is known as the Hall effect [25].

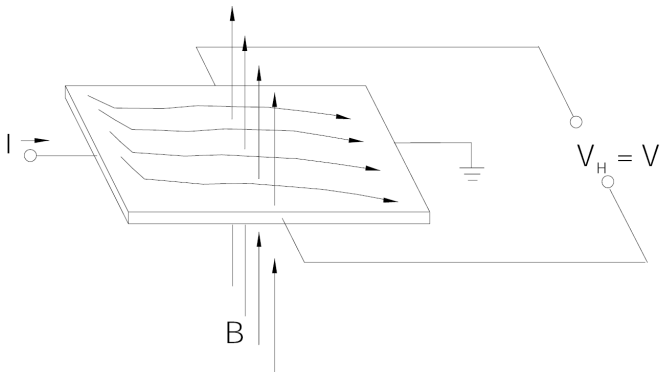


Figure 33. Representation of the Hall effect and its electrical parameters

Concentric magnetic field lines are generated around a current carrying conductor (see Figure 33). Approximating the primary current conductor as infinitely long, the magnetic field strength may be defined $B = \mu_o I / 2pr$, where μ_o is the permeability of free space, I is the current and r is the distance from the

center of the current conductor. In order to induce a larger signal out of the Hall element; the current conductor may be wrapped around a slotted ferrous toroid *N* number of times, such that $B \approx 6.9NI$.

In an open loop topology (see Figure 34), the Hall element output is simply amplified and the output is read as a voltage that represents the measured current through a scaling factor [26].

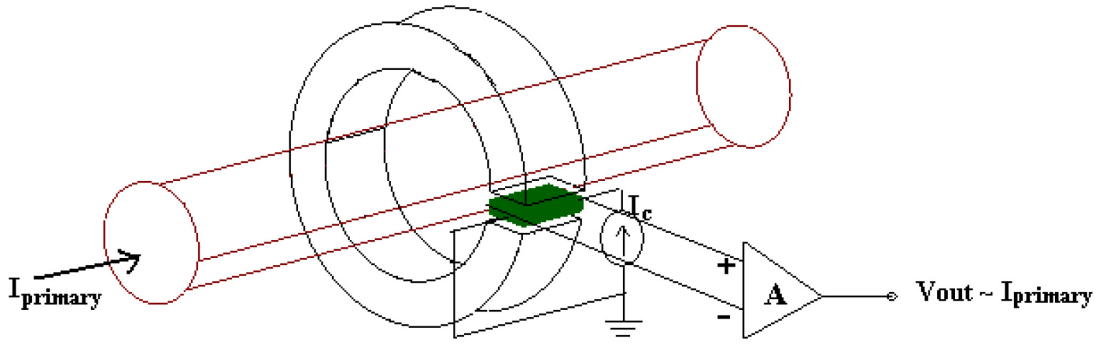


Figure 34. Basic Topology of Open Loop Hall Effect Current Sensor

In a closed loop topology (see Figure 35), the output of the Hall element drives a secondary coil that will generate a magnetic field to cancel the primary current field. The secondary current, scaled proportional to the primary current by the secondary coil ratio, can then be measured as voltage across a sense resistor.

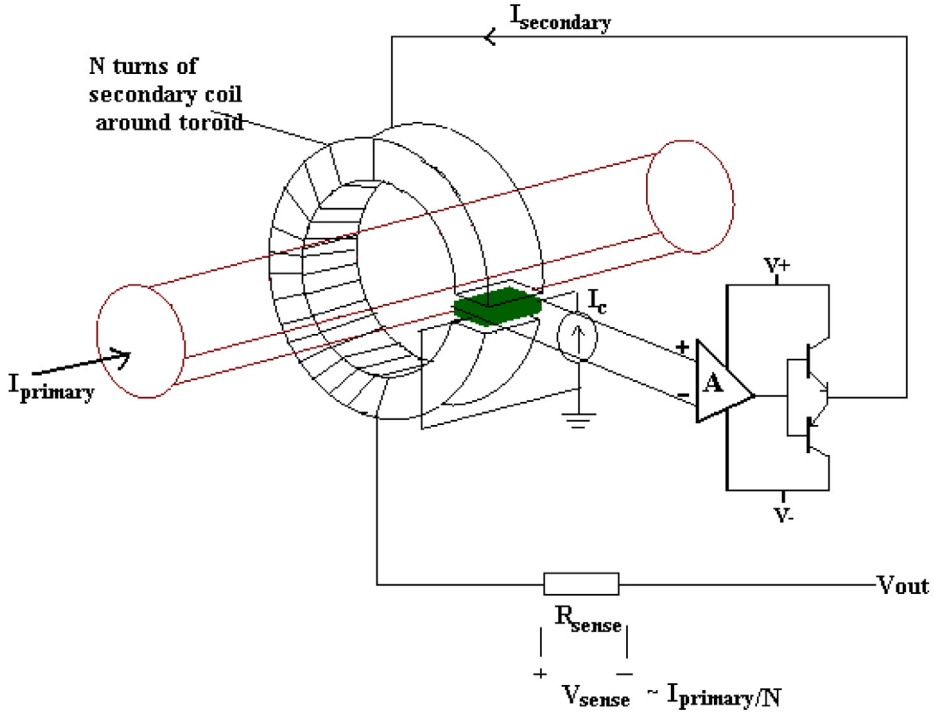


Figure 35. Basic Topology of Closed Loop Hall Effect Current Sensor

By keeping the resultant field at zero, the errors associated with offset drift, sensitivity drift and saturation of the magnetic core will also be effectively cancelled. Closed-loop Hall effect current sensors also provide the fastest

response times.

However, with a secondary coil that may be needed to drive up to several milliamps of current, power consumption is much higher in closed loop Hall effect devices, than open loop topologies. The closed loop configuration also limits the magnitude of the current that can be sensed since the device may only drive a finite amount of compensation current. Table 3, provides a simple comparison of different current sensing techniques.

After exposing the different technologies to measure current, the best one for the AquisGrain 1.0 platform and, therefore, for the ECG application is a sense resistor. With a sense resistor it is possible to achieve an accurate current measure at a very low cost. Moreover, it would be easy to integrate to the platform without significant changes in the PCB design.

Current-sensing technique	Accuracy	Galvanic isolation	Power dissipation	Relative cost	Typical current range
Sense resistor	>95%	None	High	Low	<20 A, DC – 100 kHz
Transformer	~95%	Yes	Moderate	Med.	Up to 1000 A, AC
Open loop hall effect sensor	90-95%	Yes	Low	Med.	Up to 1000 A, DC – 20 kHz
Closed loop hall effect sensor	>95%	Yes	Moderate - High	High	<500 A, DC – 150 kHz

Table 3. Current sensing techniques comparison

However, with this data the user has not enough information to estimate the remaining capacity or range of the battery. In order to have a quite accurate estimation of the remaining capacity of the battery it is necessary to add something more than a simple current sensor.

4.1.2.2 Battery Capacity Monitoring

For years, portable device system designers have used the battery voltage measurement method to determine the capacity of a battery in a given application. Although this method has been effective, it lacks accuracy. Because of the wide range of load characteristics among handheld devices, voltage measurement alone cannot determine capacity when the same device may have different power modes. A cellular phone that is browsing through different system menus is in a lower power mode than when it is accessing wireless features. The battery voltage can vary while at the same capacity level due to temperature, impedance differences, and discharge current.

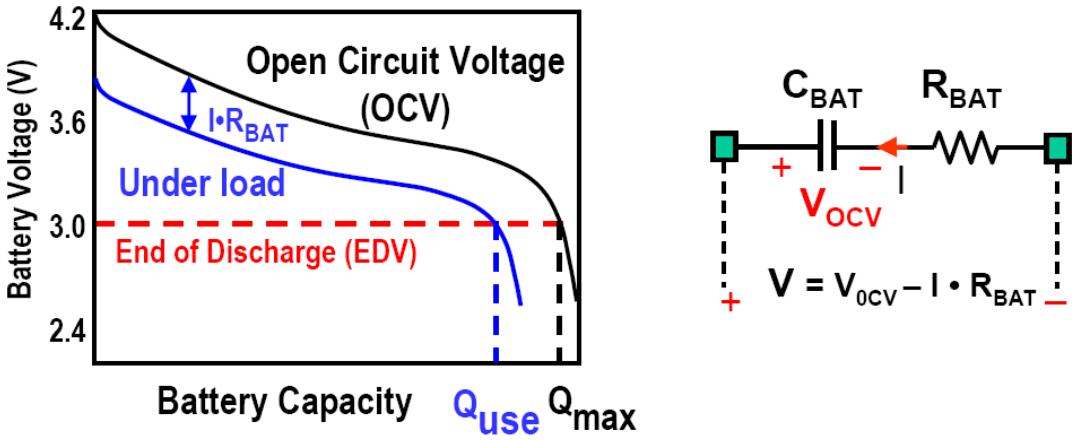


Figure 36. Battery modelling parameters.

As commented in the previous chapter, the end of discharge will be reached earlier for higher discharge current, therefore the useable capacity Q_{USE} is less than the maximum capacity (see Figure 36). This way, it is possible to define the true full charge capacity as the capacity usable.

If voltage measurement is an inaccurate scheme to determine remaining battery capacity, then on other approach is to measure the charge added or removed for determining the capacity change. It is based on measuring the amount of energy that flowed into or out from a battery. This is accomplished by constantly monitoring the current that is being supplied from a charger or drained by a load.

Given that current is related to charge as $i(t) = \frac{dq(t)}{dt}$, by measuring the integral of current over time, the capacity being added or depleted from the battery is known. Accurate tracking of the battery's capacity requires that something monitors the current activity of the battery at all times. During any time that the current is not being measured, the net amount of capacity being depleted from the battery may be unaccounted. This could result in false expectations of how much energy remains in the battery for use in a given handheld device. The main processor of a handheld system should not be tasked with measuring current to accurately report battery capacity. Rather, an independent system can measure the current and process this information so that the main processor can use it to determine power management functions and also provide battery/handheld device information to the end-user.

Battery gas gauging is the process to collect data such as current, voltage, and temperature and then use this data to make decisions concerning power management and battery capacity reporting. A gas gauge system requires at least two components. As a minimum, one subsystem must collect current, voltage, and temperature data, and another subsystem analyzes this data to obtain the desired battery conditions. A gas gauging algorithm running in a microprocessor is used to analyze the data. This algorithm is complex, due to the various characteristics of batteries that must be considered. Self-discharge and aging of batteries also must be considered when implementing gas gauging. A battery that is at rest may show decreased capacity after an extensive time because of self-discharge. As a battery ages, the impedance is

higher, and hence decreases the total battery capacity that can be used.

The most common technique used today for battery capacity monitoring is coulomb counting, tracking the current into and out of the battery. This works well with a freshly-charged battery but it is not without problems. That's because every battery wears out, or ages, according to its usage pattern. For instance, if a battery pack is charged and left unused for several days, or is never fully-charged because of several charge and discharge cycles, self-discharge becomes more evident through internal chemical reactions. There is no way to measure self-discharge, so it is corrected roughly, using a pre-defined equation of state-of-charge, temperature and the battery's cycling history. This varies from one battery model to the next.

Another problem with coulomb counting is that the value of the total capacity is updated only if a full discharge happens immediately after a full-charge. The average user of a portable device rarely fully discharges the battery, so the actual capacity may be considerably decreased before it is updated. The gas gauge incorrectly estimates available capacity during these periods, sometimes by as much as 50 percent.

A second option uses the correlation between battery voltage and remaining capacity. It seems to be straightforward, but battery voltage correlates with capacity only if the voltage has relaxed: any load has to be removed prior to the measurement as when a load is applied, there is a voltage drop caused by the battery's internal impedance.

4.1.2.2.1 Voltage-based techniques

By subtracting the voltage drop, calculated as current multiplied by resistance (I·R), from the measured voltage, the gas gauge can use a corrected voltage reading to obtain the current state of charge (SOC). However, as R is more then 10 times higher when the battery is discharged than when charged, using an average value can give an error rate of up to 100 percent in SOC.

One way to address this issue is to calculate the remaining capacity using a multi-dimensional table of voltages listing different loads against the SOC. As resistance can increase about 1.5 times with every 10 ºC temperature decrease this adds another dimension to the table.

Effective *R* will depend on the time of the load application and the transient behaviour of voltage response. However, when the internal impedance is treated like a simple ohmic resistance, without considering the amount of time since the last load change, the calculation will lead to significant errors, even if the R(SOC) dependence is determined by a table. Because the slope of the SOC(V) function depends on the SOC, transient error will range from 0.5 percent in a discharged state to as high as 14 percent in a mid-charged state.

Many new Li-ion battery cells have low-frequency direct current impedance variation of ±15 percent. This makes a significant difference in voltage correction at high loads: at 0.5 C rate a typical DC impedance of 2 Ah cells at 0.15% produces as much as 45 mV difference between cells, with an estimated SOC error of 20 percent.

Finally, the single most significant impedance-related problem occurs when a

cell ages: a typical Li-ion battery doubles its DC impedance in 70 cycles, while its no-load capacity decreases by only two to three percent. A voltage-based algorithm which works for a new battery pack may show up to 50 percent error at only 15 percent of its normal lifespan of 500 cycles.

In conclusion, the gross inaccuracy of the voltage method can be compensated by accounting for three major environmental offsets: temperature, rate of discharge, and voltage level. That, however, requires extensive extra circuitry and concomitant extra cost.

4.1.2.2.2 Current-based techniques

Current-based fuel gauging is more complex, at least in concept. It requires precise analog-to-digital conversion circuits and a low-value, high-precision sense resistor to cause a small voltage drop in the power path from the battery.

Current is measured both going to the cell and again leaving the cell through the sense resistor. The measurements are accumulated over time in memory and used to determine the remaining capacity-in effect, providing a running total of the amp/hours remaining.

While the current-based gauge is initially more complex than voltage based, it is inherently more accurate. Controlled experiments of the uncompensated current gauge show an error rate of around 10%. The inaccuracy derives from two of the environmental effects noted above: temperature and discharge rate. Compared to compensating for voltage-based errors, compensating for these effects in a current-based gauge can yield even greater accuracy at an ultimately lower cost.

At present, the AquisGrain 1.0 platform incorporates a battery monitor, the Maxim Dallas DS2436 Battery Identification/Monitor Chip [27]. In the current framework the DS2436 is only used to get a unique 64 ROM ID which is used as MAC address of the node. However, this device also performs the essential function of monitoring battery temperature, without the need for a thermistor in the battery pack. A cycle counter assists to determine the remaining cycle life of the battery. The DS2436 measures battery voltage that could be used to determine the end-of-charge or end-of-discharge or basic fuel gauge operations.

However, with only this information it is not feasible to get an accurate estimation of the state of charge of the battery. So, taking into account the different techniques to monitor the battery capacity and according to the requirements of the ECG application it is necessary to include a new integrated circuit to the AquisGrain 1.0 platform not only for fuel gauging but also to determine the battery non-linear effects. Then, we will merge the advantages of the current measuring technique and the voltage measurement technique. As result, we will achieve an accurate estimation of the battery's state of charge.

There are two main manufacturers of fuel gauging IC's: Texas Instruments and Dallas Semiconductor. The desired requirements used for searching were:

- Compatibility with 1-wire interface [28], in order to keep the application framework which uses the DS2436 to get the MAC address, and avoid-

ing to use some extra port.

- Battery type: Li-Ion, Li-Ion Polymer.
- At least 32 EEPROM, to store battery parameters and application data if necessary.
- Current, Voltage and Temperature measurement.
- Package 8TSSOP and unique ID, to make it compatible with the DS2436 shape and the features that AquisGrain 1.0 applications are using.

As shown in the tables in the Appendix A, Texas Instruments has seven microcontrollers which fulfil the requirements. However, the communication interface of six of them is one wire but at high speed. As all one wire devices on a multimode bus must operate at the same communication speed for proper operation and the DS2436 operates at normal communication speed, the only candidate integrated circuit from TI that operates at the speed is the BQ2023.

On the other hand, Dallas Semiconductor has four ICs which fulfil our requirements and they can operate on both communication speed modes. Carrying out a conscientious comparison of the five possibilities (four from DS and one from TI), we realized that some of the ICs are able to perform on-chip remaining capacity estimations. Although this was not our requirement, under the point of view of energy management it is optimal to carry out as less processing algorithms in the node as possible, so if some IC are able to perform capacity estimation by themselves that saves some energy. Only the Dallas Semiconductor DS278x series performs on-chip remaining capacity estimations. In that series, two ICs fulfil the desired requirements: DS2781 and DS2780. The only difference between these ICs is that the DS2781 is designed for operating with two cells and therefore at a higher voltage level. As the AquisGrain 1.0 platform is designed to be powered at a maximum voltage of 5.5 V, the DS2780 is enough for the node.

Figure 37. DS2780 Integrated Circuit

Summarizing, our AquisGrain 1.0 platform will be powered with one Li-Ion battery or one Li-Polymer battery. For estimating the remaining capacity and the energy consumption of the application running on the platform, the DS2780 IC will be attached to the battery. This circuit will provided all the necessary information for the energy management architecture and, besides, will perform on-chip remaining capacity estimations.

4.1.3 MAXIM DS2780 Standalone Fuel Gauge

The DS2780 provides high-precision current-flow measurement data to support battery-capacity monitoring in cost-sensitive applications. Moreover, it measures voltage, temperature and estimates available capacity for rechargeable Lithium Ion and Lithium Ion Polymer batteries. It was initially designed for battery monitoring, but can be used to monitor consumption of any device. Through its Dallas 1-Wire interface, the DS2780 allows the host system access to real-time current and accumulated data. Each device has a unique factory-programmed 64-bit address that allows it to be individually addressed by the host system. The interface can be operated with standard or overdrive timing. Figure 38 illustrates the block diagram of the MAXIM DS2780 coulomb counter [4].

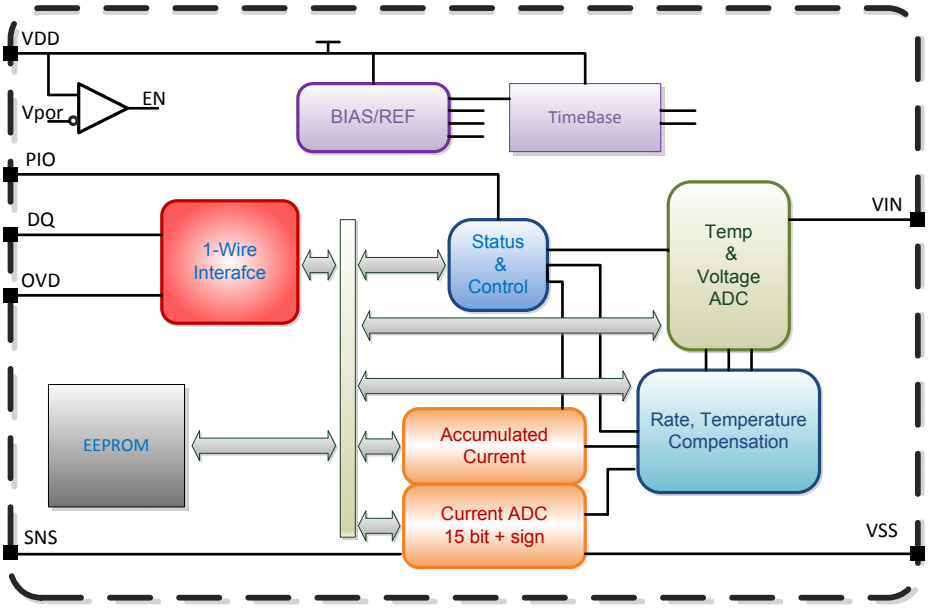


Figure 38. DS2780 block diagram

The DS2780 has two power modes: active and sleep. While in active mode, the DS2780 operates as a high-precision coulomb counter with current and accumulated current measurement blocks operating continuously and the resulting values updated in the measurement registers. Read and write access is allowed to all registers and the PIO pin is active. In sleep mode, the DS2780 operates in a low-power mode with no current measurement activity. Serial access to current, accumulated current, and status/control registers is allowed if VDD > 2 V. For our application ECG, the DS2780 will operate always in the active mode, as we need the capacity algorithm to be continuously running.

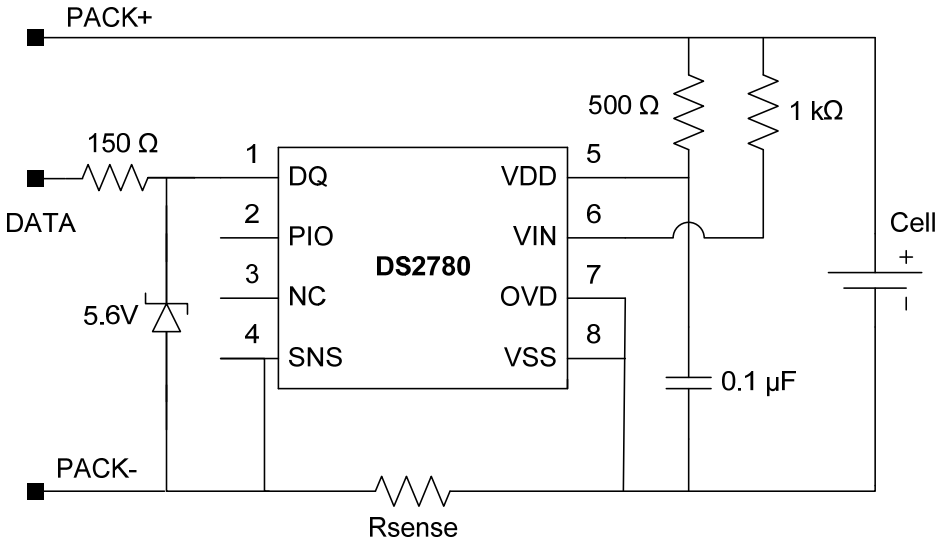


Figure 39. DS2780 designed operation circuit

The schematic of Figure 39 illustrates a possible solution for using the DS2780 in the host system of a wireless application. The pack could be a single-cell Lithium or several of them in series. The PACK+ and PACK- terminals represent the connection to the external battery pack. The 150 Ω resistor from PACK+ to VDD of the DS2740 is for ESD immunity.

The resistor helps to limit current spikes into the part, and protect against over-voltage conditions. The other passive component on the VDD line is the 0.1 µF capacitor. This capacitor helps to filter voltage spikes and keep the voltage within the specified 2.7 V to 5.5 V range.

According to the schematics, the DS2780 allows low current sensing as the current sensor element is connected between the load and ground. Current is measured by looking at the voltage drop across a resistor placed between the load and ground. The main advantages of this method are that is straightforward, easy, and inexpensive and precise. Nevertheless, it adds undesirable resistance in the ground path and may require an additional wire to the load that could otherwise be omitted.

Material	Resistivity	Temperature coefficient
Copper	1.7 x 10^{-6} Ω·cm	0.004041
Nickel	7.8 x 10^{-6} Ω·cm	0.005866
Platinum	10 x 10^{-6} Ω·cm	0.003729
Manganin	44 x 10^{-6} Ω·cm	0.000015
Nichrome	110 x 10^{-6} Ω·cm	0.00017

Table 4. Resistor materials and its main parameters

Before choosing a precision resistor it must be calculated how much power it will be dissipating. Then, a resistor and heat sink that has the proper wattage rating has to be chosen. The AquisGrain 1.0 platform consumes a maximum of 33 mA. The heating of the shunt resistor due to power dissipation or to

ambient temperature changes affects the measurement accuracy by changing the resistance of the shunt. The various metals used in resistors offer different temperature coefficients (see Table 4).

On the other hand, The DS2780 supports sense resistor values from 3.922 mΩ to 1 Ω. This value is critical to accurate fuel gauge measurements because the DS2780 measures the voltage drop across the sense resistor and divides that voltage by the sense resistor value to calculate the current. Thereby, the maximum power dissipation is:

$$P = I^2 \cdot R = (0.033)^2 = 1.089\,\text{mW}$$

Moreover, the value of the resistor determines the resolution of the current measure. According to the specification the maximum current resolution is 1.5625 µA. This resolution is achieved if a resistor of 1 Ω is placed. The ECG application can consume between 7 mA and 12 mA, with the smallest current variation being of ten miliampers approximately. So, in order to have a resolution good enough to monitor the different power states of the application, a sense resistor of 220 mΩ is placed (see Table 5).

Current Resolution DS2780				
VSS-VSNS	**Sense Resistor**			
	1 Ω	220 mΩ	20 mΩ	5 mΩ
1.5625 µV	1.5625 µA	7.1002 µA	78.13 µA	312.5 µA

Table 5. DS2780 current resolution with different sense resistors.

4.1.3.1 Voltage Measurement

Battery voltage is measured at the VIN input with respect to VSS. It has a range of 0 V to 4.992 V and a resolution of 4.88 mV. The measurement is stored in the VOLTAGE register in two's compliment form and is updated every 440 ms. If the measured voltage is above the maximum register value, then it is reported at the maximum value. On the contrary, voltages below the minimum register value are reported at the minimum value.

VIN is usually connected to the positive terminal of a single cell Lithium-Ion battery via a 1 kΩ resistor. The input impedance is large enough (15 MΩ) to be connected to a high impedance voltage divider in order to support multiple cell applications. The pack voltage should be divided by the number of series cells to present a single cell average voltage to the VIN input.

4.1.3.2 Temperature Measurement

As explained in chapter 2, the influence of the temperature over the capacity has to be taken into account in order to estimate the state of charge of the battery. The DS2780 uses the temperature measurement in its on-chip algorithm to calculate the remaining capacity.

The DS2780 uses an integrated temperature sensor to measure battery temperature with a resolution of 0.125 °C. Temperature measurements are updated every 440 ms and placed in the temperature register in two's comple-

ment form.

4.1.3.3 Current Measurement

In the active mode of operation, the DS2780 continually measures the current flow into and out of the battery by measuring the voltage drop across a low-value current-sense resistor.

The DS2780 has two registers which use the current measurement:

- Current register: stores the current measure. It is updated every 3.515 s with the current conversion result in two's complement form.
- Average Current register: reports an average current level over the preceding 28 seconds. The register value is updated every 28 s in two's complement form, and is the average of the 8 preceding Current register updates.

Current measurements are internally summed, or accumulated, at the completion of each conversion period and the results are stored in the Accumulated Current Register (ACR). The accuracy of the ACR is dependent on the current measurement and the conversion timebase. Charge currents (positive Current register values) less than 100 µV are not accumulated in order to mask the effect of accumulating small positive offset errors over long periods. To preserve the ACR value in case of power loss, it is backed up to EEPROM. The ACR value is recovered from EEPROM on power-up.

The Current A/D of the DS2780 is extremely sensitive. It is capable of measuring a voltage drop of only 1.5625 µV across the sense resistor. This kind of accuracy can only be achieved by calibrating the current measurement after the cell pack is assembled. The Current Offset Register (Address 0x33h) allows the current measurements of the DS2780 to be adjusted for accurate measurement of very small currents. This offset is subtracted internally from each current measurement and is reflected in the Current Register and the Accumulated Current Register.

There can be slight variations in the current offset of the device across the temperature and voltage range of the application. Therefore, it is necessary to calibrate the offset at the average temperature and voltage of the application. For example, the ECG application it is expected to be running in ICU where it would spend the majority of its time at approximately 24 °C and 3.8 V, which would be room temperature and the mid-range of the cell voltage. The DS2780's current measurement gain can be adjusted through the RSGAIN register. Moreover, it is capable of temperature compensating the current sense resistor to correct for variation in a sense resistor's value over temperature.

In order to fight against the non-lineal discharge battery effects, the DS2780 has a special register, the Accumulation Bias register (AB). It allows an arbitrary bias to be introduced into the current-accumulation process. The AB can be used to account for currents that do not flow through the sense resistor, estimate currents too small to measure, estimate battery self-discharge or correct for static offset of the DS2780.

4.1.3.4 Capacity Estimation Algorithm

The DS2780 is capable of perform on-chip capacity estimation algorithm using real-time measured values, stored parameters describing the cell characteristics, and application operating limits [4]. Therefore, before performing the estimation the integrated circuit must be configured with the parameters which describe the cell and application characteristics.

In order to achieve reasonable accuracy in estimating remaining capacity, the cell performance characteristics over temperature, load current, and charge termination point must be considered. Since the behaviour of Li-ion cells is non-linear these characteristics must be included in the capacity estimation to achieve a reasonable accuracy. Full and empty points are retrieved in a lookup process which re-traces a piece-wise linear model.

