UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
BACHARELADO EM ENGENHARIA ELETRÔNICA

FÁBIO HENRIQUE FUDOLI

DESENVOLVIMENTO DE UMA REDE DE SENSORES SEM FIO APLICADA A DOMÓTICA E INTERNET DAS COISAS

TRABALHO DE CONCLUSÃO DE CURSO

CAMPO MOURÃO
2017

FÁBIO HENRIQUE FUDOLI

Desenvolvimento de uma Rede de Sensores Sem Fio Aplicada a Domótica e Internet das Coisas

Trabalho de Conclusão de Curso de Graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2 - TCC2, do curso superior de Engenharia Eletrônica do Departamento Acadêmico de Eletrônica (DAELN) da Universidade Tecnológica Federal do Paraná (UTFPR), como requisito para obtenção do título de Engenheiro em Eletrônica.

Orientador: Prof. Dr. Márcio R. da Cunha
Coorientador: Prof. Msc. Paulo D. G. da Luz

CAMPO MOURÃO
2017

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Campo Mourão
Coordenação do Curso de Engenharia Eletrônica

TERMO DE APROVAÇÃO

DO TRABALHO DE CONCLUSÃO DE CURSO INTITULADO

Desenvolvimento de uma Rede de Sensores Sem Fio Aplicada a Domótica e Internet das Coisas

por

Fábio Henrique Fudoli

Trabalho de Conclusão de Curso apresentado no dia 20 de Junho de 2017 ao Curso Superior de Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná, Campus Campo Mourão. O Candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____

Prof.ª Aline Rocha Leão
(UTFPR)

_____

Prof. Jakson Paulo Bonaldo
(UTFPR)

_____

Prof. Márcio Rodrigues da Cunha
(UTFPR)
Orientador

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

Agradecimentos

Primeiramente, agradeço aos meus pais Sandra Fudoli e Nelson Fudoli que me criaram e educaram com dedicação e incentivo, pelo apoio emocional e financeiro que tornaram essa jornada possível e dedico esta conquista a eles. A minha namorada Danielle Veron que me acompanha já a oito anos, seis destes durante o curso, me dando apoio e me aguentando nos momentos difíceis, sendo compreensiva e atenciosa. Aos meus avós Cecília Moura e Edson Moura pelo apoio constante, me ajudando em tudo que precisei.

Aos meus amigos e colegas que me acompanharam e me ajudaram, as vezes me "carregando" e também sendo "carregados". Especialmente aos meus companheiros de república, Vitor Hungria, Caike Albertin e Yago Pessoa que tornaram o convívio diário uma experiência agradável, seja nos estudos, nos jogos ou nas tarefas e Lucas Blessa que me acompanhou e ajudou na maior parte das matérias e trabalhos.

Aos professores do departamento de eletrônica, especialmente aos professores Paulo da Luz pelo incentivo na área de eletrônica embarcada, pela oportunidade no projeto de extensão e pela idealização inicial deste trabalho, ao professor Moacyr Brito pela orientação e oportunidade no projeto de iniciação científica e ao professor Márcio Cunha pela orientação e suporte nesta etapa final.

"Para se ter sucesso, é necessário amar de verdade o que se faz. Caso contrário, levando em conta apenas o lado racional, você simplesmente desiste. É o que acontece com a maioria das pessoas."

(Steven Paul Jobs)

Resumo

FUDOLI, Fábio H. **Desenvolvimento de uma Rede de Sensores Sem Fio Aplicada a Domótica e Internet das Coisas**. 2016. Trabalho de Conclusão de Curso - Engenharia Eletrônica. Universidade Tecnológica Federal do Paraná. Campo Mourão, 2016.

O presente trabalho de conclusão de curso apresenta o estado da arte a cerca das principais tecnologias de automação residencial disponíveis e o desenvolvimento do projeto de uma rede de sensores sem fio aplicada a domótica e internet das coisas. Utilizando a plataforma Arduino, o módulo NRF24L01 e o ESP8266, baseado em estudos realizados, foi possível implementar uma rede capaz de obter dados de um ambiente através de sensores de umidade, temperatura e luminosidade, bem como deixá-los disponíveis para consulta via Web pela plataforma ThingSpeak e também tomar ações baseadas nos valores destes dados. A rede de sensores sem fio implementada neste trabalho é constituída de 3 nós sensores, sendo estes um que faz as leituras de umidade e temperatura, outro que faz a leitura da luminosidade do ambiente e o último fazendo um acionamento, baseado em uma dessas leituras. O *gateway* desta rede recebe os dados dos nós, os interpreta e disponibiliza-os na nuvem, através da plataforma ThingSpeak. O protótipo deste *gateway* foi implementado em placa de circuito impresso.

Palavras-chave: Domótica. IoT. Arduino. WSN.

Abstract

FUDOLI, Fábio H. **Development of a Wireless Sensor Network Applied to Home Automation and Internet of Things** . 2016. Term Paper - Electronic Engineering. Federal Technological University of Paraná. Campo Mourão, 2016.

This term paper presents the state of the art about the main residential automation technologies available and the development of the design of a wireless sensor network applied to home automation and internet of things. Using the Arduino platform, module NRF24L01 and ESP8266, it was possible to implement a network capable of obtaining data from an environment through humidity, temperature and luminosity sensors, as well as making them available for web consultation by ThingSpeak platform and also taking actions based on values of these data. The wireless sensor network implemented in this work consists of three sensor nodes, one of which is used to read humidity and temperature values, another to read the ambient light and the last to act based on one of these readings. The gateway of this network receives the data from the nodes, interprets them and makes them available in the cloud through the ThingSpeak platform. The prototype of this gateway was implemented on printed circuit board.

Keywords: Domotic. Iot. Arduino. WSN.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ALU	Arithmetic Logic Unit (Unidade Lógica Aritmética)
ASIC	Application Specific Integrated Circuits (circuitos integrados de aplicação especifica)
CPU	Central Process Unite (unidade central de processamento)
DIP	Dual In-line Package (encapsulamento em duas linhas)
DIP	Dual In-Line Package (Encapsulamento em linha dupla.)
EIB	European Installation Bus
FCC	Federal Communications Commission
FPGA	Field Programmable Gate Array (Arranjo de Portas Programáveis em Campo)
GFSK	Gaussian Frequency-shift Keying
GPIO	General Purpose Input/Output (Entrada/saida de uso geral)
GPU	Graphics Processing Unit (Unidade de Processamento Gráfico)
HDMI	High Definition Multimedia Interface (interface multimídia de alta definição)
I/O	Input Output (entrada e saída)
I2C	Inter-Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
IHC	Intelligent Home Control
IR	Infravermelho
LAN	Local Area Network
LON	Local Operating Network
MAC	Media Access Control
MIPS	Milhões de Instruções Por Segundo

MISO	Master Input Slave Output
MOSI	Master Output Slave Input
PLC	Power Line Carrier
PTH	Pin Through Hole (Pinos que atravessam a placa.)
PWM	Pulse Width Modulation (modulação por largura de pulso)
RAM	Random Access Memory (Memória de Acesso Aleatório)
RFID	Radio-Frequency Identification - Identificação por rádio-frequência
RISC	Reduced Instruction Set Computer (computador com conjunto de instruções reduzida)
ROM	Read Only Memory (Memória Somente Leitura)
SCL	Serial Clock Line
SCLK	Serial Clock
SDA	Serial Data Line
SDRAM	Synchronous Dynamic Random Access Memory (Memória de acesso aleatório dinâmico síncrono)
SIMTEEL	Simpósio de Tecnologia e Engenharia Eletrônica
SoC	System on Chip (sistema no chip)
SPI	Serial Peripheral Interface
SPI	Serial Peripheram Intergace
SPICE	Simulation Program with Integrated Circuit Emphasis
SS	Slave Select
UART	Universal Asynchronous Receive/Transmitter
USB	Universal Serial Bus (barramento serial universal)
Wi-FI	Wireless Fidelity

Sumário

1 Introdução

A palavra domótica resulta da junção de duas palavras: casa e robótica. Aliada ao aumento da qualidade de vida das pessoas, a domótica faz parte das diversas invenções da humanidade que auxiliam nesse quesito. Metas como melhorar o conforto, facilitar e automatizar tarefas do dia a dia, auxiliar pessoas fisicamente debilitadas a terem melhores condições de vida, bem como prover segurança e confiabilidade a seus usuários são tarefas da domótica.

As tecnologias já desenvolvidas e padronizadas em domótica, como por exemplo, os protocolos X10, EIB e IHC, foram criados em uma época onde a comunicação sem fio não era tão comum quanto é nos dias de hoje. Por conta disso, ficaram limitadas a apenas algumas classes sociais devido a seu alto custo de instalação e manutenção.

Outro objetivo importante da domótica é na gestão inteligente de água e energia elétrica. Por exemplo, sensores acoplados ao solo de um jardim verificam a umidade e temperatura, decidindo a melhor hora de iniciar uma irrigação e também a hora certa de parar. Em relação a energia elétrica, com uma interface homem-máquina acoplada a rede, é possível saber exatamente quais lâmpadas estão acesas e em quais cômodos da casa elas pertencem. Isso evita o desperdício, além de proporcionar uma maior sensação de controle.

> Por exemplo, num edifício inteligente, o sinal de um sensor de abertura de janela pode ser enviado simultaneamente ao sistema de alarme de intrusão, ao sistema de som, e ao sistema de climatização. O valor indicado pelo termômetro da temperatura do ar exterior pode ser enviado simultaneamente ao sistema de climatização, ao controle automático de estores, ao sistema de aquecimento da água da piscina, ao sistema de abertura e fecho da cobertura da piscina, e a um qualquer “display“ digital instalado em qualquer ponto do edifício (ALVES; MOTA, 2003).

Para Cook e Das (2004), ambientes inteligentes representam o próximo passo em construções, indústrias, casas e automação de sistemas de transporte. Como um organismo capaz de sentir, os ambientes inteligentes baseiam-se primeiramente em dados sensoriais do mundo real. Esses dados vem de diversos tipos de sensores que podem ser de temperatura, umidade, presença, luminosidade, entre outros que podem ser distribuídos em localidades diferentes de uma casa, por exemplo.

Segundo Alves e Mota (2003), são casas inteligentes as que possuem as características de tornar a vida mais simples. Algumas dessas características são:

- Segurança

- Economia
- Conforto
- Ecologia
- Integração

A tarefa de interligar todos esses sensores, atuadores e interfaces homem-máquina de maneira sem fio, é bastante complexa. Dado a quantidade possível desses sensores e atuadores de serem instalados, uma rede que comunique esses dispositivos entre si de maneira a não usar fios será desenvolvida ao longo desse trabalho.

Um dos compromissos desse trabalho é para com a conectividade, pois de nada adianta ter um sistema instalado em sua residência que fique fechado a novas tecnologias que vão surgindo ao longo do tempo. Edwards e Grinter (2001 apud RAMOS, 2015) afirmam que:

> Em sistemas domóticos os serviços são dinâmicos e a adição de novos dispositivos ao longo do tempo requer não somente que aplicações possam comunicar-se, mas também identificar/descobrir quais novos serviços foram adicionados e, quando necessário, estabelecer uma comunicação pertinente ao contexto desta nova aplicação. Além disso, se uma determinada aplicação não exerce o seu papel corretamente no sistema ela poderá afetar de forma inadequada os demais serviços. Ou seja, para que todos os serviços e tecnologias de uma residência possam interagir é necessária uma interoperabilidade tanto no nível sintático como semântico.

Atualmente, quando se fala em domótica, uma nova tendência vem interligada: a internet das coisas, popularmente conhecida como IoT. Segundo Gubbi et al. (2013), com a IoT, muito dos objetos do dia a dia estarão na rede de uma forma ou de outra. Identificação por rádio frequência (RFID) e tecnologias em redes de sensores vão surgir para encontrar esse novo desafio no qual sistemas de informação e comunicação estão inclusos no ambiente.

As redes de sensores ou (WSN - Wireless Sensor Network), com os recentes avanços da tecnologia em circuitos integrados de baixa potência e comunicações sem fio, tornaram viável e eficiente os dispositivos de sensoriamento remoto. A combinação destes fatores permite uma grande quantidade de sensores inteligentes que por sua vez coletam, processam, analisam e disseminam informações coletadas em uma série de ambientes diferentes.

1.1 Problemas e Premissas

Observando as soluções em domótica presentes no mercado, nota-se a variedade de produtos, funcionalidades e protocolos de comunicação adotados por cada fabricante. As

soluções atuais, em sua grande maioria, requerem que o consumidor faça novas instalações de cabos e fios, o que torna o processo de automação residencial mais custoso, já que além de comprar os dispositivos de um determinado fabricante, o consumidor deve se preocupar também com a instalação, seja ela através de fios e cabos e as vezes se preocupar até mesmo com a passagem de novos eletrodutos embutidos ou não.

Essas dificuldades contribuem para desestimular o mercado da domótica, tornando-o focado apenas em algumas classes e produzidos por poucos fabricantes. Uma solução para esse problema é desenvolver um sistema que funcione sem fios e que possa comunicar-se com seus periféricos através de vários protocolos de comunicação, utilizando de componentes de baixo custo para que possa ser repassado a valores mais acessíveis atingindo assim mais classes socioeconômicas.

2 Objetivos

Nesse capítulo serão descritos o objetivo geral e os objetivos específicos que esse trabalho pretende atingir.

2.1 Objetivo Geral

O presente trabalho tem como objetivo geral realizar um estudo teórico e prático no que se diz respeito a redes sem fio de sensores aplicados a domótica, bem como implementar de forma satisfatória um protótipo de uma rede de sensores.

2.2 Objetivos Específicos

São os objetivos específicos deste trabalho:

- Pesquisar livros, artigos, dissertações de mestrado, teses de doutorado, manuais que se relacionem ao tema aqui proposto;
- Efetuar uma breve pesquisa sobre outras alternativas sem fio, como por exemplo, WI-FI, *ZigBee* ou *XBee*.
- Fazer uma revisão sobre redes de malha ou redes *Mesh*;
- Realizar simulações em software SPICE dos circuitos eletrônicos relevantes do trabalho, quando possível;
- Buscar métodos, técnicas, materiais e componentes eletrônicos necessários para realização de testes a cerca deste trabalho e posteriormente apresentar resultados;
- Realizar uma pesquisa no sentido de estudar os protocolos de comunicação serial mais conhecidos, como por exemplo UART , SPI e I2C;
- Desenvolver um protocolo de comunicação próprio;
- Estabelecer a comunicação sem fio entre os módulos proposto por este trabalho;
- Verificar a conectividade do módulo proposto com outros periféricos a fim de certificar a possibilidade de comunicação via UART, SPI e I2C;
- Empregar a rede de sensores sem fio na IoT;

- Desenvolver um protótipo a fim de testes e análises de funcionamento com base nos objetivos esperados;
- Analisar os dados coletados.

2.3 Justificativa

As tecnologias atuais em domótica são pouco acessíveis, pouco conhecidas e quando adquiridas geram certo trabalho do consumidor quando a instalação é feita. Por isso, desenvolver um módulo que possa fazer uma comunicação sem fio com outros módulos e que se conecte com os periféricos através dos protocolos mais conhecidos é uma alternativa viável.

Em eventos realizados neste Campus como ExpoUT, Semana de Eletrônica e SIMTEEL (Simpósio de Tecnologia e Engenharia Eletrônica) foram apresentados em forma de artigo, banner e protótipo, alguns itens que levaram a elaboração dessa proposta de trabalho. Em algum desses eventos nos quais tiveram a ilustre visita da comunidade acadêmica e também da região através das escolas, foi possível notar o interesse da comunidade nas soluções apresentadas que foram apenas acender lâmpadas e controlar sua intensidade de maneira sem fio.

3 Estado da Arte

Neste capítulo será apresentada a revisão bibliográfica a cerca dos principais tópicos que estão incluídos no tema desse trabalho.

3.1 A Domótica

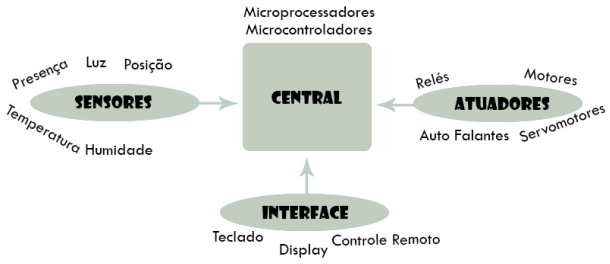
Domótica é a área que corresponde ao controle e automação de residências, proporcionando a seus habitantes experiências únicas em conforto e segurança. Sistemas de domótica podem interligar diversos sensores, como no tocante a segurança, soar alarmes de incêndio e detectar vazamento de gás ou até mesmo localizar intrusos no quintal e tomar medidas pré-programadas quando alguma dessas situações ocorrem.

A automação já é bem conhecida no ramo industrial, automatizando processos, tornando-os mais rápidos, confiáveis e mais eficazes. É natural que essa tecnologia chegasse as residências ao longo do tempo, simplificando as tarefas diárias e proporcionando mais segurança aos usuários.

> Ela é uma tecnologia de automação residencial recente que propicia a gestão de recursos habitacionais de forma integrada, controlando a residência remotamente, unindo as vantagens dos meios eletrônicos aos informáticos. Visa simplificar a vida diária das pessoas, poupando tempo com tarefas repetitivas, economizando custos diversos e aumentando o conforto e segurança (BOLZANI, 2004).

Exemplos de aplicação da domótica são controle de temperatura de ambientes com uso de sensores de temperatura internos e externos, controle de iluminação baseado em sensores de luminância e por relógio, irrigação de gramados baseados em sensores de umidade e também por relógio. No que diz respeito a segurança, podem ser destacados o controle de acesso a ambientes através de leitura de cartões RFID ou leitura biométrica. A Figura 1 mostra alguns dos sensores, atuadores e centrais que podem ser interligados num sistema de automação residencial.

