

Raphael Tucunduva Gonçalves

***Monitoramento da Capacidade de Baterias em
Sistemas Embarcados***

Florianópolis - SC, Brasil

21 de maio de 2007

Raphael Tucunduva Gonçalves

***Monitoramento da Capacidade de Baterias em
Sistemas Embarcados***

Trabalho apresentado como requisito parcial para a obtenção do grau de Bacharel em Ciências da Computação, no semestre 2007.1 do curso de Ciências da Computação da Universidade Federal de Santa Catarina.

DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA - INE
CENTRO TECNOLÓGICO
UNIVERSIDADE FEDERAL DE SANTA CATARINA

Florianópolis - SC, Brasil

21 de maio de 2007

Trabalho de Conclusão de Curso sob o título “*Monitoramento da Capacidade de Baterias em Sistemas Embarcados*”, a ser defendida por Raphael Tucunduva Gonçalves, visada por:

Prof. Dr. Antônio Augusto Medeiros Frölich
Orientador

Msc. Arliones Stever Hoeler Júnior
Co-orientador

Bel. Geovani Ricardo Wiedenhof
Avaliador

Bel. Hugo Marcondes
Avaliador

*“Quem quiser chegar
a ser o que não é,
deverá principiar
por não ser o que é.”*

Carlos Bernardo González Pecotche

Resumo

A necessidade de se gerenciar a energia consumida por processadores é cada vez maior, pois cresce a cada dia a demanda por processamento em sistemas que comumente operam alimentados unicamente por baterias. Em contrapartida, muitos mecanismos de gerenciamento de energia exigem do programador de uma aplicação que, além de se preocupar com as funcionalidades da mesma, se preocupe também com controle da energia que ela consome, entre outras tarefas de cunho operacional. Criando-se um mecanismo no sistema operacional que permita monitoramento do consumo de energia geral do sistema em tempo de execução, o dito S.O. passa a possuir a capacidade de tratar a energia como mais um recurso gerenciável. Com isso, permite-se a implementação de escalonadores e políticas de qualidade de serviço que levem em consideração o consumo de energia do sistema, com total independência da aplicação que será executada, porém ainda permitindo que as políticas de utilização desse recurso sejam configuradas para atender às necessidades da aplicação em questão. Este trabalho apresenta a implementação de um mecanismo de monitoramento do consumo de energia em um sistema operacional para sistemas embarcados com a característica de ser orientado à aplicação que se utiliza para tal do monitoramento da capacidade das baterias que alimentam o sistema e do uso de seus componentes de *hardware*.

Palavras-chave: Sistemas Embarcados, Gerência de Energia, Sistemas Operacionais Embarcados, Projeto de Sistemas Orientado à Aplicação, Predição de Consumo de Energia.

Sumário

Lista de Figuras

Lista de Tabelas

1	Introdução	p. 9
2	Trabalhos Relacionados	p. 11
3	Sistemas Embarcados	p. 12
3.1	Definição	p. 12
3.2	Sistemas Operacionais para Aplicações Embarcadas	p. 16
3.2.1	Um Sistema Operacional Orientado à Aplicação	p. 17
3.2.2	Programação Orientada a Aspectos	p. 18
3.2.3	O EPOS	p. 19
4	Gerência de energia	p. 21
4.1	Definição	p. 21
4.2	Técnicas Mais utilizadas	p. 22
4.2.1	Dynamic Voltage Scaling (DVS)	p. 22
4.2.2	Técnicas utilizadas na Compilação	p. 22
4.2.3	Hibernação de Recursos	p. 23
4.3	Baterias	p. 24
5	Monitorando o consumo de energia	p. 29
5.1	Desafios do Projeto	p. 29

5.2	Implementação	p. 32
5.2.1	Representação numérica da capacidade da bateria	p. 33
5.2.2	Configurações Gerais Necessárias	p. 35
5.2.3	Atribuição do Consumo a Componentes	p. 36
5.2.4	Atualização da Capacidade da Bateria	p. 42
5.2.5	Desempenho do monitor	p. 43
6	Conclusão	p. 45
	Referências Bibliográficas	p. 47

Lista de Figuras

3.1	Destino dos microprocessadores produzidos até 2000	p. 12
3.2	Natureza de aplicação dos processadores fabricados em 2000	p. 13
3.3	Divisão dos processadores pelo número de bits de sua arquitetura	p. 17
3.4	Esquema da interligação de aspectos	p. 19
4.1	Disparidade entre a evolução de sistemas computacionais e baterias	p. 24
4.2	Comparação de densidades energéticas (Wh/l) de diferentes tipos de baterias .	p. 25
4.3	Energia teórica e energia real das baterias	p. 26
4.4	Consumo de uma bateria alcalina em diferentes casos de uso.	p. 27
5.1	Esquemático Mica2 MPR400/MPR410/MPR420	p. 32
5.2	CrossBow Mica2 MPR400/MPR410/MPR420 e o adaptador para bateria AA	p. 33
5.3	Curva de descarga Tensão (em milivolts) x Tempo(em milissegundos) do ex- perimento realizado	p. 34
5.4	Aproximação de um intervalo: o sistema permanece mais em algumas ten- sões que em outras.	p. 35
5.5	Captura de um estado do procedimento de medição no osciloscópio	p. 44

Lista de Tabelas

5.1	Curva subdividida em intervalos de tempo iguais	p. 34
5.2	Consumo de corrente da CPU para seus diferentes estados de economia. . . .	p. 39
5.3	Consumo de corrente do ADC para seus diferentes estados de economia. . . .	p. 39
5.4	Consumo de corrente da USART para seus diferentes estados de economia. . .	p. 39
5.5	Valores obtidos nos testes.	p. 44

1 *Introdução*

A cada dia é maior o número de dispositivos eletrônicos que necessitam de algum tipo de processamento. A maioria desses dispositivos, contudo, não utilizam o processamento de microcomputadores, que muitas vezes são utilizados como sinônimo de microprocessadores, por serem a forma mais conhecida e consagrada pelo uso em que estes últimos são aplicados. Tais dispositivos, conhecidos como sistemas embarcados costumam possuir uma necessidade de processamento muito menor do que a provida por processadores de computadores de propósito geral, dentre outras variadas limitações, tais como peso, tamanho, necessidade de grande mobilidade etc. Por conta dessas características, muitos desses sistemas não possuem qualquer tipo de conexão com uma rede elétrica que lhes permita um constante atendimento de suas necessidades de consumo de energia, e acabam sendo supridos energeticamente por fontes de alimentação próprias e individuais, sendo a principal delas as baterias.

A utilização desse tipo de fonte energética, traz uma grande limitação ao sistema que a utiliza, por conta de sua natureza finita. Uma má utilização dos recursos que ela é capaz de prover pode representar um grande ônus à aplicação a que se destina o sistema, diminuindo o seu tempo de vida ou acarretando em degradações proibitivas do modo de execução da aplicação. Com isso, apresenta-se a necessidade de gerenciamento da energia que o sistema possui para ser consumida.

Nos já referidos sistemas computacionais de propósito geral, conta-se com uma miríade de soluções muito bem consolidadas que atendem a tais necessidades de gerenciamento. Contudo, a complexidade dessas soluções, bem como as dependências arquiteturais que elas apresentam (1) impossibilitam a sua utilização em um ambiente embarcado ou profundamente embarcado. Com isso, acaba tornando-se uma prática comum o próprio programador da aplicação preocupar-se com todas as questões envolvidas com o gerenciamento dos recursos energéticos do sistema que está projetando - que sejam relacionadas ao *software* -, abdicando, portanto, por falta de alternativas, das facilidades que porventura pudessem advir de um sistema externo à sua aplicação que se encarregasse de realizar essas tarefas de cunho operacional.

O objetivo deste trabalho é apresentar um sistema de monitoramento do consumo de energia implementado no sistema operacional orientado à aplicação EPOS (Embedded Parallel Operating System), projetado para sistemas embarcados. A implementação desse monitor tem a intenção de fazer com que a energia que o sistema possui para ser utilizada seja encarada como mais um recurso gerenciável em tempo de execução pelo sistema operacional, permitindo-lhe que realize reconfigurações globais que levem em consideração tais características energéticas, seja capaz de prever seu consumo, possa realizar escalonamentos dessa energia disponível, implementar sistemas de economia, e lhe permita estabelecer prioridades de consumo. A implementação do monitor se deu em um nível que permite monitorar o consumo do sistema relacionando-o a cada componente de *hardware* que ele venha a possuir.

Estrutura

No próximo capítulo, apresentamos alguns trabalhos correlatos, que também possuem a idéia de acompanhar a evolução do consumo da bateria para aprimorar as funcionalidades de algum sistema.

No capítulo 3 se apresenta uma definição a respeito de sistemas embarcados, seguindo-se uma explanação a respeito da necessidade apresentada por mecanismos dessa natureza de políticas de gerência de energia. Na seção 3.2 desse capítulo abordamos o conceito de sistemas operacionais para as arquiteturas computacionais embarcadas. Posteriormente, em suas subseções, relata-se o paradigma de Projeto de Sistemas Orientados à Aplicação, sob cujo escopo está embasada a idéia do nosso projeto. Também é apresentado o sistema operacional EPOS, sistema operacional no qual está inserido o presente trabalho como contribuição, e que foi desenvolvido dentro das especificações do dito paradigma.

Uma apresentação de técnicas de gerência de energia é apresentada no capítulo 4, bem como um estudo sobre as características de baterias que são relevantes para a elaboração do monitor a que esse trabalho se refere.

A implementação do sistema de monitoramento de consumo é apresentada no capítulo 5, explicitando as características de implementações e decisões do projeto. No capítulo 6 encontra-se a conclusão de nosso trabalho e a referência a trabalhos futuros a serem desenvolvidos como continuidade deste.

2 *Trabalhos Relacionados*

Em (2), Park introduz conceitos a serem utilizados como estratégias por um sistema que pretende aprimorar o consumo de energia. A sua abordagem leva em consideração a carga de trabalho requisitada por cada componente do sistema como um critério mais importante que uma simples estimativa de tempo de vida da bateria. Em seu experimento, baseou-se em características de baterias de íon de lítio, e utilizou-se de um simulador de baterias para a composição de seu modelo. Utilizando-se da combinação dos conceitos de *curvas de serviço e de capacidade*, apresenta duas abordagens a serem utilizadas, uma baseada em considerações feitas estaticamente a respeito da bateria, e uma segunda - baseada na primeira - que utiliza-se de dados disponíveis dinamicamente.

O trabalho apresentado por Benini et al.(3) apresenta estudos interessantes a respeito de sistemas alimentados por mais de uma bateria. Nele, o autor apresenta técnicas a serem utilizadas para um compartilhamento da requisição de carga entre as baterias contidas no sistema, embasando-se no fato de que um compartilhamento balanceado dessas requisições, previne uma sobrecarga de requisição de corrente para alguma das baterias que compõe o sistema. A partir de uma descarga de corrente direcionada, obtêm-se um maior tempo de vida de cada uma das baterias, e conseqüentemente, do sistema como um todo. No trabalho (4), Benini explora a idéia de escalonamento entre baterias com vistas a fazer com que a capacidade do conjunto mostre-se equivalente à capacidade de um sistema monolítico, que possua uma bateria equivalente ao conjunto. Isso porque um conjunto de baterias possui um desempenho menor do que um sistema monolítico de mesma quantidade de material ativo. Em um terceiro trabalho (5), Benini apresenta a idéia de que muitas vezes, redução do consumo do sistema e aumento do tempo de vida da bateria podem se tornar características extremamente distintas. A partir disso, apresenta uma série de técnicas voltadas especificamente para o aumento da vida útil de baterias utilizando-se para isso de parâmetros a respeito seu estado atual.