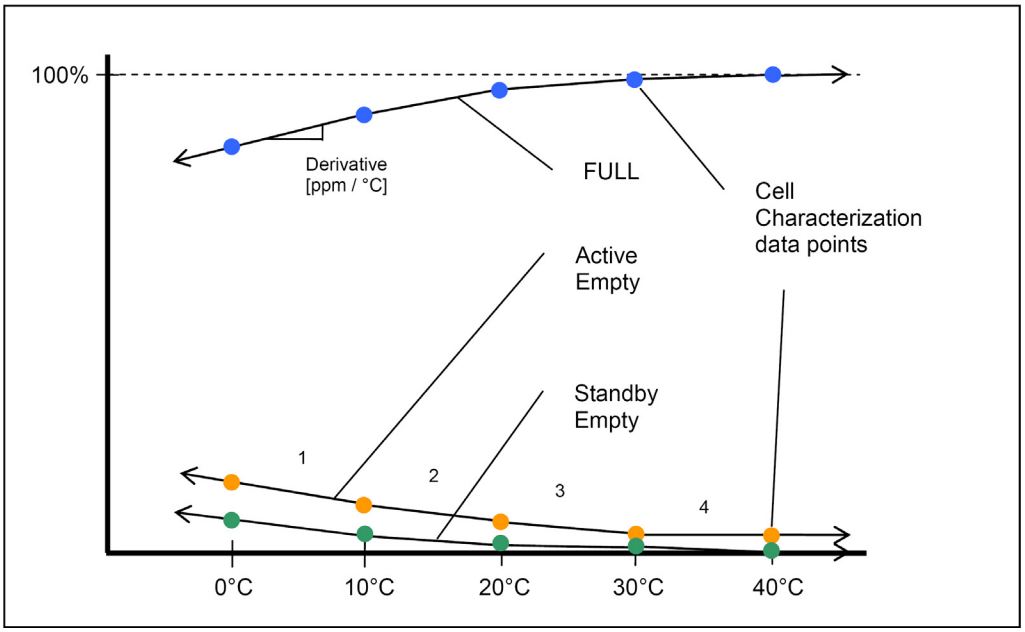


Figure 40. DS2780 cell model diagram

Three model curves are stored: Full, Active Empty and Standby Empty (see Figure 40):

- **Full:** The Full curve defines how the full point of a given cell depends on temperature for a given charge termination. The application's charge termination method should be used to determine the table values. The DS2780 reconstructs the Full line from the cell characteristic table to determine the Full capacity of the battery at each temperature.
- **Active Empty:** The Active Empty curve defines the variation of the Active Empty Point over temperature. The Active Empty Point is defined as the minimum voltage required for system operation at a discharge rate based on a high level load current (one that is sustained during a high power operating mode). This load current is programmed as the Active Empty current (IAE) and should be a 3.5 s average to correspond to values read from the Current register. The specified minimum voltage, or Active Empty voltage (VAE), should be a 220 ms average to correspond to values read from the Voltage register. The DS2780 reconstructs the Active Empty line from the cell characteristic table to de-

termine the Active Empty capacity of the battery at each temperature.

- **Standby Empty:** The Standby Empty curve defines the variation of the standby empty point over temperature. The standby empty point is defined as the minimum voltage required for standby operation at a discharge rate dictated by the application standby current. In typical PDA applications, Standby Empty represents the point that the battery can no longer support RAM refresh and thus the standby voltage is set by the RAM voltage supply requirements. In other applications, Standby Empty can represent the point that the battery can no longer support a subset of the full application operation, such as games or organizer functions on a wireless handset. The standby load current and voltage are used for determining the cell characteristics but are not programmed into the DS2780. The DS2780 reconstructs the Standby Empty line from the cell characteristic table to determine the Standby Empt capacity of the battery at each temperature.

4.1.3.4.1 Cell Model Parameters and Applications Parameters Calculations

Using the battery specification from the manufacturer, the cell model is constructed with all points normalized to the fully charged state at +40 °C. All values are stored in the cell parameter EEPROM block.

The DS2780 allows the user to customize the fuel gauge to the exact parameters of the application. However, the units stored in the device are not units that are commonly used. In the following lines, the meaning of each register used for customizing the estimation algorithm is explained. Also, all the necessary calculations in the case of the Lishen SP0405AC battery are carried out. The aim of the final application is to let the user to enter all the battery specification data on the PC side, and through a coordinator message, it configures the AquisGrain 1.0 platform. Therefore, the next calculations are an example for a specific battery, but they can easily be extended for other types of Li-Ion batteries.

- Accumulation Bias Register

  The Accumulation Bias Register is used to estimate battery currents that do not flow through the sense resistor, or battery self-discharge. This is a signed register with an LSB value of 1.5625 µV/R_{SNS}. It is stored in Address 61h, and has a range of -200.000 µV to 198.4375 µV. With the Sense Resistor of 220 mΩ, the range is -909 µA to 902 µA in 7.102 µA steps. In our case, the register will be initialized to zero.

$$AccBias\_\mu V = AccBias\_mA \cdot Sense\,Resistor\_m\Omega$$
$$= 0mA \cdot 220m\Omega = 0\mu V$$

$$AccBias\_\mu V = 0mA \cdot 220m\Omega = 0\mu V$$

$$ValueStored\ (61h) = \frac{AccBias\_\mu V}{1.5625\mu V} = 00h$$

- Aging Capacity Register

  The Aging Capacity Register stores the rated capacity used in estimat-

ing the decrease in battery capacity during normal use. It is an unsigned register with an LSB value of 6.25 µVh/R_{SNS}. It is stored in Addresses 62h and 63h, and has a range of 0 µVh to 409.59375 mVh. With the Sense Resistor of 220 mΩ, the range is 0 mAh to 1861.789772 mAh in 28.4 µAh steps.

$$AgingCapacity\_\mu Vh = AgingCapacity\_mAh \cdot SenseResistor\_m\Omega$$

Assuming that the battery is new:

$$AgingCapacity\_\mu Vh = 140mAh \cdot 220m\Omega = 30800\mu Vhrs$$

$$ValueStored\ (62h) = \frac{AgingCapacity\_\mu Vh}{6.25\mu Vhr} >> 8 = 13h$$

$$ValueStored\ (63h) = \frac{AgingCapacity\_\mu Vh}{6.25\mu Vhr} = 40h$$

- Charge Voltage Register

  The Charge Voltage Register stores the charge voltage threshold used to detect a fully charged state. This is an unsigned register with an LSB value of 19.52 mV. It is stored in Address 64h, and has a range of 0 to 4.9776 V. The charger used during the thesis has a charge voltage of 4.2 V.

$$ValueStored\ (64h) = \frac{ChargeVoltage\_V}{19.52mV} = D7h$$

- Minimum Charge Current Register

  The Minimum Charge Current Register stores the charge current threshold that detects a fully charged state. This is an unsigned register with an LSB value of 50 µV. It is stored in Address 65h, and has a range of 0 to 12.75 mV. With the Sense Resistor of 220 mΩ, the range is 0 mA to 57.95 mA in 227 µA steps.

$$ChargeCurrent\_\mu n = ChargeCurrent\_mA \cdot SenseResistor\_m\Omega$$

$$= 50mA \cdot 220m\Omega = 11000\mu V$$

$$ValueStored\ (65h) = \frac{ChargeVoltage\_\mu V}{50\mu V} = DCh$$

- Active Empty Voltage Register

  The Active Empty Voltage Register stores the voltage threshold used to detect the Active Empty point. This is an unsigned register with an LSB value of 19.52 mV. It is stored in Address 66h, and has a range of 0 to 4.9776 V. In the case of the AquisGrain 1.0 Platform, this voltage is 2.5 V, which is the minimum voltage that the voltage regulator can amplify till the 3.3 V that the platform needs.

$$ValueStored\ (66h) = \frac{AEVoltage\_V}{19.52mV} = 80h$$

- Active Empty Current Register

The Active Empty Current Register stores the discharge current threshold that detects the Active Empty point. This is an unsigned register with an LSB value of 200 µV. It is stored in Address 67h, and has a range of 0 to 51.2 mV With the Sense Resistor of 220 mΩ, the range is 0 mA to 232 mA in 0.91 mA steps.

$$AECurrent\_\mu V = AECurrent\_mA \cdot SenseResistor\_m\Omega$$
$$= 6.5mA \cdot 220m\Omega = 1430\mu V$$

$$ValueStored\ (67h) = \frac{ChargeVoltage\_\mu V}{200\mu V} = 07h$$

- Active Empty 40 Register

  The Active Empty 40 Register stores the Active Empty point value for +40°C. This is an unsigned register with an LSB value of parts per million of the Full point at +40°C. It is stored in Address 68h, and has a range of 0 to 24.9% of the Full point for +40°C. From the battery specifications it is possible to deduce all the information related to the battery response to temperature changes. So, the Active Empty point value for +40°C is approximately 6 mAh, and the full point for +40°C is approximately 150 mAh.

$$ValueStored\ (68h) = \frac{ActiveEmpty\_40\_mAh}{Full\_40\_mAh \cdot 2^{-10}} = 28h$$

- Sense Resistor Primer Register

  The Sense Resistor Prime (RSNSP) Register stores the value of the Sense Resistor that computes the absolute capacity results. This is an unsigned register with an LSB value of 1mhos. It is stored in Address 69h, and has a range of 1mhos to 255 mhos, which is 1 Ω to 3.922 mΩ. As explained in the previous point, the resistor that best fits our requirements is a sense resistor of 220 mΩ, though several tests were done with other sense resistors to proof that.

$$ValueStored\ (69h) = \frac{1}{SenseResistor\_\Omega} = 05h$$

- Full 40 Register

  The Full 40 Register stores the Full point value for +40°C. This is an unsigned register with an LSB value of 6.25µVh/R_{SNS}. It is stored in Addresses 6Ah and 6Bh, and has a range of 0 µVh to 409.59375 µVh. Assuming the Sense Resistor has a value of 220 mΩ, the range is 0 mAh to 1861.79 mAh in 284.1 µAh steps.

$$Full40\_\mu Vh = Full40\_mAh \cdot SenseResistor\_m\Omega$$
$$= 150mAhrs \cdot 220m\Omega = 33000\mu Vhrs$$

$$ValueStored\ (6Ah) = \frac{Full40\_\mu Vh}{6.25\mu Vh} >> 8 = 14h$$

$$ValueStored\ (6Bh) = \frac{Full40\_\mu Vh}{6.25\mu Vh} = A0h$$

- Full Slopes

  The Full point at +40°C is a stored value (Full 40), and the Full points at the other temperatures are calculated using the slope of the Full line. The slope for each 10-degree segment of the Full line is stored as an unsigned byte in terms of parts per million per °C (ppm/°C). It is assumed that the Full point at +40°C is the highest point on the Full line. The Full line is reconstructed in one-degree increments so that the Full point at any temperature is always less than or equal to the Full point at the next higher temperature. The slope can range from 0 to 15564 ppm/°C.

$$SlopeFrom30to40 = 1 - \frac{Full\_30\_mAh}{Full\_40\_mAh} = 1 - \frac{145}{150} = 0.033$$

$$SlopeFrom20to30 = \frac{Full\_30\_mAh}{Full\_40\_mAh} - \frac{Full\_20\_mAh}{Full\_40\_mAh} = \frac{145}{150} - \frac{138}{150} = 0.046$$

$$SlopeFrom10to20 = \frac{Full\_20\_mAh}{Full\_40\_mAh} - \frac{Full\_10\_mAh}{Full\_40\_mAh} = \frac{138}{150} - \frac{130}{150} = 0.053$$

$$SlopeFrom0to10 = \frac{Full\_10\_mAh}{Full\_40\_mAh} - \frac{Full\_0\_mAh}{Full\_40\_mAh} = \frac{130}{150} - \frac{126}{150} = 0.026$$

$$ValueStored\ (6Ch) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 36h$$

$$ValueStored\ (6Dh) = \frac{SlopeFrom20to30}{10 \cdot 2^{-14}} = 4Bh$$

$$ValueStored\ (6Eh) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 56h$$

$$ValueStored\ (6Fh) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 2Ah$$

- Active Empty Slopes

  The Active Empty line is reconstructed in a similar fashion as the Full line. The Active Empty point at +40°C is a stored value (Active Empty 40), and the Active Empty points at other temperatures are calculated from the slope between each 10-degree step. The slopes between each of the Active Empty points are stored as an unsigned byte in ppm/°C. The Active Empty line is reconstructed in one-degree increments so that the Active Empty point at any temperature is always greater than or equal to the Active Empty point at the next higher temperature. The slope can range from 0 to 15564 ppm/°C.

$$SlopeFrom30to40 = \frac{AE\_30\_mAh}{Full\_40\_mAh} - \frac{AE\_40\_mAh}{Full\_40\_mAh} = \frac{10}{150} - \frac{6}{150} = 0.026$$

$$SlopeFrom20to30 = \frac{AE\_20\_mAh}{Full\_40\_mAh} - \frac{AE\_30\_mAh}{Full\_40\_mAh} = \frac{16}{150} - \frac{10}{150} = 0.04$$

$$SlopeFrom10to20 = \frac{AE\_10\_mAh}{Full\_40\_mAh} - \frac{AE\_20\_mAh}{Full\_40\_mAh} = \frac{20}{150} - \frac{16}{150} = 0.026$$

$$SlopeFrom0to10 = \frac{AE\_0\_mAh}{Full\_40\_mAh} - \frac{AE\_10\_mAh}{Full\_40\_mAh} = \frac{23}{150} - \frac{20}{150} = 0.02$$

$$ValueStored\ (70h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 2Ah$$

$$ValueStored\ (71h) = \frac{SlopeFrom20to30}{10 \cdot 2^{-14}} = 41h$$

$$ValueStored\ (72h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 2Ah$$

$$ValueStored\ (73h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 20h$$

- Standby Empty Slopes

  The Standby Empty line is reconstructed similar to the Full and Active Empty lines. The Standby Empty point at +40°C is fixed at 0 mAh. The Standby Empty points at other temperatures are calculated from the slope between each 10-degree step. The slopes between each of the Standby Empty points are stored as an unsigned byte in ppm/°C. The Standby Empty line is reconstructed in one-degree increments so that the Standby Empty point at any temperature is always greater than or equal to the Standby Empty point at the next higher temperature. The slope can range from 0 to 15564 ppm/°C.

$$SlopeFrom30to40 = \frac{SE\_30\_mAh}{Full\_40\_mAh} = \frac{1}{150} - = 0.0066$$

$$SlopeFrom20to30 = \frac{SE\_20\_mAh}{Full\_40\_mAh} - \frac{SE\_30\_mAh}{Full\_40\_mAh} = \frac{3}{150} - \frac{1}{150} = 0.013$$

$$SlopeFrom10to20 = \frac{SE\_10\_mAh}{Full\_40\_mAh} - \frac{SE\_20\_mAh}{Full\_40\_mAh} = \frac{5}{150} - \frac{3}{150} = 0.026$$

$$SlopeFrom0to10 = \frac{SE\_0\_mAh}{Full\_40\_mAh} - \frac{SE\_10\_mAh}{Full\_40\_mAh} = \frac{10}{150} - \frac{5}{150} = 0.033$$

$$ValueStored\ (74h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 0Ah$$

$$ValueStored\ (75h) = \frac{SlopeFrom20to30}{10 \cdot 2^{-14}} = 15h$$

$$ValueStored\ (76h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 2Ah$$

$$ValueStored\ (77h) = \frac{SlopeFrom30to40}{10 \cdot 2^{-14}} = 36h$$

- Sense Resistor Gain Register

  The Sense Resistor Gain (RSGAIN) Register stores the calibration factor that produces accurate readings in the Current Register when a ref-

erence voltage is forced across SNS and VSS. This is an 11-bit value with an LSB value of 1/1024. It is stored in Addresses 78h and 79h, and has a range of 0 to 1.999. The nominal value for this register is 1.000.

$$ValueStored\ (78h) = (RSGAIN \cdot 1024) >> 8 = 04h$$

$$ValueStored\ (79h) = (RSGAIN \cdot 1024) = 00h$$

- Sense Resistor Temperature Coefficient Register

  The Sense Resistor Temperature Coefficient (RSTCO) Register stores the temperature coefficient of the sense resistor. This is an 8-bit value with an LSB value of 30.5176ppm/°C. It is stored in Address 7Ah, and has a range of 0 to 7782ppm/°C. For our ECG application it is set to zero which disables the temperature-compensation function.

$$ValueStored\ (7Ah) = \frac{RSTCO}{30.5176} = 00h$$

4.1.3.4.2 Capacity Estimation Utility Functions

The DS2780 as well as being a fuel gauge, it incorporates a series of functions that help it to perform a more accurate capacity estimation.

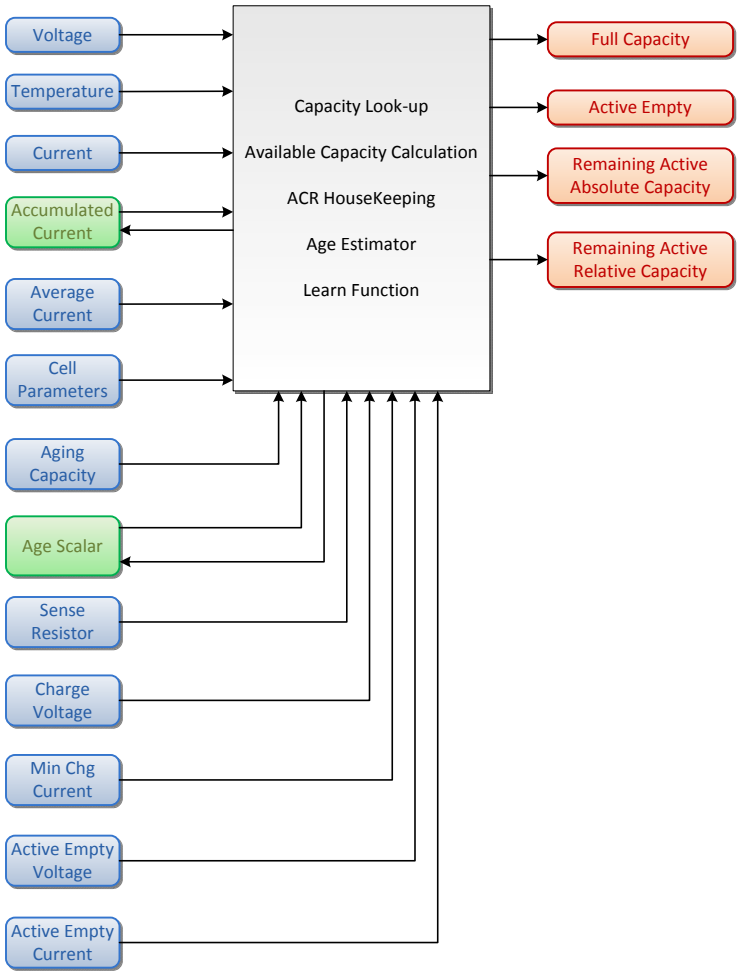


Figure 41. DS2780 top level algorithm diagram

As shown in Figure 41, the main capacity functions are:

- **Aging Estimation**

  Lithium batteries also suffer from time-related aging, which causes their capacity to fall from the moment the battery leaves the factory, regardless of usage. This effect can cause a fully charged Li-ion battery to lose 20% of its capacity per year at 25°C, and 35% at 40°C. A special register adjusts the cell capacity estimation results downward to compensate for aging. The Aging Scalar register value is adjusted occasionally based on cumulative discharge. As the ACR register decrements during each discharge cycle, an internal counter is incremented until equal to 32 times the AC. The AS is then decremented by one, resulting in a decrease in the scaled full battery capacity of 0.78% (approximately 2.4% per 100 cycles).

- **Learn Function**

  Since Li+ cells exhibit charge efficiencies near unity, the charge delivered to a Li+ cell from a known empty point to a known full point is a dependable measure of the cell capacity. A continuous charge from empty to full results in a "learn cycle". First, the Active Empty Point must be detected. The Learn Flag (*LEARNF*) is set at this point. Then, once charging starts, the charge must continue uninterrupted until the battery is charged to full. Upon detecting full, *LEARNF* is cleared, the Charge to Full (*CHGTF*) flag is set and the Age Scalar (AS) is adjusted according to the learned capacity of the cell.

- **ACR Housekeeping**

  The ACR value is adjusted occasionally to maintain the coulomb count within the model curve boundaries. When the battery is charged to full (*CHGTF* set), the ACR is set equal to the age scaled full lookup value at the present temperature. If a learn cycle is in progress, correction of the ACR value occurs after the age scalar (AS) is updated. When an empty condition is detected (*AEF* or *LEARNF* set), the ACR adjustment is conditional. If *AEF* is set and *LEARNF* is not, then the Active Empty Point was not detected and the battery is likely below the Active Empty capacity of the model. The ACR is set to the Active Empty model value only if it is greater than the Active Empty model value. If *LEARNF* is set, then the battery is at the Active Empty Point and the ACR is set to the Active Empty model value.

- **Full Detect**

  Full detection occurs when the Voltage (VOLT) readings remain above the VCHG (Charge Voltage) threshold for the duration of two Average Current (IAVG) readings, where both IAVG readings are below IMIN (Terminating Current). The two consecutive IAVG readings must also be positive and non-zero. This ensures that removing the battery from the charger does not result in a false detection of full.

- **Active Empty Point Detect**

  Active Empty Point detection occurs when the Voltage register drops below the VAE threshold and the two previous Current readings are

above IAE. This captures the event of the battery reaching the Active Empty Point. Note that the two previous Current readings must be negative and greater in magnitude than IAE, that is, a larger discharge current than specified by the IAE threshold. Qualifying the Voltage level with the discharge rate ensures that the Active Empty Point is not detected at loads much lighter than those used to construct the model. Also, the Active Empty Point must not be detected when a deep discharge at a very light load is followed by a load greater than IAE. Either case would cause a learn cycle on the following charge to include part of the Standby capacity in the measurement of the Active capacity.

4.2 Policy Specification

How the user could interact with a wireless sensor network is a challenge that is still under debate. Although users have experience with the system under observation, the programming paradigm offered to wireless sensor networks users who are not accustomed to programming remains a broadly open question. Moreover, the lack of device networking knowledge makes the policy specification rules difficult to be understood by common users. The main idea of the policy specification is to allow the user to set several directives that are dynamically checked and automatically satisfied at run-time by the application.

The language used in the specification should be sufficiently expressive for the user to manage the application. However, this expressiveness could mask all the system features. So, it is necessary to find an intermediate point between expressiveness and masking system features. Besides, the language should emphasize the application capabilities while abstracting away the complexity of the network interaction.

On the other hand, the complexity of the algorithm to support this language should also be taken into account. Usually, the application which is running in the wireless sensor network could suffer sudden changes so it must be quickly adapted to a dynamic changing environment. Again, our language must satisfy a compromise between a computationally tractable algorithm and a real time response to environmental fluctuations.

In order to bound these needs, the directives which form our policy must contain only one optimization clause (maximize/minimize) while the rest of the clauses must be binary predicates. This, when coupled with strict prioritization, essentially reduces the algorithm complexity to linear time with one variable for optimization.

Figure 42 shows a possible policy that a user could set up. It should be noted that this example is specific for the ECG application. As it will be explained in the next points, a policy consists of rules. The example policy consist of five rules divided in threes different priority levels. Each rule consists of one element which has a binary operator followed by a value. As it has been previously, a rule can include an optimization clause as the first one in the example which maximizes the lifetime.

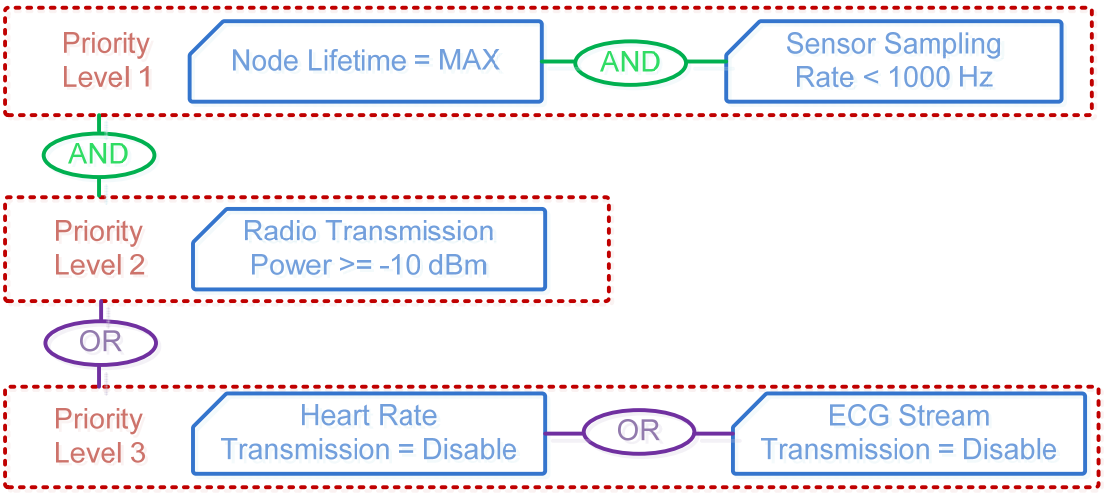


Figure 42. User Policy Specification example

4.2.1 Rules Priority

For addressing these chiefs, the specification should incorporate a prioritized list of user requirements, enabling expression of a simple enforcement policy that can be carried out by the energy manager system. This list of requirements composes the user policy which is constrained by a single shared source –energy. It is inevitable that these rules will compete among them, so it is necessary to associate each rule with a certain priority. With this priority system the architecture will be able to handle situation when a subset of the rule cannot be met. The idea of priority pervades throughout the specification, and makes a quickly rule evaluation possible. Prioritization is also the key to let the user decide which operation should see the least interruption in the case of an energy shortage.

The user specification consists of a series of levels of priorities, which are joined by logical operators (e.g. AND, OR). Each level composed of one or more rules also joined by logical operators. One rule is composed of a term (e.g. sensor sampling rate, lifetime), a binary operator (<, <=, >, >=, =) and a desired value for the specified term. This value can contain an optimization clause such as MIN or MAX, where the system attempts to perform the indicated action as much or as little as possible based on the directive.

4.2.2 Application Status

In order to help the user to introduce the right desired policy, it may be important to include the user in the feedback loop about the status of the application, and, therefore, of the policy. Then, the user will be aware in about the application status real time and will be able to choose the policy that best fits his desires.

For instance, if the user had introduced a desired lifetime of 15 hours, and after 5 hours of running the application due to the dynamic environment the energy consumption would had increased till the level that the desired lifetime could not be achieved, an exception should be raised. Therefore, the policy specification module has to handle possible exceptions.

4.3 Energy Management

This module will be in charge of regulating the energy resources, providing QoS based on user policy. The energy manager should work locally, so its decisions should affect only a node at once. Therefore, our energy manager is focused in regulate local energy resources according to the user policy.

From the energy monitor information, the energy manager uses the system energy levels and component usage profile to take decisions based on user policy. For instance, in order to estimate the remaining lifetime the energy manager can use an average current consumption provided by the energy monitor. Then, this estimation will help the user to set a desired lifetime, so the energy manager could take the right decision to achieve this lifetime.

Once the user has set a policy, the energy manager will check periodically that this policy could be reached and if it is not possible to achieve the user requirements an exception will be raised to inform the user. Therefore, the energy manager has to know in advance all the possible energy states of the AquisGrain 1.0 platform. Then, with the information of the current consumption in each state it will be able to diagnose the best energy state for the user policy.

As the user policy could have more than one possible path, as the rules are joined by logical operators, the energy manager has to check all the possible paths within a priority level and choose the one which is less energy restrictive for the application (see Figure 43). This figure shows a Tree produced by recursive parser on the given rule. Each interior node represents a logical operator (either AND or OR) while each leaf node represents a term in the rule. The relationships between the terms are encoded by the parent-child relationships within the tree. The recursive parser creates a tree for each rule, allowing the enforcement algorithm to evaluate each rule efficiently. When the user policy consists of more than one priority level, this module has also to check if all the levels could be accomplished and if it is not possible, it has to ensure that the levels with highest priority are achieved.

