UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

GUILHERME HENRIQUE EVANGELISTA DE ANDRADE

DESENVOLVIMETO DE UM SISTEMA DE MONITORAMENTO ENERGÉTICO CONTROLADO PELA INTERNET

TRABALHO DE CONCLUSÃO DE CURSO 2

PATO BRANCO
2016

GUILHERME HENRIQUE EVANGELISTA DE ANDRADE

DESENVOLVIMENTO DE UM SISTEMA DE MONITORAMENTO ENERGÉTICO CONTROLADO PELA INTERNET

Trabalho de Conclusão de Curso como requisito parcial à obtenção do título de Bacharel em Engenharia de Computação, do Departamento Acadêmico de Informática da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Fábio Luiz Bertotti
Co-orientador: Prof. Dr. Gustavo Denardin

PATO BRANCO
2016

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Engenharia de Computação

TERMO DE APROVAÇÃO

Às 10h horas e 20 minutos do dia 13 de dezembro de 2016, na sala V105, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, reuniu-se a banca examinadora composta pelos professores Fabio Luiz Bertotti (orientador), Gustavo Weber Denardin (coorientador), Cesar Rafael Claure Torrico e Robison Cris Brito para avaliar o trabalho de conclusão de curso com o título **Desenvolvimento de um sistema de monitoramento energético controlado pela internet**, do aluno **Guilherme Henrique Evangelista de Andrade**, matrícula 01147102, do curso de Engenharia de Computação. Após a apresentação o candidato foi arguido pela banca examinadora. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

Fabio Luiz Bertotti Orientador (UTFPR)	Gustavo Weber Denardin Coorientador(UTFPR)
Cesar Rafael Claure Torrico (UTFPR)	Robison Cris Brito (UTFPR)
Beatriz Terezinha Borsoi Coordenador de TCC	Pablo Gauterio Cavalcanti Coordenador do Curso de Engenharia de Computação

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

EVANGELISTA DE ANDRADE, Guilherme Henrique. Desenvolvimento de um sistema de monitoramento energético controlado pela internet. 2016. 112p. Trabalho de Conclusão de Curso de bacharelado em Engenharia de Computação - Universidade Tecnológica Federal do Paraná. Pato Branco, 2016.

A internet revolucionou a sociedade e o modo como as pessoas se relacionam com a informação e objetos. Em paralelo, a dependência por energia elétrica demanda meios de visualizar e controlar os equipamentos que consomem essa energia. Desse modo, é desenvolvido um sistema domótico que fornece índices de energia, além da capacidade de armazenar os dados coletados em um banco de dados e o controle de uma rede de unidades sensoras pela internet.
Por isso, foi proposto o desenvolvimento de um sistema de monitoramento de consumo energético e acionamento de dispositivos elétricos, através de uma unidade sensora que realize as medições e um dispositivo central que recolha e envie a informação pela internet, fornecendo para o usuário acesso desses dados através de um aplicativo.
Embora o sistema tenha encontrado sinergia, com o aplicativo comandando a unidade central, que se comunicou bem com a unidade sensora e esta adquiriu os sinais de temperatura e corrente, alguns problemas na comunicação sem fio e na aquisição dos sinais fizeram deste trabalho um grande desafio.

Palavras-chave: IoT, Controle, Energia, Sistema embarcado, Domótica, Linux

ABSTRACT

EVANGELISTA DE ANDRADE, Guilherme Henrique. Development of an energy monitoring system controlled by internet. 2016. 112p. Computer Engineering - Federal University of Technology - Paraná (UTFPR). Pato Branco, 2016.

The internet has revolutionized society and the way people relate to the information and objects. In parallel, the electric energy dependence needs ways of visualizing and controlling the equipment that consume this energy. Therefore, a domotic system is developed to provide energy indexes, also the capacity to store the data collected in a database and control a sensor network through the Internet.

Keywords: IoT, Control, Energy, Embedded System, Domotic, Linux

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE GRÁFICOS

LISTA DE SIGLAS

ANEEL	Agência Nacional de Energia Elétrica
API	*Application Programming Interface*, Interface de Programação de Aplicativos
BCS	*Basic Clock System*, Sistema de Clock Principal
BEN	Balanço Energético Nacional
CI	Circuito Integrado
DHCP	*Dynamic Host Configuration Protocol*, Protocolo de Configuração Dinâmica de Host
EPE	Empresa de Pesquisa Energética
E/S	Entrada e Saída
FPU	Unidade de Ponto Flutuante
FSM	*Finite State Machine*, Máquina de estados finitos
IA	Inteligência Artificial
IDE	*Integrated development environment*, Ambiente de desenvolvimento integrado
JDBC	*Java Database Connectivity*, Conexão Java com Banco de dados
MME	Ministério de Minas e Energia
M2M	*Machine-to-Machine*, Máquina-para-Máquina
OSL	One Single Loop
PNBL	Programa Nacional de Banda Larga
RTI	Retorno de Interrupção
RTOS	*Real Time Operating System*, Sistema Operacional de tempo real
SO	Sistema Operacional
SoC	*System on a chip*, Sistema em um chip
SQL	*Structured Query Language*, Linguagem de Consulta Estruturada
TIC	Tecnologias de Informação e Comunicação

SUMÁRIO

1 INTRODUÇÃO

Durante o século passado a matriz energética, obtida principalmente de recursos fósseis, deu impulso ao grande desenvolvimento da economia em todo o mundo. Para que este desenvolvimento econômico continue em ritmo acelerado, seria necessário que a oferta de energia acompanhasse o aumento do consumo desde então, mas isto se mostrou impraticável. A estrutura do consumo por fontes energéticas constitui um dos aspectos-chave para analisar os desafios que a sociedade enfrentará no futuro, seja ele de curto ou longo prazo. A expansão do consumo de energia, maior que o crescimento do PIB em 2013, indica a melhora na qualidade de vida das pessoas no país, embora apresente aspectos indesejados como esgotamento dos recursos finitos e danos à natureza.

O *status quo* da matriz energética mundial é ainda dominada pelas fontes naturais. Em 2011, segundo a Agência Internacional de Energia e a Secretaria Técnica da Repsol, 81% da energia utilizada no mundo foi de origem não-renovável (COMPANHIA ENERGÉTICA…, 2015). Essa dependência no consumo de energia é alarmante, uma vez que fontes de energia não renováveis têm recursos teoricamente limitados, dependendo dos recursos existentes no planeta (PORTAL ENERGIA…, 2015). Por esse motivo, o tema sobre eficiência energética e consumo inteligente é tratado como assunto *sine qua non* pelas autoridades internacionais (UNITED NATIONS…, 2015).

No contexto da energia elétrica, uma vez que a distribuição de energia pelas concessionárias é custosa, a conscientização dos consumidores sobre seus gastos e o maior controle dos dispositivos que usufruem da energia pode ser um grande aliado na economia destes recursos. A tendência pelo consumo eficiente é mostrada quando se analisam os índices de consumo dos países mais desenvolvidos que, historicamente, são maiores consumidores, mas com uma redução contínua nesses indicadores, enquanto que em países em desenvolvimento há o movimento oposto.

Em paralelo, o uso da rede mundial de computadores no Brasil vem aumentando, principalmente no setor móvel, como afirma o Ministro das Comunicações, afirmando que “a proliferação da internet no país vai ser pela via móvel” (MINISTÉRIO…, 2014a, p.1). Com a internet associada a dispositivos reais, o conceito de *Internet of Things,* IoT, é usado como uma ligação entre o mundo físico e virtual dos objetos (HU et al, 2014, p.1). O intuito é oferecer mais controle ao ser humano sobre a interação entre este e os dispositivos que fazem parte de seu cotidiano, além de aproximá-lo da tecnologia para que esta se torne

mais intuitiva, tal como as tecnologias em desenvolvimento que estão cada vez mais próximas da linguagem natural.

A pesquisa e desenvolvimento de soluções no setor de energia se torna imprescindível no sentido de que o Brasil é um pais populoso com cerca de 62 milhões de unidades consumidoras em quase todos os municípios brasileiros - 85% no contexto residencial. De fato, o setor de energia elétrica é o serviço de infra-estrutura mais universalizado. A motivação do desenvolvimento de dispositivos que auxiliem no monitoramento e economia do consumo dentro de residências, comércio e até mesmo da indústria, está no panorama atual de dependência por fontes energéticas não-renováveis, no contexto de aumento do uso da internet e no barateamento dos meios tecnológicos que viabilizam soluções inteligentes.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver um sistema de monitoramento de consumo energético e acionamento de dispositivos elétricos, através de uma unidade sensora que realize as medições e um dispositivo central que recolha e envie a informação pela internet, fornecendo para o usuário acesso desses dados através de um aplicativo.

1.1.2 Objetivos Específicos

a) Controle de dispositivos/equipamentos pela internet;
b) Medição de energia elétrica;
c) Estudo sobre firmware;
d) Desenvolvimento de hardware e firmware para unidade sensora;
e) Desenvolvimento de hardware e software para unidade mestre.

1.2 JUSTIFICATIVA

Países em que o investimento é pesado em tecnologia, como a China, a *IoT* se tornou campo de pesquisa com investimento governamental (HARBOR…, 2014). É interessante notar que, além do farto campo de pesquisas, casas inteligentes se mostram como um tema de enorme potencial de investimento privado (HU et al, 2014, p.1). De fato, com desenvolvimento e barateamento contínuos das tecnologias (MOORE, 2014, p.1), estas se tornam cada vez mais presentes no cotidiano e formam um contexto interessante para inovações e soluções criativas. Sempre se buscou o estreitamento na relação do homem com os produtos os quais este interage, como eletrônicos, eletrodomésticos e automóveis, fazendo com que essas tecnologias sejam amigáveis aos sentidos e, consequentemente, facilitar seu uso ao ponto do que se considera natural. Essa facilidade de interação se traduz em comodidade e praticidade.

A facilidade de uso e interatividade são tendências tecnológicas que podem ser observadas, por exemplo, nas telas sensíveis ao toque, que dominaram o mercado de *gadgets* nos últimos 9 anos, extinguindo a necessidade de treinamento para utilização desses dispositivos, uma vez que se comanda um sistema com movimentos mais naturais

do que em teclados físicos. Outro exemplo são os dispositivos que captam o movimento por câmeras, como o Kinect da Microsoft, ou ainda os sistemas que atuam no reconhecimento de voz, alvo de intensa pesquisa de empresas de grande porte como Microsoft, Google e Apple, assim como instituições acadêmicas como o Massachusetts Institute of Technology, MIT, apostando nos comandos por fala como principal método de interface com o computador (MIT…, 2014). Ainda, pode-se citar o emprego de numerosas formas de tecnologia nos sistemas que comandam os automóveis, que se tornam cada vez mais inteligentes, como se observa nos construídos pela empresa Tesla Motors (TESLA…, 2014).

Os esforços para trazer a tecnologia mais perto das pessoas se estendem ao emprego de ideias para conectar os sistemas eletrônicos de um determinado ambiente aos seus usuários por meio de uma unidade de controle central, a domótica, como se observa em eletrodomésticos inteligentes como lavadoras, secadoras e geladeiras, em sistemas de iluminação, segurança e jardinagem, os quais são projetados para se conectarem à internet, trazendo o mundo analógico até a comodidade e facilidade do digital. Estes são conceitos e ideias discutidas há décadas, no entanto, apenas nos últimos anos se mostrou uma proposta interessante e viável. Um exemplo disto é o Edyn (KICKSTARTER, 2014), um sistema para automatização de jardins, que rega plantações automaticamente e monitora índices de umidade e temperatura do ambiente, sendo alimentado por energia solar, ou seja, atente ao desafio atual de consumir energia eficientemente (TEXAS…, 2014, p.1).

A sociedade se encontra na terceira geração da internet, possibilitando a comunicação entre pessoas, processos e objetos em rede. Esta revolução tecnológica é o foco de uma das maiores tendências para a área de Tecnologia da Informação e Telecomunicações (STARTUPI, 2015, p.1). Dentro das residências, a IoT trará mais conforto, automação e segurança, permitindo não só o controle mas também o monitoramento de indicadores de interesse geral, como gastos de energia e detecção de anormalidades na rede. Segundo a Texas Instruments, a IoT também fornece um estilo de vida melhor, reduz gastos e, no mundo dos negócios, vende mais produtos e novos serviços são desenvolvidos (TEXAS…, 2014, p.1).

1.3 ESTRUTURA DO TRABALHO

O desenvolvimento deste trabalho se distribuiu em quatro estágios principais. Primeiramente, fez-se um panorama geral do setor de energia no Brasil, mostrando como é estruturado e sua evolução de funcionamento e consumo nacional com o passar do tempo. Depois se apresentou a estrutura da internet no país, com dados referentes à número de usuários e contexto atual. Conceituam-se também as teorias que modelaram o projeto. Posteriormente, na segunda etapa, são dadas especificações técnicas sobre o hardware e software utilizados, bem como os motivos que levaram a estas escolhas. Na terceira etapa são mostrados como foram construídos os circuitos para aquisição de dados e atuadores, como foram configurados os dispositivos que constituem o sistema proposto e como foi desenvolvida a aplicação. Por último, são apresentados os resultados individuais e integrados de cada dispositivo e algumas conclusões são discutidas. A Figura 1 mostra os processos planejados.

2 REFERENCIAL TEÓRICO

2.1 CARACTERÍSTICAS GERAIS

A facilidade com que uma determinada população usufrui da infraestrutura de energia é um dos fatores básicos de determinação do desenvolvimento de um país, uma vez que esta energia é responsável por oferecer suporte à ação humana. Assim, pode-se dividir o setor em duas vertentes: a) a que visa o desenvolvimento da melhor qualidade e mais eficiente produção e consumo de energia e b) a que visa o aumento do acesso das pessoas à estas fontes.

A primeira, muitas vezes denominada ação vertical, inclui diversas pesquisas e investimento em novas tecnologias. A substituição gradual de parte da matriz energética por fontes renováveis, a mudança nos automóveis, que antes utilizavam apenas gasolina e agora são movidos também por etanol e eletricidade, ou ainda a própria distribuição de eletricidade são exemplos disto.

A segunda, ou ação horizontal, pode ser verificada no aumento continuo de oferta de energia elétrica no país, por meio de investimentos em infra-estrutura e programas sociais, como o “Luz para Todos” e o “Baixa Renda” do Ministério de Minas e Energia que, segundo pesquisa da EPE, Empresa de Pesquisa Energética, em 2008, aumentou o consumo da região Norte do país em 23% (ANEEL, 2008, p. 22).

Mesmo na década de 70 no Brasil, em que a cocção de alimentos, processo em que a comida sofre ação do calor, era realizada pela energia fornecida pela lenha, pôde-se notar os reflexos sociais e econômicos da substituição por derivados do petróleo, que resultou em um maior número de pessoas com acesso a produtos que, além de mais eficientes do ponto de vista energético, não necessitavam ter origem local.

Com relação ao cenário mundial, segundo o anuário estatístico de energia elétrica 2013 da EPE, em 2006 foram consumidos 16.391,5 TWh, e cerca de 18.466,5 TWh em 2010, com o Brasil em 9º lugar (EPE, 2013, p.53). Essa diferença representa aumento de cerca de 11% no consumo. Em contrapartida, a geração de energia elétrica foi de 18.005,1 TWh em 2006, e 20.225,3 TWh em 2010, aumento de 10,9%, ou seja, os aumentos de produção e consumo aumentaram na mesma proporção (EPE, 2013, p.22). As tarifas de energia elétrica nos setores Industrial e Residencial mostram o Brasil em, respectivamente, 12º com US$135,19 por MWh, e 26º com US$170,15 por MWh (EPE, 2013, p.55).

O funcionamento setor elétrico no país pode ser dividido em três etapas fundamentais. São elas a geração, transmissão e a distribuição de energia, como mostra a Figura 1.

Após deixar as usinas, a energia trafega pela rede de transmissão. Esta é composta de 106,444 mil quilômetros de linhas em 2012, divididas em 64 concessionárias, em tensões que podem chegar a 750kV. A grande extensão da rede de transmissão é explicada, primeiramente, à grande extensão territorial do país, mas também devido à grande dependência das fontes hidráulicas, que estão instaladas em locais distantes dos consumidores. Pode ser classificada em dois blocos: o Sistema Interligado Nacional (SIN) e os Sistemas Isolados, instalados principalmente na região Norte.

A energia fornecida pelo setor elétrico no país é oferecida à sociedade através das distribuidoras, que realizam a conexão, atendimento e entrega efetiva ao consumidor. Por meio das linhas de transmissão a energia chega às distribuidoras, onde a tensão é rebaixada e distribuída em valores de 127V ou 220V para os centros de carga. Este mercado é composto de outras 63 concessionárias, responsáveis pelo atendimento de mais de 72 milhões de unidades consumidoras (EPE, 2013, p.152) e desenvolvem programas que incentivam inclusão social, além de serem responsáveis pelo investimento em projetos de eficiência energética e P&D (Pesquisa e Desenvolvimento). Esse investimento é previsto em lei, destinando ao propósito um percentual mínimo de sua receita líquida. Entre 1998 e 2007, este valor foi de R$1,3 bilhão.

O destino final são os consumidores, que podem ser classificados em Cativos e Livres. O consumidor cativo absorve incertezas e erros e acertos do planejamento centralizado de governo e da distribuidora, ficando exposto a riscos. Já para o consumidor livre a energia é livremente negociada. O consumidor tem obrigação de comprovar 100% de contratação, após a medição do montante consumido. O valor de sua energia é resultante de sua opção individual de compra, que poderá incluir contratos de diferentes prazos e maior ou menor exposição ao preço de curto prazo. No mercado livre o consumidor é responsável por gerir incertezas e por seus erros e acertos na decisão de contratação. Assim, o consumidor livre toma para si a tarefa de gerir suas compras de energia e os riscos associados.

Na regulamentação das atividades e deveres dos agentes deste sistema entra a Agência Nacional de Energia Elétrica, ANEEL, promovendo controle para que o mercado de energia elétrica e desenvolva com equilíbrio, visando o bem da sociedade.

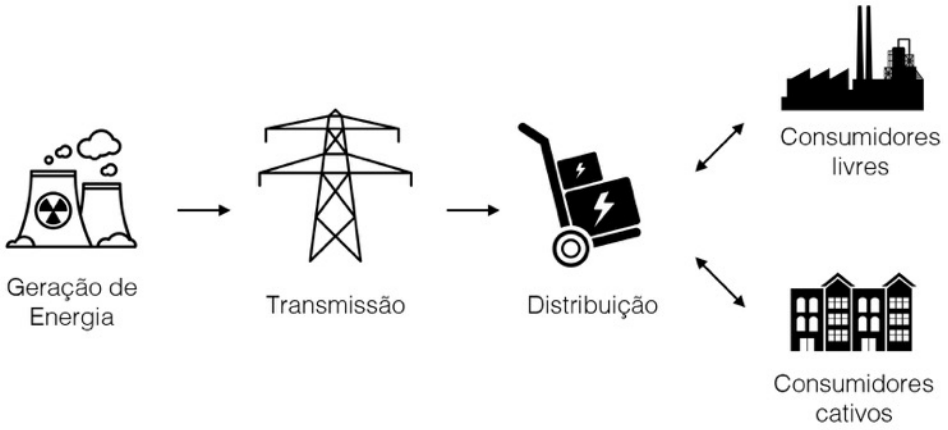


Figura 1 - Estrutura do setor elétrico
Fonte: ANEEL, 2008

O consumo de energia é um importante indicador de desenvolvimento em uma sociedade, refletindo a produção industrial e o consumo de bens e serviços pela população. No Brasil, segundo o BEN, fornecido pelo EPE em 2014, a pedido do Ministério de Minas e Energia, houve redução de 5,4% na oferta de energia elétrica devido a condições desfavoráveis (MINISTÉRIO…, 2014b, p.1). Em contrapartida, houve aumento no consumo das residências e serviços.

O racionamento ocorrido em 2001 provocou mudanças no panorama energético nacional. Ações que sugeriram mudanças para um comportamento mais responsável e práticas de eficiência energética, como utilização de lâmpadas econômicas no setor residencial, fizeram com que, em 2002, o consumo de energia elétrica verificado no país, segundo o BEN 2008, baixasse em níveis próximos aos verificados entre 1999 e 2000. Partindo desse ano, entretanto, houve crescimento nos níveis de consumo, segundo a Agência Nacional de Energia Elétrica no Brasil, ANEEL, em Atlas de Energia no Brasil – 6,5% em 2003; 5,2% em 2004; 4,2% em 2005 e 3,9% em 2006 - provocando preocupações das organizações responsáveis com relação à capacidade da oferta acompanhar esta evolução (ANEEL, 2008, p.45).

2.1.1 Consumo na Indústria, Comercio e Residências

De acordo com o BEN, em relatório realizado em 2014, no ano anterior a indústria foi responsável por 33,9% de consumo e o setor agropecuário por outros 4,1%, totalizando 38% da energia utilizada no país (MINISTÉRIO…, 2014b, p.20). Disso, a eletricidade possui a maior fatia de consumo, usando mais de 1/5 desse valor (MINISTÉRIO…, 2014b, p.23), como mostra o Gráfico 1. Este setor também se mostra como principal abrigo da tendência de autoprodução de energia, ou seja, investimentos realizados por consumidores de grande porte em usinas geradoras para suprimento próprio e venda do excedente em mercado. Em 15 anos, a variação acumulada foi de 262%.

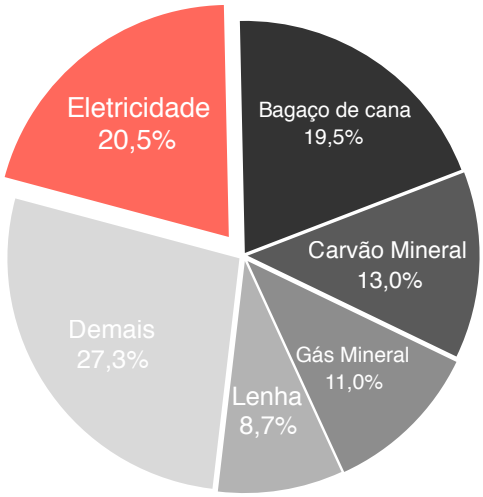


Gráfico 1 - Fontes de consumo na Indústria em 2013
Fonte: Ministério, 2014

No comércio, em matéria da Empresa Brasil de Comunicação S/A, o consumo de energia aumentou. Analisam a expansão do consumo de energia como reflexo na área de vendas, movimentação dos aeroportos e turismo (EMPRESA…, 2014,p.1). Os setores comerciais que mais consumiram em 2012 são encabeçados pelo Comércio varejista, Alimentação e Telecomunicações.

O setor residencial foi responsável, em 2013, pelo consumo de 9,1% do consumo de energia no país, com aumento de 6,2% em relação a 2012 (MINISTÉRIO…, 2014b, p.20). Em relação a energia elétrica, o setor foi responsável por 45,3% do consumo conforme o

Gráfico 2 (MINISTÉRIO..., 2014b, p.25). O motivo do aumento no consumo está no aquecimento do mercado experimentado nos últimos anos, que ocasionou a compra de novos equipamentos eletrodomésticos.

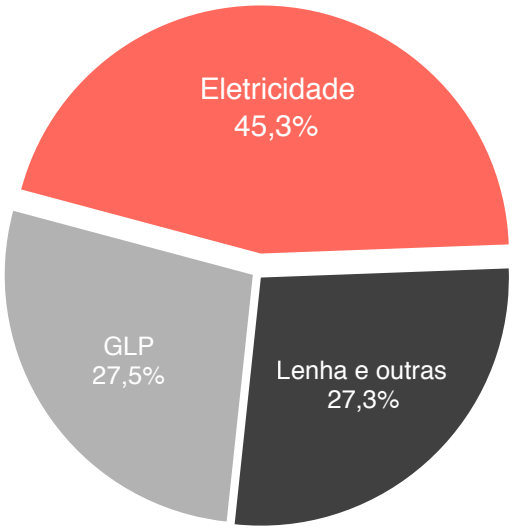


Gráfico 2 - Fontes de consumo nas residências em 2013

Fonte: Ministério, 2014.

2.2 EVOLUÇÃO DA INTERNET NO BRASIL

O acesso da sociedade às tecnologias de informação e comunicação, denominada TIC, é um fator importante no processo de desenvolvimento do conhecimento, crescimento econômico, geração de empregos e qualidade de vida. Isso mostra o motivo do investimento dos governos ao redor do mundo em políticas de democratização do acesso, com o objetivo de popularizar essa tecnologia e abrandar a fratura digital.

Acompanhando o aumento do consumo de energia, apontado anteriormente, está o aumento no número de pessoas conectadas à rede. A internet móvel cresceu 416% no Brasil de dezembro de 2010 a janeiro de 2014 (MINISTÉRIO..., 2014a, p.1), o que faz dela uma ferramenta útil na observação de padrões de consumo da população, mesmo no contexto de consumo de energia.

A CETIC.br, Centro de Estudos sobre as Tecnologias da Informação e da Comunicação, realiza anualmente a pesquisa TIC Domicílios, desde 2005, com o objetivo de mapear o acesso à infraestrutura TIC nas residências urbanas e rurais do país. O último relatório, em 2013, aponta para o crescimento constante da presença de computadores e internet nas casas, com cerca de 21 pontos percentuais em 4 anos. Também mostra que 40% dos domicílios brasileiros possuem internet em 2012, representando aumento de 22 pontos desde 2008, ou 24,3 milhões de residências (CETIC.br, 2013, p.159). O Gráfico 3 representa este crescimento.

