

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO DE ELETRÔNICA
ENGENHARIA INDUSTRIAL ELÉTRICA COM ÊNFASE EM ELETRÔNICA E
TELECOMUNICAÇÕES

HUMBERTO FERNANDO MASSAHARU CAVAMURA
MARLOS KENJY MITSUHASHI

SISTEMA DE GERÊNCIA DE VAGAS DE ESTACIONAMENTO

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2014

HUMBERTO FERNANDO MASSAHARU CAVAMURA
MARLOS KENJY MITSUHASHI

SISTEMA DE GERÊNCIA DE VAGAS DE ESTACIONAMENTO

Trabalho de Conclusão de Curso como requisito parcial à obtenção do título de Bacharel em Engenharia Industrial Elétrica, com ênfase em Eletrônica e Telecomunicações, do Departamento de Eletrônica, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Sérgio Moribe

CURITIBA
2014

TERMO DE APROVAÇÃO

HUMBERTO FERNANDO MASSAHARU CAVAMURA
MARLOS KENJY MITSUHASHI

SISTEMA DE GERÊNCIA DE VAGAS DE ESTACIONAMENTO

Este trabalho de conclusão de curso foi apresentado no dia 17 de março de 2014, como requisito parcial para obtenção do título de Bacharel em Engenharia Industrial Elétrica, com ênfase em Eletrônica e Telecomunicações, outorgado pela Universidade Tecnológica Federal do Paraná. Os alunos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Hilton José Silva de Azevedo
Coordenador de Curso
Departamento Acadêmico de Eletrônica

Prof. Dario Eduardo Amaral Dergint
Responsável pela Atividade de Trabalho de Conclusão de Curso
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

Prof. Rubens Alexandre de Faria
UTFPR

Prof. Rafael Eleodoro de Goes
UTFPR

Prof. Sérgio Moribe
Orientador - UTFPR

AGRADECIMENTOS

Agradecemos primeiramente ao professor orientador Sérgio Moribe, que nos ajudou e deu-nos conselhos muito importantes para o desenvolvimento do projeto.

Agradecemos também ao professor Rubens Alexandre de Faria, pelas sugestões e pela atenção dada à equipe.

Aos nossos pais pelo apoio e o amor incondicional, que nos ajudaram a chegar até aqui.

Aos nossos amigos que estão presentes na nossa vida desde muito tempo, oferecendo suporte e a quem temos como referência de vida.

Aos grupos de jovens de que participamos que contribuíram com suas experiências de vida que sozinhos jamais conseguiríamos.

Por fim, agradecemos à todos os professores que nos ajudaram e compartilharam idéias para o desenvolvimento desse trabalho.

RESUMO

MITSUHASHI, Marlos Kenjy. CAVAMURA, Humberto Fernando Massaharu. **Sistema de Gerência de Vaga de Estacionamento**. 2014. 107 p. Trabalho de Conclusão de Curso (Bacharelado ou Tecnologia em Engenharia Industrial Elétrica, com ênfase em Eletrônica e Telecomunicações) - Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

Devido ao grande movimento de pessoas em estabelecimentos comerciais, houve um aumento de carros na procura por um local para estacionar. O objetivo principal do projeto é desenvolver um sistema que gerencie as vagas utilizando sensores, para verificar se está ou não ocupado e uma central que informe os motoristas sobre as informações do estacionamento. A metodologia consiste em estudar os métodos empregados em redes sem fio e nos meios de sensoriamento para então desenvolver uma estrutura de comunicação apropriada, com a construção dos módulos, desenvolvimento do hardware e a escolha por um meio de interface com o usuário. O projeto é dividido em duas partes, o módulo nas vagas e uma central. A primeira utiliza microcontrolador e um sensor de ultrassom que capta, processa os dados da vaga e envia para a central, através do canal de comunicação a ser desenvolvido. Já a segunda é formada pela Raspberry Pi com Sistema operacional Linux embarcado que capta essas informações e converte em uma interface para o usuário, através de uma tela de monitor. Esse projeto visa alcançar uma redução no tempo de procurar por um local para estacionar, diminuindo o tempo dentro do estacionamento.

Palavras-chave: Sensor de estacionamento. Rede sem fio. Raspberry Pi. nRF24L01+. Ultrassom.

ABSTRACT

MITSUHASHI, Marlos Kenjy. CAVAMURA, Humberto Fernando Massaharu. **Parking Management System**. 2014. 107 p. Trabalho de Conclusão de Curso (Bacharelado ou Tecnologia em Engenharia Industrial Elétrica, com ênfase em Eletrônica e Telecomunicações) - Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

Due to the large movement of people in shops, there was an increase in demand of cars for a place to park. The main objective of the project is to develop a system that manages parking spots using sensors to check if it is busy or not and a central to inform drivers about the parking information. The methodology consists in studying the methods used in wireless networks and means of sensing to then develop an appropriate communication structure, with the construction of modules, hardware development and the choice for the user interface. The project is divided into two parts, the module present in the parking spot and a central. The first uses a microcontroller and an ultrasonic sensor that captures, processes the data and sends the status of the parking spot to the central through the communication channel to be developed. The second consists of a Raspberry Pi with an operational system Linux embedded and converts this information into a user interface through a monitor screen. This project aims to achieve a reduction in the time to look for a place to park, reducing the time inside the parking lot.

Keywords: Parking Sensor. Wireless network. Raspberry Pi. nRF24L01+. Ultrasound.

LISTA DE ILUSTRAÇÕES

Figura 1 - Diagrama do funcionamento do projeto	13
Figura 2 - Faixas de frequência do som	21
Figura 3 - Diagrama funcional do MSP430G2553	26
Figura 4 - Pinagem do MSP430G2553	27
Figura 5 - Diagrama de blocos do chip nRF24L01+	28
Figura 6 - Pinagem do módulo de comunicação sem fio nRF24L01+	29
Figura 7 - Sensor ultrassônico HC-SR04	30
Figura 8 - Diagrama da leitura da distância pelo sensor	31
Figura 9 - Raspberry Pi – Model B	32
Figura 10 - Pinagem da porta P1 da Raspberry Pi – Model B	32
Figura 11 - Esquema de uma rede independente com 2 módulos	34
Figura 12 - Pacote de configuração enviado por cada elemento da rede	35
Figura 13 - Exemplo do uso do pacote de resposta na rede	36
Figura 14 - Fluxograma do envio de configuração	39
Figura 15 - Fluxograma do recebimento de configuração	40
Figura 16 - Fluxograma do funcionamento do módulo	42
Figura 17 - Escrevendo um byte via SPI, utilizando assembly	43
Figura 18 - Interligação entre o nRF24L01+ e o microcontrolador (Component_3) ...	43
Figura 19 - Interligação entre o sensor (HC1) e o microcontrolador (M1)	44
Figura 20 - Tratamento da interrupção de timer para determinar a distância	45
Figura 21 - Ligação entre o pino do microcontrolador, borne a referência da placa ..	47
Figura 22 - Interligação entre a sinalização sonora (LS1) e o microcontrolador (Component_3)	48
Figura 23 - Circuito regulador com LM317	49
Figura 24 - Circuito regulador com LM317 para tensão de 3V	49
Figura 25 - Circuito regulador com LM317 para tensão de 5V	50
Figura 26 - nRF24L01+ com amplificador de potência (PA) e de baixo ruído (LNA) ..	53
Figura 27 - Testando fonte e estilo via SDL_TTF	57
Figura 28 - Teste da fonte conforme os diversos comandos	58
Figura 29 - Imagem sem SDL_MapRGB() e SDL_SetColorKey()	59
Figura 30 - Imagem utilizando os meios para deixar uma cor invisível	60
Figura 31 - Função tela_inicial()	61
Figura 32 - Exemplo de como funciona a leitura do mouse	62
Figura 33 - Exemplo de como funciona a leitura do teclado	63
Figura 34 - Função tela_conf()	64
Figura 35 - Função tela_add()	65
Figura 36 - Função tela_del()	66
Figura 37 - Função escolhaVaga()	67
Figura 38 - Função tela_id()	67
Figura 39 - Diferentes estados para os ícones de vagas	68
Figura 40 - Função tela_leitura	68
Figura 41 - Formas de onda do pino de trigger (azul) e de echo (amarelo)	69
Figura 42 - Forma de onda do pino de echo, para uma distância de 210 cm.	70
Figura 43 - Composição do pacote de configuração a ser enviado pela SPI	71
Figura 44 - Forma de onda no pino MOSI, ao enviar pacote de configuração	71
Figura 45 - Composição do pacote de resposta recebido pela SPI	72
Figura 46 - Forma de onda no pino MISO ao receber pacote de resposta	72
Figura 47 - Terminal ao ser iniciada a configuração da rede	75

Figura 48 - Terminal com a rede já configurada.....	76
Figura 49 - Sistema funcionando com todas as vagas disponíveis.	76
Figura 50 - Representação com o carro entrando.....	77
Figura 51 - Representação com o carro estacionado na vaga.	77
Figura 52 - Representação com o carro saindo na vaga.....	78
Figura 53 – Perda de conexão com a rede independente 0.....	78
Figura 54 - Fluxograma da venda do mapTrace ao cliente	87
Figura 55 - Estrutura por funções da SEII Automação	92
Figura 56 - Esquemático do módulo, sem a parte da sinalização luminosa.....	105
Figura 57 - Placa do circuito do módulo de estacionamento	107
Fotografia 1 - MSP-EXP430G2 LaunchPad	24
Fotografia 2 - Chip nRF24L01+	27
Fotografia 3 - Módulo sensor, sem a parte da sinalização luminosa.	41
Fotografia 4 - Sinalização luminosa.	47
Fotografia 5 - Posicionamento do módulo em relação ao carro	51
Fotografia 6 - Ligação do nRF24L01+ com os pinos de GPIO da Raspberry Pi	53
Fotografia 7 - Módulo instalado para o teste.....	73
Fotografia 8 - Área utilizada para teste com 2 vagas (rede independente 0).	74
Fotografia 9 - Área utilizada para teste com 2 vagas (rede independente 1).	74
Fotografia 10 - Foto da interface	75
Quadro 1 - Pinagem do módulo de comunicação sem fio nRF24L01+	30
Quadro 2 - Pinagem do sensor HC-SR04	30
Quadro 3 - Quantidade de estabelecimentos na região sul do Brasil.....	84
Quadro 4 - Quantidade de shoppings no Brasil.....	85
Quadro 5 - Números dos grandes supermercados na região sul em 2011 e 2012 ...	85
Quadro 6 - Quantidade de shoppings em cada região do Brasil	86
Quadro 7 - Projeção de vendas do maptrace ao longo de 5 anos	90
Quadro 8 - Cronograma	91
Quadro 9 - Quadro de pessoal ao longo de três anos e seu custos.....	93
Quadro 10 - Dados sobre o Break Even do mapTrace	94
Quadro 11 - Projeção da D.R.E da SEII Automação	94
Quadro 12 - Projeção de fluxo de caixa da SEII Automação	95

LISTA DE SIGLAS

A/D	Analógico/Digital
SPI	<i>Serial Peripheral Interface</i>
I/O	<i>Input/Output</i>
RISC	<i>Reduced Instruction Set Computer</i>
GFSK	<i>Gaussian Frequency-Shift Keying</i>
GPIO	<i>General Purpose Input Output</i>
UART	<i>Universal Asynchronous Receiver Transceiver</i>
USB	<i>Universal Serial Bus</i>
PWM	<i>Pulse Width-Modulation</i>
I ² C	<i>Inter Integrated Circuit</i>
SRAM	<i>Static Random Access Memory</i>
D/A	Digital/Analógico
EEPROM	<i>Electrically-Erasable Programmable Read-Only Memory</i>
JTAG	Joint Test Action Group
LCD	<i>Liquid Crystal Display</i>
RF	Rádio Frequência
GPIO	<i>General Purpose Interface Bus</i>
SCSI	<i>Small Computer System Interface</i>
HDMI	<i>High-Definition Multimedia Interface</i>
PA	<i>Power Amplifier</i>
LNA	<i>Low Noise Amplifier</i>
DVI	<i>Digital Visual Interface</i>
SMBus	<i>System Management Bus</i>
SDL	<i>Simple Directmedia Layer</i>
IHM	<i>Interface homem-máquina</i>
TTF	<i>TrueType Font</i>

SUMÁRIO

1 INTRODUÇÃO	12
1.1 MOTIVAÇÃO E JUSTIFICATIVAS	12
1.2 OBJETIVOS	12
1.2.1 Objetivo Geral	12
1.2.2 Objetivos Específicos	13
1.3 DIAGRAMA	13
1.4 METODOLOGIA	13
1.5 APRESENTAÇÃO DO DOCUMENTO	14
2 FUNDAMENTAÇÕES	16
2.1 COMUNICAÇÃO SPI	16
2.1.1 Transmissão de Dados	17
2.1.2 Vantagens e Desvantagens da SPI	18
2.1.2.1 Vantagens	18
2.1.2.2 Desvantagens	18
2.2 MICROCONTROLADORES	19
2.2.1 Aplicações	20
2.3 ULTRASSOM	20
2.4 REDE SEM FIO	21
2.5 COMUNICAÇÃO EM RÁDIO FREQUÊNCIA	22
2.5.1 Funcionamento	22
2.5.2 Regulamentação física e licença	23
2.5.3 Determinando o alcance	23
2.5.4 Taxa de envio de dados	24
2.6 MSP LAUNCHPAD	24
2.7 MSP430G2553	25
2.7.1 Diagrama funcional	26
2.7.2 Pinagem	26
2.8 MÓDULO DE COMUNICAÇÃO NRF24L01+	27
2.8.1 Diagrama de blocos do <i>chip</i> nRF24L01+	28
2.8.2 Estrutura física do módulo de comunicação sem fio nRF24L01+	29
2.8.3 Pinagem módulo de comunicação sem fio nRF24L01+	29
2.9 SENSOR ULTRASSÔNICO HC-SR04	30
2.9.1 Pinagem	30
2.9.2 Leitura da distância	31
2.10 RASPBERRY PI	31
3 REDE SEM FIO PARA COMUNICAÇÃO DE DADOS	34
3.1 COMPOSIÇÃO DA REDE	34
3.1.1 Pacote de configuração e de resposta	35
3.1.2 Endereço e canal de comunicação do nRF24L01+	36
3.1.3 Processamento dos estados recebidos	36
3.2 CRIAÇÃO DE REDES INDEPENDENTES	37
3.3 ENVIA CONFIGURAÇÃO E RECEBE ESTADOS DAS VAGAS	37
3.4 RECEBE CONFIGURAÇÃO E ENVIA ESTADOS DAS VAGAS	39
4 MÓDULO DE AQUISIÇÃO E COMUNICAÇÃO DE DADOS	41
4.1 MÓDULO PRESENTE NAS VAGAS	41
4.1.1 Interação entre o microcontrolador e o módulo de comunicação RF	42
4.1.2 Lendo a distância a partir do sensor ultrassônico	44
4.1.3 Verificação do estado da vaga em função da variação da distância	45

4.1.4	Sinalização na vaga para o motorista	46
4.1.4.1	Sinalização luminosa	47
4.1.4.2	Sinalização sonora	48
4.2	ALIMENTAÇÃO	48
4.3	POSIÇÃO DO MÓDULO E DA SINALIZAÇÃO LUMINOSA NA VAGA	50
5	CENTRAL	52
5.1	PAPEL NA REDE DESENVOLVIDA	52
5.2	INTERFACE	53
5.2.1	Elementos utilizados:	53
5.2.1.1	Classes	54
5.2.1.2	Listas:	54
5.2.1.3	Biblioteca de I/O	54
5.2.1.4	Biblioteca SDL	55
5.2.1.4.1	<i>SDL_Surface</i>	56
5.2.1.4.2	<i>SDL_Event</i>	56
5.2.1.4.3	<i>Sub-biblioteca SDL_TTF</i>	57
5.2.1.4.4	<i>Sub-biblioteca SDL_IMG</i>	58
5.2.2	Funções	60
5.2.2.1	Tela_inicial()	61
5.2.2.2	Tela_conf()	62
5.2.2.3	Tela_add()	64
5.2.2.4	Tela_del()	65
5.2.2.5	Tela_escolhaVaga() e Tela_id()	66
5.2.2.6	Tela_leitura()	68
6	TESTES E RESULTADOS OBTIDOS	69
6.1	TESTE COM O SENSOR ULTRASSÔNICO	69
6.2	TESTE COM O MÓDULO DE COMUNICAÇÃO RF	70
6.3	TESTE DO SISTEMA EM UM ESTACIONAMENTO COM 4 VAGAS	72
6.3.1	Montagens para o teste	72
6.3.2	Funcionamento do Sistema de Gerência das Vagas de Estacionamento na residência	75
7	PLANO DE NEGÓCIOS	80
7.1	SUMÁRIO EXECUTIVO	80
7.2	DESCRIÇÃO DO NEGÓCIO	80
7.2.1	Visão:	80
7.2.2	Missão	81
7.2.3	Valores	81
7.2.4	Descrição Do Negócio:	81
7.3	OBJETIVOS	81
7.3.1	Objetivos Principais	81
7.3.2	Objetivos Intermediários	82
7.4	PRODUTO E SERVIÇO	82
7.4.1	Descrições Do Produto E Serviços	82
7.4.2	Análise Comparativa	82
7.4.3	Tecnologia	83
7.4.4	Produtos E Serviços Futuros	83
7.5	ANÁLISE DE MERCADO RESUMIDA	83
7.5.1	Segmentação De Mercado	84
7.5.2	Segmento Alvo De Mercado	84
7.5.2.1	Necessidade do mercado	84

7.5.2.2	Tendências do mercado.....	85
7.5.2.3	Crescimento do mercado	85
7.5.3	Análise Da Indústria	86
7.5.3.1	Players	86
7.5.3.2	Modelo de distribuição	87
7.5.3.3	Modelo de competitividade.....	88
7.5.3.4	Principais players	88
7.6	PROPOSTA DE VALOR	88
7.7	ESTRATÉGIAS.....	89
7.7.1	Diferenciais Competitivos.....	89
7.7.2	Estratégia De Marketing.....	89
7.7.2.1	Estratégia de preços	89
7.7.2.2	Estratégia de promoção	90
7.7.2.3	Estratégia de distribuição.....	90
7.7.3	Estratégia De Vendas	90
7.7.3.1	Projeção de vendas	90
7.7.3.2	Plano de vendas	90
7.7.4	Alianças Estratégicas.....	91
7.7.5	Cronograma	91
7.8	GESTÃO	92
7.8.1	Estrutura Organizacional.....	92
7.8.2	Equipe.....	92
7.8.2.1	Administrador:.....	92
7.8.2.2	Engenheiro:.....	93
7.8.2.3	Analista em marketing.....	93
7.8.2.4	Auxiliar de produção	93
7.8.3	Quadro De Pessoal.....	93
7.9	PLANO FINANCEIRO.....	94
7.9.1	Considerações	94
7.9.2	Análise Do <i>Break-Even</i>	94
7.9.3	Projeção Do Resultado	94
7.9.4	Projeção Do Fluxo De Caixa	95
8	CONSIDERAÇÕES FINAIS.....	96
8.1	CONSIDERAÇÕES SOBRE O DESENVOLVIMENTO DO PROJETO	96
8.2	DIFICULDADES ENCONTRADAS	96
8.3	OBJETIVOS ALCANÇADOS	97
8.4	MELHORIAS FUTURAS	97
	REFERÊNCIAS.....	99
	APÊNDICE A - Esquemático do módulo	104
	APÊNDICE B - Placa de circuito impresso gerada para o módulo	106

1 INTRODUÇÃO

1.1 MOTIVAÇÃO E JUSTIFICATIVAS

As soluções disponíveis no que tange à gerenciamento de vagas de estacionamento não atendem à totalidade das funcionalidades requisitadas, sempre pecando em algum quesito. Existem produtos que resolvem uma parcela dos problemas, não oferecendo um ótimo funcionamento, causando *stress* e irritação nos motoristas.

Os gerenciadores existentes atualmente apresentam apenas indicação de presença do carro sobre as vagas, não existindo um monitor antes de entrar no estacionamento para verificar quais e onde estão as vagas livres, obrigando o usuário a achar uma vaga tendo que andar pelo estacionamento, o que, com o projeto desenvolvido não ocorreria, uma vez que se pode ver antes os locais exatos que estão livres.

Quanto ao sensoriamento para estacionar o carro na vaga, existe apenas o sistema instalado no carro, obrigando o motorista a ter um carro com esse sistema já instalado ou então, tendo que o instalar, o que demanda gastos. Com o sensor instalado externamente, não importa se o carro tem ou não o sensoriamento no carro, ele conseguirá utilizar o mesmo de forma a estacionar de forma segura e correta, sem causar danos ao carro.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

A proposta consiste em gerenciar um estacionamento, utilizando-se de sensores instalados em cada uma das vagas, visando verificar o estado da vaga, que auxiliará também no ato de estacionar. O usuário poderá ter a exata noção do estado de cada uma das vagas quando observar o painel eletrônico, instalado na entrada do estacionamento.

1.2.2 Objetivos Específicos

O sensor de estacionamento irá enviar, via rede sem fio, um sinal a respeito da situação da vaga (ocupada ou livre) para o gerenciador, que irá receber o sinal enviado por todos os sensores e mostrará o estado de cada uma das respectivas vagas no painel eletrônico, que corresponderá à uma miniatura do estacionamento real. Ao sensor caberá também auxiliar o motorista no ato de estacionar. Ele estará fixo na parede e possuirá um alarme sonoro e luminoso e será acionado e se tornará mais acentuado conforme chegar ao limite estabelecido como ideal em relação à parede.

1.3 DIAGRAMA

Na figura 1, é mostrado um diagrama que resume o funcionamento do projeto.



Figura 1 - Diagrama do funcionamento do projeto
Fonte: Autoria própria (2014)

1.4 METODOLOGIA

- Estudo dos métodos de sensoriamento de distância;
- Estudo de métodos de redes de comunicação sem fio;
- Estudo dos recursos disponíveis do microcontrolador utilizado;
- Construção e testes do sensor de estacionamento;

- Desenvolvimento do painel do estacionamento a ser mapeado;
- Desenvolvimento da comunicação entre o microcontrolador e os periféricos;
- Desenvolvimento da gerência do projeto, a parte de *software*;
- Integração de todas as partes do sistema, testes e correção;
- Roteamento e construção dos PCB's.
- Construção do Protótipo, que corresponde à utilização do sistema, com todos seus componentes, em um estacionamento pequeno;
- Realização de testes do protótipo;
- Documentação e apresentação do projeto.

1.5 APRESENTAÇÃO DO DOCUMENTO

Este relatório tem por finalidade apresentar a realização do trabalho de conclusão do curso de Engenharia Industrial Elétrica, com ênfase em Eletrônica e Telecomunicações da UTFPR (Universidade Tecnológica Federal do Paraná). Este documento está dividido segundo a seguinte organização:

O capítulo 1 apresenta aspectos iniciais do projeto, tais como as justificativas e motivações da equipe, objetivos geral e específico, diagrama do sistema e a proposta de divisão do projeto em etapas.

O capítulo 2 trará informações referentes à fundamentação teórica do projeto. Será feita uma breve introdução a respeito das tecnologias envolvidas, para facilitar a compreensão da etapa de desenvolvimento do projeto.

O capítulo 3 explicará sobre as características técnicas das partes construídas, explicando como funcionam os principais componentes utilizados no projeto, tais como o sensor de distância, o microcontrolador utilizado, o módulo de rede sem fio e o computador utilizado para desenvolver a central e a interface gráfica.

O capítulo 4 dissertará sobre a rede sem fio desenvolvida, explicando como foi feita, seu funcionamento e o papel da central e dos módulos na mesma.

O capítulo 5 falará sobre os módulos, explicando como funcionam, como foram construídos, como os seus componentes interagem e como se comunicam com o motorista.

O capítulo 6 explicará o funcionamento da central, tanto no contexto da rede como na construção da interface gráfica.

O capítulo 7 falará sobre os testes realizados com alguns dos componentes utilizados, bem como do sistema funcionando como um todo.

O capítulo 8 apresenta o plano de negócios, com os dados necessários para avaliar a viabilidade do Sistema de Gerência de Vagas de Estacionamento como um produto no mercado.

O capítulo 9 traz as considerações finais sobre o projeto desenvolvido, apontando as dificuldades encontradas, os objetivos alcançados e sugestões para melhorias futuras.

2 FUNDAMENTAÇÕES

Nesse capítulo será mostrada a base teórica para o Sistema de Gerência de Vagas de Estacionamento. Será mostrado o funcionamento da comunicação SPI – *Serial Peripheral Interface*, que será utilizada para desenvolver a interface entre o módulo de rádio frequência e o microcontrolador, uma breve explicação a respeito deste, explicando seu funcionamento interno, suas funcionalidades e aplicações. Também será discorrido sobre redes sem fio e sobre comunicação em rádio frequência, que é utilizada para fazer a conexão entre os transceptores a serem utilizados, além de uma explicação sobre o ultrassom. Então será discorrido sobre a base técnica para se desenvolver o projeto, explicando sobre cada uma das partes que compõe o sistema desenvolvido. Será abordado sobre o funcionamento e características do transceptor de rádio frequência utilizado, sobre o microcontrolador selecionado, o MSP430G2553, da Texas Instruments, além de uma descrição do funcionamento do sensor ultrassônico HC-SR04. Por fim, será descrito a respeito da *Raspberry Pi*, focando na explicação sobre os pinos de GPIO da mesma e na sua utilização para fazer a comunicação SPI com o transceptor.

2.1 COMUNICAÇÃO SPI

Serial Peripheral Interface ou SPI é um protocolo de dados seriais síncronos utilizado em microcontrolador para comunicação entre este e um ou mais periféricos. Também pode ser utilizado entre dois microcontroladores.

A tecnologia de comunicação SPI foi desenvolvida pela Motorola para a linha de processadores da família MC68K. É um protocolo síncrono e opera no modo *full duplex*, permitindo a comunicação de um microcontrolador com diversos outros componentes, formando uma rede. Em modo "escravo", o comporta-se como um componente da rede, recebendo o sinal de relógio. Em modo "mestre", gera um sinal de relógio e deve ter um pino de entrada/saída para habilitação de cada periférico.

No protocolo SPI, a comunicação só pode ser feita entre dois pontos, sendo um deles o mestre e outro o escravo. (BARBOSA, 2013).

