CENTRO UNIVERSITÁRIO DE BRASÍLIA - UniCEUB
FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS – FATECS
CURSO DE ENGENHARIA DA COMPUTAÇÃO

ALEX DA SILVA DE ARAÚJO

SISTEMA DE HARDWARE E SOFTWARE PARA VERIFICAÇÃO DE PREÇO DE PRODUTOS UTILIZANDO CÓDIGO DE BARRAS

Orientador: Prof. MSc. Luciano Henrique Duque

Brasília
Dezembro, 2015.

ALEX DA SILVA DE ARAÚJO

SISTEMA DE HARDWARE E SOFTWARE PARA VERIFICAÇÃO DE PREÇO DE PRODUTOS UTILIZANDO CÓDIGO DE BARRAS

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curso de Engenharia da Computação da FATECS – Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília como requisito para obtenção do título de Engenheiro da Computação.

Orientador: Prof. MSc. Luciano Henrique Duque

Brasília

Dezembro, 2015.

ALEX DA SILVA DE ARAÚJO

SISTEMA DE HARDWARE E SOFTWARE PARA VERIFICAÇÃO DE PREÇO DE PRODUTOS UTILIZANDO CÓDIGO DE BARRAS

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curso de Engenharia da Computação da FATECS – Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília como requisito para obtenção do título de Engenheiro da Computação.

Orientador: Prof. MSc. Luciano Henrique Duque

BANCA EXAMINADORA

Prof. Dr. Abiézer Amarília Fernandes

Coordenador do Curso

Prof. MSc. Luciano Henrique Duque

Orientador

Prfª. Dra. Ingrid Maria Dittert

UniCeub

Prof. MSc. Henrique Marra Taira Menegaz

UniCeub

Brasília

Dezembro, 2015.

AGRADECIMENTOS

Primeiramente agradeço a Deus por me conceder força e paciência para sempre poder adquirir mais conhecimentos nessa vida. Acredito em uma força maior e tenho fé nisso.

Agradeço a minha família por sempre ter me apoiado independente dos caminhos seguidos. Agradeço em especial ao meu pai José Júlio e a minha mãe Rosimar Araújo por ter me dado, acima de tudo, uma educação a qual levarei adiante para meus filhos.

Agradeço a minha namorada e futura esposa Raisa Gomes de Holanda Lopes por sempre estar ao meu lado com amor, carinho, respeito e companheirismo tanto no nosso relacionamento quanto nos meus estudos e vida profissional.

Agradeço aos meus amigos que sempre estiveram ao meu lado tanto em momentos bons quanto em momentos de provas. Estes amigos que nasceram em uma sala de aula e que hoje se tornaram amigos únicos para o resto da vida. Em especial agradeço ao André Luiz, Carlos Marlem, Diego Jardim, Hugo Molina, João Marcos, Pedro Lobo, Samantha Coimbra, Thiago Emanuel, Tiago Mota e Vitor Aires.

Por último e não menos importante agradeço ao Professor Mestre Luciano Henrique Duque pela paciência e ajuda no desenvolvimento deste trabalho. Agradeço também ao coordenador do curso de engenharia o Professor Doutor Abiézer Amarília Fernandes por acreditar na profissão de engenheiro de computação.

Obrigado a todos!

Alex da Silva de Araújo

CITAÇÃO

“A força não vem da vitória. Seus esforços desenvolvem suas forças. Quando você enfrenta dificuldades e decide não se entregar, isso é força. ”

Arnold Schwarzenegger

RESUMO

As inovações tecnológicas que envolvem o segmento de supermercados merecem um destaque, pois o correto emprego das tecnologias é o que determinam o diferencial da empresa. As soluções tecnológicas inovam a forma tradicional de se fazer compras utilizando como base a tecnologia de código barras sendo esta inovação por *softwares* na forma de aplicativos, *e-commerce* ou por *hardwares* que por sua vez inova em dispositivo s que automatizam o processo de fazer compras. Seja qual for a inovação tecnológica ela sempre busca uma comodidade maior ao usuário não distante dos quesitos segurança e confiabilidade. As tendências futuras irão unir o conceito de facilidade com praticidade e seguir esta tendência é o diferencial para qualquer empresa. O trabalho propõe um sistema de *hardware* e *software* capaz de verificar o preço de produtos de supermercados onde será possível, com auxílio da placa de prototipagem Arduino, ler um código de barras e retornar o preço do produto para o usuário através de uma consulta em um banco de dados. Uma solução que trará um conforto, confiabilidade e agilidade para quem frequenta a grande variedade de supermercados.

Palavras chaves: Arduino; código de barras; banco de dados; automatização.

ABSTRACT

The technological innovations involving the supermarket segment are worth emphasizing because the correct use of technologies is what determines the differential of the company. The technological solutions innovate the traditional way of shopping using as a basis the bar code technology and this can be a software solution for innovation in the form of applications, e-commerce or by a hardware solution which in turn innovation in devices that automate the process of shopping. Whatever the technological innovation it always seeks greater convenience to the user of the questions not far safety and reliability. Future trends will unite the concept of ease with practicality and follow this trend is the differential for any company. This paper proposes a hardware and software system able to check product prices in a supermarket where you can, with the help of an Arduino board, read a bar code and return the price of the product to the user through a query on a database. A solution that will bring comfort, safety and agility for those attending the wide variety of supermarkets.

Keywords: Arduino; bar code; database; automation.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

2D	*Two-dimensional*
ARM	*Advanced RISC Machines*
CE	*Chip Enable*
CSN	*Chip Select*
EAN	*European Article Number*
GND	*Ground*
GTIN	*Global Trade Item Number*
IDE	*Integrated Development Environment*
IRQ	*Maskable Interruption*
LCD	*Liquid Crystal Display*
MISO	*Master Input Salve Output*
MOSI	*Master Output Slave Input*
PWM	*Pulse width modulation*
QR Code	*Quick Response Code*
RISC	*Reduced Instruction Set Computer*
RS232	*Recommend Standard – 232*
RX	*Receiver*
SCK	*Serial Clock*
SGBD	Servidor e Gerenciador de Banco de Dados
SPI	*Serial Peripheral Interface*
SQL	*Structured Query Language*
TTL	*Transistor-Transistor Logic*
TX	*Transmitter*
ULP	*Ultra Low Power*
USART	*Universal Synchronous Asynchronous Receiver Transmitter*
VCC	*Collector supply voltage*

SUMÁRIO

1 INTRODUÇÃO

A tecnologia evolui e sofre constantes modificações que afetam o mundo dos negócios e o dia a dia das pessoas. A inovação tecnológica é importante para as empresas, pois ela permite que as empresas fiquem mais competitivas.

A tecnologia possui grande impacto no quesito novos produtos, serviços e modelos de negócios. Dentro do conceito de sistemas de informações gerenciáveis o que se pode destacar é que esses fatores são a principal ferramenta para criar novos tipos de serviços e produtos, inflamar as vendas e até mesmo criar novos conceitos e modelos de negócios (LAUDON e LAUDON, 2011).

Para determinados tipos de negócios é o uso da tecnologia que irá proporcionar vendas, maior rendimento, confiabilidade, segurança e outros fatores. No geral, o implemento de novas soluções tecnológicas é que irá garantir o sucesso da empresa no seu segmento.

Para o segmento de supermercados esta avaliação é totalmente válida, pois o uso da tecnologia neste nicho é o que abrange o assunto da proposta deste trabalho e analisar a forma como ela está inserida atualmente e como ela afeta é um objeto de estudo válido. Este trabalho tem uma visão sobre seu impacto e a forma como ela pode mudar o tipo de modelo de negócio e como as soluções tecnológicas podem afetar diretamente e indiretamente as formas de vendas de produtos de supermercados.

O código de barras foi um grande passo para a evolução tecnológica no segmento de supermercados. Com essa tecnologia foi possível agilizar e realizar o pagamento de forma automática e foi possível também facilitar o cadastro de novos produtos no inventário do supermercado. Sem o uso desta tecnologia os supermercados eram obrigados a fechar as portas para atualização de seu inventário, um procedimento que demorava entre um a dois dias. Segundo Tony Seideman este cenário mudou em 1974 após o primeiro registro oficial do uso de código de barras por um supermercado. Ao comparar o uso de leitores de código de barras em supermercados com o tempo que não havia essa tecnologia o tempo de fila de espera no caixa do supermercado para finalizar a compra diminuiu drasticamente (SEIDEMAN, 2011).

Atualmente o código de barras é a peça fundamental para as tecnologias no segmento de supermercados sendo este o mais utilizado, principalmente pela simplicidade e pelo baixo

custo de implementação, basta observar que no mercado, farmácia, lojas, shopping ou em outros lugares, todos os produtos possuem um código de barras.

Devido ao fato de ser a tecnologia mais utilizada, além da sua simplicidade e facilidade de implementação as inovações tecnológicas giram entorna do código de barras. Porém não é valido apenas inovar no segmento de supermercados sem entender o que o consumidor precisa. Segundo Laudon e Laudon (2011) para uma empresa aumentar seu lucro é preciso manter uma relação mais próxima com seu cliente de uma forma a entender suas necessidades. Atendendo este quesito é a certeza de cliente satisfeito e seu retorno a empresa é garantido (LAUDON e LAUDON, 2011).

Em exemplo o aplicativo para celulares e *tablets* de nome *Fooducate* possui uma ideia inovadora que é permitir o usuário ter acesso rápido a informações de cada produto. Ao ir em um supermercado as pessoas geralmente ficam confusas sobre o que comprar e muitas vezes, ainda é necessário seguir uma dieta com limitações e em certos produtos informações importantes são difíceis de localizar como Kcal, quantidade de proteína, gordura, carboidrato e etc. Então como saber se tal alimento possui ingredientes que não fazem mal para um usuário específico como o caso de pessoas que possuem alergia a glúten? É então que o aplicativo entra em cena, com ele é possível ler o código de barras de qualquer produto e obter instantaneamente informações relevantes para o usuário.

As inovações não se limitam a esta ideia. O exemplo do aplicativo *Fooducate* se limita a apenas um aplicativo de terceiros que possui uma ajuda para o usuário ao obter informações de cada produto. Já a rede de supermercado varejista TESCO que tem a sua sede no Reino Unido inovou além da ajuda de informação ao usuário. A franquia da TESCO localizada na Coréia inovou de forma diferente, ela resolveu implementar uma espécie de mercado virtual presencial, ou seja, nas redes de metrô foram impressas fotos de prateleiras e que os usuários entre o intervalo de cada metrô poderiam comprar vários produtos. A compra funciona da seguinte forma: O usuário com o aplicativo da TESCO em seu celular faz a leitura do código QR (um tipo específico de código de barras 2D) e a partir disto é possível fazer uma lista de compra. Como a prateleira é apenas um papel impresso com fotos dos produtos o potencial da inovação consiste em ao finalizar a compra o mercado se assegura de entregar o produto para a casa do cliente. Esta etapa pode acontecer no mesmo dia ou no próximo dia útil.

Seguindo esta tendência a Amazom.com lançou um produto inovador: o Amazon Dash. Um pequeno dispositivo que lembra um controle remoto que é capaz de fazer compras sem a necessidade de se dirigir a um supermercado. O Amazon Dash é dotado de um leitor de código

de barras com um microfone sendo este dispositivo capaz de ler um código de barras ou reconhecer por comando de voz o nome de um produto e adicioná-los na sua conta Amazon bastando apenas deixar o dispositivo conectado à rede sem fio da residência. Para finalizar as compras basta entrar na sua conta da Amazon através do seu computador pessoal, celular ou *tablet* e fazer o pagamento e esperar o produto chegar em sua casa. De forma análoga ao supermercado TESCO alguns pedidos chegam no mesmo dia ou no próximo dia útil.

Segundo o site *think with google,* existem algumas tendências tecnológicas para o ano de 2015, dentre as que podemos citar no contexto de segmento de supermercado é que a tendência futura é a diversidade de dispositivo s *online* que conversam entre si. Esta visão lembra o conceito de Internet da Coisas que são vários dispositivos conectados à internet capazes de interagir com o usuário. Outra tendência que se segue é bem conceitual que é a necessidade de resposta rápida, ou seja, o usuário que, por exemplo, queira comprar alguma que isso seja feita de maneira rápida e integrada. Esta é a visão para o futuro. Soluções integradas que possam promover melhorias tanto na forma de comprar quanto tornar o processo de fazer comprar mais simplificado, rápido e seguro para qualquer pessoa (THINK WITH GOOGLE, 2015).

Nesse cenário, o projeto proposto visa o desenvolvimento de um sistema de *hardware* e *software* capaz de verificar o preço do produto utilizando dois dispositivos: o primeiro realiza a leitura do código de barras e se comunica com o segundo dispositivo, através de troca de mensagens via rádiofrequência, que faz uma consulta ao banco de dados por intermédio de um *software*.

O projeto visa permitir ao usuário a facilidade e a comodidade na escolha de produtos em uma prateleira da forma que é possível checar seu respectivo valor, antes mesmo de chegar ao caixa. Essa facilidade gera um maior controle do usuário pois é possível ter a comprovação do preço real do produto, além do fato que o sistema por completo irá auxiliá-lo e será capaz de agilizar o processo como um todo trazendo benefícios como confiabilidade e segurança para quem frequenta a grande variedade de redes de supermercado.

1.1 Objetivo geral

O objetivo geral do projeto é desenvolver um sistema de *hardware* e *software* capaz de ler um código de barras e retornar o preço do produto para o usuário através de uma consulta em banco de dados.

1.2 Objetivos específicos

Para que o objetivo geral seja alcançado é necessário determinar os seguintes pontos específicos:

a) Construção de um sistema de *hardware* com nome de Dispositivo Carrinho, com o auxílio da placa Arduino MEGA, capaz de fazer a leitura de um código de barras e que transmita e receba dados via rádiofrequência.
b) Desenvolver uma placa de circuito impresso para a placa de desenvolvimento Arduino MEGA contendo um módulo de rádiofrequência nRF24L01, leitor de código de barras, circuito impresso max232 e alguns capacitores eletrolíticos.
c) Construção de um segundo sistema de *hardware* com nome de Dispositivo Servidor, com auxílio de uma placa Arduino UNO, capaz de receber e transmitir dados via rádiofrequência e trocar informações com um computador através de comunicação serial.
d) Desenvolver uma placa de circuito impresso para a placa de desenvolvimento Arduino UNO contendo um módulo de rádiofrequência nRF24L01 e um *buzzer*.
e) Elaborar um *software* em linguagem C# capaz de ler os dados a partir de uma porta serial e fazer instruções do tipo *select* e *insert* em um banco de dados.
f) Elaborar um banco de dados com instância local para consulta e armazenagem de dados como código de barras e preço.
g) Efetuar teste no *hardware* levando em consideração comunicação entre as plataformas de rádiofrequência e dados na porta serial.
h) Efetuar teste no *software* envolvendo conectividade, transferência de dados e verificação de instruções do tipo *select* e *insert* no banco de dados.

1.3 Metodologia

Para fins acadêmicos a melhor proposta de implementação deste sistema é utilizar a plataforma Arduino devida a agilidade, rapidez, e elaboração de protótipos testes em pequena e média escala afins de validação e análise de viabilidade de ideias. Ele possui também uma vasta quantidade de livrarias livre disponíveis *online* além de uma comunidade extensa com vários projetos básicos que servem de ponto de partida sem ter a necessidade de projetar funções básicas do primórdio. É utilizado a linguagem C# para elaborar o *software* intermediário que por sua vez possui sua implementação através do IDE Visual Studio. E por fim, será utilizado o banco de dados MySQL pois possui uma interface intuitiva para construção de banco de dados levando em consideração a agilidade para implementação e manutenção.

Dentro do cenário apresentado para alcançar os pontos explanados no objetivo específico se faz necessário demonstrar as etapas abaixo.

Primeira etapa: tempo destinado a estudos e pesquisas. Os tópicos a serem pesquisados e estudados são explicados abaixo:

a) Tecnologias utilizadas nos supermercados no brasil e exterior;
b) Estrutura do código de barras;
c) Arduino e sua compatibilidade com leitores de código de barras, porta serial, banco de dados;
d) Leitor de código de barras com porta serial padrão RS232;
e) Integração entre C# e MySQL;
f) Integração entre porta serial e C#;
g) Módulo de rádiofrequência nRF24l01 para comunicação via rádiofrequência entre dois Arduinos.

Segunda etapa: é destinada a construção do *Hardware*. Será desenvolvido o Dispositivo Carrinho para leitura de código de barras e deverá conter um módulo de rádiofrequência 2.4GHZ nRF24L01 para transferências de mensagens. E posteriormente será desenvolvido o Dispositivo Servidor para captação destas mensagens através do módulo de rádiofrequência nRF24L01 e sua função será trocar informações com o computador via comunicação serial.

Terceira etapa: é destinada a construção do *software*. Será elaborado um *software* em C# utilizando a IDE do Visual Studio e sua função será interpretar os dados que serão armazenados na porta serial e na sequência tratar estas mensagens fazendo instruções *select/insert* em um banco de dados MySQL.

Quarta etapa: elaborar um Banco de dados. Será utilizado o banco de dados MySQL e seu acesso será em instância local. A tabela a ser desenvolvida será simplificada e deverá conter um código de identificação, código de barras e o preço em colunas respectivas.

Quinta etapa: será desenvolvido uma rotina de testes levando em consideração todas as partes desenvolvidas (*hardware* e *software*) ainda em placa de ensaio (*protoboard*). A sequência dos testes é exibida abaixo:

a) Comunicação entre os Dispositivos Carrinho e Servidor via rádiofrequência;
b) Comunicação entre o Dispositivo Servidor e computador via comunicação serial;
c) Consulta com um código de barras em que o produto esteja cadastrado no banco de dados;
d) Consulta com um código de barras em que o produto não esteja cadastrado no banco de dados;
e) Função *insert* através do *software* C#;
f) Apresentação de erros no sistema.

Sexta etapa: desenvolver placas de circuito impresso. Serão projetadas as placas:

a) Placa de circuito impresso para a placa de prototipagem Arduino UNO com seus respectivos componentes eletrônicos;
b) Placa de circuito impresso para a placa de prototipagem Arduino MEGA com seus respectivos componentes eletrônicos.

Sétima etapa: elaborar protótipo final. Esta etapa consiste na montagem final do *hardware* e do *Software*.

Hardware:

a) Confecção de placa de circuito impresso para o Dispositivo Carrinho e Dispositivo Servidor.

Software:

a) Iniciar a instância local do banco de dados.
b) Executar em segundo plano o *software* intermediador.

1.4 Motivação

A motivação do trabalho é prover para os usuários de supermercados conforto, suporte e apoio ao fazer compras em um supermercado utilizando tecnologia existentes não distante dos quesitos conforto e facilidade no processo mais importante que é a escolha dos produtos a serem adquiridos.

O projeto possui a inspiração na automatização do processo de escolha de produtos em prateleiras de supermercado da forma que seja possível agilizar este processo e fazer de forma mais rápida gerando maior confiabilidade e segurança no processo de compra.

1.5 Resultados esperados

Como resultado final do projeto é esperado a possível implementação do sistema de *hardware* e *software* com a automatização do processo de consulta de preços de produtos em uma rede de supermercado. Da forma proposta que o Dispositivo carrinho reconheça um código de barras através de um leitor ótico e transfira essa informação, via rádiofrequência, ao Dispositivo servidor que através de um notebook com *software* funcionando em segundo plano e um banco de dados irá retornar o valor do produto que foi requisitado no início.

A comunicação e a interação entre *hardware* e *software* são fatores importante. É esperado que os dispositivos possuam a capacidade de troca de mensagens e que o *software* seja capaz de identificar as mensagens e fazer o devido tratamento para que assim o pedido de informação seja atendido e então passar pelo sistema de uma forma completa para que seja possível entregar a informação de resposta ao dispositivo que solicitou o pedido.

1.6 Estrutura do trabalho

O trabalho está dividido em capítulos. O capítulo 1 visa dar uma introdução fazendo uma abordagem sobre tecnologias inovadoras e como os supermercados utilizam essas tecnologias. Este capítulo também aponta exemplos de inovações tecnológicas fazendo uma abordagem para introduzir ao tema do proposto trabalho. Além de mostrar em detalhes superficiais da proposta deste trabalho entre outras informações. O capítulo 2 mostra os conceitos básicos que envolvem o tema do projeto como a tecnologia código de barras. Este capítulo está dividido em teorias de *hardware* e *software*. O primeiro faz referência ao dispositivo Arduino, módulo de rádiofrequência nRF24L01 e sobre o circuito integrado de conversão de sinal MAX 232. O segundo faz referência ao conceito de linguagem de programação C# e Banco de dados. O capítulo 3 é o desenvolvimento do trabalho proposto e contém primeiramente uma explicação do projeto desenvolvido divido em etapas com fluxograma de ideias e posteriormente a isso será explicado a divisão de blocos do projeto para que assim seja possível explicar o desenvolvimento deste trabalho em pontos específicos. O capítulo 4 são os testes e os resultados em que se serão explicados os cenários de testes aplicados ao projeto, bem como os resultados obtidos ao longo do desenvolvimento. O capítulo 5 é a conclusão do trabalho em que será o fechamento dos tópicos desenvolvidos bem como uma proposta de trabalhos futuros. Posteriormente a isso é apresentado os elementos pós-textuais como as referências bibliográficas utilizadas para a construção do trabalho bem como os apêndices contendo o código fonte.

2 REFERENCIAL TEÓRICO

Este capítulo tem por objetivo fazer um direcionamento e citar assuntos relevantes ao que diz respeito da proposta de solução. O aprofundamento de cada assunto está limitado também a proposta do trabalho. Serão abordados assuntos como código de barras afim de se compreender como é o seu funcionamento, tipo utilizado no trabalho, quantidades de dados codificados além de algumas vantagens. Demais desenvolvimento deste capítulo está segmentado em *Hardware* e *Software*. A primeira parte irá tratar sobre o Arduino e suas versões; Ambiente de desenvolvimento (IDE) e especificações técnicas. Bem como o circuito impresso MAX 232 e módulo de rádiofrequência nRF24L01. A segunda parte irá tratar de linguagem de programação C# com ambiente de desenvolvimento Visual Studio e Banco de dados MySQL com ambiente de desenvolvimento Workbench.

2.1 Código de barras

O código de barras funciona com o princípio chamado de simbologia que é o que define a forma básica do código de barras para sua codificação e interpretação. Tal codificação permite o *scanner* (neste caso seria o leitor de código de barras) saber onde exatamente começa e termina os dados a serem analisados o que é bem semelhante a representação binária (0 e 1). A matriz de dados por sua vez é definida por linhas paralelas pretas e brancas. Mesmo com os avanços tecnológicos, a tecnologia de código de barras ainda se mantém como uma melhor opção na automação em segmentos de supermercados devido ao seu baixo custo de implementação (SEIDEMAN, 2011).

O mapeamento da informação codificada pode acontecer de várias maneiras dependendo da necessidade e aplicação dessa tecnologia. Em concepção macro pode-se dividir o código de barras em duas categorias. Linear ou 1D que é a forma mais conhecida devido a vasta aplicabilidade em segmentos diferentes não somente em supermercados. E a outra categoria é a *dataglyphs* ou 2D que possui uma capacidade superior de codificação de informação ao comparado com a linear.

Os códigos de barras 2D possuem sua codificação definida por vários pontos em miniaturas ou padrões circulares e em alguns tipos específicos podem ser conjuntos de formatos e módulos inseridos em uma imagem especificada.

Um exemplo de código de barras 2D é Data Matrix que é capaz de codificar um conjunto extenso de dados. Sua forma consiste em sequencias randômicas de pares brancos e pretos. Com esse tipo de código de barras é possível armazenar aproximadamente 2.335,00 caracteres alfanumérico em um único símbolo de código de barras. O famoso *QR Code* é uma derivação deste tipo com capacidade de codificar símbolos além de caracteres alfanuméricos.

A FIGURA 2.1 abaixo mostra o exemplo de como é o resultado final da codificação da sequência de caracteres “EngenheiroAlex123” no código Data Matrix.

FIGURA 2.1 – EXEMPLO DE UM CÓDIGO DE BARRAS DATA MATRIX REPRESENTANDO OS CARACTERES “EngenheiroAlex123”.

Fonte: *Software* Zint Barcode Studio 2.4 – gerador freeware de código de barras

De forma similar aos códigos de barras 2D, os códigos de barras lineares possuem vários tipos e características individuais. Neste trabalho será utilizado apenas o código de barras linear usado em segmentos de supermercados brasileiros. O tipo mais utilizado é o EAN-13 ou GTIN (Número Global de Item Comercial) (BYTE SCOUT, 2014).

EAN é a abreviação de *European Article Number* e foi desenvolvido na Europa e adotado pelo brasil como padrão para supermercados, farmácia entre outros. Sua característica principal é a quantidade de algarismos que compõem este tipo, que são 13 números, sendo este último número um digito verificador com algoritmo próprio para verificação e veracidade das informações.

A FIGURA 2.2 abaixo ilustra um código de barras do tipo EAN 13.

FIGURA 2.2 - ESTRUTURAÇÃO DO CÓDIGO DE BARRAS EAN-13.

Fonte: Disponível em www.gs1br.org

O código EAN-13 foi desenvolvido pela *International Article Numbering* e inicialmente implementado nos países da Europa. A organização internacional de padrões GSI foi a que definiu as características do EAN-13. Seu propósito é ser um tipo de algoritmo que seja rápido de ser codificado e decodificado por isso o fato de apenas codificar números. E a sua leitura pode ser feito pela maioria dos scanners e em ângulos agudos de até 45 graus.

Por ser uma tecnologia utilizada em supermercados no brasil é objeto de estudo de importância pois os produtos analisados estarão neste padrão, sendo assim é importante compreender as informações por trás deste código de barras.