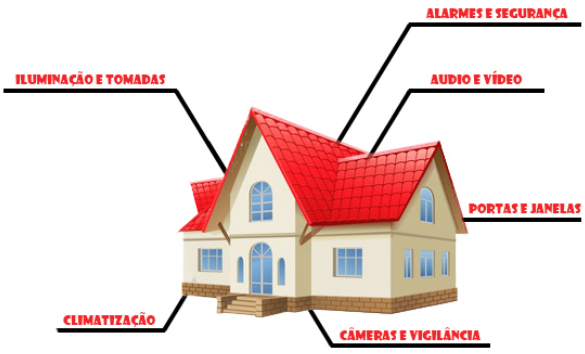
Figura 1 – Interligação de sensores e atuadores a uma central



Fonte: Autoria Própria.

E por fim, toda essa tecnologia deve oferecer certa conectividade e acessibilidade aos usuários nas residências. Com base nas tecnologias atuais, uma residência automatizada deve oferecer conectividade com celulares e computadores para uma boa experiência de comando e monitoramento. A Figura 2 exemplifica alguns dos recursos da domótica em uma residência comum.

Figura 2 – Recursos Controlados em Domótica.



Fonte: Autoria Própria.

Os recursos e comodidades vistos vêm a suprir algumas das necessidades da sociedade atual, como por exemplo, devido ao aumento da expectativa de vida da população brasileira, são mais comuns pessoas acamadas ou idosos que vivem sozinhos ou dependentes de outras pessoas para realizar tarefas simples do dia a dia. Um simples ato de desligar o interruptor de uma lâmpada para dormir pode se tornar uma complexa atividade para uma pessoa debilitada fisicamente. Um sistema de controle de iluminação por controle remoto por exemplo, auxiliaria essa pessoa.

Como afirma Werneck (1999):

> Depois de o público conhecer uma residência automatizada, não haverá como retroceder, toda a cadeia de concepção da moradia, (a arquitetura, a construção, etc.), evoluirá, e, principalmente, o ocupante do imóvel. Assim, deverão ser necessários vários profissionais que, interagindo, permitirão o real desenvolvimento das técnicas da domótica.

3.1.1 Tecnologias Atuais em Domótica

Antes de apresentar uma nova solução é necessário fazer um breve estudo das tecnologias existentes na área de domótica. Algumas delas já existem há cerca de quatro décadas. Vale ressaltar que muitas dessas tecnologias foram criadas em uma época onde comunicações sem fio limitavam-se a estações de rádio sem toda essa facilidade que existe atualmente com esse tipo de comunicação, como por exemplo, através de Wi-Fi ou Bluetooth.

Serão apresentadas a seguir quatro tecnologias que segundo Dias e Pizzolato (2004) são as que possuem destaque no mercado nacional e internacional.

3.1.1.1 O Sistema X10 – PLC (Power Line Carrier)

PLC, da sigla em inglês Power Line Carrier, ou em português "Portadora em Linha de Potência) nada mais é do que uma comunicação de dados via rede elétrica. Criada na Escócia na década de 70 e sua patente expirada em 1997, permitiu que muitas empresas utilizassem dessa tecnologia e fabricassem novos dispositivos. Sendo assim o protocolo de comunicação aplicado a domótica mais utilizado no mundo (ALVES; MOTA, 2003).

Aproveitando-se do fato de que praticamente todo cômodo de uma residência é abastecido por energia elétrica, o sistema X10 alcança dispositivos em praticamente toda a residência sem a necessidade de que sejam instalados novos cabos ou eletrodutos, evitando assim mais gastos com reformas.

O sistema X10, utiliza de uma frequência de portadora de 120 kHz para "carregar" as informações ao longo da rede elétrica. Os bits de informação são transmitidos somente quando a onda senoidal em 60 Hz cruza o zero. Essa é uma limitação do sistema X10 que não pode transmitir informações em grandes velocidades, não passa dos 60 bps. Outras limitações e complicações acontecem em residências que possuem instalações bifásicas ou trifásicas, sendo necessário replicar esses sinais para cada uma das fases. É necessário a instalação de filtros na entrada da rede residencial para evitar que os comandos dados na residência prossigam pela rede até a residência do vizinho por exemplo, acionando seus equipamentos de maneira indesejada.

3.1.1.2 O Sistema EIB (European Installation Bus)

O EIB, diferentemente do X10 foi criado já como um sistema livre, ou seja, não proprietário. Sua comunicação pode ocorrer por vias dedicadas (cabos ou fios) ou até mesmo pela rede elétrica, semelhante ao X10.

Utilizando da tecnologia de sistemas distribuídos, a tecnologia EIB possui seus dispositivos descentralizados e com tecnologia distribuída (ALVES; MOTA, 2003, p. 97),

portanto, permite que os dispositivos conectados à rede possam tanto comandar outros dispositivos como ao mesmo tempo serem comandados. Cada um dos dispositivos pode ter ou não seu próprio controle microcontrolado e logo pode tomar decisões baseadas em leitura de sensores ou opções pré-programadas.

Como o próprio nome diz, o sistema EIB é muito difundido na Europa, contudo, no Brasil, a única empresa que se tenha conhecimento de utilizar essa tecnologia é a Siemens que comercializa o sistema de nome Instabus (DIAS; PIZZOLATO, 2004).

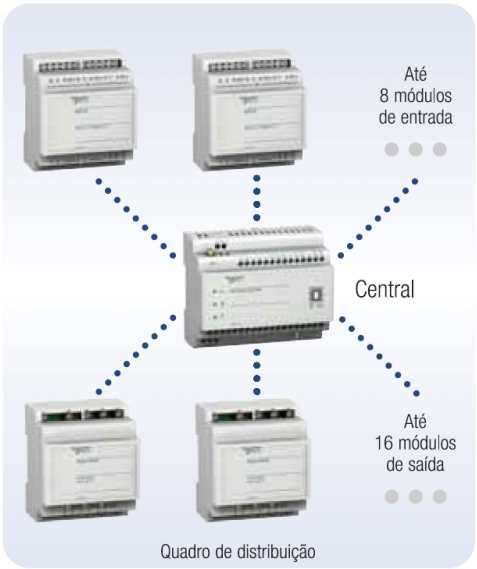
3.1.1.3 O Sistema IHC (Intelligent Home Control)

O IHC em contrapartida ao EIB, é um sistema centralizado, ou seja, todos os comandos e processamento de dados obtidos dos sensores ocorrem em uma única central. Essa tecnologia é proprietária, ou seja, pertence a uma única empresa.

Todas as suas funções podem ser programadas via software, fazendo com que os dispositivos possam ser controlados da maneira que o usuário desse sistema desejar. Esses dispositivos obedecem uma ideia modular, isto é, cada dispositivo no sistema possui uma função. Alguns desses módulos podem ser: módulo de controle, módulo de alimentação, módulo de E/S, módulo de comunicação, entre outros. Cada um desses módulos podem conter sensores, atuadores ou dispositivos de interface humana.

Como mostra a Figura 3 retirada do manual da Schneider Electric (2008), o sistema apresentado por eles suporta até 8 módulos de entrada e 16 módulos de saída interligados a uma central.

Figura 3 – Diagrama do IHC.



Fonte: Manual da Schneider Electric (2008).

3.1.1.4 O Sistema LonWorks

Semelhante ao sistema citado anteriormente de nome EIB, o LonWorks também é um sistema distribuído. Essa tecnologia depende diretamente de um chip de codinome Neuron, um protocolo de comunicação chamado LonTalk e a organização chamada LonMark que é responsável por padronizar as interfaces de equipamentos LonWorks.

As iniciais presente no nome desse sistema (LON) tem um significado pertinente as características do sistema. A sigla LON, do inglês (*Local Operating Network*) que significa rede local operante, com algumas características da conhecida LAN(*Local Area Network*), porém, não trabalha nos mesmos protocolos.

Redes LonWorks são constituídas por dispositivos de controle inteligente conhecidos como nós. Esses nós comunicam entre si através do protocolo já citado anteriormente, o LonTalk. Cada um desses nós possui um chip Neuron o que permite a implementação das funções desejadas.

Cada um desses nós ou dispositivo inteligente tem capacidade própria de processamento (sistema distribuído) e consegue enviar e receber dados diretamente de outro dispositivo sem precisar de um controlador central para intermediar o tráfego (CHERMONT, 2007).

Segundo (CUNHA, 2008 apud SILVA, 2014), as redes LonWorks podem trafegar pelas seguintes mídias:

- RF (Rádio Frequência)
- IR (Infravermelho)
- Cabo coaxial
- Cabo de par trançado
- Fibra ótica

E claro, cada tipo de mídia faz-se necessário o uso dos transceptores adequados que convertam os sinais para o meio físico desejado. Os tipos de mídia citados acima possuem vantagens e desvantagens principalmente no que se trata de velocidade e custo, portanto, deve-se escolher a alternativa que atende melhor esses dois quesitos.

Segundo Alves e Mota (2003):

> A Echelon, na sua topologia de rede LonWorks, tem desenvolvido aproximações que permitem maior flexibilidade com o intuito de obter um elevado grau de simplicidade na sua implementação, existindo mais de 700 empresas associadas a este *standard*.

3.2 As Soluções Sem Fio: *ZigBee* e *XBee*

Nesta seção serão introduzidos conceitos, características e especificações sobre essas duas populares tecnologias sem fio atuais denominadas *ZigBee* e *XBee*. Visto que essas tecnologias trabalham de forma semelhante a que será utilizada nesse trabalho, principalmente no que se diz a frequência de operação e topologias de rede.

3.2.1 O ZigBee

A aliança ZigBee é uma associação de companhias que trabalham juntas para desenvolver padrões e produtos confiáveis com baixo custo de implementação e baixa potência de operação (BARONTI et al., 2007).

O ZigBee é feito sobre o padrão IEEE 802.15.4 que define as camadas físicas e endereços MAC para baixo custo e baixas taxas de transmissão de dados para redes locais dedicadas. O padrão ZigBee permite sua associação em topologias do tipo estrela, árvore e *peer-to-peer* [1].

Suas aplicações são destinadas a dispositivos embarcados que permitem trabalhar com baixa taxas de comunicação de dados e consumindo pouca energia.

3.2.2 O *XBee*

Uma ramificação privada do *ZigBee* surgiu através do *XBee* para facilitar a implementação desse padrão. Com algumas melhorias, o *XBee* tornou os rádios *ZigBee* mais fáceis de serem utilizados em pequenos projetos embarcados. Os módulos *XBee* são testados e aprovados pela FCC (Federal Communications Commission, em português: Comissão Federal de Comunicações) e pela própria aliança *ZigBee* (DIGI INTERNATIONAL INC., 2009). A Tabela 1 mostra algumas características pertinentes do modulo XBee produzido pela Digi, como por exemplo, seu alcance e potência de transmissão.

[1] É uma arquitetura de redes de computadores onde cada um dos pontos ou nós da rede funciona tanto como cliente quanto como servidor, permitindo compartilhamentos de serviços e dados sem a necessidade de um servidor central (COULOURIS et al., 2012; TANENBAUM; STEEN, 2007, p. 424 e p. 47 respectivamente).

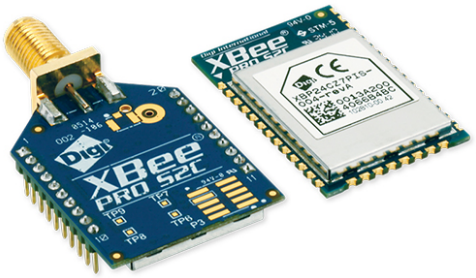
Tabela 1 – Algumas características do módulo Xbee.

Característica	Valor
Taxa de dados(Kbps)	250
Alcance interno(m)	60
Alcance externo (m)	1200
Potência de transmissão (mW)	3,1
Sensibilidade do rádio (dBm)	-100

Fonte: Adaptado de Digi International Inc. (2009).

Através da Figura 4 é possível observar o formato físico de um dos rádios *XBee* produzidos pela Digi na sua versão PRO. Essa versão possui características de desempenho melhores que as mostradas na Tabela 1 que correspondem a um módulo padrão.

Figura 4 – Módulo XBee PRO S2C.



Fonte: Digi International Inc. (2009).

3.3 As Redes *Mesh*

As redes *Mesh*, do inglês "malha", são redes sem fio, como o próprio nome já diz, em malha. Essas redes têm topologia e roteamento dinâmico e variável constituídas por nós. Essas redes evoluíram a partir das populares redes *ad-hoc* (ABELÉM et al., 2007).

Para Ramanathan e Redi (2002 apud LUIZ; PRZYBYSZ, 2007):

> Uma rede *ad-hoc*(possivelmente móvel) é um conjunto de dispositivos de rede que pretendem se comunicar, mas que não possuem infra-estrutura fixa disponível e não possuem organização pré-determinada de links de comunicação disponíveis. Os nós individuais da rede são responsáveis por descoberta dinâmica de quais são os outros nós que podem se comunicar diretamente a ele, ou seja, de quais são seus vizinhos (formando uma rede (*multi-hop*). Redes *ad-hoc* são escolhidas para serem usadas em situações onde a infra-estrutura não está disponível ou não é confiável, ou ainda em situações de emergência.

Essas redes como dito anteriormente, são auto-organizáveis e autoconfiguráveis. Seus nós dispostos em malha compõem uma rede *ad-hoc*. Em contrapartida as redes *ad-hoc*, a topologia *Mesh* possui uma localização fixa ao contrário do *ad-hoc* tradicional.

Os nós das redes *Mesh* podem ser de dois tipos: roteadores *Mesh* e clientes. Os roteadores são ocupados com múltiplas interfaces de comunicação de rede que possuam mais de uma tecnologia de acesso, provendo certa redundância de informação. Esses roteadores tem mínima mobilidade e seu agrupamento forma o backbone [2] das redes *Mesh*. Alguns desses roteadores podem fazer o papel de *gateway* e prover uma ponte entre a rede privada local com a internet, por exemplo (CABRAL; MATEUS, 2008).

Ainda para Cabral e Mateus (2008) :

> Há vários cenários de aplicação para as redes *Mesh*. Elas podem ser utilizadas para a implantação de redes domésticas, redes na vizinhança, redes corporativas e redes metropolitanas. A capacidade de auto-organização de uma rede *Mesh* reduz a complexidade de implantação e manutenção da rede.

Para Akyildiz e Wang (2005) essas características citadas acima são vantagens das redes *Mesh* perante a outros tipos de redes *ad-hoc*. Outras vantagens são a diminuição da potência desprendida em cada nó que diminui a necessidade de linha de visada direta e também gera mais robustez, confiabilidade e redundância devido à grande variedade de caminhos alternativos que a informação pode percorrer dentro da topologia de rede.

A Figura 5 mostra a topologia de uma rede *Mesh*. Um roteador *Mesh* situado em uma extremidade da topologia pode conectar-se a internet através de diferentes rotas. Essa tecnologia permite através dos nós, decidir qual é a melhor rota a ser percorrida pela informação nos diversos dispositivos interligados a rede. Nesse exemplo, os roteadores comunicam-se com a internet através de um roteador *gateway* único.

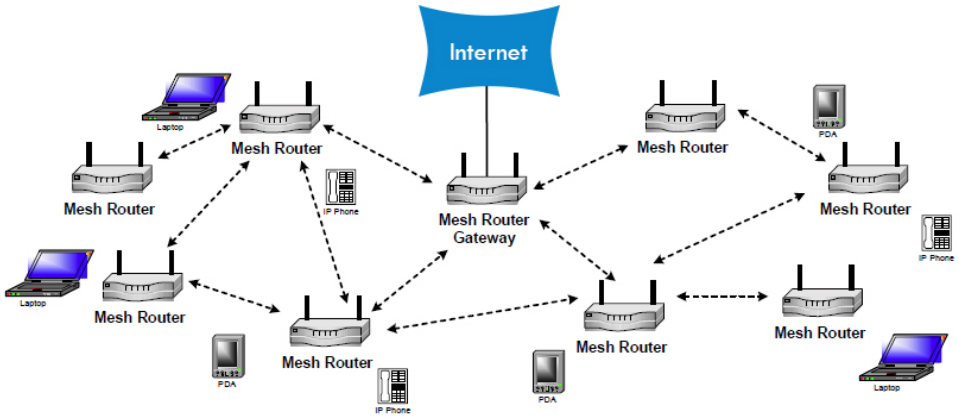
Como disse Ghosh, Basu e Das (2005 apud LUIZ; PRZYBYSZ, 2007) , as redes *Mesh* sem fio prometem ser a solução para uma série de problemas no provimento de serviços de acesso, por serem flexíveis, dinâmicas e potencialmente de baixo custo. Mas para que isso se torne efetivo, muita coisa há para ser melhorada e desenvolvida.

Uma das mais importantes características citada por Teixeira (2004) é que numa rede *Mesh* não existe um nó do qual toda a rede dependa. Se por ventura aconteça um mal funcionamento ou queda de um dos nós, a comunicação é reestabelecida através de novas rotas por outros nós sem a necessidade de interrupção da conexão ativa, já que os novos pacotes serão direcionados através da nova rota alternativa sem que o usuário perceba.

Com isso, nota-se uma grande motivação para o desenvolvimento de uma rede doméstica de sensores baseada na topologia *Mesh*. Onde em cada nó da rede possam ser

[2] Da tradução literal: Espinha dorsal. Nada mais é que a estrutura interna da rede.

Figura 5 – Topologia Mesh



Fonte: Adaptado de Abelém et al. (2007).

interligados periféricos.

Um exemplo de aplicabilidade seria em cada cômodo da casa existir um desses nós, onde o usuário possa através do periférico de controle acionar uma lâmpada conectada a um outro periférico de acionamento em um ponto remoto da casa. O comando de acionar vai pingando de nó em nó até atingir o nó de destino, caso necessário. Se o nó de destino estiver ao alcance do nó de origem, o comando acionar vai diretamente para ele. Isso mostra a versatilidade e a dinâmica adaptativa dessa topologia de rede.