A comunicação SPI sempre tem um mestre. Isto é, sempre um será o mestre e o restante será escravo. Esta comunicação contém quatro conexões:

- MISO (*master in slave out*) - Dados do escravo para o mestre;
- MOSI (*master out Slave in*) - Dados do mestre para o escravo;
- SCK (*serial clock*) - cada vez que se inserir ou receber um novo bit, é necessário fornecer um pulso de clock através deste pino;
- CSN (*negated chip select*) - é utilizado quando se deseja realizar uma comunicação, sendo esta tanto de leitura quanto de escrita. Quando este pino estiver em nível lógico 1 ele não permitirá qualquer tipo de transferência de dados. Por isso quando se configura o dispositivo, coloca-se este periférico em nível lógico 0 (HORNING, HEILMANN, 2012).

2.1.1 Transmissão de Dados

Para iniciar uma comunicação, o mestre configura primeiro o *clock*, utilizando uma frequência menor que ou igual à frequência máxima do dispositivo escravo suporta. Estas frequências podem variar de 1 a 100 MHz.

O mestre envia para o escravo desejado o bit de seleção com valor lógico zero. Este é transmitido porque a linha de *chip select* é ativa em nível baixo.

Durante cada ciclo de *clock* da SPI, ocorre uma transmissão de dados *full duplex*: ocorre a transmissão de cada bit pelo MISO e MOSI, do mestre para o escravo e do escravo para o mestre, respectivamente.

As transmissões normalmente envolvem dois registradores de deslocamento de um determinado tamanho de *word*, um no mestre e um no escravo, pois eles são conectados em um anel. Os dados são geralmente deslocados a partir do bit mais significativo para o menos. Após serem transmitidos, os valores armazenados nos buffers do mestre e o escravo são atualizados. Em seguida, cada dispositivo trata o pacote recebido e utiliza-o de alguma forma, como escrevê-lo na memória. Se não houver mais dados para trocar, os registradores de deslocamento são carregados com novos dados e o processo se reinicia.

As transmissões podem envolver qualquer número de ciclos de *clock*. Quando não há mais dados a serem transmitidos, o mestre para de alternar o seu *clock*. Normalmente, em seguida, desmarca o escravo (WIKIPEDIA, 2014a).

2.1.2 Vantagens e Desvantagens da SPI

2.1.2.1 Vantagens

- Comunicação *full duplex*;
- Maior taxa de transmissão que *I²C* – *Inter Integrated Circuit* ou *SMBus* – *System Management Bus*, pois estes protocolos são *half duplex*;
- Flexibilidade de protocolo completo para os bits transferidos, não se limita a *words* de 8 *bits* e escolha arbitrária de tamanho da mensagem, conteúdo e finalidade;
- Possui uma interface simples de *hardware*;
- Requisitos de energia normalmente inferiores *I²C* ou *SMBus* devido à menores circuitos, não arbitra ou associa modos de falha;
- Escravos usam o relógio do mestre, e não precisam de osciladores de precisão;
- Os escravos não precisam de um endereço único - ao contrário de *I²C* ou *GPIO* ou *SCSI*;
- No máximo um sinal único de barramento por dispositivo (*chip select*), todos os outros são compartilhados;
- Não se limitando a qualquer velocidade máxima do *clock*, permitindo potencialmente alta taxa de transferência (WIKIPEDIA, 2014a).

2.1.2.2 Desvantagens

- Requer mais pinos em pacotes de IC do que *I²C*, mesmo na variante de três fios;
- Sem endereçamento dentro da banda, sinais fora de banda de *chip select* são necessários em barramentos compartilhados;
- Sem controle de fluxo por *hardware* pelo escravo (mas o mestre pode atrasar a próxima borda de *clock* para diminuir a taxa de transferência);
- Sem reconhecimento de *hardware* escravo (o mestre poderia estar transmitindo para nada e não sabe);

- Não apresenta protocolo de verificação de erros definida;
- Sem um padrão formal, validar a conformidade não é possível;
- Só trabalha em curtas distâncias em relação a RS-232 e RS-485, ou barramento CAN;
- Muitas variações existentes, tornando-se difícil de encontrar ferramentas de desenvolvimento, como adaptadores de host que suportam essas variações;
- SPI não suporta conexão a quente (adicionando nós na rede dinamicamente) (WIKIPEDIA, 2014a).

2.2 MICROCONTROLADORES

Um microcontrolador contém um processador, acesso a memória e periféricos de entrada/saída. Basicamente, o uso deste consiste no processamento de dados obtidos em um de seus periféricos, tendo como saída outro conjunto de dados.

Os microcontroladores diferenciam dos processadores, pois além dos componentes lógicos e aritméticos usuais de um microprocessador de uso geral, integra elementos adicionais em sua estrutura interna, conforme (WIKIPEDIA, 2014b), tais como: gerador interno independente de clock, memória SRAM, EEPROM e FLASH, conversores A/D, D/A; vários temporizadores/contadores; comparadores analógicos, PWM, diferentes tipos de interface de comunicação, incluindo USB, UART, I²C, SPI, JTAG, relógios de tempo real, circuitos para gerenciamento de energia no chip, circuitos para controle de reset, alguns tipos de sensores, interface para LCD e outras funcionalidades de acordo com o fabricante (OKI, MANTOVANI, 2013).

A arquitetura de um microcontrolador em geral consiste em um núcleo de processamento, barramento e periféricos:

- O núcleo de processamento consiste no processador de dados (cálculos, controle de fluxo de programa) e na administração dos periféricos;
- O barramento é dividido em dados e endereços, consiste nas linhas de comunicação entre o processador e os periféricos;

- Os periféricos caracterizam o conjunto de funcionalidades disponíveis pelo microcontrolador e são controlados pelo processador. Por exemplo, memória, porta serial, porta paralela e conversor A/D (OKI, MANTOVANI, 2013).

O seu consumo em geral é relativamente pequeno, normalmente na casa dos miliwatts e possuem geralmente habilidade para entrar em modo de espera, aguardando por uma interrupção ou evento externo (WIKIPEDIA, 2014b).

2.2.1 Aplicações

Microcontroladores são geralmente utilizados em automação e controle de produtos e periféricos, como sistemas de controle de motores automotivos, controles remotos, máquinas de escritório e residenciais, brinquedos, sistemas de supervisão, etc.

Por terem reduzido tamanho, custo e consumo de energia, e se comparados à forma de utilização de microprocessadores convencionais, aliados a facilidade de desenvolvimento de aplicações, sendo uma alternativa eficiente para controlar muitos processos e aplicações (WIKIPEDIA, 2014b).

2.3 ULTRASSOM

O ultrassom é uma onda mecânica com frequência superior ao limite audível do ouvido humano, que é aproximadamente 20.000 Hz. Dispositivos ultrassônicos operam de 20 kHz até vários GHz.

Usado em muitos campos, permite detectar objetos e medir distâncias. Ultrassonografia é usada tanto em medicina veterinária quanto em medicina humana. Em testes não destrutíveis, o ultrassom é usado para detectar falhas em produtos e estruturas. Industrialmente, o ultrassom é usado para limpar, misturar e acelerar processos químicos.

Um som é caracterizado por vibrações (variação de pressão) no ar. O ser humano normal médio consegue distinguir, ou ouvir, sons na faixa de frequência que se estende de 20 Hz a 20 kHz aproximadamente. Acima deste intervalo, os sinais

são conhecidos como *ultrassons* e abaixo dele, *infrassons*. (WIKIPEDIA, 2013). Na figura 2, são mostradas as faixas de frequência do som:

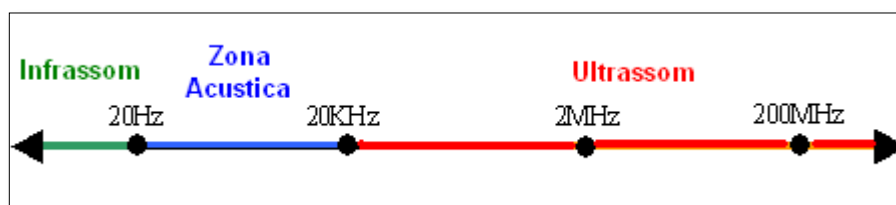


Figura 2 - Faixas de frequência do som
Fonte: Nepar (2010)

Os pulsos de ultrassom são gerados por cristais especiais, como o quartzo puro, que podem vibrar quando percorridos por uma corrente elétrica conveniente. Dessa forma, é possível gerar pulsos de ultrassom com frequências de vários MHz e com duração de apenas alguns milionésimos de segundo.

Ondas de ultrassom se propagam da mesma maneira que as ondas sonoras audíveis. Elas sofrem todos os fenômenos de reflexão, refração, difração e interferência, como todas as ondas.

Enviando-se um pulso para dentro de um material, ele será parcialmente refletido ao encontrar qualquer descontinuidade no meio, como uma mudança na densidade ou defeitos no próprio material. Enviando-se uma onda de ultrassom para dentro de uma barra de ferro que possua uma falha interna em virtude de um defeito de fabricação, esta onda será parcialmente refletida na descontinuidade e pode ser detectada por um receptor, que pode ser o próprio transdutor que a gerou ou um diferente.

Medindo o tempo decorrido entre o início do pulso e o seu retorno (eco), pode-se determinar a profundidade e dimensão da falha. Esse sistema de medida é utilizado industrialmente para detectar fissuras em tubulações de aço, em partes de motores e estruturas de edifícios e pontes (MARQUES, 2012).

2.4 REDE SEM FIO

Uma rede sem fio se refere a uma rede de computadores sem a necessidade do uso de cabos – sejam eles telefônicos, coaxiais ou ópticos – por

meio de equipamentos que usam radiofrequência (comunicação via ondas de rádio) ou comunicação via infravermelho.

O uso da tecnologia vai desde transceptores de rádio como satélites artificiais no espaço. Seu uso mais comum é em redes de computadores, servindo como meio de acesso à Internet através de locais como um escritório, um bar, um aeroporto, um parque, ou até mesmo em casa, etc.

As redes sem fio oferecem às empresas e usuários muitos benefícios, tais como a portabilidade, flexibilidade e produtividade aumentada, e baixo custo de instalação. As tecnologias *wireless* cobrem uma ampla área e isso é o que a diferencia das redes guiadas.

Além de serem adequadas a situações em que é necessária mobilidade, são flexíveis e de fácil instalação. Por outro lado, possui espectro finito e é suscetível a interferências de obstáculos, como paredes e carros. Embora os equipamentos sejam mais caros do que para redes tradicionais, a redução significativa dos custos de instalações a torna muitas vezes compensatórios. Os produtos *wireless* permitem criar, ampliar e interligar redes locais em ambientes internos ou externos sem a necessidade de utilização de fios ou cabos (ALVES et. al., 2010).

2.5 COMUNICAÇÃO EM RÁDIO FREQUÊNCIA

A comunicação RF funciona criando ondas eletromagnéticas em uma fonte e capta essas ondas em um destino particular. Essas ondas eletromagnéticas viajam através do ar a uma velocidade próxima à da luz (DIGI, 2002).

2.5.1 Funcionamento

Um transmissor RF agita um elétron, este passa a agitar os elétrons adjacentes e assim criando uma onda eletromagnética para fora do transmissor. Então essa onda irá influenciar elétrons em locais remotos e o receptor de RF detectar essa perturbação.

O sistema de comunicação de RF, em seguida, utiliza esse fenômeno em padrões específicos para representar certas informações.

Na maioria dos sistemas sem fio, um programador tem duas restrições primordiais: ele deve operar em uma determinada distância (intervalo) e transferir certa quantidade de informações dentro de um prazo (taxa de dados). Então a economia do sistema deve trabalhar fora (preço), juntamente com a aquisição de aprovações de agências governamentais (regulamentares e de licenciamento) (DIGI, 2002).

2.5.2 Regulamentação física e licença

Em 1985, a Comissão Federal de Comunicações (FCC) liberou a banda ISM (industriais, científicos e médicos) que oferece licença de operação livre dentro de determinadas frequências, com certas restrições técnicas de potência do espectro e na modulação. As bandas ISM são: de 902 a 928 MHz, de 2,4 a 2,4835 GHz e de 5 a 5.825 GHz (MELO, 2011).

2.5.3 Determinando o alcance

Existem essencialmente dois parâmetros para se determinar o alcance: A potência de transmissão, referente à quantidade de energia RF que sai da porta da antena do rádio, a sensibilidade do receptor que se refere ao sinal de nível mínimo que o rádio pode demodular e o ganho das antenas.

A potência de transmissão e a sensibilidade do receptor juntos constituem o que é conhecido como "*link budget*". O *link budget* é a quantidade total de atenuação de sinal que pode ter entre o transmissor e o receptor e ainda assim a comunicação ocorrer.

Outro fator que deve ser levado em consideração é a linha de vista. Quando se fala de radiofrequência, significa mais do que apenas ser capaz de ver a antena de recepção a partir da antena transmissora. A fim de se ter uma verdadeira linha de vista nenhum objeto (incluindo árvores, casas ou no chão) pode estar na zona de Fresnel, que é a área em torno da linha de vista que as ondas de rádio são transmitidas depois que saem da antena. Esta área deve ser clara, ou então a força do sinal vai enfraquecer (DIGI, 2002).

2.5.4 Taxa de envio de dados

As taxas de dados geralmente são ditadas pelo sistema - a quantidade de dados que deve ser transferida e com que frequência a transferência precisa tomar lugar. Menores taxas de dados permitem que o módulo de rádio tenha uma melhor sensibilidade de recepção e, portanto, mais alcance. Taxas de dados mais elevadas permitem a comunicação aconteça em menos tempo, usando mais energia para transmitir (DIGI, 2002).

2.6 MSP LAUNCHPAD



Fotografia 1 - MSP-EXP430G2 LaunchPad
Fonte: Autoria própria (2014)

O MSP-EXP430G2 *LaunchPad*, Fotografia 1, é uma placa de desenvolvimento na ordem de 10 dólares para a série de microcontroladores MSP430G2xx da Texas Instruments. O emulador integrado baseado em USB oferece todo o *hardware* e *software* necessários para desenvolver aplicativos para essa linha de microcontrolador

Esta placa dispõe de um soquete do tipo DIP que suporta até 20 pinos, permitindo que os dispositivos de linha MSP430™ possam ser colocados na mesma. Também oferece uma ferramenta de emulação de *flash on-board*, permitindo

interface direta a um computador para uma fácil programação, depuração e avaliação. A interface USB fornece uma conexão serial com um *baudrate* de 9600bps entre um dispositivo MSP430G2xxx e um computador ou uma placa conectada.

O MSP-EXP430G2 pode ser usado com IAR Embedded Workbench™ Integrated Development Environment (IDE) ou Code Composer Studio™ (CCS) IDE para escrever, fazer download e depurar aplicativos.

O depurador é discreto, permitindo que o usuário execute uma aplicação a toda a velocidade com pontos de interrupção de hardware e passo a passo disponível ao consumir sem recursos de hardware extra (TEXAS INSTRUMENTS, 2013a).

2.7 MSP430G2553

O MSP430G2553 pertence à família de microcontroladores do MSP430 de ultra baixa potência da Texas Instruments, com várias funcionalidades, tornando-o útil para as mais diversas aplicações.

O microcontrolador possui um processador RISC e registradores de 16 bits um tempo reduzido de execução.

Todas as operações, com instruções de fluxo de programa, são implementadas como operações de registrador em conjunção com sete modos de endereçamento para operandos de origem e quatro modos de endereçamento para operando de destino.

Quatro dos registradores, R0 até R3, são dedicados ao *program counter*, *stack pointers*, registrador de status e gerador de constante, respectivamente. Os registradores restantes são de uso genérico. Os periféricos são conectados com a CPU usando dados, endereço e controle de barramento e podem ser controlados com todas as instruções.

A lista de instruções consiste nas 51 instruções originais com três formatos e sete modos de endereçamento e instruções adicionais para um intervalo de endereçamento expandido. Cada instrução pode operar com dados *word* e *byte*.

Apresenta como principais funcionalidades: processador RISC de 16 bits, 8 canais de 10 bits para conversores de analógico para digital (A/D), 8 comparadores para sinal analógico, 16 kB de memória FLASH, 512 bytes de memória RAM, 2 temporizadores internos de 16 bits, interface UART e SPI e 16 pinos de entrada e saída (TEXAS INSTRUMENTS, 2013b).

2.7.1 Diagrama funcional

Na figura 3, um diagrama funcional do microcontrolador MSP430G2553:

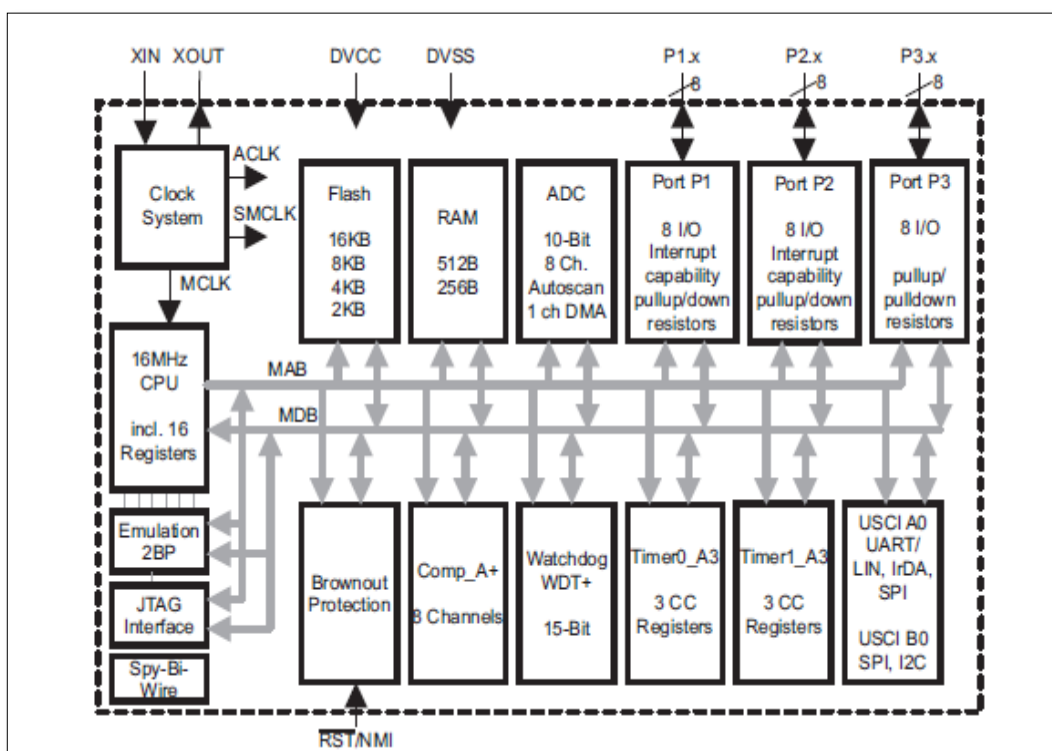


Figura 3 - Diagrama funcional do MSP430G2553

Fonte: Texas Instruments – Datasheet MSP430g2553 (2013)

2.7.2 Pinagem

Na figura 4, a pinagem do microcontrolador MSP430G2553:

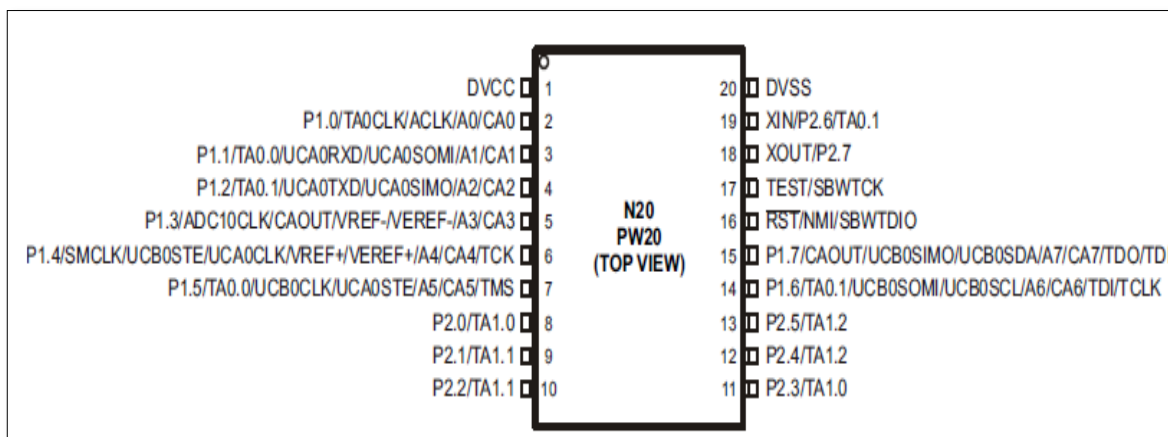
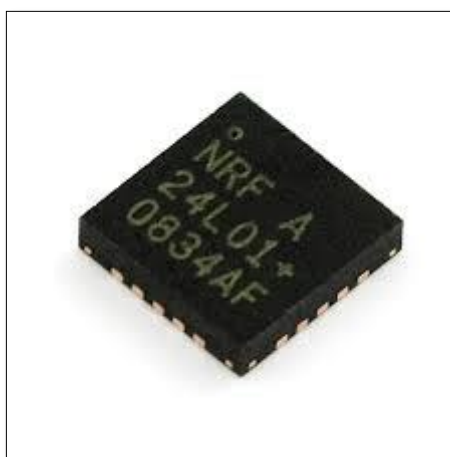


Figura 4 - Pinagem do MSP430G2553
Fonte: Texas Instruments (2013)

2.8 MÓDULO DE COMUNICAÇÃO NRF24L01+

O módulo de comunicação nRF24L01+, Fotografia 2, é utilizado para transmissão e recepção de radiofrequências. Este módulo contém em seu pacote um *chip* nRF24L01+ e um circuito periférico que se comunica diretamente com o microcontrolador através da SPI.



Fotografia 2 - Chip nRF24L01+
Fonte: Tenet Technetronics (2014)

O módulo possui um protocolo banda base integrado, sendo que é projetado para aplicações *wireless* de ultra baixa potência e para operação na banda de frequência de nível mundial ISM de 2,4 a 2,4835 GHz.

O módulo é configurado e operado por meio da SPI, que pode apresentar uma velocidade máxima de 10 Mbps. Através desta interface, o mapa de registradores está disponível, podendo enviar até 32 *bytes* de dados e utilizar até 6 armazenadores de dados para se comunicar com até seis transceptores diferentes através de uma conexão estrela.

Este circuito utiliza modulação GFSK (*Gaussian Frequency-Shift Keying*). Apresenta registradores configuráveis pelo usuário que alteram o funcionamento do transceptor, tais como canal de frequência, potência de saída e taxa de dados de ar.

O transmissor apresenta potência de saída configurável, sendo que ser de 0, -6, -12 e -18 dBm, sendo que a 0 dBm, utiliza uma corrente de 11.3 mA. Já o receptor possui filtros de canal integrados, possui uma sensibilidade -82 dBm a 2 Mbps e -85 dBm a 1 Mbps e -94 dBm para 250 Kbps, sendo que a 2Mbps consome uma corrente de 13.5 mA.

A taxa de dados de envio no ar suporta 250 kbps, 1 ou 2 Mbps. A taxa de dados de ar é combinada com dois modos de economia de energia faz com que o transceptor seja muito apropriado para aplicações de ultra baixa potência. (NORDIC SEMICONDUCTORS, 2008).

2.8.1 Diagrama de blocos do *chip* nRF24L01+

A figura 5 apresenta um diagram de blocos do *chip* nRF24L01+:

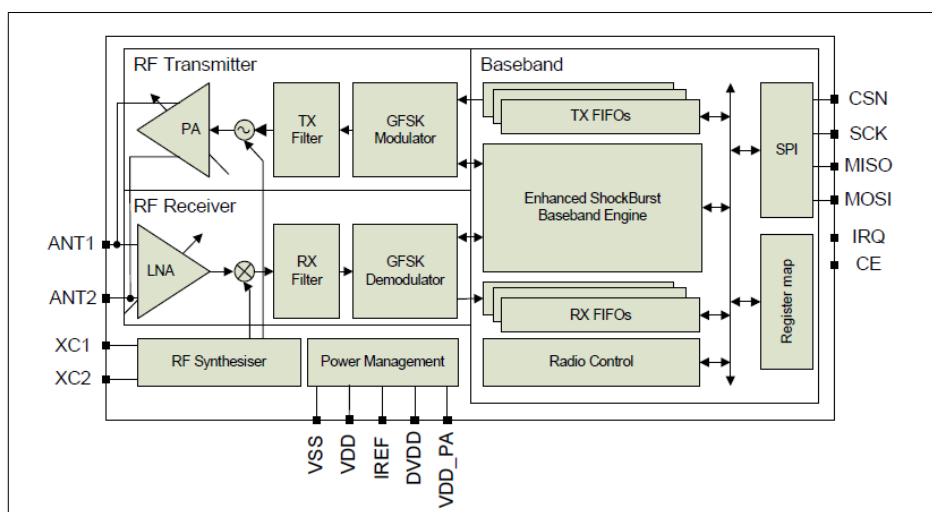
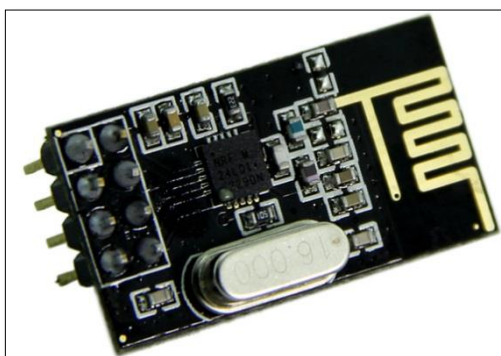


Figura 5 - Diagrama de blocos do chip nRF24L01+
Fonte: Nordic Semiconductor (2008)

2.8.2 Estrutura física do módulo de comunicação sem fio nRF24L01+

Para o desenvolvimento do projeto, foi-se utilizado o módulo de comunicação sem fio nRF24L01+, mostrado na fotografia 3.