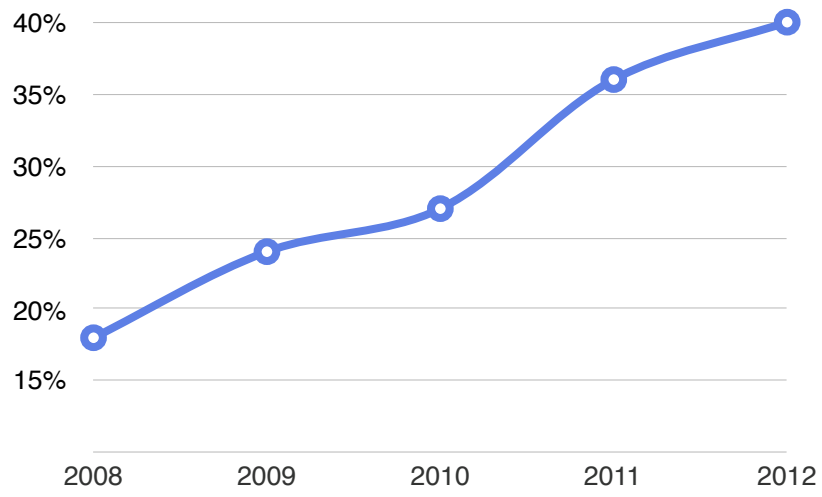


Gráfico 3 - Proporção de Residências com acesso à internet

Fonte: CETIC.br, 2013

Também houve aumento de velocidade da conexão, uma vez que 41% das casas possuíam conexões de 256kbps em 2008, caindo para 9% em 2012. Em contrapartida, as velocidades de 2Mbps, que eram de 6% em 2008, passam a ocupar 30% das residências em 2012. A pesquisa aponta para um movimento global de desenvolvimento de infraestrutura voltada a tecnologia, relacionado com a evolução dos aplicativos e conteúdos na internet, que exigem conexões mais rápidas.

Embora o crescimento seja acelerado, barreiras como preço dos meios de conexão à rede e as desigualdades sociais e econômicas limitam esse desenvolvimento, provocando grande discrepância entre as taxas de domicílios com acesso à internet na zonal rural e urbana; essa diferença chega aos 34% (CETIC.br…, 2013, p.31).

Não obstante a diferença nestas zonas, vê-se esse comportamento acentuado em diferentes regiões do país, com maior concentração de domicílios com acesso no Sudeste (48%), seguido pelo Sul (47%) e Centro-Oeste (39%). Apesar disso, houve aumento na quantidade de usuários e na frequência de uso da rede, com cerca de 90 milhões de brasileiros ativos em 2012. Também no setor empresarial, a internet é utilizada por 97% dos negócios, de pequeno ao grande porte. Isso deixa claro que qualquer mudança desejada em padrões da sociedade deverá envolver soluções que incluam a internet.

Resolvendo-se o problema do acesso e serviço universalizados, ainda existe a questão de que apenas o acesso a internet não é suficiente para promover serviços que demandam maior estrutura de rede, como o controle de processos ou o interfaceamento de objetos; para isso é preciso dispor de banda larga, que surge como necessidade na vida da população no século XXI.

Desse modo, é evidente a importância do estado na intervenção desse cenário, como aconteceu na União Européia, com proatividade de governos como França e Suécia na implantação de infraestrutra de banda larga, com grandes injeções de recursos. Essa interferência estatal surge no Brasil com o programa nacional de banda larga, PNBL, do Governo Federal, que visa massificar o acesso nas regiões mais carentes. Em 2014, a meta foi de 40 milhões de domicílios conectados.

O Ministério das Comunicações também incentiva o desenvolvimento com a desoneração de redes e fibra ótica, que resulta em renuncia de R$4bi, além da expansão da rede pública de fibra óptica e até mesmo o programa de desoneração de smartphones.

O último ponto que demostra a importância da rede no Brasil está no Marco civil da Internet, que objetiva garantir o futuro do fluxo desimpedido de ideias e opiniões. O Projeto de Lei nº 2.126/2011 estabelece princípios, garantias, direitos e deveres de quem utiliza, além de oferecer uma legislação para casos de disputas judiciais para quem consome e fornece serviço nesse meio. As opiniões a respeito são divergentes, pois enquanto uma parcela defende a manutenção de como a internet está, outros acreditam que a imparcialidade e democracia da rede está comprometida pelas práticas auto-regulamentarias do mercado. A constituição da internet, como também é denominado o projeto, possui três leis pilares fundamentais (CETIC.br, 2013, p.39):

a) Neutralidade de rede: garantia de que os dados sejam tratados de forma igualitária, sem qualquer discriminação em relação ao conteúdo, origem, destino ou ao tipo de serviço ou tecnologia.

b) Privacidade: Os provedores de conexão são proibidos de guardar registros de aplicações. Desse modo, passa a ser direito do usuário o não fornecimento a terceiros de seus registros de conexão e de acesso.

c) Liberdade de expressão: estipula que um *site* ou rede social só pode ser responsabilizado civilmente por qualquer dano causado por conteúdo postado por terceiros se, após ordem judicial, não retirar o conteúdo infringente.

2.3 INTERNET DAS COISAS

A IoT é uma estrutura que combina informação e processos energéticos para identificar e controlar diferentes objetos (BARI, 2013, p.48). Nesse contexto, os objetos possuem identidades na rede, ou seja, personalidades virtuais, com o propósito de comunicar e conectar. Pessoas, objetos e serviços, com essa ideia, estarão conectados por meio da internet, possibilitando novas maneiras de usar dispositivos e obter informação. É inquestionável o impacto que esse conceito possui no cotidiano dos potenciais usuários, tanto no campo doméstico como no trabalho, criando um novo paradigma de maior importância em um futuro próximo (ATZORI et al, 2010, p.1).

É fundamentada na Identificação por Radio Frequência, RFID e em pesquisas em nanotecnologia, ambos vistos como os principais potencializadores dessa ideia. Os avanços ao nível da miniaturização apontam o desenvolvimento para que cada vez mais pequenos objetos possuam a capacidade de interagir. Além do RFID, alguns modos de conexão são DSL, GPRS, WiFi, LAN e 3G. A IoT deve ser capaz de processar informação, configurar-se e reparar-se a si mesma, além de possuir autonomia para tomar suas próprias decisões (BARI, 2013, p.48).

Diferentemente da *Machine-to-Machine* (M2M), onde o foco é a comunicação entre máquinas, a IoT se preocupa com a comunicação entre máquinas, pessoas e tudo o mais que possa fornecer informação útil e controle de um determinado ambiente. Embora a ideia não seja nova, o desenvolvimento só ganhou impulso devido ao momento tecnológico atual, com a estrutura necessária mais desenvolvida, ou seja, energia elétrica disponível para a maioria da população, assim como internet, Wi-fi e dispositivos eletrônicos para acesso à rede mais acessíveis. Além disso, os microcontroladores, MCUs, e os dispositivos de comunicação de baixo consumo são mais baratos e dispõe de mais recursos.

Existem várias áreas que se beneficiam da IoT e as que estão em evidência são os *wearables*, como relógios e óculos inteligentes, vistos recentemente em produtos

introduzidos no mercado pela Apple, Google, Microsoft, Cisco entre outras (FORBES, 2014). O segmento residencial também está nesse caminho, com o controle de iluminação, temperatura e segurança. Cidades também aderem investindo na implementação do controle pela internet como modo facilitador de seus munícipes, como é o caso da Prefeitura de Curitiba e Pato Branco, que investem na implementação de transporte inteligente.

Pode-se dizer que a IoT é o próximo passo evolutivo da internet tradicional, oferecendo o potencial desenvolvimento de novos negócios, aplicativos e com grandes efeitos sociais. As empresas de diversos segmentos estão interessadas no assunto devido à:

- Economia de processos;
- Maximização de lucros;
- Sustentabilidade;
- Marketing;
- Integração com redes sociais (cadeia de valor)

Ainda, segundo a Intel, o grande foco no assunto hoje não está somente na construção de dispositivos e serviços que estão conectados à internet, mas também em sua interoperabilidade (INTEL, 2016, p1). Com a ajuda dessas tecnologias da IoT, as soluções de automação de ambientes estão presentes agora em empresas de grande e médio porte, e questões como aquecimento, ventilação e ar-condicionado são monitorados de forma mais eficiente e integrada do que nunca, e controlados usando sensores, inteligência artificial (IA) e atuadores também de forma integrada. Os padrões de consumo de energia podem agora ser analisados e comparados com outras residências; o sistema de segurança pode ser mais efetivo através da combinação de diversos dispositivos de monitoramento em tempo real; há uma infinidade de aplicações possíveis e lucrativas para o mercado. Embora exista o lado negativo, como a IoT ser utilizada como *spy tool* (KRAVETS, 2016, p.1), ou ainda os impactos que aumentam drasticamente quando o tema é terrorismo, existe grande entusiasmo com essa explosão de desenvolvimento.

Essa explosão de dispositivos oferece também a riqueza de detalhes necessária para a absorção de expertise humana dentro das máquinas. Relacionada com IoT está o grande número de informações que são geradas com esses dispositivos, o que foi denominado de Big Data. Esses dados, gerados principalmente nos últimos 10 anos, estão cada vez mais desestruturados e vem de todo lugar: de redes sociais a arquivos de log de servidores (DATAFLOQ, 2016, p.1). Embora essa quantidade desestruturada e não-relacionada de informações fosse um grande temor dos especialistas de tecnologia da

informação, que viam a incapacidade do homem de analisar e retirar conhecimento disso, com o crescente desenvolvimento das técnicas de IA, o Big Data é agora um aliado no combate aos grandes desafios da humanidade, como mostra a IBM ao colocar IA para auxiliar diagnósticos e análises de mercado.

O primeiro projeto de inteligência artificial foi da IBM, o Deep Blue, criado para disputar uma partida de xadrez com o campeão mundial Garry Kasparov. Embora o supercomputador tenha vencido, não se dá crédito à inteligência da máquina, mas sim ao grande poder de processamento, que fazia com que a a máquina analisasse 200 milhões de possibilidades, enquanto que o humano realizava 15 jogadas. Outro fator decisivo está também no fato de que Deep Blue construía uma árvore de jogadas futuras para simular cenários possíveis de acordo com cada movimento. Dessa maneira, o supercomputador poderia simular centenas de cenários e escolher os que pudessem render os melhores resultados.

O desafio mudou em 2006, com a conversa da IBM com produtores da BBC para colocar um computador para competir em um jogo na televisão, o Jeopardy, muito popular nos EUA. Nesse contexto, o desafio muda no sentido de que o computador não poderia utilizar força bruta para responder as questões e, antes de tudo, ele deveria ser capaz de entender a pergunta, algo que até hoje não se tem conhecimento preciso de como implementar em um sistema processado. E ainda existe o desafio de entender sarcasmos, ironias e piadas, relacionando-as com o objetivo da questão; algo além das técnicas de programação que se tem conhecimento.

Todavia, com 4 anos de desenvolvimento, milhões de documentos que alimentam a memória do supercomputador, e combinando redes neurais, técnicas de compreensão semântica e um sistema de ranking de respostas de acordo com o grau de confiabilidade, o Watson derrotou os dois maiores campeões de Jeopardy por muitos acertos de diferença. E embora o objetivo inicial fosse uma disputa em rede televisiva, descobriu-se que haviam inúmeras aplicações para essa verdadeira inteligência artificial, entre elas alimentar a memória do Watson com milhões de documentos médicos, históricos de pacientes, exames e opiniões médicas, ou seja, Big Data no contexto médico, para fornecer segundas opiniões aos especialistas sobre diagnósticos e abordagens mais eficazes de tratamentos e também compreender melhor as causas que originam o câncer e outras doenças (IBM, 2016, p.1).

2.4 DOMÓTICA

Domótica é uma área da ciência dedicada a estudar as tecnologias que podem organizar e facilitar a vida doméstica. O termo surgiu recentemente nos dicionários, formado pelas palavras do latim *domus* (casa) com “robótica” e traduz a automatização de mecanismos de um determinado espaço, de forma que operem de forma integrada. É uma tecnologia que permite a gestão de todos os recursos habitacionais, surgindo na década de 80 em construções de edifícios, que experimentavam as dificuldades de integrar segurança, climatização e iluminação.

Com o maior desenvolvimento da tecnologia e das ferramentas, o método se difundiu na integração dos cômodos nas residências, trazendo o mundo da domótica cada vez mais para a realidade do cotidiano. É um grande campo de desenvolvimento que envolve a IoT, visando o futuro da casa onde sensores e atuadores, controlados por sistemas embarcados, desempenham funções para os residentes e são controlados remotamente pela internet, permitindo o monitoramento do comportamento de cada ambiente.

Os BACS (*Building Automation and Control Systems)* podem ter impacto significante no consumo de energia. Por isso o Comitê Europeu de Padronização criou o "*EN 15232:2012 Standard: Energy Performance of Buildings – Impact of Building Automation, Controls, and Building Management*". Os padrões para construção de sistemas domóticos são referenciados na EN 15232, que inclui:

- Lista de controle, automação e técnicas que afetam a performance energética das construções;
- Método para definir os requerimentos mínimos para o controle, automação e técnicas de construção implementadas em diferentes tipos de construção;
- Procedimentos detalhados para quantificar o impacto que esses métodos possuem na performance energética;

O Padrão EN 15232 modela as bases para implementação de eficiência energética em construções. Ela introduz técnicas de construção em grupos de classes de eficiência, bem como métodos de cálculo (um detalhado e outro simplificado) usados para estimar o impacto que os sistemas de controle e automatização vão ter na performance energética. Por exemplo, Classe C é considerado o ponto de início em termos de tecnologia.

Além do desafio de conciliar dispositivos e protocolos diferentes, outro grande obstáculo é o fato de atribuir a cada objeto da casa um IP global, o que necessitaria de um

massivo número de endereçamentos que o IPv4 não suporta. Esse paradigma aliado ao crescimento de dispositivos com acesso a internet trouxe o desenvolvimento o IPv6 que resolve esse problema, mas adiciona empecilho de ser um protocolo de implementação mais complexo a nível de hardware e software.

Para minimizar o problema, uma das soluções propostas é tratar todos os dispositivos da residência como uma única rede, de modo que uma central, com maior escalabilidade, controle os objetos, colete informação e interaja com a internet, para que não seja necessário atribuir um endereço para cada componente ou função do ambiente, mas somente para rede domótica. (MIORI…, 2014, p.810). Assim, as chamadas *DomoNets* atribuem aos componentes internos da rede domótica endereços de IPv4, enquanto que o dispositivo que controla a domótica recebe endereçamento IPv6.

2.5 HARDWARE

2.5.1 Sistema Embarcado

É um sistema que possui software embarcado e hardware de computador, o que faz dele um sistema dedicado a uma aplicação ou uma parte de um sistema mais complexo (KAMAL, 2011, p.3). Todd D. Morton, autor de *Embedded Microcontrollers*, define como "sistemas eletrônicos que contém um microprocessador ou microcontrolador, mas não pensam como os computadores". São sistemas computacionais compostos de hardware, firmware e, as vezes, partes mecânicas, encapsulados em um dispositivo encarregado de executar uma tarefa específica. Diferem dos computadores pessoais justamente na objetividade, pois estes permitem navegar na internet, escrever um documento ou assistir vídeos. Sua estrutura é exemplificada na Figura 2.

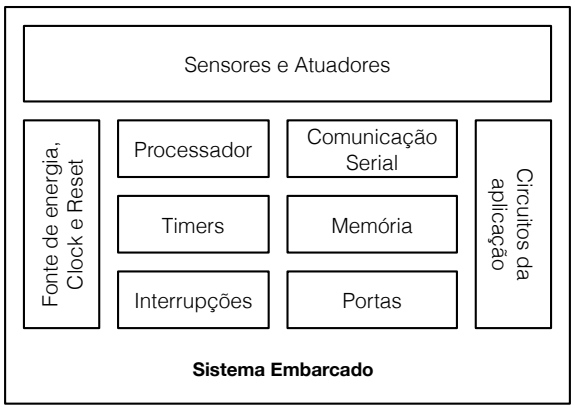


Figura 2 - Componentes de um sistema embarcado
Fonte: Kamal, 2011

Por serem mais simples que os sistemas de propósito geral, não possuem grande flexibilidade a nível de software e hardware, não permitindo que outras tarefas, além daquelas para as quais for desenhado, possam ser realizadas. Devido a isso, possuem recursos de memória e processamento mais restritos, o que reflete em seu preço, tamanho e consumo. A única flexibilidade permitida e desejada em seu projeto é o upgrade do firmware, o que permite correções e melhorias; todavia é realizada apenas pelos fabricantes. Possui três elementos fundamentais (KAMAL, 2011, p.4):

- Hardware, similar a um computador;
- Software de aplicação principal;

- Sistema de gerenciamento, que supervisiona o software de aplicação e organiza as tarefas. Sistemas maiores podem utilizar um Real-time Operating System (RTOS).

2.5.2 Microprocessador e Microcontrolador

Assim como existem dois tipos de propósito para um sistema embarcado, existem duas categorias gerais para os processadores. Os microprocessadores de uso geral são desenvolvidos para trabalhar em altíssimas frequências, porém consomem maiores quantidades de energia e necessitam de componentes externos para seu completo funcionamento. São destinados à cálculos mais complexos e com grande velocidade de processamento, como ocorre nos computadores pessoais.

Os microcontroladores são microprocessadores de propósito específico que, embora não trabalhem em frequências tão grandes, possuem recursos externos como memórias (RAM, FLASH, EEPROM), conversores A/D, Pinos de I/O para uso geral, interfaces de comunicação (serial, USB, I2C, SPI e outras), necessários para o controle de processos; além de consumirem menor quantidade de energia.

2.5.3 Filtros

A maioria dos sistemas de comunicação fazem uso dos filtros. São tipos de circuitos cujo ganho aplicado ao sinal depende de sua freqüência. Essa característica permite que sejam utilizados para selecionar uma determinada faixa ou para eliminar sinais indesejáveis.

De uma forma geral, pode-se afirmar que existem cinco tipos de filtros: Passa-baixas, que atenuam as altas frequências; Passa-altas, que atenuam as baixas frequências; Passa-faixa, que permite a passagem de uma faixa de freqüências; o Rejeita-faixa, que atua de forma inversa ao filtro Passa-faixa; e o Passa-todas (MALVINO, 2007, p.223). Na prática, o filtro ideal é impossível de se obter, mas pode-se utilizar aproximações que beneficiam resposta plana ou decaimento mais rápido (MALVINO, 2007, p.227).

Os filtros são divididos em passivos e ativos. Os filtros passivos são aqueles que utilizam apenas resistores, capacitores ou indutores. Nesse filtros, o sinal selecionado não sofre amplificações. Sua ordem é definida pelo número de indutores e capacitores no circuito. Em relação ao projeto, quanto maior a ordem, mais complexo.

Os filtros ativos, conforme o nome sugere, usam elementos ativos como amplificadores operacionais e transistores. Desse modo, ao passar por eles, o sinal selecionado pode ser amplificado, aparecendo na saída maior do que na entrada (ganho positivo). Sua ordem depende do numero de circuitos RC.

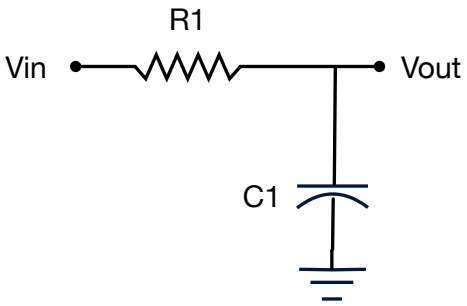


Figura 3 - Filtro passivo RC Passa-baixas
Fonte: Autor

Para um configuração simples de filtro passivo, como o RC passa-baixas da Figura 3 acima, frequências acima da frequência de corte serão atenuadas, e as frequências abaixo serão permitidas (MALVINO, 2007, 223).

O filtro PB ideal possui ganho de tensão e deslocamento de fase nulo para qualquer frequência na banda passante; todavia isso não ocorre no filtro real, que tem ganho de tensão e fase segundo as respectivas Equação 1 e Equação 2:

$$\alpha = -\,arctg\,(\omega RC) \qquad (1)$$

$$GV = \frac{1}{\sqrt{1+(\omega RC)^2}} \qquad (2)$$

Como o ganho na frequência de corte é ½√2, obtém-se de (3) os valores de R e C:

$$\omega c = \frac{1}{RC} \qquad (3)$$

Com essas equações pode-se traçar as curvas de resposta em frequência, representadas pelas Figuras 4 e 5 abaixo:

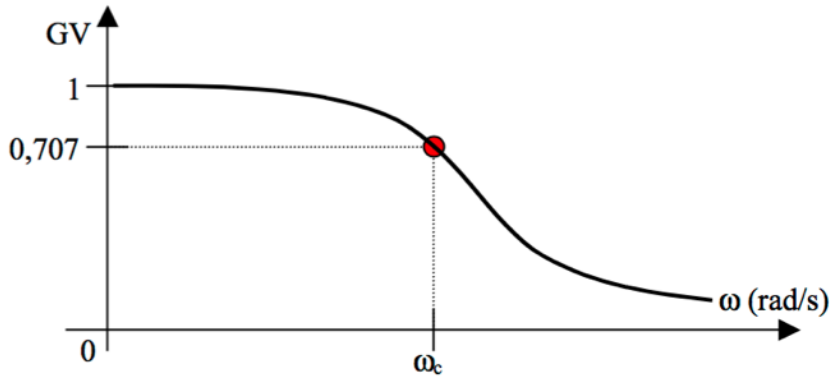


Figura 4 - Resposta em frequência - Ganho de tensão

Fonte: Autor

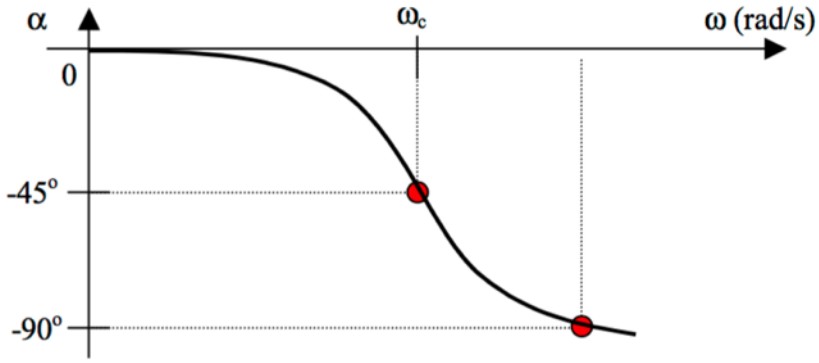


Figura 5 - Resposta em frequência - Deslocamento de fase

Fonte: Autor

2.5.4 Amplificadores Operacionais

Amplificadores Operacionais podem ser classificados como um circuito que realiza operações matemáticas. Foram utilizados, inicialmente, nos primeiros computadores analógicos, realizando operações de adição, subtração, multiplicação até cálculos mais complexos. Hoje são encontrados como circuitos integrados (MALVINO, 2007, p.57). Os Amp-Ops ideais possuem:

1. Ganho de tensão ∞ em MA;
2. Impedância de entrada ∞;
3. Impedância de saída 0;
4. Diferença de potencial entre entradas é 0 (Curto-circuito virtual);
5. Largura de banda infinita;
6. Sem deslocamento de fase.

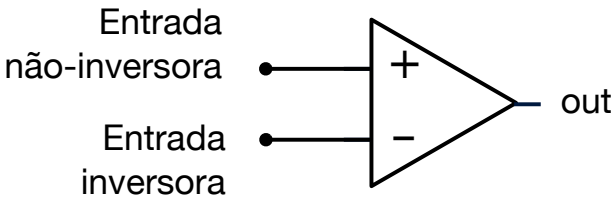


Figura 6 - Símbolo esquemático de um amp-op
Fonte: Malvino

Nos Amp-Ops reais, entretanto, essas considerações não são alcançadas plenamente, existindo parâmetros que devem ser analisados em um projeto, como as limitações de ganho, tensões e correntes de *offset*, banda de passagem e *slew-rate*.

Esses dispositivos dão abertura para uma série de configurações e circuitos, que apresentam características de saída próprias. Entre as mais conhecidas, estão o Amplificador Inversor, o Amplificador não-inversor, o Seguidor unitário (Buffer Analógico), Somador, Subtrator (Diferencial), Diferenciador e Integrador.

Na configuração de Amplificador não-inversor, que é mais estável que a configuração inversora, a fase do sinal de saída é a mesma do sinal de entrada, o ganho é determinado pela resistência de realimentação (feedback).

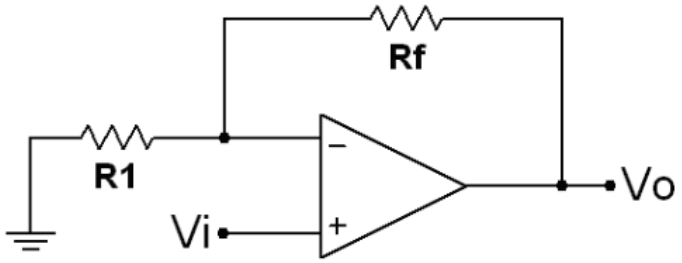


Figura 7 - Esquemático do amplificador não-inversor
Fonte: Malvino

A Equação 4 define a saída desse circuito:

$$Vout = Vin\,(1 + \frac{Rf}{R1}) \qquad (4)$$

Outra configuração utilizada é o Seguidor unitário, também conhecido como *buffer* ou Casador de impedâncias, pois não amplifica o sinal de entrada, não inverte fase e apresenta pequena impedância de saída e enorme impedância de entrada. Desse modo, a impedância de um circuito não interfere em outro, como a impedância de um sensor em um circuito de aquisição de sinal.

