CENTRO UNIVERSITÁRIO DE BRASÍLIA -UniCEUB

CURSO DE ENGENHARIA DE COMPUTAÇÃO

FELIPE SOUTO PEREIRA

SOLUÇÃO DE ETIQUETAS DE PREÇO DIGITAIS PARA COMÉRCIO CONTROLADAS REMOTAMENTE

Orientador: Professora MC Maria Marony Sousa Farias

Brasília

Novembro, 2011

FELIPE SOUTO PEREIRA

SOLUÇÃO DE ETIQUETAS DE PREÇO DIGITAIS PARA COMÉRCIO CONTROLADAS REMOTAMENTE

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.

Orientador: Professora MC Maria Marony Sousa Farias

Brasília

Novembro, 2011

FELIPE SOUTO PEREIRA

SOLUÇÃO DE ETIQUETAS DE PREÇO DIGITAIS PARA COMÉRCIO CONTROLADAS REMOTAMENTE

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.
Orientador: Professora MC Maria Marony Sousa Farias

Este Trabalho foi julgado adequado para a obtenção do Título de Engenheiro de Computação, e aprovado em sua forma final pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS.

_____
Prof. Abiezer Amarilia Fernandez
Coordenador do Curso

Banca Examinadora:

_____
Prof. nome, titulação.
Orientador

_____
Prof. nome, titulação.
Instituição

_____
Prof. nome, titulação.
Instituição

_____
Prof. nome, titulação.
Instituição

Dedico este trabalho à minha família e, meus amigos que sempre apoiaram e incentivaram a realização das minhas maluquices.

AGRADECIMENTOS

Em primeiro lugar agradeço a Deus.

Um agradecimento especial à minha família. À minha mãe Laudise e à minha tia Lêda Maria pelos conselhos para a realização do trabalho.

À meu irmão Guilherme pela ajuda no desenvolvimento do protótipo.

Agradeço ao meu amigo Rafael Coelho pela sugestão do tema do trabalho e incentivo para a realização.

À ajuda dos meus amigos: Maria Luiza, Luciana Melo, Márcio Mello, José Carlos Cruz e Aline Ataíde.

Aos meus colegas de trabalho por compreender a importância deste projeto e pelo apoio nos momentos em que precisei.

À minha querida Alessandra Correia pela força, incentivo e compreensão nos momentos de dificuldade.

À professora Maria Marony pelo seu ensinamento.

E àqueles que, de alguma maneira, contribuíram para a realização deste trabalho.

"Conta certa lenda, que estavam duas crianças patinando num lago congelado.
Era uma tarde nublada e fria, e as crianças brincavam despreocupadas.
De repente, o gelo quebrou e uma delas caiu, ficando presa na fenda que se formou.
A outra, vendo seu amiguinho preso, e se congelando, tirou um dos patins e começou a golpear o gelo com todas as suas forças, conseguindo por fim, quebrá-lo e libertar o amigo.
Quando os bombeiros chegaram e viram o que havia acontecido, perguntaram ao menino:
- Como você conseguiu fazer isso? É impossível que tenha conseguido quebrar o gelo, sendo tão pequeno e com mãos tão frágeis!
Nesse instante, um ancião que passava pelo local, comentou:
- Eu sei como ele conseguiu.
Todos perguntaram:
- Pode nos dizer como?
- É simples: - respondeu o velho.
- Não havia ninguém ao seu redor para lhe dizer que não seria capaz."

Albert Einstein

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ANATEL – Agência Nacional de Telecomunicações

API – Application Programming Interface

ASK – Amplitude Shift Keying

AWG – American Wire Gauge

bps – Bits Per Second

CCP – Compare, Capture e PWM

CI – Circuito Integrado

cm – Centí ou 10^{-2} metro, unidade de medida

CRC – Cyclic Redundancy Check

CTS – Clear to Send

DB9 – Data Bus 9

DPT – Digital Price Tag

EEPROM – Electrically-Erasable Programmable Read-Only Memory

EIA – Eletronics Industries Association

ESL – Eletronic Shelf Label

GND – Ground, terra, aterramento

GUI – Graphical User Interface

HEX – Hexadecimal

Hz – Hertz

I/O – Input/Output

ICSP – In Circuit Serial Programming

IDE – Integrated Development Environment

k – Kilo ou 10^{3}

kHz – Kilo ou 10^{3} hertz, unidade de frequência

LAM – Look At Me

LASER – Light Amplification by Stimulated Emission of Radiation

LCD – Liquid Crystal Display

LED – Light-Emitting Diode

mA – Milí ou 10^{-3} Ampére, unidade de medida de intensidade de corrente elétrica

MCLR – Master Clear

MCU – Microcontrolador

MHz – Mega-Hertz ou 10^6 Hertz, unidade de frequência

mm – Milímetros ou 10^{-3} metros, unidade de medida

m/s – Metros por segundo, unidade de velocidade

OOK – On-Off Keying

OS – Operating System

PCB – Printed Circuit Board

PDIP – Plastic Dual In Line Package

pF – Pico Farads, unidade de capacitância

PWM – Pulse Width Modulation

RAM – Random Access Memory

RF – Radio Frequency

RISC – Reduced Instruction Set Computer

RS-232 – Recommended Standard 232

RTS – Ready To Send

RXD – Receive Data

R$ – Real

TTL – Transistor-Transistor Logic

TXD – Transmit Data

UART – Universal Asynchronous Receive and Transmit

UCP – Unidade Central de Processamento

ULA – Unidade Lógica e Aritmética

USART – Universal Syncronous Asyncronous Receiver Transmitter

USB – Universal Serial Bus

V – Volt, unidade de tensão elétrica

VDC – Voltage Direct Current

WI-FI – Wireless Fidelity

µF – Micro ou 10^{-6} Farad**s,** unidade de capacitância

µS – Micro ou 10^{-6} Segundos, unidade de tempo

Ω - Ohm, unidade de resitência

RESUMO

Neste trabalho é apresentado uma solução de etiquetas de preço digitais para comércio controladas remotamente. Seu uso possibilita reduzir o desperdício de papel utilizado nas etiquetas convencionais. No projeto, o controle do conteúdo das etiquetas é feito através de um programa desenvolvido para computador, que permite maior agilidade na manutenção dos preços dos produtos. Por intervenção do usuário, os dados referentes aos produtos são enviados a um dispositivo que faz a transmissão das informações por radiofrequência. Essa transmissão utiliza um protocolo de comunicação próprio, com codificação dos dados e algoritmos de detecção de erros. Um dispositivo mantém os produtos listados no computador sincronizados com as etiquetas eletrônicas, utilizando o protocolo. As etiquetas eletrônicas captam os dados, realizam a validação do protocolo e os exibem em um display de cristal líquido. Possuem ainda a capacidade de armazenar os dados recebidos em memória não-volátil, sendo úteis nos casos de desligamento ou esgotamento da fonte de energia. Por serem dispositivos compactos, as etiquetas podem ser fixadas em gôndolas, além de possuírem mobilidade para uso em quaisquer áreas no interior do estabelecimento.

Palavras Chave: Automação, etiquetas digitais, radiofrequência, microcontroladores.

ABSTRACT

This project presents a solution for digital price tags used in commerce remotely controlled. Its use allows to reduce paper waste used in conventional tags. In this project, the control of content of tags is done by a computer program developed, allowing greater agility in the maintenance of prices. By user intervention, the data of those products are sent to the device that transmit the information by radiofrequency. This transmission uses a own communication protocol with data encoding and algorithms of error detection. A device keeps the products listed on the computer synchronized with the electronic price tags, using the protocol. The electronic price tags receive the data, perform the protocol validation and displays it in a liquid crystal display. They also have the ability to store the received data in nonvolatile memory and is useful in cases of shutdown or depletion of the energy source. By being compact devices, the tags can be affixed to shelves, besides they have mobility for use in any area within the establishment.

Keywords: Automation, digital labels, radiofrequency, microcontrollers.

CAPÍTULO 1 – INTRODUÇÃO

1.1 – Apresentação do Problema

O comércio atualmente está lotado de etiquetas de preços. Pedaços de papel que indicam o que está à venda e o seu valor. De certa forma, há um grande desperdício de papel e dinheiro na utilização desta forma de indicação de preços.

Segundo Saxena (ALTIERRE DIGITAL RETAIL, 2008), há mais de 90 milhões de folhas de papel sendo desperdiçadas em forma de etiquetas de preço.

Além disso, há um grande esforço na manutenção destas etiquetas, necessitando mão de obra fora do horário de funcionamento do estabelecimento para imprimir, recortar e distribuir os novos preços.

Neste projeto é proposta a automatização deste serviço, substituindo as etiquetas de papel por etiquetas digitais, que podem ser controladas remotamente através de tecnologia *wireless*.

Com a utilização de etiquetas eletrônicas, a atualização dos preços pode ser feita de imediato, na sala de administração do estabelecimento. Essas alterações também podem ser sincronizadas diretamente dos registros do estabelecimento, eliminando a possibilidade de superfaturamento e de diferenças entre os preços anunciados e o caixa.

1.2 – Objetivos

Este projeto tem como o objetivo geral desenvolver uma solução de etiquetas de preço eletrônicas controladas remotamente.

As tarefas a serem executadas para a realização do objetivo do projeto são:

1. Desenvolver um circuito que receba mensagens através de comunicação sem fios e exiba a mensagem recebida em um display de cristal líquido que neste projeto será chamado de *Digital Price Tag* ou DPT;
2. Elaborar um circuito intermediário entre o usuário e as etiquetas, com finalidade de encaminhar as mensagens via *wireless*, denominado no projeto de DPT *Transmitter*; e

3. Programar uma interface gráfica para o gerenciamento da solução que neste projeto será chamado de DPT *Manager*.

1.3 – Justificativa e Importância do Trabalho

São muitos os produtos que um estabelecimento pode oferecer a seus clientes. Normalmente são centenas a milhares de itens diferentes disponibilizados para a venda.

Para que estes produtos sejam dispostos para a venda, é necessário que possuam uma etiqueta contendo a descrição de cada um deles como marca, nome e preço. Estes produtos podem ser etiquetados individualmente ou estar reunidos em uma prateleira, chamada de gôndola. Neste caso a gôndola é etiquetada.

Normalmente estas etiquetas requerem uma manutenção feita de forma manual. Para uma simples atualização dos preços, é necessário imprimir, recortar e substituir as etiquetas, o que gera desgaste para os funcionários, ainda mais quando a substituição é necessária em vários produtos. Este processo manual está suscetível a erros humanos. Outro fator negativo é o desperdício de papel. Depois de substituídas, as etiquetas que não são reutilizáveis, tornam-se lixo.

Sobre o uso de etiquetas de papel, de acordo com o Setor Varejista de Alimentos, em média:

- 6% dos preços são exibidos incorretamente nas gôndolas;
- 2% das gôndolas não dispõem de etiquetas;
- 15% de compras não são realizadas devido a falta de etiquetas;
- 2 minutos é a média para alterar uma etiqueta de papel;
- 20 reclamações por semana são feitas relacionadas a preços incorretos;
- 2 auditorias nos preços são feitas por ano.

(FONTE: PRICER. 2011)

A adoção de etiquetas eletrônicas propostas neste trabalho pode ser uma solução para a solução destes problemas, pois as etiquetas só necessitariam ser substituídas nos casos de falta de bateria, o que poderia levar anos para ocorrer.

Além disso, como o gerenciamento é feito remotamente, as alterações podem ser feitas a qualquer momento através de um programa de computador, permitindo que várias etiquetas sejam atualizadas em um curto período de tempo.

1.4 – Escopo do Trabalho

O escopo do projeto consiste em elaborar um protótipo de um sistema de controle de etiquetas eletrônicas, composto pela própria etiqueta eletrônica, o dispositivo que realiza o sincronismo das etiquetas e uma aplicação que serve de interface entre o usuário e as etiquetas eletrônicas.

As etiquetas foram montadas utilizando microcontroladores da empresa Microchip Technology Inc®, família PIC e modelo 16F628A, ligado a receptores de radiofrequência de 433.92MHz e a *displays* de cristal líquido. Para o desenvolvimento do dispositivo de sincronismo o mesmo microcontrolador foi escolhido. Neste microcontrolador foram ligados o transmissor de radiofrequência, utilizando a mesma frequência das etiquetas (433.92MHz), e um circuito integrado MAX232, que permite a comunicação no padrão RS-232 (*Eletronics Industries Association* - EIA, 1969) com o computador. A interface do usuário foi desenvolvida utilizando a linguagem de programação Java, que é executada na plataforma *Windows* e é a responsável pelo gerenciamento e controle das etiquetas.

O projeto foi desenvolvido para simulação de produtos em gôndolas. A comunicação com as etiquetas inseridas nas gôndolas é feita de forma unidirecional. Com isso, não faz parte do seu escopo:

- A obtenção de informações das etiquetas;
- O funcionamento do sistema desenvolvido em outras plataformas que não *Windows*; e
- A segurança na transmissão dos dados.

1.5 – Resultados Esperados

Como resultado do desenvolvimento deste projeto, são esperados:

- Dois circuitos simulando etiquetas eletrônicas;
- Um circuito de sincronismo para atualização de conteúdo das etiquetas; e
- Uma interface de controle do sistema para a interação do usuário.

1.6 – Estrutura do Trabalho

Este trabalho é composto por 6 capítulos, iniciando com a introdução, que faz a apresentação do problema a ser tratado, os objetivos, as justificativas e a importância do projeto, o seu escopo e o que se espera como resultado do seu desenvolvimento, além da estrutura do trabalho.

No capítulo 2 são apresentadas as fundamentações sobre preços, a lei de precificação e aborda também o atual mercado das etiquetas eletrônicas.

No capítulo 3 são tratados o referencial teórico dos principais componentes do sistema, fazendo menção às teorias e aos conceitos dos itens que compõem este projeto.

No capítulo 4 são apresentados os métodos de implementação da solução, com as especificações dos componentes e descrição do funcionamento de cada um deles.

O capítulo 5 trata dos procedimentos feitos após a implementação do projeto, como os testes do protótipo e simulação do seu funcionamento, bem como a descrição dos problemas encontrados e a forma como foram resolvidos.

O capítulo 6 apresenta a conclusão. Este capítulo marca o encerramento da monografia, com a avaliação global do projeto. Nele ainda foram apresentadas propostas para trabalhos futuros, bem como recomendações e sugestões para o aprimoramento da solução ora apresentada.

CAPÍTULO 2 – PREÇO, LEI DE PRECIFICAÇÃO E O MERCADO DAS ETIQUETAS ELETRÔNICAS

Preço é uma expressão monetária do valor de um bem ou serviço transacionado no mercado. Uma das principais questões abordadas pela economia é o estudo do processo pelo qual são estabelecidos os preços dos bens no mercado.

Os preços não resultam de um processo desorganizado mas sim de milhões de decisões de empresas e consumidores que estão por trás das curvas de oferta e demanda. Para Mankiw (2005, p. 123), "os preços desempenham a função crucial de equilibrar oferta e demanda e, com isso, coordenar a atividade econômica. Quando os formadores de políticas fixam preços por decreto, obscurecem os sinais que normalmente conduzem a alocação dos recursos da sociedade" isto por que, de acordo com a teoria econômica, o valor de determinado bem é resultado direto de sua procura por parte dos consumidores e a oferta por parte dos produtores.

Segundo Mankiw (2005), nas situações em que o nível da quantidade ofertada é igual a quantidade demandada, temos o chamado preço de equilíbrio.

A Teoria da Procura diz que quanto maior o preço do bem menor será a quantidade procurada, contrária à Teoria da Oferta que afirma que quanto maior o preço do bem, maior será a quantidade oferecida.

No caso do preço estar acima do preço de equilíbrio, a quantidade que os produtores oferecem é necessariamente superior à quantidade que os consumidores procuram. Verifica-se um excesso de oferta. Assim sendo, os produtores são levados a baixarem os preços de forma a conseguirem vender os seus produtos. Inversamente, se o preço estiver abaixo do seu preço de equilíbrio, a quantidade procurada será superior à quantidade oferecida. Verifica-se, então, um excesso de procura. Neste caso, os produtores têm incentivos para aumentar os preços de forma a satisfazerem toda a procura.

2.1 – Lei Federal n° 10.962/2004

Em 11 de outubro de 2004, o presidente Luiz Inácio Lula da Silva sancionou a Lei de Precificação, que dispõe sobre a oferta e as formas de afixação de preços de produtos e serviços para o consumidor.

O artigo 2°, da Lei Federal n° 10.962/2004, refere-se as formas de afixação dos preços em vendas a varejo para o consumidor. São elas:

> I – no comércio em geral, por meio de etiquetas ou similares afixados diretamente nos bens expostos à venda, e em vitrines, mediante divulgação do preço à vista em caracteres legíveis;
> II – em auto-serviços, supermercados, hipermercados, mercearias ou estabelecimentos comerciais onde o consumidor tenha acesso direto ao produto, sem intervenção do comerciante, mediante a impressão ou afixação do preço do produto na embalagem, ou a afixação de código referencial, ou ainda, com a afixação de código de barras.
> Parágrafo único. Nos casos de utilização de código referencial ou de barras, o comerciante deverá expor, de forma clara e legível, junto aos itens expostos, informação relativa ao preço à vista do produto, suas características e código. (FONTE: BRASIL, Lei Federal n° 10.962. 2004)

A lei também contempla os casos em que há divergências no preço do mesmo produto entre os sistemas de informação utilizados pelo estabelecimento. Nesses casos o consumidor deve pagar o menor preço.

Um dos objetivos deste trabalho é evitar que ocorram divergências na afixação dos preços no estabelecimento, unificando o sistema das etiquetas com o sistema dos caixas, assegurando que a mesma base de dados alimente todos os sistemas.

2.2 – O Mercado das Etiquetas Eletrônicas

Várias empresas no mercado disponibilizam soluções de etiquetagem eletrônica. São diversas soluções, empregando várias tecnologias para resolver o mesmo problema. A maioria dessas soluções implementam somente a automatização dos valores dos produtos, utilizando ainda etiquetas de papel com informações da marca e nome do produto anunciado.

As etiquetas fornecidas pela empresa RR Etiquetas (página na web: *www.rretiquetas.com.br*), são do tipo comentado. O preço do produto é exibido em um visor e logo abaixo é afixado uma etiqueta de papel com o nome, marca e código de barras do produto como ilustrado na Figura 2.1.

Figura 2.1 - Etiqueta eletrônica de gôndola 49mm

(FONTE: RR Etiquetas. 2011)

Além da figura ilustrada, a empresa também disponibiliza as etiquetas em outros tamanhos e modelos. O modelo de 69mm permite uma legibilidade 50% maior, comparado ao modelo de 49mm conforme ilustra a Figura 2.2. Há também modelos específicos para a área de vendas de frutas, verduras e legumes como ilustra a Figura 2.3.

Figura 2.2 - Etiqueta eletrônica de gôndola de 69mm

(FONTE: RR Etiquetas. 2011)

Figura 2.3 - Etiquetas eletrônicas para frutas, legumes e verduras

(FONTE: RR Etiquetas. 2011)

A Pricer Inc.® é outra empresa que fornece a solução de etiquetas eletrônicas. Denominada pela empresa como *Eletronic Shelf Label* (ESL), "esta tecnologia é capaz de comunicar, diretamente nas prateleiras, o preço, dados sobre promoções, informações técnicas sobre o produto e até informações gerenciais como por exemplo estoque, *facing* e validade de promoções." (FONTE: www.seal.com.br/produtos/etiquetas.html).

O funcionamento do ESL é baseado em raios infravermelhos de alta frequência, livre de interferência com grande capacidade de transferir dados, dispensando o uso de cabos. A comunicação com as etiquetas é bidirecional, para assegurar que elas receberam as atualizações dos preços e de dados em geral, além de poder informar ao administrador dados sobre a bateria, a não atualização do preço ou até mesmo o furto da etiqueta.

A Pricer Inc.® possui ESL's de diversos tipos, desde modelos simples, que só exibem o preço e utilizam etiquetas de papel para outras informações, até modelos mais sofisticados, com *displays* gráficos, onde é possível exibir até mesmo a logomarca do produto anunciado.

A Figura 2.4 ilustra os diversos modelos disponibilizados pela empresa e a Figura 2.5 ilustra as informações que podem ser adicionadas às etiquetas.

Figura 2.4 - Modelos de ESL's

(FONTE: SEAL Tecnologia. 2011)

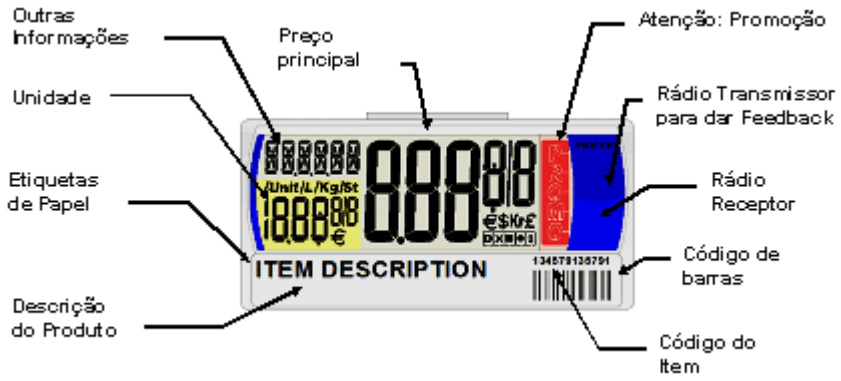

Figura 2.5 - *Eletronic Shelf Label*

(FONTE: SEAL Tecnologia. 2011)

CAPÍTULO 3 – REFERENCIAL TEÓRICO

Neste capítulo são descritos os principais itens que compõem esse projeto, são eles: os microcontroladores, os meios de transmissão e o padrão RS-232, os visores de cristal líquido e as linguagens de programação.