Fotografia 3 – Módulo de comunicação sem fio nRF24L01+
 Fonte: Microduino (2011)

2.8.3 Pinagem módulo de comunicação sem fio nRF24L01+

Na figura 6, pode-se verificar a pinagem do módulo de comunicação:

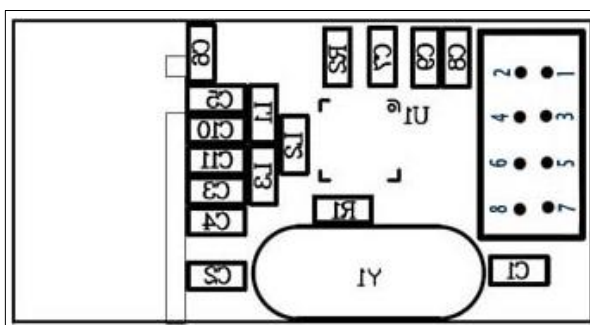


Figura 6 - Pinagem do módulo de comunicação sem fio nRF24L01+
 Fonte: Jaxcoder (2013)

Número	Pinos	Descrição Funcional
1	GND	Referência (0V)
2	VCC	Alimentação (de 1.9 a 3.6V)
3	CE	<i>Chip Enable</i> – Habilita TX/RX – Ativa em nível alto
4	CSN	<i>SPI Chip Select</i>
5	SCK	<i>SPI Clock</i>
6	MISO	<i>SPI Master In, Slave out</i>

Número	Pinos	Descrição Funcional
7	MOSI	SPI Master out, Slave in
8	IRQ	Pino de interrupção mascarável – aciona em nível baixo

Quadro 1 - Pinagem do módulo de comunicação sem fio nRF24L01+
Fonte: Nordic Semiconductor (2008)

2.9 SENSOR ULTRASSÔNICO HC-SR04

O sensor HC-SR04 (Figura 7) oferece uma medição sem contato de 2 a 400 centímetros e a precisão pode chegar a 3mm. O módulo inclui transmissores de ultrassom, receptor e circuito de controle (ELECTFREAKS, 2010).



Figura 7 - Sensor ultrassônico HC-SR04
Fonte: Deal Extreme (2014)

2.9.1 Pinagem

O sensor possui quatro pinos: VCC, Trig, Echo, GND (ELECTFREAKS, 2010), conforme mostra o quadro 2:

Pinos	Descrição Funcional
VCC	Alimentação (5V)
TRIG	Pulso de gatilho para o sensor (entrada)
ECHO	Largura de pulso correspondente à distância medida (saída)
GND	Referência (0V)

Quadro 2 - Pinagem do sensor HC-SR04
Fonte: ElecFreaks (2010)

2.9.2 Leitura da distância

Um curto pulso de ultrassom é transmitido e refletido por um objeto. O sensor recebe este sinal e o converte em um sinal elétrico. O próximo impulso pode ser transmitido quando o eco desapareceu. Este período de tempo é chamado de período de ciclo. Este não deve ser menor do que 50 ms. Se um pulso de disparo de largura de 10 μ s é enviado para o pino de sinal, o módulo de ultrassom emitirá oito pulsos de ultrassom de 40 kHz e detectar o eco de volta. A distância medida é proporcional à largura de impulso de eco (ITEADSTUDIO, 2010). A figura 8, mostra o diagrama de como funciona a leitura da distância:

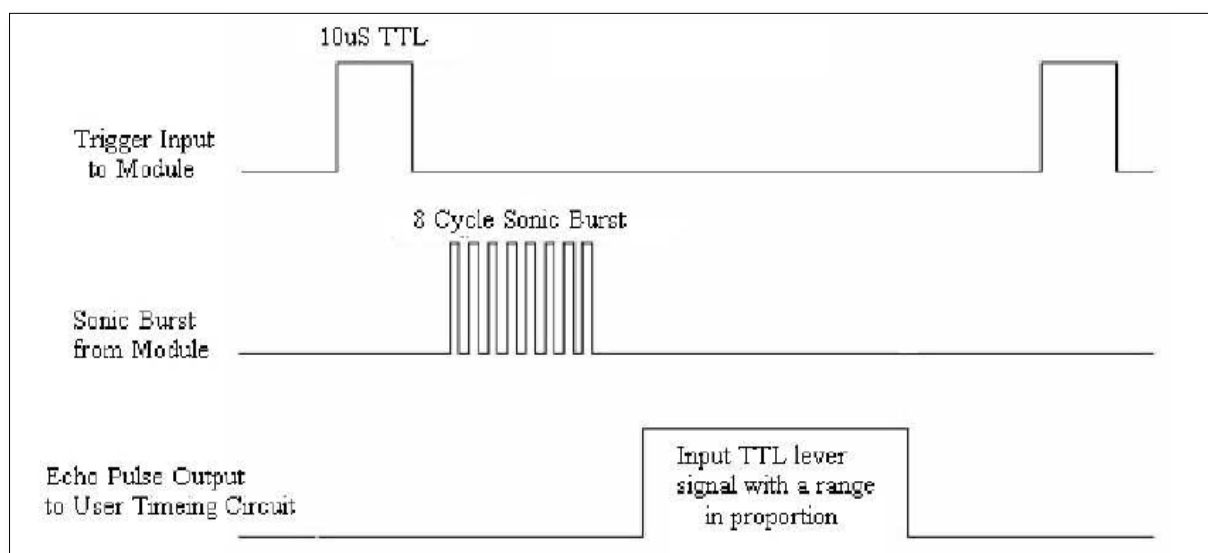


Figura 8 - Diagrama da leitura da distância pelo sensor
Fonte: Elecfreaks (2010)

2.10 RASPBERRY PI

O *Raspberry Pi* é um computador do tamanho de um cartão de crédito desenvolvido no Reino Unido pela Fundação *Raspberry Pi*. Todo o hardware é integrado em uma única placa. A CPU da *Raspberry Pi - Model B*, que é a utilizada no projeto, é baseado em um *system on a chip* (SoC) *Broadcom BCM2835*, que inclui um processador ARM1176JZF-S de 700 MHz, *GPUVideoCore IV* e 512 MB de

memória RAM em sua última revisão. O projeto não inclui uma memória não volátil, mas possui uma entrada de cartão SD para armazenamento de dados.

O *Raspberry Pi* (Figura 9) é compatível com sistemas operacionais baseados em *Linux*. O sistema operacional é normalmente armazenado no cartão SD. Qualquer linguagem que possa ser compilada na arquitetura ARMv6 pode ser usada para o desenvolvimento de software (WKIPEDIA, 2014c).



Figura 9 - *Raspberry Pi – Model B*
Fonte: Wikipedia (2014)

Além disso, também conta com 26 pinos de GPIO (figura 10), que podem ser usados para fazer uma interface serial, SPI e I²C, além de poderem ser usado como pinos de I/O.



Figura 10 - Pinagem da porta P1 da *Raspberry Pi – Model B*
Fonte: Wikipedia (2014)

Os pinos de GPIO sobre os pinos de header 2x13 incluem SPI, I²C, UART serial, 3,3V e 5V. Essas interfaces não são "plug and play" e requerem cuidados. Os níveis de tensão são GPIO 3,3V e não são tolerantes à 5V.

Todos os pinos de GPIO podem ser reconfigurados para proporcionar funções alternativas, SPI, PWM, I²C e assim por diante. Na reinicialização apenas pinos de GPIO 14 e 15 são atribuídos à função UART alternativa, estes dois podem ser ligados de volta para GPIO para fornecer um total de 17 pinos de GPIO (ELINUX, 2014).

Uma vez que se necessitava utilizar a SPI da placa para se comunicar com o módulo de comunicação sem fio nRF24L01+, teve-se a necessidade de utilizar a biblioteca "bcm2835.h". Esta é uma biblioteca em linguagem C que fornece acesso aos pinos de GPIO da *Raspberry Pi* e outras funções de I/O no chip *Broadcom BCM 2835*, permitindo que se possa controlar os mesmos e interagir com vários dispositivos externos conectados à placa.

Ela fornece funções para leitura de entradas digitais e definir saídas digitais, usando SPI e I²C, e para acessar os temporizadores do sistema. A detecção de eventos do pino é suportada por *polling* (interrupções não são suportadas).

As funções da SPI da biblioteca permitem controlar a interface SPI0 BCM 2835, que permite enviar e receber dados por SPI.

Quando a função que inicia a SPI é chamada, muda-se o comportamento dos pinos da interface SPI de seu comportamento padrão de GPIO a fim de suportar a SPI. Enquanto esta está em uso, não será capaz de controlar o estado dos pinos através do método habitual de escrever na SPI. Quando a função de encerrar a SPI é chamada, todos os pinos da SPI voltam a serem entradas e podem ser configurados e controlados com as funções normais de pinos de GPIO (AIRSPAYCE, 2012).

Foi escolhido o *Raspberry Pi* pela sua versatilidade no que tange aos periféricos, por possuir pinos de GPIO que permitem conectar o nRF24L01+ diretamente a mesma, além de contar com várias outras interfaces, como Ethernet, DVI, HDMI, que permitem que o projeto possa ter mais opções de periféricos com os quais se pode fazer a interface.

3 REDE SEM FIO PARA COMUNICAÇÃO DE DADOS

Neste capítulo, será discorrido sobre a rede sem fio criada para interligar os módulos e a central.

3.1 COMPOSIÇÃO DA REDE

A rede construída é dividida em redes independentes com conexão direta à central. A rede tem a função de enviar um pacote de configuração a um módulo de cada uma delas e receber todos os estados de vaga por este. Já os módulos são incumbidos de receber e enviar a configuração para outro módulo e assim por diante, dependendo da quantidade de módulos determinada pela central. Paralelamente, enviam e recebem os estados de vaga repassados através dos mesmos, pelo pacote de resposta, enviado cada vez que um módulo recebe a configuração.

Na figura 11 é apresentado como a comunicação sem fio funciona numa rede com 2 módulos:

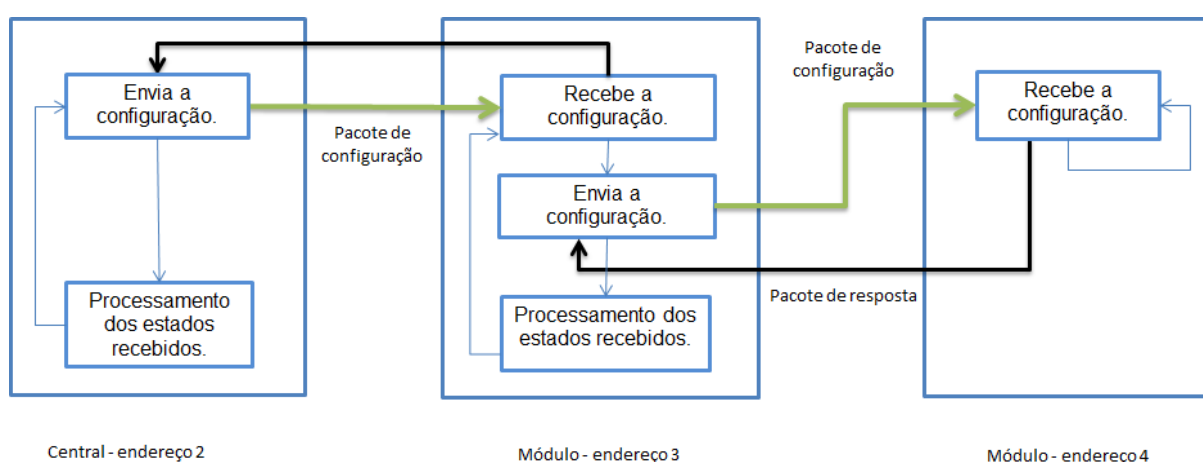


Figura 11 - Esquema de uma rede independente com 2 módulos.
Fonte: Aatoria própria (2014)

Os elementos básicos para a construção da rede e comunicação de dados entre os elementos da rede são:

- Pacote de configuração e de resposta;
- Endereço e canal de comunicação do nRF24L01+;
- Processamento dos estados de recebimento.

3.1.1 Pacote de configuração e de resposta

O pacote de configuração contém as seguintes informações: endereço de origem (endereço do módulo que está enviando), para onde está sendo enviada (que está recebendo a informação), identificação de que é um pacote de configuração, número de módulos por configurar, o canal pelo qual envia a configuração, a distância limite para o sensor e o identificador de rede. É enviado sempre que se envia a configuração para o próximo módulo. Com este, o módulo recebe um endereço próprio, um canal de recebimento de pacote, uma identificação de rede independente e a distância limite para considerar o veículo estacionado. O pacote vai sofrendo alterações ao longo da rede independente, para configurar a todos de forma correta.

Na figura 12 pode-se ver uma amostra da composição do pacote de configuração em cada um dos elementos de uma rede configurada para ter dois módulos:

Central - endereço 2	Módulo 1 - endereço 3	Módulo 2 - endereço 4
<p>Pacote de configuração que envia:</p> <p>Endereço de origem do pacote: 2 Endereço para onde vai o pacote: 3 Identificador de envio de configuração/resposta: 1 (envio de config.) Identificador de rede: 0 (Rede 0) Canal de envio de configuração: 1 Número de módulos por configurar: 2 Distância limite: 40 (cm)</p>	<p>Pacote de configuração que envia:</p> <p>Endereço de origem do pacote: 3 Endereço para onde vai o pacote: 4 Identificador de envio de configuração/resposta: 1 Identificação de rede: 0 Canal de envio de configuração: 3 Número de módulos por configurar: 2 Distância limite: 40 (cm)</p>	<p>Pacote de configuração que envia:</p> <p>Endereço de origem do pacote: - Endereço para onde vai o pacote: - Identificador de envio de configuração/resposta: - Identificação de rede: - Canal de envio de configuração: - Número de módulos por configurar: - Distância limite: -</p>

Figura 12 - Pacote de configuração enviado por cada elemento da rede.
Fonte: Autoria própria (2014)

O pacote de resposta contém o endereço de quem está enviando a mesma, o de quem está recebendo (que é quem enviou a configuração), uma identificação que é um pacote de resposta e uma da rede, além dos estados de vagas coletados. Como se pode observar, os estados são passados e acumulados do último módulo da rede independente até a central.

Na figura 13 é possível ver um exemplo do pacote de resposta enviado por cada um dos elementos de uma rede configurada para ter dois módulos:

Central - endereço 2	Módulo 1 - endereço 3	Módulo 2 - endereço 4
Pacote de resposta recebido: Endereço de origem do pacote: 3 Endereço para onde vai o pacote: 2 Identificador de envio/resposta: 2 (resposta) Identificação de rede: 0 (Rede 0) Estado coletado - Módulo 1: 0 Estado coletado - Módulo 2: 0	Pacote de resposta enviado: Endereço de origem do pacote: 3 Endereço para onde vai o pacote: 2 Identificador de envio/resposta: 2 Identificação de rede: 0 (Rede 0) Estado coletado - Módulo 1: 0 Estado coletado - Módulo 2: 0	Pacote de resposta enviado: Endereço de origem do pacote: 4 Endereço para onde vai o pacote: 3 Identificador de envio/resposta: 2 Identificação de rede: 0 (Rede 0) Estado coletado - Módulo 2: 0

Figura 13 - Exemplo do uso do pacote de resposta na rede.
Fonte: Autoria própria (2014)

3.1.2 Endereço e canal de comunicação do nRF24L01+

O módulo de comunicação possui algumas configurações básicas que não são alteradas. Já o endereço e o canal de comunicação deste são alterados de acordo com a situação (primeira configuração ou funcionamento normal) e no tipo de comunicação (envio ou recebimento de dados), de forma a fazer com que os módulos possam ser configurados, comunicarem-se e funcionarem da maneira correta.

O endereço e canal da primeira configuração são utilizados pelo módulo de comunicação apenas quando o mesmo vai ou receber ou enviar a mesma.

Uma vez recebida ou enviada, passa-se a recebê-la ou enviá-la apenas pelo endereço e canal de funcionamento normal, que corresponde ao endereço da rede independente em que o módulo está presente e o canal de recebimento de configuração, que apresenta o valor vindo do pacote de configuração. Já para o envio, o canal é alterado, pois será por este que irá enviar a configuração para o próximo módulo, que por sua vez receberá a configuração. Faz-se isso para garantir um envio correto de pacotes entre os módulos de uma mesma rede.

3.1.3 Processamento dos estados recebidos

Faz-se necessário extrair do pacote de resposta, recebido no envio da configuração, os estados das vagas dos módulos anteriores, além do próprio módulo

e colocá-los no pacote de resposta para ser enviado ao próximo no recebimento de configuração subsequente. Este processo é feito constantemente até que todos os estados cheguem à central, que irá detectar todos os estados enviados por cada uma das redes independentes.

3.2 CRIAÇÃO DE REDES INDEPENDENTES

Para possibilitar a criação de redes independentes, que permite a separação em áreas com conexão direta com a central, altera-se o endereço de comunicação pelo transceptor, em função do valor que identifica cada uma das redes. A quantidade das mesmas e quantos módulos cada uma terá são definidos dentro da central.

Antes do efetivo funcionamento da rede, há uma etapa para configurar cada módulo dentro de uma rede conforme o número de elementos definidos na central pelo usuário. Depois de configurado parte-se para a rede seguinte. Após a etapa de configuração, enviam-se os pacotes de dados de módulo a módulo para receber os estados presentes, fazendo uma varredura elemento por elemento e rede por rede, ciclo este que se repete até ser interrompido.

Nos módulos, o identificador da rede independente, na qual cada módulo estará presente, é enviada juntamente com o pacote de configuração. Uma vez recebida a primeira configuração, o módulo recebe o valor da mesma, que irá alterar o endereço de comunicação normal entre os transceptores de uma mesma rede independente.

3.3 ENVIA CONFIGURAÇÃO E RECEBE ESTADOS DAS VAGAS

Para não precisar configurar toda a rede novamente quando houver uma perda na conexão entre a central e os módulos, foi desenvolvido um código de envio de configuração. Com isso o transceptor, num primeiro momento, tenta enviar os pacotes de configuração, para o endereço e o canal de comunicação de primeira configuração, como deve funcionar com a rede independente já configurada.

Quando a central é paralisada, ou seja, pára de funcionar, os módulos ficam esperando que a central envie a configuração. Uma vez que a central volta a funcionar, a tentativa de se comunicar com o módulo será primeiramente com o endereço e canal de funcionamento normal, conectando-se automaticamente com o módulo mais próximo, permitindo assim que a rede volte a funcionar novamente, com a configuração modificada.

Quando um dos módulos da rede pára de funcionar, primeiramente a central avisa que em alguma das redes independentes ocorreu algum erro de envio em um dos módulos e também não tenta mais estabelecer conexão com a que deu problema. Na rede independente, o elemento que receberia a configuração e o que enviaria a mesma para o módulo que não está funcionando fica esperando que o problema ocorrido neste seja corrigido e seja recolocado na rede. Com o mesmo esperando para ser configurado, deve-se parar o recebimento de dados na central, colocá-la em modo de configuração, configurar a rede que perdeu a conexão, fazer ela se comunicar com a rede que os próprios módulos irão se reconectar automaticamente, bastando colocar novamente no modo de funcionamento normal para que a mesma funcione como antes.

O pacote de configuração é enviado e espera um pacote de resposta, enviado pela função que recebe a configuração no módulo com endereço imediatamente maior. O pacote recebido é verificado e se for correto, continua a execução do programa. Caso contrário, tentará mandar de novo até que a verificação dê um resultado positivo. Os estados de vagas recebidos nesse pacote serão colocados no pacote de resposta que será enviado para o módulo com endereço imediatamente inferior e, sendo o primeiro módulo da rede independente, para a central.

Na figura 14 é mostrado um fluxograma do funcionamento do envio de configuração explicado anteriormente.

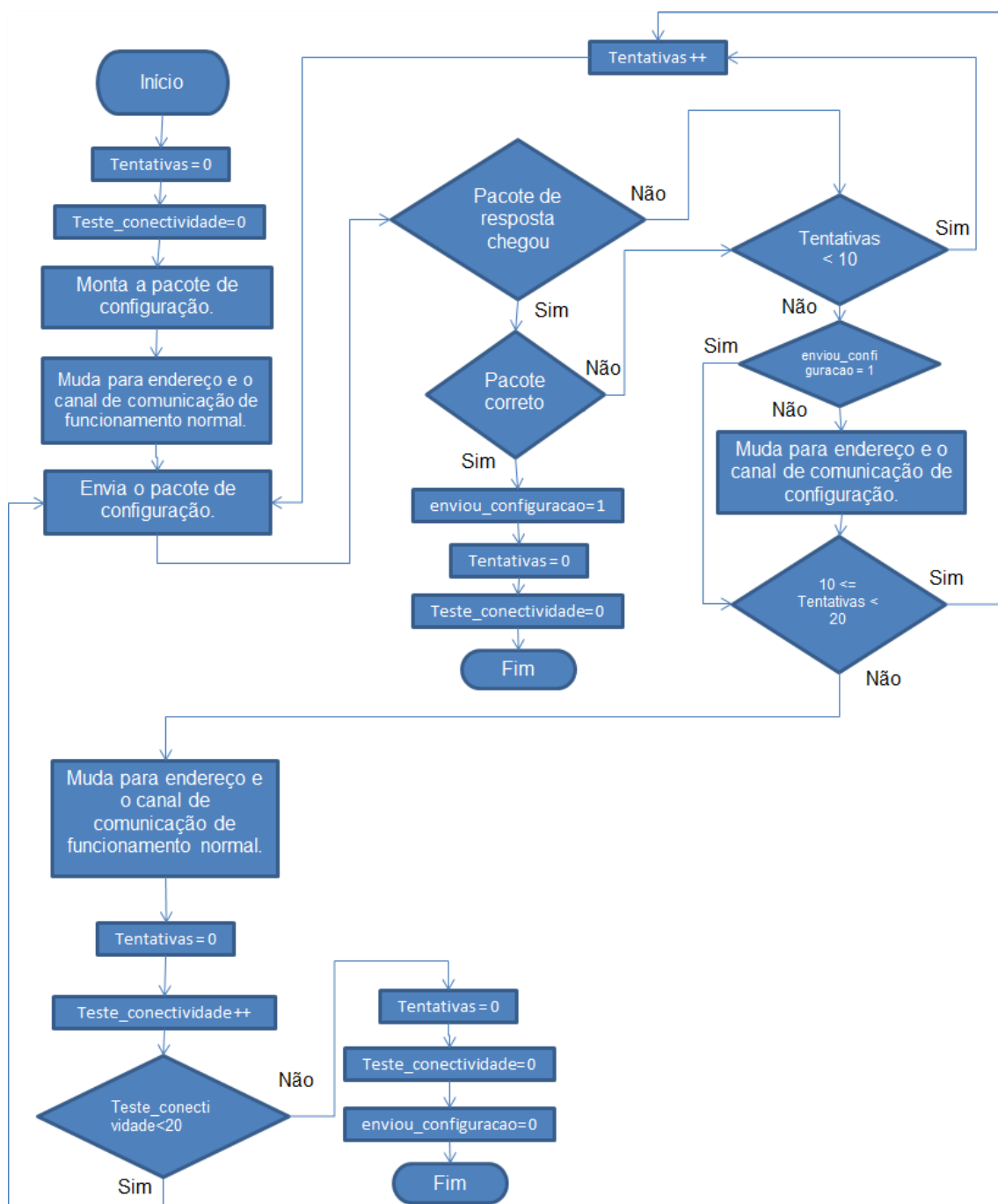


Figura 14 - Fluxograma do envio de configuração.
 Fonte: Autoria própria (2014)

3.4 RECEBE CONFIGURAÇÃO E ENVIA ESTADOS DAS VAGAS

Primeiramente, espera-se a chegada do pacote de configuração. Este chegando, é colocado num vetor. A seguir é verificado se é o pacote esperado, avaliando se o endereço de onde veio é o endereço imediatamente anterior, se o

endereço para onde está sendo enviado é o do próprio módulo e se o identificador de rede independente representa o mesmo. No caso de ser a primeira configuração, irá verificar somente se o que está recebendo é um pacote de configuração, ou não. Passando nas verificações, enviará um pacote de resposta ao módulo que está enviando a configuração, para sinalizar que o mesmo foi recebido corretamente, enviando junto também os estados das vagas coletados. O pacote recebido é então processado e altera as configurações do módulo, caso tenha sido modificado algo na informação que é enviada constantemente.

Na figura 15, é apresentado um fluxograma do funcionamento do recebimento de configuração explicado acima:

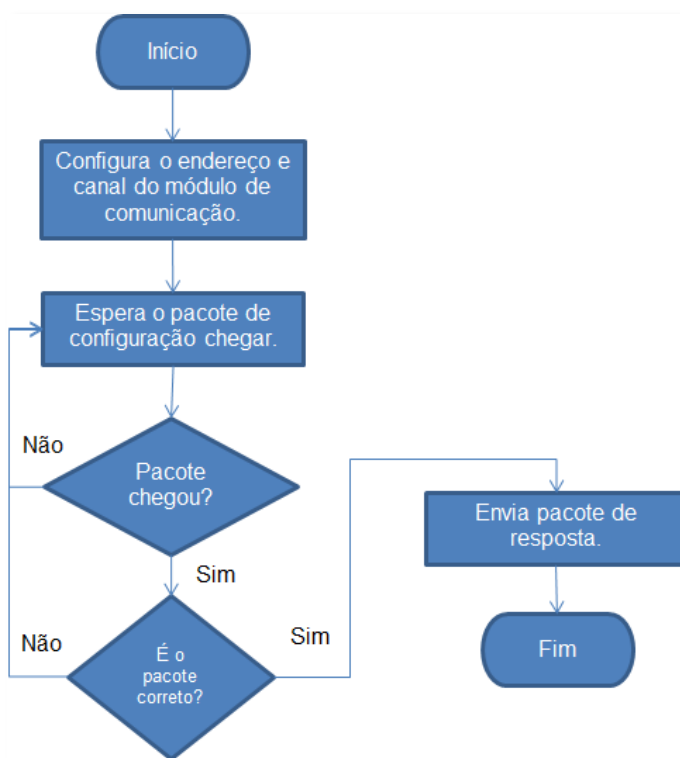
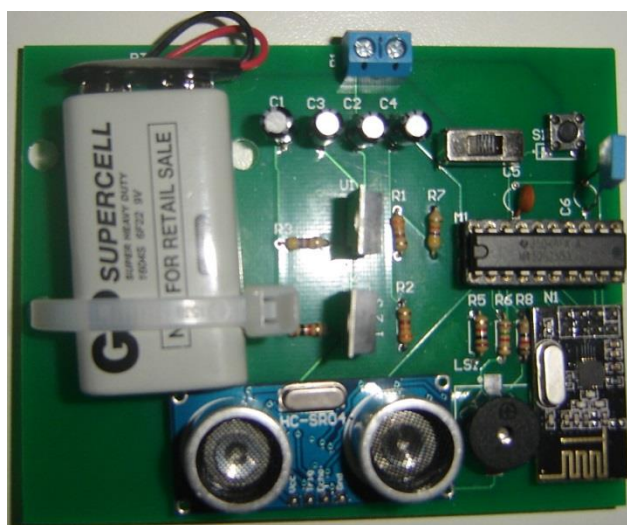


Figura 15 - Fluxograma do recebimento de configuração.
Fonte: Autoria própria (2014)

4 MÓDULO DE AQUISIÇÃO E COMUNICAÇÃO DE DADOS

Este capítulo tem por objetivo documentar o módulo que fica em cada uma das vagas e é composto por um microcontrolador, um comunicador RF e um sensor ultrassônico, explicando seus componentes e as suas principais aplicações.

