Universidade Federal de Campina Grande

Centro de Engenharia Elétrica e Informática

Curso de Graduação em Engenharia Elétrica

Departamento de Engenharia Elétrica

RAFAEL ÂNGELO VIEIRA PESSOA LIMA

TRABALHO DE CONCLUSÃO DE CURSO

MONITORAMENTO DE DADOS RFID ATRAVÉS DE UMA REDE ZIGBEE

Campina Grande
Maio 2013

RAFAEL ÂNGELO VIEIRA PESSOA LIMA

MONITORAMENTO DE DADOS RFID ATRAVÉS DE UMA REDE ZIGBEE

Trabalho de Conclusão de Curso submetido à Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Instrumentação Eletrônica

Orientador

Professor Jaidilson Jó da Silva

Campina Grande
Maio 2013

RAFAEL ÂNGELO VIEIRA PESSOA LIMA

MONITORAMENTO DE DADOS RFID ATRAVÉS DE UMA REDE ZIGBEE

Trabalho de Conclusão de Curso submetido à Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Instrumentação Eletrônica

Aprovado em / /

Professor Avaliador
Universidade Federal de Campina Grande
Avaliador

Professor Jaidilson Jó da Silva
Universidade Federal de Campina Grande
Orientador

Campina Grande
Maio 2013

AGRADECIMENTOS

À Deus, em primeiro lugar, por guiar-me durante toda a minha caminhada e escutar as minhas súplicas mais profundas. Apesar do meu ceticismo filosófico, não tenho dúvidas de sua existência.

Aos meus pais, Gabriel Ângelo Pessoa Lima e Maria Teresa Vieira Pessoa Lima, e a minha irmã, Amanda Vieira Pessoa Lima, pelo apoio constante, amor e conselhos em todos os momentos da minha vida. Devo-vos quem sou.

À minha família que me apoiaram em tudo e sem ela não chegaria até aqui.

Ao professor Jaidilson por toda atenção e orientação que dispôs para o desenvolvimento deste trabalho.

Aos meus grandes amigos Matheus Telles, João Viniccius, Arthur Farias, Túlio Vidal, Carlos Antônio e Felipe Pinheiro entre tantos outros que estiveram ao meu lado nas horas de estudo e lazer. Sem vocês tenho certeza que não conseguiria atingir os meus objetivos.

Aos meus amigos de infância Edvaldo Neto e Rafael Abreu por estarem presentes nos melhores momentos da minha vida.

Ao irmão que a vida me presenteou, Victor Cavalcante, pelo apoio e compreensão constante. De uma inteligência incomparável e de uma vontade inspiradora de fazer as coisas acontecerem.

À Bianca Leite, por simplesmente existir. Ela foi aquela coincidência da vida que acaba mudando o seu jeito de enxergar o mundo e te ensina a cuidar e proteger não só a si mesmo.

"Não deixe o barulho da opinião dos outros abafar sua voz interior. E mais importante, tenha a coragem de seguir seu coração e sua intuição. Eles de alguma forma já sabem o que você realmente quer se tornar. Tudo o mais é secundário."

Steve P. Jobs

RESUMO

O monitoramento de dados é importante em muitas aplicações industriais e comerciais, visto que através deste é identificado o estado do sistema e permite controlá-lo de forma a evitar distúrbios ou, ao menos, minimizá-los. O monitoramento de dados é possível através dos usos da instrumentação eletrônica e da aquisição de dados e interface. O primeiro é utilizado para a medição dos parâmetros de interesse no sistema e o segundo é responsável por converter tais informações que foram obtidas em valores numéricos digitais para que sejam facilmente visualizados.

Este trabalho apresenta o desenvolvimento de um sistema de monitoramento de dados desde a sua estrutura física de engenharia até sua lógica de programação. Tal sistema mostrou ser uma solução eficaz para aplicações que necessitam acesso a vários dados quase simultaneamente, distribuídos geograficamente ou não, como controle de entrada, estoque, leitura de sensores em ambientes com pouco ou moderado ruído, entre outros.

Palavras-chave: RFID, ZigBee, Arduino, Monitoramento de dados, *Tags.*

SUMÁRIO

LISTA DE SIGLAS

APL – *Application Layer*

ARAT – *Active Reader Active Tag*

ARPT - *Active Reader Passive Tag*

ECMA – *European Computer Manufacturers Association*

IDE – *Integrated Development Environment*

ISO – *International Organization for Standardization*

IP – *Internet Protocol*

NWK – *Network Layer*

MAC – *Medium Access Control*

OEMs – *Original Equipment Manufacturers*

OSI – *Open System Interconnection*

PAN – *Personal Area Network*

PHY – *Physical Layer*

PRAT – *Passive Reader Active Tag*

RAM – *Random Access Memory*

RFID - *Radio Frequency Identification*

SAP – *Service Access Point*

USB – *Universal Serial Bus*

WPAN - *Wireless Personal Area Network*

LISTA DE SÍMBOLOS

Símbolo	Descrição
V	$Tensão$
Hz	$Frequência$
m	$Metro$

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

1 INTRODUÇÃO

O monitoramento de dados é importante em muitas aplicações industriais e comerciais, visto que através deste é estimado o estado do sistema e permite controlá-lo de forma a evitar distúrbios ou, ao menos, minimizá-los. O monitoramento de dados é possível através dos usos da instrumentação eletrônica e da aquisição de dados e interface. O primeiro é utilizado para a medição dos parâmetros de interesse no sistema e o segundo é responsável por converter tais informações que foram obtidas em valores numéricos digitais para que sejam facilmente visualizados.

À medida que os sistemas industriais foram tornando-se mais complexos e maiores, foi necessário o monitoramento de uma maior quantidade de dados e consequentemente uma estrutura mais aprimorada de aquisição, instrumentação e uma rede robusta de modo a permitir a união dessa grande quantidade de dados. No início, tais estruturas de monitoramento de dados eram caras e assim foi necessário que os engenheiros criassem tecnologias em redes de comunicação e aquisição de dados de tal sorte que fosse permitido a construção de sistemas de baixo custo e alta confiabilidade que solucionassem as exigências impostas pelos novos sistemas industriais.

Duas tecnologias, que foram criadas com os mais diversos propósitos, tiveram grande utilidade para solucionar os problemas descritos anteriormente. O *Radio Frequency Identification* (RFID) e ZigBee são tecnologias robustas e de fácil utilização que permitem o desenvolvimento de sistemas de monitoramento de dados complexos de forma simples e direta. Além destas duas tecnologias-base citadas, foram usadas complementarmente para a realização do projeto a linguagem de programação C# e o Arduino.

RFID é uma tecnologia que faz uso de radiofrequência para a transferência de dados. Inicialmente, teve sua concepção voltada na complementação dos códigos de barra [AHSON e ILYAS, 2008].

Atualmente, os RFIDs têm um amplo campo de utilização que engloba a limitação de roubos, gerência de inventário, logística, aumento da produtividade, entre outros. É uma tecnologia vasta, apresentando diversas categorias de leitores e *Tags.*

O ZigBee é um conjunto de especificações para comunicação sem fio entre dispositivos eletrônicos, enfatizando a grande vida útil, baixa taxa de transferência (de 20 kbps a 900 kbps) e segurança na rede. A tecnologia ZigBee visa ser mais simples e mais

barata que outras *Wireless Personal Area Network* (WPAN), tal como *Bluetooth,* permitindo que seja usada amplamente em aplicações de controle e monitoramento [GISLASON, 2008].

As redes ZigBee operam em faixa de frequência industriais, científicas e médicas (868 MHz na Europa, 915 MHz nos Estados Unidos e Australia e 2,4 GHz na maioria dos outros países). A transmissão de dados varia de 20 a 900 kbps e permitem a configuração da rede nas principais topologias: Estrela, Malha e Árvore [GISLASON, 2008].

Neste trabalho faz-se o uso das tecnologias supracitadas para elaboração de um sistema de monitoramento de dados no qual os dados obtidos são provenientes das *Tags* RFID, mas é possível que sejam obtidos de quaisquer equipamentos cuja sua comunicação seja serial.

2 MATERIAIS E MÉTODOS

Nesta seção são apresentadas com mais detalhes as tecnologias utilizadas, o padrão de redes sem fio, como também a linguagem de programação e a plataforma de prototipagem eletrônica arduino.