3.1 – Microcontroladores

Microcontroladores são chips circuitos integrados contendo em seu interior um processador, periféricos de entrada e saída e memória. Podem ser programados para funções específicas de controle. Através da programação dos microcontroladores, pode-se controlar suas saídas tendo como referência as entradas ou um programa interno. A diferenciação entre os diversos microcontroladores basicamente são: quantidade de memória interna, velocidade de processamento, pinos de entrada/saída, arquitetura (Trevisan, 2001).

O microcontrolador escolhido para o desenvolvimento deste projeto foi o PIC16F628A por ser de prévio conhecimento do autor e possuir um amplo suporte disponível na comunidade *Web*.

3.1.1– Microchip PIC16F628A

A Microchip Technology Inc.® é uma empresa americana de semicondutores e fabricante dos microcontroladores da família PIC. O PIC está disponível em uma ampla gama de modelos para melhor adaptar-se às exigências de projetos específicos, diferenciando-se pelo número de linhas de I/O e pelo conteúdo do dispositivo. Inicia-se com modelo pequeno identificado pela sigla PIC12Cxx dotado de 8 pinos, até chegar a modelos maiores com sigla PIC17Cxx dotados de 40 pinos, conforme Trevisan (2001). Uma descrição detalhada da tipologia do PIC é disponibilizada no site da fabricante Microchip® (www.microchip.com). No site também é possível encontrar informações técnicas e exemplos de aplicações.

O modelo utilizado no desenvolvimento deste projeto foi o PIC16F628A. Este modelo foi escolhido por já possuir uma arquitetura completa, sem a necessidade de utilizar componentes externos como um cristal oscilador. Segue suas especificações:

- Arquitetura de Harvard. Nessa arquitetura a Unidade Central de Processamento (UCP) é interligada à memória de dados e à memória de programa por barramento específico;
- 18 pinos sendo que 16 de I/O, configuráveis;
- Suporta osciladores de até 20MHz;
- Atua na faixa de 2V até 5.5V;
- 2 osciladores internos (4MHz e 48kHz);
- Tamanho do *bus* de programa de 14 *bits*;
- Tamanho do *bus* de dados de 8 *bits*;
- 35 instruções;
- 10 interrupções disponíveis, entre elas estão: *timers*, externa, mudança de estado, EEPROM, USART, CCP e comparador;
- Memória de programação FLASH com 2.048 palavras;
- Permite a gravação do programa diversas vezes no mesmo chip;
- Memória EEPROM, não-volátil, interna com 128 *bytes*;
- Módulo CCP (*Capture*, *Compare* e PWM);
- Um canal de comunicação serial USART;
- 224 posições de memória de dados com capacidade de armazenamento de 8 *bits* em cada posição;
- Tecnologia RISC.

De acordo com Zanco (2005, p.34), a Figura 3.1 “ilustra o diagrama em blocos da arquitetura interna do PIC 16F628A em que se destacam as memórias de programa, RAM e EEPROM, a ULA, contador de programa e os periféricos presentes no MCU.”

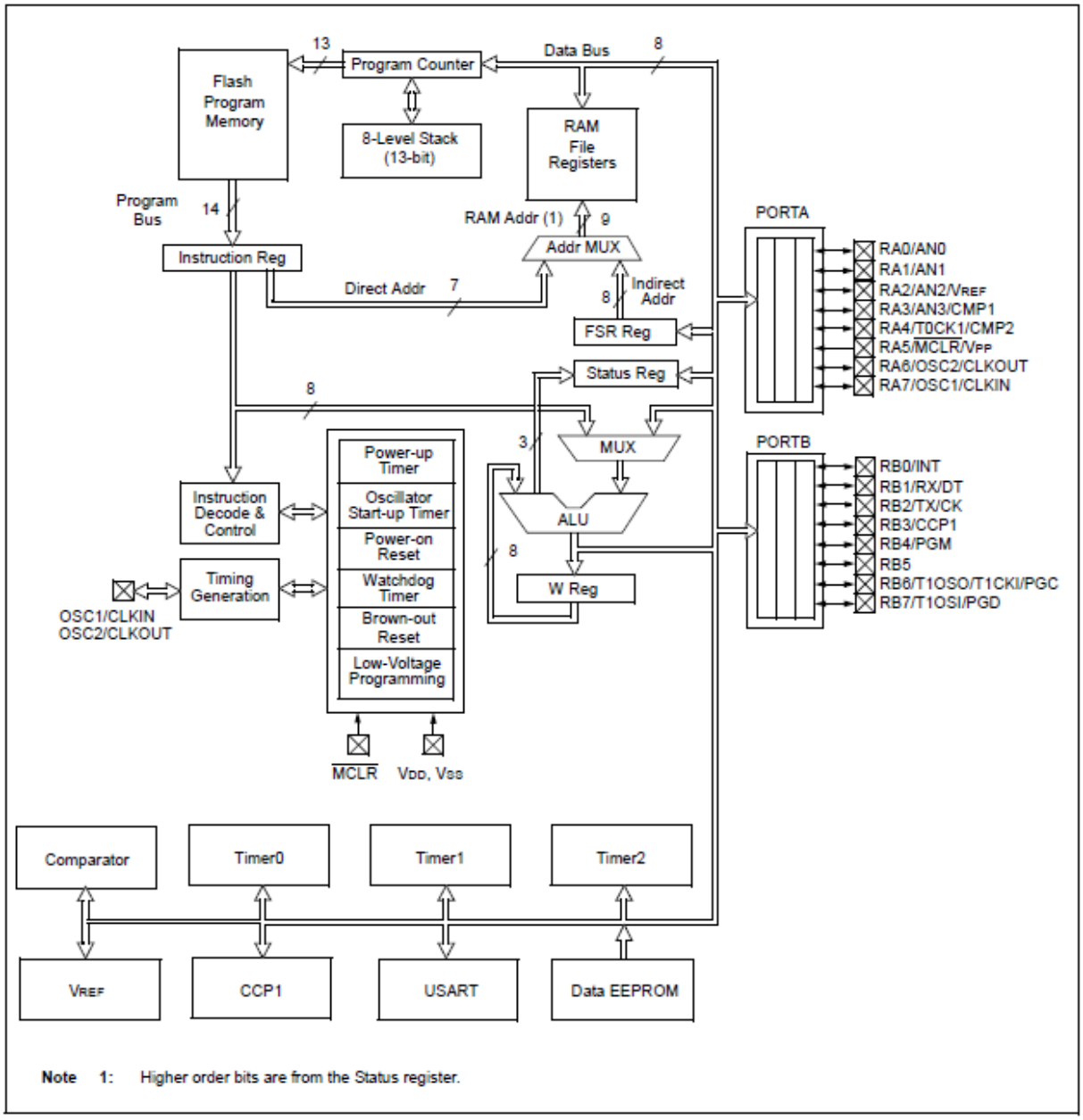

Figura 3.1 - Diagrama de Blocos do PIC16F628A.

(FONTE: Microchip Technology®, 2005, p. 10)

No diagrama também estão destacados 2 grupos, PORTA e PORTB, que se referem aos pinos configuráveis de entrada e saída do microcontrolador. Cada um desses grupos possuem 8 pinos. Na PORTA estão associados os pinos denominados RA0, RA1, RA2, RA3, RA4, RA5, RA6 e RA7. Os pinos referentes a PORTB são denominados RB0, RB1, RB2, RB3, RB4, RB5, RB6 e RB7.

3.1.2 – Pinos do PIC16F628A

A Figura 3.2 ilustra como estão dispostos os pinos das portas PORTA e PORTB no chip de encapsulamento PDIP e o **Erro! Fonte de referência não encontrada.** o ilustra as funções e descrições de cada um desses pinos.

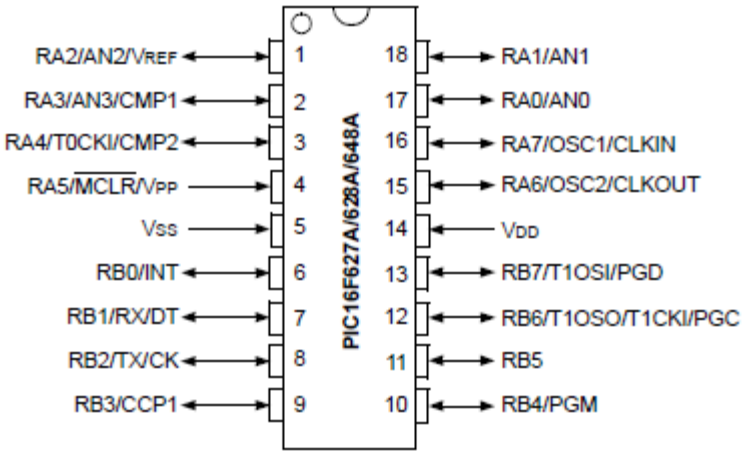

Figura 3.2 - PIC16F628A

(FONTE: Microchip Technology®, 2005, p. 4)

Quadro 3.1 - Funções e descrições dos pinos do PIC16F628A

Pino	Função	Tipo	Descrição
1	RA2/AN2/Vref	Entrada/saída	PORTA bit 2 / Entrada do comparador analógico / Saída da tensão de referência
2	RA3/AN3/CMP1	Entrada/saída	PORTA bit 3 / Entrada do comparador analógico / Saída comparador 1
3	RA4/T0CKI/CMP2	Entrada/saída	PORTA bit 4 / Entrada de clock externo do timer 0 / Saída comparador 2. *Esse pino possui saída com **dreno aberto***
4	RA5/MCLR/VPP	Entrada	PORTA bit 5 / Reset CPU / Tensão de programação
5	VSS	Alimentação	Terra
6	RB0/INT	Entrada/saída	PORTB bit 0/ Entrada de interrupção externa
7	RB1/RX/DT	Entrada/saída	PORTB bit 1/ Recepção USART (modo assíncrono) / Dados (modo síncrono)
8	RB2/TX/CK	Entrada/saída	PORTB bit 2/ Transmissão USART (modo assíncrono) / Clock (modo síncrono)
9	RB3/CCP1	Entrada/saída	PORTB bit 3 / Entrada ou saída do módulo CCP
10	RB4/PGM	Entrada/saída	PORTB bit 4 / Entrada de programação **LVP***
11	RB5	Entrada/saída	PORTB bit 5
12	RB6/T1OSO/T1CKI/ PGC	Entrada/saída	PORTB bit 6 / Entrada do oscilador do TMR1 / Entrada de clock do TMR1 / Clock na programação **ICSP***
13	RB7/T1OSI/ PGD	Entrada/saída	PORTB bit 7 / Entrada do oscilador do TMR1 / Dados na programação ICSP
14	VDD	Alimentação	Alimentação positiva (3V a 5V)
15	RA6/OSC2/CLKOUT	Entrada/saída	PORTA bit 6 / Entrada para cristal oscilador / Saída de clock
16	RA7/OSC1/CLKIN	Entrada/saída	PORTA bit 7 / Entrada para cristal oscilador / Entrada de clock externo
17	RA0/AN0	Entrada/saída	PORTA bit 0/ Entrada do comparador analógico
18	RA1/AN1	Entrada/saída	PORTA bit 1/ Entrada do comparador analógico
***LVP** - Baixa voltagem de programação. ***ICSP** - Programação in-circuit			***Dreno aberto** - Uma fonte de alimentação externa deve ser aplicada ao pino.

(FONTE: Zanco, 2005, p. 37)

Os pinos RB1 e RB2 utilizados para a comunicação serial (USART) e os pinos de alimentação V_{SS} e V_{DD} são fundamentais para o funcionamento deste projeto.

3.1.3 – Kit de Gravação PICKit2

Para gravar o *firmware* desenvolvido no microcontrolador, foi necessário um equipamento que colocasse o microcontrolador em seu modo de gravação e transfira o código compilado para a sua memória. O equipamento utilizado no desenvolvimento desse projeto foi o Gravador USB PICKit2, fornecido pela empresa RobóticaSimples® (página na *web*: http://roboticasimples.com).

Segundo o fornecedor RobóticaSimples® (2009), o gravador de PIC/EEPROM USB, integra-se ao ambiente de desenvolvimento integrado (IDE) MPLAB, grava *firmwares* criados por qualquer linguagem de programação, identifica o microcontrolador a ser gravado e possui conector ICSP (In-Circuit Serial ProgrammingTM), que permite gravar o PIC diretamente em na placa. Conta ainda com Analisador Lógico de 3 canais e conversor RS232<=>TTL sobre USB, que permite comunicar o seu computador diretamente com PIC.

3.1.3.1 – Hardware

O *hardware* do PICKit2 é composto por um circuito com conectores USB e ICSP, um *socket* ZIF de 40 pinos e um cabo USB para conexão com o computador conforme a Figura 3.3.

Figura 3.3 - Gravador PICKit2

(FONTE: Braga ,2010, p. 68)

No *socket* podem ser conectados os PIC's de 8 a 40 pinos para gravação. Também pode ser conectado a qualquer porta USB, versões 1.1 ou 2.0.

3.1.3.2 – Software

Além do circuito de gravação, o kit é composto pelo *software* PICKit2 *Programmer*. Desenvolvido pela Microchip Technology Inc.® para plataformas *Windows*, *Linux* e *MAC OS*, nele é possível o controle das funções do gravador tais como: ler, gravar, apagar e verificar o microcontrolador.

O programa também identifica o microcontrolador conectado, fornecendo ao usuário funções específicas que cada um requer.

A Figura 3.4 ilustra a tela do *software* PICKit2 *Programmer*.

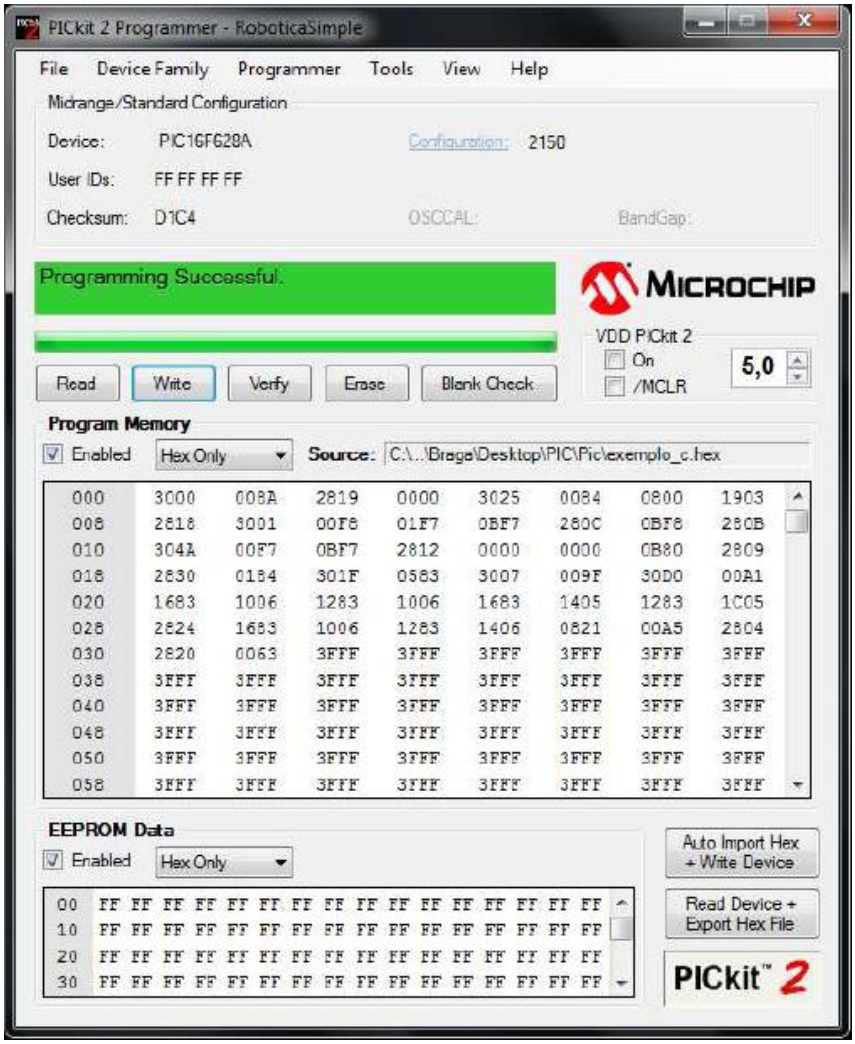

Figura 3.4 - PICKit2 *Programmer*

(FONTE: Braga, 2010, p. 69)

Para a gravação dos microcontroladores, é necessário carregar o *Programmer* com o *firmware* compilado, descrito no item 3.1.4, no seu formato hexadecimal (.HEX). O *software* aceita somente arquivos com essa extensão. Com isso, é possível a transferência do *firmware* para o microcontrolador.

3.1.4 – IDE CCS C Compiler

O IDE (*Integrated Development Environment* – Ambiente de Desenvolvimento Integrado) CCS C Compiler, fornecido pela CCS Inc.®, é um ambiente de desenvolvimento da linguagem de programação C específico para microcontralores PIC. Ele integra os operadores padrões da linguagem C e funções específicas para os registradores PIC, proporcionando aos desenvolvedores uma poderosa ferramenta para a programação do *hardware*.

Esse ambiente permite o desenvolvimento e compilação do *firmware* dos microcontroladores PIC e foi utilizado no projeto para este fim. Além disso, nele também é possível depurar o código desenvolvido.

Segundo Aho (2008, p. 1), "um compilador é um programa que recebe como entrada um programa em uma linguagem de programação – a linguagem *fonte* – e o traduz para um programa equivalente em outra linguagem – a linguagem *objeto* (ver Figura 3.5). Um papel importante do compilador é relatar quaisquer erros no programa fonte detectados durante esse processo de tradução."

Figura 3.5 - Um compilador

(FONTE: Aho, 2008, p. 1)

No caso do CCS C Compiler, o programa desenvolvido na linguagem C é traduzido para a linguagem de máquina que os microcontroladores PIC podem interpretar.

A Figura 3.6 ilustra a tela principal de desenvolvimento do ambiente, com a saída gerada do *firmware* compilado.

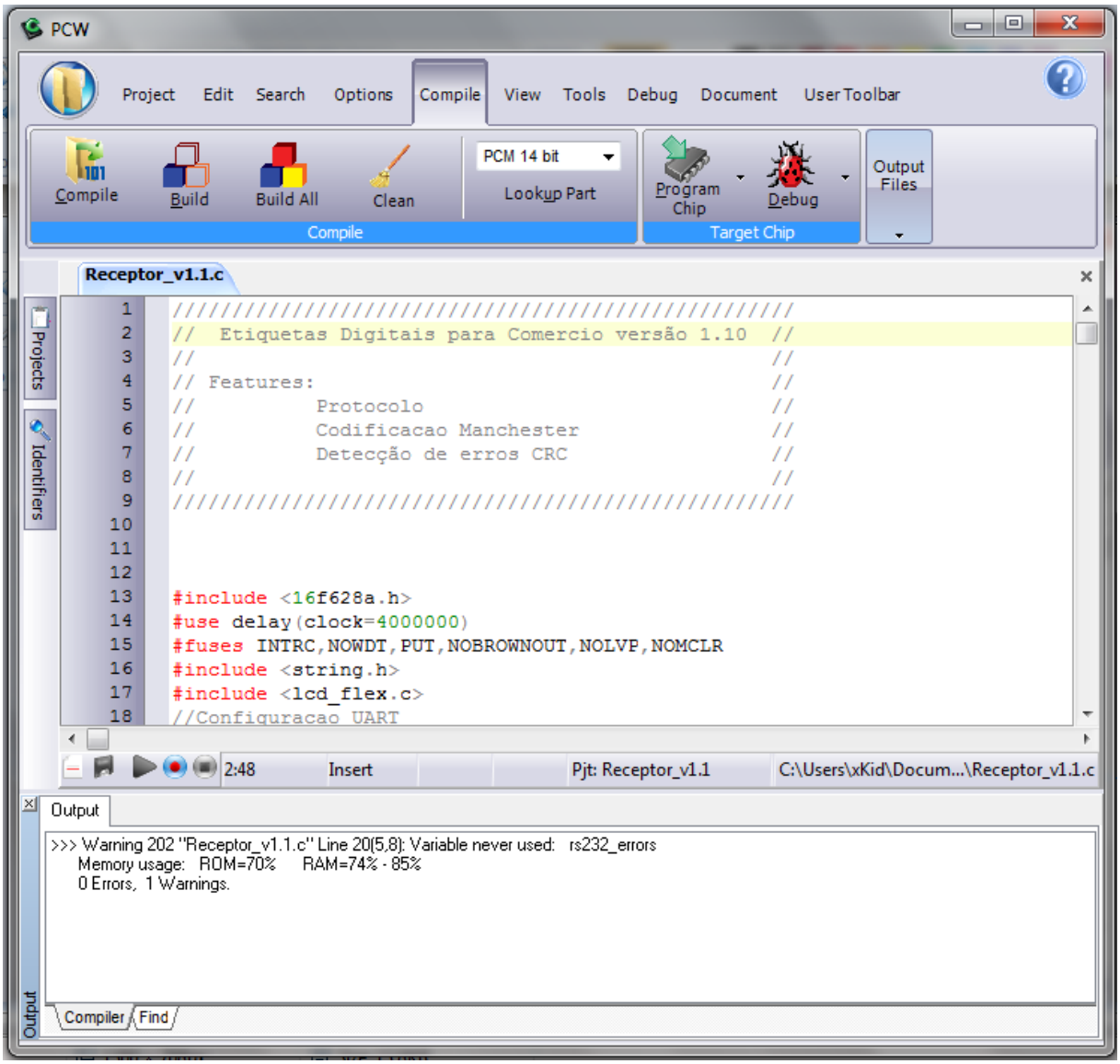

Figura 3.6 - CCS C Compiler

(FONTE: Autor)

3.2 – Transmissão dos Dados

Para a realização deste projeto foi de fundamental importância um sistema de comunicação que permitisse a troca de informações entre o sistema de gerenciamento e as etiquetas eletrônicas.

