Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

João António Silva Figueiredo

Pedro Miguel Abreu Amaral

Sistema de registo e monitorização de pesagens num processo genérico

Relatório de Projeto

Licenciatura em Engenharia Eletrotécnica

Orientador:

Professor Doutor Joaquim Delgado

Junho de 2023

RESUMO

O presente relatório descreve um sistema de pesagem suportado por 4 células de carga e um módulo HX711 que transmite os dados para o microcontrolador Arduíno MEGA.

O Arduíno MEGA, por sua vez, exibe as pesagens num LCD e envia-as para uma base de dados através do módulo *WiFi* ESP8266-01.

A base de dados foi criada em *PHP MyAdmin* e os dados podem ser também consultados num *website*.

O objetivo deste sistema é fornecer uma solução para a pesagem de objetos genéricos de forma automatizada e com registo numa base de dados *online* para facilitar o acesso e a análise das pesagens recolhidas.

O sistema pode ser utilizado em áreas como a indústria ou a agricultura, como por exemplo para monitorizar a pesagem de caixas de fruta fornecidas por uma cooperativa frutícola.

AGRADECIMENTOS

No presente relatório expõe-se a conclusão de uma etapa muito significativa para nós. Foi um percurso marcado por intenso trabalho e dedicação, mas não teríamos conseguido chegar aqui sem o apoio de muitas pessoas, às quais queremos agradecer.

Ao Professor Doutor Joaquim Delgado pela sua orientação, disponibilidade, saber que transmitiu, pelas opiniões e críticas, pela colaboração no solucionar de dúvidas que foram surgindo e por todas as palavras de incentivo.

A todos os professores e colegas do Departamento de Engenharia Eletrotécnica da ESTGV, pela sua colaboração para que este projeto fosse alcançado com sucesso.

Por fim às nossas famílias e amigos pelo incentivo e por serem modelos de coragem. Também ainda pela ajuda na superação de obstáculos que foram surgindo ao longo da realização deste projeto e pelos incentivos, sem os quais teria sido mais difícil alcançar os objetivos delineados.

ÍNDICE GERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

SIGLAS E ABREVIATURAS

ACID	*Atomicity, Consistency, Isolation and Durability*
ADC	*Analogue-to-digital converter*
ESTGV	Escola Superior de Tecnologia e Gestão de Viseu
FTP	*File Transfer Protocol*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
IDE	*Integrated Development Environment*
IP	*Internet Protocol*
IPv4	*Internet Protocol version 4*
LCD	*Liquid Crystal Display*
LED	*Light-emitting diode*
PC	*Personal computer*
PHP	*Hypertext Preprocessor*
PWM	*Pulse Width Modulation*
SD	*Secure Digital*
SQL	*Structured Query Language*
TCP	*Transmission Control Protocol*
TFT	*Thin-film transistor*
UDP	*User Datagram Protocol*
USB	*Universal serial bus*
UTFT	*Universal Thin-film transistor*
WEP	*Wired Equivalent Privacy*
WiFi	*Wireless Fidelity*
WPA	*Wi-Fi Protected Access*
WPA2	*Wi-Fi Protected Access 2*

1. Introdução

No âmbito da gestão da produção e do controlo da qualidade, a pesagem de produtos constitui uma operação necessária em inúmeros processos.

Com este projeto pretendemos desenvolver um sistema automático de aquisição e tratamento das pesagens dos produtos à saída de um setor (ou subsetor) de uma unidade industrial, como por exemplo uma cooperativa agrícola fornecedora de fruta que opera em 3 turnos.

Hipoteticamente esta fornece a fruta aos seus clientes em embalagens/caixas *standard* retornáveis com uma tara de 5 kg e com uma quantidade cujo peso depende do tipo de fruta que contém. Para a maçã, por exemplo, o peso oscila em torno dos 30 kg, sendo 5 kg para a caixa mais cerca de 25 kg para a fruta.

O sistema de pesagem será constituído por uma balança eletrónica com base no uso de quatro transdutores de força integrados nos pés de um sistema automático de elevação da carga localizado à saída de um transportador rolante que capta a grandeza peso ($P = m \cdot g$) para cada caixa de fruta a expedir.

Além disso com a pesagem deve dar-se indicação de que cada caixa não deverá possuir peso superior ou inferior a 0,5 kg do valor esperado (25 kg). Caso tal ocorra deverá ser ativado um alarme (luminoso e sonoro) (*HIGH*) ou (*LOW*) e corrigida a quantidade de fruta com a remoção ou adição de fruta, antes de a caixa seguir.

1.1 Objetivos

Este projeto situa-se no domínio das células de carga direcionadas para pesagem de produtos genéricos, como foi referido acima a pesagem de caixas de fruta. O sistema tem uma capacidade máxima de pesagem de 140 kg.

Com o objetivo de aumentar a fiabilidade da pesagem, serão captadas várias medidas de cada caixa de fruta e calculada a sua média.

O resultado (médio) será exibido e registado na memória local do sistema de registo (Arduíno).

Além da captura e exibição das medidas no ecrã local, estas devem ser enviadas para um PC via *wireless* onde corre um programa que fará a leitura, registo e processamento em tempo real e a pedido do utilizador, das medidas captadas.

Por cada turno de 8 horas o *software* deve permitir:

1. Exibir o valor de todas as medidas sequencialmente.

2. Calcular em cada instante o somatório das pesagens já efetuadas.

3. Calcular e exibir o valor máximo e mínimo das pesagens até ao instante em cada turno.

4. Fazer a análise de tendências: comparar cinco medidas sucessivas de pesagem e antever a tendência de exceder o limite superior ou inferior, exibindo um alarme visual.

5. Dispor de capacidade para iniciar o registo num novo ficheiro com nome específico para documentar as pesagens efetuadas num turno, como exemplo T1_2023-04-18 para o turno 1 do dia 18 de Abril de 2023.

6. A comunicação da balança com o PC deve ser estabelecida por ligação *wireless*.

Premissas para implementação

Para implementar os objetivos acima expostos o sistema vai ser constituído por duas unidades principais: a de captura do mesurando e a de tratamento de dados (ver Figura 1).

A unidade de captura do mesurando é constituída por: módulo de balança e células de carga/transdutores, sistema de condicionamento do sinal dos transdutores, sistema de leitura, pré-tratamento, apresentação de dados, alarmes luminosos/sonoro e módulo de comunicação *wireless*.

A unidade de tratamento de dados/informação é suportada pelo *hardware* e *software* num PC tradicional ou outro dispositivo com capacidade de comunicação via *wireless*, que pode estar muito afastado da unidade de captura do mesurando.

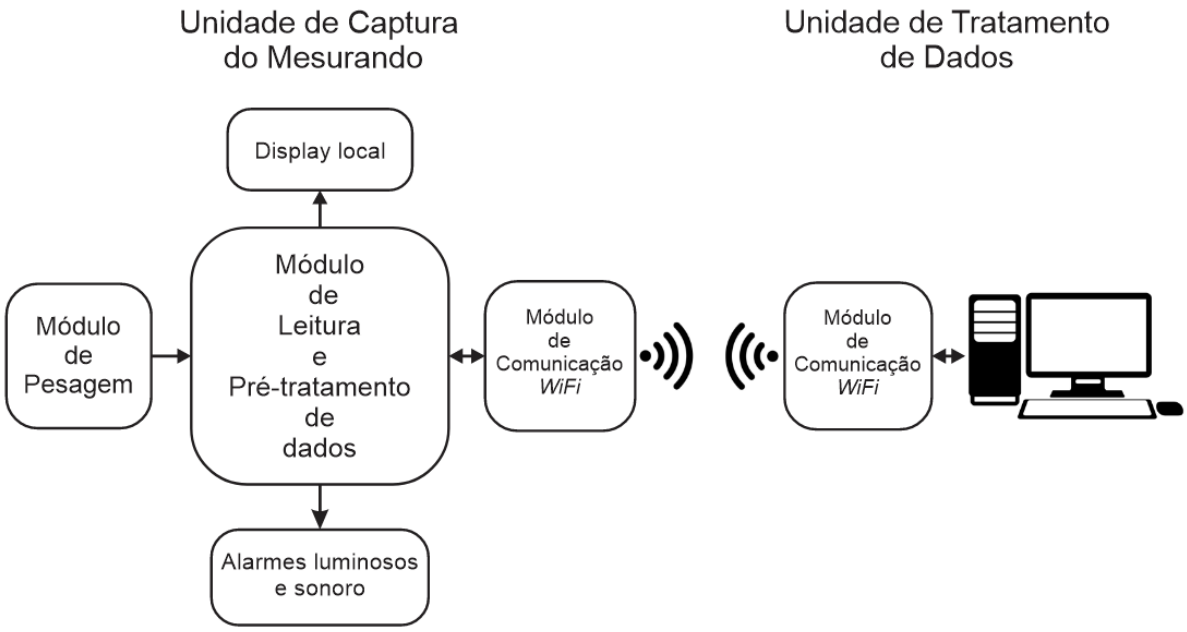


Figura 1 – Diagrama geral do sistema de pesagem.

1.2 Estrutura do relatório

Este relatório está estruturado em 5 capítulos e 11 anexos. Neste primeiro é efetuada uma breve descrição do projeto e dos seus objetivos.

No **capítulo 2** são descritos com detalhe os módulos da unidade de captura do mesurando.

No **capítulo 3** são expostas as funcionalidades suportadas pela unidade de tratamento de dados.

No **capítulo 4** é descrita a seleção e montagem dos componentes e dos processos necessários para a realização do sistema de aquisição e tratamento de dados. Sendo também abordados os procedimentos de teste efetuados para a implementação do sistema, nomeadamente a calibragem da balança através da realização de múltiplas medições com pesos padrão de valor conhecido.

No **capítulo 5** são expostas as conclusões e feitas algumas considerações sobre áreas críticas a serem alvo de maior atenção em possíveis trabalhos futuros.

2. Unidade de captura do mesurando

A unidade de captura do mesurando integra os componentes necessários para a sua captura (valor das pesagens), pré-processamento, exibição no ecrã local, alarmes luminosos/sonoro e o envio dos dados via *wireless* para a unidade de tratamento de dados/informação.

2.1 Constituição da unidade de captura do mesurando

A unidade de captura do mesurando assenta no uso de quatro transdutores de força constituídos por extensómetros, para reagir ao mesurando peso (P = m . g). Este tipo de transdutor é designado genericamente por célula de carga ou em inglês *load cell* e estão integrados em cada um dos pés da balança.

2.1.1 Funcionamento da célula de carga

O princípio de funcionamento da célula de carga é baseado na extensometria.

Extensometria

A extensometria permite medir microdeformações de um suporte quando este é submetido a forças (produto da massa pela gravidade) que lhe são aplicadas.

Dependendo da resistência mecânica do suporte, a sua sensibilidade pode detetar deformações provocadas por forças desde mN (mili *Newton*) até MN (mega *Newton*).

O elemento passivo da extensometria é o extensómetro que consiste num fio condutor longo e muito fino, disposto numa configuração como a exposta na Figura 2.

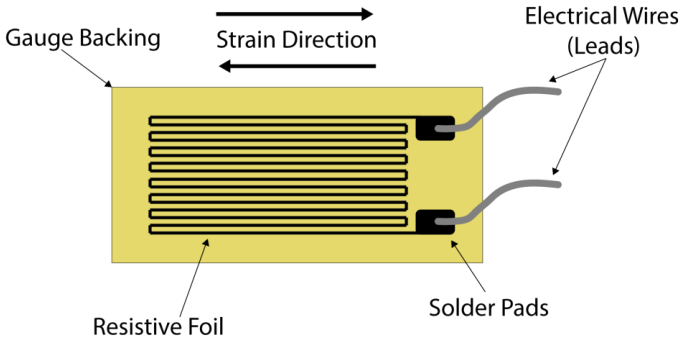


Figura 2 – Extensómetro [1].

Como podemos observar na Figura 2 o extensómetro é concretizado por um material condutor disposto sobre uma superfície que vai ser colada a um suporte metálico submetido à variação do mesurando (peso). Este suporte varia de extensómetro para extensómetro dependendo do material que é feito e da sua aplicação, por exemplo para variações do mesurando na ordem das gramas é utilizado um material mais flexível, enquanto que para medições na ordem das toneladas se utiliza um material mais robusto. Normalmente existem extensómetros com resistências de 350 Ω, 500 Ω e 1000 Ω em repouso.

Quando o extensómetro é submetido a esforços de tração ou de compressão há uma microdeformação das suas dimensões.

Se for tracionado tal provoca um alongamento longitudinal e uma diminuição de secção.

A ilustração desta situação pode ser vista na Figura 3.

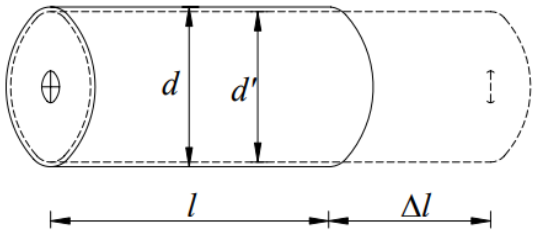


Figura 3 - Deformações num fio sob tração [2].

Como:

$$R = \rho \times \frac{l}{S} \qquad [1]$$

Onde:

R – Valor da resistência (em Ohm);

ρ – Valor da resistividade do cobre (em Ω/m);

l – Comprimento (em metros);

S – Área de secção transversal (em metros quadrados).

Na Figura 3 a linha contínua ilustra parte de um fio metálico idêntico aos utilizados nos extensómetros, onde *l* é o comprimento antes da deformação e que apresenta uma resistência *R.* A linha a traço interrompido ilustra as dimensões do fio quando submetido ao esforço e o seu comprimento é agora de *l+Δl* e a resistência *R+ΔR*.

O extensómetro também apresenta uma sensibilidade *S* que é dada pela seguinte equação:

$$S = \frac{variação\ relativa\ da\ resistência}{variação\ relativa\ do\ comprimento} = \frac{\frac{\Delta R}{R}}{\frac{\Delta L}{L}} \quad [2]$$

Para converter a variação do valor da resistência em sinal elétrico de tensão, o circuito mais utilizado é a ponte de *Wheatstone* que analisaremos na secção 2.1.2.

Relação entre a tensão e a deformação

A relação entre a tensão aplicada num dado material e a sua deformação é dada pela *Lei de Hooke*.

Quando um dado material é tracionado, a força *F* aplicada no material é proporcional à deformação *d* causada na região elástica, mantendo uma relação constante entre a magnitude da força externa e a quantidade de deformação.

$$F = k \times d \quad [3]$$

Onde:

F – Força aplicada (em Newton);

k – Constante elástica do material (em N/m);

d – Deformação (em metros).

Supondo que um dado material com comprimento *l* é tracionado e o alongamento é dado por *Δl*, a relação entre o alongamento e o comprimento original é designada de deformação *ε*, não tendo unidades de medida.

$$\varepsilon = \frac{\Delta l}{l} \quad [4]$$

Conceito de região elástica

A deformação (ε) é proporcional à força aplicada desde o repouso (ausência de esforço) até ao ponto *a*, sendo esta região designada por elástica, onde é válida a *Lei de Hooke*, como se pode observar na Figura 4.

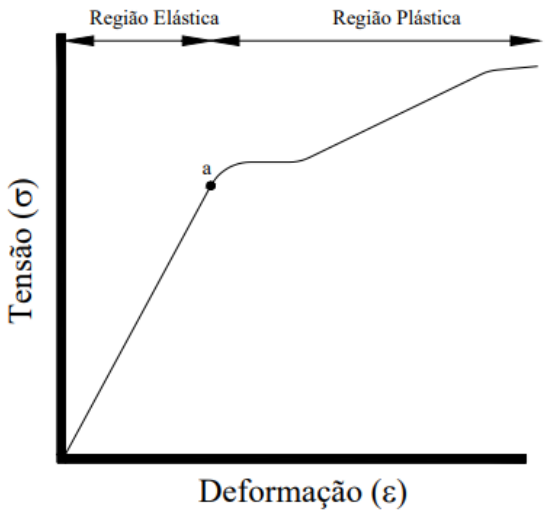


Figura 4 – Curva tensão-deformação [2].

A relação tensão-deformação na região elástica é dada pela equação (5), onde *E* é a constante de proporcionalidade, também designada por módulo de elasticidade longitudinal ou *Módulo de Young* do material em estudo.

$$E = \frac{\sigma}{\varepsilon} \qquad [5]$$

Deteção da variação da resistência

Como vemos na expressão (1), sempre que o extensómetro é tracionado provoca o aumento do seu comprimento, a diminuição da sua secção e um aumento do valor da resistência.

Se o extensómetro for comprimido provoca uma diminuição do seu comprimento, aumento da sua secção e a resistência diminui.

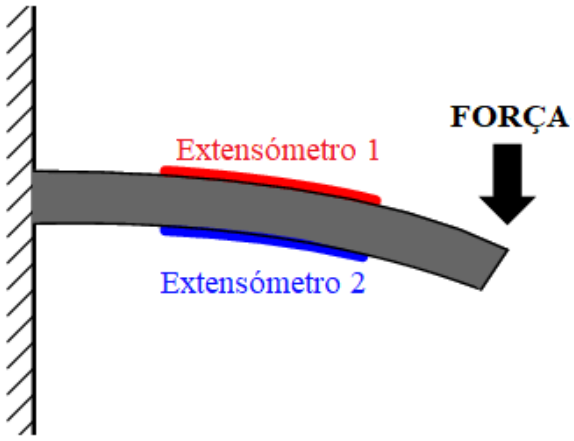


Figura 5 – Localização dos extensómetros aplicados num material sujeito a uma força [3].

Se dois extensómetros semelhantes forem colocados nas faces superior e inferior de um elemento metálico sujeito a esforço, como ilustrado na Figura 5, constata-se que o extensómetro da face superior fica submetido ao esforço de tração e a sua resistência aumenta e o colocado na face inferior fica submetido à compressão e a sua resistência diminui.

2.1.2 Funcionamento da ponte de *Wheatstone*

A ponte de *Wheatstone*, constituída por quatro braços resistivos como ilustrado na Figura 6, constitui um dos circuitos mais usado para converter de forma precisa a variação da resistência em tensão.

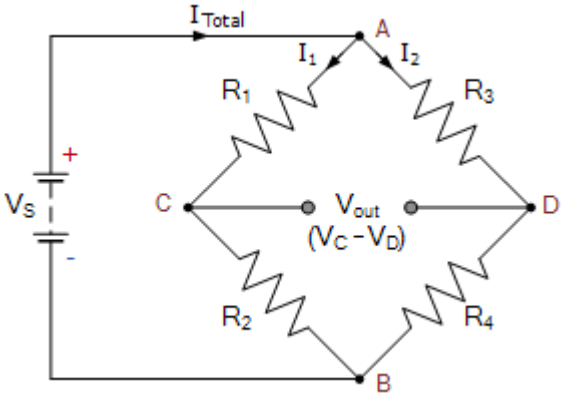


Figura 6 – Ponte de *Wheatstone* [4].

A partir da Figura 6, a tensão de saída da ponte de *Wheatstone* é dada pela expressão (6).

$$V_{CD} = V_C - V_D = (V_{AB} \times \frac{R_2}{R_1 + R_2}) - (V_{AB} \times \frac{R_4}{R_3 + R_4})$$

$$V_{CD} = V_{AB} \times \left(\frac{R_2}{R_1 + R_2} - \frac{R_4}{R_3 + R_4} \right) \quad [6]$$

Para complementarmos a explicação sobre o funcionamento da ponte de *Wheatstone* expomos abaixo um exemplo.

Se entre os pontos A e B for aplicada a tensão de alimentação fornecida por uma fonte de elevada precisão (10 V por exemplo) e os braços resistivos podem integrar resistências fixas ou variáveis (extensómetros) que vão ser submetidas aos esforços mecânicos.

Para a situação hipotética em que temos quatro resistências de valor igual em repouso (R_1= R_2= R_3= R_4=1000 Ω) e com a tensão de alimentação V_{AB} de 10 V, constata-se que:

$$V_{CD} = 10 \times \left(\frac{1000}{1000 + 1000} - \frac{1000}{1000 + 1000} \right) = 0\ V$$

Diz-se que a ponte de *Wheatstone* está em equilíbrio.

Se um dos braços da ponte, R_4 por exemplo, for constituído por um extensómetro que vai aumentar para 1100 Ω em função da força aplicada constata-se que:

$$V_{CD} = 10 \times \left(\frac{1000}{1000 + 1000} - \frac{1100}{1000 + 1100} \right) = -0.24\ V$$

Verificamos que há uma conversão da variação de R_4 em tensão V_{CD} ou um transdutor da aplicação de força que provoca variação da resistência do extensómetro e por sua vez a variação da tensão na ponte de *Wheatstone*, concluindo assim que a tensão é proporcional à força aplicada.

Tipos de ponte de *Wheatstone*

Existem três tipos de configuração para a ponte de *Wheatstone*, que são um quarto de ponte, meia ponte e ponte completa.

Tabela 1 – Número de extensómetros por tipo de ponte.

Tipo de ponte	Número de extensómetros
Um quarto de ponte	1
Meia ponte	2
Ponte completa	4

Um quarto de ponte de *Wheatstone*

Quando varia apenas um braço da ponte estamos perante uma ponte de *Wheatstone* de um quarto, como podemos observar na Figura 7.