2.1 RFID

O RFID é uma tecnologia de identificação automática (Auto-ID). Possui seu surgimento atribuída a complementação de uma outra famosa tecnologia Auto-ID, o código de barras. Em substituição das listras paralelas pretas de largura não uniformes utilizadas no código de barras, o sistema RFID tem seu código em *Tags* (etiquetas ou *transponders)*, no qual apresentam um pequeno *chip* com o código gravado e uma antena acoplada que enviará o sinal com o código único guardado para o *reader* (leitor). Ele apresenta vantagens frente às tecnologias que procura substituir, visto que seu sistema é de fácil leitura e independente quanto a necessidade de estar no campo de visão do leitor que as outras tecnologias apresentam. É também percebido a superioridade no alcance da leitura, mesmo em sistemas passivos (sem a presença de uma bateria para o aumento da energia do sinal).

Um sistema RFID usa ondas de rádio na identificação automatizada de animais, pessoas e objetos. As *tags* RFID podem conter informações sobre objetos ao qual estão associados – dimensões, localização, identificação numérica, histórico das temperaturas às quais a *tag* foi submetida e muitas outras características estáticas e dinâmicas [BHUPTANI e MORADPOUR, 2005].

Os componentes típicos que constituem a estrutura RFID são:

- *Reader* ou Leitor: Constituído por uma antena, um módulo de radiofrequência e um módulo de controle. *Readers* podem apresentar a característica de enviar sinais, chamados de interrogadores, e receber sinais de respostas das *tags* ou apenas receber sinais dos tags. Os *readers* que apresentam a primeira característica são chamados de ativos e os que não a apresentam são denominados de passivos;
- *Tags* ou Etiquetas: Formados por um *chip* semicondutor, uma antena e em alguns casos, uma bateria. As *tags* que apresentam bateria e assim podem

enviar sinais periódicos de forma autônoma recebem a denominação de ativas e os que não apresentam recebem a denominação de passivas. Além disto, as informações são guardadas eletronicamente em uma memória não-volátil, as quais podem ser de apenas leitura ou leitura/escrita;

- *Host:* Dispositivo para onde são enviadas as informações para armazenamento e/ou processamento de dados.

O acoplamento das antenas do *reader* e da *tag* é necessário para que ocorra a transferência de dados no sistema RFID. É possível observar na Figura 1 um diagrama ilustrativo de tal sistema.

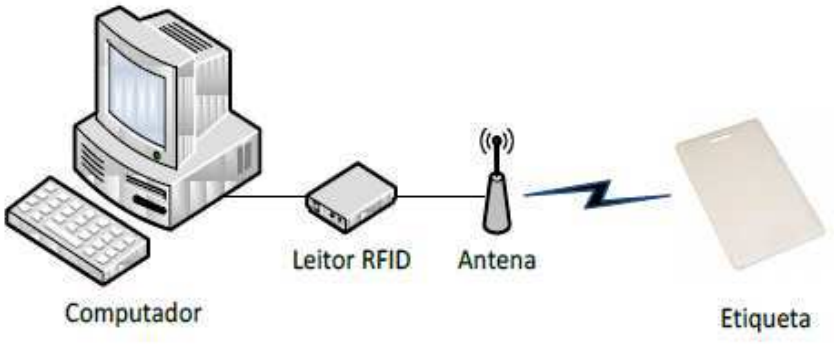


Figura 1 - Diagrama ilustrativo de um sistema RFID. [Fonte: COSTA et al, 2012]

Outro aspecto importante é quanto as configurações típicas que podem ser encontrados em sistemas RFID. A primeira é o ***Passive Reader Active Tag*** **(PRAT)** que apresenta o *reader* passivo no qual apenas recebe sinais de *tags* ativas e possui uma faixa de recepção que varia de 1 a 610 metros. A segunda configuração é denominada ***Active Reader Active Tag*** **(ARAT)** no qual *tags* ativos ficam em *standby* e apenas enviam dados quando recebem um sinal interrogador vindo do *Reader*. A última configuração é conhecida como ***Active Reader Passive Tag*** **(ARPT)** no qual apresentam *reader* ativos em que transmitem sinais interrogadores e recebem respostas de autenticação vindos de *tags* passivos. A configuração ***Passive Reader Passive Tag*** **(PRPT)** não é uma configuração válida, pois as *tags* passivas apenas transmitem dados na presença de uma excitação produzida pela antena do leitor, caso o leitor seja passivo, este campo não será produzido e consequentemente não será possível a leitura de dados das *tags.*

São evidenciadas na Tabela 1 as mais importantes classes de frequência de operação dos sistemas RFID e, na Tabela 2, informações mais generalizadas no tocante aos RFIDs.

Tabela 1 - Classificação das faixas de frequência.

Classificação das frequências	Faixa (MHz)
Baixa (LF)	0,125 - 0,1345
Alta (HF)	~13,56
Ultra alta (UHF)	915,0
Microondas (*Microwave*)	2,45 ou 5,8[1]

Geralmente, a frequência de operação define a velocidade de transferência de dados. Todavia, a velocidade não é a única característica a ser levada em consideração no desenvolvimento de uma aplicação RFID, visto que o custo, alcance e banda também são fundamentais para o desenvolvimento. [COSTA et al, 2012].

Na próxima seção será visto a tecnologia ZigBee que foi utilizada para a formação da rede sem fio.

Tabela 2 - Informações gerais sobre RFID. [DIPANKAR et al, 2009]

Banda	Alcance	Velocidade de transmissão	Utilização	Custo (2009)
120 - 150 KHz (LF)	10 cm	Baixa	Identificação Animal, uso industrial	US$1
13,56 MHz (HF)	1 m	Baixa a Moderada	Cartões inteligentes	US$0.50
433 MHz (UHF)	1 – 100 m	Moderada	Aplicações em defesa, com *tag* ativo.	US$5
865 - 928 MHz (UHF)	1 – 2 m	Moderada até Alta	EAN	US$0.15 (passivo)
2450 – 5800 MHz (*microwave)*	1 – 2 m	Alta	802.11 WLAN, *Bluetooth*	US$25 (ativo)
3,1 – 10 GHz (*microwave)*	Até 200 m	Alta		US$5

[1] A frequência de 2,45 GHz é reservada para aplicações industriais, cientificas e médicas.

2.2 ZIGBEE

ZigBee é um padrão que define um conjunto de protocolos de comunicação para redes sem fio de baixa taxa de transferência [GISLASON, 2008]. O ZigBee é utilizado para aplicações cujos principais requisitos são baixo custo, baixo consumo de energia, segurança da rede, além da baixa taxa de transferência. Em muitas aplicações ZigBee, o tempo total que os dispositivos sem fio estão engajados em algum tipo de atividade é muito limitado; os dispositivos passam a maioria do seu tempo no modo de utilização conhecido como *sleep mode*, onde sua principal característica é o baixo consumo de energia. Como resultado, estes dispositivos são capazes de estarem em operação por anos antes que haja a necessidade de troca de baterias [FARAHANI, 2008].

O padrão ZigBee é desenvolvido pela ZigBee Alliance. A ZigBee Alliance é formada por uma centena de companhias associadas, desde empresas de semicondutores e desenvolvedoras de *softwares* até fábricas de equipamentos (OEMs) e instaladoras. Teve início em 2002 como uma organização não lucrativa aberta a qualquer um que desejasse tornar-se membro [FARAHANI, 2008].

As camadas do protocolo utilizadas no padrão ZigBee são baseadas no modelo OSI do *International Organization for Standardization* (ISO). No modelo OSI existem sete camadas, mas o ZigBee implementa apenas as camadas essenciais para as redes sem fio de baixo consumo e baixa taxa de transferência. Ele adota o IEEE 802.15.4 como sua camada física (PHY) e o controle acesso ao meio (MAC). As outras camadas são *Network Layer* (NWK) e *Application Layer* (APL) definidas pelo próprio padrão ZigBee.

Cada camada comunica-se com a camada adjacente através do SAP. Um SAP é uma estrutura conceitual através da qual uma camada pode requerer o serviço de outra camada do protocolo. Na Figura 2 apresenta-se a estrutura de camadas do protocolo ZigBee.