2.2 Arduino

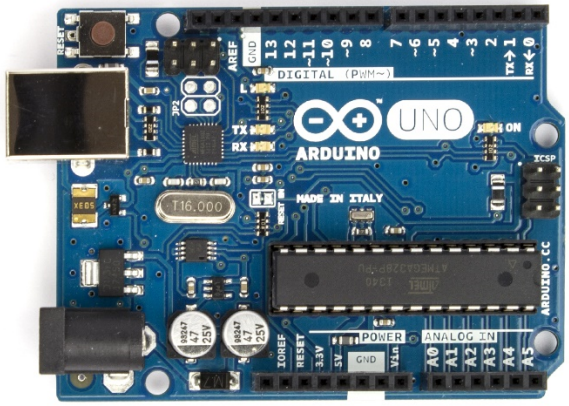
O Arduino, originalmente produzido e fabricado na Itália, é uma plataforma *open source* de prototipagem eletrônica. Possui uma placa embarcada com entradas digitais e analógicas capaz de controlar vários componentes eletrônicos, sensores, motores, saídas de Modulação de Largura de Pulso (PWM) entre outras diversas opções e funções (SOARES, 2013).

Originalmente o projeto Arduino foi destinado para o ensino em escolas, porém o potencial do projeto gerou maiores expectativas e foi possível criar uma comunidade onde é possível replicar o projeto Arduino, ou seja, é possível qualquer empresa ou pessoa física fabricar seu próprio Arduino desde que se mantenha o os padrões exigidos pelos idealizadores do projeto além de que as informações são livremente compartilhadas que é a principal vantagem de uma plataforma *open-source* (SOARES, 2013).

Existem vários modelos de Arduino e o que irá definir qual modelo utilizar é a própria necessidade do projeto e protótipo a ser desenvolvido, seja ele, menos complexo como o Arduino Uno com microprocessador de 8-bit e *clock* de 16MHz ou um projeto que exija um microcontrolador mais robusto como o caso do Arduino Due que possui um processador baseado no Atmel SAM3X8E ARM Cortex-M3 de 32-bit e *clock* equivalente a 84 MHz. Seja qual for a sua ideia, protótipo, jogo ou automação a base de prototipagem Arduino é excelente para fazer funcionar (FILIPEFLOP, 2014).

O Arduino UNO é equipado com o microcontrolador Atmega328P de 8-bit e *clock* de 16Mhz com 32KB de memória flash (0.5KB destinados ao *bootloader*). Possui 14 pinos de entradas e saídas digitais sendo que 6 podem ser utilizadas para Modulação de Largura de Pulso (PWM). Possui uma entrada USB para alimentação e porta programável e sua voltagem de operação é de 5 V. A variação de voltagem de entrada é de 6 a 20 V e a corrente DC por pino de entrada/saída é de 40 mA. O Arduino possui também um pino específico para corrente DC de 3.3 V e corrente de 50 mA. A FIGURA 2.3 abaixo ilustra o Arduino UNO R3 (ARDUINO, 2015).

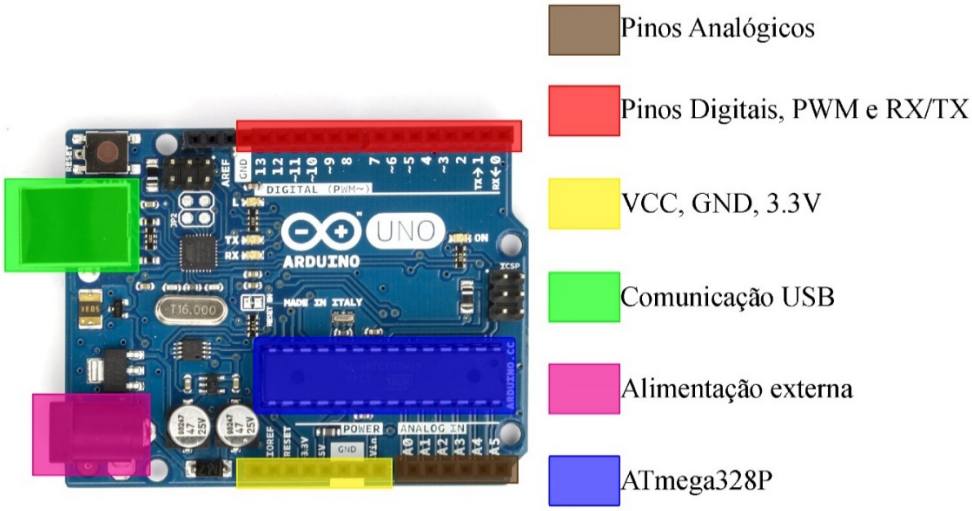
FIGURA 2.3 – ARDUINO UNO R3.



Fonte: adaptado de Arduino.cc

A FIGURA 2.4 abaixo demonstra a localização e identificação das entradas e saídas, pinos de alimentação e analógicos além do conector de fonte externa e entrada USB para comunicação USB/Serial na placa Arduino UNO.

FIGURA 2.4 – CONEXÕES DO ARDUINO UNO.

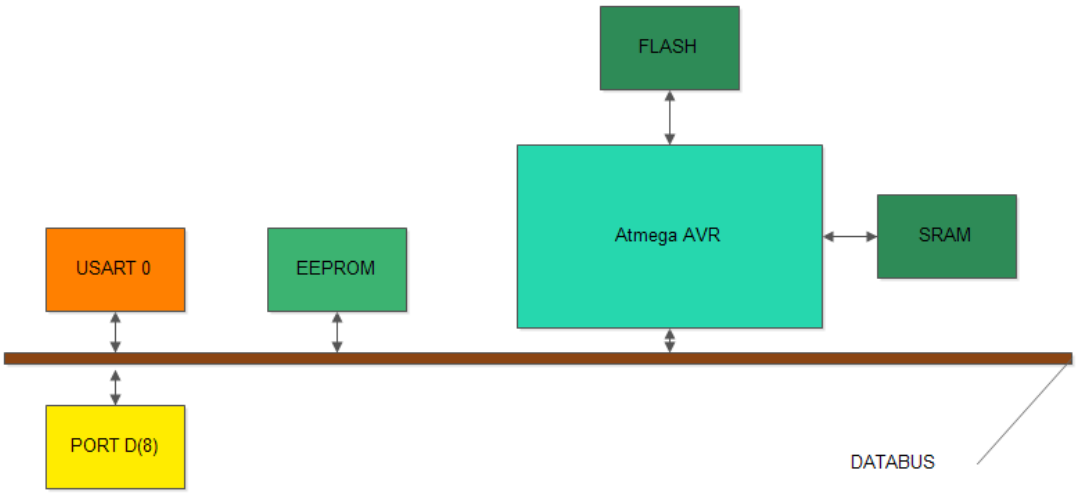


Fonte: Adaptado de Arduino.cc

O microcontrolador ATmega328P é baseado na tecnologia RISC (*Reduced Instruction Set Computer*) e sua definição são os barramentos independentes para programa e dados. O fabricante Atmel defende a sua tecnologia AVR em que os microcontroladores foram baseados e olhando dentro da sua estrutura é possível observar que possui o padrão USART (*Universal Synchronous Asynchronous Receiver Transmitter*) programável sendo responsável para a comunicação serial.

A FIGURA 2.5 abaixo representa um diagrama de bloco abaixo pertencente ao microcontrolador Atmega328P onde é possível ver, de uma forma sucinta, sua estrutura interna (ATMEL CORPORATION, 2015).

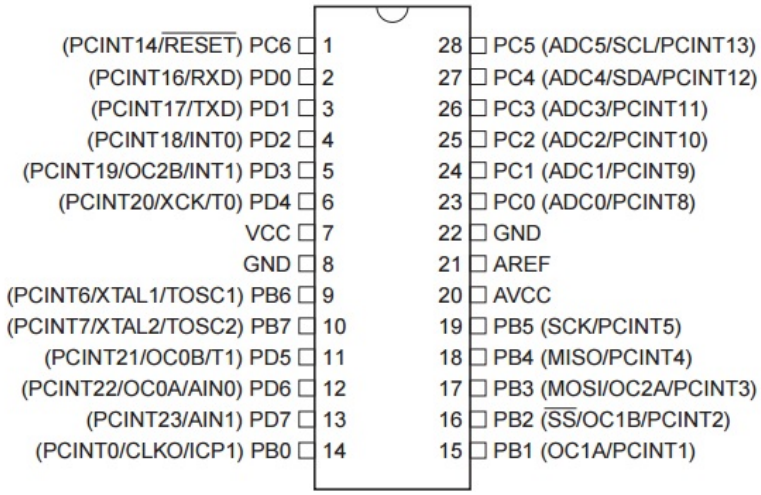
FIGURA 2.5 – DIAGRAMA DE BLOCO DO MICROCONTROLADOR ATMEGA328P.



Fonte: Adaptado do *datasheet*, disponível em <www.atmel.com>

Os pinos 0 RX e 1 TX para transmissão de dados serial da placa Arduino se conectam diretamente com o microcontrolador nas portas 2 e 3 conforme é ilustrado na FIGURA 2.6 abaixo (ARDUINO, 2015).

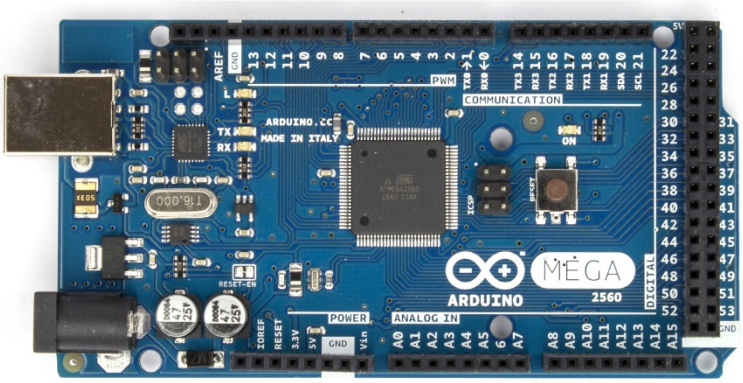
FIGURA 2.6 – DIAGRAMA DE PINO DO MICROCONTROLADOR ATMEGA328P.



Fonte: *Datasheet* disponível em: http://www.atmel.com/Images/doc8161.pdf

O Arduino Mega é bem similar ao Arduino Uno em configurações e é considerado uma atualização mais completa do mesmo. Possui o microcontrolador Atmega2560 de 8-bit e *clock* de 16Mhz com 256KB de memória flash (com 8KB destinado ao *bootloader*). Seu diferencial são as quantidades de portas digitais de entrada e saída que são no total 56 (Dentre o total, 15 podem ser usadas como Modulação de Largura de Pulso – PWM) e no total são 4 conjuntos de comunicação serial RX1/TX1, RX2/TX2 e assim por diante. As demais informações seguem o padrão do Arduino UNO. A FIGURA 2.7 abaixo ilustra o Arduino Mega2560 (ARDUINO, 2015).

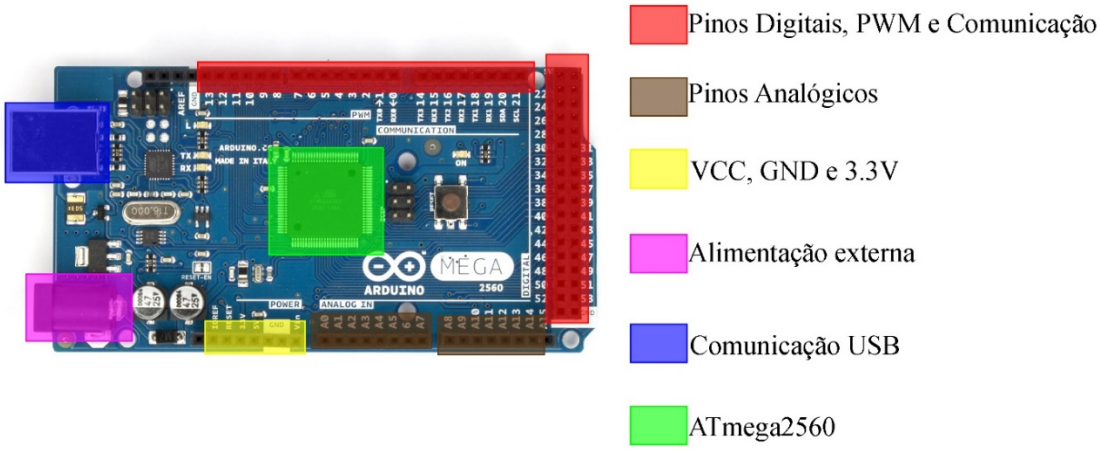
FIGURA 2.7 - ARDUINO MEGA 2560.



Fonte: disponível em https://www.arduino.cc/en/Main/ArduinoBoardMega2560

Abaixo, na FIGURA 2.8, é mostrado um explicativo indicando a localização e identificação das entradas e saídas, pinos de alimentação e analógicos além do conector de fonte externa e entrada USB para comunicação USB/Serial na placa Arduino Mega2560.

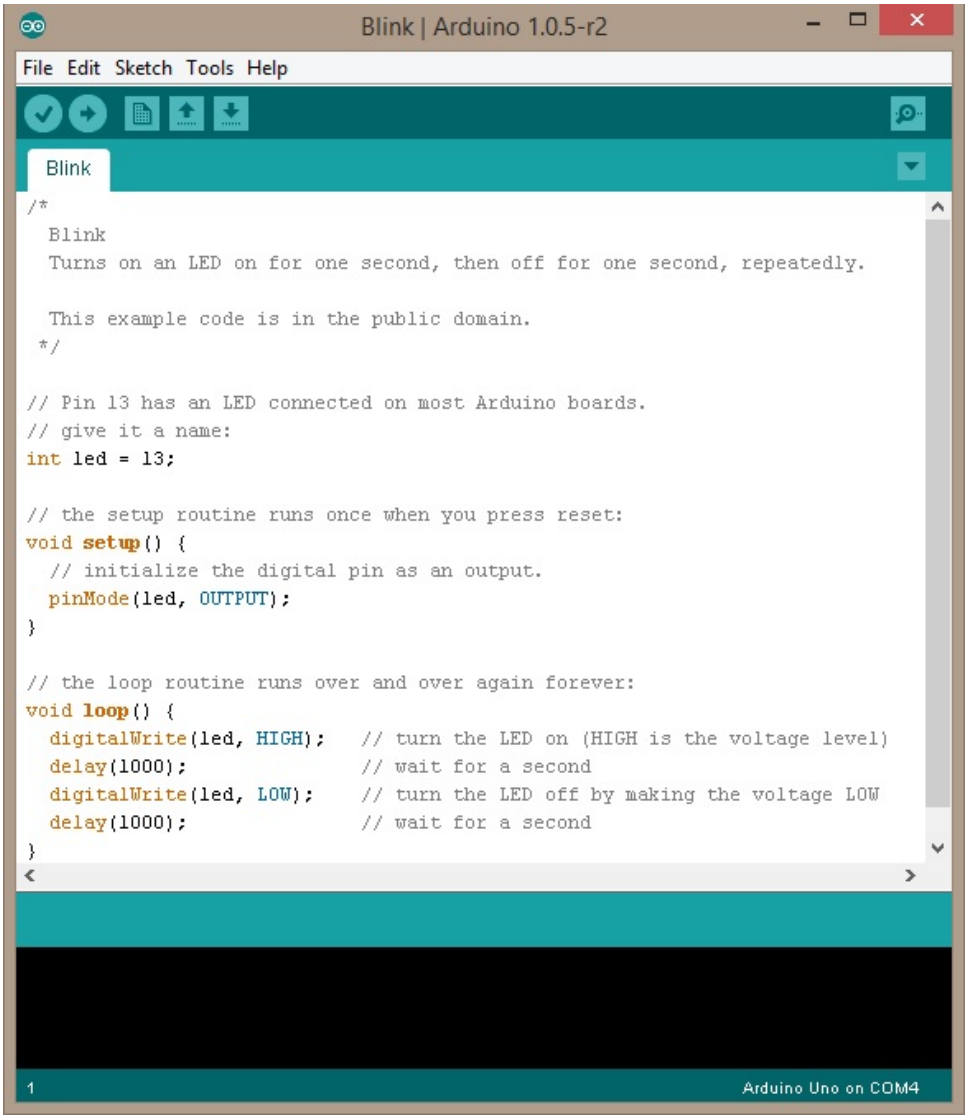
FIGURA 2.8 – CONEXÕES DO ARDUINO MEGA2560.



Fonte: Adaptado de https://www.arduino.cc/en/Main/ArduinoBoardMega2560

Seja qual for o Arduino que esteja utilizando a plataforma de desenvolvimento IDE (Integrated Development Enviroment) é a mesma. Esta plataforma é baseada em Java e sua função é fornecer um ambiente para desenvolvimento de *software*. Abaixo, na FIGURA 2.9, é ilustrado um exemplo de um código que pisca um LED na porta digital 13 a cada um segundo onde pode ser encontrado nos códigos exemplos da própria IDE (Arquivo>Exemplos>Basics>Blink) (ARDUINO, 2015).

FIGURA 2.9 – AMBIENTE DE DESENVOLVIMENTO IDE.

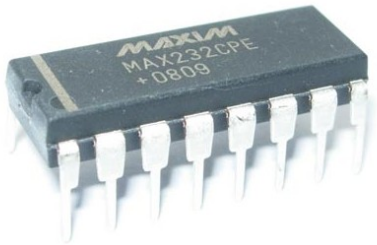


Fonte: Adaptado de Arduino.cc

2.3 Circuito integrado MAX 232

O MAX 232 é um circuito integrado lógico fabricado e distribuído pela empresa *Maxim Integrated* e sua função é fazer a conversão de sinal do padrão RS-232 vindo do leitor de código de barras para a placa Arduino que utiliza o padrão TTL. Abaixo, na FIGURA 2.10, é ilustrado uma foto do circuito integrado (MAXIM INTEGRATED, 2015).

FIGURA 2.10 - MAX232.



Fonte: Adaptado de huinfinito.com.br

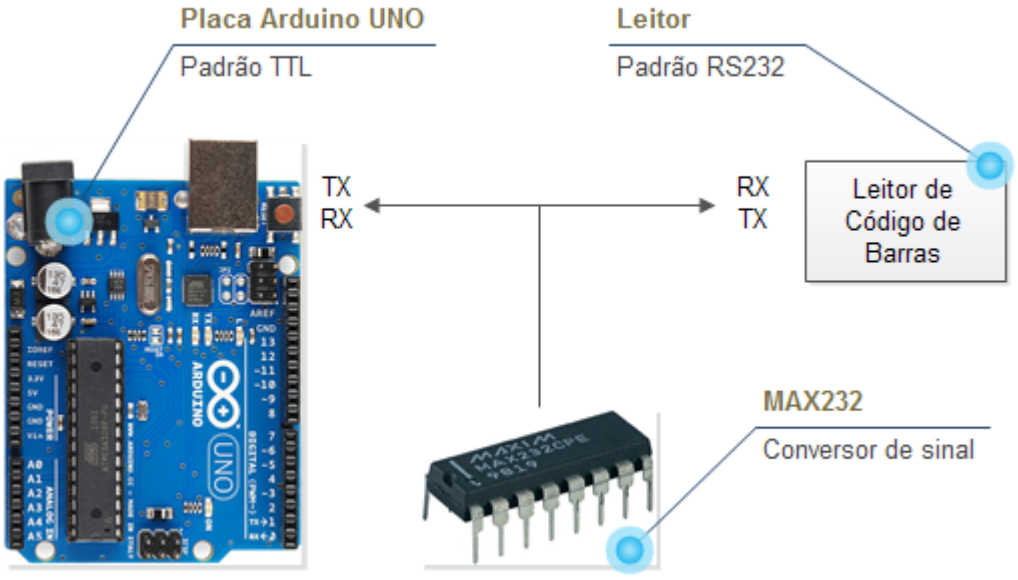
Por padrão a placa Arduino transmite e recebe dados serialmente através do padrão USART já explicado anteriormente. Este padrão transmite um bit de cada vez sendo programável a taxa de transferência (4800,9600,19200bps ou outro valor). Este método é caracterizado pelo padrão TTL (Transistor-Transistor Logic) em que sua voltagem limite de operação é de 0 a 5 V ou 3.3 V. Para transmitir um valor lógico alto a representação é um bit “1” (5V) e um valor baixo é representado por um bit “0” (0 V) (SPARKFUN, 2010).

O leitor de código de barras foi desenvolvido no padrão de comunicação RS-232 (Recommended Stantard 232). E análogo ao Arduino este padrão também transmite um bit por vez sendo que a representação de um valor alto é também um bit “1” porém esta representação é uma voltagem negativa que pode variar entre -3 a -25 V. Já a representação de um valor baixo é dada pela representação de um bit “0” e é representado por um valor positivo que pode variar entre 3 a +25 V. De uma forma mais genérica estes valores estão em uma *range* que variam entre -13 a +13 V (SPARKFUN, 2010).

Para estabelecer uma conexão segura e evitar de queimar a placa Arduino por trabalhar uma voltagem mais baixa que o leitor de código barras e não somente isso é necessário também inverter estes sinais pois o padrão TTL não é similar ao RS-232, este possui sinais invertidos e uma tensão superior a placa Arduino. Então se faz necessário o circuito integrado MAX232 que servirá como um intermediador entre o padrão RS232 do leitor de código de barras (-13 V a +13 V) e o Arduino que utiliza o padrão TTL (0 a 5 V).

Abaixo, na FIGURA 2.11, é ilustrado a ligação básica de comunicação entre Arduino, max232 e leitor de código de barras.

FIGURA 2.11 - EXEMPLO DE COMUNICAÇÃO UTILIZANDO O MAX232.



Fonte: Adaptado de https://www.sparkfun.com/tutorials/215

2.4 Módulo de rádiofrequência nRF24L01

O módulo de rádiofrequência nRF24L01 é um transceptor, ou seja, um único módulo é capaz de transmitir e receber mensagens. Muito utilizado para comunicação *wireless* em protótipos Arduino que necessitam a troca de informações. Suas principais características são de possuir um tamanho reduzido em comparação a outros módulos além de possuir antena embutida que evita a integração de antenas adicionais para seu total funcionamento. A FIGURA 2.12 abaixo ilustra o módulo de rádiofrequência nRF24L01 (FILIPEFLOP, 2014).

FIGURA 2.12 – MÓDULO DE RÁDIOFREQUÊNCIA.

Fonte: (FILIPEFLOP, 2014)

O circuito integrado nRF24L01+ que constitui o módulo transceptor é fabricado pela empresa *Nordic Semiconductor* e é da categoria de baixo consumo ULP (Ultra Low Power). Ele opera na frequência de 2.4 Ghz e com taxa de transferência de 2Mbps. Sua corrente, quando operando, é um valor abaixo de 14 mA e possui um valor inferior a μA em modo *power down* desta forma é possível prover uma longa vida utilizando apenas uma bateria simples. Sua alimentação é entre 1.9 a 3.6 V (NORDIC SEMICONDUCTOR, 2015).

O nRF24L01+ possui um protocolo interno de aceleramento de *hardware* chamado de *ShockBurst* que suporta uma alta velocidade sobre a interface SPI suportada pelos micros-controladores como é o caso da placa de prototipagem Arduino (NORDIC SEMICONDUCTOR, 2015).

O padrão de comunicação do módulo de rádiofrequência com a placa de prototipagem Arduino é o padrão SPI (Serial Peripheral Interface) e, diferentemente do padrão assíncrono como ocorre na comunicação RS232 o protocolo SPI é síncrono. Isto implica em uma comunicação *Half-duplex* em que os dos mensageiros possuem o poder de mandar e receber mensagens (não simultaneamente) pelo mesmo canal de comunicação. Este padrão suporta uma taxa de transferência de dados de 2Mbps, porém este não é o limite de velocidade do protocolo sendo que cada fabricante de Dispositivo s deste segmento pode abortar velocidades superiores (SACCO, 2014).

Para comunicação com a placa Arduino é necessário utilizar as conexões de sinais MOSI (Master Output Slave Input), MISO (Master Input Slave Output) e SCK (Serial Clock). É ilustrado, na FIGURA 2.13, o diagrama de pinos do módulo de rádiofrequência a fim de mostrar suas localizações.

FIGURA 2.13 - DIAGRAMA DE PINO DO MÓDULO DE RÁDIOFREQUÊNCIA.

Fonte: disponível em: http://blog.filipeflop.com/wireless/arduino-modulo-nrf24l01-tutorial.html

O QUADRO 2.1 abaixo ilustra a função de cada pino do modulo de rádiofrequência nRF24L01.

QUADRO 2.1 – FUNÇÃO DOS PINOS DO MÓDULO DE RÁDIOFREQUÊNCIA.

Número do Pino	Nome abreviado	Função
1	VCC	Alimentação 3.3V
2	VCC	Alimentação 3.3V
3	CE	Chip Enable RX/TX
4	CSN	SPI Chip Select
5	SCK	SPI *Clock*
6	MOSI	SPI Slave Data Input
7	MISO	SPI Slave Data Output
8	IRQ	Interrupção
9	GND	Ground
10	GND	Ground

Fonte: (FILIPEFLOP, 2014).

2.5 *Software*

A linguagem de programação C# é bem simples e moderna e totalmente orientada a objetos além de ser *type-safe* (valida o que o usuário está digitando). No início seu marketing girava em torno de que sua principal função seria o desenvolvimento de aplicativos .NET em área denominada de computação empresarial. O que pode ressaltar sobre a linguagem C# é que sua forma é a simplificação da linguagem C++ nos quesitos áreas de classes, gerenciamento de exceções, espaços de nomes e sobrecarga de métodos. A complexidade do C++ foi simplificada no C# a fim de ser reduzir a taxa de erro de programação e tonar o uso mais fácil e simplificado (WILLE, 2001).