3.4 As Comunicações

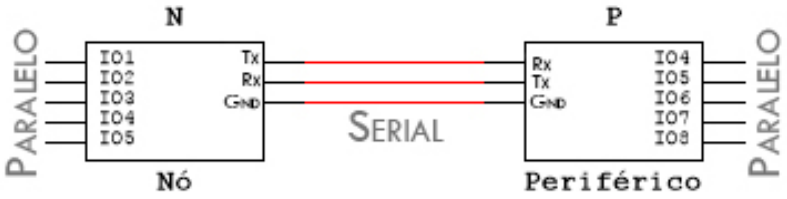
Nessa seção serão descritos os protocolos de comunicação denominados UART, I2C e SPI. Esses protocolos serão utilizados no desenvolvimento desse trabalho. Com a disponibilidade de três meios de comunicação, o módulo proposto assegura uma gama maior de periféricos que a ele possam ser conectados.

3.4.1 O Protocolo UART

A sigla UART vem do inglês Universal Asynchronous Receiver/Transmitter ou do português “Receptor/Transmissor Assíncrono Universal”. O dispositivo UART executa a conversão serial-paralela quando é um receptor e paralela-serial quando é um transmissor. Resumindo, é responsável pela comunicação de dados paralelos que trafegam por um meio serial. A Figura 6 ilustra esse tipo de comunicação. A conexão entre os terras (GND) do transmissor com o receptor faz-se necessário devido ao fato de que ambos precisam ter a mesma referência para discernir entre nível lógico baixo e nível lógico alto.

Uma comunicação é denominada serial quando o envio dos códigos dos caracteres

Figura 6 – A comunicação UART.



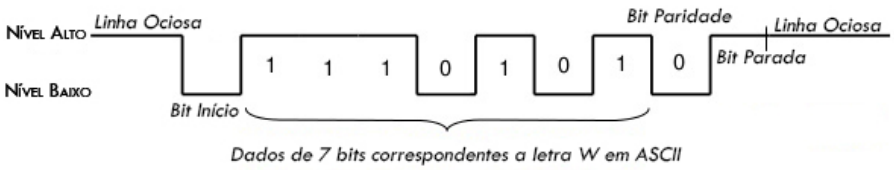
Fonte: Autoria Própria.

se processa sobre uma única linha, onde os bits enviados são encadeados em uma fila. É considerada assíncrona quando os relógios (clocks) do transmissor e do receptor não estão sincronizados. A velocidade de comunicação é chamada de baud. Essa unidade representa o número de bits transmitidos por segundo.

Quanto a sincronia dos transmissores e receptores UART pode-se dizer que a cada bit transmitido no canal deve respeitar a temporização de 1/baud segundos. Por exemplo, se for escrito no canal o bit 1, esse canal permanecerá em nível lógico alto por 1/baud segundos.

Através da Figura 7 pode-se observar um exemplo de como a transmissão do caractere W pode ser feita através de dispositivos UART. A conversão binária da letra W é feita com base na tabela ASCII [3].

Figura 7 – Exemplo de transmissão do caractere W.



Fonte: Autoria Própria.

Os dados são comumente transmitidos de uma ponta a outra em forma de pacotes os quais podem ser tão pequenos quanto um único byte. O formato desses pacotes é baseado em um número conhecido de protocolos. Cada pacote pode incluir uma verificação para detectar se o pacote esta correto ou foi corrompido ao longo da transmissão. Essa verificação poder ser através de checksum, bit de paridade ou Cyclic Redundancy Check Byte (CRC) (CURRIE; ESS, 2010).

[3] " American Standard Code for Information Interchange" é a tabela que é utilizada para representar textos em dispositivos de comunicação.

Segundo Kathik et al. (2012) existem três modos que uma comunicação UART pode assumir:

- Modo *Half-duplex*: Nesse modo a transmissão e a recepção acontecem uma de cada vez.
- Modo *Full-duplex*: Nesse modo a transmissão e a recepção acontecem ao mesmo tempo.
- Modo *Loop Back*: Esse modo é usado apenas para teste. Acontece quando o pino Tx é conectado ao pino Rx da mesma unidade UART para fins de verificar a funcionalidade da mesma.

3.4.2 O Protocolo SPI

SPI, do inglês: "Serial Peripheral Interface" ou interface serial de periféricos foi originalmente desenvolvida pela Motorola. É um protocolo que foi criado inicialmente para comunicação entre dispositivos periféricos. Ao contrário da comunicação UART, a SPI acontece de maneira sincronizada.

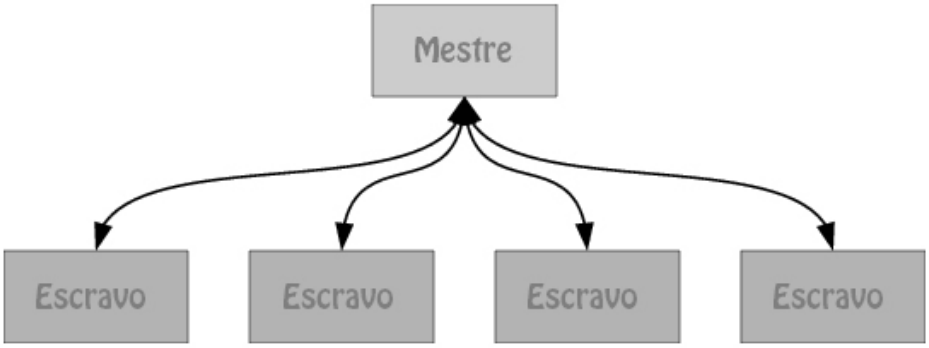
Os dados são transferidos de modo síncrono, porém, são transferidos juntamente com o sinal de clock e por conseguinte, a velocidade do clock é variável (CURRIE; ESS, 2010).

Em novidade ao UART, o protocolo SPI classifica os dispositivos que o utilizam como mestre ou escravo. O mestre, assim chamado, controla qual é a velocidade do clock no barramento. Pode ser operado no sistema de sentido único (*single wire/half-duplex*) ou até de duplo sentido (*full-duplex*). No modo *full- duplex* a transmissão ocorre em mão dupla simultaneamente. A cada bit que é enviado do mestre para o escravo acompanha um bit que é enviado do escravo para o mestre caracterizando assim o *full-duplex*. A Figura 8 mostra através de blocos a organização mestre-escravo.

Com o protocolo SPI é possível conectar quantos dispositivos o mestre possuir de pinos "SS – Seleção de Escravo" (SANDYA; RAJASEKHAR, 2012).

Os dispositivos SPI podem atingir velocidades de até 2Mbps. A Tabela 2 faz uma comparação entre o SPI e o UART em suas características.

Figura 8 – Organização mestre-escravo.



Fonte: Autoria Própria.

Tabela 2 – Comparacão entre os protocolos SPI e UART.

Tecnologia	Barramento	Taxa Máxima	Direção do Fluxo
UART	2 + GND	115200 bps	Half ou Full Duplex
SPI	3 + número de escravos	2 Mbps	Full Duplex

Fonte: Adaptado de Sacco (2014).

Existe uma nomenclatura padrão para as conexões do tipo SPI que facilitam na hora de identificar e executar a comunicação entre os dispositivos. A Tabela 3 exibe os nomes abreviados e suas funções.

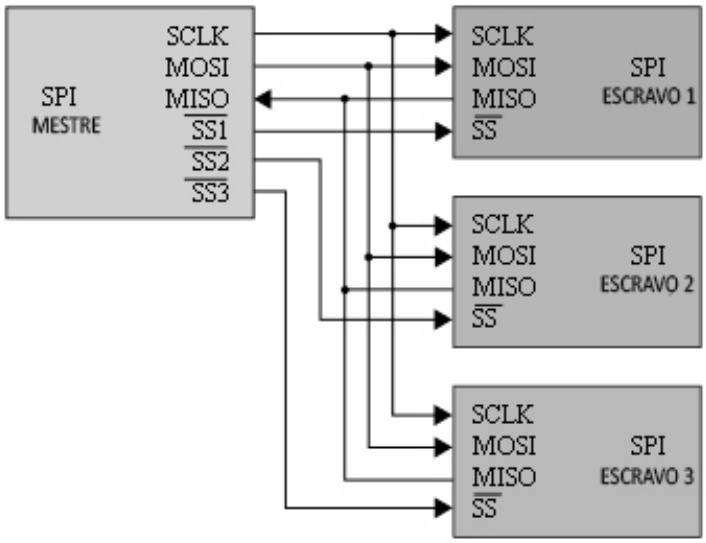
Tabela 3 – Siglas utilizadas na comunicação SPI.

Pino	Nome Abreviado	Significado
Mestre p/ escravo	MOSI	Master Output Slave Input
Escravo p/ mestre	MISO	Master Input Slave Output
Clock	SCLK	Serial Clock
Selecão de escravo	SS	Slave Select

Fonte: Adaptado de Sacco (2014).

Por exemplo, se o dispositivo mestre gostaria de enviar para o escravo um dado, ele irá escrever no barramento MOSI, ou seja, saída do mestre/entrada do escravo. Ao mesmo tempo que o SS (seleção de escravo) é ativada. O mestre, além de escrever os dados que gostaria de enviar para o escravo no barramento MOSI, deve também especificar qual escravo irá receber a informação. Essa seleção é através do pino SS. Como o barramento é compartilhado, faz-se necessário especificar para qual escravo a informação está indo. A Figura 9 exemplifica a situação de um mestre comunicando com três periféricos escravos.

Figura 9 – Esquemático de um mestre e três escravos na comunicação SPI.



Fonte: Adaptado de Sacco (2014).

3.4.3 O Protocolo I2C

O nome I2C ou até mesmo IIC vem do inglês "Inter-Integrated Circuit". Em português: Circuito Inter-Integrado. É um protocolo desenvolvido pela gigante holandesa Philips e foi inicialmente usado para anexar periféricos de baixa velocidade a placas-mãe, sistemas embarcados, celulares ou outros dispositivos digitais (SAMANTA, 2014).

> O barramento I2C permite a comunicação entre o processador do sistema e os dispositivos periféricos. É usado para minimizar o hardware a nível de sistema. A transferência de dados em um barramento I2C melhora a performance do sistema visto que a transmissão digital de dados é muito menos susceptível a erros causados por interferências do ambiente ou fontes de ruído (KRISHNAKISHORE; SHRUTI; VARSHA, 2014).

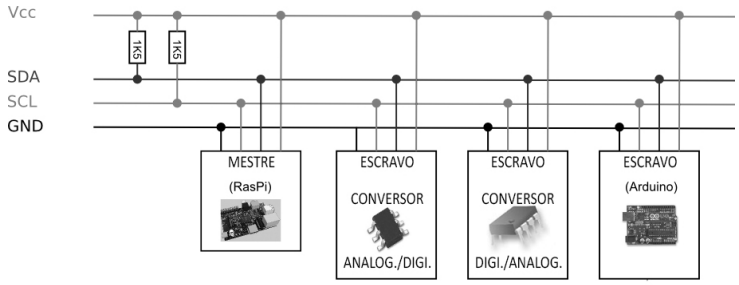
O barramento I2C fisicamente consiste em dois fios ativos e uma conexão terra (GND) para referência. Os dois fios ativos são bidirecionais e atendem pelas siglas SDA e SCL. SDA vem do inglês Serial Data Line ou "linha de dados serial" e SCL significa Serial Clock Line que em português é "linha de clock serial". Cada dispositivo conectado ao barramento I2C tem seu próprio endereço único, não importando qual sua natureza, seja ele um microcontrolador, um display, uma memória, etc. Cada um desses periféricos no barramento pode atuar como receptor e/ou transmissor, dependendo da sua funcionalidade (KUMAR; SNEHA, 2013).

Ao contrário do protocolo SPI, por exemplo, o I2C tem um barramento multi-mestre. Isso significa que mais de um periférico conectado ao barramento pode iniciar uma comunicação de dados com outro. O SPI tem a limitação de que somente o mestre

inicia a comunicação com o escravo selecionado. Quando um periférico I2C inicia uma comunicação, automaticamente todos os outros periféricos conectados ao barramento se tornam escravos.

A Figura 10 exemplifica uma situação com quatro dispositivos conectados ao barramento I2C. No instante t da figura, o dispositivo mestre é o RaspberryPI e os outros três dispositivos (conversor analógico/digital, conversor digital/analógico e um arduíno) são automaticamente escravos do mestre RaspberryPI.

Figura 10 – Dispositivos conectados ao barramento I2C.



Fonte: Adaptado de KrishnaKishore, Shruti e Varsha (2014).

Segundo KrishnaKishore, Shruti e Varsha (2014), uma comunicação I2C padrão consiste basicamente de quatro partes, são elas:

- Sinal Início (*START* signal): Quando o barramento está ocioso, isto é, quando nenhum periférico está utilizando o barramento, ambas as linhas SDA e SCL estão em nível lógico alto. Um periférico que deseja se tornar mestre, envia um sinal de *START* comumente representado pelo S-bit [4] que nada mais é que uma transição de nível lógico alto para nível lógico baixo na linha SDA enquanto SCL permanece em nível lógico alto. O sinal *START* significa o início de uma nova transmissão de dados.

- Endereço do Escravo (*Slave Address Transfer*): É o próximo byte logo após o sinal de *START* e como o próprio nome já diz, representa o endereço do periférico escravo ao qual deseja-se iniciar uma transmissão de dados. Como dito anteriormente, cada dispositivo tem seu endereço único, o que assegura que não haverá conflitos. Logo, somente o escravo com o endereço requisitado irá atender sinalizando através da linha um bit de reconhecimento que nada mais é que transitar o nível da linha SDA para baixo no ciclo de clock de SCL.

- Transferência de Dados: Uma vez que o endereçamento do escravo aconteça corretamente, a transferência de dados pode iniciar-se enviando byte a byte. A

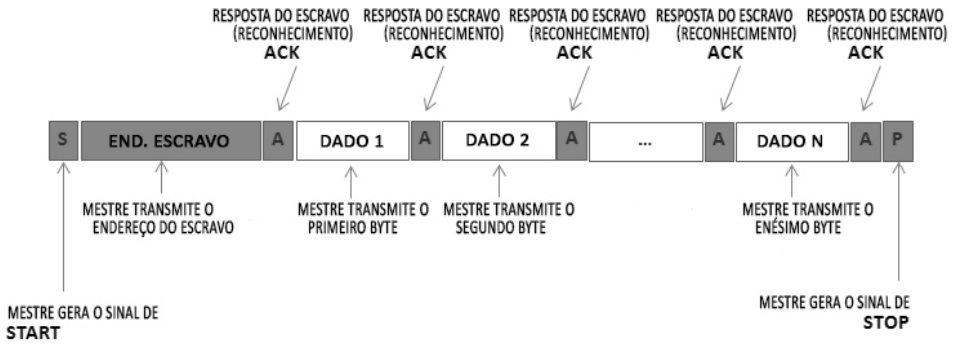
[4] Abreviação de "*Start Bit*" ou bit de início.

cada byte enviado, é enviado pelo escravo um sinal de reconhecimento no ciclo de clock da linha SCL.

- Sinal de Parada (*STOP signal*): O mestre termina a comunicação gerando um sinal de parada. Esse sinal é geralmente representado pelo bit P o que é uma transição de nível lógico baixo para nível lógico alto na linha SDA quando SCL está em nível alto.

Na Figura 11 é possível observar o formato de dados a ser transferido em um barramento I2C conforme as quatro partes citadas acima.

Figura 11 – Formato de dados que trafegam em um barramento I2C.



Fonte: Adaptado de KrishnaKishore, Shruti e Varsha (2014).

A Tabela 4 faz uma comparação entre as três tecnologias apresentadas em função de suas principais características.

Tabela 4 – Comparacão entre os protocolos UART, SPI e I2C.

Tecnologia	Barramento	Taxa Máxima	Direção do Fluxo
UART	2 + GND	115200 bps	Half ou Full Duplex
SPI	3 + número de escravos	2 Mbps	Full Duplex
I2C	2 + GND	400 Kbps	Half Duplex

Fonte: Adaptado de Sacco (2014).

3.5 WSN - As Redes de Sensores Sem Fio

Do inglês: Wireless Sensor Network, ou RSSF (rede de sensores sem fio) é uma tecnologia de comunicação sem fio que utiliza dispositivos de rede de baixo consumo de energia. Para Mini e Loureiro (2009 apud RIVERO, 2011), esses dispositivos de rede transmitem dados coletados por sensores, permitindo monitorar tanto o estado de utilização

de um equipamento eletrônico quanto o ambiente em que este equipamento está inserido, com pouco ou nenhum impacto físico neste ambiente.

Os dispositivos de rede de uma RSSF são chamados de módulos sensores e são compostos por um rádio de comunicação sem fio, um microcontrolador para processamento dos dados, sensores e uma fonte de alimentação, sendo esta fixa ou por baterias. Se este módulo sensor é alimentado por baterias, o tempo do monitoramento do ambiente é limitado a duração desta bateria. Algumas técnicas para economizar bateria podem ser aplicadas, como atualização da rede em intervalos maiores ou até mesmo funcionamento em *stand by* durante a madrugada.

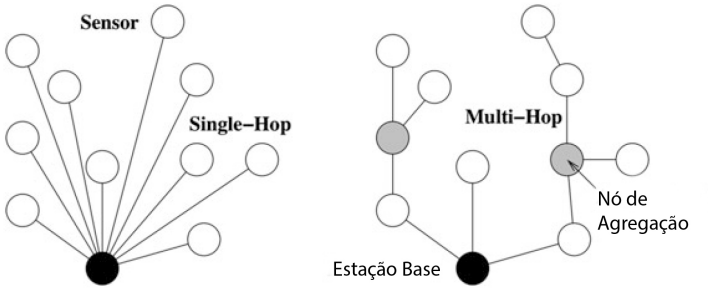
> Uma RSSF é uma rede sem fio que consiste de dispositivos autônomos distribuídos no espaço usando sensores para monitorar grandezas como temperatura, sons, vibração, pressão, movimento ou poluentes em locais diferentes. O desenvolvimento original das RSSFs foi inicialmente motivado para aplicações militares como monitoramento do campo de batalha. Entretanto, atualmente as RSSFs são usadas para diversas aplicações civis como monitoramento de habitat, saúde, domótica e controle de tráfego (EL-BENDARY et al., 2015, p.9).