4.1 MÓDULO PRESENTE NAS VAGAS



Fotografia 3 - Módulo sensor, sem a parte da sinalização luminosa.
Fonte: Autoria própria (2014)

O módulo (Fotografia 3), estará em cada uma das vagas do estacionamento, tendo como principais componentes um transceptor nRF24L01+ para fazer a comunicação sem fio, um sensor ultrassônico HC-SR04, para aferir a distância entre em relação ao carro e um microcontrolador MSP430G2553, da Texas Instruments, que interage com os componentes citados acima, administrando e controlando-os, bem como os outros periféricos conectados ao mesmo.

Cada módulo repete indefinidamente o seguinte processo: recebe o pacote de configuração vindo da central ou do módulo que o antecede, depois envia um para o próximo módulo, caso tenha, a seguir coleta os estados de vaga recebidos e acrescenta o seu próprio ao vetor de que será enviado através do transceptor. Após isso, utiliza-se do sensor ultrassônico para aferir o valor da distância e usa-se a

variação da mesma para avaliar o estado da vaga, acionando os alarmes sonoros e luminosos conforme a situação da vaga.

Os estados das vagas são repassados de módulo para módulo do último até o primeiro, os quais são repassados e interpretados pela central, que mostrará o estado de cada uma das vagas do estacionamento numa interface gráfica.

Na figura 16, tem-se um fluxograma mostrando o funcionamento do módulo:

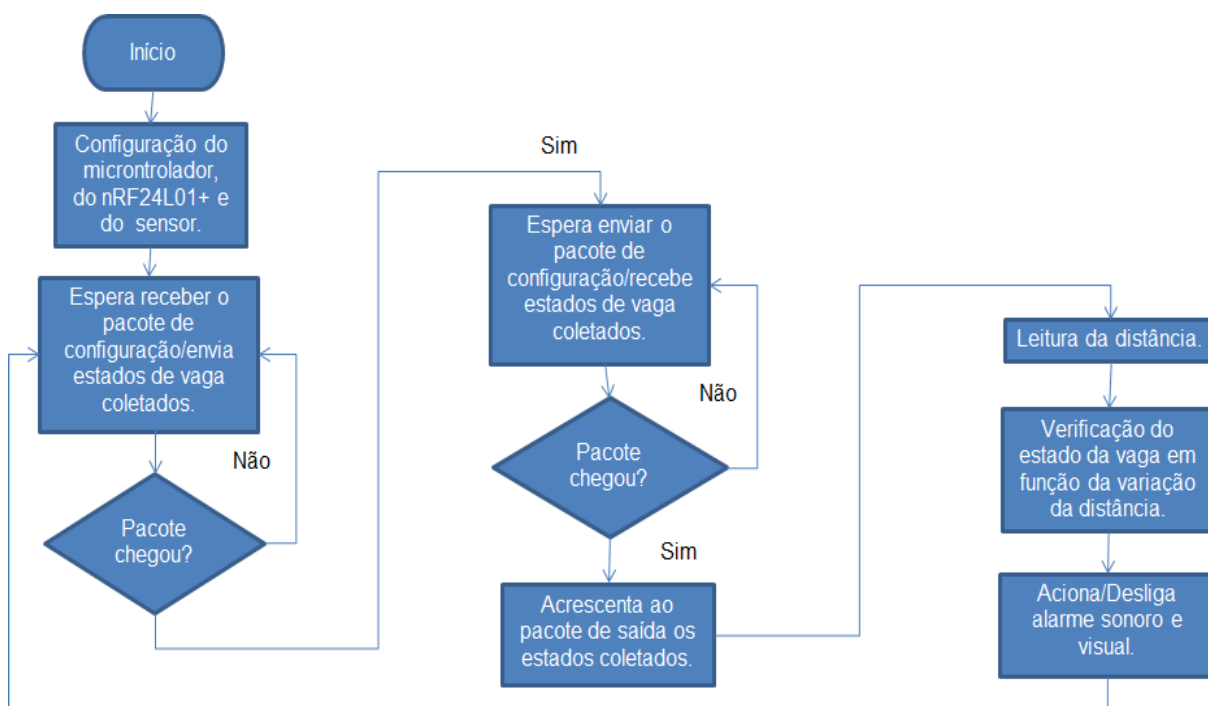


Figura 16 - Fluxograma do funcionamento do módulo
 Fonte: Autoria própria (2014)

O *software* foi desenvolvido em linguagem C e em *assembly* no IAR *Embedded Workbench™ Integrated Development Environment* (IDE) e colocado no microcontrolador utilizando o IAR integrado ao MSP *Launchpad*.

4.1.1 Interação entre o microcontrolador e o módulo de comunicação RF

O microcontrolador se comunica com o nRF24L01+ através da SPI, que no projeto feito foi programado manualmente para que os pinos se comportassem como comunicação SPI. A programação desta foi feita em linguagem *assembly* e utilizaram-se algumas rotinas para desenvolver a SPI em (HORNING, HEILMANN,

2012), sobretudo para escrever e ler um *byte* através da mesma. Como exemplo, na figura 17 se detalha a parte de escrita de um *byte*, através da SPI:

```

ORG 0FFA0h

NAME write_data

#include "msp430g2553.h"

PUBLIC write_data

DC16 write_data

RSEG CSTACK

RSEG CODE

;DATA
write_data MOV.B #8,R6

LOOPB:      RLC.B R12           ;rotaciona registrador para a esquerda
            JNC MOVE2_0       ;se for o bit de carry for 0 (não deu
                               ;overflow)

            JC MOVE2_1        ;se for o bit de carry for 1 (deu overflow)
MOVE2_0     BIC.B #80h,P1OUT   ;manda bit 0 para o pino MOSI
            JMP PULSO2
MOVE2_1     BIS.B #80h,P1OUT   ;manda bit 1 para o pino MOSI
PULSO2      BIS.B #20h,P1OUT   ;pulso de clock para o pino
            BIC.B #20h,P1OUT   ;SCLK

            DEC R6
            JNZ LOOPB ;8x     ;repete-se o processo 8 vezes

            BIC.B #20h,P1OUT

            RET

END

```

Figura 17 - Escrevendo um byte via SPI, utilizando assembly.
Fonte: Autoria própria (2014)

Na figura 18, a interligação física entre o nRF24L01+ e o microcontrolador:

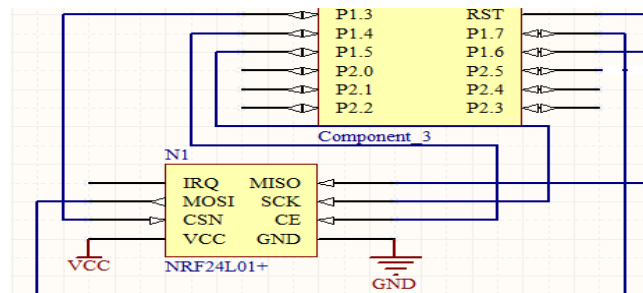


Figura 18 - Interligação entre o nRF24L01+ e o microcontrolador (Component_3)
Fonte: Autoria própria (2014)

4.1.2 Lendo a distância a partir do sensor ultrassônico

O sensor ultrassônico envia um sinal sonoro, que tem como velocidade 340 m/s, tornando possível calcular a distância entre o objeto e o sensor a partir da medição do tempo do pulso enviado através do pino de eco do sensor para o microcontrolador.

Com este objetivo, primeiramente foi necessário configurar a saída do pino de eco como entrada para o microcontrolador, mais especificamente para acionar a interrupção de timer e um pino de I/O ser usado como entrada ao trigger do sensor. Os dois estão interligados conforme a figura 19:

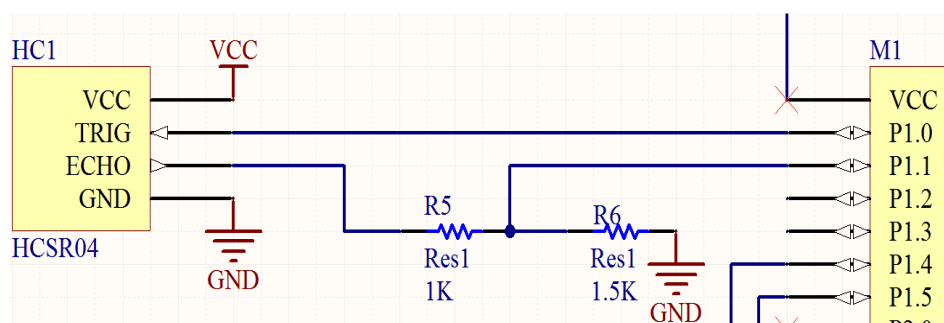


Figura 19 - Interligação entre o sensor (HC1) e o microcontrolador (M1)
Fonte: Autoria própria (2014)

Enviando através do pino conectado com o trigger do sensor um pulso de 10 us, aciona-se a leitura do sensor. A seguir, o pino de eco envia um pulso de largura proporcional ao tempo. Tanto na borda de subida quanto na descida do mesmo, é lido o valor do *timer*, é encontrada a diferença entre os valores, obtendo-se a duração do pulso. Para calcular a distância, é necessário dividir o tempo encontrado por dois, pois o pulso corresponde ao tempo de ida e volta do ultrassom e presume-se que o tempo que para o sinal emitido refletir e ser recebido novamente pelo sensor é o mesmo.

Na figura 20, tem-se o trecho de código que mostra o tratamento da interrupção de timer para ler o pulso de eco:

```

void calcula_tempo() {
    static unsigned int anterior=0;

    //se o pino 1.1 apresenta uma borda de subida, lê-se o valor do
    timer.
    if(state==0) {
        anterior = TACCR0;
        state = 1;
    }

    //se o pino 1.1 apresenta uma borda de descida, verifica-se o valor
    que esta no timer e subtrai-se do valor medido na borda de subida para
    encontrar o tempo que o ultrassom levou para ir e voltar.
    else {
        //s = v*t; como v=340 m/s e t corresponde ao tempo em
        microssegundos.
        distancia = (TACCR0 - anterior)/29/2;

        state = 0;
        leu_sensor = 1;
        __delay_cycles(20);
    }

    TACCTL0 &= ~0x01;
}

```

Figura 20 - Tratamento da interrupção de timer para determinar a distância.
Fonte: Autoria própria (2014)

Uma vez que o *clock* é configurado para se ter uma frequência de 1 MHz, o valor lido pelo *timer* corresponde ao intervalo de tempo em microssegundos. Utilizando-se a relação em que o deslocamento é igual à velocidade multiplicado pelo tempo, pode-se calcular a distância, utilizando-se a velocidade do som e o tempo contado, em microssegundos.

Como podem ocorrer algumas interferências na leitura, estas acabam variando um pouco, não ficando fixa numa mesma distância, então na interpretação da variação da distância, sobretudo nas verificações de parada, há uma tolerância introduzida para serem ignorados erros desprezíveis, que seriam de aproximadamente 2 cm.

4.1.3 Verificação do estado da vaga em função da variação da distância

O sensor utilizado mede continuamente a distância até um carro que no qual o ultrassom reflete. Então, utilizou-se a variação da distância para determinar o estado da vaga. Os estados possíveis de serem interpretados pelo programa são:

não estacionado (fora da vaga), estacionado (dentro da distancia limite), chegando e saindo da vaga.

Para verificar o estado da vaga, primeiramente verifica-se se o carro está na vaga ou não. A distância considerada dentro da vaga é a partir dos 2,2 m. Se o carro estiver dentro desse limite, é verificado se o carro está parado ou não. Caso considerado parado, mede-se a distância, e se for menor que o limite, é considerado estacionado correto. No caso do carro estiver em movimento, e a distancia diminuindo, é considerado chegando e se estiver aumentando, o estado da vaga é interpretado como saindo.

Como para distâncias muito grandes ou simplesmente sem nenhum carro, o sensor acaba lendo valores aleatórios, sendo necessária uma filtragem dos valores lidos. Para certificar se o módulo está no estado fora da vaga, é verificado se a distancia é maior que 2,2 m. Então é criada uma “barreira” que só haverá uma mudança no estado caso o sensor detecte que um carro está entrando, ou seja, se a distancia lida chegou entre 1,6m e 2,2 m, caso contrário, ele ignora e interpreta o estado como fora da vaga.

O sensor só vai interpretar se o carro estiver entrando caso a distancia esteja diminuindo. Para certificar a parada do carro, verifica-se constantemente se a distância varia ou não. Se não variar uma vez, fixa-se esta e ela passa a ser o valor de referência. Ficando na mesma distância por algumas leituras, o veículo é considerado parado.

4.1.4 Sinalização na vaga para o motorista

O carro estando dentro da distância correspondente a duas vezes a distância limite estabelecida, a sinalização sonora apita com intervalos que variam com a distância, ficando mais rápido na medida em que a mesma diminui. Uma vez que o carro encontra-se dentro da distância limite, o som fica contínuo e a sinalização luminosa é acionada para mostrar que deve parar o veículo. Uma vez detectada a parada, o som para e após certo tempo, a sinalização visual é desligada também.

4.1.4.1 Sinalização luminosa

Como os módulos precisam estar em uma altura igual ao ponto mais à frente do carro e essa região não é visível ao motorista, foi necessário desenvolver uma sinalização luminosa que deve ficar nessa área visível. Para isso, esse recurso foi feito separado do sensor. Um borne de duas saídas foi adicionado na placa do módulo para fazer a ligação da sinalização luminosa ao pino do microcontrolador e a referência da placa. Essa conexão será feita através de um cabo. A figura 21 mostra como funciona conexão entre o pino, borne no módulo e a referência:

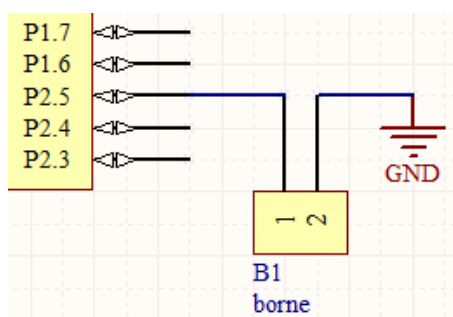


Figura 21 - Ligação entre o pino do microcontrolador, borne a referência da placa.
Fonte: Autoria própria (2014)

O circuito da sinalização luminosa é construído por um *LED*, um resistor de 150Ω e um borne, sendo este o conector com a placa do sensor, conforme a fotografia 4:



Fotografia 4 - Sinalização luminosa.
Fonte: Autoria própria (2014)

4.1.4.2 Sinalização sonora

A sinalização sonora corresponde a um *buzzer* em série com um resistor de valor bem baixo (da ordem de 10Ω), para gerar uma corrente mais alta, fazendo com que o buzzer faça um barulho o mais alto possível, sem também utilizar muita corrente. Está interligado ao microcontrolador conforme mostra a figura 22:

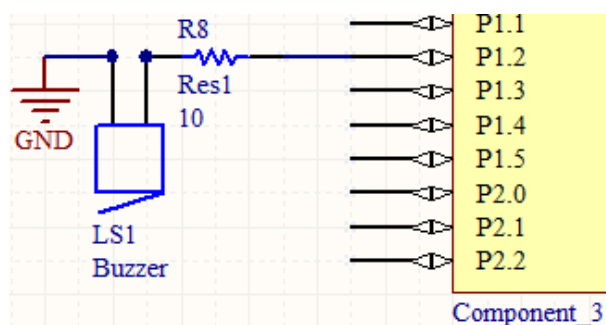


Figura 22 - Interligação entre a sinalização sonora (LS1) e o microcontrolador (Component_3).
 Fonte: Autoria própria (2014)

4.2 ALIMENTAÇÃO

O circuito do módulo é alimentado por uma bateria de 9V. Como o sensor e o microcontrolador, conforme verificado nas especificações de cada um, podem ser alimentados com uma tensão entre 1.9 e 3.6V, considerou-se uma tensão de saída para o primeiro circuito regulador um valor de aproximadamente 3V. Já para o sensor ultrassônico, faz-se necessário que seja alimentado com 5V, que é a tensão de saída determinada para o segundo regulador. Os dois circuitos reguladores utilizam o circuito integrado LM317, da Texas Instruments.

O LM317 é um regulador de tensão positiva de três terminais ajustável capaz de fornecer mais de 1,5 A, durante um intervalo de saída de voltagem de 1,25 V a 37 V. É excepcionalmente fácil de usar e requer apenas duas resistências externas para ajustar a tensão de saída. Além disso, tanto a linha quanto a regulação de carga são melhores do que nos reguladores fixos convencionais (TEXAS INSTRUMENTS, 2004). Na figura 23, um circuito genérico para regular a tensão de saída é mostrado:

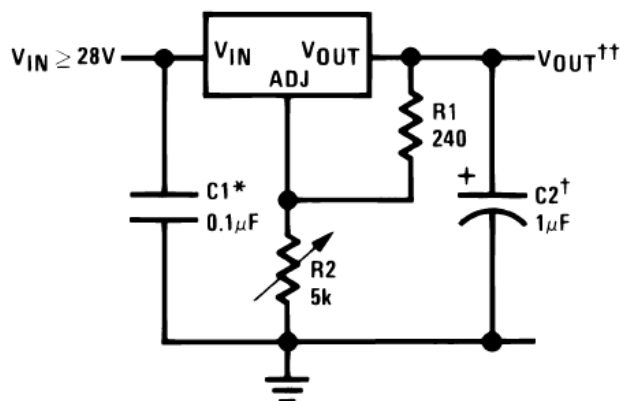


Figura 23 - Circuito regulador com LM317
Fonte: Texas Instruments (2004)

Para determinar os resistores para gerar certa tensão de saída, foi utilizada à equação oferecida pela Texas Instruments:

$$V_{out} = 1,25 * \left(1 + \frac{R_2}{R_1}\right) + I_{ADJ}(R_2).$$

Para construí-los, considerou-se a corrente que flui pelo pino de ajuste (I_{ADJ}) desprezível, escolheu-se o valor de tensão (3V e 5V), fixou-se R1 em 330Ω e calculou-se o R2 para cada tensão de saída. Para tensão de 3V, R2 deve ser igual a 470Ω e para 5V, 1kΩ. Nas figuras 24 e 25, são mostrados os dois circuitos reguladores construídos:

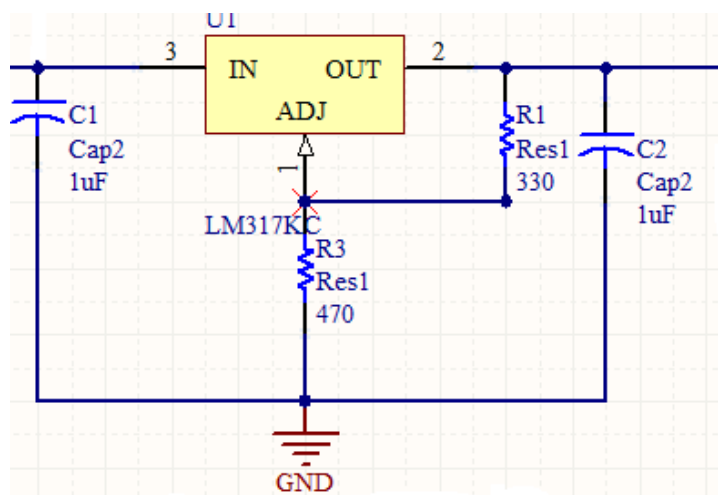


Figura 24 - Circuito regulador com LM317 para tensão de 3V
Fonte: Autoria própria (2014)

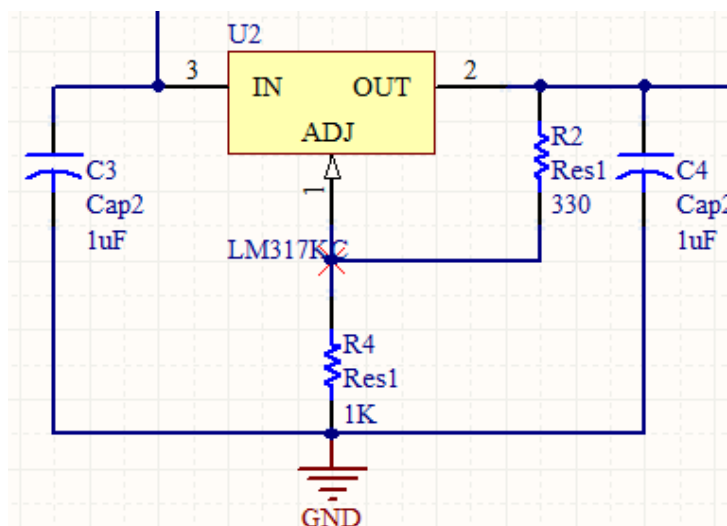


Figura 25 - Circuito regulador com LM317 para tensão de 5V
Fonte: Autoria própria (2014)

4.3 POSIÇÃO DO MÓDULO E DA SINALIZAÇÃO LUMINOSA NA VAGA

Para obter a máxima precisão no que tange a medição da distância pelo sensor entre a parede e o carro, foram medidas as distâncias em relação ao chão e em relação ao tamanho do carro para encontrar o posicionamento ideal do sensor, que corresponde ao ponto mais frente do carro. Através de medições nos carros disponíveis, pode-se observar que a distancia em largura era sempre a mesma (1,66 m aproximadamente entre os dois pneus dianteiros), com uma distância central, por consequência, de 83 cm e que a distância do chão em relação ao ponto mais a frente corresponde a aproximadamente 50 cm nos carros verificados.

Para posicionar a sinalização luminosa, avaliou-se qual a distância mínima que o motorista consegue enxergar em relação ao solo, através do vidro. Foi-se encontrada a distância de 80 cm, para que conseguisse enxergar sem ter que se mover a cabeça para frente. Visando tornar fácil a visualização, observou-se que acima de 1 metro pode-se enxergar a sinalização apenas olhando a frente, sem nenhum esforço. Assim, estipulamos uma distancia de 1,2 metros em relação ao chão para ser usada no projeto.

Na fotografia 5, é mostrado como foi posicionado o módulo em relação ao veículo, baseando-se nas medidas citadas acima:



Fotografia 5 - Posicionamento do módulo em relação ao carro
Fonte: Autoria própria (2014)

5 CENTRAL

Este capítulo tem por objetivo detalhar o papel da central na rede, que é responsável por configurar e receber os dados da rede e mostrar o estado de todas as vagas de um estacionamento real numa interface gráfica. O *hardware* da central é constituído por uma *Raspberry Pi – Model B* e um módulo de comunicação RF.

5.1 PAPEL NA REDE DESENVOLVIDA

A central tem como papel principal definir os parâmetros da rede, tais como o número de redes independentes, o de módulos que terá em cada uma delas e a distância limite para o qual o carro será considerado estacionado na vaga. Para configurar a rede, ela envia os pacotes de configuração para módulo mais próximo de cada uma das redes independentes, além de receber os estados das vagas repassados através dos módulos, que irá interpretar os dados e colocá-los na interface gráfica.

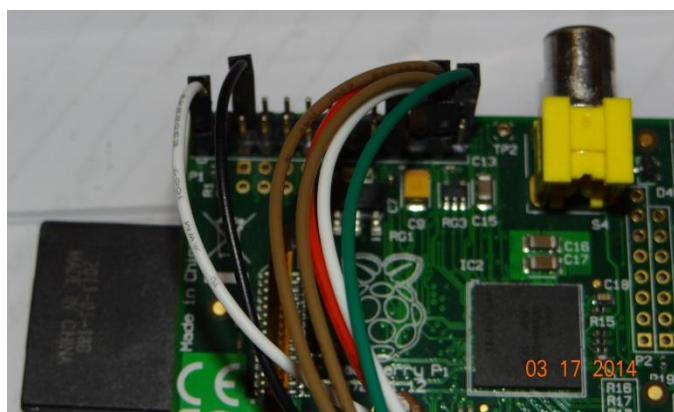
A central apresenta um módulo de comunicação nRF24L01+, mas com a adição de um circuito amplificador de potência (PA) e um circuito amplificador de baixo ruído (LNA), que permite transmitir distâncias mais longas e ter um desempenho mais estável. O aumento da PA e LNA, o interruptor de RF e o filtro passa-banda formam um excelente amplificador bidirecional de potência RF, tornando a comunicação muito mais eficaz (ELECTFREAKS, 2011). Abaixo, na figura 26, mostra o módulo de comunicação citado acima:



Figura 26 - nRF24I01+ com amplificador de potência (PA) e de baixo ruído (LNA)

Fonte: Robotshop (2013)

Este é conectado à *Raspberry Pi* através dos pinos de GPIO presentes na mesma, comunicando-se através da SPI, conforme a fotografia 6:



Fotografia 6 - Ligação do nRF24I01+ com os pinos de GPIO da *Raspberry Pi*

Fonte: Autoria própria (2014)

5.2 INTERFACE

A interface é a parte do programa na qual o usuário interage visualmente e permite aos usuários utilizar periféricos no funcionamento do projeto, é nela em que se pode verificar os estados dos módulos através ícones e figuras na tela do programa. Desenvolvido em linguagem C++ e com a biblioteca SDL (*Simple Directmedia Layer*), houve a possibilidade de programar um modelo simples de interface homem-máquina (IHM) e com uma fácil interatividade para as pessoas.

5.2.1 Elementos utilizados:

5.2.1.1 Classes

Elemento existente na linguagem C++, as classes são formas de *structs* que podem guardar uma quantidade de membros, sendo eles os atributos que armazenam valores e os métodos que fazem analogia às funções. Esses membros podem ser classificados como privados, públicos ou *protected*, isto é, ser acessível apenas por funções internas, enquanto a segunda em qualquer função interna e externa e a última, usada para casos especiais, mas o padrão é ser privado até que se diga o contrário.