A linguagem de programação C#, que executa o .NET framework, é uma linguagem de programação robusta e simples e sua aplicação é da mais variada vai desde jogos até aplicativos simples. Sua base é a programação orientada a objetos e mantêm o clássico e estilo famoso da linguagem C (MICROSOFT, 2015).

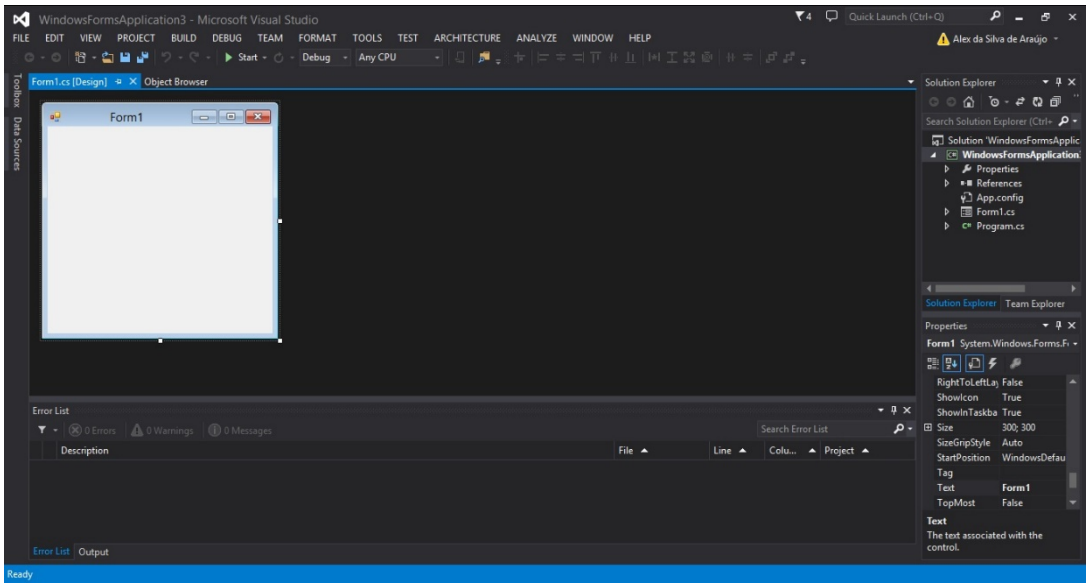
O Visual Studio é o ambiente de desenvolvimento (IDE) de linguagem de programas como C#, Visual Basic, Visual C#, Visual F#, Visual C++ e JavaScript. Com esta ferramenta é possível a criação de diferentes tipos de aplicações tais como as mais modernas Windows Phone e Windows Store a mais comuns como aplicativos para Área de trabalho, aplicativos e serviços da web. Sua característica fundamental e importante que facilita o desenvolvimento é a o padrão adotado pelo Visual Studio: “Você programa o que vê”. Desta forma o ambiente fica interativo e intuitivo (MICROSOFT, 2015).

O Visual C# é a implementação propriamente dita da linguagem C# fornecido pela Microsoft dentro do compilador. Uma vantagem, pois, desta forma é oferecida ao usuário um completo editor de código, modelos de projetos, assistente de código além de outras ferramentas que auxiliam e aceleram o processo de desenvolvimento.

O ambiente de desenvolvimento (IDE) do Visual Studio é mostrado na Figura 2.14 abaixo já configurada para o projeto “Windows Form Aplication”. É possível logo de início já ter uma previa da janela do Windows em branco antes mesmo de começar o desenvolvimento.

A FIGURA 2.14 abaixo ilustra o ambiente de desenvolvimento (IDE) do Visual Studio.

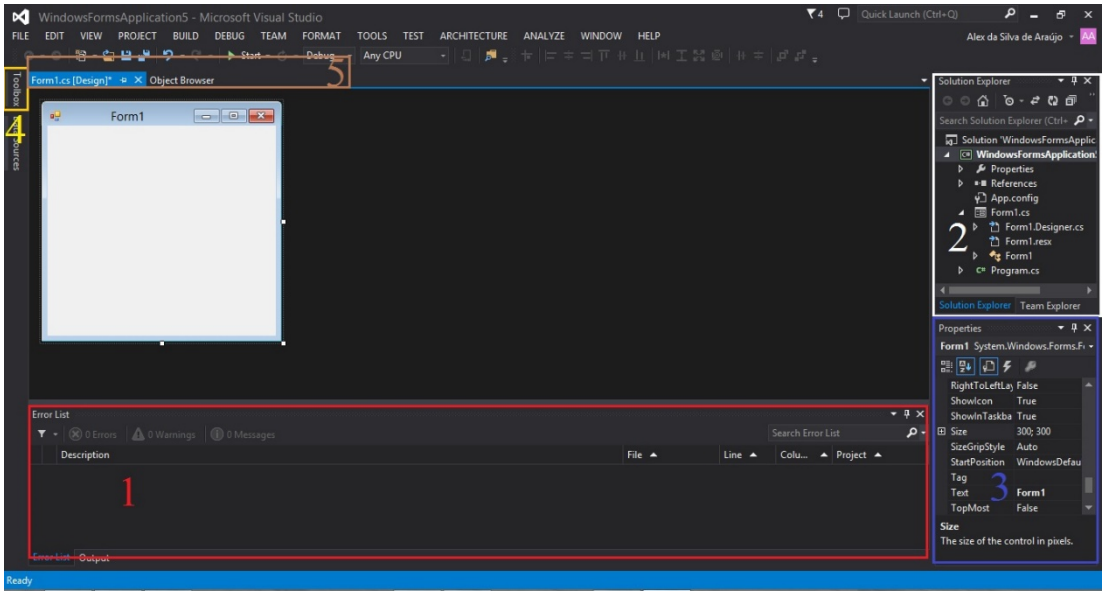
FIGURA 2.14 – AMBIENTE DE DESENVOLVIMENTO (IDE) VISUAL STUDIO.



Fonte: do autor.

A FIGURA 2.15 abaixo tem o objetivo de ilustrar as funções e mostrar o ambiente IDE Visual Studio.

FIGURA 2.15 – AMBIENTE DE DESENVOLVIMENTO (IDE) VISUAL STUDIO.



Fonte: Adaptado de Andrrew Stellman; Jennifer Greene, Use a cabeça C#

Dando sequência na criação do novo projeto Windows Forms Application a IDE retorna para o usuário uma janela em branco. A demarcação “`1” representa a depuração do código que está sendo desenvolvido e sua função é apontar e identificar erro no código. A demarcação “2” representa o *Solution Explorer* local responsável por identificar e demonstrar quais arquivos estão sendo criados em seu projeto que neste exemplo são eles: *form1.cs* e *program.cs*. A demarcação “3” representa as propriedades de cada formulário onde é possível fazer alterações e controles do que foi criado. A demarcação “4” é a toolbox que representa uma caixa de ferramentas onde é possível arrastar para a sua janela de desenvolvimentos funções visuais como botões, *checkboxs*, *textbox* entre outros. A demarcação “5” é a aba de navegação para alterar as aplicações que o usuário está desenvolvendo (STELLMAN e GREEME, 2008).

A vantagem de usar esta plataforma é a diversidade de funções pré-programadas. Em exemplo, para criar um botão não é preciso desenvolver linhas de código para criar o botão ou até mesmo para fazer o seu *layout*. Basta apenas inserir o botão que essa programação já está definida (STELLMAN e GREEME, 2008).

A ferramenta de desenvolvimento Visual Studio é robusta e completa e sua característica é apresentar para o usuário o que está sendo produzido. Outro ponto importante são as ferramentas de debug e de análise do código sendo possível não só apenas análise de erros, mas também fazer uma avaliação de performance de sua aplicação.

2.6 Banco de Dados

O uso de banco de dados nos dias atuais está presente nas mais variadas aplicações não somente a assuntos que diz respeito a Tecnologia da Informação. Outros segmentos, como o assunto deste trabalho, por exemplo supermercados, aviação, bancário, comércio de produtos e serviços, entre outra gama de diversidade de exemplos utilizam de alguma forma para armazenar suas informações de uma maneira geral (MILANE, 2006).

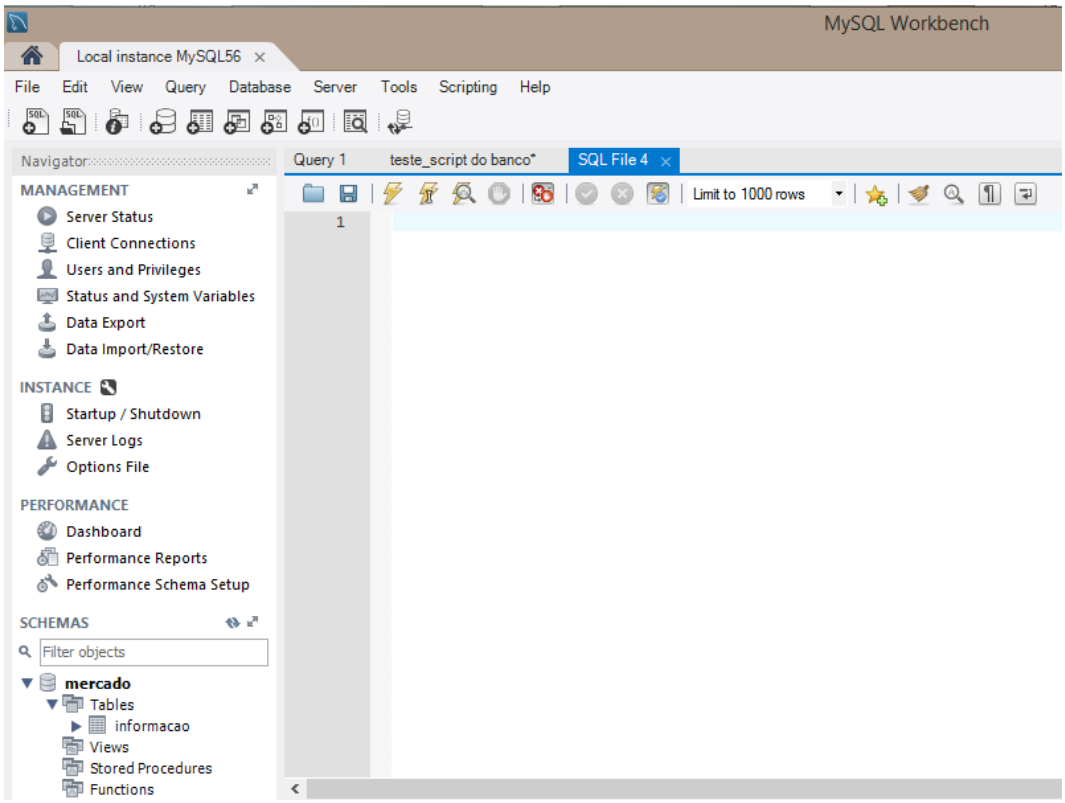
O banco de dados MySQL é um servidor completo e um SGBD (Servidor e Gerenciador de Banco de Dados) relacional que possui duas licenças disponíveis no mercado a versão paga e a versão livre que inicialmente foi projetado para atender a aplicações consideradas pequenas ou de médio porte e hoje atende vários tipos de serviços considerados de grande porte (MILANE, 2006).

O banco de dados MySQL Workbench é bem versátil, rápido e possui uma interface interativa que torna o uso mais simples e prático. Seu uso é bem difundido em universidades e o curso de banco de dados é bem aceito tanto para usuários iniciantes como avançados. Com este banco de dados é possível desenvolver modelação de dados, desenvolvimento de SQL (linguagem de banco de dados) e até mesmo configurações avançadas de administração de servidores (MYSQL, 2015).

Sua característica fundamental é o seu ambiente visual e totalmente interativo e moderno que torna o desenvolvimento e o gerenciamento de um banco de dados um processo simples desde a criação de *database*, modelação de dados até o gerenciamento de *instancias*. (MYSQL, 2015).

A FIGURA 2.16 abaixo ilustra uma parte do ambiente de desenvolvimento do MySQL Workbench.

FIGURA 2.16 – PARTE DO AMBIENTE DE DESENVOLVIMENTO (IDE) MySQL WORKBENCH.



Fonte: do autor.

O banco de dados MySQL Workbench é um ambiente interativo e simples que torna o processo de modelagem de dados uma forma rápida e será utilizado na proposta deste trabalho pois há uma necessidade em armazenar as informações do código de barras e o seu respectivo valor. Desta forma, o banco de dados estará disponível em forma de instância local, ou seja, será acessado localmente de um único computador de uma forma a representar um sistema de banco de dados de um supermercado.

3 DESENVOLVIMENTO DO TRABALHO PROPOSTO

O capítulo 3 tem por objetivo demonstrar o desenvolvimento do trabalho proposto e está organizado e dividido em etapas. A primeira parte é a apresentação de um diagrama de bloco do projeto em nível macro com as etapas do que foi produzido. Em seguida é demonstrada uma visão geral do trabalho proposto organizada por grandes blocos. Posteriormente é desenvolvido cada bloco a fim de chegar a implementação propriamente dita.

O projeto é desenvolvido em etapas e a FIGURA 3.1 abaixo ilustra sua divisão:

FIGURA 3.1 – DIAGRAMA DE BLOCO DE DESENVOLVIMENTO DO TRABALHO.

Fonte: do autor.

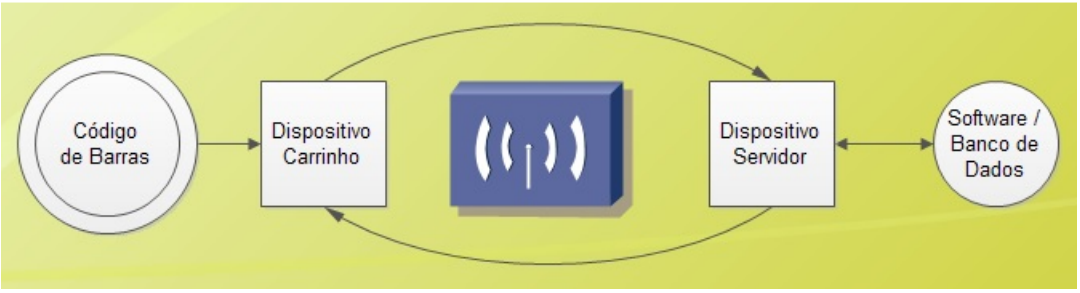
A primeira etapa desenvolvida é a de pesquisas e estudos e posteriormente a isso é elaborada toda a parte de desenvolvimento de *hardware* e em seguida a de *software*. A próxima

etapa são os testes nestas plataformas visando sua integração e por fim é montado o protótipo final.

3.1 Apresentação da proposta

A proposta deste trabalho consiste em desenvolver um sistema de *hardware* e *software* capaz de ler um código de barras e retornar o preço do produto através de uma consulta em um banco de dados. Nesse contexto, é desenvolvido dois dispositivos de *hardware*, um *software* em linguagem C# intermediador, e um banco de dados. A Figura 3.2, abaixo, apresenta o diagrama esquemático do projeto.

FIGURA 3.2 – REPRESENTAÇÃO DA PROPOSTA DE TRABALHO.

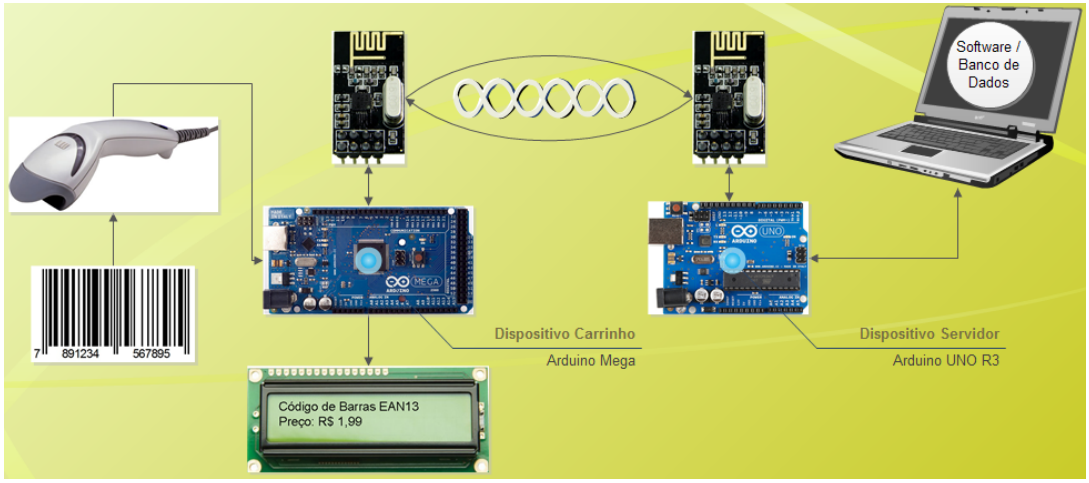


Fonte: do autor.

O código de barras é lido pelo dispositivo Carrinho e em seguida repassará essa informação via rádiofrequência para o Dispositivo Servidor que por sua vez, fará a consulta no banco de dados através do *Software* em linguagem C# para que então, a resposta seja devolvida via rádiofrequência para o primeiro dispositivo.

O processamento do Dispositivo Carrinho fica a cargo da placa Arduino MEGA que por sua vez possui um leitor de código de barras e uma tela de cristal líquido responsável por mostrar as informações em duas etapas que é informar o código que foi lido e posteriormente informar o seu equivalente preço. Já o Dispositivo Servidor tem seu processamento a cargo da placa Arduino UNO que por sua vez está conectado via USB/Serial a um computador e sua função é fazer o repasse de informações. O computador é responsável por manter a instância local do banco de dados e o *software* em C#, responsável por fazer a consulta no banco de dados, rodando em segundo plano. A FIGURA 3.3, abaixo, apresenta os componentes descritos anteriormente.

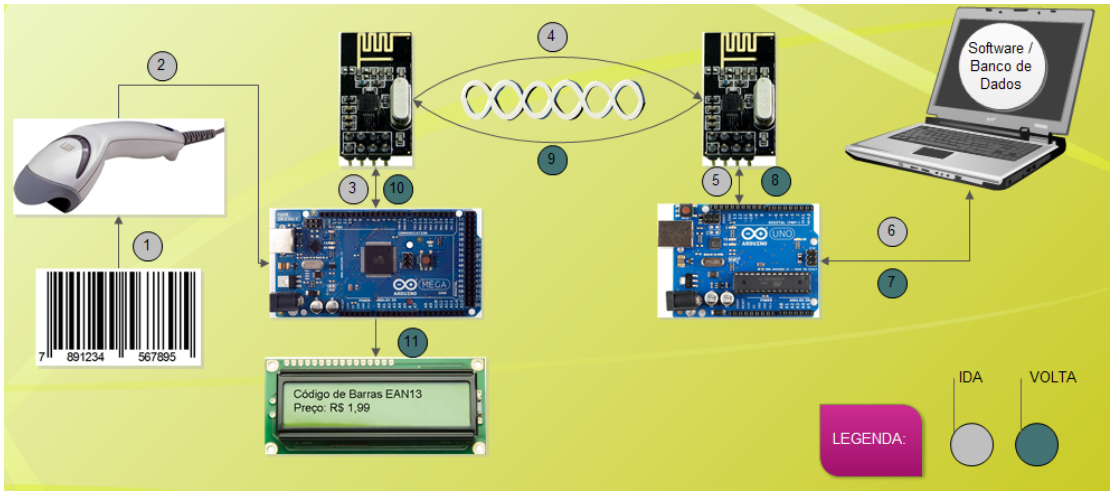
FIGURA 3.3 – IDENTIFICAÇÃO DE COMPONENTES.



Fonte: do autor.

O fluxo de informação é representado em 11 passos correlacionada aos componentes de maior importância do projeto. A FIGURA 3.4, abaixo, apresenta este processo.

FIGURA 3.4 – REPRESENTAÇÃO DO FLUXO DE INFORMAÇÃO.

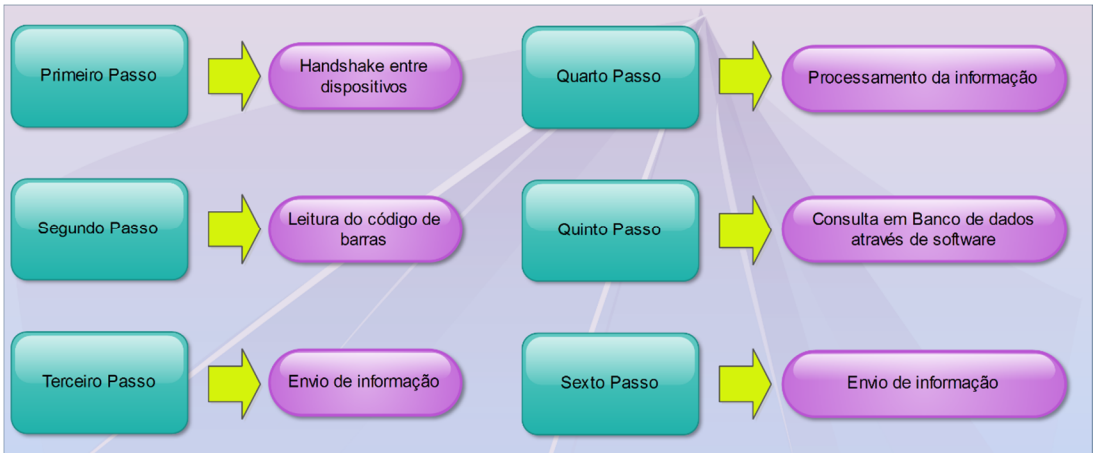


Fonte: do autor.

O círculo de cor cinza claro representa o caminho de IDA da informação, ou seja, é contado a partir do momento que o leitor de código de barras faz a varredura do código de barras e esta informação passa pelo Dispositivo Carrinho até chegar ao banco de dados. O círculo de cor verde escuro representa o caminho de VOLTA da informação, ou seja, é contado a partir do momento que a informação está saindo do banco de dados até chegar tela de LCD (*Liquid Crystal Display*) do Dispositivo Carrinho.

O início da proposta é definido quando os dois Dispositivos iniciam a comunicação. Este passo inicial tem a sua importância pois, sem esta etapa não é possível fazer a troca de informação entre os dispositivos. Bem como, é importante que o banco de dados esteja *online* em instância local e o *software* rodando em segundo plano. Diante destas informações é possível dividir em etapas o funcionamento da proposta do projeto e a FIGURA 3.5 abaixo ilustra estas etapas.

FIGURA 3.5 – ETAPAS DE FUNCIONAMENTO DO PROJETO.



Fonte: do autor.

Todas as etapas citadas na FIGURA 3.5 são em nível macro de funcionamento e seu devido detalhamento será abordado na divisão de blocos de funcionamento da proposta deste trabalho. A breve explicação abaixo se dá pela importância de cada passo, pois é necessário compreender por onde passa a informação e onde é esperado o envio, recebimento e processamento do dado ao longo do projeto.

No primeiro passo ocorre o processo de *handshake* entre os dois dispositivos que, em outras palavras é uma troca de informação inicial que caracteriza a concretização do emparelhamento na comunicação wireless entre os dois dispositivos sendo o responsável por esta função o módulo de rádiofrequência. O segundo passo, ilustra a leitura do código de barras que é responsável pela interpretação do código de barras. O terceiro passo representa o envio do código de barras do produto que está no Dispositivo Carrinho para o Dispositivo Servidor é esperado que esta mensagem não se perca e que não ocorra interferência ou *trash* somada ao pacote de mensagem tanto para o envio quanto para o recebimento explicado no próximo passo. O quarto passo demonstra o recebimento da informação que chega ao Dispositivo Servidor, que

faz a comunicação serial com o notebook. No quinto passo temos o *software* em C# recebendo a informação do Dispositivo Servidor via comunicação serial e realizando a instrução do tipo *select* via SQL no banco de dados. Por fim o sexto passo representa a resposta do *Software* C# retornando para o Dispositivo Servidor e na sequência o envio da mensagem para o Dispositivo Carrinho.

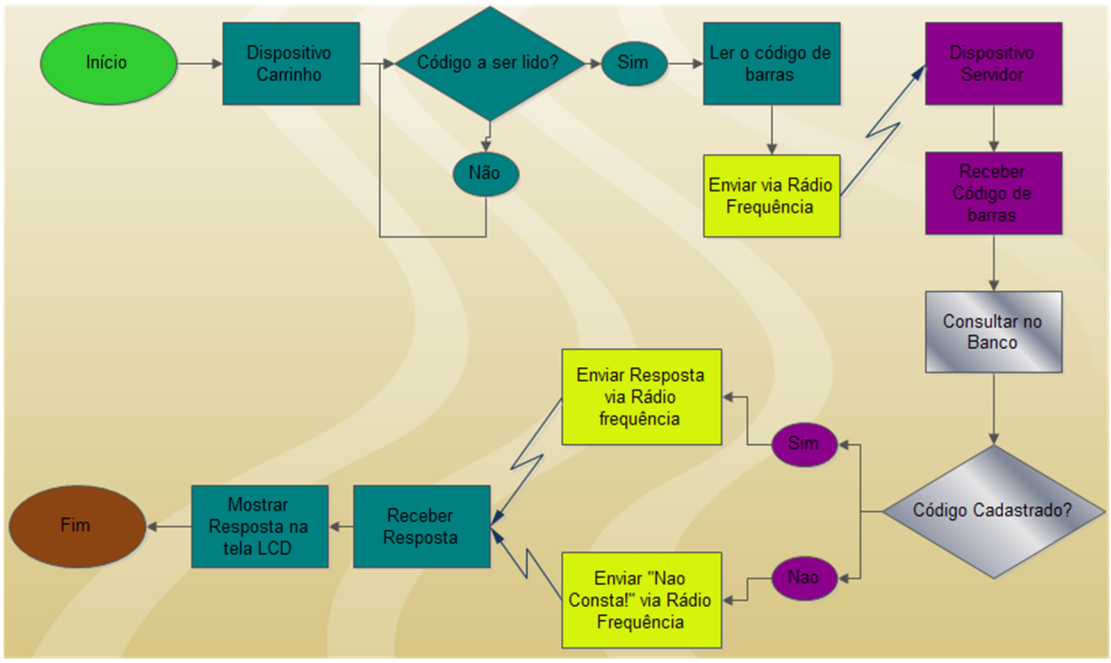
3.2 Fluxo de informação

O fluxo de informação é explicado em formato de fluxograma, como mostra a Figura 3.6. Sua função é detalhar a sequência dos processos realizados bem como as decisões tomadas por cada elemento da proposta do trabalho. É levado em consideração os seguintes Dispositivos:

a) Dispositivo Carrinho representado na cor verde escura;
b) Dispositivo Servidor representado na cor roxa;
c) Módulo de rádiofrequência nRF24L01 representado na cor amarela.
d) *Software* C# e Banco de dados representado na cor prata.