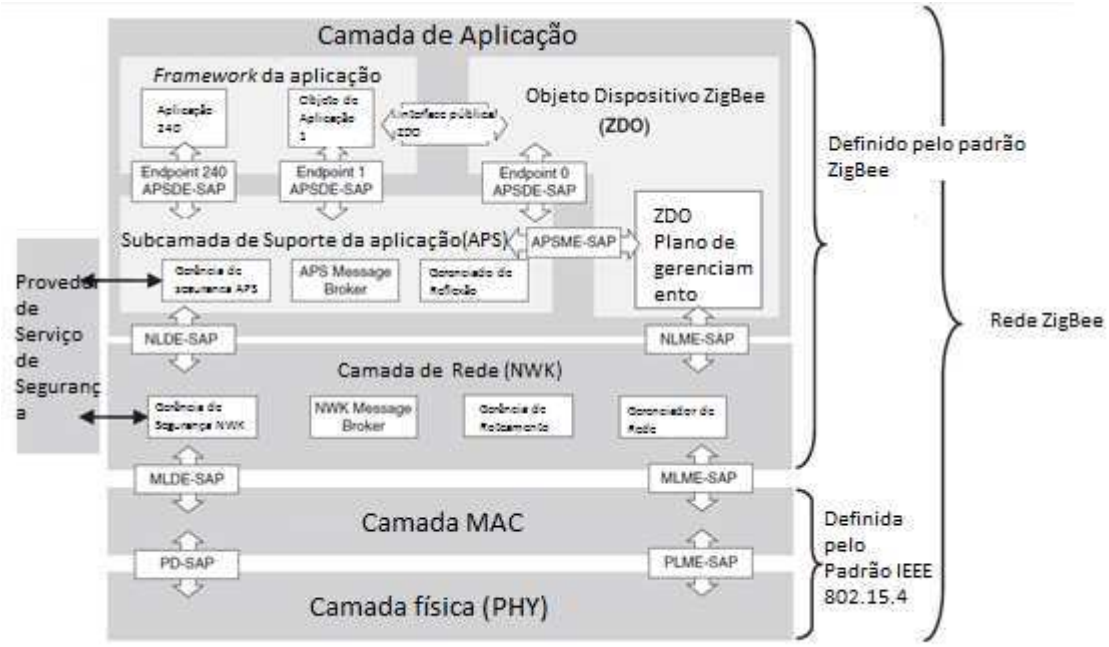


Figura 2 – Representação das camadas do protocolo de rede do ZigBee [Fonte: FARAHANI, 2008].

O padrão IEEE 802.15.4 identifica dois tipos de dispositivos:

- Dispositivos de funções completas: Estes são dispositivos que podem suportar três modos de operação, como segue:
    - Coordenador *Personal Area Network* (PAN): Este dispositivo possui sua própria rede em que outros dispositivos podem se associar;
    - Coordenador: promove a sincronização através de sinais específicos. Devem ser associados a coordenadores PAN e não tem a capacidade de criar sua própria rede. Também são conhecidos como roteadores.
    - Dispositivo Final: um dispositivo simples no qual não é implementada nenhuma das funcionalidades anteriores.
- Dispositivos de funções reduzidas: São dispositivos que operam com uma implementação mínima do IEEE 802.15.4. São voltados para aplicações que são extremamente simples, tal como um sensor de infravermelho passivo. Eles não tem a necessidade de enviar grandes quantidades de dados e podem apenas associar-se com um único dispositivo por vez.

Outro aspecto importante são as topologias de rede que são definidas no padrão IEEE 802.15.4. Os dois tipos básicos de topologia, empregadas de acordo com os requerimentos da aplicação, são as topologias estrela e ponto-a-ponto. Um terceiro tipo considerado é um caso particular da topologia ponto-a-ponto, conhecida como árvore [KOUBAA et al, 2008].

Na topologia estrela, um único nó opera como um coordenador PAN. Caso um dispositivo de funções completas seja ativado, ele poderá estabelecer sua própria rede e tornar-se o coordenador PAN. Este dispositivo escolherá um identificador PAN e este identificador não poderá ser usado por qualquer outra rede na área de atuação deste dispositivo [KOUBAA et al, 2008].

A principal característica observada desta topologia é sua centralização, ou seja, toda mensagem enviada de um dispositivo a outro deverá passar pelo coordenador PAN e assim não é permitida a comunicação direta entre os nós. Na Figura 3 é apresentada a topologia estrela.

Figura 3 - Topologia Estrela [Fonte: KOUBAA et al, 2008].

A topologia ponto-a-ponto, também denominada de malha, inclui um nó coordenador PAN que é escolhido por ser o primeiro dispositivo a se comunicar no canal. Entretanto, ao contrário da topologia supracitada, a ponto-a-ponto é descentralizada e assim cada dispositivos pode se comunicar diretamente com qualquer outro dispositivo na rede. Esta topologia habilita a flexibilidade da rede, mas induz uma adicional complexidade para prover conectividade entre todos os dispositivos da rede [KOUBAA et al, 2008].

Basicamente, a topologia ponto-a-ponto opera de forma *ad hoc* e permite múltiplos *hops* para rotear os dados de qualquer dispositivo para outro. Todavia, estas funções devem ser definidas na camada de rede (NWK) e assim não são consideradas no IEEE 802.15.4. A topologia ponto-a-ponto é apresentada na Figura 4.

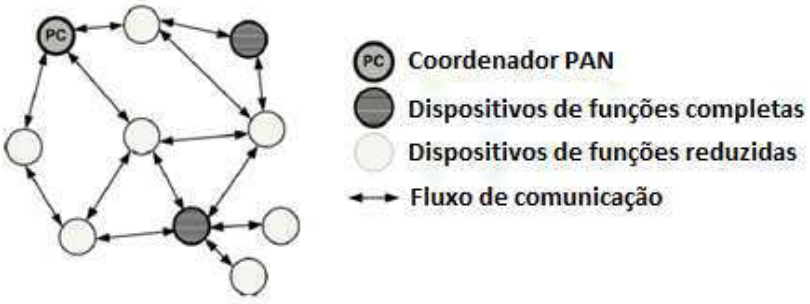


Figura 4 - Topologia Ponto-a-Ponto [Fonte: KOUBAA et al, 2008].

A topologia árvore é um caso especial da ponto-a-ponto na qual a maioria dos dispositivos são de funções completas. Apenas um dos dispositivos desta topologia é atribuída a função de nó coordenador PAN, responsável por identificar a rede inteira. Qualquer outro dispositivo de função completa pode atuar como roteador e promover a sincronização para os outros roteadores ou dispositivos finais. Por fim, os dispositivos de funções reduzidas conectam-se à rede como nós nas folhas da estrutura e ligam-se a apenas um dispositivo de funções completas. Na Figura 5 é apresentada a estrutura da topologia árvore.

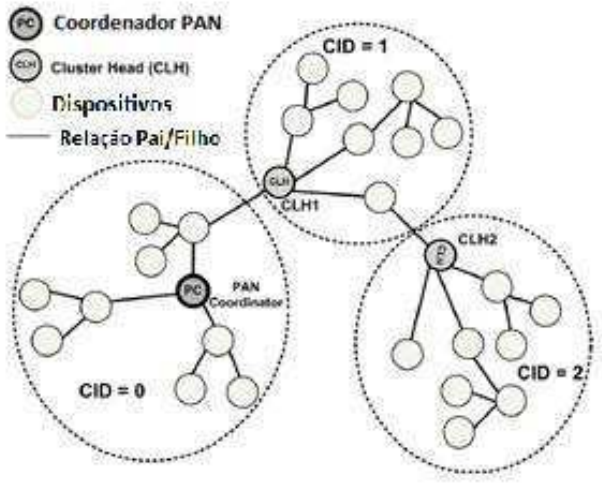


Figura 5 - Topologia árvore. [Fonte: KOUBAA et al, 2008].

O padrão IEEE 802.15.4 não define como construir uma rede estruturada na topologia árvore. Ele apenas indica que isto é possível e pode ser iniciada pelas camadas mais altas.