Existem vários termos relevantes quando se refere a uma RSSF. Os mais importantes e seus significados são:

- *RSSF* é o grupo de nós que são conectados entre si sem o uso de fios. Juntos formam a rede de sensores;
- *Sensores e Detectores* é a parte sensora do nó;
- *Área de Cobertura* é a máxima área ou seção que os sensores podem cobrir;
- *Cluster* é um grupo de nós em serviço de uma área específica;
- *Coordenador* é o cérebro do sistema, pode tomar decisões ou agir apenas como um roteador.

A estrutura de uma RSSF pode ser composta por módulos sensores, módulos roteadores e por fim um módulo coordenador para receber e processar as informações que trafegam pela rede. O nó de uma RSSF é um dos vários dispositivos presentes na rede, se refere a parte do processo de sensoriamento e também as operações associadas a este processo como a de processamento de sinais, amplificação, conversões analógica/digital, entre outros (EL-BENDARY et al., 2015, p.9). Na Figura 12 é possível observar a estrutura de rede *single-hop* e *multi-hop*. Nota-se que na estrutura *multi-hop* estão presentes os nós agregadores permitindo maior confiabilidade em função da distância física dos módulos, porém insere complexidade e custo a rede.

Figura 12 – Estrutura *single-hop* e *multi-hop*.



Fonte: Adaptado de El-Bendary et al. (2015).

Os termos *single-hop* e *multi-hop* quando traduzidos indicam facilmente seu significado. Na Figura 12 vemos a estrutura de único salto à esquerda, e logo a direita a estrutura de multi saltos. Os controladores de cada nó realiza tarefas, processa dados e controla as funcionalidades de outros componentes no nó. A maioria destes controladores são microcontroladores, mas outras alternativas que também podem ser usadas como controladores são: processador de uso geral, processador digital de sinais, FPGAs e ASICs. Os microcontroladores são mais frequentemente utilizados em aplicações embarcadas assim como em nós sensores devido ao seu baixo custo, flexibilidade em conectar-se a outros dispositivos, facilidade de programação e baixo consumo de energia.

Um processador de uso geral utilizado nos PCs, geralmente, possui consumo de energia maior que um microcontrolador, por isso não é comumente utilizado para um nó sensor. Os processadores digitais de sinais podem ser escolhidos para aplicações sem fio que requerem banda larga, o que não é o caso das RSSFs. Já as FPGAs podem ser reprogramadas e reconfiguradas de acordo com a necessidade, mas toma mais tempo e energia do que o necessário (EL-BENDARY et al., 2015, p.10).

É importante ressaltar a diferença entre um sistema embarcado e uma RSSF. O que diferencia uma RSSF dos sistemas embarcados é o grande número de nós participantes, o uso de comunicação sem fio e severas restrições de recursos, custos e potência nos nós sensores. Pode-se definir um sistema embarcado como um sistema cuja principal função é não computacional, mas é controlada por um computador embarcado (EL-BENDARY et al., 2015, p.13).

3.5.1 Definição de Sensoriamento

O princípio base das RSSFs é o processo de sensoriamento. Sensoriamento remoto pode ser definido como quaisquer processos onde informações são coletadas sobre um objeto, área ou fenômeno sem estar em contato (EL-BENDARY et al., 2015, p.24). O olho humano é um exemplo de um dispositivo de sensoriamento remoto excelente, sendo capaz

de coletar informações do ambiente através da luz refletida nos objetos, seja esta luz do sol ou de uma lâmpada. Ao contrário um termômetro que necessita estar em contato com o objeto para que uma grandeza seja medida.

O controlador tem um papel importante em monitorar e traduzir os dados obtidos dos sensores, dando sentido físico para estes. Como mostrado na Figura 13 o cérebro humano faz o papel de controlador e os olhos, o papel do nó sensor. Nota-se a diferença na imagem crua captada pela retina e a mesma imagem tratada e já processada no cérebro.

Figura 13 – A esquerda, imagem captada pela retina e a direita imagem processada pelo cérebro humano.

Fonte: El-Bendary et al. (2015).

Os sensores fazem a conexão do mundo físico com o mundo digital, convertendo dados do mundo real de uma maneira que possam ser processados, armazenados e servir como base para ações a serem tomadas.

3.5.2 Definições entre Sensor e Transdutor

Sensor é o termo empregado para designar dispositivos sensíveis a alguma forma de energia do ambiente que pode ser luminosa, térmica, cinética, relacionando informações sobre uma grandeza que precisa ser medida como, temperatura, pressão, velocidade, corrente, aceleração, posição, entre outras (THOMAZINI; ALBUQUERQUE, 2005, p.17).

Ainda para Thomazini e Albuquerque (2005, p.19), transdutor é a denominação que recebe um dispositivo completo, que contém o sensor, dispositivo usado para transformar uma grandeza qualquer em outra que possa ser utilizada em sistemas de controle. Os transdutores transformam uma grandeza física em um sinal de tensão ou corrente que possa ser facilmente interpretado por estes sistemas. Portanto, todos os transdutores contém sensores.

É possível observar através da Tabela 5 alguns tipos de grandezas e os tipos de sensores utilizados para suas respectivas medições.

Tabela 5 – Algumas grandezas e sensores.

Grandeza	Tipos de Sensores
Temperatura	Termistores, termoaclopadores
Pressão	Barômetros, sensores infravermelho
Ótica	Fotodiodos, fototransistores, fotoresistores
Acústico	Microfones, ressonadores piezoelétricos
Movimento	Acelerômetros, giroscópios
Umidade	higrômetros, sensores capacitivos
Posição	GPS, ultrasom

Fonte: Adaptado de El-Bendary et al. (2015).

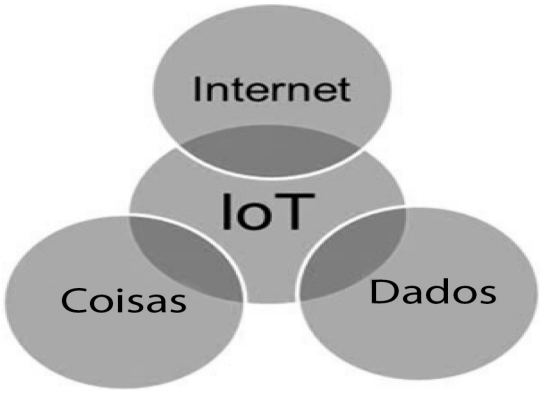
3.6 IoT - Internet das Coisas

A próxima tendência na era da computação vai ser fora do domínio do computador de mesa tradicional. No paradigma da Internet das Coisas (IoT), vários dos objetos que nos cercam estarão na rede de uma forma ou de outra. Segundo Machado (2016) IoT é uma maneira de se referenciar objetos através de um tipo de endereçamento, de maneira única, para que possam se comunicar dentro de uma rede maior. A origem da IoT é atribuída ao centro de Auto-ID do Instituto de Tecnologia de Massachusetts (MIT) que desenvolveu o sistema de reconhecimento por rádio frequência. Resumidamente, segundo Minerva e Crespi (2017, p.146) IoT é um sistema complexo auto configurável e adaptativo feito por redes de sensores e objetos inteligentes com o propósito de conectar todas as "coisas", incluindo objetos do dia a dia e objetos industriais de uma maneira que os torne inteligentes, programáveis e mais capazes de interagir com humanos.

Para Rayes e Salam (2017, p.2), o termo IoT ganhou uma grande fama não só entre os entusiastas de tecnologia como também na academia e na indústria especialmente nos últimos anos. As razões por de trás deste interesse são as importantes funcionalidades que a IoT tem a oferecer. No nível pessoal, esta tecnologia descreve um mundo onde todas as coisas ao redor do ser humano estarão conectadas a internet e se comunicarão entre si de maneira inteligente. O objetivo principal é que os objetos do ambiente pessoal possam ter sensoriamento eficiente, comunicação de baixo custo e assim criar um ambiente melhor. A Figura 14 define a IoT na sua mais simples forma.

No âmbito industrial, as companhias já estão empregando a IoT para criar novos modelos de negócios, melhorando processos e reduzindo custos e riscos. Alguns dos ganhos que esta tecnologia traz a vida pessoal se dá pelo avançado monitoramento de saúde, aprendizado otimizado e melhorias na segurança (RAYES; SALAM, 2017, p.2).

Figura 14 – A difinição de IoT na sua forma mais simples.



Fonte: adaptado de Rayes e Salam (2017).

A mundialmente famosa empresa de consultoria em tecnologia Gartner, diz que cerca de 20 bilhões de dispositivos farão uso da IoT até 2020. A CISCO estima cerca de 26,3 bilhões de dispositivos (incluindo celulares, TVs, PCs, tablets, entre outros) no mesmo período. Outros acreditam que esses valores são modestos, visto que qualquer tipo de dispositivo pode ser conectado a internet seja ele um microcontrolador ou uma modesta chave liga-desliga.

A IoT pode ser considerada uma rede de dispositivos físicos que englobam:

- **Sensores**: para coletar informações;
- **Identificadores**: para identificar a fonte dos dados;
- **Software**: para analisar os dados;
- **Conexão com a internet**: para se comunicar e notificar.

Estas conexões não se limitam apenas em informar e gerar relatórios com base nos dados mas também atuar, permitindo aos usuários que alcancem e controlem "coisas" através da IoT, como por exemplo acender uma lâmpada remotamente.

Uma pergunta interessante é: Por que monitorar e controlar coisas? Para Rayes e Salam (2017, p.5) existem diversos motivos para monitorar e controlar coisas remotamente pela internet. Por exemplo, monitoramento por especialistas da área da saúde que conseguem obter dados da temperatura do paciente ou pressão sanguínea enquanto o paciente está no conforto de sua casa ou até mesmo permitir que autoridades gerencie "coisas" em cidades inteligentes de uma maneira otimizada. Esses exemplos são grandes oportunidades de negócios.

É possível observar na Tabela 6 a previsão dos dados que indicam que em 2020 cerca de 21 bilhões de "coisas" estarão conectadas. Esse número é abastecido com os

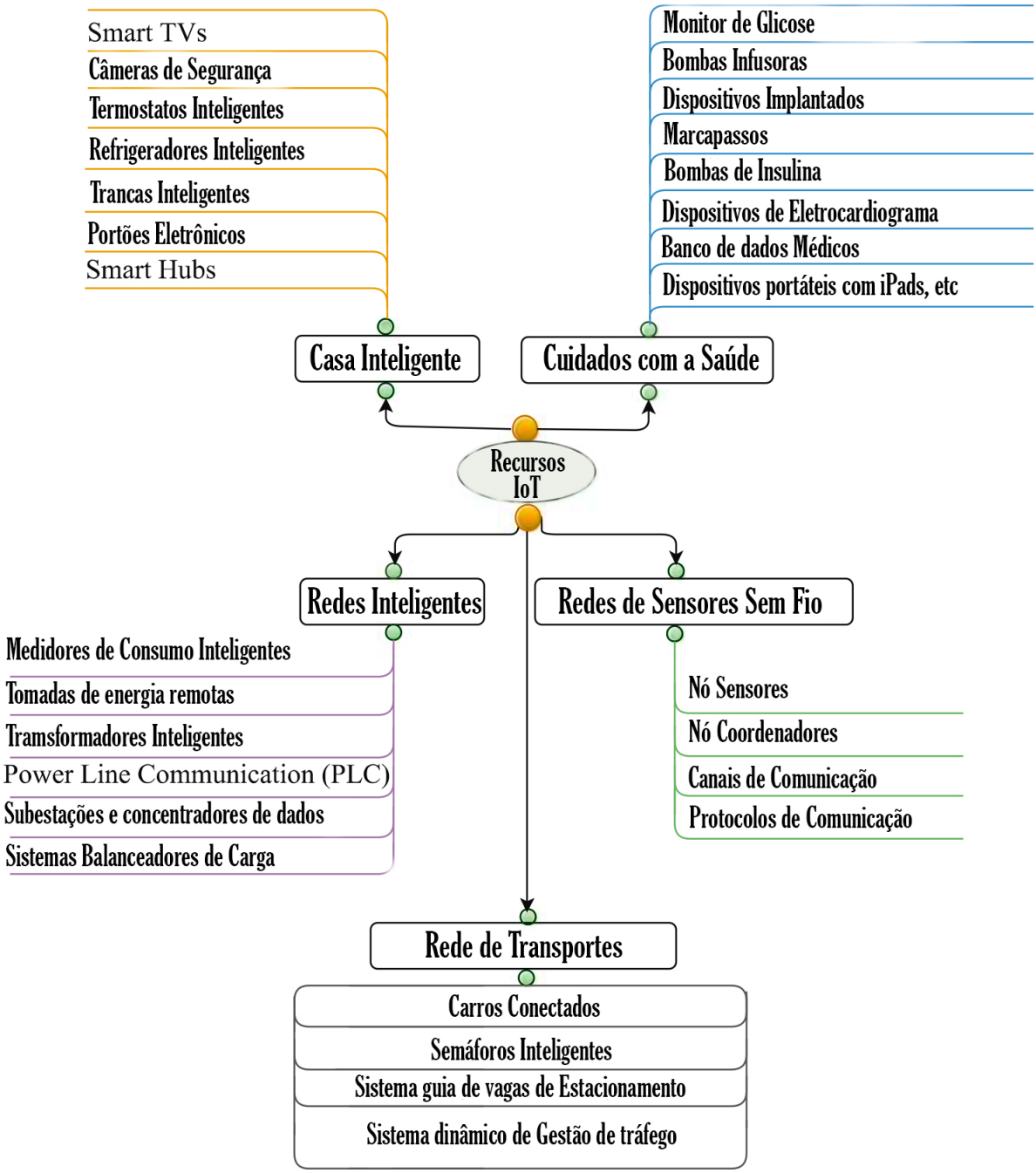
avanços nas áreas de telefonia celular, computação em nuvem e redes sociais combinadas com IoT. É visível na Figura 15 um apanhado geral das tecnologias e recursos que podem ser gerenciados através a IoT, destaque para a RSSF.

Tabela 6 – Previsão: Milhões de unidades de IoT instaladas baseada em categorias, excluindo a automotiva.

Categoria	2014	2015	2016	2020
Consumidor	2277	3023	4024	13509
Negócios Gerais	623	815	1092	4408
Negócios Verticais	898	1065	1276	2880
Total Geral	3807	4902	6392	20797

Fonte: Adaptado de Rayes e Salam (2017).

Figura 15 – Apanhado geral das tecnologias e recursos em IoT.



Fonte: Adaptado de Rayes e Salam (2017).

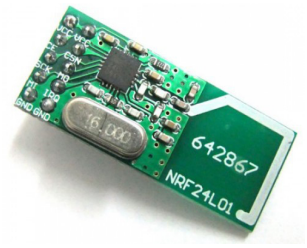
3.7 As Tecnologias Consideradas

Nesta seção serão descritas as tecnologias consideradas para aplicação no desenvolvimento prático deste trabalho.

3.7.1 O Módulo *NRF24L01*

Este transceptor de rede sem fio é fabricado exclusivamente pela Nordic Semiconductor. É um único chip que opera em 2,4 GHz com um protocolo em banda base desenvolvido para aplicações sem fio de ultrabaixa potência, além de possuir um baixo custo de aquisição.

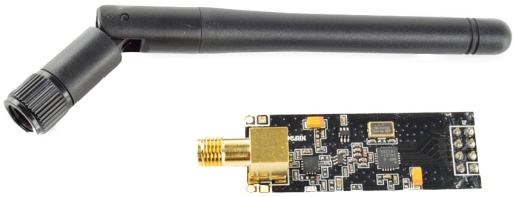
Figura 16 – Modulo NRF24L01 com antena embutida.



Fonte: Adaptado de www.webtronico.com .

O NRF24L01 (Figura 16) é configurado e operado via protocolo SPI (Serial Peripheral Interface) de comunicação. O front end do rádio usa a modulação GFSK (Gaussian frequency-shift keying) e permite algumas configurações como frequência do canal, potência de saída e velocidade de dados. Esta velocidade de dados é configurável até 2 Mbps. O alcance pode chegar até 1 Km nas versões com antena externa (Figura 17) e transmissões em baixas velocidades (NORDIC SEMICONDUCTORS, 2007).

Figura 17 – Modulo NRF24L01 com antena externa 3dB.



Fonte: Adaptado de robotshop.com.

Algumas especificações:

- Tensão de alimentação: 1,9 - 3,6 V;
- Frequência: 2,4 Ghz;
- Velocidade de Operação: 2 Mbps (máx);
- Alcance inversamente proporcional a velocidade de operação.
- Regulador de voltagem embutido;

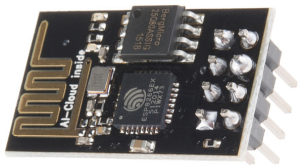
- Dimensões: 3,3 x 1,4 x 0,5 cm;
- Corrente durante a transmissão: 11.3 mA;
- Corrente durante a receptação: 12,3 mA;
- Corrente em repouso: 900 nA;
- Sensibilidade: -85 dBm;
- Temperatura de trabalho: -40 a 85ºC.

Devido a sua ótima relação entre custo e benefício, este módulo é a melhor opção para a implementação prática desta rede de sensores, permitindo a integração dos diversos dispositivos e sensores pertencentes a rede, com velocidades e distâncias satisfatórias.

3.7.2 O Módulo *ESP8266*

Ao contrário do *NRF24L01* que opera em modulação GFSK, o *ESP8266* pode se conectar as redes WiFis convencionais, ou seja, no protocolo 802.11 b/g/n. Fabricado pela Espressif Systems, este módulo SoC, do inglês *System on Chip* (sistema em um chip), além da conectividade WiFi com celulares, notebooks e outros, traz também o protocolo TCP/IP integrado.

Figura 18 – Módulo ESP8266 com antena embutida.