O diagrama de blocos é mostrado na FIGURA 3.6 abaixo.

FIGURA 3.6 – DIAGRAMA DE BLOCOS, PROCESSOS E DECISÕES.



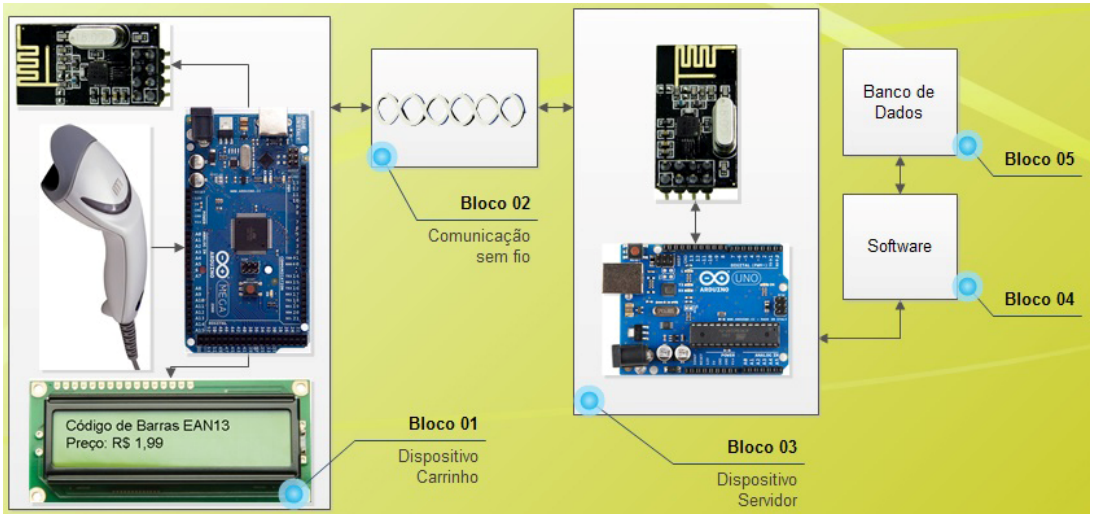
Fonte: do autor.

O processo “Código a ser lido” pertencente ao Dispositivo Carrinho representa o *loop* infinito de espera para receber um código de barras. Ou seja, a partir do momento que o Dispositivo é iniciado e após ele realizar a rotina de *handshake* o mesmo entra em rotina de espera de dados para que desta forma ele receba uma informação, processe a mesma e volte a entrar em modo de espera novamente para receber outro dado. Já o processo “Enviar via Rádiofrequência” pertencente ao módulo de rádiofrequência do Dispositivo Carrinho representa o envio de informação para o Dispositivo Servidor. O processo “Código Cadastrado” pertencente ao *software* C# e atrelado ao Banco de Dados recebe a informação e efetua a consulta no banco de dados. A resposta deste processo é se o código de barras e o preço constam ou não no Banco de dados. Caso essa informação seja verdadeira então a resposta dessa decisão (marcado de “sim” de cor roxa na figura) pertencente ao Dispositivo Servidor será o envio, para o Dispositivo Carrinho, do preço associado ao código de barras lido. Caso seja falsa essa informação, ou seja, o código de barras não esteja cadastrado no banco então a reposta da decisão (marcada de “não” de cor roxa na figura) enviará uma resposta fixa - que é a mensagem “ *Não Consta! ”* - Para o Dispositivo Carrinho. Já o processo “Mostrar Resposta na tela LCD” pertencente ao Dispositivo Carrinho é o último processo a ser realizado. Este é ponto final para o usuário que fará a consulta de um código de barras qualquer para que em seguida receba uma resposta a sua solicitação sendo esta consulta positiva (Código de barras com preço relacionado) ou negativa (Código de barras não cadastrado).

3.3 Diagrama de Blocos

A proposta desse trabalho apresenta um diagrama em blocos que será descrito e apresentado conforme ilustra a FIGURA 3.7 abaixo.

FIGURA 3.7 – DIAGRAMA DE BLOCO DE DIVISÃO DO TRABALHO.



Fonte: do autor.

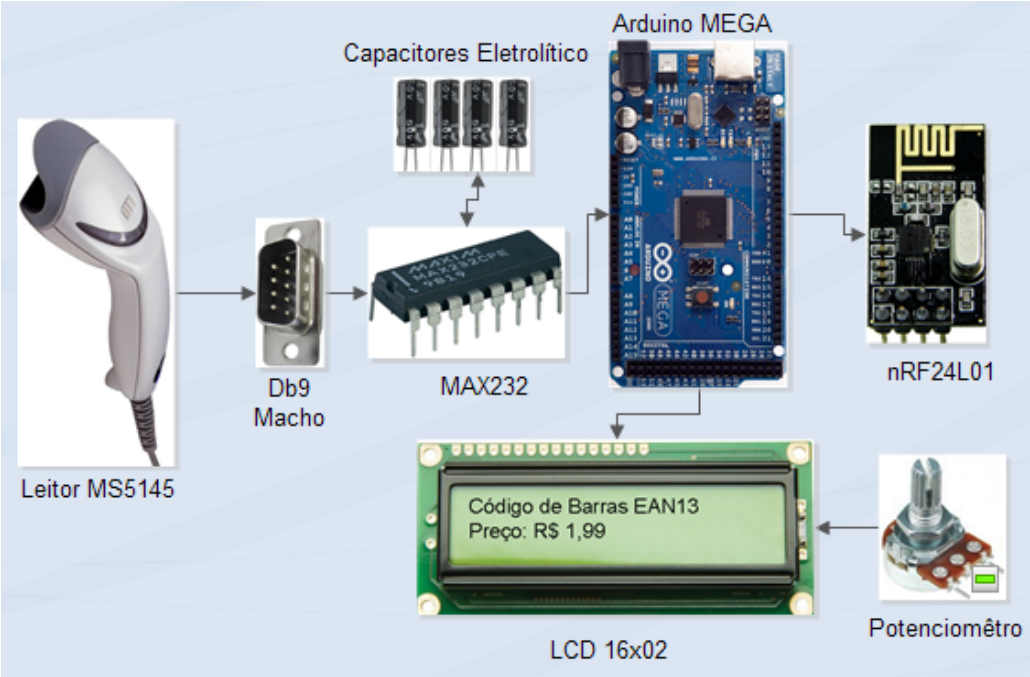
O bloco 1 é destinado ao Dispositivo Carrinho e todos os seus componentes sendo a sua função principal a leitura do código de barras e a troca de informações. O bloco 2 é associado a comunicação sem fio sendo este responsável por tratar os parâmetros, troca de mensagens, bem como a rotina de *handshake*. O bloco 3 é associado ao Dispositivo Servidor que estará conectada ao computador e sua função é o repasse e informações. O bloco 4 é destinado ao *software* C# desenvolvido com a principal função de realizar instruções do tipo *select* e *insert*. O bloco 5 é o banco de dados e sua função é armazenar o código de barras e o preço do produto equivalente.

3.4 Bloco 1 – Dispositivo Carrinho

O bloco 1 é responsável por fazer a captação do código de barras processar essa informação e enviar para o Dispositivo Servidor. Esta leitura é requisitada da placa de prototipagem Arduino MEGA para o Leitor de código de barras através da comunicação serial USART. Posteriormente a isso é criado um *buffer* temporário em formato de *Array* no intuito de ter um controle de caractere a caractere para que seja possível mostrar na tela LCD e, posteriormente, fazer o repasse dessa informação via rádiofrequência.

Os componentes utilizados no bloco 01 são representados na FIGURA 3.8, abaixo.

FIGURA 3.8 – TODOS OS COMPONENTES QUE INTEGRAM O BLOCO 1.



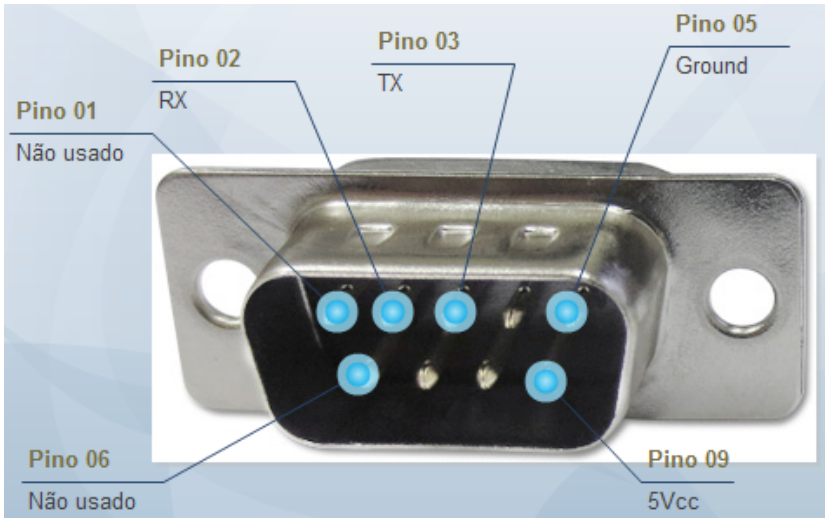
Fonte: do autor.

3.4.1 Conector Db9 Macho

O leitor de código de barras está conectado diretamente ao conector Db9 Macho que por sua vez está interligado ao max232, sendo o responsável pela conversão de sinal. Este conector se faz necessário pois ele corresponde ao padrão RS232 utilizado pelo leitor de código de barras.

O conector Db9 Macho e suas conexões utilizadas no projeto são mostradas na FIGURA 3.9, abaixo.

FIGURA 3.9 – CONECTOR DB9 MACHO E SUAS RESPECTIVAS CONEXÕES.



Legenda: os pinos sem demarcação não são utilizados neste projeto.

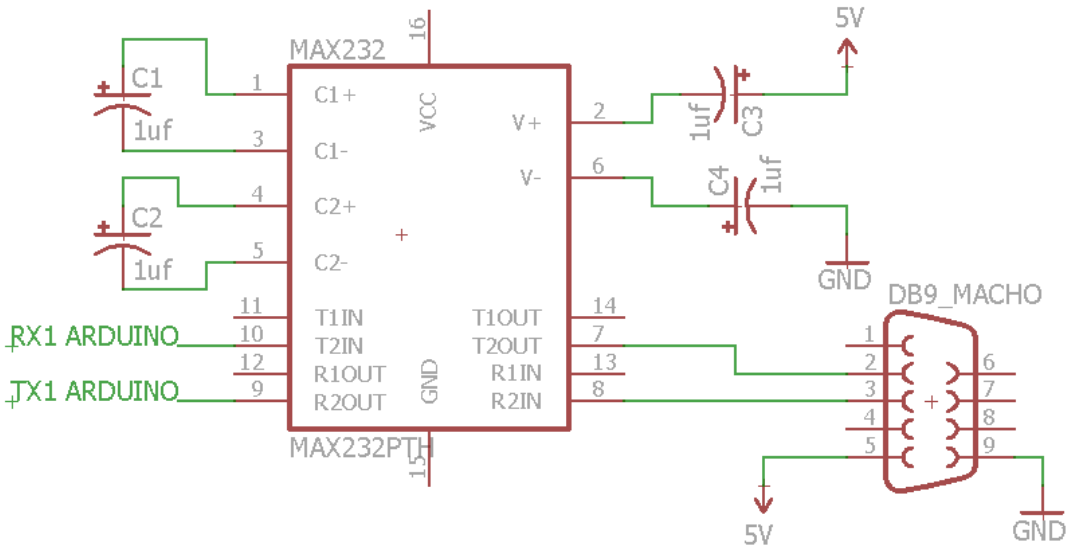
Fonte: do autor.

O pino 02 denominado de RX do conector Db9 Macho representa o recebimento de dados para o leitor de código de barras. Já o pino 03 denominado de TX representa o envio de informações do leitor para a placa Arduino MEGA. Os pinos 05 e 09 denominados de *Ground* e *Vcc* respectivamente são responsáveis pela alimentação fornecida da placa Arduino para o leitor de código de barras.

3.4.2 MAX232

O max232 é responsável por receber as conexões TX e RX do leitor de código de barras e retransmitir para os respectivos pinos TX e RX da placa Arduino MEGA. Ele se faz necessário no projeto pois é necessário converter o sinal RS-232 vindo do leitor para o padrão TTL que é o padrão reconhecido por microcontroladores em geral, que neste caso é o atmega2560 utilizado na placa Arduino MEGA. A conexão entre o DB9 Macho, max232 e Arduino Mega são representados na FIGURA 3.10, abaixo.

FIGURA 3.10 – CONEXÕES ENTRE DB9, MAX232 E PLACA ARDUINO MEGA.



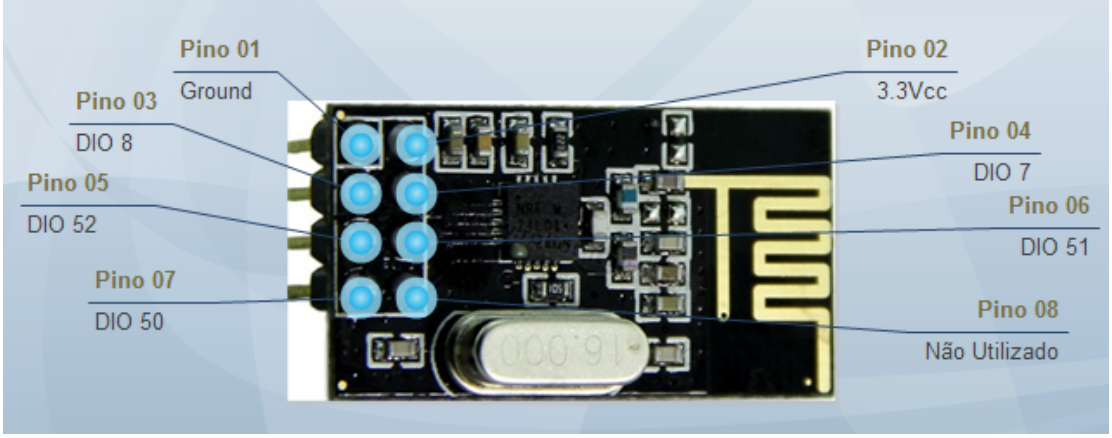
Fonte: do autor.

Como é observado na FIGURA 3.10, o max232 é alimentado com apenas 5 V no intuito de fazer a conversão de nível entre -13V/+13V (RS232) para 0 V e 5 V (TTL). Isto só é possível aos 4 capacitores incluídos nos circuitos capaz de gerar a tensão alta. Já para a tensão negativa note que os capacitores com a legenda de C3 e C4 estão ligados invertidos para tal finalidade.

3.4.3 Módulo de rádiofrequência nRF24L01

O módulo de rádiofrequência nRF24L01 se conecta na placa Arduino através dos pinos digitais do mesmo. Importante ressaltar que a tensão de alimentação deste módulo é de 3.3 V não podendo utilizar-se da alimentação convencional da placa Arduino de 5 V. As conexões entre placa Arduino MEGA e o módulo de rádiofrequência nRF24L01 são ilustradas na FIGURA 3.11, abaixo.

FIGURA 3.11 – CONEXÕES ENTRE O MÓDULO DE RÁDIOFREQUÊNCIA E PLACA ARDUINO.



Fonte: do autor.

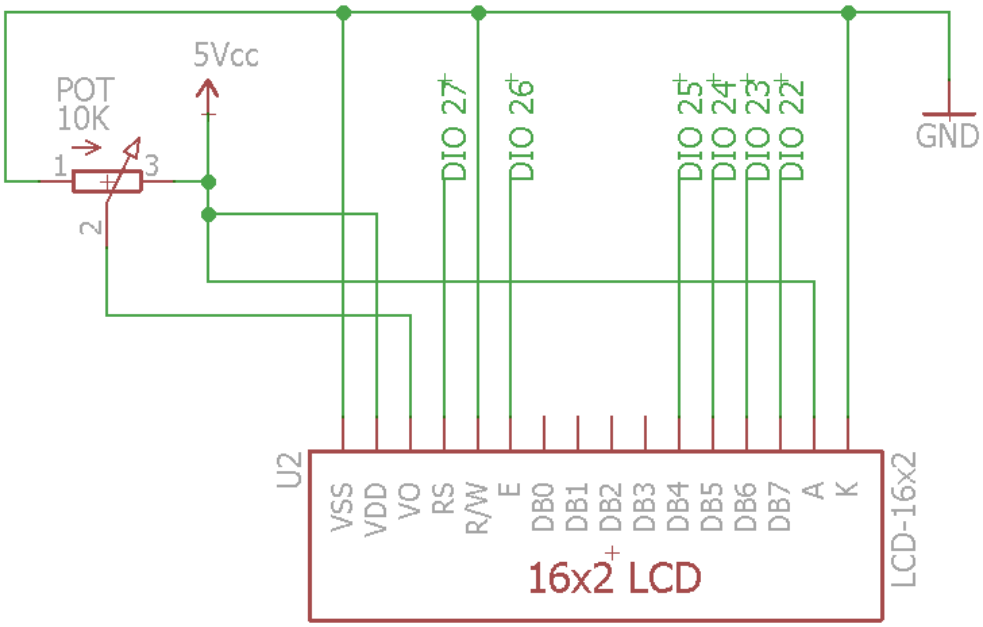
Os pinos numerados de 01 a 08 correspondem aos pinos do módulo de rádiofrequência. Já as marcações de *Ground*, *3.3V*, *DIO 8 a DIO 50* correspondem as conexões que serão utilizadas pela placa Arduino.

3.4.4 Tela de Cristal Líquido e Potenciômetro

A tela de Cristal Líquido (LCD) é responsável por mostrar os dados vindo do Arduino MEGA. A variável código de barras é apresentado na tela de forma sequencial (caractere por caractere) pois a sua escrita é realizada de forma lenta. Este método foi adotado para evitar que aconteça interferência e impressão de caracteres aleatórios. Já a variável preço é mostrado normalmente na tela LCD utilizando-se do formato *String*.

As conexões da Tela LCD e suas conexões com a placa Arduino são ilustradas na FIGURA 3.12, abaixo.

FIGURA 3.12 – CONEXÕES ENTRE A TELA LCD E A PLACA ARDUINO MEGA.



Fonte: do autor.

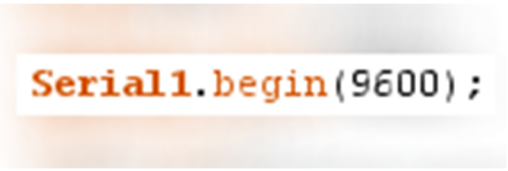
A tela LCD exibida na FIGURA 3.12 é no padrão 16 linhas por 2 colunas e sua alimentação é de 5 Volts. Para configuração e ajustes finos do contraste da tela foi adicionado um potenciômetro, identificado de POT na figura, de resistência regulável de 0 a 10K Ohms. Os pinos representados de DIO27 à DIO22 são responsáveis pela conexão com a placa Arduino MEGA.

3.4.5 Placa Arduino MEGA

O leitor de código de barras foi associado a comunicação serial denominada de RX1 e TX1 pela placa Arduino MEGA com a velocidade de troca de informações pré-definida de 9600 bits por segundo (*baud*). Como já explicado neste trabalho, o Arduino MEGA possui 3 entradas auxiliares de comunicação serial que utilizam o padrão USART. Então, através de linhas de comando, foi definido que o leitor de código de barras utilizará a porta 01.

Esta etapa é ilustrada na FIGURA 3.13, abaixo.

FIGURA 3.13 – LINHA DE COMANDO.



Legenda: código referente a indexação do leitor de código de barras a porta de comunicação serial 01 da placa Arduino Mega.

Fonte: do autor.

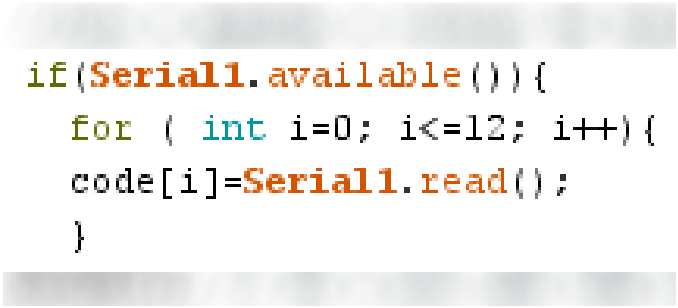
Para fazer a leitura do código de barras é necessário criar um *Array* do tipo *char* com 13 posições fixas. Desta forma é garantido que cada caractere do código de barras EAN 13 estará atribuído a uma posição fixa.

A placa Arduino possui um método chamado *Serial.available()* pertencente a sua biblioteca interna. Este método não possui nenhum parâmetro e o seu retorno é o dado obtido caso este exista em uma porta serial. Sua implementação é atrelada ao comando *if else* que tem a função de verificar se existe ou não um dado disponível para leitura. Esta etapa pertence ao *loop* infinito de espera para ler algum código de barras.

Caso o comando *if else* verificar que a condição seja verdadeira, ou seja, exista algum dado disponível na porta serial é feita então uma leitura sequencial através da função *for* utilizando o método *Serial.read()*, pertencente a sua biblioteca interna, para leitura de dados. Esta parte da implementação corresponde a obtenção definitiva do código de barras vindo do leitor através da comunicação serial. Caso o comando *if else* verificar que a condição seja falsa, ou seja, não existe um dado disponível para leitura este processo então não irá retornar nada e voltará a entrar em *loop* até que algum dado esteja disponível para leitura na porta serial.

A FIGURA 3.14 abaixo representa esta implementação em linhas de comando.

FIGURA 3.14 – IMPLEMENTAÇÃO PARA LEITURA DO CÓDIGO DE BARRAS.



Fonte: do autor.

3.5 Bloco 2 – Comunicação sem fio

O envio e recebimento de mensagens bem como a rotina de *handshake* estará a cargo do bloco 2 denominado de comunicação sem fio. Este bloco será responsável por mostrar as linhas de comando para enviar e receber mensagens que é o mesmo processo utilizado nos dois Dispositivos.

O código fonte utilizado neste procedimento foi modificado da biblioteca *nRF24L01p* em que é adaptado o código exemplo *tranceivers* onde o *PRX* é o recebedor e o *PTX* é o transmissor de dados (ELETRÔNICA ARDUINO E ANDROID, 2014).

3.5.1 Parâmetros

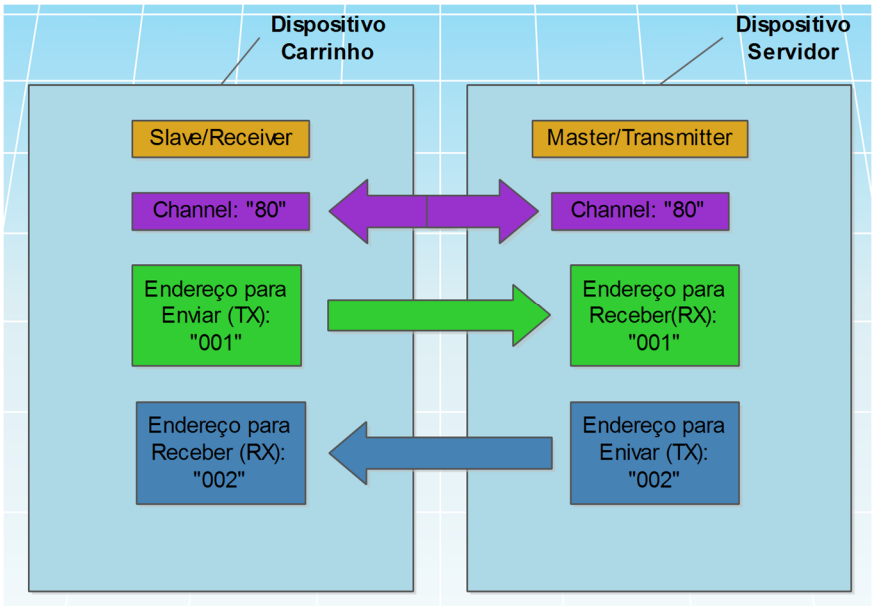
A interface de comunicação utilizada entre o módulo de rádiofrequência e a placa Arduino é a SPI (Serial Peripheral Interface) como já foi explicado neste trabalho. O método de comunicação utilizado é *Half-duplex*, ou seja, os dois Dispositivos *Master* e *Slave* poderão enviar e receber mensagens, porém não simultaneamente. Os pinos responsáveis do módulo de rádiofrequência por fazer o envio ou o recebimento de uma mensagem são os pinos MOSI e MISO explicados abaixo:

a) MOSI (*Master Output Slave Input*) – Pino responsável por mandar a mensagem do Dispositivo *Master* para o *Slave*.
b) MISO (*Master Input Slave Output*) - Pino responsável por mandar a mensagem do Dispositivo *Slave* para o *Master*.