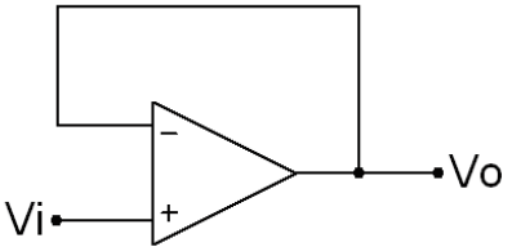


Figura 8 - Esquemático do *buffer*
Fonte: Malvino

A Equação 5 define a saída desse circuito:

$$Vout = Vin \tag{5}$$

2.5.5 Teoria de Amostragem

O grande desafio em um sistema de comunicação é determinar o número mínimo de amostras para que o receptor da mensagem receba o símbolo de forma fiel e sem ambiguidades. Harry Nyquist se preocupou com a questão e apresentou o Teorema de Nyquist (ou Teorema da amostragem de Shannon), fundamental no campo da Teoria da Informação, enunciando que, para a recuperação unívoca de todas as componentes da mensagem (até a componente de maior frequência f_{max}), a taxa de amostragem f_a deve ser, ao menos, duas vezes maior do que f_{max} (ROCHOL, 2011, p.67). O teorema pode ser expressado como:

$$f_a = 2f_{\max} \tag{6}$$

O teorema afirma então que, para um sinal analógico de banda limitada e que foi amostrado a partir de uma sequência finita de aquisições, pode ser perfeitamente reconstruído. O efeito de *aliasing* ocorre quando essa teoria não é praticada. Para prevenir ou reduzir o aliasing, duas coisas podem ser feitas:

1. Aumento da taxa de amostragem;
2. Introduzir um filtro anti-aliasing.

Esse filtro deve ser utilizado antes da amostragem para garantir que o processo obedece o teorema.

2.6 SOFTWARE

2.6.1 Firmware

Descreve um conjunto de instruções que estão diretamente ligadas com componentes de hardware. São gravados no chip de memória dos dispositivos, como EPROM ou FLASH, podendo ser encontrado nas mais diversas aplicações, de câmeras digitais, celulares e smartphones, eletrodomésticos a automóveis.

Pode ser do tipo *Foreground/Background,* um laço infinito que executa determinadas tarefas através da chamada de funções, ou um sistema em tempo real, RTOS, que surge com o objetivo de suprir certas deficiências do método anterior, como falta de prioridade das tarefas e falta de determinismo.

No desenvolvimento de um sistema embarcado sem RTOS é importante definir uma arquitetura de desenvolvimento de software. Observou-se três arquiteturas principais:

OSL (One Single Loop): um modelo simples que implementa duas funções básicas: as configurações iniciais, antes do laço principal, e o laço de repetição, que executará as funções. É a base de programação de microcontroladores como o Arduino. Apesar da vantagem no quesito simplicidade de construção, a velocidade de execução do laço principal depende das funções implementadas, de maneira que é praticamente impossível utilizar a frequência de execução para realizar alguma tarefa com requisitos de tempo. Além disso, há a possibilidade de travamento do sistema, caso alguma função entrar em *loop* infinito ou deadlock.

Interrupções: resolve o problema de execuções periódicas da abordagem anterior, possibilitando que uma função pré-definida seja executada sempre que um evento acontecer. Também há a vantagem de se poder utilizar modos de baixo consumo de energia. A desvantagem está na proteção de recursos que serão utilizados por mais de uma tarefa, como *buffers.* Outra desvantagem, e mais importante, está no fato da arquitetura do microcontrolador escolhido, o MSP430G2553, não possuir prioridades nas interrupções comuns, o que dificulta a execução de duas tarefas que utilizem interrupções, como comunicação e conversões.

FSM (Finite State Machine): o ciclo de execuções é parecido com o método OSL, mas as tarefas são, agora, estados. A diferença é que é possível que a saída de um estado pode ser baseada em condições, gerando mais de um caminho de execução. A máquina de estados pode facilmente ser escrita em código em C com uma estrutura switch-case, em que cada estado é representado por um case e a mudança para o novo estado está nas condições de saída do estado atual. Entre as vantagens, está a facilidade em implementar novas tarefas.

Essa técnica é também denominada de *Cooperative Multitasking*, implicando que as funções que estiverem no top-slot ou bottom-slot podem demorar o tempo que quiserem e apenas ao término de sua execução a próxima função começa a ser executada. A desvantagem é que a frequência com que cada função será executada não pode ser determinada facilmente.

2.6.2 Sistema Operacional (SO)

É um software que conecta o microprocessador ao usuário, cuja função é gerenciar os recursos do sistema. De forma simplista pode-se dizer que um SO possui as funções de gerenciamento de tarefas, gerenciamento de memória e gerenciamento de recursos. Existe a visão top-down, que o define como provedor de abstração do hardware, e a visão bottom-up, em que ele é gerenciador de recursos, ou seja, manipula tarefas.

Os sistemas computacionais modernos englobam um complexo conjunto de recursos, como as memórias principal e secundária, dispositivos de E/S, discos, interfaces de rede entre outros. Isso tornaria o projeto de programas otimizados que interagem com tais dispositivos uma tarefa árdua.

Por isso desenvolve-se o software denominado sistema operacional (SO), que gerencia estes recursos e oferece um modelo de alto nível para os programadores, de modo que estes não necessitem se preocupar com o gerenciamento a nível de hardware. Ou seja, o trabalho do sistema operacional é alocar de maneira ordenada os recursos do sistema entre os programas que competem por eles (TANNENBAUM, 2010, p.3). O SO normalmente opera em modo núcleo (supervisor), com maiores privilégios e o modo usuário.

Um diagrama dos componentes que compõe um SO pode ser visto na Figura 9.

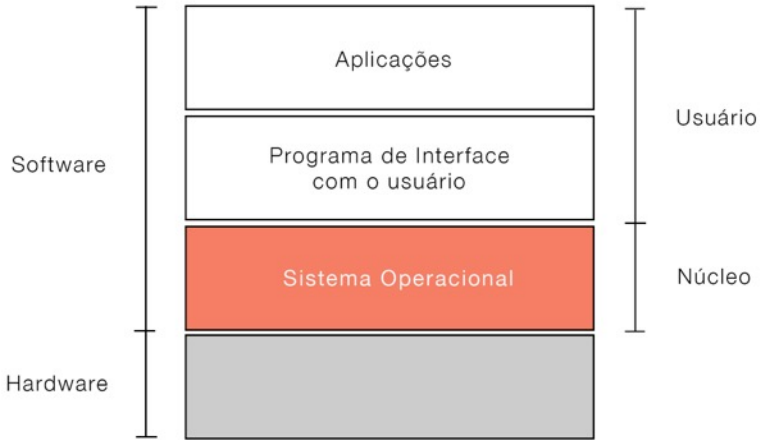


Figura 9 - Diagrama de camadas de um sistema computacional
Fonte: TANNENBAUM

O que diferencia Sistema operacional do firmware é a complexidade, uma vez que, por exemplo, o núcleo de um SO como Linux possui mais de 15 milhões de linhas de código, divididas de forma extremamente modular. Também possuem um ciclo de vida mais longo devido a dificuldade de codificar.

2.6.3 Segurança Computacional

As distribuições de linux são, em sua maioria, *open code*, ou seja, abertas para que qualquer pessoa possa analisar e desenvolver. Policitantes do código-fonte aberto possuem inclinação à ideia de que há o benefício de vários desenvolvedores depurem e revisem o código, diminuindo os erros de segurança. O *Fuzz Test*, realizado na Universidade de Wisconsin, em 1988, analisou 80 sistemas operacionais em diversas plataformas de hardware. De fato, o teste mostrou que os sistemas fechados sofriam de problemas fundamentais que não existiam nos equivalentes de código-fonte aberto.

Construir e disponibilizar um sistema deve ser um processo planejado, projetado para responder eficientemente a ameaças contra a segurança durante a sua execução. Esse processo deve possuir alguns itens (STALLINGS, 2014, p.372):

a) Planejamento: a primeira medida envolve o levantamento de requisitos de segurança para as aplicações que serão disponibilizadas e para os dados do usuário. A indicação é promover esses requisitos em tempo de projeto, maximizando a segurança e minimizando custos.
b) Fortalecimento do sistema operacional: a base de todo o sistema é o sistema operacional (OS), uma vez que ele disponibilizará e executará os demais serviços, portanto esse é um item de segurança básico. O maior problema está no fato de que as instalações padrões dos OSs maximizam os recursos disponíveis, sub-utilizando a maior parte deles, podendo deixar o sistema mais vulnerável. A melhor prática, nesse contexto é, uma vez traçados os requisitos de segurança necessários, restringir o uso desses serviços e aplicações, removendo os que não são de interesse do projeto. Além disso, manter o sistema atualizado com os *patches* disponíveis e instalar ferramentas adicionais como antivírus.
c) Manutenção: Monitoramento e geração de arquivos de registro de eventos (logs) e backup regular dos dados críticos.

Os serviços de segurança também devem ser fornecidos por um sistema para a proteção dos recursos e informações, implementando políticas e mecanismos que evitem ameaças. A mais geral definição é a capacidade de suportar requerimentos de segurança como confidencialidade, integridade ou avaliabilidade, ou todas elas.

3 MATERIAIS E MÉTODOS

A topologia do sistema é constituída de dispositivos embarcados que efetuarão medições do consumo de energia e temperatura, chaveamento e comunicação com um dispositivo central. A unidade centralizadora se comunicará com as unidades sensoras, coletando as informações e inserindo-as em um banco de dados na nuvem. O diagrama lógico de funcionamento podendo ser viso na Figura 10 e o fluxograma geral de funcionamento do sistema no Apêndice X.

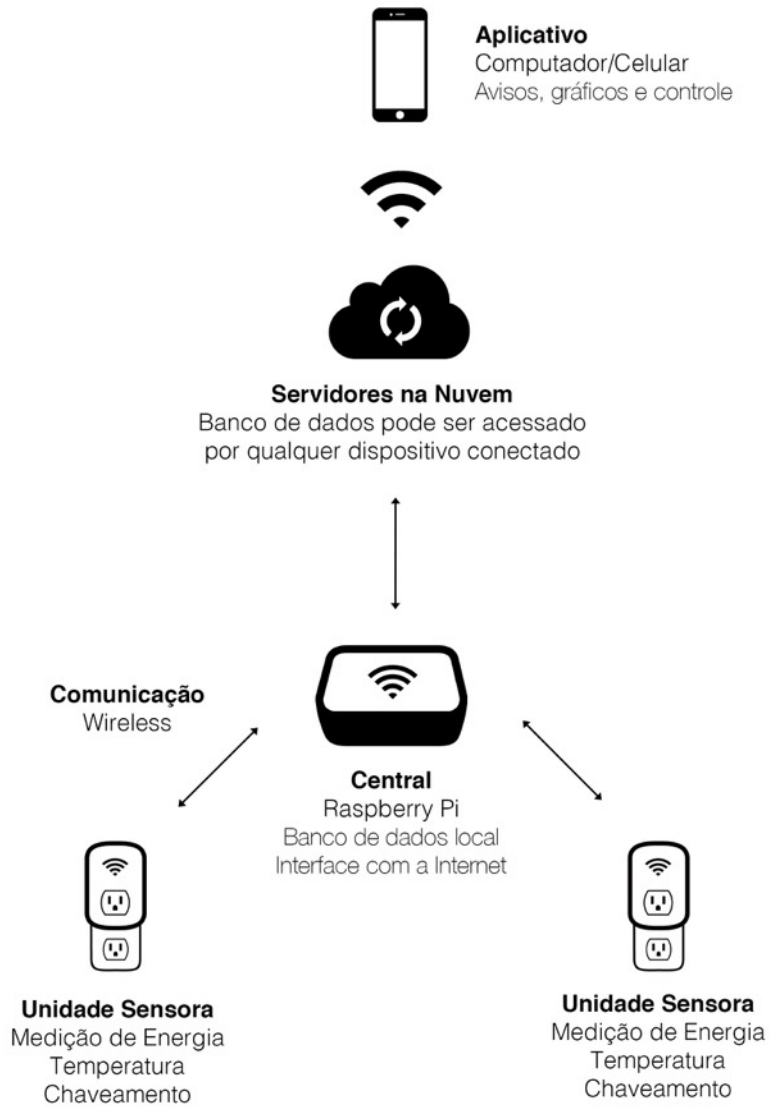


Figura 10 - Diagrama geral do sistema

Fonte: Autor

Existem duas opções para a hospedagem do banco de dados. A primeira é colocá-lo na unidade central e realizar configurações complexas no roteador para permitir que dispositivos na internet, fora da rede local, consigam acesso. Embora essa abordagem se mostre mais barata, pois dispensa a utilização de um servidor online, é de complexa implementação. Existem diversos tipos de roteadores e cada um necessitaria de uma configuração diferente para permitir o acesso de dispositivos fora da rede local, o que levaria muito tempo.

A segunda abordagem, escolhida para este trabalho, é colocar os dados em um banco de dados hospedado em um servidor online, de maneira que este seja o agente centralizador do sistema.

Um servidor é um *host* que está executando um ou mais serviços ou programas que compartilham recursos com clientes. O modelo cliente-servidor, que será explorado no projeto, tornou-se uma das ideias centrais de computação de rede. Muitos aplicativos de negócios, escritos hoje, utilizam o modelo cliente-servidor.

A unidade central acessará o servidor para inserir as informações colhidas das unidades sensores e verificará quais são os comandos a serem realizados. Essa abordagem pode possuir um custo mensal associado à hospedagem do servidor, mas é mais simples por guardar os dados em apenas um lugar; mais segura, pois o fornecedor do servidor será responsável pela segurança desses dados; e escalável, uma característica desejável em um sistema, rede ou processo.

3.1 MATERIAIS

3.1.1 Raspberry Pi B

Para controlar um conjunto considerável de unidades sensoras, interfaceá-los com a internet e prover uma rede domótica segura, é necessário usufruir de bom poder de processamento e plataformas consolidadas. O Raspberry Pi, da Raspberry Pi Foundation, que em 2012 começou a receber grande cobertura da imprensa, possui todos esses requisitos, além de ser leve, pequeno, oferecer as facilidades de programação de um computador comum e possuir grande suporte da comunidade.

O modelo escolhido é o Raspberry Pi 1 Modelo B, mostrado na Figura 11, um SoC BCM2835 com ARM11 de 700MHz, 512 MB de RAM, 26 pinos de GPIO, duas portas USB, Saída de vídeo HDMI e RCA, com 100Mb de porta Ethernet para acesso à internet (RASPBERRY, 2016, p.1).

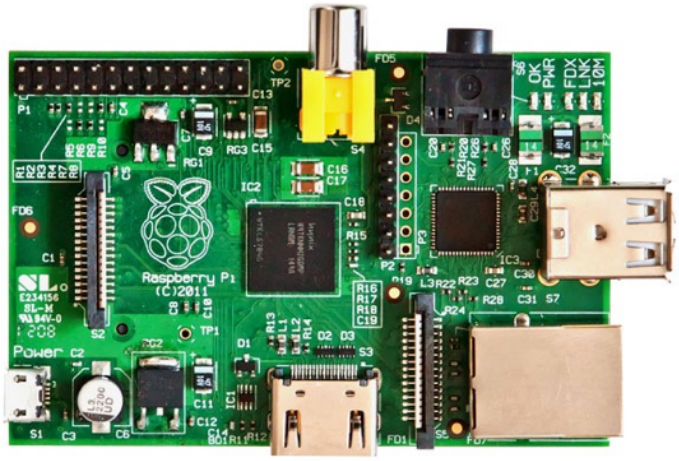


Figura 11 - Representação do Raspberry Pi
Fonte: Raspberry Foundation

O raspberry pi é amplamente utilizado em projetos disruptivos e que necessitam de uma plataforma robusta e de fácil acesso. Entre os temas de projetos em maior sinergia com o *status quo* das áreas de estudo de computação, estão Clusters (ALASDAIR, 2015, p.1), Processamento de imagem (CAMBRIDGE…, 2012, p.1) e Inteligência Artificial.

Conta com 26 pinos no total: 3 pinos de energia, 3V3 (3.3V), 5V0 (5.0V) e GND (0V); 6 pinos DNC (do not connect); e 17 pinos GPIO (General Propose I/O).

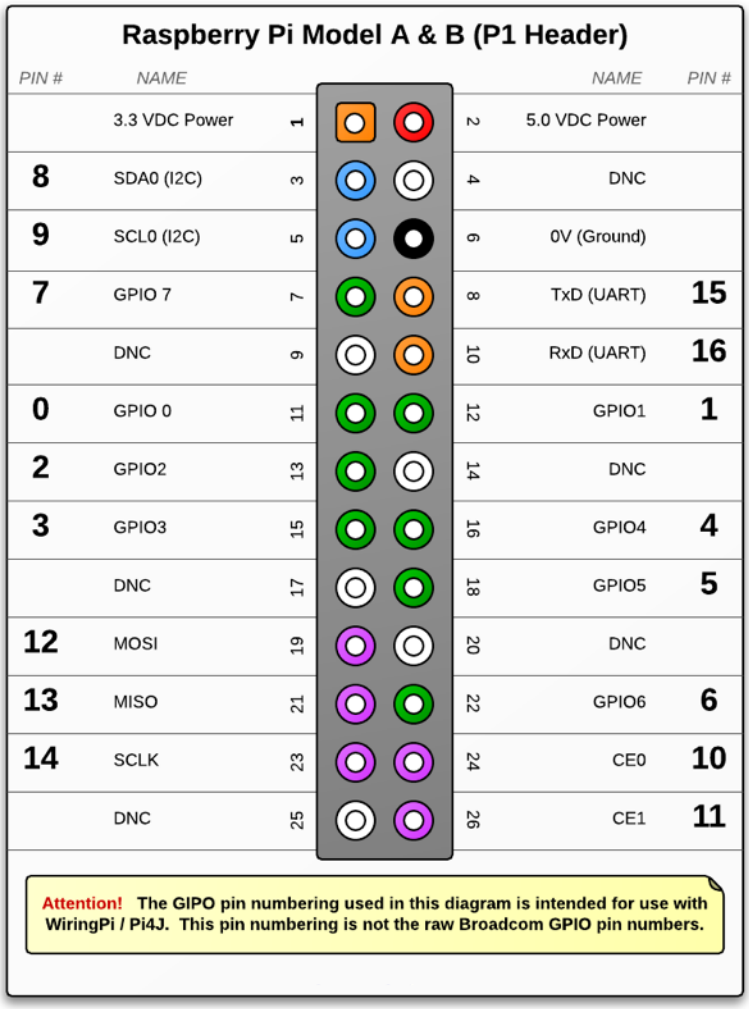


Figura 12 - Tabela de conexões

Fonte: Raspberry Foundation

3.1.2 MSP430G2553

Microcontrolador fabricado pela Texas Instruments que incorpora uma arquitetura RISC 16Bits. É constituído de amplo conjunto de periféricos, entre eles o conversor Analógico-Digital (ADC) de 10 bits, timer de 16 bits e módulos de comunicação serial. A flexibilidade, valor acessível e baixo consumo de energia também são fatores que influenciam a escolha. Ele possui 512 bytes de RAM, 16Kb de memória não-volátil e frequência de 16MHz (TEXAS…, 2013, p.24). A Figura 13 mostra o *lauchpad* desse microcontrolador.

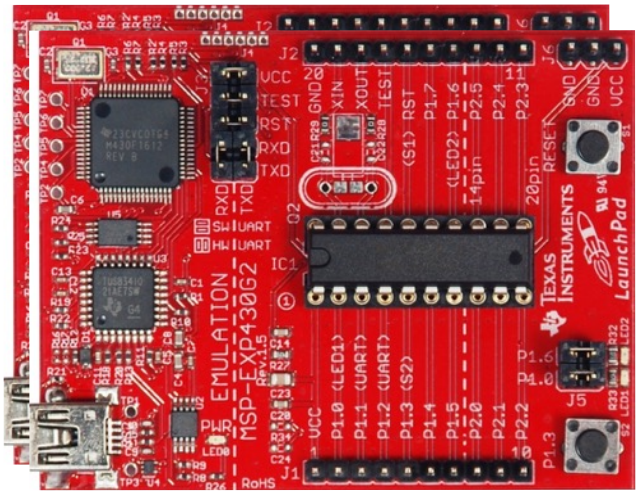


Figura 13 - Microcontrolador MSP430G2553
Fonte: Texas

a) Basic Clock System

O *Basic Clock System*, BCS, é responsável por fornecer o clock para os componentes que compõe o microcontrolador, tendo como características o baixíssimo consumo de energia e incluindo diversas fontes de clock, entre elas o auxiliar ALCK, o principal MCLK e secundário SMCLK.

b) Timers

O Timer A é um contador de 16 bits com vários registradores dedicados à captura e comparação, suportando saídas PWM e temporização. Pode obter clock internamente do ACLK e do SMCLK, oferecendo a possibilidade de dividi-lo, ampliando as possibilidades de configuração para atender as necessidades de tempo.

c) ADC

O módulo conversor analógico-digital, ADC, realiza conversões de 10bits de resolução com geração de referência e controle de transferência de informação DTC. Este é responsável pela inserção das conversões diretamente na memória sem interferência da CPU, o que deixa o processo mais rápido. Possui velocidade de 200ksps e disparo de conversão pelo timer A (TEXAS…, 2013, p.534). Opera em 4 modos:

- *Single-Channel Single-Conversion*: única conversão por canal
- *Sequence-of-Channels*: única conversão em uma sequência de canais
- *Repeat-Single-Channel*: repetidas conversões por canal
- *Repeat-Sequence-of-Channels*: repetidas converses por sequencia de canais

As fontes de clock possíveis são SMCLK, MCLK, ACLK, e o oscilador interno ADC10OSC. Todas as etapas de conversão podem ser realizadas em até 27 ciclos de *clock*, se for escolhido realizar 4 amostras por conversão (TEXAS…, 2013, p.540).

3.1.3 Code Composer Studio

É um IDE para design de firmware para sistema microcontrolados da Texas Instruments. Projetado para sistemas embarcados de baixo nível com depuração baseada em Joint Test Action Group (JTAG), este criado para desenvolver métodos para testar circuitos impressos depois de fabricados.

Os últimos lançamentos são baseados em versões do Eclipse IDE de código aberto, que pode ser facilmente estendido para incluir suporte de depuração a nível de SO e aplicativos de compilação de código aberto como o GCC.

3.1.4 Módulo XBEE

ZigBee é a especificação para comunicação WPAN construído sobre o padrão IEEE 802.15.4, desenhado para aplicações que visam dispositivos de baixo consumo e pequenas porções de dados. O XBee é um dispositivo de comunicação que implementa o protocolo Zigbee, e pode transmitir e receber dados na banda industrial, cientifica e medica (ISM) de 2.4GHz, com alcance de 30 metros (*indoor*) até 90 metros (*outdoor*) e taxa de transmissão de até 250,000 b/s (DIGI, 2009, p.4). É um dispositivo de baixo consumo de energia, cerca de 1mW, 45mA a 3,3V para transmissão e 50mA a 3,3V para recepção, e ½uA em modo *sleep.* Essas características fazem desse dispositivo uma boa escolha para automação.

Figura 14 - Módulo Xbee S2
Fonte: Digi

Implementa a técnica de modulação de espectro de propagação denominada de *Direct Sequence Spread Spectrum* (DSSS), uma tecnologia de espalhamento de espectro definida pelo padrão IEEE 802.11, usada extensamente em aplicações militares e ideal para transmitir taxas de dados mais baixas nos cabos de energia elétrica. Opera com uma faixa de frequência definida como canal, com 14 canais ao todo (BARTZ, 2012, p.140).

O software para sua configuração, o XCTU da Digi, é gratuito, disponibilizando comandos nos modos AT e API. Com o XBEE é possível fazer comunicação entre microcontroladores, computadores ou qualquer dispositivo que tenha porta serial. Pode-se fazer conexões de ponto-a-ponto até multi-conexões de rede.

Dispositivos que tem interface UART podem se conectar diretamente com os pinos do modulo RF como mostrado na Figura 15. Os dados entram no módulo pelo pino data in (DI) e o sinal deve ser HIGH quando nenhuma informação está sendo transmitida.

Cada byte da mensagem é composto de um bit de start (LOW), 8 bits de dados (LSB) e um bit de parada (HIGH) (DIGI, 2009, p.10). Comunicação serial dependerá de duas UARTs configuradas de maneira compatível, ou seja, baud-rate, paridade, start bits, stop bits e data bits devem ser equivalentes.