5.2.1.2 Listas:

Lista é uma estrutura que permite armazenar um tipo de elemento em uma sequência, também chamado de lista duplamente encadeada, pois cada elemento interno tem uma referência do elemento anterior e o seguinte. Isto permite vasculhar a lista em ambos os sentidos, do início ao fim e vice-versa. Na figura 27 mostra como é ligado

Esta estrutura oferece facilidades em diversas operações que alteram elementos internos, como inserir, apagar, mover. Mas na hora de buscar dados específicos há uma necessidade maior por processamento, pois é criado um iterador que deve percorrer componente por componente até encontrar o dado desejado. O iterador deve ser apontar para uma das pontas da lista.

Alguns comandos de lista que foram utilizados:

- `Push_back`: adiciona elementos no final da lista.
- `Erase`: elimina um elemento da lista em qualquer posição.
- `Clear`: apaga a lista inteira.
- `Size`: informa o tamanho da lista.

5.2.1.3 Biblioteca de I/O

A biblioteca `<fstream>` permite tanto a entrada quanto a saída de dados em arquivos. Dessa forma podemos salvar os dados importantes na hora de encerrar o *programa* e recarregá-la ao reiniciarmos.

Para guardar dados em um arquivo externo, deve-se utilizar uma variável *ofstream* e depois abrir o arquivo desejado com o comando `open()`, caso não exista, ele será criado. É necessário realizar alguns testes antes de prosseguir para evitar erros, como chamar o comando `is_open()`, para certificar se o arquivo realmente está aberto, e a função `good()`, para verificar se não houve uma *flag* de erro. Não apresentando problemas, pode-se começar a guardar os dados, utilizando o operador `<<`, com o qual a informação de sua direita, as nossas variáveis, é armazenado num destino na esquerda do operador, que é o arquivo. Entre cada variável armazenada deve-se inserir caracteres especiais para fazer separação dos dados, aqui podemos usar o “_” (*underline*), “;” (ponto-e-vírgula) ou “,” (vírgula), fazendo isso facilitará na hora de abrir o arquivo. No fim devemos encerrar com o comando `close()` para evitar futuros erros.

Já na hora de carregar os dados do arquivo salvo, cria-se outra variável, essa do tipo *ifstream* e executar os mesmos comandos, `is_open()` e o `good()`. Então ao terminar, navega-se dentro do documento, linha por linha e com o comando `getline()`, guardando a atual linha numa *string*. Depois se varre essa *string* letra por letra, armazenando cada uma dentro de uma variável *char* e com o comando `strtok()`, encerra-se esse processo ao encontrar um carácter de separação, é necessária uma estrutura de repetição para continuar com esse procedimento até que a *string* seja nula e saltar para a linha seguinte. O próximo passo é usar o `sscanf()` para converter essa variável *char* num formato que seja útil ao programa. No fim também é necessário utilizar o comando `close()`.

5.2.1.4 Biblioteca SDL

SDL ou *Simple Directmedia Layer* é uma biblioteca criada para várias linguagens de programação e os diversos tipos de sistemas operacionais. Muitos programadores a utilizam para desenvolvimento de games devido ao fácil acesso aos elementos de áudio, periféricos em geral, elementos de vídeo, usos de figuras e programação 3D como o Open GL.

Para utilizá-la deve-se carregar os recursos existentes com o comando `SDL_Init()`, permitindo o uso de sub-rotinas como vídeo e eventos, mas ao encerrar o programa precisamos chamar o `SDL_Quit()`. Depois de inicializada, é preciso criar

a janela em que irá rodar o programa do projeto, através do comando `SDL_SetVideoMode()` em que se escolhe o comprimento e largura em *pixels*, a quantidade de bits em cada *pixels* e uma sinalização referente aos recursos utilizados pelo vídeo, como carregar ou não em modo *fullscreen*.

5.2.1.4.1 *SDL_Surface*

`SDL_Surface` é uma estrutura utilizada como variável para elementos que vão fazer uso de *pixels* e carregar imagens na tela. Possui diversos parâmetros como as informações dos *pixels* utilizados e outra estrutura, `SDL_Rect` que se refere a uma área retangular ocupada pela imagem com coordenadas de posição e comprimento e altura em *pixels*.

5.2.1.4.2 *SDL_Event*

Um diferencial da biblioteca SDL é o uso de diversos periféricos dentro do programa, como responder aos movimentos e as ações do mouse, as teclas do teclado, os botões de um *joystick*, captações do *touch finger* e até mesmo do *multi fingers*. Isso é possível através de leituras de eventos que acontecem durante a execução do programa, onde cada tipo dessas ações corresponde uma *struct* diferente.

`SDL_Event` é uma *union* que possui todos essas estruturas de eventos possíveis dentro da SDL. Isto significa que cada uma delas compartilha a mesma região de memória e quando uma dessas ocorre, o evento anterior que está na memória é substituído, assim sempre existirá apenas um tipo de *struct* a cada leitura de eventos.

Durante a execução do programa, ocorrem diversos eventos, então a própria biblioteca se encarrega de armazená-los dentro de uma lista, para verificar o tipo de ação deve-se usar o comando `SDL_PollEvent()` que pega o evento do topo da lista a passa por referência ao usuário. Para analisá-la é só usar os comandos condicionais, *if-else* e o *switch-case*, para verificar as *flags* que caracterizam cada evento.

Um exemplo, verificar o `SDL_MouseButtonEvent` que se refere aos botões do mouse. Para certificar qual botão da ação, se foi o da direita, `SDL_BUTTON_RIGHT`, ou da esquerda, `SDL_BUTTON_LEFT`; dentro disso ainda existe a tipo do evento, se o usuário clicou em `SDL_MOUSEBUTTONDOWN`, ou se soltou `SDL_MOUSEBUTTONUP` e por ultimo verificar o estado do botão do mouse, se está pressionado, `SDL_PRESSED`, ou solto, `SDL_RELEASED`.

5.2.1.4.3 Sub-biblioteca `SDL_TTF`

Apesar de inicializada, a SDL em si não possui todos os recursos necessários para o projeto, é preciso incluir outras sub-bibliotecas, sendo uma delas a `SDL_TTF` que processa fontes no formato TTF e armazena numa variável `SDL_Surface`, uma string para carregar na tela. Parecido com a própria biblioteca `<SDL.h>`, também é necessário inicializar suas funções com o `TTF_Init()` e no fim, encerrar com o `TTF_Quit()`.

Primeiramente, é preciso abrir uma fonte de um arquivo externo e escolher tamanho da letra com o comando `TTF_OpenFont()` e depois disso trocar o estilo das letras para negrito ou itálico com o `TTF_SetFontStyle()`, ver na Figura 27.



Figura 27 - Testando fonte e estilo via `SDL_TTF`
Fonte: Autoria própria (2014)

Para carregar uma frase na tela, existe os seguintes comandos, `TTF_RenderText_Solid()`, `TTF_RenderText_Shaded()` e o `TTF_RenderText_Blended()`, todos armazenam uma string, uma fonte que carregada e uma cor, a diferença entre eles aparece na forma apresentada. A primeira é mais sólida e rústica, usando o menor processamento entre os três; já a segunda possui uma cor extra no realce de fundo, sendo necessário informar mais uma cor na referência e enquanto a última melhora a aparência usando um maior processamento para lapidar as extremidades de cada letra, ver Figura 28.

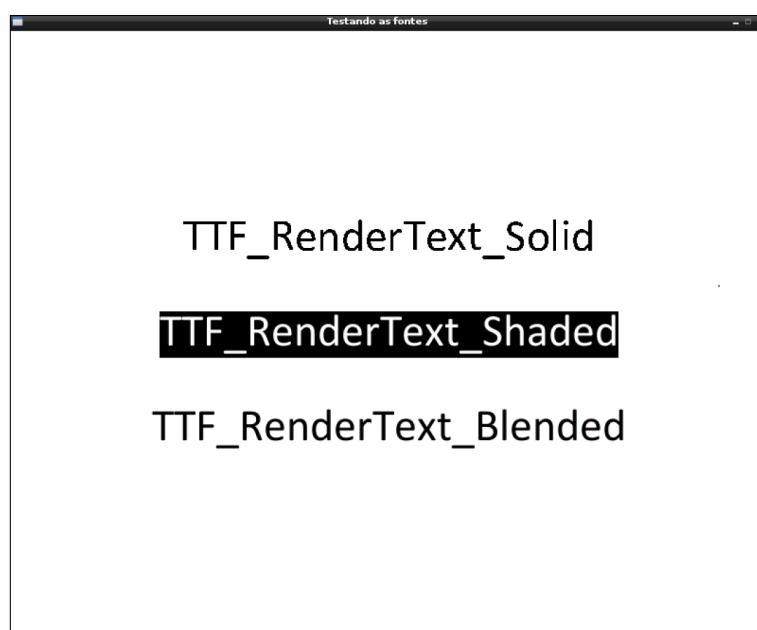


Figura 28 - Teste da fonte conforme os diversos comandos
Fonte: Autoria própria (2014)

5.2.1.4.4 Sub-biblioteca *SDL_IMG*

Para trabalhar com as imagens oriundas de arquivos externos, é preciso incluir mais uma biblioteca, a `<SDL_Image.h>` que permite além de carregar as figuras nos mais diversos formatos, como também processá-las de uma forma que auxilie na visualização.

Trabalhando mais uma vez com a `SDL_Surface`, para carregar uma gravura com o `IMG_Load()`, em que é preciso indicar o caminho do diretório onde se encontra o arquivo da imagem. Mas para que apareça na tela principal, existem outras funções, sendo a primeira a `SDL_BlitSurface()`, que insere uma figura sobre

uma superfície de referência que no nosso caso é tela do programa, é necessário fazer isso para cada variável `SDL_Surface` que queremos carregar, isso fica armazenado dentro de um buffer e então aplicar o comando `SDL_Flip()` que troca o frame atual pelo armazenado.

Outra necessidade, é que as gravuras não são necessariamente retangulares, podem ter as mais diversas formas, mas ao salvar num arquivo ela se torna retangular e as áreas ao redor da figura devem ser preenchidas com uma única cor, ver Figura 29. Pois com a `SDL_Image` pode-se fazer com que essa cor fique invisível e ela não apareça na tela e assim teremos a gravura sem o contorno, ver Figura 30. Os comandos necessários são o `SDL_MapRGB()` que traça os locais onde há a cor escolhida e depois com a `SDL_SetColorKey()` essas regiões serão alteradas na variável `SDL_Surface`.

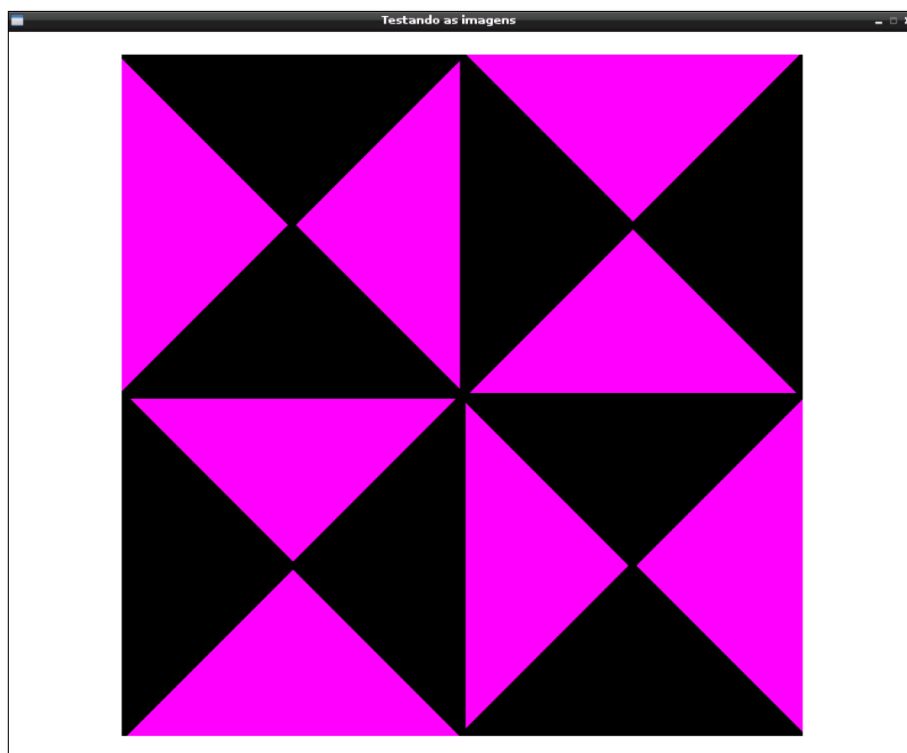


Figura 29 - Imagem sem `SDL_MapRGB()` e `SDL_SetColorKey()`
Fonte: Autoria própria (2014)

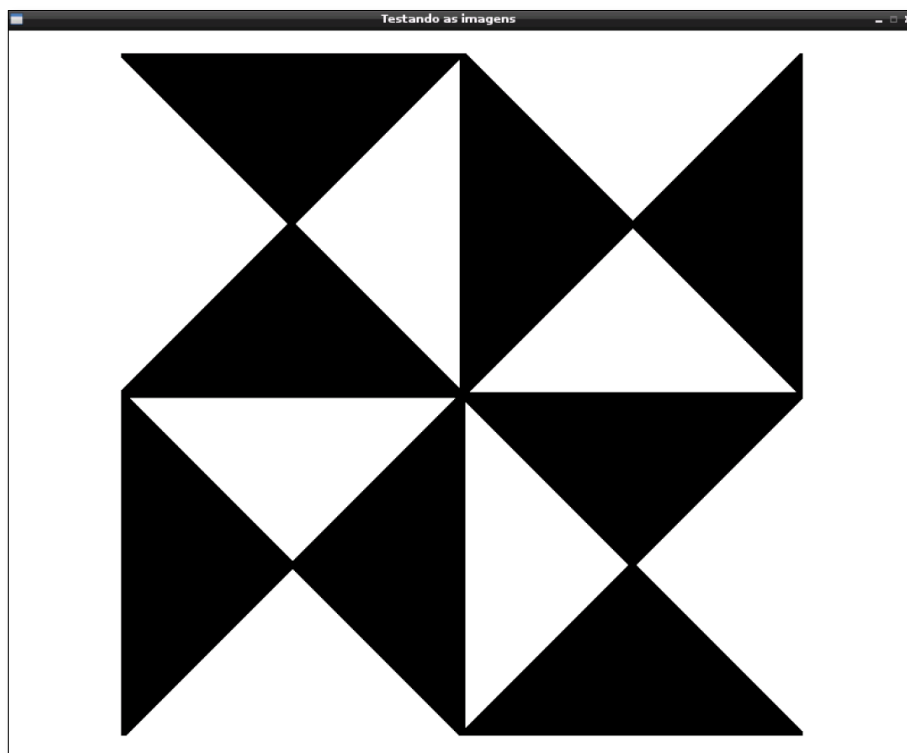


Figura 30 - Imagem utilizando os meios para deixar uma cor invisível
Fonte: Autoria própria (2014)

5.2.2 Funções

Foi criada uma classe `Vaga()` que se refere aos ícones colocados na imagem do estacionamento. Ela armazena os valores da posição dos ícones, o estado em que se encontram os sensores em campo, e as variáveis de identidade, a rede que pertence e um ID. Com isso criamos uma lista desse tipo de elemento para ser utilizado por diversas funções da interface.

As principais funções são aquelas que fazem leituras de eventos e que criam as diferentes telas do programa. São elas, a `tela_inicial()`, com *menu* de opções; `tela_conf()`, para decidir alguns parâmetros fundamentais para o funcionamento do programa, `tela_add()` para colocar ícones de vagas na figura do estacionamento; `tela_del()`, que serve para apagar um desses ícones; `tela_escolheVaga()`, que permite ao usuário escolher um ícone para que a `tela_id()` possa trocar os dados de identidade e por último a `tela_leitura()`, função que verifica os estados oriundos dos módulos e informa ao motorista através de imagens.

Ainda existem outras funções auxiliares dentro da interface, são elas a `init()` que carrega os recursos das bibliotecas da SDL e as variáveis, a `scene()`, que

imprime as imagens na tela principal conforme as funções citadas no paragrafo anterior, o `saveFile()`, para guardar os dados dos elementos da lista; `loadFile()`, para retomar esses dados e recriar a lista e a `ajusta_del()` para apagar elementos excedentes após a configuração básica pela `tela_conf()`.

5.2.2.1 Tela_inicial()

Esta função mostra uma figura do estacionamento, os ícones das vagas, caso haja alguma alocada, e os botões para opções das funcionalidades do programa. O botão CONF serve para chamar a `tela_conf()`, DEL a `tela_del()`, ID para `tela_escolheVaga()` e depois `tela_id()` e o START para `tela_leitura()`. Cada um desses botões ocupam uma região na tela principal. Função retratada na Figura 31.

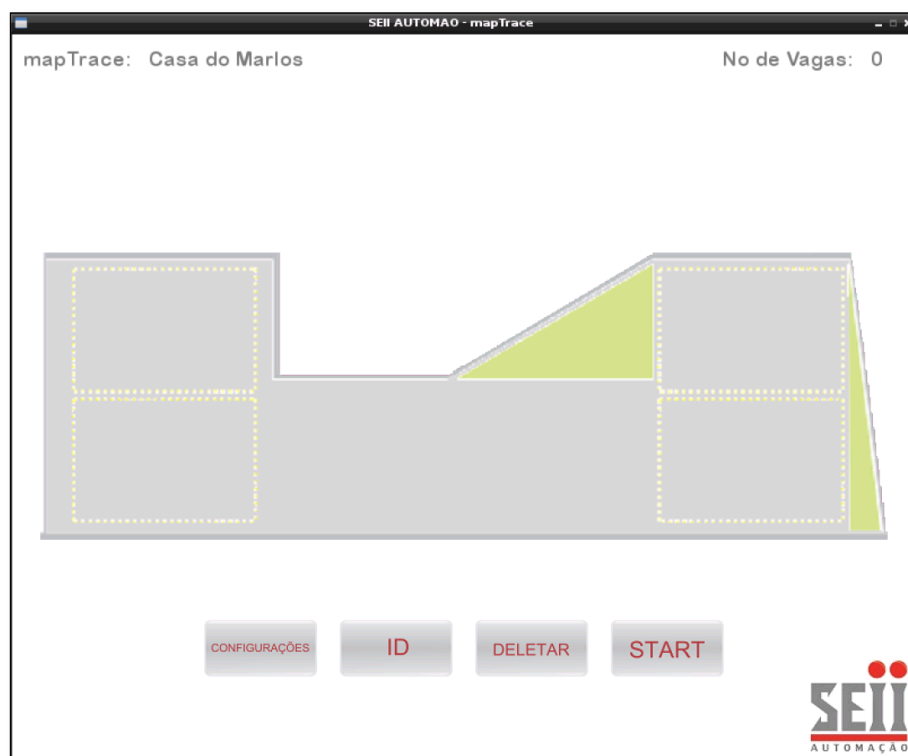


Figura 31 - Função tela_inicial()
Fonte: Autoria própria (2014)

Aqui, os eventos referem-se aos botões do mouse, a `SDL_MouseButtonEvent`. Quando o botão da esquerda do mouse é pressionado, através das coordenadas e do comando *if-else*, podemos verificar se o usuário

apertou uma região que pertence a alguma área das opções do menu. Na Figura 32, é mostrado um exemplo de como funciona a leitura do mouse:

```
// comando que pega o primeiro elemento da lista de eventos
if ( SDL_PollEvent( &event ) ) {
    // verifica se o evento foi do tipo clique no botão do mouse
    if ( event.type == SDL_MOUSEBUTTONDOWN ) {
        // verifica se o botão foi o da esquerda
        if( event.button.button == SDL_BUTTON_LEFT ) {

            // grava as coordenadas
            x = event.motion.x;
            y = event.motion.y;

            // Verifica que as coordenadas se esta dentro de uma região
            // O exemplo abaixo é o botão de deletar.
            // pos_deletar é um vetor que possui as coordenadas do botão
            // img_deletar->w é o comprimento em pixels da imagem
            // img_deletar->h é a altura em pixels

            if((x>pos_deletar[0])&&(x<pos_deletar[0]+img_deletar->w) &&
                (y>pos_deletar[1])&&(y<pos_deletar[1]+img_deletar->h ) ) {
                (... comandos adicionais)
            }
        }
    }
}
```

Figura 32 - Exemplo de como funciona a leitura do mouse

Fonte: Autoria própria (2014)

5.2.2.2 Tela_conf()

Nesta parte, o usuário informa a quantidade de redes com que irá trabalhar e o número de módulos em cada uma delas. Depois de decidido, é liberado o acesso para condicionar o número de ícones conforme os dados de entrada. Caso haja um aumento de vagas dentro de uma rede, será chamada a tela_add() para colocar os ícones extras na imagem, caso contrário é chamado à função ajusta_del() para que eliminar a quantidade extra de elementos na lista. Ainda existe o botão SCAN nessa tela para que possamos fazer a configuração dos módulos em campo. Essa função é retratada na Figura 34.

Na análise de eventos, usamos a SDL_KeyboardEvent, que analisa as teclas apertadas e procura quando for um numero, esse valor é alocado ao final de uma *string*, assim mostrar ao usuário numero que está digitando. Também usamos a tecla BACKSPACE, onde diminuimos o tamanho da *string* em um, assim apagando o ultimo dado de entrada. Por ultimo à tecla RETURN/ENTER, a função converte a

string num inteiro para guardar dentro de uma variável que será utilizado em outras funções, procedimento retratado na Figura 33. Esse processo irá continua enquanto houver necessidade de inserir parâmetros.

Essa função ainda precisa verificar os eventos de mouse se foi apertado algum dos botões ou para condicionar as vagas conforme os dados inseridos, ou chamar a função de *scan* ou sair.

```
// Pega o primeiro evento da lista
if ( SDL_PollEvent( &event ) ) {
    // verifica se o evento é referente ao teclado
    if ( event.type == SDL_KEYDOWN ) {
        // verifica qual foi o botão através de IF.
        // Botão ESC apertado
        if ( event.key.keysym.sym == SDLK_ESCAPE ) {quit = true;}
        // Botão referente aos números
        if(( event.key.keysym.unicode >= (Uint16)'0' ) &&
            ( event.key.keysym.unicode <= (Uint16)'9' ) ) {
            if ( num_redes_str == "0" ) {
                num_redes_str.erase( num_redes_str.length() - 1 );
            }
            // pega o valor do evento e adicionar no final da string
            num_redes_str += (char)event.key.keysym.unicode;
        }
        // Botão foi o BACKSPACE
        if ( ( event.key.keysym.sym == SDLK_BACKSPACE ) &&
            ( num_redes_str.length() != 0 ) ) {
            // diminui o tamanho da string em 1.
            // assim apaga o ultimo elemento inserido
            num_redes_str.erase( num_redes_str.length() - 1 );
        }
        // Botão ENTER/RETURN
        if((event.key.keysym.sym==SDLK_RETURN) &&
            (num_redes_str!="0" && num_redes_str!= "")) {
            // converte a string em inteiro
            num_redes = atoi(num_redes_str.c_str());
            (... outros comandos)
        }
    }
}
```

Figura 33 - Exemplo de como funciona a leitura do teclado
Fonte: Autoria própria (2014)



Figura 34 - Função tela_conf()
Fonte: Autoria própria (2014)

5.2.2.3 Tela_add()

Chamada pela tela_conf(), esta função (figura 35) é responsável por incluir objetos da classe vaga dentro de uma lista e assim por adicionar ícones na tela. Ela recebe como parâmetro a rede dos novos objetos e a quantidade de novas vagas que deverão ser incluídas. A função ainda vasculha a lista em busca por outros elementos dessa rede para continuar com os valores ID para as novas vagas.

Utilizando mais uma vez os botões do mouse, é pego as coordenadas para verificar a existência de algum conflito com outras vagas ou a margem do estacionamento. Não tendo problemas, o programa irá criar um objeto da classe vaga com os valores da posição, estado nulo para a vaga, a rede que pertencerá e o ID, assim incluindo no final da lista.

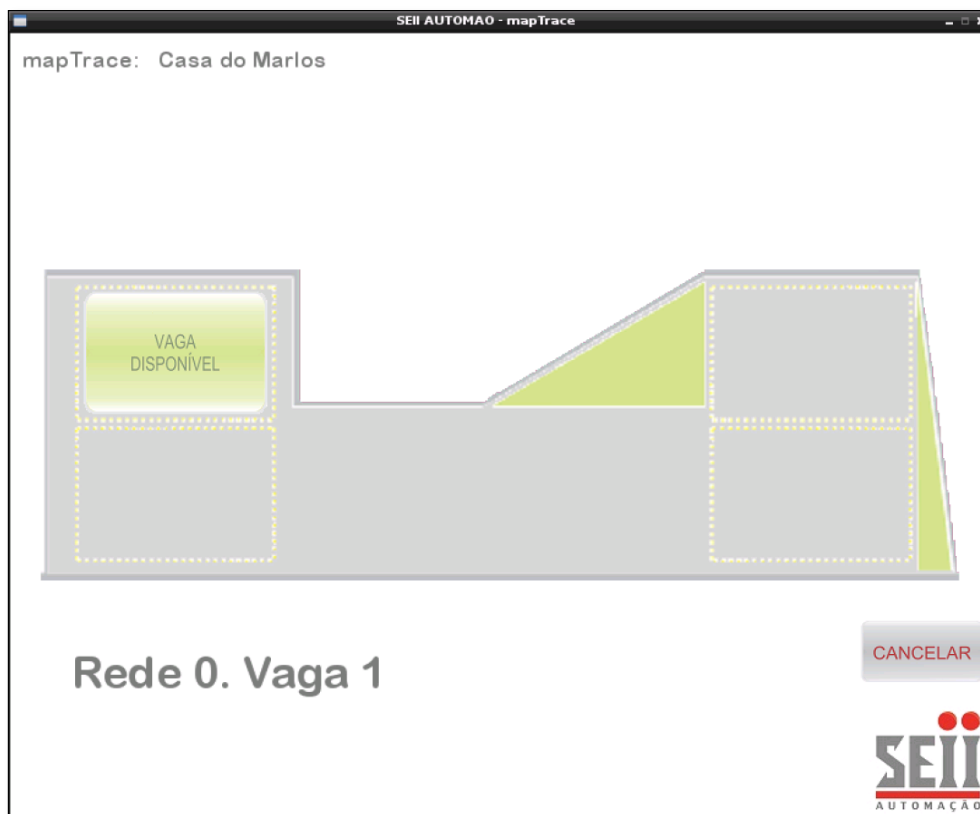


Figura 35 - Função tela_add()
Fonte: Autoria própria (2014)

5.2.2.4 Tela_del()

Caso seja preciso retirar algum objeto é só acessar a opção DEL na tela_inicial() e depois clicar no ícone desejado que o elemento será apagado da lista.

