Universidade do Minho
Escola de Engenharia

Mariana Lage da Costa

Sistema de Aquisição de Dados de *Suspension Travel*

junho de 2022

Universidade do Minho
Escola de Engenharia

Mariana Lage da Costa

Sistema de Aquisição de Dados de *Suspension Travel*

Dissertação de Mestrado
Mestrado Integrado em Engenharia Mecânica

Trabalho efetuado sob a orientação do
Professor Doutor José Mendes Machado

junho de 2022

AGRADECIMENTOS

Gostaria de agradecer a todos os que fizeram parte da realização deste projeto, que contribuíram positivamente de uma forma ou de outra. Sem toda esta ajuda, a realização deste trabalho não teria sido possível.

A toda a equipa da The Racing Factory,

Por todo o apoio que me deram, todos os ensinamentos e meses de formação, cresci imenso como engenheira e como pessoa ao trabalhar com pessoas tão profissionais.

Ao Eng. Adriano Sousa,

Por ter estado desde o primeiro dia no projeto comigo até ao fim, por me ter dado o suporte e as ferramentas para que tudo corresse como esperado. Por me ter desafiado e ensinado muito, é um engenheiro como poucos e um exemplo para mim.

Ao Prof. Dr. José Machado, o meu orientador,

Por ter estado sempre predisposto a ajudar-me e reunir comigo sempre que foi necessário, mesmo quando estava mais sobrecarregado, pelo suporte que deu a este projeto desde o início e pelo conhecimento partilhado.

À minha família,

O meu maior suporte durante esta caminhada e todas as caminhadas da minha vida. Foram a minha força, estiveram do meu lado a cada passo dado, a cada imprevisto. Passaram horas comigo a criar o sistema na oficina e outro par de horas a analisar comigo a dissertação. Palavras não chegam por toda a gratidão que tenho por vós, Gui, Pai e Mãe. A vós dedico a minha tese.

Ao meu amigo Jorge Pinto, aluno de engenharia eletrónica industrial e computadores desta mui nobre academia,

Por todas as horas que aturou as minhas perguntas, que reuniu comigo ora por vídeo chamada, ora manhãs inteiras num espaço fechado sempre a trabalhar. Foste um apoio gigante neste projeto, sem ti não teria cumprido com os meus objetivos.

Aos meus amigos, em especial Rita, Inês, Sofia, Pedrocas, Néo, Didi, Marta, Cate, Pina, Taborda, Xico, Bárbara e Cássia,

Por terem sempre puxado por mim, por estarem lá sem pensar duas vezes, por durante estes 5 anos de curso e na fase seguinte da minha vida pós curso terem sido a minha base e quem nunca me falhou. Cresci muito ao vosso lado, esta tese também é para vós e dedicada a vós.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

RESUMO

A aquisição e análise de dados em competição automóvel revolucionou a forma de condução e otimização dos *setups* dos carros e do *Motorsport,* dado que a informação é das ferramentas mais valiosas. [1] Esta fornece aos engenheiros um registo de parâmetros que são importantes para a fiabilidade e desempenho do veículo e dos seus sistemas. [2-3]

Na competição automóvel, em particular nos Ralis, a transmissão de dados via rádio e/ou telemetria é interdita [4], o que requer que o engenheiro faça, após a ida do carro à assistência, um *download* dos dados do Sistema de Aquisição de Dados (DAS), de forma a analisá-los mais tarde.

Nas mais conceituadas categorias do *Motorsport*, tais como Fórmula 1, WRC ou Moto GP, são utilizados *Data Acquisition Systems* para o *Suspension Travel* [5], sistemas esses que possibilitam o monotorização das suspensões dos carros. Dado o interesse na área de controlo e análise de dados, em conjunto com a área da competição automóvel, foi analisada e, posteriormente, realizada a possibilidade de desenvolvimento deste tema em cenários de teste.

Este projeto consiste numa análise mais aprofundada do sistema de amortecimento e suspensão de um Škoda Fabia Rally 2 Evo e ao seu comportamento, de forma a melhorar a *performance* dos veículos motorizados no *Motorsport*, através da recolha e análise de dados desse subsistema.

PALAVRAS-CHAVE: Competição Automóvel, *Motorsport*, Sistema DAS, Škoda Fabia Rally 2 Evo, *Suspension Travel System*.

ABSTRACT

Data acquisition and analysis in racing has revolutionized the way cars are driven and optimized for setups, and in Motorsport information is one of the most valuable tools[1]. It provides engineers with a method of recording parameters that are vital to the reliability and performance of the vehicle and its systems[2-3].

In motor racing, specifically in Rallys, data transmission via radio and/or telemetry is forbidden [4], which requires the engineer to download the data from the Data Acquisition System (DAS) after the car goes to the service park, in order to analyse it later.

In the most prestigious motorsport categories, such as Formula 1, WRC or Moto GP, Data Acquisition Systems are used for Suspension Travel Systems [5] that enable the monitoring of the cars' suspensions. Given the interest in the area of control and data analysis, aligned with the area of car competition, the possibility of developing this theme in test scenarios was analysed and then carried out.

This project consists of further analysis of the damping and suspension system of a Škoda Fabia Rally 2 Evo, and its behaviour, in order to improve motor vehicle performance in Motorsport by collecting and analysing data from that subsystem.

KEYWORDS: Automotive Competition, DAS System, Motorsport, Škoda Fabia Rally 2 Evo, Suspension Travel System.

ÍNDICE

Agradecimentos iii
Resumo vi
Abstract viii
Índice ix
Índice de Figuras xii
Índice de Tabelas xv
Lista de Abreviaturas, Siglas e Acrónimos xvi
1. Introdução 1
1.1 Enquadramento e Motivação 1
1.2 A Empresa 2
1.3 Objetivos 6
1.4 Organização da Dissertação 7
1.4.1 Planificação Real 8
A reter deste Capítulo 11
2. Revisão Bibliográfica 12
2.1 Sistema de Suspensão e Amortecimento 12
2.1.1 Principais Componentes 13
2.1.2 Funcionamento 14
2.2 Sistemas DAS 16
2.2.1 Sistemas DAS 17
2.2.2 Elementos Constituintes 17
2.2.3 Soluções Tecnológicas Existentes/Possíveis 18
2.2.4 *Software* utilizado 19
2.2.5 Škoda Fabia Rally 2 EVO 21
2.3 Tipos de Troço 22
2.3.1 Troço de Terra 22
2.3.2 Troço de Asfalto 23

2.4 Estudo das Variáveis 23
2.4.1 Posicionamento do Sensor e o seu Mecanismo de Barras 23
2.4.2 Gestão da Compressão e *Rebound* 25
2.5 *Suspension Travel System* 27
2.5.1 Suspensões Utilizadas - Škoda 27
2.5.2 Amortecedores Utilizados - Škoda 28
2.5.3 Modelação 3D do Sistema 29
A reter deste Capítulo 32
3. Sistema de Recolha de Dados 33
3.1 Componentes Selecionados 33
3.1.1 Sensores de Posição Angular 33
3.1.2 Placa Arduino Uno Rev 3 37
3.1.3 Placa PCB e Peças Complementares 39
3.1.4 *Software Microsoft Excel - Škoda Datasheet* 42
3.2 Mapa Geral de Tráfego dos Dados 44
3.3 Ligação Entre Componentes 44
3.3.1 Explicação Geral da Interligação dos Componentes 44
3.3.2 Carro-Sensor 45
3.3.3 Sensor-Placa 47
3.3.4 Arduino IDE - Arduino Placa 48
3.3.5 Arduino Placa-Excel 48
3.4 Código em Arduino 49
A reter deste Capítulo 61
4. Testes Práticos e Análise de Resultados 62
4.1 Montagem do Sistema Físico 62
4.1.1 Opções de Colocação dos Sensores 62
4.1.2 Ligações Elétricas 63
4.1.3 Localização Controlo do Sistema e Acionamento 65
4.2 Curva Curso-Diferença de Potencial 65

4.3 Final *DAS Dashboard* 68
4.4 Análise dos Resultados dos Dados 71
4.5 Observação da Robustez do Sistema 72
4.6 Análise SWOT 72
A reter deste Capítulo 74
5. Considerações Finais 75
5.1 Conclusões 75
5.2 Trabalhos Futuros 77
Referências 78
Apêndice 1 – Código Principal Arduino 82
Anexo 1 – *Info Book Sensor Type B* 85
Anexo 2 – Código *ReadWriteSDCard* 86
Anexo 3 – Código base efetuado em simulação *online TinkerCAD* 87

ÍNDICE DE FIGURAS

Figura 1 - Logótipo da empresa The Racing Factory 3
Figura 2 - Peugeot 208 Rally 4 e respetivas informações relevantes. 4
Figura 3 - Škoda Fabia R5/Rally2 EVO e respetivas informações relevantes. 4
Figura 4 - Škoda Fabia R5 e respetivas informações relevantes. 4
Figura 5 - Maverick X3 e respetivas informações relevantes. 5
Figura 6 - Peugeot 208 R2 e respetivas informações relevantes. 5
Figura 7 - Vista de frente da empresa 6
Figura 8 - Vista explodida de um sistema de suspensão e amortecimento usual 13
Figura 9 - Vista lateral em corte de um amortecedor hidráulico 13
Figura 10 - Tipologia MacPherson 14
Figura 11 – Modelo de ¼ de carro de um sistema de suspensão 15
Figura 12 - Configuração geral de um sistema de aquisição de dados 18
Figura 13 - Potenciómetros lineares LP 75-150 J e LPT 50-150 19
Figura 14 - Škoda Fabia Rally2 EVO do Vice Campeão Nacional, Armindo Araújo, onde é implementado o sistema. 22
Figura 15 - Localização do sensor e dos seus braços no charriot e num braço de suspensão. 24
Figura 16 - Mecanismo biela-manivela. 25
Figura 17 - Representação em 3D do catálogo online exclusivo e interdito da Škoda Motorsport 29
Figura 18 - Mecanismo sensor-braços em SolidWorks, onde é possível observar o movimento do braço menor ligado diretamente ao sensor. 30
Figura 19 - Assembly Final do sistema implementado no Škoda, em SolidWorks. 31
Figura 20 - Sensor Tipo A posicionado no carro 34
Figura 21 - Sensores Hella Tipo A e Tipo B. 34
Figura 22 - Especificações técnicas do sensor Tipo A. 35
Figura 23 - Representação dos ângulos do sensor onde não se consegue medir diferença de potencial. 36
Figura 24 - Elementos que constituem o mecanismo do sensor 37
Figura 25 - Especificações Placa Arduino Uno Rev 3 38

Figura 26 - Placa Arduino Uno Rev 3 38
Figura 27 - PCB com todos os componentes. 39
Figura 28 - Esquema das ligações entre componentes no software Autodesk Eagle 40
Figura 29 - Posicionamento final das peças e de todas as ligações para impressão da PCB. .. 41
Figura 30 - Localização dos componentes na PCB por cores. 42
Figura 31 - Back-End visível onde se encontram informações referentes aos gráficos secundários e ao sensor em estudo. 43
Figura 32 - Front-End do Excel para visualização dos gráficos finais. 43
Figura 33 - Mapa geral do tráfego de dados 44
Figura 34 - Suporte adicionado ao sensor que liga o mesmo ao charriot. 45
Figura 35 - Suporte com cavidade cilíndrica que suporta o braço maior do mecanismo no braço de suspensão. 46
Figura 36 - Ilustração da representação final do mecanismo no STS. 46
Figura 37 - Ligações de 3 pinos do sensor. Extremidade que liga ao sensor (esquerda); Extremidade que liga à PCB (direita); Pino 1 – Ground, Pino 4 – Output Signal, Pino 6 – 5V. 47
Figura 38 - Shell com os três cabos ligada à placa. 47
Figura 39 - Esquema final do projeto simulado em TinkerCAD. 48
Figura 40 - Ferramenta do Excel que transporta os dados do ficheiro .txt para o ficheiro criado para back-end em Excel. 49
Figura 41 - Documento em formato .txt criado pelo código em Arduino IDE com os dados retirados dos sensores. 49
Figura 42 - Passagem do código do botão de hexadecimal para decimal. 59
Figura 43 - Primeira opção de colocação do sensor entre as setas azuis. 62
Figura 44 - Segunda opção de colocação do sensor entre as setas vermelhas. 63
Figura 45 - Esquerda: cablagem ligar através de orifício do sensor de óleo dos travões; Direita: Cablagem ligar pela zona de cablagem principal onde ficam conectadas as principais ligações elétricas. 64
Figura 46 - Cablagem do sensor. 64
Figura 47 - Amortecedor com curso todo estendido. 66
Figura 48 - Amortecedor com curso quase todo comprimido. 66

Figura 49 - Recolha dos dados das diferenças de potencial a cada 10 mm de curso, com a ajuda de um multímetro. 67
Figura 50 - Dados retirados após iterações e respetivo gráfico padrão. 67
Figura 51 - Seleção de linhas para obter somente os valores do sensor A0..... 68
Figura 52 - Regra de 3 simples aplicada a todos os valores do sensor A0. 69
Figura 53 - Valores finais das alturas de curso, através do gráfico padrão achado nas iterações. 69
Figura 54 - Gráfico final que apresenta o curso do amortecedor num determinado espaço de tempo. 70
Figura 55 - Gráfico de dispersão por pontos do comportamento do curso do amortecedor num determinado espaço de tempo. 70

ÍNDICE DE TABELAS

Tabela 1 - Planificação atualizada real em formato Diagrama de Gantt 10
Tabela 2 - Listagem de todas as molas de suspensão disponíveis no Manual online (exclusivo para equipas parceiras de Rali) da Škoda Motorsport 27
Tabela 3 - Os amortecedores utilizados para o Kit de Terra como para o Kit de Asfalto 29
Tabela 4 - Designações dos componentes por cores, de acordo com a imagem ao lado. 42
Tabela 5 - Análise SWOT do projeto..... 73

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

ASIC *Application Specific Integrated Circuit*

DAS *Data Acquisition Systems*

DPA *Driver Performance Analysis*

DVP *Determining Vehicle Parameters*

HSC *High Speed Compression*

HSR *High Speed Rebound*

IDE *Integrated Development Environment*

LSC *Low Speed Compression*

LSR *Low Speed Rebound*

PCB *Printed Circuit Board*

PDS *Progressive Damper System*

PEC *Prova Especial de Classificação*

RL *Running Logs*

RS *Reliability and Safety*

SPI *Serial Peripheral Interface*

STS *Suspension Travel System*

SWOT *Strengths, Weaknesses, Opportunities, Threats*

TRF *The Racing Factory*

VD *Vehicle Development*

VPA *Vehicle Performance Analysis*

DEI *Departamento de Eletrónica Industrial*

PCBA *Printed Circuit Board Assembly*

PCB *Printed Circuit Board*

1. INTRODUÇÃO

Este projeto surge no âmbito do Mestrado Integrado em Engenharia Mecânica da Universidade do Minho e está inserido na aquisição e tratamento de dados para competição automóvel. Esta dissertação foi realizada em âmbito empresarial na *empresa The Racing Factory* e consiste numa análise mais aprofundada do sistema de amortecimento e suspensão de um Škoda Fabia Rally 2 Evo. Neste capítulo descreve-se o enquadramento, a motivação, a empresa, o objetivo e a organização desta dissertação.

1.1 Enquadramento e Motivação

Como já dizia *Jörge Segers*, engenheiro de *Motorsport* conceituado mundialmente,

"(…) Uma forma comprovada dos atletas terem sucesso em qualquer modalidade desportiva é registarem o seu desempenho, analisarem os acontecimentos e retirarem conclusões do fator que influencia esse mesmo desempenho. (…) Medem algo, aprendem com essa informação retirada e tentam usá-la a seu favor numa próxima vez. (…)", a aquisição e análise de dados em competição automóvel revolucionou a forma de condução e otimização dos *setups* dos carros. Atualmente, o acesso a esta tecnologia de ponta é cada vez mais eficiente, sendo necessária para todas as equipas envolvidas no desporto automóvel. Esta fornece aos engenheiros um de registo de parâmetros que são vitais para o desempenho do veículo e dos sistemas que o compõe.

Na competição automóvel, mais propriamente em Rally, a transmissão de dados via rádio e/ou telemetria é interdita, o que requer que o engenheiro execute, após a ida do carro à assistência ou até mesmo num teste, um *download* dos dados registados e guardados no sistema de aquisição de dados (DAS) existente no veículo, de forma a analisá-los mais tarde num computador e interpretar os mesmos com o piloto.

Nas mais conceituadas competições no *Motorsport*, tais como Fórmula 1, World Rally Championship (WRC) ou Moto GP, são utilizados Sistemas de Aquisição de Dados de *Suspension Travel*, sistemas esses que possibilitam a análise e o controlo em tempo real do sistema de amortecimento e suspensão dos carros. No entanto, esse género de sistemas não está acessível ao público devido às políticas de privacidade e contratos de exclusividade

criados com as marcas dos automóveis de competição, além da dificuldade em gerar novos canais nas centralinas de origem. Dado o interesse na área de controlo e análise de dados, em conjunto com a área de *race engineering*, em especial esta última por ser bastante vasta onde há espaço para inovar à medida que a tecnologia evolui, foi analisada a possibilidade de desenvolvimento deste tema, bem como a monitorização do mesmo em cenários de teste de Rally.

Quantos mais dados for possível obter acerca de um automóvel, melhor será a leitura/ compreensão que se obtém acerca do mesmo, como é que se comporta nas mais variadas situações. Como estes são veículos que estão a ser constantemente colocados em situações extremas, é crucial avaliar todos os parâmetros que poderão comprometer tanto o carro como o seu desempenho. Um destes parâmetros é o *Suspension Travel System* (STS). Este influencia diretamente o contacto entre o conjunto chassi-roda-piso, mais concretamente o comportamento, segurança, estabilidade e eficiência do carro. Todos estes parâmetros dependem do ajuste do curso do amortecedor, sendo que este contribui para que haja um aumento ou diminuição da carga da mola no amortecedor. Existem principalmente três problemas causados pela falta ou mau ajuste do curso do amortecedor, sendo esses: A ausência de estabilidade nas curvas, pois o sistema não está a absorver e libertar a energia dos impactos corretamente; A leitura incorreta do solo, uma vez que a suspensão ajuda as rodas a efetuarem a leitura correta do solo e a manter sempre a aderência, o que se não acontecer promoverá saída de estrada ou falta de controlo do carro, podendo a equipa perder segundos valiosos ou até desistir; E o desgaste desnecessário de outros componentes envolvidos no sistema, tais como os discos e calços, e desalinhamento das rodas. A motivação para este projeto baseia-se assim, numa variável tão peculiar e simples que possui um tamanho impacto neste desporto. O objetivo é adquirir dados num curto intervalo de tempo as alturas de curso do amortecedor, o que permitirá, a partir destes valores, obter informações acerca de outros parâmetros. Sabendo os dados das alturas de curso e as informações de diferença de potencial adquiridas por um sensor de movimento angular ligado a um sistema biela-manivela, será possível obter gráficos de posição e curso dos amortecedores, a fim de analisar a sua *performance* e qual será o melhor ajuste para que, assim, se promovam melhores condições às equipas de Rali nos troços de classificação.

1.2 A Empresa

A *The Racing Factory* é uma empresa nascida no ano de 2018, integralmente dedicada ao desporto automóvel e está sedeada em S. Paio de Oleiros, no concelho de Santa Maria da Feira, em Portugal. O seu logótipo está representado na Figura 1.

Figura 1 - Logótipo da empresa The Racing Factory

Esta proporciona vários serviços na área de *Motorsport*, tendo como principais os seguintes:

- Preparação e Manutenção Mecânica;
- Operações Logísticas;
- Engenharia e Desenvolvimento;
- Aluguer de Carros de Competição;
- Produtos *Motorsport*;
- Serviço de Comunicação e *Design*.

Como está enumerado acima, a empresa disponibiliza veículos de competição para aluguer, dentro dos quais os seguintes apresentados nas figuras seguintes, com as suas respetivas informações (Figura 2, Figura 3, Figura 4, Figura 5, Figura 6).

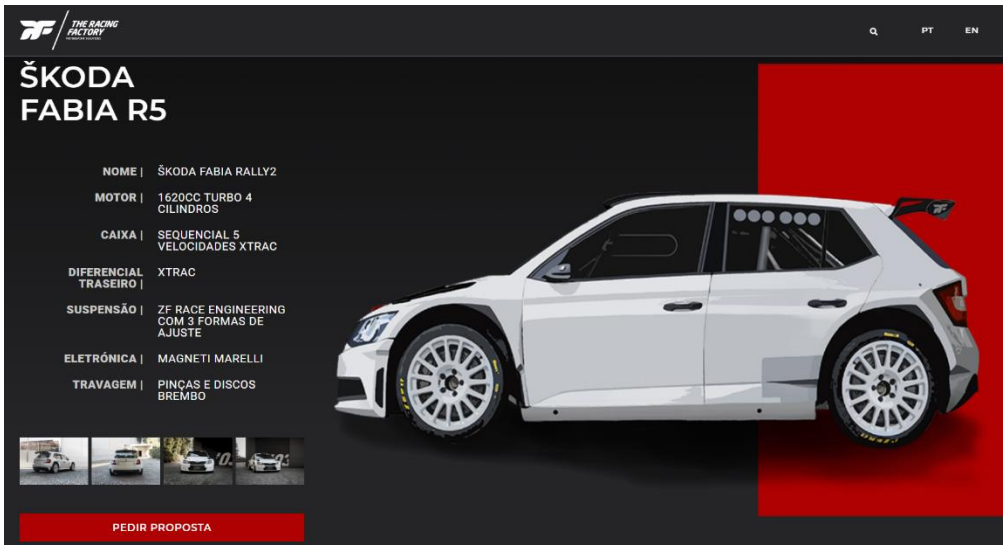


Figura 4 - Škoda Fabia R5 e respetivas informações relevantes.

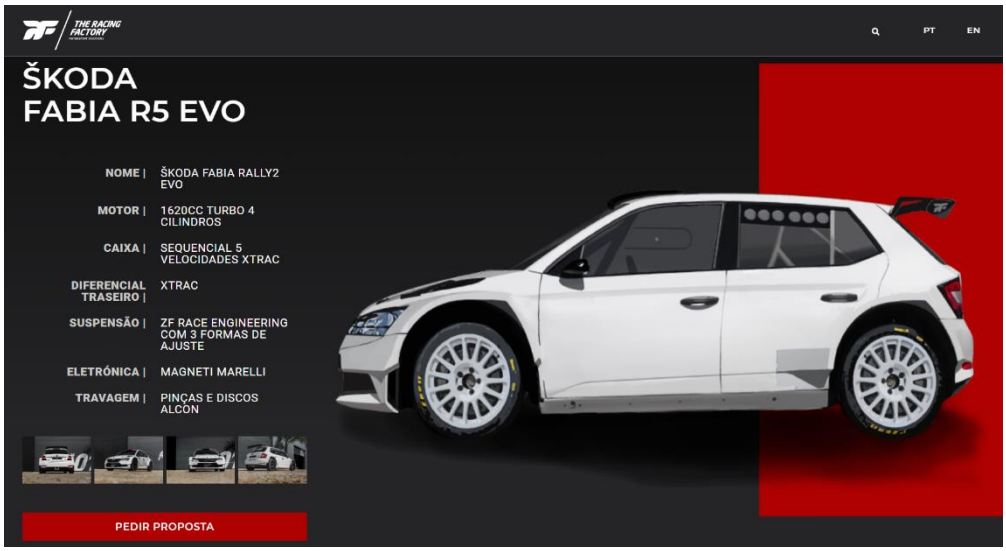


Figura 3 - Škoda Fabia R5/Rally2 EVO e respetivas informações relevantes.

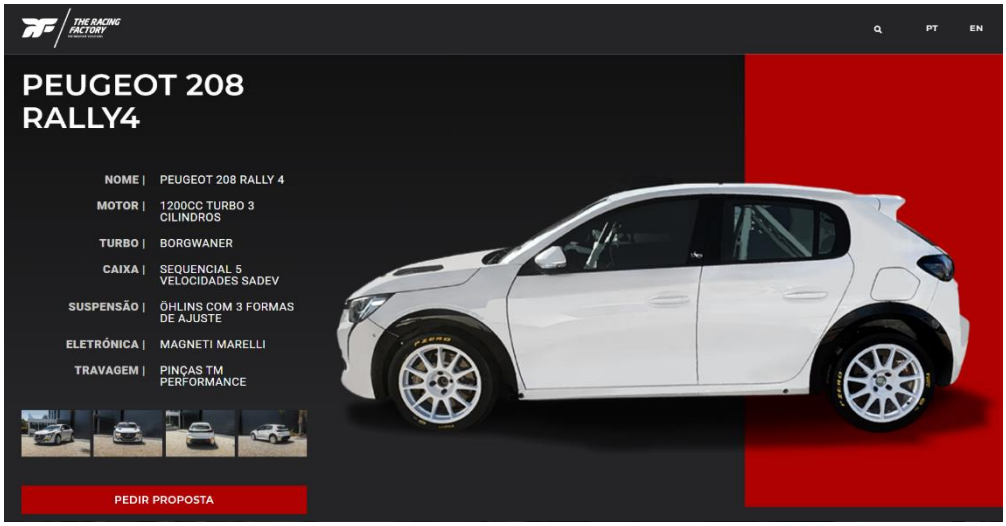


Figura 2 - Peugeot 208 Rally 4 e respetivas informações relevantes.

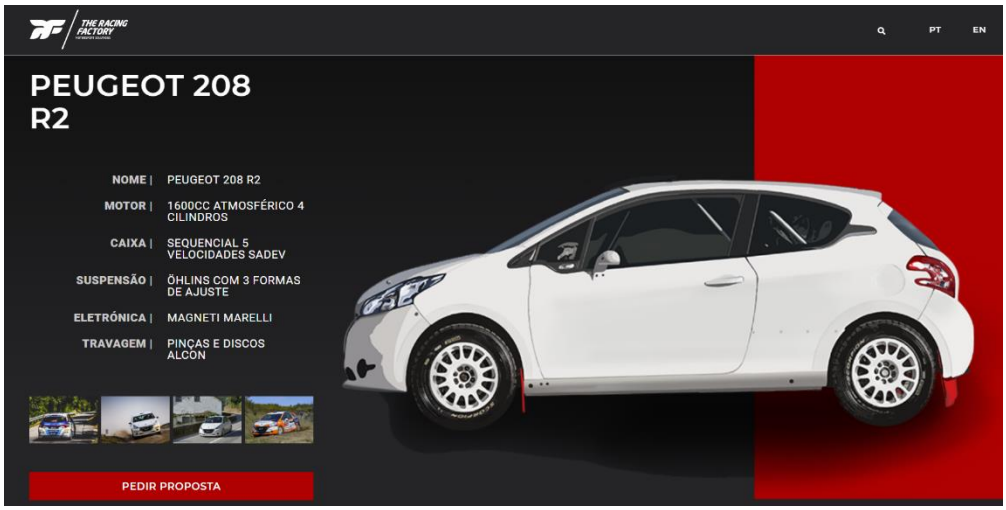


Figura 6 - Peugeot 208 R2 e respetivas informações relevantes.