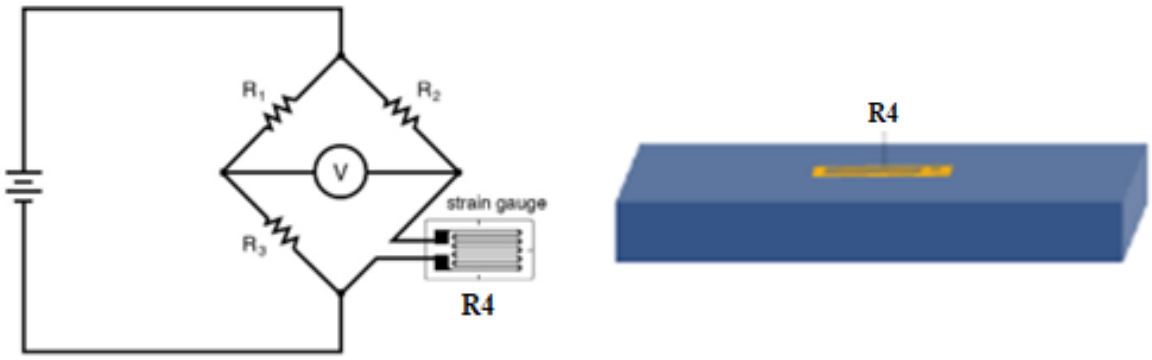


Figura 7 – Um quatro de ponte de *Wheatstone* [3][5].

Meia ponte de *Wheatstone*

Se o braço R_4 variar (aumentando o seu valor), o braço adjacente R_2 também varia (diminuindo o seu valor) estamos perante meia ponte de *Wheatstone*, como se ilustra na Figura 8. A variação simultânea do braço superior e inferior provoca a variação da tensão com maior sensibilidade.

Figura 8 – Meia ponte de *Wheatstone* [3][5].

Ponte de *Wheatstone* completa

Quando os quatro braços da ponte variarem, se R_1 e R_4 aumentarem e os braços adjacentes R_2 e R_3 diminuírem estamos perante uma ponte completa, como se pode visualizar na Figura 9.

Este tipo de ponte é o mais utilizado devido a exibir uma maior sensibilidade ou variação da tensão de saída em função da variação das resistências nos quatro braços.

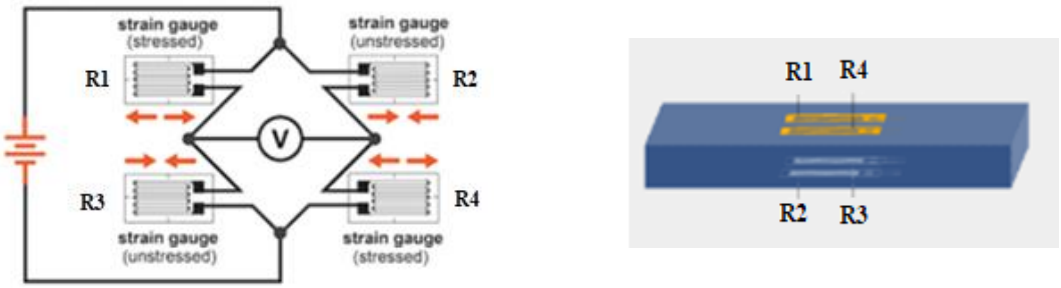


Figura 9 – Ponte de *Wheatstone* completa [3][5].

2.1.3 Células de carga

A célula de carga é um transdutor usado para converter a variação do mesurando força num sinal elétrico. Dispõe de um *design* simples e baseado na transferência de uma força aplicada, na deformação de um material de suporte e no fluxo da eletricidade. É um dispositivo muito versátil que apresenta um desempenho preciso e robusto, para uma vasta diversidade de aplicações.

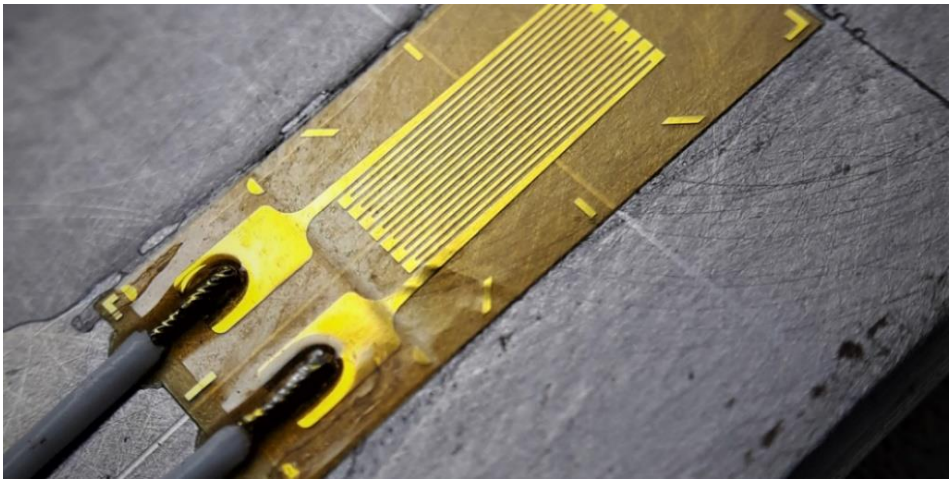


Figura 10 – Célula de carga aplicada num suporte [1].

As células de carga estão disponíveis com diferentes configurações e padrões, dependendo da aplicação e ambiente. A Figura 10 ilustra um extensómetro tradicional aplicado no suporte de uma célula de carga.

Diferentes tipos de células de carga

Existem vários tipos de células de carga, classificados de acordo com a gama de forças a medir e a forma como detetam a força aplicada, tais como flexão, deformação ou compressão.

Como neste projeto utilizamos células de carga com base em extensómetros, vamos abranger mais esse exemplo.

Ponto único

A célula de carga de ponto único (exposta na Figura 11) é a que é utilizada numa maior variedade de aplicações, como balanças pequenas e médias. Integra geralmente quatro extensómetros e pode ser muito precisa. Normalmente é posicionada sob uma plataforma que é submetida ao peso exercido na parte superior.

Figura 11 – Célula de carga de ponto único [6].

Cisalhamento

A célula de carga de cisalhamento (exibida na Figura 12) é usada para medir a durabilidade e os limites de falha de produtos em testes destrutivos, assim como para verificar a medição de força em prensas. Normalmente pode ser colocada entre dois componentes para compressão ou ser usada em tração através de furos com rosca.

Figura 12 – Célula de carga de cisalhamento [7].

Parâmetros críticos no uso de células de carga

Quando uma célula de carga é submetida a esforço ela exibe uma curva de deformação em função da força aplicada, onde se verificam fenómenos como histerése, repetibilidade e não-linearidade como se ilustra na Figura 13.

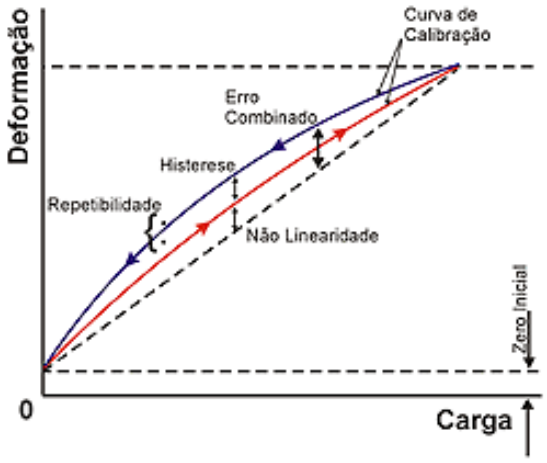


Figura 13 – Histerése, repetibilidade e não-linearidade [8].

A histerése é a diferença máxima entre as leituras de saída efetuadas pelo transdutor para a mesma força aplicada, uma obtida quando aumenta a força a partir do zero e a outra quando diminui a força até zero.

A repetibilidade é a diferença máxima entre a leitura de saída do transdutor para cargas repetidas em condições ambientais e de esforço idênticas.

A não-linearidade é o desvio máximo dos sinais da saída da célula de carga numa linha reta entre a saída que representa o zero e a máxima força aplicada.

2.1.4 Sistema de leitura da ponte de *Wheatstone* (HX711)

Para efetuar a leitura do valor da tensão à saída da ponte de *Wheatstone*, ou detetar a variação da tensão provocada pela variação da resistência nos extensómetros, utilizamos o circuito HX711 que se expõe na Figura 14.

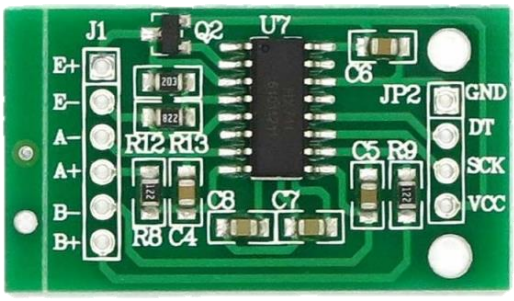


Figura 14 – Circuito HX711 [9].

Este circuito desempenha duas funções, amplificar o sinal de tensão gerado pela ponte de *Wheatstone* e convertê-lo de analógico em digital (ADC) com 24 *bits* de resolução (2^{24} = 16777216 combinações possíveis) para ser lido pelo microcontrolador. Possui uma impedância de entrada muito alta, o que faz com que seja ideal para medir com precisão pequenas alterações na resistência (peso) aplicadas sobre as células de carga.

Na Figura 15 podemos observar o circuito interno do HX711 em que do lado esquerdo estão os pinos de entrada sendo que E+ e E- são a alimentação, 5V e 0V respetivamente, e S+ e S- que estão ligados aos pinos INPA e INNA (entrada positiva e negativa do canal A) são os pinos que transmitem o sinal gerado pela ponte de *Wheatstone*.

Podemos observar na entrada um condensador ligado em paralelo com os dois pinos, para filtrar o ruído (filtro passa-baixo).

Na parte superior podemos observar um regulador de tensão no qual a tensão de entrada é 2.7 a 5.5 V e a base do transístor funciona como saída de controlo do regulador de tensão.

O pino DOUT é uma saída série e o PS_SCK a entrada de relógio série. O *pinout* completo do HX711 é exposto no Anexo 1.

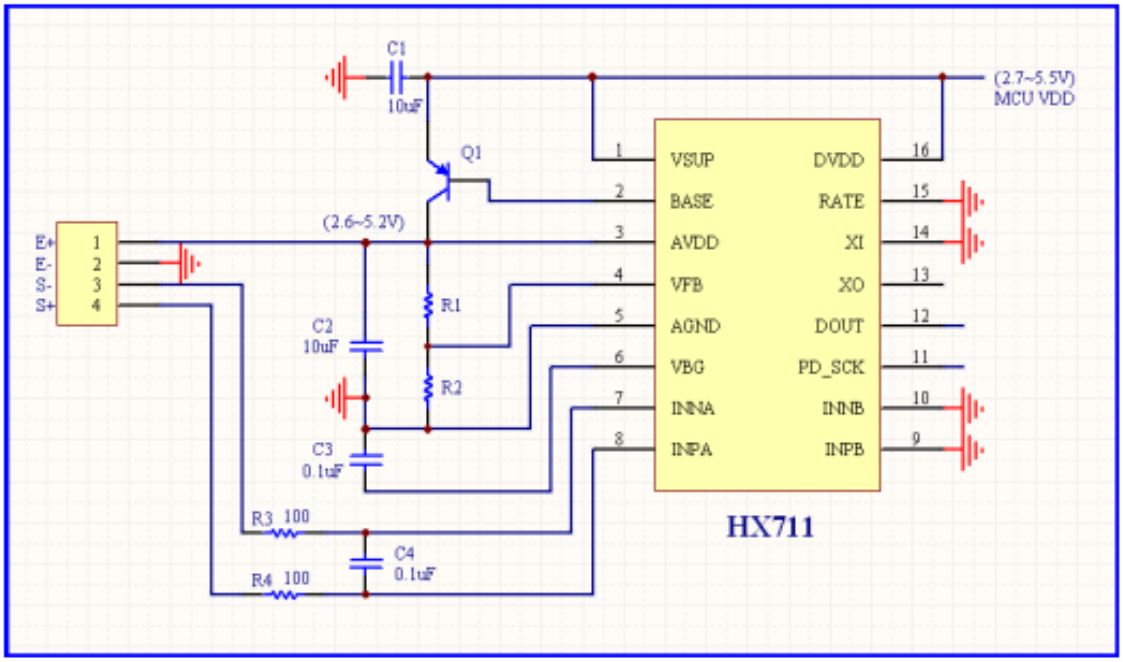


Figura 15 – Esquema interno do circuito HX711 [10].

Na Figura 16 expõe-se o diagrama de blocos do circuito HX711 onde o sinal gerado na ponte de *Wheatstone* é amplificado internamente com um ganho de 64 ou 128 (canal A) dependendo da grandeza do sinal e convertido num sinal digital de 24 *bits* com valor hexadecimal entre 800000 e 7FFFFF, que por sua vez é transmitido, via DOUT, para o microcontrolador (Arduíno MEGA).

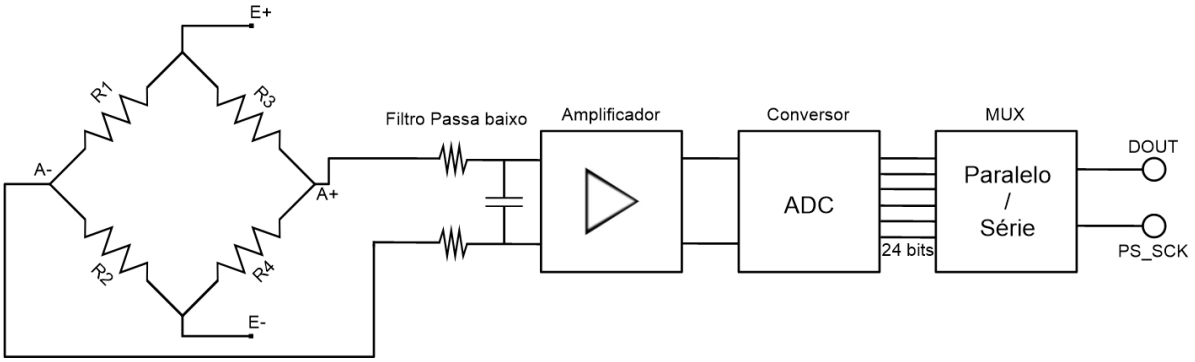


Figura 16 – Diagrama de blocos do circuito HX711.

2.2 Módulo de leitura e pré-tratamento de dados

O módulo de leitura e pré-tratamento de dados permite que os dados originados por sensores e dispositivos de aquisição de dados sejam lidos e preparados para uso em sistemas de análise, controlo e tomada de decisões.

Este é constituído por *hardware* e *software* que funcionam para ler, organizar e transformar dados em informação útil.

2.2.1 Microcontrolador Arduíno MEGA

O microcontrolador Arduíno MEGA é um dos dispositivos mais versáteis da família Arduíno, dispondo de bibliotecas *online* que facilitam a sua programação.

Figura 17 – Microcontrolador Arduíno MEGA [11].

O Arduíno MEGA tem incorporado um microcontrolador ATmega 2560 de 8 *bits* com capacidade para processar um grande volume de dados com frequência de 16 MHz. Possui 54 pinos de entradas e saídas digitais, das quais 15 podem ser usadas como saídas PWM, 16 entradas analógicas, 4 portas série e uma memória *flash* de 256 *KBytes*.

O Arduíno MEGA permite utilizar uma plataforma de desenvolvimento integrado (IDE) com base na linguagem de programação *C++*.

O *pinout* mais detalhado do Arduíno MEGA é exposto no Anexo 2.

2.2.2 Ecrã local 3.2” TFT LCD

O ecrã TFT com 3,2 polegadas na diagonal, que se expõe na Figura 18, possui resolução de 320 por 240 *pixels* e 65 mil cores. Oferece interface fácil de usar, permitindo a exibição de gráficos e textos em tempo real. O controlador deste módulo LCD é o SSD1289 e suporta uma interface de dados de 16 *bits*. Devido a ser um ecrã com interface de entrada resistiva, tem ainda a capacidade de detetar a pressão sobre o ecrã permitindo a implementação de entrada de dados e de interfaces interativas com o utilizador. Possui ainda uma *slot* para inserir um cartão de memória SD.

Figura 18 – Ecrã 3.2” TFT LCD [12].

Para efetuar a ligação do ecrã local ao Arduíno é necessário utilizar uma placa de expansão para o Arduíno (*shield* ITDB02) que permite a utilização do ecrã TFT de 3,2 polegadas.

2.3 Módulo de comunicação *wireless*

O módulo de comunicação *wireless* utilizado foi o ESP8266-01 (ilustrado na Figura 19) da família ESP8266. Este é utilizado para aplicações que requerem ligação *WiFi* e suporta linguagens de programação como *Lua*, *Python* e *C++*.

Pode ser programado através da plataforma do Arduíno e oferece recursos avançados como WEP, WPA e WPA2 para garantir a segurança no envio de dados na rede *WiFi*.

O ESP8266-01 suporta as redes 802.11 b/g/n (o número 802.11 refere-se às redes sem fio e a letra corresponde à frequência e velocidade que opera), podendo trabalhar como ponto de acesso ou como estação e também como ponto de acesso e estação, enviando e recebendo dados.

A comunicação do módulo *WiFi* com o Arduíno MEGA pode ser feita via série utilizando os pinos RX e TX, podendo ser configurada através de comandos AT (conjunto de comandos para configurar e controlar o ESP8266-01, como por exemplo verificar se o módulo está a funcionar corretamente, reiniciar o módulo, verificar a versão do *firmware* instalada, detetar pontos de acesso disponíveis na área e ligar o módulo a uma rede *WiFi*). A tensão de alimentação é 3.3V.

Este módulo comunica num intervalo de frequências de 2.4 a 2.5 GHz e tem um alcance de 90 metros. Em termos de protocolos de rede utiliza IPv4, TCP, UDP, HTTP e FTP.

O *pinout* do módulo ESP8266-01 é exposto no Anexo 3.

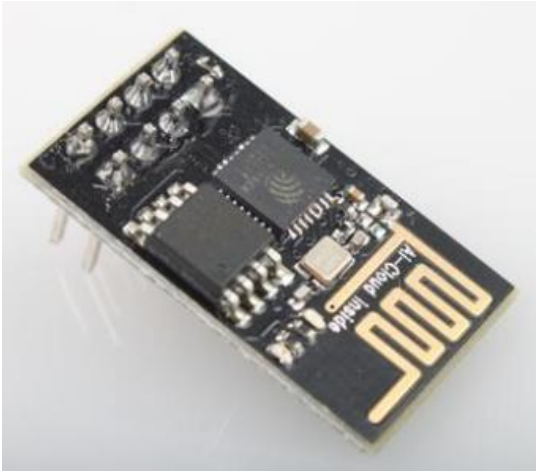


Figura 19 – ESP8266-01 [13].

3. Unidade de tratamento de dados

A unidade de tratamento de dados integra *hardware* e *software* e pressupõe a intervenção de um operador, responsáveis por recolher, armazenar, processar e analisar dados.

Esta é fundamental para realizar pesagens precisas, permitindo a monitorização de cada uma ao longo do tempo e a identificação de eventuais problemas que permitam tomar medidas antecipadamente para melhorar os processos.

3.1 Base de dados

A base de dados é um componente essencial para o sistema de pesagem, pois permite que as informações recolhidas no Arduíno MEGA sejam armazenadas, organizadas e acedidas de forma eficiente, além de conter ferramentas de análise de dados, como tendência e desvio-padrão, entre outros, que facilitam a tomada de decisões para melhorar o desempenho do sistema [14].

Existem duas formas de transferir dados do Arduíno MEGA para a base de dados: – manual e – automática. Em ambas, os dados são captados por dispositivos de medição diferindo na forma de inserir na base da dados. No modo manual são inseridos pelo utilizador, enquanto no modo automático são inseridos automaticamente.

O armazenamento de dados pode ser feito no *hardware* da balança ou à distância, existindo diversos tipos de sistemas de administração de dados disponíveis, como *MySQL*, *PostgreSQL*, *Oracle*, *SQL Server* e *MongoDB*. Para garantir que a capacidade de armazenamento seja suficiente para todos os dados, são utilizadas técnicas como criação de tabelas, compactação de dados e gestão de ficheiros [15].

A exibição dos dados armazenados deve ser eficiente e fácil de aceder, sendo comum a utilização de tabelas, gráficos, relatórios ou painéis. Cada uma dessas formas tem características próprias e a escolha da melhor depende das necessidades e objetivos do sistema de pesagem.

3.1.1 Tipo de sistema de administração de dados

Para armazenar os valores das pesagens efetuadas optamos por usar o sistema de administração de dados *MySQL*.

Este é um sistema de administração de dados disponível em código aberto, amplamente utilizado em aplicações *web* e empresariais devido à sua fiabilidade, escalabilidade e desempenho. É compatível com a maioria dos sistemas operativos, incluído *Linux*, *Windows* e *macOS* e constitui uma escolha frequente para pequenas e médias empresas. Suporta a

linguagem *SQL* padrão, além de extensões e recursos avançados, como transações *ACID*, replicação, *clustering* (agrupamento de exemplos) e separação de tabelas [16].