Apesar dos dois Dispositivos trocarem mensagens é preciso definir qual dispositivo será o *Master* e qual será o *Slave*. Para este projeto foi definido que o Dispositivo Servidor será o Master (tratado como *transmitter*) e o Dispositivo *Slave* (tratado como *receiver*) será o Dispositivo Carrinho.

Em linhas de comando é necessário referenciar os endereços físico de cada dispositivo que deve possuir um tamanho entre 3 a 5 bytes. Um atributo importante para a comunicação funcionar nos dois sentidos (*Half-duplex*) é atribuir alguns parâmetros ilustrado na FIGURA 3.15, abaixo.

FIGURA 3.15 – ENDEREÇOS ATRIBUÍDOS PARA ENVIAR E RECEBER MENSAGENS.



Fonte: do autor.

É fundamental atribuir que o endereço de envio (TX) do Master seja o mesmo endereço para receber (RX) no Dispositivo *Slave*. E que o caminho inverso esteja também neste mesmo padrão, não obstante de que os Dispositivos estejam no mesmo canal de comunicação que também é pré-definido em suas configurações iniciais. Desta forma é possível criar uma conexão simultânea para troca de mensagens em um canal exclusivo para os dois Dispositivos.

3.5.2 **Rotina de *handshake***

A rotina de *handshake* foi desenvolvida com a função de iniciar a comunicação entre os dois Dispositivos. Esta rotina funciona através do envio de uma mensagem do Dispositivo *Master* para o Dispositivo *Slave*. Após este procedimento, a troca de mensagem é habilitada.

O módulo de rádiofrequência nRF24L01, como já explicado anteriormente, é um módulo transceptor em que tanto o Dispositivo *Master* quanto o Slave têm o poder de enviar e receber mensagens. Isso só é possível graças as funções internas do módulo denominadas de *TX Mode* e *RX Mode.* O Dispositivo que foi denominado de *Slave* é iniciado em modo RX para receber mensagens. Quando há algum dado a ser enviado ele altera o seu estado para o modo TX para enviar a mensagem colocando um dado na fila. O mesmo acontece para o Dispositivo que foi denominado de Master. O mesmo inicia em modo TX para enviar mensagens. Quando não há nenhum dado a ser enviado o módulo entra em modo de *standby* para aguardar o envio de alguma mensagem e ao mesmo tempo fica disponível para também receber mensagens.

A rotina de *handshake*, bem como a troca de mensagens e alteração de estado de RX para TX é representado na FIGURA 3.16, abaixo.

FIGURA 3.16 – ROTINA DE HANDSHAKE E ALTERAÇÃO DE ESTADO ENRE OS MODOS RX E TX.

Fonte: do autor.

Quando os dois Dispositivos são iniciados o *Master* aguarda 3 segundos antes de mandar a primeira mensagem para o *Slave*. Para este projeto foi definido que esta mensagem será a “Diga-me o código” no formato *string*. Após o envio desta mensagem, e caso o Dispositivo Slave a receba com sucesso, a rotina de *handshake* é concluída e então os dois Dispositivos entram em modo de espera para que seja possível um escutar o outro. Esta rotina tem a sua importância no projeto pois desta forma é possível excluir a possibilidade de erro na comunicação inicial dos Dispositivos.

3.5.3 Envio e Recebimento de mensagens

O módulo de rádiofrequência envia mensagens de tamanho de carga (*payload)* entre 0 a 32 bytes fixos a ser definido pelo microcontrolador sendo este, o exemplo de tamanho de carga estática. Porém, o módulo permite também trabalhar com cargas dinâmicas em que a mensagem possui um tamanho variado. Para este projeto foi adotado a carga dinâmica pois é levado em consideração que o preço do código de barras cadastrado possui valores diferentes no banco de dados.

Para enviar uma mensagem no Dispositivo Carrinho (Slave ou receiver), em linhas de comando, o procedimento é simples pois o microcontrolador utiliza o que está gravado na variável e envia via rádiofrequência. Seja essa mensagem do tipo *string* ou *char*. A FIGURA 3.17 abaixo ilustra este procedimento.

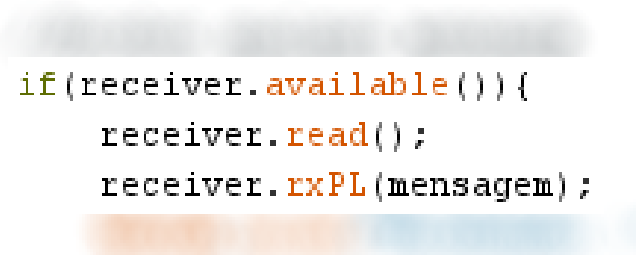
FIGURA 3.17 – LINHAS DE COMANDO PARA ENVIO DE MENSAGENS DO DISPOSITIVO CARRINHO.

```
receiver.txPL(code);
receiver.send(SLOW);
```

Fonte: adaptado de nrf24l01p. (ELETRÔNICA ARDUINO E ANDROID, 2014)

O recebimento de uma mensagem no Dispositivo Carrinho (*Slave* ou *receiver*) funciona similar a rotina de espera de um código de barras explicado anteriormente. Ou seja, foi implementando também uma função *if else* para verificar se existe alguma mensagem na fila a ser lida. Caso seja verdadeira, ou seja, exista alguma mensagem para ser recebida. O método *receiver.available(),* que não possui nenhum parâmetro, é acionado e o seu retorno é a mensagem que deve ser lida. O método *receiver.read(),* que não possui nenhum parâmetro também, tem a sua função de retirar a mensagem da fila para ser lida. Já o método *receiver.rxPL(mensagem)* tem a função de armazenar o dado na variável *mensagem* do tipo *string*. Este procedimento é ilustrado na FIGURA 3.18.

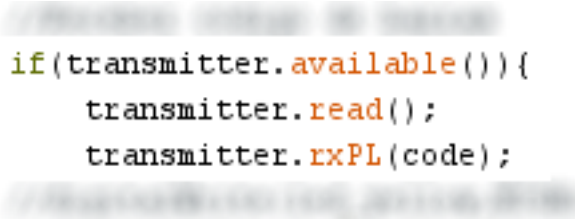
FIGURA 3.18 – LINHAS DE COMANDO PARA RECEBER UMA MENSAGEM NO DISPOSITIVO CARRINHO.



Fonte: adaptado de nrf24l01p. (ELETRÔNICA ARDUINO E ANDROID, 2014)

De forma similar, o Dispositivo Servidor (*Master* ou *transmitter*) possui o mesmo procedimento para receber uma mensagem. A função utilizada é a mesma do Slave, porém é alterado sua referência na chamada do método que agora passa a ser *transmitter*. Este processo é ilustrado na FIGURA 3.19, abaixo.

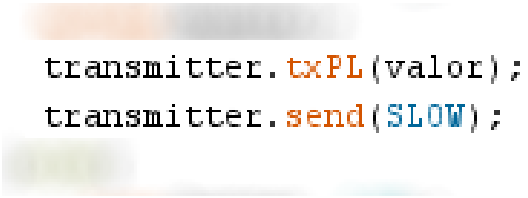
FIGURA 3.19 – LINHAS DE COMANDO PARA RECEBER UMA MENSAGEM NO DISPOSITIVO SERVIDOR.



Fonte: adaptado de nrf24l01p. (ELETRÔNICA ARDUINO E ANDROID, 2014)

O mesmo procedimento para enviar uma mensagem no Dispositivo Servidor (Master ou *transmitter*) se mantém similar ao Dispositivo Carrinho. O que irá diferenciar é que a função *if else* fará o controle de qual mensagem enviar seja o valor correspondente ao código de barras cadastrado ou a resposta de que não consta o valor referente ao código de barras. Caso a função *if else* verifique que há o valor disponível para enviar então o método *transmitter.txPL(valor)* é acionando conforme a FIGURA 3.20 que ilustra este processo.

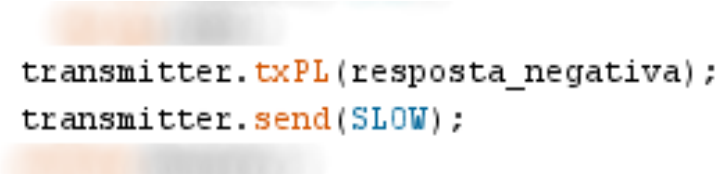
FIGURA 3.20 – LINHAS DE COMANDO DO DISPOSITVO SERVIDOR.



Legenda: envio do preço do produto referente ao código de barras.
Fonte: adaptado de nf24l01p. (ELETRÔNICA ARDUINO E ANDROID, 2014)

Caso a função *if else* retorne que não há valor cadastrado então o método *transmitter.txPL(resposta_negativa)* é acionado para transmitir a resposta "não consta" pré-definida. A FIGURA 3.21 abaixo ilustra este processo.

FIGURA 3.21 – LINHAS DE COMANDO DO DISPOSITIVO SERVIDOR.



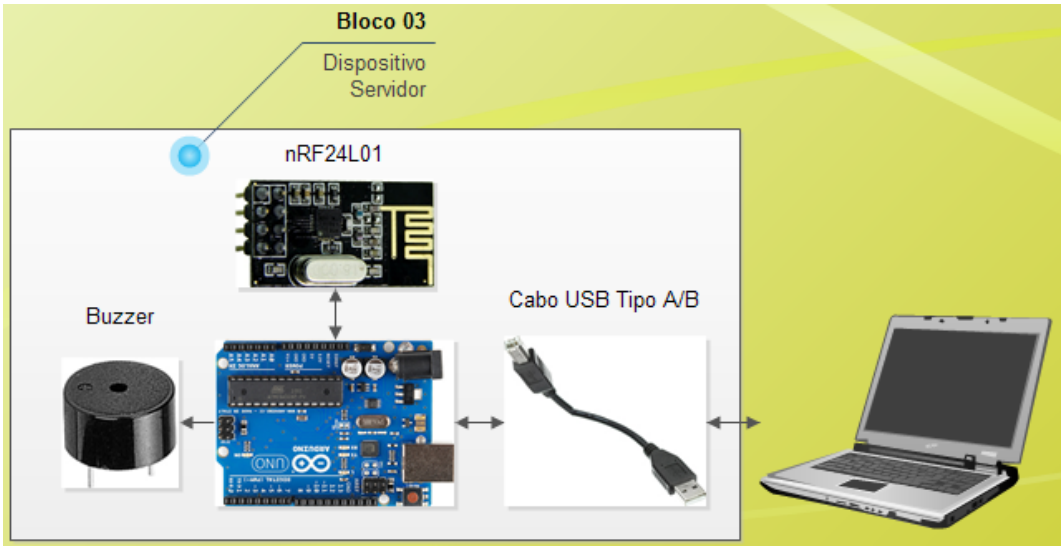
Legenda: envio da resposta "Não consta".
Fonte: adaptado de nrf24l01p. (ELETRÔNICA ARDUINO E ANDROID, 2014)

3.6 Bloco 3 – Dispositivo Servidor

O bloco 3 denominado de Dispositivo Servidor tem a função de receber o código de barras via rádiofrequência, criar um buffer em formato de *Array* afim de armazenar este dado e posteriormente a isso, se comunicar através da porta serial com o computador no intuito de repassar esta informação. Logo em seguida o Dispositivo servidor faz análise da resposta do computador e efetua o envio desta informação, via rádiofrequência, para o Dispositivo Carrinho.

Todos os componentes que integram o Bloco 3 são exibidos na FIGURA 3.22 abaixo.

FIGURA 3.22 – TODOS OS COMPONENTES QUE INTEGRAM O BLOCO DISPOSITIVO SERVIDOR.



Fonte: do autor.

3.6.1 Buzzer

O *buzzer* é um alto-falante que emite um som controlado por um sinal oriundo de um microcontrolador, neste caso o atmega328p da placa de prototipagem Arduino UNO. Os parâmetros a serem passados são a frequência e tempo deste som.

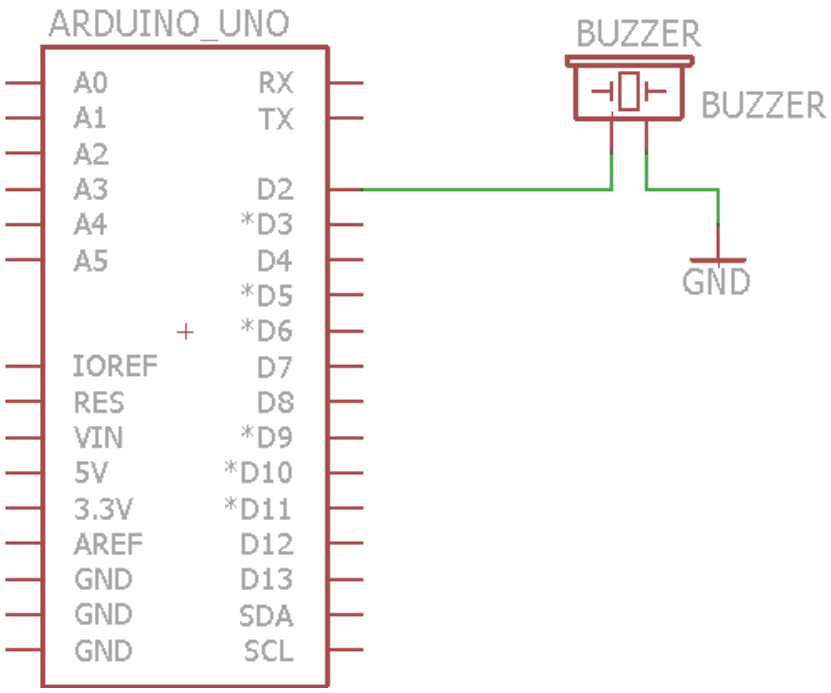
Sua função neste projeto é emitir um aviso sonoro em três cenários:

a) O primeiro cenário é alertar que a mensagem originada do Dispositivo Carrinho foi recebida com sucesso;
b) O segundo cenário é alertar que o envio da mensagem com o preço do código de barras foi efetuado com sucesso;
c) O terceiro cenário é alertar o envio da mensagem “Não consta”.

As diferenciações destes sons estão em sua frequência. O primeiro caso foi adotado que o som possui uma frequência de 1500Hz (representando um tom mais agudo) enquanto o segundo caso possui apenas 250Hz (representando um tom mais grave). Já para o terceiro caso foi adotado um som com a frequência de 0Hz (representando um som distorcido). Desta forma é possível fazer a diferenciação e monitoramento com o auxílio de alertas sonoros.

A sua conexão com a placa Arduino UNO é exibida na FIGURA 3.23 abaixo.

FIGURA 3.23 – CONEXÃO ENTRE BUZZER E PLACA ARDUINO UNO.



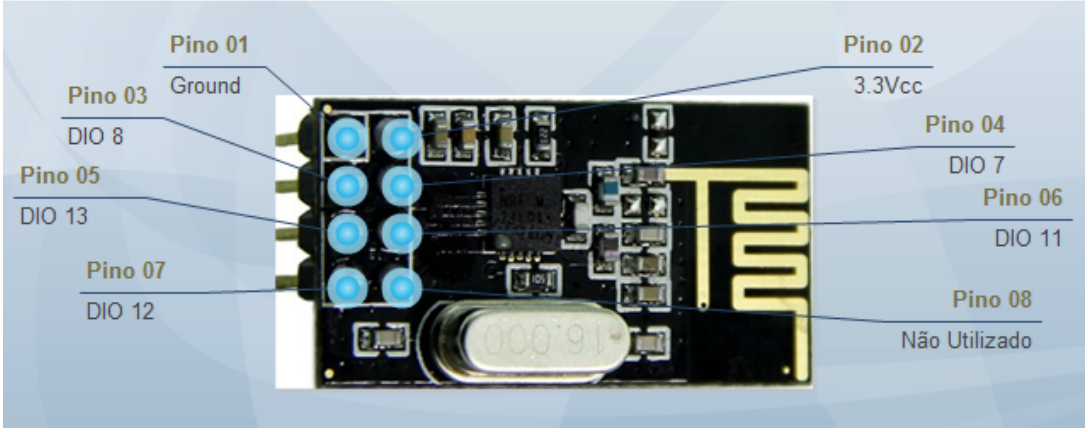
Fonte: do autor.

3.6.2 Módulo de rádiofrequência

O módulo de rádiofrequência utilizado no Dispositivo Servidor é similar ao utilizado no Dispositivo Carrinho. O que diferencia é o modo de conexão com a placa Arduino UNO que utiliza outras portas digitais para comunicação.

As portas utilizadas para comunicação entre o módulo de rádiofrequência nRF24L01 e a placa Arduino UNO são exibidos na FIGURA 3.24, abaixo.

FIGURA 3.24 – CONEXÕES ENTRE O MÓDULO DE RÁDIOFREQUÊNCIA E A PLACA ARDUINO UNO.



Fonte: do autor.

De forma similar ao módulo de rádiofrequência utilizado no Dispositivo Carrinho os pinos denominados de 01 a 08 pertencem ao módulo de rádiofrequência nRF24L01 e os pinos demarcados de *Ground*, *3.3Vcc*, DIO 8 a DIO 12 pertencem a placa Arduino UNO.

3.6.3 Placa Arduino UNO

O Dispositivo Servidor após receber o código de barras via rádiofrequência é necessário se comunicar com o computador que está conectado via cabo USB/Serial. Esta conexão é feita através da comunicação serial com o padrão de comunicação USART, já explicado neste trabalho. De forma similar ao Dispositivo Carrinho é necessário definir a porta serial de comunicação entre a placa Arduino UNO e o computador que neste caso é a única porta padrão RX e TX. Em linhas de comando é necessário declarar esta comunicação bem como a taxa de transferência definida em *bits* por segundo. A FIGURA 3.25 ilustra este processo.

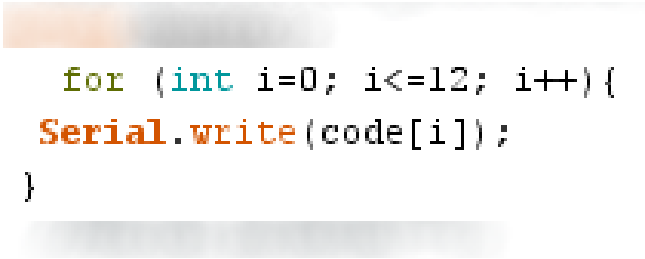
FIGURA 3.25 – LINHA DE COMANDO ATRIBUÍDO A PLACA ARDUINO UNO.

Legenda: declaração de comunicação serial.

Fonte: do autor.

O próximo passo é efetuar o repasse de informação para a porta serial do computador. Diante disso, é necessário que este repasse seja feito caractere por caractere afim de evitar a perca do pacote de informações bem como a interferência de caracteres aleatórios. Para tal finalidade o método utilizado é o *Serial.write()* e o parâmetro a ser passado é o próprio *Array* com o código de barras armazenado. A FIGURA 3.26 abaixo ilustra este processo.

FIGURA 3.26 – LINHAS DE COMANDO REFERENTE AO REPASSE.



Legenda: repasse de informação entre a placa Arduino Uno para o computador.

Fonte: do autor.

Ao fazer o repasse desta informação para o computador o Dispositivo Servidor precisa aguardar a sua resposta. Neste ponto, foi implementado a função *if else* para que seja possível fazer o controle desta resposta. Caso tenha uma resposta originada do computador, a função retorna esta mensagem via rádiofrequência. Caso aconteça de a resposta ser o que o produto não esteja cadastrado ou então uma falha seja por tempo ocioso, tempo limite de resposta ou outro erro, a função retorna a mensagem pré-definida “Não Consta” via rádiofrequência. O método utilizado é o *Serial.available()* para avaliar e esperar a resposta do computador. Esta implementação é exibida na FIGURA 3.27.

FIGURA 3.27 – LINHAS DE COMANDO REFERENTE A IMPLEMENTAÇÃO DA FUNÇÃO IF ELSE.

```
if (Serial.available()){
  transmitter.txPL(valor);
  transmitter.send(SLOW);
}else{
transmitter.txPL(resposta_negativa);
transmitter.send(SLOW);
}
```

Fonte: do autor.

3.7 Bloco 4 - *Software*

O bloco 4 é um *software* que foi desenvolvido na linguagem de programação C# e para este projeto foi atribuído o nome de “Conexão MySQL”. A sua função é receber um dado na porta serial originado da placa Arduino UNO para que logo em seguida efetue a instrução *select* no banco de dados com instância local. Após efetuar o comando *select* o *software* devolve imediatamente a informação para a porta serial do computador e a próxima etapa é o Dispositivo Servidor fazer a leitura e o processamento desta informação.

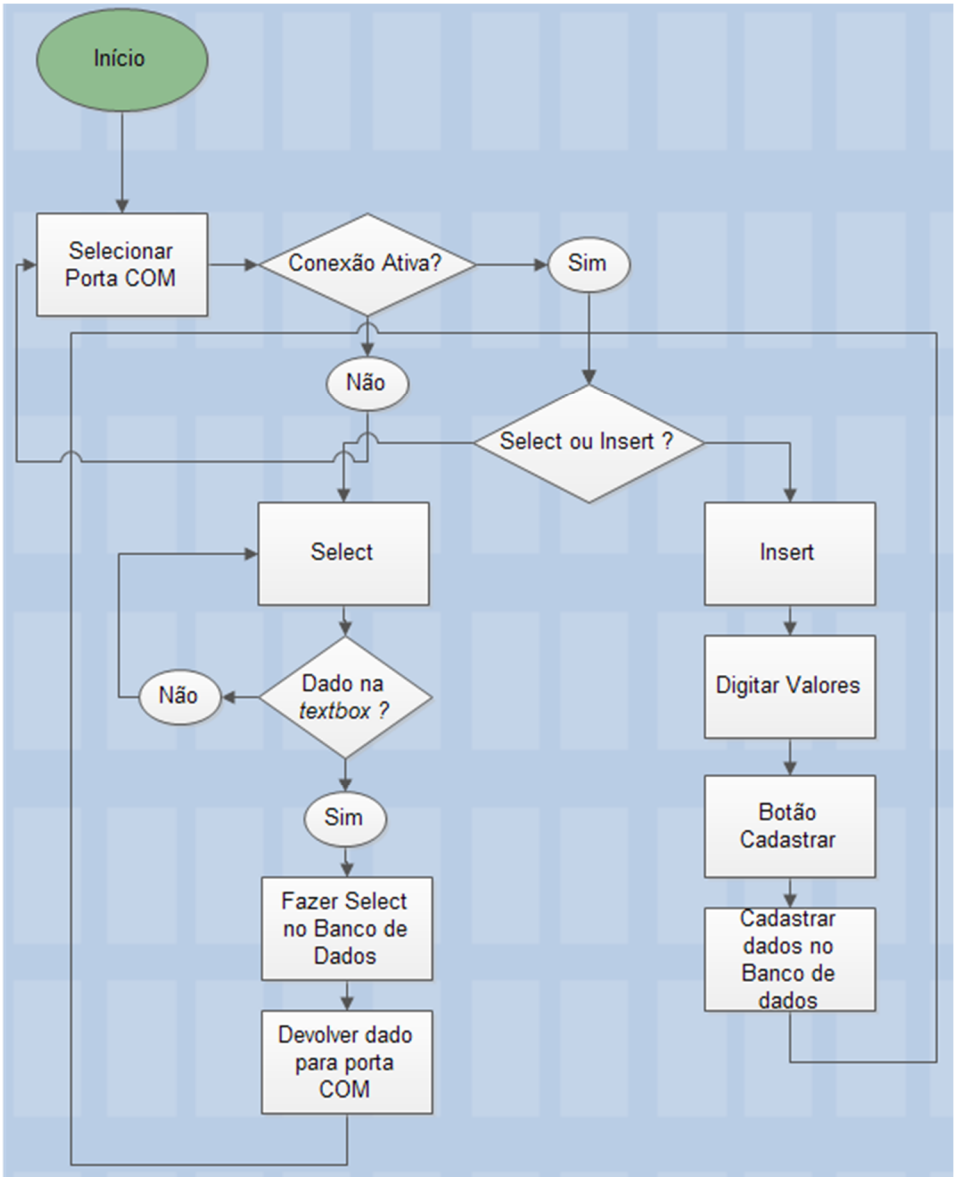
Para este projeto foi adotado que o *software* rodará em segundo plano realizando a instrução *select* e no caso de acontecer algum erro o *software* não fará nenhum tratamento, apenas irá mostrar para o usuário qual erro foi gerado. A instrução *insert* também foi implementada no *software*, e sua função foi auxiliar no desenvolvimento deste trabalho. No *software* é possível, a qualquer momento, fazer a alteração entre os modos *select* e *insert* através de uma seleção simples.

Ao iniciar o programa pela primeira vez é necessário preencher as informações necessárias para se fazer a conexão com o banco de dados. Há um campo (ou *textbox*) denominado de “*conexão”* e os parâmetros a serem passados são explicados logo a seguir:

a) Servidor: Endereço de IP que o banco de dados está instanciado.
b) *Database*: Nome atribuído a base de dados (ou *schema*).
c) *Userid*: Nome atribuído ao usuário principal do *software* de banco de dados.
d) *Password*: Senha atribuída ao usuário para acesso ao banco de dados.

O funcionamento completo do *software* bem como o seu diagrama de fluxo de informação do *software* é exibido na FIGURA 3.28.

FIGURA 3.28 – DIAGRAMA DE FLUXO DE INFORMAÇÃO DO *SOFTWARE*.