A rede ZigBee apresenta dois tipos de operações que influenciam diretamente no consumo de energia. O tipo *Beaconing* é caracterizado pela transmissão regular de sinais *beaconing* (sinalizadores) por um dispositivo com função de roteador para confirmar a sua presença na rede aos outros roteadores. Os outros nós da rede precisam estar ativos apenas no momento da transmissão e isto é feito através de sua configuração. Neste tipo, a maioria dos dispositivos permanecem dormindo (*Sleep)* e assim há uma redução significativa do consumo de energia. O Segundo tipo denominado de *Non-Beaconing* apresenta os seus receptores sempre ativos, consumindo mais energia. Na próxima seção, será apresentada a linguagem C# que foi utilizada no desenvolvimento da aplicação de monitoramento de dados.

2.3 LINGUAGEM DE PROGRAMAÇÃO C#

C# é uma linguagem de programação multiparadigma que apresenta os conceitos de programação funcional, *Strong typing,* imperativa, declarativa, orientada a objeto e baseada em componentes que foi utilizada no presente trabalho para o desenvolvimento do *software* de monitoramento de dados.

Desenvolvida pela *Microsoft* dentro do conceito .NET e aprovada pelo padrão ECMA e ISO, C# é uma das linguagens de programação *Common Language Infrastructure* que define o ambiente de alto nível que permite o uso da aplicação em diferentes plataformas sem que seja necessário a reescrita do código para arquiteturas distintas. A versão atual é 5.0, lançada em agosto de 2012, e é liderada pela pelo engenheiro de *software* Anders Hejlsberg.

A IDE utilizada para o desenvolvimento de aplicações em C# é o *Visual Studio*. Esta IDE foi desenvolvida pela Microsoft com o uso das linguagens C++ e o próprio C#. O *Visual Studio* possui diversos recursos que auxiliam a programação como o *IntelliSense* e refatoração de código.

A tela de programação do *Visual Studio* é apresentada na Figura 6.

Na próxima seção, será apresentado o Arduino, que foi utilizado para a simulação do envio de dados através da rede sem fio e para o teste de confiabilidade.

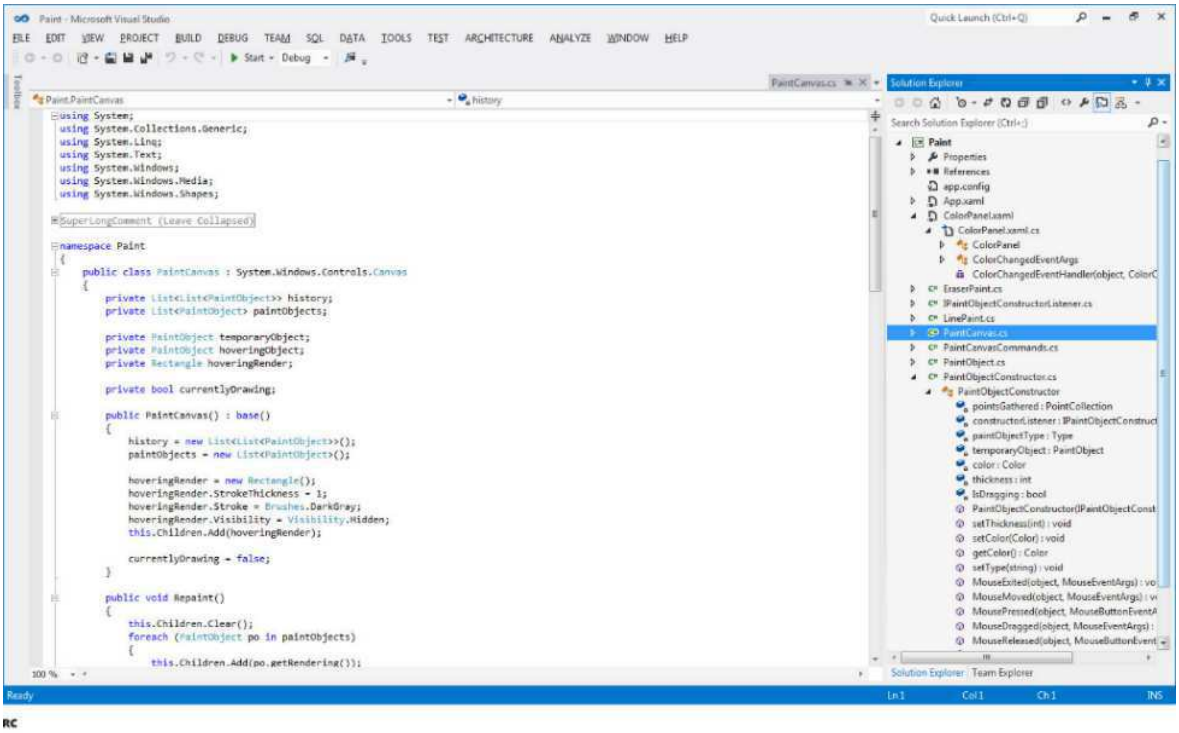


Figura 6 – Representação da tela de programação do *Visual Studio*.

2.4 ARDUINO

Arduino é uma plataforma de prototipagem eletrônica de *hardware* livre, com um microcontrolador Atmel AVR placa única, Entrada/Saída digital e entrada analógica embutida e linguagem de programação padrão orientada a objeto com base nas linguagens C/C++. O objetivo do arduino é possibilitar a criação de ferramentas de baixo custo, flexíveis e fáceis de implementar para que pessoas que não tivessem acesso a controladores mais sofisticados pudessem também elaborar projetos eletrônicos.

O microcontrolador utilizado pelos arduinos originais são membros da série *megaAVR*, especialmente os *ATMega8, ATMega168, ATMega328* e *ATMega1280.*

A grande maioria dos arduinos apresentam um regulador de 5V, um oscilador de cristal de 16 MHz e são pré-programados com um *bootloader* que facilita o carregamento de programas para a memória *flash* de forma a não precisar de um chip programador externo. Há também a presença de componentes complementares para facilitar a programação e incorporação de outros circuitos, além de um chip FTDI FT232 para que seja possível programar o microcontrolador através da porta USB. Um arduino semelhante ao utilizado neste trabalho pode ser visualizado pela Figura 7.

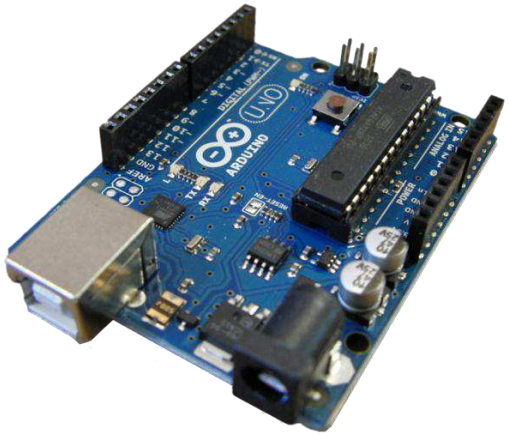


Figura 7 – Imagem ilustrativa do Arduino Uno.

A IDE para programação é uma aplicação multiplataforma escrita em Java e derivada dos projetos *Processing* e *Wiring*. Para que seja implementado um programa funcional é necessário apenas a definição de duas funções: *Setup()* e *Loop().* A primeira é responsável pela inicialização das funcionalidades e o segundo é a função de repetição das funcionalidades de comando.

O arduino apresenta também várias placas complementares, chamadas *Shields,* que estendem as suas funcionalidades como:

- *Shield Ethernet:* utilizada para adicionar funcionalidades de rede ao arduino, associando um endereço IP e MAC de forma que pode-se ler ou escrever dados para o arduino através da intranet ou Internet.
- *Shield* Xbee: Permite que o arduino comunique-se através de uma rede sem fio usando ZigBee.

3 SISTEMA DE MONITORAMENTO DE DADOS

Este trabalho de conclusão de curso fez uso da integração das tecnologias supracitadas para o desenvolvimento de um sistema de monitoramento de dados escritos em *tags* RFID, enviando-os através de uma rede ZigBee para uma aplicação executada em um computador central que comunica-se diretamente com o nó coordenador da rede. Esta aplicação permite que os dados sejam visualizados, manipulados e posteriormente salvos.

Esta seção será dividida em duas partes: *Hardware* e *Software*. A primeira será destinada ao desenvolvimento do circuito necessário para o funcionamento do sistema, enquanto a segunda será destinada ao desenvolvimento da aplicação que será responsável pela interface de monitoramento.