Na Tabela 2 expõe-se a estrutura genérica de uma base de dados hipotética para o sistema de pesagem de uma cooperativa fornecedora de fruta.

Tabela 2 – Estrutura de uma base de dados de um sistema de pesagem de fruta.

Contagem	**Peso**	**Acumulado**	**Data**
1	30.082	30.082	9/05/23 18:25
2	29.975	60.057	9/05/23 18:26
3	29.956	90.013	9/05/23 18:27
4	30.013	120.026	9/05/23 18:28

4. Implementação do sistema

A implementação do sistema de pesagem engloba a seleção e a montagem dos componentes necessários para a realização do sistema de aquisição e tratamento de dados. Sendo também expostos adiante os procedimentos obrigatórios para a calibragem da balança através da realização de pesagens com pesos padrão de valor conhecido, e ainda a criação da base de dados para armazenar as pesagens efetuadas.

4.1 Balança *Sanitas SBF-70*

A estrutura de suporte e os extensómetros da balança eletrónica *Sanitas SBF-70* foram utilizados para implementar o transdutor de pesagem devido ao módulo eletrónico desta não se encontrar funcional.

Figura 20 – Balança Sanitas SBF-10 [17].

Esta balança possui quatro células de carga, um ecrã, uma placa dedicada que tinha a função de aquisição e tratamento de dados e ainda uma transferência de dados, via *Bluetooth* para um *smartphone*, como ilustrado na Figura 20.

4.1.1 Células de carga da balança

Na Figura 21 podemos ver uma das células de carga removida da balança.

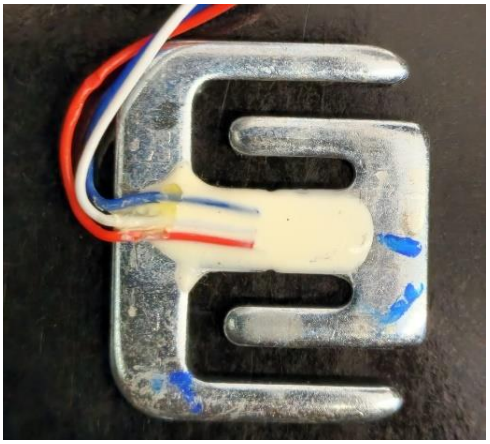


Figura 21 – Célula de carga da balança.

Estas células funcionam com base em meia ponte de *Wheatstone* em que R2 aumenta o valor da resistência e R1 diminui ao ser aplicada uma força (ver Figura 22). Apresenta uma ligação de três fios, um azul, um branco e um vermelho. O fio branco é o que fornece o sinal de variação da resistência em função da força aplicada e os fios azul e vermelho são de alimentação.

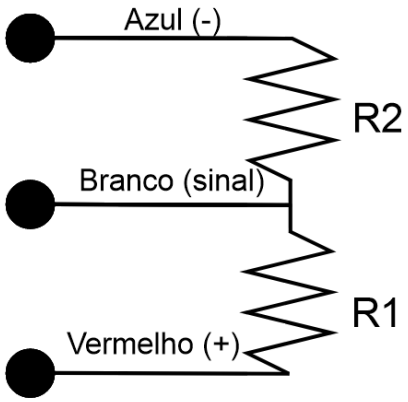


Figura 22 – Esquema de uma célula de carga removida da balança.

Com objetivo de averiguar o estado das células, efetuamos várias medições recorrendo a uma multímetro de elevada precisão.

Primeiro efetuamos medições sem remover as ligações do módulo eletrónico existente na balança e registamos os valores expostos na Tabela 3.

Tabela 3 – Medição da resistência de cada célula mantendo as ligações existentes.

Célula de carga	**Medição entre fio (Ω)**		
	Branco-azul	**Branco-vermelho**	**Azul-vermelho**
1	868	868	1502
2	867	867	1470
3	868	868	1502
4	868	867	1470

Depois removemos cada célula da balança, medimos a resistência e registamos os valores expostos na Tabela 4.

Tabela 4 – Medição da resistência de cada célula removendo as ligações existentes.

Célula de carga	**Medição entre fio (Ω)**		
	Branco-azul	**Branco-vermelho**	**Azul-vermelho**
1	999	999	1995
2	1000	1000	1996
3	1000	1000	1997
4	999	999	1995

Com estas medições concluímos que a resistência das células de carga é de 1000 Ω em repouso.

4.2 Circuito HX711

Na Figura 23 está exposta a ligação das células de carga ao circuito HX711 descrito em 2.1.4.

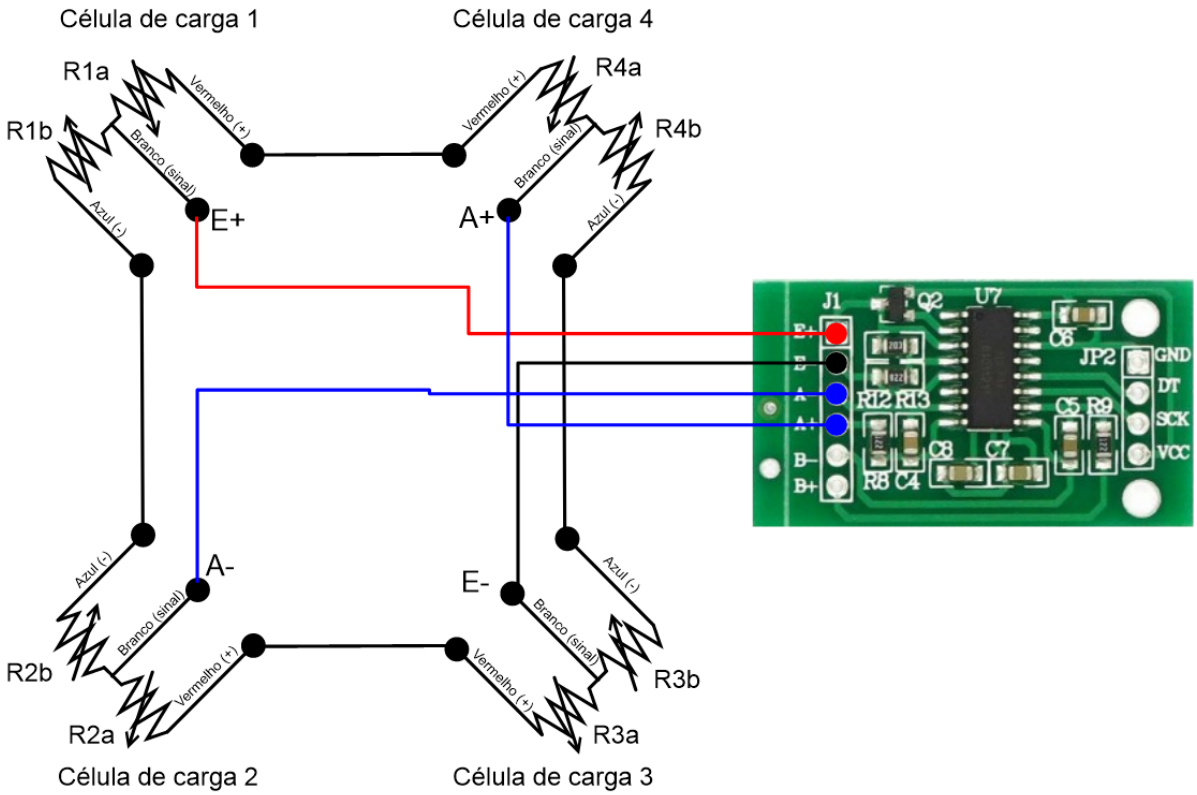


Figura 23 – Esquema de ligação das células ao circuito HX711.

Na Figura 24 expõe-se o esquema simplificado das 4 células de carga, cada uma funcionando como meia ponte de *Wheatstone*. Com o método de ligação representado na Figura 23, estas irão funcionar como uma ponte de *Wheatstone* completa, na qual ao exercer uma força, o valor da resistência R1b + R2b e R3b + R4b aumentam, enquanto R2a + R3a e R1a + R4a diminuem. Como foi referido na secção 2.1.4, a alimentação da ponte é feita através dos pinos E+ e E-, 5V e 0V respetivamente, enquanto os pinos A+ e A- captam a variação da tensão.

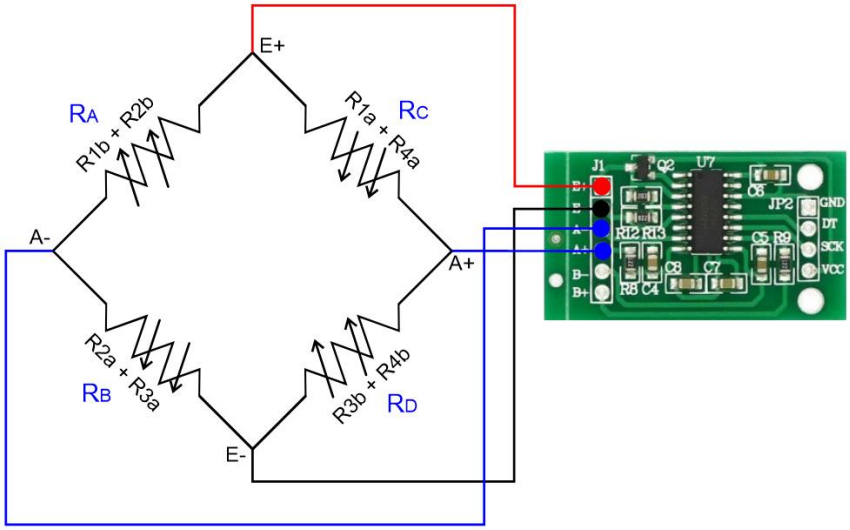


Figura 24 – Variação dos braços resistivos e ligação das células ao circuito HX711.

Com as ligações expostas na Figura 24 e com finalidade de comprovar a expressão (6) efetuamos medições de tensão entre os pontos A+ e A- com um voltímetro e registamos na Tabela 5.

$$V_{A+A-} = V_{E+E-} \times \left(\frac{R_D}{R_C + R_D} - \frac{R_B}{R_A + R_B} \right)$$

Em que R_A corresponde a R1b + R2b, R_B corresponde a R2a + R3a, R_C corresponde a R1a + R4a e R_D corresponde a R3b + R4b, V_{E+E-} é a tensão de alimentação do circuito HX711 e V_{A+A-} é a tensão de saída da ponte de *Wheatstone*.

Tabela 5 – Tensão de saída V_{A+A-} em função do valor do peso.

Peso (kg)	**Tensão de saída (mV)**
0	0.1
5	0.4
10	0.6
15	0.9
20	1.1
84	4.1
89	4.4
94	4.6
99	4.8
104	5.0

4.3 Transmissão e leitura do valor da tensão

Para efetuar a transmissão dos valores captados pelo HX711 ligámos o pino DOUT à entrada digital 8 (condutor azul) e PS_SCK (condutor verde) à entrada digital 9 do Arduíno MEGA, para realizar a sua leitura.

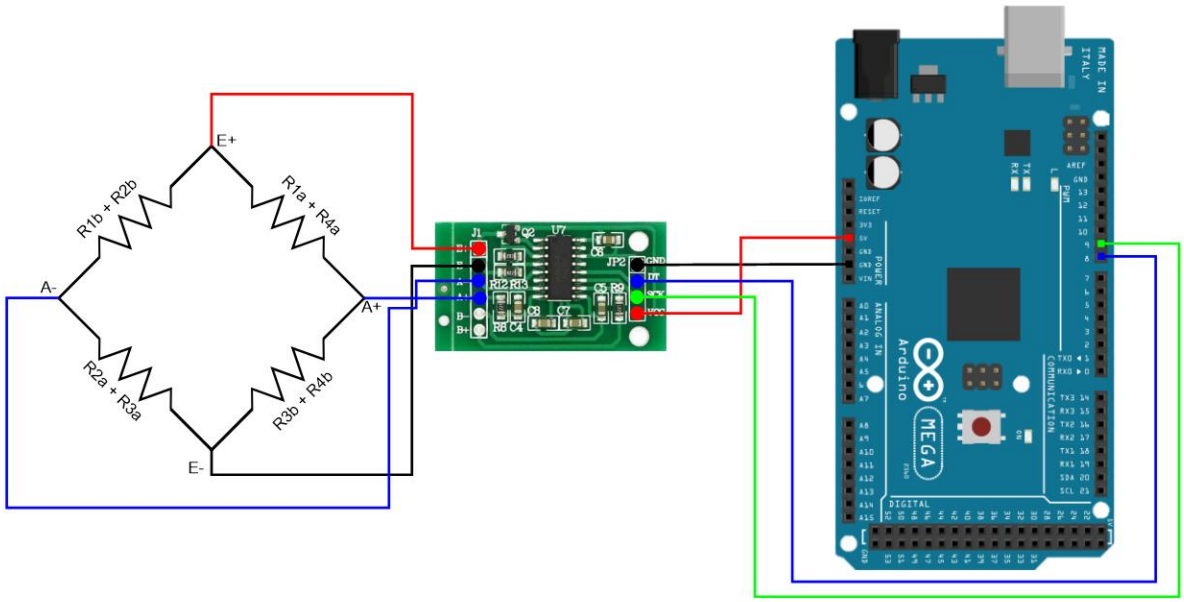


Figura 25 – Ligação do HX711 ao Arduíno MEGA.

Com a implementação das ligações expostas na Figura 25 pudemos realizar todos os testes necessários para calibrar a balança.

4.3.1 Procedimento para leitura das pesagens

Após a concretização da ligação do *hardware* de captura das medidas ao Arduíno, foi necessário desenvolver código para efetuar a leitura do peso sobre a balança, o qual é descrito a seguir:

```
// Biblioteca do circuito HX711.
#include "HX711.h"
// Ligação do circuito HX711 ao microcontrolador Arduíno.
const int LOADCELL_DOUT_PIN = 8;
const int LOADCELL_SCK_PIN = 9;
HX711 scale;
```

```
void setup() {
	// Inicialização do monitor série.
	Serial.begin(115200);

	// Inicialização do circuito HX711.
	scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN)
}
void loop() {
	// Variável “reading” é o valor transmitido pelo circuito HX711.
	float reading = scale.read();
}
```

4.4 Procedimento para calibragem da balança

Para que a balança exiba valores precisos tivemos que realizar ensaios nos quais utilizamos padrões com peso conhecido entre zero e 106 kg, como se pode observar na Tabela 6 e no Anexo 5.

Com base nestes dados procedemos de seguida à identificação de um polinómio que permita aproximar o valor da medida ao peso real em função do código hexadecimal fornecido pelo circuito HX711.

Tabela 6 – Valores disponibilizados pelo circuito HX711.

Valores Conhecidos (kg)	Valor medido pelo HX711
0	-135200
0.145	-139200
0.25	-141500
0.5	-148050
0.75	-154275
1	-160550
1.25	-166880
1.5	-173330
1.75	-179580
2	-185750
2.25	-191900
2.5	-198400
2.75	-204700
3	-210600
3.25	-217000
3.5	-223320
3.75	-229550
4	-236050
4.25	-242250
4.5	-248850
4.75	-255050
5	-261000
5.25	-267170
5.5	-273600
80.35	-2124470
82.1	-2167800
83.55	-2202500
84.55	-2228600
85.55	-2254600
86.55	-2279200
87.55	-2304350
88.64	-2332100
89.64	-2357800
90.64	-2382200
91.64	-2408200
92.64	-2432900
93.6	-2456300
94.6	-2476900
95.6	-2501800
96.6	-2527800
98.43	-2573600
99.43	-2598750
100.43	-2623800
101.43	-2648900
103.26	-2695900
104.26	-2721900
105.26	-2745900
106.26	-2769800

A primeira tentativa para encontrar esse polinómio foi o uso das funções do *Excel*, mas devido ao facto deste não fornecer um polinómio consentâneo e com o rigor exigido, como podemos verificar na Tabela 7, o que iria resultar numa aproximação muito imprecisa, tivemos de utilizar o *MATLAB* para identificar os coeficientes do polinómio que permitissem maior rigor.

Tabela 7 – Coeficientes polinomiais obtidos com o *Excel* e o *MATLAB*.

Coeficiente	***Excel***	***MATLAB***
Terceiro grau	4*10^-20	3.73634*10^-20
Segundo grau	3*10^-13	2.56745*10^-13
Primeiro grau	-4*10^-5	-3.99307*10^-5
Termo independente	-5.4406	-5.44063

A partir desta informação produzimos o código abaixo, que nos permitiu criar uma tabela com os valores das aproximações dos polinómios, como podemos ver na Tabela 8 e no Anexo 5.

```
% Ler o ficheiro Excel usando a função xlsread.
[num,txt,raw] = xlsread('dados.xlsx');
% Extrair os dados das colunas "x" e "y".
x = num(:,1);
y = num(:,2);
% Ajustar um polinómio de grau n aos dados usando a função polyfit.
% Polinómio de primeira ordem (linear).
coeficientes = polyfit(x, y, 1);
fprintf('Linear: \n');
fprintf('Coeficientes do polinómio: %.5e %.5e\n', coeficientes(1), coeficientes(2));
% Polinómio de segunda ordem.
coeficientes = polyfit(x, y, 2);
fprintf('Segunda ordem: \n');
fprintf('Coeficientes do polinómio: %.5e %.5e %.5e\n', coeficientes(1), coeficientes(2),
coeficientes(3));
% Polinómio de terceira ordem.
coeficientes = polyfit(x, y, 3);
fprintf('Terceira ordem: \n');
fprintf('Coeficientes do polinómio: %.5e %.5e %.5e %.5e \n', coeficientes(1), coeficientes(2),
coeficientes(3), coeficientes(4));
% Polinómio de quarta ordem.
coeficientes = polyfit(x, y, 4);
```

```
fprintf('Quarta ordem: \n');

fprintf('Coeficientes do polinómio: %.5e %.5e %.5e %.5e %.5e\n', coeficientes(1), coeficientes(2), coeficientes(3), coeficientes(4), coeficientes(5));

% Polinómio de quinta ordem.

coeficientes = polyfit(x, y, 5);

fprintf('Quinta ordem: \n');

fprintf('Coeficientes do polinómio: %.5e %.5e %.5e %.5e %.5e %.5e \n', coeficientes(1), coeficientes(2), coeficientes(3), coeficientes(4), coeficientes(5), coeficientes(6));

% Polinómio de sexta ordem.

coeficientes = polyfit(x, y, 6);

fprintf('Sexta ordem: \n');

fprintf('Coeficientes do polinómio: %.5e %.5e %.5e %.5e %.5e %.5e %.5e \n', coeficientes(1), coeficientes(2), coeficientes(3), coeficientes(4), coeficientes(5), coeficientes(6), coeficientes(7));
```

Tabela 8 – Resultados do ensaio para identificar o melhor polinómio.

Valores Conhecidos (kg)	Valor medido pelo HX711	Linear	2ª Ordem	3ª Ordem	4ª Ordem	5ª Ordem	6ª Ordem
0	-135200	-0.128	-0.058	-0.037	-0.064	0.000	0.019
0.145	-139200	0.034	0.103	0.123	0.097	0.157	0.175
0.25	-141500	0.127	0.195	0.215	0.190	0.248	0.264
0.5	-148050	0.391	0.458	0.477	0.453	0.506	0.520
0.75	-154275	0.643	0.708	0.726	0.704	0.751	0.763
1	-160550	0.896	0.959	0.977	0.957	0.999	1.009
1.25	-166880	1.151	1.214	1.230	1.211	1.249	1.257
1.5	-173330	1.412	1.472	1.488	1.471	1.503	1.510
1.75	-179580	1.664	1.723	1.738	1.722	1.750	1.755
2	-185750	1.913	1.971	1.985	1.971	1.995	1.998
2.25	-191900	2.161	2.218	2.231	2.218	2.238	2.240
2.5	-198400	2.424	2.479	2.491	2.480	2.496	2.496
2.75	-204700	2.678	2.732	2.744	2.733	2.745	2.745
3	-210600	2.916	2.969	2.980	2.971	2.979	2.978
3.25	-217000	3.175	3.225	3.236	3.228	3.233	3.231
3.5	-223320	3.430	3.479	3.489	3.482	3.484	3.481
3.75	-229550	3.681	3.729	3.739	3.733	3.732	3.728
4	-236050	3.944	3.990	3.999	3.994	3.990	3.985
4.25	-242250	4.194	4.239	4.247	4.243	4.237	4.231
4.5	-248850	4.461	4.504	4.511	4.509	4.500	4.493
4.75	-255050	4.711	4.753	4.760	4.758	4.747	4.740
5	-261000	4.951	4.992	4.998	4.997	4.984	4.976
5.25	-267170	5.200	5.240	5.245	5.246	5.230	5.222
5.5	-273600	5.460	5.498	5.503	5.504	5.486	5.478

A partir dos resultados da Tabela 8, obtivemos a quantificação do erro entre o valor do peso real e o aproximado pelo polinómio, assim como a sua média.

Tabela 9 – Erro e valor médio obtidos em cada polinómio.