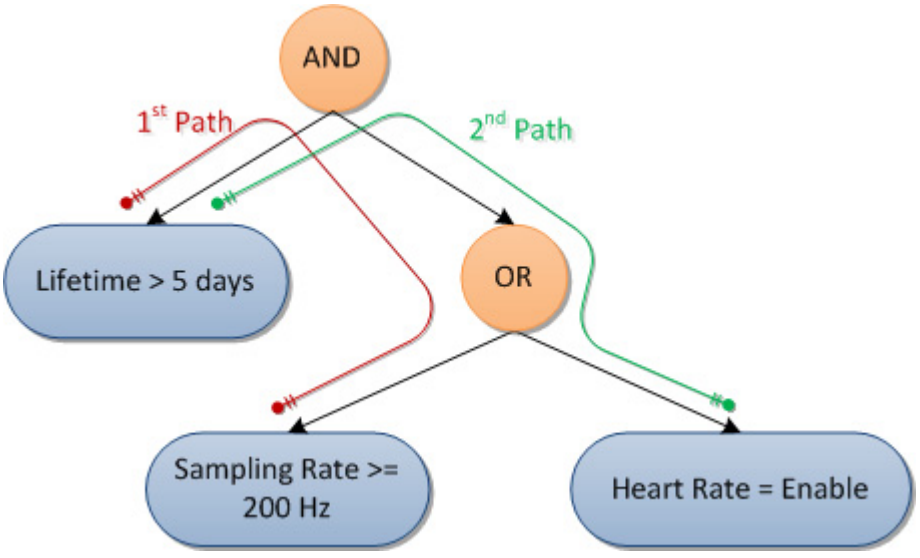


Figure 43. Example of a tree produced from a policy rule

However, in order to be able to apply any kind of energy management, a

power save application must be running in the AquisGrain 1.0 platform. The main modules which determine the energy consumption of this platform are the ATmega 128L and the CC2420 RF transceiver.

4.3.1 Overview of CHIPCON CC2420

The CHIPCON CC2420 is a single-chip 2.4 GHz IEEE 802.15.4 compliant RF transceiver designed for low-power and low-voltage wireless applications. The CC2420 includes a digital direct sequence spread spectrum base band modem providing a spreading gain of 9 dB and an effective data rate of 250 kbps. The CC2420 is a low-cost, highly integrated solution for robust wireless communication in the 2.4 GHz unlicensed ISM band. The RF transceiver also provides extensive hardware support for packet handling, data buffering, burst transmissions, data encryption, data authentication, clear channel assessment, link quality indication and packet timing information. The configuration interface and transmit / receive FIFOs of the CC2420 are accessed via an SPI interface. In a typical application the CC2420 will be used together with a microcontroller and the necessary passive and active components.

Figure 44 illustrates a simplified block diagram of the CC2420, which features a low-IF receiver. The received RF signal is amplified by a low-noise amplifier (LNA) and down-converted in quadrature (I and Q) to the intermediate frequency (IF). At IF (2 MHz), the complex I/Q signal is filtered and amplified, and then digitized by the ADCs. Automatic gain control, final channel filtering, de-spreading, symbol correlation and byte synchronization are performed digitally. The SFD pin goes high when a start of frame delimiter has been detected. The CC2420 buffers the received data in a 128 byte receive FIFO. The user may read the FIFO through an SPI interface. Cyclic Redundancy Check bytes are verified in the hardware. RSSI and correlation values are appended to the frame. CCA is available on a pin in receive mode. Serial (unbuffered) data modes are also available for test purposes.

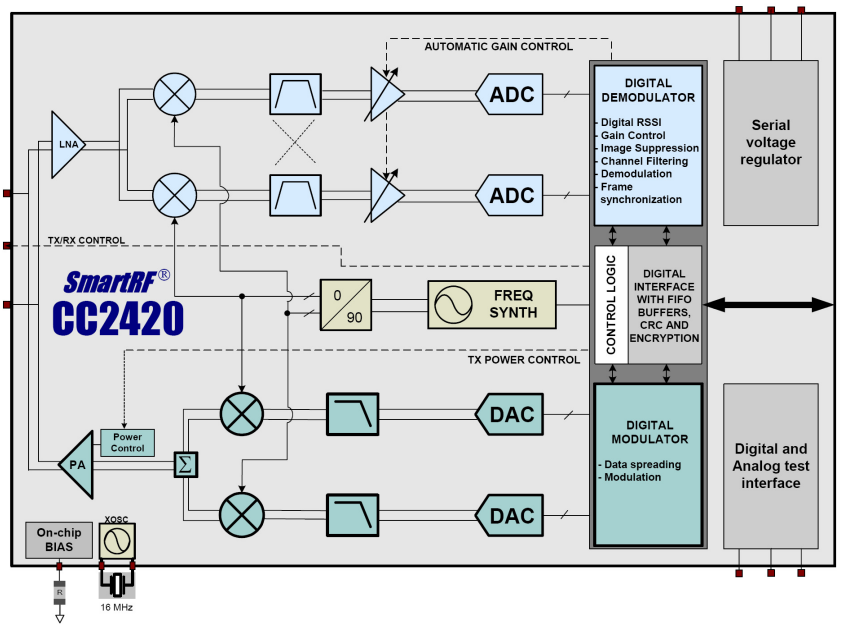


Figure 44. CC2420 simplified block diagram

The CC2420 transmitter is based on direct up-conversion. The data is buffered in a 128 byte transmit FIFO (separate from the receive FIFO). The preamble and start of frame delimiter are generated by hardware. Each symbol (4 bits) is spread using the IEEE 802.15.4 spreading sequence to 32 chips and output to the digital-to-analog converters (DACs). An analog low-pass filter passes the signal to the quadrature (I and Q) up-conversion mixers. The RF signal is amplified in the power amplifier (PA) and fed to the antenna.

Table 6 shows the main parameters of the CC2420.

Parameter	Nominal Value
TX current	17.4 mA
RX current	19.7 mA
Output power	0 dBm
Sensitivity	-94 dBm
Range (Line-of-Sight)[2]	230 meter

Table 6. Main CC2420 parameters

The RF output power of the device is programmable and is controlled by the TXCTRL.PA_LEVEL register. Table 7 contains all the possible transmission power modes of the CC240, relating the power mode with the expected power consumption.

Output Power	Current Consumption
0 dBm	17.4 mA
-1 dBm	16.5 mA
-3 dBm	15.2 mA
-5 dBm	13.9 mA
-7 dBm	12.5 mA
-10 dBm	11.2 mA
-15 dBm	9.9 mA
-25 dBm	8.5 mA

Table 7. CC2420 possible transmission powers

In addition, the CC2420 has four power save modes (see Table 8). The normal operational mode is when the reception is always on, but this is the most restrictive consumption mode. On the other hand, if the radio reception is off while the chip is in the idle state, the power consumption decreases 96% but it takes 30 µs to turn the reception on. The other two states are not feasible for the ECG application as the time that it takes to turn the radio on will produce an unacceptable delay.

Radio mode	Power con-sump-tion	Time to return to full working mode
RX_IS_ON_WHEN_IDLE	24 mA	--
RX_OFF_WHEN_IDLE	1 mA	30 micro sec
MPM_CC2420_XOSC_OFF	> 0.5 mA	1200 micro sec
MPM_CC2420_XOSC_AND_VREG_OFF	> 0.2 mA	2400 micro sec

Table 8. CC2420 radio modes

4.3.2 Overview ATMEGA 128L

The ATmega 128L is a low-power CMOS 8-bit microcontroller based on the AVR Enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega 128L achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed. The AVR core features 32 general purpose working registers which are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega 128L provides the following features: 128K bytes of in-system programmable flash with read-while-write capabilities, 4K bytes of EEPROM, 4K bytes of SRAM, 53 general purpose I/O lines, 32 general purpose working registers, real time counter (RTC), four flexible timer/counters with compare modes and PWM, 2 USARTs, a byte oriented two-wire serial interface, a 8-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable watchdog timer with internal oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the on-chip debug system and programming, and six software selectable power saving modes.

During the Idle mode the CPU is disabled while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The power-down mode saves the register contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset. In power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC noise reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC to minimize switching noise during ADC conversions. In standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In extended standby mode, both the main oscillator and the asynchronous timer continue to run .

The on-chip ISP Flash allows the program memory to be reprogrammed in-

system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an on-chip boot program running on the AVR core. The boot program can use any interface to download the application program in the application flash memory. Software in the boot flash section will continue to run while the application flash section is updated, providing true read-while-write operation. By combining an 8-bit RISC CPU with in-system self-programmable flash on a monolithic chip, the Atmel ATmega 128L is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications. The ATmega 128L AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Atmega128L working mode	Power consumption	Time to return to full working mode
Normal working mode	12 mA	--
Power IDLE mode	4.1 mA	2 micro sec Wake up by timer.
Power DOWN mode	100 µA	1K CK Wake up by external IRQ.
Power standby mode	250 µA	1K CK Wake up by timer0

Table 9. Main Atmega128L operating modes

The ATmega 128L has six sleep modes which enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the application's requirements. The preferred working mode for the microcontroller is the IDLE mode, in which it consumes about one third of the power consumed in the normal mode (see Table 9). The other modes are not feasible for the ECG application as the time that it takes to turn the microcontroller on will produce an unacceptable delay.

4.3.3 Electrocardiogram Power Saver Application

After exposing the main power characteristics of the ATMEGA128L and the CC2420, it is obvious that a power save module is necessary in our architecture. In order to determine which power mode is the best one, it is necessary to understand how the ECG application works.

A medical sensor in the ECG environment measures periodically various values, so it is active very often. However, the radio is needed only for shorter intervals. In this way, the radio have to stay in a save power mode as much time as possible to reduce the energy consumption. Moreover, the microcontroller will remain in the power idle mode with the same purpose.

On the one hand, The ECG is sampled with a variable rate of 200, 500 or

1000 samples/second. The sensor sampling rate depends on the user policy. Every 5, 2 or 1 millisecond a sample is captured from ADC and stored into a message buffer. As the maximum data length is limited to 84 bytes, the application can keep 39 samples so that ECG stream is sent in a 195, 78 or 39 millisecond interval respectively.

On the other hand, the sensor has to propagate the evaluated heart rate every second.

These requirements give the timing diagram shown below:

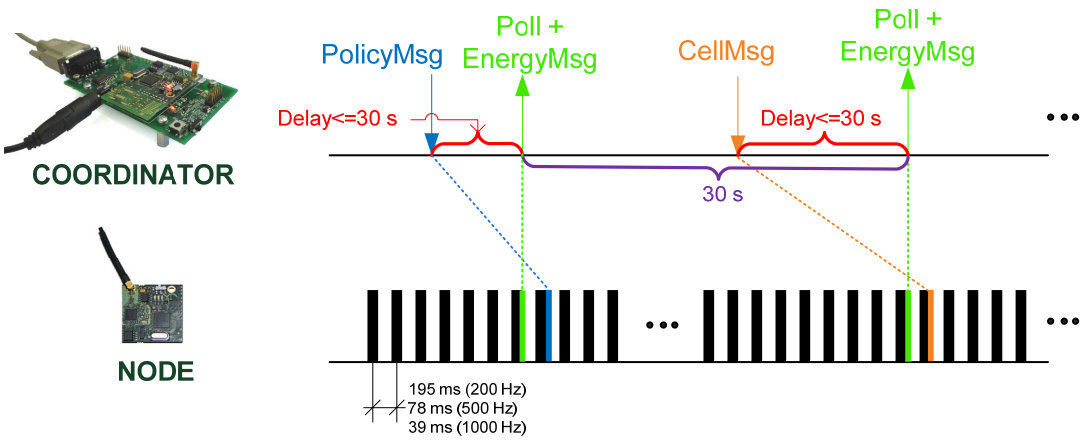


Figure 45. Timing diagram showing message exchange protocol

With these requirements, the optimal energy situation is that the CC240 remains in the RX_OFF_WHEN_IDLE power mode as much time as possible, and the Atmega128L in the power idle mode.

In order to save as much energy as possible, it is decided to use a non-beacon system. Therefore, no time synchronization message from coordinator is sent to the nodes, and the radio reception can remain much time turned off.

However, the sensor must be able to receive messages from the coordinator. These messages are completely asynchronous, as the user sets the policy in an undefined time. This dilemma suggests the use of a polling protocol. The coordinator will store all the policy messages and cell messages in a buffer until the node polls it.

With this design, an estimation of the expected win of power is carried out:

Streaming activity:

- The radio is on for 4…10 milliseconds and off the rest of the 195 (200 Hz) / 78 (500 Hz) /39 (1000 Hz) milliseconds depending on the sampling rate.
- The microcontroller is in power idle mode for most of the time (but ADC, timer running) and in full power mode for the sampling (200, 500, 1000 samples) for 100 microseconds each.

Architecture messages activity:

- The radio is on for the duration of an energy message (~2 milliseconds)

and off the rest of the 30 seconds. This activity in terms of energy consumption is negligible.

- The microcontroller is in full power mode for the same interval.

Power consumption: **No power save mode**	**Power consumption:** **Practical duration** **(assuming send = 10ms)**
Radio: 24 mA -	Radio: 24 mA * 13% + 1 mA * 87%
MMCU: 12 mA	MMCU: 12 mA * 28% + 4 mA * 72%
ECG amp 2 mA	ECG amp 2 mA
Total 38 mA	Total 12.23 mA
Power consume**: 100 %**	Power consume: **34 %**
Battery lifetime: ***Lishen SP0405AC* ~= 3.7 hours** ***Varta 653450UC* ~= 29.8 hours**	**Battery lifetime:** ***Lishen SP0405AC* ~= 11.5 hours** ***Varta 653450UC* ~= 92.4 hours**

Table 10. Power consumption comparison whit and without power saver module

5 Implementation

Once the design principles have been established, the implementation of each module of the energy management architecture is carried out. First of all, we must decide where each module is going to be working. The energy monitor must work in each AquisGrain 1.0 node as it has to monitor the battery status. Similarly, the user policy specification must work in the coordinator side, as the user has to interact with a computer in order to set the rules. The dilemma arises with the energy manager, as it could work in the node or in the coordinator side. However, as the main purpose of wireless sensor networks, and therefore, of the ECG application, is to optimize the energy consumption it seems reasonable that the energy manager works in the coordinator side. Working in the coordinator side, all the policy specification processing will be done by the computer in which no special energy restrictions are expected, and only some messages will be sent to the involved nodes to set the policy. In this way, the microprocessor of each node only has to process the incoming messages. These locations are shown in Figure 46.

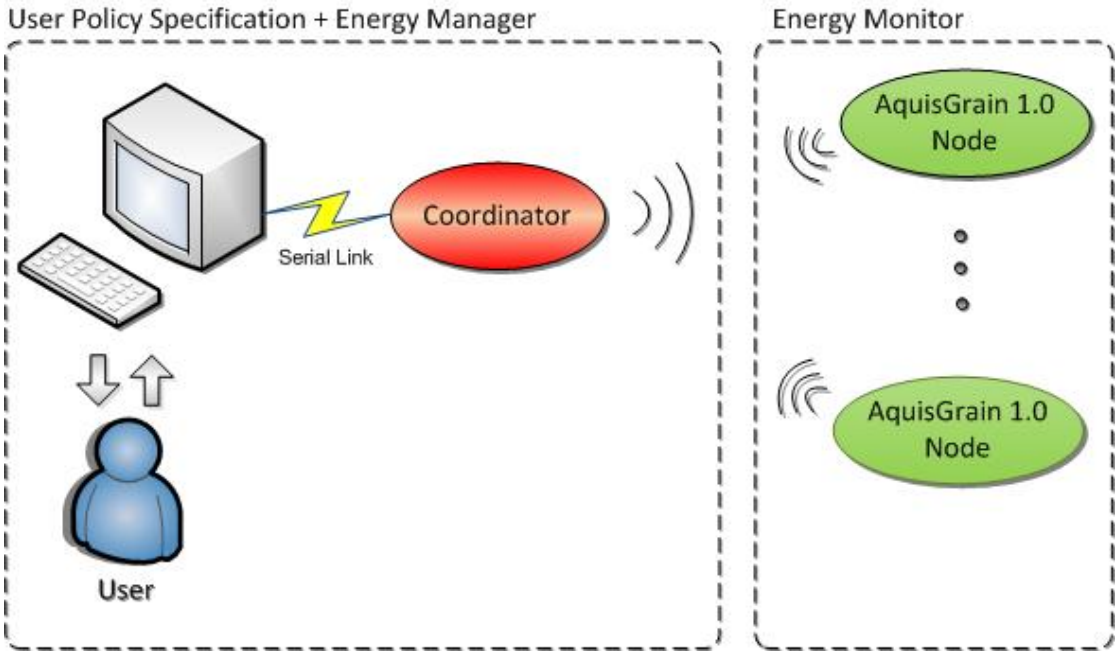


Figure 46. Architecture implementation elements scheme

In this chapter, the implementation of each module of the proposed architecture is explained in detail.

5.1 Energy Monitor

According to the design and the requirements, it is possible to differentiate two parts in the implementation of the energy monitor. The first part will be in charge of getting all the information from the battery and send it to the coordinator through the AquisGrain 1.0. The second part will consist of a PC application to show the user this information and let him introduce the battery specification data to configure the application running in the node. The solu-

tion here described is totally compliant with the Chipcon MAC used in AquisGrain 1.0 sensors and its implementation allows to reliably monitor Li-Ion or Li-Polymer batteries.

As discussed in the previous chapter, in order to implement the first part we decided to insert the DS2780 integrated circuit between the AquisGrain 1.0 and the selected battery. The DS2780 uses the 1-Wire™ Interface to communicate with the Atmega128L.

In this section, it is explained a simple possible software solution for basic 1-Wire™ communication between the AquisGrain 1.0 and the fuel gauge integrated circuit. Moreover, the features and the capabilities of the PC application are presented.

5.1.1 AquisGrain 1.0 Hardware Configuration

The AquisGrain 1.0 is equipped with a 5-pin battery connector, as Figure 47 shows. The five lines are used as followed:

- PACK+, which should be connected to the positive input power of the AquisGrain 1.0. In this line the PACK+ output of the DS2780 schematics is connected.
- Data, which is a bi-directional data line based in the Dallas Semiconductor's proprietary solution one-wire interface, and is connected together with the one-wire interface output of the DS2436 battery monitor and the DS2780 fuel gauge.
- A reserved line.
- VCharge, in which the charger power output is supposed to be connected
- PACK-, which should be connected to the negative input power of the AquisGrain 1.0. In this line the PACK- output of the DS2780 schematics is connected.

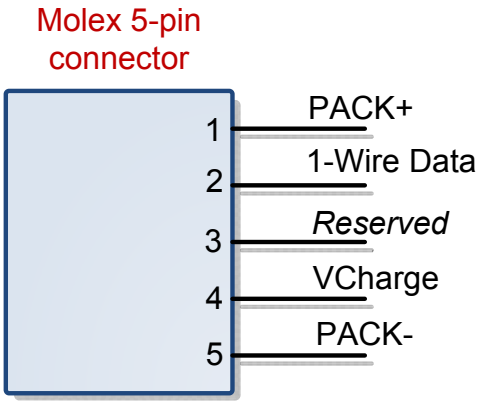


Figure 47. AquisGrain 1.0 battery connector

The DATA line of the 5-pin connector is a single-wire bus which provides communication access and power to both devices: the DS2436 identifier chip and the DS2780 fuel gauge (see Figure 48). Power to the bus is provided through the 4.7 kΩ pull-up resistor from a 3.3 V supply rail. This resistor value

is chosen to allow tolerance for highly resistive contacts and to provide good logic levels at both ends of a short cable. Usually the reading circuitry of the master will accept a voltage of up to 0.8 V as logic 0. Since the loading from the cable between master and the devices influences the time constant of the 1-Wire bus, it may be necessary to use pull–up resistors below 5 kΩ.

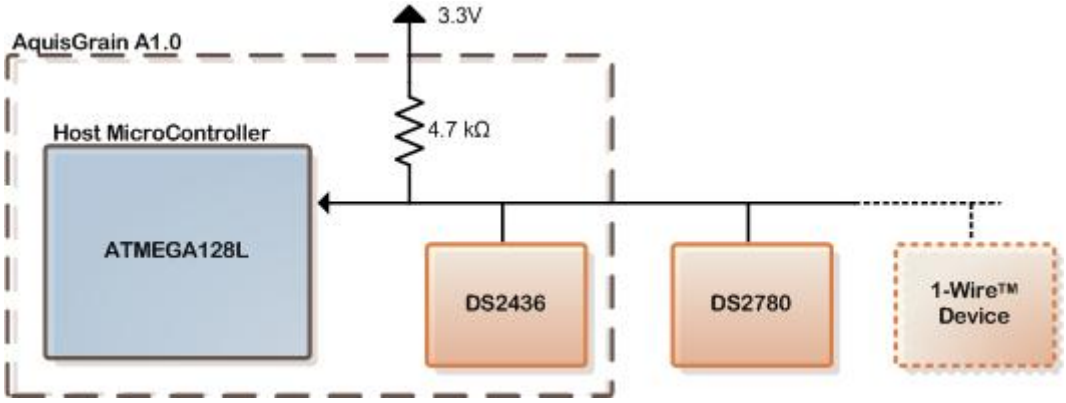


Figure 48. AquisGrain 1.0 1-Wire elements connections

1-Wire bus has a single line so it is important that each device on the bus must be able to drive it at the appropriate time. To ensure this, each device attached to the 1-wire bus must be connected to the bus with open-drain or tri-state output drivers. If a bidirectional pin is not available on the bus master, separate output and input pins can be connected together. As each device has a unique 64-bit ROM code identifier, the devices can be connected to the bus without interfering one each other.

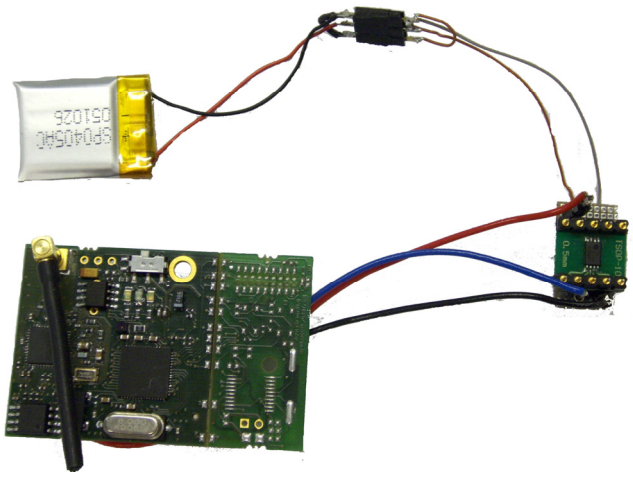


Figure 49. DS2780 connected to an AquisGrain 1.0

5.1.2 Interfacing the DS2780

The 1-Wire timing protocol has specific timing constraints that must be followed in order to achieve successful communication. The timing logic provides a means of measuring and generating digital pulses of various widths. Data transfers are bit–asynchronous and half–duplex. Communication with 1-Wire components is done in time slots of nominal 60 ms duration. Each time slot transfers 1 bit. An extra long low pulse acts as a reset. In response to a reset, 1-Wire components generate a Presence Pulse.

At present, the AquisGrain 1.0 libraries developed by Philips Research only allow to perform the 64-bit ROM code identifier reading of only one device. As our solution needs to connect the DS2780 in the same bus that the DS2436 is connected, these functions are useless. Therefore, in order to control accurately the special timing requirements of the 1-Wire interface, certain key functions must first be established. In this way, the file *mau_serialid.c* is modified in order to allow accessing to the different registers of the DS2780. In addition, a header file *onewire.h* is added to the root folder. This header file contains the definitions of all the DS2780 registers, 1-Wire commands and 1-Wire families.

The first function needed is the “delay” function which is integral to all read and write control. This function is already implemented, though it is named *halWait(uint8 miliseconds)*. Using this function we have to take into account that the time spent in interruptions will be added to the timeout. So to avoid that each time a 1-Wire function is called, the interruptions are disabled.

Once the delay function, which is entirely dependent on the speed of the microcontroller, is implemented the write and read functions can be implemented.

5.1.2.1 Write Time Slots

Timing relations in the DS2780 are defined with respect to time slots. To provide maximum margin for all types of tolerances, it samples the status of the data line in the middle of a time slot. By definition the active part of a 1-Wire time slot (t_{SLOT}) is 60 ms. Under nominal conditions, it will sample the line 30 ms after the falling edge of the start condition.

The internal time base of the DS2780 may deviate from its nominal value. The allowed tolerance band ranges from 15 ms to 60 ms. During this time frame the voltage on the data line must stay below V_{ILMAX} or above V_{IHMIN}. The duration of a low pulse to write a 1 (t_{LOW1}) must be shorter than 15 ms (see Figure 50); to write a 0 the duration of the low pulse (t_{LOW0}) must be at least 60 ms to cope with worst–case conditions (see Figure 51).

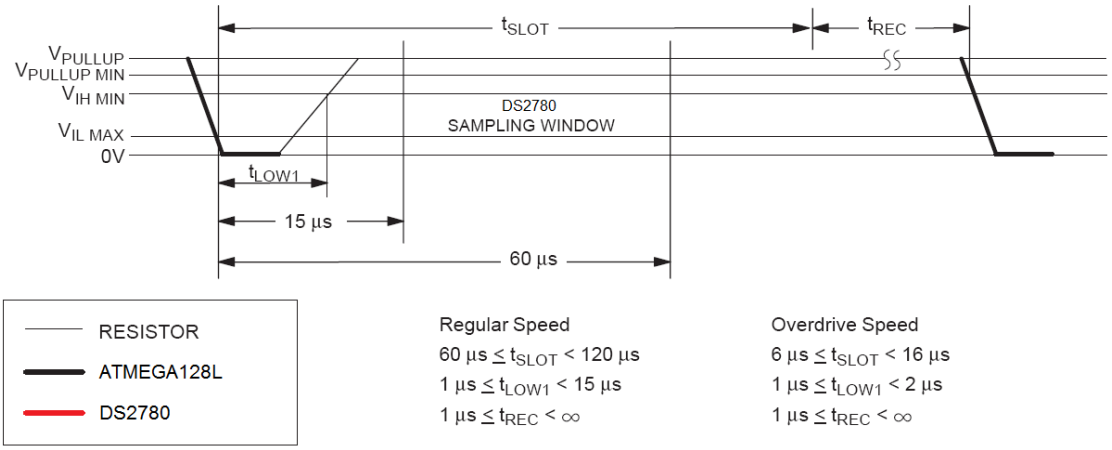


Figure 50. DS2780 write 1 timing slot

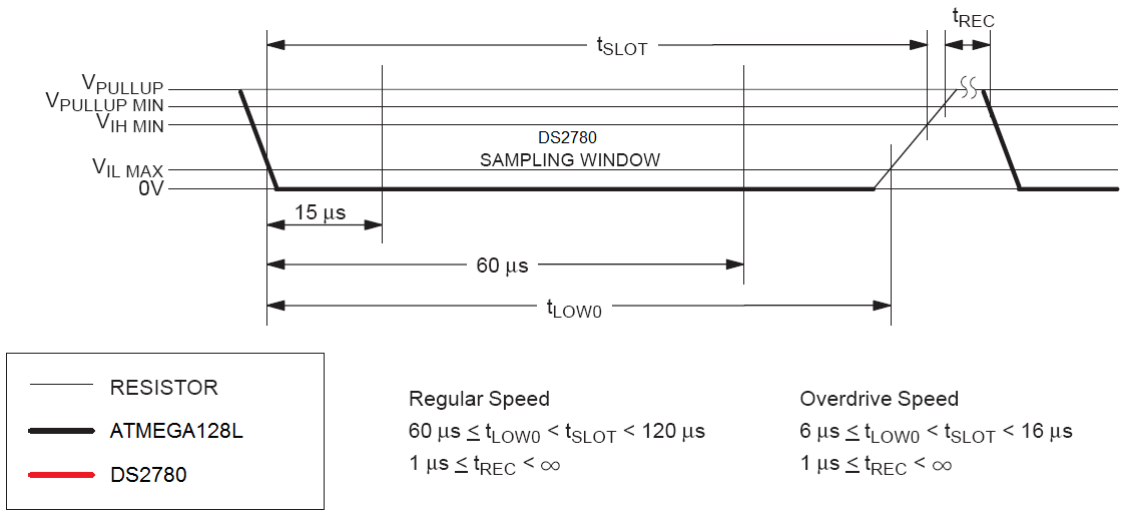


Figure 51. DS2780 write 0 timing slot

The duration of the active part of a time slot can be extended beyond 60 ms. The maximum extension is limited by the fact that a low pulse of a duration of at least eight active time slots (480 ms) is defined as a Reset Pulse. Allowing the same worst–case tolerance ratio, a low pulse of 120 ms might be sufficient for a reset. This limits the extension of the active part of a time slot to a maximum of 120 ms to prevent misinterpretation with reset.