3 *Sistemas Embarcados*

3.1 Definição

Comumente, estamos acostumados a relacionar processamento de informações com computadores completos, da forma como os vemos e os temos em casa (monitor, teclado, mouse etc.). Porém, computadores de propósito geral não são a única forma existente de se utilizar recursos computacionais, e tampouco são capazes de atender a todas as espécies de necessidades de processamento existentes. Para muitas realidades de utilização, utilizar um processador de propósito geral como o existente em computadores pessoais (PCs) é um desperdício de recursos, ou simplesmente inviável.

Por conta disso, há um mercado em crescente expansão que trabalha com os chamados "Sistemas Embarcados". Tal mercado já é o destino da ampla maioria dos processadores existentes hoje no mundo, como pode ser constatado nos gráficos da pesquisa demonstrada por Tennenhouse(6) (figura 3.1). Também conhecidos como *embedded systems* ou sistemas embutidos, estes dispositivos costumam ser utilizados por sistemas que são desenvolvidos para desempenhar uma tarefa específica nas mais diferentes áreas do conhecimento, pesquisa e produção. Dentre elas, segundo Marwedel(7), As principais são:

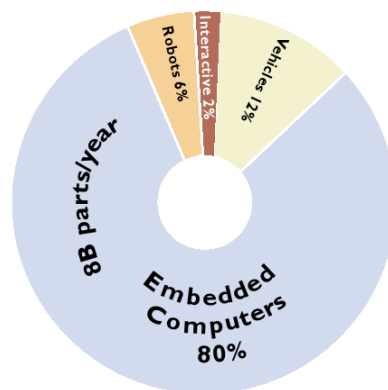


Figura 3.1: Destino dos microprocessadores produzidos até 2000
Fonte:Tennenhouse(6)

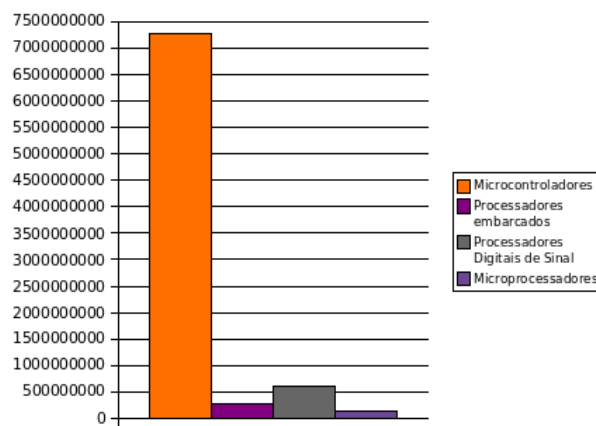


Figura 3.2: Natureza de aplicação dos processadores fabricados em 2000
Fonte: Tennenhouse(6)

- *Eletrônica Automotiva:* Um carro com um nível mediano de sofisticação pode ultrapassar facilmente a marca de 50 processadores embarcados (8). Isso ocorre por conta de muitas das funções de controle serem desempenhadas por esses sistemas, tais como o sistema de freios *anti-braking system* (ABS), air-bag, ar condicionado, injeção eletrônica, dentre muitas outras.
- *Eletrônica Aeronáutica:* Controladores de vôo, sistemas de informações que são dadas ao piloto de uma aeronave, sistemas anti-colisão, são alguns dos empregos que possuem para sistemas embarcados nesse ramo
- *Transporte Ferroviário:* Utilizados no controle de tráfego, e em uma série de mecanismos de segurança
- *Telecomunicações:* Amplamente caracterizado pelos telefones celulares, é o ramo em que o emprego de sistemas embarcados é mais conhecido do grande público, é utilizado para a transmissão e controle dos dispositivos envolvidos, e a cada dia englobam mais funcionalidades.
- *Sistemas Médicos:* Na área médica, os sistemas embarcados possuem uma grande gama de aplicações, auxiliando e aprimorando em muito o trabalho realizado nessa área, e que pode tomar vantagem pelo uso de equipamentos mais sofisticados.
- *Aplicações Militares:* Como não poderia deixar de ser, os sistemas embarcados possuem uma grande aplicabilidade para o processamento de informações militares, sendo utilizado no sensoriamento de regiões, transmissão e captação de sinais, dentre várias outras necessidades. Esta é uma grande área geradora de novos estudos e inovações tecnológicas.

- *Sistemas de Autenticação:* O número de sistemas embarcados que são utilizados com o propósito de autenticação é cada vez maior. Leitores de íris, impressão digital e reconhedores de voz são algumas das aplicações mais proeminentes.
- *Eletroeletrônicos Domésticos (Eletrônica de Consumo):* A cada dia que passa existem mais novidades sendo lançadas que visam facilitar ou trazer mais conforto e comodidade para os domicílios. Todas essas novidades são amplas utilizadoras da tecnologia que os sistemas embarcados são capazes de proporcionar. Consoles de video-games, aparelhos de áudio e vídeo são alguns representantes dessa vertente de utilização.
- *Aplicações Industriais:* Esta é uma área em que os sistemas embarcados possuem já uma grande tradição de utilização. São muito importantes para o monitoramento de processos de produção, bem como para aumentar a confiabilidade, velocidade ou segurança de atividades que devam ser desenvolvidas com a finalidade industrial.
- *Robótica:* Sistemas embarcados podem ser considerados o núcleo principal de um sistema de robótica. Nessa área, eles são usados principalmente para o controle de atividades mecânicas, bem como para a tomada de decisões.

A utilização de sistemas embarcados em todas as áreas citadas acima lhes permitiu inúmeros avanços. Além desses avanços, existem ainda outras características relevantes que podem ser consideradas vantagens encontradas em sistemas embarcados. Heath(9) ressalta do conjunto dessas as seguintes:

- Com a necessidade de sistemas cada vez mais sofisticados, que possuíssem a possibilidade de prover ao usuário uma maior quantidade de funcionalidades, a construção de sistemas baseados totalmente no comportamento lógico de circuitos foi ficando cada vez mais complexa - utilizando centenas de circuitos integrados -, ao ponto de tornar-se impraticável para muitas aplicações. Isso tem gerado uma gradativa substituição de tais conjuntos de circuitos integrados por sistemas embarcados, uma vez que com estes se possui a possibilidade de transladar toda essa complexidade para o *software* a ser executado no processador.
- Por conta da mesma necessidade já mencionada de melhorias constantes em funcionalidades de aplicações, ganha importância em um sistema que necessita de tais características a facilidade com que modificações no mesmo possam ser realizadas. Utilizando-se um sistema embarcado nesse tipo de aplicação, costumeiramente a única tarefa necessária de se realizar para tanto é uma modificação do código a ser executado pelo processador

(*upgrade* do *firmware*), permitindo com isso desde a simples inclusão de mais funcionalidades ao sistema ou sua manutenção (correção de *bugs*) até a utilização do mesmo *hardware* para finalidades completamente distintas.

- Sistemas eletromecânicos necessitam com muita frequência de uma alta precisão em seu controle, uma capacidade que facilmente costuma ir muito além das habilidades e possibilidades humanas, tais como mover um braço mecânico com precisão milimétrica ou acionar dispositivos em parcelas de tempo extremamente precisas. Uma falha no comportamento do controle desse tipo de sistema pode causar danos que vão desde uma diminuição do tempo de vida do sistema até um resultado perigoso de seu funcionamento. Utilizando-se do controle através de um processador embarcado, a tarefa de obter o controle com a confiabilidade e precisão necessitadas é muito facilitada.
- Ao se construir um dispositivo eletrônico utilizando-se de circuitos integrados com conexões externas, dificulta-se o trabalho de proteção da propriedade intelectual sobre o trabalho desenvolvido, uma vez que para se obter um esquema completo do sistema basta que se identifique cada um dos componentes eletrônicos envolvidos e se faça um mapeamento de todas as conexões, uma tarefa que embora seja trabalhosa não deixa de ser factível em muitos casos. Em contrapartida, ao utilizar-se um único *chip* que apresenta as mesmas funcionalidades do circuito anteriormente mencionado implementadas em seu *firmware*, torna-se toda a produção intelectual envolvida na construção do sistema inacessível.
- Para muitas necessidades, tem-se optado em substituir o trabalho com sinais analógicos pelo processamento digital desses sinais. Isso acontece cada vez com maior frequência e para um maior número de aplicações, pois apesar de muitas vezes a complexidade envolvida no tratamento digital ser maior e a precisão da amostra de sinal menor, existem muitos atrativos para o uso dessa alternativa. Dentre elas, podemos citar como preponderantes o baixo custo desse processamento que é exigido pela complexidade alcançada, a reconfiguração do tratamento digital pela simples troca dos coeficientes envolvidos nos cálculos, sem a necessidade de troca de componentes físicos como capacitores ou resistores envolvidos no circuitos, além de uma maior imunidade do sistema a ruídos e do fato de sistemas digitais não perderem desempenho com seu tempo de uso.

Em (7), Marwedel define sistemas embarcados como "sistemas de processamento de informação que são embutidos em um produto maior"(MARWEDEL, 2003, p. 1), e define como suas principais características as seguintes:

- Frequentemente, possuem **sensores**, que coletam informações do ambiente externo, e

atuadores, que interagem com o dito ambiente;

- Devem ser sistemas resistentes a falhas, seguros, confiáveis e robustos;
- Devem ser eficientes quanto ao consumo de energia, tamanho de código a ser executado, tempo de execução, tamanho e custo;
- Costumam ser dedicados à execução de uma determinada aplicação;
- Utilizam interfaces de comunicação com o ambiente diferentes de computadores comuns, tais como simples botões, pedais, luzes etc;
- Muitos sistemas embarcados necessitam executar uma tarefa em tempo real, sem espaços para falhas e atrasos;
- São sistemas híbridos, isto é possuem dispositivos analógicos e digitais operando conjuntamente;

3.2 Sistemas Operacionais para Aplicações Embarcadas

Sistemas embarcados costumam possuir, quando comparados com computadores de propósito geral, uma série de restrições de variadas naturezas. As frequências dos processadores utilizados em sistemas embarcados dificilmente passam de algumas dezenas de megahertz, sendo extremamente comuns processadores que trabalham na ordem dos kilohertz. Além disso, conforme pode ser visto na figura 3.3 existe uma grande predominância nesse mercado de processadores desenvolvidos em uma arquitetura de 8 bits, em oposição aos 32 ou 64 bits costumadamente encontrados nas arquiteturas de processadores de propósito geral.

Essas características tornam inviável a utilização pelo mundo de sistemas embarcados dos sistemas operacionais já consagrados e tradicionalmente utilizados pelas arquiteturas desenvolvidas para computação de propósito geral. Estes sistemas foram desenvolvidos para arquiteturas com um desempenho muito superior, e causariam na maioria das arquiteturas embarcadas um *overhead*¹ extremamente proibitivo. Além disso, é muito difícil no mundo dos computadores de propósito geral saber antecipadamente qual será o destino dado pelo usuário final a um sistema, por isso tais sistemas são desenvolvidos visando atender à maior quantidade possível de necessidades que esse usuário possa apresentar. Para isso acaba-se implementando previamente o maior número de funcionalidades que o sistema operacional possa necessitar para atender a essas demandas.

¹Processamento que não é utilizado diretamente na resolução do propósito da aplicação.

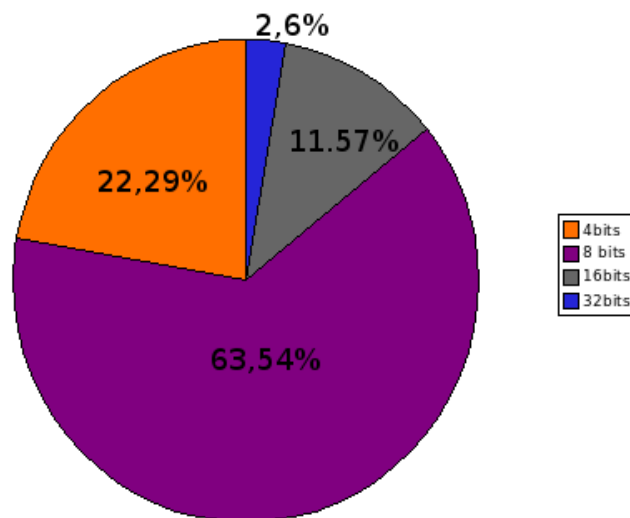


Figura 3.3: Divisão dos processadores pelo número de bits de sua arquitetura
Fonte:Tennenhouse(6)

Em se tratando de sistemas embarcados, uma decisão de projeto como essa pode indicar uma péssima alocação de recursos de memória e processamento no sistema, o que poderá afetar sensivelmente o seu funcionamento, acarretando em uma degradação de sua qualidade que pode condenar todo o desenvolvimento de um projeto.