O padrão da comunicação de todo o projeto foi o RS-232 e dois meios de transmissão foram utilizados. Primeiro foi a transmissão de dados do computador para o microcontrolador através de um cabo serial, com conector DB9 e ligado a um conversor de nível RS-232 para TTL. O segundo foi a transmissão entre os microcontroladores – *Transmitter* para as etiquetas – realizada através de ondas de rádio, caracterizando uma transmissão sem fios.

3.2.1 – Padrão Serial RS-232

Segundo Alcântara (2008), o RS-232, também conhecido por EIA RS-232C ou V.24, é um padrão muito antigo de transmissão que por sua simplicidade e confiabilidade, continua sendo bastante utilizado.

A transmissão serial é caracterizada pelo envio sequencial dos *bits*. Um a um, os *bits* são enviados de forma assíncrona, ou seja, o transmissor e receptor precisam efetuar o controle de tempo para saber o início e o fim de cada *bit*.

O padrão RS-232 utiliza dois sinais de controle de fluxo via *hardware*, o *Ready To Send* (RTS) e o *Clear To Send* (CTS). A transmissão só é iniciada quando o receptor seta o pino CTS, sinalizando para o transmissor enviar os dados. Para cada *byte* transmitido existem *bits* de *start* e de *stop*. A Figura 3.7 ilustra a transmissão de um *byte*.

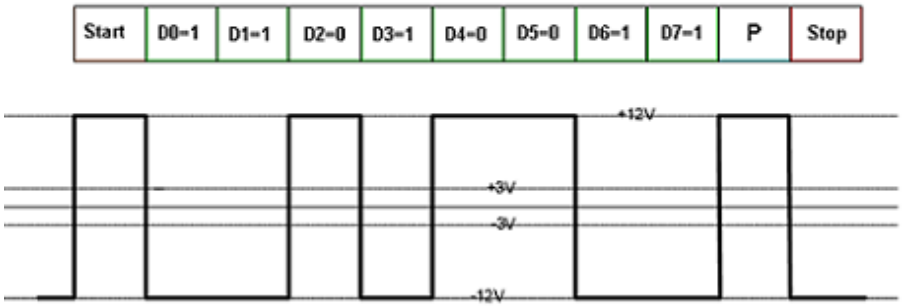

Figura 3.7 - Transmissão de um *byte* em RS-232

(FONTE: Alcântara. 2008)

Como dito anteriormente, a transmissão é assíncrona, ou seja, sem uma linha de relógio (*clock*). Então, para que seja possível a identificação dos *bits*, é preciso configurar a taxa de transmissão (*baud rate*) nos dispositivos.

O *baud rate* informa quantos *bits* no período de um segundo serão transferidos na linha. A unidade do *baud rate* é dada por bits por segundo (bps), os mais utilizados são 2400, 4800 e 9600 bps.

Nos microcontroladores há um periférico chamado UART (*Universal Asynchronous Receiver Transmitter*), que é o encarregado de todo esse controle da transmissão e gera apenas interrupções quando um *byte* é recebido.

De acordo com Alcântara (2008), "na interface RS232 o nível lógico "1" corresponde à uma tensão entre -3V e -12V e o nível lógico "0" à uma tensão entre 3V e 12V. Valores de tensão entre -3V e +3V são indefinidos e precisam ser evitados". Entretanto, o microcontrolador utilizado neste projeto não utiliza os níveis de tensão do padrão RS-232 e foi necessário um circuito adicional de conversão de níveis RS-232/TTL. O circuito integrado mais comum para efetuar essa conversão é o MAX232 (Figura 3.8).

Figura 3.8 - Circuito integrado conversor de níveis TTL/RS-232

(FONTE: Braga, 2010, p. 56)

3.2.2 – Cabo Serial DB9

A aplicação mais usual deste cabo é para comunicações seriais RS-232 e neste projeto foi utilizado para ligar o computador com o DPT *Transmitter*.

O referido cabo é composto por conectores DB9 que possuem nove pinos, conforme ilustra a Figura 3.9 e as especificações de cada pino são listadas no **Erro! Fonte de referência não encontrada.**.

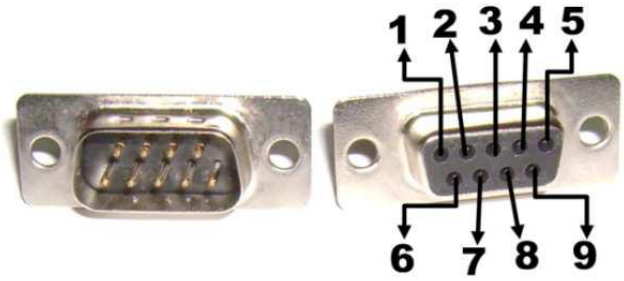

Figura 3.9 - Conector DB9

(FONTE: Braga, 2010, p. 53)

Quadro 3.2 - Pinos DB9

Número	Nome	Designação
1	CD – Carrier Detect	Detecção de portador
2	RXD – Receive Data	Recepção de dados
3	TXD – Transmit Data	Transmissão de dados
4	DTR – Data Terminal Ready	Terminal pronto
5	GND – Signal Ground	Massa lógica
6	DSR – Data Set Ready	Dados prontos
7	RTS – Request to send	Pedido de emissão
8	CTS – Clear to send	Empréstimo a emitir
9	RI – Ring Indicator	Indicador de campainha elétrica

(FONTE: Braga, 2010, p. 54)

Neste projeto não foi necessário utilizar todas as propriedades do padrão RS-232 tais como os sinalizadores RTS e CTS. Foram utilizados somente os pino 2 (RXD) e 3 (TXD) referentes, respectivamente, a recepção e transmissão de dados e o pino 5 (GND), que corresponde ao terra.

3.2.3 – Sistemas de Comunicação via Rádio

Para a comunicação entre o DPT *Transmitter* e as etiquetas eletrônicas, o meio de transmissão utilizado foi o de radiofrequência.

Sistemas de radiofrequência utilizam ondas eletromagnéticas como elemento de ligação entre transmissor e receptor. Uma das propriedades destas ondas se irradiarem pelo espaço, dispensando outros meios físicos para a transmissão.

Quando comparada à sistemas cabeados, a transmissão via rádio apresenta algumas características: confiabilidade menor e dependência das condições de propagação da onda, custo de implementação menor para transmissão em distâncias superiores e flexibilidade para

ampliação. Com essas características, este sistema é adequado tanto para comunicação a longa distância quanto para mobilidade, duas características deste projeto.

A composição de um sistema de comunicação via rádio é feita por, pelo menos, duas estações de rádio, uma transmissora e outra receptora conforme Figura 3.10.

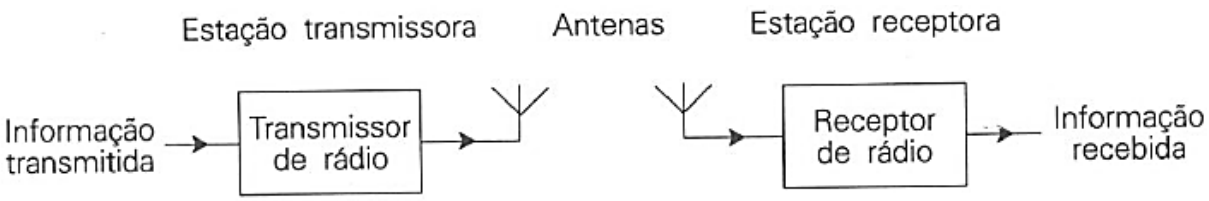

Figura 3.10 - Diagrama de blocos de um sistema de comunicação via rádio

(FONTE: Nascimento, 2000, p. XXVI)

Segundo Nascimento (2000, p. XXVI), “cada estação é composta por um equipamento de rádio, uma linha de transmissão e uma antena. A função dos equipamentos de rádio é, no caso do transmissor, gerar sinais de radiofreqüência e, no caso do receptor, recebê-los; da linha de transmissão é conduzir o sinal de radiofreqüência do transmissor até a antena, ou da antena até o receptor. Finalmente, a função da antena é gerar ou captar ondas eletromagnéticas.”

Para equipar o transmissor do projeto, foi escolhido o componente TXC1 (Figura 3.11) da empresa Keymark Technology®, que opera na frequência de 433.92MHz e utiliza modulação por chaveamento de amplitude.

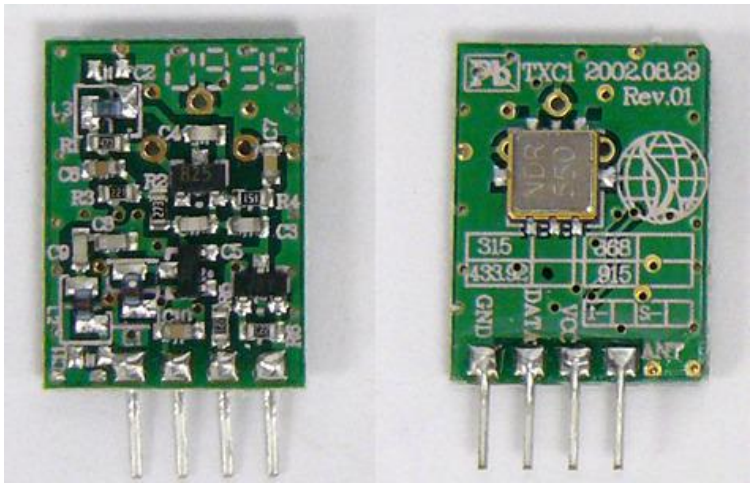

Figura 3.11 - Transmissor 433.92MHz TXC1

(FONTE: Keymark Technology. 2011)

As especificações de seus pinos são:

- Pino 1 – Vcc 5v;
- Pino 2 – Data;
- Pino 3 – GND;
- Pino 4 – Antena.

ASK ou *Amplitude Shift Keying*, é uma técnica de modulação utilizada em sinais digitais. Consiste na alteração da onda portadora em função do sinal digital a ser transmitido. A amplitude da portadora é comutada entre dois valores, ligado ou desligado, resultando em pulsos de radiofrequência representando o sinal binário "1" e em espaços que são equivalentes ao sinal binário "0" (Figura 3.12).

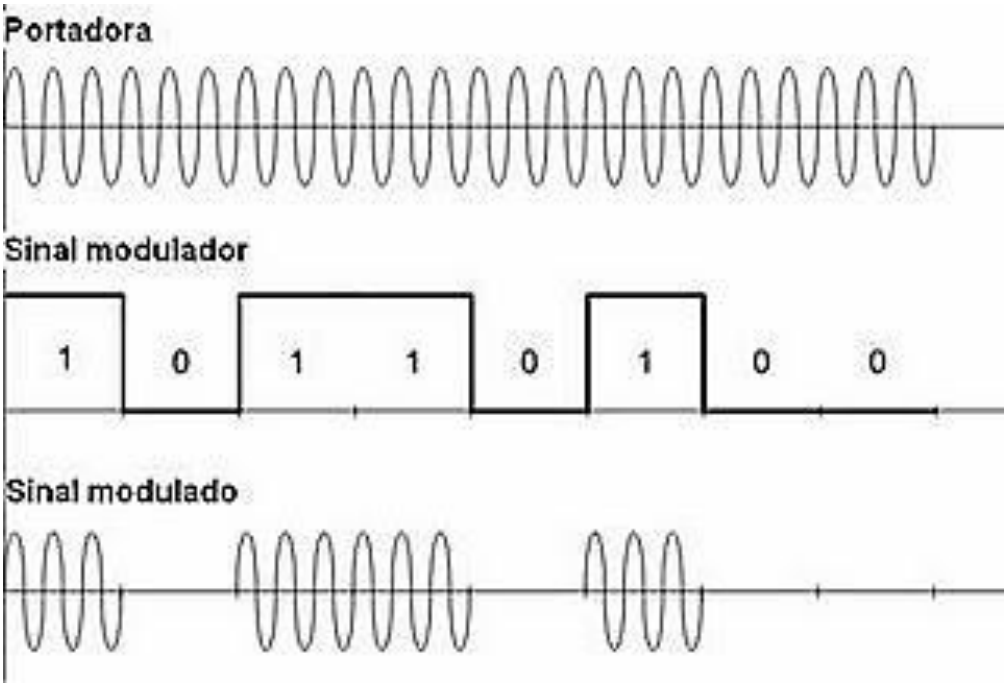

Figura 3.12 - Modulação ASK

(FONTE: UFRGS, 2011)

Para completar o sistema de comunicação via rádio, o receptor que foi utilizado nas etiquetas eletrônicas também é da empresa Keymark Technology®, modelo RXD1.

De acordo com o fabricante, o RXD1 (ver Figura 3.13), é um módulo que recebe sinais modulados em *On-Off Keyed* (OOK), caso particular da modulação ASK, e demodula para um sinal digital. Este módulo também opera na frequência de 433.92MHz.

Figura 3.13 - Receptor 433.92Mhz RXD1

(FONTE: Keymark Technology. 2011)

As especificações dos seus pinos são:

- Pino 1 – Antena;
- Pino 2 – GND;
- Pino 3 – GND;
- Pino 4 – V_{CC};
- Pino 5 – V_{CC};
- Pino 6 – Data;
- Pino 7 – Data;
- Pino 8 – GND.

Para o funcionamento dos módulos transmissor e receptor foi necessário a instalação de uma antena para irradiar as ondas e recebê-las.

Diz Nascimento (2000, p. XXVII) que, "como é grande o número de estações transmissoras existentes, uma antena receptora irá captar inúmeros outros sinais, além do sinal desejado. Portanto, antes de recuperar a informação contida em um determinado sinal, é necessário separá-lo."

Para resolver o problema da captação de outros sinais, neste projeto foi implementado uma técnica chamada *Look At Me* (LAM) ou preâmbulo de sincronização que consiste em

enviar uma sequência de 0’s e 1’s, antes de transmitir a informação propriamente dita. O receptor ao receber essa sequência, sincroniza com esse transmissor até o final da transmissão. Os outros sinais indesejados não são captados nesse tempo. No CAPÍTULO 4, item 4.3.2.1 o procedimento é melhor detalhado.

3.3 – Visor de Cristal Líquido – LCD

O LCD (*Liquid Crystal Display)* ou, em português, visor de cristal líquido é um dispositivo utilizado em aparelhos eletrônicos com a finalidade de exibir resultados ou informações. Muitos dispositivos fazem uso deste *display*, alguns exemplos são:

- Calculadoras;
- Telefones;
- Televisores;
- Painel de instrumentos.

Por ter baixo consumo de energia, pode ser utilizado em aparelhos portáteis alimentados inclusive por bateria.

Pelas características apresentadas este dispositivo se adequa a este projeto, que tem como proposta exibir as informações referentes aos produtos anunciados como a marca, o nome e o valor.

A Figura 3.14 ilustra o *display* utilizado para a confecção das etiquetas eletrônicas e a **Erro! Fonte de referência não encontrada.** exibe as especificações dos pinos.

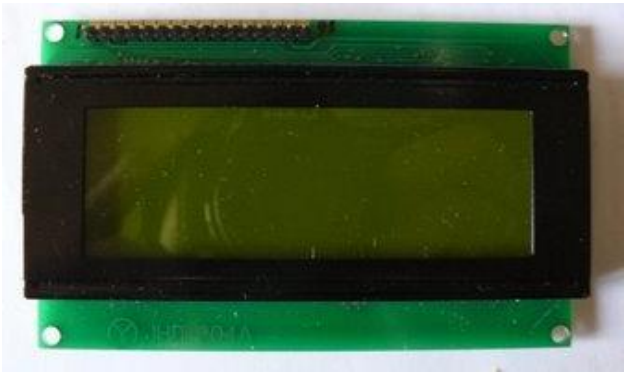

Figura 3.14 - LCD JHD 204A

(FONTE: Autor)

Tabela 3.1 - Pinos do *display*

Pino	Símbolo	Função
1	VSS	Alimentação – GND
2	VCC	Alimentação – 5V
3	VEE	Contraste
4	RS	Registrador de Comandos/Dados
5	R/W	Leitura/Escrita
6	E	*Enable*
7 a 14	DB0 à DB7	Dados
15	LED	Anodo do LED
16	LED	Catodo do LED

(FONTE: Autor)

3.4 – Linguagens de Programação

Entende-se por linguagens de programação instruções escritas com uma sequência lógica para serem interpretadas por um processador. São códigos escritos por pessoas para serem executados em uma máquina.

As linguagens são divididas em dois tipos: as de baixo nível e as de alto nível. Processadores são capazes de reconhecer somente o código binário e as linguagens de baixo nível. Entretanto a programação utilizando essas linguagens é muito difícil. Já as de alto nível são mais fáceis de se trabalhar e entender e foram criadas para facilitar o entendimento das pessoas. Antes de serem interpretadas, é necessário a tradução do código por um compilador, comentado anteriormente no item 3.1.4 na página 33.

Duas linguagem foram utilizadas neste projeto. Os microcontroladores que compõem a solução foram programados utilizando a linguagem C. Já a interface do usuário, que é executado pelo computador, foi desenvolvida utilizando a linguagem Java.

3.4.1 – Linguagem de Programação Java

Jandl (2002, p. 5), define Java como “mais do que uma simples linguagem de programação, ou seja, é todo um ambiente de desenvolvimento e execução de programas que

reúne um conjunto ímpar de facilidades: é completamente orientada a objetos, muito robusta, possui extrema portabilidade, permite a operação em rede e distribuição, incorpora diversas características voltadas a segurança e, finalmente, permite sua distribuição em diversos ambientes, destacando-se a Internet.”

A linguagem Java foi escolhida por conta de suas características, é uma linguagem robusta e fácil de ser implementada, além de encontrar facilmente suporte e documentação na *Web*. Outra característica é ser orientada a objetos. Segundo Deitel (2006), um objeto pode ser entendido assim como na vida real, por exemplo um carro, um telefone ou uma casa. Cada objeto possui uma classe que contém atributos e métodos que o caracterizam.

A interface gráfica desenvolvida para este projeto foi feita utilizando uma API (*Application Programming Interface*) do Java chamada *Swing* e com ela foi possível desenhar as janelas do programa.

A ilustra uma aplicação desenvolvida utilizando a API *Swing*.

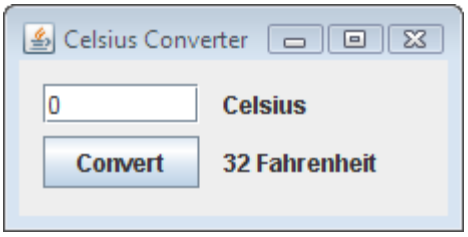

Figura 3.15 - Aplicação *Swing*

(FONTE: ORACLE. 2011)

CAPÍTULO 4 – IMPLEMENTAÇÃO

Neste capítulo são tratados os métodos implementados para o desenvolvimento do projeto. Ele está dividido em duas partes principais: a primeira parte trata do modelo do *hardware* desenvolvido para exemplificar a solução que é composta por um transmissor de dados e os receptores que simulam a etiquetas eletrônicas. Na segunda parte é apresentada a implementação dos *softwares* desenvolvidos para o funcionamento do sistema de etiquetas. São descritos o *firmware* que é executado no microcontrolador e o programa que intermedia o administrador do sistema e as etiquetas.

4.1 – *Hardware* do Módulo Transmissor - DPT *Transmitter*

O módulo transmissor, ou como chamado no projeto de DPT *Transmitter*, é o *hardware* que interliga o computador através da comunicação serial RS-232 com as etiquetas eletrônicas, por comunicação sem fio por radiofrequência.

Na montagem deste módulo, três componentes principais foram utilizados para realizar a tarefa descrita acima. São eles:

1. Microcontrolador PIC16F628A;
2. Circuito Integrado MAX232;
3. Transmissor RF TXC1 433.92MHz.

O PIC16F628A é a parte central do módulo, responsável pelo processamento das informações recebidas pela comunicação serial, que é convertida para sinais TTL pelo CI MAX232. A conversão para TTL é necessária devido ao microcontrolador não suportar os sinais originais do padrão RS-232.

Após o processamento, as informações são enviadas ao transmissor TXC1, que realiza a modulação do sinal e o envia para as etiquetas por ondas com frequência de 433.92MHz. Estes dados estão disponíveis para qualquer receptor que capte esta frequência, porém somente podem ser interpretadas por dispositivos que conheçam o protocolo de comunicação implementado. Este protocolo foi de autoria própria e os detalhes são abordados no item 4.3.2.2, página 72. O processo de transmissão é ilustrado na Figura 4.1.

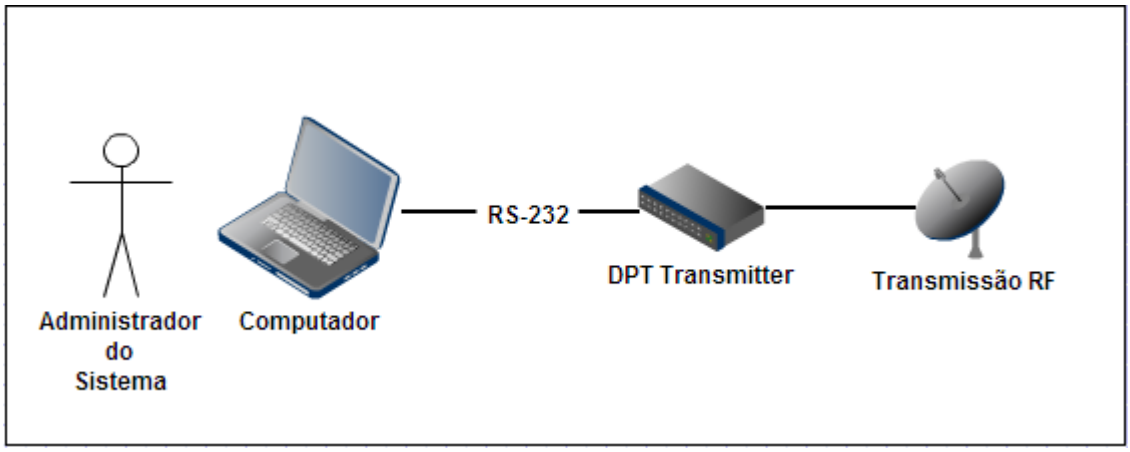

Figura 4.1 - Diagrama da transmissão

(FONTE: Autor)

Primeiramente, o administrador do sistema solicita o sincronismo dos dados por meio de uma interface desenvolvida e, em seguida, o computador os envia ao módulo transmissor. Ao receber as informações, o módulo processa os dados recebidos e os adaptam em um pacote que segue o formato de um protocolo de comunicação, denominado no projeto de XP (*X-Protocol*). Este pacote é composto por: *bytes* de controle, dados codificados e *bytes* de detecção de erros. O formato e as características do pacote é melhor detalhado no item 4.3.2.2, página 73.