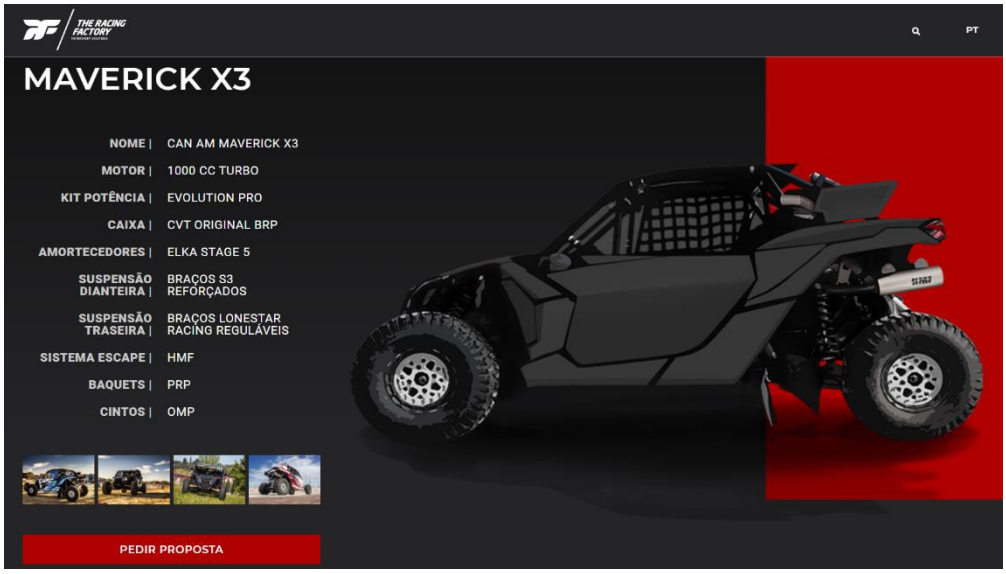


Figura 5 - Maverick X3 e respetivas informações relevantes.

Para a *The Racing Factory*, os principais objetivos passam por proporcionar aos clientes e pilotos um serviço de excelência, completo e adequado às exigências do desporto motorizado dos dias de hoje e serem uma referência nesta área que tanto tem para explorar. Com uma estrutura de recursos humanos forte e coesa e em constante formação e desenvolvimento, possui um grupo de trabalho de quinze elementos que, todos os dias, se empenha para executar e atingir os objetivos dos clientes e estratégias de crescimento delineadas.

Cumprindo com os objetivos planeados, marcaram desde logo presença a nível internacional na *Peugeot Rally Cup Iberica*, e no Campeonato *FIA ERC* com o desenvolvimento

da *FPAK Portugal ERC Team*, equipa representante de Portugal nesse campeonato, conquistando a 3ª posição final entre as seleções europeias. Num curto período, conquistaram vários títulos, entre eles o Campeonato de Portugal de Ralis (2020) e Campeonato da Madeira de Ralis de Coral (2020).

Com os olhos postos numa estratégia de crescimento sólido, sustentado e com espírito vencedor, a *The Racing Factory* pretende estar presente nas mais diversas modalidades como, ralis, todo-o-terreno, velocidade, mas também estar preparada e seguir ativamente as mudanças e desafios que o futuro do desporto automóvel trará.

A *The Racing Factory* está localizada em Santa Maria da Feira, na Rua de Vila Boa, nº165, 4535-457, São Paio de Oleiros. De seguida é possível observar a entrada principal da empresa, ilustrada na Figura 7[6].

Figura 7 - Vista de frente da empresa

1.3 Objetivos

Esta dissertação tem como objetivo global desenvolver um sistema robusto que recolha dados precisos do sistema de amortecimento e suspensão para veículos de competição, sendo que este sistema terá de ser adaptado para ser usado num *Škoda Fabia Rally 2 EVO*. Numa primeira fase, é necessário estudar e compreender o estado da arte referente aos sistemas de aquisição de dados para *suspension travel* existentes no mercado, de forma a obter um ponto de partida onde se pode analisar possíveis erros e melhorias. Para que este sistema seja funcional e duradouro, é importante fazer um estudo de necessidades e entraves ao projeto, de forma a obter o resultado esperado. De seguida, é imperativo o desenvolvimento e uma pesquisa aprofundada do sistema de suspensão e de amortecimento

num contexto prático, de maneira a possibilitar uma melhor compreensão de todos os componentes e realização das modelações de todos os constituintes da STS. É necessário também analisar quais os componentes (microcontroladores, sensores, atuadores, placa, etc.) que são mais indicados e como devem ser colocados no chassi do carro sem comprometer o seu movimento e *performance*. Numa fase posterior de recolha e tratamento de dados, é essencial considerar qual será o *software DAS* que realizará uma análise adequada, equacionar as variáveis e expor os resultados para uma compreensão mais intuitiva dos mesmos.

Ao nível dos resultados, é expectável um sistema desenvolvido robusto e funcional, que consiga assegurar a integridade do veículo e dos seus componentes, para uma posterior aplicação em automóveis de competição, de forma a que os dados retirados do STS possam ser analisados e apoiar nos *Setups*, além da busca de melhores tempos finais em prova.

1.4 Organização da Dissertação

De forma a alcançar os objetivos propostos desde o início, A dissertação está organizada de uma forma simples, de acordo com a ordem de realização do projeto de tese.

No segundo capítulo, referente à Revisão Bibliográfica, é feita uma introdução ao documento, abordando o enquadramento e motivação, os objetivos para o projeto e a empresa onde está a ser realizado o estágio curricular. De seguida, é realizada uma revisão bibliográfica, mais concretamente um estado da arte, dos conteúdos necessários para a compreensão deste trabalho, dentro dos quais o sistema de suspensão e amortecimento do carro em estudo, o sistema DAS, o carro de competição que terá o sistema implementado, uma caraterização dos tipos de troço existentes em rali, as variáveis que influenciarão o projeto o STS mais aprofundadamente e, por fim, as modelações de estudo inicial do mesmo. Este último permite compreender ao pormenor no que consiste este subsistema e executar um levantamento e estudo de todos os elementos envolventes.

No capítulo Sistema de Recolha de Dados estão inseridas todas as componentes que constituem o sistema de recolha e tratamento de dados, assim como um mapa de tráfego dos dados para uma melhor compreensão do sistema numa perspetiva global. Será explicado todo

o processo de interligação de componentes eletrónicos, o código usado e a interface entre os dados e o programa que os vai tratar.

No capítulo 4: Testes Práticos e Análise de Resultados, está presente a explicação da montagem do sistema físico, o respetivo *output* que se pretende ter, e também são descritos os testes práticos que foram realizados ao sistema, além da observação da robustez do projeto. De forma a complementar todo o trabalho realizado, também existe uma análise dos resultados adquiridos pelos sensores e as iterações realizadas, que remetem para uma avaliação final do projeto e possíveis sugestões de melhoria para uma fase subsequente. Neste capítulo também está presente uma análise SWOT que permitirá desenvolver, de uma perspetiva mais global, um ponto de situação final do trabalho.

No capítulo das Considerações Finais, são tecidas conclusões e sugestões de trabalhos futuros.

1.4.1 Planificação Real

Devido ao vírus covid-19 e à forma como este limitou o quotidiano de todos, foi necessário reorganizar a estruturação e fases de trabalho da dissertação. De maneira a conseguir obter o conhecimento necessário tanto no desporto motorizado em questão como nos carros de competição, foi proporcionado um estágio curricular na *The Racing Factory*, no qual foi permitido trabalhar ativamente desempenhando várias funções, o que a longo prazo culminasse numa entrega mais tardia da dissertação. No entanto, sem todo este conhecimento e sem o apoio de toda a equipa, seria muito mais complicado obter bases fundamentadas para desenvolver este projeto de dissertação.

<u>Julho - Agosto</u>

- Pesquisa de sistemas existentes no mercado de aquisição de dados para *suspension travel* (estado da arte).

- Pesquisa de estado da arte referente às suspensões utilizadas na competição automóvel, mais especialmente no Campeonato Português de Ralis (CPR).

<u>Agosto - Setembro</u>

- Pesquisa de protocolos/códigos a ter em conta em contexto desportivo e estudo das suspensões num carro específico em contexto prático.

- Pesquisa de controladores e condicionadores de sinal mais adequados às exigências do sistema.

- Estudo de *softwares DAS* (funcionamento, recolha de dados, análise de dados, ajustes).

Setembro - Dezembro

- Pesquisa de qual o melhor *software DAS* para gravação e análise dos dados.

- Estudo aprofundado do sistema de suspensão existente, de forma a averiguar quais os componentes a acrescentar.

- Desenvolvimento das modelações necessárias que irão compor o sistema inicial (só suspensão atual).

- Elaboração das modelações do sistema de *suspension travel*, com os componentes extra.

- Simulação dos modelos desenvolvidos.

Novembro - Janeiro

- Realização de análise de erros e possíveis melhoramentos e fase de teste do equipamento, aplicado no veículo.

Janeiro – Março

- Realização de testes para testar robustez do sistema.

- Efetuar ligação do sistema e *software DAS* e interpretar valores obtidos.

- Finalização de ajustes ao sistema e à sua ligação ao *software* e desenvolvimento do relatório da dissertação.

Março – Junho

- Desenvolvimento/finalização do relatório da dissertação.

A planificação final foi a seguinte, disposta também num diagrama de *Gantt* na Tabela 1.

	Nº	Tarefa	Prazos											
			JUL	AGO	SET	OUT	NOV	DEZ	JAN	FEV	MAR	ABR	MAI	JUN
Fases do Projeto	T01	Pesquisa de sistemas no mercado de aquisição de dados para S.T.												
	T02	Pesquisa de estado da arte												
	T03	Pesquisa de protocolos/códigos												
	T04	Estudo das suspensões em contexto prático												
	T05	Pesquisa de controladores e condicionadores de sinal												
	T06	Estudo de softwares DAQ												
	T07	Pesquisa de qual o melhor software DAQ												
	T08	Estudo aprofundado do sistema de suspensão existente												
	T09	Desenvolvimento das modelações necessárias												
	T10	Elaboração das modelações do sistema de suspension travel												
	T11	Simulação dos modelos desenvolvidos												
	T12	Realização de análise de erros e possíveis melhoramentos												
	T13	Fase de teste do equipamento, aplicado no veículo												
	T14	Realização de provas de teste em troço de terra												
	T15	Efetuar ligações necessárias entre os sistemas												
	T16	Finalização de ajustes ao sistema e à sua ligação ao software												
	T17	Desenvolvimento do relatório da dissertação												
	T18	Desenvolvimento/Finalização do relatório da dissertação												

Tabela 1 - Planificação atualizada real em formato Diagrama de *Gantt*

A RETER DESTE CAPÍTULO

Com o passar dos anos, a tecnologia tem vindo a aliar-se cada vez mais à área do *Motorsport*, e com essa aliança vem a inovação constante e a procura da eficiência e rentabilidade máximas. Não descorando a componente mecânica, que está em constante evolução, chega-se a um certo ponto em que são mais valiosos os dados impercetíveis a olho nu, sendo esta a informação que dita se os componentes mecânicos estão em conformidade ou há espaço para melhoria.

O objetivo global desta dissertação consiste em desenvolver um sistema robusto que recolha dados precisos do Sistema de Amortecimento e Suspensão (STS) de veículos de competição automóvel e, assim, melhorar de forma considerável o desempenho em provas, visto que este subsistema dos carros em questão é subvalorizado e possui variáveis ainda por explorar.

2. REVISÃO BIBLIOGRÁFICA

Neste capítulo é realizada uma revisão bibliográfica a esta dissertação, de forma a obter não só um plano de partida para o(a) autor(a), como também ajudar o(a) leitor(a) na compreensão do tema através das ferramentas e informação dispostas. Devido à elevada confidencialidade que existe ao redor deste ramo, a informação disposta para quem pretende desenvolver este género de sistemas é escassa, o que dificulta inicialmente todo o processo de recolha de informação. De forma a solucionar esse entrave, foi comprada bibliografia especializada na área e foram contactados profissionais que trabalham em *Motorsport*. É explicado, numa perspetiva geral, os sistemas de amortecimento e suspensão de um automóvel e, de forma mais aprofundada, o *Suspension Travel System* (STS), enunciando quais as suspensões e amortecedores que são utilizados no carro. É efetuado um estudo das variáveis do sistema, de maneira a que seja possível observar o seu comportamento e interpretar os resultados obtidos a partir daí. Os Sistemas DAS são outra componente fulcral neste projeto, pelo que neste capítulo serão explorados os vários tipos de sistemas que existem no mercado. Também é efetuada uma enumeração e explicação dos tipos de troço existentes nos Ralis, além de uma modelação em 3D de todo o sistema para uma melhor análise e compreensão nos capítulos seguintes.

2.1 Sistema de Suspensão e Amortecimento

Atualmente, é possível denotar uma melhor compreensão da ciência do amortecimento, como resultado da investigação e desenvolvimento que levou à conceção desta componente crucial da suspensão moderna. O objetivo básico do amortecedor numa suspensão é converter a energia de movimento, ou energia cinética gerada pela massa em movimento de um carro (armazenada na mola comprimida), em calor ou energia térmica. A energia térmica é então dissipada através do corpo metálico do amortecedor que atua como um permutador de calor.

A compressão da suspensão é controlada por forças amortecedoras, abrandando o curso da suspensão em compressão, impedindo as molas da absorção de mais energia cinética do que a necessária do que quando este processo não seria amortecido. O amortecedor abranda e controla o movimento da suspensão tanto no *rebound* como na compressão.

Cada aspeto do amortecimento hidráulico é sujeito a uma constante investigação e desenvolvimento, sendo melhorado e testado em laboratório e no terreno. O amortecimento hidráulico é ainda uma componente complexa no que toca à afinação da suspensão, daí ser imperativa uma compreensão básica do amortecimento, como funciona a suspensão e os efeitos do mesmo na utilização do veículo, pois a sua dinâmica será muito útil para qualquer pessoa envolvida em engenharia de carros de corrida.

2.1.1 Principais Componentes

Existem três fatores cruciais num sistema de suspensão que devem ser tomados em conta, sendo esses: a flexibilidade, proporcionada por uma mola que distorce e recupera (normalmente comprime e expande) à medida que a roda atravessa perturbações na superfície da estrada; o amortecimento, que é essencial para proteger o corpo e a roda de movimentos ressonantes; e a localização da roda ou do eixo.

Os elementos de um sistema de suspensão dividem-se geralmente em dois grupos: aqueles que controlam a função de mola (ou seja, as molas principais, amortecedores e barras anti torsão), e as que localizam o subconjunto das rodas e controlam as geometrias dos seus movimentos em condições de carga dinâmica [7].

Nas Figura 9 e Figura 8 é possível analisar uma vista lateral em corte de um amortecedor hidráulico e também uma vista explodida de um sistema de suspensão e amortecimento usual.

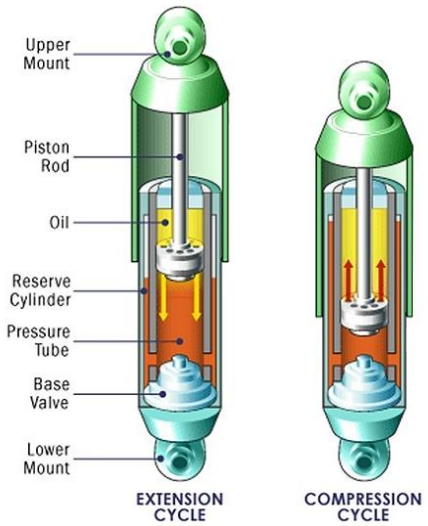


Figura 9 - Vista lateral em corte de um amortecedor hidráulico

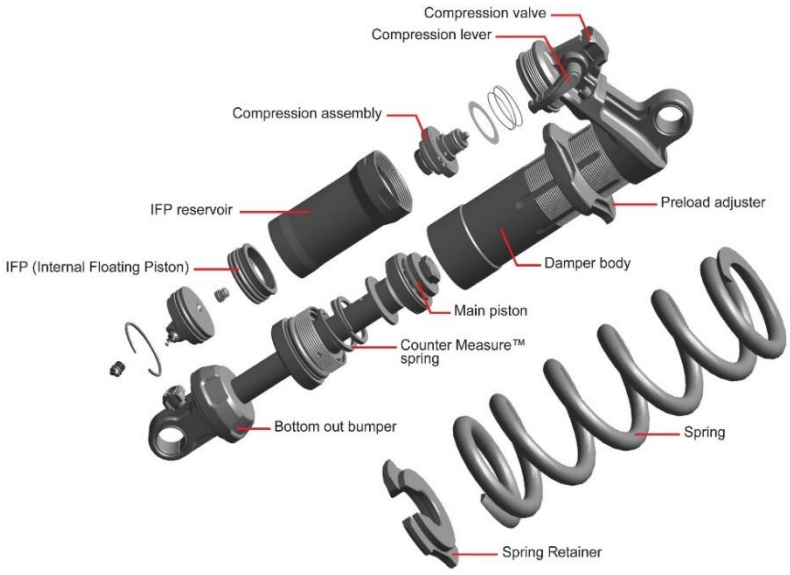


Figura 8 - Vista explodida de um sistema de suspensão e amortecimento usual

As suas localizações e tipologias no conjunto representado também são importantes pois é a partir destas que há transferência de movimento e de forças envolvidas, como podemos observar na Figura 10, onde a tipologia representada é a *MacPherson*, a estrutura de suspensão automóvel utilizada no *Škoda Fabia Rally2 EVO*.

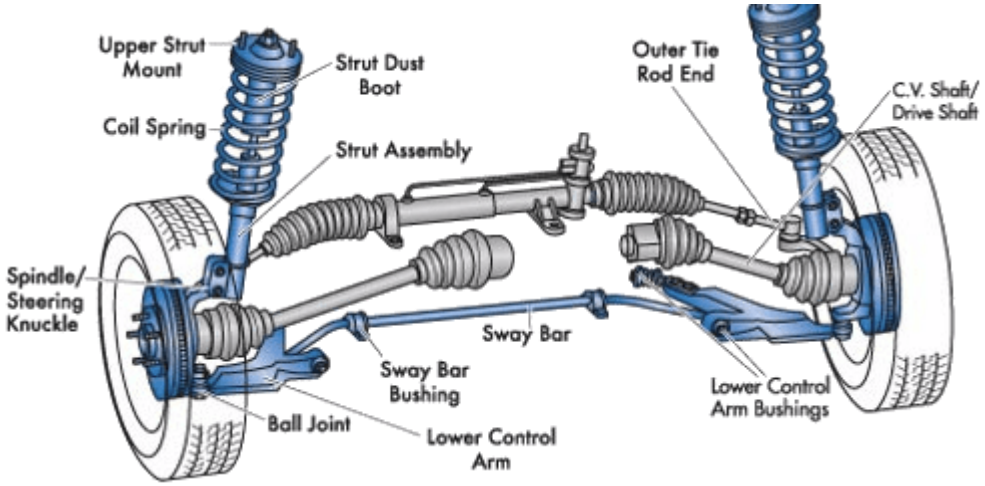


Figura 10 - Tipologia *MacPherson*

2.1.2 Funcionamento

Para um veículo rodoviário, o sistema de suspensão isola os ocupantes ou carga dos choques e vibrações induzidos pela superfície da estrada. Este isolamento dos choques e vibrações induzidos também melhora a durabilidade do veículo. Igualmente importante, o sistema de suspensão permite que as rodas mantenham contacto com a estrada, assegurando a estabilidade e o controlo do mesmo.

A Figura 11 mostra o conceito básico de um sistema de suspensão em termos de mola e amortecimento (mas neste caso ignora o aspeto da localização da roda). Este esquema é frequentemente utilizado em trabalhos técnicos relacionados com a modelação matemática de sistemas de suspensão e é designado de modelo de quarto de carro, representando assim uma única secção de roda. A figura ilustra a massa suspensa do carro (neste caso, a massa suportada por uma roda - normalmente um quarto da massa total suspensa), apoiada na roda e do conjunto da suspensão (referida como massa não suspensa) que, por sua vez, é suportada acima da superfície da estrada pela mola do pneu pneumático. O pneu atua como um sistema de mola secundária, atuando tanto sobre a massa não suspensa da suspensão e a massa suspensa do veículo carregado.

Entre as massas suspensas e não suspensas estão localizados os mecanismos de amortecimento, incluindo o amortecedor e o atrito inerente resultante das juntas e dos contactos deslizantes no sistema de suspensão.

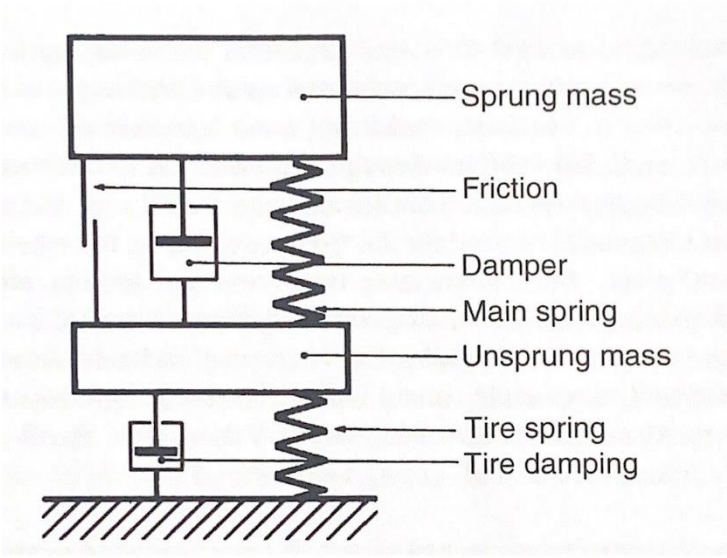


Figura 11 – Modelo de ¼ de carro de um sistema de suspensão

Uma haste de pistão inserida num cilindro cheio de líquido deslocará uma quantidade de líquido igual ao volume da haste do pistão, a quantidade de líquido é forçada a sair do cilindro e só pode fluir através do sistema de válvula para o reservatório de fluidos, gerando forças de amortecimento.

No amortecedor monotubo de gás/hidráulico (sendo esta a tipologia dos amortecedores deste projeto) a parte inferior do corpo de trabalho o cilindro funciona como reservatório com um pistão divisor flutuante, mantendo o volume de gás e o volume de fluido separados um do outro. O amortecedor de gás/hidráulico de tubo simples pode ser equipado com um reservatório externo que absorve a quantidade de fluido deslocado para fora do cilindro, poupando o comprimento da embalagem, o que é importante quando é necessário um amortecimento curto. Conduzindo o fluido através de válvulas e orifícios, são criadas forças de amortecimento. A quantidade de amortecimento é determinada pelo número de orifícios e de válvulas carregadas por mola.

Durante o curso de retorno/ descompressão, quando a haste é puxada para fora do cilindro, o fluido pressurizado retido entre o pistão e a guia flui através do sistema de válvula do pistão para a parte inferior do cilindro, enquanto que a quantidade de fluido que foi

deslocada para fora do cilindro durante o curso de compressão flui de volta para o cilindro, ocupando o lugar disponível da haste do pistão. [8]

Atualmente, a maioria (mas não todos) os desenhos de suspensão utilizam molas helicoidais numa grande variedade de formas, tipicamente utilizadas concentricamente com amortecedores telescópicos, frequentemente referido como "*coil-over-damper*". A flexibilidade dos seus critérios de conceção e a sua capacidade de serem fabricados a um preço relativamente baixo, em grandes quantidades e com características consistentes, trouxeram as molas helicoidais para a vanguarda de *design* moderno de suspensão. A forma mais comum de suspensão frontal em bobina é a tipologia escora *MacPherson*. O mesmo apresenta sistemas de suspensão passiva, que representam a maioria dos sistemas em funcionamento nos dias de hoje.

O caminho agora está no foco do desenvolvimento de sistemas adaptativos, que consistem em, mediante a resposta do sistema, haver um ajuste automático para as mais diferentes condições. Isto pode ser conseguido de forma passiva ou, cada vez mais, sob controlo informático. Os sistemas semi-ativos utilizam normalmente molas passivas para suportar o peso estático do carro e fornecer suspensão para baixas frequências e grandes deslocamentos, e os atuadores hidráulicos são controlados por computador para fornecer uma suspensão a altas frequências e pequenos deslocamentos. Na modalidade que o carro em estudo está inserido, neste caso em Rally, este sistema semi-ativo não é utilizado uma vez que além de ser impossível a telemetria em prova, os pisos dos troços são bastante irregulares, pelo que as forças e o desgaste em questão são extremas e não compensariam este género de sistemas adaptativos. [9]

2.2 Sistemas DAS

Uma das vantagens competitivas mais importantes para uma equipa de carros de corrida é a informação disponível. Quanto mais informação for possível recolher e processar, melhor será a interpretação aquando da tomada de decisão. A aquisição de dados fornece aos engenheiros as informações que, tanto estes como a equipa, necessitam para uma avaliação do desempenho e do estado do veículo.

2.2.1 Sistemas DAS

Simplificando, um sistema de aquisição de dados de veículos de corrida é uma unidade de memória eletrónica que armazena parâmetros definidos pelo utilizador em função do tempo, enquanto o carro estiver na pista ou troço. Os parâmetros armazenados nos dados podem ser descarregados para um computador onde poderão ser analisados, muitas vezes com pacotes de *software* incorporados no veículo.

Existem diversas categorias para aquisição de dados, dentro dos quais a *Vehicle Performance Analysis* (VPA), a *Driver Performance Analysis* (DPA), o *Vehicle Development* (VD), a *Reliability* and *Safety* (RS), os *Determining Vehicle Parameters* (DVP) e os *Running Logs* (RL). Embora muitos dos sinais estejam frequentemente inter-relacionados, os dados que o sistema mede podem ser divididos geralmente nas seguintes categorias: As funções importantes do carro, a atividade do piloto e os parâmetros do chassis.

Dependendo do orçamento disponível para a aquisição de um sistema assim completo, as possibilidades são quase infinitas. Existem sistemas de aquisição de dados para quase todas as aplicações. A configuração tradicional de aquisição de dados consiste numa unidade de registo adequada, que mede o mais variado número de sinais disponíveis para posterior tratamento a gosto do(a) engenheiro(a) e do(a) piloto.

2.2.2 Elementos Constituintes

Tudo o que um *data register* faz é armazenar números. E mesmo com taxas de amostra modestas, os dados de uma única volta formam uma lista muito vasta de números, pois quanto menor for o intervalo de tempo entre estes, melhor a precisão dos resultados. Devido a isso, é necessária uma ferramenta para auxiliar a visualização desta lista que ajude a compreender facilmente o que está a acontecer com o carro. Numa pista ou troço de corridas, durante uma prova ou sessão de testes, o tempo disponível para analisar os dados do diário de bordo é limitado. O(A) engenheiro(a) de dados deve fornecer respostas claras num tempo muito curto. O *software* de análise utilizado é um fator importante para fazer face a essa entrave, daí que escolher o pacote de *software* certo que execute o que é pretendido é essencial.

A Figura 12 permite compreender qual o processo efetuado desde que se pretende analisar um certo parâmetro até que os dados são analisados. Um parâmetro físico (por

exemplo, pressão, temperatura, velocidade, força) de interesse é capturado por um sensor que transforma a medição num sinal eletrónico proporcional a este parâmetro e compreensível para a unidade de registo de dados. A propriedade mais importante da unidade de registo de dados é que armazena os parâmetros medidos numa memória eletrónica.

Um dispositivo de saída (computador) pode comunicar com o dispositivo de registo de dados através de uma ligação externa. Esta ligação é muito frequentemente bidirecional porque a maioria dos sistemas oferece alguns parâmetros a serem configurados pelo utilizador [3].