Será ilustrado na Figura 8 um diagrama das arquiteturas de *hardware* e *software* que compõe o sistema de monitoramento de dados de forma à permitir um melhor entendimento dos tópicos seguintes.

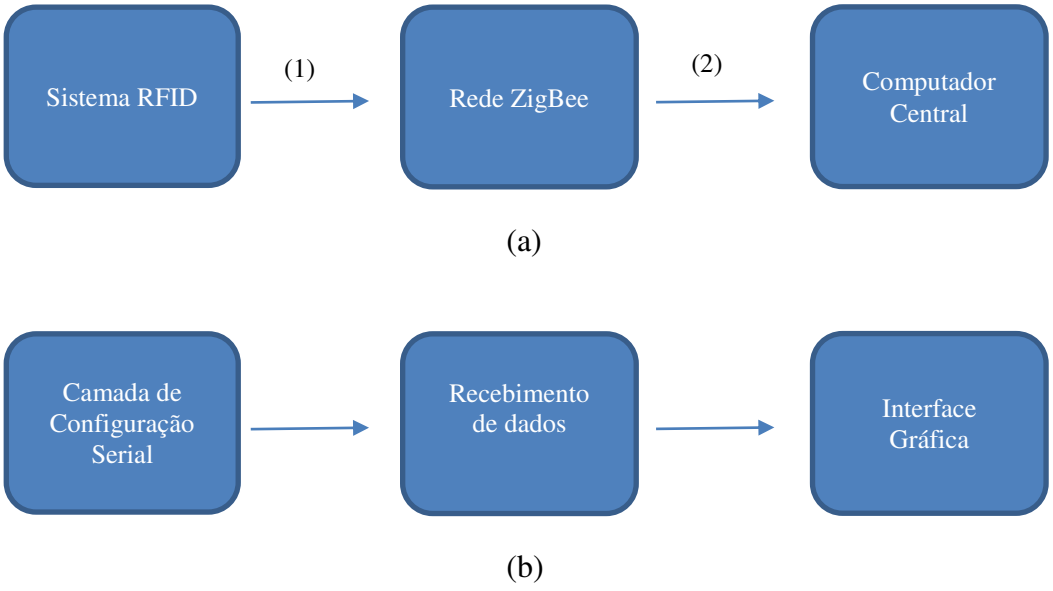


Figura 8 - Diagrama da estrutura de (a) *hardware* e (b) *software* do sistemas de monitoramento de dados.

A Figura 8 (a) é a ilustração do fluxo de dados no sistema de *hardware* onde a transferência de dados (1) e (2) representa uma comunicação serial e a Figura 8 (b) ilustra o sistema de *software.*

3.1 CIRCUITO ELÉTRICO

A estrutura de *hardware* desenvolvida no presente trabalho é responsável por todo o tratamento dos sinais, partindo do recebimento da informação vinda da *tag* RFID até o recebimento via serial pelo computador central. A explanação da estrutura de *hardware* do sistema de monitoramento de dados foi dividida em duas partes para melhor apresentação da estrutura e conceito.

3.1.1 HARDWARE DO SISTEMA RFID

O *hardware* utilizado para a estrutura RFID foi:

- Leitor RFID da *Texas Instruments* modelo RI-STU-MB2A-03;
- Antena *Ferrite Rod* ou *Stick* com dimensões de 20 mm x 140 mm;
- *Tags* do tipo cartão de leitura e escrita com dimensões de 85 mm x 53 mm.

O leitor provê o controle das funções necessárias para interação com *tags* RFID de 134,2 kHz, sendo responsável pelo envio de sinal ao *tag* e pela decodificação e verificação dos dados recebidos [COSTA et al, 2012]. Este leitor possui uma memória de 64 kB de memória EPROM para firmware, 1 kb memória EEPROM para configuração, 32 kB de memória RAM para dados, apresenta uma interface de comunicação RS232 com parâmetros de comunicação configuráveis, uma arquitetura *point-to-point*.

O leitor inicia a comunicação com o envio de um sinal para o *tag*, transmitindo em seguida dados modulados em radiofrequência. A transmissão dar-se-á através de uma interface serial (RS232).

O processo de comunicação via porta serial com o leitor é realizado através do protocolo ASCII. A imagem ilustrativa do leitor RFID é apresentada na Figura 9.

Figura 9 - Imagem ilustrativa do leitor RFID.

A antena utilizada pelo leitor, do tipo *stick*, para captura dos dados é capaz de criar uma zona de leitura de características específicas para atender às necessidades do sistema. Este tipo de antena fornece uma zona mais focada e uma capacidade de discriminar entre tipos de *tags* [*Texas Instruments,* 2002 – Modificado].

Para finalizar, para o *hardware* da estrutura para captura de dados RFID, foram utilizados *Tags* RFID do tipo cartão com uma frequência de 134,2 kHz como apresentada na Figura 10. A configuração utilizada para o sistema RFID foi ARPT.

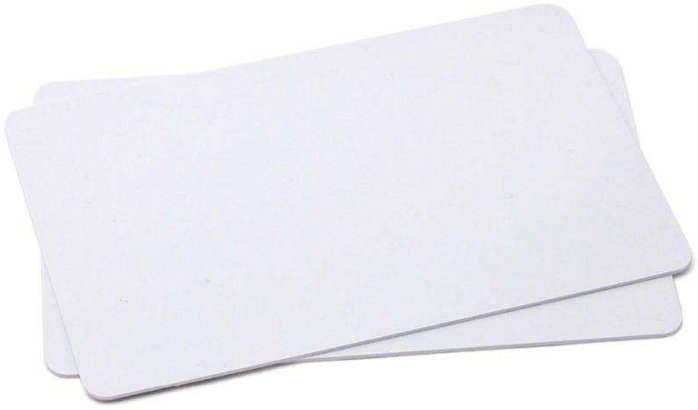


Figura 10 - Imagem ilustrativa dos *Tags* RFID utilizados.

Inicialmente, foi utilizado um circuito de teste para comunicação formado por um arduino com o propósito de simular o comportamento do sistema de dados RFID de forma a testar a rede ZigBee. Este circuito auxiliar será apresentado em seguida.

3.1.2 HARDWARE DO CIRCUITO DE TESTE PARA COMUNICAÇÃO

O circuito de teste para comunicação foi criado para substituir o sistema RFID quando o mesmo não pudesse ser utilizado. Este circuito é formado por um arduino cuja programação simulava o envio de dados pela interface serial para o nó ZigBee e assim os dados eram enviados para o nó coordenador da rede ZigBee para que fosse processado pelo computador central.

A ligação entre o arduino e o módulo ZigBee pode ser observado na Figura 11. O TX e o RX são, respectivamente, o transmissor e receptor de dados da interface serial e é inserida uma resistência de 1 kΩ entre o pino DTR e VCC.

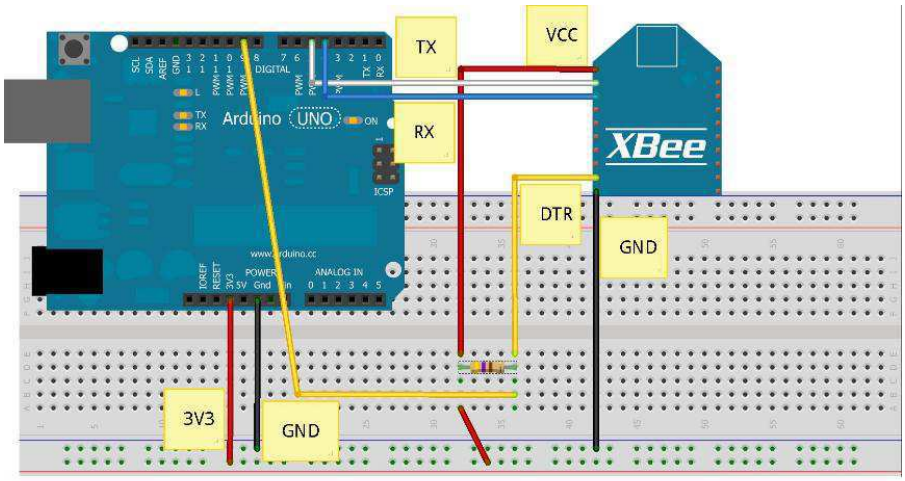


Figura 11 – Representação do esquema de ligação do circuito para teste de comunicação [Fonte: UESPI, 2013].