4.1.1 – Montagem do transmissor

A montagem dos componentes do módulo foi realizada seguindo instruções dos próprios fabricantes e adequada às necessidades do projeto.

4.1.1.1 – Comunicação Serial com MAX232

Como os computadores atuais não possuem mais portas seriais para a ligação do computador com o módulo, foi necessário um cabo conversor de USB para serial, adquirido em lojas de informática, ilustrado na Figura 4.2. Este conversor foi ligado a um cabo serial montado, utilizando um conector DB9 fêmea, conectado aos pinos 2, 3 e 5 - RXD, TXD e GND respectivamente - e ligados ao CI MAX232.

Figura 4.2 - Conversor USB – Serial

(FONTE: Autor)

O MAX232 foi montado de acordo com o *datasheet* do fabricante. Segue sua configuração:

- Pinos 1 e 3 – Capacitor de 1µF;
- Pinos 4 e 5 – Capacitor de 1µF;
- Pino 6 e GND – Capacitor de 1µF;
- Pinos 2 e 16 – Capacitor de 1µF;
- Pinos 15 e 16 – Capacitor de 10µF;
- Pino 15 – GND;
- Pino 16 – 5V;
- Pino 12 – Pino 7 do PIC16F628A;
- Pino 13 – Pino 3 do DB9;
- Pino 14 – Pino 2 do DB9;
- Pino 15 – Pino 5 do DB9;

A Figura 4.3 representa o circuito do MAX232 montado no *software* de simulação de circuitos, Proteus®. A Figura 4.4 ilustra o circuito montado na placa para montagem de circuitos experimentais, *protoboard*.

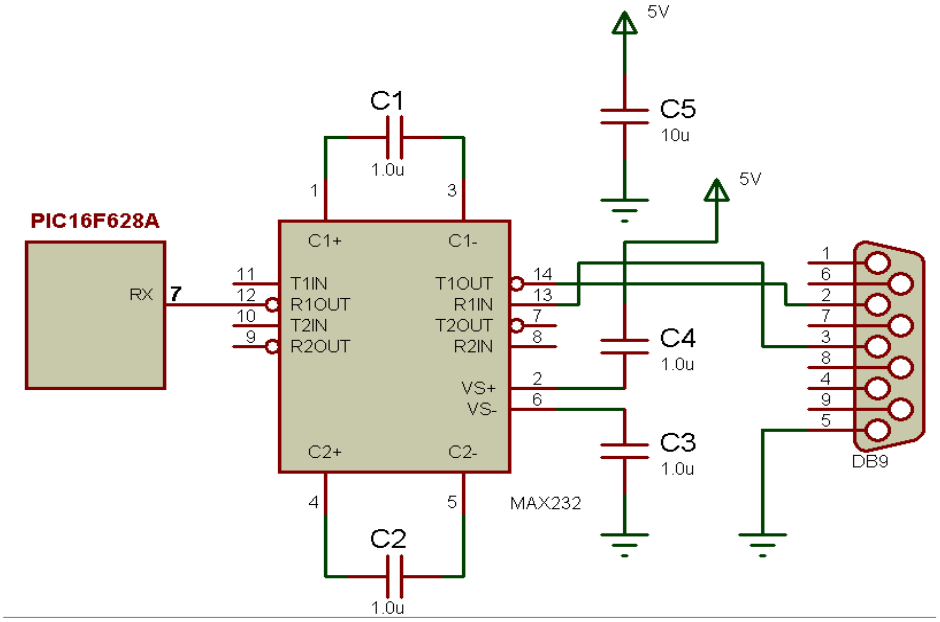

Figura 4.3 - Representação do circuito MAX232 no Proteus®

(FONTE: Autor)

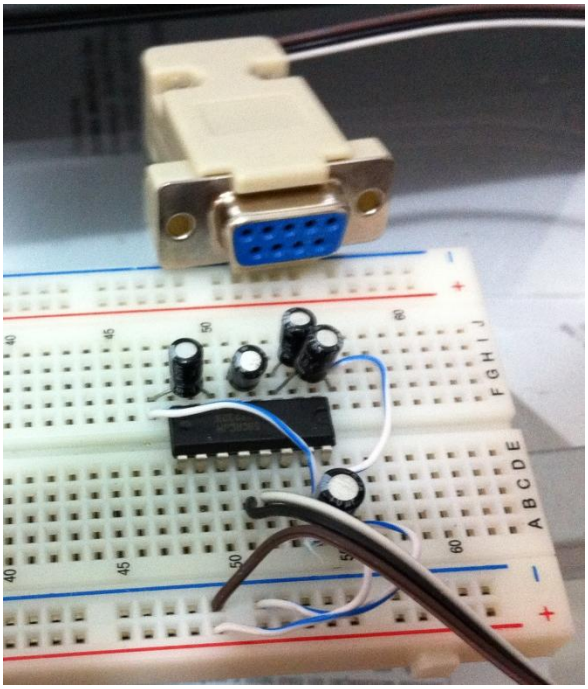

Figura 4.4 - Circuito MAX232 na *protoboard*

(FONTE: Autor)

4.1.1.2 – PIC16F628A

O microcontrolador PICF628A foi ligado de forma a receber os dados do MAX232, processá-los e encaminhá-los ao transmissor de radiofrequência. Nele foram conectados LED's indicadores, além do resistor de 10KΩ e do capacitor de 1pF recomendados pela Microchip Inc.®. A Figura 4.5 ilustra o microcontrolador utilizado neste projeto.

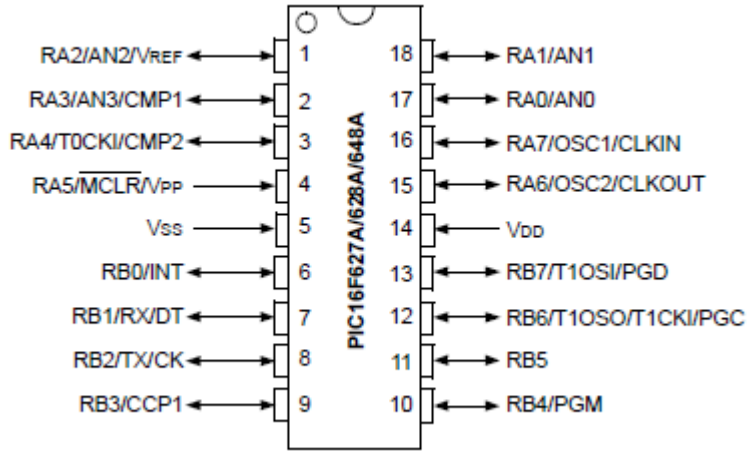

Figura 4.5 - PIC16F628A

(FONTE: Microchip Technology Inc. ®, 2005, p. 2)

Os pinos do microcontrolador ficaram conectados da seguinte forma:

- Pino 4 – Resistor 10KΩ e 5V;
- Pino 5 – GND;
- Pino 7 – Pino 12 do MAX232;
- Pino 8 – Pino 2 do TXC1;
- Pino 9 – LED indicativo de transmissão e pino 3 do TXC1;
- Pino 14 – 5V e capacitor de 1pF;

O pino 4 do PIC é para *reset* do microcontrolador. O resistor conectado a ele é do tipo *pull-up*, que eleva a tensão do pino ao nível alto (5V) evitando que ocorram *resets* inesperados.

Os pinos 5 e 14 são de alimentação do microcontrolador e são conectados a um capacitor para eliminar ruídos da fonte externa.

O pino de recepção de dados (RXD) do microcontrolador é o pino 7, e os dados provenientes do MAX232 são recebidos por este pino. Já o pino 8, que é o TX do microcontrolador, está conectado ao pino 2 (RXD) do transmissor de radiofrequência.

Um LED foi ligado em série com um resistor de 220Ω e ao pino 9 e foi utilizado para indicar quando há transmissão pelo módulo. Este pino também alimenta o transmissor TXC1.

4.1.1.3 – Transmissor TXC1

O módulo transmissor TXC1 possui quatro pinos e sua ligação é bem simplificada, como segue:

- Pino 1 – GND;
- Pino 2 – Pino 8 do PIC;
- Pino 3 – Pino 9 do PIC;
- Pino 4 – Antena.

O pino 2 do TXC1 é o de recepção de dados e por ele são recebidos os dados do microcontrolador, que serão modulados e transmitidos em radiofrequência (RF).

Para que o TXC1 não transmita dados aleatórios enquanto estiver fora de uso, a configuração foi feita para que o microcontrolador controle o momento de ligar e desligar o transmissor e para isso o pino de alimentação (3) foi ligado ao pino 9 do PIC.

Na Figura 4.6 é ilustrado o circuito do transmissor montado em *protoboard*.

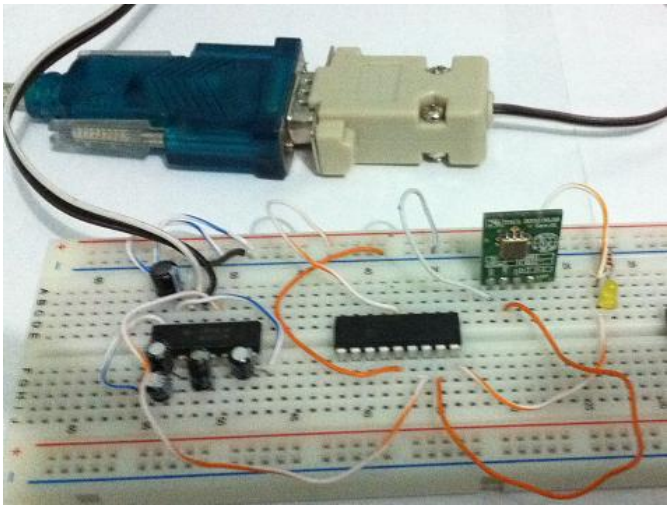

Figura 4.6 - Circuito transmissor em *protoboard*

(FONTE: Autor)

4.1.1.4 - Alimentação

Para alimentar o módulo transmissor, uma fonte externa de 9VDC e de 350mA foi utilizada. Como a tensão da fonte era superior a tensão de trabalho dos demais componentes, foi necessário um regulador de tensão que a ajustasse para 5VDC.

O regulador de tensão em questão foi o LM7805 o qual foi conectado segundo instruções recomendadas no *datasheet* do fabricante. O regulador possui 3 pinos, um de entrada, um terra e uma saída. A entrada suporta tensões de 6VDC até 35VDC e faz a regulagem para 5VDC. Em tensões superiores a 16VDC é recomendado utilizar um dissipador de calor para evitar que o aquecimento gerado por ele afete seu funcionamento.

Como recomendação do fabricante para eliminar ruídos provenientes da rede elétrica, foi utilizado um capacitor de 100µF/50V entre o pino de entrada e o terra do LM7805, além de outro capacitor de 10µF/50V entre a saída e o terra. Neste circuito também foi utilizado um LED para indicar quando o mesmo estiver ligado. O LED foi conectado em série com um resistor de 220Ω entre os pinos 2 e 3 do regulador.

A Figura 4.7 representa o esquemático no Proteus® da ligação do regulador.

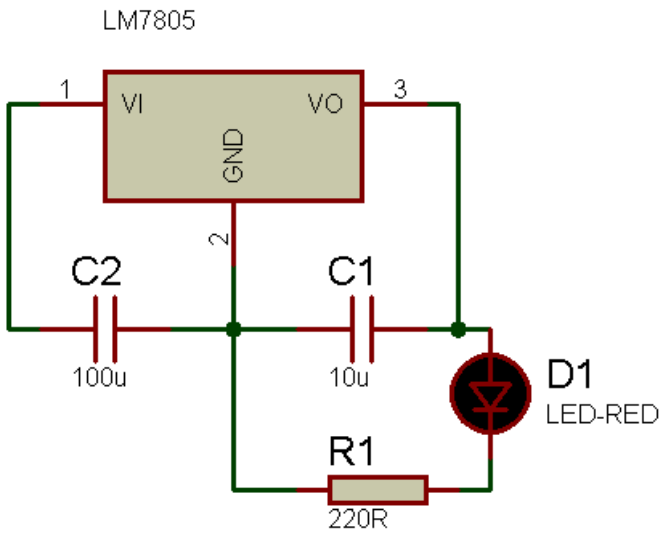

Figura 4.7 - Ligação do regulador de tensão

(FONTE: Autor)

4.1.2 – Elaboração do Circuito Impresso do Transmissor

Com o protótipo testado em *protoboard*, o circuito foi redesenhado para ser impresso em uma placa de fenolite. Assim possibilitou ocorrer a montagem da placa de circuito impresso do transmissor.

O *layout* do circuito foi desenhado utilizando o *software* ARES®, específico para esta função e pode ser visto na Figura 4.8.

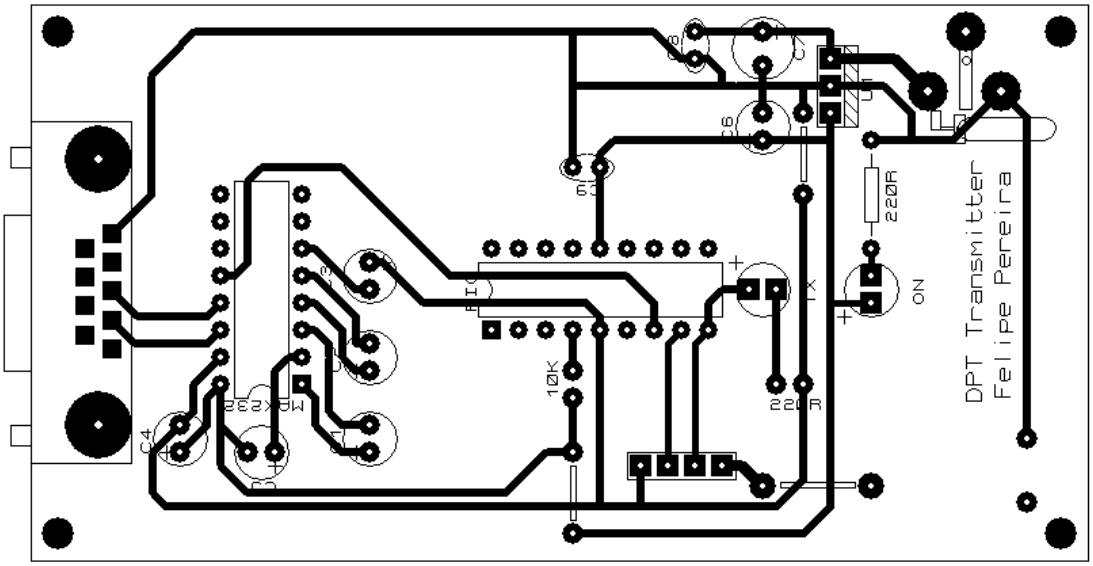

Figura 4.8 - *Layout* da placa do transmissor

(FONTE: Autor)

O desenho foi impresso em transparência para retroprojetor em impressora LASER e o método de transferência do desenho para a placa de fenolite foi a transferência térmica.

Com o desenho transferido, a placa foi imersa em uma solução química de percloreto de ferro para proceder com a corrosão do cobre nas áreas não cobertas pelo desenho das trilhas. Após a corrosão, a placa foi limpa com thinner e palha de aço para remover os resíduos de tinta.

Para evitar a oxidação do cobre, um verniz a base de breu e thinner foi aplicado na placa, que também foi perfurada para a fixação dos componentes.

A Figura 4.9 e a Figura 4.10 ilustram os dois lados da placa de fenolite, e a Figura 4.11 ilustra o transmissor completamente montado.

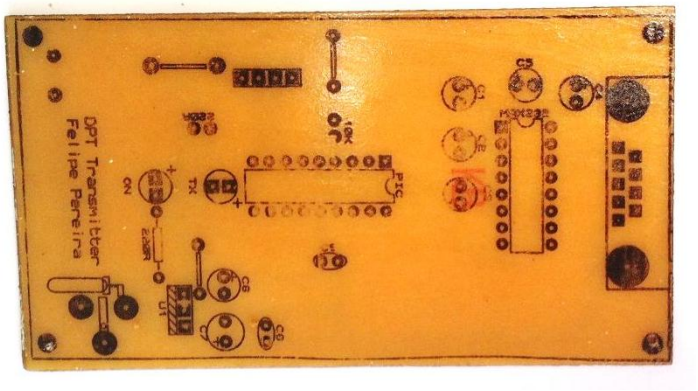

Figura 4.9 - Placa fenolite – lado dos componentes

(FONTE: Autor)

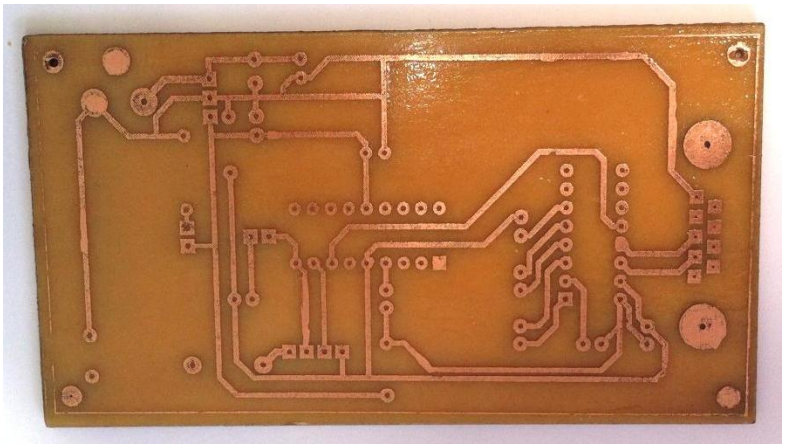

Figura 4.10 – Placa fenolite – lado das trilhas da placa

(FONTE: Autor)

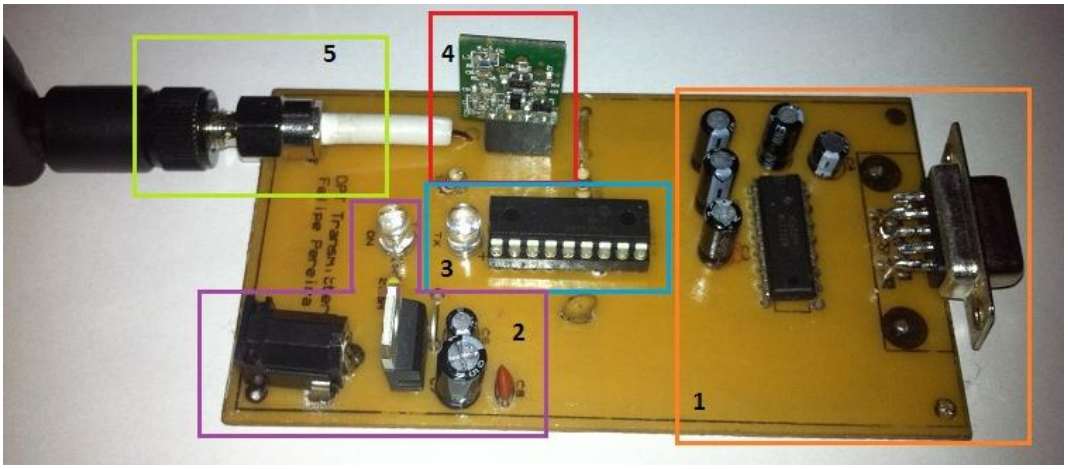

Figura 4.11 - Transmissor montado e circuitos enumerados

(FONTE: Autor)

Na figura anterior, são enumerados os cinco itens que compõem a placa. São eles:

1. Circuito MAX232 e conector DB9 fêmea;
2. Conector Jack para fonte externa e circuito LM7805
3. PIC16F628A;
4. Transmissor TXC1;
5. Conector da antena e a própria antena.

4.2 – *Hardware* do Módulo Receptor – Digital Price Tag

O *hardware* desenvolvido para simular as etiquetas eletrônicas são compostos por dois circuitos que foram implementados com configurações físicas idênticas. A diferença entre os protótipos foi apenas em *software*, diferenciando somente o identificador de cada etiqueta, de modo que cada uma fosse endereçado distintamente.

Como no módulo transmissor, há componentes principais no modelo desenvolvido. São eles:

1. Microcontrolador PIC16F628A;
2. Receptor RF RXD1 433.92MHz;
3. LCD 20 linhas por 4 colunas.

A Figura 4.12 ilustra o princípio de funcionamento da etiqueta eletrônica.

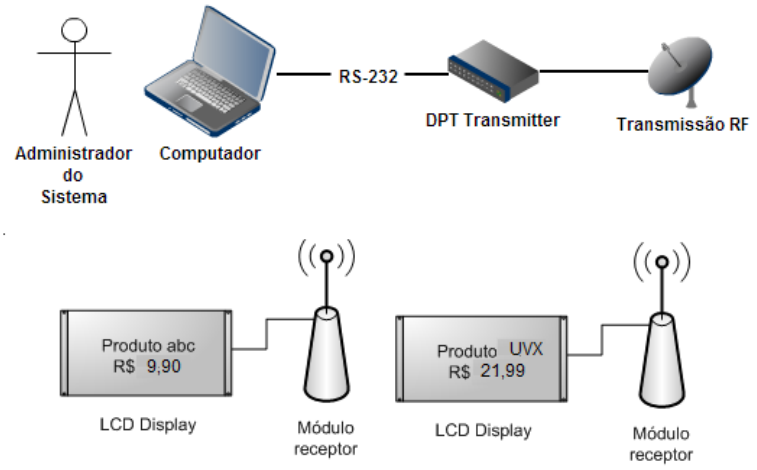

Figura 4.12 - Diagrama etiquetas eletrônicas

(FONTE: Autor)

Aqui também, o principal componente para o processamento das informações é o PIC16F628A, que é ligado ao receptor RXD1 e ao LCD de 20 colunas e 4 linhas.