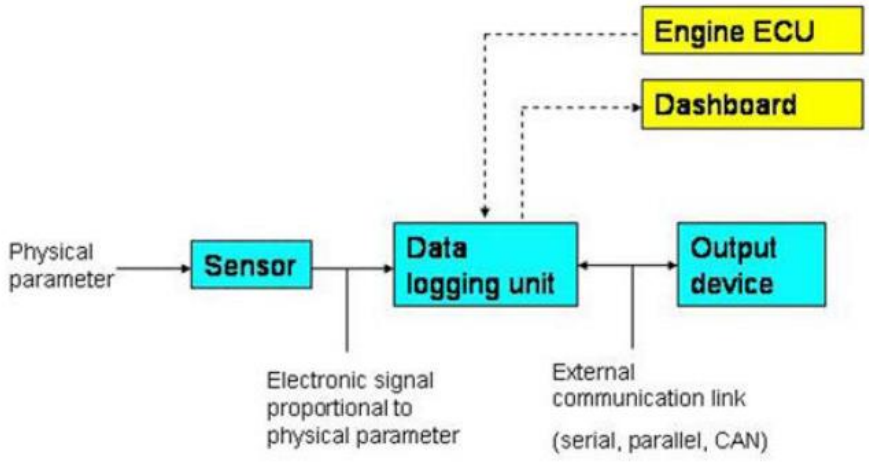


Figura 12 - Configuração geral de um sistema de aquisição de dados

2.2.3 Soluções Tecnológicas Existentes/Possíveis

Existem *softwares* mais eficientes e robustos para este tipo de aquisição e tratamento de dados. No entanto, o que é pretendido pela *The Racing Factory* é não adquirir licenças de *software* mais caras, criar um sistema simples, mas viável, com um código fácil de programar e que permita sofrer alterações. É também importante referir que as placas Arduíno são bastante acessíveis e intuitivas de usar, o que promoveu a escolha deste equipamento.

Existem bastantes *softwares* para *Motorsport* no mercado, tais como o *MoTeC* ou o *AiM*, ambos semelhantes ao *WinTax* utilizado nas placas da *Marelli*, no que toca ao tratamento dos dados, no entanto no caso dos Ralis são exclusivos e impenetráveis sem a devida autorização dos fabricantes, não é possível adicionar canais com mais informação de

parâmetros, daí ser necessário criar um sistema do zero e independente para o efeito pretendido.

Relativamente aos sensores existentes no mercado automóvel para este fim, existe um tipo de sensor que é quase sempre utilizado para retirar dados do *Suspension Travel*, que são os sensores de posição linear ou potenciómetros lineares. Estes são mais dispendiosos, foram criados para outras funções, mas foram adaptados para serem aplicados nos sistemas de suspensão e amortecimento dos carros de competição. São assim, por sua vez, robustos, precisos e já possuem inserção de canais para este parâmetro nos programas de análise de dados. No caso da Magneti Marelli, existem dois tipos de sensores deste género, próprios da marca, o LP 75-150 J e o LPT 50-150, representados na Figura 13[10].

Figura 13 - Potenciómetros lineares LP 75-150 J e LPT 50-150

Potenciómetros lineares como estes existem no mercado e são fáceis de encontrar, o problema encontrado é que, para adquirir tanto o equipamento como a possibilidade de adicionar os canais ao *software* DAS, é caro e não compensa os milhares de euros que se investe, a não ser que o sensor já venha incorporado no carro aquando da sua aquisição.

2.2.4 *Software* utilizado

Para esta trabalho, foi necessária uma abordagem diferente, visto que não era possível aceder ao *software* implementado no carro, da *Magneti Marelli*. Se assim fosse, só bastaria

adicionar um canal para recolha e análise de dados e tudo seria mais simples e eficaz, pois todos os dados estariam a ser tratados na mesma *data logging unit* e poderiam ser estudados em simultâneo para comparar valores de vários parâmetros.

Para isso, foi pensado utilizar uma abordagem simples, relativamente robusta para o comando que teria de executar, através de Arduino e Excel. Outros programas foram usados para auxiliar todo o processo e desenvolvimento deste sistema, pelo que serão descritos de seguida.

Arduino

O Arduino é uma plataforma eletrónica baseada em uc de código aberto baseada em *hardware* e *software* fáceis de usar. As placas Arduino são capazes de ler entradas e transformá-las em saídas, ativando um motor, ligando um LED, etc. É possível comandar a placa enviando um conjunto de instruções para o microcontrolador. Para isso, é necessário utilizar a linguagem de programação Arduino, baseada em *Wiring*, uma plataforma computacional física semelhante, e o Software *Arduino (IDE - Integrated Development Environment)*, baseado ambiente de programação *Processing*, feito em Java [9].

Numa perspetiva de desenvolver um sistema simples e de fácil acesso para possíveis alterações, foi verificado que a junção do Arduino com o Excel seriam uma opção mais vantajosa do que um programa mais complexo da *National Instruments*, por exemplo, além de mais dispendioso também devido à necessidade de aquisição de licenças. Para este projeto, o objetivo era desenvolver uma solução que fosse facilmente alterável tanto ao nível da aquisição de dados (componentes mecânicas e eletrónicas) como ao seu tratamento (programação). Em conjunto com a empresa parceira deste projeto ficou então decidido que seriam esses então os dois programas principais que se iria utilizar.

Neste projeto, o Arduino servirá para efetuar ordens de comando de certos componentes acessórios (tais como sensor infravermelhos, LEDs, leitor de cartão de memória SD, entre outros) mas, principalmente, para registar os dados retirados dos sensores de posição angular que serão implementados no carro.

Microsoft Excel

O Microsoft Excel é um programa muito útil e poderoso usado para análise de dados e documentação. É um programa de folha de cálculo, que contém várias colunas e linhas, onde

cada interseção de uma coluna e uma linha é uma célula. Cada célula contém um ponto de dados ou uma informação. Organizando as células dessa forma, é possível tornar as mesmas mais fáceis de localizar e obter informações de forma automática. [11]

Para este projeto, o Excel tem a função de aceder a um ficheiro de texto, que vai ser gerado através de comandos dados pelo Arduino e que vai conter todos os dados retirados do sensor. Este executa, de forma automática, a passagem desses dados para células do mesmo, que posteriormente e através de uma equação específica, cria a tão esperada curva de comportamento do curso do amortecedor.

2.2.5 Škoda Fabia Rally 2 EVO

De maneira a compreender melhor o sistema que será implementado, é importante conhecer o veículo de competição desta época que usufruirá dessa funcionalidade, entender a visão da marca construtora, a *ŠKODA Motorsport,* as suas respetivas evoluções e, com isto, analisar a importância que uma maior eficiência e melhor uso das novas tecnologias tem em todos os subsistemas de um carro de competição. O objetivo da *ŠKODA Motorsport* era desenvolver um carro que pudesse melhorar ainda mais as habilidades consideráveis da conceituada primeira versão. O carro anterior conquistou três títulos no *World Rally Championship 2* (WRC 2) e centenas de vitórias em Rallys em dezenas de países, tornando-se o carro de Rally de maior sucesso do mundo no grupo Rally2.

Esta nova versão está equipada com um motor de 1,6 litros atualizado com mais potência, sistemas de arrefecimento e lubrificação mais eficazes e um sistema eletrónico mais rápido. A caixa de velocidades também foi modificada, aumentando a sua vida útil. Toda a carroçaria foi reforçada e a suspensão agora possui um curso maior para uma maior aderência em condições extremas.

Para 2021, o *FABIA Rally2 EVO*, apresentado na Figura 14, recebeu outro conjunto de atualizações. O pacote inclui uma evolução no motor, bem como um *intercooler* redesenhado e coletor de escape. Especialmente desenvolvido para superfícies de baixa aderência, a *ŠKODA Motorsport* homologou um terceiro conjunto de rampas diferenciais. Uma nova configuração de amortecedores ZF oferece ainda melhor tração e estabilidade ao carro em todos os tipos de superfícies. [12]

Figura 14 - *Škoda Fabia Rally2 EVO* do Vice Campeão Nacional, Armindo Araújo, onde é implementado o sistema.

2.3 Tipos de Troço

Cada troço em Rally tem especificações próprias e possui demasiadas variáveis diferentes para se conseguir controlar todas as Provas Especiais de Classificação (PECs) a 100%. De uma forma geral, existem dois tipos de troço, os de terra e os de asfalto. Para cada um deste género de troços é necessária toda uma preparação e adaptação prévia dos carros na oficina, preparação essa que passa por aplicar aos veículos os seus respetivos *kits* de terra/asfalto e realizar o *setup* consoante o tipo de prova.

2.3.1 Troço de Terra

Os troços de terra são normalmente classificados como instáveis de solo, possuem muitos desníveis, abrasivos e buracos, o que irá fazer com que seja necessário preparar o carro para essas condições, focando bastante na proteção dos seus componentes na zona de maior contacto com o solo. Normalmente são utilizadas jantes menores mas mais robustas, pneus com piso de taco para o mesmo efeito, molas de suspensão mais macias para permitir ao carro ter uma maior liberdade no *Suspension Travel System* (STS) na passagem, adaptação dos amortecedores no seu curso, compressão, *rebound*, entre outros.

2.3.2 Troço de Asfalto

Por sua vez, os troços de asfalto são troços mais regulares, nivelados, no entanto com mais *grip*, sendo mais abrasivos. Como não sofrem tantas agressões, não necessitam de tanta proteção ou robustez, nem tanta liberdade de movimento do STS. É mais usual nestes sistemas encontrar-se um *kit* de molas de suspensão mais rígido, pois as mesmas promovem uma maior estabilidade. São necessárias jantes maiores, no entanto, não se usam pneus tão rugosos. Como foi abordado antes, são necessários discos maiores para este género de troço. Tudo o que envolva amortecedores, *setup* e outros componentes, também terá de ser adaptado para asfalto e para a PEC em especial, ou seja, não existem só dois tipos de *setups*, estes são criados e ajustados dependendo do solo que se está a analisar.

2.4 Estudo das Variáveis

Para este projeto, existirão duas variáveis principais a ter em conta, sendo estes os dados da diferença de potencial retirados do sensor de posição angular numa determinada posição do curso do amortecedor. A partir desses parâmetros será possível observar o comportamento dos amortecedores do carro, se estes necessitam de eventuais ajustes.

2.4.1 Posicionamento do Sensor e o seu Mecanismo de Barras

Para que sejam retiradas informações precisas neste sistema, é necessário um ponto fixo e um ponto movível de fixação, que serão em princípio o charriot e o braço de suspensão, respetivamente. Na Figura 15 é possível observar tanto o *charriot* da frente como o braço de suspensão da roda direita e o sensor de posição angular no seu posicionamento final escolhido.

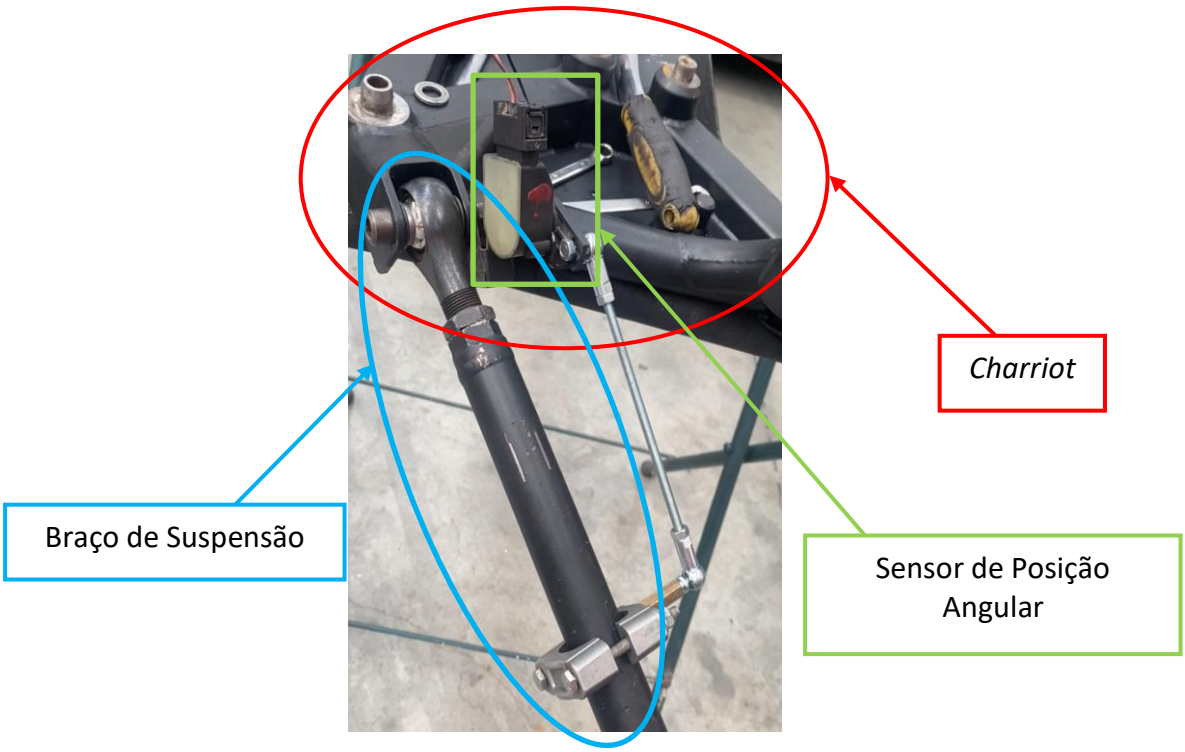


Figura 15 - Localização do sensor e dos seus braços no *charriot* e num braço de suspensão.

O objetivo é que, a cada 10 milissegundos (valor escolhido para que existisse um intervalo de tempo de registo de dados curto e obter uma curva de comportamento mais precisa), sejam registados os dados de diferença de potencial entre 0.25V e 5V do sensor representado. A cada valor dessa diferença de potencial irá corresponder uma altura específica do curso do amortecedor. Para tal, esses valores serão medidos previamente através de iterações e serão implementados numa equação linear. Essa equação será necessária para indicar, numa fase seguinte, quais serão os valores finais do curso do amortecedor a partir dos valores da diferença de potencial, em função do tempo.

Com o sensor de posição angular, veio acoplado um braço de 44,5 mm que media valores de diferença de potencial entre 0 e 90 graus, com dois batentes para que esses ângulos não fossem ultrapassados. Foi necessário criar um mecanismo Biela-Manivela com 1 GDL adicionando outro braço ao existente, com 205,3 mm, que se ligasse ao braço de suspensão através de um suporte metálico fixo. Na Figura 16 está representada a tipologia do sistema Biela-Manivela utilizado. [13]

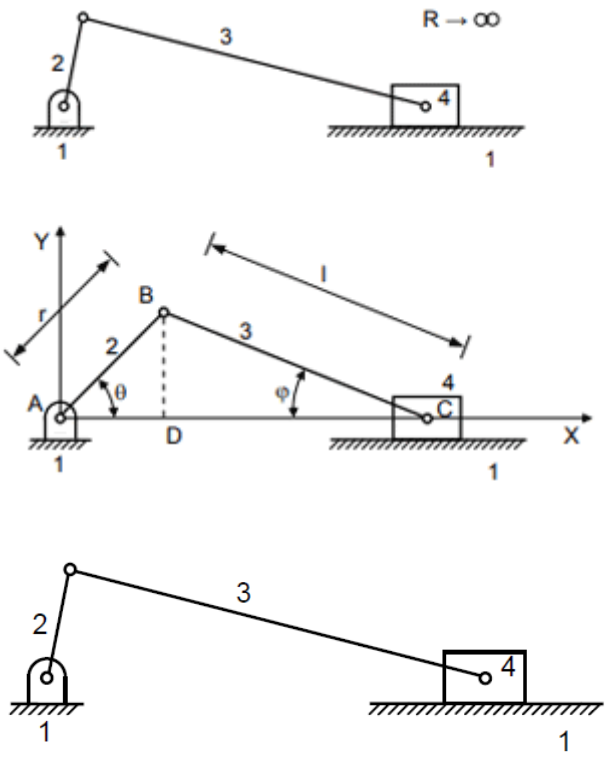


Figura 16 - Mecanismo biela-manivela.

2.4.2 Gestão da Compressão e *Rebound*

O STS é composto por uma mola e um amortecedor. A mola é um componente que promove resistência à carga. O amortecedor, que é como um pistão num tubo de óleo, resiste ao movimento (e fornece resistência apenas quando se encontra em movimento), ao contrário da mola, que fornece resistência sempre que a quantidade de compressão é superior a zero.

Quanto mais rápido o pistão se movimentar, mais resistência oferece. Infelizmente, a gama de velocidades que a suspensão comprime é demasiado alta para um pistão simples se movimentar corretamente. Para esse défice foram criadas umas aberturas que abrem e fecham a diferentes velocidades, de maneira a manter um nível de força adequado.

Certos amortecedores possuem mecanismos separados que estão ativos durante a compressão e durante o movimento de retorno do choque, designado de *rebound*. O *rebound* é o movimento de retorno da suspensão ou amortecedor depois de absorver a energia de impacto um obstáculo. Essa velocidade de atuação é muito importante, porque se for muito lenta e existirem vários obstáculos seguidos, a suspensão atuará corretamente no primeiro, mas não terá tempo de recuperar para amortecer os seguintes.

Se, pelo contrário, o *rebound* for muito rápido, o sistema será muito reativo e nervoso, e tenderá a desviar o carro do percurso pretendido, uma vez que vai fazer com que as rodas percam, em vários momentos pontuais, o contacto com o solo e deixem de agarrar, sendo que o controlo sobre a direção será muito impreciso.

A situação ideal é ter um *rebound* o mais rápido possível, mas sem que retire precisão à condução. É necessário, portanto, procurar o equilíbrio perfeito de velocidade.

Assim, há a possibilidade de obter vários modos para várias velocidades e para a direção do movimento. Os quatro modos principais são:

- Amortecimento por compressão a alta velocidade (HsC – *High Speed Compression*);
- Amortecimento por compressão a baixa velocidade (LsC – *Low Speed Compression*);
- Amortecimento a alta velocidade de *rebound* (HsR – *High Speed Rebound*);
- Amortecimento a baixa velocidade de *rebound* (LsR – *Low Speed Rebound*).

A principal diferença entre a compressão de alta e baixa velocidade (HsC e LsC) baseia-se no controlo da rapidez com que a suspensão é comprimida em diferentes tipos de impacto. Alguns amortecedores permitem ajustar ou os quatro modos separadamente, mas nem todos possuem essa particularidade e ajustam um ou outro separadamente.

O modo de funcionamento mais usual do amortecedor é para o mecanismo de *Low Speed* engatar, no entanto, se a força se torna bastante elevada devido a um movimento mais rápido, as válvulas abrem-se para reduzir a força, permitindo que o óleo flua mais livremente através do mecanismo de *High Speed*. Quando o movimento abranda o suficiente – abrandando a força de igual forma - estas válvulas fecham-se e o fluido volta a correr através do mecanismo de *Low Speed*, ou seja, mais restritivo. [14]

No caso da compressão, a velocidade de impacto está a ditar a velocidade de compressão, que é controlada pelos circuitos de amortecimento. Com o *rebound*, a velocidade é ditada pela mola, pois uma mola mais forte irá recuperar mais rápido. Independentemente da força da mola, é sempre necessário um circuito de amortecimento de *rebound* para controlar a taxa de ressalto, para que a suspensão não salte para trás muito rapidamente e cause uma perda de controlo do carro. O objetivo também não passa por existir muito amortecimento de recuperação, pois se a suspensão responder muito devagar, esta não estará pronta para o próximo impacto e, eventualmente, ficará sobrecarregada por sucessivos impactos [15].

2.5 *Suspension Travel System*

2.5.1 Suspensões Utilizadas - Škoda

Para cada tipo de troço, seja este de asfalto ou de terra, é necessário aplicar um género específico de suspensões, sendo estas de três tipos: Macias, Médias ou Rijas. Dependendo do troço e da necessidade de estabilidade imposta pelo mesmo, essas suspensões são escolhidas pelo(a) engenheiro(a). No caso dos *Škodas*, existe esta seleção de três pares de suspensões para ambos os tipos de troço, todos variando no seu valor de rigidez (em N/mm).

O *ŠKODA FABIA Rally 2 EVO* está equipado com molas de compressão helicoidais com uma característica linear. As molas são feitas de aço de alta resistência e as extremidades são retificadas para uma melhor distribuição da carga nas plataformas das molas.

A mola principal é suportada por uma mola auxiliar. É recomendado pela marca a utilização de molas principais com molas auxiliares para evitar qualquer dano ao amortecedor ou às plataformas das molas e para manter consistentemente o funcionamento adequado das mesmas. Existem as seguintes gamas disponíveis, tanto para terra como para asfalto, na Tabela *2*.

Tabela 2 - Listagem de todas as molas de suspensão disponíveis no Manual *online* (exclusivo para equipas parceiras de Rali) da *Škoda Motorsport*

	Part No.	Rate [N/mm]	Approx. free [mm]	Colour
Gravel	6VT 411 102 H	15.0	498	Green*
	6VT 411 102 J	17.5	460	Green*
	6VT 411 101 AF	20.0	459	Black
	6VT 411 101 AG	22.5	442	Black
	6VT 411 101 AH	25.0	428	Black
	6VT 411 101 AJ	27.5	416	Black
	6VT 411 101 AK	30.0	407	Black
	6VT 411 101 AL	32.5	399	Black
	6VT 411 101 AM	35.0	392	Black
	6VT 411 101 AN	37.5	386	Black
Tarmac	6VT 411 102	30.0	308	Green*
	6VT 411 102 A	35.0	292	Green*
	6VT 411 102 B	40.0	283	Green*
	6VT 411 102 C	45.0	274	Green*
	6VT 411 101 S	50.0	257	Black
	6VT 411 101 T	55.0	251	Black
	6VT 411 101 AA	60.0	247	Black
Helper	6VT 411 104	2.0	190	Black

2.5.2 Amortecedores Utilizados - Škoda

A *ŠKODA Motorsport* introduz a nova especificação de amortecedores para troços de terra, que foi desenvolvida pela equipa de engenharia com base na experiência adquirida e comprovada por pilotos experientes. O desempenho e a fiabilidade desta especificação foram significativamente melhorados em comparação com amortecedores anteriores.

De seguida, estão dispostas algumas informações relativas à nova atualização dos amortecedores utilizados no veículo em estudo.

- Nome da regulação: 8657A (Frente);

  9559A (Traseira);
- Fabricante: *ZF Race Engineering;*
- 3 vias ajustáveis (de altos e baixos *bump*/*rebound* independentemente) com reservatório externo;
- Ajuste de *Low Speed* e *High Speed Bump* no reservatório;
- Ajuste de *rebound* através da haste do pistão;
- Batente de colisão hidráulico não regulável;
- Batente de ricochete hidráulico não regulável;
- Design monotubo com carga de gás.

Nestes amortecedores, é possível alterar a gama de regulação da compressão. Nas ilustrações seguintes estão descritos todos os amortecedores utilizados tanto para o Kit de Terra como para o Kit de Asfalto(Tabela 3), assim como uma representação em 3D do site oficial da *Škoda Motorsport* dos principais componentes envolvidos neste sistema(Figura 17).

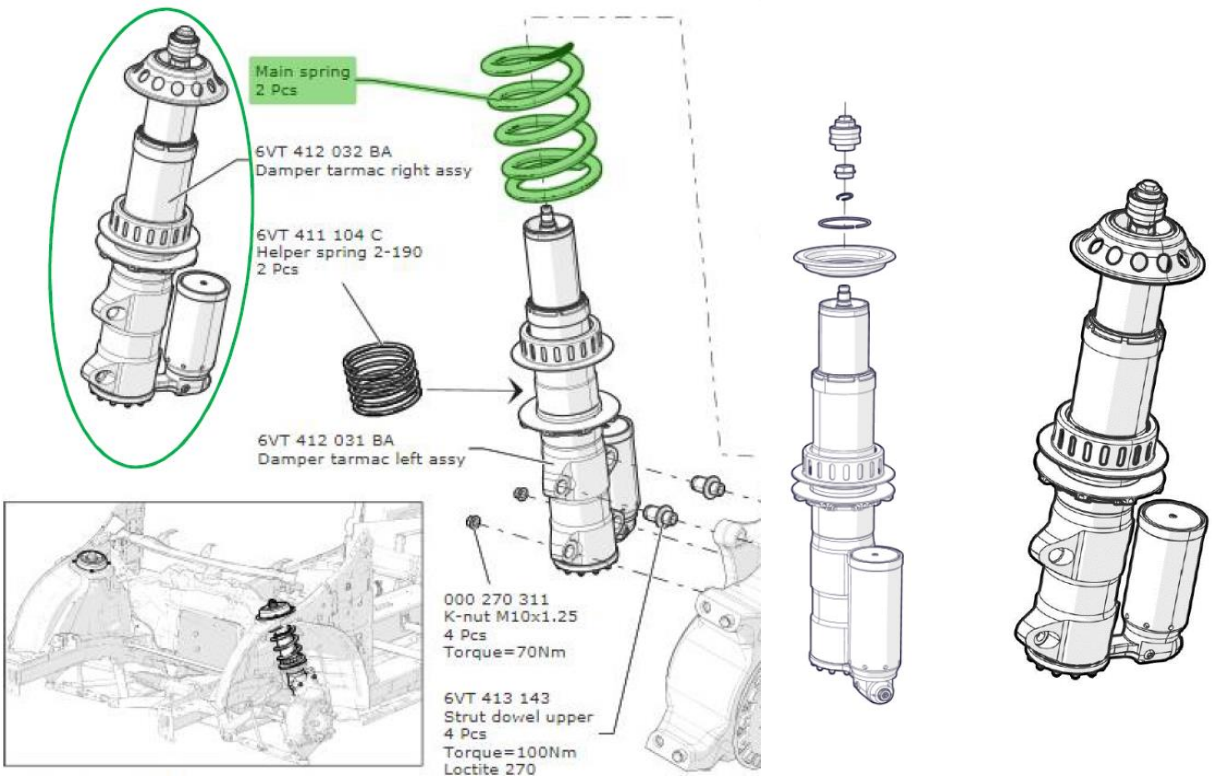


Figura 17 - Representação em 3D do catálogo *online* exclusivo e interdito da *Škoda Motorsport*

Tabela 3 - Os amortecedores utilizados para o Kit de Terra como para o Kit de Asfalto

Position	Part name	Part number	Setting name
Tarmac			
FL	Damper tarmac left assy	6VT 412 031 BK	P12549B
FR	Damper tarmac right assy	6VT 412 032 BK	P12549B
RL	Damper tarmac left assy	6VT 412 031 BM	P12753B
RR	Damper tarmac right assy	6VT 412 032 BM	P12753B
Gravel			
FL	Damper gravel left assy	6VT 412 031 BN	P12957A
FR	Damper gravel right assy	6VT 412 032 BN	P12957A
RL	Damper gravel left assy	6VT 412 031 BP	P13159A
RR	Damper gravel right assy	6VT 412 032 BP	P13159A

2.5.3 Modelação 3D do Sistema

Devido à impossibilidade da *Škoda Motorsport* fornecer na modelação em 3D do sistema em questão por motivos de confidencialidade, foi necessário desenhar uma modelação 3D do STS que permitisse visualizar os componentes mais importantes que intervém neste subconjunto. Como não existe a necessidade de obter equações de movimento e análises estáticas e dinâmicas neste projeto, foi decidido que, para este

trabalho, se iria obter mais proveito de todo o conhecimento à disposição se fossem entendidas a fundo todas as componentes do STS. Dessa forma, não se focou o tempo em modelações mais complexas, mas sim em oficina a desmontar e montar os equipamentos minuciosamente.