Também com a SDL_MouseButtonEvent, verifica-se as coordenadas do clique do mouse e para vasculhando a lista de vagas e verificando se a região dos ícones bate com as coordenadas do mouse, quando é encontrado será utilizado o comando erase() para apagá-lo.

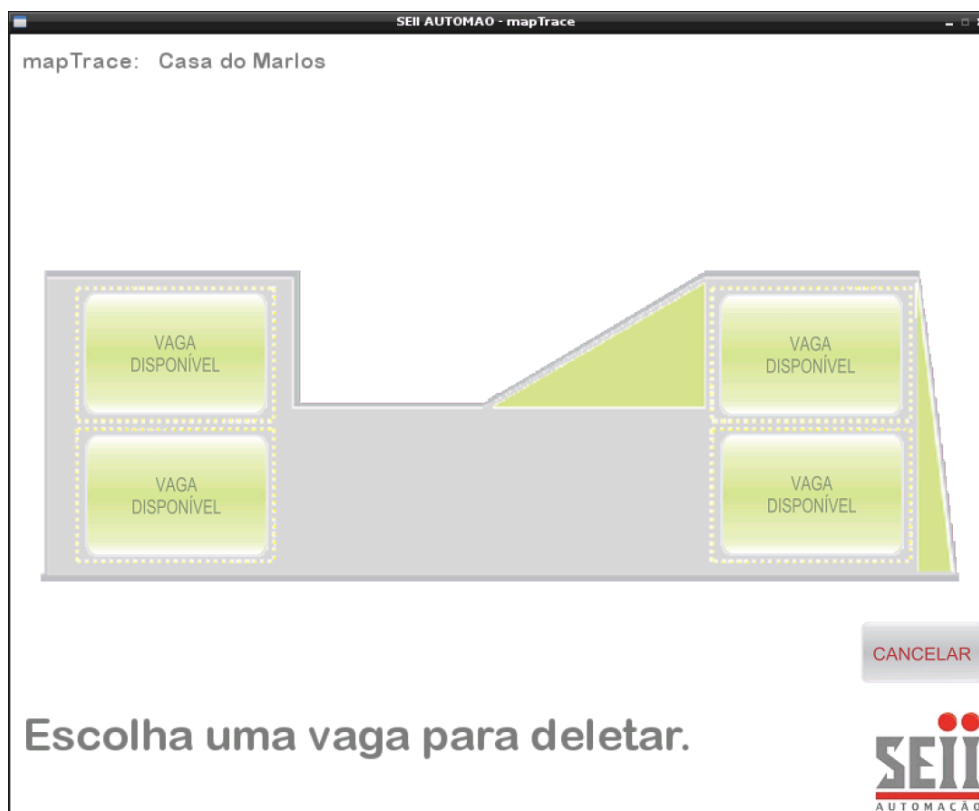


Figura 36 - Função tela_del()
Fonte: Autoria Própria (2014)

5.2.2.5 Tela_escolhaVaga() e Tela_id()

Para os casos em que é preciso alterar dados de uma vaga, clica-se num ícone da lista e assim podemos alterar a rede e o ID através do teclado.

A função `tela_escolhaVaga()` (Figura 37), realiza a leitura dos eventos de mouse que é parecido com a `tela_del()`, em que se vasculha a lista de vagas, mas ao invés de apagar, é chamado a função `tela_id()` (Figura 38) que analisa os eventos do teclado, parecido com a 5.2.2.2 `Tela_conf()`, para verificar os dados de entrada e armazenar no objeto alvo.

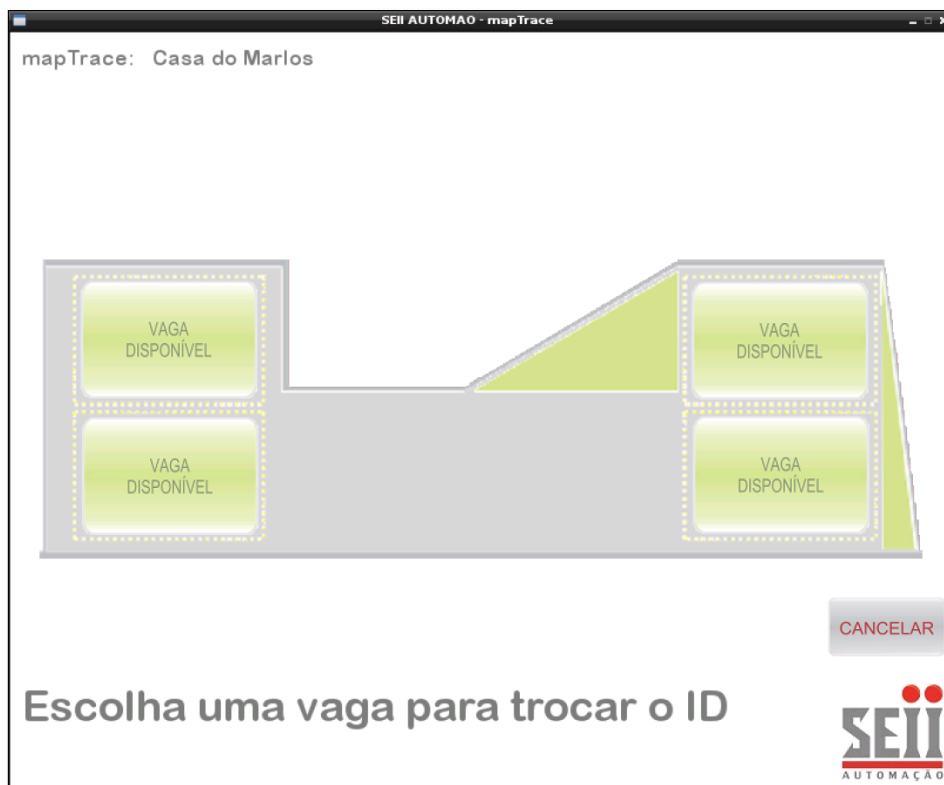


Figura 37 - Função escolhaVaga()
Fonte: Autoria própria (2014)

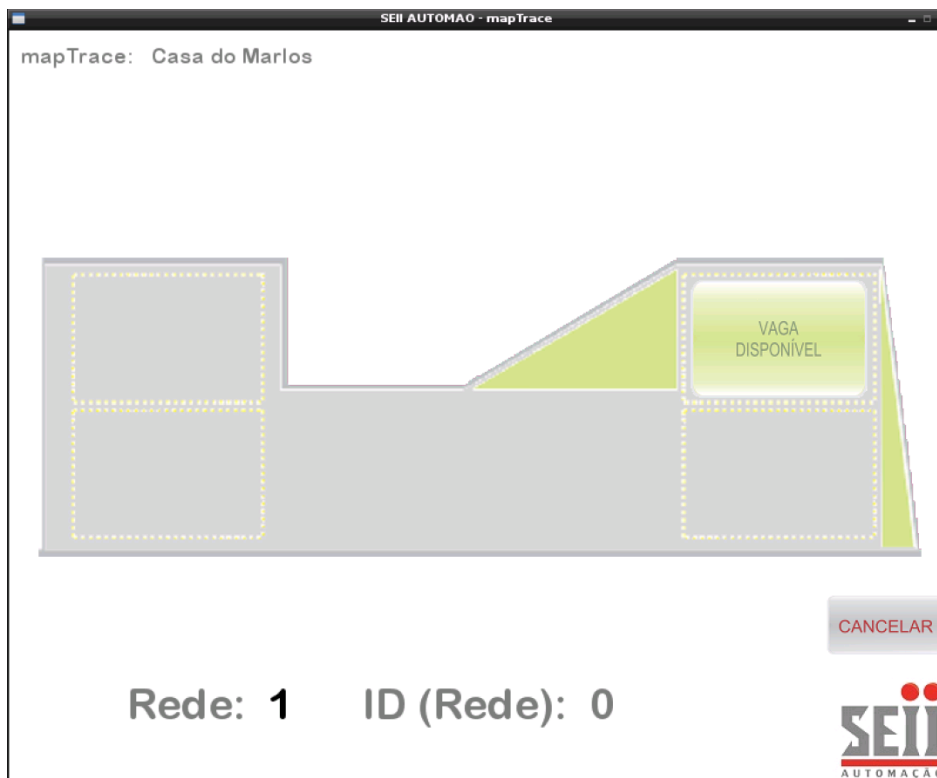


Figura 38 - Função tela_id()
Fonte: Autoria própria (2014)

5.2.2.6 Tela_leitura()

Última função referente aos tipos de tela, servindo para o sincronismo dos módulos com os ícones na imagem, podendo alternar entre VAGA DISPONIVEL, OCUPADO, CARRO SAINDO e CARRO ENTRANDO, conforme mostra a figura 39.

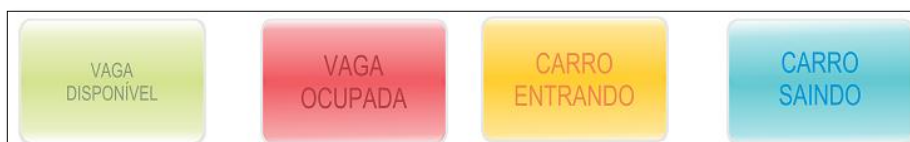


Figura 39 - Diferentes estados para os ícones de vagas
Fonte: Autoria própria (2014)

Através de uma matriz com o tamanho do número total de redes e o maior valor de ID entre as redes. Quando se verifica um elemento da lista é acessada a matriz usando as variáveis de identificação, assim se pega o valor ali inserido e carrega-se no atributo referente ao estado da classe.

Esta função não utiliza leitura de eventos muito complexas apenas verifica uma tecla do teclado, a BACKSPACE e uma região para o mouse, unicamente para sair dessa função, ocmo pode ser visto na figura 40.



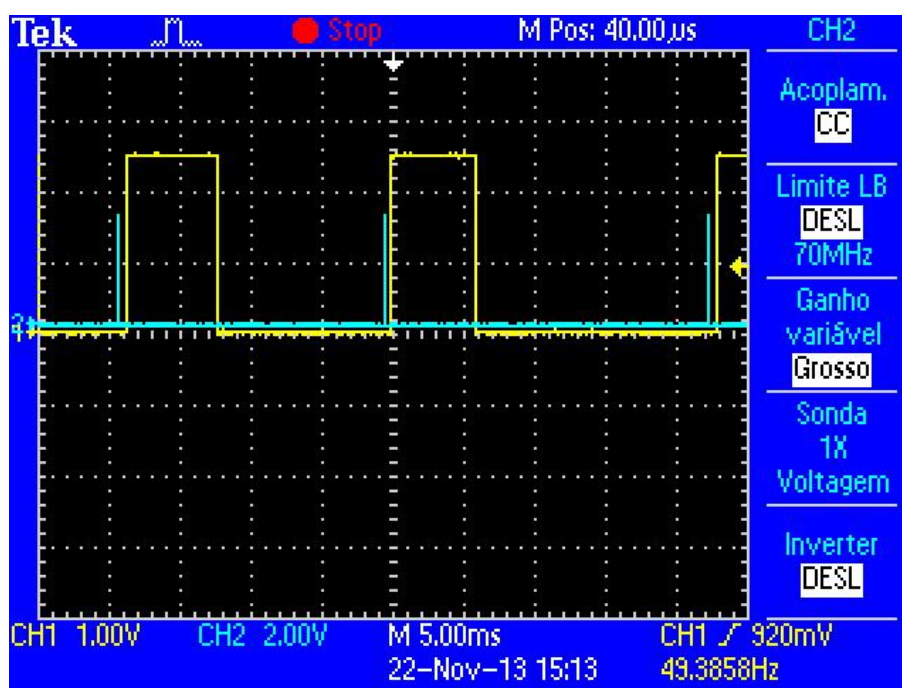
Figura 40 - Função tela_leitura
Fonte: Autoria própria (2014)

6 TESTES E RESULTADOS OBTIDOS

Neste capítulo será comentado a respeito dos testes feitos com alguns dos principais componentes do módulo, como o sensor ultrassônico e o modulo de comunicação sem fio, além de mostrar o Sistema de Gerência de Vagas de Estacionamento funcionando num estacionamento real.

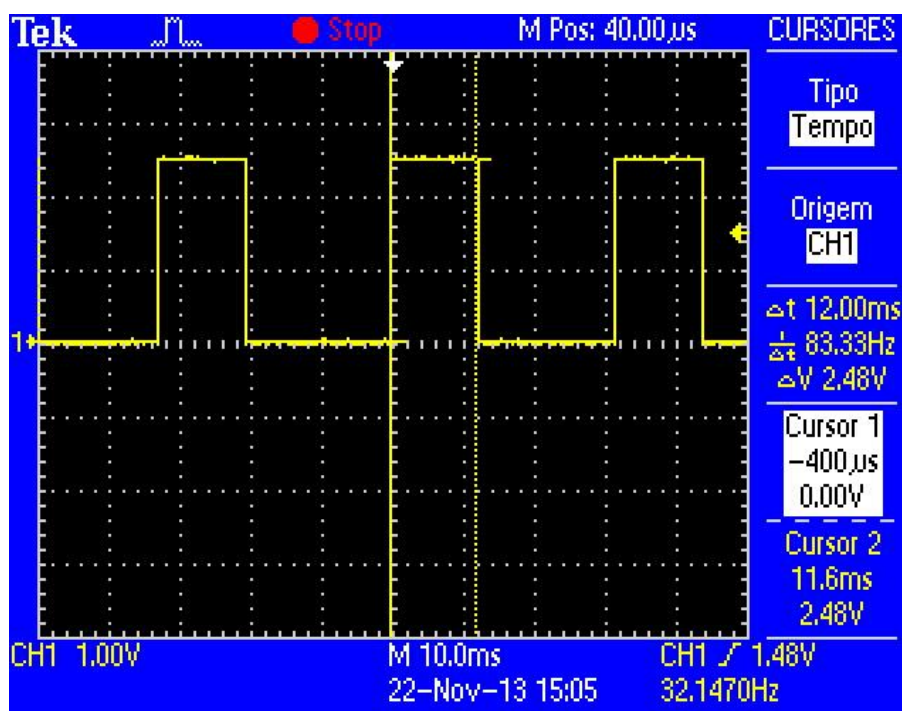
6.1 TESTE COM O SENSOR ULTRASSÔNICO

Com o intuito de testar o sensor, foram lidas as formas de onda dos pinos de *trigger* e *echo* do mesmo, com o auxílio de um osciloscópio digital, como mostra a figura 41:



**Figura 41 - Formas de onda do pino de trigger (azul) e de echo (amarelo).
Fonte: Autoria própria (2014)**

Para verificar se a medição da distância em função da largura de pulso do pino de *echo* funciona, foi-se colocado um objeto a uma distância conhecida e lido o pulso gerado que é recebido pelo microcontrolador. Na figura 42, um exemplo de um pulso recebido pelo mesmo por uma distância verificada de 210 cm:



**Figura 42 - Forma de onda do pino de echo, para uma distância de 210 cm.
Fonte: Autoria própria (2014)**

Como se pode ver o pulso do pino de *echo* tem uma largura de 12 milissegundos, que, convertendo para distância, através da relação da física que esta é igual à velocidade vezes o tempo, sendo que a primeira é a do som (340 m/s) e o segundo, é a metade do tempo contado do pulso recebido (tempo de ida do sinal de ultrassom), em microssegundos, resulta num valor teórico de aproximadamente 204 cm, porém como no programa utilizam-se valores inteiros, acaba-se obtendo um valor de 207 cm, mostrando que a leitura do sensor pelo microcontrolador está funcionando, apesar de um pequeno erro introduzido.

6.2 TESTE COM O MÓDULO DE COMUNICAÇÃO RF

Para verificar o recebimento e envio de dados, faz-se necessário verificar, através do osciloscópio, as formas de onda dos pinos MISO e MOSI, da SPI, que fazem a comunicação entre o módulo e o microcontrolador.

O pacote enviado usado como exemplo é o do primeiro módulo (endereço 3) enviando a configuração para o segundo módulo presente na rede independente. Este tem a seguinte composição, conforme mostra a figura 43:

Expression	Value	Location	Type
data_out	'.' (0x03)	Memory: 0x21D	<8-bit unsign...
[0]	3	Memory: 0x21D	unsigned char
[1]	4	Memory: 0x21E	unsigned char
[2]	1	Memory: 0x21F	unsigned char
[3]	0	Memory: 0x220	unsigned char
[4]	3	Memory: 0x221	unsigned char
[5]	1	Memory: 0x222	unsigned char
[6]	40	Memory: 0x223	unsigned char
[7]	0	Memory: 0x224	unsigned char

Figura 43 - Composição do pacote de configuração a ser enviado pela SPI.
Fonte: Autoria própria (2014)

O envio do pacote de configuração pode ser verificado através do pino MOSI, situação verificada pela figura 44:

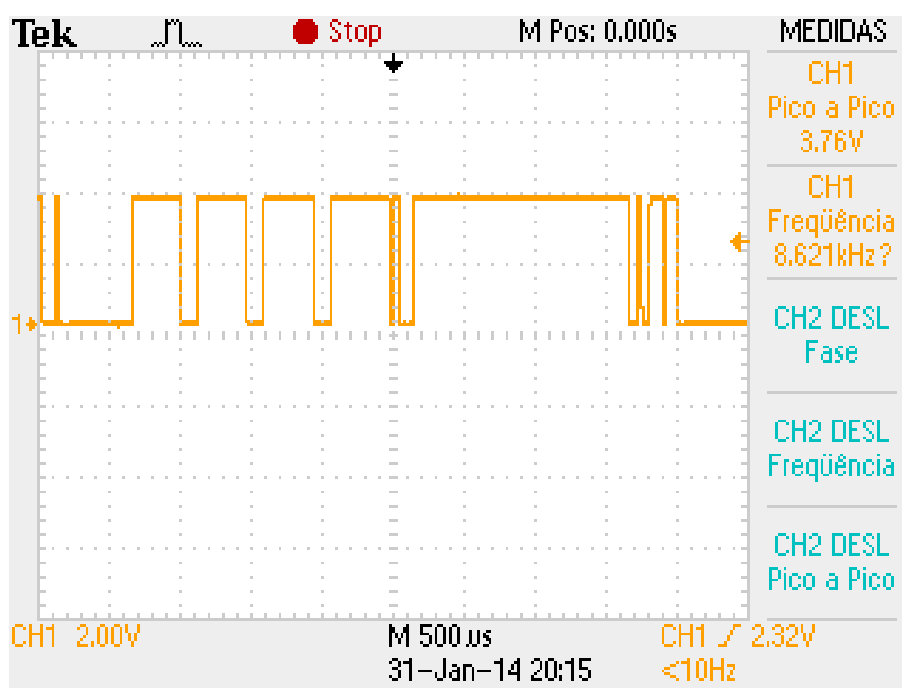


Figura 44 - Forma de onda no pino MOSI, ao enviar pacote de configuração.
Fonte: Autoria própria (2014)

O pacote de resposta usado como exemplo é o que vai do segundo módulo da rede independente (endereço 4) para o primeiro módulo presente na mesma, quando este enviou o pacote de configuração. Este tem a seguinte composição, como mostra a figura 45:

Expression	Value	Location	Type
data_in	"J L "	Memory: 0x213	unsigned ch...
[0]	4	Memory: 0x213	unsigned char
[1]	3	Memory: 0x214	unsigned char
[2]	2	Memory: 0x215	unsigned char
[3]	0	Memory: 0x216	unsigned char
[4]	0	Memory: 0x217	unsigned char
[5]	0	Memory: 0x218	unsigned char
[6]	0	Memory: 0x219	unsigned char
[7]	0	Memory: 0x21A	unsigned char

Figura 45 - Composição do pacote de resposta recebido pela SPI.
 Fonte: Autoria própria (2014)

O recebimento do pacote de resposta pode ser verificado através do pino MISO, situação presente pela figura 46:

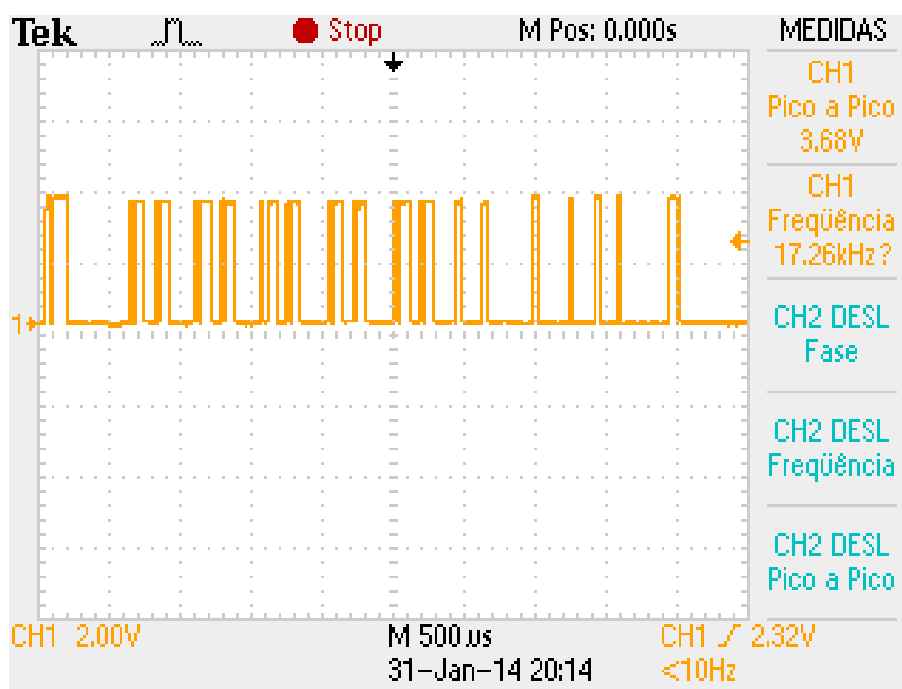


Figura 46 - Forma de onda no pino MISO ao receber pacote de resposta
 Fonte: Autoria própria (2014)

6.3 TESTE DO SISTEMA EM UM ESTACIONAMENTO COM 4 VAGAS

6.3.1 Montagens para o teste

O teste do sistema foi feito na residência de um dos membros da equipe, que apresenta 4 vagas disponíveis. Em cada uma delas, foi colocado um módulo que verifica o estado da vaga, sinaliza para o motorista se o carro está estacionado corretamente e envia os dados pela rede sem fio. Como se encontram em dois agrupamentos de duas vagas foram divididas, no contexto da rede, em duas redes independentes, cada uma com conexão direta com a central.

Esta foi colocada em um ponto central entre as duas para facilitar o enlace e evitar problemas de conexão.

O módulo foi instalado na vaga da seguinte forma na vaga, conforme mostra a fotografia 7:



**Fotografia 7 - Módulo instalado para o teste.
Fonte: Autoria própria (2014)**

As áreas utilizadas para o teste estão representadas nas imagens Fotografia 8 e na Fotografia 9.



Fotografia 8 - Área utilizada para teste com 2 vagas (rede independente 0).
Fonte: Autoria própria (2014)



Fotografia 9 - Área utilizada para teste com 2 vagas (rede independente 1).
Fonte: Autoria própria (2014)

A central utilizada no teste foi a *Raspberry Pi* utilizado junto com um *laptop* para, através deste, visualizar e interagir com a central, como é mostrado na fotografia 10:



Fotografia 10 - Foto da interface
Fonte: A autoria própria (2014)

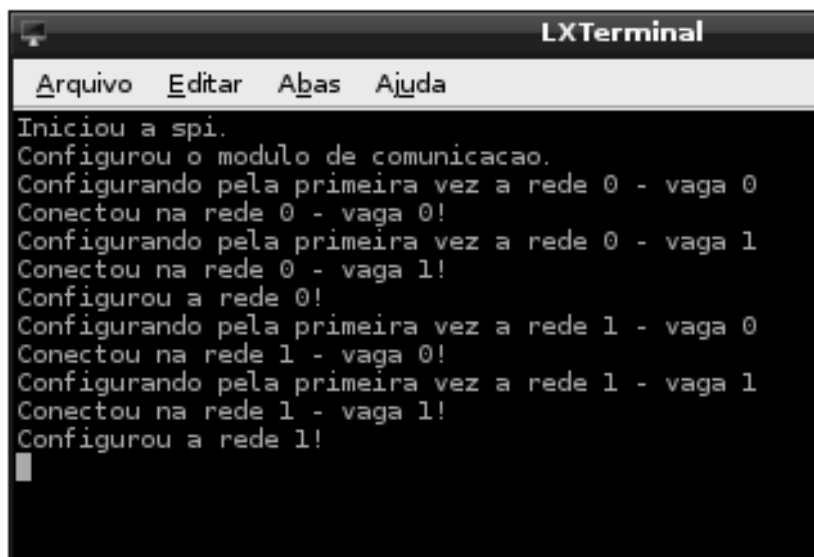
6.3.2 Funcionamento do Sistema de Gerência das Vagas de Estacionamento na residência

Após abrir o programa, é configurada uma situação de teste em que o sistema possui duas redes independentes e dois módulos em cada uma delas. Depois de posicionado os ícones das vagas, conforme o título 5.2.2.3 Tela_add(), e na tela de configuração, 6.2.2.2 Tela_conf(), é preciso configurar a rede, clicando no botão “SCAN”, e assim os dados do terminal é representado pela Figura 47:

```
LXTerminal
Arquivo  Editar  Abas  Ajuda
Iniciou a spi.
Configurou o modulo de comunicacao.
Configurando pela primeira vez a rede 0 - vaga 0
█
```

Figura 47 - Terminal ao ser iniciada a configuração da rede.
Fonte: A autoria própria (2014)

Pelo terminal é que será mostrado se os módulos se conectaram com a central. Uma vez a rede configurada, mostrará a seguinte tela, apresentada na Figura 48:



```
LXTerminal
Arquivo Editar Abas Ajuda
Iniciou a spi.
Configurou o modulo de comunicacao.
Configurando pela primeira vez a rede 0 - vaga 0
Conectou na rede 0 - vaga 0!
Configurando pela primeira vez a rede 0 - vaga 1
Conectou na rede 0 - vaga 1!
Configurou a rede 0!
Configurando pela primeira vez a rede 1 - vaga 0
Conectou na rede 1 - vaga 0!
Configurando pela primeira vez a rede 1 - vaga 1
Conectou na rede 1 - vaga 1!
Configurou a rede 1!
```

Figura 48 - Terminal com a rede já configurada.
Fonte: Aatoria própria (2014)

Voltando para tela de configuração, basta clicar em “SAIR” e na tela inicial, clicar em “START” e o sistema começará a funcionar, como mostra a Figura 49.