O módulo receptor, neste projeto, foi o único desenvolvido para interpretar o protocolo de comunicação. Quando um pacote válido deste protocolo for recebido pela etiqueta eletrônica, a informação é decodificada e armazenada na memória interna EEPROM de 128 *bytes* do microcontrolador, e posteriormente transcritas no LCD.

A memória EEPROM é fundamental para que nos casos de desligamento do módulo receptor ou falta de bateria, as informações que foram sincronizadas não se percam. Ao reestabelecer a alimentação do módulo, a memória é lida e se houver informações gravadas, estas serão novamente transcritas no LCD.

4.2.1 – Montagem da Etiqueta Eletrônica

Como feito no transmissor, os componentes da etiqueta foram ligados de acordo com as especificações dos fabricantes e algumas alterações foram realizadas para se adequar ao projeto, como será visto a seguir.

4.2.1.1 – PIC16F628A

Na ligação entre o principal circuito integrado da etiqueta eletrônica e os demais componentes, a configuração dos pinos ficou disposta do seguinte modo:

- Pino 4 – Resistor *pull-up* 10KΩ e 5V;
- Pino 5 – GND;
- Pino 7 – Pino 2 do receptor RXD1;
- Pino 9 – LED indicativo;
- Pino 10 – Pino 11 do LCD;
- Pino 11 – Pino 12 do LCD;
- Pino 12 – Pino 13 do LCD;
- Pino 13 – Pino 14 do LCD;
- Pino 14 – 5V;
- Pino 17 – Pino 4 do LCD;
- Pino 18 – Pino 6 do LCD.

Da mesma forma que no transmissor, um resistor do tipo *pull-up* foi ligado entre o MCLR (pino 4) do microcontrolador e os 5V para evitar *resets* inesperados.

Os sinais recebidos pelo receptor RXD1 são demodulados e lidos pelo microcontrolador através da porta RXD (pino 7).

Os pinos 10 a 14 do PIC são ligados aos pinos de dados do LCD, D4 a D7. A comunicação entre o microcontrolador e o *display* é feita por transmissão paralela de 4 bits. Os pinos 17 e 18, que também são conectado ao *display*, se referem as portas de controle do LCD, registradores de comandos – RS (pino 4) – e *Enable* (pino 6).

A alimentação do PIC fica por conta dos pinos 5 e 14, sendo o 5 o terra e o 14 o V_{CC}. Entre eles também foi ligado um capacitor cerâmico de 1pF para eliminar ruídos da fonte.

4.2.1.2 – Receptor RF RXD1 433.92MHz

O RXD1, responsável por captar sinais enviados por ondas de radiofrequência na faixa de 433.92MHz, foi ligado na etiqueta eletrônica para permitir a comunicação entre o transmissor e a etiqueta. O receptor contém oito pinos, quatro em cada extremidade, e foram conectados da seguinte maneira:

- Pinos 1, 6 e 7 – GND;
- Pino 2 – Dados;
- Pinos 4 e 5 – 5V;
- Pino 8 – Antena.

Apesar dos vários pinos, a ligação do receptor é basicamente a alimentação, os dados e a antena.

O pino 2 é onde o receptor dispõe os dados recebidos por radiofrequência já demodulados e prontos para serem interpretados pelo microcontrolador. Ele é ligado ao pino RXD do PIC (pino 7).

No pino 8 foi ligada uma antena para melhorar a recepção das ondas eletromagnéticas. Para isso, uma antena foi desenvolvida com características semelhantes às vendidas no mercado. Seguindo especificações[1] de uma antena helicoidal de 433.92MHz comercial,

[1] As dimensões da figura referenciada são dadas em milímetros.

conforme ilustrado na Figura 4.13, foi montada uma antena utilizando um tubo de plástico com o mesmo diâmetro interno e externo, e um fio de cobre de 26 AWG.

A antena montada para o protótipo é ilustrada na Figura 4.14.

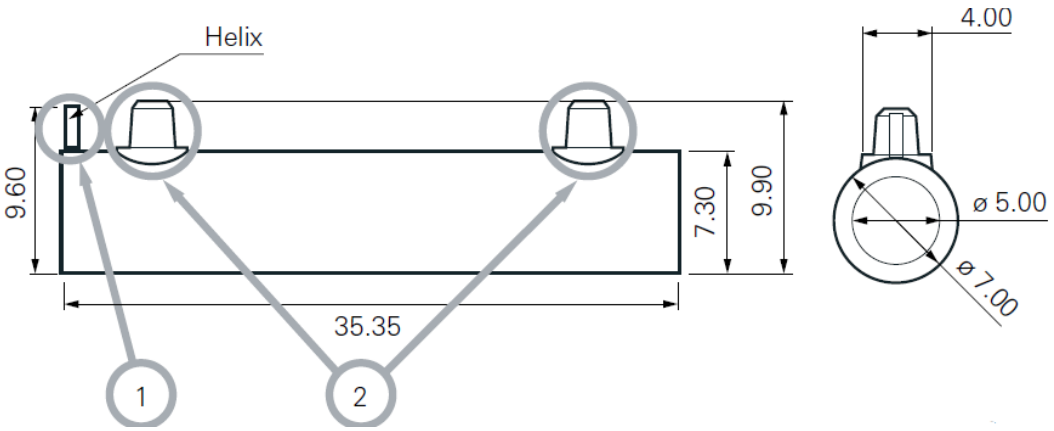

Figura 4.13 - Antena Helicoidal 433.92MHz

(FONTE: PULSEENG. 2011)

Figura 4.14 - Antena montada para o projeto

(FONTE: Autor)

Entretanto, os resultados obtidos com a antena desenvolvida não foram eficientes e novos modelos com diferentes tamanhos foram testados.

No item 5.1 do CAPÍTULO 5 encontra-se a explicação do desenvolvimento de outros modelos de antenas e dos testes feitos para a solução do problema.

4.2.1.3 – LCD

Neste protótipo, as informações sincronizadas são exibidas no visor de cristal líquido. O *display* foi ligado ao microcontrolador para que este envie os dados a serem exibidos. A configuração dos pinos são:

- Pino 1 – GND;
- Pino 2 – 5V;
- Pino 3 – Potenciômetro de 10KΩ e GND;
- Pino 4 – Pino 17 do PIC;
- Pino 5 – GND;
- Pino 6 – Pino 18 do PIC;
- Pino 11 a 14 – Pinos 10 a 13 respectivamente.

Além da alimentação do *display*, entre os pinos 1 e 2, um potenciômetro de 10KΩ foi ligado ao pino 3 para ajuste de contraste do visor.

Os pinos 4 e 6, já comentados anteriormente, são portas de controle do *display*.

Os pinos de dados, 11 ao 14 são ligados para receber as informações do microcontrolador por uma comunicação paralela de 4 *bits*. A comunicação de 4 *bits* foi escolhida para economizar portas de saída do microcontrolador.

A exibição dos dados no visor ficou disposta de modo que na primeira linha do *display* seja exibido o nome do produto, na segunda linha, entre as colunas 1 a 11 a marca e entre as colunas 12 a 20 símbolo da moeda brasileira – R$ – e o valor do produto.

A Figura 4.15 ilustra um exemplo de exibição das informações.

Figura 4.15 - Descrição de um produto na etiqueta eletrônica

(FONTE: Autor)

4.2.1.4 – Alimentação do Circuito

Para alimentar todos os circuitos do protótipo da etiqueta eletrônica, foi utilizado uma bateria de 9V.

A tensão máxima suportada por todos os componentes é de 5V e para o ajuste da tensão para o nível correto, um regulador de tensão foi montado. O regulador utilizado foi o LM78L05, ligado de acordo com o *datasheet* do fabricante. Dos 3 pinos do regulador, o pino 1 recebe o positivo da bateria, o 2 é ligado ao negativo e o 3 é a saída ajustada de 5V. Entre os pinos 1 – 2 e 2 – 3 são ligados dois capacitores cerâmicos de 10pF, conforme recomendado pelo fabricante.

A Figura 4.16 ilustra o esquemático do circuito desenvolvido no Proteus®.

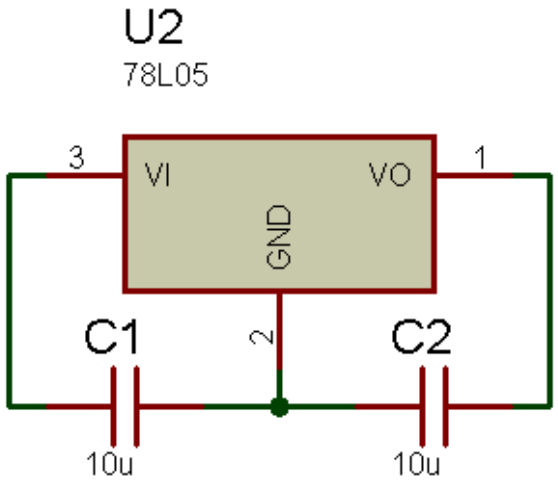

Figura 4.16 - Circuito LM78L05

(FONTE: Autor)

4.2.1.5 – Elaboração do Circuito Impresso Receptor

Os componentes do módulo receptor foram primeiramente conectados em uma *protoboard* para possibilitar o desenvolvimento do *firmware* e testes. A interligação dos circuitos e a descrição de um produto é ilustrada na Figura 4.17.

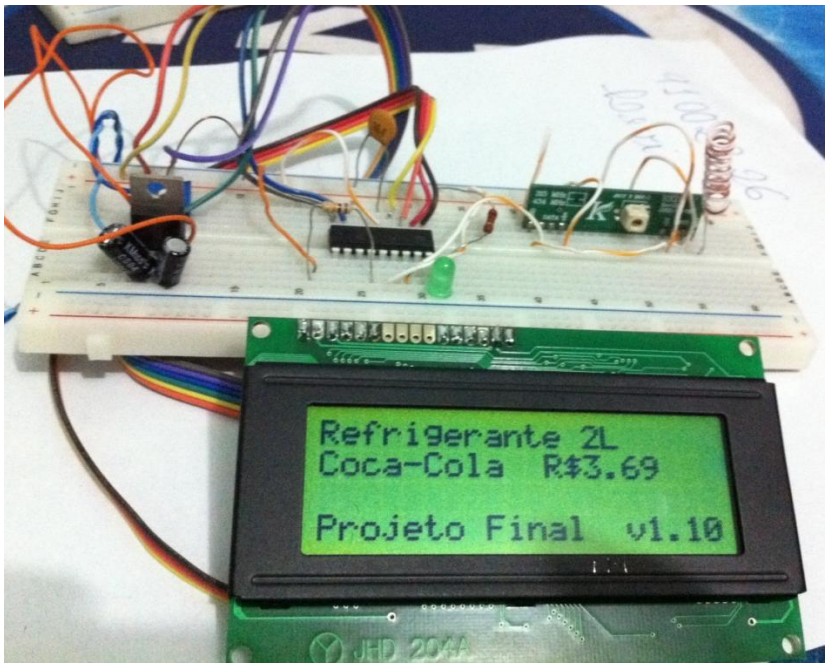

Figura 4.17 - Circuito receptor e exemplo de descrição de um produto

(FONTE: Autor)

Com o *firmware* desenvolvido e os testes finalizados, o próximo passo foi a elaboração do circuito impresso das etiquetas eletrônicas.

Utilizando o *software* de PCB *Layout* ARES®, as trilhas da placa foram desenhadas conforme ilustração da Figura 4.18.

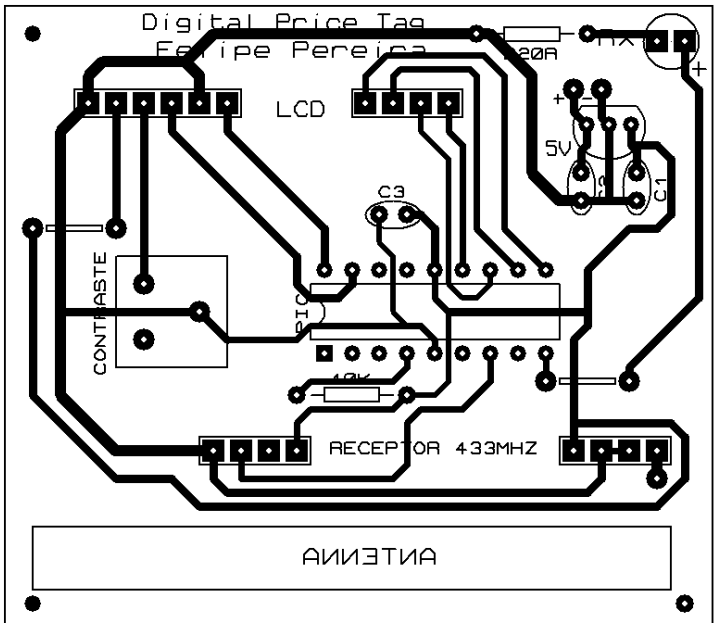

Figura 4.18 - *Layout* da placa da etiqueta eletrônica

(FONTE: Autor)

O método para impressão nas placas de fenolite foi o mesmo utilizado no desenvolvimento do transmissor, descrito anteriormente. Os desenhos foram passados para as placas por transferência térmica, em seguida foram corroídas, envernizadas, perfuradas e montadas.

As próximas figuras ilustram uma etiqueta eletrônica ao final do processo de montagem.

A Figura 4.19 ilustra as trilhas da placa de fenolite, a antena conectada, a parte traseira do LCD e a bateria de alimentação da etiqueta. A face dos componentes é ilustrada na Figura 4.20 juntamente com o LCD. A Figura 4.21 contém os circuitos enumerados. Logo, a Figura 4.22 ilustra uma etiqueta eletrônica em funcionamento.

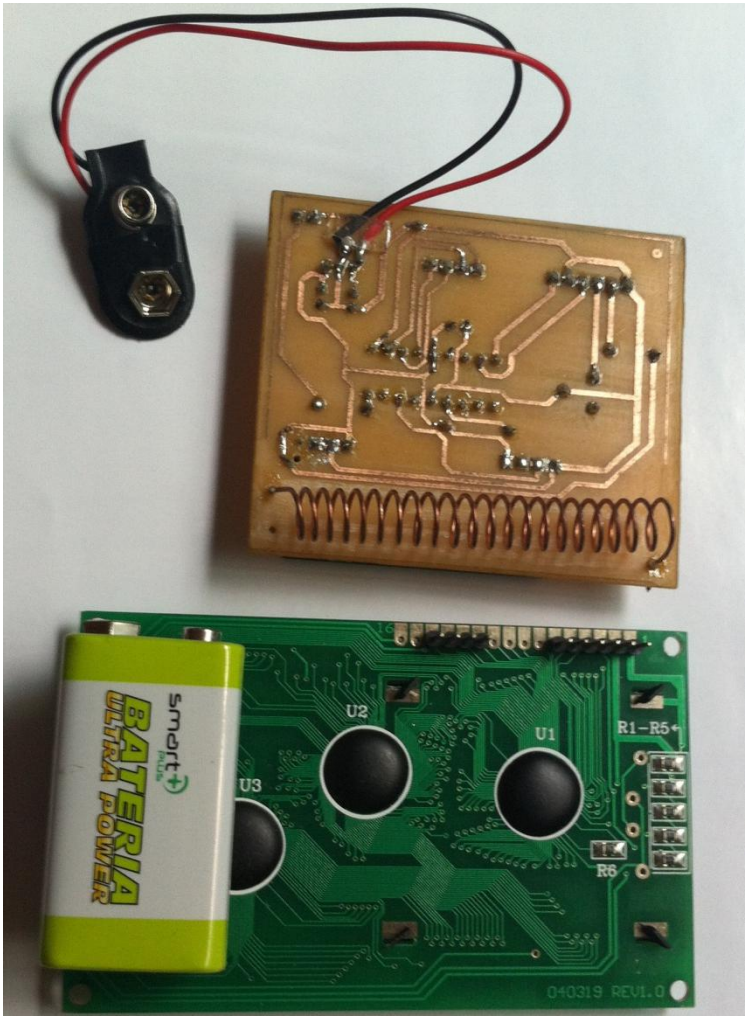

Figura 4.19 - Parte traseira da placa, do LCD e bateria de 9V

(FONTE: Autor)

Figura 4.20 - Frontal da placa e LCD

(FONTE: Autor)

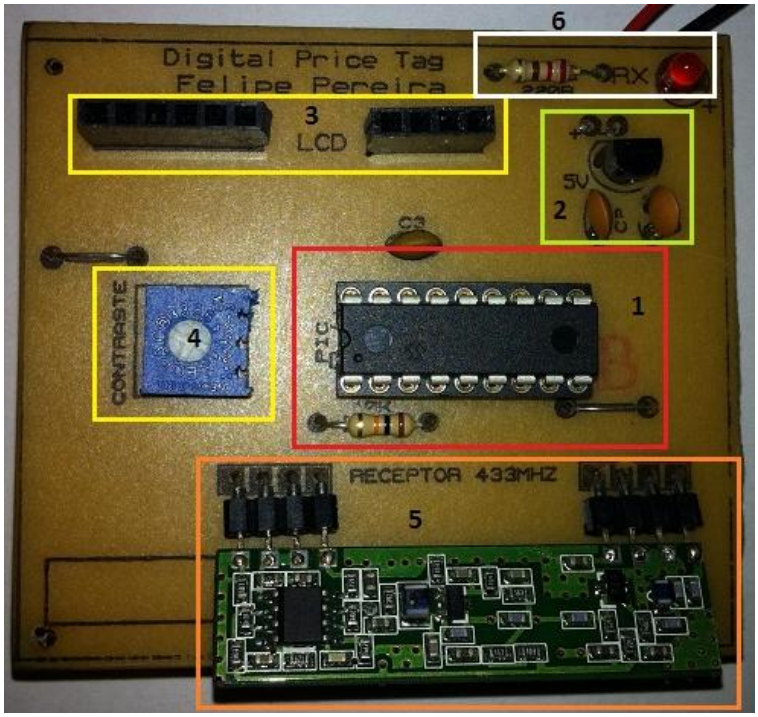

Figura 4.21 - Placa com circuitos enumerados

(FONTE: Autor)

Enumeração dos circuitos da etiqueta:

1. Microcontrolador PIC;
2. Circuito regulador de tensão;
3. *Socket* do display;
4. Potenciômetro para ajuste de contraste;
5. Receptor RF RXD1;
6. LED indicador de recepção de dados.

Figura 4.22 – Digital Price Tag

(FONTE: Autor)

4.3 – *Softwares*

Dois *softwares* foram implementados, um aplicativo *desktop* utilizado para administração da solução e o *firmware* dos microcontroladores.

O aplicativo *desktop* refere-se a interface gráfica, ou GUI (*Graphical User Interface*), que permitiu facilitar e tornar mais prático o controle do sistema de etiquetas para o administrador.

Firmware é denominado como as instruções operacionais programadas diretamente no *hardware*. Neste caso refere-se ao *software* que foi gravado nos microcontroladores, onde são executados.

Para facilitar o entendimento, os *softwares* foram descritos de acordo com o fluxo do sistema: em primeiro lugar ocorre o processo de transmissão e em seguida o de recepção.

Os fluxogramas que ilustram cada um dos processos podem ser vistos respectivamente na Figura 4.23 e na Figura 4.24.