Todas as modelações deste sistema foram realizadas através do *software* de modelação 3D em *SolidWorks*. Para esta dissertação foi pensado um desenho 3D para que este servisse de apoio na compreensão de todos os elementos envolvidos nesta dissertação e, o mais importante, visualizar onde seria o local mais apropriado para a colocação do sensor de posição angular.

Nas imagens seguintes estão representados todos os elementos que constituem o *Suspension Travel System*, dentro dos quais fazem parte o *charriot* (representados só os braços ligados a um extremo lateral do *charriot*), a manga de eixo, o amortecedor com suspensão montada no mesmo e os braços de direção que suportam da manga de eixo para cima todo o movimento existente neste subconjunto. Primeiramente, é possível observar na Figura 18, uma perspetiva aproximada só do mecanismo adicionado ao sistema, mais propriamente nos braços de suspensão e numa parte do *charriot* do *Škoda*. É possível observar o movimento que os braços do sensor realizam quando existe movimento nos braços de suspensão.

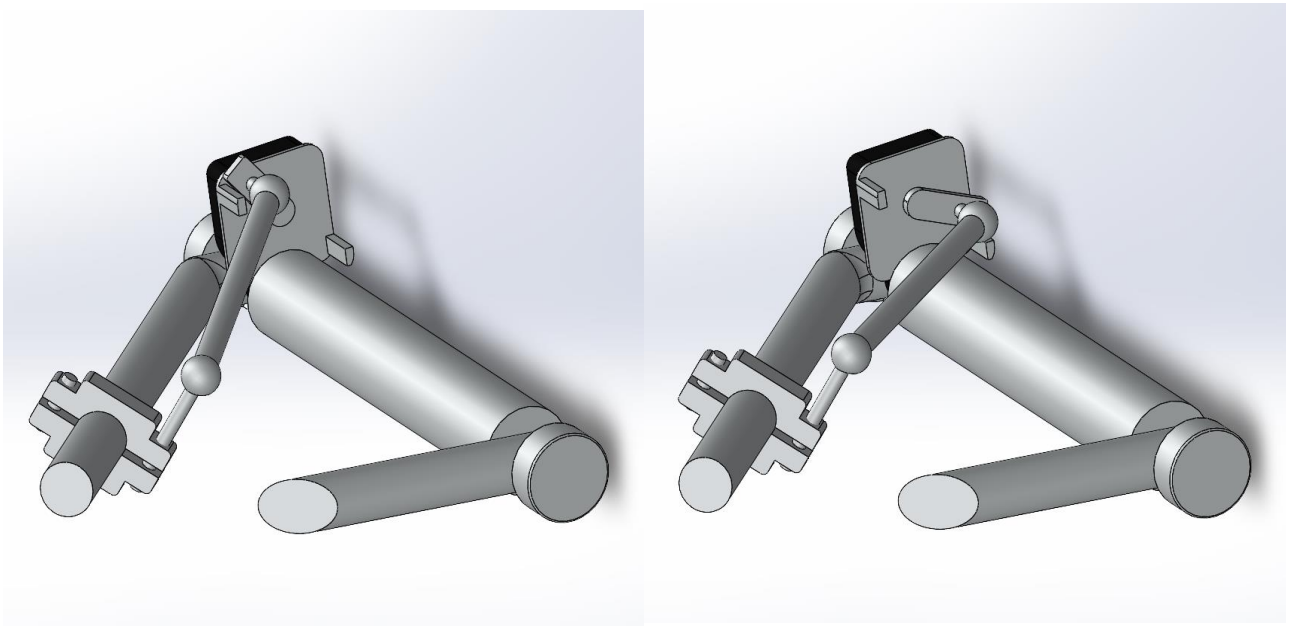


Figura 18 - Mecanismo sensor-braços em *SolidWorks*, onde é possível observar o movimento do braço menor ligado diretamente ao sensor.

Na Figura 18, é apresentado todo o conjunto em *assembly* final, com uma manga de eixo, um amortecedor com suspensão e um excerto da carroçaria do carro.

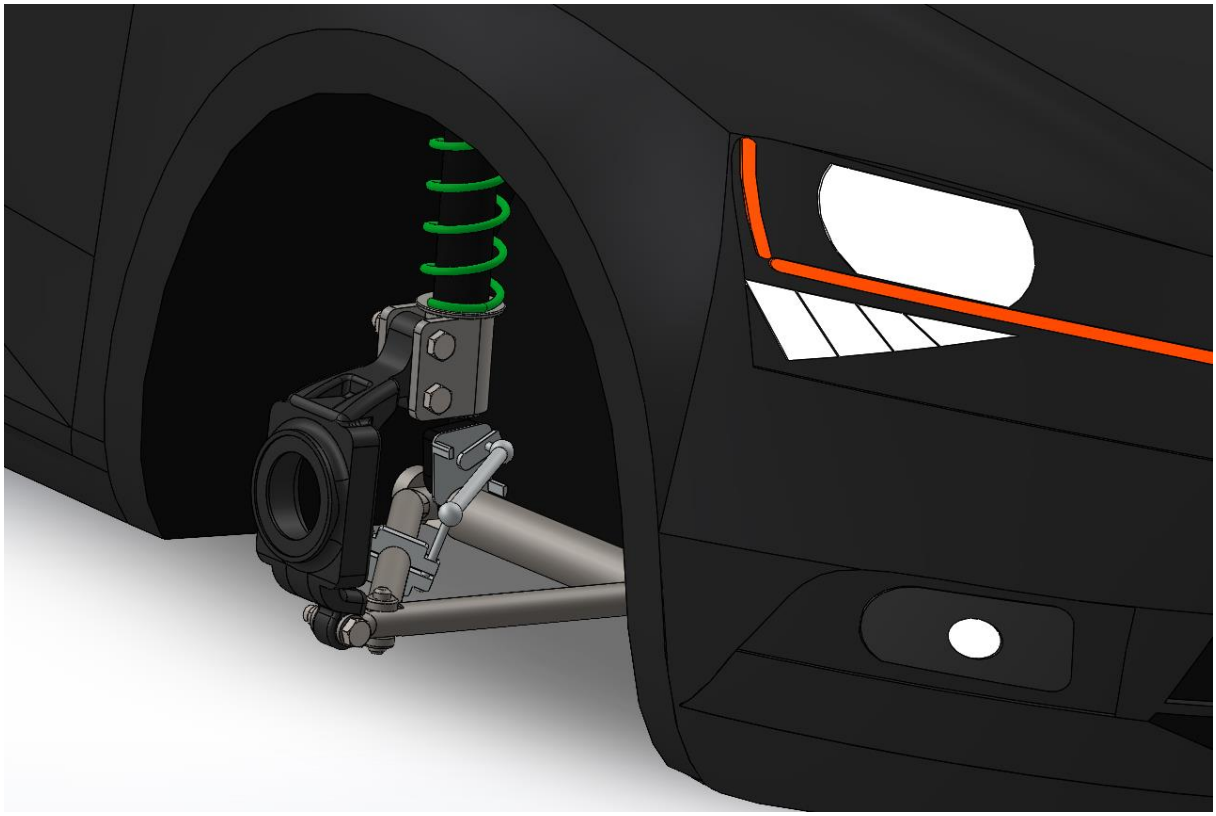


Figura 19 - *Assembly* Final do sistema implementado no *Škoda*, em *SolidWorks.*

A RETER DESTE CAPÍTULO

A base bibliográfica, ou estado da arte, é importante em qualquer projeto, pois visa dar suporte factual a tudo o que se aborda e desenvolve numa fase seguinte. Para isso, é importante entender os componentes constituintes do veículo em estudo, mais propriamente os que estão presentes no STS (*Suspension Travel System*), de forma a observar quais as necessidades, os desafios e as variáveis antes de criar um sistema de aquisição e tratamento de dados que se interligue corretamente. Como o maior desafio que se tenta combater é o desgaste do sistema, são abordados também os tipos de troço existentes em Rally. Também é necessário realizar uma análise mais aprofundada dos sistemas DAS no geral e dos tipos de sensores existentes ou não no mercado, utilizados para este parâmetro de medição em específico. Para isso, realizar as devidas modelações em 3D permite complementar o projeto e criar uma melhor noção espacial de onde e como se localiza o sistema DAS e de todas as partes envolvidas.

3. SISTEMA DE RECOLHA DE DADOS

Neste capítulo são efetuados diversos estudos, dentro dos quais o estudo dos sensores utilizados no sistema, a análise de quais os *softwares* que apoiam, tanto na recolha dos dados como em todo o projeto, uma explicação da escolha e o contributo de todos os componentes posteriormente adicionados, um mapa de tráfego de dados que explica todo o processo desde a leitura à análise e tratamento dos mesmos, entre outros.

Também aqui é explicada a fase mais importante deste projeto de tese. É aqui desenvolvido e apresentado o *software* DAS e o respetivo tratamento dos dados. Para isso são explicadas e desenvolvidas as ligações entre componentes, o respetivo código desenvolvido em Arduino para aquisição dos dados e a ligação do Arduino ao Excel para a leitura e tratamento dos mesmos.

3.1 Componentes Selecionados

3.1.1 Sensores de Posição Angular

Para este projeto, pretende-se colocar algum tipo de sensor que recolha algum tipo de dado de cada uma das quatro rodas e que, através desse dado, se possa chegar a um valor da posição do amortecedor. O movimento do curso do amortecedor é linear, o que quer dizer que uma possível abordagem seria colocar um sensor num ponto fixo do chassi e que estivesse ligado a um componente movível que acompanhasse o movimento do amortecedor. Após uma pesquisa e análise dos sensores que já existiam para este efeito e de outros sensores que executassem funções semelhantes, foi colocada a hipótese de se utilizar sensores de posição angular, neste caso potenciómetros (dão-nos uma diferença de potencial) para medir o movimento linear. Os potenciómetros são sensores analógicos utilizados normalmente para medir a rotação angular absoluta ou o movimento linear de um mecanismo. Um potenciómetro é um dispositivo de três terminais (ou mais, neste projeto usaram-se três) que utiliza um contacto móvel a partir de um divisor de resistência variável. Quando os contactos externos são ligados do Arduino a 5V e à Massa e o contacto variável é ligado a uma entrada analógica, a entrada analógica terá uma tensão analógica que varia à medida que o potenciómetro é rodado[16].

Os sensores escolhidos pertenciam a um *Renault* e serviam para regular a altura da luz dos faróis, no entanto, neste caso, será dada uma nova utilização e propósito a estes sensores da *Hella*, observados na Figura 21. Na Figura 20 está uma ilustração real do sensor Tipo A implementado no mecanismo.

Figura 21 - Sensores *Hella* Tipo A e Tipo B.

Figura 20 - Sensor Tipo A posicionado no carro

Os sensores de posição angular do tipo CIPOS® (sem contacto, sensores de posição indutiva) são concebidos para muitas aplicações diferentes para medir ângulos de forma precisa e fiável, mesmo em ambientes difíceis. A sua insensibilidade aos campos magnéticos e o seu elevado nível de estabilidade de temperatura em particular são as qualidades características da tecnologia CIPOS®, utilizada em todos os sensores de posição angular. Os ângulos são medidos indutivamente utilizando um método sem contacto e, portanto, resistente ao desgaste. Isto garante um elevado grau de precisão de medição durante toda a

vida útil do sensor. Os sensores redundantes (sensores duplos) são especialmente concebidos para a deteção de falhas, melhorando assim a fiabilidade do sistema no seu conjunto.

Abaixo está representado o Sensor do Tipo A, com apenas 4 pinos, dentro dos quais só se utilizarão três, o +5V (Pino 5), o GND (Pino 1) e o Output Signal (Pino 4). Na Figura 22 estão descritas todas as especificações técnicas e o desenho técnico deste sensor fornecidos pela marca, além dos pinos e uma representação gráfica da relação entre a diferença de potencial e do ângulo.

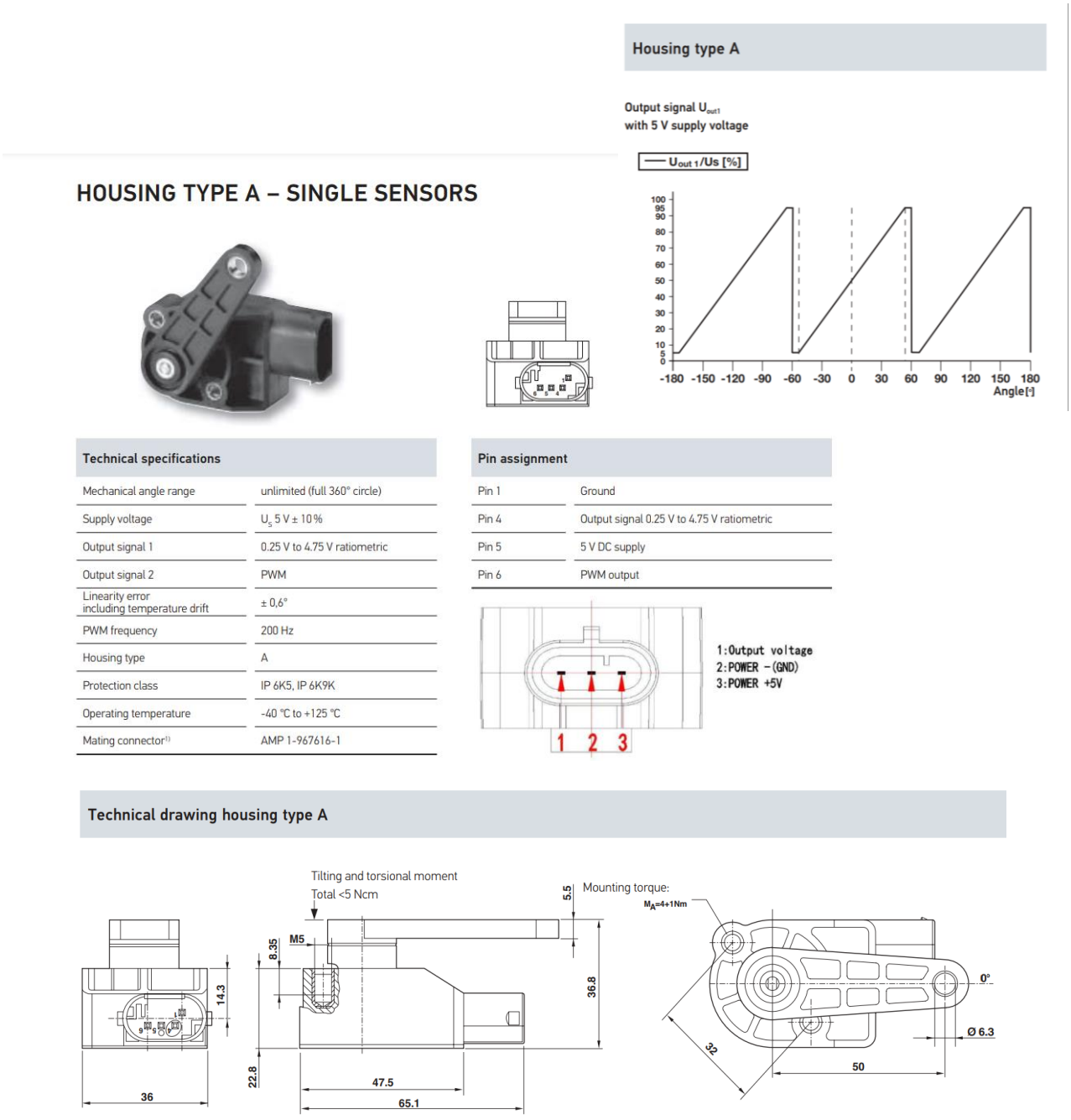


Figura 22 - Especificações técnicas do sensor Tipo A

Numa fase intermédia do projeto, são usados quatro sensores, dois do Tipo A e dois do Tipo B (Anexo 1), não por uma razão em específico, mas sim porque foram os únicos sensores deste género que se conseguiu adquirir numa sucata e correspondessem aos requisitos, após muita procura. A principal diferença entre estes está focada na capacidade de suportar tensões maiores que 5V (de 9V a 32V) e ter à disposição um número superior de pinos, tendo este dois potenciómetros em paralelo, o que para este projeto nada interfere pois podiam só ser utilizados 4 dos 8 pinos e estes estão preparados para funcionar também a 5V. Numa fase final da dissertação, é analisado que basta efetuar os teste num dos sensores pois todo o processo para um deles corresponde ao mesmo para os restantes. O sensor utilizado é o do Tipo A. Com tempo e como trabalhos futuros, o objetivo é repetir tudo o que será efetuado com o sensor do Tipo A para os restantes sensores.

A curva característica do sensor de posição angular repete-se a cada 120°. O sensor não tem, portanto, de ser instalado na posição de montagem apresentada, mas pode ser instalado qualquer ângulo de *offset* que seja um múltiplo de 120°. Isto não afetará o comportamento do sistema ligado de qualquer forma. A gama de ângulos de medição é de 108°. Se o sinal continua a cair ou a subir até 6°, o sinal de saída permanece no valor limite da gama de medição. Se o sinal exceder estes valores, a secção seguinte da curva característica é aplicada. Os intervalos de medição resultantes e as posições zero são mostrados no gráfico da Figura 22.

Os segmentos do círculo mostrados a cinzento na Figura 23 representam os ângulos que não podem ser medidos.

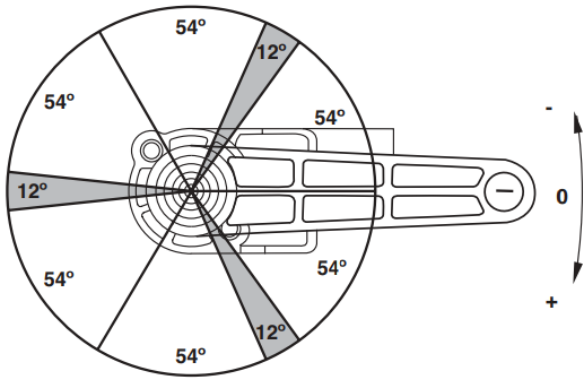


Figura 23 - Representação dos ângulos do sensor onde não se consegue medir diferença de potencial.

Passando a uma explicação do seu funcionamento, no interior da caixa de poliamida soldada a laser (PA66), a rotação do braço da alavanca é transferida para o rotor e medida por indução. Um ASIC (*Application Specific Integrated Circuit*) calcula com precisão a posição do rotor. São possíveis várias posições de montagem graças à curva característica de repetição do sinal de saída (que depende da estrutura do sensor que é utilizado), o que aumenta a flexibilidade do mesmo. Na Figura 24 são observados todos os elementos constituintes e o respetivo funcionamento. [17]

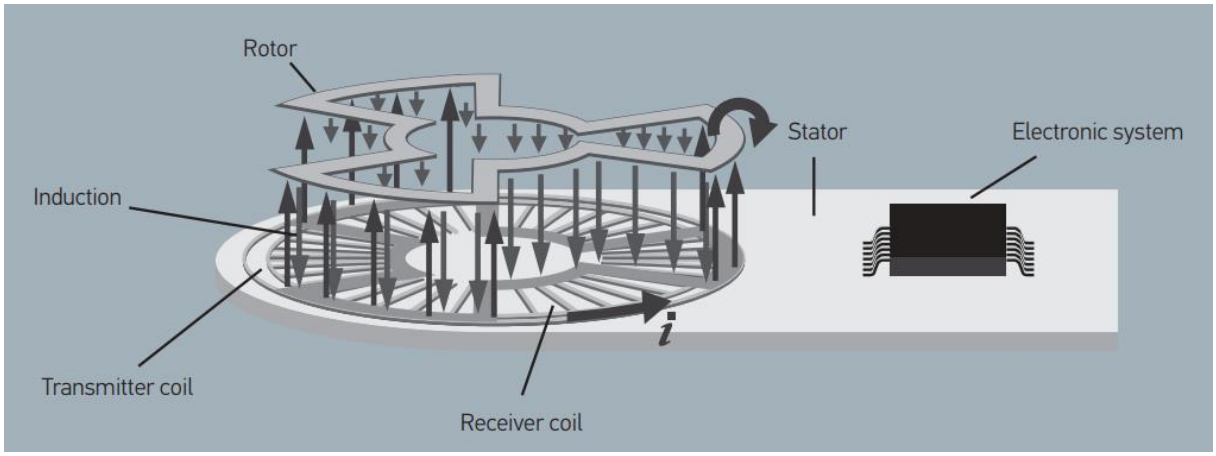


Figura 24 - Elementos que constituem o mecanismo do sensor

3.1.2 Placa Arduino Uno Rev 3

A placa física Arduino *Uno*, ilustrada na Figura 26, é uma placa microcontrolada baseada no ATmega328P. Possui 14 pinos de entrada/ saída digital (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um condensador de tensão variável de cerâmica de 16 MHz (CSTCE16M0V53-R0), uma conexão USB, um conector de alimentação, um conector ICSP e um botão de *reset*. Esta placa contém tudo o que é necessário para dar suporte a um microcontrolador, bastando conectá-lo a um computador com um cabo USB ou ligá-lo com um adaptador AC-DC ou bateria para funcionar. [18] É uma placa versátil, segura e relativamente resistente. As suas restantes especificações estão presentes na Figura 25.

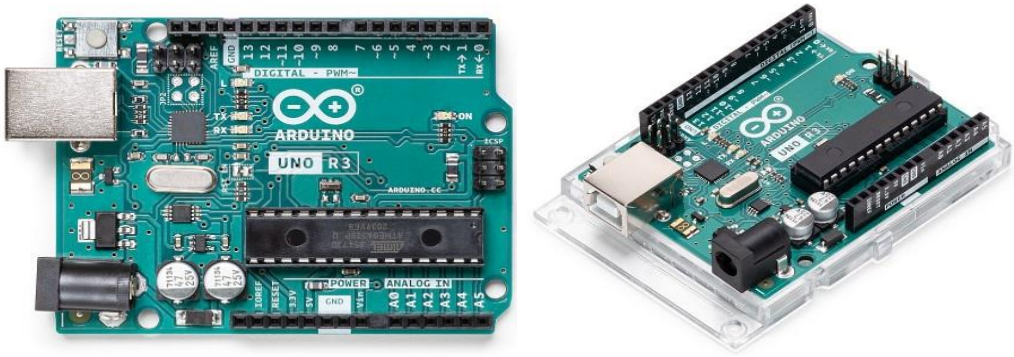


Figura 26 - Placa *Arduino Uno Rev 3*

Microcontrolador	ATmega328P
Tensão operacional	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limite)	6-20V
Pinos de E / S digitais	14 (dos quais 6 fornecem saída PWM)
Pinos de E / S digital PWM	6
Pinos de entrada analógica	6
Corrente DC por pino de I / O	20 mA
Corrente DC para pino de 3,3 V	50 mA
Memória flash	32 KB (ATmega328P) dos quais 0,5 KB usados pelo bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Velocidade do relógio	16 MHz
LED_BUILTIN	13
Comprimento	68,6 mm
Largura	53,4 mm
Peso	25 g

Figura 25 - Especificações Placa Arduino Uno Rev 3

Esta placa Arduino serve como base para controlar a PCB conectada à mesma, que tem conectado a si o sensor de posição angular, através de um código específico criado para comandar todas as funcionalidades do sistema DAS pretendido. Todos os componentes que completam a placa Arduino, tal como o código em questão, estão apresentados de seguida.

3.1.3 Placa PCB e Peças Complementares

Placa PCB

Durante o planeamento do projeto, observa-se que utilizar só a placa Arduino e uma *breadboard* é uma opção demasiado frágil, as ligações por cabos não aguentam movimentos mais extremos e não é tão funcional devido à quantidade de componentes.

Para isso, encomendou-se uma impressão de uma PCB.

PCB ou Placa de Circuito Impresso é o nome tradicional dado a uma placa, que é utilizada para suportar mecanicamente e ligar eletricamente componentes eletrónicos utilizando caminhos condutores, pistas ou traços de sinal gravados a partir de folhas de cobre, laminadas, a um substrato não condutor. Atualmente, as placas de circuitos impressos são utilizadas em praticamente todos os dispositivos eletrónicos, exceto nos mais simples produzidos comercialmente, e permitem processos de montagem totalmente automatizados que não eram possíveis ou práticos nos processos de montagem de circuitos da era anterior. Uma PCB preenchida com componentes eletrónicos é chamada de montagem de circuito impresso (PCA), montagem de placa de circuito impresso ou *Printed Circuit Board Assembly* (PCBA). [19]

Através do laboratório do Departamento de Eletrónica Industrial (DEI) é possível encomendar uma placa *Printed Circuit Board* (PCB) que liga todos os componentes que se pretende soldar à placa e as suas respetivas ligações, obtendo um protótipo mais organizado, de dimensões menores e mais viável, como a Figura 27 ilustra.

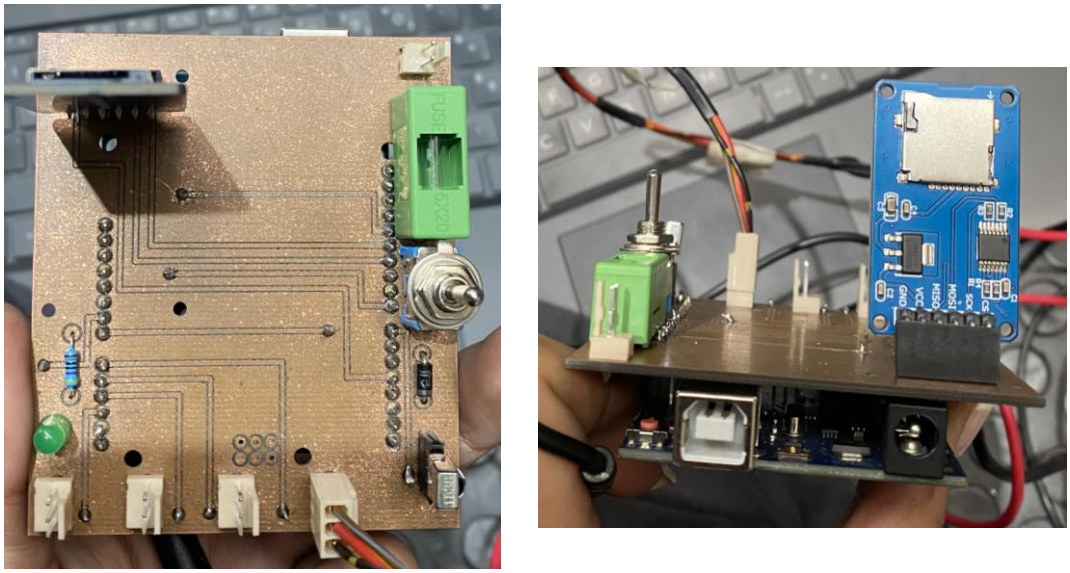


Figura 27 - PCB com todos os componentes.

Antes de se proceder à impressão, é necessário utilizar um *software* para desenhar onde se localizam os componentes e qual deve ser a disposição das ligações entre os mesmos na PCB, pelo que se recorre ao *Autodesk Eagle* para esse efeito, programa que é explicado de seguida.