Valores Conhecidos (kg)	Valor medido pelo HX711	Diferença de ERRO	Linear	2ª Ordem	3ª Ordem	4ª Ordem	5ª Ordem	6ª Ordem
0	-135200							
0.145	-139200		0.76578	0.292501	0.1545	0.329526	0.083996	0.203744
0.25	-141500		0.492747	0.220435	0.141605	0.24074	0.009128	0.056981
0.5	-148050		0.217524	0.084475	0.046767	0.093005	0.011401	0.03988
0.75	-154275		0.143277	0.056535	0.032464	0.061231	0.001437	0.017691
1	-160550		0.104134	0.040551	0.023293	0.043352	0.001386	0.008858
1.25	-166880		0.078873	0.029187	0.016015	0.030869	0.001194	0.005525
1.5	-173330		0.058802	0.018395	0.007949	0.01934	0.002219	0.006643
1.75	-179580		0.04908	0.01527	0.006749	0.015715	0.000244	0.003122
2	-185750		0.043403	0.014527	0.007442	0.014615	0.002735	0.000954
2.25	-191900		0.039346	0.014302	0.008328	0.014125	0.005331	0.004358
2.5	-198400		0.030449	0.0085	0.003426	0.008107	0.001796	0.00146
2.75	-204700		0.026105	0.006669	0.002319	0.006115	0.001724	0.001859
3	-210600		0.027869	0.010493	0.006729	0.009825	0.006902	0.007376
3.25	-217000		0.02315	0.00755	0.004295	0.006779	0.005133	0.005894
3.5	-223320		0.020028	0.005943	0.003119	0.005093	0.004479	0.005458
3.75	-229550		0.018291	0.005512	0.003056	0.004602	0.004827	0.005969
4	-236050		0.014046	0.002422	0.000293	0.00146	0.002411	0.003685
4.25	-242250		0.01315	0.002528	0.000678	0.001531	0.003053	0.004417
4.5	-248850		0.008766	0.000948	0.002544	0.001978	5.44E-05	0.00149
4.75	-255050		0.008242	0.000679	0.002058	0.001728	0.000695	0.002173
5	-261000		0.00979	0.001569	0.000378	0.00051	0.003239	0.004737
5.25	-267170		0.009498	0.001919	0.000901	0.000853	0.003848	0.005357
5.5	-273600		0.007324	0.000338	0.000521	0.000734	0.002494	0.004003

	Linear	2ª Ordem	3ª Ordem	4ª Ordem	5ª Ordem	6ª Ordem
Media dos ERROS (%)	1,663	0,712	0,462	0,777	0,192	0,357

A partir da identificação da média do erro de cada polinómio, foi possível escolher o que apresenta a melhor aproximação do valor real.

Como podemos ver na Tabela 9, o polinómio de quinta ordem apresenta a melhor aproximação no intervalo de valores de peso conhecidos. No entanto, devido ao limite de peso suportado pela balança ser 140 kg e apenas conseguirmos efetuar medições até 106 kg, os valores acima dependem exclusivamente do polinómio escolhido, como está exposto na Figura 26, retirada do *MATLAB*.

Com a análise da Figura 26 constatou-se que para valores acima de 115 kg, a aproximação do polinómio de quinta ordem deixa de ser viável.

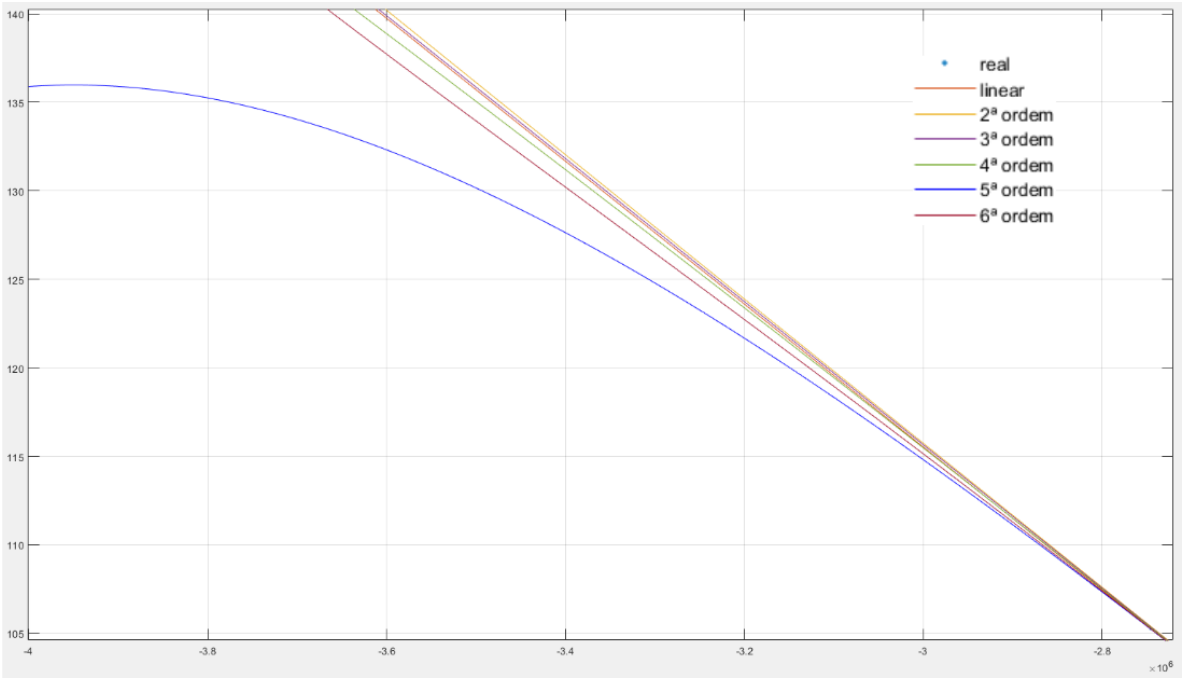


Figura 26 – Gráfico da aproximação entre polinómios.

Para prosseguir com a implementação do sistema de pesagem poderíamos ter escolhido a aproximação do polinómio de sexta ordem devido a apresentar o segundo valor mais baixo da média do erro, mas não optamos por essa solução **por ser um polinómio que exige muito maior processamento ao Arduíno**.

Como as opções anteriores não eram viáveis, optamos pelo polinómio de terceira ordem que exibe o menor valor médio de erro seguinte.

Para a programação da função polinomial escolhida na linguagem de programação do Arduíno, utilizamos o código abaixo, onde a variável "**peso**" devolve o valor da pesagem em kg e a variável "**reading**" é o valor disponibilizado pelo circuito HX711.

```
// Polinómio de 3ª Ordem.
peso = 3.73634e-20*pow(reading,3)+ 2.56745e-13*pow(reading,2) -3.99307e-05*reading -
5.44063;
```

Com a concretização do procedimento da calibragem foi possível concluir que o sistema de pesagem tem uma gama de medição entre zero e 140 kg.

Cada pesagem exibe um erro de ± 0.462%, ou seja, o valor exibido estará sujeito a um erro igual ao valor apresentado pelo polinómio de aproximação.

4.4.1 Testes ao sistema com o polinómio escolhido

Pesagem de 200 gramas

Na medição de 200 gramas, o valor exibido da pesagem irá ser 0.2 kg ± 0.462%, como demonstra na Figura 27.

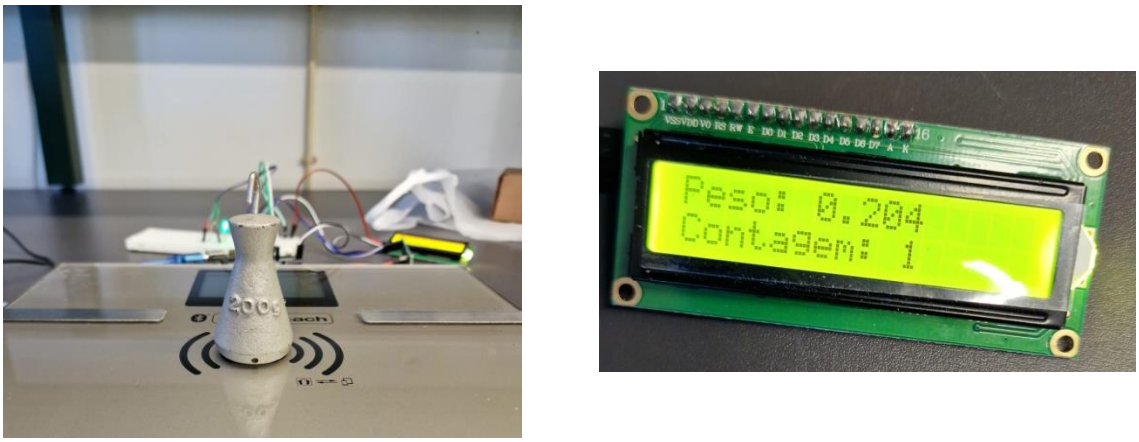


Figura 27 – Pesagem de 200 gramas.

Pesagem de 1 kg

Na medição de 1 kg, o valor exibido da pesagem irá ser 1 kg ± 0.462%, como demonstra na Figura 28.

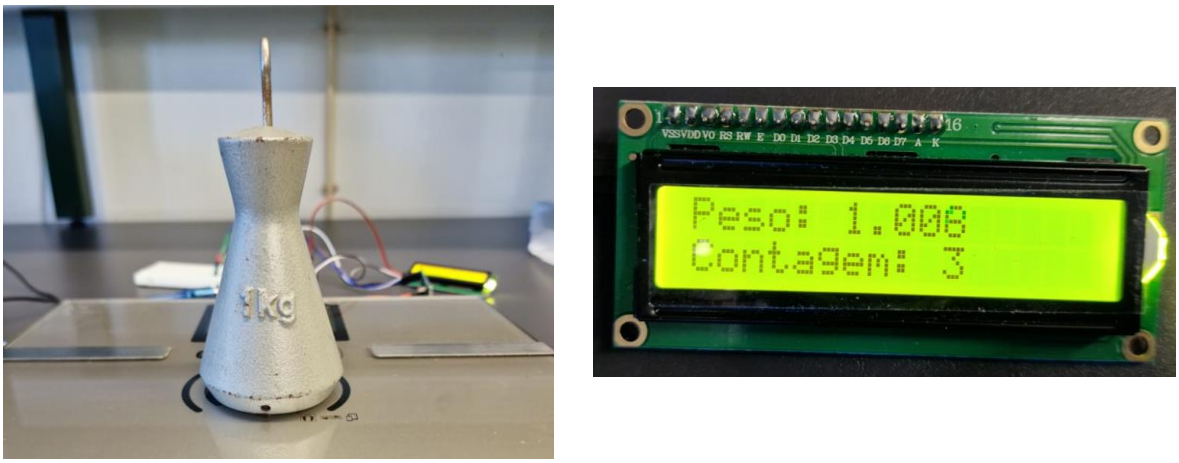


Figura 28 – Pesagem de 1 kg.

Pesagem de 4.7 kg

Na medição de 4.7 kg, o valor exibido da pesagem irá ser 4.7 kg ± 0.462%, como demonstra na Figura 29.

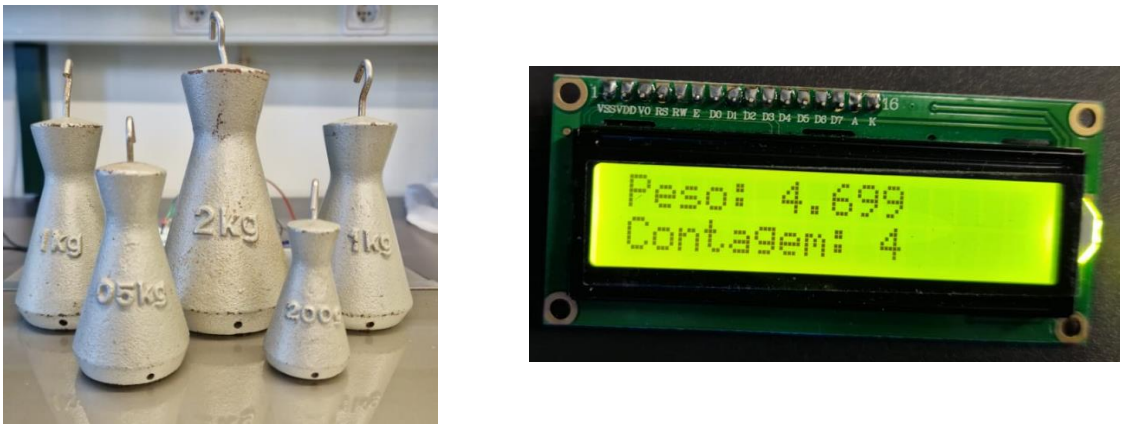


Figura 29 – Pesagem de 4.7 kg.

4.5 Funções suportadas pelo sistema de pesagem

As funções a serem suportadas pelo sistema de pesagem são:

- Cálculo do valor das pesagens através do polinómio;
- Contagem do número de pesagens efetuadas;
- Cálculo do somatório das pesagens;
- Análise de tendências;
- Identificação do peso máximo e mínimo;
- Ativação dos alarmes luminosos e sonoro;
- Atualização dos limites superior e inferior.

O código exposto nas secções seguintes implementa cada uma das funções a serem suportadas pelo sistema de pesagem.

4.5.1 Valor das pesagens através do polinómio

A função “void setup()” é a função que vai obter o valor da tara da balança para que esta tenha uma referência de zero.

```
float peso,tara;

void setup() {
  // Definição da tara inicial.
  float reading = scale.read();
  // Polinómio de 3ª Ordem.
  peso = 3.73634e-20*pow(reading,3) + 2.56745e-13*pow(reading,2) -3.99307e-05*reading -
5.44063;
  tara=peso;
  // Define o zero da balança.
  peso=peso-tara;
}
```

A função “void loop()” é um ciclo infinito para a leitura do HX711.

```
void loop() {
  float reading = scale.read();
  // Polinómio de 3ª Ordem.
  peso = 3.73634e-20*pow(reading,3) + 2.56745e-13*pow(reading,2) -3.99307e-05*reading -
5.44063;
  // Caso varie até 50 gramas depois de ter sido definida a tara, a variável “tara” é redefinida.
  if(peso < tara+0.05){
    tara=peso;
  }
  peso=peso-tara;
}
```

4.5.2 Contagem de número de pesagens efetuadas

```
int contador = 0;
// Flag que indica se o peso atual é zero.
bool CZero = false;
void loop() {
  // Contador de pesagens efetuadas.
  if (CZero==true){
    if (peso>0.05){
      contador++;
      CZero = false;
    }
  }
  else if (peso <=0.05){
    CZero = true;
  }
}
```

4.5.3 Somatório do valor das pesagens

```
float somaTotal, mediaFinal;

void loop (){
  // Somatório das pesagens efetuadas.
  somaTotal+=mediaFinal;
}
```

4.5.4 Análise de tendências

```
// Sinal de tendência.
  if(numValor==1 && mediaFinal!=0){
    values[0] = mediaFinal;
  }else if(numValor==2 && mediaFinal!=0){
    values[1] = mediaFinal;
  }else if(numValor==3 && mediaFinal!=0){
    values[2] = mediaFinal;
  }else if(numValor==4 && mediaFinal!=0){
    values[3] = mediaFinal;
  }else if(numValor==5 && mediaFinal!=0){
    values[4] = mediaFinal;
  }

  if(values[4]!=0 && mediaFinal!=0){ // Quando o array está cheio faz o processo de
identificar a tendência.
  diff = abs(values[4] - values[0]);
    for (int i = 0; i < ARRAY_SIZE-1; i++) {
      if (values[i+1] >= values[i]) {
        diminuir = false;
        if(abs(values[i+1]-values[i])<=0.3){
          diminuir = true;
        }
      }
      if (values[i+1] <= values[i]) {
        aumentar = false;
        if(abs(values[i+1]-values[i])<=0.3){
          aumentar = true;
        }
      }
    }
  }
// Quando o array estiver cheio elimina o primeiro valor do array, passando todos os valores
para a posição anterior e regista o novo valor.
  if(values[4]!=0 && mediaFinal!=0){
    memmove(values, values + 1, (ARRAY_SIZE-1) * sizeof(float));
      values[ARRAY_SIZE-1] = mediaFinal;
  }
  if (numValor>=5){
```

```
  if (diff >= 0.5 && (aumentar || diminuir)) {
    LEDTEND = 1;
  } else{
    LEDTEND = 0;
  }
 }
```

4.5.5 Peso máximo e mínimo

```
float PesoMax=0.0,PesoMin=150.0;

void loop() {
 // Atualiza os valores máximo e mínimo.
 if (mediaFinal > PesoMax) {
  PesoMax = mediaFinal;
 }
 if (mediaFinal < PesoMin && mediaFinal > 0.1) {
  PesoMin = mediaFinal;
 }
}
```

4.5.6 Alarmes luminosos e sonoro

```
// Inicialmente as variáveis são definidas como 35 e 25, mas podem ser alteradas através do website.
int LimiteSup=35, LimiteInf=25;
void loop() {
        // Sinais luminosos/sonoro.
        if (peso>=LimiteSup){
          // Se o peso for maior que o limite superior o LED vermelho irá ligar.
         digitalWrite(12,HIGH);
        }else{
          // Se o peso for menor que o limite superior o LED vermelho irá desligar.
       digitalWrite(12,LOW);
        }

        if (peso<=LimiteInf & peso >0.1){
          // Se o peso for menor que o limite inferior o LED azul irá ligar.
          digitalWrite(13,HIGH);
        }else{
          // Se o peso for maior que o limite inferior o LED azul irá desligar.
```

```
          digitalWrite(13,LOW);
        }
        if (LEDTEND==1){
          // O LED amarelo da tendência irá ligar.
          digitalWrite(11,HIGH);
        }else{
          // O LED amarelo da tendência irá desligar.
          digitalWrite(11,LOW);
        }
}
```

4.5.7 Atualização do limite superior e inferior

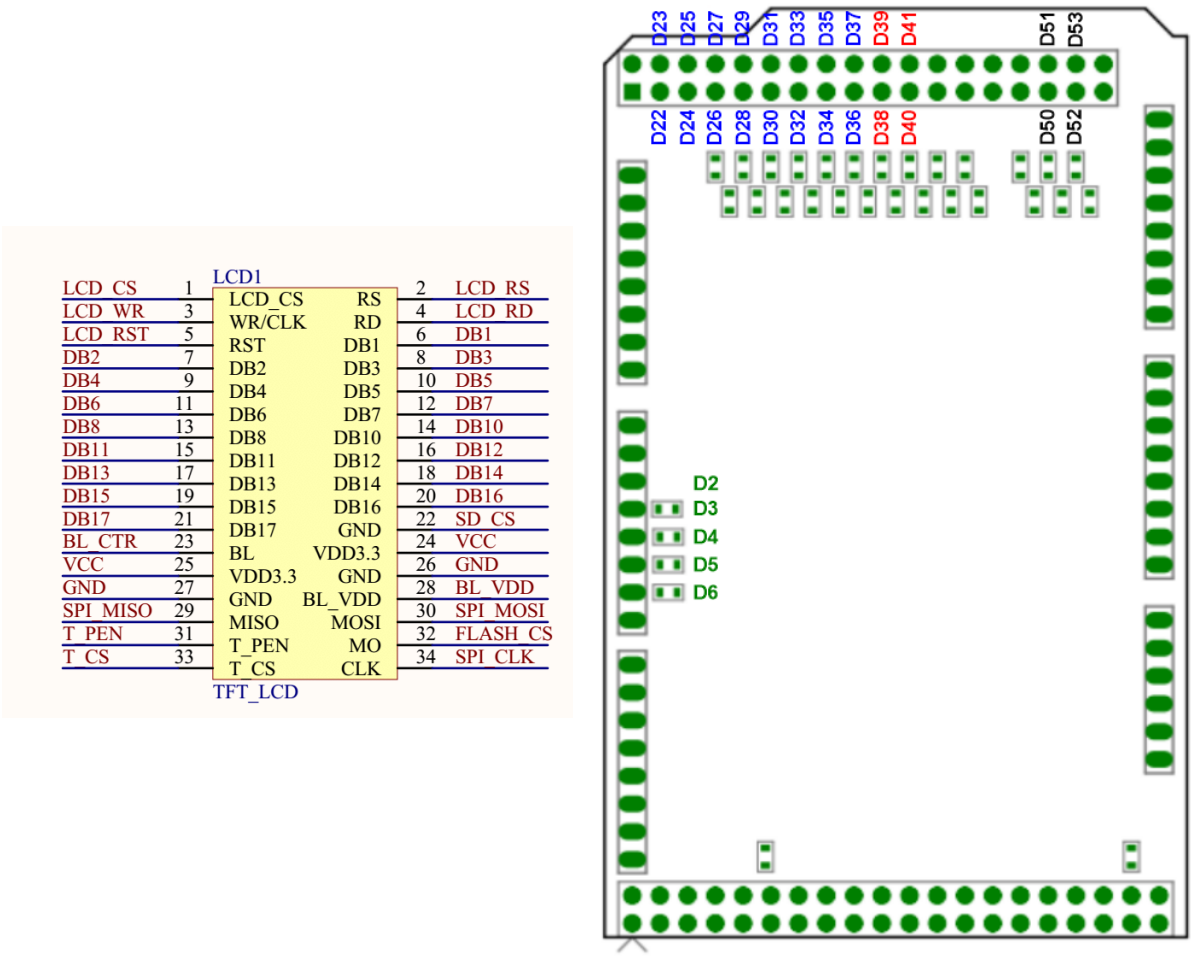
```
int LimiteSupDef, LimiteInfDef;

void loop() {
// Definição do novo “LimiteSup” e “LimiteInf” através dos dados enviados da base de dados. Exemplo de mensagem "<xxx,xxx>".
  if (Serial1.available()) {
    // Copia a mensagem que o ESP8266-01 envia para a porta série.
    String inputData = Serial1.readString();
    if(inputData.indexOf("<") != -1){ // Deteta se a mensagem que começa por o caracter "<".
      String dataLimites = inputData.substring(inputData.indexOf("<")+1, inputData.indexOf(">"));
      Serial.println(dataLimites);  // Retira os caracteres "<" e ">" da mensagem.
      // Separa as variáveis das vírgulas.
      LimiteSupDef= dataLimites.substring(0,dataLimites.indexOf(",")).toInt();
      LimiteInfDef= dataLimites.substring(dataLimites.indexOf(",")+1).toInt();
      Serial.println(LimiteSupDef);
      Serial.println(LimiteInfDef);
    }
  }
  // Faz a atualização dos limites superior e inferior se estes forem diferentes de zero e diferentes dos limites superior e inferior já existentes.
  if(LimiteSupDef!=0 && LimiteInfDef !=0){
    if(LimiteSup != LimiteSupDef || LimiteInf != LimiteInfDef){
      LimiteSup = LimiteSupDef;
      LimiteInf = LimiteInfDef;
    }
  }
}
```

4.6 Programação do ecrã local

Para a exibição local do valor das pesagens, do número da contagem, do peso máximo e mínimo e o somatório das pesagens, decidimos utilizar um ecrã local e escolhemos um LCD TFT de 3.2 polegadas.