At the end of the active part of each time slot, the DS2780 needs a recovery time t_{REC} of a minimum of 1 ms to prepare for the next bit. This recovery time may be regarded as the inactive part of a time slot, since it must be added to the duration of the active part to obtain the time it takes to transfer one bit. The wide tolerance of the time slots and the non–critical recovery time allow even slow microprocessors to meet the timing requirements for 1–Wire communication easily.

In this way, two functions are added to the *mau_serialid.c* file. The *write_bit* function which writes a bit to the one-wire bus, and the *write_byte* which writes a byte to the one-wire bus.

5.1.2.2 Read Time Slots

Commands and data are sent to the DS2780 by combining Write–Zero and Write–One time slots. As Figure 52 shows, to read data, the Atmega128L has to generate Read–Data time slots to define the start condition of each bit. The Read–Data time slot looks essentially the same as the Write–One time slot from the microprocessor's point of view. Starting at the high–to–low transition, the DS2780 sends one bit of its addressed contents. If the data bit is a 1, the DS2780 leaves the pulse unchanged. If the data bit is a 0, it will pull the data line low for t_{RDV} or 15 ms.

In this time frame data is valid for reading by the Atmega128L. The duration t_{LOWR} of the low pulse sent by the microprocessor should be a minimum of 1 ms with a maximum value as short as possible to maximize the Atmega128L sampling window.

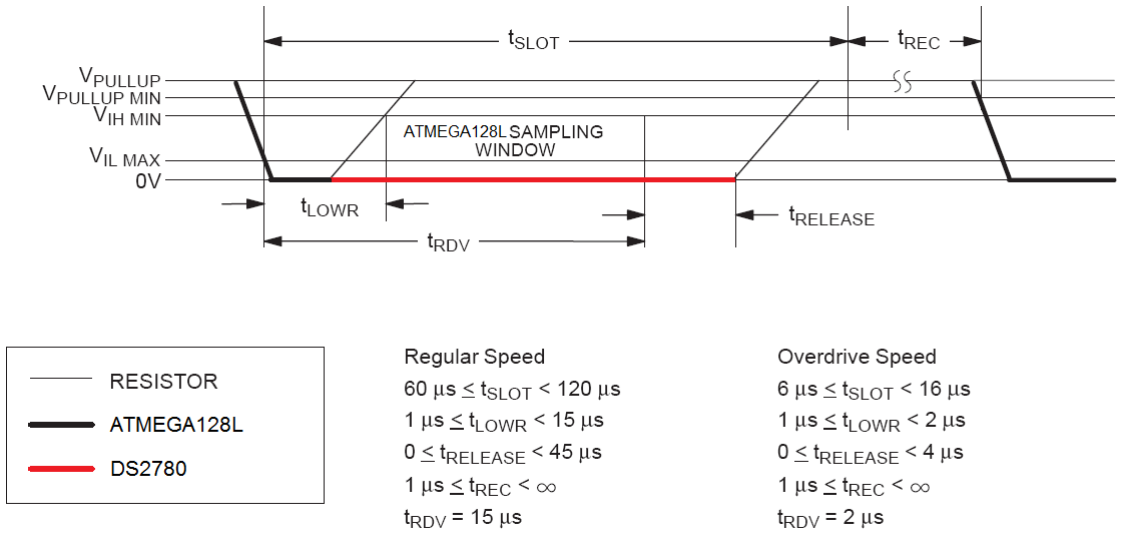


Figure 52. DS2780 read timing slot

In order to compensate for the cable capacitance of the 1–Wire line the master should sample as close to 15 ms after the synchronization edge as possible. Following t_{RDV} there is an additional time interval, $t_{RELEASE}$, after which the DS2780 releases the 1–Wire line so that its voltage can return to V_{PULLUP}. The duration of $t_{RELEASE}$ may vary from 0 to 45 ms; its nominal value is 15 ms.

In this way, two functions are added to the *mau_serialid.c* file. The *read_bit* function which reads a bit to the one-wire bus and the *read_byte* which writes a byte to the one-wire bus.

5.1.2.3 Presence Detection

As mentioned above, 1–Wire timing also supports a Reset Pulse. This pulse is defined as a single low pulse of minimum duration of eight time slots or 480 ms followed by a Reset–high time t_{RSTH} of another 480 ms (see Figure 53). This high time is needed for the DS2780 to assert its Presence Pulse. During t_{RSTH}, no other communication on the 1–Wire line is allowed.

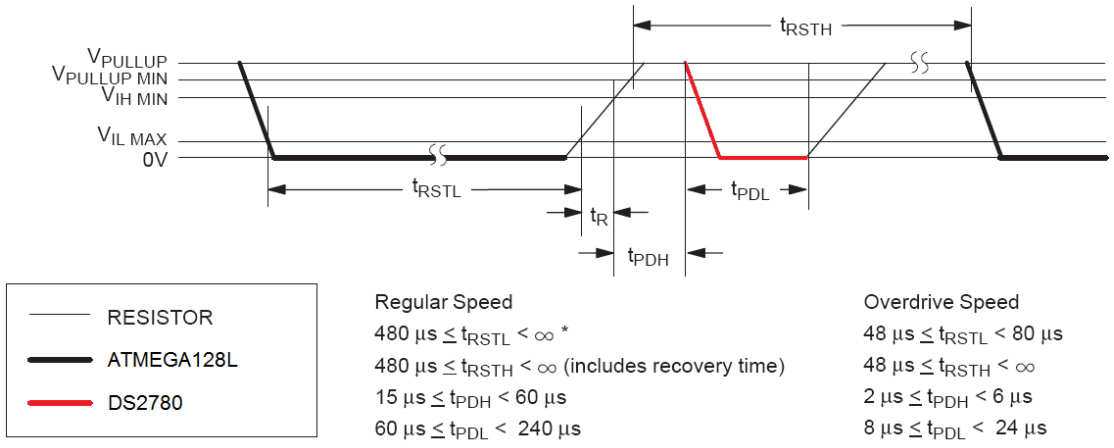


Figure 53. DS2780 reset/presence timing slot

The Reset Pulse is intended to provide a clear starting condition that supersedes any time slot synchronization. In an environment with uncertain con-

tacts it is essential to have a means to start again if the contact is broken. If the master sends a Reset Pulse, the DS2780 will wait for the time t_{PDH} and then generate a Presence Pulse of the duration t_{PDL}. This allows the master to easily determine whether a 1-wire component is on the line.

The *reset_onewire* function is added to the *mau_serialid.c* file in order to allow this functionality. Due to our line conditions and the number the devices, the timing is not very restrictive. Nevertheless, a regular timing has been chosen in order to make it compatible with future developments.

5.1.2.4 DS2780 Transmission Protocol

The 1-Wire bus requires strict signalling protocols to ensure data integrity. The four protocols used by the DS2780 are as follows: the initialization sequence (reset pulse followed by presence pulse), write 0, write 1, and read data. All of these types of signalling except the presence pulse are initiated by the Atmega128L.

To operate standalone as well as on a bus, DS2780 support the following ROM–based Networking Commands: Read ROM, Skip ROM, Match ROM and Search ROM. After execution of any ROM command, the Transport layer is reached. The command Read ROM, code 33H, is used to identify a device on the 1–Wire bus or to find out if several devices are connected at the same time. After sending this command the master has to generate 64 read time slots. The DS2780 will send its ROM contents least significant bit first, starting with family code, followed by the serial number and the CRC byte. If several 1-wire devices are connected, no reading will provide a matching CRC–byte. In this case, the command Search ROM F0H must be used to determine the ROM contents of the devices before they can be addressed. If the ROM contents are not of interest because there can be only one device on the data bus, the search can be skipped by sending the Skip ROM command, code CCH. Immediately after this command, the device reaches the Transport layer.

Usually the normal sequence of communication is:

1. Atmega128L: sends Reset Pulse.

   DS2780: sends Presence Pulse.

2. Atmega128L: sends ROM Command, possibly followed by data or read time slots.

   DS2780: listens or sends data.

3. Atmega128L: sends Memory Function Command, possibly followed by data or read time slots.

   DS2780: listens or sends data.

The flowchart of Figure 54 shows the process flow during the use of the networking commands in the DS2780.

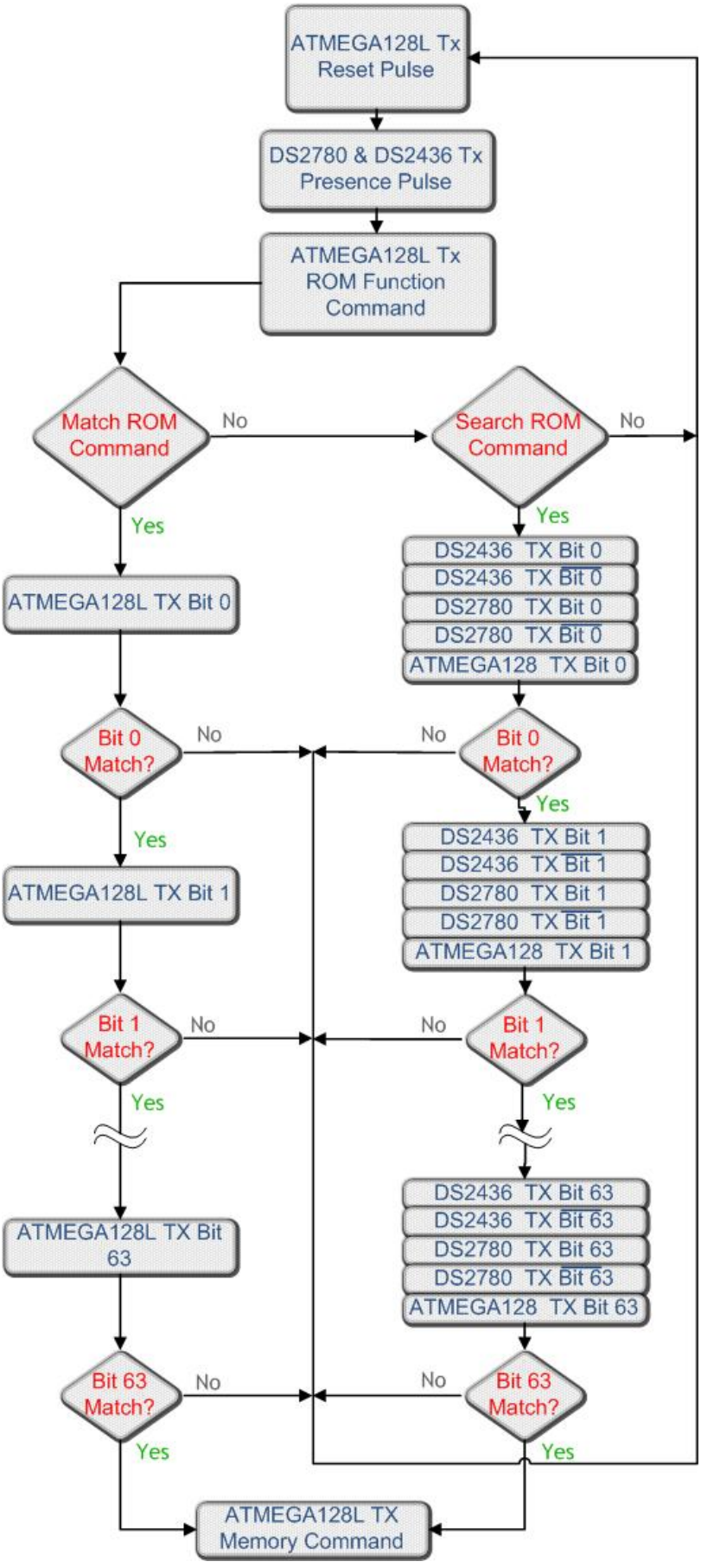


Figure 54. Flowchart of ROM functions used to communicate DS2780 with Atmega128L

However, in our case this sequence needs an extra step because in the 1-wire bus not only the DS2780 is connected but also the DS2436. So, the first step is to identify the ROM code of each device connected to the bus. In order to make the monitoring compatible with other possible AquisGrain 1.0 applications, the DS2436 ROM code will be used as the MAC address of the node, as it was before. Even if the master does not know the serial numbers of the devices connected to the 1–Wire bus, it is possible to address one single device at a time. This is done by using the command Search ROM, code F0H. This command acts like Read ROM combined with Match ROM.

All devices connected to the 1-wire bus will sequentially send the true and the false value of the actual ROM bit during the two Read time slots following the Search ROM command. If all devices have a 0 in this bit position, the reading will be 01; if the bit position contains a 1, the result will be 10. If both, a 1 and a 0 occur in this bit position, reading will result in two 0 bits, indicating a conflict. The master now has to send the bit value 1 or 0 to select the devices that will remain in the process of selection. All deselected devices will be idle until they receive a Reset Pulse.

After the first stage of selection, 63 reading/selecting cycles will follow, until finally the master has learned one device's ROM code and simultaneously has addressed it. Each stage of selection consists of two Read time slots and one Write time slot. The complete process of learning and simultaneous addressing is about three times the length of the Match ROM command, but it allows selection of all the connected devices sequentially without knowing the ROM values beforehand.

In order to implement this algorithm, three functions are added to the *mau_serialid.c*. The *first* function searches on the 1-Wire for the first device. This is performed by setting LastDiscrepancy and LastDeviceFlag to zero and then doing the search. The resulting ROM number can then be read from the ROM_NO register. If no devices are present on the 1- Wire the reset sequence will not detect a presence and the search is aborted. The *next* function searches on the 1-Wire for the next device. This search is usually performed after a *first* operation or another *next* operation. It is performed by leaving the state unchanged from the previous search and performing another search.

The resulting ROM number can then be read from the ROM_NO register. If the previous search was the last device on the 1-Wire then the result will be FALSE and the condition will be set to execute a *first* with the next call of the search algorithm. Finally, the *FindDevices* function begins with a 1-Wire reset to determine if any devices are on the net, and if so, to wake them up. The *first* function is then called, to keep track of the discrepancy bits and return to *next*, which finds each unique device on the net. Once, the first byte of one device is read, it compares it to the DS2780 or the DS2780 family and stores this value to use it later in the MATCH command. Figure 55 shows the flowchart of this algorithm.

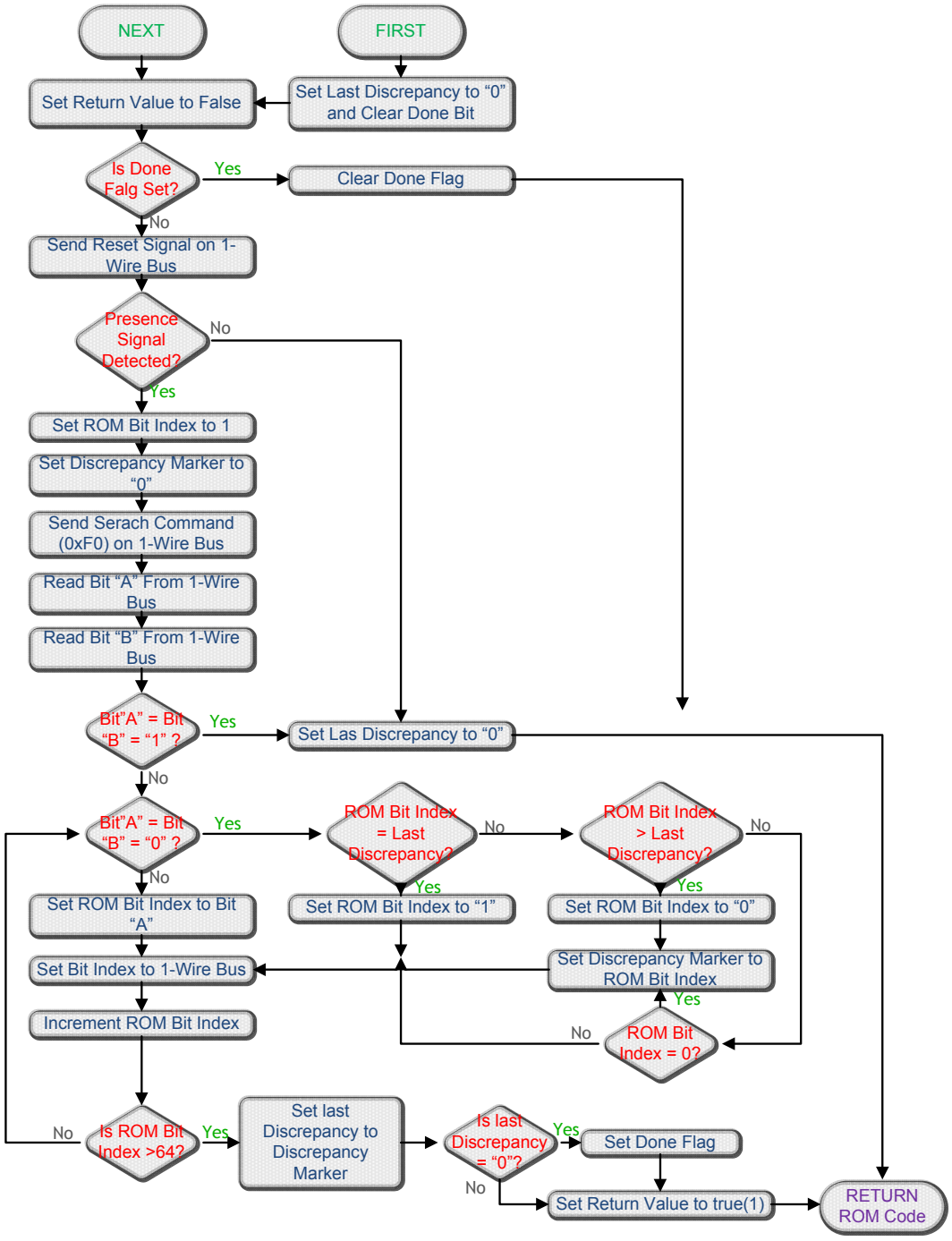


Figure 55. Implemented 1-wire devices search algorithm.

After the search ROM algorithm, we are able to distinguish the address which belongs to the DS2436 and the one which belongs to the DS2780. As they have different families, the first ROM code byte is different:

1-Wire Component	Hexadecimal Family Code
DS2436	1B
DS2780	32

Now, we can address to a particular device through the MATCH ROM command. The *Send_MatchRom* function is implemented to facilitate the access

to a particular device. Therefore, the monitoring application can access to the DS2780 only by executing *Send_MatchRom* previously.

Once the MATCH ROM command is sent, the application can access to the DS2780 EEPROM. In order to access any DS2780 register and return its value, the *Read_Register* function is implemented. This function needs six parameters to access to a particular register:

1. One-Wire component address: to access to a particular device connected dot the 1-wire bus. This address is previously obtained from the *FindDevices* functions.
2. Function command: to specify if we are going to read or to write in the register. The possible values are defined in the header *onewire.h*.
3. Register address: the particular EEPROM address where the desired value it has to be written/read.
4. Mask bits: number of bits of the register which don't contain information.
5. Register bytes: number of bytes that form the register. The DS2780 has two types of registers according to their length: 1-byte or 2-byte.
6. Sign bit: it indicates if the register has one bit reserved for the sign.

Hence, it is possible to access to the registers in the EEPROM of whichever 1-wire component, in order to read or write data through the *Read_Register* function added to the *mau_serialid.c* file.

5.1.3 Communication Protocol

According to the design requirements, information about the battery status has to be retrieved from the DS2780 and sent it to the user. In order to do that an AquisGrain 1.0 performing as coordinator of the network must be connected to the PC. Since the AquisGrain 1.0 board has no RS-232 interface a support board (MIB510CA) is used to adapt the AquisGrain's serial interface (see Figure 56). This board also supplies power and allows programming AquisGrain (also debugging with additional hardware).

Figure 56. MIB510CA+AquisGrain 1.0 used as coordinator

The PC runs the battery monitor application, which is part of the *EnergyManagement* solution. Another application called *Basumat* supplies the radio packets retrieved via the serial port from AquisGrain 1.0. Then, the energy messages received from the node are processed and the data is converted to the standard units.

A new message structure is created, named *EnergyMsg.h*, in order to allow the node to send the information about the battery status. This structure contains the necessary data to estimate the lifetime of the battery and the component energy usages:

- Message Type: which in the case of an energy message it is defined as 33.
- Group: this field is common to all AquisGrain messages and indicates the network group.
- Options code: this field is used to indicate to the coordinator when a successful message has been received. It is used as acknowledgment of the reception of other types of messages (policy messages or cell messages) and it is also used for polling the coordinator.
- Temperature: it contains the value of the DS2780 temperature register.
- Voltage: it contains the value of the DS2780 voltage register.
- Current: it contains the value of the DS2780 current register
- Remaining Active Absolute Capacity (RAAC): it reports the remaining battery capacity under the current temperature conditions at the Active Empty discharge rate (IAE) to the Active Empty Point in absolute units of milli-amp-hours.
- Remaining Active Relative Capacity (RARC): reports the remaining battery capacity under the current temperature conditions at the Active Empty discharge rate (IAE) to the Active Empty Point in relative units of percent. RARC is 8 bits.
- Time: it represents the time that the node has been working in absolute units of seconds.
- Radio On: it represents the time that the CC2420 has been in the normal working mode in absolute units of seconds.
- MMCU On: it represents the time that the ATMEGA128L has been in the normal working mode in absolute units of seconds.

Every 30 seconds an energy message is sent to the coordinator to update the current status of the battery. This way, if the current consumption changes the user will be notified and in the case that some policy had been set, the application will be able to reprogram the node with the new information. Figure 57 represents the energy messages exchange protocol during an energy update.

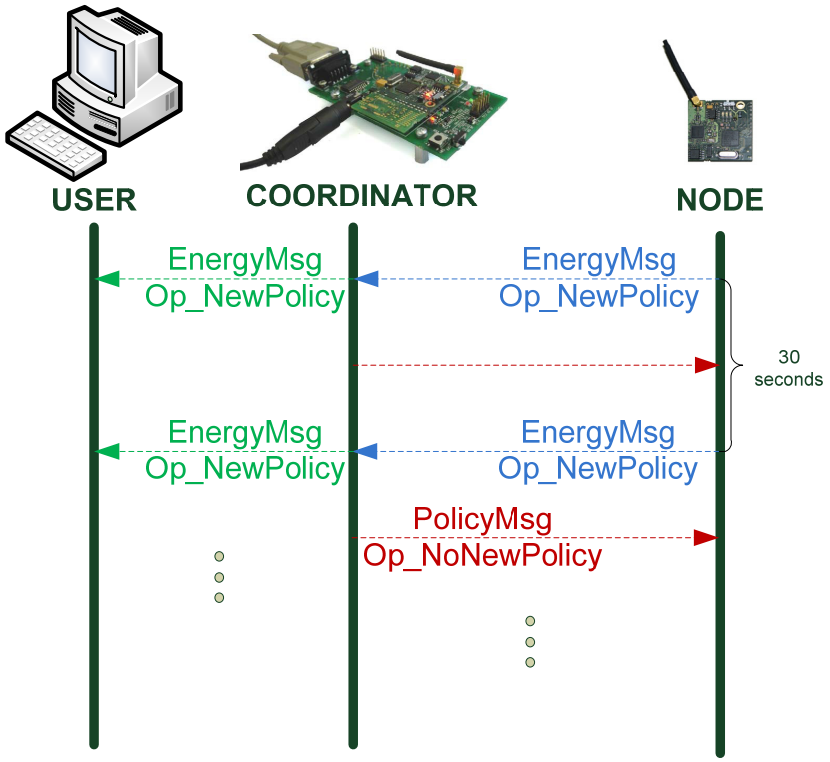


Figure 57. Messages exchange protocol during an energy update

On the other hand, according to the design specifications the user has to be able to set the battery characteristics through the PC application. In order to allow that, another message type is created, *CellMsg*. This structure contains the necessary data to specify the battery specifications needed to set the DS2780:

- Message Type: which in the case of a cell message it is defined as 35.
- Group: this field is common to all AquisGrain messages and indicates the network group.
- Options code: this field is used to indicate if the message is for retrieving the actual battery characteristics stored in the DS2780 EEPROM or it is for setting them up.
- Capacity: indicates the full capacity of the battery at +40°C.
- Charge Voltage: indicates the voltage at which the battery has been charged. It is used to know when the battery is fully charged.
- Terminating Current: indicates the current which the battery charger has used at the end of the charging process. It is used to know when the battery is running out.
- Empty Voltage: indicates the minimum voltage that our device needs to work in a proper manner.
- Empty Current: indicates the minimum current that our device needs to work in a proper manner
- Resistor: represents the value of the sense resistor which the DS2780 uses to measure the current consumption.

- Initialize Accumulated Current Register: used to indicate to the DS2780 that a new full battery has been placed, so it has to reset the ACR register.

The user request about the cell characteristics is completely asynchronous. According to energy power saver module which is controlling the ECG application, the radio is most of the time turned off in order to reduce the energy consumption. Hence, the coordinator cannot send the request message as soon as it is received. The coordinator must wait until the moment in which the node polls it. So, the message sending can be delayed a maximum of 30 seconds. This delay is feasible as the cell characteristics will be checked/set only once per battery usually. As soon as the node receives the request, it sends an acknowledge message to the user in order to let him know that the operation has been carried out successfully (see Figure 58).

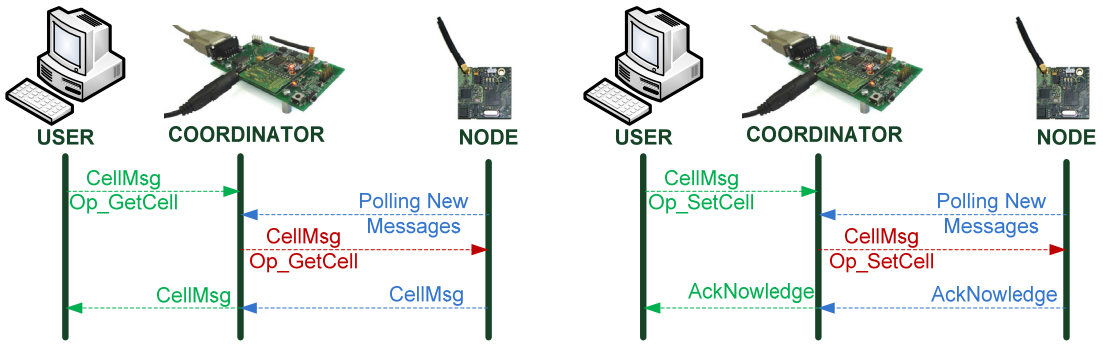


Figure 58. Message exchange protocol during (a) a cell characteristics request (b) a cell characteristics set up

The application *appECGpwrSave.c* includes in its makefile the *mau_powersaver.c* file developed to reduce power consumption of the ECG application. It has been integrated in the Chipcon MAC application framework and includes the basic functions and events common for all applications (appInit, appMainloop, appDataIndication, appDataConfirm and appExRecvSerial). Besides, it implements some additional functions that complement the generic ones. This is showed in Figure 59.

It must be noted that this file includes the functions to handle the message exchange due to the energy messages and the policy messages. The *Energy Sent* function uses the 1-wire functions implemented in the *mau_serialid.c* to read the DS2780 registers and create an energy message. The *PolicySent* function is used to report any change that occurs in the policy state. Finally, the *setPolicy* function implements the processing of the received policy messages. Therefore, according to the fields of the received policy messages this functions changes the ECG sampling rate, the CC2420 power transmission and enables/disables the transmission of the ECG stream/Heart rate.