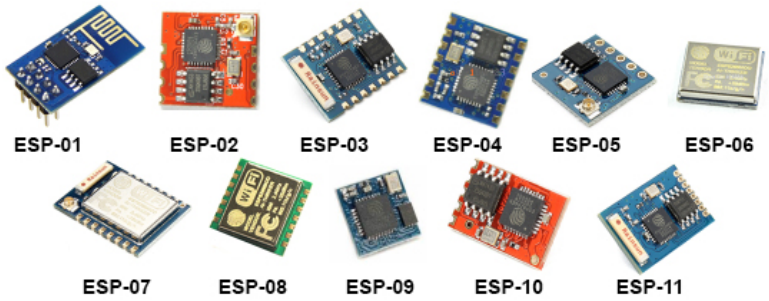
Fonte: Adaptado de sparkfun.com.

A Figura 18 mostra uma de suas versões, a 01. Existe também a capacidade deste módulo funcionar em modo de Ponto de Acesso (AP), agindo como um roteador. Sua comunicação com um microcontrolador pode ser feita via serial e sua configuração é feita por comandos AT (ESPRESSIF SYSTEMS, 2013). Já na Figura 19 é possível observar alguns dos vários tipos de módulos ESPs que trazem consigo diferentes recursos, portas GPIO, capacidade de transmissão, entre outros. Algumas especificações:

- Tensão de operação: 3,3 V;
- Suporte à redes: 802.11 b/g/n;
- Alcance: 90 m aproximadamente;

- Modo de segurança: OPEN/WEP/WPA_PSK/WPA2_PSK/WPA_WPA2_PSK;
- Dimensões: 25 x 14 x 1 mm;
- Peso: 7 g.

Figura 19 – Alguns dos vários tipos de módulos ESPs.



Fonte: Adaptado de sparkfun.com.

Esta solução de baixo custo, aproveitando-se da conexão serial provida pelo módulo servidor deste trabalho, poderá conectar a rede de sensores a rede mundial de computadores, seguindo a tendência da IoT (internet das coisas).

3.7.3 O Microcontrolador *ATmega328*

Um microcontrolador é um sistema de computador inteiro contido em um único chip de circuito integrado. A operação de um microcontrolador é controlada por um programa criado pelo usuário interagindo com a arquitetura de hardware fixa do microcontrolador (BARRETT, 2010, p.15). Um microprocessador, utilizado nos PCs, são chamados de microprocessadores de uso geral, como a família Intel x86. Estes processadores não possuem RAM, ROM e nenhuma porta I/O no chip em si. Ao contrário dos microcontroladores que possuem uma CPU (um microprocessador) em adição uma quantidade fixa de RAM, ROM, *timers* e portas I/O que suprem suas necessidades (MAZIDI; NAIMI; NAIMI, 2012, p.40).

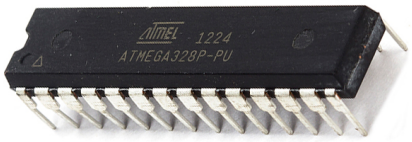
A escolha de um microcontrolador se dá em função de várias características. Dentre elas está o fato deste ser acessível em termos de disponibilidade comercial e o custo deste componente levando em consideração sua aplicação em projetos e orçamentos universitários. Assim como, a disponibilidade deste microcontrolador em relação a seus periféricos internos, como timers, conversores analógico/digital ou velocidade do clock (MACHADO, 2016).

Diversas empresas produzem diferentes famílias e modelos de microcontroladores. Além dos fatores já citados, a disponibilidade do ambiente de desenvolvimento (IDE -

Integrated Development Environment) e a existência de bibliotecas de software gratuitas também são importantes.

Levando em consideração estes fatores, foi escolhido o ATmega328P da fabricante ATmel, devido ao fato de ter fácil aquisição, baixo custo, ambiente de desenvolvimento gratuito e bem fundamentado e sua presença em placas de desenvolvimento para kits didáticos e de hobistas. Além disto, este microcontrolador permite a gravação de um *bootloader*[5] específico, permitindo com que se carregue programas diretamente pela USB. Sua pinagem, em formato DIP facilita a fabricação de protótipos.

Figura 20 – Microcontrolador ATmega328 em sua versão DIP.



Fonte: Adaptado de rhydolabz.com.

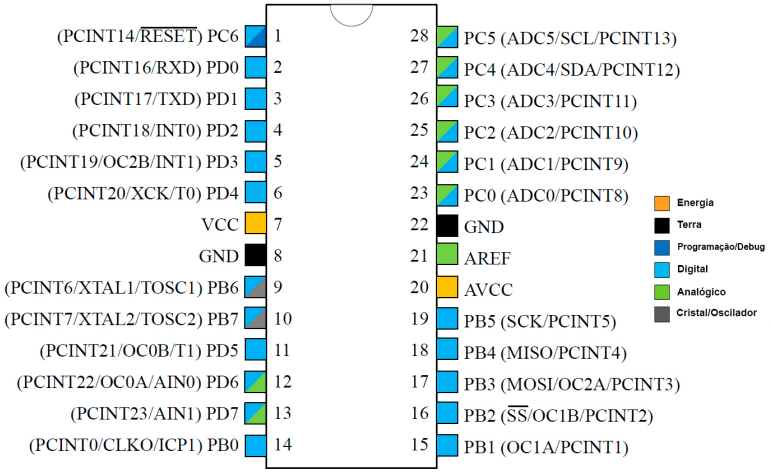
O ATmega328 (Figura 20) é equipado com 32 registradores de uso geral de 8 bits, estes registradores estão conectados diretamente à Aritmetic Logic Unit (ALU) - Unidade de Lógica e Aritmética, permitindo ao processador trabalhar com dois registradores simultaneamente em apenas um ciclo de clock. Através da Figura 21 é possível observar a disposição dos pinos no encapsulamento DIP. O processador foi desenvolvido seguindo a arquitetura de Harvard. Esta arquitetura baseada em registradores juntamente com o conjunto de instruções RISC, do inglês *Reduced Instruction Set Computer* (Computador com Conjunto Reduzido de Instruções) permite uma rápida e eficiente execução de programas e proporciona ao processador a capacidade de completar uma instrução em linguagem Assembly a cada ciclo de clock. O ATmega328 possui um conjunto de 131 instruções, executada em um ciclo de clock, opera na faixa de tensão entre 1,8 a 5,5V e na faixa de temperatura entre -40ºC e 85ºC (ATMEL, 2016).

A empresa americana Atmel (atualmente em posse da Microchip), desenvolvedora deste microcontrolador informa que este pode executar 20 MIPS (milhões de instruções por segundo) quando operado em seu clock máximo, 20MHz (BARRETT, 2010, p.16). Este microcontrolador conta também com outros recursos como conversores analógico/digital, geradores de PWM e diferentes tipos de portas de comunicações, como por exemplo UART, SPI e I2C, permitindo a interação com outros dispositivos (MAZIDI; NAIMI; NAIMI, 2012, p.45).

Quando se fala em microncontrolador, logo vem em mente sistemas embarcados. Segundo Anderson (2008 apud BARRETT, 2010), um sistema embarcado contém um

[5]Programa que reside na área de *boot* do microcontorlador.

Figura 21 – Configuração de pinos do microcontrolador Atmega328.



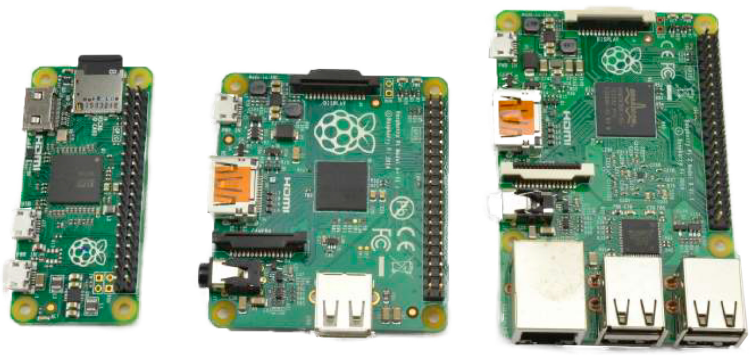
Fonte: Adaptado de Atmel (2016).

microcontrolador para realizar o trabalho de processar as entradas e gerar as saídas do sistema. A conexão entre entradas e saídas é resultado de um algoritmo codificado armazenado na memória.

3.7.4 A Plataforma Raspberry Pi

O Raspberry Pi é um computador do tamanho de um cartão de crédito. Desenvolvido e produzido no Reino Unido com a intenção de fornecer computadores baratos com objetivos educacionais. Desde seu lançamento comercial em 2012, a plataforma passou por inúmeras revisões e agora está disponível em duas versões, o modelo A e o modelo B. O modelo A é o mais simples e barato, já o modelo B é mais potente, incluindo suporte a rede ethernet (DENNIS, 2016, p.2). É possível observar na Figura 22 os modelos encontrados atualmente no mercado.

Figura 22 – Raspberry Pi Zero(esquerda), Modelo A+(centro) e Raspberry Pi 2 modelo B(direita).



Fonte: Adaptado de Monk (2016).

Como mostrado na Figura 22, o modelo A+ é menor que o Pi 2 modelo B e tem apenas uma porta USB e nenhuma porta RJ45. O Pi Zero é ainda menor, ganhando espaço utilizando a porta mini HDMI e uma porta micro USB (MONK, 2016, p.2). O Raspberry Pi 2 é um dispositivo pequeno, medindo cerca de 85,6x56x21 mm e com peso de aproximadamente 45 g. Seu tamanho reduzido o torna excelente para aplicações embarcadas, serviços de automação residencial, entre outros (DENNIS, 2016, p.3).

A arquitetura SoC consiste em uma CPU, GPU, SDRAM e uma porta USB. Todos esses recursos implementados pela fabricante Broadcom, no chip de modelo BCM2836. Sua central de processamento (CPU), é de arquitetura Arm, modelo A7 com 900 MHz de clock. A GPU, o chip especialista em gerenciar os dados complexos de renderização gráfica, possui 250 MHz e proporcina uma resolução de 1080p. Em termos de memória, o Raspberry Pi conta com 1 GB de memória SDRAM e porta de cartão SD para armazenamento do sistema operacional, programas e arquivos do usuário. Além de sua porta *ethernet* para comunicação com a rede mundial, placa de áudio, saídas de vídeo HDMI e áudio analógico (RCA), suas portas de GPIO são de extrema importância para aplicações embarcadas, contando com 40 pinos no total com capacidade de serem entradas ou saídas e sendo controlados por uma variedade de linguagens de programação (DENNIS, 2016, p.5).

3.7.5 O Sensor de Umidade e Temperatura DHT22

O DHT22 (Figura 23) é um sensor de temperatura e umidade que permite fazer leituras de temperaturas entre -40 a +80ºC e umidade entre 0 a 100%, sendo muito fácil de usar com Arduino, Raspberry e outros microcontroladores, pois possui apenas um pino com saída digital. Este dispositivo é formado por um sensor de umidade capacitivo e um termistor para medir o ar ao redor, enviando no pino de dados um sinal digital (SENSOR. . . , 2016).

Figura 23 – DHT22.

Fonte: Adaptado de Sensor. . . (2016).

Este sensor utiliza uma técnica exclusiva do fabricante Aosong Electronics para assegurar a confiabilidade e estabilidade. Seu microcontrolador interno é de 8 bits permi-

tindo uma auto-calibração e compensação de temperatura. Vem em um encapsulamento único com quatro pinos, facilitando sua instalação (AOSONG ELECTRONICS, 2014). Na Tabela 7 lista-se as especificações técnicas deste sensor.

Tabela 7 – Especificações técnicas do sensor DHT22.

Especificação	Valores
Alimentação	3.3-6V DC
Limite Operação	UR[a] 0-100% e Temperatura -40 à 80°C
Exatidão	±2% UR e ±0,5°C
Sensibilidade	0,1% UR e 0,1°C
Período de Leitura	a cada 2 segundos
Dimensões	14x18x5,5mm

Fonte: Adaptado de Aosong Electronics (2014).

[a]UR se refere a umidade relativa do ar

Várias bibliotecas para este sensor podem ser encontradas para diversas plataformas com uma rápida busca. Como no fragmento de código em C, Quadro 3.1.

```
1  #include <DHT.h>
2  int main(void){
3  double t = dht.readTemperature();
4  double h = dht.readHumidity();
5  }
```

Quadro 3.1 – Fragmento de código em C para leitura da umidade e temperatura.

Neste código, o programa chama as funções ***dht.readTemperature()*** e ***dht.readHumidity()*** e armazena os valores obtidos do retorno desta função em variáveis do tipo ***double*** adequada aos valores decimais de umidade e temperatura. Estes valores podem ser armazenados, utilizados em operações matemáticas e então transmitidos.

3.7.6 O Sensor de Luminosidade ALS-PT243-3C

O sensor ALS-PT243-3C (Figura 24) consiste de um fototransistor em um encapsulamento DIP. É uma solução para sensoriamento de luz ambiente de baixo consumo para aplicações como telefonia móvel, notebooks, PDAs, ganho automático de contraste em telas LCD, entre outros. Devido a sua alta rejeição a luz na frequência do infravermelho, a resposta espectral deste sensor é próxima a área sensibilizada pelo olho humano (EVERLIGHT, 2007).

Figura 24 – ALS-PT243-3C.

Fonte: Adaptado de Everlight (2007).

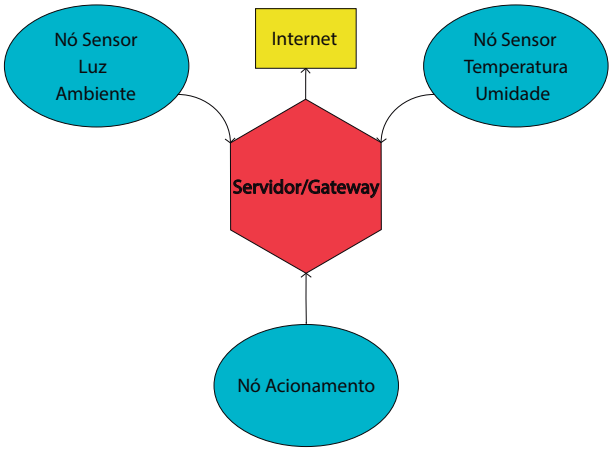
Algumas características segundo Everlight (2007):

- Tensão de alimentação entre 2,5 a 5,5 V;
- Lente plana de 5 mm;
- Saída linear;
- Pico de sensibilidade na frequência de 630 nm.

4 Metodologia

Neste capítulo serão descritos os materiais e métodos utilizados para atingir os objetivos propostos por este trabalho. Na Figura 25 é possível visualizar a topologia de rede *single-hop* que será implementada com a metodologia a seguir.

Figura 25 – Diagrama da topologia de rede.



Fonte: Autoria Própria.

4.1 Materiais e Recursos

Nesta seção serão apresentados de forma objetiva os materiais e recursos, sejam eles físicos ou computacionais utilizados para a elaboração deste trabalho. Todas as tecnologias da seção 3.7 foram utilizadas como materiais, com exceção do Raspberry PI, devido a seu alto custo no mercado brasileiro.

4.1.1 A IDE Arduino

A IDE, do inglês *Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado do Arduino permite a conexão com as várias placas de desenvolvimento produzidas pela empresa de hardware livre. A linguagem de programação é basicamente C ou C++ com algumas funções específicas para a plataforma, auxiliando nas tarefas e rotinas que costumam ser úteis na programação de microcontroladores AVR.

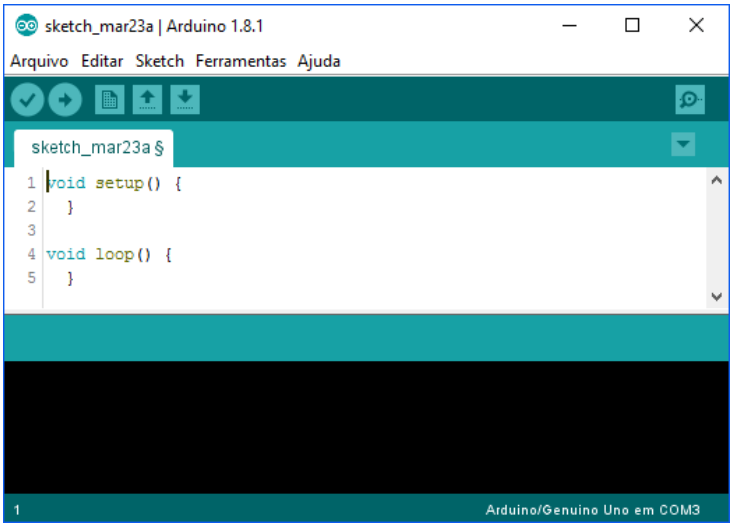
Através da placa de desenvolvimento de nome UNO, é possível carregar um *sketch*[1] via USB, facilitando o processo de gravação do chip evitando assim o uso de conver-

[1] Como é chamado um programa criado nesta IDE.

sores USB/Serial ou USB/SPI. Todo o processo de programação, compilação, *debug* e carregamento do software para o chip é feito via IDE.

A estrutura básica de um *sketch*, como mostrado na Figura 26, possui as funções **setup()** e **loop()**. A função **setup()** caracteriza-se pelo fato de que é a primeira função a ser executada assim que o microcontrolador é ligado e também as instruções dentro desta são executadas apenas uma vez. Já na função **loop()**, todas as instruções contidas nesta função irão executar em repetição infinita, semelhante ao **while(1)**.

Figura 26 – Apresentação da IDE Arduino.



Fonte: Autoria Própria.

Ainda na Figura 26, nota-se, os símbolos ✓ e → que correspondem respectivamente a verificar/compilar e carregar ao microcontrolador. Ao clicar no botão verificar/compilar é acionado o compilador da IDE e os erros ou status de compilação são exibidos na área em preto na parte inferior da janela. A Tabela 8 exibe alguns dos recursos oferecidos pela IDE adicionais as linguagens C/C++.

Tabela 8 – Alguns comandos especiais da IDE Arduino.