Software Autodesk Eagle

O *Autodesk EAGLE* é um *software* de automatização de desenho eletrónico (EDA) que permite aos desenhadores de placas de circuito impresso (PCB) ligar sem problemas diagramas esquemáticos, efetuar uma colocação de componentes corretamente, os caminhos e ligações da PCB, entre outras funcionalidades. [20]

Tomando conhecimento, numa fase mais avançada do projeto, quais são os componentes que devem estar presentes na placa, estes são colocados num desenho eletrónico, através de bibliotecas posteriormente instaladas. Nas figuras Figura *28* e Figura *29* estão representados os esquemas finais da PCB, que servem para proceder à impressão da mesma. Na Figura 28, está representado o esquema com as ligações de forma simplificada,

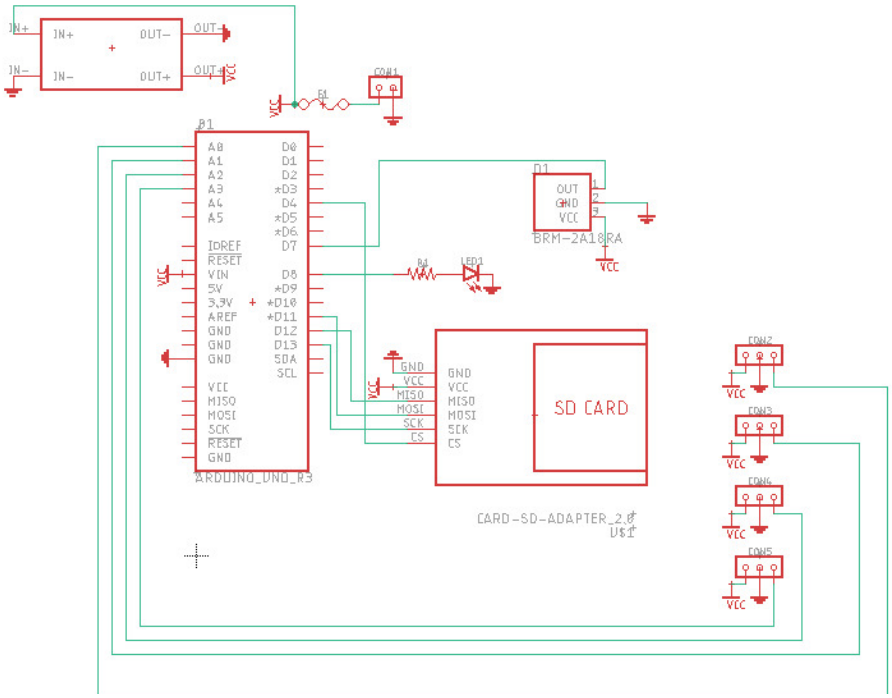


Figura 28 - Esquema das ligações entre componentes no *software Autodesk Eagle*

ainda sem preocupação de como se vai dispor tudo num PCB e quais são as interseções das ligações.

Já na Figura 29, é efetuada uma disposição mais cuidada de todas as peças, principalmente a colocação das ligações para unir a placa ao Arduino, a localização do leitor de cartões SD e, principalmente, os caminhos das ligações para que nenhuma entre em curto-circuito. Como se pode ver na Figura 27, o leitor acaba por não ficar na disposição pensada inicialmente visto que, na altura da montagem e soldadura dos componentes, não havia um *header* que permitisse que esta ficasse posicionada a 90º deitada na PCB, pelo que teve de ficar ao alto. De resto, todos os componentes estão soldados no local pretendido e na posição projetada.

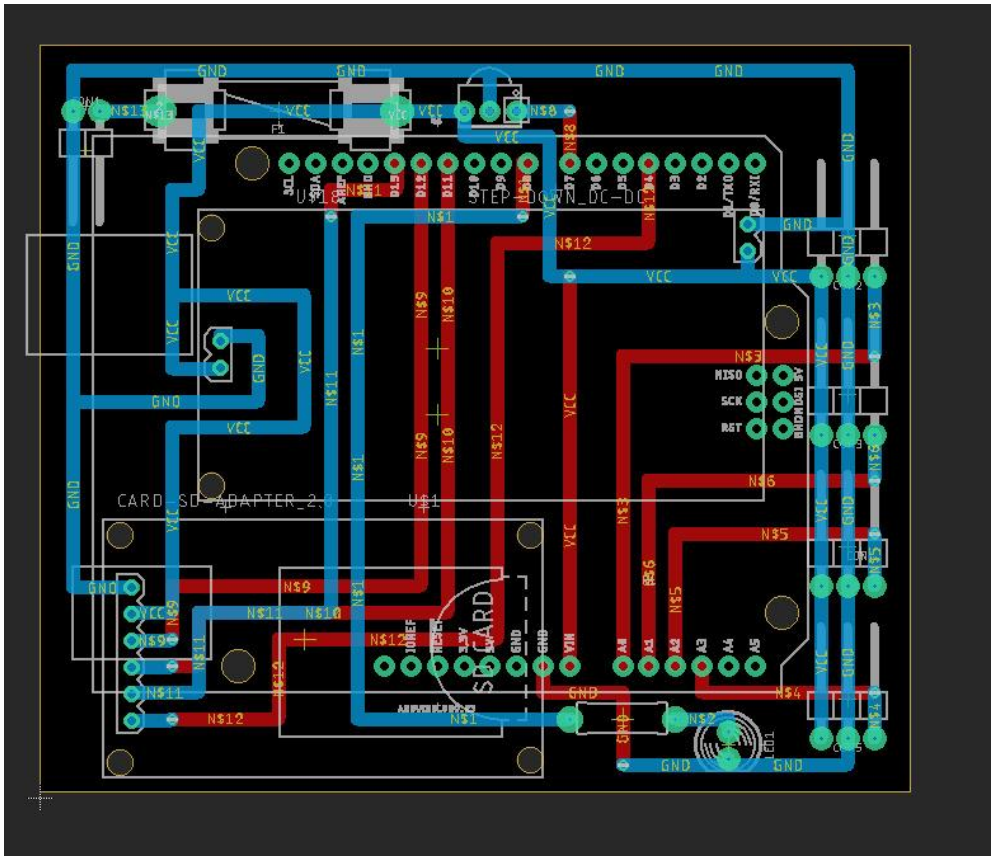


Figura 29 - Posicionamento final das peças e de todas as ligações para impressão da PCB.

Peças Complementares Soldadas à PCB

Nesta PCB são necessários certos equipamentos em específico, mas de momento só estão enunciados para uma compreensão geral do sistema, as suas funções estão também explicadas numa fase mais avançada deste capítulo. São utilizados conectores de 2 e 3 pinos

e as respetivas *shells*, terminais para ligar fichas às *shells*, *headers*, resistências, díodos, um fusível de 4A, um sensor infravermelhos, um interruptor, um LED verde e um cartão de leitor micro SD. Todos estes componentes estão apresentados na Figura 30 com as respetivas designações através de cores na Tabela 4 ao lado. Além de todos estes componentes eletrónicos também está ligada à PCB a placa Arduino.

Conector 2 pinos - Alimentação
Conector 3 pinos - Sensores
Shells 2/3 pinos
Headers
Resistência
Díodo
Fusível 4A
Sensor Infravermelhos
Interruptor
LED verde
Leitor cartão micro SD

Tabela 4 - Designações dos componentes por cores, de acordo com a imagem ao lado.

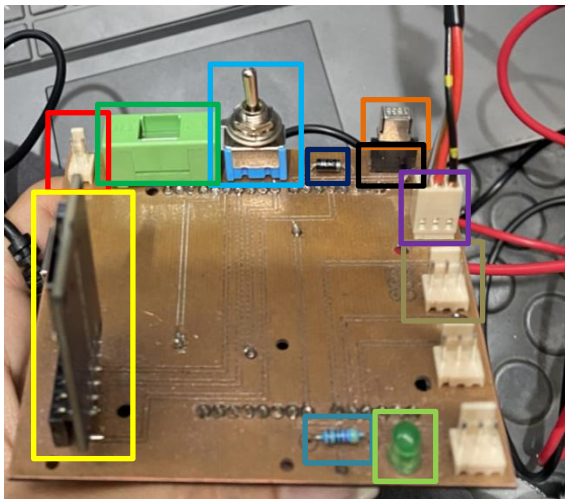


Figura 30 - Localização dos componentes na PCB por cores.

3.1.4 *Software Microsoft Excel - Škoda Datasheet*

De maneira a conseguir transportar, para uma posterior análise, todos os dados lidos pelos sensores e guardados pela placa Arduino, é criado um documento Excel.

O objetivo, com as folhas de Excel utilizadas, é criar um género de sistema de análise de dados direcionado para o *Suspension Travel System* (STS), onde seja possível observar com clareza os dados retirados, obtendo conclusões acerca da performance do carro. Neste sistema de análise está pensado dividir-se cada campo por *back-end's* e *front-end's*, no qual os front-ends dispõe de todas as componentes mais ilustrativas, tais como gráficos, imagens, principais variáveis, entre outros. Nos *back-ends* encontram-se as respetivas equações usadas para o tratamento dos dados, os dados retirados dos sensores diretamente, os novos dados devidamente tratados através das equações, informações relevantes acerca dos sensores, entre outros.

Podem ser observados, nas figuras seguintes, Figura 32 e Figura 31, exemplos das folhas criadas de *front* e *back-end*, respetivamente, ainda sem *data*, onde se observam os dados finais deste projeto e se retiram as devidas conclusões. Estas folhas podem ser utilizadas tanto para situações de teste em asfalto como em terra de momento, pois como o sistema é testado em casos extremos que se aproximam a troços de terra, os troços de asfalto nunca atingem valores tão altos de curso de amortecedor. Numa fase posterior deste projeto, é ponderada a análise mais ao fundo outras diferenças de *Setup* que se poderiam correlar e diferenciar, nesse caso, os dois tipos de troço distintos num novo Excel.

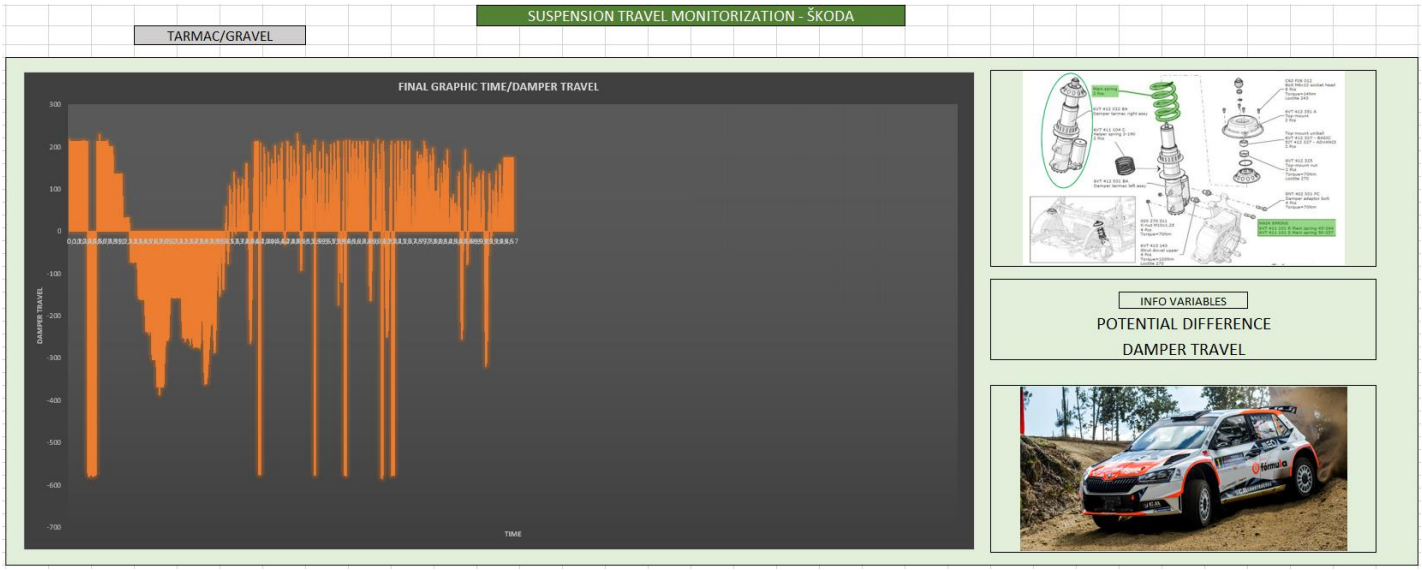


Figura 32 - *Front-End* do Excel para visualização dos gráficos finais.

Figura 31 - *Back-End* visível onde se encontram informações referentes aos gráficos secundários e ao sensor em estudo.

3.2 Mapa Geral de Tráfego dos Dados

Como é possível observar na Figura 33, o início de trajetória dos dados entre componentes do sistema é, claramente, aquando da ativação de leitura das diferentes diferenças de potencial no sensor de posição angular escolhido. De seguida, esses dados serão transportados para a placa Arduino, que os armazenará durante todo o percurso a cada 10 ms no cartão micro SD. Numa fase seguinte pós teste, esses dados são retirados através da ligação da placa Arduino ao computador, que não é efetuada diretamente necessária visto que foi adicionado um leitor de cartões micro SD que gera automaticamente um documento de texto, exibido uma vez que inserido o macro SD no computador. Posteriormente, é então aberto o ficheiro Excel com as devidas equações e efetua-se um *upload* dos dados do documento .txt dado pelo código do Arduino. Assim é possível observar todos os dados recolhidos e tratá-los como for mais conveniente pelo o/a engenheiro/a.

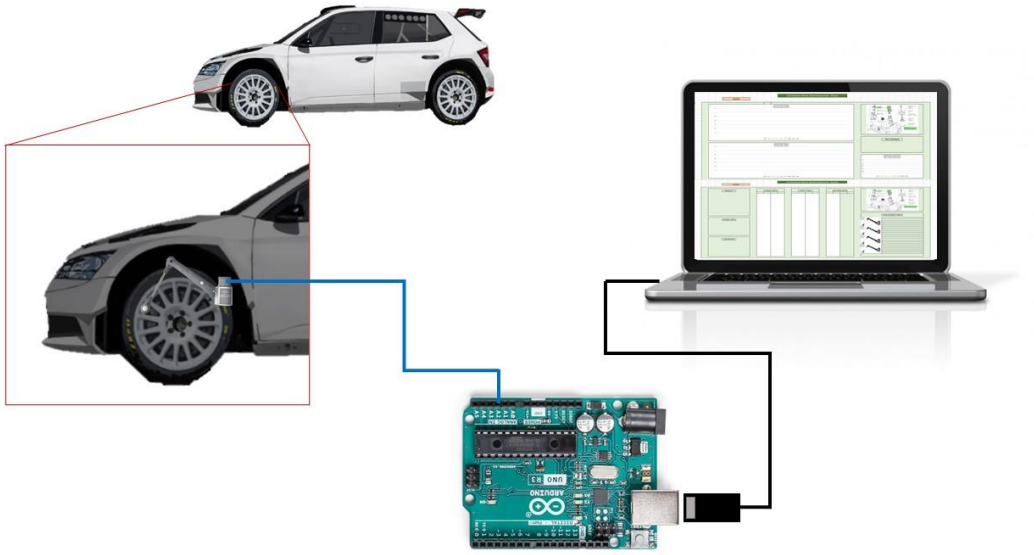


Figura 33 - Mapa geral do tráfego de dados

3.3 Ligação Entre Componentes

3.3.1 Explicação Geral da Interligação dos Componentes

Para começar, há uma ativação do sistema via infravermelhos – um comando ativa um sensor IV ligado à placa, que vai iniciar o registo dos valores e guardar na memória do micro SD. Aquando dessa ativação via comando infravermelhos, um LED verde acenderá para indicar

de forma visual que o sistema já está a registar. Durante o percurso de um troço serão retirados e guardados dados da diferença de potencial dos 4 sensores em paralelo, todos a funcionar no sistema como um mecanismo de 2 barras biela-manivela. Esses dados, mais tarde, serão transferidos e trabalhados através de equações calculadas numa fase mais avançada do projeto, de forma a informar do trabalho/posição do amortecedor nos momentos exatos dos registos, para posteriormente ser realizada uma correlação de resultados e uma análise final.

Quando o carro terminar a passagem, o comando ordenará ao sistema para parar a recolha de dados, o que fará desligar o LED verde, indicando assim que o sistema parou o registo.

Existe ainda outra funcionalidade adicionada ao sistema, que é a de um interruptor manual que deverá ser atuado quando se quiser desligar o sistema por completo, de forma a garantir a integridade de todos os componentes.

3.3.2 Carro-Sensor

Após várias tentativas e estudos de onde poderia ser colocado o sensor (Sub Capítulo 4.1.1.) esta foi a única solução viável e que permitisse que o movimento dos braços fosse executável. Este é então montado a um suporte que, por sua vez, se suporta num parafuso que liga o braço ao *charriot*, representado na Figura 34.

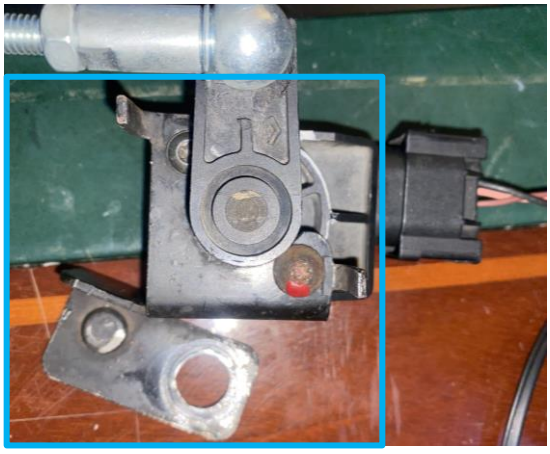


Figura 34 - Suporte adicionado ao sensor que liga o mesmo ao *charriot*.

Escolhido o ponto fixo, é necessário pensar em como se pode ligar o segundo braço, constituído por um ferro com rosca em espiral com duas rótulas nas extremidades. Escolhe-se assim um suporte com uma cavidade cilíndrica que abraça o braço e não permite nenhum tipo de movimento além do linear, como se pode observar na Figura *35*.

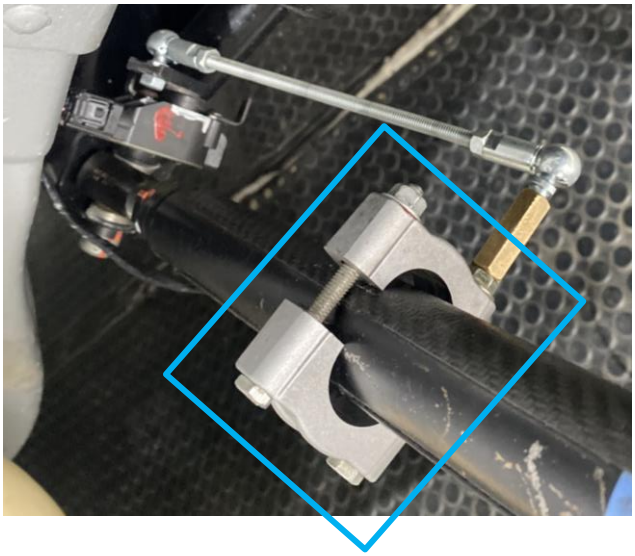


Figura 35 - Suporte com cavidade cilíndrica que suporta o braço maior do mecanismo no braço de suspensão.

De seguida e testando a linearidade do sistema, verifica-se que este mantém o movimento e a robustez pretendidos, resultando na colocação final, disposta na Figura 36.

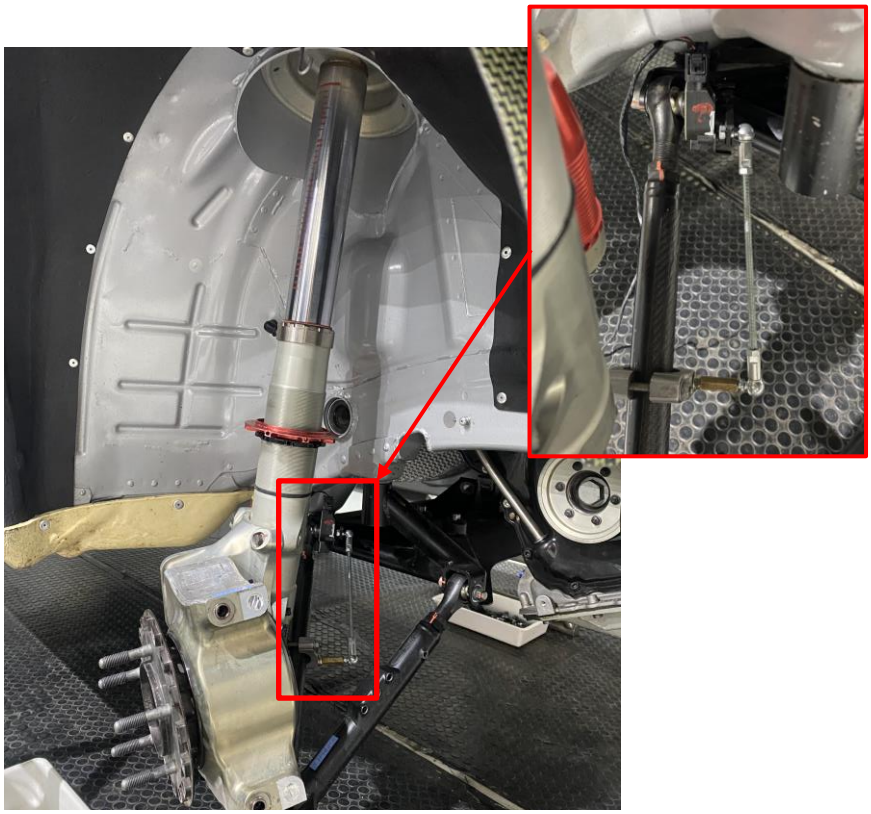


Figura 36 - Ilustração da representação final do mecanismo no STS.

3.3.3 Sensor-Placa

Relativamente à ligação do sensor à PCB, este conecta-se através de três pinos principais: o *Ground*, o VCC (+5V) e o *Output Signal*, tal como a Figura 37.

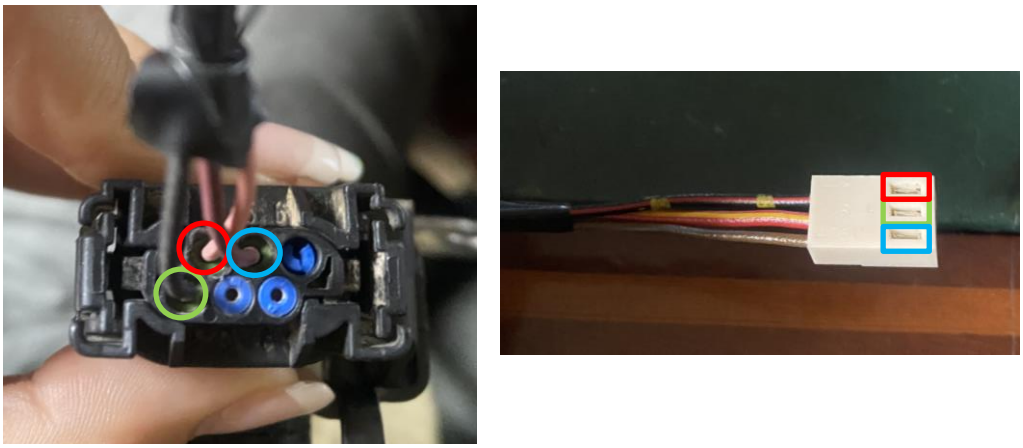


Figura 37 - Ligações de 3 pinos do sensor. Extremidade que liga ao sensor (esquerda); Extremidade que liga à PCB (direita); Pino 1 – *Ground*, Pino 4 – *Output Signal*, Pino 6 – 5V.

A *shell* onde foram inseridos os cabos previamente estendidos (daí as cores da cablagem não serem as originais) vai ser ligada a um dos quatro conectores de três pinos soldados na placa, como ilustra a Figura 38.

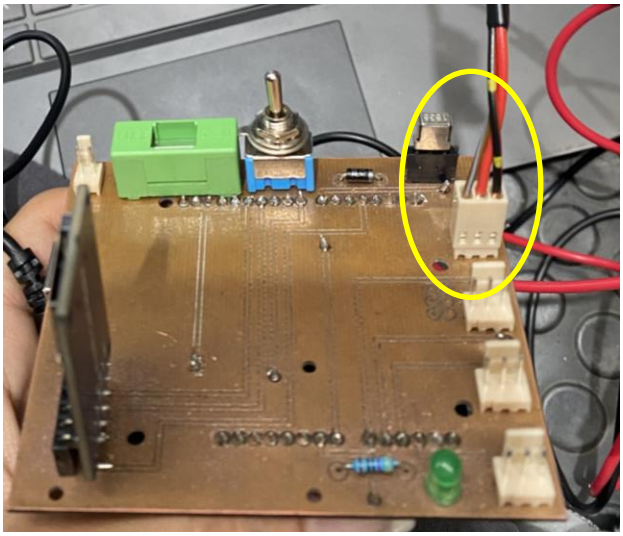


Figura 38 - *Shell* com os três cabos ligada à placa.

3.3.4 Arduino IDE - Arduino Placa

De maneira a fazer uma simulação prévia em Arduino, é utilizada uma ferramenta *online* da *Autodesk*, o *TinkerCAD*. Trata-se de um programa CAD gratuito e intuitivo, que recentemente inseriu uma expansão para incluir circuitos com o Arduino na sua capacidade de conceção. O *TinkerCAD Circuits* permite criar e programar virtualmente projetos Arduino sem a necessidade de montar um *hardware* físico[21]. No entanto, como cenário de testes primordiais e para ter certezas do que se quereria colocar ou não na PCB, monta-se também um *hardware* físico.

Na Figura 39 está apresentado o esquema final do projeto em TinkerCAD, o código que resultou desta simulação (explicado no Sub Capítulo 3.4.) serve como base para aprofundar e adicionar código para outros componentes, como o leitor de cartões SD.

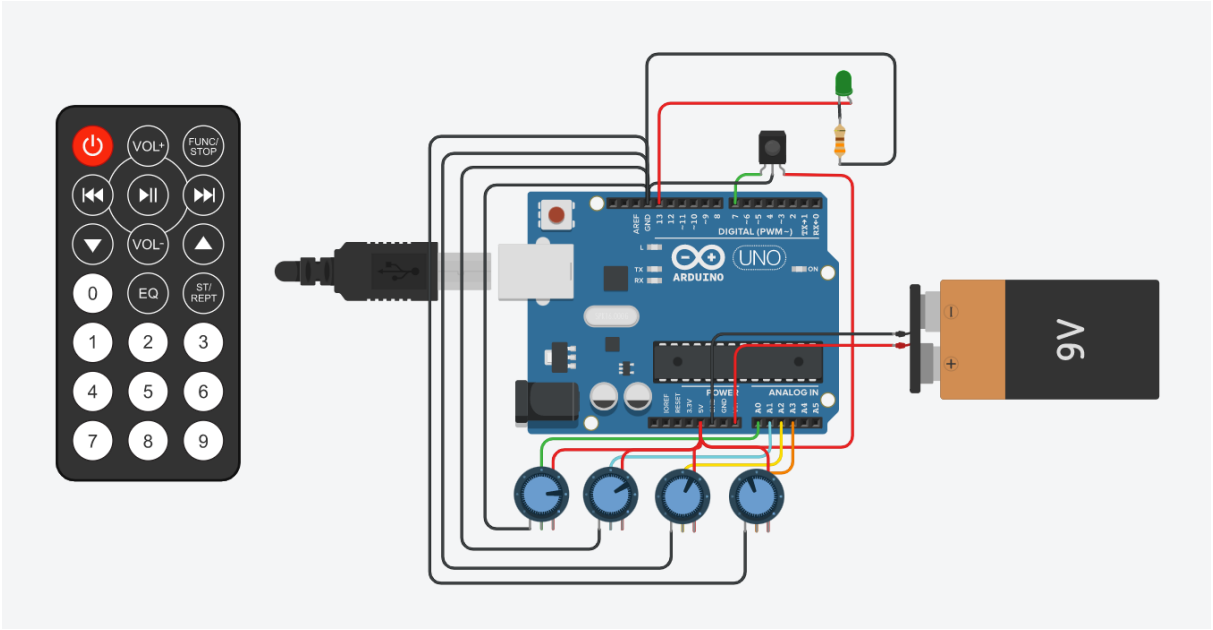


Figura 39 - Esquema final do projeto simulado em *TinkerCAD*.

