UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GABRIEL DE SOUZA ALVES

DISPOSITIVO PARA ADQUIRIR, APRESENTAR E GUARDAR OS DADOS NO VEÍCULO DO FEB RACING

BAURU
Novembro/2019

GABRIEL DE SOUZA ALVES

DISPOSITIVO PARA ADQUIRIR, APRESENTAR E GUARDAR OS DADOS NO VEÍCULO DO FEB RACING

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. Renê Pegoraro

BAURU
Novembro/2019

Gabriel de Souza Alves Dispositivo para Adquirir, Apresentar e Guardar os Dados no Veículo do Feb Racing/ Gabriel de Souza Alves. – Bauru, Novembro/2019-
36 p. : il. (algumas color.) ; 30 cm.
Orientador: Prof. Dr. Renê Pegoraro
Trabalho de Conclusão de Curso – Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Ciência da Computação, Novembro/2019.
1. Automobilismo 2. Sensores 3. Dados 4. Dispositivo 5. *Arduino* 6. *Raspberry Pi*

Gabriel de Souza Alves

Dispositivo para Adquirir, Apresentar e Guardar os Dados no Veículo do Feb Racing

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Renê Pegoraro
Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Profª. Drª. Simone das Graças Domingues Prado
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Prof. Dr. Humberto Ferasoli Filho
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Bauru, _____ de _____ de _____.

Agradecimentos

Agradeço a minha família por todo apoio até o presente momento, me apoiando para que eu tente alcançar meus sonhos, não importa onde eu tenha que ir para realiza-los.

Agradeço a todos os meus amigos que durante todo esse ano, me apoiaram e me ajudaram a passar por essa fase.

Sou grato a todos os professores que contribuíram com a minha trajetória acadêmica, especialmente ao orientador Prof. Dr. Renê Pegoraro por acreditar que eu conseguiria desenvolver esse trabalho, pela ajuda e apoio desde o começo, também ao Prof. Dr. Wilson Massashiro Yonezawa, por ter emprestado o Raspbery Pi, que sem ele, não teria sido possível ter feito o dispositivo e a Profª. Drª. Simone das Graças Domingues Prado pela ajuda e pela compreensão ao longo do desenvolvimento do trabalho, desde o começo do ano.

Agradeço ao Vitor Marchi, que faz parte do Sagui Lab, que em parceria com a Mousta Impressoras 3D, cedeu a impressora 3D pra usar no evento Feira Maker no Interdesigners, que é a semana de Design da UNESP Bauru. Ele ajudou com a parte de impressão 3D, desde a geração do arquivo para colocar na impressora até em particularidades da mesma e ainda imprimiu para mim duas vezes a capa do *Arduino* e a base que desenvolvi, uma vez para validação com menos resolução e a segunda, com mais resolução, já com alguns acertos.

Agradeço ao Prof. Dr. Dorival Campos Rossi por incentivar a cultura maker, inclusive o fundador do Sagui Lab, um laboratório destinado a cultura maker e que, graças a uma palestra que ele ministrou em 2015 sobre o tema, num evento de computação, me inspirou a fazer meu trabalho nesse ramo.

Agradeço a Profª. Drª. Ana Beatriz Pereira de Andrade, professora de fotografia do Departamento de Design, que permitiu que eu utiliza-se espaço no laboratório de foto e ao Henry Santana, assessor técnico de fotografia do Estúdio Fotográfico FotoLab, do departamento de design, pelo acesso ao laboratório e pela ajuda para tirar as fotos dos equipamentos no laboratório.

Agradeço pela parceria com a equipe da FEB Racing, em especial o capitão, que acreditaram e toparam fazer parte do desenvolvimento do dispositivo, fazendo a parte de delimitação e aquisição dos sensores, além de terem emprestado para os testes durante o desenvolvimento.

Quem não é humilde – o que é diferente de ser subserviente – acaba perdendo demais, porque fica aprisionado dentro de si.

Mário Segio Cortella

Resumo

Com o avanço da tecnologia, cada vez mais é visto aplicações de computação em diversas áreas e com o automobilismo não é diferente. O uso de sensores para adquirir os dados é útil para comprovação de cálculos e uso no refinamento dos ajustes do carro. Para tal aplicação de tecnologia nos veículos, pode-se utilizar microcontroladores e até mesmo computadores com tamanho reduzido, para não somente trazer e gravar esses dados, mas também mostra-los para quem dirige e transmiti-los para algum lugar para um melhor entendimento do que está acontecendo, permitindo uma utilização melhor do veículo. O trabalho mostra o desenvolvimento de um dispositivo, utilizando um *Arduino* e um *Raspberry Pi*, para adquirir, gravar e mostrar os dados em uma tela sensível ao toque, além da impressão 3D de uma capa para o *Arduino* e uma base desenvolvida exclusivamente para o dispositivo desenvolvido.
Palavras-chave: Automobilismo, sensores, dados, dispositivo, *Arduino*, *Raspberry Pi*.

Abstract

With the advance of technology, it's seem more and more computer applications on various areas and with motorsports it's not different. The use of sensors to acquire data is useful to testify calculation and tweaking the car's adjustments. To such application of technology on vehicles, it's possible to use microcontrollers and even smaller computers, not only to collect and store these data, but show them to the driver and send them elsewhere to a better understanding on what's happening, leading to a better usage of the vehicle. This piece of work shows the development of a device, using a *Arduino* and a *Raspberry Pi*, to collect, store and show the data on a touch screen, in addition to 3D printed case to fit the *Arduino* and a base exclusively designed to the developed device.

Keywords: Motorsport, sensors, data, device, *Arduino*, *Raspberry Pi*.

Lista de figuras

Lista de quadros

Lista de tabelas

Lista de abreviaturas e siglas

ECU	Unidade de Controle Eletrônico do inglês Engine Control Unit
ABS	Sistema de Frenagem Anti Travamento do inglês Anti-lock Braking System
FEB	Faculdade de Engenharia de Bauru
GUI	Interface Gráfica com o Usuário do inglês Graphical User Interface
SCL	Clock Serial do inglês Serial Clock
SDA	Dados Serial do inglês Serial Data
I2C	Circuito Inter Integrado do inglês Inter-Integrated Circuit
CI	Circuito Integrado
GND	Terra, Potencial elétrico zero do inglês Ground
I/O	Entrada ou Saída do inglês In Out

Sumário

1 Introdução

Como visto em (PRASAD; BROY; KRUEGER, 2010) os carros evoluíram muito com o passar do tempo e os softwares começaram a fazer parte desse processo a partir do fim de 1960 e o começo de 1970. A sua aplicação inicialmente era um simples controle de ignição e esse começo lento da aplicação de softwares no mundo automotivo se deve ao controle de custos que faz parte da industria automobilística, dado o custo de computadores e coisas do gênero naquela época, como pode ser lido também em (PRASAD; BROY; KRUEGER, 2010) e também (BEREISA, 1983). É possível observar também em (BEREISA, 1983) que logo em 1978 já começou a ser lançados modelos de carros que tinham funções, controladas por microcontroladores, de entretenimento, conforto, conveniência e do painel de instrumentos.