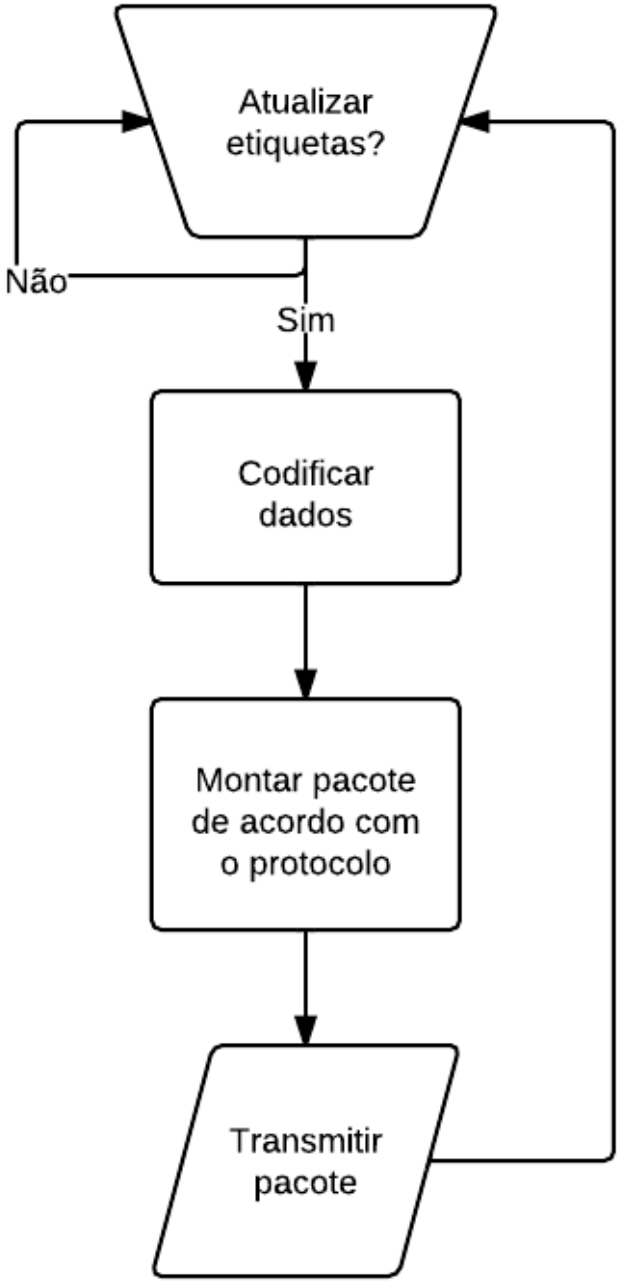

Figura 4.23 - Fluxograma da transmissão dos dados

(FONTE: Autor)

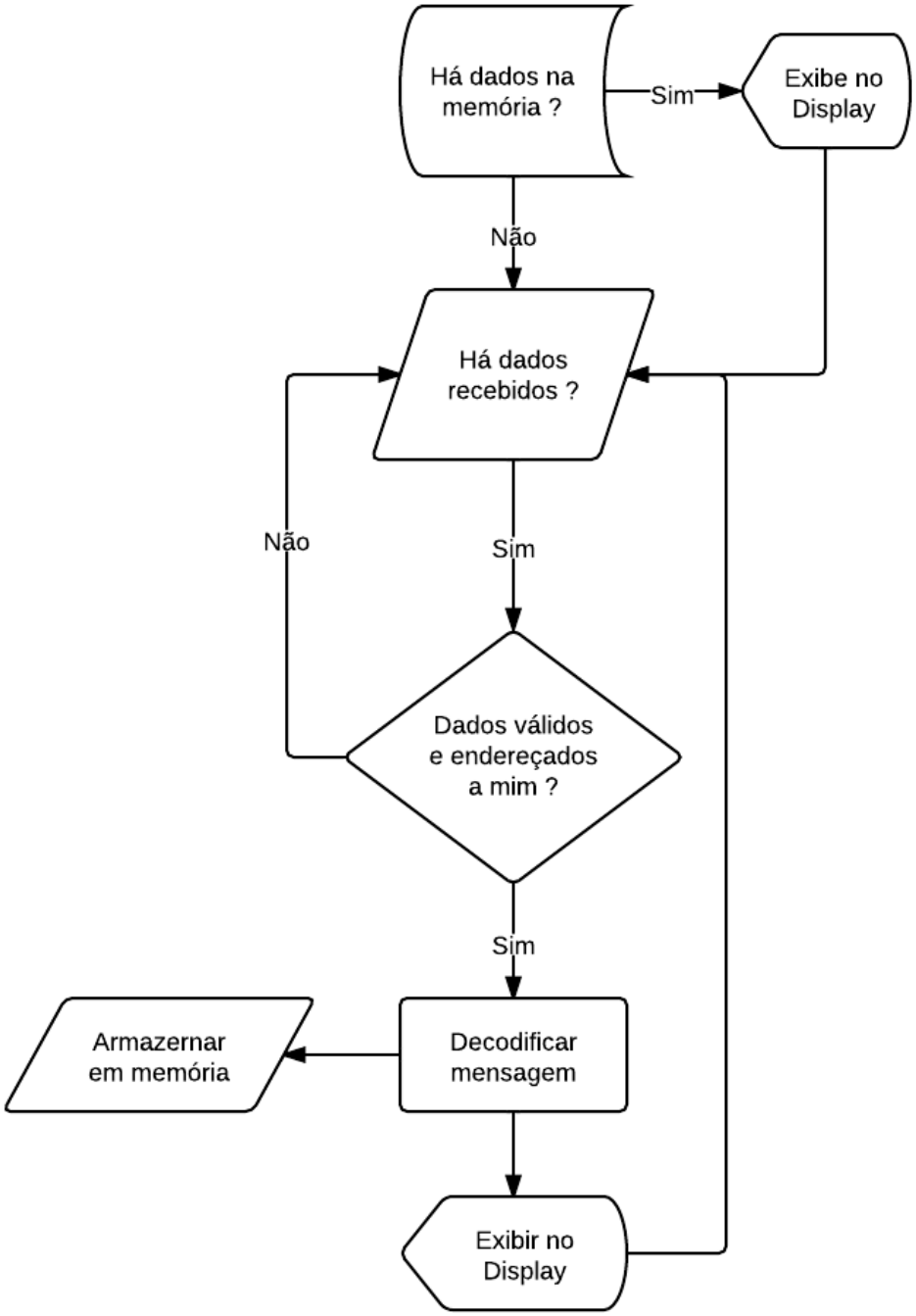

Figura 4.24 - Fluxograma do receptor

(FONTE: Autor)

4.3.1 – Interface Gráfica do Usuário

Para facilitar a utilização do sistema, preferiu-se o desenvolvimento de uma interface amigável para o controle dos dados que seriam enviados para as etiquetas eletrônicas.

A API gráfica *Swing* do Java, associada com a biblioteca de comunicação RXTXComm e o conector para banco de dados MySQL® permitiu o desenvolvimento do aplicativo *desktop* DPT *Manager*, para integração com o DPT *Transmitter*.

A interface é composta por diversas classes diferentes, são elas: Ajuda, Autentica, Database, DigitalPriceTags, Serial, Sobre, serialSetup e tagCombo. Excepcionalmente, a classe Serial foi obtida da biblioteca RXTXComm e todas as demais foram escritas pelo autor.

A DigitalPriceTags é a classe principal do programa, responsável pela tela principal e coordenação das ações de transmissão dos dados. A classe Ajuda é responsável pela exibição de uma janela com informações ao usuário sobre os botões e funcionalidades do programa. Autentica é a classe que faz a autenticação do usuário quando o mesmo deseja alterar as informações da tabela de dados. Database é a classe que executa os comandos no banco de dados, como obtenção e modificação dos dados. A classe Serial trata da comunicação do computador com o transmissor por RS-232. Esta classe foi baseada na classe de exemplo disponível no fornecedor da biblioteca RXTXComm (RXTX, 2011) e se encontra anexada ao final deste trabalho no anexo A. Sobre é uma classe simples que exibe detalhes referentes a este projeto. A classe serialSetup é a janela de configuração da comunicação serial, onde é escolhida a porta na qual se conecta o transmissor e também a velocidade da transmissão. A tagCombo é a classe que permite a associação de uma etiqueta eletrônica a um produto. As classes desenvolvidas neste projeto estão localizadas no apêndice A.

Como observação pode-se citar que para o correto funcionamento do projeto desenvolvido, o transmissor deve estar conectado ao computador onde o programa está sendo executado, através de uma porta serial ou de um conversor de USB para serial.

A interface, ao ser executada exibe ao usuário uma janela com sete botões, uma tabela de dados e uma área de *log* para acompanhamento das ações executadas. Cada botão possui uma ação específica e quatro deles são apresentados ao usuário através de ícones, e de imagens que referem-se às suas ações. Os botões, conforme ilustrado na Figura 4.25, possuem as ações que são, respectivamente: conectar ou desconectar o transmissor, recarregar a tabela de dados, liberar ou bloquear a edição dos dados da tabela e o painel de ajuda.

Figura 4.25 - Ícones dos botões da interface

(FONTE: Autor)

Há também outros três botões na interface: o botão “Salvar”, que permite que alterações efetuadas na tabela sejam gravadas no banco de dados; o “Sincronizar”, que envia todos os dados da tabela para o transmissor e o “Fechar”, que encerra a execução do programa.

Ao ser iniciado, por meio da classe Database, o programa consulta uma tabela de dados contendo a descrição dos produtos como nome, marca e valor e também o endereço da etiqueta eletrônica que está associada ao produto. Os dados consultados são exibidos na interface pelo componente Java denominado *JTable*. Tal componente exibe uma simples tabela de dados compostas pelas linhas e colunas obtidos do banco de dados. Este componente foi alterado para não permitir a edição dos dados. A edição da tabela somente é permitida por meio de uma senha, que foi implementada por meio da classe Autentica.

O *log* da aplicação é uma área de texto formada pelo componente *JTextArea*, configurado para exibir somente dados escritos pelo programa, não sendo possível editar dados neste componente.

4.3.1.1 – Ações dos Botões do Programa

Internamente no Java, existem interfaces de escuta chamadas *Listeners*. Estas interfaces são protótipos de funções que, quando utilizadas, monitoram a execução dos componentes. Cada botão da interface gráfica foi programado com um *Listener* chamado *ActionListener*, que monitora quando uma ação é executada no botão, como por exemplo, o clique do *mouse* e, a partir desta ocorrência, executa a função destinada àquele botão.

O botão conectar, quando clicado, faz uso da classe serialSetup e exibe uma janela de configuração da porta serial. Esta nova janela exibe as opções de portas seriais disponíveis e as velocidades de transmissão em dois componentes Java denominados *JComboBox*. Além disso, dois botões dão continuidade a execução da janela: o “Cancelar”, que interrompe a ação de conectar ao transmissor e o “OK”, que permite que a conexão serial seja estabelecida a partir das opções selecionadas nos combos. A Figura 4.26 ilustra a janela de configuração da porta serial.

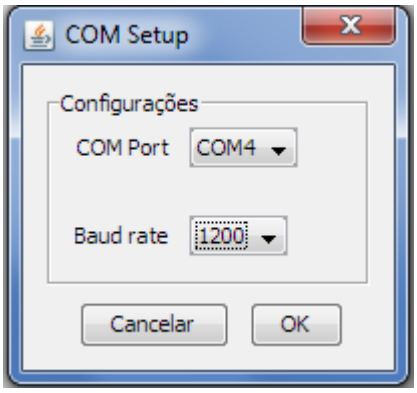

Figura 4.26 - Janela de configuração da porta serial

(FONTE: Autor)

O botão de recarregar a tabela executa uma função programada para limpar todos os dados da atual tabela. Além disso, obter os dados novamente no banco de dados e inseri-los atualizados no *JTable*.

O botão de autorizar a edição da tabela cria uma instância da classe Autentica onde uma nova janela é aberta para o usuário digitar uma senha. O usuário pode proceder de duas formas: na primeira informando a senha e clicando no botão "OK", momento em que a senha é verificada e, caso seja a mesma senha cadastrada, permite a edição da tabela. A segunda, clicando no botão "Cancelar", ou seja, desistindo assim da ação.

A Figura 4.27 ilustra a janela de autenticação do programa.

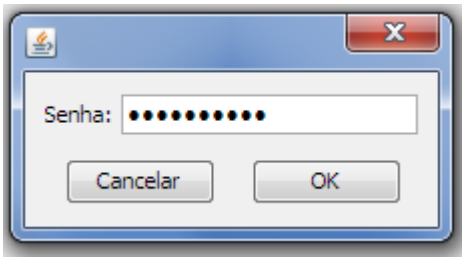

Figura 4.27 - Janela de autenticação

(FONTE: Autor)

O botão de ajuda ao ser pressionado, abre uma nova janela onde são exibidas informações ao usuário, para auxiliá-lo no uso do *software*. Nesta janela há também o botão "Sobre este programa", que exibe a descrição do *software* desenvolvido. As janelas são ilustradas respectivamente na Figura 4.28 e na Figura 4.29.

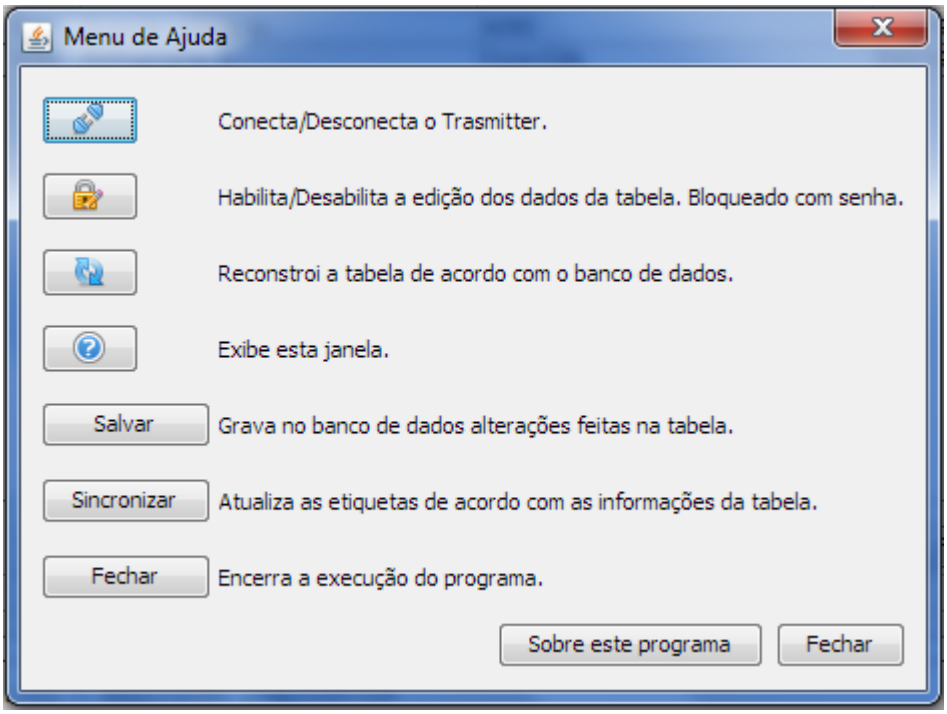

Figura 4.28 - Menu de ajuda

(FONTE: Autor)

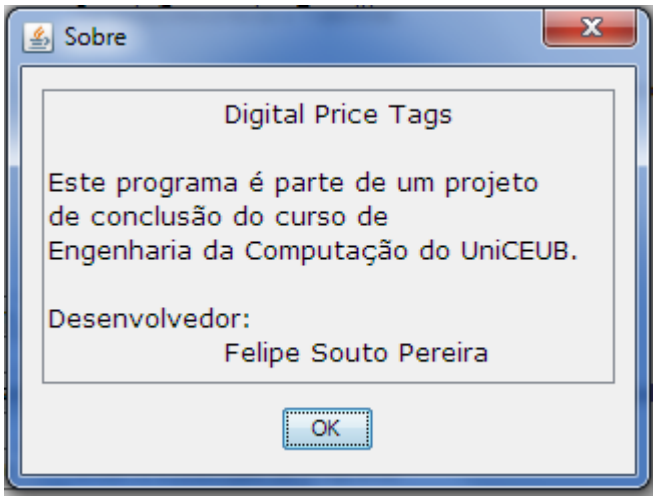

Figura 4.29 - Janela Sobre

(FONTE: Autor)

O botão “Salvar” é desabilitado por padrão. No caso de alguma alteração nos dados da tabela, automaticamente o botão é habilitado. Quando pressionado, sua ação é executar uma *query* no banco de dados para salvar as alterações feitas na tabela. Após os dados serem gravados com sucesso, o botão é desabilitado novamente.

Por fim, o botão “Sincronizar” é o responsável pela comunicação do computador com o transmissor. Quando solicitado, este botão permite duas ações. A primeira, caso a comunicação serial ainda não esteja estabelecida, a janela de configuração da porta serial é exibida ao usuário. A segunda acontece com a conexão já estabelecida. Deste modo, o programa lê cada linha da tabela e repassa os dados ao transmissor.

No início da transmissão, uma barra de progressão é exibida ao lado do botão “Salvar” informando o progresso do envio dos dados da tabela para o transmissor. Ao completar o envio de uma linha, é exibido no *log* a confirmação de sincronização da etiqueta. No fim da transmissão dos dados, a barra de progressão é novamente ocultada na interface gráfica.

A Figura 4.30 ilustra a tela principal do programa.

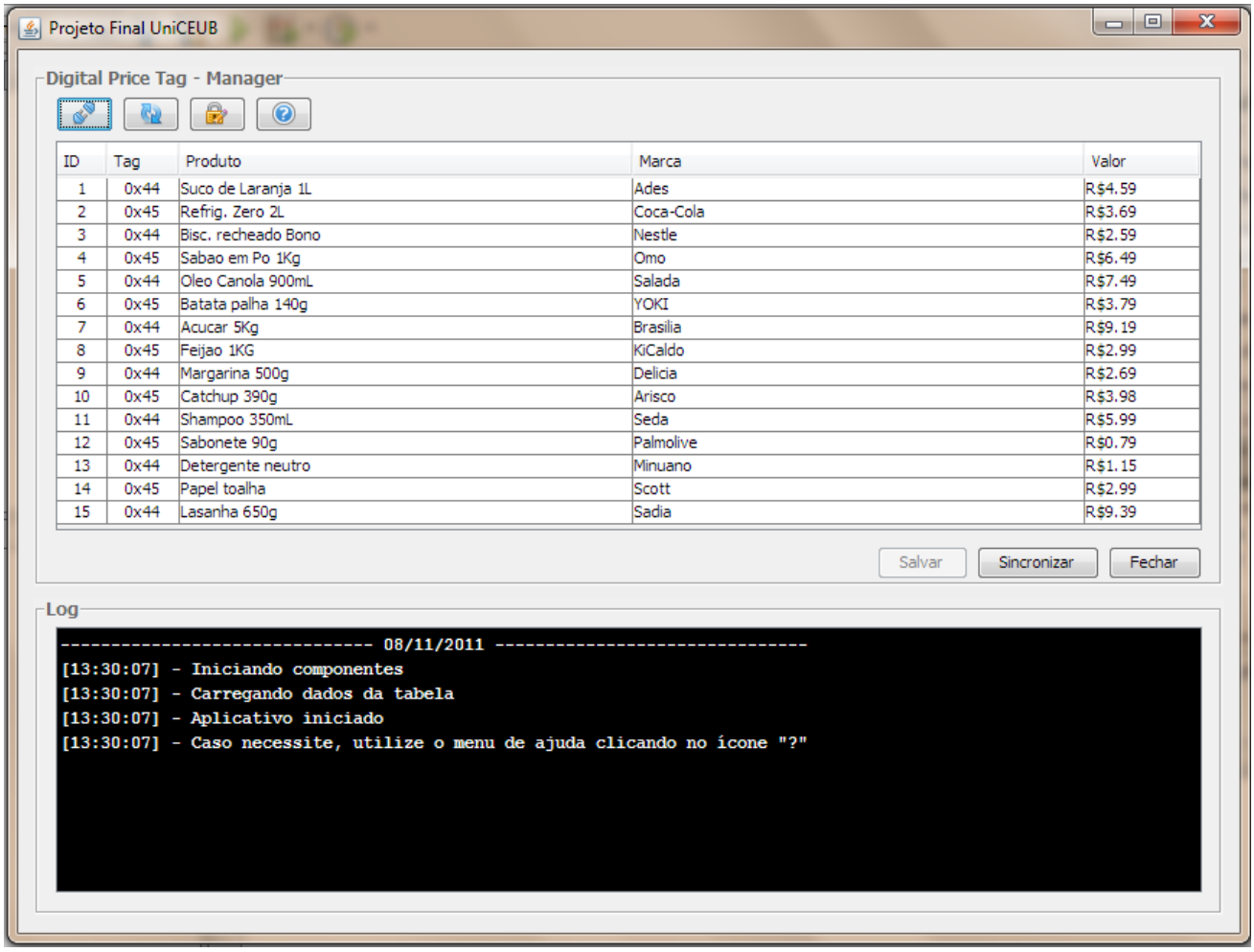

Figura 4.30 - Tela principal do programa

(FONTE: Autor)

4.3.1.2 – Endereçamento das etiquetas

A forma escolhida para endereçar as etiquetas eletrônicas foi atribuir um identificador formado por 8 *bits* em cada um dos receptores. Neste formato, até 256 (2^8) dispositivos poderiam ser endereçados. No programa, esse *byte* identificador foi representado no formato hexadecimal e foi referenciado na coluna "Tag" da tabela. Para facilitar a associação das etiquetas aos produtos, uma interface de edição foi implementada.

Ao clicar em um endereço da coluna "Tag", uma nova janela é exibida ao usuário, instanciada da classe tagCombo que, ao ser aberta, exibe três componentes Java: O *JTextField*, que exibe o endereço da *tag* clicada para ser modificada, o *JComboBox*, que exibe uma lista de *tags* disponíveis e, além destes, três botões (*JButton*), que encerram a execução da janela. São eles: o "Remover", que remove a associação da *tag* ao produto, o "Cancelar" que encerra a janela sem efetuar mudanças e o "OK", que é habilitado somente se algum item for escolhido na lista e aplica-se à mudança na tabela. A janela de edição das *tags* é ilustrada conforme Figura 4.31.

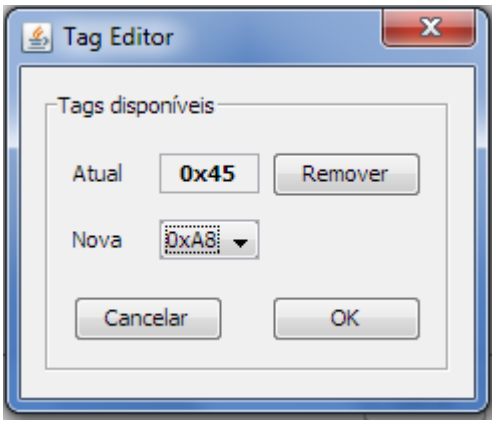

Figura 4.31 - Editor de endereços das etiquetas

(FONTE: Autor)

4.3.2 – *Firmwares* dos Microcontroladores

A principal função dos microcontroladores é aguardar a chegada de dados para realizar algum procedimento específico. O microcontrolador do transmissor aguarda dados vindos do computador para repassá-los às etiquetas eletrônicas, enquanto o microcontroladores das etiquetas aguardam dados enviados via rádio para exibi-los no LCD.

Nos dois casos foram inseridos *bytes* de controle durante a transmissão para indicar início e fim da transmissão.

Para obter a transmissão sem fios com qualidade, além de *bytes* de controle, foi necessário implementar algoritmos para codificar os dados e utilizar um protocolo de comunicação com detecção de erros. Codificar os dados e utilizar o protocolo foi de fundamental importância para o projeto, já que estes procedimentos viabilizaram a comunicação sem fios e diminuíram a ocorrência de erros e de interferências durante a transmissão.

4.3.2.1 – Codificação Manchester

Toda transmissão de dados é dada por meio de pulsos elétricos. O conceito básico desses pulsos é que o *bit* 1 representa ligado e o *bit* 0 representa desligado.