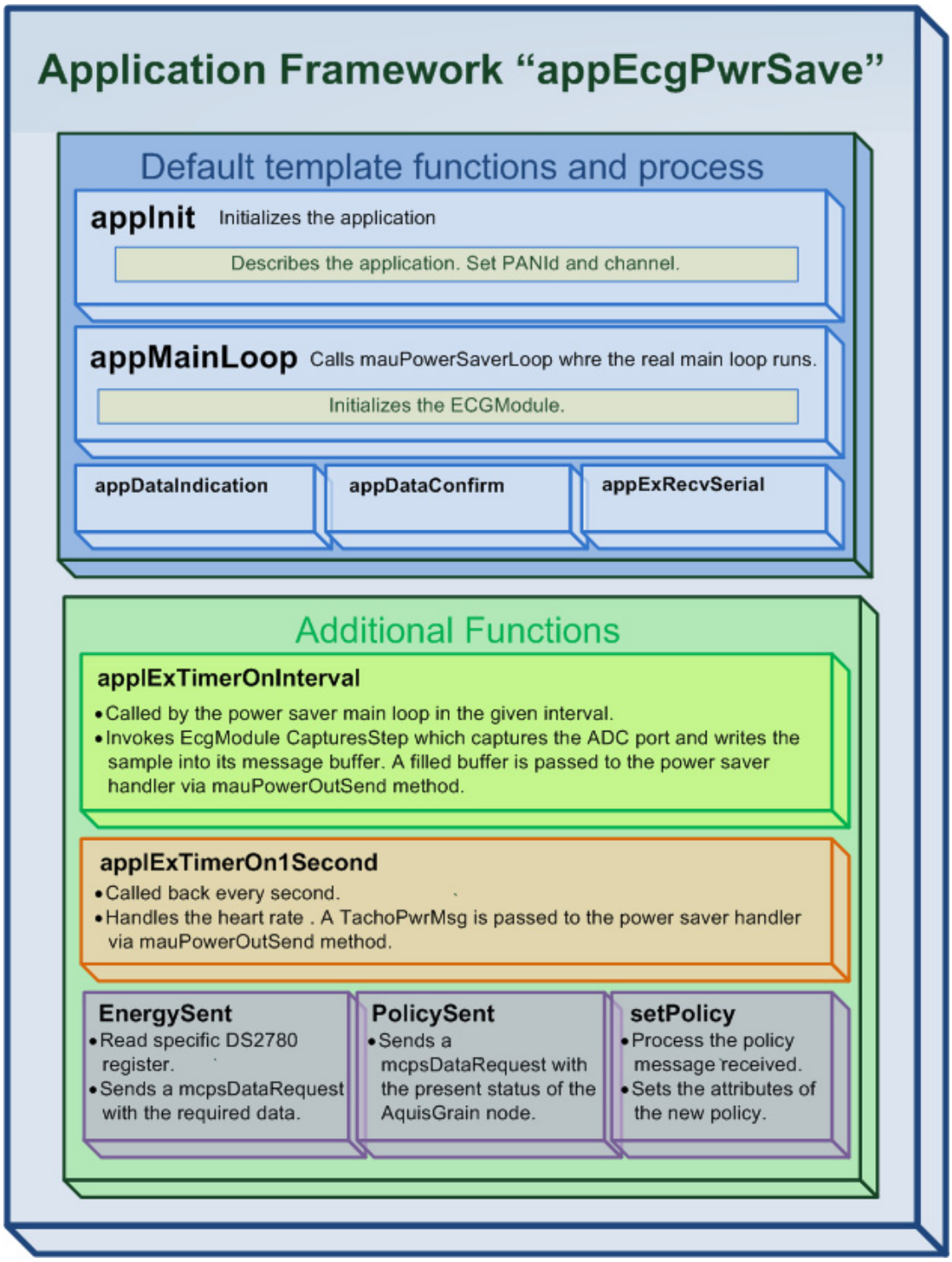


Figure 59. Structure of the appEcgPwrSave application

The first function executed is *appInit*, which is responsible for initialising the application itself. It first describes the application as ECG in the network and, after this, sets the variable *gAF_AppInfo.appCoordinator,* in the application information, to the value IAM_DEVICE. See flowchart in Figure 60.

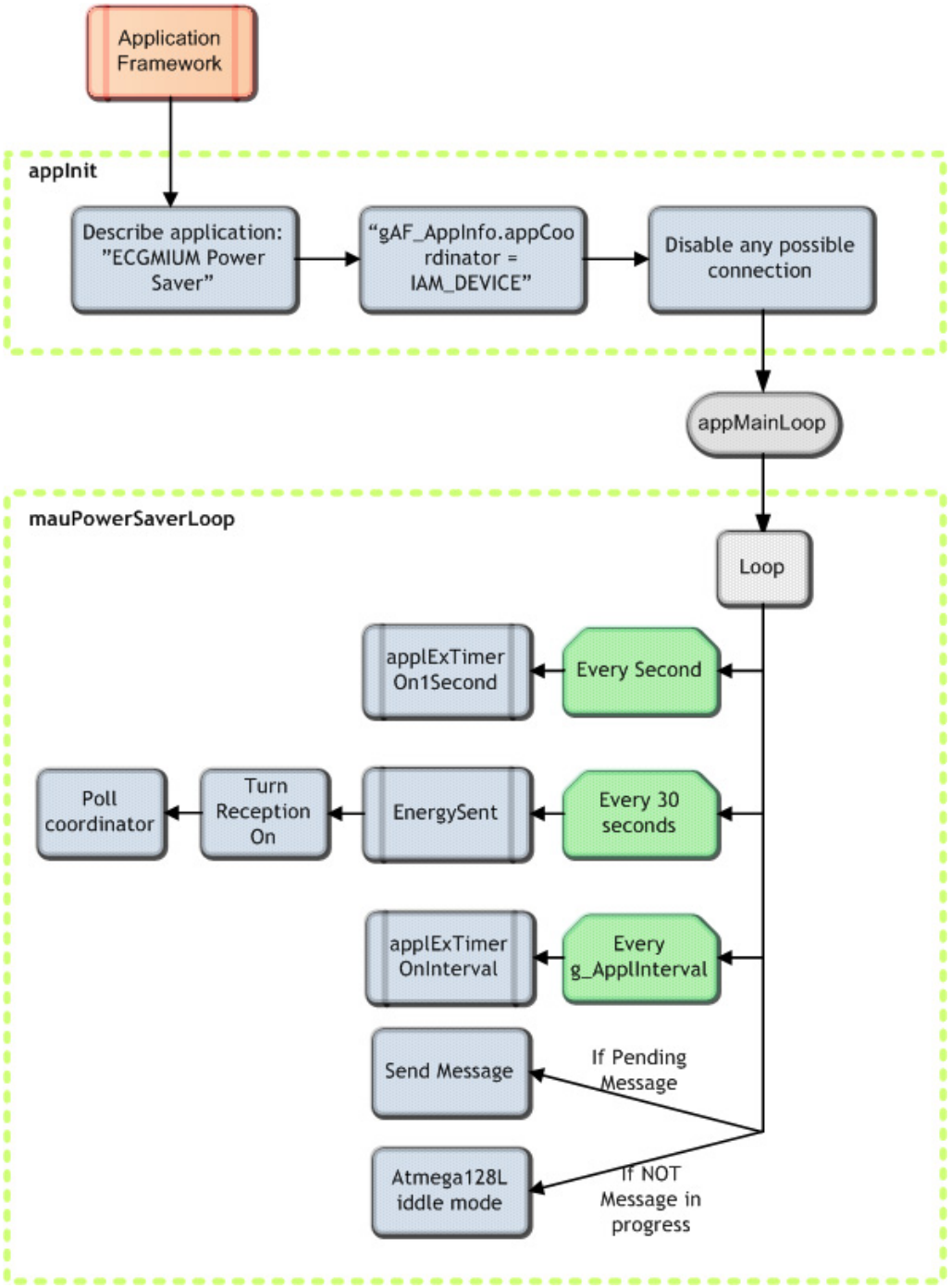


Figure 60. ECG application init and power saver module flowchart

The *appInit* is followed by the *appMainLoop* function, which call the *mauPowerSaverLoop*. It contains a never-ending loop where the power management takes place.

5.1.4 PC monitoring application

All the data retrieved from the battery which the node sends to the coordinator must be shown to the user. Moreover, the user must have the capability to ask the nodes for the battery specifications or set them. For all these reasons, a graphical user interface is needed.

In order to follow the previous Philips research projects guidelines, the chosen language for developing the PC application is C#. In this way, a single solution is developed to implement the whole PC side architecture. This solution includes three main modules: the battery monitoring, the policy specification and the energy manager. Each module runs in a different thread sharing some variables which allow the information exchange.

The battery monitoring graphical interface module allows the user know which is the present state of charge of the battery, as well as a lifetime estimation. Basically, it consists of two classes: *Battery.cs* and *BatteryInterface.cs*. The first class implements the main characteristics of a battery monitor. So the attributes are:

```
        // Battery temperature in degrees Celsius units
        private float temperature;
        // Battery voltage in absolute volts units
        private float voltage;
        // AquisGrain average consumption current
        private float averageCurrent;
        // Battery Remaining Absolute Active Capacity
        private float RAAC;
        // Battery Reamaning Relative Active Capacity
        private short RRAC;
        // Estimated Battery Lifetime
        private float ExpireTime;
        // Number of displayed bars
        private short bar_number;
        // Beeps when the RRAC is under 5.
        private bool Beep_stat;
        // Blink the last bar.
        private bool Blink;
```

Besides these attributes, the class implements methods to get and set the private ones, and two different constructors which allow creating a battery object with the data obtained from the DS2780 integrated circuit.

```
    // Default Constructor. Initializes the battery,
    // setting the battery as fulfilled, not blinking and
    // whithouth beeping.
    public Battery()
    {
      this.Blink = false;
      this.Beep_stat = false;
      this.bar_number = 9;
    }
    // Constructor to initializes the battery with specific values,
    // and the same graphical status than the default constructor.
    public Battery(float temp, float volt, float avgCurr, float RAAC, short
RARC)
    {
      this.temperature = temp;
      this.voltage = volt;
      this.averageCurrent = avgCurr;
      this.RAAC = RAAC;
      this.RRAC = RARC;
      this.ExpireTime = this.RAAC / this.averageCurrent;
      this.Blink = false;
      this.Beep_stat = false;
      this.bar_number = 9;
    }
```

On the other hand, the class *BatteryInterface* inherits from the *Windows.Form* class to implement the graphical items. Basically it is a display of the battery state and characteristics. In order to allow the other modules to get the information from the battery a *Battery* object is declared as protected and internal. This way each one of the three threads can access to information about the status of charge.

It is possible to distinguish several use cases of the energy monitor (see Figure 61). A first use case is when the user only wants to display the information about the lifetime and the energy status, so the application only must show the *Battery* object attributes. A second case is when the user wants to get the cell specifications so the application must send a request to the node and show the retrieved information. A third case, directly related to the previous one, is when the user sets the battery specification, so a cell message has to be sent to the node. The last case is when the user wants to know the evolution of the energy status of the node.

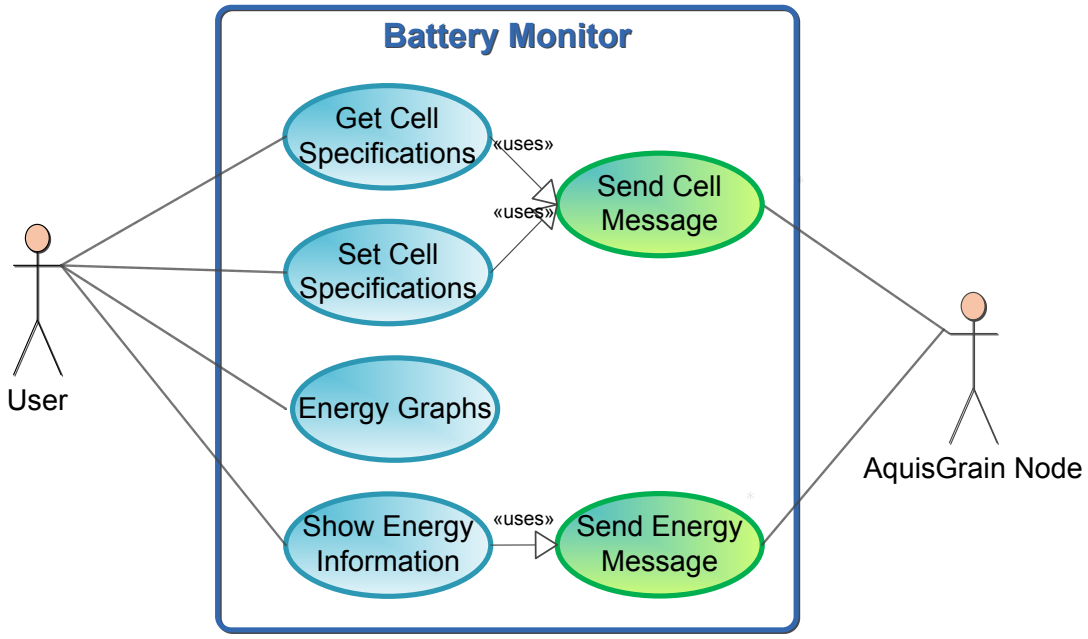


Figure 61. Battery Monitor application use cases

In order to display the energy information, the *BatteryInterface* class includes a timer of 1 millisecond interval which is in charge of updating the graphical percentage bars (see Figure 62) and the battery settings, and another timer of 1 second interval which checks if the battery state of charge is below the 5% percent. If the condition is true, the application switches to the blink states and emits an alarm to alert the user that the battery must be changed. In this way we succeed to inform the user about a critical state as the application can be minimized to the system tray so the user will not be able to see the graphical interface.

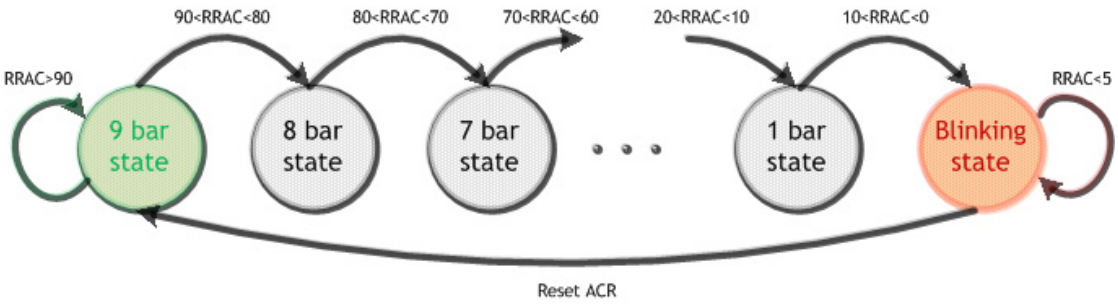


Figure 62. Battery Monitor GUI bar finite states machine

According to the first and second use cases, the *BatteryInterface* class allows to input the necessary data to create *CellMsg*. Then, the user can set or get the battery specifications that are stored in the EEPROM of the DS2780.

Next figure shows a screenshot of the final battery monitor application. It consists of three different parts: one for the battery state of charge, another for the battery energy parameters, and a last one for the battery specifications.

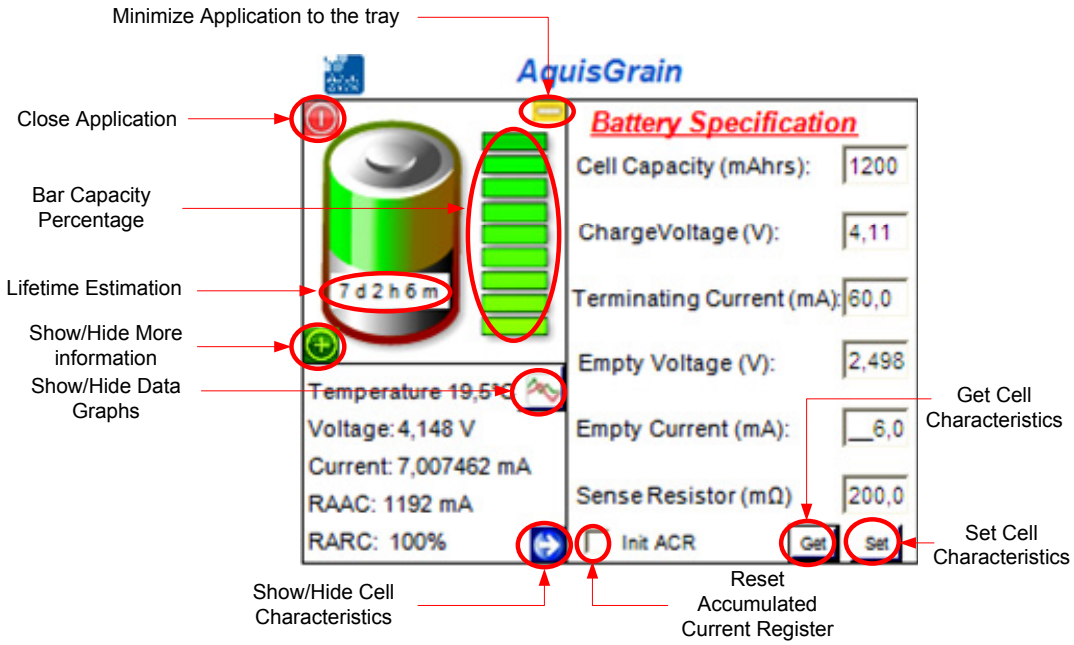


Figure 63. Extended battery monitor GUI screenshot

Finally, a database is necessary to implement the last use case. All the data since the node starts sending energy information must be stored to make it possible to draw a graph. From the several available databases technologies, SQL Server 2005 Express is chosen. The main benefits of this election are:

- Advanced query optimizer that automatically optimizes queries
- Computer manager for starting and stopping services
- Rich Database Functionality
- Transact-SQL support
- Full Text Search
- Always free to obtain and use

- Native XML data type
- XQuery support
- Deep Integration with Visual Studio 2005
- In-process data access with ADO.NET

This database is shared by the other threads, so that they are able to add or modify the data. Three different evolution graphs are represented: temperature, voltage and current. The database includes a table to store these three attributes besides the time.

In order to be able to draw graphs showing the evolution of the energy data, a new class is added to the package: *Graphs.cs*. This class mainly consists of a panel in which the graph is inserted. The user has the possibility to choose the parameter to graph. The free license *ZedGraph* library is included in the class in order to facilitate the graphing functions [29]. This library allows drawing XY graphs from a specific data type. The data in the graphs is updated in real time, so each time an energy message arrives the corresponding data is read from the database and represented in the graph.

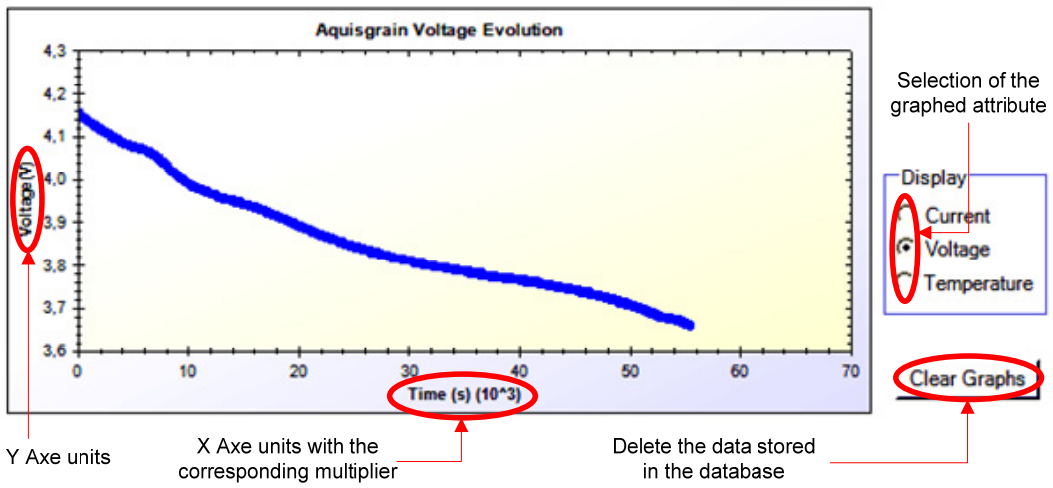


Figure 64. Graph from the battery GUI screenshot

Besides the possibility of being able to choose the parameter to graph, the user also can clear the graph panel by deleting the data stored in the database. Figure 64 shows the voltage during the battery discharge process.

5.2 Policy Specification

The implementation of this module relays, in a great deal, on the ECG application. Following the design and the requirements exposed in the previous chapter, the policy specification must allow the user to set a series of rules with priority levels. These rules are highly related to the application. Therefore, the language used must be defined according to the ECG application.

Analyzing the ECG application, the elements that can influence in the energy that the node is consuming are included in the language specification. These elements have to be understandable to the user. We cannot assume that the

user has programming knowledge. Each one of these elements has a series of possible values assigned. Therefore, the language specification is constituted by a series of elements and their corresponding values (see Table 11).

Element	Operand	Value
Sensor Sampling Rate	>,>=,=	200
	>,>=,=,<=,<	500
	=,<=,<	1000
Power Transmission	>,>=,=	0 dBm
	>,>=,=,<=,<	-5 dBm
		-10 dBm
		-15 dBm
	=,<=,<	-25 dBm
ECG messages transmission	=	Enable
		Disable
Heart Rate messages transmission	=	Enable
		Disable

Table 11. Possible ECG rule elements

Besides these elements, the user should be able to set up a specific battery lifetime. Therefore, the lifetime element is included in the list of possible rule elements. The value of the lifetime element is related to the battery capacity. So, the user has the possibility to choose between a series of values between the maximum and the minimum lifetime (see Table 12).

Battery	Nominal Capacity	Lifetime	
		Maximum	*Minimum*
Lishen SP0405AC	140 mAhrs	~21 hours	~11 hours
Varta 653450UC	1130 mAhrs	~13 days	~4 days

Table 12. Expected batteries lifetime under default mode

According to the elements number, only five different levels of priority will be possible to set up in the policy. However, as each level will typically consist of more than one rule, the most used policies will have up to three priority levels.

After user has entered the desired policy and it is processed, it is necessary to send the right commands to the nodes. So, a communication protocol it is necessary to manage these messages.

5.2.1 Communication Protocol

The design of the power saver application consists on reducing the time that the radio and the microcontroller spend in the normal mode. In this way, the radio reception is turned off most of the time. This fact forces the coordinator

to send messages to the nodes only in a specific period of time. So, the communication protocol between the coordinator and the nodes must be adapted to this restriction.

Moreover, a new type of message is necessary to send the policy information. The energy manager will process the policy set up by the user, and it will translate it to some commands that the AquisGrain application can understand. In this way, a new header file is implemented: *PolicyMsg.h*.

```
struct PolicyMsg
{
      // Generic AG message item specifying message type
      UINT8   msgType;
      // Generic AG message item specifying netowrk group
      UINT8   group;
      // Options code. Indicates the type of information inside the policy
      // message
      UINT8   opcode;
      // Sensor Sampling Rate value (200,500,100)
      UINT16  SSR;
      // CC2420 Power Transmission
      UINT8   PTX;
      // Enable/Disable transmission of heart rate
      UINT8   HeartRate;
      // Enable/Disable transmission of ecg
      UINT8   ECGStream;
};
```

The ECG application is modified in order to allow the possibility of applying a policy. Two new functions are added to the *appEcgPwrSave.c*: *PolicySent*, *setPolicy*.

The aim of the first function is to send the current policy of the AquisGrain node. With this information, the user will know the present state of the node before setting up another policy. The *PolicySent* function behaviour consists of reading the different elements which are possible to change with a policy. With these values, a new *PolicyMsg* is created and it is send to the coordinator through the MAC primitive *mcpsDataRequest* (see Figure 65).

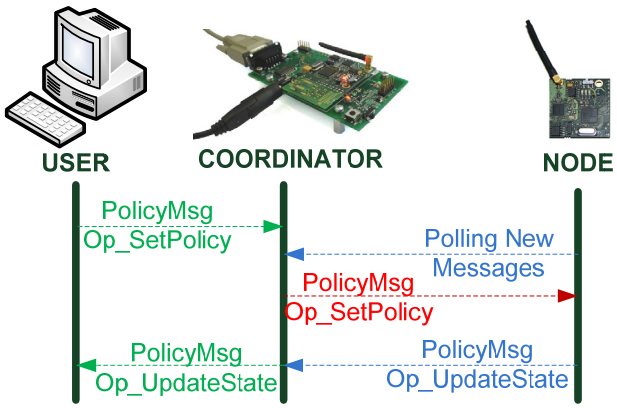


Figure 65. Messages exchange protocol during a policy setting up

The *setPolicy* function is implemented in order to process the reception of a policy message from the user. Once the message is received in the node, the value of the policy elements are changed with the ones specified in the message. By means of a series of control variables, the state of the policy ele-

ments are managed. Figure 66 shows the flowchart of the application running on the node which control, among other things, the policy.

However, managing the CC2420 transmission power requires to access to the MAC primitive *msupSetTransmitPower*. This primitive requires changing the value of the structure attribute *ppib.phyTransmitPower* with the desire transmission power. All the values that this attribute can have are declared as constants in the common libraries. So, the *setPolicy* only sets the constant that corresponds to the received value,

Therefore, the node can received two types of messages: policy messages or cell messages. Depending on the type and the options code, the application takes the corresponding decisions.

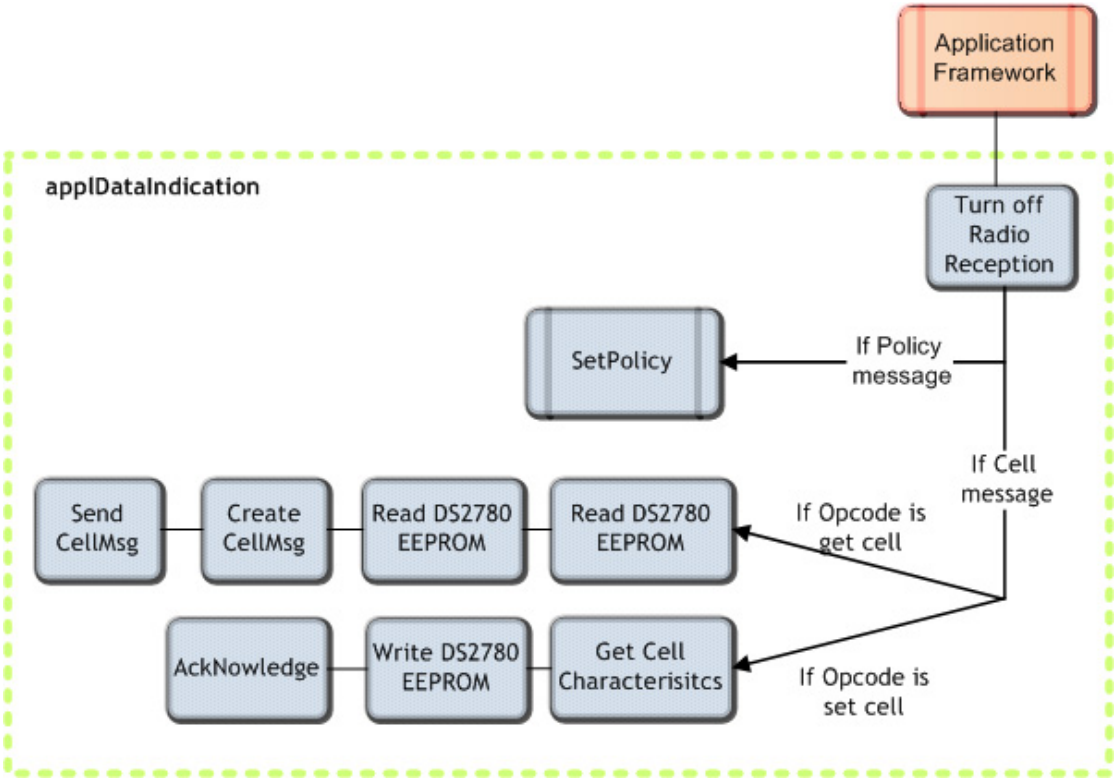


Figure 66. Node's *applDataIndication* flowchart

5.2.2 PC Policy Specification application

The main goal of the policy specification application is to allow the user to enter the desired policy. As explained in the previous chapter, this policy is specific for the ECG application. This involves a big constraint when specifying the policy, as the different types of elements are known in advance.

As the ECG restricts the policy elements, the policy specification application will show the different elements which can be used in the rules. This avoids the possibility of implementing a complex parsing algorithm, as we are using the information that we know in advance. Therefore, simple combo boxes are shown to let the user set up rules.

Each rule element has a series of specific values. So, the element must be selected before the value. In this way, the application does not allow to select

any value if an element is not selected. The same reasoning is done with the operands.