Com o passar do tempo, os carros começaram a ganhar mais funcionalidades, como sistema de airbags e sistema anti travamento do freio(ABS), todas controladas por microcomputadores e, juntamente, os softwares neles instalados para tais controles e funções, como citado no (PRASAD; BROY; KRUEGER, 2010).

Os carros não são utilizados somente no nosso dia a dia, a fim de ser um meio de ir do ponto A ao ponto B, mas também em competições, como o carro de competição da F1 e da FEB Racing, que pode ser observado na Figura 1. Para estas competições, dedica-se tempo com estudos e desenvolvimento com o objetivo de ter o melhor carro, mais acertado, com melhor desempenho nas curvas, que consiga chegar mais rapidamente em sua velocidade final, que nas frenagens consiga diminuir a sua velocidade no menor distância possível sem travar as rodas, que tenha a melhor aerodinâmica para fazer as curvas na maior velocidade possível sem perder muita velocidade final, isso dentro das regulamentações de determinada categoria.

Um dos meios de se atingir esse objetivo é utilizando a telemetria, que é a coleta de um aglomerado de informações em tempo real pela equipe, desde o que está acontecendo no motor, até o que acontece com o piloto e também o log, que é a coleta dos dados e a gravação desses num arquivo num dos dispositivos que fica no carro, com os dados dos sensores durante um período, para posterior analise. De acordo com (WALDO, 2005) os carros de F1, e os pilotos, são os objetos mais instrumentados do planeta e uma corrida gerava, em 2005, mais de 1 gigabyte de dados.

O objetivo deste trabalho é construir um sistema de medição, apresentação em tempo real e salvamento dos dados obtidos durante cada prova da equipe FEB Racing, que pode ser observada na Figura 2, fornecendo dados para que a equipe possa tentar identificar situações que possam ser trabalhadas para melhorar o carro nas próximas competições.

Figura 1 – Foto do carro da FEB Racing, que será aplicado o dispositivo.

Fonte: Cedida pelo capitão da equipe FEB Racing

Figura 2 – Foto da equipe FEB Racing.

Fonte: Cedida pelo capitão da equipe FEB Racing

2 Conceitos e Materiais Envolvidos

Neste capítulo será abordado os materiais envolvidos no desenvolvimento do dispositivo de auferir dados do carro da FEB Racing, que envolvem os dispositivos utilizados, *Arduino*, *Raspberry Pi*, um multiplexador CD4052, quatro sensores de temperatura MLX90614, quatro sensores de campo magnético KY003, quatro potenciômetros, três acelerômetros/giroscópios MPU6050 e os conceitos envolvidos para o dispositivo funcionar, a transmissão de dados por barramento I2C, uso de *thread*, conversão de dados vindo de sensores e impressão 3D para o alojamento e fixação dos controladores no carro.

2.1 Arduino

Figura 3 – Foto do *Arduino Mega 2560* utilizado no dispositivo.

Fonte: Produzido pelo autor

Arduino é composto por dois produtos, um são algumas tipos de placas, de desenvolvimento aberto, ou seja, o processo de produção delas é aberto e qualquer um pode replica-las e produzi-las, desde de acordo com as políticas do registro da mesma e uma IDE para desenvolvimento de programas para serem executados nessas placas. A Figura 3 mostra a placa *Arduino* que será utilizada no dispositivo.

De acordo com o (INTRODUCTION, 2019):

> *Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.*

A placa, constituída por um microcontrolador, é de fácil desenvolvimento de protótipos, que são projetos em processo de desenvolvimento, que envolvam lidar com componentes de

baixo nível, como sensores e atuadores, com possibilidade de fácil mudanças de ligações e reprogramações para testes e validações.

O *Arduino* ATMega2560, utilizado no dispositivo, contém pinos I/O analógicas, ou seja, que tem um conversor de analógico para digital interno. Os conversores analógico digitais transformam um sinal analógico, que é um sinal contínuo que varia entre 0V e 5V, para valores discretos, definidos pela definição do conversor, por exemplo, um conversor analógico digital de 10 bits, ao ler um valor analógico, irá transforma-lo num número binário entre 0 à 1023, como está exemplificado na Tabela 1, que é o caso do conversor presente nos pinos I/O analógicos do *Arduino*. O *Arduino* também contém pinos I/O digitais, ou seja, que sua entrada ou saída é sensível ou pode assumir os valores de 5V ou 0V sem valores intermediários. Há também pinos I/O para aplicações mais específicas, como os pinos para comunicação I2C, no caso, pinos de Dados Serial(SDA) e de Clock Serial(SCL).

Tabela 1 – Tabela exemplo de um conversor analógico digital de 10 bits com a conversão de um sinal analógico, entre 0V e 5V, para um sinal digital entre 0 e 1023. Os valores da tensão de entrada estão aproximados

Intervalo de tensões de entrada analógicas contínuas	**Valores digitais de saída**
0V à 0,00487V	00000 00000
0,00488V à 0,00975V	00000 00001
0,00976V à 0,01463V	00000 00010
0,01464V à 0,01951	00000 00011
0,01952V à 0,02439	00000 00100
0,02440V à 0,02927	00000 00101

Arduino é ideal para prototipagem, dado sua fácil manipulação tanto do hardware tanto do software, que utiliza C++, baseado no *Wiring*, para o controle do mesmo. *Wiring* é um *framework* de programação de microcontroladores de código aberto, que qualquer um pode contribuir e se basear nele, que foi o caso do software para fazer os programas para o *Arduino*.

2.2 *Raspberry Pi*

Figura 4 – Foto do *Raspberry Pi* Utilizado no Dispositivo.

Fonte: Produzido pelo autor

Raspberry Pi envolve todo um grande projeto para espalhar o uso dos computadores. Ele é um computador completo, em um tamanho reduzido, que pode ser usado de várias formas, inclusive como um computador embarcado. Ele pode ser observado na Figura 4.

De acordo com o (FAQS, 2019):

> *Raspberry Pi is the third best-selling computer brand in the world. The Raspberry Pi is a credit card–sized computer that plugs into your TV or display, and a keyboard and mouse. You can use it to learn coding and to build electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, browsing the internet, and playing games. It also plays high-definition video. The Raspberry Pi is being used by adults and children all over the world to learn programming and digital making.[...]*

O *Raspberry Pi* é ideal para protótipos que envolvam a utilização de aplicações de alto nível, como utilização de banco de dados, de programas com interfaces gráficas mais complexas caso ele esteja ligado a uma tela, acesso a internet nativo e utilização de protocolos de rede, utilização de dispositivos ligados as suas portas USB, cálculos mais complexos e até mesmo utilizar seus pinos I/O, parecidos com o *Arduino* pois é um computador, com sistema operacional Raspibian baseado em Linux, que possibilita uma gama maior de ferramentas de software para serem utilizadas.