Ocorre que, segundo Tanenbaum (1997, p. 319), "se uma estação enviar o *string* de *bits* 0001000, outras poderão erroneamente interpretá-lo como 1000000 ou 01000000, pois não conseguem identificar a diferença entre um transmissor inativo (0 volt) e um *bit* 0 (0 volt)". Pela falta de relógio externo para a sincronização das estações, é necessário haver uma maneira de os receptores determinarem exatamente o início, o fim ou o meio de cada *bit*. Denominado de codificação *Manchester*, os *bits* são determinados por uma transição na parte intermediária, tornando fácil para o receptor sincronizar-se com o transmissor. Um *bit* 1 binário é representado pela transição de estado 1 para 0. No 0 binário o inverso acontece, a representação é dada pela transição de 0 para 1.

A Figura 4.32 ilustra dois tipos de codificação para o envio dos bits "10000101111".

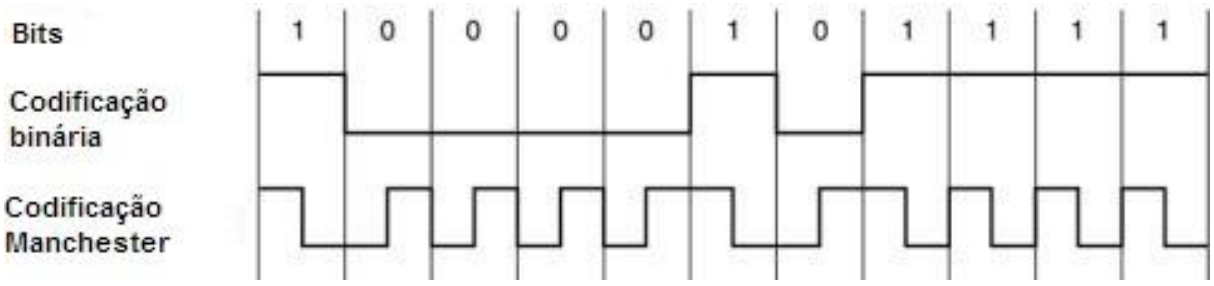

Figura 4.32 – Codificação binária e representação em *Manchester*, com alterações

(FONTE: Tanenbaum, 1997, p. 320)

O trecho do código dos *firmwares* onde os dados são codificados e decodificados em *Manchester* tiveram como base o artigo *Manchester Encoding Using RS232*, publicado em 2005, e está localizado ao final deste trabalho no anexo B.

Além da codificação dos dados, no projeto foi necessário implementar uma solução via *software*, para a utilização dos módulos de recepção de radiofrequência, devido a vulnerabilidade a interferências e ruídos que esses módulos podem captar.

Desta forma, antes de cada transmissão, foi necessário enviar *bytes* de sincronismo para estabelecer um canal entre transmissor e receptor. O envio desses *bytes* permitiram aos módulos receptores e as etiquetas eletrônicas executarem três funções:

- Inicializar o demodulador do módulo receptor;
- Habilitar o decodificador RS-232 a sincronizar o *Start bit* de cada *byte* RS-232;
- Indicar ao receptor (etiqueta eletrônica) que foi feito o sincronismo e os dados válidos serão enviados.

Designado como LAM (*Look At Me*) ou preâmbulo de sincronização, o sincronismo dos módulos é feito enviando uma sequência de 1's e 0's ou vice-versa. Neste projeto foi utilizado o *byte* 0xF0 (HEX) que é representado por 11110000 em binário. Cinco destes *bytes* enviados a cada 250µS foram suficientes para inicializar a comunicação entre os módulos.

4.3.2.2 – *X-Protocol* e Detecção de Erros CRC

Para a comunicação do transmissor com as etiquetas eletrônicas, um conjunto de regras e procedimentos foram implementados para enviar e receber os dados de forma correta.

O protocolo implementado teve como base um exemplo disponível no IDE CCS Compiler que foi alterado para se adequar a este projeto. O código-fonte exemplo foi anexado ao final deste trabalho no anexo C.

Para a correta comunicação entre o transmissor e as etiquetas eletrônicas, a mensagem transmitida necessitou seguir o padrão ilustrado na Tabela 4.1.

Tabela 4.1 - *X-Protocol*, Protocolo de comunicação

Início (START)	ID Emissor (ID.TX)	ID Receptor (ID.RX)	Tamanho (LENGTH)	Dados Manchester (DATA)	Detecção de Erros (CRC)	Fim da Mensagem (STOP)
1 *Byte*	1 *Byte*	1 *Byte*	2 *Bytes*	0 – 56 *Bytes*	2 *Bytes*	1 *Byte*

(FONTE: Autor)

Neste formato, o tamanho máximo que o pacote pode ter são 64 *bytes*. Esta limitação se deve a limitações físicas da memória do microcontrolador.

Os três primeiros *bytes* são para controle e são verificados a cada transmissão. Se alguma das verificações falhar, o pacote é descartado e o processo de verificar os dados é reiniciado.

Ao receber os *bytes* de controle corretamente, os dados seguintes são armazenados em um *buffer* até a detecção do *byte* de fim da mensagem. Com isso, os dados são lidos do *buffer* e é feita a verificação do tamanho da mensagem, para determinar a localização dos *bytes* de detecção de erro ou CRC.

Baseado em Tanenbaum (1997), o CRC (*Cyclic Redundancy Check*) é um código detector de erros que é gerado pelo transmissor a partir da mensagem que se deseja transmitir. O algoritmo do CRC realiza um cálculo de divisão polinomial com os dados que serão transmitidos e um polinômio gerador escolhido. Na divisão dos polinômios é obtido o resto desta divisão. O resto corresponde ao código CRC que é concatenado ao final da mensagem que se deseja transmitir. Estes códigos polinomiais se baseiam no tratamento das mensagens como coeficientes binários de um polinômio variável X qualquer. Quando o método de código polinomial é utilizado, o transmissor e o receptor devem concordar em relação ao polinômio gerador.

Segundo Tanenbaum (1997, p. 219), os seguintes polinômios se tornaram padrões internacionais:

$$\text{CRC-12} = x^{12} + x^{11} + x^3 + x^2 + x^1 + 1 \quad (1)$$

$$\text{CRC-16} = x^{16} + x^{15} + x^2 + 1 \quad (2)$$

$$\text{CRC-CCITT} = x^{16} + x^{12} + x^5 + 1 \quad (3)$$

No projeto, o polinômio gerador foi o CRC-CCITT de 16 *bits*.

Quando os receptores, neste caso as etiquetas eletrônicas, recebem a mensagem e obtém o código CRC, o mesmo cálculo é feito. Se o código gerado pelo receptor for idêntico ao código que foi recebido pelo transmissor, a transmissão ocorreu com sucesso. Desta forma, o microcontrolador procede com a decodificação da mensagem, armazenamento em memória não volátil e exibição no LCD. Caso contrário, a mensagem é descartada e o processo de verificação de dados reiniciado.

O código CRC foi utilizado neste projeto por ser um método de detecção de erros muito eficiente e devido aos receptores captarem interferências de outras fontes, o CRC foi fundamental para que as etiquetas só recebessem os dados corretos. A Tabela 4.2 ilustra a

eficiência do código CRC de 16 *bits* e a Figura 4.33 demonstra o cálculo de divisão polinomial da mensagem binária “1010001101” pelo polinômio gerador “110101”.

Tabela 4.2 - Eficiência do CRC de 16 *bits*

Tipos de Erros	**Eficiência**
Erros de um único *bit*	100%
Erros de 2 *bits*	100%
Erros com número impar de *bits*	100%
Erros de estouro < 16 *bits*	100%
Erros de estouro = 17 *bits*	99,9969%
Outros erros de estouro	99,9984%

(FONTE: TANEMBAUM. 1997)

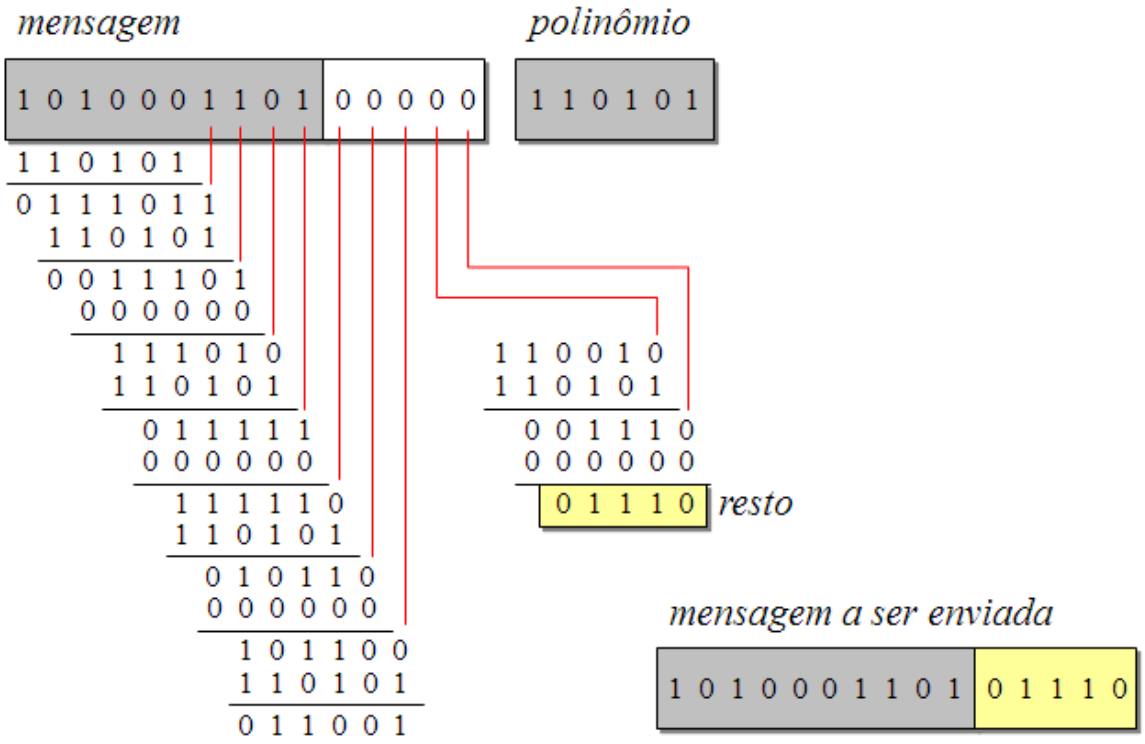

Figura 4.33 - Divisão polinomial do código CRC

(FONTE: TANEMBAUM. 1997)

4.3.2.3 – Funções do Código Fonte

Várias funções foram escritas para serem executadas pelos microcontroladores. Todo o código fonte está dividido em funções que são chamadas quando solicitadas pelo programa principal ou por algum evento externo, como o recebimento de dados.

Dentre as funções do transmissor estão: rda_isr, send_packet e main (programa principal). Já as etiquetas eletrônicas executam as seguintes funções: isr_rda, output_toggle, Decode_Data, get_packet, PrintLCD e também a função main.

As funções rda_isr e isr_rda são as duas funções que são chamadas no transmissor e receptor, respectivamente, quando há o recebimento de dados. O microcontrolador, ao identificar que há dados disponíveis, gera uma interrupção para execução dessas funções. No transmissor o seu papel é armazenar os dados em um *buffer* e avisar ao programa principal sobre o recebimento dos dados através da variável booleana "flag". Nas etiquetas eletrônicas, essa função verifica se os *bytes* de controle estão corretos e na ordem certa e, em caso positivo, os dados seguintes são armazenados em um *buffer* e a mesma variável *flag* é setada para avisar ao programa principal sobre os dados. No caso de recebimento incorreto dos *bytes* de controle, os dados são descartados e a função é reiniciada.

A função send_packet é executada pelo transmissor quando se deseja enviar dados às etiquetas eletrônicas. Esta função recebe como parâmetros o ponteiro da mensagem a ser enviada e o tamanho da mensagem. Em seguida o pino de alimentação do módulo transmissor TXC1 é acionado e a transmissão dos dados é iniciada. A função também é responsável pelo cálculo CRC que é transmitido antes do *byte* STOP. Terminada a transmissão, a função finaliza sua execução desligando o pino onde está ligado o módulo TXC1.

No programa principal do transmissor são realizadas as configurações básicas para o funcionamento do microcontrolador, como habilitar as interrupções e configurar a porta RS-232. Esta função também executa um *loop* para manter o microcontrolador em execução contínua. Neste *loop* são verificados se há dados a serem transmitidos para as etiquetas e, se houver, esses dados são armazenados em um *buffer*, que é o protocolo de transmissão. Os dados são inseridos na sequência que foi detalhada no item 4.3.2.2, se iniciando com a inserção do *byte* START e encerrando com a inserção do *byte* STOP. Durante o preenchimento do *buffer*, o trecho de código responsável pela codificação *Manchester* é executado, inserindo no *buffer* a mensagem codificada. Ao final do preenchimento do *buffer*, o programa principal faz a chamada da função send_packet e procede com a transmissão dos dados. Logo após este procedimento, o *loop* é reiniciado.

As etiquetas eletrônicas, além da função isr_rda já descrita, executam a função output_toggle para sinalizar através de um LED o recebimento de dados.

A função Decode_Data ao ser executada, recebe como parâmetro um *byte* codificado em *Manchester* e retorna o *byte* decodificado para a função que a chamou.

A função get_packet é chamada através da função principal quando houver sinalização de que dados foram recebidos pela porta UART. Esta função lê o *buffer* de dados recebidos e faz a validação dos *bytes* de controle, a decodificação da mensagem e a verificação do código CRC.

A função PrintLCD é executada para exibir os dados recebidos no visor de cristal líquido. Os dados são lidos da memória EEPROM e transcritos no visor.

O programa principal, assim como no transmissor, realiza as primeiras configurações do microcontrolador e executa um *loop*. Durante a execução do *loop* são verificadas algumas variáveis para a chamada das demais funções. A variável *flag*, quando setada pela função isr_rda permite ao programa principal obter os dados recebidos e decodificá-los. Os dados decodificados são armazenados na memória EEPROM e exibidos no visor.

CAPÍTULO 5 – Testes, Simulação e Dificuldades

Após a implementação do projeto foi necessária a realização de procedimentos para verificar a sua funcionalidade. Foram realizados diversos testes com as etiquetas eletrônicas para verificação de distância de funcionamento, velocidades de transmissão e imunidade a interferências.

Além destes testes, uma simulação de uso das etiquetas foi realizada, utilizando uma tabela de dados de diversos produtos vendidos no comércio.

Durante a implementação e realização dos testes surgiram dificuldades que alteraram o fluxo do trabalho induzindo a novos procedimentos para resolução dos problemas encontrados.

5.1 – Testes

Dentre os testes realizados após a implementação do protótipo, o primeiro foi a verificação da distância de funcionamento, analisando vários tipos de antenas para a comunicação sem fios. Foram realizados em ambiente fechado e com obstáculos para obter a melhor performance e qualidade das antenas.

Além da antena já citada anteriormente que apresentou baixa eficiência, outros dois modelos de antenas foram desenvolvidos e testados. A antena de $^1/_4$ de onda e a antena helicoidal.

De acordo com Tanenbaum (1997, p. 107), "no vácuo, todas as ondas eletromagnéticas viajam na mesma velocidade, independente de sua frequência. Essa velocidade, geralmente chamada de velocidade da luz, c, é cerca de 3 x 10^8 m/s. ou, aproximadamente de 30 cm por nanossegundo". A relação fundamental entre a frequência (f), comprimento de onda (λ) e velocidade da luz no vácuo (c) é:

$$\lambda f = c \tag{4}$$

Para definir o tamanho da antena de $^1/_4$ de onda, foi necessário calcular o comprimento de onda do módulo utilizado no projeto, através da fórmula:

$$\lambda = \frac{c}{f} \quad (5)$$

Na equação 5, c é a velocidade da luz no vácuo (3x10^8 m/s) e f é a frequência da onda em Hertz, que neste caso é 433.92MHz. Com este cálculo foi possível definir o comprimento de onda de $\lambda = 0.69137$m e $\lambda/4$ do comprimento é equivalente a 0.17284m ou 17.2cm.

Para a confecção da antena foi utilizado um fio de cobre rígido de 26 AWG com o tamanho definido pelo cálculo, que é de 17.2 centímetros.

A partir dos testes realizados com esta antena foi constatada sua eficiência, pois foram atingidas distâncias superiores a 20 metros do transmissor. A desvantagem deste tipo de antena foi o seu tamanho, que não atendeu às expectativas do projeto, por ser um protótipo compacto.

Como alternativa de resolução do problema relativo ao tamanho da antena, outro modelo foi desenvolvido para atender o projeto. A antena helicoidal é um modelo de antena com formato de mola, com o comprimento duas a três vezes superior à antena de $1/4$ de onda.

A confecção desta antena também foi realizada utilizando fio de cobre de 26 AWG que foi enrolado em um cilindro metálico de 7 milímetros. A sintonização deste tipo de antena é definida pelo espaçamento entre as voltas e o comprimento da antena. Devido às falhas do primeiro modelo helicoidal desenvolvido, essa sintonia foi feita por aproximação, devido a falta de ferramentas e aparelhos disponíveis para este propósito.

A configuração final da antena foi de 3 mm de espaçamento entre as voltas e 6 cm de comprimento. Esta configuração permitiu alcance superior a 25 metros. Além disso, a antena ficou compacta o bastante para ser fixada atrás da placa de circuito impresso e foi a escolhida para o protótipo.

Mesmo atendendo as expectativas do projeto, a antena helicoidal possui algumas desvantagens. Objetos próximos a antena podem facilmente alterar a sintonização e prejudicar o seu funcionamento, diminuindo assim o alcance.

Outro teste realizado nas etiquetas eletrônicas foi relacionado a velocidade da transmissão. Os módulos utilizados no projeto não possuíam especificação da velocidade de transmissão em seu *datasheet*, fazendo com que fossem testadas as velocidades usuais encontradas na comunidade *Web* de módulos parecidos com os utilizados no projeto. As velocidades testadas foram 2400bps e 1200bps. Na velocidade de 2400bps a transmissão ocorreu com falhas e os dados exibidos no leitor continham caracteres inválidos, necessitando diminuir esta velocidade. A velocidade inferior próxima a 2400bps foi a de 1200bps. Com esta velocidade a comunicação entre o transmissor e as etiquetas ocorreu sem falhas.

Por fim, o protótipo foi testado em um ambiente com várias outras fontes de ondas eletromagnéticas próximas as etiquetas eletrônicas, como antenas WI-FI, rádios, televisores e outros. Em todos os testes, dentro do raio de funcionamento das etiquetas, todas receberam os dados corretamente e sem interferências dos outros aparelhos.

5.2 – Simulação do Protótipo

Para realizar a simulação de funcionamento do protótipo foi utilizada uma base de dados de testes com uma lista de vários produtos distintos. O servidor do banco de dados é o MySQL® e possui apenas uma base com uma tabela.

A tabela foi composta por quinze produtos e foi dividida em cinco colunas: ID, Tag, Marca, Produto e Valor.

A coluna ID é o identificador do número da linha. A Tag é o campo onde permanece armazenado o endereço da etiqueta eletrônica associada àquele produto. As colunas Marca, Produto e Valor são destinadas para a descrição dos produtos.

O banco de dados foi utilizado pela interface gráfica desenvolvida. Na inicialização da interface foi feita uma consulta na tabela e os dados foram exibidos na interface gráfica. A simulação foi feita alternando o endereço das etiquetas eletrônicas a cada produto.

Nesta simulação também foi possível definir o tempo decorrido para a sincronização das etiquetas. O menor tempo conseguido sem que houvesse perda de dados foi sincronizando um produto entre 3~4 segundos. Durante a simulação dos quinze produtos da tabela, o menor tempo atingido foi de 55 segundos. Seguindo esta média, no período de uma hora é possível sincronizar até 981 produtos distintos.

5.3 – Dificuldades do Projeto

Durante a implementação do projeto surgiram várias dificuldades que fizeram com que o fluxo do trabalho fosse alterado, no entanto, as dificuldades foram contornadas e resolvidas de forma simples e rápida.

A comunicação sem fios foi a maior dificuldade encontrada durante o desenvolvimento do projeto. Também surgiram limitações referentes ao microcontrolador utilizado e a comunicação do computador com o transmissor.

Para o correto funcionamento da comunicação sem fios, foi inevitável a utilização de um protocolo de comunicação e a codificação *Manchester*.

Os módulos receptores utilizados nas etiquetas eletrônicas, quando ligados, captam muitas interferências de outros dispositivos transmissores de ondas eletromagnéticas. Sem um protocolo para validar esses dados, o funcionamento do microcontrolador foi prejudicado devido a grande quantidade de interferências recebidas.

O protocolo permitiu que o microcontrolador validasse os primeiros dados recebidos pelo módulo receptor e, caso não correspondessem a dados válidos, estes seriam descartados, livrando o microcontrolador de processamento desnecessário.

A codificação *Manchester* também foi fundamental para a transmissão de vários dados sequenciais pelo transmissor. A característica básica desta codificação é identificar os *bits* por sua transição de 0 para 1 ou vice-versa. Quando vários dados são transmitidos, essa transição facilita ao módulo receptor a interpretação do *bit* recebido. Se uma sequência de 1's ou 0's for recebida, alguns *bits* podem ser perdidos e a transmissão ficar comprometida. Assim, mantendo a transição dos *bits*, este problema não ocorre.

Como consequência da utilização da codificação *Manchester*, foi necessário lidar com a limitação de memória do microcontrolador.

Um dado de 8 *bits* de tamanho, ao ser codificado em *Manchester*, necessita de 16 *bits* de espaço em memória para armazenamento. Deste modo, a transmissão de uma mensagem de tamanho N, ao ser codificada, requer um espaço de memória de 2N.

Para resolver esta questão, foi necessário limitar o tamanho máximo para transmissão e dividir a transmissão dos dados em pacotes. No projeto, o envio da descrição do produto foi realizada em três partes. Em primeiro lugar foi enviado o nome do produto, em seguida a marca e por último o valor. Na transmissão do último pacote, um *byte* é enviado para sinalizar o fim da mensagem, e o receptor pode proceder com a decodificação e exibição dos dados no LCD.