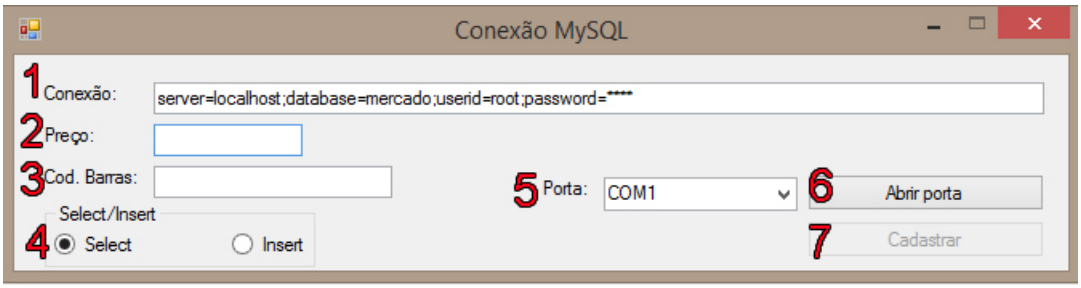
Fonte: do autor.

Ao iniciar o *software* o primeiro passo é localizar qual a porta COM a qual a placa Arduino UNO está se comunicando com o computador. Feito isso o *software* cria automaticamente uma comunicação com esta porta, caso ela exista e esteja disponível. A próxima etapa é fazer a seleção da chave para o método *select* ou *insert*. A função *insert* no *software*, funciona de maneira manual, ou seja, é preciso que o usuário insira os dados (código

de barras e valor correspondente) para que então clique no botão “Cadastrar”. Ao clicar neste botão o *software* faz a conexão com o banco de dados e executa a *query* correspondente em que faz a inserção na tabela referenciando as colunas código e valor. Já a função *select* espera o dado vir da placa Arduino UNO, via comunicação serial, para então fazer o devido *select* no banco de dados. Para esta função funcionar de forma automática é realizado um método que executa, repetidamente, a função *select* na *textbox* referente ao código de barras possuindo um dado disponível ou não.

A FIGURA 3.29 ilustra a tela única do *software*:

FIGURA 3.29 – TELA ÚNICA DO *SOFTWARE* DESENVOLVIDO.



Fonte: do autor.

As demarcações enumeradas de 1 a 7 são explicadas logo abaixo:

a) 1 – Parâmetros a serem preenchidos pelo usuário para fazer conexão com o banco de dados. (Servidor; base de dados; Usuário e senha).
b) 2 – Campo (ou *textbox*) destinado ao preço do produto.
c) 3 – Campo (ou *textbox*) destinado ao código de barras.
d) 4 – Seleção entre instruções de *select* ou *insert.*
e) 5 – Seleção de porta serial para comunicação com a placa Arduino UNO.
f) 6 – Botão para iniciar a comunicação com a placa Arduino UNO.
g) 7 – Botão para cadastrar manualmente algum produto no banco de dados. Este botão só fica disponível ao selecionar a função *insert.*

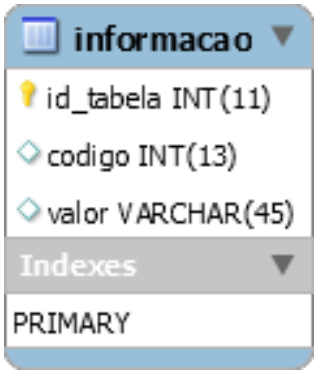
3.8 Bloco 5 – Banco de dados

O bloco 05 denominado de banco de dados é a base de dados de todo o projeto. Nele consta a instância do servidor funcionando localmente para simular um ambiente de uma rede de supermercados bem como uma tabela simples para armazenar as informações de código de barras e o seu respectivo valor.

Foi determinado o índice da tabela como um valor inteiro de 11 caracteres. Este valor representa um índice de chave primária da tabela do banco de dados. Já a coluna denominada de "*código*" foi atribuída o valor inteiro de 13 caracteres já que este é o responsável por armazenar o código de barras do tipo EAN13. E a coluna denominada de "*valor*" foi atribuído a variável do tipo *varchar*, que representa os caracteres variados, pois o campo valor possui caracteres especiais como "R$" e ", " além de representar os números de 0 a 9. O tamanho máximo de 45 caracteres foi atribuído pois existem uma gama de variação de preços de produtos em um supermercado.

O modelo da tabela de dados desenvolvida é representado na FIGURA 3.30, abaixo.

FIGURA 3.30 – TABELA REFERENTE AO BANCO DE DADOS DESENVOLVIDO.



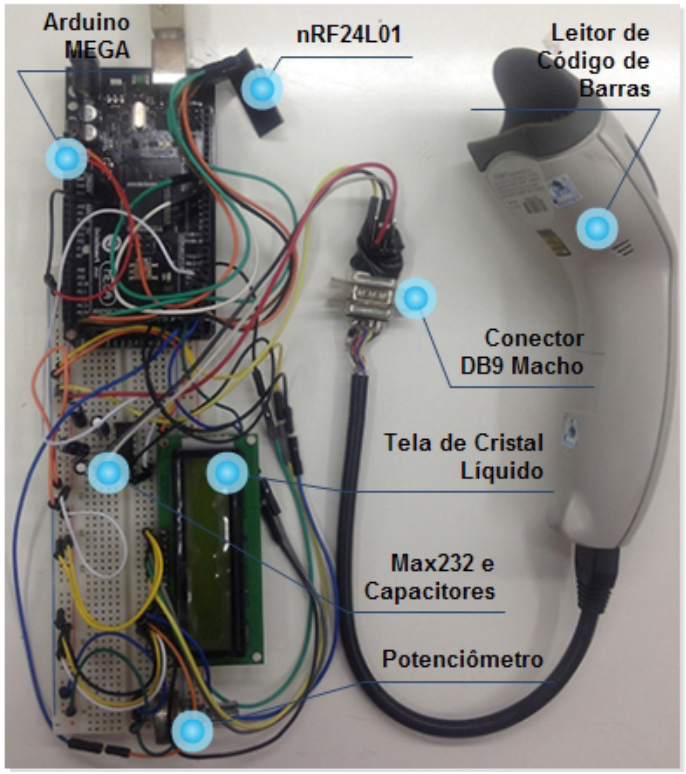
Fonte: do autor.

3.9 Montagem do protótipo final

A montagem do protótipo foi desenvolvida em duas etapas a fim de se obter uma prototipação e validação da proposta deste trabalho. A primeira etapa desenvolvida foi a montagem do protótipo em uma placa de ensaio (ou *protoboard*) em que os componentes não são soldados. Esta parte tem a sua importância pois desta forma foi possível testar todos os componentes que compõem este projeto.

O Dispositivo Carrinho em montagem de placa de ensaio bem como todos os componentes que o integram são exibidos na FIGURA 3.31 abaixo.

FIGURA 3.31 – DISPOSITIVO CARRINHO EM PLACA DE ENSAIO.

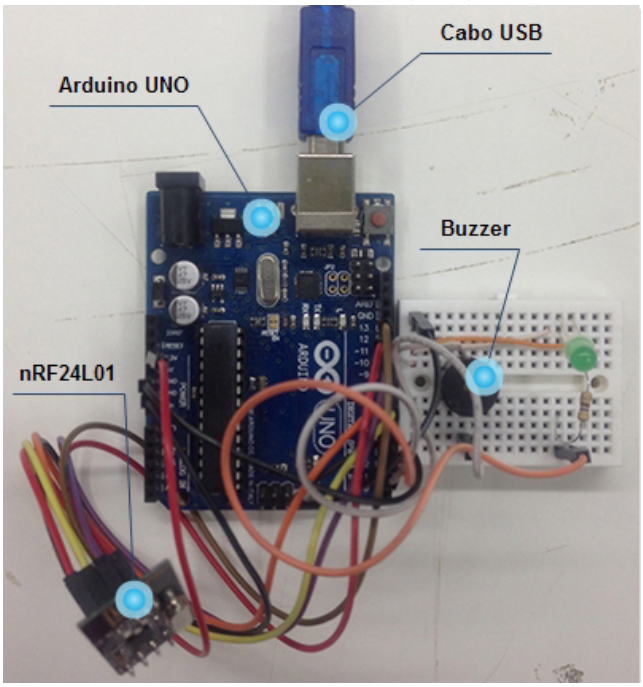


Fonte: do autor.

Na FIGURA 3.31 é possível observar o leitor de código de barras conectado ao *hardware* através do conector DB9 Macho. Na sequência é observado o circuito integrado max232 conectado aos capacitores. A figura ilustra também os outros componentes como a tela LCD, potenciômetro, módulo de rádiofrequência e pôr fim a placa de prototipagem Arduino Mega.

O Dispositivo Servidor bem como todos os componentes que o integram montados em uma placa de ensaio é exibido na FIGURA 3.32 abaixo.

FIGURA 3.32 – DISPOSITIVO SERVIDOR EM PLACA DE ENSAIO.



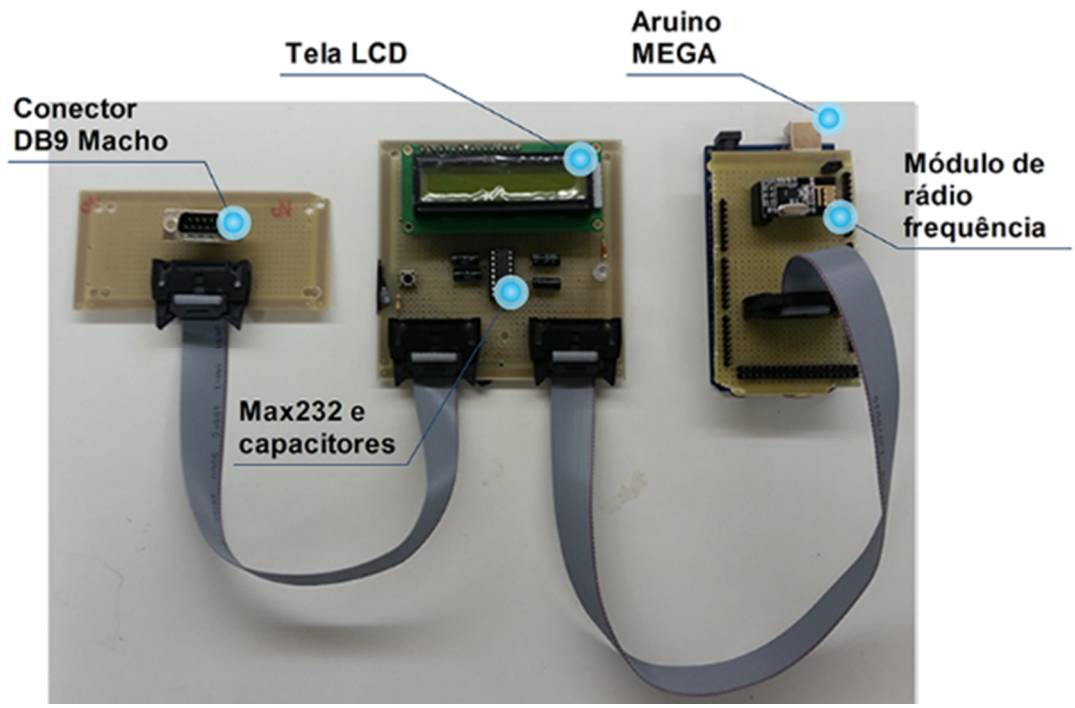
Fonte: do autor.

Na FIGURA 3.32 é possível observar o módulo de rádiofrequência e o *buzzer* conectado a placa de prototipagem Arduino UNO. O cabo USB ilustrado na figura é do tipo A/B usado na comunicação entra placa Arduino UNO e o computador via comunicação serial.

A segunda etapa da montagem final do protótipo é a passagem dos componentes eletrônicos da placa de ensaio para a placa de circuito impresso em que os mesmos são soldados. Desta forma é possível garantir um menor índice de interferências e mal contato já que não há fios expostos ao redor de componentes eletrônicos.

O Dispositivo Carrinho com todos os seus componentes soldados em uma placa de circuito impresso é exibido na FIGURA 3.33.

FIGURA 3.33 – DISPOSITIVO CARRINHO EM PLACA DE CIRCUITO IMPRESSO.

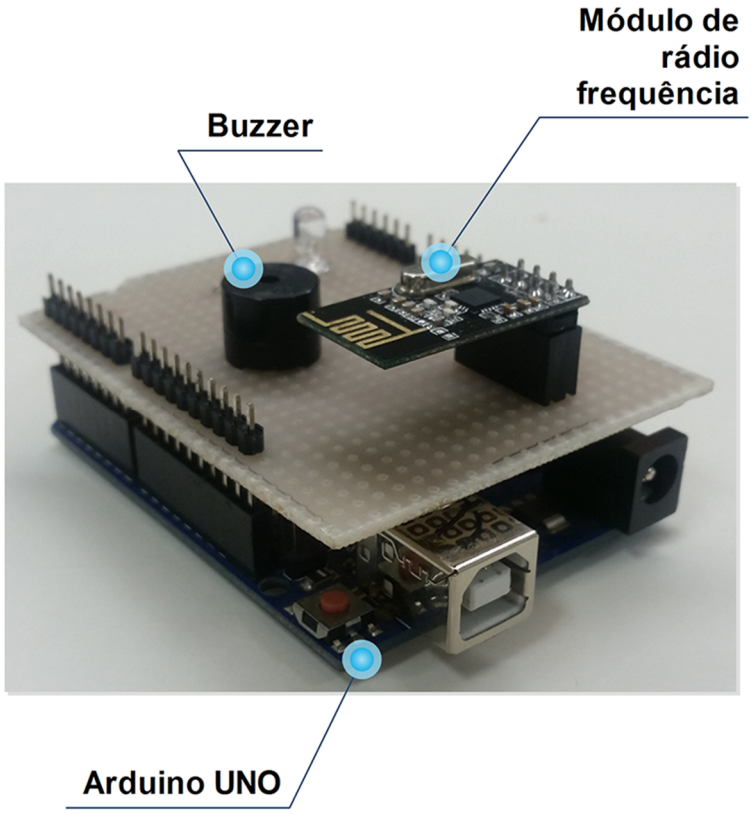


Fonte: do autor.

Na FIGURA 3.33 é possível observar os componentes soldados em três placas distintas conectados através do cabo tipo *lacth* de 14 pinos. Devida a grande quantidade de conexões elétricas entre os componentes a confecção foi dividida em três placas: uma placa dedicada ao conector DB9 macho; uma placa para a tela LCD e o circuito impresso max232 e seus respectivos capacitores e por fim uma placa com o módulo de rádiofrequência conectada a placa de prototipagem Arduino MEGA.

O Dispositivo servidor e seus componentes soldados em uma placa de circuito impresso é exibido na FIGURA 3.34 abaixo.

FIGURA 3.34 – DISPOSITVO SERVIDOR EM PLACA DE CIRCUITO IMPRESSO.



Fonte: do autor.

Na FIGURA 3.34 é possível observar a placa de circuito impresso conectada a placa de prototipagem Arduino UNO onde os componentes módulo de rádiofrequência e *buzzer* estão soldados.

4 TESTES E RESULTADOS

Este capítulo tem objetivo de apresentar os testes e resultados obtidos ao longo do desenvolvimento deste trabalho. Afins de organização os testes foram divididos em cenários:

a) **Primeiro cenário:** Apresentação da troca de informações entre o Dispositivo Carrinho e Dispositivo Servidor.
b) **Segundo cenário:** Apresentação da troca de mensagens entre o Dispositivo Servidor e o computador.
c) **Terceiro cenário:** Apresentação da rotina de *Handshake.*
d) **Quarto cenário:** Consulta no banco de dados com o produto cadastrado.
e) **Quinto cenário:** Consulta no banco de dados sem o produto cadastrado.
f) **Sexto cenário:** Apresentação da função *insert* operada pelo *software* de forma manualmente.
g) **Sétimo cenário:** Apresentação de problemas que ocasionalmente ocorrem no sistema de *hardware* e *software* desenvolvido.

Após a apresentação dos cenários de teste e resultados obtidos será exibido o custo estimado de implementação do projeto.

4.1 Primeiro cenário

4.1.1 Descrição

O primeiro cenário é a simulação da troca de informações entre o Dispositivo Carrinho e o Dispositivo Servidor em que o primeiro irá ler e mandar, via rádiofrequência, um código de barras e o segundo irá receber esta mensagem. Este estará conectado ao computador e com auxílio do ambiente de desenvolvimento do Arduino será possível monitorar a comunicação serial e mostrar na tela qualquer mensagem recebida pelo Dispositivo Servidor.

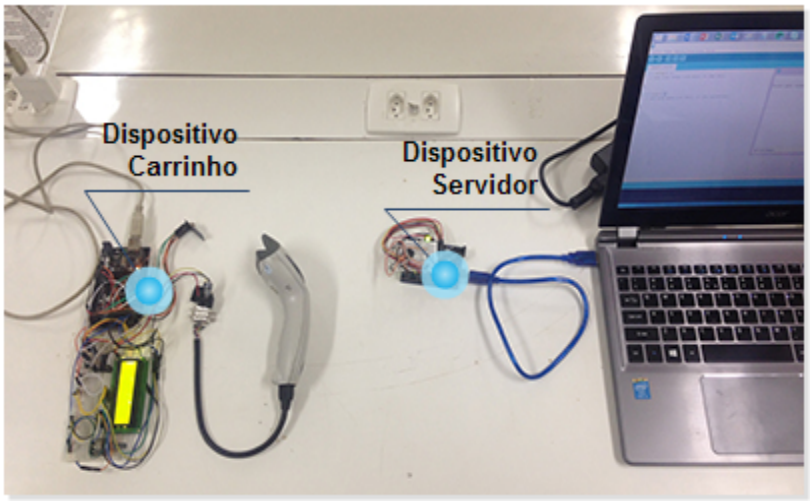
4.1.2 Pré-requisitos

Os pré-requisitos para este cenário de teste são:

a) Dispositivo Carrinho deverá estar conectado a uma fonte de alimentação externa.
b) Dispositivo Servidor deverá estar conectado ao computador via cabo USB/Serial.
c) O *software* (IDE) do Arduino deverá ser executado e com a tela "*Monitor Serial*" disponível.
d) A rotina de *handshake* deve ser realizada ao início do teste.

O ponto de partida deste cenário de teste é único e se repete para os demais cenários. A FIGURA 4.1 abaixo ilustra esta etapa.

FIGURA 4.1 – CENÁRIO DE TESTE.



Fonte: do autor.

O cenário é composto pelo Dispositivo Carrinho conectado a fonte externa de alimentação e do Dispositivo Servidor conectado ao computador via cabo USB/Serial. O computador por sua vez com o *software* Conexão MySQL em segundo plano e o banco de dados instanciado localmente.

4.1.3 Resultados

A leitura do código de barras “7891150029392” é realizada para demonstrar a comunicação entre os dois Dispositivos. Este teste tem o único objetivo de ilustrar a troca de mensagens, já que este procedimento é interno e o usuário não irá visualiza-lo ao efetuar uma consulta de preço através do código de barras.

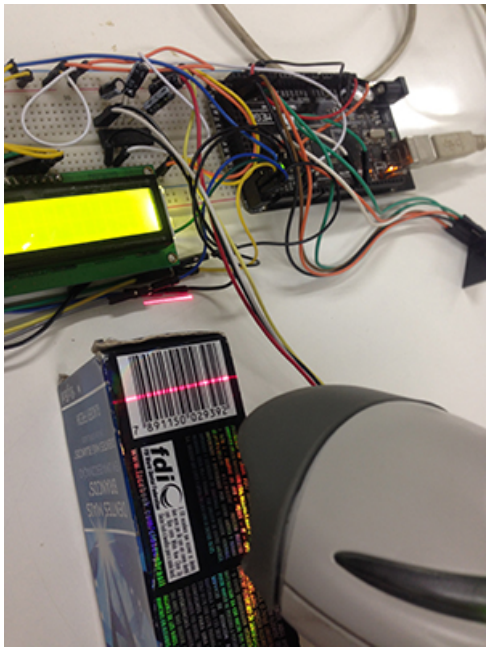
O código de barras a ser lido é exibido na FIGURA 4.2 abaixo.

FIGURA 4.2 –CÓDIGO DE BARRAS DO TIPO EAN-13.

Fonte: do autor.

A primeira etapa do teste é realizada a partir da leitura do código de barras pelo Dispositivo Carrinho e na sequência o dado será enviado via rádiofrequência para o Dispositivo Servidor. Esta etapa é exibida nas FIGURA 4.3 e FIGURA 4.4 abaixo.

FIGURA 4.3 – POSICIONAMENTO DO LEITOR DE CÓDIGO DE BARRAS.



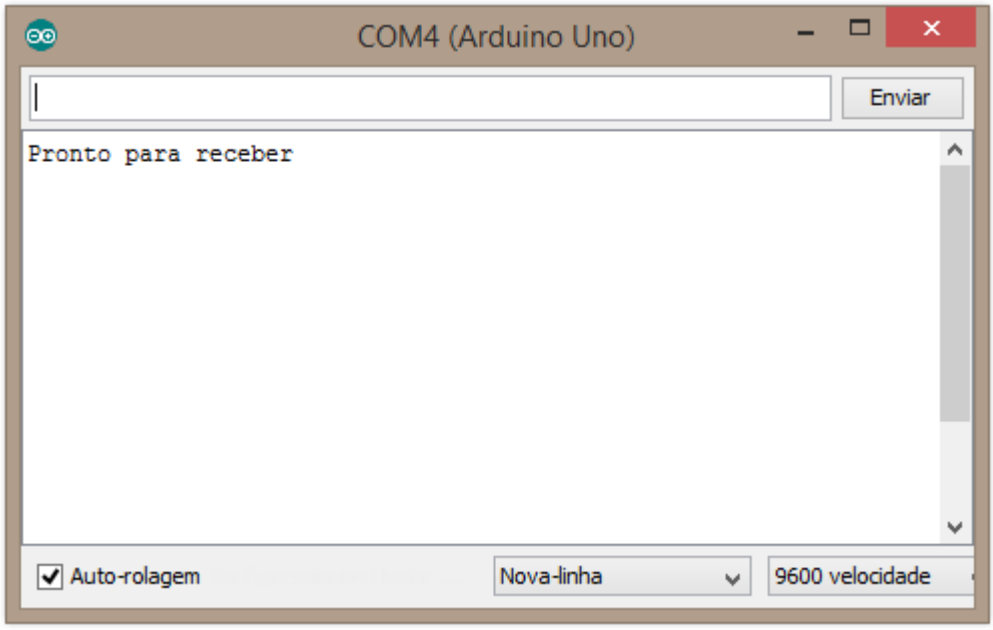
Fonte: do autor.

FIGURA 4.4 – CÓDIGO DE BARRAS SENDO EXIBIDO NA TELA LCD.

Fonte: do autor.

Em sequência o próximo passo é visualizar o código de barras através do ambiente de desenvolvimento da plataforma Arduino. A FIGURA 4.5 abaixo ilustra o *Monitor Serial* onde os dados serão visualizados.

FIGURA 4.5 – TELA DO *SOFTWARE* DA PLATAFORMA ARDUINO.

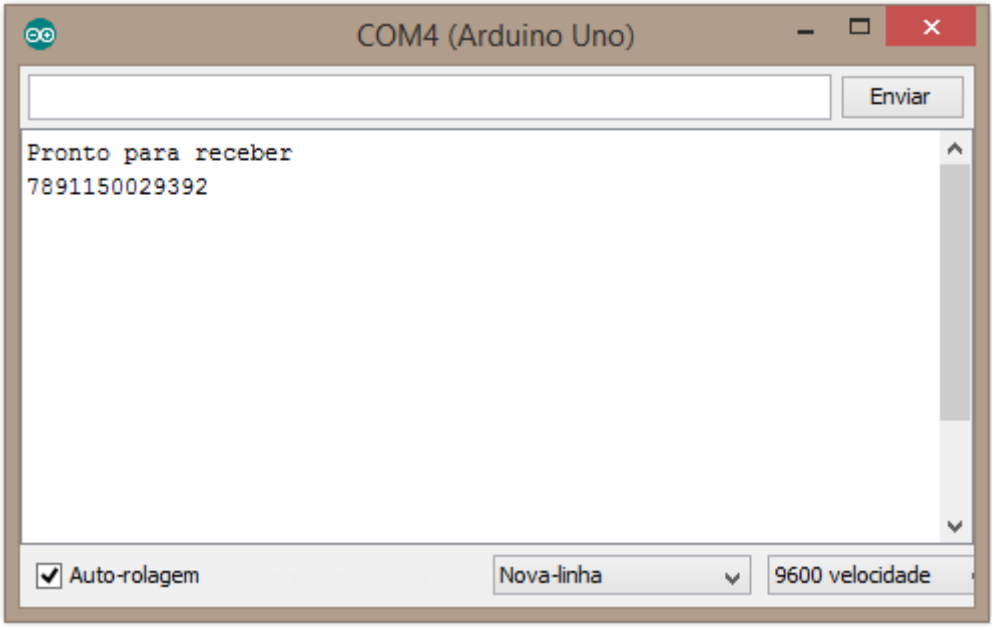


Legenda: os dados serão visualizados na tela *Monitor Serial*.

Fonte: do autor.

O dado recebido pelo Dispositivo Servidor é exibido no monitor na FIGURA 4.6 abaixo.

FIGURA 4.6 – TELA DO *SOFTWARE* DA PLATAFORMA ARDUINO.



Legenda: dados recebidos do Dispositivo Carrinho.

Fonte: do autor.

A partir do momento que o dado é visualizado no computador através do *software* do ambiente de desenvolvimento é a comprovação de que o dado foi enviado via rádiofrequência e recebido pelo Dispositivo Servidor e exibido na tela *Monitor Serial*.