2.3 Sensor de Temperatura MLX90614

O sensor MLX90614 funciona lendo, por infra vermelho, sem a necessidade de contato, a temperatura do objeto e também mede a temperatura ambiente, que não será utilizada nesse dispositivo.

Figura 5 – Foto de um MLX90614 com a placa, já pronto para ser ligado diretamente num *Arduino*, por exemplo.

Fonte: Site de venda ImagingSys [1]

[1] https://www.imagingsys.com/hiletgo-mlx90614-to-39-ir-infrared-thermometer-non-contact/

Ele utiliza comunicação I2C para transmitir, sem parar, os valores das duas temperaturas o tempo todo, com uma precisão padrão de 0,02°C utilizando esse protocolo. Sua tensão de funcionamento pode ser 5V ou 3,3V.

O sensor pode ser observado na Figura 5, no qual nota-se os dois pinos de alimentação VIN, que é ligado no positivo da fonte de alimentação e GND, ligado no negativo da bateria ou no 0V. Também estão presentes os pinos de comunicação pelo barramento I2C, SCL e SDA.

2.4 Sensor Acelerômetro Giroscópio MPU6050

Figura 6 – Foto de um MPU6050 com a placa, já pronto para ser ligado diretamente num *Arduino*, por exemplo.

Fonte: Site FlipeFlop [2]

O sensor MPU6050 oferece muitos recursos, mas para o dispositivo serão utilizados um giroscópio de 3 eixos e um acelerômetro de 3 eixos, são usados internamente um conversor analógico digital de 16 bits para o giroscópio e um para o acelerômetro.

Ele utiliza comunicação I2C para transmitir, sem parar, os valores vindos dos conversores, com a precisão, para o giroscópio, de ± 250, ±500, ±1000 e ±2000°/s e para o acelerômetro, de ±2, ±4, ±-8 e ±16g(gravidade). Para o dispositivo será utilizado a precisão de ±250°/s para o giroscópio e ±2g para o acelerômetro. Sua tensão de funcionamento pode ser 5V ou 3.3V. Para controlar seu endereço, ele tem o pino AD0 para selecionar.

O sensor pode ser observado na Figura 6, no qual nota-se os pinos de alimentação do sensor VCC e GND, onde serão ligados, respectivamente, o 5V e o 0V. Além deles, observa-se os pinos de comunicação que serão utilizados no dispositivo para liga-lo ao barramento I2C, sendo eles o SCL e SDA e o pino AD0 que controla qual endereço o sensor está utilizando.

[2] https://www.filipeflop.com/produto/acelerometro-e-giroscopio-3-eixos-6-dof-mpu-6050/

2.5 Sensor de Campo Magnético KY003

O sensor KY003 mede a presença ou não de um campo magnético próximo à ele utilizando o efeito hall. Quando próximo à um campo magnético, ele envia sinal, através de uma saída analógica que também pode ser lida como digital pela característica do sensor pois varia entre 0V até a tensão que o alimenta, que pode ser entre 4,5V e 5V.

Figura 7 – Foto de um KY003 com a placa, já pronto para ser ligado diretamente num *Arduino*, por exemplo.

Fonte: Site de Venda Tecnotronic [3]

Apesar de seu funcionamento simples, ele será utilizado para medir a velocidade da roda, sendo contado quantas vezes o sensor variou sua entrada em determinado intervalo de tempo, assim calculando a velocidade da roda de acordo com a frequência de acionamento do sensor.

O sensor pode ser observado na Figura 7, no qual nota-se os pinos GND, 5V e o pino de sinal, da esquerda para direita, tendo como referencial a letra "S"presente imediatamente ao lado do pino de sinal. Através dele será mandado 0V ou 5V, indicando a presença do campo eletromagnético.

2.6 Potenciômetro

O potenciômetro é um resistor variável, que tem um acionador, dependendo de sua posição, modifica a resistência entre os seus terminais. Se percorrido por uma corrente elétrica, quando varia sua resistência, também varia a tensão sobre ele.

Para o dispositivo, o potenciômetro será utilizado para medir o posicionamento da suspensão, pois deixando o potenciômetro preso ao chassi do carro e prendendo o acionador do potenciômetro na parte móvel da suspensão, pelo valor da tensão de saída é possível saber o posicionamento da suspensão. Utilizando uma das entradas analógicas descritas no tópico 2.1, com um conversor A/D, é possível saber a distância entre o chassis e o solo de acordo com a tensão de saída sobre o potenciômetro.

[3] https://www.tecnotronics.com.brsensor-hall-magnetico-ky003-arduino.html

Figura 8 – Foto de um potênciometro com acionador por rotação.

Fonte: Site de Venda Bau da Eletrônica [4]

O sensor pode ser observado na Figura 8, no qual nota-se os pinos para a ligação, no qual liga-se o terminal central no 5V e o outro, em qualquer um dos lados, na entrada do conversor A/D do *Arduino*.

2.7 Protocolo I2C

De acordo com (SILVA; KASCHNY et al., 2012) o barramento I2C fisicamente é constituído de dois fios, sendo eles, um o que comunica o clock (SCL) e o outro que faz a comunicação serial bidirecional dos dados (SDA). No barramento, além dos fios serem ligados nos dispositivos, também são ligados no positivo da fonte de alimentação, utilizando resistores para tal. A fonte de alimentação para o barramento pode ser tanto 5V como 3.3V.

Os dispositivos do barramento são constituídos de um mestre e inúmeros escravos. O dispositivo mestre é quem começa a comunicação no barramento e ele, comumente, dita o clock do barramento.

Na comunicação de dados pela via de dados serial SDA, trafega informações de endereços, dados e comandos. Tais endereços pertencem aos dispositivos escravos e são utilizados pelo mestre para mandar e receber as informações sem que se confunda com quem ele está se comunicando.

Os endereços desses dispositivos tem certa flexibilidade, mas quem dita os endereços disponíveis e se haverá mais de um endereço disponível é o fabricante do dispositivo. No caso dos sensores utilizados nesse dispositivo que utilizam esse protocolo de comunicação, o MLX90614 e o MPU6050, cada um com um endereço diferente. O MLX90614, mais simples, só tem um endereço disponível para ser colocado no barramento, em notação hexadecimal, 5A, sendo definido pelo fabricante e o MPU6050, tem um pino de controle de endereço, o AD0, como mencionado na seção 2.4, com os endereços possíveis, em hexadecimal, 68 e 69, sendo definido pelo fabricante e o controle de qual endereço está sendo utilizado pelo sensor se dá

[4] https://www.baudaeletronica.com.br/potenciometro-linear-de-10k-10000.html

pelo sinal no pino de controle, sendo o endereço 68 com AD0 em 0V e endereço 69 com AD0 em 5v/3.3V., mas tem a limitação desses dois endereços apenas.

O dispositivo utiliza quatro dos sensores MLX90614 e MPU6050, e como mencionado anteriormente, o MLX90614 tem a limitação de um endereço definido possível e o MPU6050 tem a limitação de dois endereços possíveis, mas será utilizado quatro MLX90614 e três MPU6050 ligados no barramento I2C no dispositivo desenvolvido, então para resolver esse problema, foi utilizado um multiplexador.