Ao implementar a solução de dividir a mensagem enviada, foi detectado que o computador enviava os dados mais rápido do que o microcontrolador poderia processar e repassar por radiofrequência. O problema foi contornado adicionando um tempo de espera a cada transmissão, no *software* de controle. A interface de controle, ao enviar cada pacote ao microcontrolador, executa uma função para interromper a execução do programa por um período de 1200 milissegundos. Este tempo foi suficiente para que o microcontrolador executasse todas as funções e retornasse ao seu modo de espera.

CAPÍTULO 6 – CONSIDERAÇÕES FINAIS

6.1 – Conclusões

Este projeto introduziu uma solução de etiquetas de preço digitais que se propõe a melhoria do sistema de etiquetagem encontrado no comércio, através de tecnologias específicas para controle e gerenciamento dos preços dos produtos. A solução desenvolvida foi capaz de substituir satisfatoriamente as etiquetas convencionais de papel, contribuindo para a melhoria da qualidade do serviço, reduzindo desperdícios, tais como, de papel, mão de obra e outros. É importante ressaltar que as etiquetas eletrônicas são reaproveitáveis. Todo conteúdo exibido nas etiquetas pode ser editado e os dispositivos serem movidos de um local para outro, já que funcionam através de tecnologias de comunicação sem fio. O sistema de etiquetas eletrônicas também contribuiu pelo modo como são gerenciadas, facilitando as alterações de preços e também as alterações na descrição dos produtos.

Os objetivos propostos neste trabalho foram alcançados, na medida em que forneceram um protótipo funcional da solução, composto por duas etiquetas eletrônicas, um transmissor e a interface de gerenciamento. Os resultados obtidos através da simulação da solução mostraram relativa eficiência do sistema para o uso no comércio, dado que a etiqueta possui boa legibilidade e ainda é compacta, podendo facilmente ser acoplada em gôndolas.

Nas pesquisas realizadas durante o desenvolvimento do trabalho não foi possível obter a relação custo/benefício da solução, devido a fatores envolvidos para produção em grande escala. Vários fatores teriam que ser considerados, tais como a quantidade de itens vendidos, área do estabelecimento, dentre outros. No entanto, por se tratar de um projeto acadêmico, estas considerações fogem ao escopo do projeto.

6.2 – Sugestões para Trabalhos Futuros

Para o desenvolvimento de projetos futuros sugere-se o aprimoramento da solução apresentada neste trabalho tais como, a utilização de outras tecnologias de transmissão sem fios, melhorias nos modos de exibição dos dados e melhorias na capacidade do sistema.

Este trabalho utilizou a tecnologia de transmissão via radiofrequência para a comunicação sem fios, através de módulos vendidos no comércio. O uso de transmissões por

radiofrequência deve ser regulamentado por agências reguladoras, que no caso do Brasil é a Agência Nacional de Telecomunicações (ANATEL), devido às interferências que esse tipo de comunicação pode gerar em outras transmissões de dados. A implementação de outro tipo de tecnologia, como a transmissão por raios infravermelhos, dispensa essa regulamentação, pois este tipo de tecnologia não gera interferências que prejudicam o funcionamento de outros dispositivos.

Como complemento à utilização de outra tecnologia de transmissão, pode-se escolher um meio que permita a comunicação bidirecional. Com isso, as etiquetas poderiam enviar dados ao sistema de controle com informações úteis ao seu funcionamento, como *status* da bateria, recebimento correto dos dados e até mesmo o furto da etiqueta.

O uso de outras formas de exibição dos dados é outro aprimoramento que pode ser incluído em novos projetos. Através do uso de visores gráficos podem ser exibidos detalhes mais precisos dos produtos anunciados, como a logomarca do produto, maior legibilidade das informações e melhor qualidade de resolução. Ainda podem ser incluídas ações para anunciar avisos ao consumidor como promoções e quantidade de itens em estoque.

O *hardware* escolhido para o desenvolvimento do protótipo deste trabalho implicou em limitações para a exibição das informações. Por ter pouca capacidade de armazenamento, a quantidade de caracteres exibida nos visores teve de ser limitada. O aumento dessa capacidade permite que mais informações possam ser recebidas pelas etiquetas e exibidas aos consumidores, acrescentando maior qualidade à solução.

Outra sugestão de melhoria do projeto é o estudo da relação custo/benefício da implementação da solução em larga escala. Através desse estudo seria possível definir se a solução apresentada é capaz de efetivamente substituir o tradicional sistema de etiquetas de papel comumente usado.

REFERÊNCIAS

AHO, A. V.[et al.]. **Compiladores : princípios, técnicas e ferramentas** / Alfred V. Aho...[et al.];Tradução Daniel Vieira; revisão técnica Mariza Bigonha. – 2ª edição – São Paulo: Pearson Addison-Wesley, 2008.

ALCÂNTARA Filho, Roberto. (2008). **Padrão Serial RS-232**. Disponível em: <http://www2.eletronica.org/artigos/eletronica-digital/padrao-serial-rs-232>. Acesso em: 12 out. 2011,

ALTIERRE DIGITAL RETAIL. (2008). ***Digital price tags save energy and costs***. Disponível em: <http://vator.tv/news/2008-12-08-digital-price-tags-saves-energy-and-costs>. Acesso em: 21 set. 2011.

BRAGA, Maria Luiza de Oliveira. **Janela automatizada para smart houses com sensor de chuva e aviso por SMS**. Projeto acadêmico em Engenharia da Computação do Centro Universitário de Brasília. 2010. Brasília – DF.

BRASIL, Lei Federal n° 10.962. (2004). **Dispõe sobre a oferta e as formas de afixação de preços de produtos e serviços para o consumidor.** Legislação Federal. Disponível em <www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/lei/l10.962.htm>. Acesso em 03 out. 2011.

DEITEL, H. M; DEITEL, P. J. **Java Como Programar –** 6 Ed. – Prentice Hall Brasil, 2006.

ENCICLOPÉDIA TEMÁTICA. (2011). **Conceito de Preço**. Disponível em: <http://www.knoow.net/cienceconempr/economia/preco.htm>. Acesso em 03 out. 2011.

JANDL Junior, Peter. Introdução ao Java / Peter Jandl Junior. – São Paulo : Berkeley Brasil, 2002.

JOSÉ, A. (2009). **Microcontroladores**. Disponível em: <http://lusorobotica.com/index.php?topic=1198.0>. Acesso em: 26 out. 2011.

KEYMARK TECHNOLOGY. ***Datasheet 315/434 MHz Hybrid Receiver***. Disponível em: <http://www.tato.ind.br/files/RxD1.pdf>. Acessado em 29 set. 2011.

KEYMARK TECHNOLOGY. ***Datasheet RF Transmitter Module for 315/433.92 MHz***. Disponível em: < http://www.tato.ind.br/files/TX-C1.pdf>. Acessado em 29 set. 2011.

MANKIW, N. G. **Introdução à economia**. Edição compacta / N. Gregory Mankiw; [tradução Allan Vidigal Hasting]; revisão técnica Carlos Roberto Martins Passos. – São Paulo: Pioneira Thomson Learning, 2005.

MICROCHIP TECHNOLOGY INC. (2007). ***PIC16F627A/628A/648A***. Datasheet: Flash-Based, 8-Bit CMOS Microcontrollers with nano Watt Technology. DS40044F. Disponível em: <http://ww1.microchip.com/downloads/en/devicedoc/40044f.pdf>. Acesso em: 12 de set. 2011.

MILLS, Adrian. (2005). ***Manchester Enconding Using RS232***. Disponível em: <http://www.fatiherdem.net/yuklenenler/Manchester_encoding_using_RS232.pdf >. Acesso em: 15 out. 2011.

NASCIMENTO, Juarez do. Telecomunicações – 2ª edição – São Paulo : Makron Books, 2000.

NATIONAL SEMICONDUCTOR CORPORATION. (2000**). *Datasheet LM78XX Series Voltage Regulators***. Data do documento: Maio de 2000. DS007746. Disponível em: <http://www.national.com/ds/LM/LM7512C.pdf>. Acesso em: 22 de out. 2011.

NOTA POSITIVA. (2011). **Preço de Mercado**. Disponível em: <http://www.notapositiva.com/dicionario_economia/precomercado.htm>. Acesso em 04 out. 2011.

ORACLE. (2011). ***Java Platform SE 6 Docs*.** Disponível em: <http://download.oracle.com/javase/6/docs/api/>. Acesso em: 20 set. 2011.

PEREIRA, F. **Microcontroladores PIC: Programação em C** / Fábio Pereira. – 7. Ed. – São Paulo: Érica, 2007.

PEREIRA, F. **Microcontroladores PIC: Técnicas Avançadas** / Fábio Pereira. – 3.Ed. – São Paulo: Érica, 2004.

PETERSEN, L. (2002). ***UART test program for 16F628***. Montagem do Circuito MAX232. Disponível em: <http://www.oz1bxm.dk/PIC/628uart_c.htm>. Acesso em: 12 out. 2011.

PRICER. (2011). ***Eletronic Shelf Labels***. Disponível em: <http://www.pricer.com>. Acesso em: 09 out. 2011.

PULSEENG. (2011). ***ISM 433MHz Helical Antenna***. Disponível em: <http://pulseeletronics.com/download/w3127.pdf>. Acessado em: 26 out. 2011.

ROBÓTICA SIMPLES. (2010). **Aula 03 - O Microcontrolador PIC 16F628A**. Disponível em: <http://www.roboticasimples.com/cursos.php?acao=15>. Acesso em: 18 set. 2011.

RR ETIQUETAS. **Etiquetas Eletrônicas**. Disponível em: <http://www.rretiquetas.com.br>. Acesso em: 06 out. 2011.

RXTX. (2011). ***Java library, using a native implementation, providing serial and parallel communication for the Java Development Toolkit (JDK).*** Disponível em: <http://rxtx.qbang.org>. Acesso em: 24 set. 2011.

SEAL TECNOLOGIA. (2011). **Etiquetas de prateleira**. Disponível em: <http://www.seal.com.br/solucoes/etiquetas_de_prateleira.html>. Acesso em: 09 out. 2011.

SOUZA, D. J. **Desbravando o PIC: Ampliado e Atualizado para PIC 16F628A** / David José de Souza. – 8. Ed. – São Paulo: Érica, 2005.

SYSTEM, M. I. (2007). ***RS232 Pinouts Designation***. Disponível em: <http://www.machine-informationsystems.com/RS232_Pinouts.html >. Acesso em: 21 out. 2011.

TANENBAUM, Andrew S., 1944 – **Redes de Computadores**. Rio de Janeiro. Andrew S. Editora Campus, 1997.

TAROUCO, Liane Margarida Rockenbach.
Redes de comunicação de dados / Liane Margarida Rockenbach Tarouco, – 3ª ed. – Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora S.A., 1985.

TEXAS INSTRUMENTS. (2004). ***MAX232, MAX232I, DUAL EIA-232***. *Datasheet* do MAX232. Disponível em: <http://pdf1.alldatasheet.com/datasheetpdf/view/27230/TI/MAX232N.html>. Acesso em: 25 out. 2011.

TIPLER, P. A. **Física para Cientistas e Engenheiros, Volume 1: Mecânica, Oscilações e Ondas, Termodinâmica** / Paul A Tipler, Gene Mosca; tradução e revisão técnica Paulo Machado Mors. –Rio de Janeiro: LTC, 2009

TREVISAN, Pedro V. T. **Microcontroladores PIC**. Disponível em: <http://www.eletroeletronica.org/index.php?post/Microcontrolador-PIC>. Acesso em: 08 out. 2011.

UFRGS. **MODULAÇÃO EM AMPLITUDE POR CHAVEAMENTO – ASK.** Disponível em: <http://penta2.ufrgs.br/Alvaro/ask.html>. Acesso em: 12 out. 2011.

ZANCO, W. S. **Microcontroladores PIC 16F628A/648A: Uma Abordagem Prática e Objetiva /** Wagner da Silva Zanco **–** 1.Ed – São Paulo, SP, Brasil: Érica 2005.

APÊNDICE A – Classes Java do projeto

O conteúdo desta seção encontra-se anexo em uma mídia óptica.

APÊNDICE B – Código Fonte dos Microcontroladores

O conteúdo desta seção encontra-se anexo em uma mídia óptica.

ANEXO A – Classe Java Serial

```
package br.uniceub.projetofinal;

import gnu.io.*;
import java.io.IOException;
import java.io.OutputStream;
import java.util.Enumeration;
import java.util.TooManyListenersException;
import javax.swing.JOptionPane;

public class Serial implements SerialPortEventListener { // Classe da comunicacao
serial
    static Enumeration          portList;
    static CommPortIdentifier portId;
    static SerialPort           serialPort;
    static OutputStream       outputStream;
    static boolean       outputBufferEmptyFlag = false;
    static boolean            opened = false;
    public  void  openSerial(CommPortIdentifier  Id,  Integer  baudrate)  {  //Funcao
para conectar na porta serial

        try {
            serialPort = (SerialPort) Id.open("serialComm",1000);
            serialPort.setSerialPortParams(baudrate,
                                           SerialPort.DATABITS_8,
                                           SerialPort.STOPBITS_1,
                                           SerialPort.PARITY_NONE);
            outputStream = serialPort.getOutputStream();

            opened = true;
        } catch (PortInUseException e) {
            JOptionPane.showMessageDialog(null, "Não foi possível conectar.\nPorta
em uso "+Id.getName(), "Aviso", JOptionPane.WARNING_MESSAGE);
        } catch (UnsupportedCommOperationException e) {
        } catch (IOException e) {
        }
    }

    public void initListener() { //Inicia o listener para verificar se o buffer de
saída está limpo
        try {
            serialPort.addEventListener(this);

            serialPort.notifyOnOutputEmpty(true);
        } catch (TooManyListenersException e) {

        }
    }

    public void closeSerial() { //Encerra a conexão da porta Serial
        if (isOpen()) {
            try {
                serialPort.removeEventListener();
                outputStream.close();
                serialPort.close();
                opened = false;
            } catch (IOException e) {

            }
```

```
        }
    }

    public boolean isOpen() { // Returna se há conexão com a porta serial
        return opened;
    }

    public void writeSerial(String packet) { // Envia um String pela porta serial.
        try {
            outputBufferEmptyFlag = false;
            outputStream.write(packet.getBytes());
        } catch (IOException e) {}
    }

    public boolean bufferIsEmpty() { //Verifica se o buffer está vazio
        return outputBufferEmptyFlag;
    }

    public synchronized void serialEvent(SerialPortEvent spe) {
        if(spe.getEventType() == SerialPortEvent.OUTPUT_BUFFER_EMPTY) {
            outputBufferEmptyFlag = true;
            notify();
        }
    }
}
```

ANEXO B – Código-fonte da Codificação Manchester

```
//Função que testa o BYTE recebido e o codifica para Manchester, cria a transição
dos bits
void SEND_DATA(BYTE txbyte)
{
   int i,j,b,me;
   b = txbyte;
   for (i=0; i<2; i++) {
      me = 0; // manchester encoded txbyte
      for (j=0 ; j<4; j++) {
         me >>=2;
         if (bit_test(b,0) )
            me |= 0b01000000; // 1->0
         else
            me |= 0b10000000; // 0->1
         b >>=1;
      }
      putc(me);
   }
}

//Recebe um BYTE codificado em Manchester e realiza o procedimento de
decodificação, removendo a transição dos bits e retornando o BYTE correto.
BYTE DECODE_DATA(BYTE encoded)
{
   BYTE i,dec,enc,pattern;
   enc = encoded;
   if (enc == 0xf0) // start/end condition encountered
      return 0xf0;
      dec = 0;
      for (i=0; i<4; i++) {
         dec >>=1;
         pattern = enc & 0b11;
         if (pattern == 0b01) // 1
            bit_set(dec,3);
         else if (pattern == 0b10)
            bit_clear(dec,3); // 0
         else
            return 0xff; // illegal code
         enc >>=2;
      }
      return dec;
}
```

ANEXO C – Código-fonte Exemplo de Protocolo e CRC

```
#if defined(__PCM__)
#include <16F877.H>
#device *=16
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_B1, rcv=PIN_B0)

#elif defined(__PCH__)
#include <18F452.H>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_B1, rcv=PIN_B0)
#endif

#include <crc.c>

// CONSTANTS
#define MACHINE_ADDRESS 0x01
#define SEND_ADDRESS    0x02
#define ACK             0x01
#define NACK            0xFF
#define BUFFER_SIZE     64

// GLOBAL VARIABLES
int packet_buffer[BUFFER_SIZE];
int ext_buffer[BUFFER_SIZE];
int ext_buffer_next_in;
int ext_buffer_next_out;

#define MESSAGE_SEND    (!input(PIN_B2))
#define DATA_IN         (ext_buffer_next_in != ext_buffer_next_out)


// EXTERNAL INTERRUPT
// Funcao para ler os dados da porta serial
#INT_EXT
void ext_isr()
{
   ext_buffer[ext_buffer_next_in] = getc();     // get a byte, put it in buffer

   if(++ext_buffer_next_in == BUFFER_SIZE)      // increment counter
      ext_buffer_next_in = 0;
}

// GET_BUFF_INT
// Extrai os dados do buffer
int get_buff_int()
{
   int retval;

   while(!DATA_IN);                             // wait until data available

   retval = ext_buffer[ext_buffer_next_out];    // get the byte
   if(++ext_buffer_next_out == BUFFER_SIZE)     // increment counter
      ext_buffer_next_out = 0;
```

```
   return retval;
}

// SEND_PACKET
// Funcao que envia o pacote pela porta serial
void send_packet(int* packet_ptr, int16 packet_length)
{
   int *ptr;
   int16 CRC,i;

   ptr = packet_ptr;                              // set pointer

   CRC = generate_16bit_crc(ptr, packet_length, CRC_CCITT);
                                                  // make CRC
   for(i=0; i<packet_length; i++)                 // send packet
      putc(packet_ptr[i]);

   putc((int)(CRC>>8));                           // send CRC
   putc((int)(CRC));
}

// GET_PACKET
// Recebe o pacote do buffer e interpreta o protocolo
short get_packet(int* packet_ptr)
{
   short retval;
   int16 length;
   int16 CRC;
   int16 i;

   retval = TRUE;

   packet_ptr[0] = get_buff_int();                // get the address of send to
   packet_ptr[1] = get_buff_int();                // get the address of send from

   if(packet_ptr[0] != MACHINE_ADDRESS)
      retval = FALSE;

   packet_ptr[2] = get_buff_int();                // get the control byte
   if(packet_ptr[2] == NACK)
      retval = FALSE;

   packet_ptr[3] = get_buff_int();                // get the length of the data
   packet_ptr[4] = get_buff_int();

   length = (int16)(packet_ptr[3])<<8;
   length += packet_ptr[4];

   for(i=5; i<(length+5); i++)                    // get the data
      packet_ptr[i] = get_buff_int();

   packet_ptr[length+5] = get_buff_int();         // get the CRC
   packet_ptr[length+6] = get_buff_int();

   CRC = (int16)(packet_ptr[length+5])<<8;
   CRC += packet_ptr[length+6];

   if(CRC != generate_16bit_crc(packet_ptr, length+5, CRC_CCITT))
      retval = FALSE;

   return retval;
```

```
}

// Change RS-232 IO pins
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)  // Jumpers: 8 to 11, 6 to 14
//Funcao principal, configure o microcontrolador e mantém em modo contínuo
void main()   {

   ext_buffer_next_in = 0;                      // init variables
   ext_buffer_next_out = 0;

   ext_int_edge(H_TO_L);                        // init interrupts
   enable_interrupts(INT_EXT);
   enable_interrupts(GLOBAL);

   while(TRUE)                                  // loop always
   {
      if(MESSAGE_SEND)                          // if button pushed
      {
         packet_buffer[0] = SEND_ADDRESS;
         packet_buffer[1] = MACHINE_ADDRESS;
         packet_buffer[2] = 0;
         packet_buffer[3] = 0;
         packet_buffer[4] = 9;
         packet_buffer[5] = 'H';
         packet_buffer[6] = 'i';
         packet_buffer[7] = ' ';
         packet_buffer[8] = 't';
         packet_buffer[9] = 'h';
         packet_buffer[10] = 'e';
         packet_buffer[11] = 'r';
         packet_buffer[12] = 'e';
         packet_buffer[13] = '!';
         send_packet(packet_buffer, 14);        // send message
      }

      delay_ms(100);

      if(!DATA_IN)                              // if no data in
         continue;                              // loop back

      if(get_packet(packet_buffer))             // if valid packet
      {
         int16 length,i;
         // int16 CRC;

         printf("Message from unit# %U\r\n",packet_buffer[1]);

         length = ((int16)(packet_buffer[3]<<8)) + packet_buffer[4];

         if(packet_buffer[2] == ACK)
            printf("Previous message sent was received by unit.\r\n");

         if(length)                             // display message
         {
            printf("Message is:\r\n\n");
            for(i=0; i<length; ++i)
               putc(packet_buffer[i+5]);
         }
         printf("\r\n\n... Message End ...\r\n\n\n");

         if(length)
```

```
         {
            packet_buffer[0] = packet_buffer[1];//send an ACK
            packet_buffer[1] = MACHINE_ADDRESS;
            packet_buffer[2] = ACK;
            packet_buffer[3] = 0;
            packet_buffer[4] = 0;
            send_packet(packet_buffer, 5);
         }
      }
      else                                    // if not valid packet
      {
         if(packet_buffer[0] != MACHINE_ADDRESS)
            break;                            // message is not for this PIC
         else if(packet_buffer[2] == NACK)      // tried to send and failed error
            printf("Previous message sent was not received by unit.\r\n");
         else
         {                                        // tried to receive and failed
error
            printf("Message received was corrupted.\r\n");

            packet_buffer[0] = packet_buffer[1];//send a NACK
            packet_buffer[1] = MACHINE_ADDRESS;
            packet_buffer[2] = NACK;
            packet_buffer[3] = 0;
            packet_buffer[4] = 0;
            send_packet(packet_buffer, 5);
         }
      }
   }
}
```