Este ecrã local é ligado ao *shield* ITDB02 através dos pinos representados na Figura 30a e é alimentado com 5V. O *shield* ITDB02, por sua vez, é ligado ao Arduíno pelos pinos exibidos na Figura 30b. Os pinos digitais 38 a 41 (representados a vermelho) são usados para transmissão de dados, os pinos 2 a 6 (representados a verde) pela funcionalidade tátil do ecrã e os pinos 50 a 53 (representados a preto) são utilizados para a ligação e comunicação do cartão SD. O restante *pinout* do *shield* ITDB02 está exposto no Anexo 4.



a) Pinos de ligação do ecrã ao *shield* [18]. b) Pinos de ligação do *shield* ao Arduíno.

Figura 30 – Pinos de ligação do ecrã local para o Arduíno MEGA.

Na Figura 31 podemos observar as ligações descritas na Figura 30, onde o LCD e o *shield* encaixam sobre Arduíno.

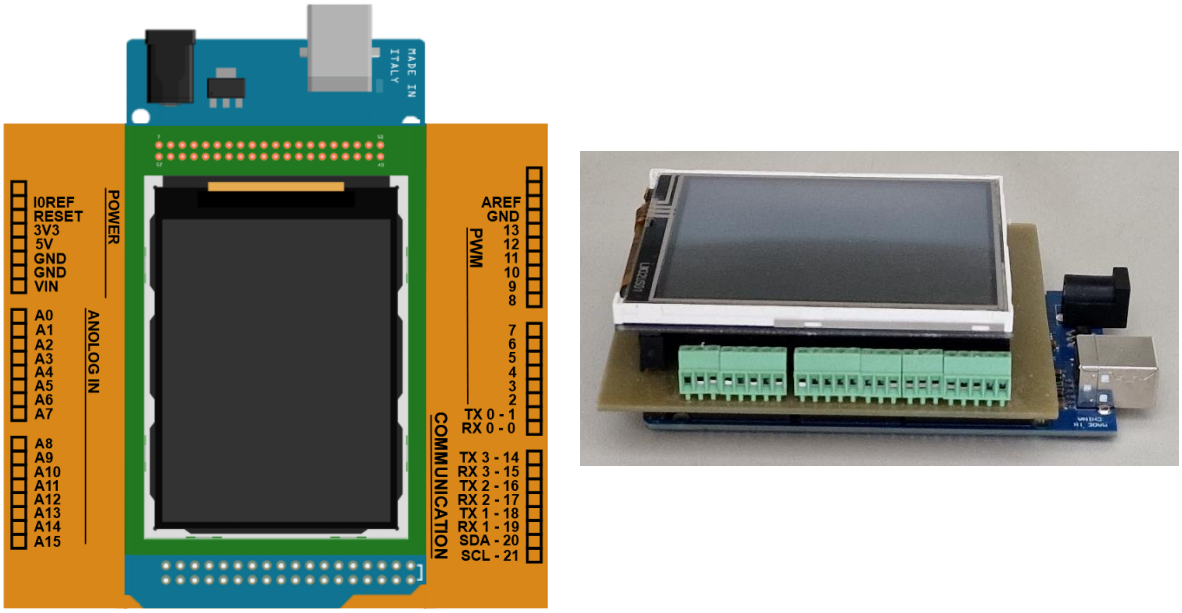


Figura 31 – Esquema de ligação entre o LCD, o *shield* e o Arduíno.

4.6.1 Código de programação

Para a utilização do ecrã local foi necessário utilizar a biblioteca UTFT a qual disponibiliza funções para definir o tamanho e cor da letra, exibição de imagens, desenho de formas, entre outros.

```
// Biblioteca do TFT LCD.
#include <UTFT.h>
// Inicializa o objeto UTFT com os pinos correspondentes ao shield ITDB02.
UTFT myGLCD(ITDB32S,38,39,40,41);
// Chama a função SmallFont da biblioteca UTFT.
extern uint8_t SmallFont[];
// Chama a função BigFont da biblioteca UTFT.
extern uint8_t BigFont[];
// Chama o array onde está guardada a imagem do logotipo do Departamento de Engenharia Eletrotécnica.
extern unsigned short icon_logo_dee[];
// Chama o array onde está guardada a imagem do logotipo do curso de Engenharia Eletrotécnica.
extern unsigned short Logo_LEE[];
```

Para organizar a apresentação da informação no ecrã local criamos duas telas: uma de arranque que apresenta o nome do projeto e uma segunda que exibe a contagem, o valor das pesagens, o valor máximo e mínimo e o somatório das pesagens.

Durante os primeiros 5 segundos após a inicialização do ecrã, é mostrada a tela da Figura 32, cujo código de programação é descrito abaixo.

Figura 32 – Tela de arranque do ecrã local.

```
void setup(){
  myGLCD.InitLCD(); // Inicializa o ecrã local.
  myGLCD.clrScr(); // Limpa a tela.
  myGLCD.drawBitmap(0, 0, 125, 62, icon_logo_dee); // Desenha a imagem no LCD.
  myGLCD.drawBitmap(195, 0, 125, 62, Logo_LEE); // Desenha a imagem no LCD.
  myGLCD.setFont(BigFont); // Define o tamanho da letra para “grande” (BigFont).
  myGLCD.setColor(255, 255, 255); // Define a cor do texto para branco.
  myGLCD.print("Sistema de Pesagem", CENTER, 100); // Exibe a mensagem no centro da tela.
  myGLCD.setFont(SmallFont); // Define o tamanho da letra para “pequeno” (SmallFont).
  myGLCD.print("Projeto realizado por:", 135, 180); // Exibe a mensagem.
  myGLCD.print("Joao Figueiredo 18111", 135, 200); // Exibe a mensagem.
  myGLCD.print("Pedro Amaral 18149", 135, 215); // Exibe a mensagem.
  delay(5000); // Aguarda 5 segundos.
  myGLCD.clrScr(); // Limpa a tela novamente.
}
```

Após 5 segundos surge a segunda tela, exibida na Figura 33 e o código que a gera é exposto abaixo.

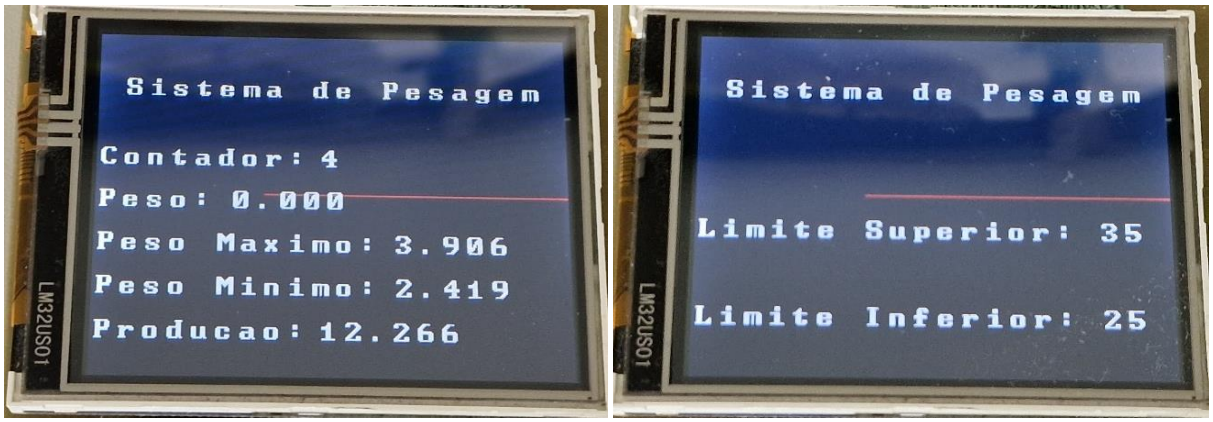


Figura 33 – Tela de apresentação dos valores das pesagens.

```
void loop() {
  // Envio de dados para o LCD.
    myGLCD.setFont(BigFont); // Define o tamanho da letra para “grande” (BigFont).
    myGLCD.print("Sistema de Pesagem", CENTER, 30); // Exibe a mensagem no centro da tela na posição Y=30.
    myGLCD.print("Contador: ", 0, 80); // Exibe a mensagem à esquerda da tela na posição Y=80.
    myGLCD.printNumI(contador, 150, 80); // Exibe o valor do contador.
    myGLCD.print("Peso: ", 0, 110); // Exibe a mensagem à esquerda da tela na posição Y=110.
    myGLCD.printNumF(peso, 3, 90, 110); // Exibe o valor da pesagem com três casas decimais.
    myGLCD.print("Peso Maximo: ", 0, 140); // Exibe a mensagem à esquerda da tela na posição Y=140.
    myGLCD.printNumF(PesoMax, 3, 200, 140); // Exibe o valor do peso máximo com três casas decimais.
    myGLCD.print("Peso Minimo: ", 0, 170); // Exibe a mensagem à esquerda da tela na posição Y=170.
    myGLCD.printNumF(PesoMin, 3, 200, 170); // Exibe o valor da peso mínimo com três casas decimais.
    myGLCD.print("Producao: ", 0, 200); // Exibe a mensagem à esquerda da tela na posição Y=200.
    myGLCD.printNumF(somaTotal, 3, 150, 200); // Exibe o somatório das pesagens com três casas decimais.
```

```
// Faz a atualização dos limites superior e inferior se estes forem diferentes de 0 e diferentes dos limites superior e inferior já existentes.
  if(LimiteSupDef!=0 && LimiteInfDef !=0){
    if(LimiteSup != LimiteSupDef || LimiteInf != LimiteInfDef){
      LimiteSup = LimiteSupDef;
      LimiteInf = LimiteInfDef;
      myGLCD.clrScr(); // Limpa a tela.
      myGLCD.print("Sistema de Pesagem", CENTER, 30); // Exibe a mensagem no centro da tela.
      myGLCD.print("Limite Superior: ", 0, 130); // Exibe a mensagem à esquerda da tela na posição Y=130.
      myGLCD.printNumI(LimiteSup, 270, 130); // Exibe o valor do novo limite superior.
      myGLCD.print("Limite Inferior: ", 0, 190); // Exibe a mensagem à esquerda da tela na posição Y=190.
      myGLCD.printNumF(LimiteInf, 270, 190); // Exibe o valor do novo limite inferior.
      delay(2000); // Espera 2 segundos.
      myGLCD.clrScr(); // Limpa a tela.
    }
  }
```

4.7 Ativação dos alarmes luminosos e sonoro

Para implementar os alarmes luminosos e sonoro de valores de pesagem acima (*HIGH*) ou abaixo (*LOW*) do valor desejado e para assinalar a tendência, utilizamos três LED's e um besouro.

Na Figura 34 podemos ver que o aviso *HIGH* é representado por um LED vermelho e um besouro ligados à entrada digital 12 do Arduíno, o aviso *LOW* é representado por um LED azul e um besouro ligados à entrada digital 13 e o aviso da tendência por um LED amarelo ligado à entrada digital 11. As resistências foram ligadas ao terminal negativo de cada LED para limitar a corrente e garantir o seu funcionamento seguro.

Os LED's são ativados pela função exposta na subsecção 4.5.6.

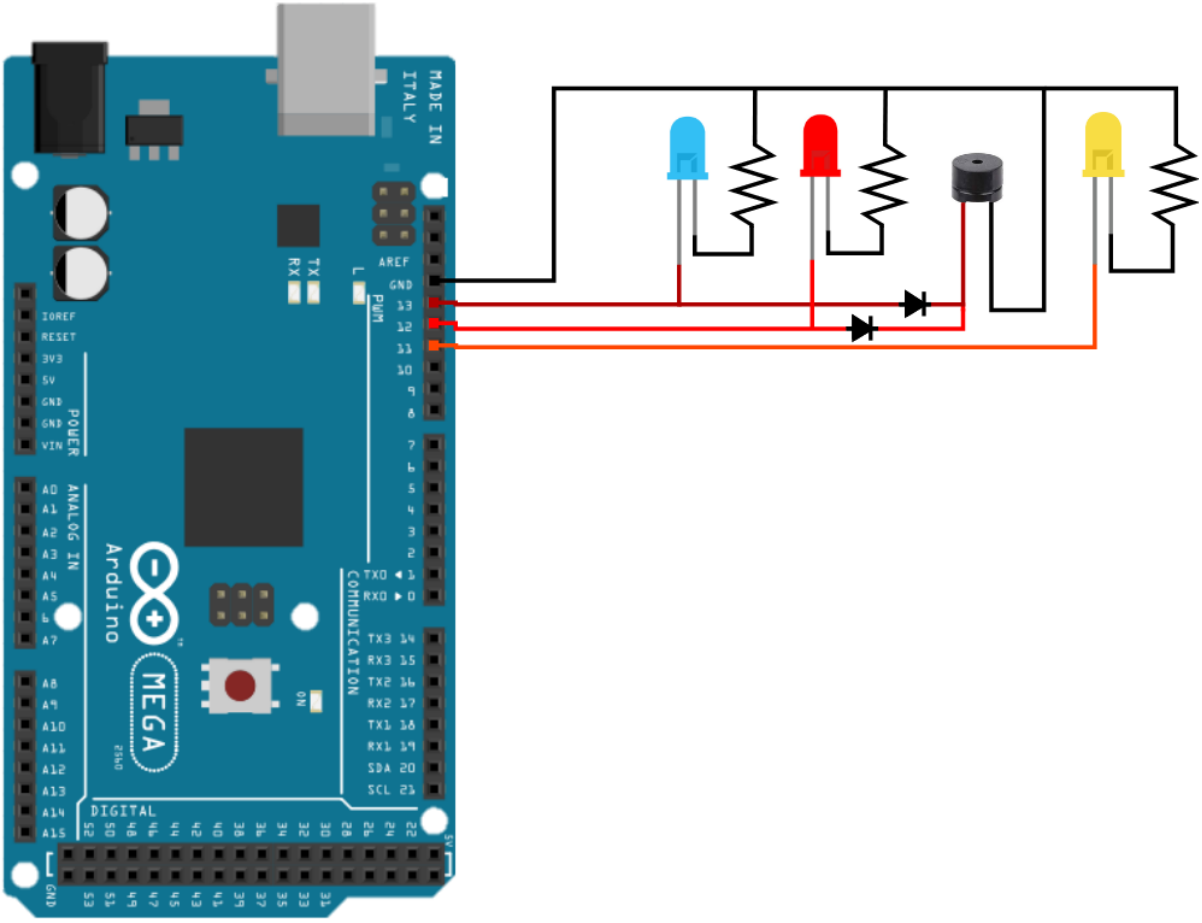


Figura 34 – Esquema de ligação dos alarmes luminosos e sonoro.

4.8 Envio de dados do Arduíno para o módulo de comunicação *wireless*

Para enviar os dados para a base de dados é necessário enviá-los primeiro para o módulo *WiFi* ESP8266-01. Para essa finalidade as portas TX e RX do Arduíno são utlizadas para enviar uma mensagem (*string*) que contém dados separados por vírgulas.

Um exemplo de mensagem é composto pelos dados nesta ordem: valor da pesagem, peso máximo, peso mínimo, contagem, limite superior e limite inferior de pesagem, LED máximo, LED mínimo, somatório de pesagens, limite superior e limite inferior definido pelo utilizador e LED da tendência: “ç83.895,89.332,77.217,7,80,50,1,0,327.239,80,50,0ç”.

Para enviar o valor da variável "peso" para a base de dados com a maior precisão possível foi necessário criar um *array* para armazenar os valores captados pela balança enquanto o objeto estiver sobre ela.

Quando um objeto é colocado ou retirado da balança, a captação dos valores de subida e descida afeta a precisão da balança, tornando-a mais baixa. Para contornar este problema, calculamos a média dos valores captados pela balança armazenados no *array*, incluindo os valores de subida e descida da pesagem. Em seguida, calculámos uma nova média apenas com valores acima de 80% da média total dos valores armazenados no *array*. Desta forma, os valores de subida e descida da pesagem são excluídos do cálculo da média enviada para a base de dados, garantindo assim valores de pesagens mais precisos. Para que esta solução funcione, após a colocação de um peso sobre a balança, **é necessário um tempo de espera para que o sistema estabilize e faça a medição**.

O código seguinte é responsável pelo envio dos dados do Arduíno para o módulo *WiFi* ESP8266-01.

```
#define MAX_VALORES 50 // Tamanho do array.
// Array para armazenar os vários valores do peso para depois calcular a média dos mesmos.
float ValoresP[MAX_VALORES];
int numValores = 0; // Número atual de valores no array.
float mediaTotal,mediaFinal,soma=0,sum=0;

void loop() {
// Tratamento dos dados do "peso" para encontrar uma média dos pesos através do array.
  if (peso > 0.1 && numValores < MAX_VALORES) {
    // Se o peso é maior que 0 e o array ainda não estiver cheio.
    ValoresP[numValores] = peso; // Adiciona o valor ao array.
    numValores++; // Incrementa o contador de valores.
  }
  // Calcula a média dos valores no array.
```

```
  if(peso<0.1 && numValores>0){
    for (int i = 0; i < numValores; i++) {
      sum += ValoresP[i];
    }
    mediaTotal = sum / numValores;
    // Calcula a média apenas dos valores acima de 80% da média do array, excluindo os valores de subida e descida da balança.
    for(int j=0 ; j < numValores; j++){
      if(ValoresP[j]>mediaTotal*0.8){
        soma += ValoresP[j];
        contaValores++;
      }
    }
    mediaFinal =  soma / contaValores;
    numValor++;
  }
// Envio do estado dos sinais luminosos/sonoro para a base de dados.
  if (mediaFinal>=LimiteSup){
    LEDMAX = 1;
  }else{
    LEDMAX = 0;
  }
  if (mediaFinal<=LimiteInf && mediaFinal > 0.1){
    LEDMIN = 1;
  }else{
    LEDMIN = 0;
  }
  // Envio de dados via porta série para o ESP8266-01.
  if(peso<=0.05 && mediaFinal>0.1){
    String data =  "ç" + String(mediaFinal, 3) + "," + String(PesoMax, 3) + "," +
String(PesoMin, 3) + "," + String(contador) + "," + String(LimiteSup) + "," +
String(LimiteInf) + "," + String(LEDMAX) + "," + String(LEDMIN) + "," +
String(somaTotal, 3) + "," + String(LimiteSupDef) + "," +String(LimiteInfDef) + "," +
String(LEDTEND) +"ç";
    Serial.println(data);
    delay(500);
    // Reset das variáveis para uma nova pesagem.
    mediaFinal=0;
    mediaTotal=0;
      for(int i = 0; i < numValores; i++) {
        ValoresP[i] = 0;
```

```
      }
    contaValores =0;
    numValores = 0;
    soma=0;
    sum = 0;
  }
}
```

4.9 Transmissão das leituras para a base de dados

Os valores das pesagens são transmitidos para a base de dados através do módulo *WiFi* ESP8266-01. Para facilitar a ligação utilizamos um adaptador que possui um divisor de tensão que reduz a tensão de 5V para 3.3V.

Na Figura 35 está exposto o esquema de ligação do ESP8266-01 ao Arduíno, no qual o pino RX deste módulo está ligado ao TX1 do Arduíno e o pino TX do módulo ao RX1 do Arduíno, devido ao facto do módulo necessitar de receber e enviar dados e os restantes pinos são a alimentação (5V e 0V).