Por conta dessas limitações, os *softwares* desenvolvidos para sistemas embarcados costumam muitas vezes serem construídos sem a presença de um sistema operacional, o que traz a necessidade de acessar a partir da própria aplicação detalhes específicos da plataforma de *hardware* escolhida. Essa estratégia de implementação faz com que todo o controle necessário para a utilização dos componentes de *hardware* seja implementado conjuntamente à aplicação a qual o sistema embarcado se destina. Isso torna a atividade do desenvolvedor mais complexa, já que este necessita conhecer de detalhes do *hardware*, e limitante, pois uma vez escolhida a arquitetura alvo, uma modificação dessa decisão no meio do projeto pode representar a não utilização do código já implementado que seja específico para o hardware anteriormente escolhido, o que leva a sua reimplementação de acordo com as necessidades apresentadas pela nova arquitetura.

3.2.1 Um Sistema Operacional Orientado à Aplicação

O uso de um sistema operacional independente da arquitetura de *hardware*, que seja caracterizado por um baixo *overhead* e que possua unicamente as necessidades de controle requisitadas pela aplicação em questão, colabora muito com a tarefa de desenvolvimento de código para sistemas embarcados. Isso porque, além de possibilitar uma reusabilidade por conta da independência adquirida com relação à arquitetura, abstém o programador da aplicação de conhecer

detalhes extremamente específicos da plataforma em questão, permitindo-lhe usar de algumas das facilidades já existentes para programadores de computadores de propósito geral.

A construção de um sistema operacional embarcado que seja projetado respeitando o paradigma de orientação à aplicação (Application Oriented System Design - AOSD) proposto por Frölich em (10), pretende fazer com que as características utilizadas para a elaboração de um sistema operacional provenham diretamente das necessidades apresentadas pela aplicação que este se destina a gerenciar. Isso permite que o S.O. seja moldado de acordo com as necessidades apresentadas por esta aplicação. Essa abordagem segue uma vertente oposta à comumente utilizada na computação em geral, na qual a aplicação é que geralmente necessita adaptar-se às características do sistema operacional. Com essa mudança de foco e hierarquia de projeto, pretende-se obter um conjunto aplicação/S.O. mais conciso, enxuto e, por consequência, muito mais eficiente e conveniente para uma aplicação embarcada.

A configurabilidade atribuída ao sistema operacional é conquistada através de uma modelagem capaz de decompor o domínio de atuação do sistema operacional em abstrações de componentes de hardware e software. Na construção de tais abstrações procura-se eximir ao máximo cada um dos componentes construídos de especificidades arquiteturais, visando a maior reusabilidade de código possível por conta independência de contexto que se conquista. Para tanto, a modelagem é feita levando em consideração quais são as características essenciais (e portanto, globais) de cada abstração, e quais são as características específicas, que só existem dentro de um determinado contexto. Assim criam-se, respectivamente, famílias de abstrações - que são agrupadas de acordo com suas características comuns - e aspectos de cenário, que definem um contexto de utilização. Realizada tal decomposição do domínio, e fornecendo-se ao programador da aplicação interfaces de acesso aos componentes gerados, permite-se em tempo de compilação um grande processo de configuração do sistema operacional.

3.2.2 Programação Orientada a Aspectos

Outro paradigma útil no esforço de atribuir aos componentes modelados dentro do conceito de orientação à aplicação é o de programação orientada a aspectos. Em um desenvolvimento baseado em componentes, define-se como aspecto todas as características não funcionais de um componente. A separação dessas características do conjunto das que são essenciais à definição de um componente é mais um fato que colabora com a configurabilidade do sistema, bem como auxilia na independência arquitetural.

Essas características não funcionais são reunidas em unidades distintas dos componentes, denominadas **aspectos**. Sua utilização se dá através de definições do usuário e de características

do contexto geral do sistema, e é consolidada através de interligadores de aspectos (*weavers*) (figura 3.4).

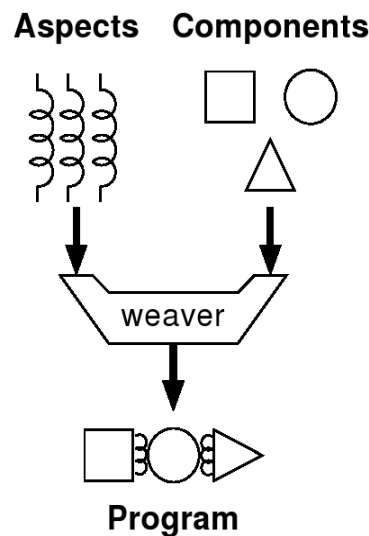


Figura 3.4: Esquema da interligação de aspectos
Fonte:Frölich(10)

3.2.3 O EPOS

Os conceitos apresentados nas seções anteriores são características do sistema operacional EPOS (*Embedded Parallel Operating System*), sendo este sistema o resultado do trabalho apresentado por Frölich em (10), com respeito a projetos de sistemas orientados à aplicação.

O EPOS é, portanto, um sistema operacional construído seguindo o paradigma orientado à aplicação. Foi desenvolvido para plataformas dedicadas de alto desempenho, focalizando especialmente a computação paralela e embarcada. Consiste em um repositório de componentes de *software* que encapsulam abstrações de sistema e aspectos de contexto, uma série de ferramentas para a sua seleção e configuração automática, e faz uso de técnicas sofisticadas de programação orientada a objetos com a intenção de gerar instâncias de sistemas voltadas às necessidades apresentadas pela aplicação.

O sistema implementa abstrações orientadas à aplicação que definem gerenciamento de memória, gerenciamento de processos, coordenação, comunicação e *I/O*, dentre outras. Entidades com forte dependência arquitetural tais como CPU, são modeladas separadamente como mediadores de hardware, e são utilizadas pelas interfaces de hardware do sistema. É pelo uso dessas interfaces que o EPOS pretende manter sua portabilidade, uma vez que, independente do mediador de hardware a ser utilizado para uma arquitetura, a sua utilização pelo usuário final se dará da mesma forma.

As características do sistema operacional que dizem respeito a compartilhamento, alocação, proteção, atomicidade, invocação remota, depuração etc., utilizam-se do já mencionado paradigma de orientação a aspecto para tal, definindo aspectos de cenário e posteriormente criando um contexto de execução através de seus adaptadores.

4 *Gerência de energia*

4.1 Definição

Dentre a grande variedade de ambientes e necessidades nas quais se utiliza de sistemas embarcados, muitas não permitem a conexão do sistema a uma rede elétrica que lhe forneça uma alimentação energética constante e permanente. Isso pode ser causado por conta da necessidade de mobilidade apresentada por um problema que a aplicação pretende solucionar, pela inexistência de uma rede elétrica utilizável no local de atuação do sistema, ou por outras questões específicas e variadas. Com isso, surge a necessidade de muitos desses sistemas embarcados serem alimentados por fontes alternativas de energia, principalmente baterias, que possuem capacidade limitada de prover recursos energéticos. Por isso, mostra-se necessária uma utilização inteligente e eficiente da energia da qual se dispõe.

Por conta disso, existem hoje estudos em variadas áreas focalizando o aperfeiçoamento e criação de técnicas de redução de consumo. Park(11), define os esforços existentes hoje em três classes distintas. A primeira engloba os estudos que visam o desenvolvimento de *hardware* de baixo consumo. Como o escopo de nosso trabalho diz respeito a técnicas utilizadas em software, não trataremos aqui desse tipo de técnicas.

A segunda classe é composta por *softwares* que implementam algoritmos eficientes quanto ao consumo de energia. Tais algoritmos conquistam essa eficiência efetuando transições de estado de consumo de componentes de hardware do sistema. Exemplos dessa categoria são apresentados na sessão 4.2.

Na terceira e última classe, encontram-se as técnicas denominadas por Sung como "Abordagens Cientes do Consumo" (*Power-Aware*). Essa classe de técnicas, na qual o autor já citado inclui o seu trabalho, e é também onde o nosso trabalho deve ser enquadrado, é caracterizada pela capacidade de monitorar dinamicamente as modificações que vão ocorrendo com os recursos de energia que o sistema possui, em decorrência do seu uso. Para isso, é necessário que se haja a capacidade de monitoramento de tais recursos de hardware em tempo de execução

e formas de monitorar e acompanhar o consumo da aplicação, com o intuito de tornar o sistema capaz de realizar previsões relacionadas com o seu tempo de vida segundo seu comportamento, e assim permitir a tomada de decisões embasadas nesses critérios.

4.2 Técnicas Mais utilizadas

Por conta do escopo do presente trabalho, esta seção pretende apresentar somente as principais técnicas de redução de consumo existentes atualmente no âmbito de programação, sem levar em consideração as técnicas voltadas ao aprimoramento da arquitetura de *hardware* do sistema. A grande maioria das técnicas amplamente difundidas e consolidadas seriam classificadas como sendo da segunda classe de técnicas, dentro da taxonomia apresentada na seção 4.1.

4.2.1 Dinamic Voltage Scaling (DVS)

Sendo a mais amplamente utilizada de todas as técnicas, DVS consiste em uma configuração dinâmica, ou seja, com o processador em atividade, da tensão de alimentação do sistema e de sua frequência de operação combinadamente. O preceito principal é que a tensão de operação de um processador pode variar - e, conseqüentemente, sua frequência - de acordo com a carga de trabalho necessária de ser desempenhada em um dado momento. Isso porque é comprovadamente mais econômico no que diz respeito ao consumo de energia utilizar-se de todo o tempo que se possui para a execução de uma tarefa. Dessa forma, se o tempo máximo para que a execução de uma dada tarefa seja de 10 segundos e, com a frequência atual o processador consegue executá-la em 1 segundo, existindo a possibilidade de torná-lo mais lento de forma a utilizar todos os 10 segundos que se possui, os recursos energéticos serão melhores aproveitados do que se esta tarefa for realizada em 1 segundo e o sistema permanecer ocioso pelos 9 segundos restantes.

4.2.2 Técnicas utilizadas na Compilação

Algumas abordagens visam apurar uma boa utilização dos recursos do sistema atribuindo ao compilador a capacidade de tomar decisões que auxiliem na economia de energia. As principais otimizações que podem ser feitas em um compilador para isso são as seguintes:

- **Reordenamento de Instruções:** Enquanto a modificação da ordem de execução das instruções não deturpar a semântica do sistema, é possível a realização de reordenamentos de

código. Isso pode ser interessante para diminuir o número de transições lógicas de sinal no barramento de instruções no processador e, conseqüentemente, poupando energia.

- **Seleção de instruções que consomam menos:** Geralmente existem várias formas de se implementar em código de máquina a mesma funcionalidade descrita em uma linguagem de mais alto nível. Dessa forma, o compilador pode utilizar-se de técnicas para classificação das instruções conforme o seu consumo, e sempre definir por usar as mais convenientes para cada situação. Esta é a técnica apresentada por Steinke et al.(12). Contudo, esta é uma técnica bastante complexa de ser utilizada, uma vez que não pode-se estabelecer uma relação absoluta entre o consumo de uma instrução e o seu real significado dentro de um sistema como um todo. A decisão pela utilização de uma instrução que consuma menos em detrimento de outra, pode, por exemplo significar a necessidade maior paralelismo de hardware ou maior quantidade de ciclos de execução, o que no fim pode não significar uma economia de energia.
- **Redução de acessos à memória:** É conhecido que um dos grandes responsáveis pelo consumo de energia em tempo de execução são os acessos à memória. Por isso, é uma prática interessante para prevenir consumo desnecessário evitar acessos redundantes à memória, utilizando-se o máximo possível da alocação de registradores para acesso a variáveis, bem como de técnicas que favoreçam uma boa utilização da memória cache disponível.