In order to implement the concept of rule, a new class is created: *Rule.cs*. This class is composed by a series of attributes which compose a rule, and the methods to access them.

```
// Priority level of the rule in the policy
private int priority;
// Transaction name of a named ECG account
private int element;
// Binary operator {<,>,=,<=,>=}
private int operand;
// Element value
private string val;
// Logical operand {AND,OR}
private int logical;
```

In the same way, a class named *Policy.cs* is implemented. Basically, this class is composed by the definition of a "level" structure, and the different kinds of enumerations that are possible while setting up a rule. These enumerations facilitate the translation from the user interface policy to the policy object. In addition, all the methods which allow a translation from the graphic interface to the message format are implemented in this class.

```
public struct Level
 {
      /// Number of rules which the level contains
      public int numRules;
      /// Logical operand which links the next level.
      public int ANDOR;
      /// Rules that composed the level
      public Rule[] rules;
 }
```

Therefore, a user policy is composed of one or more levels which in turn are composed of one or more rules. Figure 67 shows a policy example which is composed of three levels each one which a different number of rules. Each level is linked to the next one through a logical operator (AND, OR) implementing the concept of priority. So do the rules.

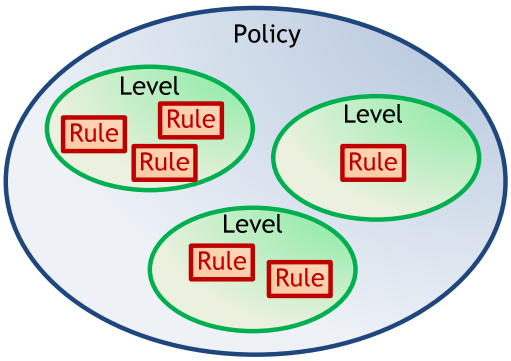


Figure 67. User policy architecture

The graphic user interface is implemented in the class named *PolicyInterface.cs*. It inherits from *Windows.Forms*. The graphic interface is composed by a series of combo boxes, and three buttons: one for adding a rule, one for deleting a rule, and another one for enabling the logical level setting up. Once

a rule is set up, the add rule button is enabled as well as the set logical button. So, the user is only able to add a new rule once the last rule is established. When the user finishes introducing the policy rules, then it must click the *set* button. Afterwards, the logical relationship between each priority level has to be set up. After the whole policy is introduced, a button appears allowing the user to send the policy to the energy manager. Figure 68 shows a screenshot of the policy GUI where a user has introduced several rules.

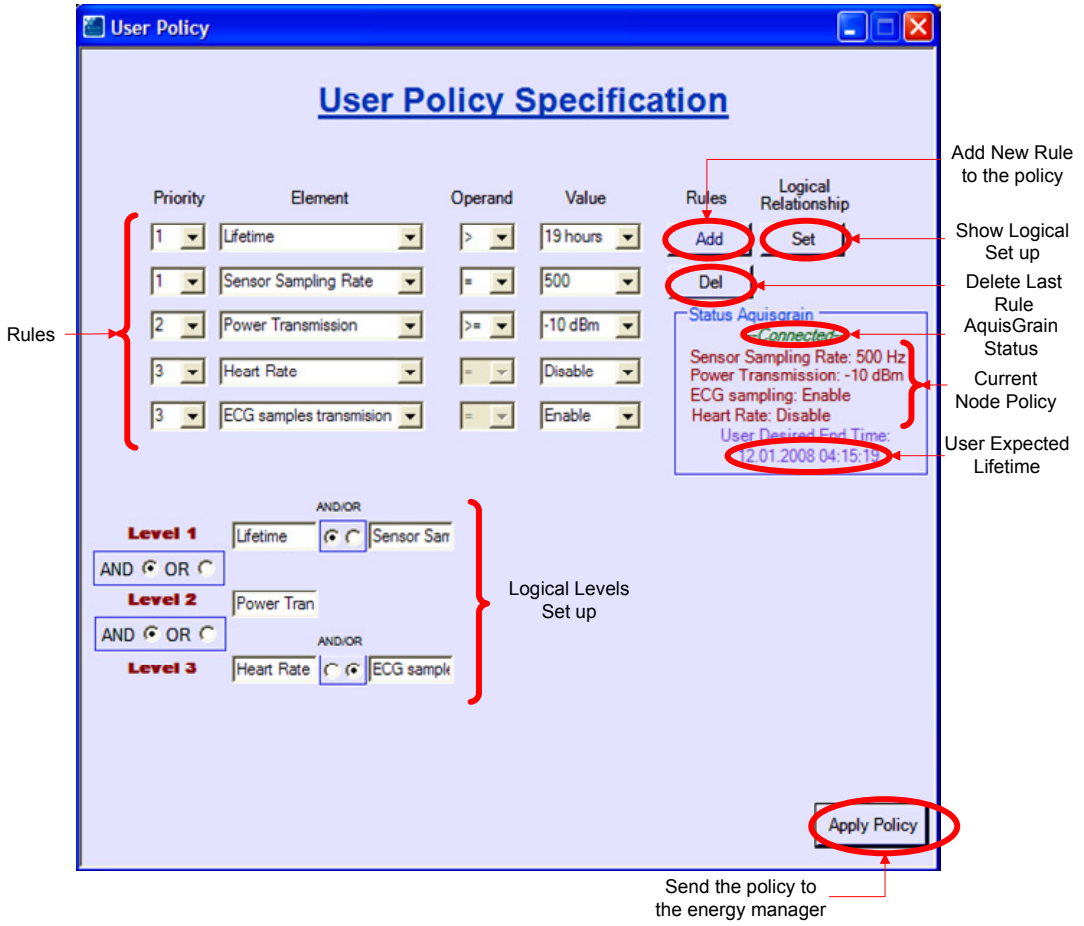


Figure 68. Policy graphical user interface screenshot

The policy introduced by the user is stored as a *Policy* object. This object is declared a protected and internal, in order to allow other modules to access it. In addition, an event is raised when the *Apply Policy* button is clicked. So, the energy manager can subscribed this event, and process the policy.

On the other hand, the GUI also shows the current policy which is set up in the node at the present. So, the user can know at any time which is the status of the node. This information is received with the same rate than the energy data. The node every 30 seconds sends a policy message with the options code set to *Op_UpdatePolicy*. In this way, the node takes advantage of the CC2420 power mode to send two different types of messages. So, it is saved some power due to the fact that only a power mode change is necessary. 30 microseconds are spared, which means that 30 nA are saved every 30 seconds.

Finally, it should be noted this class is set as the main class of the solution. So, during the initialization, two more threads are created: one for the energy

manager and another one for the battery monitor.

5.3 Energy Manager

As the battery monitor, the energy manager works in the PC side as well as in the node. It is in the PC where the user introduces the policy, so the energy manager has to process the policy and returns the value of the elements which best fit the policy. Inside the node, the power saver designed in the previous chapter is controlling the ECG application in order to save as much energy as possible.

5.3.1 Node Power Saver application

According to the design requirements, this application controls the ECG application, so it is called in the *applInit* function of the *applEcgPwrSaver.c* (see Figure 69). After this invocation, the *mau_powersaver.c* controls the process flow .

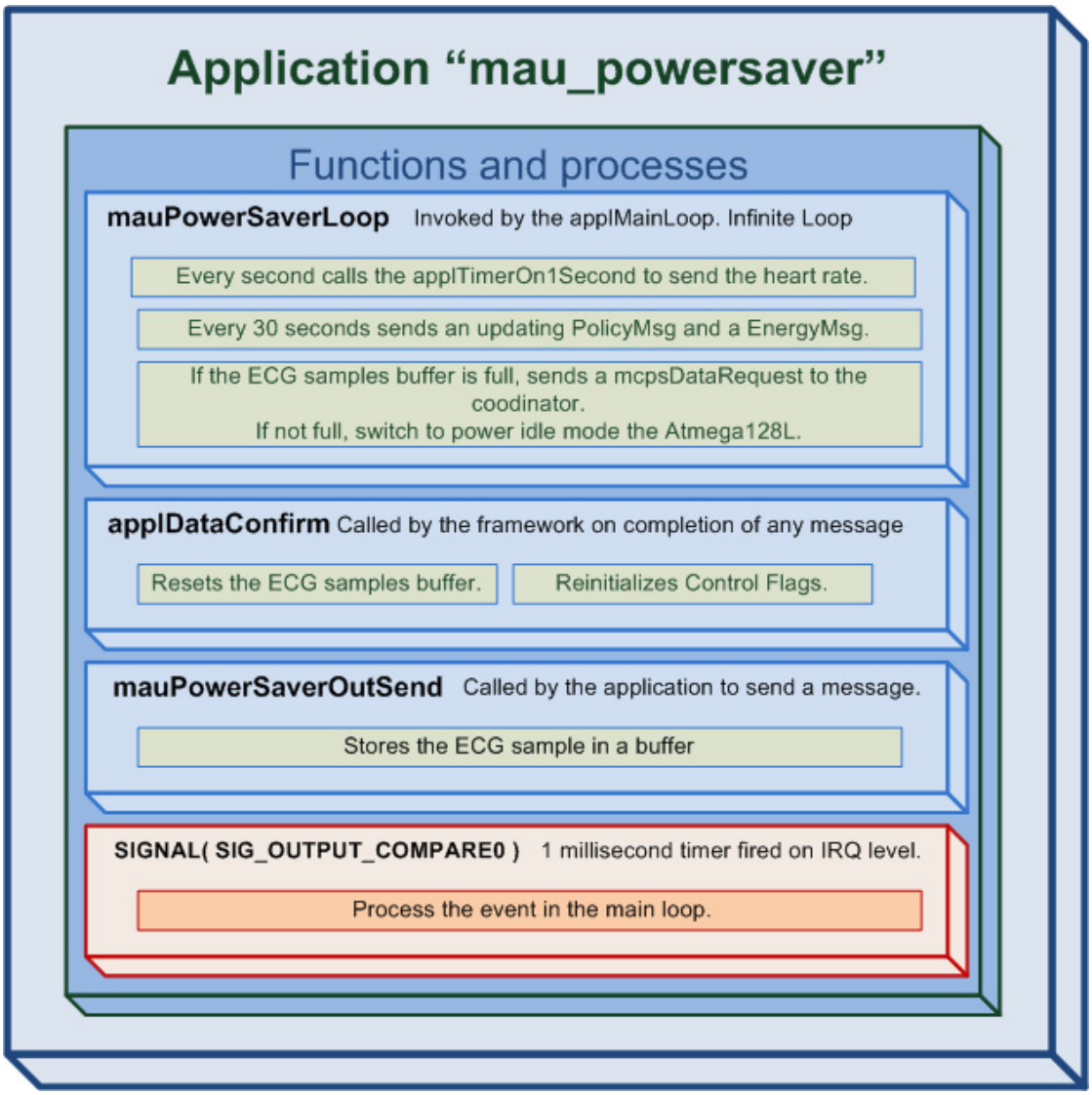


Figure 69. Function contained in the *mau_powersaver.c* file

The power saver loops acts with a 1-milisecond timer. Three counters are used for controlling the requested application interval, the energy messages rate, and the heart rate. Interrupt mode is enable for the timer 0, because it must be able to wake up the microcontroller. The main loop is triggered by the timer 0 interrupt. Also any other interruption on UART, MAC timer, radio, etc. wakes up the Atmega128L. Messages in the sender queue are identified by a MAC Service Data Unit handle, defined by the ECG application. A queue entry is inserted by application call *mauPowerSaverOutSend* and removed on confirmation by *mauPowerServerOutConfirm*. Pending messages in queue with same MAC Service Data Unit handle are overwritten.

In order to change the power mode of the atmega128L, the SE bit in MCUCR must be written to logic one and a SLEEP instruction must be executed. If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, it executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the register file and SRAM are unaltered when the device wakes up from sleep. The power idle mode is set by writing a logic zero to the SM2, SM1, and SM0 bits, which select the sleep mode.

```
cbi(MCUCR, SM2);
cbi(MCUCR, SM1);
cbi(MCUCR, SM0);
sbi( MCUCR, SE );// do enable power reduction
asm volatile ("sleep\n\t" ::);      // enter power reduction
cbi( MCUCR, SE );// do enable power reduction
```

Figure 70. C Code for switching ATMEGA128L to power idle mode

The above code is called every time there is no message pending to be sent. Therefore, most of the time the microcontroller is in the power idle mode.

In the same way, in order to switch the power mode of the CC2420 it is necessary to invoke the MAC primitive *mlmeSetRequest*. This primitive has as parameters the MAC_PIB_ATTR, which is the attribute to be changed, and a pointer to the PIB attribute. In our case the MAC_PIB_ATTR will get be assigned to MAC_RX_ON_WHEN_IDLE. So, depending on the pointer value (true, false), the CC2420 switches the power mode.

Only after a poll message is sent, the radio reception is turned on. After a poll message, the coordinator responses the node with the corresponding message. If no message was pending, a *Op_noData* policy message is sent. After the node receives the coordinator notification, it turns the radio reception off. This way, the radio power consumption is optimized.

With this module all the timers are running, including the TIMER1 which is the heart of the 802.15.4 MAC. It should be stated that this power saver is not foreseen for applications with short measurement and very long sleep times. This is specific for medical sensor applications.

5.3.2 PC energy manager application

Once the desired policy is introduced, it has to be processed. As it was specified in the previous chapter, all the policy processing is done in the PC as no special energy constraints are expected in the computer. Therefore, continu-

ing the development of the solution which includes the battery monitor and the policy specification, another class is added: *EnergyManager.cs*.

The main aim of this class is to translate the policy introduced by the user, to a series of values which are understandable by the ECG application running in the node. Moreover, this class implements the serial communication with the coordinator.

5.3.2.1 Serial Communication

In order to make possible the communication between a computer and the MIB510CA, Philips Research has developed an application which forwards the message received in the coordinator to the PC, the so-called Basumat Software (see Figure 71). This application also graphs the ECG streams, among other things.

Furthermore, Philips Research has developed a C# library which implements the serial communication between the PC and the Basumat Software, the so-called *AGBase* library. This library is added to the global solution in order to allow our *EnergyManagement* solution to communicate to the coordinator.

So, the *EnergyManager* class inherits from the *IBasumatClient*. In this way, our application connects to a *Basumatclient* which is instanced during the application initialization, so it can send messages to the coordinator and all the messages that the coordinator received are forwarded to our application. Two methods are inherited from this interface: *MessageReceived* and *MessageExit*.

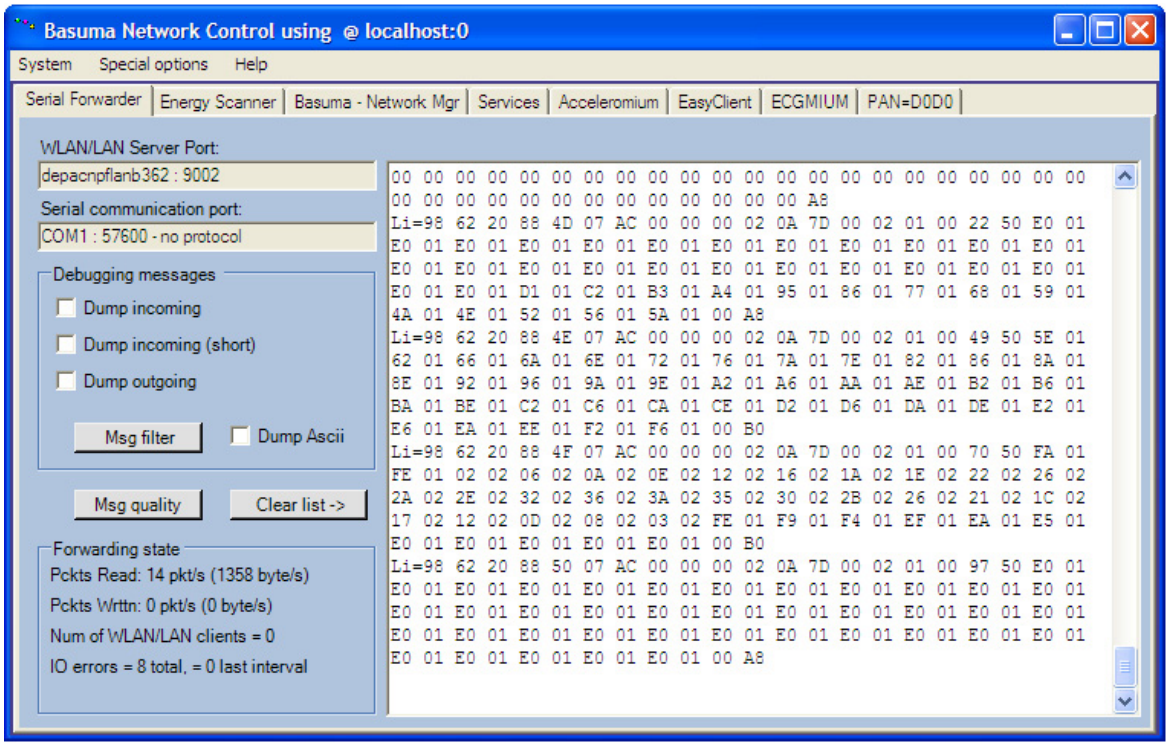


Figure 71. Basumat serialforwarder screenshot

On the one hand, the *MessageReceived* method is called when a message is forwarded to the application. This method processes three different types of

messages: policy, cell and energy messages. If it is a policy message, the policy current status is updated in the policy GUI. In the same way, if it is a cell message the information about the battery characteristics is updated in the battery monitor GUI. However, if it is an energy message a complex process is carried out. First of all, the shared battery object is updated with the new data, except from the average current. The current data in the message is stored in a buffer. When this buffer is fulfilled with 100 current measurements, the average is done. This average current is stored in the shared battery object. Moreover, all the energy messages are stored in a database table in order to allow graphing them later.

On the other hand, the *MessageExit* method is called when the connection to Basumat client is lost. Therefore, the application destroys *IBasumatClient* object and closes the entire environment.

Finally, in order to send a message via serial, we only have to invoke the *SendMsg* method over the *IBasumatClient* object. We have to pass as arguments any type of message which inherits from *Struct_TOS_Msg*. This class defines the general structure of an AquisGrain message.

5.3.2.2 Processing Policy Algorithm

Once the policy object is created, the energy manager processes the rules having into account the logical operator which link them and their priority.

The algorithm to process the policy requires several steps. While setting the relationship among the different rules, two different logical operators can be chosen: AND, OR. When the relationship is OR, two different paths are created (see Figure 43). Then, the algorithm has to evaluate each path and determine which the less energy restrictive one is. Moreover, the levels can also be linked with OR operator. So, more different paths are created and, each one of them with certain priority.

In order to evaluate which current consumption is associated to each possible different policy, the application needs to know in advance the different current consumption states. Therefore, before being able to set a specific lifetime a series of measurements must be carried out. The results obtained are stored in a table inside the same database where the DS2780 data was stored (see Appendix B – Policy State Current Measurement). Then, when the algorithm requires knowing which state is the one that entails a precise current consumption, it is only necessary to query the database with this current. So, we are able to evaluate which will be the estimated energy consumption of each different path of the policy thought querying the database.

One particular policy case that should be noted is when a specific lifetime is set. The translation from the desired battery lifetime to specific element values of the ECG application can be done by comparing the current consumption with the desired one. Once the desired current is calculated, the elements values are obtained from the database. If the desired current is not achievable, a pop-up window is shown and the best approximated policy is applied.

The major drawback of the algorithm is that its performance relays on the database information. So, this data has to be as accurate as possible. However, at this point a paradox appears. On the one hand, for predicting the

lifetime we need measurements very precise. On the other hand, the current consumption depends on the environment conditions (interferences, obstacles, distance, temperature…) largely. That paradox is solved by becoming adaptative to the environment. So, the first measurements, which are stored in the database, are replaced by the new ones if measuring during 100 samples the average is not in the range of ±50 μA. After that database updating, we can assure that the information is enough accurate to estimate the lifetime. Next figure shows the flowchart of this algorithm.

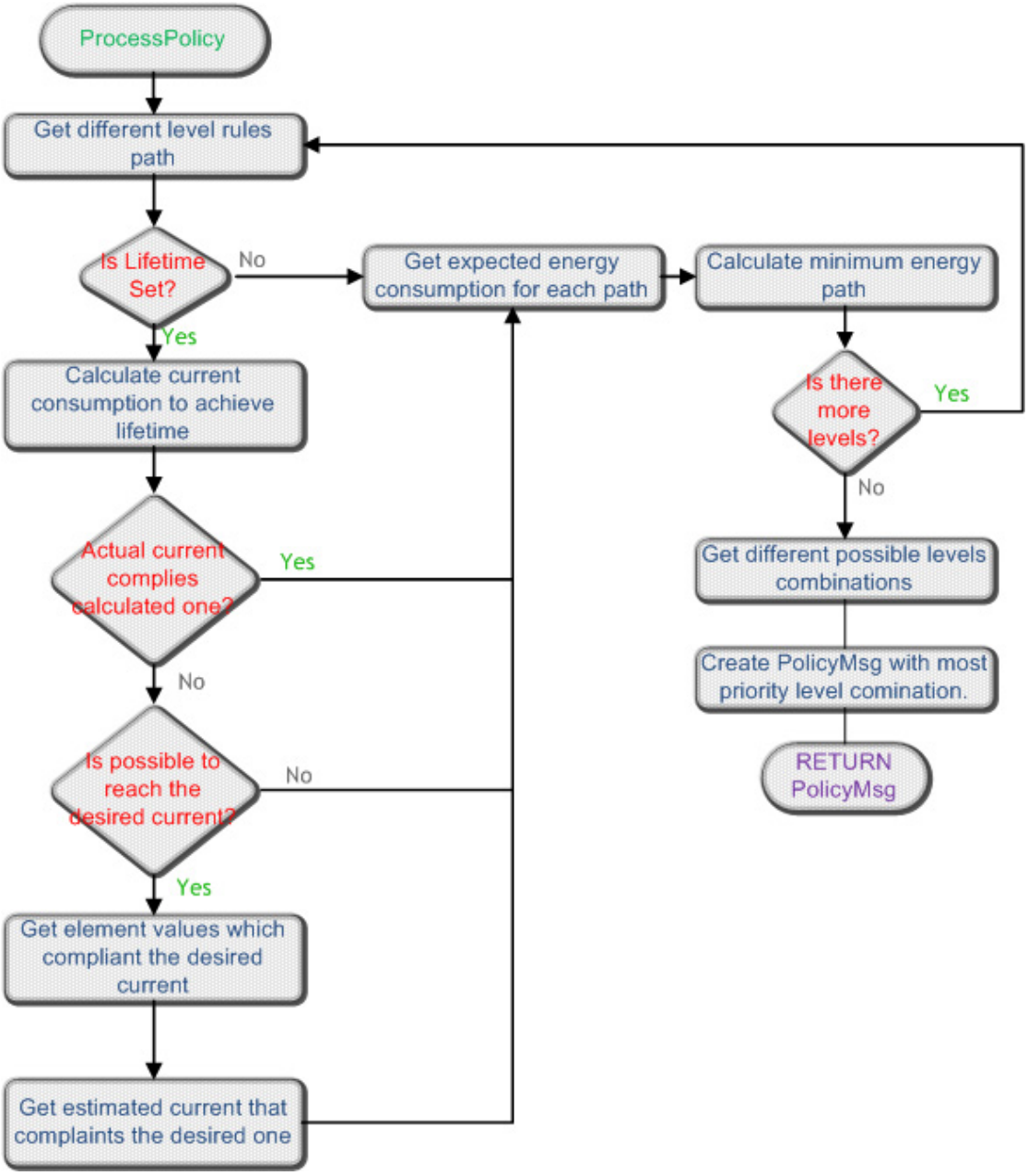


Figure 72. Processing policy algorithm flowchart

Due to the environment changes, the current consumption in a particular policy state can change, as it was exposed. So, it may happen that the lifetime set in a previous time could not be achieved with the applied policy. In this case, two options could be possible. The first one is that due to the current change, the specified lifetime could not be reached. Then, the application

notifies that fact to the user and sets best approximated policy. The second possibility is that the desired current could be achieved by changing some element values. So, as the database is adaptative to the environment, in this second case the requirements are achieved by processing again the user policy. In order to take into account this, a timer is implemented in the application which checks every 30 seconds if a current change has occurred and reprocesses the policy.

Finally, after running the processing algorithm, a *PolicyMsg* is created with the elements values that compliant the policy. Then, this message is sent to the coordinator via serial through the *IBasumatClient* object. In this way, the energy management architecture changes the application state which is working in the node according to the user specifications and the energy state.

6 Evaluation

Several tests are carried out in order to evaluate the performance of the proposed architecture. First of all, the hardware of the architecture is tested. Our architecture mainly relies on the accuracy of the measurements obtained by the DS2780. So, the DS2780 performance is evaluated under different test. After evaluating that the fuel gauge, the whole application is evaluated altogether.

6.1 Testing and Calibrating a DS2780

The first step after making the printing circuit board corresponding with the circuit design of the DS2780 is to place the DS2780 registers into known states and determine if there are any direct shorts on the board. Successful communication between the AquisGrain node and the device in this step verifies the DQ connection, as well as battery positive to the VDD and PACK+ pins, and battery negative to the VSS and PACK- pins.

Running this test we ensure that no interference is being produced in the one-wire bus between the DS2436 and the DS2780. So the *FindDevices* function gets the DS2436 ROM address and uses it as MAC address of the node, and it also gets the DS2780 to later use.

After that, we verify VIN pin connection to battery positive (see Figure 39). In this case, the latest voltage from the DS2780 is read to ensure that the connection is satisfactory. The procedure is the following:

1. Force 4.0 V from BAT+ to BAT-, connecting one of the selected batteries fully charged.
2. Wait 10 ms, executing the *halWait* function with.
3. Read 2 bytes voltage register, through the *Read_Register* function. In this function we specified the LSB voltage register address (0x0C) and the Read Data command (0x69).
4. Send test message to the coordinator, specifying DS2436 and DS2780 ROM address and the read data.

The results obtained are:

DS2436 ROM code	1B-28-2E-43-2A-AC-AC-68
DS2780 ROM code	32-FB-0F-12-00-00-00-B4
Voltage	4,197 V ± 4.88 mV

Analyzing the results, they are quite reasonable. The ROM codes should be composed of 8 bytes, and the first byte should be the family code. In our case the ROM codes agree with the theory, as the DS2436 family code is 0x1b and the DS2780 family code is 0x32. On the other hand, the voltage obtained is a valid measurement as 4.2 V is the value expected from a fulfilled battery. Measuring the battery directly with a voltmeter, the result obtained (4.201 V) is in the resolution margin.

Next step is to verify SNS pin connection. This connection is verified with a valid current measurement. While validating the capacitor is difficult, a successful current measurement will prove that the pins are not shorted. The same procedure carried out with the voltage register reading is followed. The result is a current measure of 8,612 mA ± 7,1002 µA. From a first approach to the current measurement, the value obtained is acceptable. However, measuring the same environment with an amperemeter, the result obtained (8,75 mA ± 5 µA) is out of the resolution error.

6.1.1 Calibrating RSGAIN for DS2780 IC

In order to get a precise measure, it is necessary to calibrate the current measurement. The current A/D of the DS2780 is extremely sensitive. It is capable of measuring a voltage drop of only 1.5625 µV across the sense resistor. This kind of accuracy can only be achieved by calibrating the current measurement. The DS2780 is factory calibrated to provide the accuracy specified in the data sheet. However, in our case we need to reprogram the current measurement gain factor (RSGAIN) to improve current-measurement accuracy. The RSGAIN is adjusted to correct for variation in the external sense resistor's nominal value.

The DS2780 measures the current and then multiplies that value by the RSGAIN scaling factor to provide an accurate current measurement, which is reported in the current register and accumulated in the Accumulated Current Register (ACR).

Reported Current (mA) = Measured Current (mA) × RSGAIN

When shipped from the factory, the gain calibration value is stored in two separate locations in the Parameter EEPROM Block: RSGAIN (reprogrammable) and FRSGAIN (read only). RSGAIN determines the gain used in the current measurement. The read-only FRSGAIN is provided to preserve the factory value only and is not used in the current measurement. In the event that an incorrect value is inadvertently written to the RSGAIN register, the FRSGAIN value can be used to recover the original RSGAIN value.