2.8 Multiplexador CD4052

Como visto no datasheet (CD4051BC. . . , 2002), o CI CD4052 faz parte da família dos multiplexadores CD4051, CD4052 e CD4053, no qual o CD4051 é um multiplexador único de 8 canais, ou seja, em uma saída, é possível variar entre 8 entradas diferentes, o CD4052 é um multiplexador duplo de 4 canais, ou seja, em 2 saída, em cada uma delas, é possível escolher entre 4 entradas diferentes. Já o CD4053 é um multiplexador triplo com 2 canais, ou seja, através de 3 saídas, é possível escolher entre 2 entradas diferentes.

Quadro 1 – Quadro retirado do Datasheet com a Tabela Verdade da família CD405X

Truth Table

INPUT STATES				"ON" CHANNELS		
INHIBIT	C	B	A	CD4051B	CD4052B	CD4053B
0	0	0	0	0	0X, 0Y	cx, bx, ax
0	0	0	1	1	1X, 1Y	cx, bx, ay
0	0	1	0	2	2X, 2Y	cx, by, ax
0	0	1	1	3	3X, 3Y	cx, by, ay
0	1	0	0	4		cy, bx, ax
0	1	0	1	5		cy, bx, ay
0	1	1	0	6		cy, by, ax
0	1	1	1	7		cy, by, ay
1	*	*	*	NONE	NONE	NONE

*Don't Care condition.

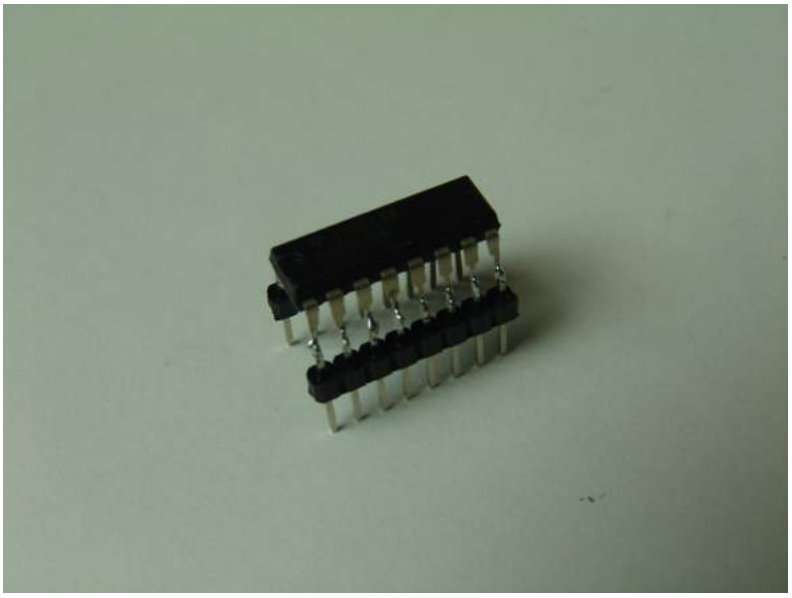
Fonte: Datasheet disponível online[5]

A seleção do canal que está saindo se faz mudando os valores das entradas do multiplexador. O multiplexador é constituído pelos canais de entrada de informação, a(s) saída(s) e os seletores de sinal. Por exemplo, no multiplexador que será utilizado no dispositivo, há 2 canais de seleção, A e B, há 8 entradas, sendo separadas em duas, sendo X0 à X3 e Y0 à Y3. Para um melhor entendimento, pode-se conferir o Quadro 1.

Como o problema em questão envolve ligar os sensores no barramento I2C, é necessário haver duas saídas, uma para o SDA e outra para o SCL. Outra necessidade do dispositivo é a quantidade de sensores por barramento e o sensor com maior limitação nessa questão é

[5] http://dalincom.ru/datasheet/CD4052.pdf

Figura 9 – Foto do multiplexador CD4052 utilizado no dispositivo com terminais soldados para facilitar a ligação com os jumpers.



Fonte: Produzido pelo autor

o MLX90614, que tem somente um endereço para ser ligado no barramento I2C, logo são necessárias pelo menos 4 entradas de dados no multiplexador. O sensor MPU6050, por ter a possibilidade de 2 endereços, necessita somente de duas entradas.

Logo, com as limitações acima citadas, o melhor CI a ser utilizado é o CD4052, como pode ser observado na Figura 9, por ter exatamente 2 saídas e 4 duplas de entradas, como exemplificado pela tabela 2.

Tabela 2 – Tabela Verdade Multiplexador CD4052 aplicado no Dispositivo.

Sinais nos Seletores	**Saída X**	**Saída Y**
A = 0, B = 0	SDA1	SCL1
A = 1, B = 0	SDA2	SCL2
A = 0, B = 1	SDA3	SCL3
A = 1, B = 1	SDA4	SCL4

2.9 Tela 3,5"sensível ao toque

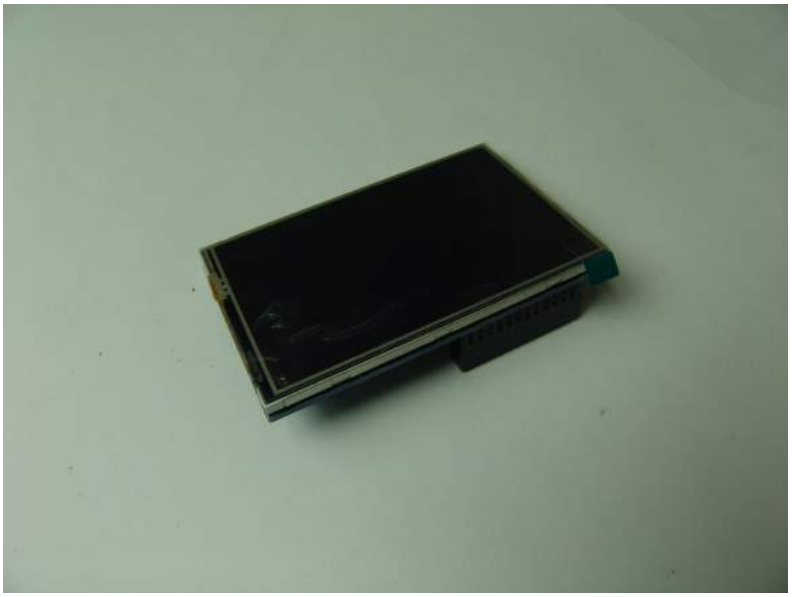
Para apresentar as informações coletadas dos sensores, será utilizada uma tela de 3,5 polegadas, que pode ser observada na Figura 10, sensível ao toque, compatível com o *Raspberry Pi*. Muito simples de ser instalada e utilizada, por ter sido feita para o *Raspberry Pi* e por ser um dispositivo popular existem muitos tutoriais disponíveis online [6] que podem ser seguidos para

[6] Tutorial utilizado para a instalação da tela presente no vídeo https://www.youtube.com/watch?v=0084bZfQG3k&t=1020s

instala-la. A sua instalação se dá encaixando os terminais da tela nos pinos I/O do *Raspberry Pi* e pela troca de arquivos do sistema operacional por arquivos já prontos, bastando troca-los.