Figura 49 - Sistema funcionando com todas as vagas disponíveis.
Fonte: Aatoria própria (2014)

Como exemplo para mostrar o sistema funcionando, utilizou-se um carro para variar o estado da vaga 0, da rede independente 0, posicionada na parte de cima e na esquerda na Figura 49. Com o carro entrando na mesma, será apresentada a Figura 50 na interface gráfica.

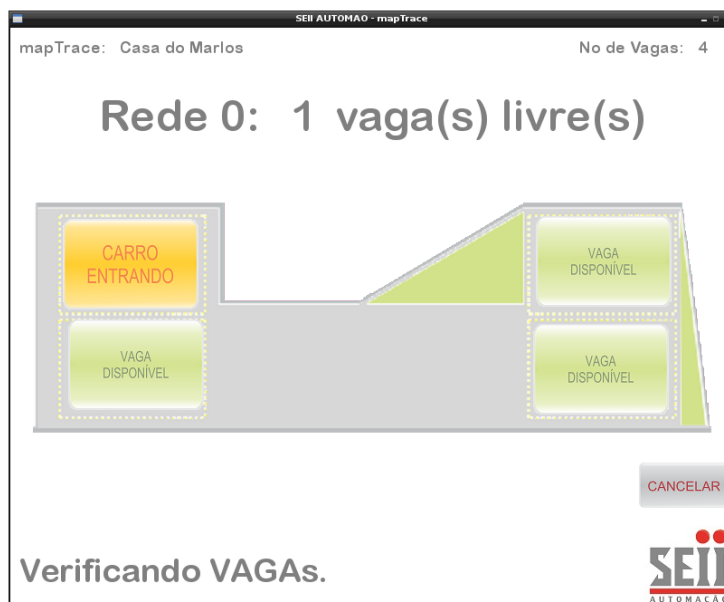


Figura 50 - Representação com o carro entrando.
Fonte: Autoria própria (2014)

Na Figura 51, o carro está estacionado na vaga.

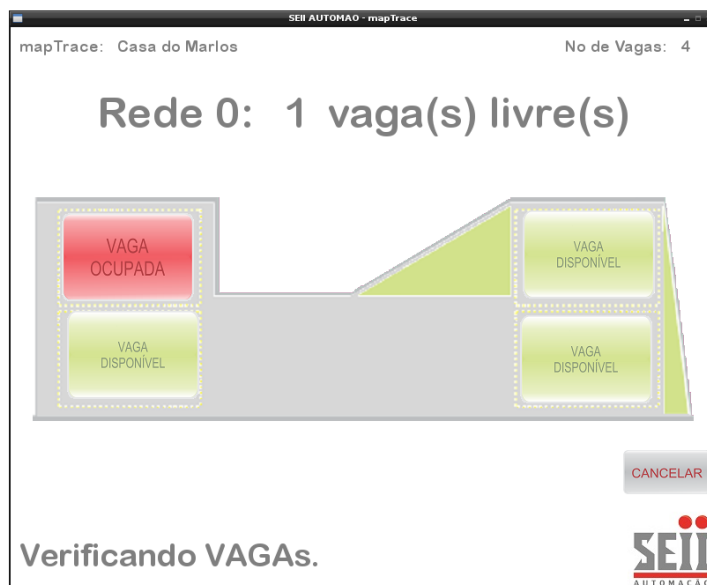


Figura 51 - Representação com o carro estacionado na vaga.
Fonte: Autoria própria (2014)

Na Figura 52, o carro está saindo da vaga.

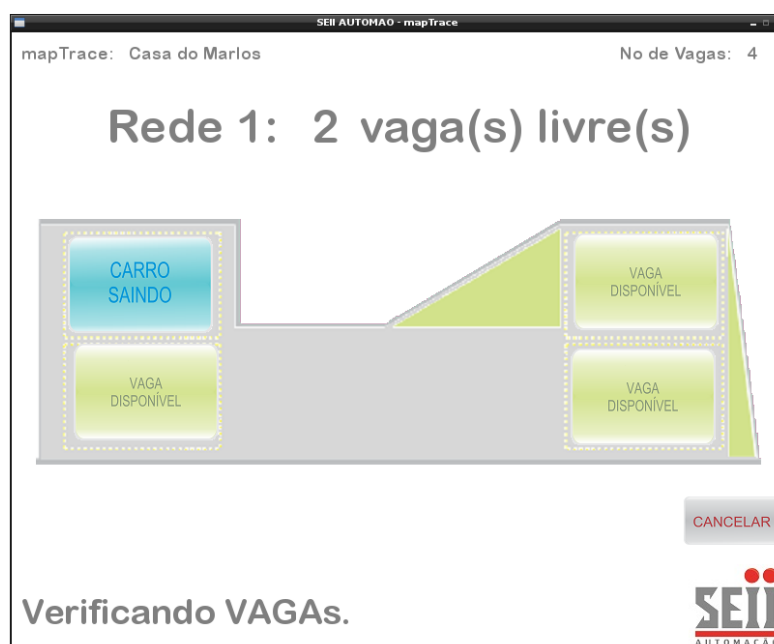


Figura 52 - Representação com o carro saindo na vaga.
Fonte: Autoria própria (2014)

Caso ocorra alguma falha algum dos módulos de alguma das redes independentes, será mostrada no terminal, pois foi considerado que este tipo de informação não é interessante aparecer na interface, apenas para o configurador saber que ocorreu uma perda de comunicação, conforme mostra a figura 53 (no caso, um dos módulos da rede independente 0 teve algum problema que o fez perder a conexão):

```
LXTerminal
Arquivo  Editar  Abas  Ajuda
Iniciou a spi.
Configurou o modulo de comunicacao.
Configurando pela primeira vez a rede 0 - vaga 0
Conectou na rede 0 - vaga 0!
Configurando pela primeira vez a rede 0 - vaga 1
Conectou na rede 0 - vaga 1!
Configurou a rede 0!
Configurando pela primeira vez a rede 1 - vaga 0
Conectou na rede 1 - vaga 0!
Configurando pela primeira vez a rede 1 - vaga 1
Conectou na rede 1 - vaga 1!
Configurou a rede 1!
Perdeu a conexao com a Rede 0.
```

Figura 53 – Perda de conexão com a rede independente 0.
Fonte: Autoria própria (2014)

Sendo assim, deve-se pausar o funcionamento do sistema recolocar o módulo em funcionamento e executar novamente a configuração da rede, não tendo a necessidade de fazer o reposicionamento, apenas a parte de configuração da rede independente que perdeu a conexão.

7 PLANO DE NEGÓCIOS

7.1 SUMÁRIO EXECUTIVO

A SEII Automação é uma empresa que atua na área de automação em estacionamentos para shoppings e supermercados, oferecendo produtos que trazem benefícios não apenas aos clientes, mas como aqueles que venham fazer uso deles. Situado em Curitiba, cidade uma das cidades importantes da região sul do Brasil em no mercado de shoppings.

Os sócios são engenheiros formados em engenharia eletrônica pela UTFPR, os quais desenvolveram ao longo do curso diversas matérias e projetos ligados à eletrônica e também compartilham experiência na área de comunicações e também área de pesquisa em controle. Essas diferentes experiências auxiliam no desenvolvimento novos métodos através de diversas áreas da eletrônica.

A nossa oportunidade surge em oferecer produtos e serviços para estacionamentos sob uma regionalidade, já que as grandes empresas situam-se em São Paulo e região Nordeste, deixando assim uma lacuna no mercado. Através dos sites das associações de shoppings e supermercados o número atual é de 86 para shoppings e 120 para supermercados com potencial de compra. Além disso, até o final do ano, cinco novos shoppings deverão ser inaugurados na região sul do brasil.

Este plano visa adquirir investimento total de R\$ = 150.00,00. A meta é baseada em alcançar o *payback* em no máximo quatro anos e atingir a região sul adquirir um lucro líquido em torno de R\$ = 55.000,00 por ano.

7.2 DESCRIÇÃO DO NEGÓCIO

7.2.1 Visão:

Ser marca reconhecida pela sociedade brasileira e símbolo de confiança e inovação.

7.2.2 Missão

Inovar e escutar clientes e usuários para desenvolver um produto confiável e que atende as necessidades de todos.

7.2.3 Valores

- Empatia
- Inovação
- Qualidade
- Segurança
- Responsabilidade Ambiental

7.2.4 Descrição Do Negócio:

SEII Automação é uma empresa que desenvolve produtos na área de automação em estacionamentos. Participa de uma indústria que vem crescendo nos últimos cinco anos, junto com o aumento do número de pessoas e de carros. Apesar de essa área ser pouco conhecido pelas pessoas a SEII visa desenvolver e manter contato direto com seus clientes e os envolvido com os produtos da marca, através de meios de comunicação como redes sociais e outros meios usados pela sociedade.

7.3 OBJETIVOS

7.3.1 Objetivos Principais

- Alcançar parcela de 20% do mercado na região sul do Brasil em cinco anos.
- Alcançar o payback em três anos e meio.

7.3.2 Objetivos Intermediários

- No primeiro ano vender quatro produtos em Curitiba.
- Durante o segundo ano vender oito sistemas na região metropolitana de Curitiba e cidades em até 150 km da capital paranaense.
- No ano seguinte, expandir para outras cidades no interior do Paraná e também cidade em Santa Catarina próximas de Curitiba.
- Durante o quarto, alcançar o *payback* de investimentos do produto.
- No quinto ano, alcançar outras cidades em Santa Catarina e também no Rio Grande do Sul.

7.4 PRODUTO E SERVIÇO

7.4.1 Descrições Do Produto E Serviços

O mapTrace é um sistema integrado entre módulos de estacionamento e a central de gerenciamento e esta ainda faz interface com um monitor. Os módulos servem para informar a central sobre a situação da vaga e também para auxiliar os motoristas que não se aproximem muito da parede ou de obstáculo no fim da vaga. Já o módulo central processa os dados oriundos dos periféricos e informa os motoristas através de um display, numa uma representação gráfica do estacionamento, mostra as vagas livres e também alguns dados sobre o estacionamento como a quantidade de vagas.

Esse sistema irá auxiliar de forma mais intuitiva o motorista sobre onde estão as vagas, oferecendo mais agilidade para estacionar, resolvendo o problema do tempo desperdiçado pelo motorista dentro do estacionamento. Isso permite uma melhor rotatividade dos carros nas vagas, tornando as vagas menos ociosas.

7.4.2 Análise Comparativa

A maioria das empresas na área de automação para estacionamentos oferecem produtos como cancelas, catracas e formas de pagamento, são meios para facilitar apenas na entrada e na hora da cobrança. Já as empresas com produtos concorrentes são bem reduzidas e o produto ainda possui algumas diferenças como informar o motorista sobre a quantidade de vagas em uma área do estacionamento sem dizer onde exatamente estão. Alguns desses produtos ainda exigem uma instalação mais complexa como passagem de cabos de alimentação e também os de comunicação entre os módulos.

7.4.3 Tecnologia

O mapTrace conta com comunicação sem fio entre os módulos, retirando a necessidade do uso de cabos para interligar o sistema, e, além disso, os módulos periféricos contam com bateria interna para alimentação. Tudo isso permite reduzir gastos com cabos e facilitar na instalação do produto, causando menos impacto no estacionamento e reduzindo as alterações na instalação.

7.4.4 Produtos E Serviços Futuros

- Aplicativo para smartphone em que o usuário poderá acessar o mapa do estacionamento que se encontra e observar as situações das vagas, transmitido via rede interna do estabelecimento.
- Sensores de estacionamento que serão embutidos nas vagas e enviam para o motorista num aplicativo de celular como ele está estacionando.

7.5 ANÁLISE DE MERCADO RESUMIDA

Observando a oportunidade do mercado, existe uma expansão em todo o Brasil e na região Sul. A SEII Automação participa do segmento com o sistema gerenciador de vagas para shoppings e supermercados. São no total, 86 shoppings desde o Rio Grande do Sul até o Paraná, e com previsão de lançamento de mais cinco shoppings ainda no ano de 2013. Ainda não são muitos locais que possuem

um sistema desses, em Curitiba são apenas dois em 16 shoppings que usam um sistema de gerência. Outro ponto é o fato de as empresas concorrentes não serem e nem possuírem uma filial na região sul, deixando um setor para ser aproveitado.

7.5.1 Segmentação De Mercado

Procurando por possíveis clientes, podemos separar os estacionamentos privados que não sejam residenciais. Sendo eles estacionamentos de shoppings, supermercados, empresas, universidades e outros menores.

Estabelecimentos	Quantidade
Universidades	46
Shoppings	86
Supermercados (alvos)	120
Grandes Empresas	49.555

Quadro 3 - Quantidade de estabelecimentos na região sul do Brasil

Fonte: Aatoria própria (2013)

7.5.2 Segmento Alvo De Mercado

Observando os dados podemos observar que existe um grande número de empresas que possam ter estacionamento. Porém usando como foco vender um produto que ofereça melhor rotatividade de carro nas vagas, o produto se torna interessante para estabelecimentos em que o motorista vai para o local, mas que não fique durante muito tempo, no máximo algumas horas. Shoppings e Supermercados possuem essa característica. Confirmando, há estabelecimentos que estimam uma rotatividade de 20.000 carros por dia e 1,5 milhões de pessoas por mês.

7.5.2.1 Necessidade do mercado

Aumentar a eficiência das vagas de estacionamentos, através da rotatividade de carros. Manter as vagas menos ociosa possível e reduzindo o tráfego

dentro do estacionamento, assim permitindo que os motoristas usem mais tempo dentro do estabelecimento do que perdido no estacionamento.

7.5.2.2 Tendências do mercado

Apesar do advento da internet e as pessoas realizando compras eletronicamente, os shoppings e supermercados ainda possuem um nível de tráfego alto de pessoas. Isso se deve ao fato de as pessoas procurem por lazer e usos de serviços. Esses locais oferecem além de vendas de produtos, existem serviços básicos como lavanderia, salão de beleza, cinema, caixas eletrônicas de banco, banquinhas de revistas, *lan house*, enfim, muitos outros são oferecidos. De todas as atividades nos shoppings, as vendas ainda lideram com 38%, segundo com 19% as pessoas vão a passeio e 12% pela praça de alimentação.

7.5.2.3 Crescimento do mercado

Ano	Número de shoppings	Tráfego de pessoas (milhões/mês)
2008	376	325
2009	392	328
2010	408	329
2011	430	376
2012	457	398

Quadro 4 - Quantidade de shoppings no Brasil

Fonte: <http://www.portaldoshopping.com.br/numeros-do-setor/evolucao-do-setor>

Supermercado	Quantidade em 2011	Quantidade em 2012
Carrefour	210	218
Wal-Mart	521	547
Angeloni	23	26
CIA Zaffari	29	30
Irmãos Muffato	35	37
Condor	33	35
Total	851	893

Quadro 5 - Números dos grandes supermercados na região sul em 2011 e 2012

Fonte: <http://www.sm.com.br/Ranking-Supermercados-38>

Podemos ver conforme a tabela 2, que durante os últimos seis anos o número de shoppings e supermercados continua crescendo e também o tráfego de pessoas. A existência da internet não se torna um problema e não influencia a redução de movimento.

Região	Número de shoppings	% do Total
Norte	21	4,4%
Nordeste	64	13,4%
Centro-Oeste	43	9,0%
Sudeste	262	55,0%
Sul	86	18,1%
Total	476	100%

Quadro 6 - Quantidade de shoppings em cada região do Brasil

Fonte: <http://www.portaldoshopping.com.br/numeros-do-setor/dados-regionais-de-shopping-centers>

Pela tabela 4, podemos ver que a região sul é a segunda em potencial, perdendo para os estados do sudeste. Aliado a esses dados existe previsão de quatro novos shoppings em Rio Grande do Sul e um no Paraná até o final do ano de 2013 e em Curitiba existe projetos para três novos shoppings na cidade sem uma data específica ainda.

7.5.3 Análise Da Indústria

7.5.3.1 Players

Dentro da indústria de estacionamento, participam os estabelecimentos que fazem uso de um local para os motoristas possam vir e deixar seus carros. Os shoppings, supermercados de todos os tamanhos, as empresas com estacionamento, os clubes e associações, os parques e alguns museus encaixam-se nesse padrão. Mas cada um deles possui características especiais que influenciam no nível requerido de automação em seus estacionamentos.

Nessa área podemos ver ainda as diversas empresas atuando cada qual com um foco diferente. Existem empresas que oferecem apenas o serviço de cobrança pelo tempo estacionado, como a AutoPark. Outras que oferecem produtos de catracas, cancelas e guichês de pagamentos que também são empresas que

cuidam apenas da entrada do motorista e do tempo utilizados por ele. Algumas empresas desenvolvem produtos para segurança interna, como câmeras e barreiras para impedir colisão carros. Poucas empresas oferecem um sistema que interage mais com o cliente e as vagas, os sistemas de gerencia de vagas.

7.5.3.2 Modelo de distribuição

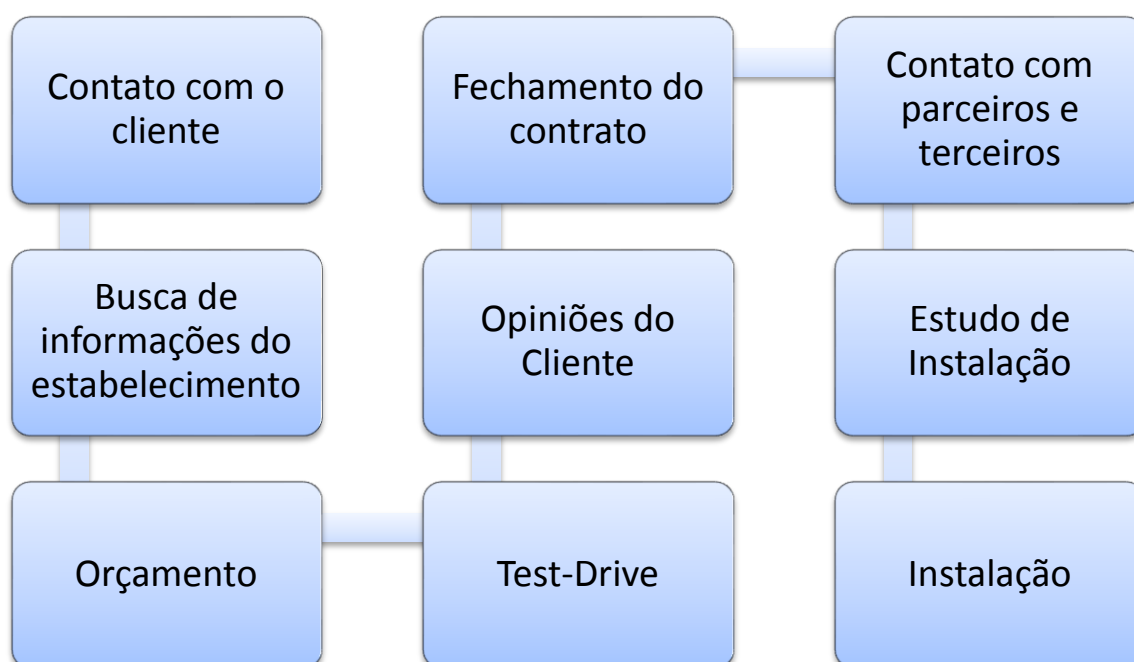


Figura 54 - Fluxograma da venda do mapTrace ao cliente
 Fonte: Autoria própria (2013)

Nossa distribuição baseia-se em relacionamento direto com o cliente, através de vista ou pelo contato do cliente. É necessário então recolher informações e dados do estacionamento do cliente, para então elaborar um projeto e o orçamento. O cliente pode requerer por testar o produto num pequeno espaço e tempo. Acabado o tempo de teste, recolhe-se a opinião do cliente e caso aceite instalar o produto, é fechado o acordo. Entra na etapa de instalação que é primeiro entrar em contato com os terceiros, fazendo um estudo da instalação. Fechado isso ocorreu à instalação do produto.

7.5.3.3 Modelo de competitividade

Dentro do mercado, as empresas vendem seus produtos na forma direta, com preço relativamente elevado e atendendo todo o país.

Os clientes devem pedir um orçamento e informar os dados do estacionamento em questão. As empresas então enviam um técnico para avaliação do local. O preço é elevado para gastar os gastos com despesas, como transporte ida ao local; Por ultimo as empresas fazem uso de revendedores de produtos, isso deixa uma lacuna por faltar à presença da empresa numa região.

7.5.3.4 Principais players

Participando com o mapTrace, a SEII Automação tem como principais concorrentes três empresas, sendo duas de São Paulo e a outra de Pernambuco. A Nepos, Prosiga e a FastPark. As três empresas vendem o produto de forma parecida. Porém a Prosiga participa do mercado nacional através de revendedores nas diversas cidades do país.

Outro player fundamental são os fornecedores da matéria para a fabricação dos produtos. Entre eles se destaca a RS Components, empresa que fornece a base da central dos módulos e a Texas Instruments, que fabrica os componentes principais para a produção dos módulos periféricos.

7.6 PROPOSTA DE VALOR

O mapTrace é um sistema de gerência de estacionamento que permite ao usuário observar onde estão as vagas livres através de um monitor contendo um mapa e as informações do estacionamento. Todas as informações são atualizadas através dos módulos de estacionamento que colocados nas vagas também auxiliam para estacionar, apitando caso esteja muito próximo do sensor. O sistema permite maior rapidez na hora de procurar um local para deixar o carro, trazendo um diferencial, agilidade e comodidade ao motorista. Auxilia também na rotatividade dos

carros e melhor aproveitamento das vagas. Assim que um carro sai outro já pode ir pegar a vaga e também pode melhorar o aproveitamento mostram as vagas e os motoristas se distribuem melhor.

7.7 ESTRATÉGIAS

7.7.1 Diferenciais Competitivos

O mapTrace oferece uma comodidade maior para o motorista, uma agilidade ao visualizar as vagas de forma mais preciso e intuitiva, usando o monitor ou displays espalhados pelo estacionamento e módulos sem fio, para diminuir o trabalho de implantação e impacto para o estabelecimento.

7.7.2 Estratégia De Marketing

Entrar em contato com futuro cliente apresentando o produto e oferecendo pra teste. Assim que for ganho mercado, utilizar os clientes como exemplos de caso de sucesso.

7.7.2.1 Estratégia de preços

Para ganhar espaço no mercado, o sistema integrado será vendido num preço um pouco menor que da concorrência. Nosso custo fixo permanece em pouco mais de 60 mil total, que envolve o aluguel, depreciação, segurança do imóvel e o salario do pessoal da produção.

O Custo variável envolve a matéria prima e empresa terceiras que dependendo do tamanho do estacionamento do cliente.

O preço será R\$ 250,00 dos módulos de estacionamento enquanto o há empresa que vende por R\$ 300,00. Um preço que permite pagar as despesas. Já as centrais serão vendidas por R\$ 350,00. Esses valores permitem que o retorno seja de 100% sobre o custo total e ajude a pagar as despesas.

7.7.2.2 Estratégia de promoção

Para incentivar a venda do produto, oferecemos alguns módulos periféricos junto de uma central para que sejam testados pelo cliente. Dessa forma há um preço ser cobrado, que será revertido na hora da compra do sistema completo.

7.7.2.3 Estratégia de distribuição

A venda se dará de forma direta. Na qual o cliente irá entrar em contato com a empresa de forma de diversos tipos como site, telefone, e-mail cadastro, redes sociais. Será necessário dados do cliente e do estabelecimento, e então é enviado um técnico da empresa para analisar fisicamente o local, e depois disso os módulos são enviados por frete até o local e a equipe de instalação instala no local necessário. Isso tudo tendo a presença de um técnico da SEII.

7.7.3 Estratégia De Vendas

7.7.3.1 Projeção de vendas

Projeção de vendas	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5
mapTrace	4	7	9	10	11
Vendas Totais	4	7	9	10	11

Quadro 7 - Projeção de vendas do maptrace ao longo de 5 anos

Fonte: Aatoria própria (2013)

7.7.3.2 Plano de vendas

Consiste em entrar contato com o cliente, mostrar os benefícios do produto. Fazer um orçamento e oferecer o produto para teste no estabelecimento. Caso seja aceito fazer o orçamento final com melhorias desejadas e vender produto.

7.7.4 Alianças Estratégicas

Para a venda do mapTrace, a SEII Automação conta com participação de empresas para desenhar o layout do estacionamento, outra para fazer a instalação dos módulos e mais uma empresa que fabrica monitores.

7.7.5 Cronograma

SEII Automação	Data Início	Data Fim	Budget	Responsável	Departamento
Desenvolvimento do mapTrace	-	30/11/2013	R\$ -	Humberto e Marlos	Engenharia
Adquirir investimento	01/11/2013	30/04/2014	R\$ -	Humberto e Marlos	Administrativo e Marketing
Constituição legal da empresa	01/01/2014	28/02/2014	R\$ 2.590,00	Humberto e Marlos	Administrativo
Montagem do site, rede social.	01/03/2014	31/05/2014	R\$ 50,00	Humberto e Marlos	Engenharia e Marketing
Pesquisa de local de trabalho	01/02/2014	28/02/2014	R\$ -	Humberto e Marlos	Administrativo
Locação de imóvel	01/03/2014	01/03/2014	R\$ 3.000,00	Humberto e Marlos	Administrativo
Compra de móveis e equipamentos	01/03/2014	31/03/2014	R\$ 45.000,00	Humberto e Marlos	Administrativo
Contratação de profissionais	01/04/2014	15/04/2014	R\$ -	Humberto e Marlos	Administrativo
Realizar parcerias	01/04/2014	30/04/2014	R\$ -	Humberto e Marlos	Administrativo
Início da produção	01/06/2014	-	R\$ -	Marlos e auxiliar de produção	Engenharia
Lançamento do site e dos meios de rede sociais.	01/06/2014	-	R\$ -	Analista de Marketing	Marketing
Lançamento de marketing	01/06/2014	-	R\$ -	Analista de Marketing	Marketing
Vender quatro produtos em Curitiba	-	Final do primeiro ano	R\$ -	SEII Automação	
Alcançar o Payback de investimento	-	Início do quarto ano	R\$ -	SEII Automação	
Total			R\$ 50.640,00		

Quadro 8 - Cronograma

Fonte: Autoria própria (2013)

7.8 GESTÃO

7.8.1 Estrutura Organizacional



Figura 55 - Estrutura por funções da SEII Automação
Fonte: Autoria própria (2013)

7.8.2 Equipe

7.8.2.1 Administrador:

Cuida dos assuntos internos não ligados a parte de engenharia da empresa. Primeiro são os assuntos que envolvem capital da empresa, a parte financeira e a contabilidade. Outra atividade é a parte de gestão de pessoas, remuneração, treinamentos, motivação, contratação. Realiza também as tarefas de logística da entrega de produto e o envio de pessoal para as visitas aos clientes. Por último é o responsável pela gestão de projetos, planejamento, desenvolvimento, direção e controle das atividades.