Podemos observar na Figura 11 que os pinos utilizados para a comunicação serial não são os pinos TX e RX do arduino. Isto foi feito porque não era possível o envio de dados para o nó-sensor da rede ZigBee usando as portas de comunicação serial padrão em virtude da sua utilização para a comunicação com o computador, sendo necessário emular uma interface serial que permitisse o uso de outras portas digitais. Utilizou-se uma biblioteca de programação para arduino, descrita na seção destinada ao *software* do sistema, para sanar tal empecilho.

3.1.3 HARDWARE DO SISTEMA ZIGBEE

O *hardware* utilizado na composição do Sistema ZigBee foram:

- Módulo transceptor XBee e XBee Pro da *MaxStream*;
- Circuito elétrico nó-sensor.

Os módulos XBee e XBee Pro operam na banda de 2,4 GHz e são facilmente integráveis em sistemas que utilizam radiofrequência na transmissão de sinais. Estes módulos atendem o protocolo ZigBee e desta forma podem ser configurados com diferentes topologias.

Outra informação importante relacionada aos módulos é a ausência de configuração externa, ou seja, é permitido a realização das operações necessárias através de comandos AT (comandos enviados ao módulo localmente precedidos da sigla AT que indica atenção aos comandos seguintes os quais definirão as ações do módulo) e de operações específicas de controle, ou realizando as modificações necessárias diretamente sobre o padrão IEEE 802.15.4, o qual permite uma maior interação com o *hardware* do dispositivo, minimizando os custos com outros componentes envolvidos no processo de aquisição de dados e processamento da informação. Em contrapartida, existe a necessidade de um conhecimento mais aprimorado do padrão, visto que as modificações necessárias à implementação de uma aplicação customizada são realizadas através de primitivas, diretamente na pilha do padrão. Assim, destinam-se a interagir com dispositivos por meio de uma interface RS-232, encarregando-se da transmissão e recepção dos dados pretendidos [JUNIOR, 2011].

Diante do exposto acima, iremos detalhar um pouco o que vem a ser os comando AT. Comandos AT são expressões utilizadas para a configuração dos módulos ZigBee. Eles possuem uma estrutura fixa, mostrada a seguir, no qual são sempre precedidos por um comando de três caracteres “+++” que indica para o módulo ZigBee que as linhas seguintes são comandos AT. Para cada comando AT bem sucedido, o módulo ZigBee deverá retornar a mensagem de resposta “OK” ou o valor do endereço, no caso de parametrização de endereços de rede dos módulos, para que seja possível o envio do próximo comando AT. Por fim, deve ser enviado ao módulo ZigBee o comando ATCN para informar que a configuração foi concluída e não será enviado nenhum comando AT em seguida.

Um exemplo de configuração dos módulos ZigBee através de comando AT pode ser vista na Figura 12.

$$+ + +$$

$$ATDL < ENTER > \ - Leitura\ de\ Endereço\ Atual$$

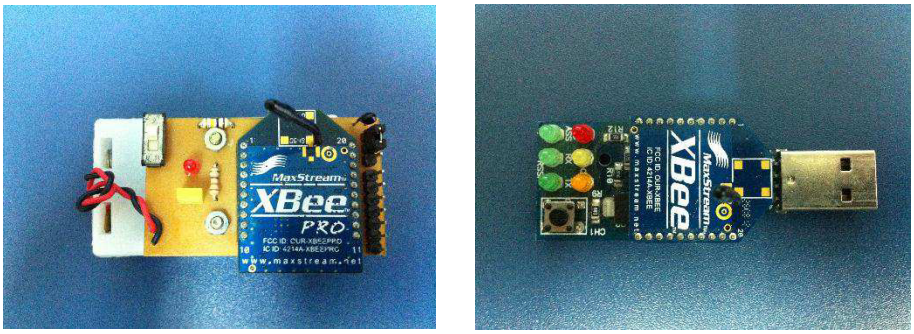
$$ATDL1A0D < ENTER > \ - Modificação\ de\ endereço\ para\ 1A0D$$

$$ATWR < ENTER > - Escrita\ das\ alterações\ para\ a\ memória\ não\ volátil$$

$$ATCN < ENTER > \ - Saída\ do\ comando\ AT$$

Figura 12 - Exemplo de configuração do módulo ZigBee por comandos AT.

Os modelos de transceptores, XBee e XBee Pro, diferem no alcance da transmissão e na potência exigida para a mesma por consequência. Os módulos utilizados podem ser vistos na Figura 13.



(a) (b)

Figura 13 – Imagem ilustrativa do Módulo transceptor: (a) XBee Pro, (b) Xbee.

Os transceptores apresentam uma estrutura interna compostas por um processador com *buffers* de E/S individuais, um transmissor, um receptor e uma chave seletora de radiofrequência, como pode ser percebido na Figura 14. O processador utilizado pelo transceptor é o M9S08GT60 e o transceptor é o MC13193 produzidos pela Freescale.

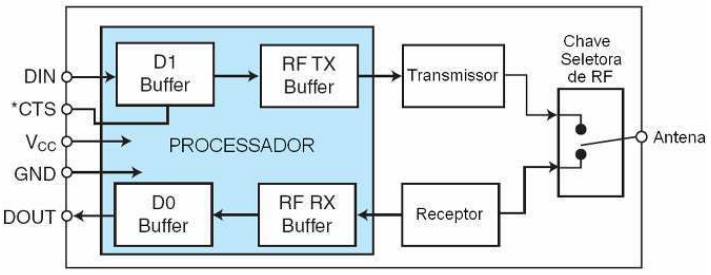


Figura 14 – Representação da estrutura interna dos tranceptores XBee [Fonte: JUNIOR, 2011].

É possível observar na Figura 13 (a) que o transceptor em questão, o nó-sensor, está acomodado em um circuito eletrônico. Este circuito foi utilizado para realizar o condicionamento de modo fácil, eficiente e pequenas dimensões, sendo alimentado apenas por uma pilha de 9 V. O esquemático do circuito em questão pode ser observado na Figura 15.

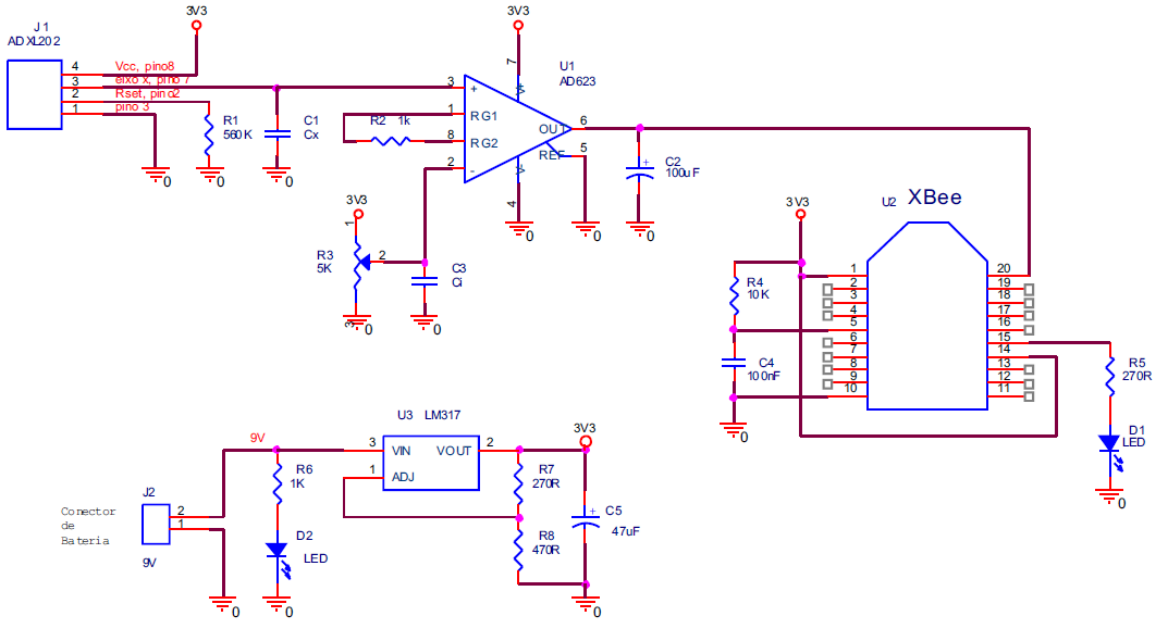


Figura 15 - Diagrama Elétrico do circuito de condicionamento do nó-sensor [Fonte: Junior, 2011].