4.2.3 Hibernação de Recursos

É muito difícil que um sistema opere todo o tempo utilizando a plenitude de suas capacidades e recursos de *hardware*. Contudo, os componentes do sistema continuam consumindo energia, mesmo quando ociosos. A técnica de hibernação de recursos utiliza-se da possibilidade de desligamento de determinadas parcelas do dispositivo nos momentos em que estas estejam ociosas. O caso mais simples é o que permite escolher entre deixar um componente, tal como o *clock* do sistema ou algum periférico, ligado ou desligado. Porém, muitas plataformas possuem diversos níveis diferentes dentre os quais cada componente do sistema pode ser configurado para operar, cada qual com as suas vantagens de economia e suas restrições de capacidade e desempenho. Existe uma grande quantidade de estudos realizados nesse âmbito, que tem o propósito de encontrar a melhor forma - para cada necessidade e característica do sistema - de se transicionar o sistema entre os estados de economia que este possui.

Em (13), Hoeler apresenta uma abordagem de tratamento hierárquico de componentes de

sistemas embarcados que preza pela necessidade de baixa utilização de recursos necessária nesse tipo de sistema. Simunic et al.(14) representa outro exemplo dessa categoria, que também trabalha com os conceitos de transições de estados, porém para plataformas com mais recursos, tais como *laptops*.

4.3 Baterias

Algumas Características Gerais

A presente seção visa contextualizar e esclarecer alguns breves detalhes técnicos A respeito das baterias. Pela definição de Linden(15), baterias são dispositivos que convertem a energia química contida em seu material ativo diretamente em energia elétrica através de uma reação eletroquímica de óxido-redução, caracterizada pela transferência de elétrons a partir de um material reagente (ânodo), passando em um circuito elétrico (dispositivo alimentado pela bateria) e chegando a outro material reagente (cátodo).

Embora a indústria de baterias tenha conquistado importantes avanços nos últimos anos, a sua taxa de evolução não está acompanhando a necessidade de potência cada vez maior requisitada pelos sistemas computacionais atuais, criando um abismo cada vez maior nessa relação (figura 4.1). Além disso, a potência dos equipamentos computacionais, que é diretamente relacionada à energia consumida pelos mesmos, é um fator determinante para o desempenho dos processadores. Por isso, mostra-se necessário um estudo de relação que possibilite amenizar tais disparidades de avanços tecnológicos.

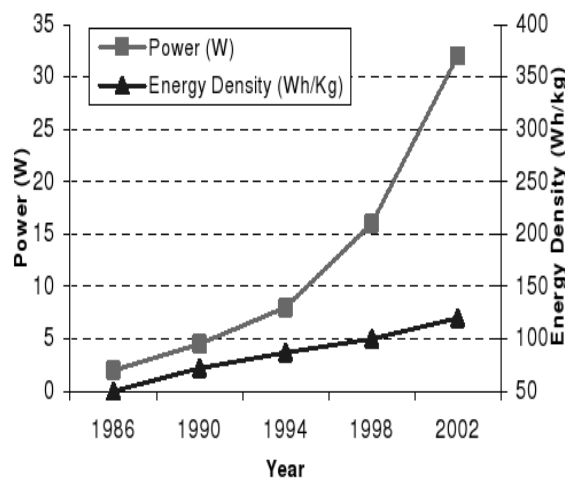


Figura 4.1: Disparidade entre a evolução de sistemas computacionais e baterias

Fonte: Park(11)

As principais unidades de medida envolvidas na classificação de capacidade de uma bateria são:

- Energia Teórica: é o valor máximo que um dado sistema eletroquímico é capaz de prover, calculado em Wh (Watt-hora)
- Energia Específica: Esse valor define a capacidade eletroquímica que um determinado tipo de material possui, é medida em Wh/l (Watt-Hora por litro). Detalhes na figura 4.2.
- Capacidade Nominal: Medida em miliampère-hora (*mAh*), é o valor utilizado para definir o tempo de vida de uma bateria. Será explicada com maiores detalhes posteriormente.

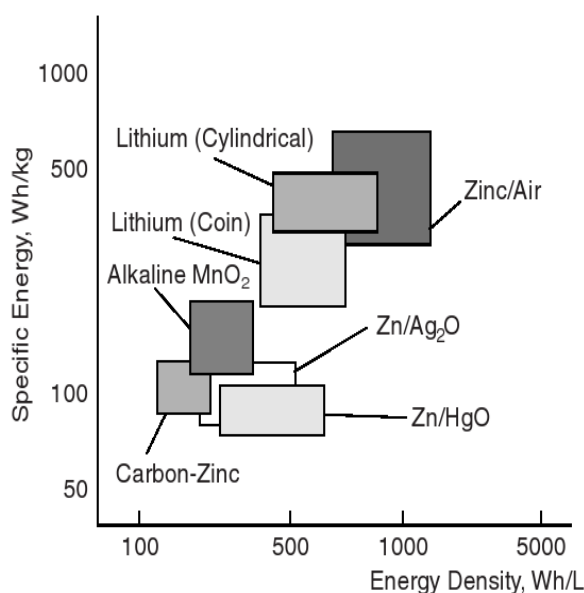


Figura 4.2: Comparação de densidades energéticas (Wh/l) de diferentes tipos de baterias
Fonte:Linden(15)

Diferentemente do que se costuma supor, as baterias não possuem a capacidade de esgotar todo o seu potencial energético. A figura 4.3 demonstra essa importante característica das baterias, apresentando as peculiaridades existentes para cada tipo de composto.

A principal distinção feita entre o conjunto existente de baterias se dá com dependência da sua capacidade de ser eletricamente recarregada. Por conta da existência ou não dessa característica, elas são classificadas entre baterias **secundárias** e **primárias**, respectivamente. Baterias secundárias costumam ser utilizadas em meios nos quais se possui acesso a uma fonte primária de energia que possibilite a sua constante recarga. Bons exemplos de baterias recarregáveis são as baterias automotivas. Costumam possuir uma densidade energética ¹ menor do que as

¹Capacidade de provisão de energia relacionada à quantidade de material ativo - maiores informações são encontradas em (16) e (15).

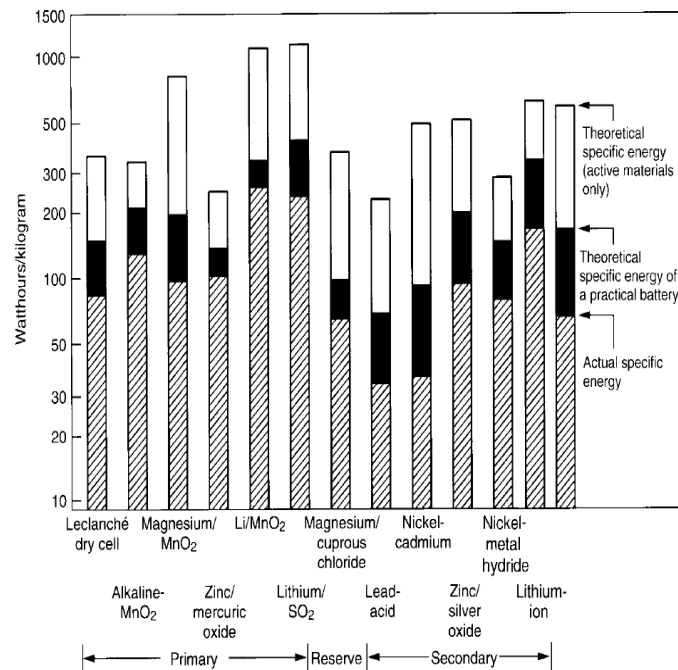


Figura 4.3: Energia teórica e energia real das baterias
Fonte:Linden(15)

baterias primárias, bem como uma menor capacidade de retenção de sua carga. Embora atualmente a utilização de baterias secundárias em componentes eletrônicos tenha crescido, as baterias primárias ainda são mais amplamente utilizadas nesse mercado. Isso porque estas últimas costumam ser mais baratas e, conforme já mencionado, possuem uma maior densidade energética, o que para muitas necessidades é prioritário sobre a capacidade de recarga. Em virtude dessa maior utilização, nos voltaremos no restante dessa sessão a características específicas das baterias primárias do tipo alcalinas.

Baterias Alcalinas

As baterias do tipo alcalina são o principal tipo de baterias utilizadas em sistemas portáteis. Desde que foi introduzida nesse mercado, têm construído uma ótima reputação e mostrado-se como a melhor solução diante de todas as suas concorrentes. Algumas de suas características mais relevantes em comparação com as demais são: Uma maior densidade energética, melhor rendimento tanto em situações de demanda contínua de carga quanto intermitente, menor resistência interna e, principalmente, uma maior vida útil. Em algumas aplicabilidades, chega a possuir um rendimento 50% maior que o de suas concorrentes. Esta bateria é encontrada no mercado em dois formatos, cilíndrico que, comumente em nosso país recebe o nome de **pilha**, e no formato "moeda". A figura 4.4 apresenta curvas de desempenho desse tipo de bateria em alguns de seus empregos mais comuns. Por conta de suas características, é um tipo de bateria

amplamente utilizado por dispositivos embarcados, sendo, portanto, o tipo de bateria escolhida como base do estudo e trabalho realizado em nosso monitor.

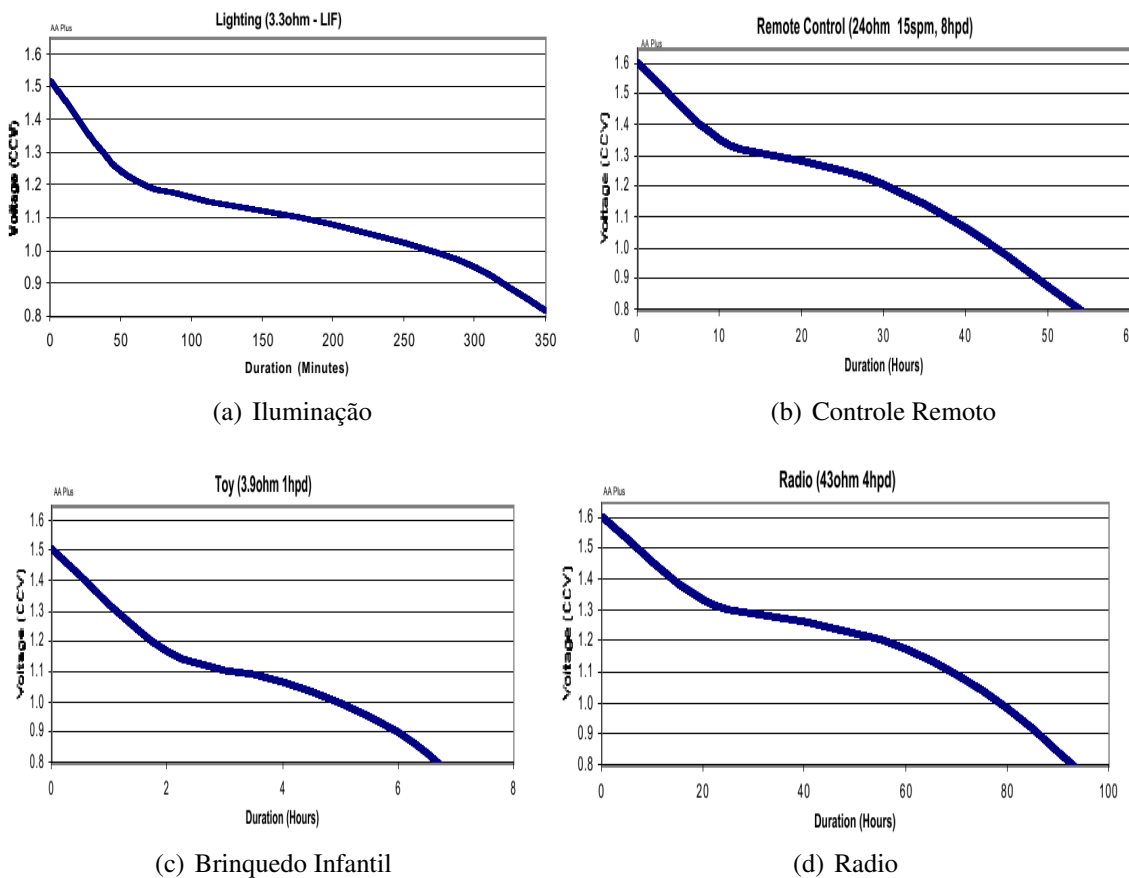


Figura 4.4: Consumo de uma bateria alcalina em diferentes casos de uso.