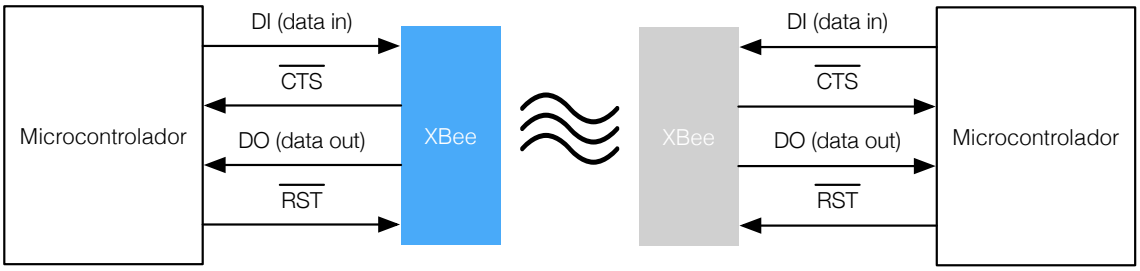


Figura 15 - Diagrama do ambiente UART para fluxo de dados

Fonte: Autor

Pode-se utilizar os modos de operação Transparente e Application Programming Interface (API). O primeiro é o padrão de funcionamento, em que toda informação enviada pelo pino DI através da UART é transmitido pelo RF. O modo API é uma alternativa ao modo padrão, onde a aplicação pode interagir com a rede. Assim, toda informação que entra e sai pelo módulo é contida em *frames* que definem operações específicas ou eventos.

Para um configuração ponto-a-ponto, alguns parâmetros são essenciais:

PAN ID: *Personal Area Network* é o número de identificação da rede Xbee ao qual o módulo irá se conectar.Para juntar dois ou mais módulos Xbee à mesma rede, basta que estejam com o mesmo PAN ID.

MY 16-bit Source Address - Número que identifica o módulo Xbee na rede PAN.

DH Destination Address High - Determina a parte mais significativa de 32 bits do endereço de 64 bits que podemos atribuir aos módulos Xbee.

DL Destination Address Low - Determina a parte menos significativa de 32 bits do endereço de 64 bits que podemos atribuir aos módulos Xbee.

3.1.5 Sensor de corrente

O HWCT-5A é um sensor de corrente por efeito hall de baixo custo e reduzidas dimensões físicas. O fator de conversão é de 1000:1, ou seja, para 1A medido, obtém-se 1mA na saída. Possui resposta linear de 0 até 5A, resistência dielétrica de 3kV e faixa de temperatura que varia de -40ºC a 70ºC. Embora seja barato e pequeno, insere fase de até 20º na medição. Para sua instrumentação, segundo o fabricante, pode-se utilizar na saída do sinal uma resistência ou um amplificador operacional.

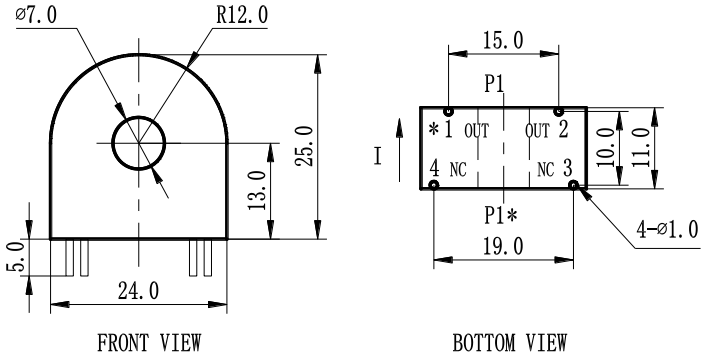


Figura 17 - Sensor HWCT5A

Fonte: Datasheet

3.1.6 Sensor de temperatura

O termistor é um tipo especial de resistência que modifica seus valores de acordo com a temperatura a que é exposto. Existem dois tipos: termistor PTC (Positive Temperature Coefficient), que aumenta a resistência conforme aumenta a temperatura; e o termistor NTC (Negative Temperature Coefficient), inverso ao PTC, ou seja, a resistência diminui a medida em que a temperatura aumenta.

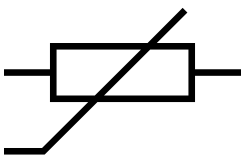


Figura 18 - Símbolo do termistor

Fonte: Autor

Os materiais de construção do componente são, normalmente, platina, níquel, cobalto, ferro e óxidos de silício. Os termistores são divididos em 3 grupos: esfera, disco e vidro. O termistor escolhido é o NTC MF52 de 3mm e 10kΩ da Catherm, que permite a leitura de temperaturas entre -55ºC e 125ºC. É do tipo esfera, sintetizado em um corpo de cerâmica. Possui vasta utilização na indústria, seja para refrigeração e ar condicionado, linhas automotivas, controle de temperatura, sistemas de detecção e alarmes contra incêndio.

Entre as facilidades dessa abordagem pode-se listar as reduzidas dimensões do componente e rápida resposta, o baixo custo se comparado a sensores de temperatura mais complexos, como o LM35 da Texas Instruments, e disponível em valores populares de resistência (CATHERM, 2012, p.3). Todavia, o maior problema está no cálculo da temperatura, que utiliza equações que não podem ser processadas facilmente por processadores sem FPU. A Equação de Steinhart-Hart relaciona a temperatura T ambiente, em Kelvin, com a resistência do termistor R e a constante do tipo de termistor, β, e é definida pela Equação 7 abaixo.

$$\frac{1}{T} = \frac{1}{To} + \frac{1}{\beta} \ln\left(\frac{R}{Ro}\right) \qquad (7)$$

Um fenômeno que deve ser observado é o Self-Heating effect, que ocorre sempre que há uma corrente fluindo pelo termistor. Visto que este é um tipo especial de resistor, existe a dissipação de calor, o que pode afetar a precisão da medição.

3.1.7 Relé

É um interruptor que abre ou fecha quando corrente elétrica percorre o dispositivo. Há dois tipos principais de relés: eletromecânicos e de estado sólido. A diferença entre as categorias é que a segunda diz respeito a dispositivos completamente eletrônicos, sem partes mecânicas em movimento.

Para o dispositivo eletromecânico, em volta da bobina é gerado um campo eletromagnético quando o relé é energizado, fechando o contato. Entre as aplicações, estão (1) a proteção de entradas e saídas de CLP através da isolação galvânica e (2) segurança para acionamentos de cargas de alta corrente ou tensão através de sinais de baixa

corrente. São encontrados na indústria em geral, em automação predial e residencial, na geração, transmissão e distribuição de energia e em máquinas e equipamentos.

Sua vantagem está no baixo custo, baixa tensão de contado, exclusão de dissipadores de calor e isolação entre entrada e saída (FINDER, 2015, p.1).

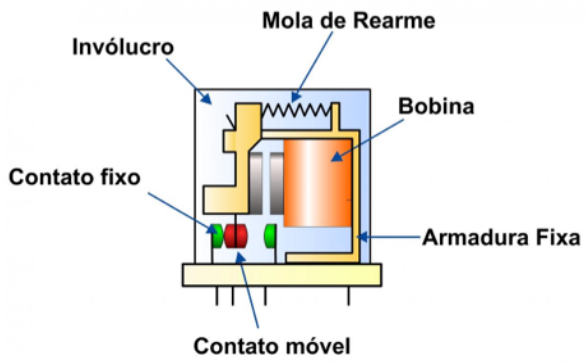


Figura 19 - Relé eletromecânico
Fonte: Finder, 2016

O SRD-S-105D, da Sanyou Relays, é um relé eletromecânico com contato de prata construído para automatização de aparelhos da casa como ar condicionado e iluminação, oferecendo a capacidade de chavear até 12A e 250V em baixo consumo, com dissipação de 0,36W. Possui preço acessível e pequenas dimensões físicas (SANYOU, 2015, p.1).

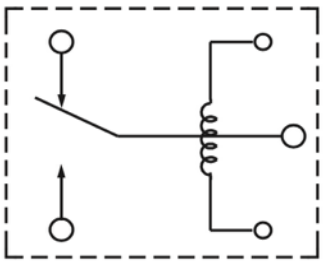


Figura 20 - Relé SRDS105D
Fonte: Sanyou, 2015

3.1.8 Amp-Op

O LM324, da Texas Instruments, possui alto ganho independente e elimina a necessidade de duas fontes, podendo ser alimentado de 3V até 32V para fonte simples, e de +-1,5V até +-16V para fontes duplas (TEXAS…, 2015, p.1). Pode ser utilizado em comparadores, multivibradores, integradores ou diferenciadores e filtros ativos.

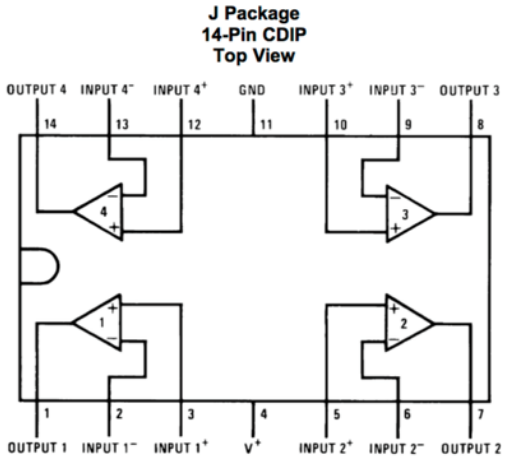


Figura 21 - Pinagem do amplificador operacional
Fonte: Texas, 2015

3.1.9 API Java

As funções de conexão poderiam ser construídas sobre sockets, mas a complexidade do protótipo aumentaria bastante, uma vez que os protocolos proprietários deveriam ser entendidos e implementados. Uma alternativa é a utilização de APIs, que são encontradas no pacote *Java Database Connectivity* (JDBC), que é o padrão da indústria para conexões entre java e bancos de dados SQL. Para cada banco de dados existe um JDBC. Os métodos implementados são Date, DriverManager, DriverPropertyInfo, SQLPermission, Timer, Timestamp e Types (ORACLE, 2016, p.1).

Esse conjunto de classes realiza a ponte entre o código cliente que usa a API JDBC e o banco de dados. São elas que sabem se comunicar através do protocolo proprietário do banco de dados. Esse conjunto de classes recebe o nome de driver. O JDBC possui dois pacotes: o java.sql e o javax.sql, que são obtidos realizando o download do software Java Platform Standard Edition (Java SE) 7 ou pelo download direto da biblioteca.

3.1.10 AppleScript

É uma linguagem de programação orientada a objetos criada pela Apple, que fornece controle direto sobre aplicações e partes programáveis do sistema operacional (APPLE, 2016, p.2). Através de uma interface de editor presente nativamente no Mac SO, é possível editar, compilar executar scripts, podendo exportar a aplicação desenvolvida diretamente como um executável. Também é possível consultar em um dicionário específico todas as funcionalidades e integrações com outras aplicações do sistema operacional que são oferecidas.

Segundo a empresa, a utilização dessa linguagem é indicada para quando se cria uma aplicação para um dispositivo Apple ou quando se deseja automatizar processos que utilizem o SO, *exempli gratia*, quando se deseja enviar uma mensagem a algum de seus contatos sempre que um site específico for acessado. Por outro lado, existem limitações como a eficiência face à cálculos intensos e ações complexas, como acesso a um servidor. Mas isso pode ser contornado aliando essa linguagem a outros tipos de scripts como Shell scripts (APPLE, 2016, p.3).

3.2 UNIDADE SENSORA

3.2.1 Hardware

Os circuitos devem ser desenvolvidos para possibilitar o chaveamento da carga pelo relé, a leitura correta dos valores de corrente e temperatura, além da comunicação com o dispositivo central. O diagrama de blocos da Figura 22 mostra a disposição do sistema de circuitos que compõe a unidade sensora.

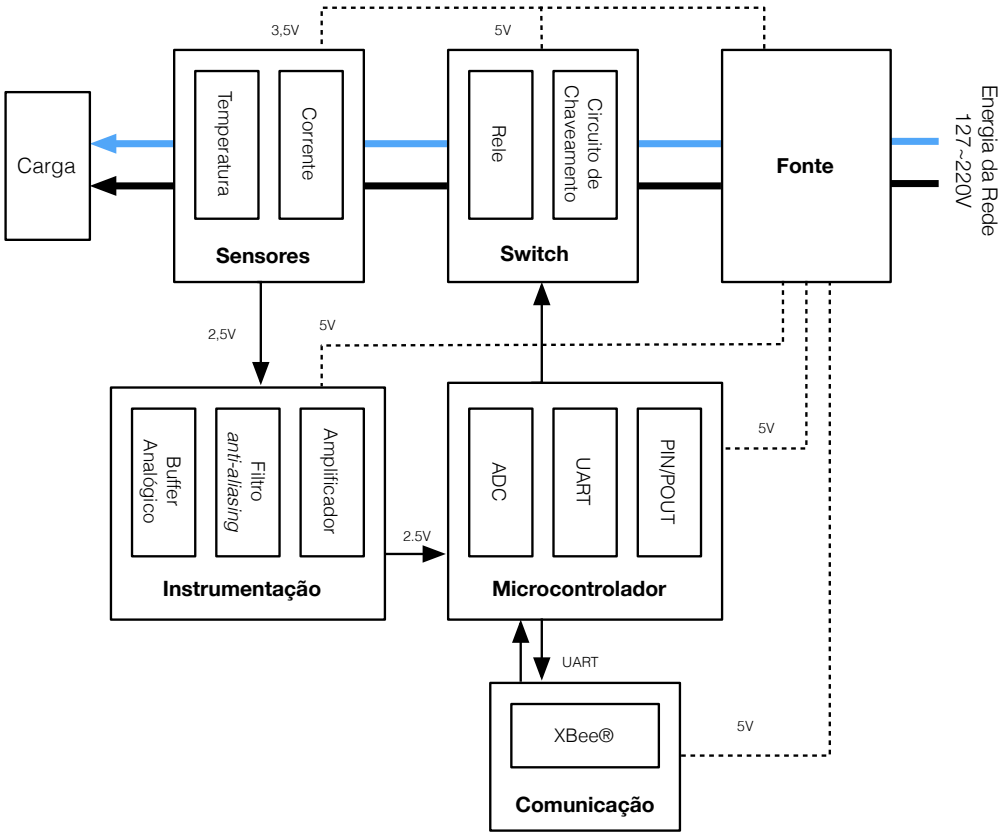


Figura 22 - Diagrama de circuitos da unidade sensora

Fonte: Autor

No bloco *Switch*, o circuito de acionamento será a maneira que o microcontrolador controlará a carga, utilizando o relé para abrir e fechar contato, como uma chave. Como dito anteriormente, a grande vantagem está no fato de que a tensão da rede não estará em

contato com os circuitos de baixa potência, devido à isolação galvânica propiciada pelo SRD-S112D.

Para que o acionamento ocorra satisfatoriamente é necessário prover uma corrente de 75 mA para 5V, que o microcontrolador não pode sustentar. Então utilizou-se o transistor BC337-25 como chave para suprir esses valores.

Um efeito interessante nessa configuração está no momento em que o relé é desenergizado. As linhas de força do campo magnético da bobina, que se encontram em seu estado de expansão máxima, começam a se contrair. Nesta contração há a indução de uma tensão de polaridade oposta àquela que criou o campo e que pode atingir valores altos. Por isso, para proteger o microcontrolador desses picos de corrente inversa, fez-se uso do diodo 1N4004, como mostra o esquemático na Figura 23.

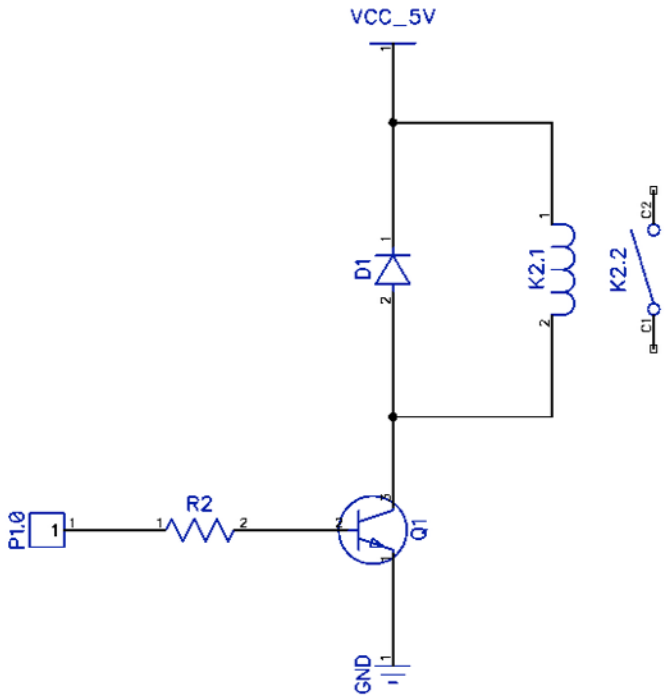


Figura 23 - Circuito de acionamento de carga

Fonte: Autor

Para o sensor de corrente, a saída produzida deve ser amplificada na escala de 2,5V do ADC do microcontrolador. Com a utilização de um circuito amplificador não-inversor, mostrado na Figura 24, obtém-se um sinal de saída 5 vezes maior que o de entrada. A Equação 8 descreve a modelagem desse amplificador.

$$Vout = (1 + \frac{R7}{R6})\, Vin = 5Vin \qquad (8)$$

Desse modo, para a saída máxima do sensor, 5mA*100Ω = 500mV, tem-se o sinal máximo amplificado de 2,5V. Então o ADC lê o valor de tensão presente em seu terminal e converte para uma palavra digital de 10 bits (0 a 1023) proporcional a este sinal de entrada.

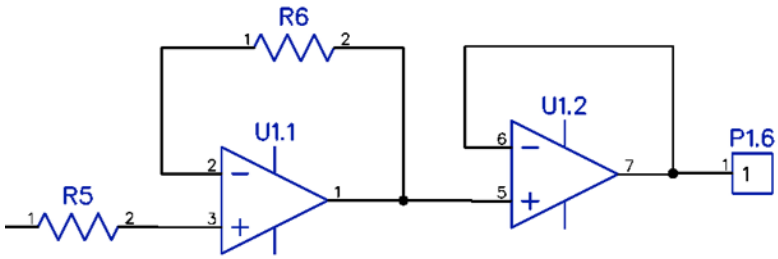


Figura 24 - Sensor de Corrente

Fonte: Autor

Para o sensor de temperatura obtida pelo termistor, montou-se um circuito divisor de tensão com o resistor R5 de 10kΩ em série ao NTC. Este divisor tem como tensão de entrada VCC do microcontrolador, 3,3V, e o sinal de saída do divisor de tensão será conectado à entrada analógica do microcontrolador, conforme a Figura 25

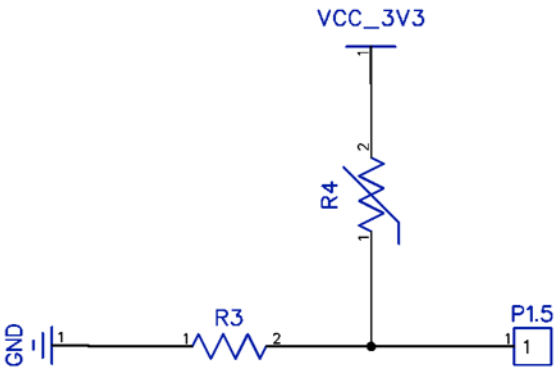


Figura 25 - Circuito para leitura do termistor

Fonte: Autor

A equação utilizada para o pré-processamento dos valores de temperatura em uma tabela relaciona a equação do divisor de tensão do termistor com o valor obtido quando realizada a conversão, como mostra a Equação 9.

$$Vtemp = \frac{R_5}{(R + R_5)} Vcc = \frac{ADC}{1023} 2,5 \quad (9)$$

Isolando R em (9), que é a resistência do termistor, encontra-se a Equação (10). Substituindo-a em (7) e simplificando, tem-se a equação de descreve a temperatura em função da resistência do termistor (11):

$$R = \frac{14322}{ADC} k - 10k \quad (10)$$

$$T = \frac{1185146}{3977 + 298 \ln(\frac{R}{10k})} \quad (11)$$

Depois de amplificar o sinal dos sensores, é necessário um filtro para evitar *aliasing*. Definindo-se o número de conversões que deveriam ser realizadas para alcançar cálculos com precisão satisfatória, obtém-se a frequência para construção desse filtro.

Pela teoria de amostragem, para a frequência de f_{max} = 60Hz da rede, precisaríamos de uma frequência de amostragem no mínimo duas vezes maior, f_a = 120Hz. Todavia, devido aos ruídos da rede e pelo fato de querermos as harmônicas além dessa frequência, definiu-se a frequência de Nyquist para dez vezes acima da frequência da rede. Com isso, temos f_{max} = 600Hz, que será a frequência de corte do filtro.

Como a frequência de amostragem deve ser, ao menos, o dobro de f_{max}, ao realizar 79 amostras a cada ciclo da onda da rede, respeita-se o teorema e obtém-se uma frequência quase oito vezes maior que f_{max} (f_a = 4740Hz).

Com um filtro passivo passa-baixas de primeira ordem, obtém-se atenuação suficiente para o propósito do protótipo e simplicidade de projeto. Utilizando as equações (1), (2) e (3), escolhendo ωc = 500Hz um pouco abaixo dos 10*60Hz devido a atenuação mais lenta desse tipo de filtro, tem-se o filtro desejado.

O ADC do microcontrolador não possui buffer analógico de entrada, então adicionou-se um na entrada P1.6, para que a saída do circuito de instrumentação não interferisse nos valores da medição. A construção do filtro para ambos os sensores de tensão e de corrente é equivalente e mostrada na Figura 26.

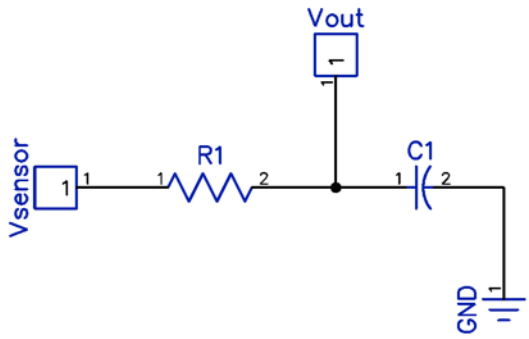


Figura 26 - Filtro *anti-aliasing*
Fonte: Autor

O esquemático completo do sistema pode ser visto no Apêndice XI.

3.2.2 Firmware

Para realizar todas as atividades propostas para a unidade sensora, encontrou-se o desafio de inserir todas e ainda respeitar requisitos de tempo do sinal colhido da rede. Utilizando-se a abordagem de FSM com a adição de um contador de tempo, obtém-se o determinismo desejado no projeto.

A abordagem lembra a construção de um RTOS, de modo que se implementa um *tick-count* para manipular a execução das atividades do sistema. As tarefas que deverão ser desempenhadas estão descritas no Quadro 1.

Tarefa	Descrição
Configuração Inicial	Configuração dos clocks, portas, timers, conversores e interrupções.
Central de Gerenciamento	Chamar as tarefas
Energia	Conversões de corrente e temperatura
Cálculo	Realiza os cálculos de conversão
Temperatura	Conversão de temperatura utilizando *look-up table*
Liga/desliga	Controle da carga
Comunicação	Recebe e transmite dados
Sincronização	Assegura restrição temporal

Quadro 1 - Tarefas da unidade sensora

Apesar de não ser ideal, já que o processador fica ocioso para padronizar os tempos de execução das tarefas com uma função de sincronização, o determinismo atingido por esse método traz imensas vantagens no desenvolvimento. Além disso, é uma abordagem possível de se implementar em sistemas que não são capazes de suportar um RTOS e necessitam de algum grau de determinismo temporal.

O fluxograma da Figura 27 apresenta o modelo conceitual de funcionamento do firmware da unidade sensora.

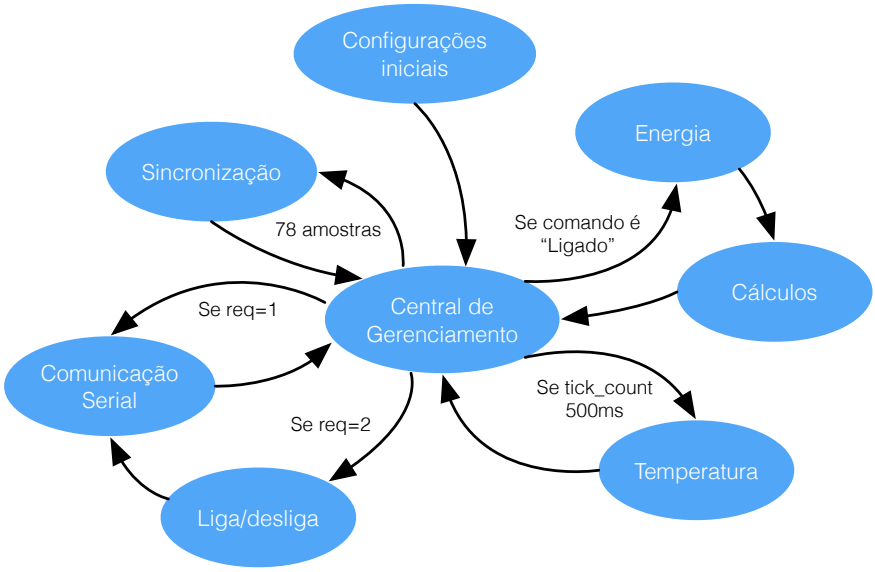


Figura 27 - Fluxograma do firmware da unidade sensora

Fonte: Autor

Cada tarefa será executada por uma biblioteca específica, e a central de gerenciamento, que está na main.c, acessará as funções implementadas por elas para fazer o sistema funcionar. A Figura 28 mostra essas bibliotecas.