4.2 Segundo cenário

4.2.1 Descrição

O segundo cenário é troca de mensagens, via comunicação serial, entre o Dispositivo Servidor e o computador. Esta etapa é a verificação onde o Dispositivo Servidor irá mandar uma mensagem para a porta serial e logo em seguida o computador irá ler esta mensagem, através do *software* Conexão MySQL, e então irá processar (realizando instruções do tipo *insert* ou *insert*) e devolver esta mensagem de volta para a porta serial.

Levando em consideração que a porta serial não permite comunicações simultâneas esta parte será testada virtualmente onde o Dispositivo Servidor será simulado por *software* e os dados serão inseridos manualmente na porta serial para que então seja possível monitorar e visualizar a resposta.

O teste será realizado simulando o Dispositivo Servidor através do *software Proteus* e os dados serão inseridos em um *Virtual Terminal*. Para tal finalidade o *software Virtual Serial Ports Emulator* é utilizado para simular uma porta serial a qual o *software Proteus* irá utilizar. Desta forma irá possibilitar que o Dispositivo Servidor seja emulado virtualmente.

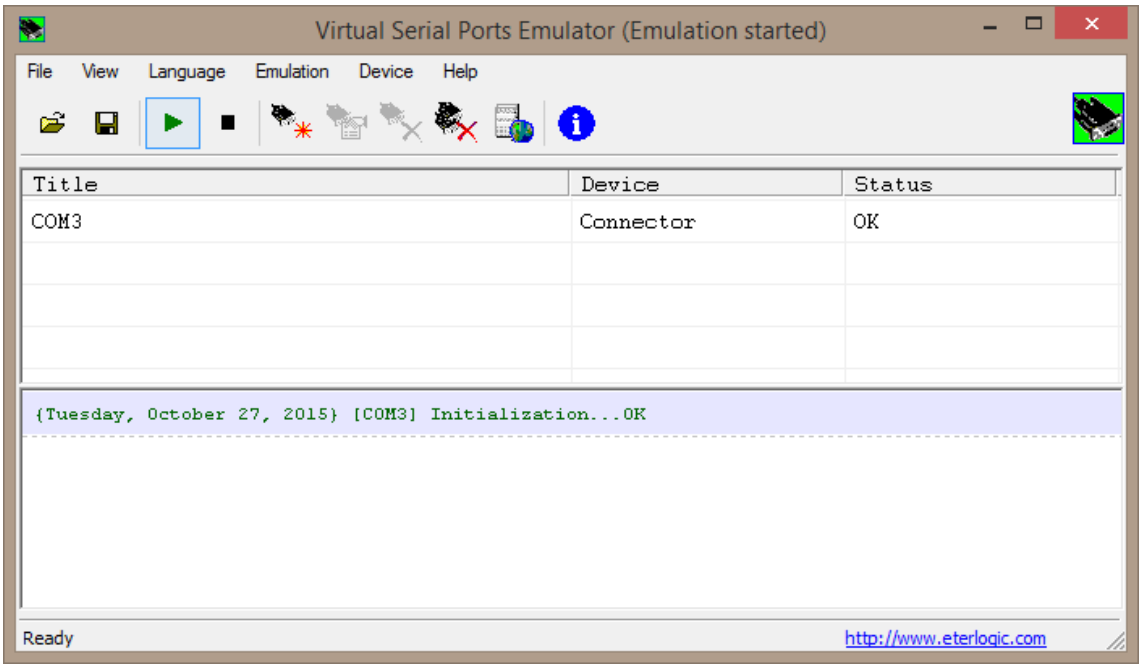
4.2.2 Pré-requisitos

Os pré-requisitos para este cenário são:

a) Porta Serial simulada em *software Virtual Serial Ports Emulator*
b) *Virtual Terminal* simulado em *software Proteus*
c) *Software* Conexão MySQL funcionando em segundo plano.
d) Banco de dados instanciado localmente.
e) O produto deve estar cadastrado no banco de dados.

Para este cenário de teste foi simulado uma porta aleatória no *software Virtual Serial Ports Emulator*. A virtualização desta porta serial, no caso a COM3, é exibida na FIGURA 4.7 abaixo.

FIGURA 4.7 – SIMULAÇÃO DA PORTA SEIRAL COM3.

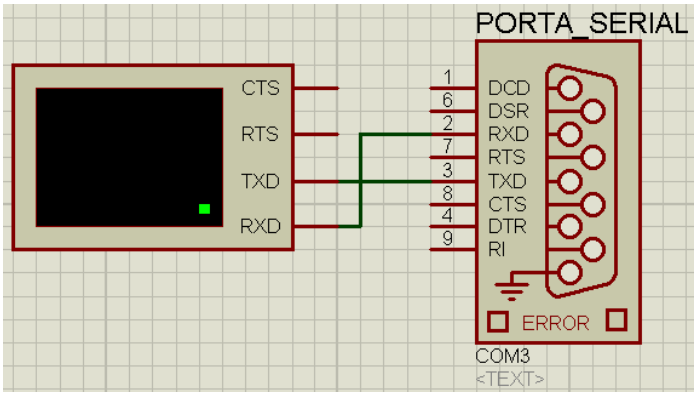


Legenda: simulação a partir do *software Virtual Serial Ports Emulator*.

Fonte: do autor.

O circuito base para simular um *Virtual Terminal* pelo *software Proteus* é exibido na FIGURA 4.8.

FIGURA 4.8 – CIRCUITO BÁSICO PARA SIMULAR UM TERMINAL VIRTUAL.

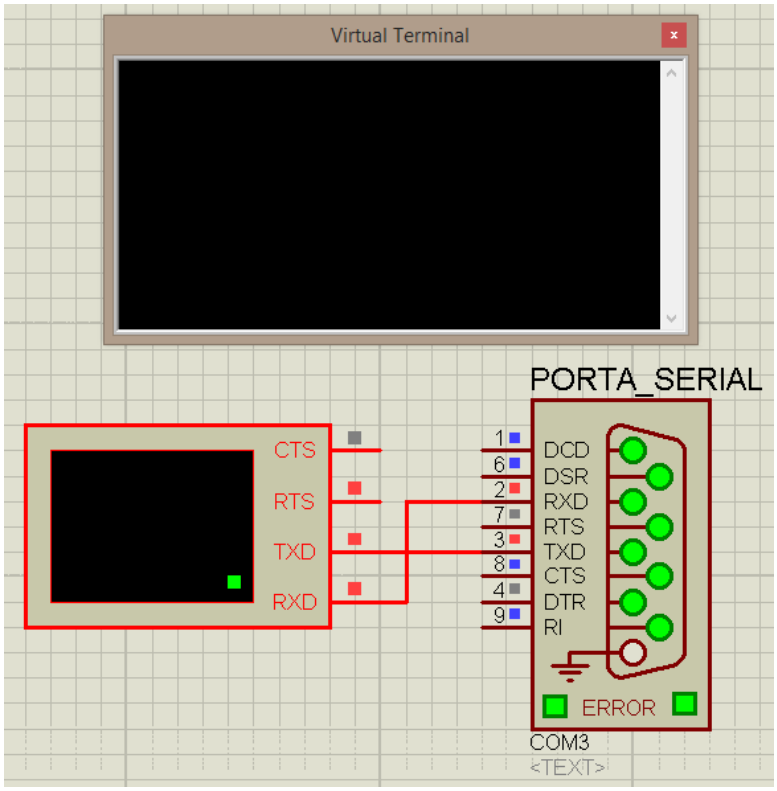


Legenda: simulação a partir do *software Proteus*.

Fonte: do autor.

O *Virtual Terminal* em modo de simulação é exibido na FIGURA 4.9 abaixo.

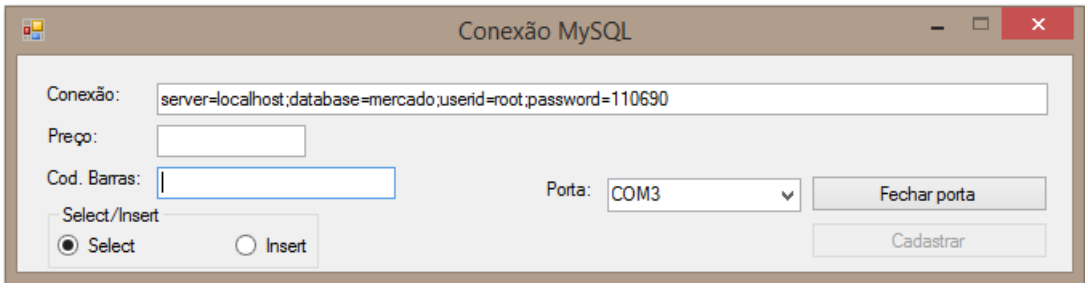
FIGURA 4.9 – VIRTUAL TERMINAL SIMULADO PELO *SOFTWARE* PROTEUS.



Fonte: do autor.

O *software* Conexão MySQL bem como os parâmetros de conexão com o banco de dados e a seleção de porta Serial é exibido na FIGURA 4.10 abaixo.

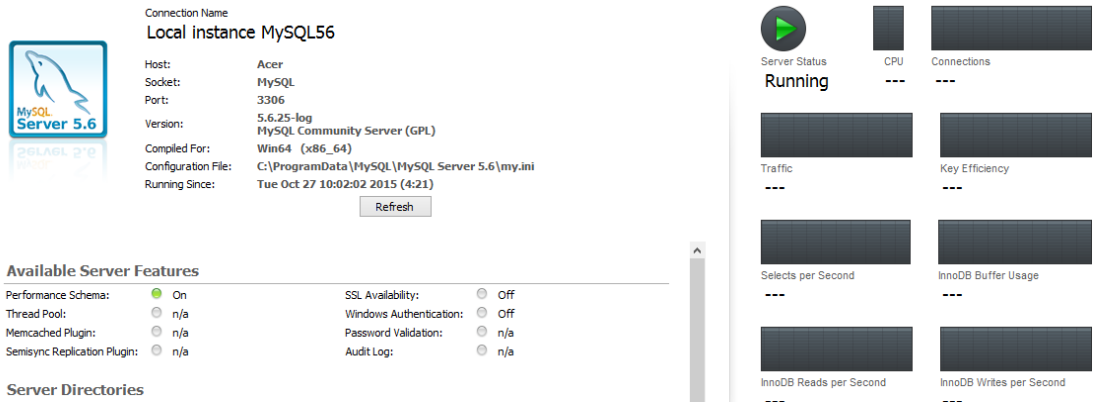
FIGURA 4.10 – *SOFTWARE* CONEXÃO MYSQL COM PARÂMETROS DE CONEXÃO COM O BANCO DE DADOS E SELEÇÃO DE PORTA SERIAL PARA COMUNICAÇÃO TESTE.



Fonte: do autor.

A instância local do banco de dados é exibida na FIGURA 4.11 abaixo.

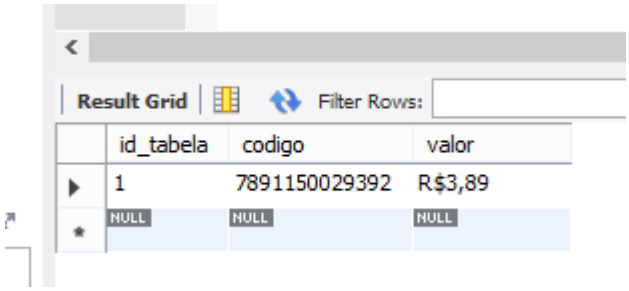
FIGURA 4.11 – INSTÂNCIA LOCAL DO BANCO DE DADOS ONLINE.



Fonte: do autor.

O produto cadastrado na tabela do banco de dados com o código de barras e o seu respectivo preço é exibido na FIGURA 4.12 abaixo.

FIGURA 4.12 –TABELA DO BANCO DE DADOS CONTENDO O CÓDIGO DE BARRAS E VALOR.



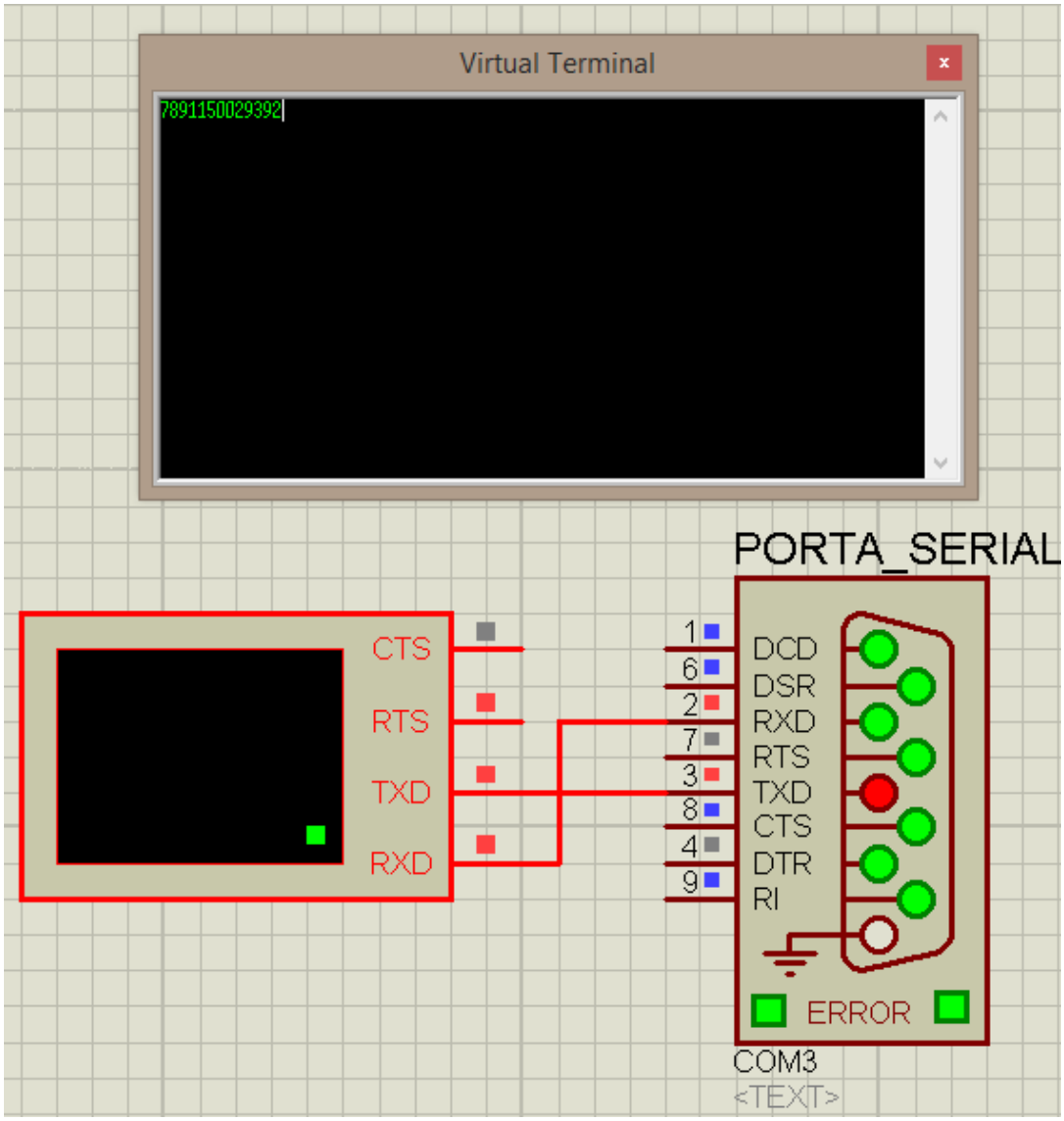
Fonte: do autor.

4.2.3 Resultados

O teste foi realizado através do *Virtual Terminal* do *software Proteus* em que o dado será inserido manualmente para que então o *software* Conexão MySQL faça a leitura e posteriormente devolva a resposta.

A execução do teste se dá pela inserção manualmente do código “7891150029392” no *Virtual Terminal*. FIGURA 4.13 abaixo ilustra este processo.

FIGURA 4.13 – ETAPA QUE MOSTRA O CÓDIGO DE BARRAS SENDO INSERIDO MANUALMENTE EM UMA PORTA SERIAL.



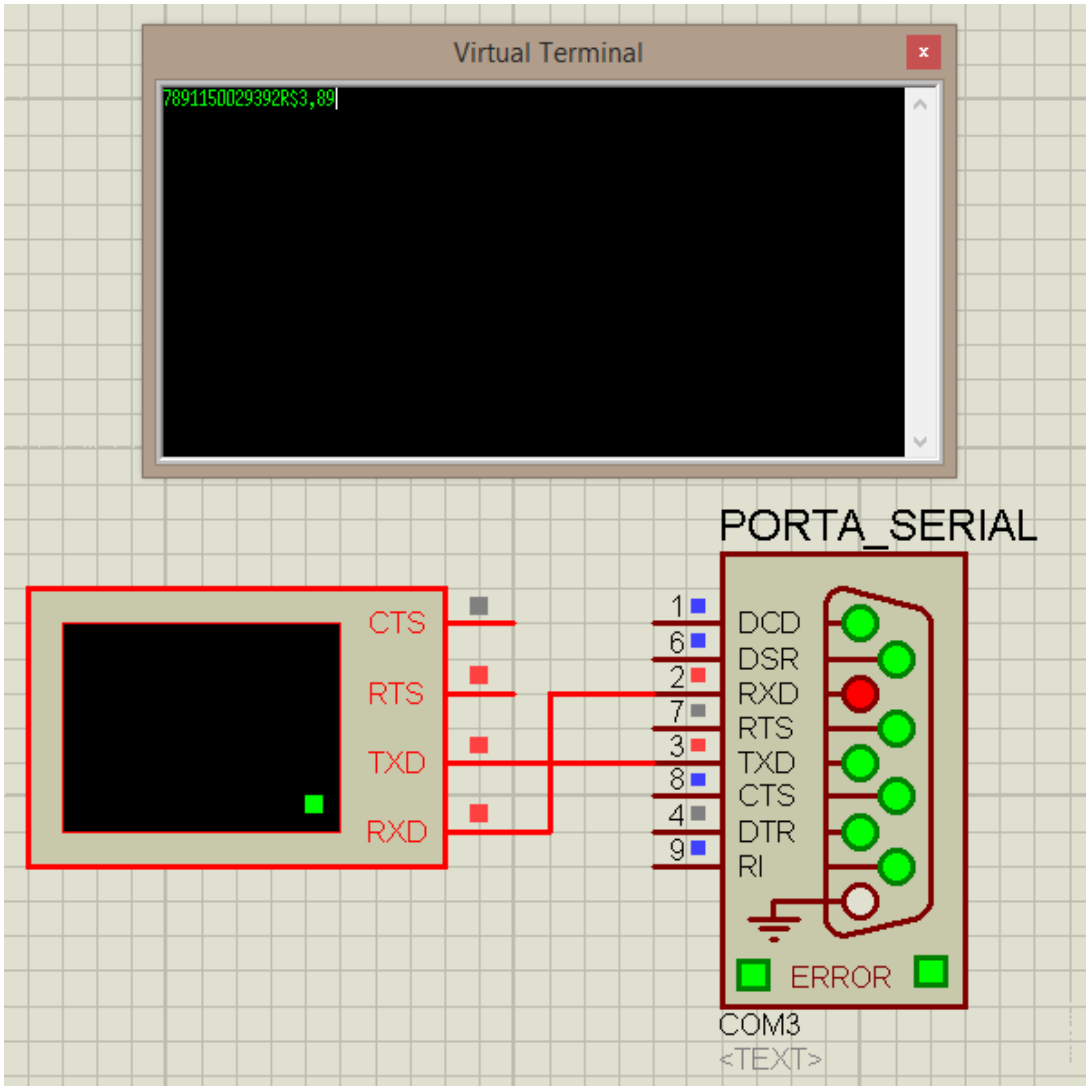
Fonte: do autor.

Note que pino denominado de TXD está representado de coloração avermelhada e isto representa o dado sendo enviado da porta serial COM3 virtual para o computador. Outro ponto importante é o dado no *Virtual Terminal* onde é possível visualizar que ele foi inserido.

Em sequência é esperado que o *software* Conexão MySQL efetue a instrução *select* no banco de dados e então retorne esta resposta para que seja possível visualiza-lo no *Virtual Terminal*. Como este *Virtual Terminal* não apresenta uma forma automática de limpeza de dados em tela então o dado será exibido logo abaixo do código de barras.

A FIGURA 4.14 abaixo ilustra esta última etapa do teste.

FIGURA 4.14 – REPRESENTAÇÃO DA RESPOSTA DO COMPUTADOR.



Fonte: do autor.

Note que o pino representado de RXD está representado de coloração avermelhada e isto representa a resposta do *software* Conexão MySQL para a porta serial COM3. Esta etapa é equivalente ao computador devolvendo a resposta para o Dispositivo Servidor.

Outro ponto importante é o dado representado no *Virtual Terminal* em que logo abaixo do código de barras inserido é apresentado a resposta do computador com o valor cadastrado no banco de dados: “RS3,89” dando por fim o teste deste cenário.

4.3 Terceiro cenário

4.3.1 Descrição

O terceiro cenário é a apresentação da rotina de *handshake* em que serão utilizados o Dispositivo Carrinho e o Dispositivo Servidor. Esta etapa do teste será ligar os dois Dispositivos s e aguardar os mesmos fazerem a rotina de *handshake* de forma automática.

4.3.2 Pré-requisitos

Os pré-requisitos para este teste são:

a) Dispositivo Carrinho alimentado por uma fonte externa;
b) Dispositivo Servidor conectado ao computador via cabo USB/Serial.

4.3.3 Resultados

Ao iniciar o Dispositivo Carrinho é possível visualizar uma mensagem de saudação na tela de cristal líquido. Esta etapa demonstra apenas que o Dispositivo foi iniciado com sucesso.

O Dispositivo Carrinho ligado e demonstrando a sua mensagem de saudação é exibido na FIGURA 4.15 abaixo.

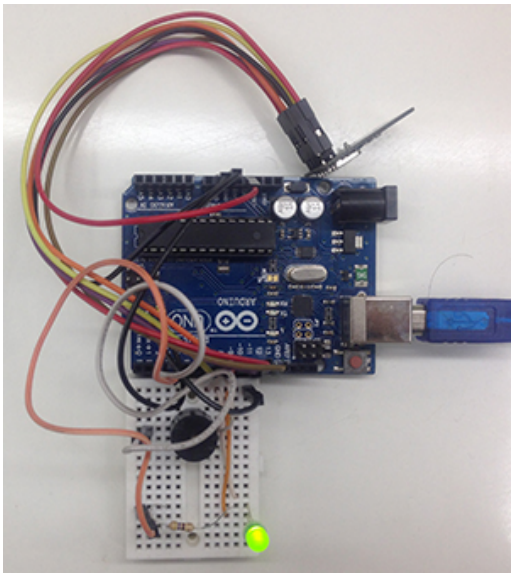
FIGURA 4.15 – DISPOSITIVO CARRINHO LIGADO.

Legenda: mensagem de saudação.

Fonte: do autor.

O Dispositivo Servidor ligado e em modo de operação é exibido na FIGURA 4.16 abaixo.

FIGURA 4.16 – DISPOSITVO SERVIDOR CONECTADO AO COMPUTADOR.



Fonte: do autor.

Após os dois Dispositivos ligados, a próxima etapa é aguardar três segundos e verificar a conclusão da rotina de *handshake* que é marcada com o recebimento da mensagem “diga-me o código” pelo Dispositivo Carrinho. A FIGURA 4.17 abaixo ilustra esta etapa.

FIGURA 4.17 – DISPOSITIVO CARRINHO RECEBENDO MENSAGEM.

Fonte: do autor.

Após visualizar a mensagem “Diga-me o código” os Dispositivos estão prontos para trocarem mensagens dando por fim este cenário de teste.

4.4 Quarto cenário

4.4.1 Descrição

O quarto cenário é a consulta efetuada pelo Dispositivo Carrinho em que o produto em questão deve está cadastrado no banco de dados. Este teste será considerado o mais importante pois ele constitui o objetivo deste trabalho.

Para este cenário será utilizado o Dispositivo Carrinho e o Dispositivo Servidor além do *software* Conexão MySQL funcionando em segundo plano e o banco de dados instanciado localmente.

Este teste é marcado pela leitura de um código de barras pelo Dispositivo Carrinho e posteriormente a isso será verificado, através da resposta via rádiofrequência do Dispositivo Servidor, o preço do produto cadastrado.

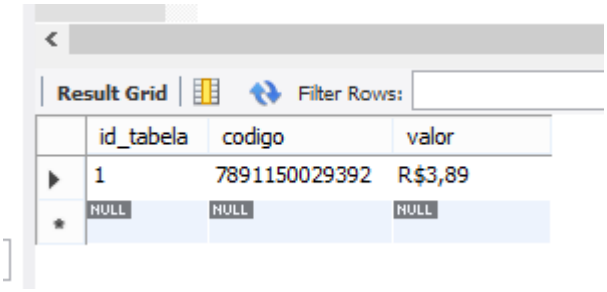
4.4.2 Pré-requisitos

Os pré-requisitos para este cenário são:

a) Dispositivo Carrinho conectado a uma fonte de alimentação externa;
b) Dispositivo Servidor Conectado ao computador através do cabo USB/Serial;
c) *Software* Conexão MySQL funcionando em segundo plano;
d) Banco de dados instanciado localmente;
e) Produto a ser requisitado deverá estar cadastrado no banco de dados.

O produto que será requisitado a consulta é mostrado abaixo na FIGURA 4.18 abaixo.

FIGURA 4.18 – CÓDIGO DE BARRAS CADASTRADO NO BANCO DE DADOS.