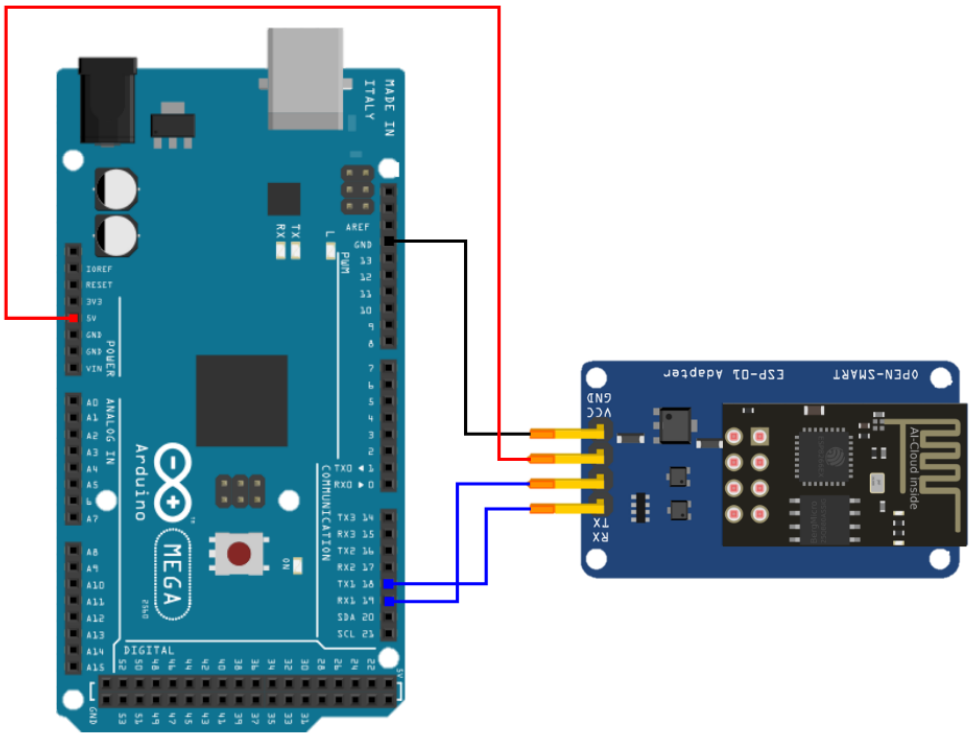


Figura 35 – Esquema de ligação do ESP8266-01 ao Arduíno.

4.9.1 Programação do módulo ESP8266-01

Para programar o módulo ESP8266-01 foi necessário utilizar um programador USB, exibido na Figura 36a. Foi ainda necessário fazer a adição de um interruptor de pressão entre o pino *GPOI0* com o pino *GND*, como se verifica na Figura 36b, para que seja possível ativar o modo de programação.

a) Sem alteração [5]. b) Com alteração.

Figura 36 – Programador USB.

Se o sistema de pesagem for deslocado do local inicial, é necessário mudar os dados da rede *WiFi* local alterando o conteúdo entre aspas nas linhas a azul do código abaixo.

Para ligar o ESP8266-01 à base de dados tivemos de criar um programa do qual são expostos trechos de código abaixo. O código completo encontra-se no Anexo 7.

```
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
WiFiClient client;
const char* ssid = "labelect"; // Nome da internet.
const char* password = "elect2016"; // Password da internet.
// Link para aceder ao ficheiro PHP que irá encaminhar os dados para a base de dados.
const char* serverName = "http://sistemadepesagem2023.000webhostapp.com/post-
data.php";

void setup() {
  Serial.begin(115200); // Inicia o monitor série.
  WiFi.begin(ssid, password); // Função que inicia o módulo WiFi.
  Serial.println("Connecting to the Wifi Network");
// Aguarda até ligar ao WiFi.
```

```
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
// Mensagem a ligação do módulo WiFi.
  Serial.print("WiFi is Connected at this IP Address : ");
  Serial.println(WiFi.localIP()); // Confirmação do endereço IP.
}
```

4.9.2 Funções de transferência de dados

Para realizar a transferência de dados entre o Arduíno e a base de dados, foram desenvolvidas duas funções: "*SendData*" e "*ReceiveData*". A primeira responsável por enviar os dados coletados pelo Arduíno para a base de dados, enquanto a segunda tem como função enviar dados armazenados na base de dados para o Arduíno.

Para que as duas funções possam funcionar em simultâneo foi necessário implementar duas novas bibliotecas, "*Thread"* e "*ThreadController"*.

A biblioteca "*Thread*" permite a criação e a execução simultânea de múltiplas tarefas, tais como "*SendData*" e "*ReceiveData*", garantindo que um possível atraso numa delas não afete o restante do código do programa, melhorando o seu desempenho.

A biblioteca "*ThreadController*", por sua vez possibilita o controlo eficiente das instruções em execução e possibilita iniciar, parar, suspender e retomar a execução de todas as tarefas ou instruções individuais de forma conveniente.

Funções "*Thread"* e "*ThreadController"*

```
// Biblioteca Thread, permite que várias funções sejam executadas simultaneamente.
#include <Thread.h>
#include <ThreadController.h>
ThreadController controll = ThreadController();
// Cria as threads.
Thread thread1 = Thread();
Thread thread2 = Thread();

void setup() {
  // A função "SendData" vai estar sempre a ser executada.
  thread1.onRun(SendData);
  thread1.setInterval(0);
  // A função "ReceiveData" vai ser executada de 5 em 5 segundos.
```

```
  thread2.onRun(ReceiveData);
  thread2.setInterval(5000);
  controll.add(&thread1);
  controll.add(&thread2);
}

void loop() {
  // Esta função é responsável por executar a lógica de uma thread num ciclo contínuo enquanto gere os recursos da thread.
  controll.run();
}
```

Função "*SendData*"

```
// Chave para que quando o ficheiro PHP receber os dados, só envia se este tiver recebido a chave no início da mensagem.
String apiKeyValue = "#54321";
float peso,PesoMax,PesoMin, peso_dup;
int LEDMAX,LEDMIN,MAX,MIN,contador=0;

void SendData(){
  // Verifica o estado da ligação WiFi.
  if(WiFi.status()== WL_CONNECTED){
    if (Serial.available()) {
      // Copia a mensagem que o Arduíno envia para a porta série.
      String datafull = Serial.readString();
      // Verifica se a mensagem começa e acaba com o caracter "ç". Por exemplo: "ç83.895,89.332,77.217,7,80,50,1,0,327.239,80,50,0ç".
      if(datafull.startsWith("ç") || datafull.indexOf("ç")) {
        int idxinicio = datafull.indexOf("ç");
        int idxfim = datafull.indexOf("ç",idxinicio+2);
        String data = datafull.substring(idxinicio+2,idxfim); // Retira o caracter "ç" da mensagem.
        // Separa os valores das vírgulas e guarda os valores em cada variável.
        int idx1 = data.indexOf(',');
        int idx2 = data.indexOf(',', idx1 + 1);
        int idx3 = data.indexOf(',', idx2 + 1);
        int idx4 = data.indexOf(',', idx3 + 1);
        int idx5 = data.indexOf(',', idx4 + 1);
        int idx6 = data.indexOf(',', idx5 + 1);
        int idx7 = data.indexOf(',', idx6 + 1);
```

```
      int idx8 = data.indexOf(',', idx7 + 1);
      int idx9 = data.indexOf(',', idx8 + 1);
      int idx10 = data.indexOf(',', idx9 + 1);
      int idx11 = data.indexOf(',', idx10 + 1);
        peso = data.substring(0,idx1).toFloat();
        PesoMax = data.substring(idx1 + 1, idx2).toFloat();
        PesoMin = data.substring(idx2 + 1, idx3).toFloat();
        contador = data.substring(idx3 + 1, idx4).toInt();
        MAX = data.substring(idx4 + 1, idx5).toInt();
        MIN = data.substring(idx5 + 1, idx6).toInt();
        LEDMAX = data.substring(idx6 + 1,idx7).toInt();
        LEDMIN = data.substring(idx7 + 1,idx8).toInt();
        somaTotal = data.substring(idx8 + 1,idx9).toFloat();
        LimiteSupDef = data.substring(idx8 + 1,idx9).toInt();
        LimiteInfDef = data.substring(idx9 + 1).toInt();
        LEDTEND = data.substring(idx11 + 1).toInt();
    }
  }
// Envia somente os dados se o peso for maior que 0 e se estes não tivessem sido enviados anteriormente.
if (peso>0.00 && peso!=peso_dup){
  // Cria a mensagem com os valores das variáveis.
  String httpRequestData = "api_key=" + apiKeyValue + "&LimiteSupDef=" + String(LimiteSupDef) + "&LimiteInfDef="+ String(LimiteInfDef) + "&Peso=" + String(peso, 3) + "&PesoMax=" + String(PesoMax, 3) + "&PesoMin=" + String(PesoMin, 3) + "&Contagem=" + String(contador) + "&LimiteSup=" + String(MAX) + "&LimiteInf=" + String(MIN) + "&somaTotal=" + String(somaTotal, 3) + "&PesoMaxSinal=" + String(LEDMAX) + "&PesoMinSinal=" + String(LEDMIN) + "&TendenciaSinal=" + String(LEDTEND) + "";
  HTTPClient http;
  http.begin(client,serverName);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  int httpResponseCode = http.POST(httpRequestData); // Envia os dados para a base de dados.

  if (httpResponseCode>0) { // Verifica se os dados foram enviados com sucesso.
    Serial.print("HTTP Response code: ");
    Serial.println(httpResponseCode);
    // Iguala a variável “peso_dup” a “peso”, para que não sejam enviadas duplicações para a base de dados.
    peso_dup = peso;
```

```
    }
    else {
      // Se ocorrer um erro ao enviar os dados para a base de dados, escreve o erro.
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  }
  }
  else {
    Serial.println("WiFi Disconnected");
  }
}
```

Função "*ReceiveData*"

```
// Variável para verificar se os dados que se recebe da base de dados são iguais aos já recebidos (previne que duplicações sejam enviadas para o Arduíno).
String DataLimiteDup;

void ReceiveData(){
 if (WiFi.status() == WL_CONNECTED) {
  HTTPClient http;
  http.begin(client, serverName);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");

  int httpResponseCode = http.GET();

    if (httpResponseCode == HTTP_CODE_OK) {
      String DataLimite = http.getString();
      if(DataLimite!=DataLimiteDup){
        delay(200);
        Serial.println(DataLimite);
        // Iguala a variável "DataLimiteDup" a "DataLimite", para que não sejam enviadas duplicações para o Arduíno.
        DataLimiteDup=DataLimite;
        delay(200);
      }
    }
  }
}
```

4.10 Estrutura da base de dados

Com o objetivo de armazenar a informação proveniente do sistema de pesagem, implementámos uma base de dados. Optamos pelo uso da ferramenta *web PHP MyAdmin* para desenvolver a base de dados em *MySQL*.

4.10.1 *PHP MyAdmin*

O *PHP MyAdmin* é uma ferramenta de *software* escrita na linguagem *PHP* destinada a administrar base de dados através de um navegador. Suporta uma ampla gama de operações em *MySQL* e *MariaDB*, tais como criar e excluir base de dados, tabelas e campos, além de executar consultas *SQL* diretamente da interface. Devido a estar em constante evolução, esta ferramenta constitui uma escolha fiável para administração de base de dados em *MySQL*.

Implementação da tabela

Para implementar a tabela responsável pelo armazenamento dos dados foi necessário criar as variáveis pretendidas e definir o tipo de cada (*int*, *float*, *boolean*).

Nome	Tipo	Tamanho/Valores*	Predefinido	Agrupamento (Collation)	Atributos	Nulo	A_I
ID	INT	11	Nenhum			☐	☑
LimiteSupDef	INT	11	Nenhum			☐	☐
LimiteInfDef	INT	11	Nenhum			☐	☐
Peso	FLOAT		Nenhum			☐	☐
PesoMax	FLOAT		Nenhum			☐	☐
PesoMin	FLOAT		Nenhum			☐	☐
Contagem	INT	11	Nenhum			☐	☐
LimiteSup	INT	11	Nenhum			☐	☐
LimiteInf	INT	11	Nenhum			☐	☐
somaTotal	FLOAT		Nenhum			☐	☐
PesoMaxSinal	BOOLEAN	1	Nenhum			☐	☐
PesoMinSinal	BOOLEAN	1	Nenhum			☐	☐
TendenciaSinal	BOOLEAN	1	Nenhum			☐	☐
Data	DATETIME		CURRENT_TIME			☐	☐

Figura 37 – Implementação da tabela.

Na Figura 37 expõem-se as variáveis da tabela da base de dados:

- A variável "ID" é utilizada para identificar sequencialmente as entradas de dados;
- As variáveis "LimiteSupDef" e "LimiteInfDef" armazenam os valores definidos pelo utilizador no *website* para estabelecer novos limites superior e inferior;
- A variável "Peso" guarda o valor da pesagem, enquanto "PesoMax" e "PesoMin" guardam os valores máximo e mínimo obtidos nas pesagens;
- A variável "Contagem" contém o número de pesagens efetuadas;
- As variáveis "LimiteSup" e "LimiteInf" armazenam os valores de limite máximo e mínimo que irão acionar os LED's quando ultrapassados;
- A variável "somaTotal" armazena o valor do somatório das pesagens efetuadas;
- As variáveis "PesoMaxSinal", "PesoMinSinal" e "TendenciaSinal" indicam o estado de cada LED (sendo "1" quando o LED está ativo e "0" quando desativo);
- A variável "Data" é responsável por registar a hora e a data em que a pesagem foi realizada.

Na Figura 38 está exposta uma ilustração com valores reais da tabela descrita no parágrafo anterior.

	ID	LimiteSupDef	LimiteInfDef	Peso	PesoMax	PesoMin	Contagem	LimiteSup	LimiteInf	somaTotal	PesoMaxSinal	PesoMinSinal	TendenciaSinal	Data
Edita Copiar Apagar	1	0	0	0.502	0.502	0.502	1	3	1	0.502	0	1	0	2023-05-10 17:36:07
Edita Copiar Apagar	2	0	0	0.508	0.508	0.502	2	3	1	1.011	0	1	0	2023-05-10 17:36:18
Edita Copiar Apagar	3	0	0	1.002	1.002	0.502	3	3	1	2.013	0	0	0	2023-05-10 17:36:25
Edita Copiar Apagar	4	0	0	1.006	1.006	0.502	4	3	1	3.019	0	0	0	2023-05-10 17:36:36
Edita Copiar Apagar	5	0	0	1.506	1.506	0.502	5	3	1	4.525	0	0	1	2023-05-10 17:36:45

Figura 38 – Tabela da base de dados.

4.10.2 Parametrização de novos limites

A fim de armazenar os novos valores dos limites superior e inferior introduzidos pelo utilizador foi necessário criar um ficheiro em *PHP* denominado de "*update_values*".

Quando o utilizador introduz novos valores dos limites superior e inferior, o ficheiro "*update_values*" entra em ação. Ao receber estas informações, realiza o processamento necessário para atualizar as variáveis correspondentes aos limites, e posteriormente as variáveis são enviadas para o Arduíno, permitindo assim que o sistema de pesagem utilize os novos limites definidos pelo utilizador.

O código do ficheiro "*update_values*" é exposto no Anexo 8.

4.10.3 Transferência de dados entre o módulo *WiFi* e a base de dados

Com a finalidade de transferir dados entre o módulo *WiFi* e a base de dados foi necessário criar um ficheiro em *PHP* intitulado "*post-data*".

O ficheiro "*post-data*" funciona como intermediário, recebendo os dados obtidos pelo Arduíno e enviando estes para a base de dados, garantindo o seu armazenamento. Este ficheiro também é responsável por enviar dados das variáveis atualizadas pelo ficheiro "*update_values*", da base de dados para o Arduíno. Isto permite que o sistema de pesagem tenha acesso a todas as informações, para que o seu funcionamento seja correto.

O código do ficheiro "*post-data*" é exposto no Anexo 9.

4.10.4 Exibição e acesso à base de dados

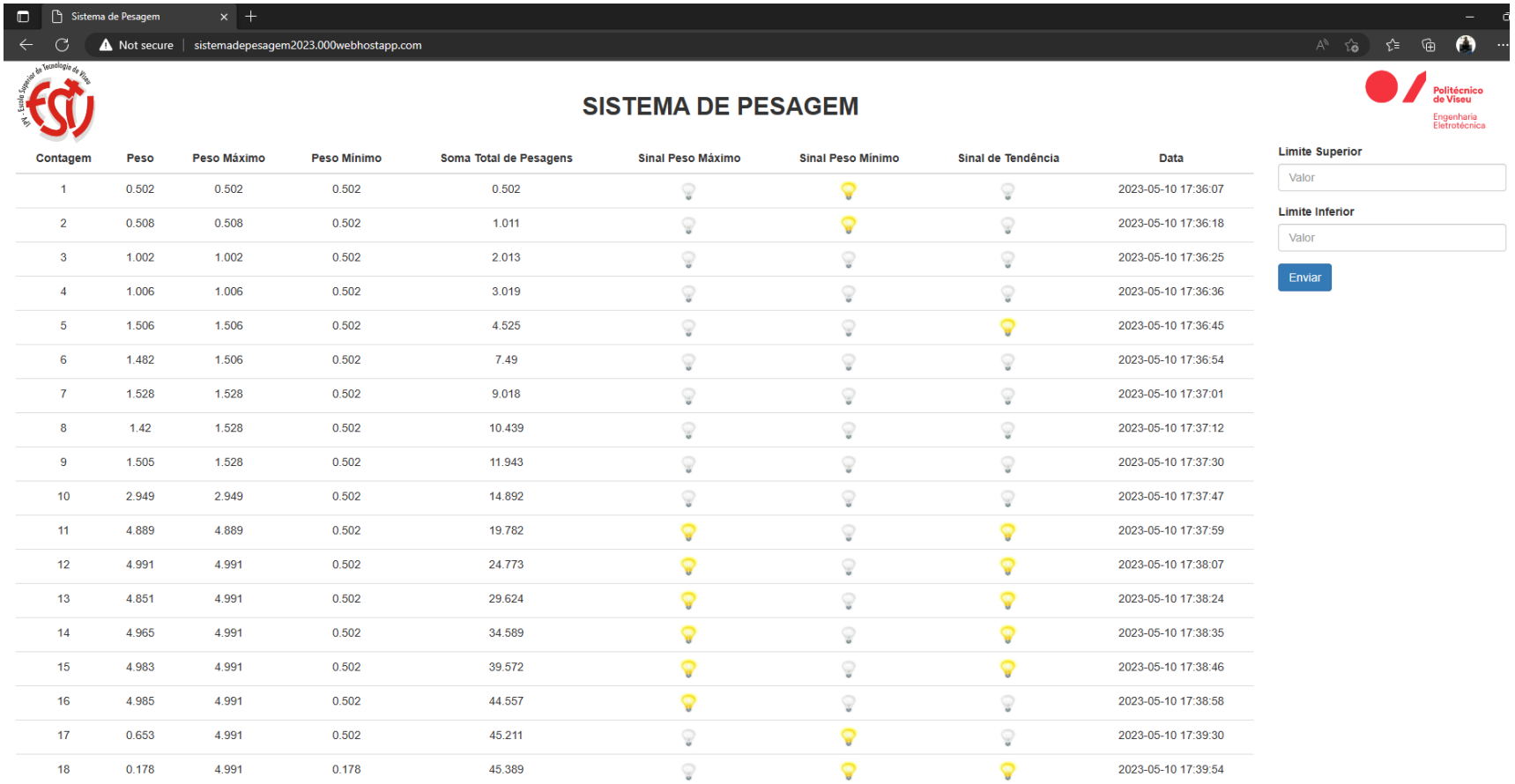
Para exibir a base de dados é necessário utilizar uma plataforma de hospedagem para o *website*, para a qual existem diversas opções, tais como www.webnode.com, www.wix.com, cloud.google.com, www.infinityfree.net e www.000webhost.com.

Optamos por utilizar a plataforma www.000webhost.com para hospedar o nosso *website*, https://sistemadepesagem2023.000webhostapp.com, devido a ser um serviço gratuito e de fácil utilização para pessoas sem grande experiência na área.

Para exibir a informação contida na base de dados, foi necessário elaborar um ficheiro em *HTML* chamado "*index*", o qual permite aceder à base de dados, recolher a informação e apresentá-la numa tabela. Além disso permite definir um novo limite superior e inferior.

O código do ficheiro "*index*" é exposto no Anexo 10.

Na Figura 39 expõe-se a visualização do *website* da base de dados que é estruturado pelo ficheiro "*index*".