Fonte: PANASONIC Ind. Comp.(17)

Capacidade das Baterias

A parametrização dos dados de capacidade energética das baterias mostra-se uma tarefa complexa. Isso porque não existe um padrão de comportamento que possa ser atribuído a todos os tipos de baterias existentes, mesmo quando se limita seu modelo e composição como fizemos, tratando apenas de baterias AA de composição alcalinas. Apesar de dentro de uma classificação como esta as baterias mostrarem-se bastante similares, possuindo inclusive curvas de descargas muito parecidas, possibilitando até uma relação de equivalência nesse quesito (as figuras em 4.4 são um exemplo dessa similaridade entre as curvas), existem disparidades a respeito da capacidade de energia que estas são capazes de prover ao sistema. Esta capacidade é costumeiramente referida como um valor medido em mAh (miliampére-hora) e varia por conta de compostos diferentes, quantidades diferentes desses compostos, qualidade desses ou

até mesmo por métodos de fabricação diferenciados. Apesar disso, o valor dessa capacidade nominal para cada tipo de bateria costuma ser de fácil acesso aos seus usuários em geral, vindo assinalado na própria embalagem da bateria ou sendo facilmente encontrado em especificações difundidas por seus fabricantes. Para facilitar o entendimento do que significa essa unidade de medida, basta utilizar-se o seguinte exemplo: Definir que uma bateria possui a capacidade nominal de 10 *mAh* significa dizer que esta bateria é capaz de fornecer para o sistema que alimenta uma corrente de 1 miliampére pelo período de 10 horas, 10 miliampéres por um período de 1 hora, 5 miliampéres por 2 horas ou qualquer outra combinação de valores cuja multiplicação resulte no valor nominal, representado por 10, no nosso exemplo².

²Conforme apresentado por Bellosa(18), e comprovado pela Lei de Peukert, a capacidade de provisão de energia de uma bateria sofre grande influência segundo a taxa com que a corrente é drenada. As baterias possuem a propriedade de aumentar consideravelmente a sua capacidade se são submetidas a baixas taxas de drenagem de corrente, e têm a sua capacidade diminuída caso contrário.

5 *Monitorando o consumo de energia*

5.1 **Desafios do Projeto**

A idéia principal do projeto é tornar os recursos de energia do sistema passíveis de serem gerenciados pelo sistema operacional via técnicas de monitoramento do seu consumo. Conforme já mencionado anteriormente, o sistema operacional para o qual o mecanismo foi desenvolvido, o EPOS, é um sistema orientado à aplicação, e seu projeto de desenvolvimento é baseado em uma clara estrutura de componentes. Adicionando a tal sistema modificações que permitam a implementação do monitor, este conquistará uma capacidade de gerenciamento da energia do sistema. Isso lhe possibilitará o oferecimento de mais um importante serviço às aplicações clientes que porventura necessitem de um gerenciamento aprimorado de energia pois, por conta dessa implementação, os recursos de energia do sistema poderão ser encarados como sendo de primeira classe na visão do sistema operacional. A partir daí, é possível realizar reconfigurações globais do sistema que levem em consideração tais características energéticas, e conquista-se a possibilidade de monitorar o comportamento de consumo, realizar predições a seu respeito, dentre outras características interessantes.

Benini et al.(5) apresenta uma taxonomia interessante para sistemas de gerenciamento de energia, a saber:

"(...) as regras do tipo *closed-loop* utilizadas para controlar os estados de operação do sistema são baseadas na observação da tensão amostrada na bateria (que não é relacionada linearmente com a taxa de descarga). Isso vem em contraposição às soluções *open-loop*, que tomam suas decisões a respeito do delisgamento de componentes de forma independente da tensão medida na bateria." (BENINI et al., 2001, p. 1, tradução nossa).

Com base nessa afirmação, podemos classificar o nosso trabalho como sendo uma ferramenta que pretende tornar-se uma alternativa que facilite ou até viabilize a utilização de técnicas do tipo *closed-loop* sem a necessidade de um constante acompanhamento real da tensão da bateria. Justamente por conta de possuírem um maior número de requisitos nos quais embasar as suas decisões - propiciados pela medida da tensão - as técnicas têm a capacidade de toma-las de forma mais confiável¹.

A primeira decisão que mostrou-se necessária ser tomada para a implementação do monitor foi definir em que consistia o monitoramento. Qual seria a característica mensurável da bateria capaz de denotar o seu estado atual? As características a serem medidas em uma bateria podem modificar-se dependendo do tipo de bateria utilizado. Em conformação com o apresentado na seção 4.3, em nossa abordagem foi decidido pela utilização de baterias primárias do tipo alcalinas, já que este é um tipo amplamente utilizado em um variado conjunto de plataformas.

Como já foi apresentado, esse conjunto de baterias possui a característica de ter a sua tensão diminuída em decorrência de seu uso (Figura 4.4). Como a maioria dos sistemas embarcados existentes possuem um Conversor Analógico Digital (ADC) como periférico, o acompanhamento da tensão da bateria em um dado momento é feito pela utilização desse dispositivo, através de leituras (*samplings*) efetuadas em sua tensão amostrada e uma decorrente conversão para um valor digital utilizável pelo sistema.

Apesar de ser possível um constante acompanhamento do nível de tensão em que a bateria se encontra utilizando-se dos recursos providos pelo ADC, cada amostragem realizada implica por si só em consumo de energia do sistema, pela utilização do referido periférico, além de um *overhead* considerável, e nenhuma dessas características é interessante. Para tentar diminuir os efeitos e interferências causadas pela utilização do monitor no comportamento do sistema como um todo, partiu-se para a implementação de uma estrutura que permitisse acompanhar a evolução do consumo de forma aproximada. Essa abordagem utiliza-se de informações previamente conhecidas a respeito de características específicas da bateria e dos componentes de *hardware* do sistema a ser monitorado - ambas já contidas no S.O. ou fornecidas pelo usuário previamente à compilação. A dita estrutura define no início da utilização do sistema os valores do estado atual de cada um dos componentes envolvidos no monitoramento, atribuindo à bateria um valor de capacidade e criando meios para que possa ser atribuído a cada componente a quantidade de energia consumida em tempo de execução. Isso é feito através de incrementações e decrementações envolvidas em ambos os lados (bateria e sistema), de forma a retirar

¹Nosso sistema não simula a tensão da bateria em tempo de execução, e se utiliza dela apenas para início do processo de monitoramento, porém os dados providos por ele podem ser utilizados para os mesmos fins, no tocante a definir o estado atual da bateria.

da capacidade de energia da bateria a quantidade consumida por um componente, e atribuir esta quantidade ao conjunto de energia consumido por esse componente. Com isso, obtém-se uma aproximação da capacidade de energia que ainda é capaz de ser fornecida pela bateria, além de um acompanhamento acerca da parcela de participação referente a cada componente de hardware dentro do montante geral de energia consumida pelo sistema. Todo o esquema de funcionamento apresentado será explicitado com maiores detalhes no decorrer do presente capítulo.

A partir do valor de tensão da bateria amostrado pelo ADC, é necessário definir o que esse valor obtido significa em termos quantitativos de energia. Isso porque, como já foi demonstrado na seção 4.3, uma bateria não tem a capacidade de converter toda a tensão que possui em recurso utilizável pelo sistema que ela alimenta. Concomitantemente, sistemas computacionais também possuem limiares a partir dos quais a tensão que os alimenta torna-se utilizável ou inutilizável, segundo esteja dentro ou fora da faixa necessária para sua perfeita operação. Além disso, para que se consiga realizar as atribuições de consumo para cada um dos componentes e o decremento da capacidade energética do modelo, é necessário estabelecer uma relação numérica discreta entre o valor amostrado e quanto deve ser decrementado da bateria/atribuído a um componente.

Para o estabelecimento dessas convenções, nos inspiramos nos conceitos de *Currentcy* e *Utility*, apresentados por Zeng et al.(19) e Pillai, Huang e Shin(20), respectivamente. Assim como estas, a nossa técnica de medição pretende possuir a capacidade de indicar o atual estado energético do sistema, e ainda possibilitar a sua utilização em uma modelagem de escalonamento votada à energia, servindo como uma espécie de unidade de crédito a ser concedida a uma entidade escalonável do S.O., possibilitando extensões desse trabalho.

Não é possível definir-se previamente a capacidade nominal de uma bateria. Mesmo dentro de baterias de mesma classificação, muitos fatores acabam colaborando para a existência de disparidade entre as suas capacidades. Para a solução do problema mostra-se a necessidade de esse ser um dado configurável pelo programador da aplicação, que definirá então a capacidade da bateria que utilizará no seu sistema. Apesar de apresentar uma limitação, já que o sistema só estará configurado para um tipo de bateria, para situações que permitam a recompilação do sistema sua reconfiguração pode ser realizável modificando-se esse valor numérico e efetuando-se essa recompilação.

5.2 Implementação

Por conta de o sistema operacional EPOS possuir o propósito de ser uma ferramenta para a construção de aplicações livre de dependências arquiteturais, adaptáveis à plataforma alvo via recursos de configuração providos pela própria ferramenta (abstrações de entidades do sistema operacional, componentes, mediadores de hardware e aspectos) em tempo de compilação, existe uma grande parcela da implementação do monitor que torna-se compartilhável entre todas as plataformas para as quais o EPOS tenha sido portado. Contudo, há uma série de características que são dependentes de detalhes do hardware. Com isso, escolhemos para a implementação e validação do nosso monitor a arquitetura **Atmel AVR8**. Esta é a arquitetura do processador **ATMega128L** ATMEL Corp.(21) (figura 5.1), presente no dispositivo CrossBow Mica2 (22), família MPR400/MPR410/MPR420, um dispositivo que contém um rádio-transmissor e permite o acoplamento de uma placa de expansão com conjunto de sensores de diversas funcionalidades, dentre luminosidade, aceleração, temperatura, etc.

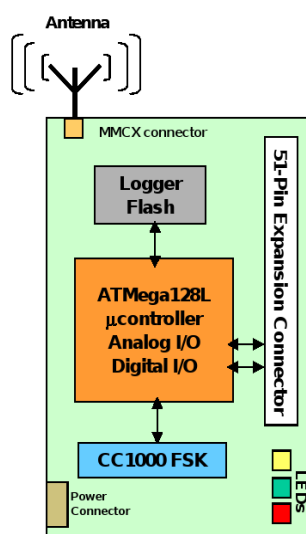


Figura 5.1: Esquemático Mica2 MPR400/MPR410/MPR420
Fonte: CROSSBOW Technology Inc.(22)

A alimentação energética do Mica2 é feita por duas pilhas AA (figura 5.2), e a tensão mínima dentro da qual o sistema opera é de 2.1V.(23). A consideração dessa característica é muito importante, uma vez que é uma das que são necessárias observar para que se possa estabelecer um valor numérico monitorável para a capacidade do sistema. Utilizaremos o procedimento de obtenção dos valores para o nosso próprio sistema como exemplo.