Construtor de Código	Descrição
HIGH	saída em nível lógico alto
LOW	saída em nível lógico baixo
INPUT	pino em modo entrada
OUTPUT	pino em modo saída
analogRead()	Aciona os conversores A/D e retorna um valor entre (0-1023)
digitalRead()	Faz a leitura de um pino e retorna HIGH ou LOW
analogWrite()	saída D/A
digitalWrite()	saída HIGH ou LOW
Serial.begin()	inicia o monitor da porta serial do microcontrolador
Serial.print()	Exibe um vetor de caracteres na porta serial
delay(ms)	pausa a execução da rotina por um valor em milisegundos

Fonte: Adaptado de Javed (2016).

4.1.2 ThingSpeak

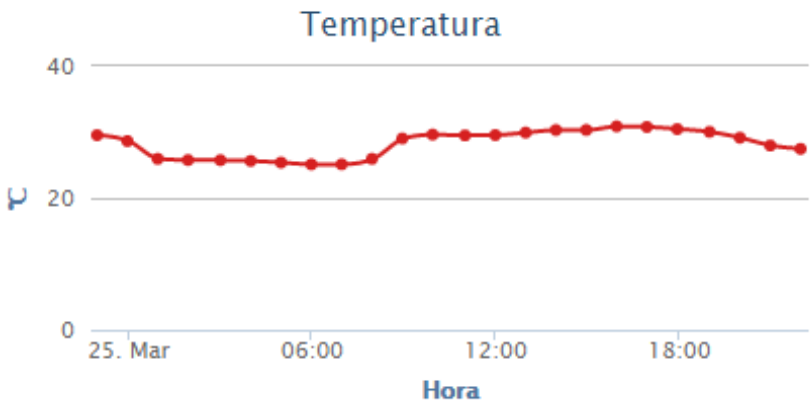
ThingSpeak, ao pé da letra: coisas falam, é uma plataforma de IoT em posse da MathWorks (dona do popular MatLab) que possibilita coletar e armazenar dados de sensores na nuvem e desenvolver aplicações em IoT. Esta plataforma fornece aplicativos para analisar e visualizar os dados no MatLab e então gerar ações baseadas nesses dados. Esses dados de sensores podem ser enviados de diversas plataformas, inclusive a Arduino (THINGSPEAK, 2017).

O elemento primário do ThingSpeak é o chamado canal. Esse contém campos onde dados, localizações e status são mostrados. É uma ferramenta descentralizada, de código aberto e licenciada pela ioBridge sob licença GPLv3. O uso comercial desta ferramenta requer um acordo prévio. Além disso, o ThingSpeak fornece um servidor que pode ser usado para armazenar dados IoT e exportar esses dados em formatos como JSON, XML e CSV permitindo sua visualização em programas como MatLab, Microsoft Excel, entre outros.

A Figura 27 exibe dados de temperatura coletados em um ambiente ao longo de um dia. O intervalo de leitura do sensor é de aproximadamente 30 segundos. Vale ressaltar que a própria ferramenta fornece opções de operações matemáticas para o traçado do gráfico, por exemplo, a possibilidade fazer uma média dos resultados coletados dos sensores e assim exibir dados mais precisos devido a suavização de alguns picos de medidas.

Outra forma de apresentação de dados é através de medidores. Esses simulam os

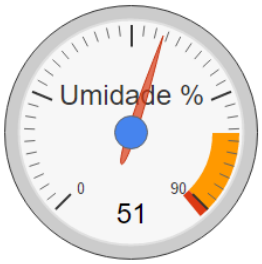
Figura 27 – Gráfico da variação da temperatura ao longo de 1 dia.



Fonte: Autoria Própria.

antigos mostradores analógicos de dados de diversas grandezas físicas. Por exemplo, na Figura 28, o medidor indica a umidade relativa do ar no ambiente onde o sensor está instalado, que no momento é de 51%.

Figura 28 – Valor da umidade relativa do ar em um medidor analógico virtual.



Fonte: Autoria Própria.

4.1.3 Bibliotecas de Programação

Durante a elaboração deste trabalho diversas bibliotecas foram estudadas a fundo em seu código fonte para uma escolha ideal. Nesta subseção serão descritas de forma sucinta as bibliotecas utilizadas neste trabalho bem como suas principais funções a fim de ajudar na compreensão do código.

4.1.3.1 A Biblioteca Espduino

Esta biblioteca permite o uso do módulo ESP8266 com comandos via porta UART baseando-se na API (Application Programming Interface - Interface de Programação de Aplicativos) chamada REST. Esta API é utilizada por muitas aplicações web e significa

REpresentational **S**tate **T**ransfer, formando a sigla REST. É uma arquitetura criada em 2000 e define várias restrições a serem seguidas (SCHWARTZ, 2015), como por exemplo:

- Uma comunicação cliente/servidor: um cliente envia uma requisição ao servidor e o servidor responde de acordo;
- Uma comunicação sem estados: cada requisição deve conter toda a informação necessária para entendimento do servidor, sem a necessidade de conhecimento prévio do servidor;
- Uma interface uniforme: para identificar facilmente recursos disponíveis no servidor.

As funções mais importantes desta biblioteca que serão utilizadas neste trabalho são:

```
1  ESP meuWifi(&Serial, 4);
2  REST restApi(&meuWifi);
3  meuWifi.enable();
4  meuWifi.reset();
5  meuWifi.ready();
6  meuWifi.connect("minha_SSID", "minha_senha");
7  meuWifi.process();
8  restApi.get((const char*)dados);
```

Quadro 4.1 – Funções importantes da biblioteca ESPDUINO.

Na linha 1, cria-se um objeto do tipo ESP de nome “meuWifi” onde são passados como parâmetro a porta serial virtual para fins de debug e o pino no qual o chip enable do módulo ESP8266 está conectado. Cria-se então uma instância da API REST para transmissão ao servidor do ThingSpeak, de nome “restApi” passando como parâmetro o endereço do objeto “meuWifi”. A função **meuWifi.enable()** ativa o chip, **meuWifi.reset()** por sua vez faz o reinício do mesmo e a função **meuWifi.ready()** retorna verdadeiro ou falso se o dispositivo já esta pronto para uso.

Já a função **meuWifi.connect** recebe como parâmetros o nome da rede WiFi a ser conectada e sua senha de acesso. Na linha 7, **meuWifi.process()** é a função que deve ser chamada em loop infinito pois mantém o processo do ESP8266 em execução, e por último, a função **restApi.get** transmite para o servidor os dados armazenados na string de nome “dados”.

4.1.3.2 A Biblioteca RadioHead

A biblioteca RadioHead agrupa um compilado de diversos drivers criados para os mais diversos rádios disponíveis no mercado, como o NRF24L01 alvo deste trabalho. Com esta biblioteca é possível o controle do módulo NRF24L01 através de algumas funções básicas. O método de comunicação é do tipo cliente/servidor, sendo um módulo adotado como servidor e os demais como clientes.

Com esta biblioteca é possível com que cada módulo atue como emissor ou receptor de dados, ou seja, quando o mesmo não está transmitindo ele esta "escutando" novos comandos. É possível também selecionar a velocidade com a qual os dados serão transmitidos via rede, visto que uma menor velocidade permite um maior alcance. Por padrão, esta biblioteca seleciona através da função **setRF** uma velocidade de transmissão de 2 Mbps com a máxima potência do rádio. No Quadro 4.2 é possível conhecer as principais funções utilizadas desta biblioteca neste trabalho.

```
1              RH_NRF24 radio(9,10);
2              radio.init();
3              radio.setChannel(3);
4              radio.setRF(RH_NRF24::DataRate2Mbps, RH_NRF24::
   TransmitPower0dBm)
5              radio.available();
6              radio.recv(pacote_rec, &tamanho)
7              radio.send(pacote_env, sizeof(pacote_env))
8              radio.waitPacketSent();
```

Quadro 4.2 – Principais funções da biblioteca RadioHead

Semelhante a biblioteca *espduino*, cria-se através do construtor o objeto cujo nome escolhido foi "radio", do tipo **RH_NRF24** passando como parâmetro os pinos de *chip enable* e SS (slave select) do protocolo SPI, respectivamente. A função **radio.init()** inicializa o funcionamento do rádio preparando a comunicação SPI do microcontrolador com o módulo NRF24L01 e limpa os buffers de recepção e transmissão.

Em seguida a função da linha 3 seleciona o canal de transmissão como sendo o 3 que corresponde a frequência 2400+3 MHz, a função **radio.setRF** seleciona a velocidade de comunicação como sendo de 2 Mbps e a potência máxima de transmissão. Todas essas funções são do tipo **boolean** e retornam **true** (verdadeiro) ou **false** (falso) caso algum comando não possa ser executado ou um erro foi encontrado.

A função da linha 5 (Quadro 4.2) retorna **true** ou **false** se uma mensagem foi ou não recebida. Caso sim (**true**), a função **radio.recv()** deve ser chamada em seguida para copiar a mensagem do buffer para o vetor de strings juntamente com seu comprimento.

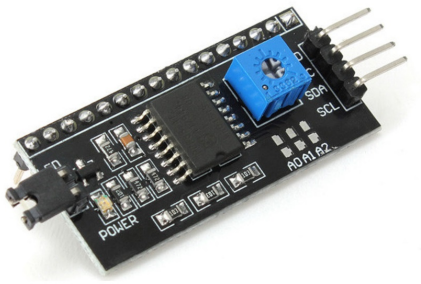
Estas funções são chamadas em loop infinito para que o módulo sempre esteja "escutando" a comunicação.

Na função **radio.send()** coloca-se o módulo em modo de transmissão e recebe como parâmetros o vetor de strings que contém a mensagem juntamente com o comprimento dessa mensagem. Retorna **true** se a mensagem foi enviada com sucesso. Já a função da linha 8 deve ser chamada em seguida para aguardar que este pacote seja transmitido.

4.1.3.3 A Biblioteca LiquidCrystal_I2C

Esta biblioteca tem como principal função utilizar o I2C1602 (Figura 29) que é um módulo capaz de conectar um display LCD de 20 colunas por 4 linhas utilizando apenas dois fios, SDA e SDL do protocolo I2C, ao invés das 14 conexões convencionais, economizando assim pinos do microcontrolador. Respeitando o formato de dados da Figura 11, este módulo interpreta os comandos recebidos e os converte em sinais elétricos para os diversos pinos do display LCD.

Figura 29 – Módulo I2C para Display LCD.



Fonte: Adaptado de www.filipeflop.com .

Seu uso simplifica o processo de interface homem máquina exibindo as informações relevantes ao usuário do sistema. As funções exibidas no Quadro 4.3 permitem a exibição dessas informações no display. Se necessário, o endereço I2C do módulo pode ser mudado, bastando fazer uma combinação de curto circuito nos pinos A0, A1 e A2 no módulo, conforme Figura 29.

```
1  LiquidCrystal_I2C lcd(0x3F, 20, 4);
2  lcd.init();
3  lcd.backlight();
4  lcd.setCursor(0,0);
5  lcd.print();
6  lcd.write();
```

Quadro 4.3 – Principais funções da biblioteca LiquidCrystal_I2C

Na linha 1, utilizando o construtor padrão cria-se o objeto LCD passando como parâmetros o endereço do módulo no barramento I2C, a quantidade de colunas e a quantidade de linhas, respectivamente. Na linha 2 o processo de inicialização da comunicação I2C acontece. A função **lcd.backlight()** liga a luz de fundo do display.

Observa-se na linha 4 a função que é responsável por reposicionar o cursor para escrita no display no formato colunas, linhas. Por sua vez, a função **lcd.print()** escreve na posição onde está o cursor o conteúdo de uma váriavel ou uma mensagem, desde que esta esteja entre aspas. Na última linha, esta função se encarrega da conversão de dados de string para ASCII, por exemplo, se chamada a função como **lcd.write(65)** será exibido no display o caractere A.

4.2 Métodos

Nesta seção serão descritos os métodos utilizados para obter os objetivos propostos neste trabalho. São métodos os algoritmos, estratégias e linguagens de programação bem como estes foram aplicados nos materiais utilizados.

4.2.1 O Protocolo de Comunicação

Para a conversação dos nós da rede entre o cliente e o servidor faz-se necessário o uso de um protocolo que satisfaça as necessidades de cada nó sensor da RSSF. Logo, um protocolo para a comunicação deve ser adotado. Foi adotado um protocolo com o tamanho de 16 bytes, iniciando sempre com o caractere **'#'** seguido do destinatário, identificador do tipo de dado, valores dos dados, caracteres randômicos e por último os bits de verificação da integridade do pacote.

PCT1=

#	**S**	**0**	**1**	**T**	**M**	**P**	**D**	**D**	**D**	**D**	**R**	**R**	**R**	**CS**	**CS**
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

No pacote acima (PCT1), as posições de 1 a 3 indicam "servidor 1", de 4 à 6: *TMP* é a abreviação para temperatura. Em seguida, as posições de 7 à 10 tem a letra *D* indicando o valor da temperatura lido no sensor que inclui as casas da dezena, unidade, vírgula e uma casa após esta vírgula.

Identificadas pela letra *R*, as posições de 11 à 13 são compostas por caracteres randômicos com função de aumentar a confiabilidade da transmissão, pois numa suposta situação onde o valor de temperatura não tenha mudado, o pacote sem estes 3 valores randômicos seria o mesmo do último enviado. As duas últimas posições (14 e 15) são reservadas a verificação chamada de *checksum* que significa soma de verificação.

O pacote abaixo (PCT2) representa uma situação onde o cliente envia para o servidor o dado de umidade relativa do ar obtida pelo sensor DHT22, semelhante ao pacote da temperatura com exceção das três letras das posições de 4 à 6 indicando agora o prefixo *UMI*.

PCT2=

#	S	0	1	U	M	I	D	D	D	D	R	R	R	CS	CS
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

O pacote 3 (PCT3) tem a função de informar uma porcentagem da luz ambiente captada pelo sensor de luz ambiente ALS-PT243, novamente utilizando o mesmo formato com exceção das posições de 4 à 6 agora identificadas pelo prefixo *LUZ*.

PCT3=

#	S	0	1	L	U	Z	D	D	D	D	R	R	R	CS	CS
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Já o pacote 4 tem a função de acionar um LED quando a luminosidade captada pelo sensor de luz ambiente atinge certo valor, ficando:

PCT4=

#	S	0	1	L	I	G	1	1	1	1	R	R	R	CS	CS
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

E para desligar este mesmo LED, o pacote 5 foi adotado.

PCT5=

#	S	0	1	O	F	F	0	0	0	0	R	R	R	CS	CS
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

4.2.2 A Soma de Verificação (*Checksum*)

As somas de verificação são utilizadas para garantir a integridade de dados que são transmitidos ou armazenados. Transmissões de dados via redes podem produzir erros como, bits trocados de lugar, perdidos ou duplicados. Como resultado, os dados recebidos pelo destinatário podem não ser os mesmos enviados pelo remetente, como numa brincadeira infantil de telefone sem fio.

Devido a esses erros o transmissor calcula o *checksum* e o envia junto a informação. O receptor, por sua vez, irá calcular localmente o *checksum* com o mesmo algoritmo utilizado no transmissor e compara-lo com o que está na informação recebida. Se o valor calculado não for idêntico ao contido na informação recebida, um erro de transmissão é detectado e o pacote recebido deve ser descartado, pois seu conteúdo não é confiável.

A soma de verificação adotada neste trabalho preza pela simplicidade mantendo os dados transmitidos confiáveis. Esta soma consiste em somar os valores contidos na posição 0 a 13 dos protocolos e armazena-los nas duas últimas posições do pacote, 14 e 15, respectivamente.

Para realizar esta soma, alguns recursos são necessários como a estrutura **union** da linguagem C. Este recurso permite armazenar diferente tipos de dados dentro do mesmo espaço de memória nivelando pelo tipo de maior tamanho. No caso deste trabalho, os tipos de dados dentro da **union** são **unsigned short int** e um *array* de **unsigned char** com duas posições. O tipo **unsigned short int** possui 2 bytes de tamanho e o tipo **unsigned char** 1 byte apenas. Como é um array de duas posições, os dois tipos ocupam o mesmo espaço na memória. No Quadro 4.4 declara-se o **union**.

```
1  union checksum{
2    unsigned short int soma;
3    unsigned char checksum[2];
4  };
```

Quadro 4.4 – Estrutura do **union**.

A parte **unsigned short int** é utilizada para armazenar os valores numéricos resultantes da soma dos valores inteiros que correspondem cada caractere na tabela ASCII. Por sua vez, a parte **unsigned char** tem como função separar esse valor em 2 bytes para sua inserção no final do protocolo. A função **ChecaSoma()** mostrada no Quadro 4.5 exibe o algoritmo de cálculo da soma de verificação.

```
1  void ChecaSoma(){
2    union checksum chk1;
3    chk1.soma = 0;
4    for(int j=0; j<=13; j++){
5      chk1.soma += pacote_env[j];
6    }
7    pacote_env[14] = chk1.checksum[0];
8    pacote_env[15] = chk1.checksum[1];
9  }
```

Quadro 4.5 – Função que calcula a soma de verificação no cliente.

Esta função (Quadro 4.5) através do laço de repetição **for** soma os valores de cada posição do *array* "pacote_env" e armazena na variável chk1.soma que é do tipo **short int**. O conteúdo deste endereço de memória ocupa 2 bytes e nas linhas 7 e 8, as duas últimas posições do protocolo a ser enviado recebem cada um desses bytes respectivamente.

Para exemplificar a operação desta função, no Quadro 4.6, nota-se o *array* "pacote" com 16 posições, quatorze destas ocupadas pelo caractere A cujo valor decimal na tabela ASCII vale 65. O resultado desta soma é dado pela expressão $14 \cdot 65 = 910$. Porém, quando lê-se a **union** como um *array* de **unsigned char**, encontra-se na posição [0] e [1] o valor

$8E_{16}$ e 3_{16} respectivamente. Como a posição [1] é o byte mais significativo, o valor fica $38E_{16}$ que quando convertido para decimal resulta no mesmo 910.