In order to calculate an RSGAIN value, an accurate reference current of 30 mA is forced across the sense resistor. The reference current is divided by the current reported by the device. Before hand, the factory gain is measured. The ratio of reference current to reported current then is multiplied by the existing RSGAIN value to determine the new RSGAIN value.

$$New\ RSGAIN = Factory\ RSGAIN \cdot \frac{Reference\ current}{Reported\ current} = 1.02637 \cdot \frac{30\ mA}{29.53\ mA} = 1.042$$

The RSGAIN calculated value should be written to Addresses 78h and 79h, and then copied into EEPROM. This ensures that the current reported by the DS2780 device matches the reference current that is forced across the sense resistor.

Finally, if we evaluate again the last test of the previous point, the current measurement is 8.749 mA, which is much more approximate to the one obtained with an accurate amperemeter.

At this point, the communication between the DS2780 and the AquisGrain, as

well as the performance of the voltage and current registers, is checked. Now, it seems necessary to analyze the behaviour of the DS2780 during continuous measuring. In order to test it, we send current measurements to the PC during two hours and a half. The results are graphed in the Figure 73.

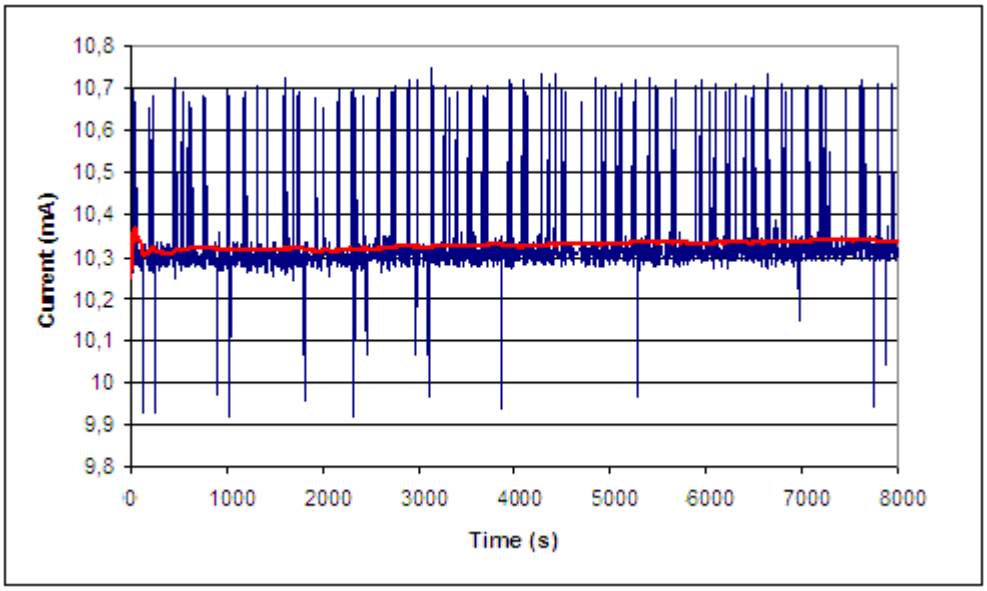


Figure 73. Current Evolution during 2.5 hours

Analyzing previous figure, all the current values lie inside the range of less than 1 mA. This error is mainly due to the timing point when the DS2780 samples the voltage dropped in the sensor resistor. If this sample is taken when the AquisGrain is transmitting, then the measurement will be above the average. In table 15, the main statistical figures are calculated. As expected, the standard deviation is below the DS2780 resolution. In addition, the probability of getting a measurement out of the 95% confidence level is less than 0.5%.

Current Measurements	
Mean	10,3294
Standard Error	0,0024
Median	10,3089
Mode	10,3017
Standard Deviation	0,1024
Sample Variance	0,0105
Kurtosis	8,9315
Range	0,8190
Minimum	9,9282
Maximum	10,7471
Confidence Level(95,0%)	0,0047

Table 13. Current measurements statistics.

Figure 74 shows the histogram of the current measurements. The histogram suggests a Gaussian distribution. In order to check this hypothesis, two statistical tests are calculated. First, the kurtosis which is a measure of whether the data are peaked or flat relative to a normal distribution. For univariate data Y_1, Y_2, ..., Y_N, the formula for kurtosis is:

$$Kurtosis = \frac{\sum_{i=1}^{N}(Y_i - \mu)^4}{(N-1)\cdot\sigma^4}$$

where μ is the mean, σ is the standard deviation, and N is the number of data points. The kurtosis is 8.9 while the kurtosis for a standard normal distribution is three. Therefore, our distribution is not normal. To analyze how different is our distribution from a normal one, the Kolgomorov-Smirnov test is calculated in order to confirm the previous result. The Kolmogorov-Smirnov test for normality is based on the maximum difference between the sample cumulative distribution and the hypothesized cumulative distribution:

$$D = \max_{1\le i\le N}\left(N(Y_i) - \frac{i-1}{N}, \frac{i}{N} - N(Y_i)\right)$$

where N is the normal distribution. If the D statistic is significant, then the hypothesis that the respective distribution is normal should be rejected. After running the test over 1865 samples, we obtain a distance of 0,343. So, in conclusion, the current measurement distribution is not normal but it is quite similar.

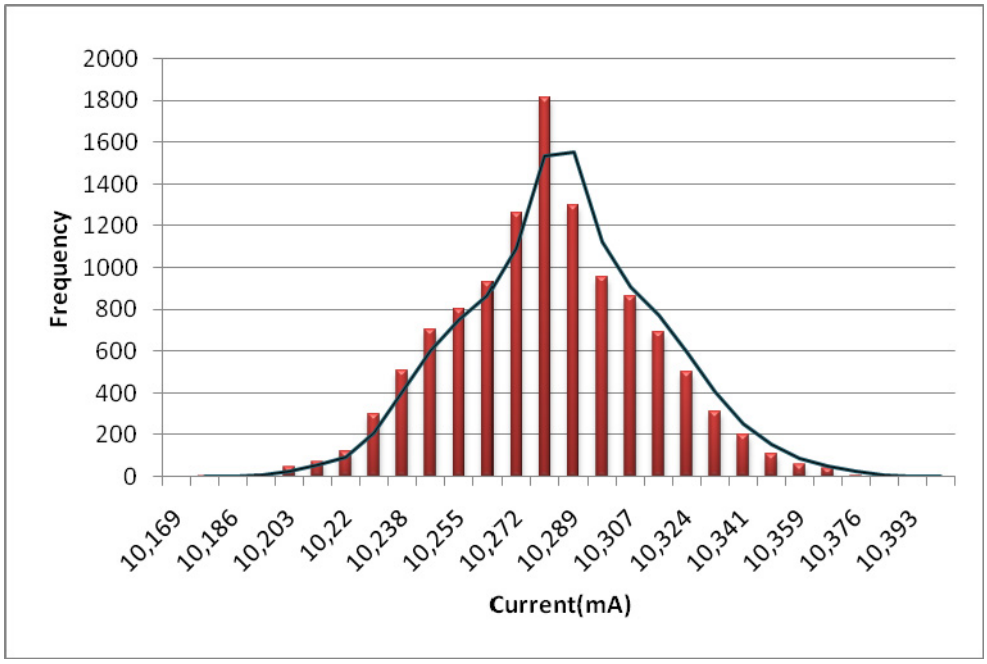


Figure 74. Current Measurements Histogram

6.1.2 Evaluating accuracy fuel gauge

The accuracy of the coulomb counting fuel gauges as the DS2780 depends on numerous variables, many outside the immediate control of the fuel gauge. Fuel gauging systems fall into two categories: integrating system and sam-

pling system. As explained in the previous chapter, the DS2780 belong the second category. The main sourced of error in the DS2780 are similar to those found in an A/D converter. These include *gain*, *linearity*, *inout offset*, and *resolution errors.* The DS2780 datasheet lumps the gain and linearity together and specifies them as a percentage.

In addition, input offset error is specifies as a voltage. Therefore, assuming our typical sense resistor of 220 mΩ is known:

$$I_{ERROR} = \frac{V_{OFFSET}}{R_{SENSE}} = \begin{cases} MIN & \frac{-7.82\mu V}{220m\Omega} = -35.54 \ \ \mu A \\ MAX & \frac{12.5\mu V}{220m\Omega} = 56.8\mu A \end{cases}$$

This error is fixed and does not change with the current sense signal amplitude. Thus, this term contributes more error as measured current becomes small. Increasing R_{SENSE} can reduce the impact of offset error, but the cost of this is increased power dissipation in the application. Our designed PCB includes a sense resistor of 220 mΩ, as it was explained in chapter 4.1.3. This value was chosen according to the needed resolution.

Resolution error becomes an issue when the resolution is as large as the signal level of interest. In order to proof how the sense resistor affects to the current measurements a test is carried out. This test consists in forcing the AquisGrain node to consume more current by inserting resistors between the DS2780 circuit output and the AquisGrain battery input. This way different currents are achieved. Moreover, the sense resistor is also variable among the following values: 220 mΩ, 100 mΩ, 50 mΩ, 20 mΩ and 10 mΩ. After that, the DS2780 current registers are read. The error of the obtained measure is calculated as:

$$ERROR(\%) = \left| \frac{Nominal\ \ Current \cdot Measured\ \ Current}{Nominal\ \ Current} \right| \cdot 100$$

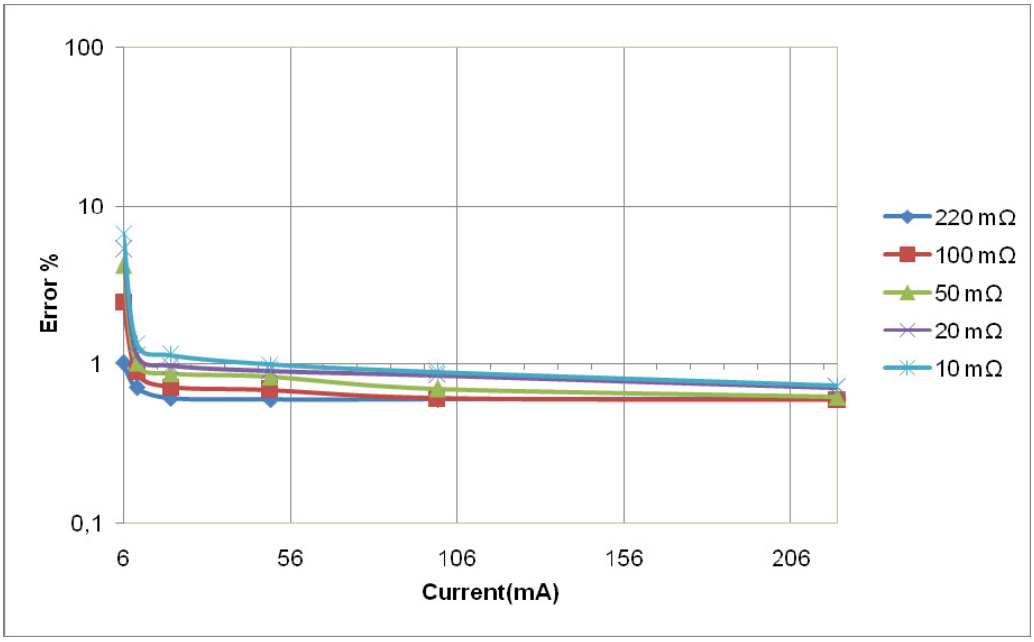


Figure 75. DS270 accuracy as a function of current and R_{SENSE}

As expected, the results of the test agree with the theory (see Figure 75). The measurement error gets greater as the current consumption gets smaller. In this point, bigger resistor has less error. So, as our ECG application is going to consume between 6 mA and 13 mA, the 220 mΩ is the best choice. With this resistor we get a maximum error of 1.03% percent during the test. This error is feasible for our energy management architecture.

6.1.3 Evaluation the on-chip SOC determination algorithms

Once the current measurement is calibrated and the sense resistor provides enough resolution, the DS2780 capacity algorithms are evaluated. In order to carry out this test the following steps are followed:

1. Store the battery specifications of the Lishen SP0405AC in DS270 EEPROM registers (see point 4.1.3.4.1)
2. Connect a new fulfilled Lishen SP0405AC battery to the DS2780 PCB.
3. Initialize the ACR.
4. Send the RRAC and the RAAC values.
5. Wait until no message is received.

The test is carried out with a constant discharge current of 24 mA ± 0.5 mA. The battery runs out after 6 hours and 5 minutes, approximately. Analyzing the evolution of the relative and absolute capacity values, the obtained values are quite accurate with the physical results. When the RRAC and the RAAC reach the zero, the AquisGrain only lasts 4 minutes more. That approximation is quite accurate as the prediction error is only 1.1% of the expected lifetime. This accuracy is reached due to several factors. First, the battery is new, so no aging deterioration has occurred in the battery chemistry yet. Second, the discharge current is quite high compared with the battery nominal capacity. Finally, the test is run under constant temperature (20°C). Results are graphed in Figure 76.

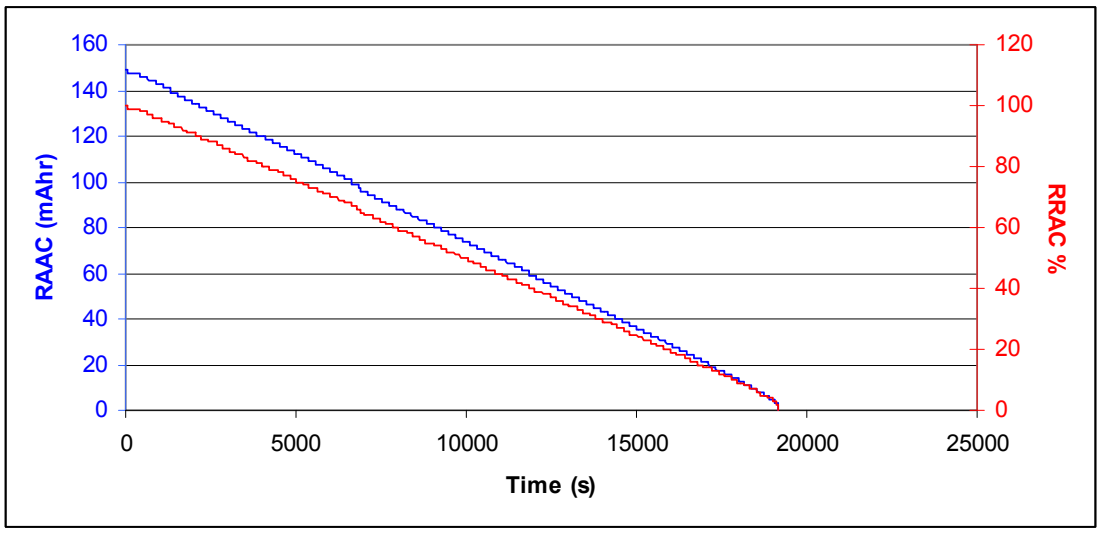


Figure 76. RAAC and RRAC time evolution during a 150 mAhr battery discharge

Once the accuracy of the capacity estimation algorithms is checked under a constant temperature, the next step is to analyze the behaviour of the imple-

mented cell model. Therefore, we test the active capacity performance while temperature is varying. The test steps are:

1. Store the battery specifications of the Lishen SP0405AC in DS270 EEPROM registers (see point 4.1.3.4.1)
2. Connect a fulfilled Lishen SP0405AC battery to the DS2780 PCB.
3. Initialize the ACR.
4. Set desired temperature.
5. Send Full capacity register value.
6. Repeat steps 4 and 5 with different temperatures.

The obtained results are graphed in Figure 77. The DS2780 behaves as expected and the results comply with cell model specifications. So in a range of ten degrees (from 20°C to 30°C) the full capacity changes almost 10 mAhrs. This change is considerable for the test 150 mAhrs battery, but it is negligible in typical batteries which will be used for the ECG application.

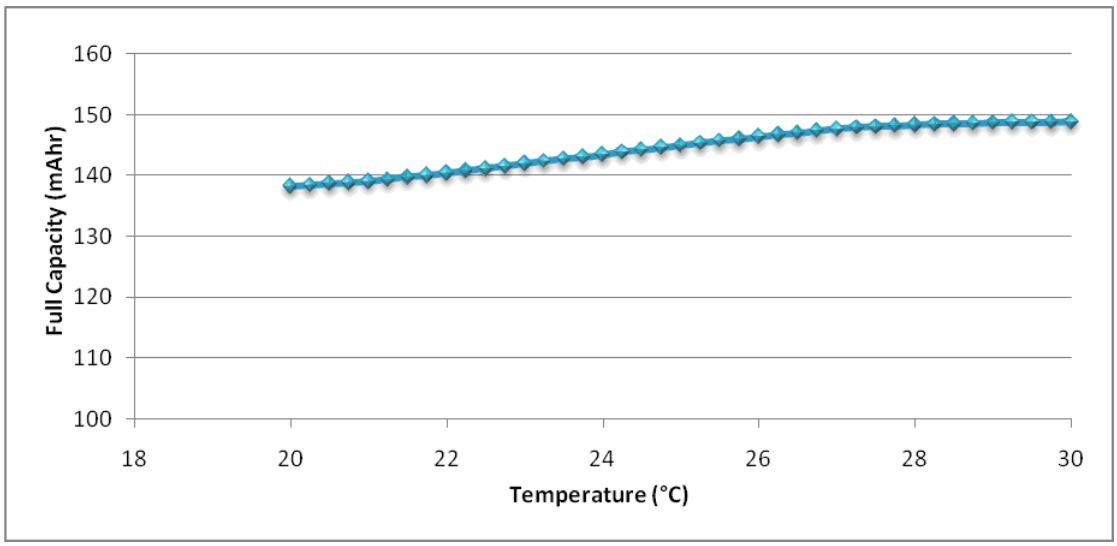


Figure 77. Full battery capacity evolution with variable temperature

6.2 Testing ECG power saver module

After running the previous tests, we can assure that the battery monitor works as expected for the AquisGrain 1.0. The next tests are done in order to ensure that the ECG power saver application can allow different energy states.

6.2.1 Testing ECG sampling rate

Our architecture will allow changing the rate which the ECG sensor is sampling depending on the user policy. Many discussions were devoted to the topic of the adequate sampling frequency for the ECG data. The total error produced by the finite sampling of the ECG contributes an additive white noise component to the spectra of the PR interval, which is inversely proportional to the frequency squared. Due to the shorter duration and inherent limited variability of intrabeat cardiac intervals such as the PR or QT intervals, sampling

frequencies of 250 Hz or higher should be considered. These sampling rates may be implemented by direct sampling of the ECG or through interpolation methods. The Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology recommended the use of 250–500 Hz or higher sampling frequency for HRV measurements without interpolation. Poor-quality records due to a low sampling rate can produce inaccurate measures. On the other hand, unnecessarily fast sampling results in extreme memory usage, processing time and production costs. So, we let user decides which sampling rate is better for his hardware.

The sampling rate choice will affect directly on the AquisGrain consumption. In order to analyze this effect, we run a test changing the sensor sampling rates. The followed steps are:

1. Set the AquisGrain policy to its default state:
    - Radio power transmission= 0 dBm
    - Sensor sampling rate = 500 Hz
    - Heart Rate transmission = Enable
    - ECG samples transmission = Enable
2. Maintaining the power transmission, we change the other parameters to all the possible combinations.
3. Get current measurements from the DS2780.

After receiving the current measurements for the 16 possible states, we conclude that as the sampling rate increases the current consumption also increases. Disabling the heart rate transmission or the ECG transmission decreases the current consumption but not in a proportional way (see Figure 78)

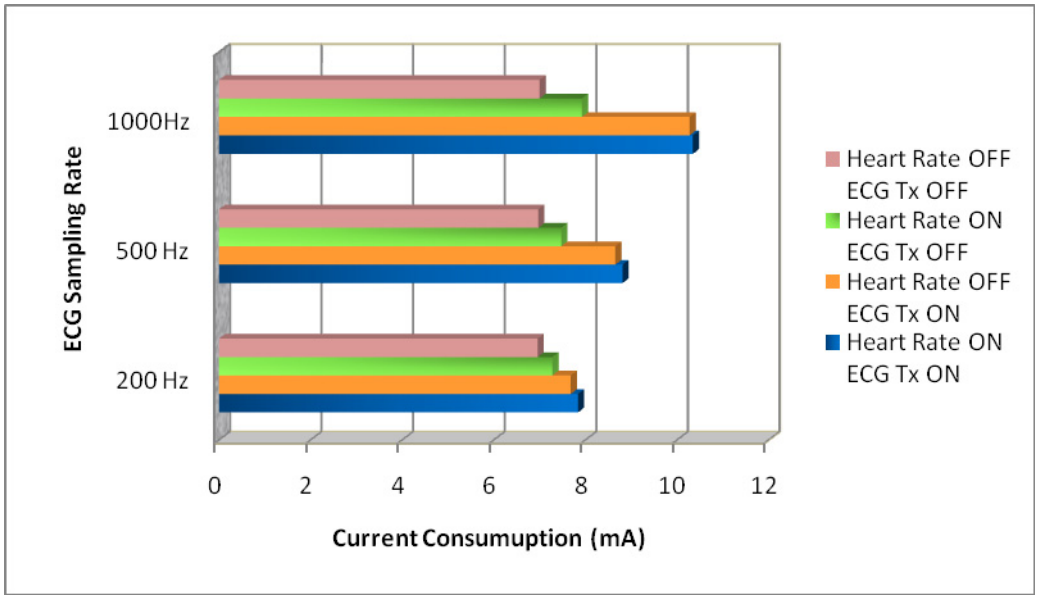


Figure 78. Current consumption evolution for different states

This behaviour can be understood through the analysis of the energy usage. Each AquisGrain component is consuming some current, but this current consumption varies according to the sampling rate in some of these compo-

nents. The Atmega128L is the component which consumes more current. After the microprocessor, it is the CC2420 the second one which consumes more current. Then, the other components (voltage regulator, ID chip, fuel gauge IC …) consume a negligible amount of current.

A second test is carried out with similar specifications that the previous one. Now, only the ECG sampling is variable, the rest of components remain in their default state. In addition, the current consumed by each AquisGrain component is also measured.

After analysing the results, we noticed that the current consumption increment after a sensor sampling rate increase is due to the time increase that the CC2420 is transmitting. As it is shown in Figure 79, with 1000 Hz the current consumed by the radio is more than the double than the one consumed with 200 Hz. The current consumed by the microprocessor also increases but in a smaller amount than the radio does. The current consumed by the rest of AquisGrain components remains almost constant.

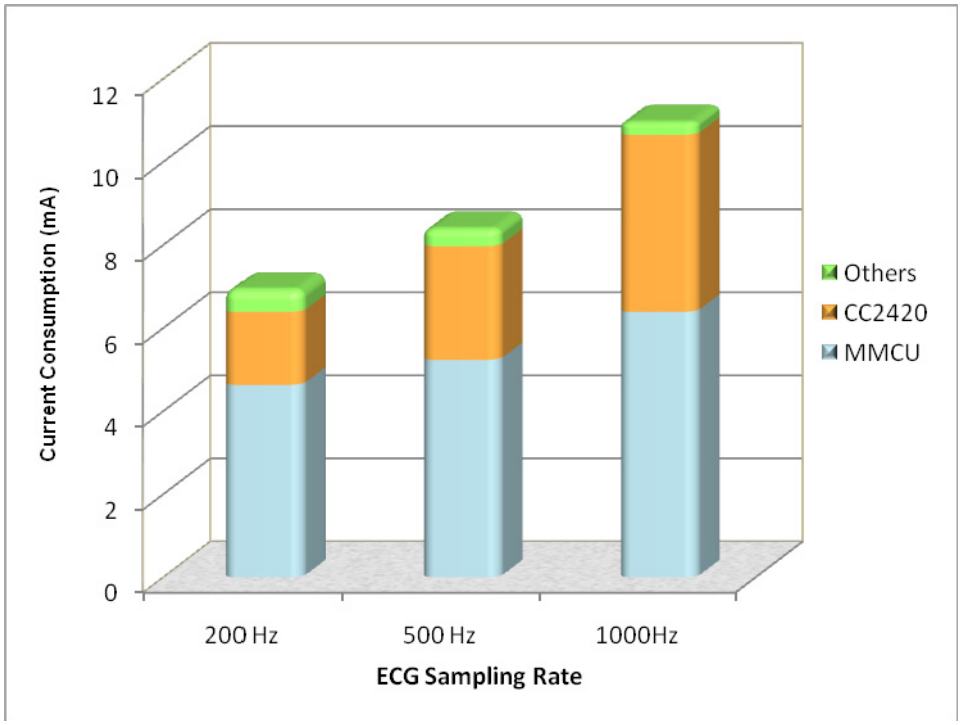


Figure 79. Energy component usage for different sampling rates

6.2.2 Testing CC2420 Power Modes

In order to save energy the policy specification allows the user to change the power transmission mode of the radio. Reducing the transmission power could be also exploited for increasing the reuse of channels, thereby more networks can co-exist. However, the transmission power strongly affects the communication range of IEEE 802.15.4. So, the user must have into account the area he desires to cover.

The test of the CC2420 consists on changing the power transmission mode of the radio and measure the power consumption. One resistor of 1 Ω is inserted between the battery and the AquisGrain node. Then, the voltage drop across

the resistor is measured with a calibrated oscilloscope.

Power Transmission (dBm)	Theoretical Current Consumption (mA)	Measured Current Consumption (mA)
0	17.4	16.33
-1	16.5	15.09
-3	15.2	14.11
-5	13.9	13.89
-7	12.5	11.88
-10	11.2	10.78
-15	9.9	9.73
-25	8.5	8.48

Table 14. Current Consumption of the CC2420

As it is shown in Table 14, if the power transmission decreases, the current consumption decreases too. The current measurements are always lower than the theoretical consumption. In this way, the CC2420 consumes less than what we expected from the specifications. In Figure 80, a snapshot from the oscilloscope monitoring a resistor of 1 Ω shows the current consumption (in this case voltage≈current) during the ECG transmission. Analyzing the figure, we see that the sampling rate is 500 Hz as the ECG stream period is 78 ms. The maximum power is 22.4 mA which mainly belongs to the radio and the microprocessor.

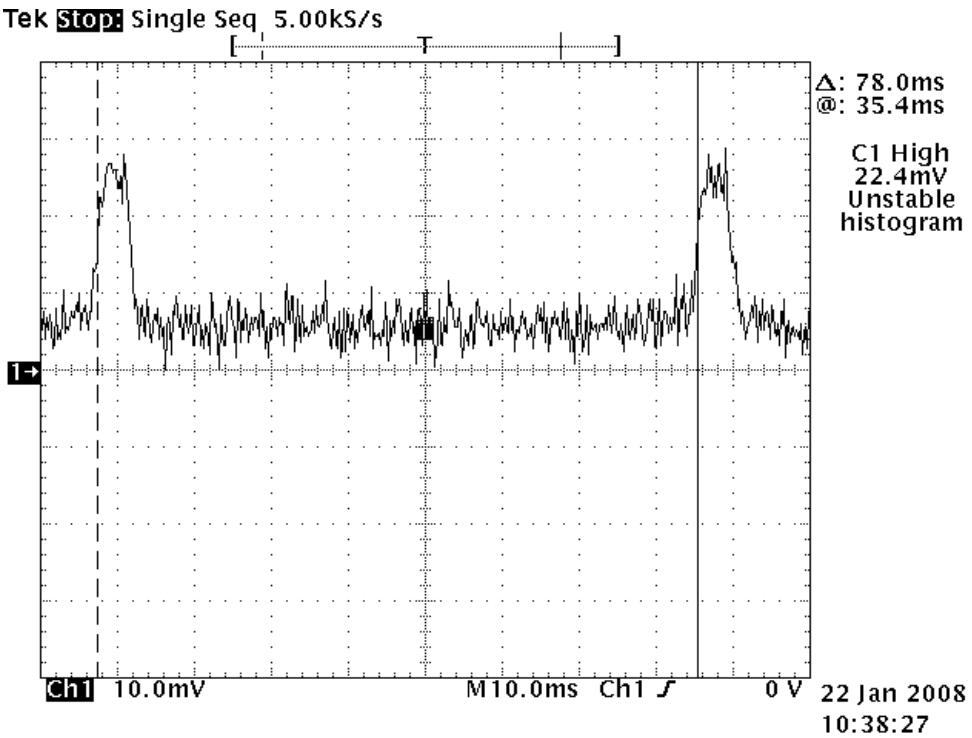


Figure 80. ECG stream 0 dBm transmission

Therefore, the user can decrease more than half of the current consumption

by decreasing the power transmission mode. However, it should be noted that the lower the transmission power is, the more vulnerable the radio link gets to position changes. So, the user has to be aware of this drawback.