3.3.5 Arduino Placa-Excel

O código final tem como objetivo comandar a placa Arduino do sistema e a PCB, a respetiva ordem de acontecimentos e possíveis respostas. Este vai registar a informação recolhida dos sensores no cartão micro SD e transportar, por ordem de recolha, para um Excel, Excel esse previamente preparado com equações e fórmulas que tratam e cruzam as informações dos sensores de posição angular com as informações-padrão recolhidas numa

fase anterior. Estas informações-padrão resultam de um processo de iterações onde são recolhidos, de 10 em 10 mm de curso de amortecedor, um valor correspondente de diferença de potencial dada por um sensor, resultando assim numa equação-padrão que permite identificar a que curso corresponde cada diferença de potencial dada nos testes práticos.

Nesse seguimento, criado um documento de texto, Figura 41, a mando do código Arduino IDE, este é exportado para o Excel de análise de dados, através da ferramenta do Excel apresentada na Figura 40.

O objetivo final é obter um gráfico final de variação de curso do amortecedor por diferença de potencial, aquando inserido o cartão no computador e aberto o Excel.

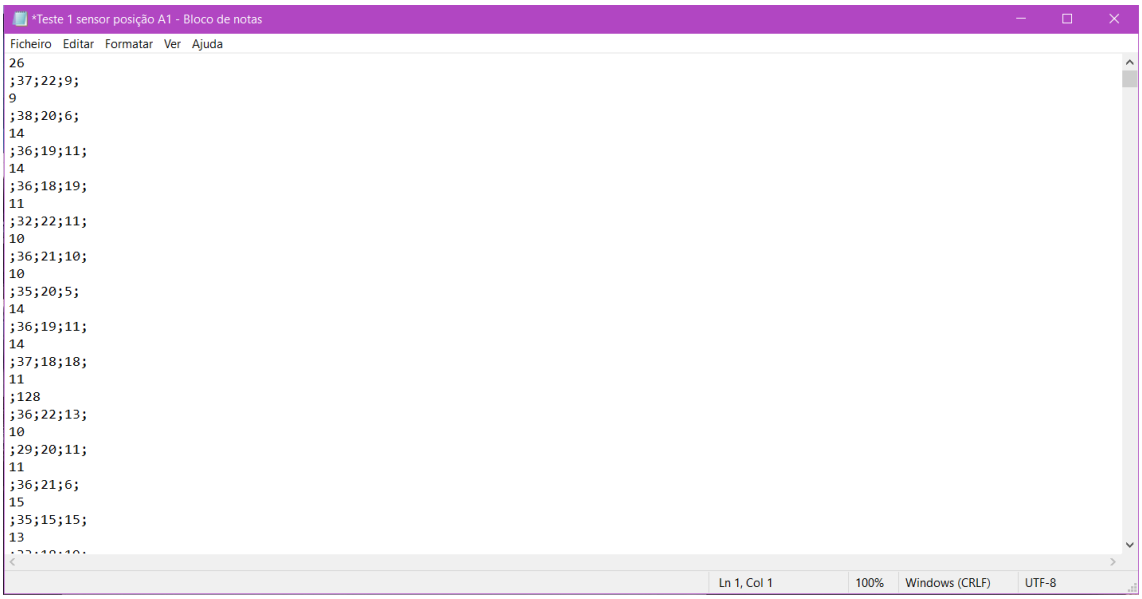


Figura 41 - Documento em formato .txt criado pelo código em Arduino IDE com os dados retirados dos sensores.

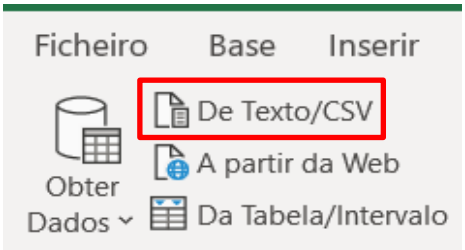


Figura 40 - Ferramenta do Excel que transporta os dados do ficheiro .txt para o ficheiro criado para *back-end* em Excel.

3.4 Código em Arduino

Após efetuado um estudo desta ferramenta e ao código que a compõe, é obtido primeiramente um código base em simulação de *TinkerCAD*, presente no ANEXO 3, que permite o funcionamento correto do sistema num teste *online*. A partir desse código são

efetuadas alterações e junções, como por exemplo uma parte de código que ordena ao Arduino para registar os dados no cartão de memória. Esse código que serve como auxílio para obter o código final está presente no ANEXO 2. Poder-se-ia questionar a razão de adicionar um leitor de cartões de memória à PCB, visto que o Arduino possui já memória *Electrically-Erasable Programmable Read-Only Memory* (EEPROM). Analisando os requisitos que estavam a ser pedidos à placa Arduino, no qual este armazenar dados a cada 10 ms, e efetuando os devidos cálculos, observa-se que este não tem capacidade para armazenar dados de quatro sensores em paralelo, durante um período de 30 minutos em média (este espaço de tempo é uma estimativa pensada para casos extremos).

Para corresponder a esses requisitos, são necessários 2 bytes (cada pino com a resolução de 10bits. Cada pino, ao ler 2 bytes são necessários 2*4=8 bytes p/ leitura. A cada segundo, efetuam-se 100 leituras e, num minuto, 6000. Ou seja, em 30 minutos, são lidas 180 mil leituras. Portanto é necessário 180000*8 bytes, que dá um total de 1.44MB. Este valor ultrapassa consideravelmente a capacidade de memória da EEPROM, que corresponde a 1kB, daí a necessidade de adicionar esta componente extra para promover um sistema mais preciso em resultados e flexível na sua utilização.

De forma a compreender todas as partes fulcrais do código deste projeto, este está analisado por partes.

Numa primeira fase, são apresentadas as bibliotecas, que possuem funções desenvolvidas especificamente para executar tarefas personalizadas.

#include <SPI.h>

Esta biblioteca permite uma comunicação com os dispositivos *Serial Peripheral Interface* (SPI), tendo o Arduino como dispositivo controlador. Neste caso esta biblioteca auxilia na comunicação com o cartão SD [22][23].

#include <SD.h>

O SD.h permite a leitura e escrita em cartões SD. Quando um cartão de memória SD é ligado à interface SPI da placa Arduino é possível criar ficheiros e ler/escrever nos mesmos[24].

```
#include <IRremote.h>
```

O *IRremote* permite receber ou transmitir códigos de controlo remoto por infravermelhos e que o projeto seja controlável por um comando, pelo que neste caso tem de existir devido a haver um sensor infravermelhos no sistema e o seu respetivo comando[25].

```
#define RECV_PIN 7
#define LED_ON 8
```

Esta função permite ao programador dar um nome a um valor constante antes do programa ser compilado, sendo definidos como constantes neste caso o sensor infravermelhos e o LED verde[26].

```
IRrecv irrecv(RECV_PIN);
decode_results results;
File myFile;
```

Esta parte remete para o IRremote, pois este atua como 2 bibliotecas, uma para enviar e outra para receber. Neste caso vai receber, pois é mais fácil começar pela receção primeiro para depois encontrar o código mais rápido[27].

```
// previous time for the tasks depending upon time.
unsigned long prevTime_T1 = millis();
```

As *unsigned longs* são variáveis de tamanhos maiores para armazenamento de números, neste caso o tempo, e armazenam 32 bits (4 bytes), além de não armazenarem números negativos[28].

```
// time intervals for the tasks
long interval_T1 = 10; // read every 10 miliseconds
bool running = false; //flag to start reading values
```

Aqui, como está referido no texto auxiliar do código, é o local onde se define o intervalo de tempo que as tarefas devem ser executadas, sendo que, logo de seguida, dá-se permissão para os valores começarem a ser lidos.

```
int sensor0Value;// array with values of the sensor
```

```
int sensor1Value;// array with values of the sensor
int sensor2Value;// array with values of the sensor
int sensor3Value;// array with values of the sensor
```

Um *array* é uma coleção de variáveis que são acedidas com um número de índice. Aqui são inseridos *arrays* de memória para cada sensor registar os dados que são recolhidos.

```
void setup()
{
```

A função *void setup* é uma função que se cria no início de cada código. Dentro dos parênteses "**{**" vai-se encontrar o código que se pretende executar assim que o programa comece a correr[29].

```
pinMode(LED_ON, OUTPUT);
 // Open serial communications and wait for port to open:
 Serial.begin(9600);
 while (!Serial) {
  ; // wait for serial port to connect. Needed for native USB port only
 }
```

A função pinMode() é utilizada para configurar um pino específico para se comportar quer como *input* quer como *output*. Diz-se que os pinos configurados como *output* com a função *pinMode()* estão num estado de baixa impedância. Isto significa que eles podem fornecer uma quantidade substancial de corrente a outros circuitos. As saídas Atmega podem fornecer fonte até 40 mA para outros dispositivos/circuitos, sendo que isto é corrente suficiente para iluminar um LED[30]. Como se pretende ativar durante todo o processo o LED verde, utiliza-se essa função.

Já a função seguinte, a *Serial.begin*, define a taxa de dados em bits por segundo para uma transmissão de dados em série[31].

```
 Serial.print("Initializing SD card...");
 if (!SD.begin(4)) {
  Serial.println("initialization failed!");
  while (1);
```

```
 }
 Serial.println("initialization done.");
```

Nesta situação, o comando *serial.print* manda para a porta serie a seguinte frase: "*Initializing SD card*..." Depois incia o *SD card*, e caso o tenha conseguido fazer manda para a porta serie "*initialization done".* Caso contrário envia "*initialization failed*!" e não faz mais nada. O *while (1);* vai "prender" o processador neste momento, este não faz mais nada.

O número 4 no *SD.Begin* representa o pino que está ligado ao *chip select*. Esta função vai retornar 1 se tiver sucesso, caso contrário retorna 0[32].

```
 // open the file. note that only one file can be open at a time,
 // so you have to close this one before opening another.
 myFile = SD.open("test.txt", FILE_WRITE);
```

Neste momento, é efetuado um comando que pede ao Arduino para abrir à vez a cada registo de dados o ficheiro de texto no cartão SD.

```
 // if the file opened okay, write to it:
 if (myFile) {
  Serial.print("Successfully open.");

 } else {
  // if the file didn't open, print an error:
  Serial.println("error opening test.txt");
 }
 irrecv.enableIRIn();
}
```

Nesta secção, está-se a pedir ao código que abra um ficheiro de texto (.txt) no cartão SD que leia os valores dos sensores, no entanto é necessário verificar se esse ficheiro foi, ou não, aberto, informando o programador através do texto indicativo dentro das funções *Serial.print()* e *Serial.println().*

A função *irrecv.enableIRIn()* começa aqui o processo de receção da informação[33].

```
void loop()
```

```
{
 unsigned long currentTime = millis();
```

A função *void loop()* é uma função que executa os comandos que nela são colocados infinitamente, enquanto a placa estiver ligada, permitindo ao programa mudar e responder a essas mudanças[34].

Utilizando a *millis()* com a função *unsigned long*, esta imprime na porta de série o número de milissegundos passados desde que a placa Arduino começou a executar o próprio código[35].

```
 if (running){
  if (currentTime - prevTime_T1 > interval_T1) {
```

Running é um bit que fica a 1 quando for para começar a recolha dos dados.

Assim quando este estiver a 1 e se a condição abaixo for verdadeira irá executar o código dentro das chavetas que servem para recolher os dados.

A condição seguinte serve para se saber se já passaram os 10 milissegundos e se nesse caso pode recolher dados novamente. Assim, se a diferença entre o tempo atual menos o tempo da última leitura for igual ou superior a 10 ms, o código vai efetuar uma nova leitura.

Após toda a preparação para a leitura dos dados, é nesta parte de código que realmente se começa a efetuar a recolha, a apresentação dos valores no serial monitor e a escrita automática no ficheiro de texto. Como os quatro sensores funcionam da mesma maneira e os dados estão a ser recolhidos em paralelo, a explicação para o código de um é igual para os restantes. De notar que, apesar de se ter optado no final por efetuar o teste com um dos sensores apenas, o código está preparado para a implementação dos quatro.

```
    sensor0Value = analogRead(A0);
```

A função *AnalogRead()* lê o valor de um pino analógico específico. A placa Arduino possui um conversor analógico-digital 10 bits de 6 canais. Isto significa que este irá mapear tensões entre 0 e a tensão operacional (5V neste caso) para valores inteiros entre 0 e 1023. No Arduino UNO, por exemplo, permite uma resolução entre leituras de: 5 volts / 1024 unidades, ou .0049 volts (4.9 mV) por unidade.

```
Serial.print(sensor0Value);
```

```
Serial.print("//**//");
```

Nesta situação, é dado o nome de *sensor0Value* ao sensor analógico 0, que posteriormente receberá uma ordem para ser escrito o valor dado entre 0 e 1023 no serial monitor [36] e que terá à frente desse valor este género de separador **"//**//"** para espaçar os valores dos quatro sensores.

```
myFile = SD.open("test.txt", FILE_WRITE);
// if the file opened okay, write to it:
if (myFile) {
  myFile.print(sensor0Value);
  myFile.print(";");
  // close the file:
  myFile.close();
  }
```

Mais acima, na parte de código do *setup(),* cria-se um novo ficheiro com o *SD.open()* chamado "*test.txt*". Nomeia-se a instância do ficheiro aberto como "*myFile*". O "*FILE_WRITE*" que aparece no *SD.open()* permite o acesso de leitura e escrita ao ficheiro. Se por acaso um ficheiro "*test.txt*" já estivesse no cartão, esse ficheiro é aberto.
Uma vez aberto, utiliza-se o *myFile.println()* para escrever uma *string* no cartão de memória. Depois de escrever o valor, ordena-se ao código que feche o ficheiro, com a função *myFile.close()*[37]. Efetuar este fecho e abertura constantes do ficheiro aumenta consideravelmente o tempo de execução desta tarefa no total, pelo que em trabalhos futuros deste projeto isso será algo a ter em conta e melhorar.

Assim, executa-se o mesmo código para os três sensores restantes, apresentado de seguida.

```
sensor1Value = analogRead(A1);
Serial.print(sensor1Value);
Serial.print("//**//");
myFile = SD.open("test.txt", FILE_WRITE);
```

```
// if the file opened okay, write to it:
if (myFile) {
   myFile.print(sensor1Value);
   myFile.print(";");
   // close the file:
   myFile.close();
   }

sensor2Value = analogRead(A2);
Serial.print(sensor2Value);
Serial.print("//**//");
 myFile = SD.open("test.txt", FILE_WRITE);
// if the file opened okay, write to it:
if (myFile) {
   myFile.print(sensor2Value);
   myFile.print(";");
   // close the file:
   myFile.close();
   }

sensor3Value = analogRead(A3);
Serial.print(sensor3Value);
Serial.println("//**//");
 myFile = SD.open("test.txt", FILE_WRITE);
// if the file opened okay, write to it:
if (myFile) {
   myFile.print(sensor3Value);
   myFile.println(";");
   // close the file:
   myFile.close();
   }
```

```
  digitalWrite(LED_ON, HIGH);
    prevTime_T1 = currentTime;
   }
 }
 else
  digitalWrite(LED_ON, LOW);
```

A atribuição *prevTime_T1 = currentTime* está dentro do bloco de código que é executado caso tenham passado 10 milissegundos, assim é necessário ser executada nesse bloco para se saber em que instante se começou a executar esse bloco (leitura de dados), de modo a ser possível saber se depois já passaram 10 milissegundos desde a última leitura ou não.

A função apresentada, a *digitalWrite()*, aciona um valor *HIGH* ou *LOW* num pino digital.

Como o pino é configurado como saída (*OUTPUT*) com a função *pinMode()*, a tensão é acionada para o valor correspondente: 5V para o valor HIGH, 0V (ou *ground*) para LOW[38].

Neste momento, caso o sensor não esteja a recolher mais dados, o LED verde entra na posição LOW, ou seja, desliga.

```
 if(irrecv.decode(&results)){
   do_response(results.value);
   Serial.print(results.value);
   irrecv.resume();
 }
}

/*
 respond to specific remote-control keys with different behaviors
*/

void do_response(int key)
{
```

O *do_response* é quando se dá uma resposta ao código, acionando um botão de um comando especíico.

A condição if *(irrecv.decode(&results))* devolve 0 se não existirem dados prontos, ou 1 se os dados estiverem prontos.

O *irrecv.resume();* reinicia o código do estado de interrupção e receber o próximo valor.

Os tipos de dados "*int*" servem para armazenar números. No caso do Arduino Uno, um "*int*" armazena um valor de 16 bits (2 bytes)[39]. Estes 2 bytes são utilizados para inserir os números e as letras do comando de botões que liga por infra vermelhos, na fase seguinte.

```
switch (key)
{
 case 25245:  // FUNC/STOP turns off/on the reading of the analog ports //16 736 925
  running = !running;
 // in order to check if is reading
  if(running)
   digitalWrite(LED_ON, HIGH);
  else
   digitalWrite(LED_ON, LOW);

  Serial.println("FUNC/STOP");
  break;
```

O comando envia 3 Bytes mas apenas se vai usar os 2 menos significativos (cada número/letra em hexadecimal corresponde a 4bits, logo 2 números/letras serão 1byte (8bits)). Como no código, tipicamente, escreve-se sempre em decimal, colocou-se o número 25245.

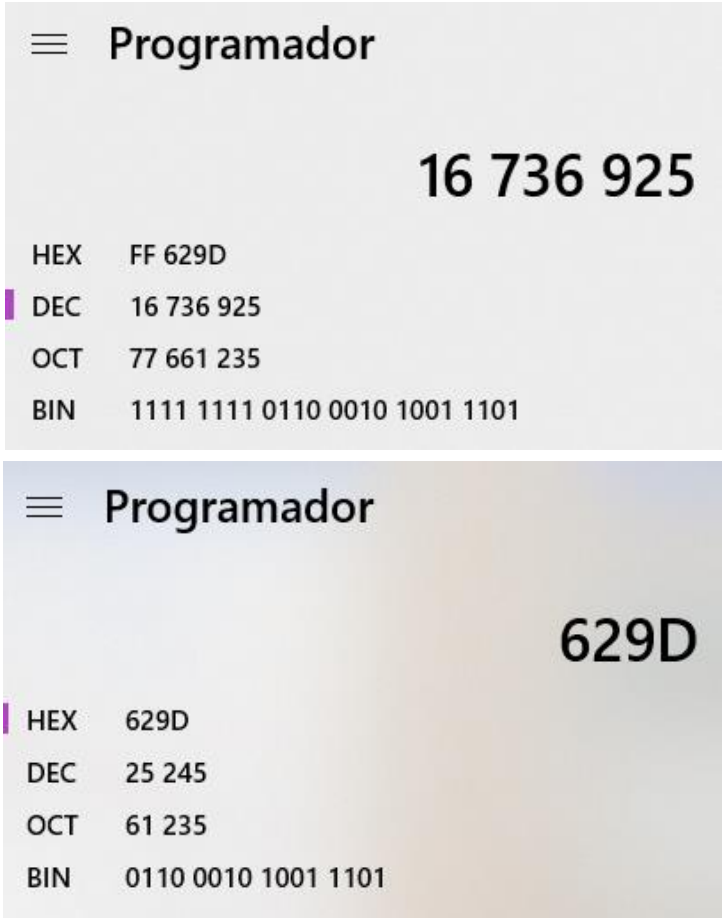


Figura 42 - Passagem do código do botão de hexadecimal para decimal.

Esta parte recorre à referência correspondente do comando IV (FF629D passada para 629D que corresponde ao valor 25245 no código) (Figura 42) e, caso seja pressionado, efetua uma ação, sendo que neste caso existe um botão para correr o sistema, o *FUNC/STOP*, que começa a registar os dados na memória do cartão SD. Para trabalhos futuros, poderia ser uma opção adicionar mais botões além do existente, como por exemplo um botão RESET que reiniciaria a memória do cartão SD para futuras utilizações, não acumulando memória desnecessária após cada passagem do carro.

```
   default:
    {
      Serial.print("Key ");
      Serial.print(key);
      Serial.println(" not programmed");
    }
   break;
 }
}
```

Por fim, no *switch case* acima, caso não exista um caso para o botão carregado, este vai fazer o que está dentro do *default* do *switch case.* Neste caso vai realizar *print*, ou seja, escrever, para o *serial monitor* de *"Key"* o número que a tecla que foi clicada corresponde e dizer que não tem nenhuma função associada a esse botão, ou seja, *"Not progammed".*

A RETER DESTE CAPÍTULO

A partir deste capítulo é possível compreender qual o trajeto ideal, e também real, do tráfego dos dados, estudando assim os componentes que compõe este projeto e as respetivas interfaces entre eles, analisando cada etapa tanto de forma individual como num conjunto final. Neste capítulo é também ilustrado o código utilizado no sistema de recolha de dados e o caminho para lá chegar.

4. Testes Práticos e Análise de Resultados

Neste capítulo são apresentadas todas as fases do desenvolvimento prático deste projeto de dissertação, nomeadamente a montagem do sistema físico, os testes realizados, o processo de construção da *dashboard* em Excel que permite tecer conclusões acerca do comportamento dos amortecedores, além de uma reflexão acerca da robustez do sistema numa perspetiva geral. De forma a ajudar a tecer conclusões e analisar numa perspetiva geral todo o sistema de recolha de dados, é também apresentada uma análise SWOT. Através desta análise procede-se a um estudo de possíveis melhorias.

4.1 Montagem do Sistema Físico

Para a montagem do sistema físico, são necessários alguns testes prévios no que toca a organizar os possíveis espaços livres, de maneira a conseguir colocar todos os equipamentos. São efetuadas várias medições e propostas várias opções, de forma a analisar qual a localização mais vantajosa tanto para as peças em si, como para as ligações elétricas. Este é o subcapítulo que aborda tanto essas questões como também a fase final da análise dos dados recolhidos, nomeadamente a curva padrão diferença de potencial-altura de curso, a Interface final entre os documentos de texto fornecidos pelo Arduino e o tratamento realizado em Excel, de forma a obter as conclusões pretendidas.

4.1.1 Opções de Colocação dos Sensores

<u>Opção 1 – Menos Viável</u>

- Colocar o sensor do lado exterior ao *charriot*, entre o chassi e a zona do *charriot* onde é montado um dos braços de suspensão, como é possível observar na Figura 43. Através do suporte em ferro pensado para o sensor, este apoia-se no parafuso que liga o *charriot* ao braço.

Figura 43 - Primeira opção de colocação do sensor entre as setas azuis.

Problemas encontrados:

- Sensor não tem espaço para ser montado nessa posição e as ligações nas extremidades ficam comprometidas;

- Braços do sensor impedem o movimento da transmissão e da ponteira de direção.

Opção 2 – Mais Viável e Escolha Final

- Colocar o sensor do lado interior do *charriot*, como é observável Figura 44. Através do suporte em ferro pensado para o sensor, este apoia-se no parafuso que liga o *charriot* ao braço de suspensão, como na situação anterior. No entanto está mais protegido e não impede o movimento de outros componentes do veículo em questão.

- Após testada essa opção, avalia-se que esta é a correta pois não existem problemas na sua colocação.

Figura 44 - Segunda opção de colocação do sensor entre as setas vermelhas.

4.1.2 Ligações Elétricas

Nesta etapa do projeto, após saber que os sensores utilizados necessitam de uma alimentação de 5V e que estes não devem exceder esse valor, é importante analisar se essa opção é ou não viável, uma vez que se não existisse algum cabo com diferença de potencial de 5V seria obrigatório adicionar à PCB um regulador de tensão. É então encontrado um cabo que corresponde às requisições de tensão, no entanto para prevenir é adicionado um fusível na PCB. Inicialmente foi ponderado colocar a cablagem do sensor, que está circundada a vermelho na Figura , pelo orifício onde passa a cablagem do sensor de pressão de óleo dos travões, ilustrado na Figura 45 à esquerda. Como este local aparenta ser mais exposto a

condições adversas, é encontrada uma opção mais flexível, perto não só da cablagem principal do carro como também de onde se pretende colocar a PCB com a placa Arduino. Como o cabo de 5V encontra-se perto desse local, este fica então definido para a passagem das ligações elétricas, representado na Figura no lado direito, circundado a azul.

Figura 45 - Cablagem do sensor.

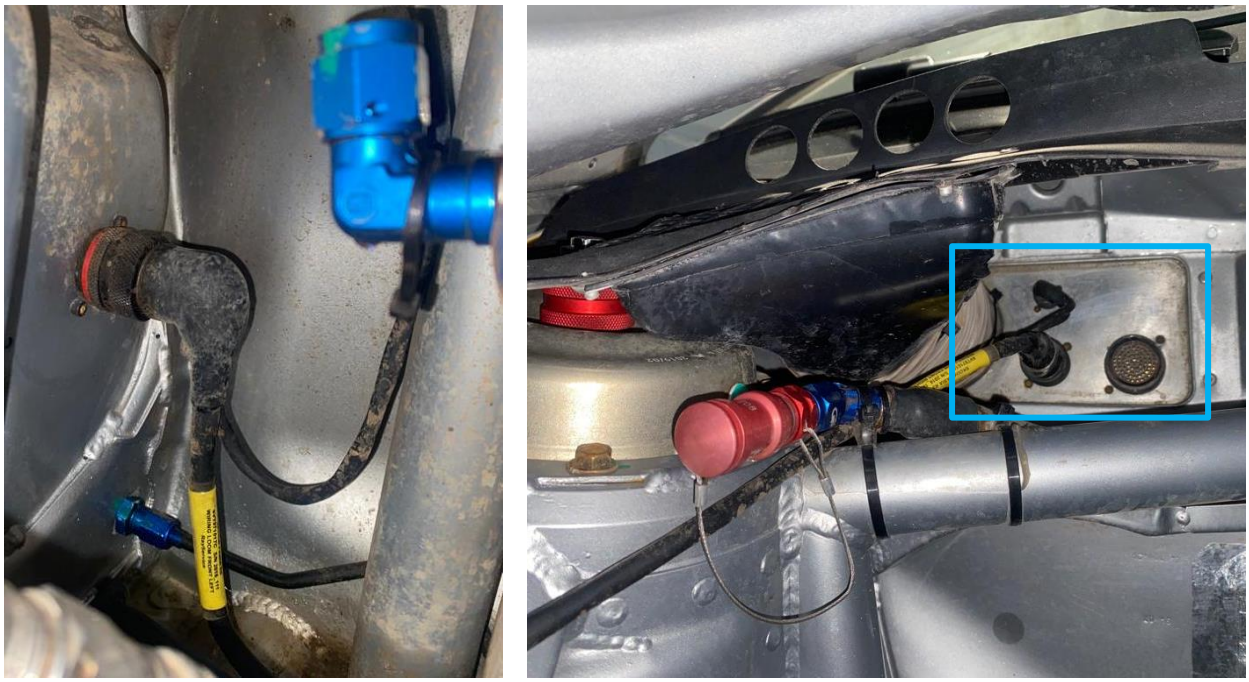


Figura 46 - Esquerda: cablagem ligar através de orifício do sensor de óleo dos travões; Direita: Cablagem ligar pela zona de cablagem principal onde ficam conectadas as principais ligações elétricas.