Ao utilizar a tela e trocar os arquivos, a saída de vídeo pelo HDMI do *Raspberry Pi* não acontece mais. Para reverter, basta baixar de novo os arquivos originais do sistema.

Figura 10 – Tela 3,5"sensível ao toque que será utilizada no *Raspberry Pi*



Fonte: Produzido pelo autor

2.10 Thread

De acordo com (O. . . , 2004) a *thread* é um recurso computacional para que um programa possa executar mais de uma tarefa ao mesmo tempo. Esse tipo de recurso é interessante quando tratamos de aplicações mais modernas, que contém uma interface com o usuário e que precisa executar uma tarefa enquanto a interface continua responsiva e que pode executar outras tarefas ao mesmo tempo.

Como visto, o dispositivo envolve a leitura de sensores em tempo real, sua apresentação e a gravação dos mesmos num arquivo para posterior análise. Então, para isso, o conceito de *thread* é utilizado para fazer essas atividades ao mesmo tempo num mesmo programa.

2.11 Conversão de Dados de Sensores

Como visto anteriormente nas Seções 2.3, 2.4, 2.5 e 2.6 esses sensores capturam informações do mundo real e os transferem para o mundo digital, mas ao ler as informações mandadas pelo mesmo, será percebido que o valor lido não é o valor já na sua unidade que entendemos, por exemplo no caso do potenciômetro, explicado na Seção 2.6, não lemos o

valor analógico da tensão sobre o potenciômetro, mas sim um valor inteiro entre 0 e 1024. Isso se deve pois esses sensores trabalham convertendo o valor analógico num valor digital, utilizando um conversor A/D, que depende da precisão desse sensor e também da sensibilidade ou range escolhido, como no caso de sensores mais complexos como o acelerômetro e giroscópio explicado na Seção 2.4.

Por isso, para que o usuário que está utilizando o equipamento entenda os valores mostrados, é necessário uma conversão prévia desses valores e cada um dos sensores envolvidos no dispositivo tem suas características próprias para a conversão, que são explicados como entender esses valores nos seus respectivos *datasheets*.

3 Metodologia

3.1 Ferramentas Envolvidas

Nesse capítulo será abordado as ferramentas utilizadas para fazer a parte funcional utilizando dispositivos e conceitos anteriormente citados na 2 para o funcionamento do dispositivo.

3.2 Visual Studio 2015

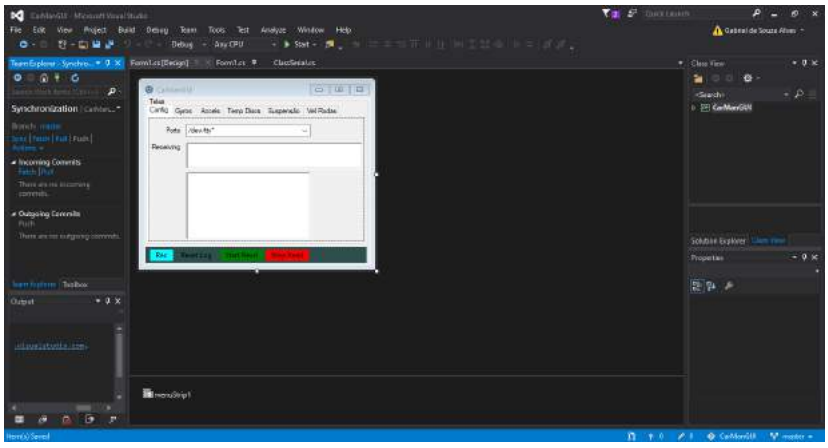
C# é uma linguagem de programação orientada a objeto multi paradigma, derivada do C++, com elementos que lembram Java, desenvolvida pela Microsoft.

A IDE *Visual Studio 2015*, que pode ser observada na Figura 11 é uma ferramenta da Microsoft para programação que utiliza a linguagem C# como principal para escrever os programas. É utilizada para fazer uma interface com o usuário (GUI) e juntamente programa-la e fazer sua parte funcional.

O *Visual Studio 2015* foi utilizado pois, oferece ferramentas muito poderosas como controle de *thread*, comunicação serial e o controle da interface, com a presença de componentes visuais padrões versáteis e de fácil uso.

A IDE tem a opção de programar em C# e o dispositivo ser compilado para ambientes .NET, necessário para poder executar no *Rasberry Pi* descrito na Seção 2.2 utilizando a ferramenta da Seção3.3.

Figura 11 – Visual Studio 2015, IDE utilizada para desenvolver a interface e parte funcional do *Raspberry Pi*.



Fonte: Produzido pelo autor

3.3 Mono

O Mono é uma biblioteca que roda programas baseados em .NET em ambientes que utilizam um sistema operacional baseado em Unix, que é o caso do *Raspberry Pi*, descrito na Seção 2.2, que utiliza o Raspibian, sistema operacional baseado no sistema operacional Linux, que foi baseado em Unix.

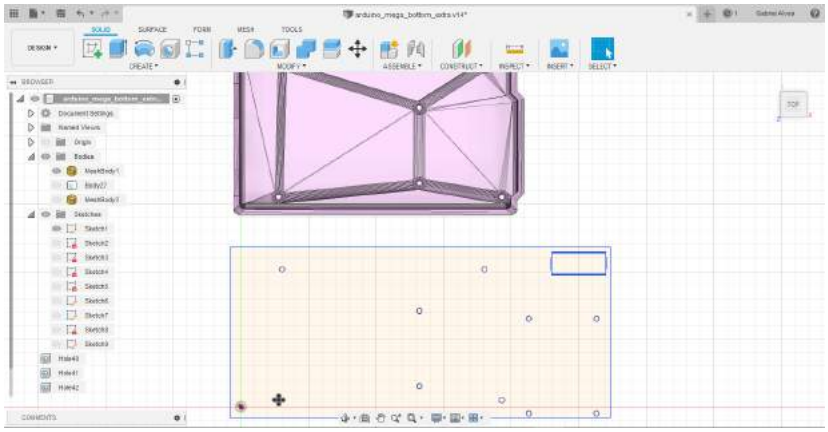
Para o dispositivo, seu uso é bem simples, é necessário apenas instala-lo no Raspibian via terminal e utiliza-lo para rodar o executável da janela feita no *Visual Studio*.

3.4 *Fusion 360*

O programa *Fusion 360*, pertencente aos softwares da AutoDesk, é um programa que permite modelar em 3D, fazer simulações e gerar um arquivo para ser impresso por uma impressora 3D posteriormente. Não somente isso, mas sua interface simples, como pode ser notada na Figura 12, permite que até mesmo a pessoa com pouco conhecimento consiga utiliza-la e faça um objeto.