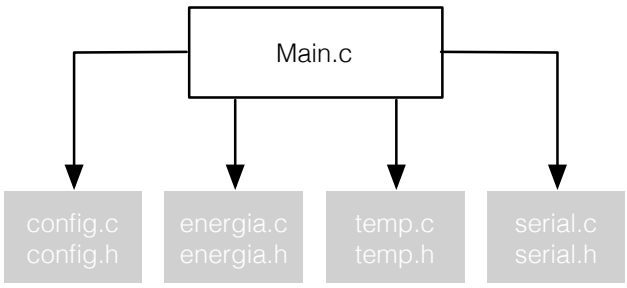


Figura 28 - Bibliotecas implementadas no firmware da unidade sensora

Fonte: Autor

Configurações iniciais: o primeiro estado a ser executado deve realizar as configurações do BCS, inicio das estruturas e variáveis que receberão os valores de corrente, cálculos e de incremento. Configura-se também as portas de I/O, *timers* e ADC10. Para o BCS, configurou-se o ACLK com 32768Hz, por meio do oscilador LFXT1CLK; o MCLK com oscilador DCOCLK de cerca de 16MHz e o SMCLK também com DCOCLK, de 16MHz. As configurações presentes nesse estado serão realizadas apenas uma vez, quando o dispositivo é ligado. O estado é implementado utilizando as bibliotecas *config.c* e *config.h*, presentes no Apêndice I.

Para a configuração do ADC utilizou-se a referência de 2,5V, o que define o degrau de 2,441mV por unidade digital. No registrador de controle ADC10CTL0, além da referência, é habilitada a interrupção de conversão.

No registrador ADC10CTL1 são definidos os canais de entrada do conversor, A5, A6 e A7, que foram determinados analisando-se as funções de cada porta disponível no microcontrolador. Por exemplo, as portas P1.0 e P1.1 serão utilizadas para a comunicação UART, portanto os canais A0 e A1 devem ficar livres. Também defini-se a fonte de *clock,* que será o MCLK configurado para oferecer 16 MHz. Desse modo, pode-se realizar uma aquisição mais refinada de cada amostra, com até 64 ciclos de relógio, e devido a máxima frequência, isso não impactará em demasia no tempo final de conversão. As conversões devem ser realizadas em sequência de canais, intercalando-se dois canais, um para o sinal de corrente e outro para temperatura, nessa ordem.

Por último, habilita-se as portas analógicas correspondentes aos canais de conversão. Dessa maneira, P1.0 corresponderá ao canal A0 até P1.7 correspondente à A7. A configuração do ADC pode ser vista no Apêndice I.

É importante determinar do número de amostras que serão efetuadas, pois este implicará no tempo de conversão. São determinantes desse tempo a frequência configurada para o conversor f_{ADC}, o período de amostra $t_{amostra}$ e o tempo para $N_{amostras}$.

O tempo limite para executar todas as tarefas do sistema é determinado pela frequência da rede elétrica que, no caso do Brasil, é de 60Hz, ou seja, antes de 16,667ms todas as tarefas devem ter sido executadas 79 vezes. A frequência do ADC foi configurada para utilizar o MCLK de 16 MHz e período da amostra de 64 ciclos do clock do ADC. Logo, com 79 amostras de corrente e outra para temperatura, o tempo gasto pelo MSP430G2553 para a amostragem do sinal é mostrado na Equação (12).

$$t_{exec} = \frac{N_{amostras} t_{amostras}}{f_{ADC}} = \frac{80(64+12+1)}{16M} = 0,388ms \quad (12)$$

Considerou-se apenas uma conversão de temperatura, uma vez que essa conversão ocorre a cada 500ms, ou seja, vai ocorrer uma vez a cada 2370 conversões de corrente; então o pior caso será aquele em que tal conversão ocorre.

Isso mostra que os tempos para todas as conversões estão dentro dos limites estipulados, utilizando 3% do tempo total de 1/60ms, o que deixa margem suficiente para as demais tarefas executarem.

A interrupção do *timer* levará o fluxo de execução do código ao tratamento de sua RTI, que realizará a contagem de tempo do sistema. Definiu-se o modo contínuo de contagem, escolhendo-se a fonte SMCLK de 16MHz para gerar os tempos necessários.

O *tick_count* escolhido foi de 160/16MHz = 10us. Sua escolha se justifica no fato de que cada ciclo de execução das tarefas, que será executada 79 vezes, deverá consumir o tempo máximo de (1/60ms)/79 = 210us, então o valor de contagem deveria ser múltiplo de 210. A configuração do timer pode ser vista no Apêndice I.

Um detalhe importante na escolha do *tick_count* : provavelmente o tempo de cada ciclo dos 79 não será exato. Nesse caso, esse tempo é de 210,97us e não exatos 210us. Uma solução seria diminuir o *tick_count* para aumentar a precisão, mas nesse microcontrolador haveria problemas com outras interrupções ocorrendo no sistema. Outra abordagem é arredondar valor do *tick_count* para baixo e deixar que uma tarefa de sincronização resolva.

Central de Gerenciamento: é a função principal, responsável por trocar as tarefas que devem ser executadas. Isso é realizado com a utilização das seguintes variáveis: f_e (flag da tarefa Energia), f_t (flag da tarefa Temperatura), f_ld (flag da tarefa Liga/desliga), e f_en (flag da tarefa Comunicação), que não permitem que as tarefas que foram executadas sejam repetidas antes do ciclo terminar. Quando todas as *flags* são desativadas, chega-se à condição final, onde há a sincronização de tempo e o reset das *flags* para que o processo ocorra novamente.

Energia: Implementa as funções necessárias para os cálculos de consumo. Sua função mais importante é a `converte_energia()`, que intercala os canais de conversão A6, habilita a conversão no registrador ADC10CTL0, guarda os valores convertidos, desabilita a conversão e incrementa a variável `num_amostras`, que diz respeito a quantas conversões já foram realizadas. O estado é implementado utilizando as bibliotecas `energia.c`, presente no Apêndice III.

Cálculo: depois de obtidos os valores de corrente e antes de voltar à tarefa Central de Gerenciamento, o fluxo de execução vai para a tarefa de cálculos. Depois de eliminar ruídos observados nas conversões durante os testes realiza-se, a cada uma das 79 conversões, o somatório delas, com o objetivo de calcular a potência média para aquele ciclo de onda. Para calcular a potência, precisa-se saber a tensão e corrente que estão presentes na carga:

$$ADC_{current} = (\frac{current_{conversion} 2,5}{1024}) \frac{1}{5} \frac{1}{100} \qquad (13)$$

A Equação (13) traduz qual será a tensão, em mV, de saída do sensor de corrente que entra na porta do ADC. Para saber a corrente real na carga em ampere, utiliza-se a Equação (14):

$$Real_{current} = (\frac{current_{conversion} 2,5}{1024}) \frac{1}{5} \frac{1}{100} 1000 \qquad (14)$$

Com esta equação, pode-se calcular a potência real utilizando-se a Equação 15:

$$Real_{power} = (\frac{current_{conversion} 2,5}{1024}) \frac{1}{5} \frac{1}{100} 1000 V_{load} \qquad (15)$$

Todavia, a Equação (15) não poderia ser utilizada no circuito que será construído devido a uma simplificação que impede que as tensões negativas provenientes do sensor sejam amplificadas. Uma alternativa seria a retificação de onda completa desse sinal antes do estádio de amplificação, de maneira que a fase negativa da onda fosse espelhada e o conversor pudesse mensurar seu valor correto. Mas, dependendo do valor do sinal do sensor, a diferença de potencial da ponte retificadora pode ser muito grande.

Por isso, deve-se colher os resultados do ciclo positivo e duplicar seu resultado; dessa forma teremos uma aproximação para um ciclo completo de onda. Além disso, a potência será calculada no final de todas as 79 aquisições, então a equação mais detalhada é a (16):

(16)

Temperatura: Para a tarefa de leitura da temperatura, a mesma abordagem da conversão de corrente é utilizada, mas a troca é para o canal A5 do microcontrolador. A função `converte_temperatura()` realiza essa troca e, após a conversão de temperatura do termistor, chama a função `temperatura()`, que realiza a consulta dos valores pré-processados de conversão de temperaturas, relacionando-os com valores obtidos do ADC. O estado é implementado utilizando a biblioteca `temperatura.c`, presentes no Apêndice IV.

Essa abordagem é denominada *Look-up table*, e consiste em uma tabela gerada por algum software matemático que calcula valores de alguma determinada função previamente. Para esse projeto, a equação utilizada foi (7). Uma vez calculados, cria-se uma lista que pode ser consultada a cada conversão do ADC, de forma que os cálculos complexos descritos anteriormente são contornados. Embora seja vantajoso nesse sentido, a precisão desse método dependerá da precisão da criação da tabela, ou seja, quanto mais valores de entrada, melhor a resolução da saída. Como a temperatura do circuito não necessita de grande precisão, utilizou-se 27 valores com 1ºC de variação, de 18ºC a 45ºC.

Liga/desliga: controla a carga que será ligada ao sistema, ligando-a ou desligando-a por meio do acionamento do relé. Além disso, comuta a flag `liga_desl`, uma vez que esta é responsável por habilitar ou não a tarefa de Energia e, consequentemente, a tarefa Cálculo. Ela é importante no sentido de que se o sistema está desligado, então teoricamente não há energia sendo consumida, dispensando a necessidade do fluxo de execução passar por essas duas tarefas.

A execução dessa tarefa depende de uma condição de verificação. A *flag* `requis_central` possui dois estados '1' e '2'. Se '1' então a central enviou uma requisição de informação, e a tarefa de comunicação será executada. Se for '2', além da requisição de informação, a central quer mudar o estado atual de funcionamento, ligando ou desligando a carga.

Comunicação: a parte crítica no firmware era criar um mecanismo de transmissão e recepção de dados sem utilizar muito tempo de interrupção da UART, uma vez que isso pode atrapalhar a interrupção mais importante do sistema, que é a do *timer*. Correspondente ao envio, todavia, mesmo com um pacote de poucos bytes, somente essa tarefa consumia muito mais que o estipulado para um ciclo de 210us. A solução foi dividir o pacote, de 16

bytes, em sub-pacotes de 2 bytes. Dessa forma, a cada período de 1/60 ms poderiam ser enviados 79*2 bytes = 158 bytes, e o tempo gasto seria menor que 100us. O estado é implementado utilizando as bibliotecas `serial.c`, presentes no Apêndice V.

A mensagem enviada pela interface UART possui um byte de início S0; 3 bytes de numeração de pacote N2, N1 e N0; 3 bytes de endereço da unidade sensora A2, A1 e A0; 1 byte de status da carga ST0, para saber se ela está ligada ou desligada; 4 bytes de consumo C2, C1 e C0; 2 bytes de temperatura T1 e T0; 2 bytes livres L1 e L0; e 1 byte de término de mensagem E0. O pacote possui o seguinte escopo:

```
S0 | N2 N1 N0 | A2 A1 A0 | ST0 | C2 C1 C0 | T1 T0 | L1 L0 | E0
```

É Interessante notar que os valores de Status, Potência e Temperatura são enviados pela unidade sensora depois de converter cada número no seu correspondente ASCII. Desse modo, na unidade central esses valores devem ser novamente traduzidos nos valores decimais para inseri-los no servidor e exibí-los na tela.

Condizente ao recebimento de comandos da unidade central para a unidade sensora, utilizou-se os caracteres R e S, este apenas para coletar as informações da unidade central e aquele para coletar as informações e mudar o status dessa unidade sensora; ou seja, se a unidade central enviar ‘R’, a unidade sensora envia informações; caso a unidade central envie ‘S’, a unidade sensora envia informações e liga ou desliga a carga. O comando é inserido na RTI da UART, mas só ocorrerá, de fato, quando a Central de Gerenciamento chegar no estado de Liga/desliga.

Sincronização: faz com que o fluxo de execução só saia dessa tarefa quando o tempo decorrido do ciclo atual corresponder ao tempo de um ciclo de tarefas, ou seja, 210us ((1/60s)/79). Isso é feito com um laço que aguarda a *flag* `flag_210` ser setada dentro da RTI do *timer*. Também há a preocupação de sincronizar o tempo de um ciclo de onda da rede, de modo que se respeite as 79 conversões para os 1/60s.

3.2.3 Otimização de código

Para obter os menores tempos em cada tarefa a ser executada na unidade sensora, é necessário compreender que a falta de uma Unidade de Ponto Flutuante (FPU) no uC implica em cálculos extremamente demorados caso envolvam ponto flutuante. Por isso, uma

das medidas de otimização mais importantes está na utilização de métodos que evitem esse tipo de cálculo.

Base Q, ou *IQ number format,* possibilita ao processador realizar cálculos com operandos em ponto flutuante como cálculos de ponto fixo, executando o código em menos ciclos de *clock*, liberando o uso do processador em menos tempo. A técnica também possui a vantagem de ser facilmente implementada em tempo de projeto, definindo-se uma base Q_n e, a partir dela, utilizando qualquer número dentro do corpo do código como sendo um produto deste número pela base escolhida. Assim, para converter um número flutuante em uma base Q_n, utiliza-se a Equação (17):

$$num_Q = num_{flutuante} \bullet Q_n \tag{17}$$

De um número em base Qn para flutuante, utiliza-se (18):

$$num_{flutuante} = \frac{num_Q}{Q_n} \tag{18}$$

Como o desenvolvimento do firmware foi baseado no modelo clássico de Engenharia de Software, o iterativo e Incremental, a adoção de uma base Q foi substituída pela atenção de evitar cálculos com decimais, ou seja, não há emulação desse tipo de processamento em qualquer parte do código.

Outra maneira de evitar problemas no sistema é colocar o mínimo de código possível dentro das interrupções, não impactando na execução das demais tarefas.

3.3 UNIDADE CENTRAL

3.3.1 Configurações Iniciais

A unidade central, que realizará cálculos mais complexos e a inserção dos dados das unidades sensoras na nuvem, será desenvolvida sobre o Raspberry, que pode receber diversos tipos de SO. A distribuição Debian foi escolhida considerando-se a comunidade de milhares de colaboradores que compõe o desenvolvimento e manutenção do sistema, caracterizando-o como o maior projeto de software livre do planeta (MOTA…, 2014, p.82). Grandes empresas e instituições utilizam Debian e, no Brasil, os maiores deles são o Exército Brasileiro, o Banco do Brasil, Caixa Econômica Federal e SUS, Sistema Único de Saúde.

Os motivos para a escolha do Debian também estão no fato dele ser multiplataforma, suportando 9 arquiteturas de hardware (MOTA…, 2014, p. 93), entre elas ARM, que é a utilizada no Raspberry Pi. Além disso, o kernel é portável e há pouca ocupação de disco que, sem ambiente gráfico, é de cerca de 600MB de memória. O suporte também é extenso, e respostas a incidentes de segurança são rapidamente tratadas pelos desenvolvedores. O processo de configuração inicial da unidade central segue três passos: 1) Download do SO; 2) Preparação do disco SD e 3) Instalação.

Para o primeiro passo, no link `https://www.raspberrypi.org/downloads/raspbian/` pode-se efetuar o download da imagem do SO.

Na segunda etapa, depois de efetuar o download do Rasbian, é preciso escrever a imagem do SO no cartão de memória SD, podendo-se realizar a tarefa com um software específico ou por alguns comandos no terminal. Para realizar a instalação o cartão SD deve possuir, ao menos, 4GB.

Uma vez que o cartão SD está formatado em FAT32, primeiramente deve-se saber o nome do cartão SD. Isto pode ser feito como comando `<diskutil list>`, que fornece uma tabela com os dispositivos montados no sistema. A montagem de um disco descreve uma técnica avançada de gerenciamento de arquivos, em que se mapeia uma partição para uma pasta vazia em outra partição formatada com o sistema de arquivos, como o disco principal do computador. Unidades montadas geralmente recebem um rótulo ou nome no lugar de uma letra da unidade, *e.g.* `disk2`, permitindo aumentar o armazenamento de uma unidade ou partição.

Depois de identificado qual número de disco se refere ao cartão de memória, deve-se desmontar a imagem, por meio do comando <diskutil unmountDisk /dev/disk#, onde disk# representa o nome do cartão de memória.

Posteriormente, na última etapa, com sudo dd bs=1m if=image.img of=/dev/<disk#> efetua-se a transferência da imagem do Rasbian SO para o disco, em que image.img é o diretório da imagem. Esse último passo consome um tempo considerável, pois depende da velocidade de transferência do disco principal. Com estes passos, basta inserir o cartão na entrada SD do Raspberry e ligá-lo para iniciar o *boot* do sistema.

3.3.2 Configurações de Rede

Como a ideia do sistema é a fácil utilização por parte do usuário, de modo que poucas configurações devam ser realizadas, o sistema deve, inicialmente, ser capaz de aderir a qualquer rede, surgindo a questão de como configurá-lo para obter conexão à internet automaticamente. Desse modo, o protocolo de comunicação dinâmica de host (DHCP) soluciona este problema, fazendo com que o *host* receba de um servidor DHCP, após envio de um pacote UDP de requisição, um pacote com configurações onde constará, pelo menos, um endereço IP, uma máscara de rede e outros dados opcionais, como o gateway e servidores de DNS.

Há dois modos de configurar a rede pelo terminal do raspberry. No primeiro, modifica-se o arquivo localizado dentro do cartão SD “cmdline.txt”, que passa argumentos para o *kernel* do SO logo na inicialização do sistema. O segundo método é iniciar o SO e, por linha de comando, modificar as configurações. Optando-se pela configuração por comandos, é recomendável desligar a interface eth0 antes, com <sudo ifdown eth0>, para que as configurações sejam realizadas. O comando <sudo nano /etc/network/interfaces> possibilita a edição do arquivo *interfaces*, que define as configurações desejadas de rede. Nesse arquivo, eth0 se refere à porta Ethernet e wlan0 à wireless, caso haja alguma. Como utilizou-se a conexão cabeada, configurou-se eth0 salvando no arquivo o texto <iface eth0 inet dhcp>. Depois de salvar o arquivo, reiniciou-se a interface de rede com <sudo service networking restart>.

3.3.3 Instalação de Pacotes e configurações gerais

Antes de iniciar as configurações, é necessário verificar se o SO utilizado está em sua última versão, com o comando pelo terminal <sudo apt-get update>.

No linux, instalações são realizadas por meio de pacotes, uma maneira mais rápida e limpa de instalação, com programas pré-compilados que são obtidos pelos gerenciadores de pacotes. No Rasbian SO existe o gerenciador *apt-get* que, assim como os similares, baixa os pacotes, verifica as dependências e, caso necessário, baixa outros programas e bibliotecas que o programa inicial precisa para executar.

O Servidor Apache é a primeira ferramenta a ser instalada. Popular servidor livre, com amplo suporte da comunidade web, representa cerca de 47% dos servidores ativos no mundo. A instalação é realizada com o comando <sudo apt-get install apache2>.

Para o acesso e gerenciamento do banco de dados, escolheu-se o Mysql versão 5.4, lançada em 2011, pois apresenta melhor desempenho e uma estrutura já consolidada. É o mais popular no segmento, que utiliza linguagem de consulta estruturada, tendo como usuários NASA, Banco Bradesco, Dataprev, HP, Nokia, Sony, Lufthansa, U.S. Army, U.S. Federal Reserve Bank, Cisco Systems e Google. O comando para adquirir o pacote é <apt-get install mysql-server mysql-client>.

Também é necessário instalar o BC (*Basic Calculator*), aplicação tipicamente utilizada para scripts matemáticos, com linguagem similar ao C. Será um componente utilizado na obtenção da informação contida na mensagem enviada pela UART, que devem ser quebrados, traduzidos e inseridos no servidor online. Para a instalação, <sudo apt-get install bc>.

3.3.4 Comunicação serial com GPIO

Para realizar a comunicação serial utilizando a GPIO do raspberry pi, primeiramente é necessário desabilitar o Getty, serviço que mantém a porta ocupada e impede a utilização. Para isso, é necessário acessar o arquivo /etc/inittab e comentar a seguinte linha:

```
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Também é indicado em alguns fóruns que se previna que o S.O. envie dados pela porta serial no *boot*. Para isso, no arquivo de inicialização do sistema /boot/cmdline.txt remova o seguinte trecho de texto:

```
console=ttyAMA0,115200
```

3.3.5 Shell script

Para manter a Unidade Central conectada com o servidor e em comunicação com cada unidade sensora, criou-se um script que realizará toda a interface com banco de dados e porta serial continuamente. Detalhes do código podem ser vistos no Apêndice VIII.

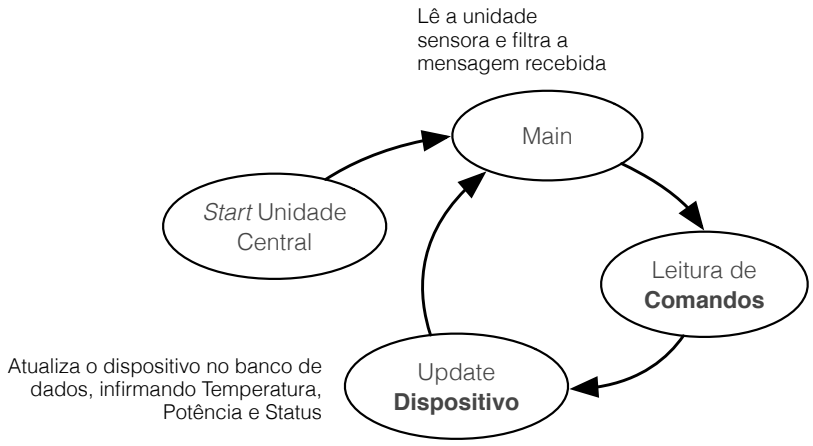


Figura 29 - Fluxograma do script da unidade central

Fonte: Autor

A primeira etapa, na função `main()`, consiste em ler as informações da unidade sensora (Potência, Temperatura e Status) para atualizar a tabela Dispositivos no servidor. Nesse ponto são realizados filtros para atribuir a cada variável o valor correto contido na mensagem, além de converter ASCII em decimal.

Na segunda etapa, na função `sql_read_comandos()`, depois de ler a tabela Comandos, em ordem decrescente de inserção, se houver algum comando, ele executa na unidade sensora, insere um alerta em Alertas e deleta o comando executado.

Na terceira etapa, na função `sql_update_dispositivo`, ele insere Potência, Temperatura e Status na tabela Dispositivos do servidor, que será lida pelo app java e *applescript*. Depois disso, o ciclo reinicia e o processo é repetido.

3.4 APLICATIVO

3.4.1 Organização

A Aplicação envolve um software java, onde as principais informações e interface de comando serão oferecidas ao usuário, e um aplicativo em *applescript* para tornar a experiência de notificações mais natural para o usuário.

Toda a informação que é coletada pela unidade central é inserida em um banco de dados no servidor online e acessada pelo software de aplicação em java. A aplicação poderia ser realizada de muitas formas, porém a linguagem java se mostrou mais eficaz para a construção rápida de um protótipo por incluir bibliotecas que facilitam a interface com a estrutura de banco de dados. Além disso, é multiplataforma, então o controle das unidades sensoras pode ser realizado em qualquer computador devidamente configurado. O algoritmo da aplicação segue as etapas descritas na Figura 30.

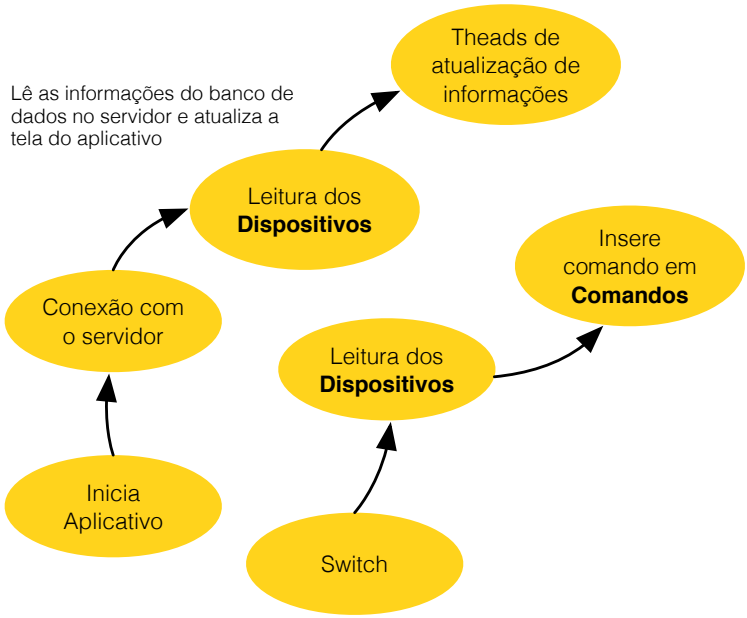


Figura 30 - Fluxograma do Aplicativo JAVA

Fonte: Autor

Quando inicia-se o software java, a primeira ação é conectar a aplicação com o servidor pelo botão INICIAR. Isso faz com que o usuário esteja conectado a cada unidade sensora que existir na rede. Uma vez iniciado, duas *threads* são inseridas no sistema, que

serão responsáveis pela atualização da tela principal e dos gráficos exibidos ao usuário, a cada 500ms e 300ms respectivamente. Dessa maneira, o aplicativo fica livre para as ações que o usuário desejar, como ligar uma determinada unidade sensora.

No momento em que o usuário deseja alterar o estado de uma unidade, o evento de botão de SWITCH ocorre, inserindo-se no servidor, na tabela Comandos, uma ação. Esse comando, que possui o código da unidade sensora, notificará a unidade central, que está conectada a esse banco de dados, e o comando será enviado a unidade sensora correspondente. Quando o comando é finalizado, a unidade sensora envia à central um pacote contendo as informações, a central interpreta os dados e atualiza o banco de dados com as informações, notificando sobre a ação concluída na tabela Alertas; então o processo recomeça.

Na inserção do alerta, o *applescript* desenvolvido, que está em execução e verificando o banco de dados, recebe a notificação e, utilizando a central de notificações do SO, informa o usuário. Para realizar as exibições necessárias na tela, o aplicativo deve seguirá o esquema da Figura 31.

Figura 31 - Interface do aplicativo JAVA

Fonte: Autor

A tabela conterá abas identificando as informações, como o nome do dispositivo, temperatura e os demais dados que estão na tabela Dispositivos. Essa tabela criará uma lista de todos os dispositivos que compõe o sistema, ou seja, quando um novo dispositivo for adicionado, ele deverá aparecer nessa lista.

O app construído em *applescript* utiliza uma biblioteca de shell scripts básicos que realizam a leitura e modificações no banco de dados. Seu código em *applescript* pode ser visto no Apêndice IV. Com ele, assim que houver mensagens na tabela Alertas no servidor, um *banner* deslizará pela tela e informará o usuário sobre o ocorrido, como mostra a Figura 32.

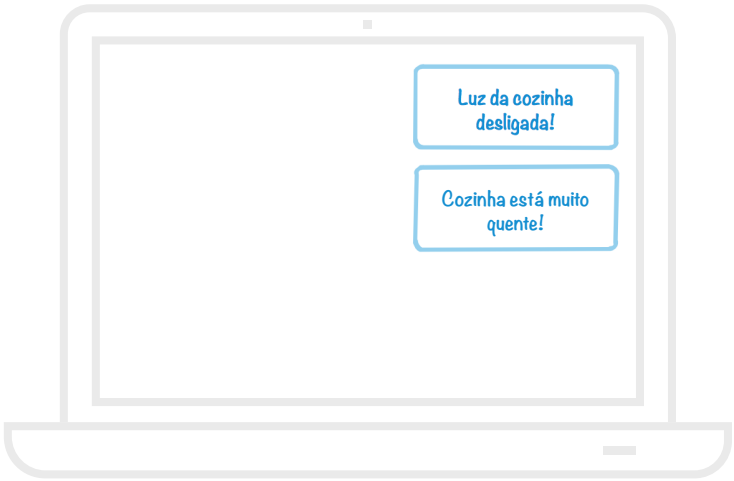


Figura 32 - Integração entre SO e Aplicação
Fonte: Autor

3.4.2 Conexão com o banco de dados

Para realizar a conexão deve-se primeiramente importar as APIs do JDBC no projeto da aplicação, que são encontradas para download no site da Oracle. Dessa API utilizou-se a classe DriverManager que permite a customização e acesso do banco pela aplicação. Entre os métodos disponíveis, usou-se o getConnection(String url, String user, String password) que realiza o acesso ao banco com as informações de endereço do banco, usuário e senha. Também existe a interface Connection que define métodos para executar uma query (como um insert e select), comitar transações e fechar a conexão. A função de conexão construída pode ser vista no Apêndice III.

3.4.3 Estrutura do Servidor/Banco de dados

O banco de dados foi estruturado para armazenar informações de todas as ações que ocorrem durante a execução do sistema. Isso envolve os dados obtidos da unidade sensora como potência, temperatura e status, o histórico desses dados ao longo do tempo e as datas de cada comando. A estrutura simples desse banco é mostrada na Figura 33.

Figura 33 - Tabelas do banco de dados
Fonte: Autor

O servidor MySQL utilizado fica disponível no serviço grátis db4free. O db4free.net fornece um serviço para testes de aplicações com a mais recente - as vezes versões em desenvolvimento - do MySQL Server.

De acordo com o próprio site, o db4free é um serviço de testes o que significa que não é feito para produtos finais, pois pode haver queda de energia, perda de dados, lentidão e não haverá funcionalidades de segurança esperadas em serviços de hospedagem.

As tabelas que estruturam o banco de dados podem ser vistas abaixo.

ALERTAS				
id	mensagem	disp		
Type: INT PRIMARY KEY auto_increment	Type: VARCHAR	Type: VARCHAR		
Lenght: 11	Lenght: 10	Lenght: 10		
COMANDOS				
id	cmd	date		
Type: INT PRIMARY KEY auto_increment	Type: CHAR	Type: CHAR		
Lenght: 11	Lenght: 10	Lenght: 10		
DISPOSITIVO				
id	status	temp	consumo	disp
Type: INT PRIMARY KEY auto_increment	Type: INT	Type: INT	Type: DECIMAL	Type: CHAR
Lenght: 11	Lenght: 11	Lenght: 3	Lenght: 6,2	Lenght: 10
HISTORICO				
id	temp	consumo	date	hour
Type: INT PRIMARY KEY auto_increment	Type: INT	Type: DECIMAL	Type: VARCHAR	Type: VARCHAR
Lenght: 11	Lenght: 3	Lenght: 6,2	Lenght: 10	Lenght: 10

Quadro 2 - Estrutura das tabelas do banco de dados

Essa tabela pode ser implementada pelo *script* no Apêndice IX.

4 RESULTADOS

Com os métodos apresentados, encontrou-se os valores de potência, consumo e temperatura com erro aceitável e que, com os algoritmos definidos para a unidade sensora, unidade central e aplicativo, existisse sinergia no sistema completo. O Apêndice X explica o funcionamento geral.

4.1 UNIDADE SENSORA

Devido à preocupação do microcontrolador da unidade sensora não realizar os cálculos nos tempos necessários, impactando em resultados sem integridade, evitou-se ao máximo os cálculos com decimais, utilizando as técnicas explicitadas anteriormente.

Para observar o impacto dessas medidas no tempo de execução, realizou-se simulações e testes comparativos entre diferentes arquiteturas. O resultado pode ser visto no Quadro 3.

Processador	Biblioteca Math.h	Iterativo com Ponto Flutuante	Iterativo com Base Q
HCS08 8bits - 40MHz	54001	64115	18008
MSP430G2553 16bits - 16MHz	41600	25150	3834
Coldfire V1 32bits - 50MHz	2070	11993	1496

Quadro 3 - Comparativo entre processadores

O quadro acima mostra o número de ciclos de cada microcontrolador para realizar um valor constante de divisões com ponto flutuante.

Com três microcontroladores, dois em simulação e um em mãos, embora a diferença entre a execução por ciclos de clock seja grande entre os processadores, normalizando os dados em relação à frequência do barramento, obtém-se tempos de execução próximos entre a arquitetura de 8bits do HCS08 e a do MSP430G2553.

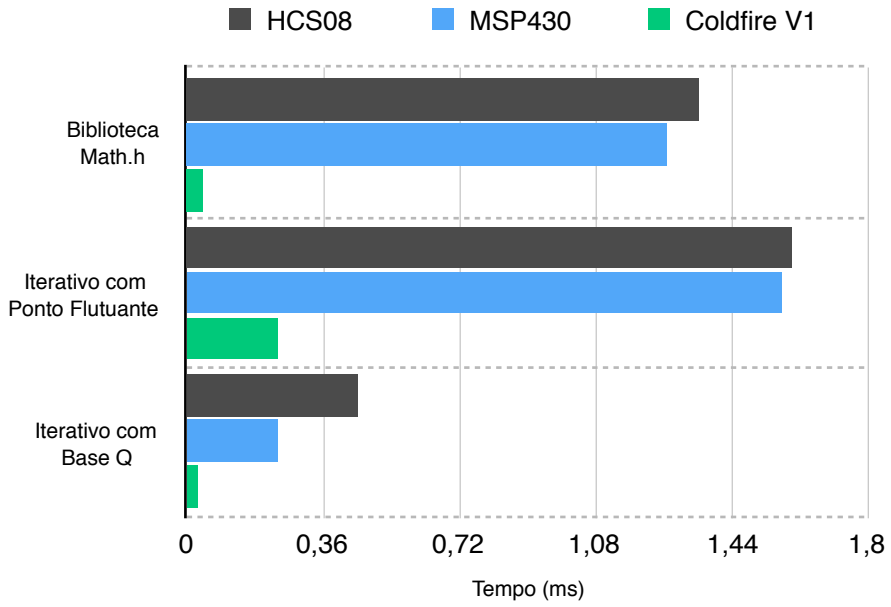


Gráfico 4 - Comparativo entre tempos de execução

Esses resultados foram determinantes para a escolha do microcontrolador, que não precisaria ser necessariamente o de maior poder de processamento, e mostrou que as técnicas de programação voltadas para evitar os cálculos com ponto flutuante e o bom planejamento do firmware são extremamente importantes para o bom desempenho do sistema.

Com relação ao sensor, alguns testes foram realizados para verificar os dados e precisão do sensor de corrente HWCT-5A, utilizando um resistor de 100Ω na saída do sensor e lâmpadas de 40W, 60W e 80W como carga, para fornecer diferentes correntes. Os resultados são mostrados no Quadro 4.

Potência (127V)	40W	60W	100W
Corrente Teórica	0,787A	0,472A	0,315A
Corrente Prática (média)	0,769A	0,46A	0,303A
Erro Relativo	2,28%	2,54%	3,8%

Quadro 4 - Testes com sensor de corrente

Fonte: Autor

Na etapa de amplificação do sinal de corrente, o amplificador não-inversor com buffer de ganho de 5 vezes apresentou reposta satisfatória, que pode ser vista no canal CH1 da Figura 34 abaixo, testado com uma entrada de 500mV a 100Hz, vista no canal CH2.

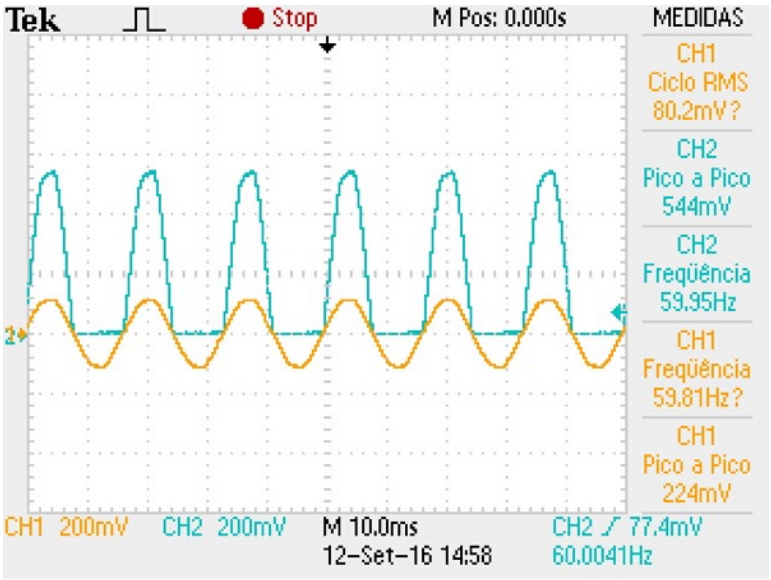


Figura 34 - Estágio de amplificação do sinal do sensor de corrente

Fonte: Autor

Na implementação do filtro RC passa-baixas, realizou-se anteriormente uma simulação para verificar se os requisitos de projeto foram alcançados, como atenuação mínima na frequência mais importante - 60Hz. Na Figura 35 está o filtro implementado e na Figura 36 as respostas no domínio da frequência.

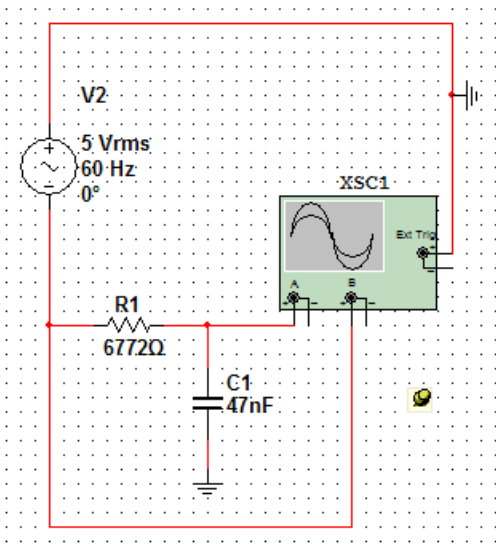


Figura 35 - Esquemático do filtro *anti-aliasing*

Fonte: Autor

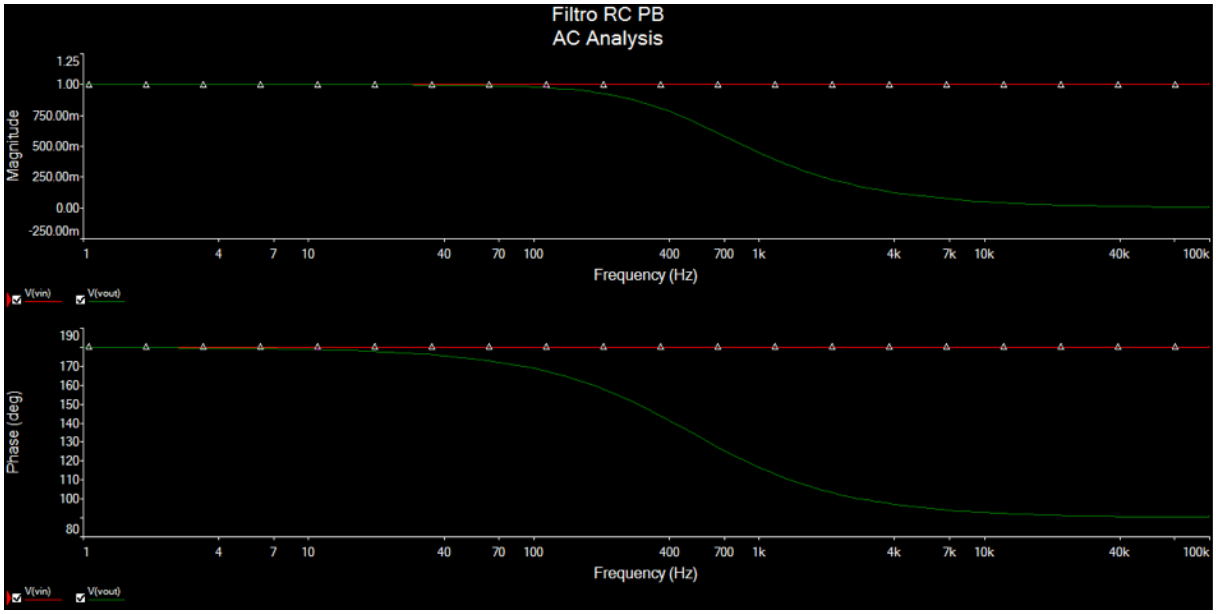


Figura 36 - Análise em frequência do filtro *anti-aliasing*

Fonte: Autor

A comunicação serial funcionou conforme estipulado em projeto, com envio e leitura dentro do esperado. Quando se envia, por exemplo, uma requisição de informação, representada pelo comando ‘R’, sem carga ligada e com a temperatura ambiente de 21ºC, a resposta da unidade sensora pode ser vista na Figura 37 abaixo:

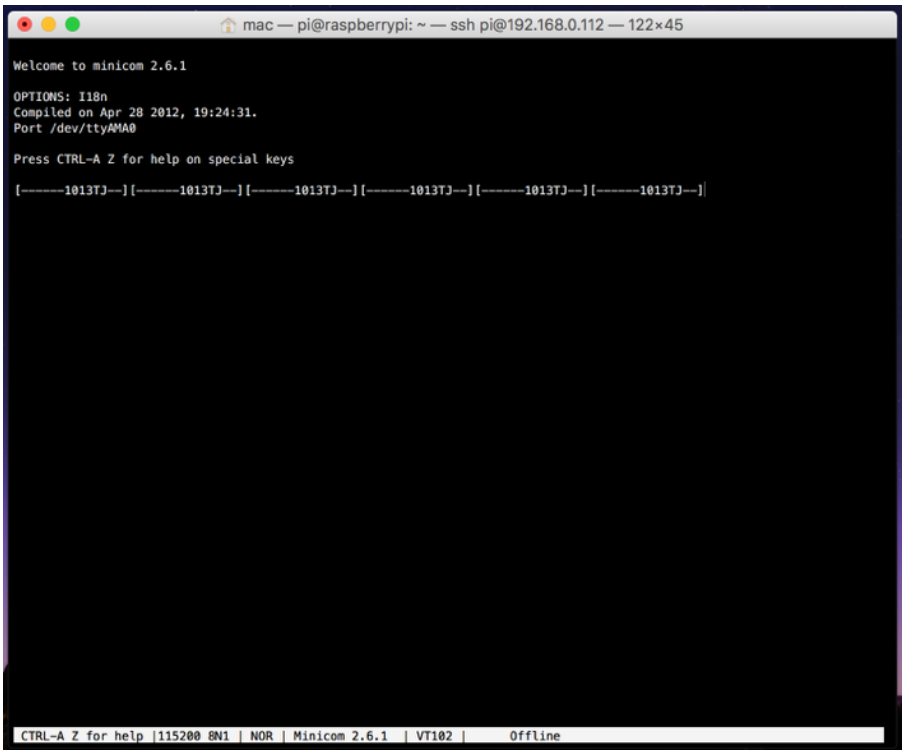


Figura 37 - Comunicação serial no modo verbose

Fonte: Autor

A mensagem segue claramente o padrão estipulado anteriormente. Os hífens são os campos que ainda não foram utilizados. Os dados estão em ASCII, e logo abaixo vê-se os dados traduzidos pela unidade central, como a temperatura J, que representa 74-48 = 26ºC. Essa diferença de temperatura ocorre porque o ambiente interno de uma residência conserva mais calor.

Para a aquisição da temperatura, para a utilização da técnica de *lookup-table* com as equações descritas anteriormente para o termistor, obteve-se os valores do Quadro 5 abaixo, que estão implementadas em `temperat.c`.

Conversão ADC10	Temperatura (ºC)	Conversão ADC10	Temperatura (ºC)
603	18,053	840	33,015
619	19,029	855	34,040
635	20,006	870	35,080
652	21,045	884	36,066
668	22,026	898	37,068
685	23,072	911	38,013
701	24,061	925	39,050
717	25,056	938	40,031
733	26,058	951	41,030
749	27,067	964	42,049
764	28,022	976	43,008
780	29,050	989	44,070
796	30,089	1001	45,072
810	31,009	1013	46,097
826	32,072		

Quadro 5 - Lookup table

Fonte: Autor

Com o Code Composer Studio, foi possível acompanhar o tempo de cada tarefa. Esses também estavam dentro do estipulado de 210us por ciclo de conversão, com a tarefa de comunicação ocupando o maior tempo de processamento, como vê-se abaixo:

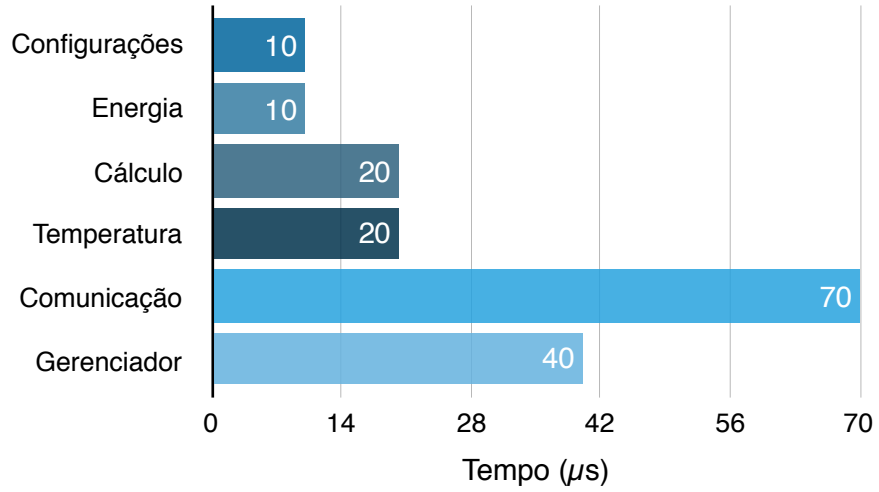


Gráfico 4 - Tempos das tarefas do firmware

Fonte: Code Composer Studio 6.1

O tempo total de uma execução de tarefas é de, em média, cerca de 170μs, respeitando o limite de 210μs proposto. Abaixo, na Figura 38, pode-se ver o tempo de uma execução.

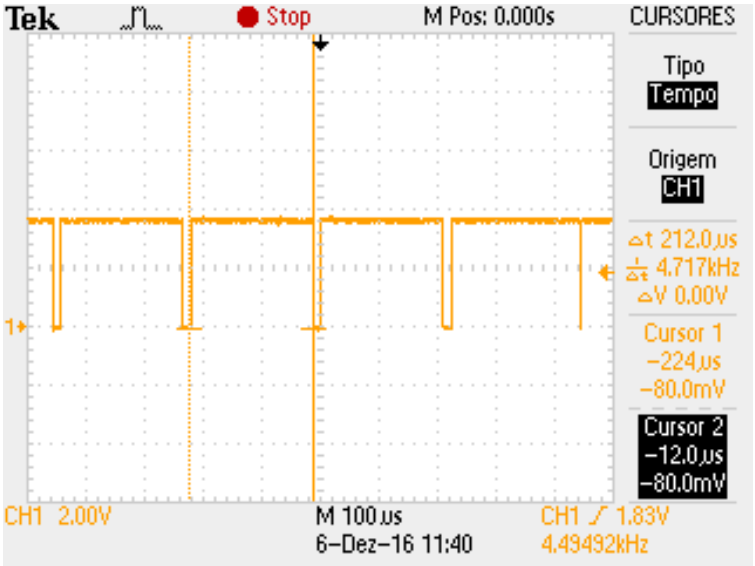


Figura 38 - Tempo das tarefas do firmware

Fonte: Autor

Para a execução de 79 ciclos, ou seja, realizadas todas as conversões, obteve-se a frequência de cerca de 60Hz. Com isso, todas as tarefas foram realizadas dentro do tempo estabelecido. A Figura 39 mostra os tempos coletados.

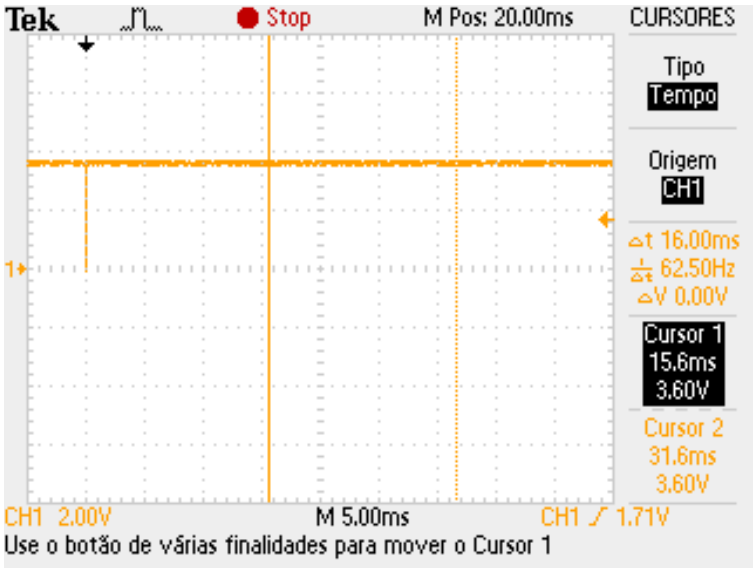


Figura 39 - Tempo para todas as conversões

Fonte: Autor

Nos Gráficos 5, 6 e 7, mostra-se os valores calculados pela unidade sensora para 3 diferentes cargas.

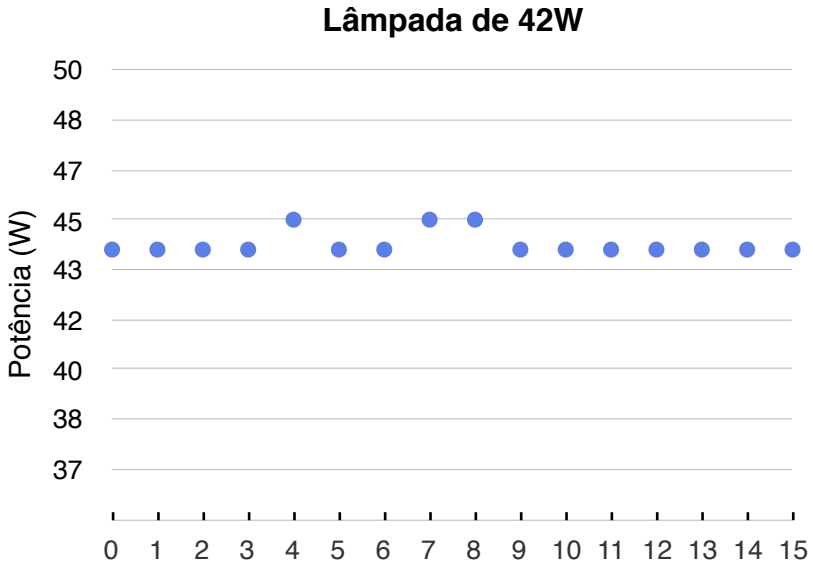


Gráfico 5 - Carga de teste de 42W

Fonte: Code Composer Studio

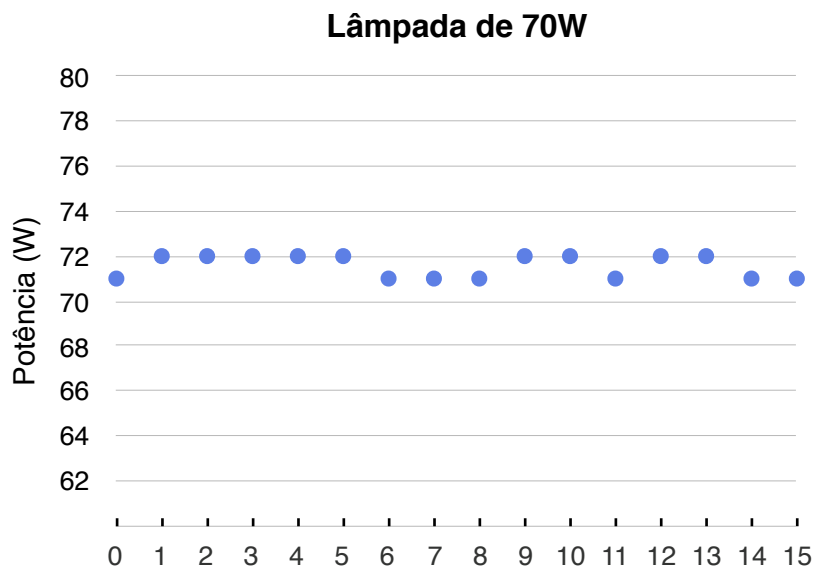


Gráfico 6 - Carga de teste de 70W

Fonte: Code Composer Studio

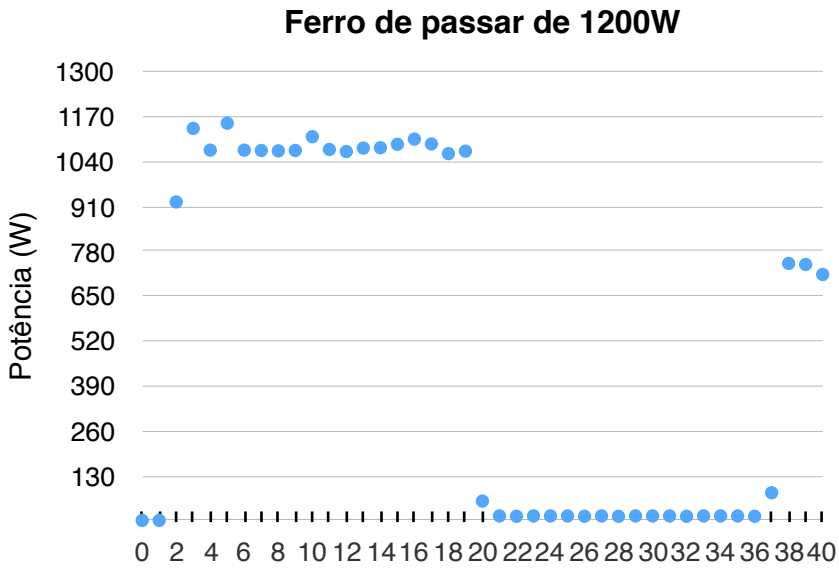


Gráfico 7 - Carga de teste de 1200W

Fonte: Code Composer Studio

As médias e erros calculados para esses testes são mostradas no Quadro 7 abaixo.

Potência nominal (W)	Pico de potência aferida (W)	Erro absoluto (%)
42	45,0	6,667
70	72,0	2,778
1200	1174,0	2,215

Quadro 6 - Comparativo entre potência real e potência calculada

Fonte: Autor

É interessante notar que, devido aos métodos utilizados para processar os dados advindos do sensor no microcontrolador, quanto maior a potência do equipamento a ser medido, menor será o erro absoluto.

4.2 UNIDADE CENTRAL

Neste dispositivo, obteve-se sucesso na instalação do S.O. e nas configurações de rede, possibilitando a instalação dos diversos pacotes que dão corpo à comunicação com unidade sensora e servidor. As movimentações dentro do banco, executadas pelo *script central.sh* cumpriram com os objetivos estabelecidos e, devido à estrutura modular dos códigos, futuras modificações podem ser implementadas facilmente. Também obteve-se sucesso em colocar o banco de dados no servidor.

Com o script em funcionamento no Raspberry Pi, a unidade central vai enviar continuamente o comando 'R', de forma que a resposta da unidade sensora pode ser vista na Figura 40 abaixo. Quando a central envia "Pronto!", então o processo se repete.