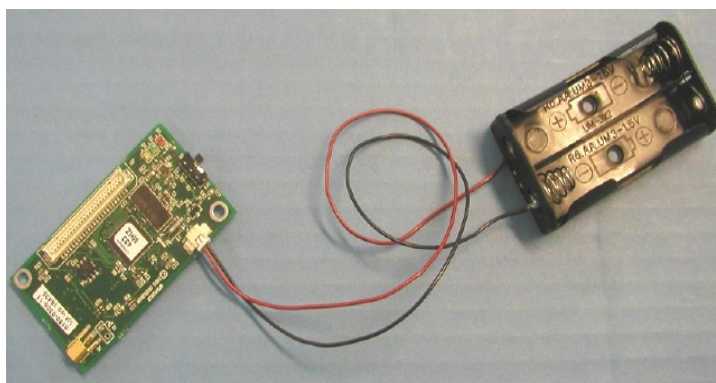


Figura 5.2: CrossBow Mica2 MPR400/MPR410/MPR420 e o adaptador para bateria AA
Fonte: CROSSBOW Technology Inc.(22)

5.2.1 Representação numérica da capacidade da bateria

Dentro do esforço para se encontrar uma representação numérica proporcional para os valores amostrados da bateria, surgiu a necessidade de se responder a seguinte pergunta: *Como saber que uma dada tensão significa uma determinada capacidade da bateria?* Com todas as características das baterias que foram mencionadas até aqui, fica comprovada a complexidade e até a inviabilidade de desenvolver um padrão válido para todas as categorias de baterias. Então, para que pudéssemos estabelecer esse relacionamento para o nosso caso de uso definido, mostrou-se necessário o acesso a parâmetros com um bom nível de detalhes a respeito do comportamento da descarga de uma pilha alcalina. Para isso, realizamos um teste simples, que consistiu em deixar uma aplicação que utilizava todos os recursos de hardware do sistema, inclusive uma placa extensora de sensores, ser executada com uma bateria nova, até esgotar totalmente a sua capacidade². Durante a sua execução, monitoramos a evolução da descarga da bateria, e obtivemos o gráfico de relação Tensão x Tempo apresentado na figura 5.3.

As diferentes espessuras do traço da curva que podem ser observadas nessa figura indicam uma maior ou menor quantidade de amostragens da bateria que anunciaram que esta encontrava-se em um dado valor, como pode ser visto na sua aproximação de um intervalo (figura 5.4).

Com base nessa característica observada, mostra-se que é possível definir o percentual de uso de uma bateria conforme a tensão indicada pela mesma em uma leitura do ADC. Para tanto, a parcela da curva de descarga compreendida entre o início da execução da aplicação e a tensão mínima de operação definida para o dispositivo que, conforme mencionado anteriormente, é de

²Em nosso experimento utilizamos uma alta taxa de drenagem que, por ação da Lei de Peukert, consequentemente diminuiu a capacidade da bateria. Contudo, isso não influencia de forma preocupante o comportamento da curva de descarga da mesma, apenas torna-a mais rápida. Como a velocidade de descarga foi a mesma em todos os momentos, manteve-se a proporcionalidade entre as curvas de uma taxa de descarga maior e menor, não causando um grande prejuízo aos resultados do experimento.

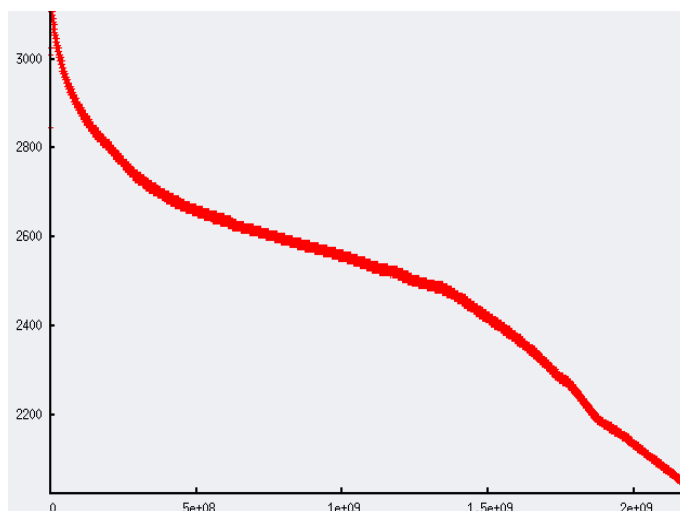


Figura 5.3: Curva de descarga Tensão (em milivolts) x Tempo(em milissegundos) do experimento realizado

2.1V nesse caso³, foi subdividida em parcelas que correspondam ao mesmo intervalo de tempo, que podem ser interpretadas como uma *época* da bateria. Com isso, ao realizar-se uma leitura da tensão, o que se deve fazer para definir a sua capacidade atual é simplesmente identificar a que época a tensão amostrada pertence, e assim consegue-se definir o percentual de tempo já passado desde o início da vida útil da bateria até o presente momento, em quantidades de épocas, que representam uma quantidade de energia já consumida. Com isso, é possível definir a atual situação da carga da bateria. A tabela 5.1 demonstra como ficou a dita subdivisão em nosso caso de uso.

Tabela 5.1: Curva subdividida em intervalos de tempo iguais

Lista de Intervalos	NumSamplings	% Tempo do Total	Intervalo (mV)
3124 a 2796 mV	130667	9,98%	328 mV
2795 a 2688 mV	123528	9,43%	107 mV
2687 a 2631 mV	131923	10,07%	56 mV
2630 a 2588 mV	135245	10,33%	42 mV
2587 a 2551 mV	125019	9,54%	36 mV
2550 a 2505 mV	133654	10,20%	45 mV
2504 a 2447 mV	128432	9,81%	57 mV
2446 a 2346 mV	133209	10,17%	100 mV
2345 a 2199 mV	135575	10,35%	146 mV
2200 a 2100 mV	132567	10,12%	100 mV
Total	1309819	100,00%	

³É importante frisar que, embora o sistema porventura continue funcionando após a tensão especificada, esse funcionamento já se encontra fora dos limites confiáveis definidos pelo fabricante, por isso descartamos essa parcela da execução.

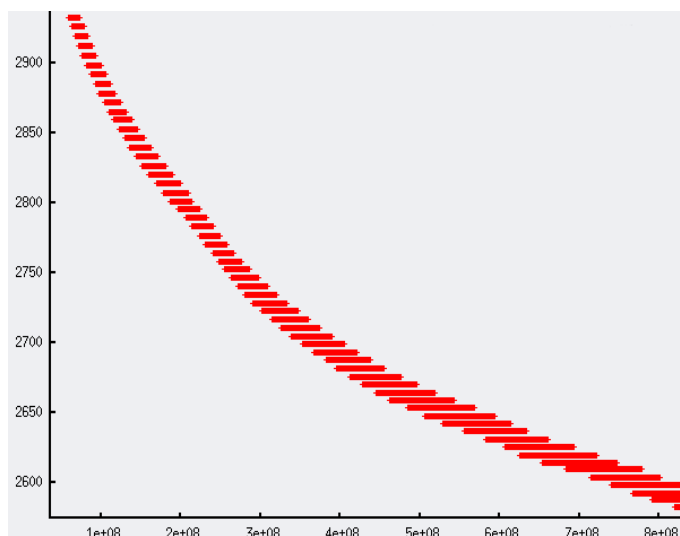


Figura 5.4: Aproximação de um intervalo: o sistema permanece mais em algumas tensões que em outras.

Assim, conseguimos estabelecer uma relação entre o percentual de carga existente na bateria e sua tensão. Agora, *que valor é esse, que representa o percentual de carga que a bateria ainda possui?* É nesse ponto que consideramos a anteriormente citada **capacidade nominal** da bateria, medida em *mAh*. Em nosso experimento, por exemplo, cada uma das duas baterias utilizadas⁴ possui uma capacidade nominal de 2500 *mAh*. O que significa que nosso sistema possui em seu início de operação um total de 5000 *mAh* a serem consumidos. À medida que utilizamos a energia das baterias, a capacidade das baterias vai diminuindo, até que chegue a zero, quando não possuem mais carga a ser cedida - sua tensão tornou-se insuficiente. Assim, o valor numérico que procuramos é justamente um percentual dessa capacidade nominal, que será definido com base na amostragem da tensão das baterias. A principal função do monitor é, portanto, a partir dos dados de consumo dos componentes, decrementar esse valor obtido de forma a torna-lo correspondente à carga real que a bateria possui a cada momento.

5.2.2 Configurações Gerais Necessárias

Para que o EPOS tenha a capacidade de se adaptar às necessidades que a aplicação que ele irá gerenciar possui é, naturalmente, necessário que ele as conheça. Muitas dessas necessidades o sistema consegue assinalar simplesmente através da programação desta aplicação. Um exemplo disso são os periféricos do sistema para os quais a aplicação necessitará de suporte. Para identifica-los, basta instanciar os que ela declarar explicitamente no seu código. Contudo, nem todas as necessidades da aplicação são definíveis através da sua simples programação, e

⁴Baterias do fabricante Maxell modelo Alkaline ACE LR6(SN)/AA/1.5V. Detalhes em: http://www.maxell.de/alkalineace_c.html

outras podem possuir mais de uma alternativa de apresentação, e o usuário do sistema deve possuir a liberdade de escolher entre cada uma delas. Para permitir que essas configurações sejam realizadas, o EPOS possui um mecanismo que utiliza técnicas de *metaprogramação estática* implementado de forma a realizar as configurações necessárias em tempo de compilação, não representando uma sobrecarga na execução do sistema final. Esse mecanismo é denominado *Traits*, e engloba um conjunto de questões configuráveis em vários níveis do sistema, desde o comportamento de mediadores de hardware até parâmetros de funcionamento de abstrações operacionais e serviços providos.

Em primeiro lugar, para que o sistema utilize o monitoramento de energia, é necessário que o usuário solicite esse serviço. Isso é realizado definindo a variável *consumption_monitoring* pertencente a *Traits* como *true*. Nosso sistema de monitoramento também utiliza o dito mecanismo para permitir algumas configurações de funcionamento e requisitar algumas informações do usuário. É através do mecanismo *Traits* que o usuário define a capacidade nominal da bateria, necessidade demonstrada na seção 5.2.1.

Como será demonstrado na seção 5.2.3, a parcela de consumo atribuída a cada componente é, muitas vezes, da ordem de microampéres ($1\mu A = 10^{-6}A$). Além disso, a contagem de tempo e conseqüente incrementação/decrementação de valores será feita com resolução de milissegundos. Como a capacidade nominal costuma ser referida em miliampéres ($10^{-3}A$) com relação a horas, mostra-se necessária uma conformação de ordens de grandeza desses valores. Essa adaptação também é realizada através de *Traits*. No momento em que o compilador necessitar do valor da capacidade nominal inserido pelo usuário, este será convertido de *mAh* para μAms , através de cálculos realizados internamente ao mecanismo *Traits*. O algoritmo 1 demonstra essa adaptação.

Outra definição importante para o nosso sistema realizada nesse nível é a tensão de operação do sistema. Esse valor - que em realidade, não é literal - é utilizado posteriormente como índice da tabela de correntes de cada um dos componentes, em conjunto com o estado de economia do mesmo que também possuem parâmetros definidos nesse âmbito, que já eram utilizados pelo aspecto gerente de energia. O algoritmo 2 demonstra como se dá essa definição.

5.2.3 Atribuição do Consumo a Componentes

Outra etapa importante do monitoramento é o processo de atribuição da quantidade de energia consumida por um componente do sistema em um período de tempo. Na implementação do sistema operacional EPOS, essa tarefa é facilitada em decorrência da modelagem baseada em componentes e pela existência de aspectos de contexto. A estrutura de aspectos nos dá a capaci-

Algoritmo 1 Solicitando utilização do monitor e adaptando valor de capacidade

```
1  template <class Imp>
2  struct Traits
3  {
4  (... )
5  static const bool consumption_monitoring = true;
6  };

7  template <> struct Traits<ATMega128_Battery>:
8  public Traits<void>
9  {
10 static const unsigned long long  MA_HR = 3000;
11 static const
12 unsigned long long  CONVERTED_UA_MS =
13 3600000 * 1000 * MA_HR;
14 };
```

Algoritmo 2 Configurando a tensão de operação

```
template <> struct Traits<AVR8>: public Traits<void>
1 {
2 (... )
3 static const char _3_0V = 0;
4 static const char _3_5V = 1;
5 static const char _4_0V = 2;
6 static const char _4_5V = 3;
7 static const char _5_0V = 4;
8 static const char _5_5V = 5;
9 };

10 template <> struct Traits<ATMega128>:
11 public Traits<ATMega128_Common>
12 {
13 (... )
14 static const unsigned int OPERATING_TENSION = Traits<AVR8>::_3_0V;
15 };
```

dade de atribuir aos componentes características decorrentes da necessidade apresentada por um determinado contexto de utilização, como por exemplo, a nossa necessidade de monitoramento do consumo (esquema apresentado na figura 3.4).