Para o dispositivo, foi desenvolvido um suporte para colocar o *Arduino*, já dentro de uma capa[1], juntamente com o CI do multiplexador e lugar para a placa MCP2515 que será utilizada posteriormente no dispositivo. Esse suporte também está pronta para ser usada como a base de uma caixa, a qual as outras faces serão desenvolvidas para encaixar e proteger os componentes e ligações.

Figura 12 – AutoDesk Fusion 360, software utilizado para desenvolver a placa do suporte para o *Arduino*, MCP2515 e CD4052.



Fonte: Produzido pelo autor

[1] Disponível em: https://www.thingiverse.com/thing:1145811

3.5 Impressão 3D

Após a criação do objeto no *Fusion 360* e o arquivo exportado, podemos imprimi-lo utilizando impressoras 3D que apesar de não ser financeiramente muito acessível, existem empresas que você pode levar seu arquivo e eles irão imprimir para você. Será pago o valor do material utilizado, calculado pelo peso do produto impresso, tempo de máquina ligada e uma margem de lucro.

O processo de impressão 3D consiste no depósito de plástico, através de uma "cabeça"que se mexe em dois eixos, deposita o plástico derretido em pequena quantidade e em camadas, primeiramente numa placa de aderência e conforme a placa desce, deslocando o terceiro eixo, o depósito de plástico se faz sobre a camada já impressa. O tamanho da camada e a precisão da cabeça ao depositar o plástico são configurados no arquivo de impressão.

4 Características do Dispositivo

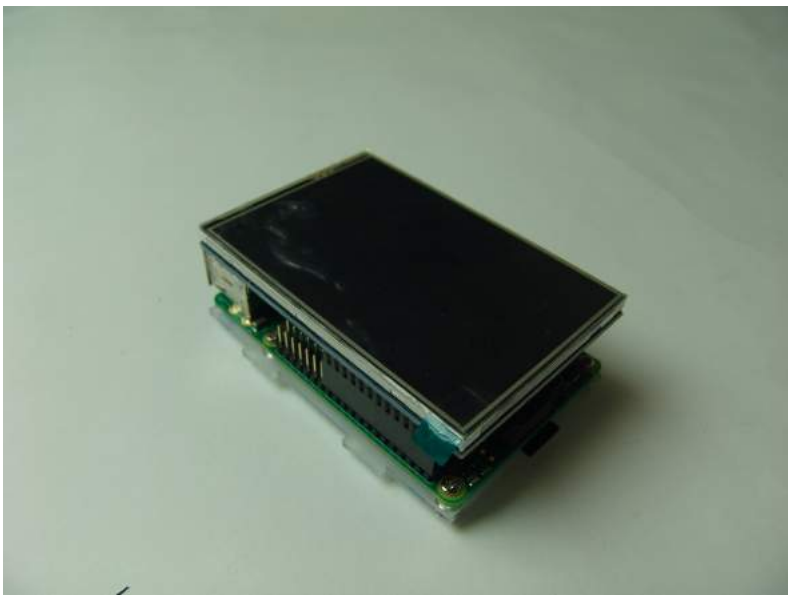
O dispositivo desenvolvido para funcionar em parceria com a FEB Racing, lê os sensores do carro, armazena os dados coletados e os mostra na tela sensível ao toque.

4.1 Hardware do Dispositivo

Os sensores presentes no carro são quatro sensores de temperatura por infra vermelho MLX90614 para os discos de freio, três acelerômetros giroscópios MPU6050, quatro sensores detectores de campo magnético por efeito hall KY003 para cálculo de velocidade das rodas e quatro potenciômetros para detectar o curso da suspensão.

A leitura dos sensores é feita por um *Arduino* Mega 2560 e para armazenar as informações e apresentá-las na tela sensível ao toque um *Raspberry Pi* B+ de 512MB de Ram, como pode ser observado na Figura 13 somente com a tela e na Figura 14 já com sua capa também. A comunicação entre o *Arduino* e o *Raspberry Pi* é feita via USB, como pode-se observa-los conectados na Figura 17.

Figura 13 – *Raspberry Pi* somente com a tela sensível ao toque.



Fonte: Produzido pelo autor

Para que os quatro sensores MLX90614, que tem o mesmo endereço, pudessem ser conectados ao barramento I2C, é usado o multiplexador CD4052, que contém duas saídas para serem utilizados como SDA e SCL, que podem ser controladas pelos pinos I/O digitais do *Arduino*.

Figura 14 – *Raspberry Pi* com a tela sensível ao toque, dentro da capa.

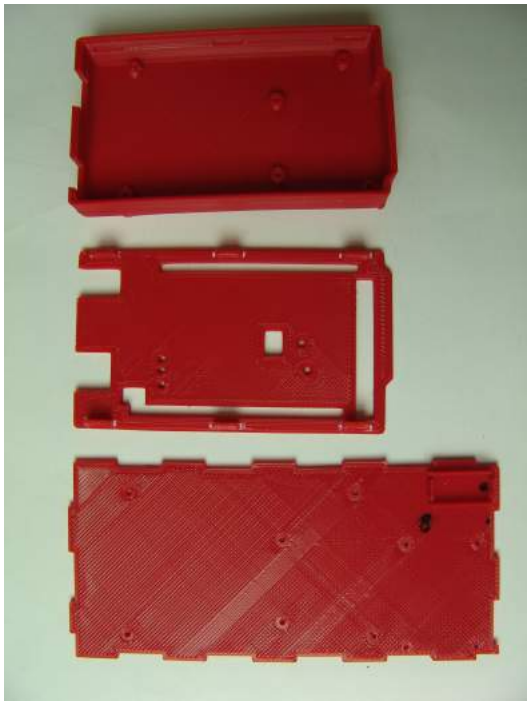
Fonte: Produzido pelo autor

O dispositivo conta com uma base desenvolvida exclusivamente para o dispositivo, como pode ser visto na Figura 15, para fixar o *Arduino* dentro de sua capa, que é uma caixa em que ele vai encaixado, juntamente com uma tampa que tem buracos para a ligação dos pinos do mesmo, além de buracos para visualizar os LEDs da placa do *Arduino*. Ela também fixa o multiplexador e o MCP2515, como pode ser observado na Figura 16.