```
pi@raspberrypi ~ $ sudo bash central.sh
UNIDADE CENTRAL Activity Simulation
Device is ready!
Reading serial port [...]
Message: [------1000TJ--
1 000 26
0 Wtt
Pronto!
Device is ready!
Reading serial port [...]
Message: [------1000TJ--
1 000 26
0 Wtt
Pronto!
Device is ready!
Reading serial port [...]
Message: [------1000TJ--
1 000 26
0 Wtt
Pronto!
Device is ready!
Reading serial port [...]
Message: [------1000TJ--
1 000 26
0 Wtt
Pronto!
```

Figura 40 - Unidade central em modo verbose

Fonte: Autor

4.3 APLICATIVO

O servidor respondeu bem ao sistema. A estrutura do banco de dados foi implementada tranquilamente com os comandos básicos do MySQL como CREATE TABLE, e ele pode ser acessado online, como mostra a Figura 41. As tabelas propostas, embora sem grande complexidade de estrutura, foram suficientes para receber os dados e alimentar todas as aplicações que dependem dela.

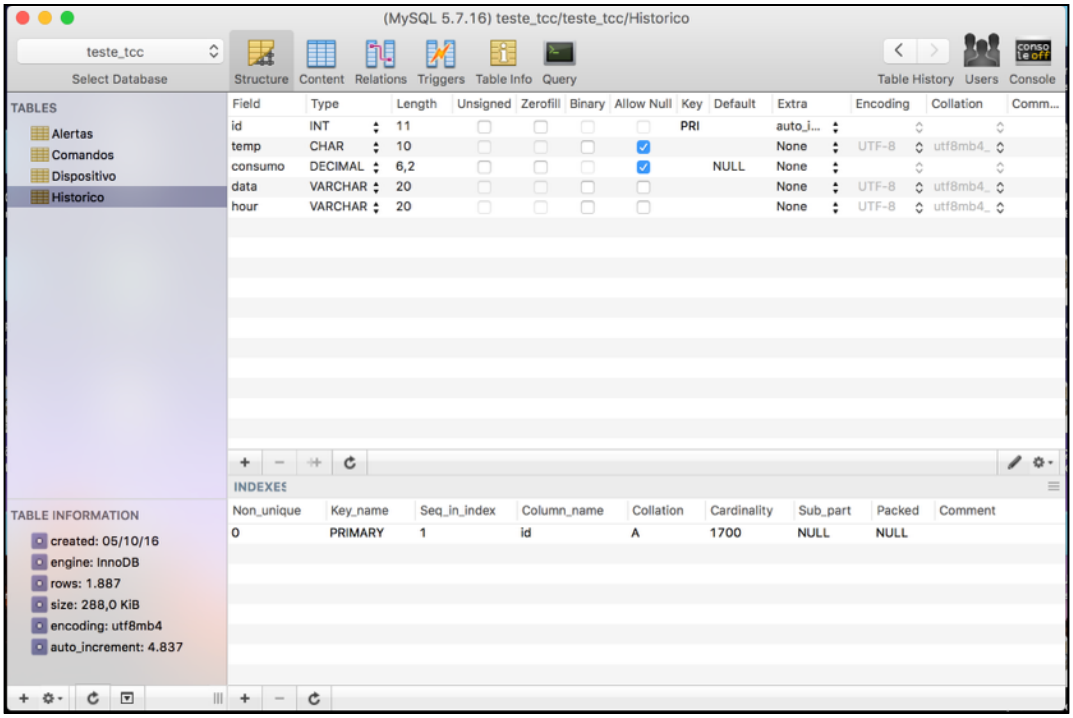


Figura 41 - Acesso ao banco de dados e suas tabelas
Fonte: Autor

Embora o servidor tenha se encaixado bem no projeto, o fato de ser um produto sem fins lucrativos para testes de protótipos faz com que a instabilidade esteja presente com certa frequência. Experimentou-se extrema lentidão durante alguns horários específicos do dia e existem oscilações de disponibilidade.

Para o aplicativo que comandará o sistema, construído em JAVA, foi possível respeitar com grande precisão o desenho proposto anteriormente, como mostra a Figura 42 abaixo.

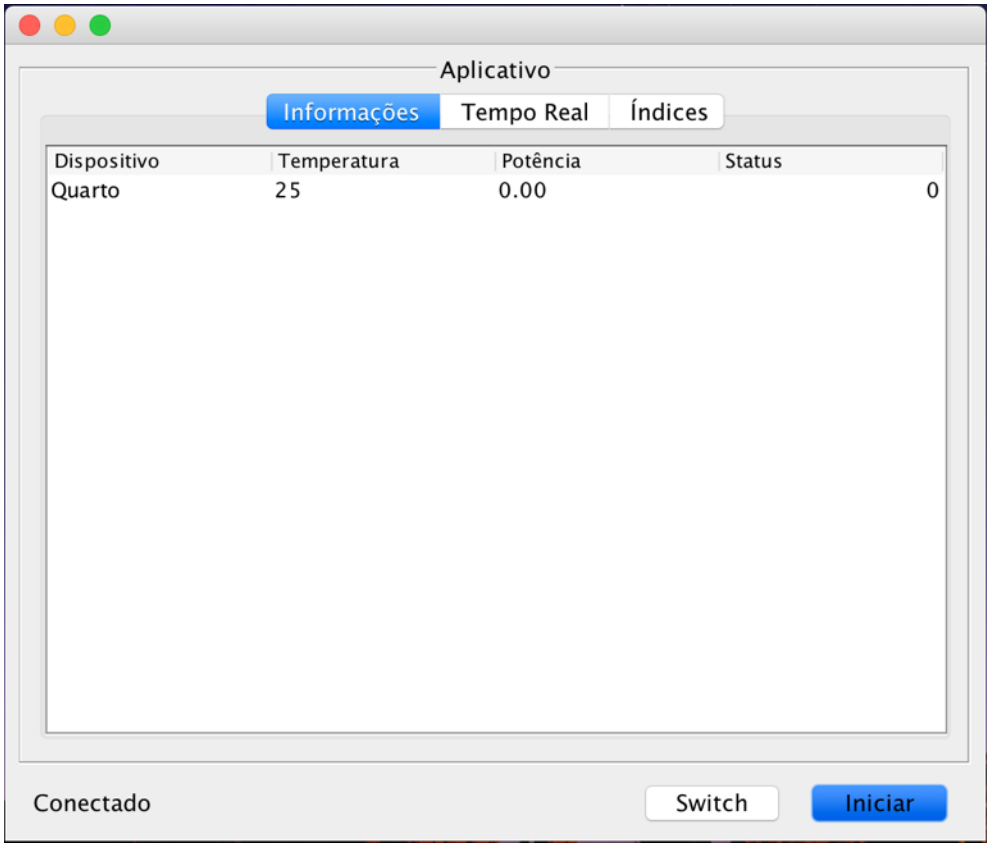


Figura 42 - Interface do aplicativo: Aba Informações
Fonte: Autor

Um dos grandes resultados alcançados diz respeito à maneira com que todas as abas são atualizadas simultaneamente. Duas threads, `atualiza_screen()` e `atualiza_graphs()`, chamadas na ação associada ao botão de INICIAR do aplicativo, são responsáveis por, respectivamente, atualizar a aba “Informações” a cada 100ms e atualizar todos os gráficos a cada 250ms. Mais detalhes pode ser vistos no Apêndice IV.

Na aba “Tempo Real”, onde são exibidas as informações em tempo real, a Figura 42 mostra como o aplicativo exibe os dados do servidor. Na aba “Índices”, são exibidas as informações das últimas 24h, além da energia consumida até o momento e pico de potência, como mostra a Figura 43 abaixo.

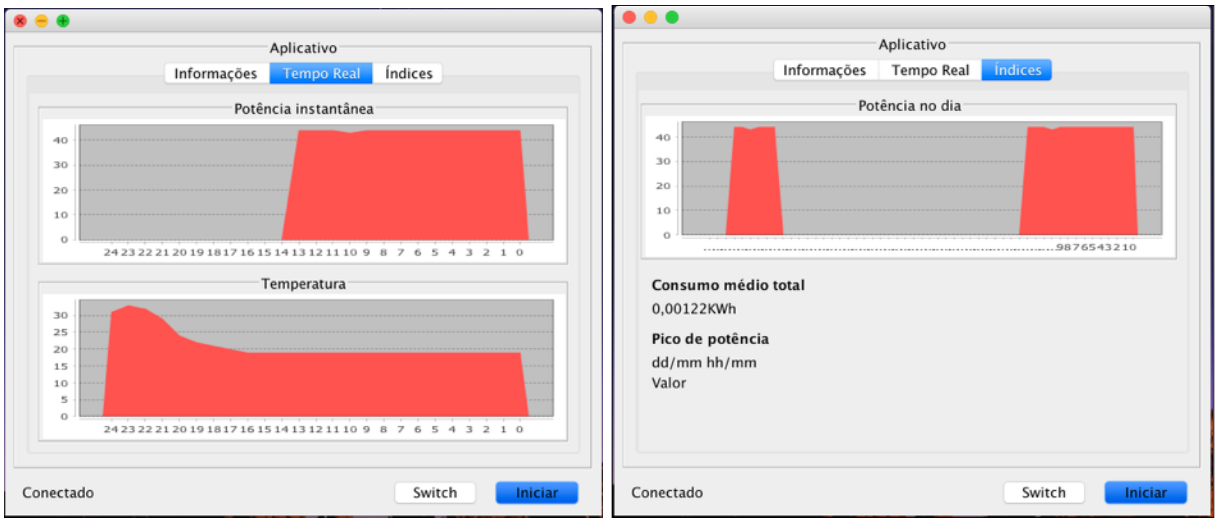


Figura 43 - Interface de gráficos do aplicativo

Fonte: Autor

5 CONSIDERAÇÕES FINAIS

Analisando-se os resultados, pode-se dizer que os objetivos propostos foram alcançados. Todas as medições são realizadas nos tempos corretos, respeitando o teorema da amostragem de Nyquist–Shannon e, embora existam erros pequenos na mediação de energia e temperatura, para o propósito do protótipo são toleráveis.

O problema no pequeno erro nas medições pontuais é que o fator acumulativo dos cálculos para o consumo de energia amplificam esses erros, fazendo com que, com bastante tempo, eles sejam significativos. Por isso, é muito importante aprimorar os métodos e circuitos para a máxima acurácia.

Mais especificamente no contexto da unidade sensora, entre as maiores dificuldades estava englobar todas as tarefas propostas no tempo limite da onda da rede elétrica. O interessante na arquitetura do firmware desenvolvido está no fato de ela ainda suportar outras atividades; ele se torna escalável em certa forma.

Outra dificuldade encontrada na unidade sensora diz respeito à construção de diversas estruturas de hardware que deveriam estar em sintonia. Desde o circuito de acionamento da carga pelo relé, que não poderia permitir que uma corrente parasita do indutor pudesse ocasionar algum dano ao microcontrolador, até os circuitos de amplificação do sinal do sensor.

Com relação a comunicação sem fio utilizando-se o módulo Xbee, devido ao

Também observou-se dificuldades em criar um sistema com maior precisão devido ao fato de existirem muitos processos ocorrendo simultaneamente e o microcontrolador não possuir FPU para cálculos de ponto flutuante. Isso levou ao desenvolvimento com simplificações como a divisão por deslocamento de bits que, embora nos fizesse ganhar tempo no processo de cálculo, trazia alguma perda de informação.

Condizente à unidade central, a maior dificuldade residiu no desenvolvimento do script para acessar o banco de dados e realizar a comunicação serial com a unidade sensora. Muitas versões foram desenvolvidas até se alcançar o algoritmo que funcionasse.

De um modo geral, o trabalho proposto possui uma gigantesca variedade de outros temas de projeto que podem derivar deste. Em primeiro lugar, otimizar os circuitos para filtros anti-aliasing mais complexos, ADC mais precisos e PCIs de menor dimensão. O maior desenvolvimento desses circuitos levaria o sistema a um nível maior de detalhes da rede, como componentes harmônicas e fator de potência. No quesito comunicação, avançar mais um passo utilizando RFs mais baratos e com alcance maior.

Em relação ao servidor, buscar outras formas de interagir com o banco de dados que sejam mais rápidas, atém de refinar o algoritmo da unidade central para realizar o menor número possível de transações com o servidor. Outro ponto que poderia ser intensamente explorado está na estruturação do banco de dados para um sistema domótico, com relações mais elaboradas entre as tabelas e técnicas que objetivem a segurança dos dados e do controle das unidades sensoras.

Também há campo extenso de melhorias em segurança computacional, tema bastante discutido devido ao aumento de dispositivos inteligentes no mercado. A utilização de melhores e mais baratos meios de comunicação sem fio, por exemplo, necessitaria de encriptação dos dados trocados entre unidade sensora e unidade central. Outro ponto é criar um banco de dados específico para cadastro de usuários, de forma que login e senhas sejam requeridas para acesso ao sistema.

No contexto atual e relevante da Inteligência artificial, existe também espaço para a utilização da grande quantidade de dados colhidos para identificar padrões, realizar ações baseando-se no conhecimento extraído desse bando de dados e até mesmo estimar o consumo utilizando técnicas de regressão.

REFERÊNCIAS

ALASDAIR, ALLAN. Build a Compact 4 Node Raspberry Pi Cluster. Disponível em: <http://makezine.com/projects/build-a-compact-4-node-raspberry-pi-cluster/>. Acesso em 07 de novembro de 2016.

APPLE. AppleScript Overview. Disponível em: <https://developer.apple.com/library/mac/documentation/AppleScript/Conceptual/AppleScriptX/AppleScriptX.html#//apple_ref/doc/uid/10000156-BCICHGIE>. Acesso em 27 de janeiro de 2016.

ANEEL. Atlas de Energia Elétrica no Brasil. Disponível em: <http://www.aneel.gov.br/arquivos/PDF/atlas3ed.pdf>. Acesso em 12 de maio de 2015.

ATZORI, Luigi. The Internet of Things: a survey. Disponível em: <http://www.elsevier.com/__data/assets/pdf_file/0010/187831/The-Internet-of-Things.pdf>. Acesso em 29 de março de 2015.

BARI, Nima. Internet of Things as a Methodological Concept. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6602039>. Acesso em 29 de março de 2015.

BARTZ, Robert J. CWTS: Certified Wireless Technology Specialist Official Study Guide. 2ª edição. Indiana: Sybex, 2012. 600 páginas.

CAMBRIDGE UNIVERSITY. Image Pi – Basic image processing. Disponível em: <https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/image-processing/>. Acesso em 07 de novembro de 2016.

CATHERM. MF52 Pearl-Shaped Precision NTC Thermistor for Temperature Measurement. Disponível em <http://www.cantherm.com/media/productPDF/cantherm_mf52_1.pdf>. Acesso em 19 de janeiro de 2016.

COMPANHIA ENERGÉTICA RESOL. Matriz Energética Mundial. Disponível em: <https://www.repsol.com/pt_pt/corporacion/conocer-repsol/contexto-energetico/matriz-energetica-mundial/>. Acesso em 9 de março de 2015.

CETIC.br. Pesquisa TIC Domicílios e Empresas em 2012. Disponível em: <http://cetic.br/media/docs/publicacoes/2/tic-domicilios-e-empresas-2012.pdf>. Acesso em 29 de março de 2015.

DIGI INTERNATIONAL INC. XBee®/XBee-PRO® RF Modules. Disponível em: <https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf>. Acesso em 24 de janeiro de 2016.

EMPRESA BRASIL DE COMUNICAÇÃO S/A. Consumo de energia sobe em residências e comércio mas cai na indústria. Disponível em: <http://www.ebc.com.br/noticias/economia/2014/12/consumo-de-energia-sobe-em-residencias-e-comercio-mas-cai-na-industria>. Acesso em: 13 de maio de 2015.

EPE. Anuário estatístico de energia elétrica 2013. Disponível em <http://www.epe.gov.br/AnuarioEstatisticodeEnergiaEletrica/20130909_1.pdf>. Acesso em 27 de maio de 2015.

FINDER. O que você precisa saber sobre relés? Disponível em: <http://www.findernet.com/en/node/47658>. Acesso em 19 de janeiro de 2016.

FORBES. Apple and Google dominate 'Internet of Things' influence with home automation Efforts. Disponível em: <http://www.forbes.com/sites/brucerogers/2014/07/08/apple-and-google-dominate-internet-of-things-influence-with-home-automation-efforts/>. Acesso em: 13 de maio de 2015.

HARBOR RESEARCH. IoT in the News: China Invests Heavily in the Internet of Things. Disponível em: <http://harborresearch.com/iot-in-the-news-china-invests-heavily-in-the-iot/>. Acesso em 9 de março de 2015.

HU, Shushan. Connected intelligent home based on the Internet of Things. Disponível em: <http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6617476&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6617476>. Acesso em 9 de março de 2015.

IBM. Fazendo o Watson trabalhar. Disponível em: <http://www-03.ibm.com/systems/br/power/advantages/watson/>. Acesso em 19 de março de 2016.

PURCELL, Lee. Intel: The Past, Present and Future of IoT. Disponível em:<https://software.intel.com/pt-br/articles/the-past-present-and-future-of-iot>. Acesso em 19 de março de 2016.

KAMAL, Raj. Embedded Systems 2E. 2ª edição. Nova Delhi: Tata McGraw-Hill Publishing Company Limited, 2011. 681 páginas.

KICKSTARTER. Edyn: Welcome to the connected garden. Disponível em: <https://www.kickstarter.com/projects/edyn/edyn-welcome-to-the-connected-garden?ref=category>. Acesso em 9 de março de 2015.

KRAVETS, David. Internet of Things to be used as spy tool by governments: US intel chief. Disponível em: <http://arstechnica.com/tech-policy/2016/02/us-intelligence-chief-says-iot-climate-change-add-to-global-instability/>. Acesso em 19 de março de 2016.

MALVINO, Albert. Eletrônica: volume 2 / Albert Malvino, David J. Bates. 7ª edição. Porto Alegre: AMGH, 2007. 556 páginas; 28cm.

MIORI, Vittorio. Domotic evolution towards the IoT. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6844739>. Acesso em 13 de maio de 2015.

MINISTÉRIO DAS COMUNICAÇÕES. Matéria: “Bernardo: Crescimento da internet móvel 'salta aos olhos'”. Disponível em: <http://www.mc.gov.br/sala-de-imprensa/todas-as-noticias/institucionais/30310-crescimento-da-internet-movel-salta-aos-olhos-afirma-bernardo>. Acesso em 9 de março de 2015.

MINISTÉRIO DE MINAS E ENERGIA. Balanço Energético Nacional. Disponível em: <https://ben.epe.gov.br/downloads/S%C3%ADntese%20do%20Relatório%20Final_2014_Web.pdf>. Acesso em 9 de março de 2015.

MIT TECHNOLOGY REVIEW. Where Speech Recognition Is Going. Disponível em: <http://www.technologyreview.com/news/427793/where-speech-recognition-is-going/>. Acesso em 9 de março de 2015.

MOORE, Gordon E. Cramming more components onto integrated circuits. Disponível em: <http://www.monolithic3d.com/uploads/6/0/5/5/6055488/gordon_moore_1965_article.pdf>. Acesso em 9 de março de 2015.

MOTA FILHO, João Eriberto. Descobrindo o Linux: entenda o sistema operacional GNU/Linux. 3ª edição revisada e ampliada. São Paulo: Novatec, 2012. 928 páginas.

ORACLE. Packages. Disponível em: < http://docs.oracle.com/javase/7/docs/api/java/sql/package-summary.html > Acesso em: 26 de janeiro de 2016.

PORTAL ENERGIA ENERGIAS RENOVÁVEIS. Fontes de energia renováveis e não renováveis. Disponível em: <http://www.portal-energia.com/fontes-de-energia/>. Acesso em 9 de março de 2015.

STARTUPI. Internet das Coisas: expectativas e desafios em 2015. Disponível em: <http://startupi.com.br/2015/01/internet-das-coisas-expectativas-e-desafios-em-2015/>. Acesso em 22 de março de 2015.

UNITED NATIONS FOUNDATION. Improving Energy Efficiency. Disponível em: <http://www.unfoundation.org/what-we-do/issues/energy-and-climate/improving-energy-efficiency.html>. Acesso em 9 de março de 2015.

ROCHOL, Juergen. Comunicação de Dados: Série Livros Didáticos Informática UFRGS. 1ª edição. São Paulo: Bookman, 2011. 25 cm.

RASPBERRY FOUNDATION. RASPBERRY PI 1 MODEL B. Disponível em <https://www.raspberrypi.org/products/model-b/>. Acesso em: 26 de janeiro de 2016.

SECHI, Francesco. Design of a Distributed Embedded System for Domotic Applications. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4669267>. Acesso em 13 de maio de 2015.

STALLINGS, William. Segurança de Computadores: princípios e práticas. 2ª edição. Rio de Janeiro: Elsevier, 2014. 28 cm.

SANYOU, Relays. Miniature Power Relay. Disponível em: <http://www.sanyourelay.ca/public/products/pdf/SRD.pdf>. Acesso em: 8 de julho de 2015.

TANENBAUM, Andrew S. Sistemas operacionais modernos. 3ª edição. São Paulo: Pearson Prentice Hall, 2010. 27,5 cm.

TESLA MOTORS. An Evolution in automobile engineering. Disponível em: <http://www.teslamotors.com/models> Acesso em: 13 de maio de 2015.

TEXAS INSTRUMENTS. LMx24-N LM2902-N LOW-POWER, QUAD-OPERACIONAL AMPLIFIERS. Disponível em: <http://www.ti.com/lit/ds/symlink/lm2902-n.pdf>. Acesso em 18 de setembro de 2016.

TEXAS INSTRUMENTS. MSP430x2xx Family User's Guide. Disponível em: <http://www.ti.com/lit/ug/slau144j/slau144j.pdf>. Acesso em 12 de maio de 2015.

TEXAS INSTRUMENTS. The Internet of Things: Opportunities & Challenges. Disponível em: <http://www.ti.com/ww/en/internet_of_things/pdf/14-09-17-IoTforCap.pdf>. Acesso em 22 de março de 2015.

APÊNDICE I - UNIDADE SENSORA: MAIN