4.1.3 Localização Controlo do Sistema e Acionamento

Apesar do local do posicionamento do sistema e acionamento ter sido testado e terem sido analisadas opções para tal, não se viu vantagens em colocar, devido a não ter sido realizado um teste real do sensor em treino de Ralis. Dessa forma, os testes ao STS foram realizados no exterior do carro, estando este suspenso por cavaletes, de maneira a ser mais prático e funcional retirar tanto os dados das iterações, como os dados do sensor, através de uma simulação manual do movimento linear do subconjunto em estudo.

Idealmente o sítio mais adequado é dentro do carro, perto da cablagem principal e também das restantes *dashboards* do mesmo, onde é possível o navegador ativar recolha dos dados antes do início de cada percurso.

4.2 Curva Curso-Diferença de Potencial

Ao analisar que o teste efetuado em estrada teria o mesmo efeito ao ser realizado na oficina, é efetuada uma simulação manualmente no STS de forma a obter resultados semelhantes analisáveis em Excel.

Os quatro sensores estão projetados para funcionar de forma igual, em paralelo, no código apresentado no capítulo anterior. Tanto as suas ligações como os seus apoios são praticamente iguais, devido à tração do *Škoda* ser 4x4. Para facilitar todo o processo e obter resultados viáveis o mais breve possível para permitir mais do que um teste, é montado somente um dos sensores na roda dianteira direita. Os dados que são apresentados e analisados de seguida são referentes a esse sensor.

Como abordado nos capítulos iniciais, não se optou por obter equações de movimento em modelação 3D, recorrendo a um método que, apesar de ser mais moroso, permitiu obter resultados mais reais para efetuar a relação direta entre a diferença de potencial do sensor com o curso do amortecedor. Para tal, recorreu-se a iterações que possibilitaram retirar uma alta *range* de pontos que permitissem realizar uma curva de valores. De 10 em 10 milímetros e com o amortecedor comprimido até 30 mm do limite, é iniciada assim a medição dos valores até atingir 400mm de curso, altura de curso essa já definida como situação limite do amortecedor, como ilustrado nas Figura e Figura 45.

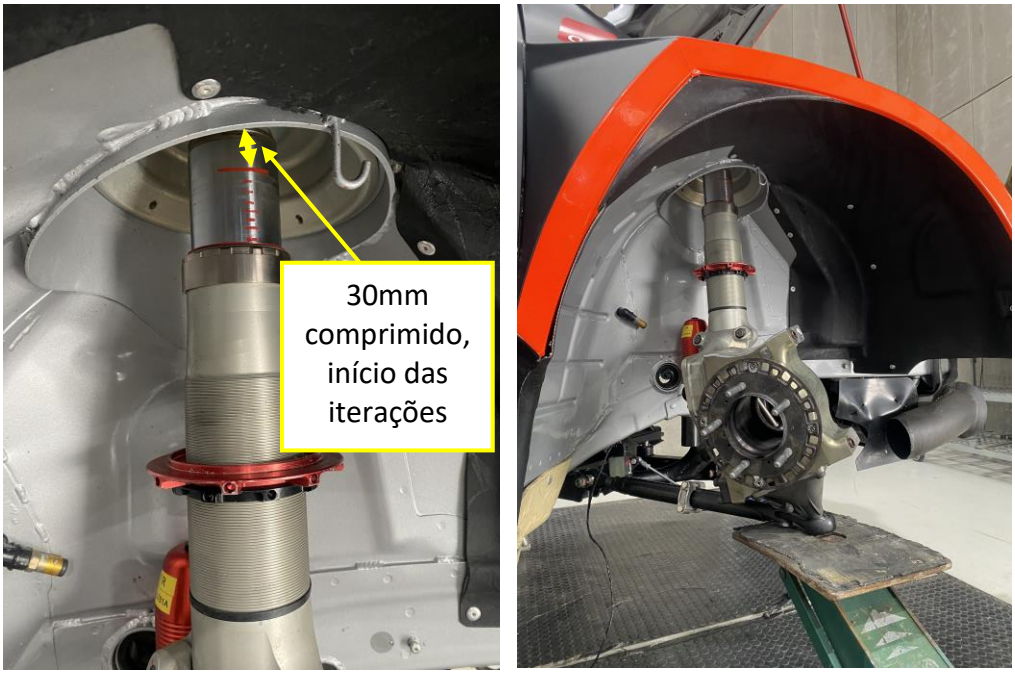


Figura 47 - Amortecedor com curso quase todo comprimido.

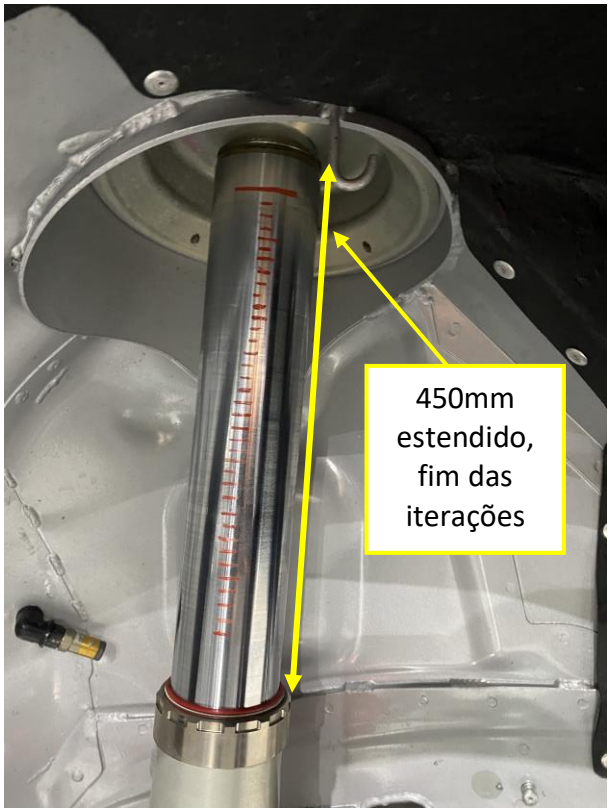


Figura 45 - Amortecedor com curso todo estendido.

Através de um multímetro, e como representado na Figura , é retirada a diferença de potencial à medida que se desce o *Suspension Travel System*, chegando à seguinte listagem de valores presentes na Figura 46.

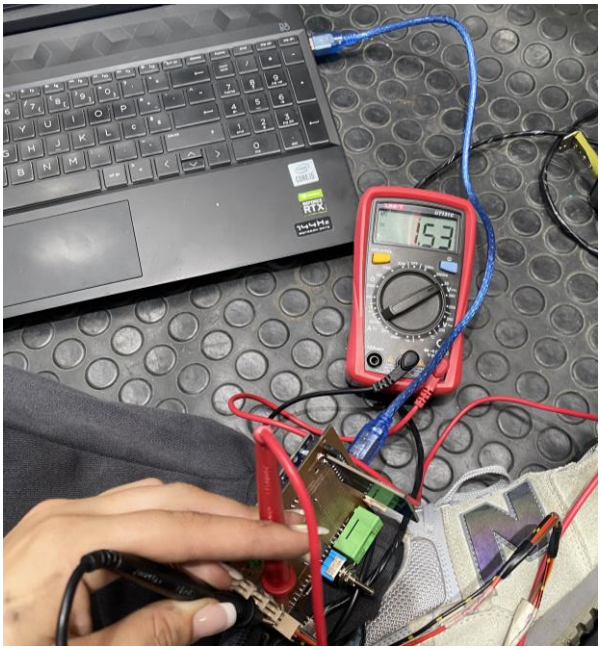


Figura 49 - Recolha dos dados das diferenças de potencial a cada 10 mm de curso, com a ajuda de um multímetro.

Dif Pot 0-5V	Damper Travel (mm)
3,12	30
3,17	40
3,22	50
3,28	60
3,31	70
3,34	80
3,39	90
3,42	100
3,47	110
3,52	120
3,56	130
3,61	140
3,67	150
3,71	160
3,75	170
3,79	180
3,86	190
3,9	200
3,95	210
3,99	220
4,06	230
4,1	240
4,12	250
4,13	260
4,12	270
4,3	280
4,34	290
4,44	300
4,47	310
4,53	320
4,56	330
4,6	340
4,66	350
4,7	360
4,75	370
4,82	380
4,82	390
4,82	400

Figura 46 - Dados retirados após iterações e respetivo gráfico padrão.

Posteriormente, com as ferramentas do Excel é possível obter uma função do gráfico que relaciona estes dois parâmetros, que permitiu posteriormente calcular o valor de curso

do amortecedor correspondente aos valores de diferença de potencial dados pelo sensor em teste. A função desse gráfico é dada por: y = 207,93x - 613,27. É possível denotar um ligeiro desvio na função final, desvio esse provocado pela decrescência de um valor, quando deveria ter um comportamento crescente (4.12V -> 4.13V -> 4.12V).

Além dessa situação, repentinamente foi registado outro valor mais dispare do que os anteriores. Poderá eventualmente ter sido um erro de leitura do sensor, uma vez que todos os valores representados foram lidos em teste mais do que cinco vezes, de maneira a não aumentar a margem de erros de caráter aleatório.

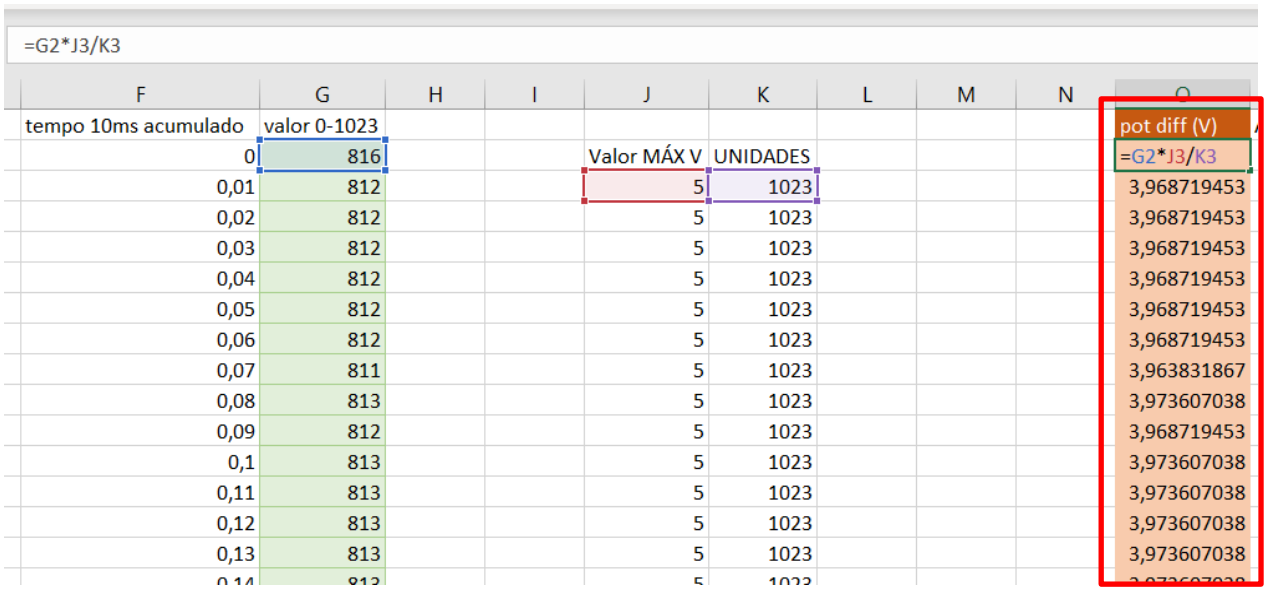
4.3 Final *DAS Dashboard*

A placa *Arduino Uno Rev3* permite mapear tensões entre 0 e a tensão operacional (5V neste caso) para valores inteiros entre 0 e 1023 unidades. Ou seja, quando se observa no *Serial Monitor* do Arduino IDE a recolha em tempo real dos dados, não se observam valores diretos entre 0 e 5 V, mas sim valores com uma range de 0 a 1023. Para configurar esses dados é necessário primeiro filtrar os valores do sensor da roda dianteira direita, o A0, que aparece sempre primeiro no ficheiro .txt. Para isso recorre-se a uma seleção de linhas e obtém-se cerca de 800 valores, como é possível observar na Figura .

	A	B	C	D	E	F	G	H
1						tempo 10ms acumulado	valor 0-1023	
2	Column1		0	0		0	816	
3	816		1	0,01		0,01	812	
4	;417;84;15;		2	0,01		0,02	812	
5	812		3	0,01		0,03	812	
6	;421;84;25;		4	0,01		0,04	812	
7	812		5	0,01		0,05	812	
8	;420;93;16;		6	0,01		0,06	812	
9	812		7	0,01		0,07	811	
10	;423;95;15;		8	0,01		0,08	813	
11	812		9	0,01		0,09	812	
12	;428;89;12;		10	0,01		0,1	813	
13	812		11	0,01		0,11	813	
14	;424;88;15;		12	0,01		0,12	813	
15	812		13	0,01		0,13	813	
16	;423;85;25;		14	0,01		0,14	813	
17	811		15	0,01		0,15	813	
18	;422;93;21;		16	0,01		0,16	812	
19	813		17	0,01		0,17	813	
20	;422;96;16;		18	0,01		0,18	812	
21	812		19	0,01		0,19	813	

Figura 51- Seleção de linhas para obter somente os valores do sensor A0.

De seguida, após realizar uma regra de três simples para saber a que valor entre 0 e 1023 corresponde o número retirado do sensor, obtém-se os valores reais de teste das diferenças de potencial, presentes na Figura 47.

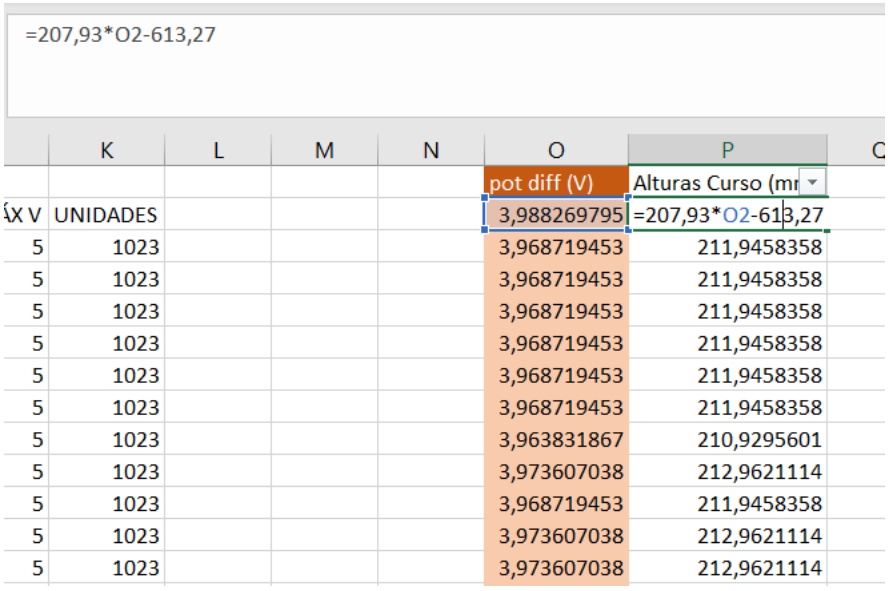


=G2*J3/K3

F	G	H	I	J	K	L	M	N	O
tempo 10ms acumulado	valor 0-1023								pot diff (V)
0	816			Valor MÁX V	UNIDADES				=G2*J3/K3
0,01	812			5	1023				3,968719453
0,02	812			5	1023				3,968719453
0,03	812			5	1023				3,968719453
0,04	812			5	1023				3,968719453
0,05	812			5	1023				3,968719453
0,06	812			5	1023				3,968719453
0,07	811			5	1023				3,963831867
0,08	813			5	1023				3,973607038
0,09	812			5	1023				3,968719453
0,1	813			5	1023				3,973607038
0,11	813			5	1023				3,973607038
0,12	813			5	1023				3,973607038
0,13	813			5	1023				3,973607038
0,14	813			5	1023				

Figura 47 - Regra de 3 simples aplicada a todos os valores do sensor A0.

Após achar os valores reais da diferença de potencial e ao calcular previamente a função da curva de comportamento padrão Diferença de Potencial-Curso do Amortecedor, permite-se assim achar os valores das Alturas de Curso, visíveis na Figura 48.



=207,93*O2-613,27

K	L	M	N	O	P	Q
				pot diff (V)	Alturas Curso (m	
ÁX V	UNIDADES			3,988269795	=207,93*O2-613,27	
5	1023			3,968719453	211,9458358	
5	1023			3,968719453	211,9458358	
5	1023			3,968719453	211,9458358	
5	1023			3,968719453	211,9458358	
5	1023			3,968719453	211,9458358	
5	1023			3,968719453	211,9458358	
5	1023			3,963831867	210,9295601	
5	1023			3,973607038	212,9621114	
5	1023			3,968719453	211,9458358	
5	1023			3,973607038	212,9621114	
5	1023			3,973607038	212,9621114	

Figura 48 - Valores finais das alturas de curso, através do gráfico padrão achado nas iterações.

Por fim, os valores das alturas dadas e o tempo acumulado em milissegundos são selecionados para efetuar o gráfico final, semelhante a um sistema de aquisição de dados habitual, observável na Figura 49.

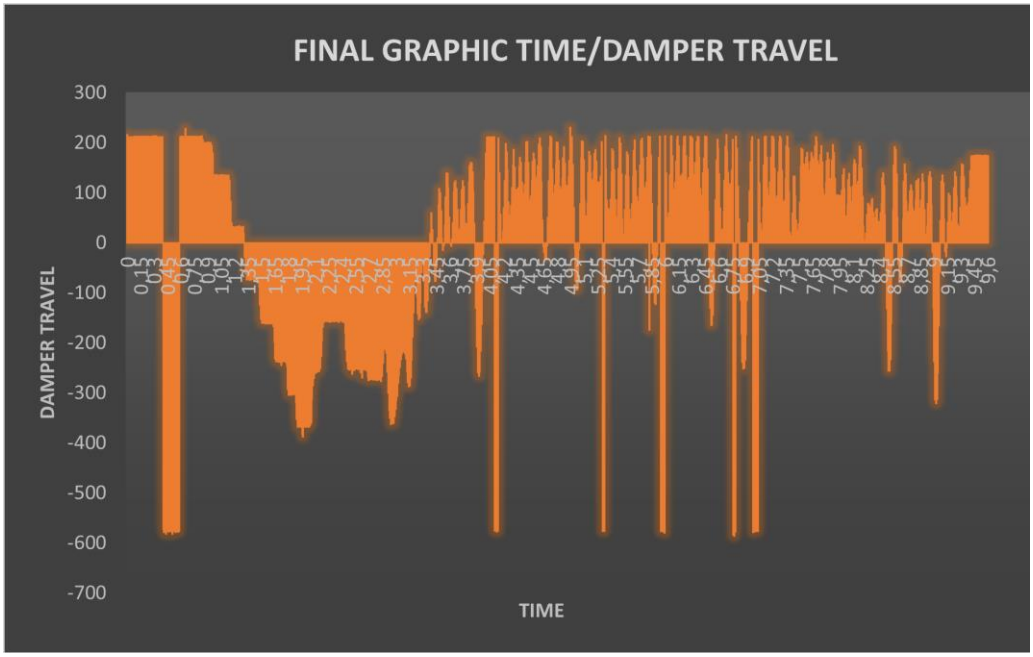


Figura 49 - Gráfico final que apresenta o curso do amortecedor num determinado espaço de tempo.

De maneira a ser mais percetível para o(a) engenheiro(a) observar e comparar os resultados em situações mais pontuais, adiciona-se também um gráfico de dispersão por pontos, como é ilustrado na Figura 50.

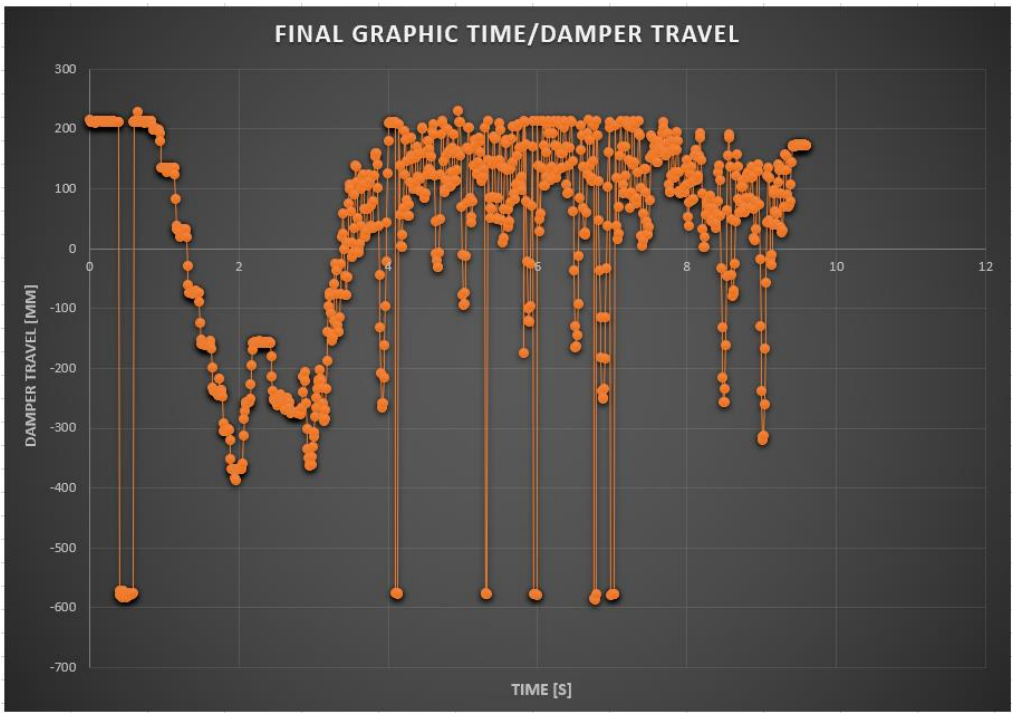


Figura 50 - Gráfico de dispersão por pontos do comportamento do curso do amortecedor num determinado espaço de tempo.

4.4 Análise dos Resultados dos Dados

Partindo destes valores e de todas as variáveis dispostas, é possível calcular toda uma range de valores distintos, como velocidades, acelerações pontuais, entre outros. É denotável que, em certos instantes do teste, alguns dos valores obtidos deram negativo e não permitiram obter um gráfico final conciso e de fácil análise.

Crê-se que esses valores resultaram de uma má leitura dos sensores nos testes, o que se terá de analisar em trabalhos futuros. Sabe-se através da curva Curso-Diferença de Potencial ($curso = f(v)$), que os valores mínimos atingidos eram de 3,12V para 30 mm de curso, além de que os valores máximos atingidos foram 4,82V para 400 mm de curso. Aos valores de curso de amortecedor correspondem números que não se encontram dentro do intervalo de diferença de potencial, pelo que ou existiu uma falha pontual na leitura dos sensores, ou nos momentos onde existiram movimentos extremos e mais imediatos, estes atingiram de tal forma os limitadores do suporte dos sensores que fizeram registar movimentos ainda mais extremos do que os previamente calculados. Isto significa que estes dados negativos não são relevantes nem apresentam valores válidos, pois numa situação real os amortecedores não se desviam da curva Curso-Diferença de Potencial previamente calculada.

Também existe outro pormenor que terá de ser melhorado numa fase posterior deste trabalho, que é o tempo da recolha dos dados. No teste realizado ao sistema, o tempo em média decorrido para retirar estes dados foi perto de 30-40 segundos. No entanto, como se pode observar na Figura *49*, esta indica que todo o teste teve uma duração de 9,6 segundos. Isto deve-se aos tempos para executar cada tarefa do Arduino. Admite-se que, idealmente e se o código cumprisse com os 10 ms de *download* total dado a dado (com as tarefas incluídas), o intervalo de tempo seria maior e a quantidade de dados também.

Cumpre-se assim o objetivo final deste trabalho, obtendo no *Front-end* um gráfico que, ao longo de um certo espaço de tempo, efetua uma leitura das diferentes alturas de curso do amortecedor.

4.5 Observação da Robustez do Sistema

Este sistema, ao longo do desenvolvimento deste projeto, foi sofrendo alterações de forma a aproximar-se cada vez mais de um sistema robusto e funcional, adaptável a mais do que um modelo de carro de Rali. Numa fase inicial de estudo, é unicamente utilizada a placa Arduino diretamente com todas as ligações, pois opta-se por uma abordagem mais rápida e menos robusta, que permita alterações numa fase pouco avançada do projeto. À medida que o sistema ia ganhando forma, é decidido desenvolver uma PCB personalizada, o que aumentou tanto a eficácia das ligações como a simplificação da montagem. Os sensores, os suportes e os braços utilizados no sistema, apesar de resistentes, foram desenvolvidos pela *Renault* para estarem protegidos dentro da carroçaria do carro, o que para esta aplicação não acontece pois, mesmo procurando-se uma localização mais defensível possível, não deixarão de estar sujeitos às condições climáticas e de desgaste. Com o tempo, a humidade pode vir a danificar os componentes eletrónicos, e detritos como pedras na estrada também podem danificar os componentes mecânicos. Além de tudo isto, deve-se ter em conta também o facto de não ser permitido ter este sistema montado no carro em dias de prova, só em testes. Estar sempre a retirar e a colocar de novo causará, por consequência de uso, desgaste.