4.2 Software do Dispositivo

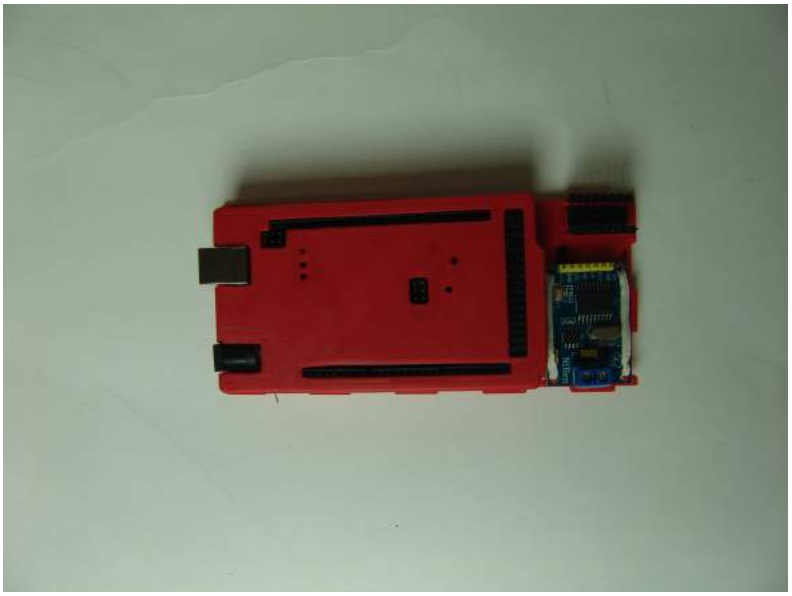
Para o desenvolvimento do software do dispositivo utilizando o hardware foi considerado o Algoritmo 1:

Figura 15 – Todas as peças da base e capa impressos em 3D.



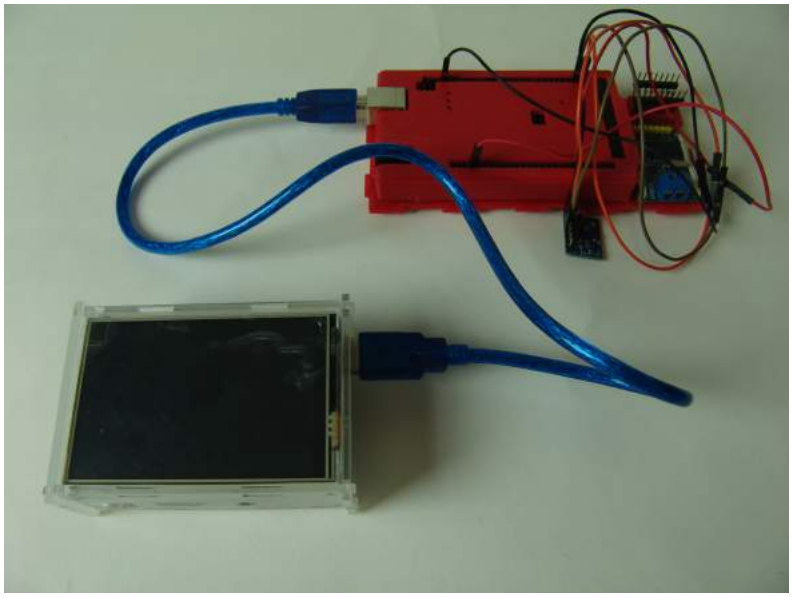
Fonte: Produzido pelo autor

Figura 16 – *Arduino* montado na capa e base, juntamente com a placa do MCP2515 e o multiplexador CD4052.



Fonte: Produzido pelo autor

Figura 17 – Conjunto *Arduino*, com o acelerômetro giroscópio MPU 6050 ligado a ele, e *Raspberry Pi* ligado ao *Arduino* pelo cabo USB.



Fonte: Produzido pelo autor

Algoritmo 1 – Algoritmo Geral do Funcionamento Desejado do Dispositivo

Entrada: Dados vindos dos Sensores D

1. ***Enquanto*** *carro ligado gerando dados dos sensores,* ***faça***
2. ***Seja*** *Dados dos sensores* D*;*
3. ***Seja*** *Arduino* A *recebendo* D*;*
4. A ***lê*** D *dos potenciômetros;*
5. A ***verifica*** *o estado do sensor de velocidade de* D *e calcula a velocidade de acordo com a contagem*
6. ***Para cada*** *grupo de entradas do multiplexador* Mi*,* ***faça***
7. ***Se*** Mi *é* $M1$*,* ***então***
8. ***Lê*** D *do primeiro e segundo acelerômetro e giroscópio e o primeiro sensor de temperatur*
9. ***Se*** Mi *é* $M2$*,* ***então***
10. ***Lê*** D *do terceiro acelerômetro e giroscópio e o segundo sensor de temperatura;*
11. ***Se*** Mi *é* $M3$*,* ***então***
12. ***Lê*** D *do terceiro sensor de temperatura;*
13. ***Se*** Mi *é* $M1$*,* ***então***
14. ***Lê*** D *do quarto sensor de temperatura;*
15.
16. ***Seja*** *Raspberry Pi* R *ligado via serial por USB com* A *executa o software com a GUI;*
17. R *em sua GUI* ***mostra*** D *recebidos via USB, separados por telas com dados de* D *diferentes;*
18. R *em sua GUI tem a opção de* ***gravar*** *os dados* D *num aquivo de log;*
19.

Para a comunicação serial entre o *Raspberry Pi* e o *Arduino*, é utilizado *thread* no *Raspberry Pi* para fazer a comunicação serial enquanto a janela é atualizada e espera por comandos do usuário como começar a gravar, parar armazenar os dados ou selecionar uma guia no GUI com informações diferentes que estão vindo do *Arduino*.

A GUI está dividida em alguns componentes, a maior parte da tela é composta pelas guias de informações, com uma barra com botões na parte de baixo da tela para os comandos. As guias são divididas em uma guia com os valores do acelerômetro, uma com os valores dos giroscópios, uma com os valores dos sensores de temperatura dos discos de freio, uma com os valores da velocidade das rodas e uma com a posição da suspensão. Também foi adicionada uma guia com um lista de portas seriais para seleção da porta serial em que o *Arduino* se encontra e alguns campos de texto para verificação se a comunicação está ocorrendo de forma correta. Os botões na parte inferior são: um para iniciar a comunicação serial, um para começar a gravar e outro para parar de gravar.

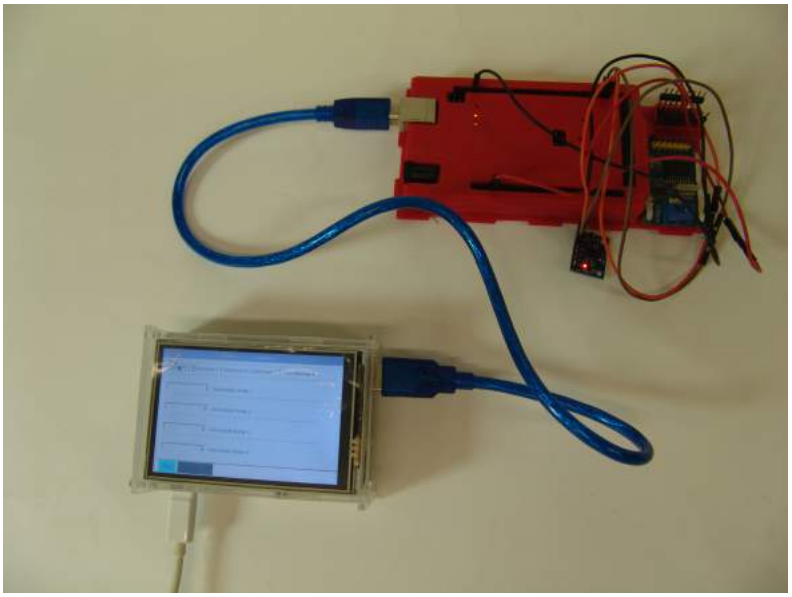
A Figura 20 mostra a GUI desenvolvida executada no *Raspberry Pi*.