```
#include <msp430.h>
#include "config.h"
#include "energia.h"
#include "temperatura.h"
#include "serial.h"

//-------- Variáveis do sistema ----------------------------//
unsigned int volatile flag_210 = 0;
unsigned int volatile processo = 0;
//flags para estados
unsigned int volatile f_e = 1, f_t = 1, f_ld = 1, f_en = 1;
char mensagem[TAMAN_MENSAGEM];

//-------- Variáveis de tempo ------------------------------//
unsigned int volatile TIME_us = 0;
unsigned int volatile TIME_ms = 0;

//-------- Variáveis de Energia ----------------------------//
unsigned int volatile num_amostras = 0;
unsigned int volatile conv_corrente = 0;
unsigned int volatile conv_tensao = 0;
unsigned int volatile vxi = 0;
unsigned int volatile potencia = 0;
unsigned int volatile buffer_potencia = 0;
unsigned int volatile consumo = 0;

//-------- Variáveis de Temperatura ----------------------//
unsigned int volatile conv_temp = 0;
unsigned int T = 0;

//-------- Variáveis para Switch --------------------------//
// 0 - desligado | 1 - ligado
unsigned int volatile liga_desl = 1;

//-------- Variáveis para transmissão --------------------//
unsigned int volatile requis_central = 0;

void main(void)
{
    int state = 1;
    start_config();

    while(1)
    {
        switch(state)
        {
            case 1:

                if(liga_desl&&f_e)                          //Medidor de
                    consumo ligado
                {
                    f_e = 0;
                    state = 2;
                }

                else if((TIME_ms%500==0)&&f_t)              //Temperatura a
                    cada 500ms
                {
                    f_t = 0;
                    state = 4;
```

```
    }

    else if((requis_central==2)&&f_ld)
    {
        f_ld = 0;
        state = 7;
    }

    else if((requis_central==1)&&f_en)
    {
        f_en = 0;
        state = 6;
    }

    else if(num_amostras>0)                     //Primeira
        conversão não sincroniza
    {
        f_e = 1; f_t = 1; f_ld = 1; f_en = 1;
        state = 5;
    }

break;

case 2:

    converte_energia();
    state = 3;

break;

case 3:

    elimina_ruido();
    potencia_inst();
    state = 1;

break;

case 4:

    converte_temperatura();
    state = 1;

break;

case 5:

    flag_210 = 0;
    if(TIME_ms>=1000)                           //Passou 1s
    {
        TIME_ms = 0;
    }

    while(flag_210!=1);                         //Sincroniza
        FSM para 210us a cada ciclo

    if(num_amostras==(NUM_AMOSTRAS-1))
    {
        while(TIME_us<16660);       //Sincroniza em 16,660ms
        potencia_media();
```

```
                buffer_potencia = potencia;                 //
                    buffer_potencia (última conversão)
                num_amostras = 0;
                potencia = 0;
                vxi = 0;
                clear_time();
            }

            state = 1;

        break;

        case 6:

            envia_bytes();
            state = 1;

        break;

        case 7:

            TOG(P1OUT,BIT0);                            //Liga/Desliga
                Relé
            if(liga_desl==0)
            {
                liga_desl=1;
            }
            else
            {
                liga_desl=0;
                buffer_potencia=0;                      //Desligado não há
                    gasto de energia
            }

            requis_central = 1;
            state = 6;

        break;
      }
    }
}
```

APÊNDICE II - UNIDADE SENSORA: CONFIGURAÇÕES INICIAIS

```
#include <msp430.h>
#include "config.h"

void config_clock(void)
{
    WDTCTL = WDTPW + WDTHOLD;
    DCOCTL = CALDCO_16MHZ;
    BCSCTL1 = CALBC1_16MHZ;
    BCSCTL2 = DIVS_0;
    BCSCTL3 = XCAP_3;

    while(BCSCTL3 & LFXT1OF);
    __enable_interrupt();
}

void config_PORT(void)
{
    P1DIR = BIT0 + BIT3 + BIT4;
    P1OUT = 0x00;
    P2DIR = 0xFF;
    P2OUT = 0x00;
}

void config_TA0(void)
{
    TACCTL0 = CCIE;
    TACCR0 = 160;
    TA0CTL = TASSEL_2 + ID_0 + MC_1 + TACLR;
}

void config_ADC(void)
{
    //Gerador ref ON | 2.5V ref | ADC10 ON | 64xADC_clock
    ADC10CTL0 = SREF_1 + REFON + REF2_5V + ADC10ON + ADC10SHT_3;
    ADC10CTL1 = INCH_6 + ADC10SSEL_3;
    ADC10AE0 = BIT6;
}

void config_UART9600(void)
{
    P1SEL |= BIT1 + BIT2;
    P1SEL2 |= BIT1 + BIT2;
    UCA0CTL1 = UCSWRST;
    UCA0CTL1 = UCSSEL_2;
    UCA0BR0 = 0x68;
    UCA0BR1 = 0x00;
    UCA0MCTL = UCOS16 + UCBRF_3;
    UCA0CTL1 &= ~UCSWRST;
    UC0IE |= UCA0RXIE;
}

void start_config(void)
{
    config_clock();
    config_PORT();
    config_ADC();
    config_UART9600();
    SET(P1OUT,BIT0);
    config_TA0();
}
```

APÊNDICE III - UNIDADE SENSORA: ENERGIA

```
#include <msp430.h>
#include "energia.h"
#include "config.h"

void potencia_inst(void)
{
    vxi = (conv_corrente);
    potencia = (potencia + vxi);
}

void potencia_media(void)
{
    //Somatório das potências instantâneas 5-220 6-127
    potencia = (potencia>>5);
}

void energia(void)
{
    consumo = consumo + potencia*(TIME_ms);
}

void elimina_ruido(void)
{
    if(conv_corrente<=10)
    {
        conv_corrente = 0;
    }

    if(conv_tensao<=10)
    {
        conv_tensao = 0;
    }
}

void converte_energia(void)
{
    SET_ADC_A6;                             //Troca canal para Corrente A6
    ADC10CTL0 |= ENC + ADC10SC;             //Habilita e inicia conversão
    while(ADC10CTL1 & BUSY);                //Espera a conversão
    conv_corrente = ADC10MEM;               //Guarda valor em conv_corrente
    ADC10CTL0&=~ENC;                        //Desabilita conversão

    SET_ADC_A7;                             //Troca canal para Tensão A7
    ADC10CTL0 |= ENC + ADC10SC;             //Habilita e inicia conversão
    while(ADC10CTL1 & BUSY);                //Espera a conversão
    conv_tensao = ADC10MEM;                 //Guarda valor em conv_corrente
    ADC10CTL0&=~ENC;                        //Desabilita conversão

    num_amostras++;
}
```

APÊNDICE IV - UNIDADE SENSORA: TEMPERATURA

```
#include <msp430.h>
#include "temperatura.h"
#include "config.h"

void temperatura(void)
{
    if(conv_temp<=603)
    {
        T = 18;
    }
    else if(conv_temp<=619)
    {
        T = 19;
    }
    else if(...)
    {
        [...]
    }
    else if(conv_temp<=1001)
    {
        T = 45;
    }
}

void converte_temperatura(void)
{
    SET_ADC_A5;
    ADC10CTL0 |= ENC + ADC10SC;
    while(ADC10CTL1 & BUSY);
    conv_temp = ADC10MEM;
    ADC10CTL0&=~ENC;

    temperatura();

    if(T>=30)
    {
        SET(P1OUT,BIT4);
    }
    else CLR(P1OUT,BIT4);
}
```

APÊNDICE V - UNIDADE SENSORA: COMUNICAÇÃO

```
#include <msp430.h>
#include "serial.h"
#include "config.h"

void bloco_mensagem(void)
{
    //| S0 | N2 N1 N0 | A2 A1 A0 | ST0 | C2 C1 C0 | T1 T0 | L1 L0 | E0
    //| START | NUMERAC√O | ENDERECO | STATUS | CONSUMO | TEMPERATURA | LIVRE
        | END |

    //Zera para que a converão anterior não apareca caso a carga seja
        desligada
    potencia_b = 0; potencia_a = 0; aux = 0;

    //Start S0
    mensagem[0] = '[';
    mensagem[1] = '-';
    mensagem[2] = '-';
    mensagem[3] = '-';
    mensagem[4] = '-';
    mensagem[5] = '-';
    mensagem[6] = '-';
    //Status da carga (Lig/Des) ST0
    mensagem[7] = liga_desl + 48;
    //Potencia C2 C1 C0
    trata_num_potencia();
    mensagem[8] = potencia_b + 48;
    mensagem[9] = potencia_a + 48;
    mensagem[10] = aux2 + 48;
    //Temperatura T1 T0
    mensagem[11] = 'T';
    mensagem[12] = T + 48;
    mensagem[13] = '-';
    mensagem[14] = '-';
    //Fim de mensagem E0
    mensagem[15] = ']';
}

void envia_bytes(void)
{
    unsigned int i = 0;
    bloco_mensagem();
    while((processo<(TAMAN_MENSAGEM/BLOCO_MENSAGEM))&&(i<BLOCO_MENSAGEM))
    {
        UCA0TXBUF = mensagem[i+processo*BLOCO_MENSAGEM];
        i++;
        while(!(IFG2&UCA0TXIFG));           //Aguarda buffer TX estar
            preparado para nova transmisão
    }

    processo++;

    if(processo>=(TAMAN_MENSAGEM/BLOCO_MENSAGEM))
    {
        processo = 0;
        requis_central = 0;                 //Finalizou a transmissão de toda
            a mensagem
    }
}
```

APÊNDICE VI - APLICAÇÃO: APP JAVA

```
public class conexaoMySQL
{
    private Connection connection;

    private int id;
    private String disp;
    private String temp;
    private float potencia_media;
    private float consumo_medio;
    private float tempo_hora;
    private String consumo;
    public String[] potencia_instant;
    public String[] potencia_total;
    public String[] temp_total;
    public String[] date;
    private int status;
    private int num_disp;
    public static final int NUM_AMOSTRAS_LIVE = 25;
    public int NUM_AMOSTRAS_MES;

    conexaoMySQL()
    {
        potencia_instant = new String[NUM_AMOSTRAS_LIVE];
        temp_total = new String[NUM_AMOSTRAS_LIVE];
    }

    public void conectar() throws SQLException
    {
        try
        {
            Class.forName("com.mysql.jdbc.Driver");

    } catch (ClassNotFoundException e)
        {
            System.out.println("Driver MySQL JDBC não encontrado.");
            return;
    }

    public void consulta_ultimas_potencias() throws SQLException
    {
        int i = 0;
        Statement st = connection.createStatement();
        //Consulta os últimos NUM_AMOSTRAS_LIVE registros em ordem decrescente
        ResultSet res = st.executeQuery("SELECT consumo FROM Historico ORDER
            BY id DESC LIMIT " + NUM_AMOSTRAS_LIVE);

        while (res.next())
        {
            potencia_instant[i] = res.getString("consumo");
            i++;
        }
    }

     public void consulta_datahora(int rowcount) throws SQLException,
         ParseException
    {
        int i=0;
        float dif_tempo_seg=0;
        date = new String[rowcount];
        Statement st = connection.createStatement();
```

```
    //Consulta os últimos NUM_AMOSTRAS_LIVE registros em ordem decrescente
    ResultSet res = st.executeQuery("SELECT hour FROM Historico ORDER BY
        id DESC LIMIT " + rowcount);

    SimpleDateFormat formater = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss"
        );
    }

    long d1, d2;
    while (res.next())
    {
        // the mysql insert statement
        date[i] = res.getString("hour");
        i++;
    }
        // create the mysql insert preparedstatement
        PreparedStatement preparedStmt = connection.prepareStatement(query);
    i=0;
        preparedStmt.execute();
    while (i<rowcount-1)
    {
        if(Float.parseFloat(potencia_total[i])>0)
    public void atualiza_disp() throws SQLException
    {   {
        String query = "UPDATE Dispositivo SET status = '" + String.valueOf
            d1=formater.parse(date[i].getTime();
            (status) + "' WHERE id=" + String.valueOf(id) + "';";
            d2=formater.parse(date[i+1].getTime();
            dif_tempo_seg = Math.abs((d2-d1)/(1000));
            tempo_hora = tempo_hora + dif_tempo_seg;
        PreparedStatement preparedStmt = connection.prepareStatement(query);
        }
        preparedStmt.execute();
    }
        i++;
    }
    public void insere_Alertas(String mensagem) throws SQLException
    {
    //Consulta para "INSERT INTO Alertas (disp,mensagem) VALUES (?,?)";
    tempo_hora = tempo_hora/(60*60);
        PreparedStatement preparedStmt = connection.prepareStatement(query);
}       preparedStmt.setString (1, disp);
        preparedStmt.setString (2, mensagem);
public void consulta_potencia_geral() throws SQLException, ParseException
{       preparedStmt.execute();
    }
    int i = 0, rowcount = 0;
    Statement st = connection.createStatement();
    public void insere_Dispositivo(String disp) throws SQLException
    {
    //Consulta os últimos NUM_AMOSTRAS_CONSUMO registros em ordem
        decrescente
        String query = "INSERT INTO Dispositivo (status,temp,consumo,disp)
    ResultSet res = st.executeQuery("SELECT consumo FROM Historico ORDER
            VALUES (?,?,?,?)";
        BY id DESC LIMIT 2000");
        PreparedStatement preparedStmt = connection.prepareStatement(query);
        preparedStmt.setInt(1, 1);
        preparedStmt.setString (2, "0");
    //Conta número de linhas da consulta
        preparedStmt.setString (3, "0");
    if (res.last())
    {   preparedStmt.setString (4, disp);
        preparedStmt.execute();
        rowcount = res.getRow();
    }   res.beforeFirst(); // not rs.first() because the rs.next() below
}           will move on, missing the first element
    }

    potencia_total = new String[rowcount];
    while (res.next())
    {
        potencia_total[i] = res.getString("consumo");
        i++;

    }

    consulta_datahora(rowcount);
    calcula_consumo_kwh();
}
```

```
public void calcula_consumo_kwh()
{
    potencia_media = 0;
    consumo_medio = 0;
    int i=0, j=0;

    while(i<potencia_total.length)
    {
        //Se o cunsumo é maior que 0, então calcula a média
        if(Float.parseFloat(potencia_total[i])>0)
        {
            potencia_media = potencia_media + Float.parseFloat
                (potencia_total[i]);
            consumo_medio = consumo_medio + Float.parseFloat
                (potencia_total[i])*tempo_hora/1000;
            System.out.println("Consumo KWh: " + consumo_medio);
            j++;
            //Consumo em KWh = (tempo*Potencia/1000)
        }

        i++;
    }

    consumo_medio = (consumo_medio/j);
    potencia_media = (potencia_media/j);
    System.out.println("Consumo KWh: " + consumo_medio + " | " + "Potencia
        Wtt: " + potencia_media);
}

public void consulta_temperatura() throws SQLException
{
    int i = 0;
    Statement st = connection.createStatement();
    //Consulta os últimos NUM_AMOSTRAS_CONSUMO registros em ordem
        decrescente
    ResultSet res = st.executeQuery("SELECT temp FROM Historico ORDER BY
        id DESC LIMIT " + NUM_AMOSTRAS_LIVE);

    while (res.next())
    {
        temp_total[i] = res.getString("temp");
        i++;
    }
}

public int read_dispositivos() throws SQLException
{
    Statement st = connection.createStatement();
    ResultSet res = st.executeQuery("SELECT * FROM Dispositivo");

    while (res.next())
    {
        id = res.getInt("id");
        disp = res.getString("disp");
        temp = res.getString("temp");
        status = res.getInt("status");
        consumo = res.getString("consumo");
    }
```

APÊNDICE VII - APLICAÇÃO: APP EM APPLESCRIPT

```
do shell script "sh /Users/mac/Desktop/App/config.sh"

repeat while (flag > 0)

    set num_alertas to do shell script "sh /Users/mac/Desktop/App/requisit.sh"
        -- verifica alertas
    set num_alertas to text -1 thru -1 of num_alertas

    if num_alertas > 0 then

        liga_desliga()

        -- Acessa a tabela Alertas para recolher e filtrar a mensagem
        set disp to do shell script "sh /Users/mac/Desktop/App/read_disp.sh"
        set disp to text -1 thru -10 of disp

        set disp to text 1 thru ((offset of "@" in disp) - 1) of disp

        set alerta to do shell script "sh /Users/mac/Desktop/App/
            read_alerta.sh"
        set temp to text -6 thru -2 of alerta
        set consumo to text -3 thru -3 of alerta
        set alerta to text -1 thru -1 of alerta

        if (alerta is "1") then

            set alerta to 1
            display notification disp & " ligado(a)"
            beep 1

        else if (alerta is "0") then

            set alerta to 0
            display notification disp & " desligado(a)"

        end if

        delay 1.5

        if (temp > 40) then

            display notification disp & " está muito quente!"

        end if

        delay 1.5

        if (consumo = "A") then

            display notification disp & " consumindo muito!"

        end if

        set num_alertas to (num_alertas - 1)
        do shell script "sh /Users/mac/Desktop/App/remove_alerta.sh"
        temp = -1

    end if

end repeat
```

APÊNDICE VIII - UNIDADE CENTRAL: SHELL SCRIPT

```
#!/bin/bash
#Variáveis
set OneByte=0
set potencia_a=0
set potencia_b=0
set potencia_c=0
set potencia=0
set temperatura=0
set status=0
set cmd_exist=0

main()
{
    #Configuração da porta serial
    stty -f /dev/cu.uart-73FF4A7AB74A3E3C raw ispeed 9600
    echo Device is ready!

    #Demilitador do fim de mensagem
    delim=$']'

    #Comando R para leitura das informações da Unidade Sensora
    echo Reading serial port [...]
    echo R > /dev/cu.uart-73FF4A7AB74A3E3C
    read -r -d $delim -n 32 -t 1 OneByte < /dev/cu.uart-73FF4A7AB74A3E3C

    #Mensagem recebida pela UART da Unidade Sensora
    echo Mensagem: $OneByte

    #Mineirando da string enviada os valores de Status, Potencia e Temperatura
    status=${OneByte:7:1}
    potencia_c=${OneByte:8:1}
    potencia_b=${OneByte:9:1}
    potencia_a=${OneByte:10:1}
    temperatura=${OneByte:12:1}

    #ASCII to char
    char="$status"
    status=$( printf "%d" "'${char}" )
    char="$potencia_c"
    potencia_c=$( printf "%d" "'${char}" )
    char="$potencia_b"
    potencia_b=$( printf "%d" "'${char}" )
    char="$potencia_a"
    potencia_a=$( printf "%d" "'${char}" )
    char="$temperatura"
    temperatura=$( printf "%d" "'${char}" )

    #Converte ASCII char em Decimal para enviar ao banco de dados
    status=$(echo "$status - 48"|bc)
    potencia_c=$(echo "$potencia_c - 48"|bc)
    potencia_b=$(echo "$potencia_b - 48"|bc)
    potencia_a=$(echo "$potencia_a - 48"|bc)
    temperatura=$(echo "$temperatura - 48"|bc)
}

sql_update_dispositivo()
{
    mysql -h db4free.net -P 3306 -u user_root -puser_root <<EOF
    USE teste_tcc;
    UPDATE Dispositivo SET status=$status WHERE id=4;
```

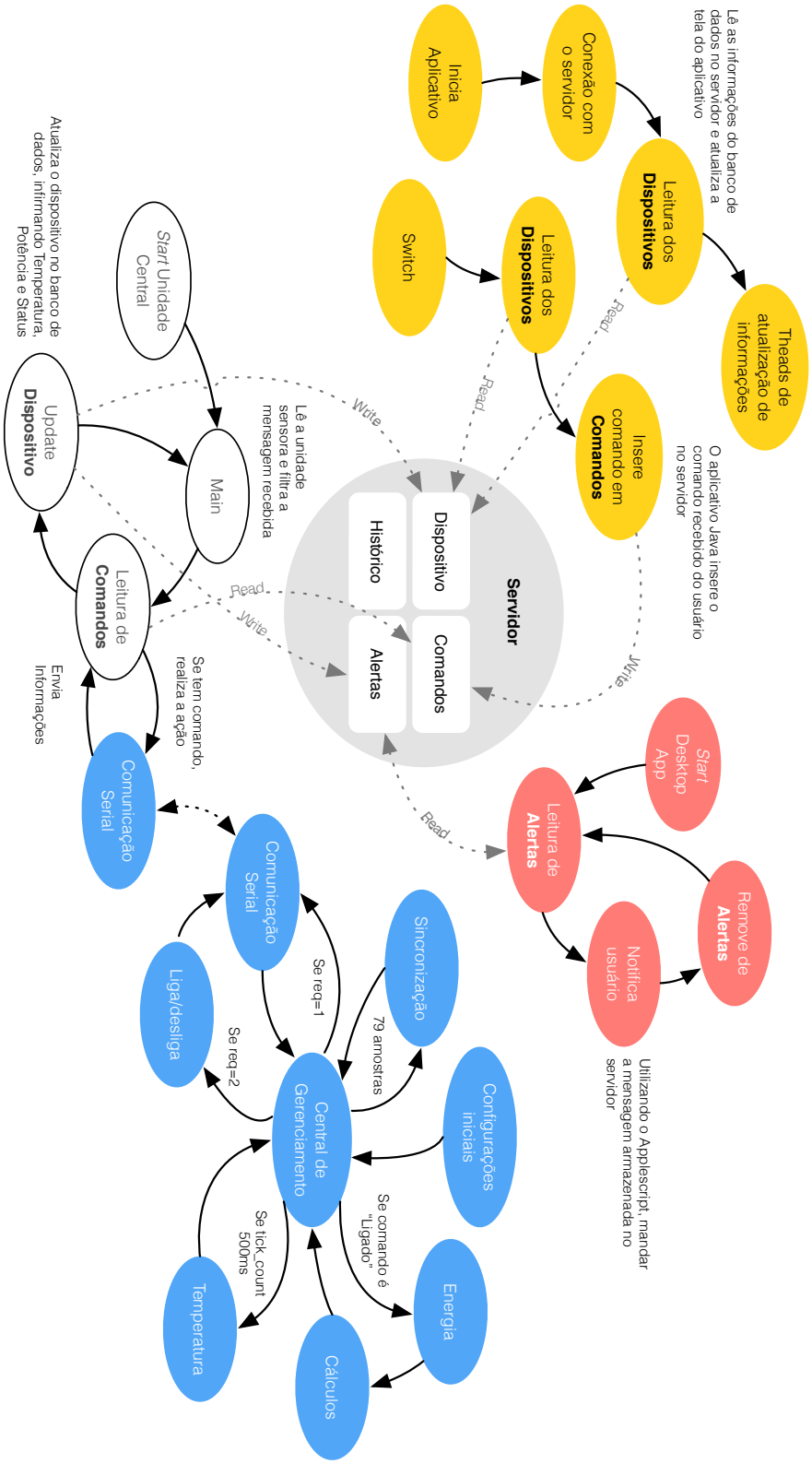
APÊNDICE IX - UNIDADE CENTRAL: CONFIGURAÇÃO DO BANCO DE DADOS

```
config_server()
{
    mysql -h db4free.net -P 3306 -u user_root -puser_root <<EOF

    USE teste_tcc;
    CREATE TABLE IF NOT EXISTS Comandos (id INT AUTO_INCREMENT PRIMARY KEY NOT
        NULL, cmd CHAR(10) NOT NULL, data CHAR(10));
    CREATE TABLE IF NOT EXISTS Alertas (id INT AUTO_INCREMENT PRIMARY KEY,
        mensagem CHAR(10) NOT NULL, disp CHAR(10) NOT NULL);
    CREATE TABLE IF NOT EXISTS Dispositivo (id INT AUTO_INCREMENT PRIMARY KEY
        NOT NULL, status INT NOT NULL, temp INT(3) NOT NULL, consumo DECIMAL(6,
        2) NOT NULL, disp CHAR(10) NOT NULL);
    CREATE TABLE IF NOT EXISTS Historico (id INT AUTO_INCREMENT PRIMARY KEY
        NOT NULL,temp CHAR(10) NOT NULL, consumo DECIMAL(6,2) NOT NULL, data
        VARCHAR(20) NOT NULL, hour VARCHAR(20) NOT NULL)

EOF
}
```

APÊNDICE X - FLUXOGRAMA GERAL



APÊNDICE XI - ESQUEMÁTICO DO SISTEMA