7.8.2.2 Engenheiro:

Responsável pela engenharia da empresa. Realiza pesquisas e desenvolver novas tecnologias. Responsável por garantir a instalação dos produtos, da manutenção e dos serviços técnicos oferecidos. Responsável ainda pela equipe de produção da empresa.

7.8.2.3 Analista em marketing

Responsável por assuntos externos da empresa. Contato com o cliente, vendas, pesquisas de satisfação e *feedback*. Desenvolve meios promocionais dos produtos. Pesquisa por inovações, procurar dados e as tendências do mercado.

7.8.2.4 Auxiliar de produção

Parte da equipe de produção que produz os módulos e realização testes de funcionalidade.

7.8.3 Quadro De Pessoal

Produção (em milhares de R\$)	Ano 1	Ano 2	Ano 3
Auxiliar de Produção	11,40/1	22,80/2	22,80/2
Subtotal	11,40	22,80	22,80

Marketing e Vendas (em milhares de R\$)	Ano 1	Ano 2	Ano 3
Analista em Marketing	27,60/1	27,60/1	27,60/1
Subtotal	27,60	27,60	27,60

Administração e outros (em milhares de R\$)	Ano 1	Ano 2	Ano 3
Sócios	24,00/2	24,00/2	24,00/2
Subtotal	24,00	24,00	24,00

Total de Pessoas	4	5	6
Total Folha (em milhares de R\$)	63,00	74,40	74,40
Benefícios e Obrigações (em milhares de R\$)	29,80	35,19	35,19
Total Gasto com Folha (em milhares de R\$)	92,80	109,59	109,59

Quadro 9 - Quadro de pessoal ao longo de três anos e seu custos

Fonte: Autoria própria (2013)

7.9 PLANO FINANCEIRO

7.9.1 Considerações

- Alcançar os objetivos necessários de 20% do mercado.
- Expansão do mercado ao longo de cinco anos.
- Venda de 500 módulos para shoppings e 100 para supermercados.
- Imposto sobre as venda de 25% para vendas até R\$ 350.000,00 e acima disso, 35% das vendas.
- Imposto de renda de 10%.
- Salários congelados ao longo dos cinco anos.
- Aluguel congelado ao longo dos cinco anos.
- Valor das matérias-primas congelados ao longo dos cinco anos.

7.9.2 Análise Do *Break-Even*

Break-Even	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5
Venda de break even (mínimo)	514	723	844	1012	1179
Custo de break-even (R\$)	128.424	180.606	210.866	252.788	294.710
Custo Total por unidade (R\$)	107,02	95,06	84,35	84,26	84,20

Quadro 10 - Dados sobre o *Break Even* do mapTrace

Fonte: Autoria própria (2013)

7.9.3 Projeção Do Resultado

D.R.E. (em milhares de reais)	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5
Receita Bruta	321,40	512,45	673,15	803,50	933,85
(-) % dos Impostos sobre Vendas	80,35	179,36	235,60	281,23	326,85
Receita Líquida	241,05	333,09	437,55	522,28	607,00
(-) CSP	128,42	180,61	210,87	252,79	294,71
Lucro Bruto	112,63	152,49	226,68	269,49	312,29
(-) Despesas	143,41	168,41	179,91	204,91	237,91
Lucro Operacional	-30,78	-15,92	46,77	64,58	74,38
(-) Imposto de Renda	3,08	1,59	4,68	6,46	7,44
Lucro Líquido	-33,86	-17,51	42,10	58,12	66,95
Margem de Lucro Líquido	-10,5%	-3,4%	6,3%	7,2%	7,2%

Quadro 11 - Projeção da D.R.E da SEII Automação

Fonte: Autoria própria (2013)

7.9.4 Projeção Do Fluxo De Caixa

FLUXO DE CAIXA	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5
Saldo inicial	150,00	55,89	27,30	58,32	105,97
Entradas	321,40	512,45	673,15	803,50	933,85
Receita de Serviços	321,40	512,45	673,15	803,50	933,85
Saídas	415,51	541,04	642,13	755,85	868,77
Aluguel+Seguro	42,00	42,00	42,00	42,00	42,00
Matéria prima	60,52	95,91	126,17	151,30	176,43
Folha Administrativa	100,41	100,41	100,41	100,41	100,41
Folha Operacional	22,18	44,37	44,37	66,55	88,74
Terceirizados	23,00	39,00	50,50	57,50	64,50
Publicidade	24,00	24,00	24,00	36,00	48,00
Luz + Água	5,40	5,40	5,40	5,40	5,40
Telefone + Internet	9,00	9,00	9,00	9,00	9,00
Equipamentos + Móveis	45,56	0,00	0,00	0,00	0,00
Imposto de Renda	3,08	1,59	4,68	6,46	7,44
Impostos sobre Vendas	80,35	179,36	235,60	281,23	326,85
Saldo final	55,89	27,30	58,32	105,97	171,05
Payback Simples	-94,11	-66,80	-8,48	97,49	268,54

Quadro 12 - Projeção de fluxo de caixa da SEII Automação

Fonte: Autoria própria (2013)

8 CONSIDERAÇÕES FINAIS

8.1 CONSIDERAÇÕES SOBRE O DESENVOLVIMENTO DO PROJETO

O projeto do Sistema de Gerência de Vagas de Estacionamento foi certamente um desafio para a equipe. Além do conhecimento de *hardware* e *software*, houve a necessidade de aprofundamento na área de comunicação sem fio, além do estudo de materiais que servissem ao nosso propósito, como o sensor de distância, microcontrolador, alimentação e a *Raspberry Pi*.

Este projeto teve como objetivo criar um sistema para auxiliar motoristas para encontrar as vagas na hora de estacionar. Foi-se alcançado o objetivo técnico de desenvolver o produto final, que é o conjunto central-sensores e ambos comunicarem entre si. Vale ressaltar que foi possível desenvolver o projeto com a comunicação sem fio e baterias. Isso facilita na hora de implantar o sistema, pois não é necessário que a passagem de muitos cabos que poderiam ser usados para comunicação e a alimentação dos sensores, assim não é necessária alterar a infraestrutura de um estacionamento.

8.2 DIFICULDADES ENCONTRADAS

O projeto apresentou algumas dificuldades durante o seu desenvolvimento, dentre as quais se pode destacar a utilização da *Raspberry Pi*, em que se tiveram problemas no que tange à compatibilidade de linguagens de programação e na capacidade da placa. Outra dificuldade foi o entendimento do dispositivo utilizado para fazer a rede sem fio, pois não se tinha nenhum conhecimento algum prévio, o que demandou um tempo razoável de estudo para entender como funcionava, como poderia ser utilizado e suas limitações.

Outra dificuldade foi no desenvolvimento da rede sem fio como um todo, pois foi necessário encontrar uma maneira de fazer com que a comunicação ocorresse para redes com muitos módulos, porque o dispositivo de comunicação sem fio utilizado tem uma limitação no número com que pode se comunicar com outros elementos.

Apesar de o projeto ter sido desenvolvido com o intuito de facilitar os motoristas no estacionamento, não foi possível realizar tal teste, para que pudesse quantificar os benefícios da utilização do sistema, uma vez que é necessário um estacionamento maior, como cenário de teste. Outra dificuldade ainda apresentada pelo sistema é quando as baterias ficam sem carga, sendo necessário ainda fazer a troca manual delas.

8.3 OBJETIVOS ALCANÇADOS

O sistema mostrou-se muito eficiente para sistemas pequenos, conseguindo mostrar em tempo real os estados das vagas do estacionamento. Os módulos consegue ler corretamente as alterações da vaga, apesar das muitas interferências que podem ocorrer dentro da vaga. A rede sem fio consegue fazer a configuração dos módulos, enviar de forma correta os estados das vagas até a central e consegue reestabelecer a conexão da rede sem ter reconfigurar tudo, apenas o módulo que por algum motivo (acabar a bateria, ser danificado ou mesmo se o transceptor parar de funcionar) parou de se comunicar com as demais.

A interface gráfica mostrou-se fácil de utilizar, permitindo que o motorista consiga enxergar os estados das vagas numa planta do estacionamento real. Para quem configura a central, é fácil interagir, pois permite que possa configurar, mudar a configuração da rede a qualquer momento, parar o funcionamento da mesma a qualquer momento, verificar perdas de conexão para restabelecer o funcionamento de forma simples e fácil, tudo através da interface gráfica.

8.4 MELHORIAS FUTURAS

Como propostas futuras de melhorias do projeto, primeiramente realizar um teste num estacionamento maior e com mais vagas. Para verificar as respostas e aperfeiçoar o sistema. Desenvolver uma estrutura para informar ao motorista também possa ver as vagas livres de longe, melhorar a interface para o motorista e

aprimorar os elementos dos sensores e assim diminuir o consumo e o aproveitamento das baterias.

Além disso, outras melhorias seriam construir uma estrutura de proteção para que o módulo não fique exposto, a comunicação entre os módulos e desenvolver os testes que faltaram desenvolver para completar a estrutura para o plano de negócios.

REFERÊNCIAS

AIRSPAYCE. **C library for Broadcom BCM 2835 as used in Raspberry Pi**. Julho 2011. Disponível em: < <http://www.airspayce.com/mikem/bcm2835/>>. Acesso em: 19 fev.2014.

ALVES, Francinildo Ramos; ARAÚJO FILHO, José Sérgio de; BRANDÃO, Nilton da Silva; MATIAS, Rafael de Almeida; PINHEIRO JUNIOR, Francisco Aroldo. **Redes Sem Fio I: Análise de Vulnerabilidade**. Outubro 2010. Disponível em: <<http://www.teleco.com.br/tutoriais/tutorialredesemfio1>>. Acesso em: 19 fev. 2014.

AYCOCK, Steve. **The history of microcontroller**. Agosto 2011. Disponível em: <http://www.ehow.com/info_10018768_history-microcontroller.html>. Acesso em: 12 fev. 2014.

Cplusplus.com. **Classes (I)**. Abril 2013. Disponível em: <<http://www.cplusplus.com/doc/tutorial/classes/>>. Acesso em: 20 fev 2014.

Cplusplus.com. **std::LIST**. Abril 2013. Disponível em: <<http://www.cplusplus.com/reference/list/list/>>. Acesso em: 20 fev 2014.

Cplusplus.com. **std::fstream**. Abril 2013. Disponível em: <<http://www.cplusplus.com/reference/fstream/fstream/>>. Acesso em: 20 fev 2014.

Cplusplus.com. **std::ofstream**. Abril 2013. Disponível em: <<http://www.cplusplus.com/reference/fstream/ofstream/>>. Acesso em: 20 fev 2014.

Cplusplus.com. **std::ifstream**. Abril 2013. Disponível em: <<http://www.cplusplus.com/reference/fstream/ifstream/>>. Acesso em: 20 fev 2014.

DIGI. **RF Basics**. Outubro 2002. Disponível em: <<https://www.digi.com/technology/rf-articles/rf-basics>>. Acesso em: 19 fev. 2014.

ELECFREAKS. **2.4G Wireless nRF24L01p with PA and LNA**. Outubro 2011. Disponível em: <http://www.electfreaks.com/wiki/index.php?title=2.4G_Wireless_nRF24L01p_with_PA_and_LNA>. Acesso em: 28 fev. 2014.

ELECFREAKS. **Ultrasonic Ranging Module HC-SR04**. Abril 2010. Disponível em: <<http://users.ece.utexas.edu/~valvano/Datasheets/HCSR04b.pdf>>. Acesso em: 12 fev. 2014.

ELINUX. **RPi Low-level peripherals**. Fevereiro 2014. Disponível em: <http://elinux.org/RPi_Low-level_peripherals>. Acesso em: 19 fev.2014.

MICRODUINO. **nRF24L01+ Module | microduino.org**. Março 2011. Disponível em: <<http://www.microduino.org/shop/arduino-shields/nrf24l01-module/>>. Acesso em: 20 fev. 2014.

MSP430G2xxx Stand Alone. Março 2012. Disponível em: <http://gpio.kaltpost.de/?page_id=1079>. Acesso em: 24 fev.2014.

HORNING, Eduardo S. e HEILMANN, Erich A. **Tutorial nRF24L01+**. UTFPR, 2012.

INSIDE GADGETS. **nRF24 Multi-Network – Allowing for 255 addresses**. Junho 2013. Disponível em: < <http://www.insidegadgets.com/2013/06/09/nrf24-multi-network-allowing-for-255-addresses/>>. Acesso em: 12 julho. 2013.

ITEADSTUDIO. **Ultrasonic ranging module: HC-SR04**. Novembro 2010. Disponível em: <ftp://imall.iteadstudio.com/Modules/IM120628012_HC_SR04/DS_IM120628012_HC_SR04.pdf>. Acesso em: 12 fev. 2014.

JAXCODER. **C++ Firmware for nRF24L01+ Transceiver**. Janeiro 2013. Disponível em: <<http://jaxcoder.com/Products.aspx?id=8>>. Acesso em: 19 fev. 2014.

MANTOVANI, Suely Cunha Amaro; OKI, Nobuo. **TEEE I - Projeto de Robôs Móveis**. Julho 2013. Disponível em: <http://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/microcontrolador_atmel_1-1.pdf>. Acesso em: 12 fev. 2014.

MARQUES, Domiciano. **Ultrassom. o que é ultrassom?**. Janeiro 2012. Disponível em: < <http://www.brasile scola.com/fisica/ultrassom.htm/>>. Acesso em: 19 fev. 2014.

MELO, Carlos. **Princípios de transmissão em radio frequência**. Agosto 2011. Disponível em: <<http://carlosvmelo.wordpress.com/2011/08/09/346/>>. Acesso em: 19 fev. 2014.

NEPAR. **BASES ULTRASSONOGRÁFICAS > Conceitos Gerais**. Abril 2011. Disponível em: <<http://www.neparus.com.br/?ver=noticia¬icia=53>>. Acesso em: 12 fev. 2014.

NORDIC SEMICONDUCTORS. **nRF24L01+ Product Specification V1.0**. Setembro 2008. Disponível em: <http://www.nordicsemi.com/eng/content/download/2726/34069/file/nRF24L01P_Product_Specification_1_0.pdf>. Acesso em 08 fev. 2014.

PORTAL DO SHOPPING. **Números do setor**. Setembro 2013. Disponível em: <<http://www.portaldoshopping.com.br/numeros-do-setor/dados-regionais-de-shopping-centers>>. Acesso em: 20 fev 2014.

ROBOTSHOP. **2.4G nRF24L01 Wireless Module w / PA and LNA**. Outubro 2013. Disponível em: <<http://www.robotshop.com/en/2-4g-wireless-module-w-pa-lna.html>>. Acesso em: 28 fev. 2014.

SDL 2.0. **SDL_Init**. Disponível em: <http://wiki.libsdl.org/SDL_Init>. Acesso em: 20 fev 2014.

SDL 2.0. **Introduction to SDL 2.0**. Disponível em: <<http://wiki.libsdl.org/Introduction>>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_Surface**. Disponível em: <http://wiki.libsdl.org/SDL_Surface>. Acesso em: 20 fev 2014.

SDL Library Documentation. **SDL_SetVideoMode**. Disponível em: <<http://www.libsdl.org/release/SDL-1.2.15/docs/html/sdlsetvideomode.html>>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_image 1.2.8**. Disponível em: <http://www.libsdl.org/projects/SDL_image/docs/SDL_image.html#SEC11>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_MapRGB**. Disponível em: <http://wiki.libsdl.org/SDL_MapRGB >. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_BlitSurface**. Disponível em: <http://wiki.libsdl.org/SDL_BlitSurface>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_SetColorKey**. Disponível em: <<http://www.libsdl.org/release/SDL-1.2.15/docs/html/sdlsetcolorkey.html>>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_ttf 2.0.10**. Disponível em: <http://www.libsdl.org/projects/SDL_ttf/docs/SDL_ttf.html#SEC51>. Acesso em: 20 fev 2014.

SDL 2.0. **SDL_Event**. Disponível em: <http://wiki.libsdl.org/SDL_Event >. Acesso em: 20 fev 2014.

SUPERMERCADO MODERNO. Abril 2013. **Ranking supermercados**. Disponível em: <<http://www.sm.com.br/Ranking-Supermercados-38>>. Acesso em: 20 fev 2014.

TENET TECHNETRONICS. **Tenet TechnetronicsProduct Reviews2.4GHz Transceiver IC - nRF24L01+2.4GHz Transceiver IC - nRF24L01+**. Agosto 2008. Disponível em: <<http://www.tenettech.com/product/854/24ghz-transceiver-ic-nrf24l01>>. Acesso em: 12 fev. 2014.

TEXAS INSTRUMENTS. **Slau 318d**. Abril 2013. Disponível em: <<http://www.ti.com/lit/ug/slau318d/slau318d.pdf>>. 2013a (acesso em: 24 fev.2014).

TEXAS INSTRUMENTS. **Datasheet – MSP430G2x53**. Maio 2013. Disponível em: <<http://www.ti.com/lit/ds/symlink/msp430g2553.pdf>>. 2013b (acesso em: 19 fev.2014).

WIKIPEDIA. **Serial Peripheral Interface**. 2014. Disponível em: <http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus>. 2014a (acesso em: 12 fev. 2014).

WIKIPEDIA. **Microcontrolador**. 2014 Disponível em: <<http://pt.wikipedia.org/wiki/Microcontrolador>>. 2014b (acesso em: 12 fev. 2014).

WIKIPEDIA. **Raspberry Pi**. 2014 Disponível em:
<http://pt.wikipedia.org/wiki/Raspberry_Pi>. 2014c (acesso em: 19 fev. 2014).

APÊNDICE A - Esquemático do módulo

O esquemático feito para o módulo, sem a parte da sinalização luminosa.

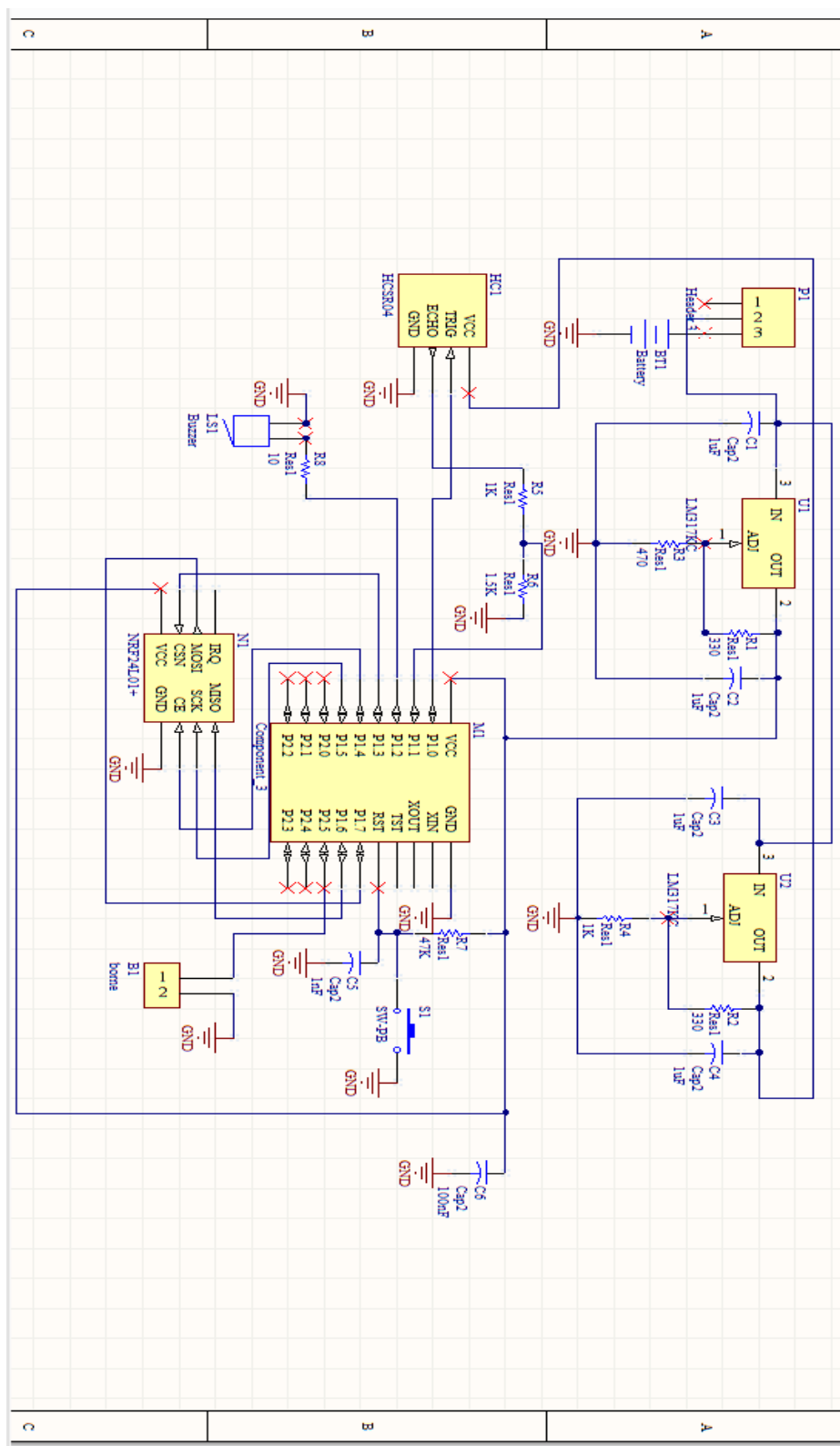


Figura 56 - Esquemático do módulo, sem a parte da sinalização luminosa
 Fonte: Autoria própria (2014)

APÊNDICE B - Placa de circuito impresso gerada para o módulo

A placa de circuito impresso feita para o módulo, sem a parte da sinalização luminosa.

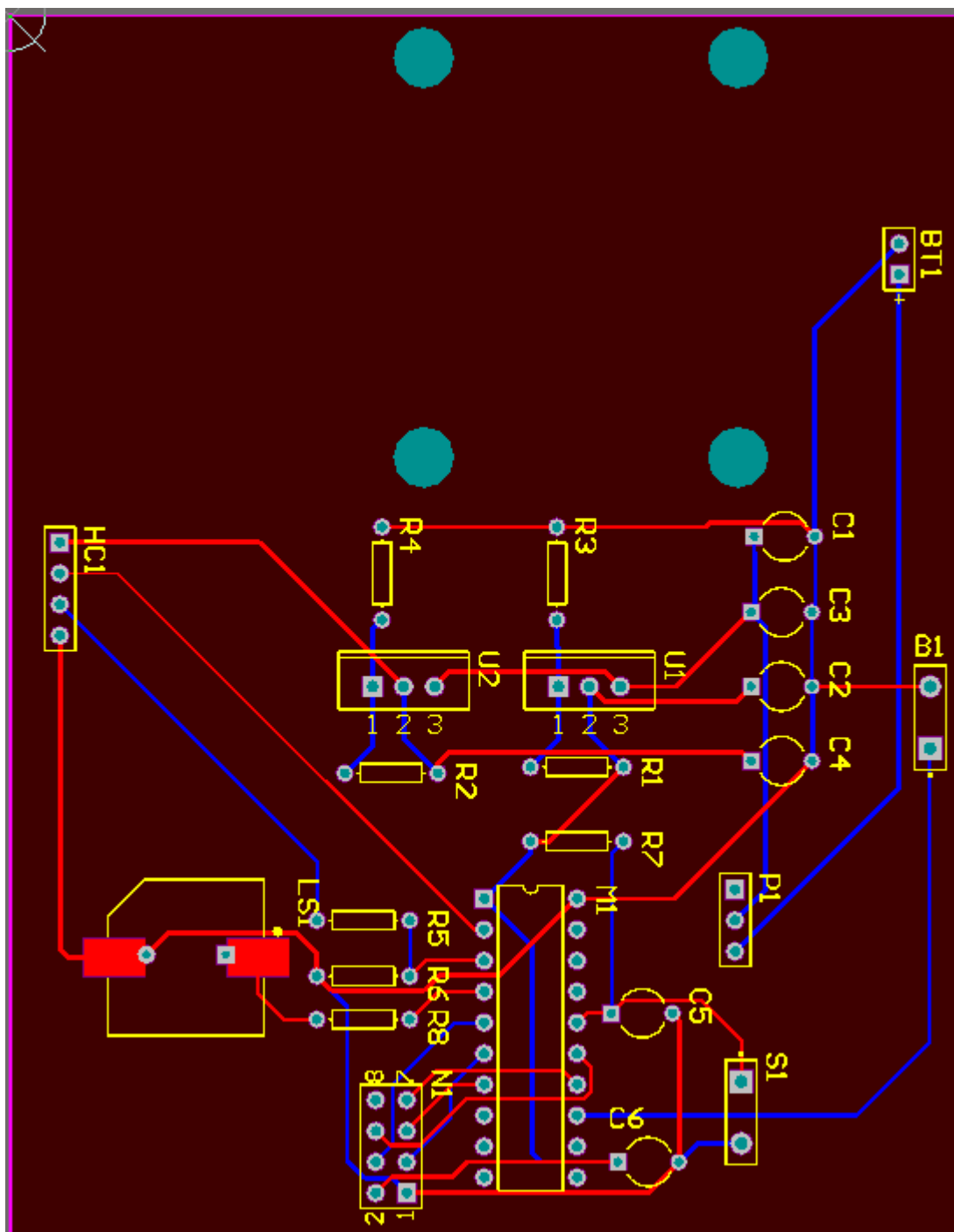


Figura 57 - Placa do circuito do módulo de estacionamento
Fonte: Autoria própria (2014)