Utilizando o mesmo algoritmo no servidor que irá receber este pacote, é possível verificar se a soma de verificação calculada no servidor é igual a encontrada nos dois últimos bytes do protocolo, se sim, o pacote é válido, se não, é descartado.

```
1   unsigned char pacote[16] = "AAAAAAAAAAAAAA";
2   void ChecaSoma(){
3     union checksum chk1;
4     chk1.soma = 0;
5     for(int j=0; j<=13; j++){
6       chk1.soma += pacote[j];
7     }
8     pacote_env[14] = chk1.checksum[0];
9     pacote_env[15] = chk1.checksum[1];
10  }
```

Quadro 4.6 – Exemplo de cálculo da soma de verificação.

4.2.3 O Servidor/*Gateway* da RSSF

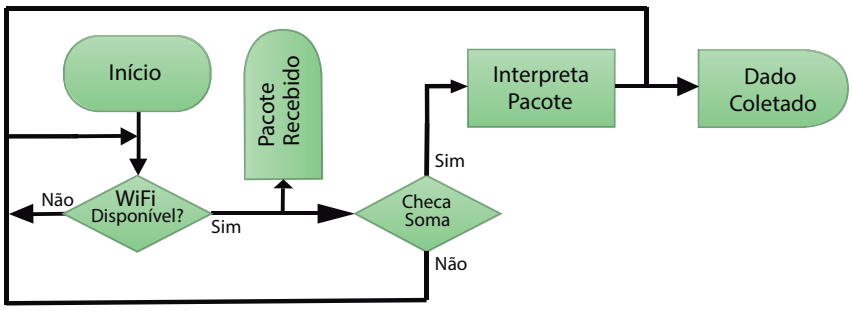
O servidor ou *gateway* da Rede de Sensores Sem Fio fica encarregado de receber os pacotes advindos dos clientes, checa-los quanto a sua integridade e interpreta-los de acordo com o protocolo. Sua função de *gateway* consiste em enviar os dados recebidos dos clientes para a internet, no caso os servidores do ThingSpeak. Suas principais atribuições são:

- "Escutar" a rede em busca dos protocolos enviados pelos clientes;
- Gerenciar e enviar dados via módulo WiFi ESP8266;
- Gerenciar o módulo NRF24L01;
- Exibir as informações corretas no display LCD;
- fazer as verificações a cerca dos pacotes recebidos;
- Montar o pacote a ser enviado para o ThingSpeak via WiFi.

Este servidor da rede consiste de um microcontrolador ATmega328P, um módulo WiFi ESP8266, um módulo da RSSF NRF24L01 e um display LCD 20x4. O fluxograma da função responsável pela escuta da rede pode ser visto na Figura 30. Esta função é chamada em loop infinito e mantém a conexão WiFi com o roteador ativa, além de escutar a rede composta pelos módulos NRF24L01. Quando um novo pacote é recebido, seu conteúdo é exibido no display LCD e seguidamente a função de checar a soma de verificação é

chamada, se esta retornar **true** (sim) o pacote é válido e deverá ser interpretado de acordo com seu tipo de dado.

Figura 30 – Fluxograma da função de escuta da rede do servidor.

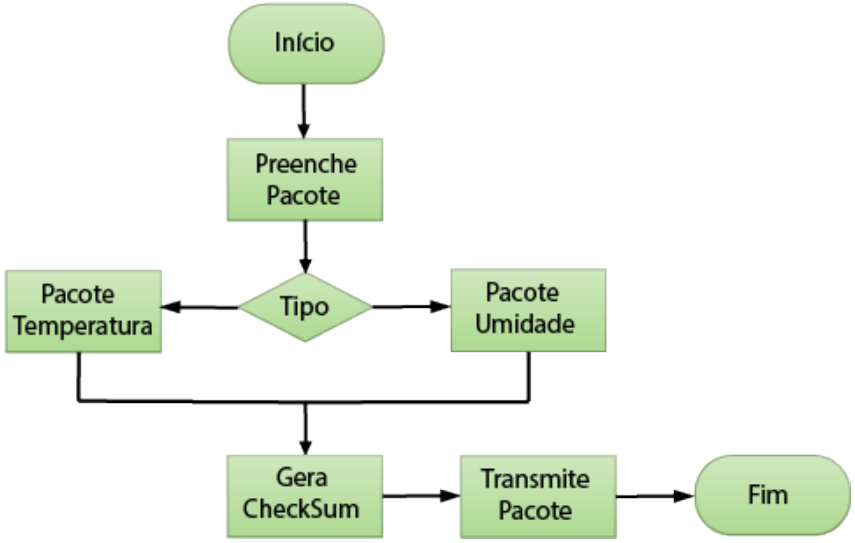


Fonte: Autoria Própria.

4.2.4 Os Nós Sensores

As extremidades da RSSF chamadas de nós sensores estão equipadas com sensor de umidade e temperatura e outra com sensor de luminosidade ambiente. Para o nó que faz o sensoriamento do clima vale destacar a função geradora de pacotes, onde é possível preencher as 16 posições do pacote padrão com caracteres randômicos e em seguida ir adicionando os dados importantes. No fluxograma da Figura 31 nota-se a função geradora de pacotes que espera como parâmetro um número inteiro indicando qual tipo de pacote deve ser gerado. Se for o tipo 1, pacote de temperatura, preenche-se as 16 posições com valores randômicos, em seguida adiciona-se a inicialização do protocolo "#S01TMP".

Figura 31 – Fluxograma da função de geração de pacote.



Fonte: Autoria Própria.

Cria-se então um *array* local do tipo **unsigned char** de nome "temp_local" com quatro posições para receber o retorno da função da biblioteca do sensor DHT22 que é do tipo **double**. Como um tipo é **unsigned char** e outro **double** é necessário o uso da função **dtostrf** que é uma abreviação para "*double to string function*" presente na biblioteca da arquitetura AVR que fará a conversão do valor de temperatura permitindo que se encaixe no *array*. Para colocar este valor dentro do pacote a ser enviado foi usada a funçao **memmove** que sobrescreve o *array* "temp_local" dentro do pacote a ser enviado somente a partir da posição 7. Logo em seguida, a função de cálculo da soma de verificação é chamada.

Para o nó que contém o sensor de luminosidade o mesmo princípio se aplica, apenas trocando o identificador do pacote para "LUZ". A leitura do sensor é feita através da função **analogRead()** (Tabela 8) que retorna um valor entre 0 e 1023. O sensor foi calibrado para 1023 ser o valor de quando uma lanterna esta sobre o sensor e 0 no escuro. Para adequar-se ao display, este valor foi dividido por dez e então exibe uma porcentagem que se refere a luminosidade do ambiente em que o sensor está.

Já para o nó de acionamento do LED, o mesmo aguarda o pacote que é enviado quando o valor de luminosidade detectado pelo sensor de luz ambiente atinge um certo nível, fazendo com que o LED seja acionado ou desligado.

5 Resultados

Neste capítulo serão exibidos os resultados obtidos pela execução deste trabalho bem como uma análise de suas características a fim de verificar se os objetivos foram realizados.

5.1 Projeto Eletrônico

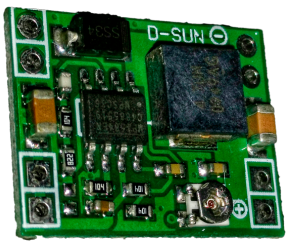
Para o nó sensor de luminosidade, conforme descrito anteriormente, o sensor de luz ambiente ALS-PT243 foi empregado. Como resultado de sua montagem física, na Figura 32 é possível observar a placa juntamente com seus resistores de escala e os conectores macho.

Figura 32 – Placa do sensor de luz ambiente.

Fonte: Autoria Própria.

Conforme Apêndice A, nota-se no circuito de alimentação 3,3 V um conversor Buck (conversor estático abaixador de tensão) em miniatura. Este conversor (Figura 33) possui 17x22 mm de tamanho e é capaz de fornecer correntes de até 3 A e abaixar tensões de até 28 V de entrada (MPS, 2009).

Figura 33 – Conversor Buck em miniatura.



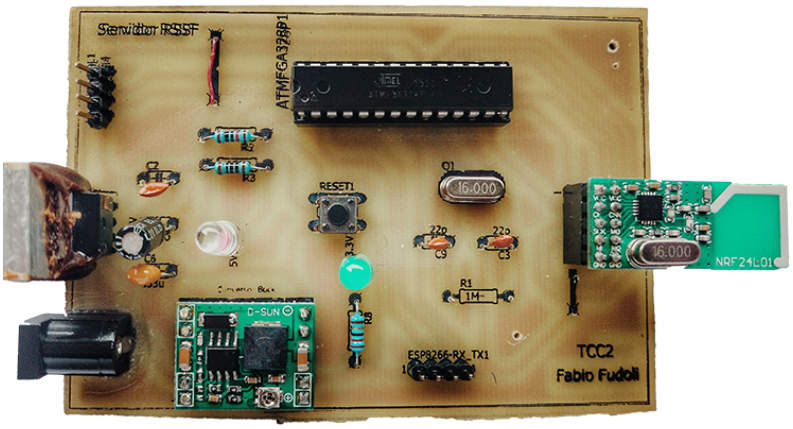
Fonte: Autoria Própria.

O uso deste conversor foi de suma importância para o projeto, visto que é necessária a alimentação dos módulos NRF24L01 e do ESP8266 já que ambos operam com 3,3 V. E o fato deste ser um conversor estático, tem um rendimento maior e portanto, exige menos

da fonte de alimentação quando comparado a um regulador de tensão, como por exemplo o LM7805.

Na Figura 34 é possível visualizar o protótipo desenvolvido para desempenhar o papel de Servidor/Gateway da RSSF. Nota-se abaixo e a esquerda a aplicação do conversor buck, no centro acima o microcontrolador ATmega328 e a direita o módulo NRF24L01 responsável pela comunicação com os nós sensores. As barras de pinos vazias são destinadas ao display LCD (superior esquerdo) e para o módulo de rede WiFi ESP8266 (inferior direito). O Apêndice B exibe a parte de baixo do protótipo.

Figura 34 – Protótipo desenvolvido como servidor da RSSF.

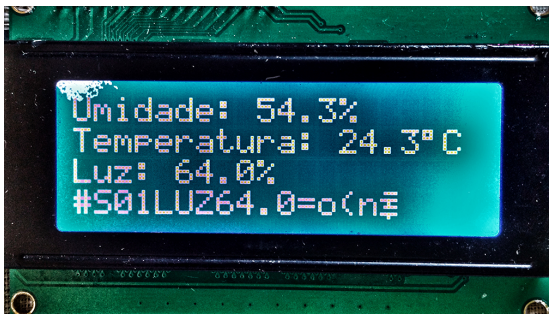


Fonte: Autoria Própria.

Este protótipo apresentou desempenho satisfatório sendo capaz de operar por horas ininterruptas transmitindo dados na rede WiFi, bem como comunicando com os nós sensores da rede interna. Para atingir este objetivo um dissipador de calor foi adicionado no regulador de tensão para evitar superaquecimento do mesmo garantindo assim a robustez do protótipo. Em medições realizadas com o auxílio de um amperímetro, a corrente nominal consumida pelo protótipo é cerca de 150 mA quando alimentado com uma tensão de 9 V, sendo que grande parte deste consumo devido a iluminação do display LCD.

As informações são mostradas localmente através do display LCD (Figura 35). Na primeira linha visualiza-se a informação da umidade relativa do ar, em sequência a temperatura do ambiente, em terceiro a informação de luminosidade e por fim, o último pacote válido recebido pelo servidor é mostrado. Nota-se que o mesmo é um pacote que informa a luz ambiente medida, dado seu identificador, acompanhado do valor, dos caracteres aleatórios e por fim da soma de verificação (checksum).

Figura 35 – Display LCD 20x4 conectado ao servidor da RSSF.

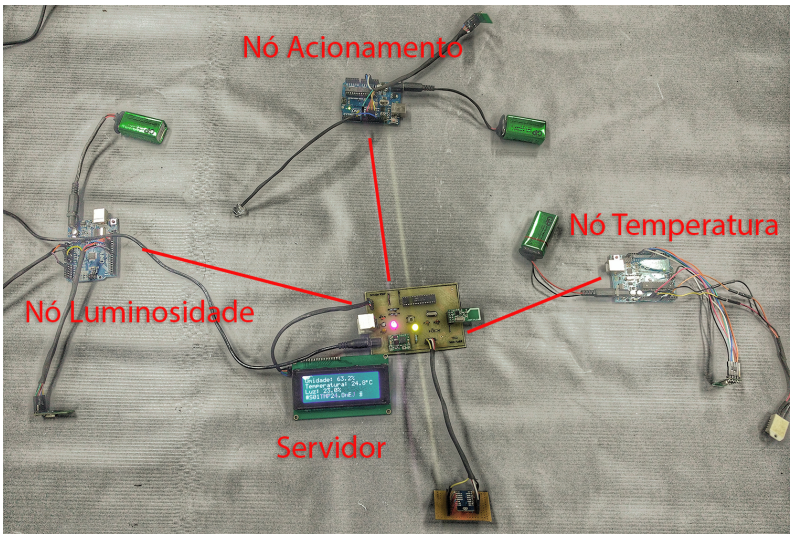


Fonte: Autoria Própria.

Através da Figura 35 é possível observar também o correto funcionamento da RSSF visto que cada pacote enviado pelos nós sensores é mostrado no display e o seu conteúdo é armazenado e mostrado na posição designada.

Já na Figura 36 tem-se uma visão geral da RSSF que acompanha o servidor/gateway, e seus nós sensores. Nota-se a esquerda o nó responsável pela detecção da luz ambiente, nó esse que envia periodicamente o valor medido através do protocolo desenvolvido na subseção 4.2.1 . Se este valor medido for menor que 10%, imediatamente o nó de acionamento (acima) liga o LED acoplado.

Figura 36 – Visão geral da RSSF.



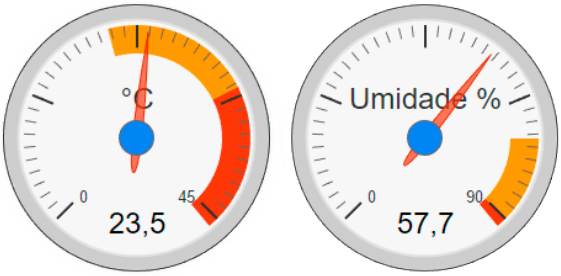
Fonte: Autoria Própria.

O lado esquerdo é ocupado pelo nó sensor de umidade e temperatura, acompanhado do sensor DHT22. Os nós foram alimentados por baterias de 9 V e o servidor por uma fonte conectada a uma tomada AC.

5.2 Aplicação a IoT

Conforme a subseção 4.1.2, o recurso da plataforma ThingSpeak foi utilizado para receber e armazenar os dados coletados pela RSSF deste trabalho. Através do *gadget* da Figura 37 que pode ser incorporado em qualquer página da web, visualiza-se os valores atuais da umidade e temperatura.

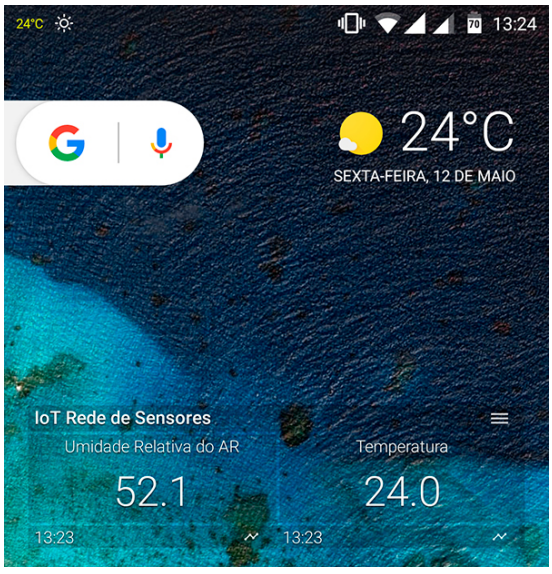
Figura 37 – Valores instantâneos de temperatura e umidade.



Fonte: Autoria Própria.

Além dos *gadgets* a plataforma permite que os dados possam ser lidos por outros sistemas, até mesmo *widgets* no sistema operacional da Google, o Android em um smartphone. Na Figura 38 nota-se o valor mostrado que foi coletado da RSSF juntamente com o valor de temperatura informado pelos provedores de previsão do tempo. Existe uma gama de aplicativos com a mesma função na *Google Play Store*. O utilizado foi "IoT ThingSpeak Monitor Widget".

Figura 38 – Gadjet para smartphone Android.



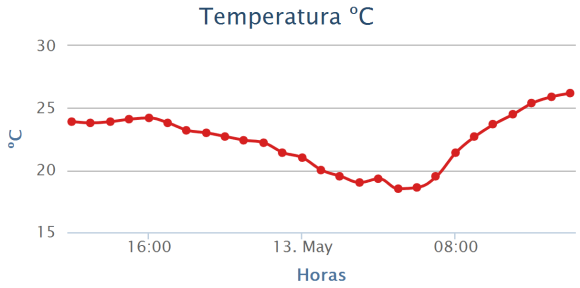
Fonte: Autoria Própria.

Os gráficos exibidos abaixo contém dados coletados pela RSSF ao longo de 26 horas, entre os dias 12 e 13 de maio de 2017 dentro de um cômodo de uma residência na cidade de Campo Mourão, Paraná. Vale ressaltar que durante todo o período de coleta de dados os sensores ficaram agrupados próximos a uma janela aberta o que permitiu que maiores variações nas medidas fossem coletadas.

Para todos os gráficos, a cada trinta segundos uma informação é transmitida da RSSF para o ThingSpeak. Esta informação contém os três dados, umidade, temperatura e luminosidade. Para filtrar as pequenas variações destas medidas, a plataforma oferece uma função de média dos valores e no caso foi utilizado a média dos 60 minutos, portanto, cada ponto no gráfico indica uma hora.