Cada componente de hardware de um dispositivo possui suas próprias características de consumo de energia. Além disso, muitas vezes essas características são configuráveis em tempo de execução, através dos estados de economia providos pela plataforma. O EPOS já possuía implementado previamente ao nosso trabalho um modelo de aspecto denominado *Power_Manager*, através do qual o sistema permite ao usuário a configuração dos parâmetros relacionados à gerência de energia.

Para a arquitetura escolhida em nosso experimento, o referido aspecto do EPOS possui a capacidade de atribuir novas características necessárias à gerência de energia aos seguintes mediadores de hardware do sistema: *CPU*, Conversor Analógico Digital (*ADC*) e *USART (Universal Synchronous and Assynchronous serial Receiver and Transmitter*⁵). Para que nosso trabalho seja incluído nesse sistema operacional respeitando os seus paradigmas de modelagem, é necessária que a sua construção seja feita através de um aspecto que defina e implemente necessidades do monitor. Como o monitoramento do consumo está intimamente ligado ao gerenciamento da energia do sistema, nos utilizamos do aspecto referido para a nossa implementação, apenas incluindo nossas novas necessidades e atributos. Consequentemente, nessa primeira implementação nosso monitor acompanha o consumo apenas dos três componentes de *hardware* citados.

Definir que um componente do sistema consumiu uma quantidade de x *mAh* da capacidade da bateria, significa dizer que este componente drenou uma corrente de y miliampéres da bateria por z horas, de forma que $x = y * z$. Com isso surge a necessidade de conhecermos a necessidade de corrente que cada um dos componentes monitorados possui para funcionar. Esses valores são diferentes para cada estado de economia nos quais esses componentes se encontram, variando também de acordo com a tensão que alimenta o dispositivo. Dessas características surge, portanto, a necessidade de construção de uma tabela tensão x estado de economia que indique qual é a corrente drenada por um componente em cada um dos casos.

Como em nosso experimento tratamos de um dispositivo alimentado por baterias, podemos estipular para ele uma tensão média de operação, sendo, com isso, necessário que se defina somente a corrente drenada para cada estado de economia quando o sistema opera nessa tensão.⁶. Nas tabelas 5.2, 5.3 e 5.4 podemos encontrar os valores definidos para a tensão média em todos

⁵Transmissor e Receptor serial Síncrono e Assíncrono Universal

⁶A tensão de operação do MICA2 alimentado por bateria varia entre 2.7 e 3.6 volts (22).

os estados de economia da CPU, ADC e USART, respectivamente.

Tabela 5.2: Consumo de corrente da CPU para seus diferentes estados de economia.

Estado de Economia	Corrente (μA)
Full	7600
Idle	3300
ADC Noise	1000
Power Down	116
Power Save	124
Standby	237
Ext Standby	243

Fonte: Landsiedel, Wehrle e Gotz(24)

Tabela 5.3: Consumo de corrente do ADC para seus diferentes estados de economia.

Estado de Economia	Corrente (μA)
Full	90
Off	0

Fonte: ATMEL Corp.(21)

Tabela 5.4: Consumo de corrente da USART para seus diferentes estados de economia.

Estado de Economia	Corrente (μA)
Full	1000
Send Only	5000
Receive Only	5000
Off	0

Fonte: ATMEL Corp.(21)

A atribuição de novas características aos componentes através de aspectos é feita utilizando-se de técnicas de *metaprogramação estática*, que permitem a inclusão de atributos novos aos componentes através de parâmetros metaprogramados. Quando um usuário do sistema operacional instancia uma entidade de algum componente que possui esse aspecto ativado, esse componente é automaticamente instanciado como sendo uma sub-classe do dito aspecto, o que lhe faz herdar todas as características pertencentes a este último. Esse foi o procedimento utilizado para a inclusão dessas tabelas no aspecto *Power_Manager*. O algoritmo 3 refere-se à implementação das tabelas de correntes drenadas pela CPU e ADC.

Algoritmo 3 Definindo tabelas de corrente

```
1  template<> const unsigned int
2    Power_Manager<AVR8,false,true,
3      true,true>::consumption_table [7][1] =
4    /*[power_mode][tension]*/
5    /*{3.0V, 3.5V, 4.0V, 4.5V, 5.0V, 5.5V}*/
6
7  {
8    {7600 /*,0,0,0,0,0*/},//FULL
9    {3300 /*,0,0,0,0,0*/},//IDLE
10   {1000 /*,0,0,0,0,0*/},//ADC_NOISE_REDUCTION
11   {116 /*,0,0,0,0,0*/}, //POWER_DOWN
12   {124 /*,0,0,0,0,0*/}, //POWER_SAVE
13   {237 /*,0,0,0,0,0*/}, //NATIVE_STANDBY
14   {243 /*,0,0,0,0,0*/} } //EXTENDED_STANDBY
15
16 template<> const unsigned int
17   Power_Manager<ATMega128_ADC,false,
18     true,true,true>::consumption_table [2][1] =
19   /*[power_mode][tension]*/
20   /*{ 3.0V, 3.5V, 4.0V, 4.5V, 5.0V, 5.5V}*/
21   {
22     {90 /*,0,0,0,0,0*/},//FULL
23     {0 /*,0,0,0,0,0*/} //OFF
24   };
```

Como pode ser observado na implementação dessas tabelas, ambas são denominadas *consumption_table*, e a definição por qual deve ser utilizada em cada circunstância é feita em tempo de compilação, através da resolução do *template Power_Manager*, implementador da metaprogramação, diferenciado pelo tipo a ser instanciado - nesse caso, *AVR8* ou *ATMega128_ADC*. Os números comentados representam os valores para as outras tensões de operação possíveis, que não são necessários para o funcionamento de nosso sistema atualmente.

Possuindo o valor de corrente necessário para o cálculo de consumo de cada componente, ficam faltando apenas as informações a respeito do tempo pelo qual o componente esteve operando em cada modo de economia. O próprio EPOS já possui implementada uma abstração de *hardware* que permite acesso ao contador de tempo do sistema. Na inicialização de cada um dos componentes monitorados, é realizada uma leitura do dito valor de tempo (cedido em milissegundos) que é então definido como o valor inicial da contagem de tempo de vida do componente. Esta é, portanto, mais uma variável definida no aspecto gerente de energia.

Algoritmo 4 Inicializando os Componentes

```
1  template<> unsigned long
2    Power_Manager<ATMega128_UART, false,
3      true, true, true>::_last_pow_change;

4  template<>
5    void Power_Manager<ATMega128_UART, Traits<AVR8>::_shared,
6      Traits<AVR8>::_instances,
7      Traits<Active_Power_Manager>::_enabled,
8        true>::_init_component()
9  {
10     last_pow_change(TSC::time_stamp());
11 }
```

O algoritmo 4 demonstra a inicialização da variável a ser usada para armazenamento das parcelas de tempo (variável *_last_pow_change*, linha 3) na instanciação de um componente da classe *ATMega128_UART* e define como é feita a sua inicialização.

Em posse dos valores de corrente dos componentes e de um período inicial de contagem do tempo, o cálculo de energia consumida pode ser feito em qualquer momento desejado. A nossa implementação realiza a atualização dos valores consumidos por cada componente nos momentos de transição entre os estados de economia. Para tanto, antes de realizar a transição de estado do componente, verifica-se o tempo passado entre o momento atual e o tempo atribuído à variável *last_pow_change*. O valor obtido Δt é então multiplicado pela corrente selecionada na tabela do componente, e somado ao montante já consumido pelo mesmo. O algoritmo 5

demonstra todo esse procedimento para o componente *CPU*. Detalhes a respeito da ordem de grandeza dos valores envolvidos serão tratados posteriormente.

Algoritmo 5 Atualizando o consumo dos componentes

```
1  template<>
2  void Power_Manager<AVR8,Traits<AVR8>::shared,
3     Traits<AVR8>::instances,Traits<Active_Power_Manager>::enabled,
4     true>::consumption_update()
5  {
6     unsigned long now = TSC::time_stamp();
7     unsigned long t_delta = now - last_pow_change();
8     unsigned long energy = t_delta *
9         consumption_table[_op_mode][Traits<Machine>::OPERATING_TENSION];
10    consumed_energy_increment(&energy);
11    last_pow_change(now);
12    Battery::remaining_charge_decrement(&energy);
13 }
```

5.2.4 Atualização da Capacidade da Bateria

Para respeitar as diretivas de um sistema baseado em componentes, foi incluído no EPOS um mediador de *hardware* especialmente para tratar das baterias. Esse mediador é responsável por oferecer ao sistema uma interface para acesso e manipulação das características desse componente. Apesar de o sistema possuir um mediador de *hardware* para o ADC, não é possível a sua utilização para a amostragem da tensão da bateria, pois isso causa uma referência circular insolúvel no sistema, por conta desta ser uma subclasse do próprio monitor que a instancia e que necessitaria utiliza-la. Com isso, se tornou necessário que o mediador da bateria replicasse as funcionalidades contidas no mediador do ADC, com vistas a obter independência desse componente. Além disso, ele também possui os atributos responsáveis pelo monitoramento da capacidade da bateria.

Na inicialização do sistema é necessário atribuir à bateria seu valor inicial de capacidade. É nesse momento que se utilizam os dados adquiridos através do experimento que verificou o comportamento de sua descarga. Para atribuir esse primeiro valor, é feita uma amostragem de sua tensão, e verifica-se a época à qual essa tensão pertence. Como cada uma das épocas representa uma parcela de 10% da capacidade da bateria, ao definir-se a qual época pertence a tensão amostrada, fixa-se o valor da bateria como sendo exatamente o meio dessa época. Através dessa técnica, faz-se com que o maior erro possível nessa definição inicial seja de 5%, para mais ou para menos.

A partir daí, os decrementos necessários ao monitoramento são realizados conjuntamente aos incrementos à energia consumida por cada um dos componentes, como pode ser verificado na linha 12 do algoritmo 5.

5.2.5 Desempenho do monitor

Para avaliar os resultados apresentados pelo monitor, fizemos um acompanhamento da execução do sistema. Implementamos uma aplicação de testes que realiza uma utilização intensiva dos recursos monitorados, e comparamos a descarga da bateria anunciada pelo monitor com dados obtidos a partir de um acompanhamento simultâneo feito a partir de um osciloscópio digital.

A aplicação executada consiste em um laço de execução que utiliza os componentes do sistema monitorados e realiza transições de estados de economia dos mesmos. Antes de iniciar a execução do laço, é verificado o estado em que se encontra a capacidade da bateria e o contador de tempo do sistema, e realizam-se novamente essas verificações ao fim de sua execução, para que se possa conhecer a quantidade decrementada de sua capacidade e o tempo transcorrido na execução do laço. O osciloscópio utilizado para acompanhar a execução dos testes foi o modelo Tektronix TDS5034B (25). Para a realização das medições foi utilizada a ponteira de prova de corrente Tektronix TCP 202, que permite ao osciloscópio realizar medições de corrente do sistema sem interferência física no mesmo através da exploração do Efeito Hall em um segmento de fio do dispositivo. As medições realizadas pelo osciloscópio possuíam um intervalo entre si de $2\mu s$ e foram agrupadas para efeito de cálculos do osciloscópio em intervalos de $1ms$, de forma a manter uma compatibilidade entre a sua amplitude e a do contador do dispositivo.