As configurações dos transceptores é feita, como dito anteriormente, por comandos AT enviados ao módulo. Entretanto, há uma outra possibilidade mais cômoda e rápida de configuração dos módulos fornecida pela *Digi International* nomeada X-CTU.

O X-CTU é uma *software* de programação que possui suporte a comunicação USB com os módulos e uma interface intuitiva de forma a permitir a configuração sem a necessidade de escrita dos comandos AT. Desta forma, torna a programação dos módulos de alto nível visto que a atribuição dos valores dos parâmetros são atribuídos por meio da interface de programação do X-CTU, apresentada na Figura 16.

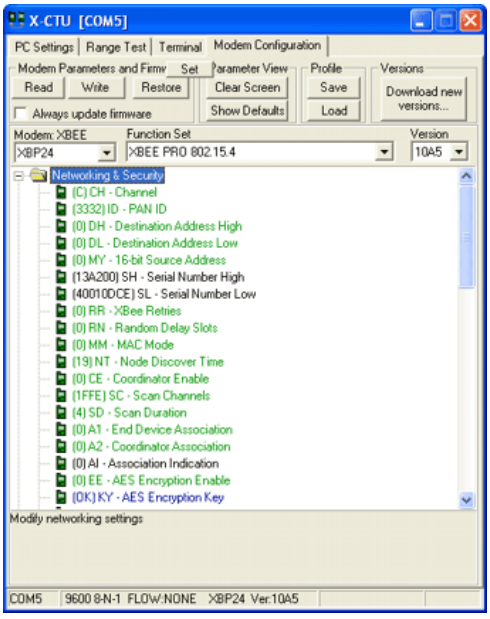


Figura 16 – Representação da tela de configuração dos módulos ZigBee.

Uma perspectiva interessante de utilização do X-CTU, amplamente empregado neste trabalho, é o monitoramento dos dados serial recebidos ou enviados pelo nó coordenador conectado à porta USB e neste trabalho fez-se o uso desta funcionalidade do X-CTU para a checagem do tratamento de dados pelo programa de controle desenvolvido e explicado a posteriori. Pode-se notar na Figura 17 a interface do terminal de visualização de dados do X-CTU.

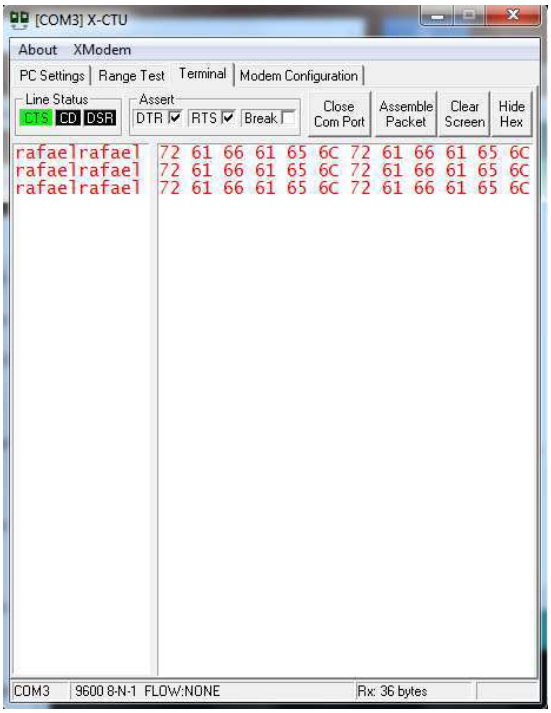


Figura 17 - Interface do terminal de visualização de dados do X-CTU.

3.2 PROGRAMA DE CONTROLE E PROCESSAMENTO DE DADOS

A segunda parte integrante do sistema de monitoramento de dados é a camada de *software* desenvolvida para o controle e processamento dos dados entregues ao nó coordenador através da rede ZigBee. O *software* principal deste sistema é uma aplicação desenvolvida com a linguagem C# que apresenta-se hospedada no computador central e é responsável pelo recebimento, processamento e visualização dos dados do sistema.

A interface da aplicação possuí um terminal de visualização, no qual os dados RFID recebidos pela rede ZigBee são mostrados, e uma zona de configuração da interface serial que é composta por áreas *drop-down* para uma seleção fácil dos parâmetros de comunicação. A interface do programa pode ser vista na Figura 18.

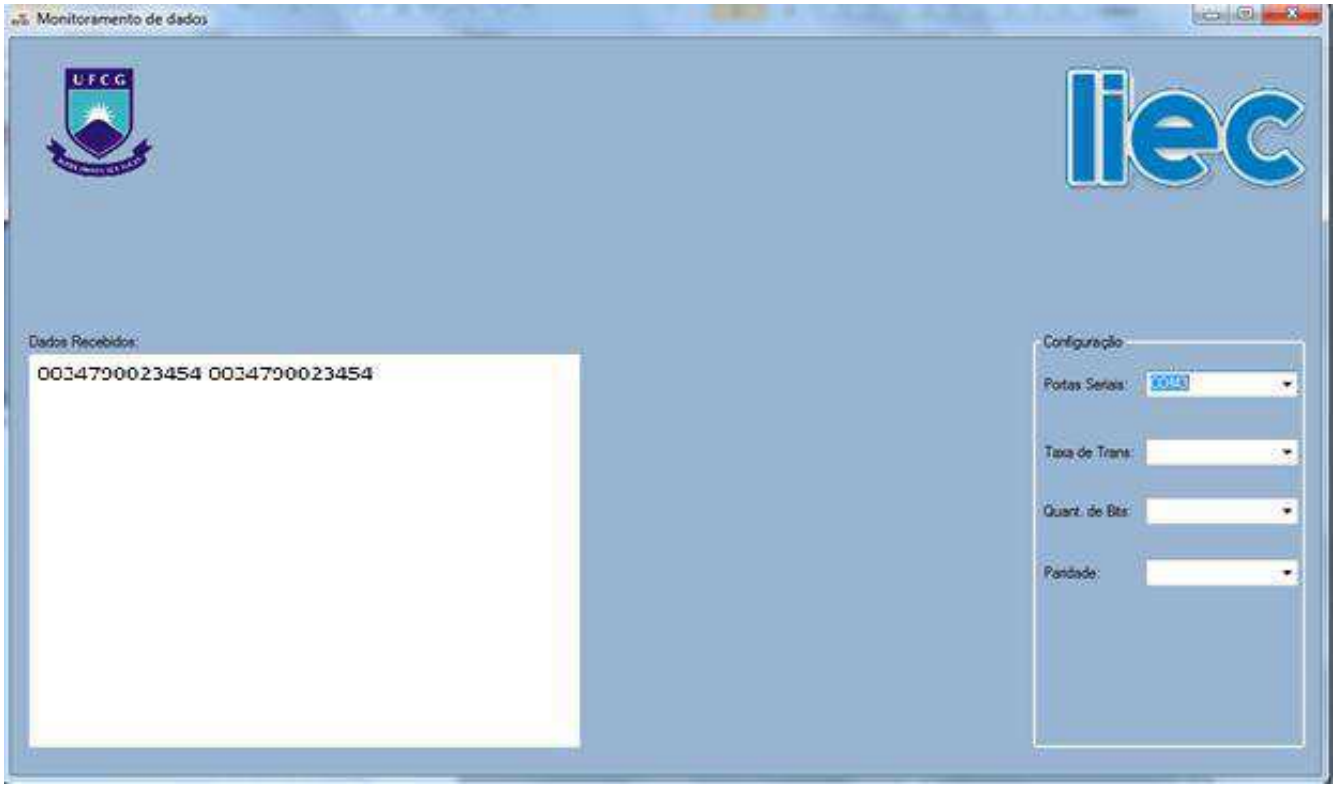


Figura 18 - Interface do programa de controle dos dados.