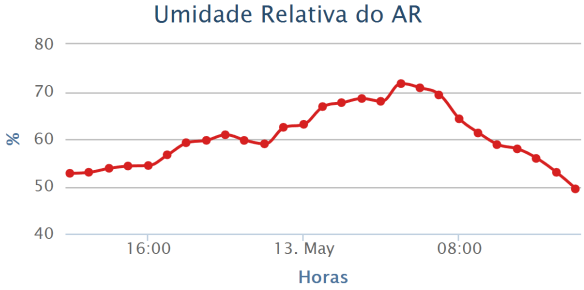
Na Figura 39 nota-se a variação de temperatura ao longo de pouco mais de um dia. Por volta de 5 da manhã a temperatura atingiu seu menor valor, 18,5ºC. Consequentemente, verifica-se na Figura 40 variações inversas em relação a temperatura. Na mesma hora a umidade relativa do ar apresentou seu pico, de 71,69%.

Figura 39 – Temperatura ao longo do dia.



Fonte: Autoria Própria.

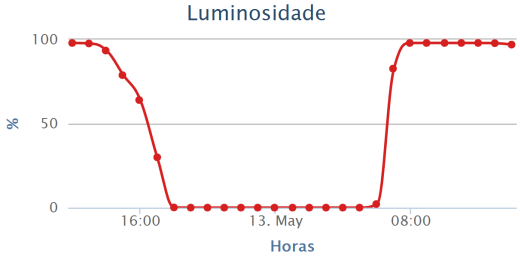
Figura 40 – Umidade ao longo do dia.



Fonte: Autoria Própria.

Já na Figura 41 é possível visualizar os dados coletados pelo sensor de luz ambiente ao longo de pouco mais de um dia. Nota-se que por volta de 6 da tarde já não há luz no cômodo e por volta de 7 da manhã a iluminação já é quase máxima. É interessante ressaltar que, no período onde não há luz, a Figura 39 mostra uma diminuição de temperatura e a Figura 40, um aumento na umidade.

Figura 41 – Luminosidade ao longo do dia.



Fonte: Autoria Própria.

6 Conclusões

O foco do desenvolvimento deste trabalho é implementar uma rede de sensores sem fio voltada a domótica e internet das coisas. Fazendo um estudo das tecnologias já consagradas em automação residencial, como por exemplo a tecnologia PLC ou até mesmo as outras soluções sem fio, como o ZigBee, foi possível definir uma abordagem do assunto que levasse em consideração as melhores características dessas tecnologias visando a aplicabilidade e o custo reduzido.

Inicialmente, excluiu-se a possibilidade do uso dos módulos ZigBee devido ao seu alto custo, mesmo considerando sua confiabilidade e o fato deste módulo ser específico para essas aplicações. Por sua vez, o módulo escolhido (NRF24L01) se mostrou capaz de realizar todos os objetivos propostos e ainda mantendo o custo do projeto baixo. Já o estudo dos protocolos de comunicação (UART, SPI...) permitiu uma correta aplicação dos recursos do microcontrolador utilizado e também minimizar a necessidade de dispositivos adicionais como multiplexadores, demultiplexadores, entre outros.

Para a implementação da RSSF, a estrutura *single-hop* se mostrou eficiente já que sua configuração é descomplicada e dada a quantidade pequena de nós sensores utilizados, não houve a necessidade de utilizar os nós agregadores presentes na estrutura *multi-hop*, logo, a rede opera de maneira mais simples facilitando a execução física e lógica.

A integração com a Internet das Coisas foi realizada com sucesso através da plataforma ThingSpeak, permitindo que as leituras de umidade, temperatura e luz ambiente realizada pelos sensores possam ser vistas em qualquer lugar do mundo, desde que este lugar possua uma conexão com a internet. O módulo ESP8266 se mostrou eficiente para conectar a RSSF na rede mundial de computadores, devido a sua robustez, conectividade simples via UART e com uma biblioteca de configuração já implementada.

Através dos materiais e métodos descritos anteriormente, a realização deste trabalho seguiu e atingiu os objetivos gerais e específicos propostos inicialmente como foi mostrado no capítulo de resultados.

Como sugestão para trabalhos futuros vale destacar a implementação em topologia *Mesh*, a adição de mais nós sensores e o sensoriamento de ambientes maiores que uma residência.

Referências

ABELÉM, A. J. G. et al. Redes mesh: Mobilidade, qualidade de serviço e comunicação em grupo. 2007. Citado 2 vezes nas páginas 25 e 27.

AKYILDIZ, I. F.; WANG, X. A survey on wireless mesh networks. *Communications Magazine, IEEE*, IEEE, v. 43, n. 9, p. S23–S30, 2005. Citado na página 26.

ALVES, J. A.; MOTA, J. *Casas inteligentes*. [S.l.]: Centro Atlântico, 2003. Citado 3 vezes nas páginas 14, 21 e 23.

ANDERSON, M. Embedded engineers why the united states is losing its edge in embedded systems. *IEEE-USA Today's Engineer*, 2008. Citado na página 44.

AOSONG ELECTRONICS. *Digital-output relative humidity e temperature sensor/module*. Guangzhou, China, 2014. Citado na página 47.

ATMEL. *ATmega328/P*. San Jose,California - EUA, 2016. Citado 2 vezes nas páginas 44 e 45.

BARONTI, P. et al. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications*, Elsevier, v. 30, n. 7, p. 1655–1695, 2007. Citado na página 24.

BARRETT, S. F. *Embedded Systems Design with the Atmel AVR Micocontroller - Part I*. Williston, Vermont - USA: Morgan e Claypool, 2010. Citado 2 vezes nas páginas 43 e 44.

BOLZANI, C. A. M. *Residências inteligentes*. [S.l.]: Editora Livraria da Física, 2004. Citado na página 19.

CABRAL, G.; MATEUS, G. Planejamento de redes mesh sem fio. *XL Simpósio Brasileiro de Pesquisa Operacional*, p. 2357–2368, 2008. Citado na página 26.

CHERMONT, M. G. *Proposta de desenvolvimento de um agente Proxy SNMP para gerenciamento de redes LonWorks.* Tese (Doutorado) — Universidade de São Paulo, 2007. Citado na página 23.

COOK, D.; DAS, S. *Smart environments: technology, protocols and applications*. [S.l.]: John Wiley & Sons, 2004. v. 43. Citado na página 14.

COULOURIS, G. et al. *Distributed Systems: Concepts and Design*. 5. ed. Boston, USA: Pearson, 2012. Citado na página 24.

CUNHA, R. d. A. *Aplicação de técnicas de Inteligência Artificial para o gerenciamento dinâmico de dispositivos de um PABX distribuído, desenvolvido com a tecnologia LonWorks.* Tese (Doutorado) — Universidade de São Paulo, 2008. Citado na página 23.

CURRIE, E. H.; ESS, D. V. *PSoC3/5 Reference Book*. California: Cypress Semiconductor Corporation, 2010. Citado 2 vezes nas páginas 28 e 29.

DENNIS, A. K. *Raspberry Pi Computer Architecture Essentials*. Birmingham B3 2PB, UK: Packt Publishing Ltd, 2016. Citado 2 vezes nas páginas 45 e 46.

DIAS, C. L. de A.; PIZZOLATO, N. D. Domótica: Aplicabilidade e sistemas de automação residencial. *Vértices*, v. 6, n. 3, p. 9–32, 2004. Citado 2 vezes nas páginas 21 e 22.

DIGI INTERNATIONAL INC. *Product Manual v1.xEx - 802.15.4 Protocol*: For rf module part numbers: Xb24-a...-001, xbp24-a...-001. Minnetonka, 2009. Citado 2 vezes nas páginas 24 e 25.

EDWARDS, W. K.; GRINTER, R. E. At home with ubiquitous computing: Seven challenges. In: SPRINGER. *Ubicomp 2001: Ubiquitous Computing*. [S.l.], 2001. p. 256–272. Citado na página 15.

EL-BENDARY, M. A. et al. *Developing security tools of WSN and WBAN networks applications*. Cairo, Egito: Springer, 2015. Citado 4 vezes nas páginas 34, 35, 36 e 37.

ESPRESSIF SYSTEMS. *Espressif Smart Connectivity Plataform: ESP8266*. Shanghai, China, 2013. Citado na página 42.

EVERLIGHT. *ALS-PT243-3C/L177*. Taiwan, 2007. Citado 2 vezes nas páginas 47 e 48.

GHOSH, S.; BASU, K.; DAS, S. K. An architecture for next-generation radio access networks. *Network, IEEE*, IEEE, v. 19, n. 5, p. 35–42, 2005. Citado na página 26.

GUBBI, J. et al. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, Elsevier, v. 29, n. 7, p. 1645–1660, 2013. Citado na página 15.

JAVED, A. *Building Arduino Projects for the Internet of Things: Experiments with Real-World Applications*. Illinois, USA: Apress Media, 2016. Citado na página 51.

KATHIK, T. K. et al. Design and verification of uart ip core using vmm. *International Journal of Soft Computing and Engineering*, v. 2, 2012. Citado na página 29.

KRISHNAKISHORE, G.; SHRUTI, K.; VARSHA, M. Design and simulation of i2c bus using verilog. *International Journal of Engineering Trends and Technology*, v. 4, 2014. Citado 3 vezes nas páginas 31, 32 e 33.

KUMAR, M. A.; SNEHA, B. Video acquisition through i2c using vhdl. *International Journal of Engineering Trends and Technology*, v. 4, 2013. Citado na página 31.

LUIZ, O. J.; PRZYBYSZ, A. L. Infra-estrutura e roteamento em redes wireless mesh. *Synergismus scyentifica UTFPR*, v. 2, n. 1, 2007. Citado 2 vezes nas páginas 25 e 26.

MACHADO, F. M. *Proposta de Protocolo de Telemonitoramento Sob Demanda de Sinais Biomédicos Usando Internet das Coisas, Computação Móvel e Armazenamento em Nuvem*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, Curitiba, 2016. Citado 2 vezes nas páginas 37 e 43.

MAZIDI, M. A.; NAIMI, S.; NAIMI, S. *The AVR Microcontroller and Embedded System*. Upper Saddle River, New Jersey - USA: Prentice Hall, 2012. Citado 2 vezes nas páginas 43 e 44.

MINERVA, R.; CRESPI, N. *Networks and New Services: A Complete Story*. Cham, Suíça: Springer, 2017. Citado na página 37.

MINI, R. A.; LOUREIRO, A. A. Energy in wireless sensor networks. In: *Middleware for Network Eccentric and Mobile Applications*. [S.l.]: Springer, 2009. p. 3–24. Citado na página 33.

MONK, S. *Raspberry Pi Cookbook*. Sebastopol, CA - USA: O'Reilly Media, 2016. Citado 2 vezes nas páginas 45 e 46.

MPS. *MP1584 - Step Down Converter*. Taiwan, 2009. Citado na página 62.

NORDIC SEMICONDUCTORS. *nRF24L01 - Single Chip 2.4GHz Transceiver*. Trondheim, Noruega, 2007. Rev. 2. Citado na página 41.

RAMANATHAN, R.; REDI, J. A brief overview of ad hoc networks: challenges and directions. *IEEE communications Magazine*, v. 40, n. 5, p. 20–22, 2002. Citado na página 25.

RAMOS, L. *Uma proposta de ontologia para residências inteligentes buscando a integração de dispositivos*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2015. Citado na página 15.

RAYES, A.; SALAM, S. *Internet of Things - From Hype to Reality*. Cham, Suíça: Springer, 2017. Citado 4 vezes nas páginas 37, 38, 39 e 40.

RIVERO, I. *Rede de Sensores sem Fio para Monitoramento de Equipamentos Eletrônicos*. Dissertação (Mestrado) — Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2011. Citado na página 33.

SACCO, F. *Comunicação SPI*. 2014. Acessado em: 21/10/2015. Disponível em: <http://www.embarcados.com.br/spi-parte-1/>. Citado 3 vezes nas páginas 30, 31 e 33.

SAMANTA, A. G. Interfacing of pic 18f252 microcontroller with real time clock via i2c protocol. *Int. Journal of Engineering Research and Applications*, v. 4, p. 152–156, 2014. Citado na página 31.

SANDYA, M.; RAJASEKHAR, K. Design and verification of serial peripheral interface. *International Journal of Engineering Trends and Technology 3 (4)*, p. 522–524, 2012. Citado na página 29.

SCHNEIDER ELECTRIC. *IHC - Intelligent Home Control*: Automação residencial com estilo, segurança e economia de energia. São Paulo, 2008. 9 p. Citado na página 22.

SCHWARTZ, M.-O. *A REST API for Arduino e the CC3000 WiFi Chip*. 2015. Acessado em: 23/03/2017. Disponível em: <https://learn.adafruit.com/a-rest-api-for-arduino-and-the-cc3000-wifi-chip/overview>. Citado na página 53.

SENSOR de Umidade e Temperatura AM2302 DHT22. 2016. Acessado em: 20/03/2017. Disponível em: <http://www.filipeflop.com/pd-137442-sensor-de-umidade-e-temperatura-am2302-dht22.html>. Citado na página 46.

SILVA, A. V. D. Proposta de interfaceamento entre rede de controle lonworks e rfid. *Revista Científica Semana Acadêmica*, v. 59, 2014. Citado na página 23.

TANENBAUM, A. S.; STEEN, M. V. *Distributed Systems: Principles and Paradigms*. 2. ed. Upper Saddle River, USA: Pearson, 2007. Citado na página 24.

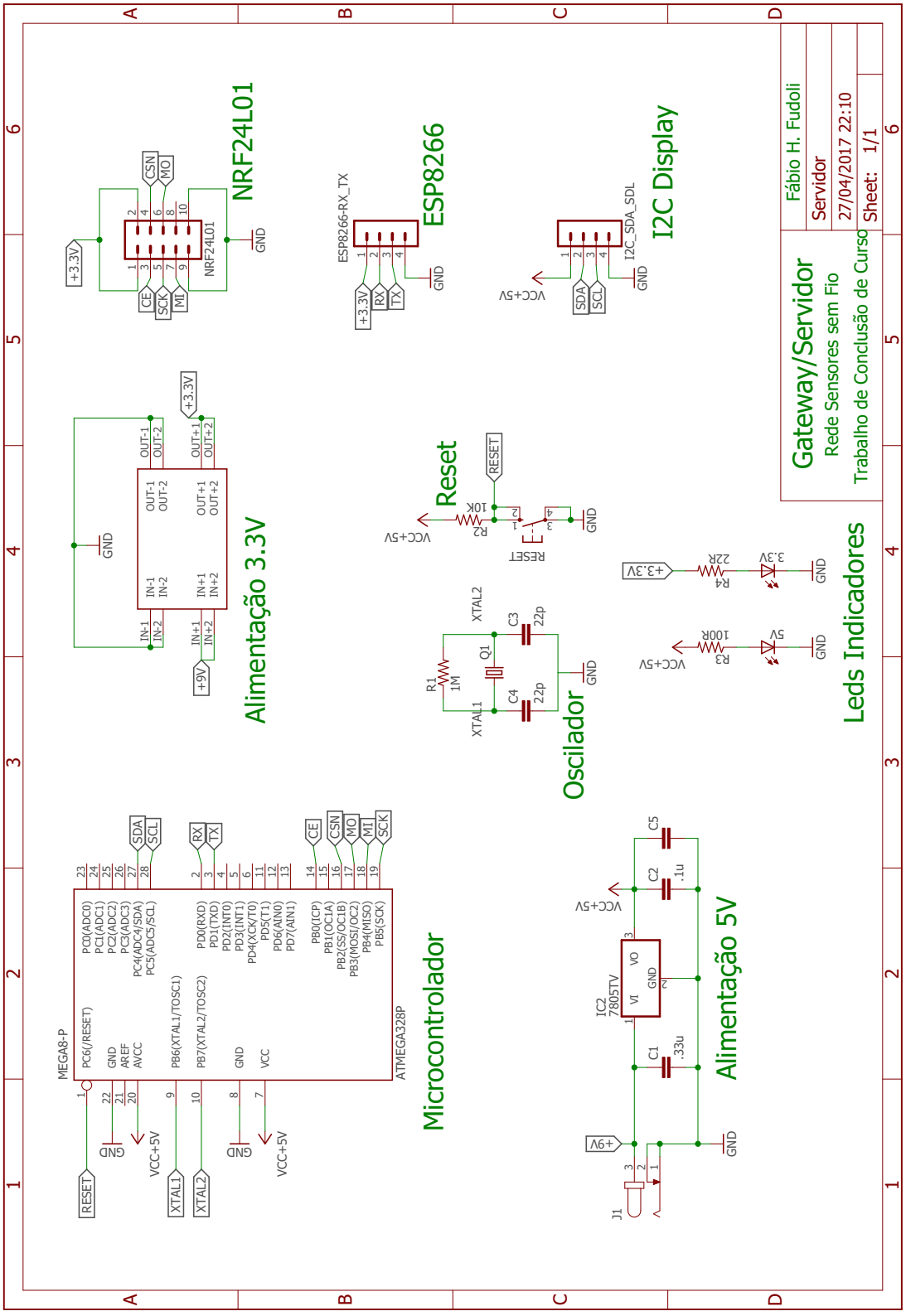
TEIXEIRA, E. R. D. *Wireless mesh networks*. [S.l.]: Teleco, 2004. Citado na página 26.

THINGSPEAK. 2017. Acessado em: 23/03/2017. Disponível em: <https://www.mathworks.com/help/thingspeak/>. Citado na página 51.

THOMAZINI, D.; ALBUQUERQUE, P. U. B. *Sensores industriais: fundamentos e aplicações*. 7. ed. São Paulo: Érica, 2005. Citado na página 36.

WERNECK, S. B. d. F. *Domótica: União de arquitetura e tecnologia da informação na edificação residencial urbana*. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro. Rio de Janeiro, 1999. Citado na página 20.

APÊNDICE A – Esquemático do Gateway



APÊNDICE B – Parte de Baixo da Placa do Protótipo