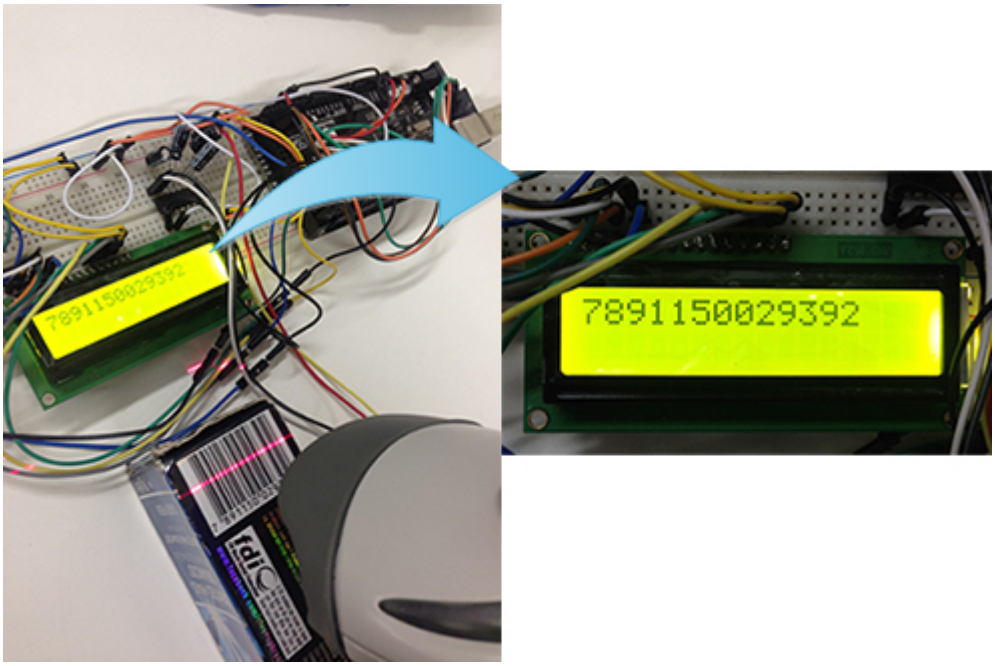


Fonte: do autor.

4.4.3 Resultados

A primeira etapa do teste é marcada pela leitura do código de barras pelo Dispositivo Servidor. A FIGURA 4.19 abaixo ilustra esta etapa.

FIGURA 4.19 – DISPOSITIVO CARRINHO FAZENDO A LEITURA DO CÓDIGO DE BARRAS.



Legenda: leitura a esquerda da imagem e o resultado sendo exibido em *zoom* na tela LCD a direita da imagem.

Fonte: do autor.

Na sequência é aguardado entre três a quatro segundos o Dispositivo Servidor mandar a resposta para que então seja possível visualizar o preço do produto cadastrado. A FIGURA 4.20 abaixo ilustra esta etapa.

FIGURA 4.20 – DISPOSITIVO CARRINHO MOSTRANDO NA TELA LCD A RESPOSTA.

Fonte: do autor.

A partir do momento que o Dispositivo Carrinho recebe a resposta, via rádiofrequência, do Dispositivo Servidor é pontuado o fim deste cenário de teste.

4.5 Quinto cenário

4.5.1 Descrição

O quinto cenário é a consulta efetuada pelo Dispositivo Carrinho em que o produto não está cadastrado no banco de dados. Para este cenário será utilizado o Dispositivo Carrinho, Dispositivo Servidor, *software* Conexão MySQL e banco de dados instanciado localmente.

Este cenário é similar ao anterior o que irá diferenciar é que o produto não estará cadastrado no banco de dados. O objetivo deste teste é observar o comportamento do sistema ao fazer uma consulta e esperar uma resposta negativa.

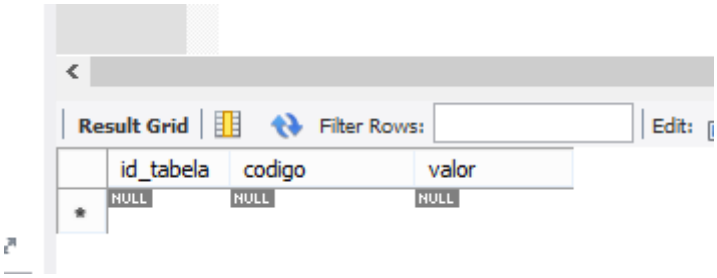
4.5.2 Pré-requisitos

Os pré-requisitos para este cenário são:

a) Dispositivo Carrinho conectado a uma fonte de alimentação externa;
b) Dispositivo Servidor Conectado ao computador através do cabo USB/Serial;
c) *Software* Conexão MySQL funcionando em segundo plano;
d) Banco de dados instanciado localmente;
e) Produto a ser requisitado não deve estar cadastrado no banco de dados.

O banco de dados sem nenhum registro é mostrado na FIGURA 4.21 abaixo.

FIGURA 4.21 – BANCO DE DADOS SEM REGISTRO.



Fonte: do autor.

4.5.3 Resultados

O início do teste é marcado pela leitura do código de barras o qual não está cadastrado no banco de dados. O código de barras que foi lido é o mesmo utilizado no teste anterior sendo esta etapa ilustrada na FIGURA 4.22 abaixo.

FIGURA 4.22 – DISPOSITIVO CARRINHO FAZENDO LEITURA DO CÓDIGO DE BARRAS NÃO CADASTRADO.

Fonte: do autor.

A próxima etapa do teste é esperar entre três a quatro segundos a resposta encaminhada do Dispositivo Servidor sendo esta parte ilustrada na FIGURA 4.23 abaixo.

FIGURA 4.23 – DISPOSITIVO CARRINHO EXIBINDO A RESPOSTA.

Fonte: do autor.

A partir do momento em que o Dispositivo Carrinho recebe a mensagem “Não consta” é marcado o fim deste cenário de teste.

4.6 Sexto cenário

4.6.1 Descrição

O sexto cenário é o exemplo de funcionamento da função *insert* do *software* Conexão MySQL em que o código de barras e seu respectivo preço é cadastrado manualmente no banco

de dados. Para este cenário serão utilizados o Dispositivo Carrinho e o Dispositivo Servidor além do *software* Conexão MySQL e o banco de dados instanciado localmente.

Este cenário de teste é fazer a inserção dos dados (código de barras e preço) manualmente através do *software*, porém é possível fazer a leitura do código de barras através do Dispositivo Carrinho e visualizar o código de barras na tela do *software* Conexão MySQL e posteriormente adicionar o preço do produto e fazer o cadastro. Desta forma é possível agilizar a etapa de cadastramento de um produto qualquer.

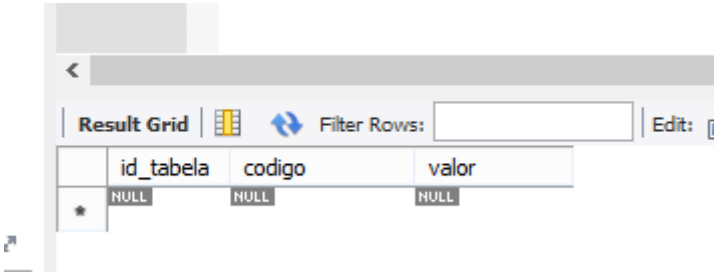
4.6.2 Pré-requisitos

Os pré-requisitos para este cenário são:

a) Dispositivo Carrinho conectado a uma fonte externa de alimentação;
b) Dispositivo Servidor conectado ao computador via cabo USB/Serial;
c) *Software* Conexão MySQL em modo de funcionamento;
d) Banco de dados instanciado localmente.

Será partido da premissa que o banco de dados não irá possuir nenhum registro para que desta forma seja mais simples de localizar o dado após ser inserido no mesmo. O banco de dados sem registro é exibido na FIGURA 4.24 abaixo.

FIGURA 4.24 – BANCO DE DADOS SEM REGISTRO.



Fonte: do autor.

4.6.3 Resultados

A primeira etapa do teste é fazer a leitura do código de barras em questão. O posicionamento do leitor é ilustrado na FIGURA 4.25 abaixo.

FIGURA 4.25 – POSICIONAMENTO DO LEITOR DE CÓDIGO DE BARRAS.

Fonte: do autor.

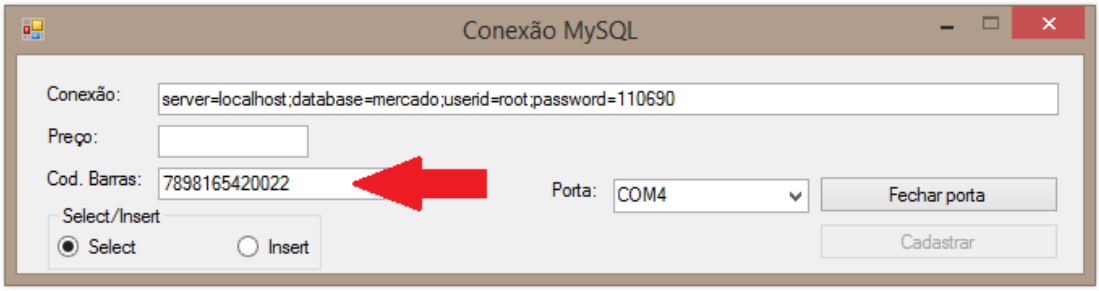
O código de barras lido é exibido na tela de cristal líquido do Dispositivo Carrinho. Esta etapa é mostrada na FIGURA 4.26 abaixo.

FIGURA 4.26 – DADO SENDO OBTIDO NA TELA LCD.

Fonte: do autor.

Em sequência, já na tela do *software* Conexão MySQL é possível ver o dado inserido no campo “*Cod. Barras: ”*. Esta etapa é ilustrada na FIGURA 4.27 abaixo.

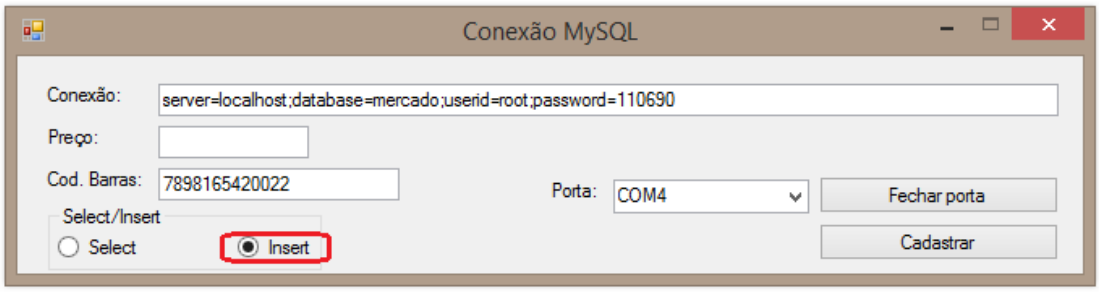
FIGURA 4.27 – INSERÇÃO DO CÓDIGO DE BARRAS.



Fonte: do autor.

O próximo passo é fazer a escolha da função *insert* no *software* para que seja habilitada a função de cadastramento do produto. Esta etapa é ilustrada na FIGURA 4.28 abaixo.

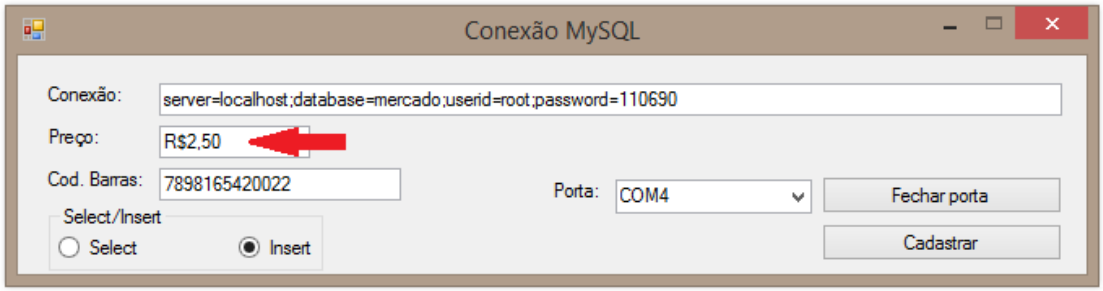
FIGURA 4.28 – FUNÇÃO DO TIPO INSERT SELECIONADA.



Fonte: do autor.

Após fazer a seleção da função *insert* será necessário digitar o preço equivalente do produto que no caso será “R$2,50”. Esta etapa é ilustrada na FIGURA 4.29 abaixo.

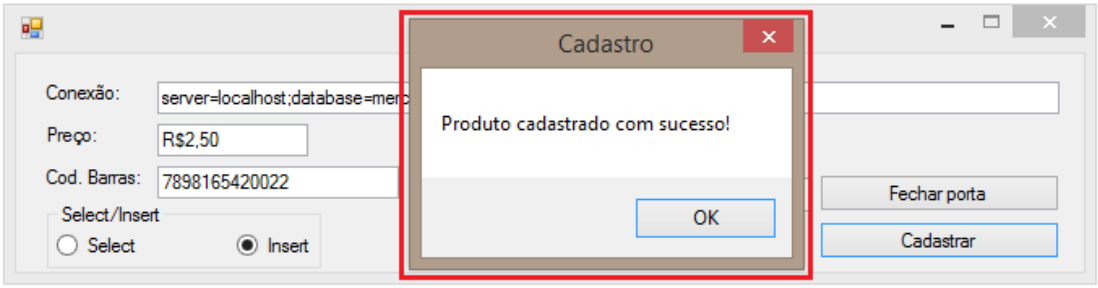
FIGURA 4.29 – PREÇO DO PRODUTO INSERIDO MANUALMENTE.



Fonte: do autor.

Após digitar o preço do produto basta clicar em “cadastrar” no *software* e aguardar a mensagem de confirmação: “Produto cadastrado com sucesso”. Esta etapa é ilustrada na FIGURA 4.30 abaixo.

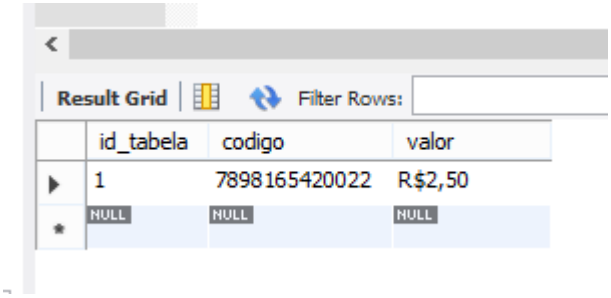
FIGURA 4.30 – CONFIRMAÇÃO DO *SOFTWARE* AO CADASTRAR.



Fonte: do autor.

Por fim, o último passo é verificar o dado no banco de dados. A FIGURA 4.31 abaixo ilustra o registro contido no banco de dados.

FIGURA 4.31 – PRODUTO CADASTRADO NO BANCO DE DADOS.



Fonte: do autor.

Após a confirmação do dado cadastrado no banco de dados é pontuado o fim deste cenário de teste.

4.7 Sétimo cenário

4.7.1 Descrição

O sétimo cenário é a apresentação dos possíveis erros que podem ocorrer ao utilizar o sistema de *hardware* e *software* construído. Este cenário é apenas uma descrição dos erros em que os mesmos não estão sendo tratados pelo sistema.

4.7.2 Identificação de possíveis erros

a) Perda de pacote de mensagem pelo módulo de rádiofrequência;
b) Perda de comunicação da porta serial entre o computador e Dispositivo Servidor;
c) Seleção de porta serial incorreta ao tentar comunicar entre o computador e Dispositivo Servidor;
d) Porta serial do Dispositivo Servidor ocupada impossibilitando a conexão;
e) Parâmetros incorreto ao fazer conexão com o banco de dados;
f) Banco de dados sem comunicação.

O comportamento do sistema com qualquer erro descrito acima é o mesmo. Na tela LCD do Dispositivo Carrinho é mostrado a mensagem negativa “Não Consta”.

Após o apontamento dos possíveis erros é pontuado o fim deste cenário de teste.

4.8 Custo estimado do projeto

O custo estimado do projeto para implementação dos sistemas de *hardware* em que é levado em consideração preços de mercado é exibido no QUADRO 4.1 abaixo.

QUADRO 4.1 – CUSTO DO PROJETO.

Custo de implementação do projeto		
Sistema de Hardware - Dispositivo Carrinho		
Descrição	Quantidade	Preço (Unitário)
Placa Arduino MEGA	1	R$ 79,90
Leitor de Código de Barras	1	R$ 112,00
Circuito Integrado *Max232*	1	R$ 1,45
Capacitor 1uF 50V	4	R$ 0,05
Tela de Cristal Líquido 16x2	1	R$ 29,90
Potenciômetro 5K	1	R$ 0,81
Conector DB9	1	R$ 1,35
Led	1	R$ 0,38
Resistor 100 Ohms 1/4W	1	R$ 0,04
Módulo de Rádiofrequência	1	R$ 14,90

QUADRO 4.1 – CUSTO DO PROJETO.

Custo de implementação do projeto		
Sistema de Hardware - Dispositivo Carrinho		
	Parcial	**R$ 240,93**
Sistema de hardware - Dispositivo Servidor		
Placa Arduino UNO	1	R$ 38,90
Buzzer	1	R$ 1,20
Led	1	R$ 0,38
Resistor 100 Ohms 1/4W	1	R$ 0,04
Módulo de Rádiofrequência	1	R$ 14,90
	Parcial	**R$ 55,42**
	TOTAL	**R$ 296,35**

Fonte: Elaborado pelo autor.

5 CONSIDERAÇÕES FINAIS

5.1 Conclusão

As inovações tecnológicas estão relacionadas a solucionar problemas do dia-a-dia das pessoas. A proposta deste trabalho está relacionada a auxiliar o usuário do segmento de supermercados na escolha do produto de uma forma que seja possível gerar uma certeza de que o preço do produto proposto é realmente o preço real.

A proposta deste trabalho é um sistema de *hardware* e *software* capaz de fazer uma consulta de um código de barras e por meio de um segundo dispositivo, via comunicação sem fio, é esperado a resposta do preço do produto por intermédio de um *software* e um banco de dados. Esta solução trará um conforto para os usuários que frequentam supermercado em seu dia-a-dia pois com esta proposta é possível gerar uma segurança, confiabilidade e um conforto maior no importante processo de escolha de um produto.

No desenvolvimento do trabalho proposto foi escolhido uma metodologia que gerou expectativas positivas pois foi possível com uma breve pesquisa adquirir os conhecimentos adicionais para a construção do trabalho. Pelo fato de se tratar de um sistema de *hardware* e *software*, foi optado pelo primeiro desenvolvimento no *hardware* em que desta forma facilitou a integração com o posterior desenvolvimento do sistema de *software*. Ainda que a escolha da plataforma de prototipagem Arduino possibilitou um rápido desenvolvimento para testes de componentes eletrônicos bem como a comunicação entre os blocos desenvolvidos.

Os resultados obtidos foram totalmente satisfatórios em que foi possível integrar as diferentes plataformas usadas como a placa de prototipagem Arduino MEGA e Arduino UNO com o *software* em linguagem C# além do banco de dados em MySQL.

Foi possível também integrar todos os diferentes protocolos de comunicação usados em que com o uso da plataforma Arduino Mega foi possível integrar os padrões de comunicação RS232 e TTL com auxílio do conversor de sinal *Max232*. Foi possível também integrar as duas plataformas Arduino MEGA e Arduino UNO através do padrão de comunicação SPI com auxílio do módulo de rádiofrequência. Com o padrão de comunicação serial foi possível também integrar a plataforma Arduino UNO com o computador ao auxílio do *software* em C# e o banco de dados.

O sistema de *hardware* e *software* desenvolvido se mostrou eficiente a fazer o processo de consulta de uma forma a apresentar uma resposta aceitável para o usuário. Outro ponto importante é que o sistema desenvolvido apesar de possuir seu foco para a consulta de um produto cadastrado o sistema se mostrou versátil em fazer o cadastramento de produtos no inventário (banco de dados) de uma forma a otimizar a leitura do código de barras deixando apenas o usuário fazer a inserção do preço de forma manual.

5.2 Proposta de trabalhos futuros

Em relação aos trabalhos futuros serão ressaltados alguns pontos interessantes que foram observados ao decorrer do desenvolvimento deste trabalho.

A primeira proposta de trabalho futuro é desenvolver sistemas de *hardware*s adicionais para exercer a mesma função do Dispositivo Carrinho para que desta forma seja possível aproximar o trabalho ainda mais do ambiente real de implementação. Para esta proposta deverá ser levado em consideração o gerenciamento das requisições de informações onde seja possível identificar cada pedido de informação de código de barras e o sistema seja capaz de gerenciar, por meio de uma fila e devolver a resposta para cada Dispositivo solicitante.

A segunda proposta de trabalho futuro é fazer o devido tratamento de erro do sistema de *hardware* e *software* em que seja possível, através do Dispositivo Carrinho, identificar qual erro que foi gerado pelo sistema. Desta forma é possível tornar o sistema mais robusto onde o mesmo irá identificar e apontar para o usuário onde que o erro foi gerado, seja ele no banco de dados, *software* ou até mesmo na comunicação sem fio.

Por fim, é sugerida uma terceira proposta de trabalho futuro onde deverá ser feita uma análise sobre a taxa de erro do sistema como um todo. Esta parte do trabalho inclui uma análise estatística onde seja possível, dentro de um cenário limitado de amostras, apontar a taxa de erro do projeto, ou seja, qual a probabilidade de erro esperada do mesmo. Outro ponto importante que deve ser analisado também é o tempo de resposta do projeto. Para esta parte deverá ser feito uma análise do sistema, tanto no *hardware* como o *software* de uma forma a analisar a eficiência do projeto por meio de um estudo aprofundado.

REFERÊNCIAS BIBLIOGRÁFICAS

ARDUINO. **Arduino Mega Board**, 2015. Disponivel em: <https://www.arduino.cc/en/Main/ArduinoBoardMega>. Acesso em: 12 Março 2015.

ARDUINO. **Arduino *Software***, 2015. Disponivel em: <https://www.arduino.cc/en/main/*software*>. Acesso em: 19 Março 2015.

ARDUINO. **Arduino Uno Board**, 2015. Disponivel em: <https://www.arduino.cc/en/Main/ArduinoBoardUno>. Acesso em: 5 Abril 2015.

ATMEL CORPORATION. **Atmega328P**, 2015. Disponivel em: <www.atmel.com/devices/atmega328p.aspx>. Acesso em: 15 Abril 2015.

BYTE SCOUT. **Introduction Into Barcodes**, 2014. Disponivel em: <https://bytescout.com/books/introduction-into-barcodes-book.html>. Acesso em: 10 Maio 2015.

ELETRÔNICA ARDUINO E ANDROID. **Como usar o módulo de Rádiofrequência NRF24L01 no Arduino**, 2014. Disponivel em: <https://www.youtube.com/watch?v=_1_ag3Y17fc>. Acesso em: 10 Março 2015.

FILIPEFLOP. **O que é Arduino?**, 2 setembro 2014. Disponivel em: <http://blog.filipeflop.com/arduino/o-que-e-arduino.html>. Acesso em: 3 Março 2015.

FILIPEFLOP. **Tutorial:** comunicação wireless com Arduino e módulo NRF24L01, 25 Março 2014. Disponivel em: <http://blog.filipeflop.com/wireless/arduino-modulo-nrf24l01-tutorial.html>. Acesso em: 4 Maio 2015.

LAUDON, K.; LAUDON, J. **Sistemas de Informação Gerenciais**. São Paulo: Figurativa Editoral MM Ltda, 2011. 3-11 p.

MAXIM INTEGRATED. **MAX323A**, 2015. Disponivel em: <https://www.maximintegrated.com/en/products/interface/transceivers/MAX232A.html/tb_tab0>. Acesso em: 18 Maio 2015.

MICROSOFT. **Guia de introdução ao Visual Studio**, 2015. Disponivel em: <https://msdn.microsoft.com/pt-br/library/ms165079.aspx>. Acesso em: 20 Maio 2015.

MICROSOFT. **Visual C#**, 2015. Disponivel em: <https://msdn.microsoft.com/pt-br/library/kx37x362.aspx>. Acesso em: 18 Maio 2015.

MILANE, A. **MySQL Guia do programador.** São Paulo: Novatec Editora, 2006.

MYSQL. **MySQL Workbench**, 2015. Disponivel em: <https://www.mysql.com/products/workbench/>. Acesso em: 20 Maio 2015.

NORDIC SEMICONDUCTOR. **nRF24L01+**, 2015. Disponivel em: <http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P>. Acesso em: 25 Maio 2015.

SACCO, F. **Comunicação SPI**, 05 maio 2014. Disponivel em: <http://www.embarcados.com.br/spi-parte-1/>. Acesso em: 15 agosto 2015.

SEIDEMAN, T. **Barcodes Sweep the World**, 15 Setembro 2011. Disponivel em: <http://www.barcoding.com/information/barcode_history.shtml>. Acesso em: 24 Março 2015.

SOARES, K. **O que é um Arduino e o que pode ser feito com ele?**, 4 setembro 2013. Disponivel em: <http://www.techtudo.com.br/noticias/noticia/2013/10/o-que-e-um-arduino-e-o-que-pode-ser-feito-com-ele.html>. Acesso em: 7 Março 2015.

SPARKFUN. **RS-232 vs. TTL Serial Communication**, 23 Novembro 2010. Disponivel em: <https://www.sparkfun.com/tutorials/215>. Acesso em: 25 Maio 2015.

STELLMAN, A.; GREEME, J. **Use a cabeça! C#.** Rio de Janeiro: Alta Books, 2008.

THINK WITH GOOGLE. **Top 3 tech trends - marketers should watch in 2015**, Janeiro 2015. Disponivel em: <http://think.storage.googleapis.com/docs/top-3-tech-trends-marketers-should-watch-in-2015_infographics.pdf>. Acesso em: 18 Abril 2015.

WILLE, C. **Apresentando C#:** Um novo conceito de linguagem de programação orientada a objeto. São Paulo: Berkeley, 2001.