Já no *Arduino*, os valores são todos lidos e passados pela serial no formato de inteiros.

5 Resultados

O dispositivo montado em sua versão final para a apresentação deste trabalho está como segue na Figura 18.

Figura 18 – Dispositivo montado, com *Arduino* e *Raspberry Pi* ligados, juntamente com o acelerômetro giroscópio MPU6050 utilizado para testes.



Fonte: Produzido pelo autor

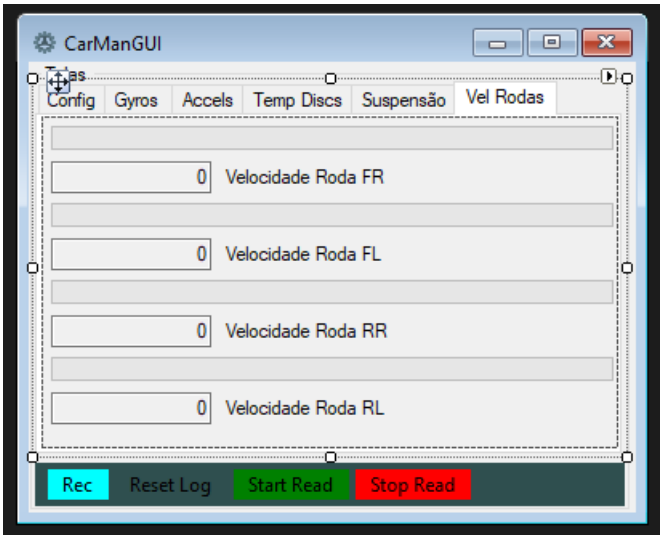
A tela sensível ao toque funciona bem e acompanha uma caneta, que será mantida em algum lugar próximo ao dispositivo, para uma manipulação mais precisa. A interface com o usuário que aparece na simulação do *Visual Studio* não tem exatamente a mesma aparência no *Raspberry Pi*, mas isso se deve ao fato do programa rodar no Raspibian e lá os componentes padrões tem sua aparência levemente diferente dos componentes padrões nativos do Windows. Pode-se observar essa diferença entre a Figura 19 e a Figura 20, mas essas diferenças não influenciam e não mudam a experiência do usuário com a interface.

O código do programa que roda no *Raspberry Pi* contém todo o funcionamento mais alto nível, lendo da serial os valores, os convertendo, gravando e mostrando na tela de acordo com a guia selecionada.

5.1 Endereços Virtuais para Contribuição com o Dispositivo

Caso haja o interesse, o dispositivo é aberto para quem quiser contribuir ou quiser acompanhar seu desenvolvimento:

Figura 19 – GUI do dispositivo vista no Windows.



Fonte: Produzido pelo autor

Figura 20 – GUI do dispositivo vista no Raspibian.

Fonte: Produzido pelo autor

https://github.com/gabrielsouza95/CarManGUI

Há também o link para a contribuição da base do dispositivo que foi desenvolvido no trabalho. Ela utilizou como base a capa do *Arduino* e está disponível no repositório do Github.

Link com a capa original utilizada de referência para a base:

https://www.thingiverse.com/thing:1145811

6 Conclusão

O dispositivo funciona assim que o carro é ligado. Após sua inicialização, deve-se selecionar a porta serial que está conectado o *Arduino*, assim o dispositivo começa a receber as informações pelo USB. A partir desse ponto, seleciona-se a tela com as informações desejadas, pode-se começar a armazenar os dados lidos.

O dispositivo será usado para a coleta de dados e a verificação em tempo real do que está acontecendo, de acordo com as informações passadas pelos sensores. Utilizando a função de armazenar as informações, é possível fazer um estudo do comportamento do carro em pista, como melhora-lo e se os cálculos feitos batem com o que acontece com o carro em pista.

6.1 Futuro do Dispositivo

O dispositivo mostrado está apenas no início de seu desenvolvimento, os planos para futura implementação no mesmo são:

- Testar o dispositivo no carro da FEB Racing;
- A leitura das informações advindas da ECU, feita através da placa MCP2515 que converse o sinal CAN, que é o protocolo mais comumente utilizado para comunicação entre os módulos dos veículos;
- Finalizar a caixa para a proteção dos componentes, que foi feita somente a base que prende a capa, a placa MCP2515 e o CI CD4052;
- Utilizar o barramento I2C do *Raspberry Pi* para a leitura dos sensores que utilizam o protocolo, para focar o uso do *Arduino* para a leitura da CAN;
- A aquisição de um *Raspberry Pi* 4 para uma maior possibilidade de desenvolvimento para fazer a interface, até mesmo utilizando aplicações web para uma melhor portabilidade;
- Implementar a comunicação sem fio no dispositivo, para o envio de dados para uma base remota, para então trabalhar com telemetria propriamente dita;
- Desenvolver uma interface utilizando componentes customizados, ao invés dos componentes padrões do sistema, para uma melhor experiência do usuário.

Referências

BEREISA, J. Applications of microcomputers in automotive electronics. *IEEE Transactions on Industrial Electronics*, IEEE, n. 2, p. 87–96, 1983.

CD4051BC • CD4052BC • CD4053BC Single 8-Channel Analog Multiplexer/Demultiplexer • Dual 4-Channel Analog Multiplexer/Demultiplexer • Triple 2-Channel Analog Multiplexer/Demultiplexer. 2002. Disponível em: <<http://dalincom.ru/datasheet/CD4052.pdf>>. Acesso em 27 de Outubro de 2019.

FAQS. 2019. Disponível em: <<https://www.raspberrypi.org/documentation/faqs/#introduction>>. Acesso em 27 de Outubro de 2019.

INTRODUCTION. 2019. Disponível em: <<https://www.arduino.cc/en/Guide/Introduction>>. Acesso em 27 de Outubro de 2019.

O que é um thread? 2004. Disponível em: <<http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/threads/threads1.html>>. Acesso em 30 de Outubro de 2019.

PRASAD, K. V.; BROY, M.; KRUEGER, I. Scanning advances in aerospace & automobile software technology. *Proceedings of the IEEE*, IEEE, v. 98, n. 4, p. 510–514, 2010.

SILVA, I. S. Lima e; KASCHNY, J. R. de A. et al. Aplicações do protocolo i2c em sistemas microcontrolados. In: *VII CONNEPI-Congresso Norte Nordeste de Pesquisa e Inovação*. [S.l.: s.n.], 2012.

WALDO, J. Embedded computing and formula one racing. *IEEE Pervasive Computing*, IEEE, v. 4, n. 3, p. 18–21, 2005.