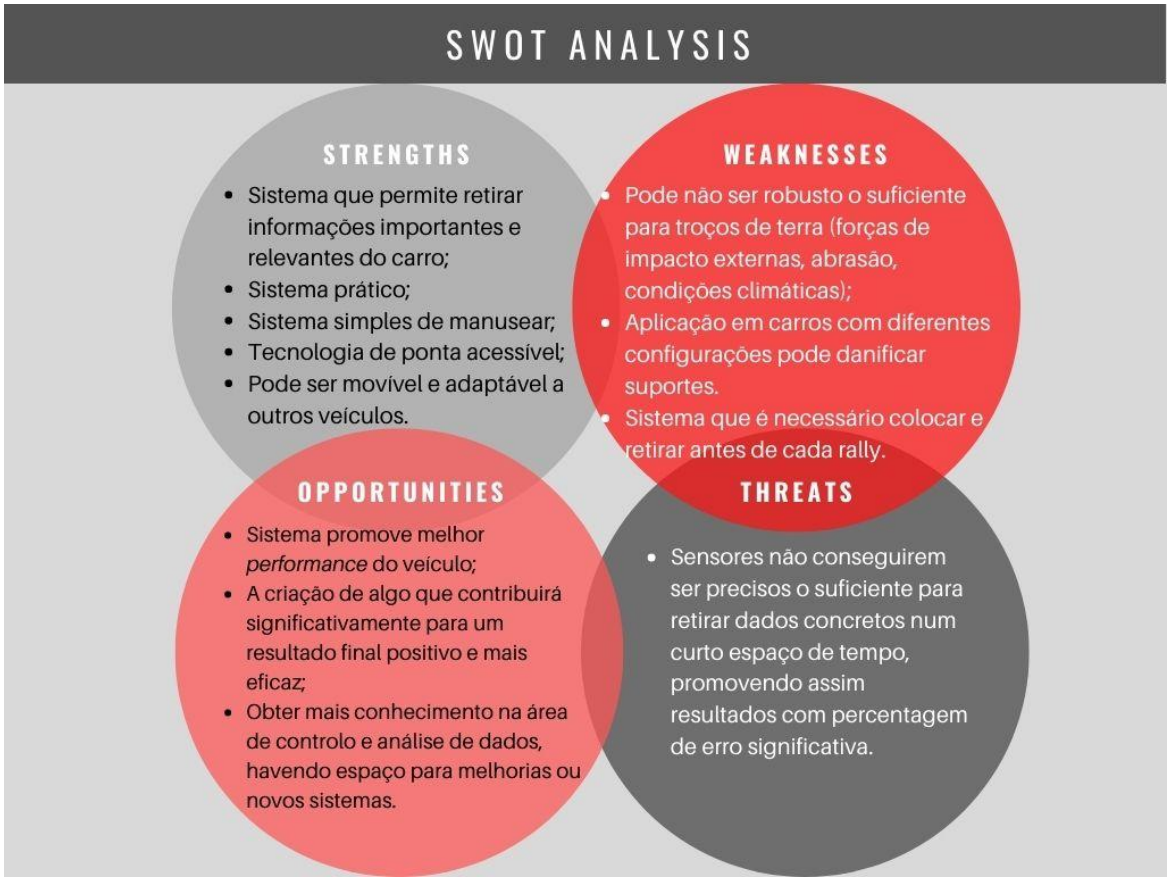
Em suma, avaliando os limitados recursos de *software* e equipamento, foi obtido o sistema final pretendido, resistente, funcional e com boa durabilidade.

4.6 Análise SWOT

Todo e qualquer projeto/produto está sujeito a influências, quer internas quer externas, que condicionam a sua utilidade e desenvolvimento. Para poder avaliar estas influências recorre-se à ferramenta de análise SWOT. O objetivo principal de uma análise SWOT é ajudar as organizações a desenvolver uma consciência completa de todos os fatores envolvidos na tomada de uma decisão de negócios. Identificam-se as forças (*Strengths*) e as fraquezas (*Weaknesses*) do projeto, sendo usualmente controláveis e, relativamente ao ambiente externo, apresentam-se as oportunidades (*Opportunities*) e as ameaças (*Threats*) ao produto, sendo estas, normalmente, difíceis de controlar[40].

Desta forma, é importante a adaptação da análise SWOT ao sistema que esta dissertação se baseou, representada na Tabela 5.

Tabela 5 - Análise SWOT do projeto.



Esta análise permite a toda e qualquer equipa/empresa/negócio estabelecer as suas prioridades para atuar sobre o mercado existente, determinar os seus riscos, assim como as vantagens e oportunidades a analisar, obtendo assim uma análise mais aprofundada do meio envolvente, havendo assim espaço para melhorias e aumento de eficiência do produto/negócio em questão.

De acordo com a Tabela 5 e tecendo conclusões de acordo com os pontos apresentados, é notável que, apesar de carecer em precisão e resistência a longo prazo, necessitando de melhorias no futuro, este é um projeto que auxilia as equipas de engenharia e fornece-lhes ferramentas para que a performance das suas equipas seja cada vez melhor. Esta é uma variável impactante na prestação dos veículos em Rally e a existência de outros métodos/sensores/*softwares* DAS não é novidade. No entanto, para tentar combater os preços exorbitantes que advém com a compra desses pacotes extra, esta solução satisfaz o objetivo pretendido inicialmente.

A RETER DESTE CAPÍTULO

Por fim, este é o capítulo onde se consolida todos os conhecimentos adquiridos e a informação recolhida ao longo destes meses de trabalho. Desde o estudo das ligações elétricas do sistema e dos locais onde se poderiam colocar os sensores as placas, à montagem final, tudo culmina nesta fase final, onde é permitido retirar informações dos amortecedores e monitorizá-los. Desenvolveu-se, assim, uma interface em Excel onde se encontram presentes todas as informações recolhidas e trabalhadas dos sensores de posição angular instalados. Para terminar, é adicionada uma análise SWOT, rematando assim o fim de um projeto e a sua posterior análise.

5. CONSIDERAÇÕES FINAIS

Neste capítulo final são tecidas algumas conclusões relativamente ao projeto desenvolvido, bem como algumas sugestões para trabalhos futuros.

5.1 Conclusões

Em *Motorsport*, uma das ferramentas mais importantes para as equipas é *data* pois, citando *Jörge Segers* "(…)Uma das armas mais importantes para uma equipa de carros de corrida é a informação. Quanto mais informação for possível recolher (e processar), melhor será o julgamento ao tomar decisões-chave. A aquisição de dados fornece aos engenheiros as informações que estes e a equipa exigem para avaliar o desempenho do veículo(…)". Além de melhorar consideravelmente a prestação das equipas em Rali, este sistema DAS foi desenvolvido para facilitar a compreensão da variável em estudo, a variação do curso do amortecedor ao longo de um determinado percurso.

Numa primeira fase, mais propriamente na introdução e na revisão bibliográfica, foi permitido apresentar o projeto, os seus objetivos, as finalidades para o qual estaria a ser desenvolvido e obter uma base factual que facilitasse a sua compreensão. Desde os Sistemas DAS existentes no mercado até à própria análise do sistema STS do carro em estudo, foram recolhidas informações que sustentassem o sistema que se pretendia desenvolver. Também foi efetuado o desenvolvimento de um estudo prévio de como seria o novo sistema STS, desenhando em simulação 3D uma modelação do mesmo.

De seguida, olhando para o mecanismo em si, para o possível trajeto dos dados e estudando como se procederia à recolha e análise dos mesmos, foram selecionados os componentes e os *softwares* que viriam a compor o sistema. Para tal, foram recolhidas informações relativas a todas as peças e programas, foram realizados testes prévios em ferramentas de simulação *online* de forma a verificar se o sistema estaria a ser bem projetado eletronicamente. A partir desses testes-base adquiriram-se possíveis versões de código, que mais tarde foram adaptadas para o sistema final com todas as peças envolventes.

Após tudo isto, procedeu-se ao estudo dos possíveis locais de montagem do sistema e respetiva ligação de todos os componentes, onde foi permitido efetuar testes ao STS sem ser necessário o carro se deslocar, observando o seu curso e comportamento através de Excel.

Tendo todo o sistema sido perfeitamente montado no carro e ter sido possível obter valores exatos das alturas de curso do amortecedor num determinado espaço de tempo, a análise deste projeto e da sua robustez correspondeu ao esperado.

Existiram efetivamente desafios desde o início do trabalho, começando logo pela escassez de informação em tudo o que envolve *Motorsport*, seja análise de dados, seja informações dos equipamentos que constituem o *Škoda* em estudo, necessitando do apoio tanto da equipa da *The Racing Factory* ao longo dos meses de estágio para adquirir o conhecimento necessário para realizar o tema, como também na busca incessante e exaustiva de informações bibliográficas adquiridas no estrangeiro. O facto de entender tão vagamente no início como funcionavam alguns dos *softwares* utilizados, o Arduino IDE e as placas (de Arduino e PCB) também foi algo que consumiu tempo e recursos, no entanto permitiram obter os resultados pretendidos. O último desafio sentido foi no código em Arduino, que se verificou possíveis melhorias de forma a aproveitar este sistema no seu máximo potencial.

Em suma, a aquisição de dados sempre fará parte de tudo o que envolva os desportos motorizados pois, com o avanço das tecnologias, existirão sempre *softwares* e sistemas mais evoluídos, numa área que ainda tem bastante por explorar.

5.2 Trabalhos Futuros

Após realizado este projeto de dissertação, foi pensado adicionar mais alguns componentes e código extra ao Arduino, de forma a que o sistema pudesse ser usado com telemetria. Como este sistema DAS terá utilidade única em testes, estas melhorias não afetariam de forma alguma regulamentos, facilitando o trabalho aos engenheiros. Adicionar-se-ia também uma *dashboard* que, em tempo real, fornecesse informações do comportamento dos amortecedores e do STS.

No que toca ao código executado, várias melhorias poderão ser implementadas. No que diz respeito ao sensor de infravermelhos, este criou muito "ruído" nos resultados. Acredita-se que é possível que tenha sido algo relacionado com as bibliotecas *IRemote* para esse fim. Seria importante executar uma análise mais aprofundada e verificar onde é que se poderia trabalhar o código e eliminar o problema. Também em relação à programação, o código foi comandado para abrir o ficheiro *.txt* sempre que queria escrever. Essa ação normalmente demora cerca de 15 ms, o que se verificou ser um problema a resolver, visto que no gráfico final de tempo-curso do amortecedor observou-se um intervalo de tempo muito menor que o esperado, devendo-se a esses *delays* na execução de determinadas tarefas em Arduino. Para tal prevê-se que comandando o Arduino para abrir o ficheiro só no início e o manter aberto sempre até quando não se pretendesse escrever mais resolveria a situação.

Por fim, na fase do tratamento e análise final dos dados em Excel, foram obtidos valores de altura negativos. Aos valores de curso de amortecedor corresponderam valores que não se encontravam dentro do intervalo de diferença de potencial aceitável. O objetivo numa fase posterior passaria por analisar qual foi a razão e obter um gráfico final sem valores *default*.

REFERÊNCIAS

VI : [1] SANTOS, Rodrigo, "*DATA ACQUISITION: 6 REASONS TO USE AND THE FUNDAMENTALS YOU MUST KNOW"*, Available: https://racingcardynamics.com/data-acquisition-fundamentals/ (accessed Dec. 03, 2020).

VI : [2] SMITH, Josh, *FUNDAMENTALS OF MOTORSPORT ENGINEERING*, Oxford University Press, 2013.

VI : [3] SEGERS, Jörge, *ANALYSIS TECHNIQUES FOR RACECAR DATA ACQUISITION*, SAE International, 2014.

VI : [4] FIA, FPAK, *ANEXO "J" AO CÓDIGO DESPORTIVO INTERNACIONAL*: *Art. 255 A — REGULAMENTAÇÃO ESPECÍFICA PARA AS VIATURAS SUPER 2000 (RALIS) / WRC,* Fevereiro de 2016, Portugal.

VI : [5] *"POSITION SENSORS UK.",* Available: https://www.position-sensors.co.uk (accessed Dec. 07, 2020).

6 : [6] THE RACING FACTORY, "*The Racing Factory Official Website*", Available: https://www.theracingfactory.pt/ (accessed Feb. 16, 2021).

13 : [7] D. Bastow, G. Howard, J. P. Whitehead, "*Car Suspension and Handling, 4th Edition*", 2004.

16 : [8] J. Zuijdijk, *Vehicle Dynamics and Damping,* 2013.

16, 20 : [9] Arduino Portugal, "*O que é Arduino*?", Available: https://www.arduinoportugal.pt/o-que-e-arduino/ (accessed Mar. 08, 2021).

19 : [10] Magneti Marelli, "*Catalogue Racing Ahea", 2016.*

21 : [11] IT Connect, "What is Excel", Available: https://itconnect.uw.edu/learn/workshops/online-tutorials/microsoft-office-365/microsoft-excel-2010/, University Of Washington (accessed Mar. 08, 2021).

21 : [12] Škoda Motorsport, "ŠKODA FABIA Rally2: The Queen of the Rally", Available: https://www.skoda-motorsport.com/en/skoda-fabia-r5-the-queen-of-the-rally/ (accessed Jun. 12, 2021).

24 : [13] FLORES, P., & PIMENTA CLARO, J. C. "*Cinemática de Mecanismos: Vols. 2. Análise Descritiva de Mecanismos*", Universidade do Minho, Guimarães, 2007.

26 : [14] BICIMAX, “O QUE É EXACTAMENTE O REBOUND?”, Available: https://www.bicimax.pt/servicio/tablas-de-regulacion-de-rebote (accessed Jun. 12, 2021).

26 : [15] BIKERUMOR!, “Suspension Tech: What’s the difference between High & Low speed rebound?” Available: https://bikerumor.com/suspension-tech-whats-the-difference-between-high-low-speed-rebound/ (accessed Jun. 12, 2021).

33 : [16] Worcester Polytechnic Institute (WPI), “Potentiometers to measure joint angle or linear motion”, Available: https://wpilib.screenstepslive.com/s/3120/m/7912/l/85771-potentiometers-to-measure-joint-angle-or-linear-motion (accessed Jun. 12, 2021).

37 : [17] HELLA, “*BRIEF INFORMATION Angular Position Sensors Single and double sensors”*, 2013.

37 : [18] Arduino Store, “Buy Arduino Uno Rev 3”, Available: https://store.arduino.cc/arduino-uno-rev3 (accessed Jun. 13, 2021).

39 : [19] EuroCircuits, “What is a PCB or Printed Circuit Board?”, Available: https://www.eurocircuits.com/pcb-printed-circuit-board/ (accessed DEC. 20, 2021).

40 : [20] Autodesk, “What is EAGLE?”, Available: https://www.autodesk.com/products/eagle/overview?term=1-YEAR&tab=subscription (accessed Jan. 11, 2022).

48 : [21] ALL3DP, C. Favela, “Tinkercad & Arduino: How to Design and Simulate Circuits”, Available: https://all3dp.com/2/tinkercad-arduino-how-to-design-simulate-circuits/ May 5, 2020 (accessed Nov. 20, 2021).

50 : [22] SparkFun, MIKEGRUSIN, “Serial Peripheral Interface (SPI)”, Available: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all (accessed Apr. 10, 2022).

50 : [23] Arduino, “SPI”, Available: https://www.arduino.cc/reference/en/language/functions/communication/spi/ (accessed Apr. 10, 2022).

50 : [24] Arduino, “SD”, Available: https://www.arduino.cc/reference/en/libraries/sd/ (accessed Apr. 10, 2022).

51 : [25] Arduino, “IRremote”, Available: https://www.arduino.cc/reference/en/libraries/irremote/ (accessed Apr. 10, 2022).

51 : [26] Arduino, “#define”, Available: https://www.arduino.cc/reference/en/language/structure/further-syntax/define/ (accessed Apr. 10, 2022).

51 : [27] PJRC, Eletronic Projects Components Components Available Worldwide, “Receiving”, Available: https://www.pjrc.com/teensy/td_libs_IRremote.html (accessed Apr. 29, 2022).
[28] Arduino, “unsigned long”, Available: https://www.arduino.cc/reference/en/language/variables/data-types/unsignedlong/ (accessed Apr. 29, 2022).

52 : [29] ALEPH, “What is void setup and void loop?”, Available: https://aleph.org.mx/what-is-void-setup-and-void-loop (accessed Apr. 30, 2022).

52 : [30] TutorialsPoint, “Arduino - I/O Functions”, Available: https://www.tutorialspoint.com/arduino/arduino_io_functions.htm# (accessed Apr. 30, 2022).

52 : [31] Arduino, “Serial.begin()”, Available: https://www.arduino.cc/reference/en/language/functions/communication/serial/begin/ (accessed May 04, 2022).

53 : [32] Arduino, “Serial.print()”, Available: https://www.arduino.cc/reference/pt/language/functions/communication/serial/print/ (accessed May 04, 2022).

53 : [33] PJRC, Eletronic Projects Components Components Available Worldwide, “Receiving”, Available: https://www.pjrc.com/teensy/td_libs_IRremote.html (accessed May 04, 2022).

54 : [34] Arduino, “millis()”, Available: https://www.arduino.cc/reference/en/language/functions/time/millis/ (accessed May 06, 2022).

54 : [35] FluxodeInformação, “Para que serve o Void loop no Arduino?”, Available: https://fluxodeinformacao.com/biblioteca/artigo/read/3876-para-que-serve-o-void-loop-no-arduino, Feb.10, 2022 (accessed May 06, 2022).

55 : [36] Arduino, “analogRead()”, Available: https://www.arduino.cc/reference/pt/language/functions/analog-io/analogread/ (accessed May 07, 2022).

55 : [37] Arduino Docs, “Guide to Arduino & Secure Digital (SD) Storage.”, Available: https://docs.arduino.cc/learn/programming/sd-guide (accessed May 07, 2022).

57 : [38] Arduino, “digitalWrite()”, Available: https://www.arduino.cc/reference/pt/language/functions/digital-io/digitalwrite/ (accessed May 07, 2022).

58 : [39] Arduino, “int”, Available: https://www.arduino.cc/reference/en/language/variables/data-types/int/ (accessed May 07, 2022).

72 : [40] Business News Daily, S. Schooley, “SWOT Analysis: What It Is and When to Use It” https://www.businessnewsdaily.com/4245-swot-analysis.html Apr 14, 2022 (accessed Oct. 09, 2021).

APÊNDICE 1 – CÓDIGO PRINCIPAL ARDUINO

stunning_uusam1 | Arduino 1.8.15

Ficheiro Editar Rascunho Ferramentas Ajuda

stunning_uusam1 §

```
/*
  SD card read/write

  This example shows how to read and write data to and from an SD card file
  The circuit:
   SD card attached to SPI bus as follows:
 ** MOSI - pin 11
 ** MISO - pin 12
 ** CLK - pin 13
 ** CS - pin 4 (for MKRZero SD: SDCARD_SS_PIN)

  created   Nov 2010
  by David A. Mellis
  modified 9 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

*/

#include <SPI.h>
#include <SD.h>
#include <IRremote.h>

#define RECV_PIN 7
#define LED_ON 8

IRrecv irrecv(RECV_PIN);
decode_results results;
File myFile;

// previous time for the tasks depending upon time.
unsigned long prevTime_T1 = millis();

// time intervals for the tasks
long interval_T1 = 10; // read every 10 miliseconds
bool running = false; //flag to start reading values

int sensor0Value;// array with values of the sensor

int sensor1Value;// array with values of the sensor

int sensor2Value;// array with values of the sensor

int sensor3Value;// array with values of the sensor


void setup()
{
  pinMode(LED_ON, OUTPUT);
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }


  Serial.print("Initializing SD card...");

  if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    while (1);
  }
  Serial.println("initialization done.");

  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE);
```

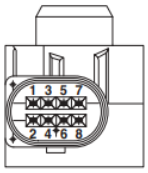
```
  // if the file opened okay, write to it:
  if (myFile) {
    Serial.print("Successufuly open.");

  } else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }
  irrecv.enableIRIn();
}

void loop()
{
 unsigned long currentTime = millis();


  if (running){
    if (currentTime - prevTime_T1 > interval_T1) {

      sensor0Value = analogRead(A0);
      Serial.print(sensor0Value);
      Serial.print("//**//");
      myFile = SD.open("test.txt", FILE_WRITE);
      // if the file opened okay, write to it:
      if (myFile) {
          myFile.print(sensor0Value);
          myFile.print(";");
          // close the file:
          myFile.close();
          }

      sensor1Value = analogRead(A1);
      Serial.print(sensor1Value);
      Serial.print("//**//");
      myFile = SD.open("test.txt", FILE_WRITE);
      // if the file opened okay, write to it:
      if (myFile) {
          myFile.print(sensor1Value);
          myFile.print(";");
          // close the file:
          myFile.close();
          }

      sensor2Value = analogRead(A2);
      Serial.print(sensor2Value);
      Serial.print("//**//");
       myFile = SD.open("test.txt", FILE_WRITE);
      // if the file opened okay, write to it:
      if (myFile) {
          myFile.print(sensor2Value);
          myFile.print(";");
          // close the file:
          myFile.close();
          }

      sensor3Value = analogRead(A3);
      Serial.print(sensor3Value);
      Serial.println("//**//");
       myFile = SD.open("test.txt", FILE_WRITE);
      // if the file opened okay, write to it:
      if (myFile) {
          myFile.print(sensor3Value);
          myFile.println(";");
          // close the file:
          myFile.close();
          }

      digitalWrite(LED_ON, HIGH);
      prevTime_T1 = currentTime;
```

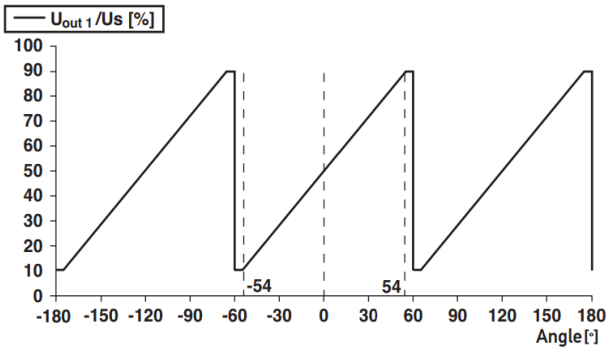
```
  }
  else
   digitalWrite(LED_ON, LOW);


  if(irrecv.decode(&results)){
    do_response(results.value);
    Serial.print(results.value);
    irrecv.resume();
  }
}
/*
  respond to specific remote-control keys with different behaviors
*/

void do_response(int key)
{

  switch (key)
  {
    case 25245:  // FUNC/STOP turns off/on the reading of the analog ports //16 736 925
      running = !running;
    // in order to check if is reading
      if(running)
        digitalWrite(LED_ON, HIGH);
      else
        digitalWrite(LED_ON, LOW);

      Serial.println("FUNC/STOP");
      break;


    default:
      {
        Serial.print("Key ");
        Serial.print(key);
        Serial.println(" not programmed");
      }
    break;
  }
}
```

Housing type B

Output signal U_{out1}
with 5 V supply voltage

HOUSING TYPE B – DOUBLE SENSORS





Output signal $U_{out 2}$ = 100 % – $U_{out 1}$ /Us [%] (opposite curve)

Technical specifications

Mechanical angle range	unlimited (full 360° circle)
Supply voltage	5 V ± 10 % or 9 V to 32 V
"Crossed Scale" output signal	
Supply voltage U_s = 5 V	Output U_{out1} 0.5 V to 4.5 V ratiometric U_{out2} 4.5 V to 0.5 V ratiometric
Supply voltage U_s = 9 V to 32 V	Output U_{out1} 0.5 V to 4.5 V U_{out2} 4.5 V to 0.5 V
Linearity error including temperature drift	± 0.3°
Housing type	B
Protection class	IP 6K9K
Operating temperature	-40 °C to +85 °C
Mating connector[1)]	AMP 1394416-1

Pin assignment

Voltage supply at 5 V DC*	
Pin 1	5 V DC sensor 2
Pin 2	Output U_{out1} 0.5 V to 4.5 V ratiometric
Pin 3	Not assigned
Pin 4	5 V DC sensor 1
Pin 5	Output U_{out2} 4.5 V to 0.5 V ratiometric
Pin 6	Not assigned
Pin 7	Ground sensor 2
Pin 8	Ground sensor 1

* The power supply lines (pin 1 and pin 4) and the ground supply lines (pin 7 and pin 8) can be bridged externally (e.g. in the mating connector) to reduce the number of wires.

Voltage supply at 9 V to 32 V DC**	
Pin 1	Bridge to pin 4 (external)
Pin 2	Output U... 0.5 V to 4.5 V

ANEXO 2 – CÓDIGO *READWRITESDCARD*


```
/*
  SD card read/write

  This example shows how to read and write data to and from an SD card file
  The circuit:
   SD card attached to SPI bus as follows:
 ** MOSI - pin 11
 ** MISO - pin 12
 ** CLK - pin 13
 ** CS - pin 4 (for MKRZero SD: SDCARD_SS_PIN)

  created   Nov 2010
  by David A. Mellis
  modified 9 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

*/

#include <SPI.h>
#include <SD.h>

File myFile;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }


  Serial.print("Initializing SD card...");

  if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    while (1);
  }
  Serial.println("initialization done.");

  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  myFile = SD.open("test.txt", FILE_WRITE);

  // if the file opened okay, write to it:
  if (myFile) {
    Serial.print("Writing to test.txt...");
    myFile.println("testing 1, 2, 3.");
    // close the file:
    myFile.close();
    Serial.println("done.");
  } else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }

  // re-open the file for reading:
  myFile = SD.open("test.txt");
  if (myFile) {
    Serial.println("test.txt:");

    // read from the file until there's nothing else in it:
    while (myFile.available()) {
      Serial.write(myFile.read());

    }
    // close the file:
    myFile.close();
  } else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }
}

void loop() {
  // nothing happens after setup
}
```

ANEXO 3 – CÓDIGO BASE EFETUADO EM SIMULAÇÃO *ONLINE TINKERCAD*

```
#include <IRremote.h>
#define RECV_PIN 7
#define LED_BUILTIN 13
#define SIZE_ARRAY 32

IRrecv irrecv(RECV_PIN);
decode_results results;

// previous time for the tasks depending upon time.
unsigned long prevTime_T1 = millis();
//unsigned long prevTime_T4 = millis();

// time intervals for the tasks
long interval_T1 = 10; // read every 10 miliseconds
//long interval_T4 = 5000; // print brightness of LED3 every 5 seconds
bool running = false; //flag to start reading values
bool send = false; //flag to start sending values

int sensor0Value;// array with values of the sensor
//byte index0 = 0;// index of array0

int sensor1Value;// array with values of the sensor
//byte index1 = 0;// index of array1

int sensor2Value;// array with values of the sensor
//byte index2 = 0;// index of array2

int sensor3Value;// array with values of the sensor
//byte index3 = 0;// index of array3


void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  irrecv.enableIRIn();
}

void loop()
{
 unsigned long currentTime = millis();

  if (running){
    if (currentTime - prevTime_T1 > interval_T1) {
      sensor0Value = analogRead(A0);
      //index0++;
      sensor1Value = analogRead(A1);
      //index1++;
      sensor2Value = analogRead(A2);
      //index2++;
      sensor3Value = analogRead(A3);
      //index3++;
      digitalWrite(LED_BUILTIN, HIGH);
      prevTime_T1 = currentTime;
    }
  }
  else
    digitalWrite(LED_BUILTIN, LOW);
  if(send){
      Serial.print(sensor0Value);
      Serial.print(";");
      Serial.print(sensor1Value);
      Serial.print(";");
      Serial.print(sensor2Value);
      Serial.print(";");
```

```
      Serial.println(sensor3Value);
    send=false;
  }
  if(irrecv.decode(&results)){
    do_response(results.value);
    Serial.print(results.value);
    irrecv.resume();
  }
}
/*
  respond to specific remote-control keys with different behaviors
*/

void do_response(int key)
{
 /*
  if (output_key)
   {
      Serial.print("Key ");
      Serial.println(key);
   }
  */
  Serial.print(key);

  switch (key)
  {
    case 16575:  // FUNC/STOP turns off UUT power
      running = !running;
    // in order to check if is reading
      if(running)
        digitalWrite(LED_BUILTIN, HIGH);
      else
        digitalWrite(LED_BUILTIN, LOW);

      Serial.println("FUNC/STOP");
      break;

    case 28815:  // ST/REPT Positive tests Select Test and Repeat Test
      send = !send;
      Serial.println("ST/REPT");
      break;

    case 32640:  // turns on UUT power
      Serial.println("POWER");
      break;

    default:
      {
        Serial.print("Key ");
        Serial.print(key);
        Serial.println(" not programmed");
      }
    break;
  }
}
```