O código fonte criado no desenvolvimento desta aplicação encontra-se no Apêndice A deste trabalho. Nele é possível checar a estrutura lógica utilizada para tal desenvolvimento.

Para conferir a fidelidade no tratamento das informações recebidas e enviadas no sistema, comparou-se os dados mostrados pelo programa de controle de dados com os dados mostrados pelo terminal do X-CTU.

Outro código desenvolvido neste projeto foi a lógica para o arduino com a finalidade de proporcionar uma simulação do sistema RFID quando o mesmo não pudesse ser utilizado.

O programa permite envio de dados pela porta serial do arduino utilizando as mesmas propriedades de redes que os transceptores XBee estão configurados e então estes dados são recebidos pelo computador central e conferidos pelo terminal do arduino e pelo terminal do X-CTU.

Um detalhe importante no desenvolvimento do programa para o arduino foi a utilização de um biblioteca destinada a comunicação serial. Esta biblioteca, conhecida como *SoftwareSerial,* foi fundamental na implementação no sentido de solucionar o problema de utilizar portas diferentes das TX e RX padrões descrito anteriormente.

Após a finalização do programa do arduino, foi possível o envio/recebimento de dados serial e sua conferência atrás do terminal da IDE do arduino simultaneamente.

Com o sistema RFID, o leitor era configurado de forma a permitir a leitura de dados constantemente e de forma automática, ou seja, não era necessário o envio de comandos por *software* para que o leitor fizesse a aquisição dos dados. Desta forma, a aquisição de dados através do sistema RFID é feito apenas pela passagem do *tag* RFID nas proximidades exigidas da antena do leitor RFID.

Com o *hardware* e *software* concluídos, realizou-se a integração das duas partes para a formação do sistema e então a realização dos testes de funcionamento. A integração e as funcionalidades foram testadas com sucesso tornando o sistema operacional.

4 CONSIDERAÇÕES FINAIS

Neste trabalho apresentou-se o desenvolvimento de um sistema de monitoramento de dados desde a sua estrutura física de engenharia até sua lógica de programação. Tal sistema mostrou ser uma solução eficaz para aplicações que necessitam acesso a vários dados quase simultaneamente, distribuídos geograficamente ou não, como controle de entrada, estoque, leitura de sensores em ambientes com pouco ou moderado ruído, entre outros.

Os teste mostraram que, em curtas e médias distâncias[2], os dados transportados não mostraram alterações entre sua recepção no sistema e sua visualização no programa. Isto mostra que a solução oferece uma boa confiabilidade.

Outro aspecto importante que deve ser levado em consideração foi a preocupação constante em desenvolver uma estrutura física e programas que permitissem uma fácil substituição de itens sem que o sistema ficasse comprometido, sendo assim, é possível substituir o leitor RFID, a estrutura de rede ZigBee, adicionar tecnologias de leitura ou escrita ao sistema sem prejuízo na estrutura, bastando apenas que tais componentes apresentem uma interface de comunicação serial.

Portanto, foi exposto neste trabalho mais uma alternativa eficaz na solução de problemas de monitoramento de dados de baixo custo, fácil adaptação e boa confiabilidade.

Diante do exposto, o sistema de monitoramento de dados desenvolvido neste trabalho de conclusão de curso pode servir de base para muitos sistemas futuros. A análise do comportamento da rede para diferentes topologias fazendo-se uso do *hardware* descrito nas seções anteriores é uma possibilidade de tentar comprovar a fidelidade e robustez do sistema para qualquer estrutura de rede empregada.

Outra possibilidade interessante para trabalhos futuros é desenvolver a funcionalidade de configuração de rede de forma automática através do *software* de monitoramento de dados, permitindo que a identificação das características da interface serial seja feita sem a interação do usuário. Poder-se-ia, também, aumentar a quantidade de leitores, nó-sensores, roteadores ZigBee, computadores centrais e a área de atuação do sistema de forma a testar os limites máximos de *hardware* que poderiam ser agregados ao sistemas de forma a permitir que o mesmo funcione de forma confiável.

[2] Entende-se por médias distancias aproximadamente 100 metros na ausência de repetidores ou nós roteadores.

REFERÊNCIAS

AHSON, S.; ILYAS, M.; RFID Handbook: Applications, Technology, Security and Privacy. *CRC Press*, 2008.

BHUPTANI, M.; MORADPOUR, S.; Field Guide: Deploying Radio Frequency Identification System, *New Jersey*: *Prentice Hall,* 2005.

COSTA, M.; NETO, J.; NETO, J. S.; BARROS, P.; A Influência de Diferentes Superfícies de Fixação no Processo de Leitura e Etiquetas RFID, 2012.

DIPANKAR, S.; PROSENJIT, S; ANAND, D.; RFID For Energy and Utility Industries, *PennWell*, 2009, pp. 1-48, 2009.

FALUDI, R.; Building Wireless Sensor Networks: A practical guide to the ZigBee Mesh Networking Protocol. *O'reilly,* 2010.

FARAHANI, S.; ZigBee Wireless Networks and Transceivers. *Elsevier,* 2008.

GISLASON, D.; ZigBee Wireless Networking. *Elsevier,* 2008.

HALIT, E.; Wireless Sensors and Instruments Networks, Design, and Applications. *Taylor and Francis,* 2005.

JOHNSON M. T.; Redes de Sensores Sem Fio (RSSF): Uso de Comunicação sem fio. Abril 2008.

JUNIOR, L.; Plataforma para Detecção de Incrustação Utilizando o Método do Impacto do Martelo e a Tecnologia ZigBee. Novembro 2011. Pág 58. Dissertação. Universidade Federal de Campina Grande. Campina Grande.

KOUBÂA, A.; ALVES, M.; TOVAR, E.; IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview. *IPP-HURRAY*, 2008.

Texas Instruments, 2002, Series 2000 stick antenna [pdf] Disponível em: http://www.ti-rfid.com [Acessado em 17 de maio de 2013].

UESPI, 2013, Configurando Xbee no modo Sleep. Disponível em: http://www.uespi.br/pesquisa/opala/configurando-o-modo-sleep-no-xbee/ [Acessado em 17 de maio de 2013].

APÊNDICE A: CÓGIDO FONTE DO PROGRAMA DE PROCESSAMENTO DE DADOS.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.IO.Ports;

namespace Serial
{
    public partial class Form1 : Form
    {

        static bool _continue = true;
        static SerialPort _serialPort = new SerialPort();

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            string name;
            string command;
            string[] portas;
            StringComparer stringComparer = StringComparer.OrdinalIgnoreCase;
            Thread t = new Thread(Read);

            portas = SerialPort.GetPortNames();

            foreach (string porta in portas)
            {
                PortasLista.Items.Add(porta);
            }

            // Allow the user to set the appropriate properties.
            _serialPort.PortName = "COM3";
            _serialPort.BaudRate = 9600;
            _serialPort.Parity = 0;
            _serialPort.DataBits = 8;

            // Set the read/write timeouts
            _serialPort.WriteTimeout = 500;

            _serialPort.Open();
            t.Start();

            while (_continue)
            {
                name = Console.ReadLine();
                if(stringComparer.Equals("sair", name)) _continue = false;

                if (!stringComparer.Equals("+++", name))
```

```
                {
                    command = name + "\r\n";
                    _serialPort.Write(command);
                }

                _serialPort.Write(name);
            }

            t.Join();
            _serialPort.Close();
        }

        public void Read()
        {
            Form1 objeto = new Form1();
            byte[] buffer = new byte[1];
            int value;

            while (_continue)
            {
                try
                {
                    value = _serialPort.Read(buffer, 0, 1);
                    string va = ASCIIEncoding.ASCII.GetString(buffer);

                        this.textBox1.Invoke(new Action(() =>
                        {
                            this.textBox1.AppendText(va + " ");
                        }));

                }

                catch (TimeoutException) { }
            }
        }

    }

}
```

APÊNDICE B: CÓDIGO FONTE DA LÓGICA DO ARDUINO

```
#include <SoftwareSerial.h>

SoftwareSerial xbee(2, 3); // RX, TX

void setup()  {


   // set the data rate for the SoftwareSerial port
   xbee.begin( 9600);
}

void loop()  {
  // send character via XBee to other XBee connected to Mac
  // via USB cable
  xbee.print( "Rafael\r\n" );

  delay( 1000 );
}
```