Contagem	Peso	Peso Máximo	Peso Mínimo	Soma Total de Pesagens	Sinal Peso Máximo	Sinal Peso Mínimo	Sinal de Tendência	Data
1	0.502	0.502	0.502	0.502				2023-05-10 17:36:07
2	0.508	0.508	0.502	1.011				2023-05-10 17:36:18
3	1.002	1.002	0.502	2.013				2023-05-10 17:36:25
4	1.006	1.006	0.502	3.019				2023-05-10 17:36:36
5	1.506	1.506	0.502	4.525				2023-05-10 17:36:45
6	1.482	1.506	0.502	7.49				2023-05-10 17:36:54
7	1.528	1.528	0.502	9.018				2023-05-10 17:37:01
8	1.42	1.528	0.502	10.439				2023-05-10 17:37:12
9	1.505	1.528	0.502	11.943				2023-05-10 17:37:30
10	2.949	2.949	0.502	14.892				2023-05-10 17:37:47
11	4.889	4.889	0.502	19.782				2023-05-10 17:37:59
12	4.991	4.991	0.502	24.773				2023-05-10 17:38:07
13	4.851	4.991	0.502	29.624				2023-05-10 17:38:24
14	4.965	4.991	0.502	34.589				2023-05-10 17:38:35
15	4.983	4.991	0.502	39.572				2023-05-10 17:38:46
16	4.985	4.991	0.502	44.557				2023-05-10 17:38:58
17	0.653	4.991	0.502	45.211				2023-05-10 17:39:30
18	0.178	4.991	0.178	45.389				2023-05-10 17:39:54

Figura 39 – *Website* da base de dados.

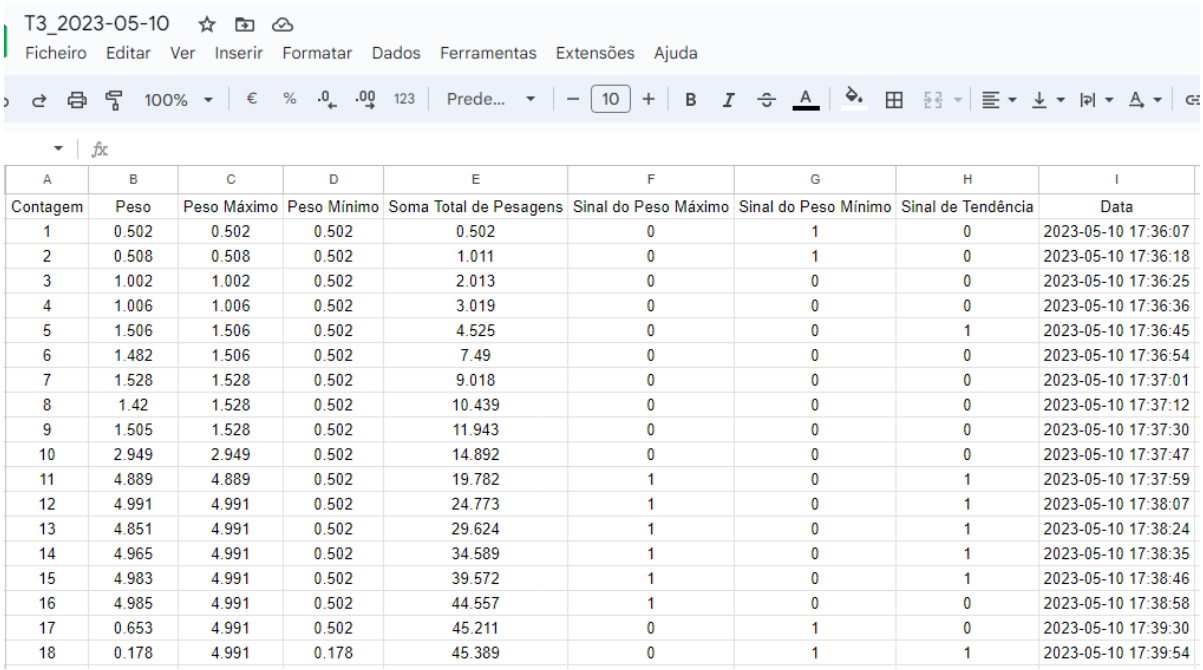
4.10.5 *Download* dos ficheiros

Com o objetivo de permitir efetuar o *download* dos ficheiros com as pesagens realizadas ao fim de cada turno em formato *Excel*, foi desenvolvido um ficheiro em *PHP* denominado de "*download_tabela*" e tem como objetivo permitir aos utilizadores obterem facilmente a tabela com os dados das pesagens.

A implementação do ficheiro "*download_tabela*" oferece um modo prático de disponibilizar as informações das pesagens ao utilizador, tendo acesso imediato ao ficheiro *Excel* e a possibilidade de efetuar análises das pesagens, como por exemplo gráficos para comparar os valores das pesagens efetuadas em cada turno.

O código do ficheiro "*download_tabela*" é exposto no Anexo 11.

Na Figura 40 expõe-se o conteúdo do ficheiro *Excel* denominado de T3_2023-05-10, com a informação das pesagens realizadas pelo turno 3 no dia 10 de Maio de 2023.



A	B	C	D	E	F	G	H	I
Contagem	Peso	Peso Máximo	Peso Mínimo	Soma Total de Pesagens	Sinal do Peso Máximo	Sinal do Peso Mínimo	Sinal de Tendência	Data
1	0.502	0.502	0.502	0.502	0	1	0	2023-05-10 17:36:07
2	0.508	0.508	0.502	1.011	0	1	0	2023-05-10 17:36:18
3	1.002	1.002	0.502	2.013	0	0	0	2023-05-10 17:36:25
4	1.006	1.006	0.502	3.019	0	0	0	2023-05-10 17:36:36
5	1.506	1.506	0.502	4.525	0	0	1	2023-05-10 17:36:45
6	1.482	1.506	0.502	7.49	0	0	0	2023-05-10 17:36:54
7	1.528	1.528	0.502	9.018	0	0	0	2023-05-10 17:37:01
8	1.42	1.528	0.502	10.439	0	0	0	2023-05-10 17:37:12
9	1.505	1.528	0.502	11.943	0	0	0	2023-05-10 17:37:30
10	2.949	2.949	0.502	14.892	0	0	0	2023-05-10 17:37:47
11	4.889	4.889	0.502	19.782	1	0	1	2023-05-10 17:37:59
12	4.991	4.991	0.502	24.773	1	0	1	2023-05-10 17:38:07
13	4.851	4.991	0.502	29.624	1	0	1	2023-05-10 17:38:24
14	4.965	4.991	0.502	34.589	1	0	1	2023-05-10 17:38:35
15	4.983	4.991	0.502	39.572	1	0	1	2023-05-10 17:38:46
16	4.985	4.991	0.502	44.557	1	0	0	2023-05-10 17:38:58
17	0.653	4.991	0.502	45.211	0	1	0	2023-05-10 17:39:30
18	0.178	4.991	0.178	45.389	0	1	1	2023-05-10 17:39:54

Figura 40 – Ficheiro *Excel* com a informação das pesagens realizadas ao fim de cada turno.

4.11 Esquema geral da unidade de captura do mesurando

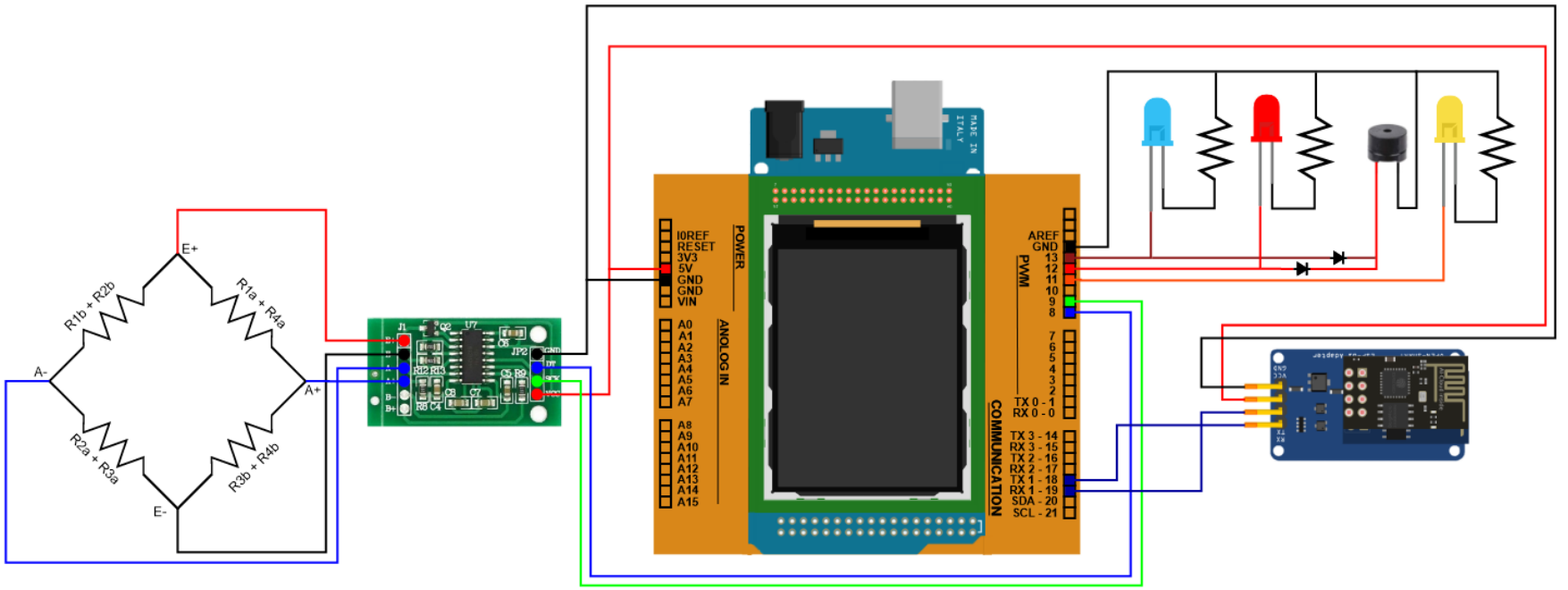


Figura 41 – Diagrama final da unidade de captura do mesurando.

4.12 Aspeto final do sistema de pesagem

A fim de concluir o sistema de pesagem projetemos um suporte utilizando o *software SOLIDWORKS* para sustentar a balança e para acomodar todos os componentes. O aspeto final do sistema de pesagem está exposto na Figura 42.

Figura 42 – Aspeto final do sistema de pesagem.

5. Conclusão

Com a concretização deste sistema de pesagem foram adquiridos conhecimentos em múltiplos domínios, garantindo maior precisão e eficiência dos processos. Especificamente a instrumentação industrial, a automação, a programação em linguagem *C++* e *PHP*, a estatística e a análise matemática que foram áreas fundamentais para o desenvolvimento do projeto.

A compreensão dos princípios da instrumentação industrial permitiu aplicar conhecimentos sobre as células de carga, como funcionam (extensometria) e os diferentes tipos existentes na indústria. Além disso foi possível obter informação sobre os componentes eletrónicos necessários para a aquisição do peso de objetos.

A automação, por sua vez, possibilitou a integração dos componentes para garantir que o sistema de pesagem operasse de forma autónoma. Para isso, utilizamos o microcontrolador Arduíno MEGA devido à sua versatilidade e capacidade de processamento, além de ter mais memória *flash* para armazenar programas mais extensos e ser compatível com a maioria dos *shields* disponíveis.

A programação em linguagem *C++* foi fundamental para a implementação de algoritmos necessários para o funcionamento do sistema, enquanto a programação em *PHP* permitiu a interação entre o utilizador e o sistema através da base de dados.

A utilização de técnicas estatísticas, como análise de tendência, cálculo de média, somatório de pesagens e a identificação de valores máximos e mínimos nos dados obtidos pelo sistema, possibilitou um funcionamento otimizado do mesmo, garantindo precisão nas medições realizadas. Por sua vez, a análise matemática foi utilizada para encontrar o melhor polinómio de aproximação das pesagens para que o erro do sistema de pesagem fosse o mais preciso possível e conhecido.

Por último, concluímos que o desenvolvimento deste projeto nos proporcionou a aquisição de conhecimentos e aptidões fundamentais em diversas áreas da indústria, enfatizando a importância da integração destas na conceção de sistemas automatizados.

REFERÊNCIAS

[1] Nachazel, T; "*What is a strain gauge and how does it work?*", disponível em: https://www.michsci.com/what-is-a-strain-gauge/?cn-reloaded=1, acedido em 26 de fevereiro de 2023.

[2] Piernas, R., Jefferson, A., Camacho, S., & Antônio De Brito, G.; "*Extensometria Básica*", disponível em: https://www.feis.unesp.br/Home/departamentos/engenhariacivil/nepae/extensometria-basica.pdf, acedido em 27 março de 2023.

[3] All About Circuits; "*Strain Gauges | Electrical Instrumentation Signals | Electronics Textbook*", disponível em: https://www.allaboutcircuits.com/textbook/direct-current/chpt-9/strain-gauges/, acedido em 27 março de 2023.

[4] Electronics Tutorial; "*Wheatstone Bridge Circuit and Theory of Operation*", disponível em: https://www.electronics-tutorials.ws/blog/wheatstone-bridge.html, acedido em 17 de fevereiro de 2023.

[5] HBK World; "*Wheatstone Bridge Circuit | Strain Gauge*", disponível em: https://www.hbkworld.com/en/knowledge/resource-center/articles/2023/strain-measurement-basics/strain-gauge-fundamentals/wheatstone-bridge-circuit#!ref_www.hbm.com, acedido em 27 de fevereiro de 2023.

[6] Tm2A; "*Célula de Carga Tipo Beam*", disponível em: https://www.tm2a.pt/produto/celula-de-carga-tipo-beam/, acedido em 25 de fevereiro de 2023.

[7] Load Cell Central; "*How a Load Cell Works | Types of Load Cells | Load Cell Central | Load Cell Central*", disponível em: https://www.800loadcel.com/white-papers/how-a-load-cell-works.html, acedido em 27 de fevereiro de 2023.

[8] Portal Célula de Carga; "*Strain Gages*", disponível em: http://celuladecarga.com.br/17/strain-gages-extensometros-eletricos/, acedido a 27 de fevereiro de 2023.

[9] Botnroll; "*HX711 - Amplificador p/ células de carga/sensores de peso duplo canal*", disponível em: https://www.botnroll.com/pt/forca-pressao-vibracao/2980-amplificador-p-c-lulas-de-carga-sensores-de-peso-hx711.html, acedido em 10 março de 2023.

[10] AVIA; "*HX711 - 24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales*", disponível em: https://www.digikey.com/htmldatasheets/production/1836471/0/0/1/hx711.html, acedido em 16 de abril de 2023.

[11] Arduino Store; "*Arduino Mega 2560 Rev3*", disponível em: https://store.arduino.cc/products/arduino-mega-2560-rev3, acedido em 16 de abril de 2023.

[12] ITEAD Wiki; "*ITDB02-3.2S Arduino Shield V1*", disponível em: https://wiki.iteadstudio.com/ITDB02-3.2S_Arduino_Shield_V1, acedido em 16 abril de 2023.

[13] AI-Thinker; "*ESP-01 WiFi Module Version 1*", disponível em https://www.microchip.ua/wireless/esp01.pdf, acedido em 16 de abril de 2023.

[14] Microsoft; "*Noções básicas da base de dados*", disponível em: https://support.microsoft.com/pt-pt/office/noções-básicas-da-base-de-dados-a849ac16-07c7-4a31-9948-3c8c94a7c204, acedido em 20 de abril de 2023.

[15] Catunda, H; "*SGBDS MAIS USADOS NO MUNDO*", disponível em: https://www.hashtagtreinamentos.com/sgbds-mais-usados-no-mundo-sql, acedido em 20 de abril de 2023.

[16] MySQL; "*MySQL 5.7 Reference Manual*", disponível em: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html, acedido em 28 de abril de 2023.

[17] Sanitas Onlineshop; "*With app for documenting your measured values SBF 70 Bluetooth® - diagnostic scale*", disponível em: https://sanitas-online.de/en/p/sbf-70-bluetooth-diagnostic-scale/, acedido em 24 de fevereiro de 2023.

[18] Geeetech Wiki; "*3.2TFT LCD*", disponível em: https://www.geeetech.com/wiki/index.php/3.2TFT_LCD, acedido em 25 de abril de 2023.

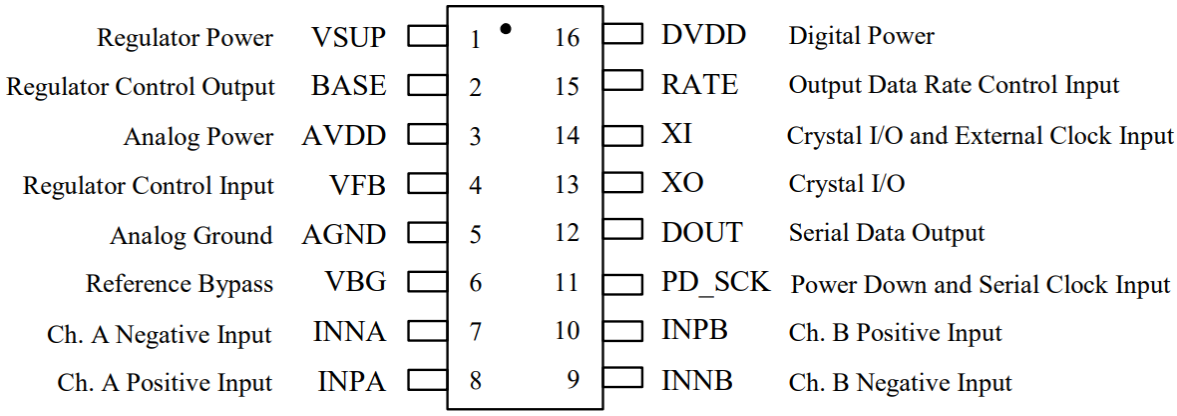
[19] Botnroll; "*Programador USB série para ESP-01 ESP8266*", disponível em: https://www.botnroll.com/pt/usb/3567-programador-usb-s-rie-para-esp-01-esp8266.html, acedido em 25 de abril de 2023.

[20] PHPMyAdmin contributors; "*Bringing MySQL to the web*", disponível em: https://www.phpmyadmin.net, acedido em 29 de abril de 2023.

Anexos

ANEXO 1 – ESPECIFICAÇÕES TÉCNICAS DO CIRCUITO HX711

Descrição do *pinout*



Pin #	Name	Function	Description
1	VSUP	Power	Regulator supply: 2.7 ~ 5.5V
2	BASE	Analog Output	Regulator control output（NC when not used）
3	AVDD	Power	Analog supply: 2.6 ~ 5.5V
4	VFB	Analog Input	Regulator control input（connect to AGND when not used）
5	AGND	Ground	Analog Ground
6	VBG	Analog Output	Reference bypass output
7	INA-	Analog Input	Channel A negative input
8	INA+	Analog Input	Channel A positive input
9	INB-	Analog Input	Channel B negative input
10	INB+	Analog Input	Channel B positive input
11	PD_SCK	Digital Input	Power down control (high active) and serial clock input
12	DOUT	Digital Output	Serial data output
13	XO	Digital I/O	Crystal I/O (NC when not used）
14	XI	Digital Input	Crystal I/O or external clock input, 0: use on-chip oscillator
15	RATE	Digital Input	Output data rate control, 0: 10Hz; 1: 80Hz
16	DVDD	Power	Digital supply: 2.6 ~ 5.5V