Ao executar a aplicação, estabelece-se através do osciloscópio uma média de drenagem de corrente do sistema para cada milissegundo percorrido com uma resolução bastante confiável. Em posse desse número, ao qual nos referiremos como I_{ms} , e do intervalo de tempo transcorrido entre o início e o fim da execução do teste (Δt), é possível calcular a corrente drenada pelo dispositivo com uma simples multiplicação:

$$I_{real} = I_{ms} * \Delta t$$

A partir disso, é possível estabelecer uma comparação entre a capacidade decrementada da bateria pelo nosso monitor (I_{mon}) e o valor obtido em I_{real} , estabelecendo o erro inerente ao monitor. A tabela 5.5 demonstra os valores obtidos em três desses testes realizados.

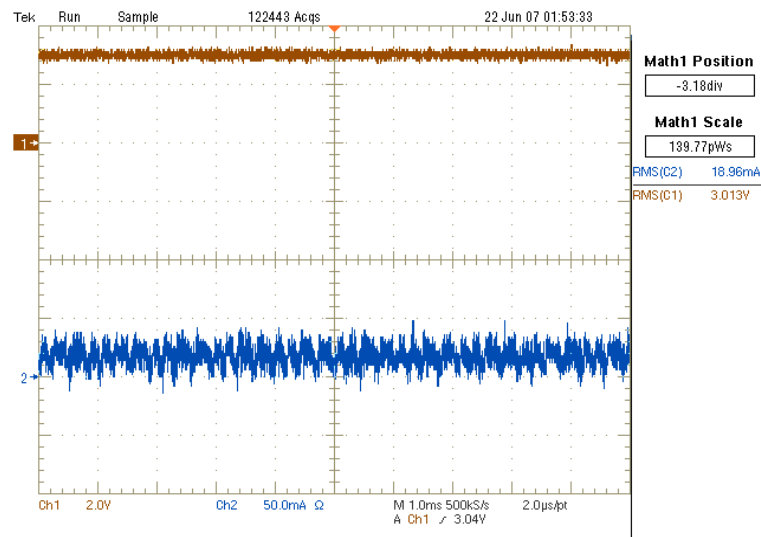


Figura 5.5: Captura de um estado do procedimento de medição no osciloscópio

Tabela 5.5: Valores obtidos nos testes.

Δt	I_{ms}	$I_{real}(I_{ms} * \Delta t)$	I_{mon}	% Erro
493179 ms	20.5 mA	10110169 mAms	8685132 mAms	16.4%
1197148 ms	15 mA	17957220 mAms	22075523 mAms	22.9%
493179 ms	18.85mA	9296424 mAms	8685135mAms	7%

Como pode ser verificado nos resultados apresentados, o monitor acompanha a descarga real, representada pelas medidas do osciloscópio, com um erro que varia entre 23% e 7%. Sabemos que muitos fatores interferem para que o valor assinalado no monitor não corresponda ao valor real. Os principais deles são a taxa de descarga, que possui forte influência na capacidade da bateria, a imprecisão nos cálculos realizados, por conta de um erro da faixa de 10% da contagem de tempo realizada pelo processador atmega128, a diferença de comportamento de baterias entre fabricantes diferentes, a corrente dissipada estaticamente pelo dispositivo (*leak*) e demais características de menor influência inerentes ao meio.

6 *Conclusão*

Este trabalho apresentou e conceitualizou o ambiente computacional de sistemas embarcados, colocando as suas principais características aplicabilidades, bem como sua inserção e importância no meio computacional. Foram relatadas brevemente iniciativas de estudos similares a nossa proposta, que levam em vista a idéia de gerenciamento de energia baseada em um acompanhamento das capacidades da bateria que alimenta esse sistema. Explicitamos os aspectos relevantes para a implementação de um sistema operacional para aplicações embarcadas, e nesse contexto foram colocados em pauta os conceitos de programação de sistemas operacionais baseados em componentes orientados à aplicação, e o paradigma de programação orientada a aspectos. Posteriormente foi apresentado o sistema operacional EPOS, que foi utilizado como base para o trabalho proposto. Foram trazidas as necessidades de gerenciamento de energia para o ambiente de sistemas embarcados e, nesse ponto, abordamos características importantes das baterias, com vistas a conceitualiza-las através de suas características mais importantes, tratando também das que são necessárias serem levadas em consideração no momento de monitorar seu comportamento.

Foi trazida como solução para a idéia apresentada nesse trabalho a inclusão de mecanismos no sistema operacional EPOS de modo a este adquirir características que lhe permitam o monitoramento da capacidade de uma bateria que alimenta o dispositivo gerenciado pelo dito sistema operacional. Tais características foram introduzidas no sistema sob a forma de parâmetros modificáveis e configuráveis através de um aspecto de contexto previamente existente no sistema, que introduzem funcionalidades e informações aos componentes do sistema que possuam ligação com esse aspecto. Apresentamos as abstrações de hardware incluídas no sistema com vistas a incluir um interfaceamento com a bateria. Tratou-se também de uma interpretação dos resultados obtidos na utilização do monitor, efetuando-se uma comparação com o comportamento real através de mecanismos externos ao sistema.

Trabalhos Futuros

A partir da implementação e validação da estrutura de monitoramento apresentada nesse trabalho, permite-se uma série de novas iniciativas que visem o englobamento de novas características e funcionalidades, bem como a consideração de novas variáveis inerentes ao meio de operação do modelo que permitam o seu refinamento e maior fidelidade à evolução do comportamento da bateria monitorada. A implementação de mais abstrações ligadas ao aspecto gerente de energia é um desses trabalhos. A partir disso, o sistema conquistará a capacidade de monitorar o uso de uma maior quantidade de periféricos, aumentando sua aplicabilidade. Os principais componentes que necessitam de interligação com os aspectos de gerência de energia são os *LEDS (Light Emitting Diode)* e o dispositivo rádio-transmissor. Várias iniciativas podem ser tomadas com a intenção de atribuir ao sistema a capacidade de efetuar não só o monitoramento, mas também tarefas de economia de energia. Uma das alternativas nesse sentido é atribuir ao sistema a capacidade de monitorar a taxa de solicitação de carga da bateria, de forma a torna-la o mais linear possível, uma vez que uma curva de descarga contínua mais baixa que altas solicitações intermitentes, segundo a Lei de Peukert, aumenta o tempo de vida da bateria. Todas essas informações podem constituir a implementação de um escalonador que conheça detalhes tanto da bateria quanto das características da aplicação a ser escalonada, de forma a tentar uniformizar essas taxas de descarga. Outra iniciativa interessante é a extensão do modelo visando torna-lo configurável para operar com mais de um tipo de bateria. Nosso modelo inicial opera com baterias alcalinas que, apesar de representarem uma grande parcela dos tipos de baterias utilizadas em sistemas embarcados, não representa a sua totalidade.

Referências Bibliográficas

- 1 HEWLETT-PACKARD Corp.; INTEL Corp.; MICROSOFT Corp.; PHOENIX Technologies Ltd.; TOSHIBA Corp. **Advanced Configuration and Power Interface (ACPI) Specification**. 3.0b. ed. [S.l.], Outubro 2006.
- 2 PARK, S.; SRIVASTAVA, M. B. Dynamic battery state aware approaches for improving battery utilization. In: **CASES '02: Proceedings of the 2002 international conference on Compilers, architecture, and synthesis for embedded systems**. New York, NY, USA: ACM Press, 2002. p. 225–231. ISBN 1-58113-575-0.
- 3 BENINI, L. et al. Discharge current steering for battery lifetime optimization. **Computers, IEEE Transactions on**, v. 52, n. 8, p. 985–995, Agosto 2003.
- 4 BENINI, L. et al. Extending lifetime of portable systems by battery scheduling. In: **Design, Automation and Test in Europe, 2001. Conference and Exhibition 2001. Proceedings**. [S.l.: s.n.], 2001. p. 197–201.
- 5 BENINI, L. et al. Battery-driven dynamic power management. **Design & Test of Computers, IEEE**, v. 18, n. 2, p. 53–60, Março-Abril 2001.
- 6 TENNENHOUSE, D. Proactive computing. **Commun. ACM**, ACM Press, New York, NY, USA, v. 43, n. 5, p. 43–50, 2000. ISSN 0001-0782.
- 7 MARWEDEL, P. **Embedded Systems Design**. 3300 AA Dordrecht, Holanda: Kluwer Academic Publishers, 2003.
- 8 GSG - Australia Motorola. **Requirements Management for Embedded Systems**. 2007. URL: <http://undergraduate.csse.uwa.edu.au/units/CITS3220/>.
- 9 HEATH, S. **Embedded Systems Design**. 2a. ed. Grã-Bretanha: EDN, 2003.
- 10 FRÖLICH, A. A. M. **Application-Oriented Operating Systems**. 200 p. Tese (Doutorado em Computação) — Institut für Rechnerarchitektur und Softwaretechnik FIRST, Berlim, Alemanha, 2001.
- 11 PARK, S. I. **The Design of Power Aware Embedded Systems**. 170 p. Tese (Doutorado em Engenharia Elétrica) — University of California, Los Angeles, Estados Unidos, 2003.
- 12 STEINKE, S. et al. **An Accurate and Fine Grain Instruction-Level Energy Model supporting Software Optimizations**. 2001. In: International Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS), Setembro 2001. Disponível em: citeseer.ist.psu.edu/steinke01accurate.html.
- 13 HOELLER, A. S.; WANNER, L. F.; FRÖLICH, A. A. M. **A Hierarchical Approach for Power Management on Mobile Embedded Systems**. Florianópolis - SC, Brasil, 2005.

- 14 SIMUNIC, T. et al. Event-driven power management. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, v. 20, n. 7, p. 840–857, Julho 2001.
- 15 LINDEN, D. **Handbook of Batteries**. 3. ed. Nova Iorque, EUA: McGRAW-HILL INC., 1995.
- 16 POWERS, R. Batteries for low power electronics. **Proceedings of the IEEE**, v. 83, n. 4, p. 687–693, Abril 1995.
- 17 PANASONIC Ind. Comp. **Panasonic Industrial AA Alkaline Battery Specification**. Secaucus, New Jersey, EUA, Setembro 2005. Disponível em: <<http://www.panasonic.com/industrial/battery/oem/>>.
- 18 BELLOSA, F. The benefits of event: driven energy accounting in power-sensitive systems. In: **EW 9: Proceedings of the 9th workshop on ACM SIGOPS European workshop**. New York, NY, USA: ACM Press, 2000. p. 37–42. ISBN 1-23456-789-0.
- 19 ZENG, H. et al. Ecosystem: managing energy as a first class operating system resource. **SIGPLAN Not.**, ACM Press, New York, NY, USA, v. 37, n. 10, p. 123–132, 2002. ISSN 0362-1340.
- 20 PILLAI, P.; HUANG, H.; SHIN, K. G. **Energy-Aware Quality of Service Adaptation**. Michigan, EUA, 2001.
- 21 ATMEL Corp. **Atmel 8-Bit AVR Microcontroller Datasheet**. San Jose, Califórnia, EUA, Outubro 2006. Revisão 2467. Disponível em: <www.atmel.com>.
- 22 CROSSBOW Technology Inc. **MPR-MIB Users Manual**. San Jose, Califórnia, EUA, Junho 2006. Revisão B. Disponível em: <www.xbow.com>.
- 23 CROSSBOW Technology Inc. **MICA2 AA Battery Pack Service Life Test**. San Jose, Califórnia, EUA, 2004. Disponível em: <www.xbow.com>.
- 24 LANDSIEDEL, O.; WEHRLE, K.; GOTZ, S. Accurate prediction of power consumption in sensor networks. In: **Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on**. [S.l.: s.n.], 2005. p. 37–44.
- 25 TEKTRONIX, I. **TDS500B Series Digital Phosphor Oscilloscopes 071-1355-02**. Beaverton, OR.