During the reception, the CC2420 also works as expected, as it shown in Figure 81. In this oscilloscope waveform, the transmission of an energy message and the reception of a policy message are shown. Therefore, the pulse width is about 8.14 ms which belongs to the transmission and the reception, as the radio is turned on until the coordinator sends an acknowledgement after the node poll.

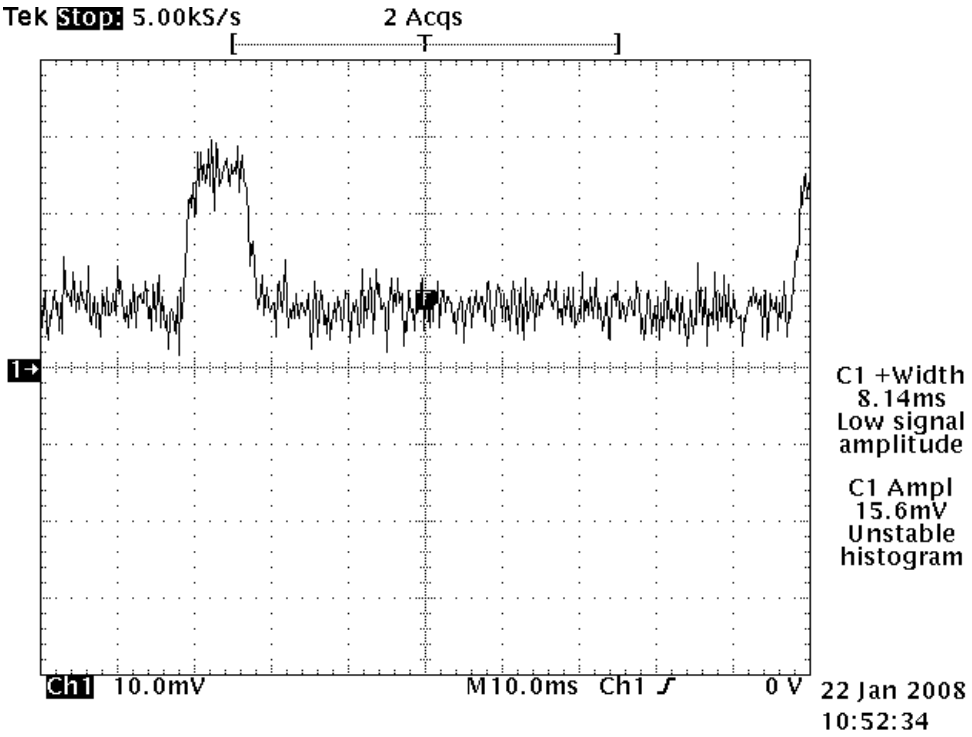


Figure 81. Energy message transmission and policy message reception

6.3 Testing the Policy Specification

In order to evaluate the impact of the policy specification in the lifetime of the battery, different policies are applied to the ECG application. As explained during the implementation chapter, the policy can change different values which have certain influence in the current consumption.

The right performance of the policy specification module is checked by sending several different policies to the energy manager. Once the policy is set, the application shows which the present status of the node is. For instance, we change the radio power transmission or the ECG sampling rate, and the new policy state is shown. Therefore, once the policy has changed, the current consumption also changes. However, the critical point of the policy specification is the lifetime. When the user sets a specific lifetime the energy manager has to compute the less energy constraint path and decide which elements are going to be change to reach this lifetime. Two different polices are evaluate to exemplify the architecture performance.

The first policy consists of three rules with two different priority levels. In the first policy level, there are two rules linked by a logical operand OR. One rule set the lifetime as maximum and the other one sets the sensor sampling rate to 1000 Hz. In the second level, the unique rule sets the power transmission to -10 dBm. The levels are related by a logical operand AND. In this case the previous policy state does not matter; as the lifetime clause does not set a specific time only maximizes it. One the policy is applied it is processed by the energy manager module. The policy is divided in different energy paths according to the logical operands and the priorities:

- Maximize Lifetime and change power transmission to -10 dBm.
- Maximize Lifetime.
- Sensor sampling rate of 1000 Hz and power transmission to -10 dBm.
- Sensor sampling rate of 1000 Hz.

From all these possible paths, the energy manager chooses the one which is less energy constraining. In the first level, maximizing the lifetime consumes less energy than setting the sampling rate to 1000 Hz. So, the sampling rate rule is rejected. As the second level is linked by a logical operand AND to the first one, and it is compatible with it, then the rule is also taken into account. Therefore, the final policy tries to maximize the lifetime by setting the sensor sampling rate to 200 Hz, disabling the heart rate transmission and also disabling the ECG stream transmission, but the power transmission is set to -10 dBm.

The second policy only has one priority level composed of three rules. The first rule set the lifetime to more than 14 hours. This rule is linked to the second one by a logical operator AND. The second rule sets the ECG transmission enable and it is linked by a logical operator OR to the third rule. This last rule set the sensor sampling rate to 1000 Hz. This policy is split in two different paths:

- Lifetime>14 Hours AND ECG transmission enable
- Sensor Sampling rate = 1000 Hz

As in the previous policy, the energy manager selects the path which consumes less current. In this case, both policies could consume the same amount of current, but setting the lifetime has more degrees of freedom. In contrast to the first policy, now the previous AquisGrain policy state is essential as it determines if the desired lifetime could be achieved. So, before selecting which path consumes less power, the lifetime has to be processed.

The energy manager calculates the minimum current consumption which is necessary to reach the lifetime. In this test the previous state is the default one (Sensor Sampling Rate=500 Hz, Power Transmission = 0 dBm, Heart Rate = Enable, ECG Transmission = Enable). This state consumes approximately 8.8 mA. The test battery is a new battery of 150 mA. With this battery in this state, the lifetime is the 17 hours. So, the energy manager determines that for applying this new policy is not necessary to change the AquisGrain status.

With these two tests the right performance of the policy specification and

some features of the energy manager are tested. However, it is not demonstrated that the lifetime policies achieved the user restrictions. For testing the accuracy of the lifetime predictions different lifetimes are set with the two test batteries. Then, we wait until the battery is completely discharged and we compare the real lifetime with the expected one.

The results of this test are shown in Table 17. The lifetime obtained and the one predicted are quite similar. The maximum error obtained is 14 minutes. It should be noted that the expected lifetime and the measured lifetime is always higher than the one expected. So, the prediction is always pessimistic which allows the user to have some extra minutes to replace the battery.

Battery Model	Discharge Current	Expected Lifetime	Measured Lifetime
Lishen SP0405AC (140 mAhrs)	6.98 mA	19 h 47 min	19 h 55 min
	7.81 mA	17 h 33 min	17h 37 min
	7.90 mA	17 h 10 min	17 h 24 min
	8.83 mA	15h 15 min	15 h 24 min
	10.25 mA	13 h 20 min	13 h 25 min
	10.45 mA	13 h 5 min	13 h 10 min
Varta 653450UC (1130 mAhrs)	8.74 mA	5 days 9 hours 30 min	5 days 9 hours 43 min
	10.39 mA	4 days 12 hours 48 min	4 days 12 hours 59 min

Table 15. Lifetime Predictions measurements

This accuracy has been reached carrying out the test under constant environment conditions and always charging the battery performing a *Learn Cycle* in order to fight the aging process. However, a problem appears during the fully discharge of the battery. Due to the voltage regulator in the AquisGrain, when the battery voltage level is below the 3.5 V, the current consumption increases considerably (see Figure 82).

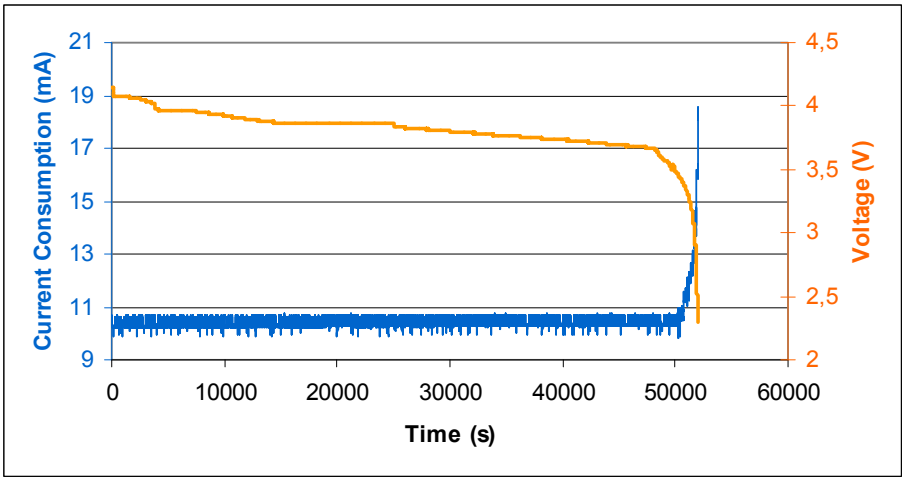


Figure 82. Fully discharge battery curves

Therefore, during the last minutes the prediction becomes totally inaccurate. This fact is solved by communicating to the user that the battery has run out of energy. So, the estimation is pessimistic and always would have a fixed error of a few minutes which depend on the discharging rate.

The final stage to ensure the architecture performance consists of evaluating it under variable environments. The steps followed to carry out the test are:

1. Set up the default environment: Connect the AquisGrain to a fulfilled battery.
2. Run the policy application in the PC and set a lifetime greater than 15 hours. This policy can be achieved with the 140 mAhrs battery and the default ECG application state which consume 8.8 mA approximately.
3. Wait some time and then force a current increase in the node by inserting a resistor.
4. Wait for the energy manager to change the policy automatically to reach the desired lifetime.

After running the test, we conclude that the energy manager adapts to the environment changes successfully. After we force a current increase, the energy manager processes again the policy and determines that it is necessary to decrease the sensor sampling rate. As it is shown in Figure 83, the energy manager last around two minutes while determining that there has occurred a current change, then it processes the rule again and decides that is necessary to change the state to achieve the desired lifetime.

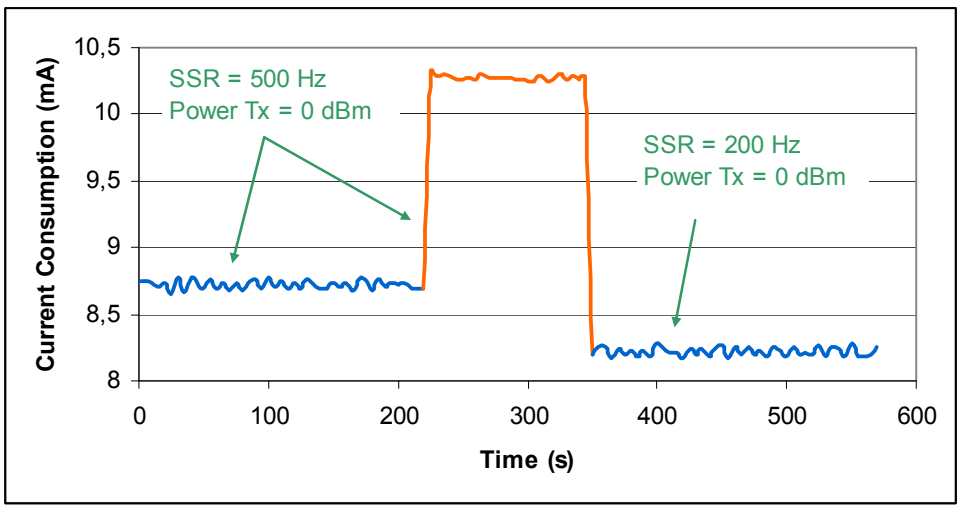


Figure 83. Environment change test

7 Conclusions

Energy management is a classic resource management problem. Towards the vision of energy efficient consumption in wireless sensor networks, we have designed an energy management architecture based on many archetypical concepts. These include classifying energy as a first-class resource, prioritizing resource request, accounting for fine-grained energy consumption, allocating resources based on dynamic constraints and providing quality-of-service guarantees by using feedback.

Our proposed architecture implements services for energy monitoring and management. An integral part of the architecture is the monitoring component. While software emulation provides an approximation, integrated circuits will provide much more detailed and accurate energy profiles. By means of a fuel gauge integrated circuit, it is possible to monitor the capacity evolution of the battery. We have showed that the DS2780 fuel gauge is able to meet or exceed challenging requirements unique to wireless sensor networks. This IC employs a simple architecture which uses an ADC current/voltage/temperature converter to provide a large dynamic range, a coulomb counter to provide capacity estimations. These features are useful for profiling a range of systems with different types of batteries and power consumptions.

In addition, we have designed a user policy specification which cooperates with the energy manager. The policy is simple to compose for non-programming-oriented users, and clearly expresses different levels of priority. It has been implemented for the particular case of an ECG application over Philips AquisGrain 1.0 platform. Therefore, the user must be literate of the ECG performance in order to specify the rules. The specification component empowers the user with the ability to directly express energy-related policies such as lifetime and sensing rates. It has been demonstrated that the policy allows user to reduce more than 50% the power consumption. Nevertheless, this power consumption reduction entails some drawbacks. The user has to be aware of the policy consequences, in order to manage it. Through the policy is also possible to reduce the transmission power for increasing the reuse of channels and thereby more networks can co-exist.

Due to the need for an energy-centric architecture in wireless sensor networks, this architecture is designed to reach this goal. However, this vision is by no means conclusive since there are many different approaches to managing energy. The designed energy manager is the heart of the energy information. With the received energy information from the network and the user policy specifications, the manager decides how the energy has to be used.

The need for adaptive energy management extends beyond communication protocols. The implemented energy manager responds effectively and dynamically to the changing conditions. To achieve this level of adaptation, a collaborative relationship between the coordinator and the node is established. Thus, the monitor application is able to gather real-time information about node resources and the environment. Then, the energy manager selects proper tradeoffs between energy consumption and other policy require-

ments, and finally modifies the nodes behaviour dynamically to conserve energy. Since some adaptations cannot be anticipated at design time, the policy enables the user to introduce/remove functionalities which are not crucial.

To prove the efficacy of the battery monitor and the fuel gauge, we have shown an implementation that shows responsiveness to current changes and runtime satisfaction of user requirements. With a fixed delay, the energy manager is able to process the current consumption changes, process the user policy and establish a new set up in the node.

On the other hand, to evaluate the lifetime prediction, several different lifetime policies have been applied to the network node, achieving an error in lifetime estimations below the 1.4 %. This error is due to the resolution of the energy monitoring and because of the non-linear behaviour of the AquisGrain 1.0 at the end of the battery life. The error could be reduced by increasing the resolution of the DS2780, but the current consumption will increase subsequently.

When analyzing the cost of implementing our architecture on the existing IEEE 802.15.4-based platform AquisGrain 1.0, only a power saver module to control the application working on the node is necessary. This module is in charge of switching among power modes the radio and the microcontroller in order to reduce the energy consumption. It also controls the exchange of the messages related to the architecture between nodes and coordinator. In fact, the changes implied that only some extra processing from the microcontroller is required in order to control the messages exchange. In summary, the approach followed makes our solution quite efficient, accurate and easy to adapt to other applications.

8 Future Work

Our energy management architecture proposal has been implemented and evaluated at the node level. In wireless sensor networks the energy consumption can be optimized not only at node-level but also at network-level. A significant amount of energy can be preserved by carefully determining the data that should be stored in designated view nodes as well as coordinating the data dissemination to these nodes.

Therefore, future research lines can improve the energy consumption by managing the routing and topology of the network. So, the goal should be to achieve a uniform energy distribution among the sensor nodes. It should be noted that sensor nodes around a sink become bottlenecks. Then, it will be needed energy-aware routing schemes for static sinks and mobile sinks. In this way, our architecture could integrate different routing schemes that could be set up by the user policy.

In addition, in some scenarios it could be useful to improve the network lifetime by taking advantage of mobile nodes. Using these nodes as relays it would be possible to decrease the power consumption of static nodes which are further from the coordinator. Then, the user could set through the policy which nodes are mobile can perform as a relay.

Furthermore, data aggregation can be used when data coming from different sources routes is similar. So, if several nodes have the same policy, we can eliminate redundancy and thus saving energy. Finding routes from multiple sources to a single destination that allows in-network consolidation of redundant data could improve the network lifetime.

It is clear that any piece of work leaves many open issues and lines of future research. It is even more so in a Thesis such as this where an architecture of very general applicability is presented. Nevertheless it is important to ask oneself whether the ground that has been built is solid enough so as to continue constructing from it. In this sense we believe that we have accomplished our goals and we hope to have contributed to the advance of not only present but also future research in our field.

Appendix A – Existing Fuel Gauges

A.1 Maxim Dallas Semiconductors Fuel Gauges

Dallas Semiconductor							
Part Number	*Battery Type*	*User Data Storage (bytes)*	*Parameters Measured*	*Supply Voltage (min) (V)*	*Supply Voltage (max) (V)*	*Features*	*Price (US$)*
DS2751	3-Cell NiMH, 1 Cell Li-Ion	16 SRAM, 32 EEPROM	Current, Temperature, Voltage	2,5	5,5	Digital Current Accumulation, Unique ID	1.99
DS2756	3-Cell NiMH, 1 Cell Li-Ion	96 EEPROM, 8 SRAM	Current, Temperature, Voltage	3	5,5	Suspend Mode, One-Shot Mode, Digital Current Accumulation, Unique ID	1.90
DS2780	1/2 Cell Li-Ion	40 EEPROM	Current, Temperature, Voltage	2,5	5,5	On-chip Remaining Capacity Algorithms, Digital Current Accumulation, Unique ID	2.70
DS2781	1/2 Cell Li-Ion	40 EEPROM	Current, Temperature, Voltage	2,5	10	On-chip Remaining Capacity Algorithms, Digital Current Accumulation, Unique ID	2.90

A.2 Texas Instruments Fuel Gauges

Texas Instruments								
Part Number	*NiCd*	*NiMH*	*Lead Acid*	*Li-Ion*	*Communication Interface*	*Li-Polymer*	*Price (US$)*	*Description*
BQ2018	Yes	Yes	Yes	Yes	HDQ	Yes	1.90	Multi-Chemistry Charge/Discharge Counter W/ High-Speed 1-Wire I/F (HDQ)
BQ2019	Yes	Yes	Yes	Yes	HDQ	Yes	1.95	FLASH-Based Precision Multi-Chemistry Charge/Discharge Counter W/High-Speed 1-Wire I/F (HDQ)
BQ2023	Yes	Yes	Yes	Yes	SDQ	Yes	2.00	Single-Wire Advanced Battery Monitor IC for Cellular and PDA Applications
BQ26200	Yes	Yes	Yes	Yes	HDQ	Yes	2.00	FLASH-Based Precision Multi-Chemistry Charge/Discharge Counter W/1-Wire I/F (HDQ)
BQ26220	Yes	Yes	Yes	Yes	HDQ	Yes	2.05	FLASH-Based Precision Multi-Chemistry Charge/Discharge Counter with Voltage Measurement
BQ26221	Yes	Yes	Yes	Yes	HDQ	Yes	2.05	FLASH-Based Precision Multi-Chemistry Charge/Discharge Counter w/Voltage Measurement
BQ26231	Yes	Yes	Yes	Yes	HDQ	Yes	1.50	Cost-Efficient Coulomb Counter for Battery Capacity Monitoring In Embedded Portable Applications

Appendix B – Policy State Current Measurement

B.1 Table with current consumption in each policy states

Power Transmission (dBm)	Sensor Sampling Rate (Hz)	Heart Rate Transmission	ECG Stream Transmission	Current Consumption (mA)
0	200	False	False	7,1
0	200	False	True	7,25
0	200	True	False	7,7
0	200	True	True	7,9
-5	200	False	False	7,02
-5	200	False	True	7,24
-5	200	True	False	7,64
-5	200	True	True	7,68
-10	200	False	False	7,02
-10	200	False	True	7,23
-10	200	True	False	7,6
-10	200	True	True	7,63
-15	200	False	False	7,02
-15	200	False	True	7,22
-15	200	True	False	7,58
-15	200	True	True	7,61
-25	200	False	False	7,02
-25	200	False	True	7,22
-25	200	True	False	7,58
-25	200	True	True	7,61
0	500	False	False	7,03
0	500	False	True	7,51
0	500	True	False	8,75
0	500	True	True	8,91
-5	500	False	False	7,03
-5	500	False	True	7,5
-5	500	True	False	8,54
-5	500	True	True	8,6
-10	500	False	False	7,03
-10	500	False	True	7,5
-10	500	True	False	8,43
-10	500	True	True	8,48
-15	500	False	False	7,03
-15	500	False	True	7,49
-15	500	True	False	8,37
-15	500	True	True	8,4
-25	500	False	False	7,03
-25	500	False	True	7,49

Power Transmission (dBm)	Sensor Sampling Rate (Hz)	Heart Rate Transmission	ECG Stream Transmission	Current Consumption (mA)
-25	500	True	False	8,33
-25	500	True	True	8,36
0	1000	False	False	7,04
0	1000	False	True	7,97
0	1000	True	False	10,02
0	1000	True	True	10,88
-5	1000	False	False	7,04
-5	1000	False	True	7,95
-5	1000	True	False	10,02
-5	1000	True	True	10,01
-10	1000	False	False	7,04
-10	1000	False	True	7,94
-10	1000	True	False	9,78
-10	1000	True	True	9,81
-15	1000	False	False	7,04
-15	1000	False	True	7,93
-15	1000	True	False	9,68
-15	1000	True	True	9,71
-25	1000	False	False	7,04
-25	1000	False	True	7,93
-25	1000	True	False	9,58
-25	1000	True	True	9,61

References

[1] IEEE Computer Society: *IEEE Std 802.15.4-2003; Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),* February 2003.

[2] Jorba, J.: *A New Wireless Node for Patient Monitoring*. Philips Research Aachen, May 2004.

[3] Gunther, S.H.:*Managing the Impact of Increasing Microprocessor Power Consumption*, Intel Technology 2001

[4] Maxim Dallas Semiconductors: *DS2780 Standalone Fuel Gauge IC,* May 2007.

[5] Atmel Corporation: *ATMEL ATmega 128L AVR core microcontroller specifications*, February 2003.

[6] Chipcon AS: *CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver specifications*, June 2006.

[7] Holger, K.; Willig, A.: *Protocols and Architectures for Wireless Sensor Networks.* Wiley, 2005.

[8] Landsiedel, O.; Wehrle, K.; Gotz, S.: *Accurate Prediction of Power Consumption in Sensor Networks*, 2nd IEEE Workshop on Embedded Networked Sensors, 2005.

[9] Titzer, B. L.; Lee, D. K.; Palsberg, J.: Avrora: *Scalable Sensor Network Simulation with Precise Timing*. In Proc. Of IPSN, 2005.

[10] Levis, P.: TOSSIM: *Accurate and Scalable Simulation of Entire TinyOS Applications*. In Proc. of SenSys, 2003.

[11] Polley, J.: *ATEMU: A Fine-Grained Sensor Network Simulator*. In Proc. of SeCon. 2004.

[12] Sun, L.; Sun, Y.; Shu, J.; He, Q.: MotePlat: *A Monitoring and Control Platform for Wireless Sensor Networks*. In Proc. Of IPSN, 2007.

[13] Werner-Allen, G.; Swieskowski, P., Welsh, M.: *MoteLab: A Wireless Sensor Network Testbed,* ISPN, 2005.

[14] Jiang, X.; Dutta, P.; Culler, D.; Stoica, I.: *Micro Power Meter for Energy Monitoring of Wireless Sensor Networks at Scale*. ACM IPSN-SPOT, 2007

[15] Crompton, T. R: *Battery reference book*, 3rd ed. Oxford, England ; Boston : Newnes 2000.

[16] Buchmann, I.: *Batteries in a Portable World: A Handbook on Rechargeable Batteries for Non-Engineers*, Cadex Electronics, 2001.

[17] Barsukov, Y: *Battery Selection, Safety, and Monitoring in Mobile Applications,* Texas, 2005

[18] Doyle, M.; Fuller, T.F.; Newman, J.: *Modeling of Galvanostatic Charge and Discharge of the Lithium/ Polymer/Insertion Cell*, J. Electrochemical Soc, 1993.

[19] Pedram, M.; Wu, Q.: *Design Considerations for Battery-Powered Electronics*, Proc. 36th ACM/ IEEE Design Automation Conf., ACM Press, 1999

[20] Benini, L.: *Discrete-Time Battery Models for System-Level Low-Power Design,* IEEE Trans. VLSI Systems, 2001.

[21] Chiasserini, C.F.; Rao, R.R.: *Energy Efficient Battery Management*, IEEE J. Selected Areas in Comm., 2001.

[22] Rakhmatov, D.N.; Vrudhula, S.B.K.: *An Analytical High-Level Battery Model for Use in Energy Management of Portable Electronic Systems*, Proc. IEEE Press, 2001.

[23] Jiang, X.; Taneja, J.; Ortiz, J.; Tavakoli, A.; Dutta, P.; Jeong, J.; Culler, D.; Levis, P.; Shenker, S.: *An Architecture for Energy Management in Wireless Sensor Networks*. WSNA, 2007.

[24] Cahill, W.; Render, M.: *Dynamic simulation modeling of ICU bed availability*, Simulation Conference Proceedings, 1999.

[25] Gilbert, J.; Dewey R.: *Linear Hall-Effect Sensors*, Application Note 27702A, Allegro MicroSystems Inc., Massachusetts, USA, 1998.

[26] Dickinson, R.; Milano S.: *Isolated Open Loop Current Sensing Using Hall EffectTechnology in an Optimized Magnetic Circuit*, Allegro MicroSystems Inc., Concord NH, USA, 2002.

[27] Maxim Dallas Semiconductors: *DS2436 Battery ID/Monitor Chip specifications,* July 2007.

[28] Maxim Dallas Semiconductors: *Overview of 1-Wire Technology and Its Use*, March 2003.

[29] ZedGraph Project [Online]. Available WWW: http://zedgraph.org

List of Acronyms and Abbreviations

AB	Accumulated Bias
AC	Alternating Current
ACR	Accumulated Current Register
ADC	Analog to Digital Converter
AEON	Accurate Prediction of Power Consumption in Sensor Networks
ALU	Arithmetic Logic Unit
ANSI	American National Standards Institute
API	Application Programming Interface
ATEMU	Atmel Emulator
AVR	Advanced Virtual RISC
BAN	Body Area Network
CCA	Clear Channel Assesment
CISC	Complex Instruction Set Computer
CRC	Cyclic redundancy check
CT	Current transformer
DC	Direct Current
DOD	Depth of Discharge
ECG	Electrocardiogram
EEPROM	Electrically-Erasable Programmable Read-Only Memory
ESD	Electrostatic Discharge
FET	Field Effect Transistor
FIFO	First-In First-Out
GPIO	General Purpose Input/Output
IAE	Intensity Active Empty
IC	Integrated Circuit
ICU	Intensive Care Unit
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IF	Intermediate Frequency
IRQ	Interruption request

JTAG	Join Test Action Group
LOS	Length of Stay
LR-WPAN	Low Rate Wireless Personal Area Network
LSB	Less Significant Byte
MAC	Medium Access Control
MIL-SPEC	Military Specifications
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
MSB	More Significant Byte
OS	Operating System
PAN	Personal Area Network
PCB	Printed Circuit Board
PDA	Portable Digital Assistant
PFL	Philips Forschungslaboratorien
PHY	Physical layer
PIO	Programable I/O Pin
PSPICE	Personal computers Simulation Program with Integrated Circuit Emphasis)
RF	Radio Frequency
ROM	Read-Only Memory
RSNSP	Sense Resistor Primer Register
RSSI	Received Signal Strength Indicator
RARC	Remaining Active Relative Capacity
RAAC	Remaining Active Absolute Capacity
SOC	State of Charge
SPI	Serial Port Interface
SQL	Structured Query Language
SRAM	Shadow Random Access Memory
TinyOs	Tiny Operating System
TOSSIM	Tiny Operating System Simulator
TSSOP	Thin-Shrink Small Outline Package
UART	Universal Asynchronous Receiver/Transmitter
VAE	Voltage Active Empty
VHDL	VHSIC Hardware Description Language
WSN	Wireless Sensor Network
XML	Extensible Markup Language

List of Figures

List of Tables