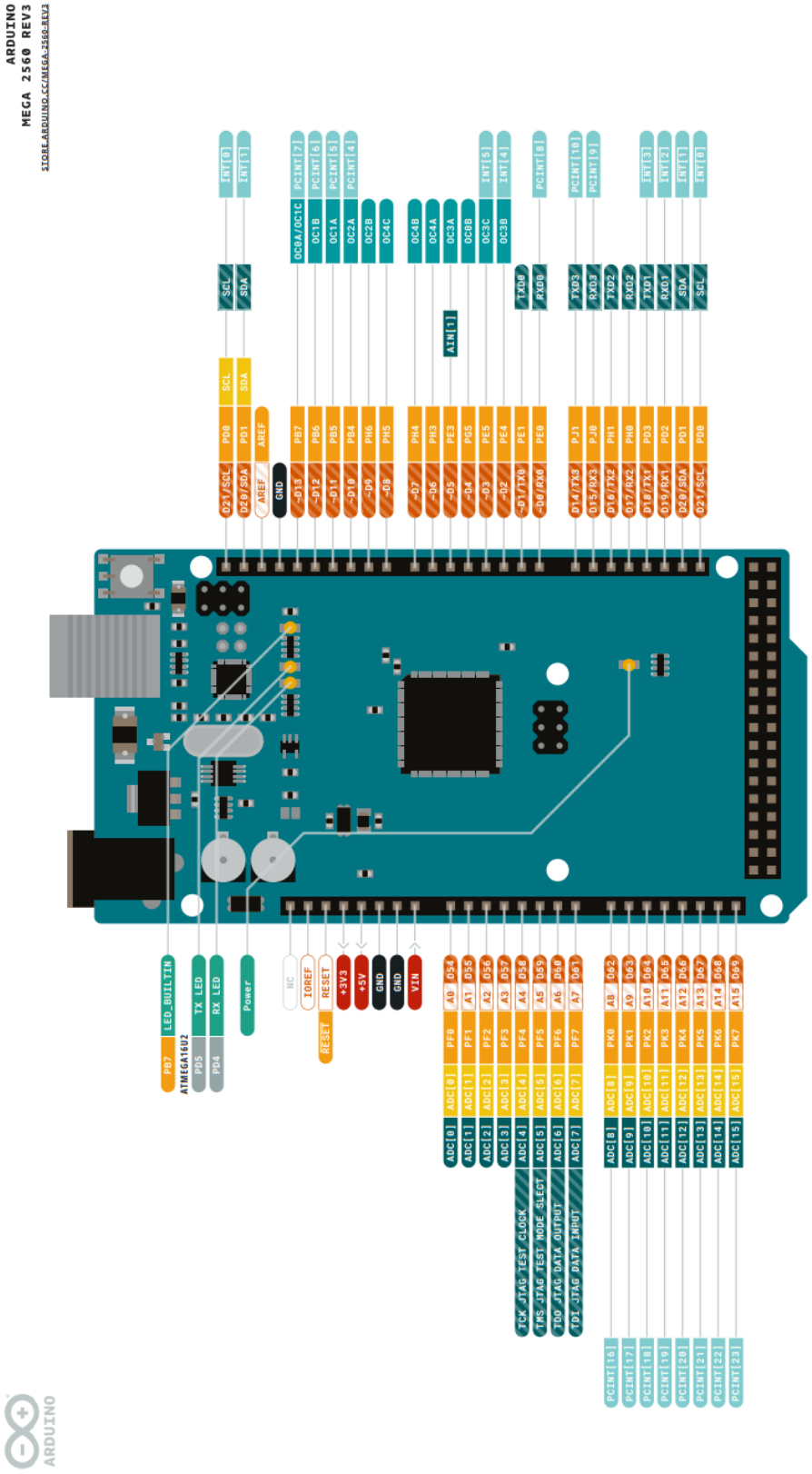
Características elétricas

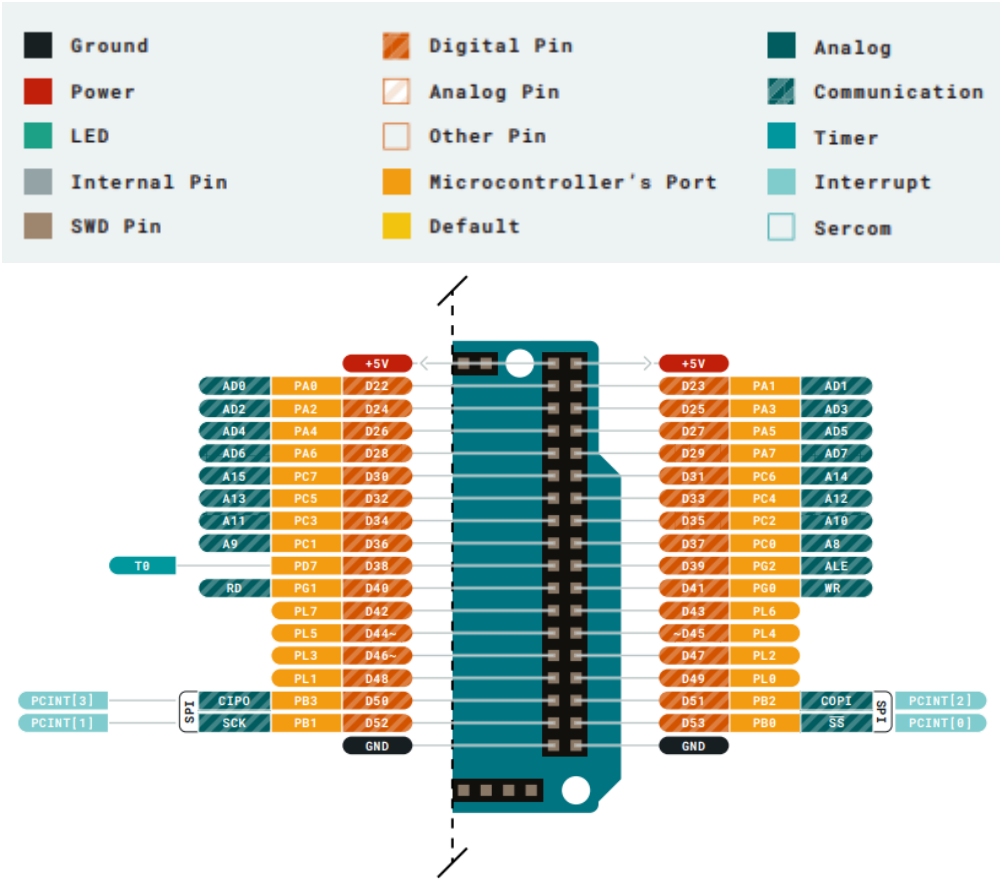
Parameter	Notes	MIN	TYP	MAX	UNIT
Full scale differential input range	V(inp)-V(inn)		±0.5(AVDD/GAIN)		V
Common mode input		AGND+1.2		AVDD-1.3	V
Output data rate	Internal Oscillator, RATE = 0		10		Hz
	Internal Oscillator, RATE = DVDD		80		
	Crystal or external clock, RATE = 0		f_{clk}/1,105,920		
	Crystal or external clock, RATE = DVDD		f_{clk}/138,240		
Output data coding	2's complement	800000		7FFFFF	HEX
Output settling time (1)	RATE = 0		400		ms
	RATE = DVDD		50		
Input offset drift	Gain = 128		0.2		mV
	Gain = 64		0.4		
Input noise	Gain = 128， RATE = 0		50		nV(rms)
	Gain = 128， RATE = DVDD		90		
Temperature drift	Input offset（Gain = 128）		±6		nV/℃
	Gain（Gain = 128）		±5		ppm/℃
Input common mode rejection	Gain = 128， RATE = 0		100		dB
Power supply rejection	Gain = 128， RATE = 0		100		dB
Reference bypass （V_{BG}）			1.25		V
Crystal or external clock frequency		1	11.0592	20	MHz
Power supply voltage	DVDD	2.6		5.5	V
	AVDD，VSUP	2.6		5.5	
Analog supply current (including regulator)	Normal		1400		μA
	Power down		0.3		
Digital supply current	Normal		100		μA
	Power down		0.2		

（1）Settling time refers to the time from power up, reset, input channel change and gain change to valid stable output data.

ANEXO 2 – ESPECIFICAÇÕES TÉCNICAS DO ARDUÍNO MEGA

Descrição do *pinout*




Ground
Power
LED
Internal Pin
SWD Pin
Digital Pin
Analog Pin
Other Pin
Microcontroller's Port
Default
Analog
Communication
Timer
Interrupt
Sercom
+5V
AD0 PA0 D22
AD2 PA2 D24
AD4 PA4 D26
AD6 PA6 D28
A15 PC7 D30
A13 PC5 D32
A11 PC3 D34
A9 PC1 D36
T0 PD7 D38
RD PG1 D40
PL7 D42
PL5 D44~
PL3 D46~
PL1 D48
PCINT[3] SPI CIPO PB3 D50
PCINT[1] SCK PB1 D52
GND
+5V
D23 PA1 AD1
D25 PA3 AD3
D27 PA5 AD5
D29 PA7 AD7
D31 PC6 A14
D33 PC4 A12
D35 PC2 A10
D37 PC0 A8
D39 PG2 ALE
D41 PG0 WR
D43 PL6
~D45 PL4
D47 PL2
D49 PL0
D51 PB2 COPI SPI PCINT[2]
D53 PB0 SS PCINT[0]
GND

ANEXO 3 - ESPECIFICAÇÕES TÉCNICAS DO ESP8266-01

Descrição do *pinout*

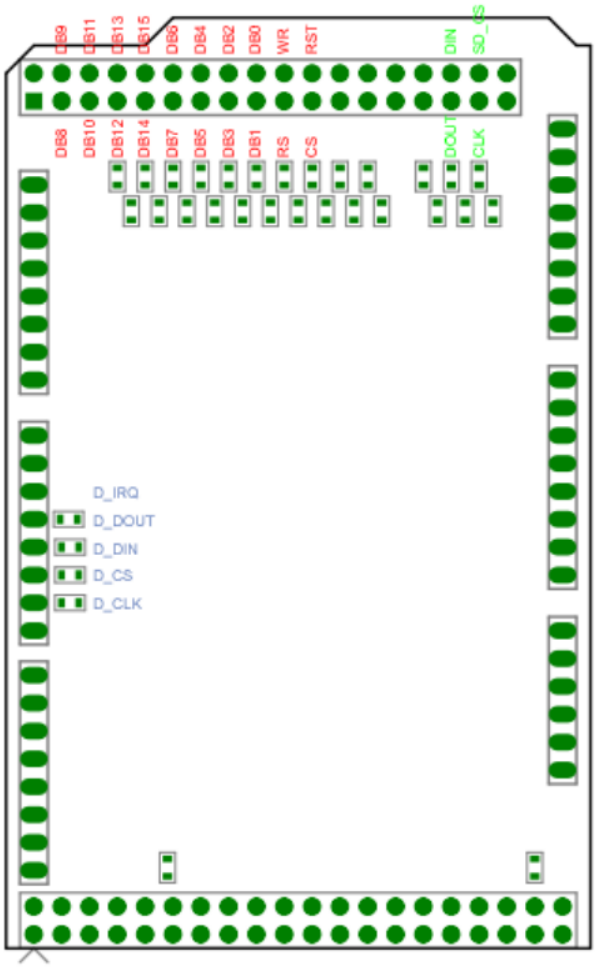
NO.	Pin Name	Function
1	GND	GND
2	GPIO2	GPIO,Internal Pull-up
3	GPIO0	GPIO,Internal Pull-up
4	RXD	UART0,data received pin RXD
5	VCC	3.3V power supply (VDD)
6	RST	1) External reset pin, active low 2) Can loft or external MCU
7	CH_PD	Chip enable pin. Active high
8	TXD	UART0,data send pin RXD

Modo de funcionamento

Mode	GPIO15	GPIO0	GPIO2
UART	Low	Low	High
Flash Boot	Low	High	High

ANEXO 4 – ESPECIFICAÇÕES TÉCNICAS DO *SHIELD* ITDB02

Descrição do *pinout*



Pino do Arduíno MEGA	Com ITDB02	
5V	VCC	
3V3	LED_A	RD
GND	GDN	
D53	SD_CS	-
D51	SD_IN	-
D50	SD_OUT	-
D52	SD_CLK	-
D37	DB0	-
D36	DB1	-

D35	DB2	-
D34	DB3	-
D33	DB4	
D32	DB5	
D31	DB6	
D30	DB7	
D22	DB8	-
D23	DB9	-
D24	DB10	-
D25	DB11	-
D26	DB12	
D27	DB13	
D28	DB14	
D29	DB15	
D38	RS	
D39	WR	
D40	CS	
D41	RST	
D6	D_CLK	
D5	D_CS	
D4	D_DIN	
D3	D_DOUT	
D2	D_IRQ	

ANEXO 5 - PROCEDIMENTO PARA CALIBRAGEM DA BALANÇA

Valores das pesagens obtidas através dos polinómios

Valores Conhecidos (kg)	Valor medido pelo HX711	Linear	2ª Ordem	3ª Ordem	4ª Ordem	5ª Ordem	6ª Ordem
0	-135200	-0.128	-0.058	-0.037	-0.064	0.000	0.019
0.145	-139200	0.034	0.103	0.123	0.097	0.157	0.175
0.25	-141500	0.127	0.195	0.215	0.190	0.248	0.264
0.5	-148050	0.391	0.458	0.477	0.453	0.506	0.520
0.75	-154275	0.643	0.708	0.726	0.704	0.751	0.763
1	-160550	0.896	0.959	0.977	0.957	0.999	1.009
1.25	-166880	1.151	1.214	1.230	1.211	1.249	1.257
1.5	-173330	1.412	1.472	1.488	1.471	1.503	1.510
1.75	-179580	1.664	1.723	1.738	1.722	1.750	1.755
2	-185750	1.913	1.971	1.985	1.971	1.995	1.998
2.25	-191900	2.161	2.218	2.231	2.218	2.238	2.240
2.5	-198400	2.424	2.479	2.491	2.480	2.496	2.496
2.75	-204700	2.678	2.732	2.744	2.733	2.745	2.745
3	-210600	2.916	2.969	2.980	2.971	2.979	2.978
3.25	-217000	3.175	3.225	3.236	3.228	3.233	3.231
3.5	-223320	3.430	3.479	3.489	3.482	3.484	3.481
3.75	-229550	3.681	3.729	3.739	3.733	3.732	3.728
4	-236050	3.944	3.990	3.999	3.994	3.990	3.985
4.25	-242250	4.194	4.239	4.247	4.243	4.237	4.231
4.5	-248850	4.461	4.504	4.511	4.509	4.500	4.493
4.75	-255050	4.711	4.753	4.760	4.758	4.747	4.740
5	-261000	4.951	4.992	4.998	4.997	4.984	4.976
5.25	-267170	5.200	5.240	5.245	5.246	5.230	5.222
5.5	-273600	5.460	5.498	5.503	5.504	5.486	5.478
5.75	-279950	5.716	5.753	5.757	5.759	5.740	5.731
6.025	-287150	6.007	6.042	6.046	6.049	6.027	6.018
6.275	-293250	6.253	6.287	6.290	6.294	6.271	6.262
6.525	-299850	6.519	6.552	6.555	6.559	6.535	6.525
6.775	-306000	6.768	6.799	6.801	6.806	6.781	6.771
7.045	-312750	7.040	7.071	7.072	7.078	7.051	7.041
7.295	-318700	7.280	7.310	7.310	7.317	7.289	7.279
7.545	-325250	7.545	7.573	7.573	7.580	7.551	7.542
7.795	-331450	7.795	7.822	7.821	7.829	7.799	7.790
8.075	-339010	8.100	8.125	8.124	8.133	8.102	8.093
8.325	-345350	8.356	8.380	8.379	8.388	8.356	8.347
8.575	-351650	8.611	8.633	8.631	8.641	8.609	8.600
8.825	-357980	8.866	8.887	8.885	8.895	8.863	8.854
9.07	-364200	9.117	9.137	9.134	9.145	9.112	9.104
9.32	-370400	9.368	9.386	9.383	9.395	9.361	9.353
9.585	-376950	9.632	9.650	9.646	9.658	9.624	9.616
9.835	-383100	9.880	9.897	9.892	9.905	9.871	9.864
10	-386400	10.013	10.029	10.025	10.037	10.004	9.996
10.5	-398900	10.518	10.532	10.526	10.540	10.506	10.499
11	-411200	11.015	11.026	11.020	11.034	11.000	10.995
11.5	-423500	11.511	11.520	11.513	11.528	11.495	11.490
12	-436200	12.024	12.031	12.023	12.038	12.006	12.002
12.5	-448300	12.512	12.517	12.509	12.524	12.493	12.490
13	-460800	13.017	13.019	13.010	13.026	12.997	12.995
13.5	-472800	13.501	13.502	13.492	13.508	13.480	13.479
14	-485500	14.014	14.012	14.002	14.018	13.992	13.992
14.5	-497400	14.495	14.491	14.480	14.496	14.472	14.473
15	-509900	14.999	14.993	14.982	14.998	14.976	14.978
15.5	-522100	15.492	15.484	15.472	15.489	15.468	15.471
16	-534900	16.008	15.999	15.986	16.003	15.984	15.988
16.5	-547200	16.505	16.493	16.480	16.497	16.481	16.485
17	-559950	17.020	17.006	16.993	17.009	16.995	17.001
17.5	-572400	17.522	17.507	17.493	17.509	17.498	17.504
18	-585050	18.033	18.016	18.001	18.017	18.008	18.016
18.5	-597400	18.532	18.512	18.498	18.513	18.507	18.515
19	-610390	19.056	19.035	19.020	19.035	19.032	19.041
19.5	-622720	19.554	19.531	19.516	19.530	19.530	19.539
20	-635340	20.063	20.039	20.023	20.037	20.039	20.049

21	-658400	20.994	20.966	20.950	20.964	20.970	20.982
22	-683100	21.991	21.960	21.944	21.956	21.968	21.980
23	-707650	22.982	22.948	22.932	22.943	22.959	22.972
24	-732200	23.973	23.936	23.920	23.930	23.951	23.964
25	-758300	25.027	24.987	24.970	24.979	25.004	25.018
26	-783300	26.036	25.994	25.977	25.984	26.014	26.027
27	-808400	27.050	27.004	26.987	26.993	27.027	27.040
28	-833920	28.080	28.032	28.015	28.019	28.056	28.069
29	-857500	29.032	28.982	28.965	28.968	29.008	29.020
30	-883000	30.061	30.009	29.993	29.993	30.036	30.047
31	-907500	31.050	30.996	30.980	30.979	31.024	31.034
32	-932000	32.039	31.983	31.968	31.965	32.011	32.020
33	-955450	32.986	32.928	32.913	32.909	32.956	32.964
34	-980300	33.989	33.929	33.915	33.910	33.957	33.964
35	-1005600	35.011	34.949	34.935	34.928	34.976	34.981
36	-1030270	36.007	35.943	35.930	35.922	35.970	35.973
37	-1054780	36.996	36.931	36.919	36.909	36.956	36.958
38	-1079375	37.989	37.923	37.912	37.901	37.946	37.947
39	-1104750	39.013	38.946	38.936	38.923	38.967	38.966
40	-1129300	40.004	39.936	39.927	39.913	39.955	39.952
41	-1154900	41.038	40.969	40.960	40.946	40.985	40.981
42	-1179450	42.029	41.959	41.951	41.936	41.973	41.967
43	-1204970	43.059	42.989	42.982	42.966	42.999	42.992
44	-1230000	44.070	43.999	43.993	43.976	44.006	43.997
45	-1255850	45.113	45.042	45.037	45.019	45.045	45.036
46	-1280400	46.104	46.033	46.029	46.011	46.033	46.022
47	-1305000	47.098	47.026	47.023	47.004	47.022	47.011
48	-1329400	48.083	48.011	48.009	47.990	48.004	47.992
49	-1354850	49.110	49.039	49.038	49.018	49.028	49.015
50	-1379500	50.105	50.034	50.034	50.015	50.019	50.007
51	-1404150	51.100	51.029	51.031	51.011	51.011	50.999
52	-1428850	52.097	52.027	52.030	52.010	52.006	51.993
53	-1454620	53.138	53.068	53.072	53.053	53.043	53.031
54	-1479300	54.134	54.065	54.070	54.051	54.038	54.025
54.7	-1496300	54.820	54.752	54.757	54.739	54.723	54.711
55.7	-1520770	55.808	55.741	55.747	55.729	55.709	55.698
56.7	-1545400	56.803	56.736	56.744	56.727	56.702	56.692
57.7	-1570000	57.796	57.731	57.739	57.723	57.694	57.685
58.7	-1595335	58.818	58.755	58.764	58.749	58.717	58.709
59.7	-1620050	59.816	59.754	59.764	59.750	59.715	59.708
60.7	-1644600	60.807	60.747	60.758	60.745	60.707	60.701
61.7	-1669100	61.796	61.737	61.749	61.738	61.697	61.693
62.7	-1693000	62.761	62.704	62.717	62.706	62.664	62.661
63.7	-1717360	63.745	63.689	63.703	63.694	63.650	63.649
64.7	-1742800	64.772	64.719	64.733	64.725	64.680	64.680
65.7	-1767560	65.771	65.720	65.735	65.729	65.683	65.685
66.7	-1792050	66.760	66.711	66.727	66.722	66.676	66.680
67.7	-1816650	67.753	67.707	67.723	67.720	67.674	67.679
68.7	-1843050	68.819	68.775	68.792	68.791	68.745	68.753
69.7	-1867700	69.814	69.773	69.790	69.791	69.746	69.755
70.7	-1892300	70.807	70.769	70.786	70.789	70.746	70.756
71.7	-1916900	71.800	71.765	71.783	71.787	71.746	71.758
72.7	-1942500	72.834	72.802	72.820	72.826	72.788	72.800
73.7	-1967450	73.841	73.812	73.830	73.838	73.804	73.817
74.8	-1993500	74.892	74.867	74.886	74.896	74.864	74.879
75.9	-2020000	75.962	75.941	75.959	75.971	75.944	75.959
77.02	-2046200	77.020	77.002	77.020	77.034	77.012	77.027
78.16	-2072500	78.082	78.068	78.086	78.101	78.085	78.099
79.25	-2098400	79.127	79.118	79.135	79.152	79.141	79.155
80.35	-2124470	80.180	80.174	80.191	80.210	80.205	80.218

82.1	-2167800	81.929	81.931	81.947	81.967	81.973	81.985
83.55	-2202500	83.330	83.338	83.353	83.375	83.388	83.399
84.55	-2228600	84.383	84.396	84.411	84.433	84.453	84.461
85.55	-2254600	85.433	85.451	85.464	85.487	85.513	85.520
86.55	-2279200	86.426	86.449	86.461	86.484	86.516	86.520
87.55	-2304350	87.442	87.469	87.480	87.503	87.541	87.543
88.64	-2332100	88.562	88.595	88.604	88.627	88.671	88.670
89.64	-2357800	89.599	89.638	89.646	89.668	89.716	89.713
90.64	-2382200	90.584	90.628	90.634	90.656	90.708	90.702
91.64	-2408200	91.634	91.683	91.688	91.708	91.763	91.754
92.64	-2432900	92.631	92.686	92.688	92.707	92.764	92.753
93.6	-2456300	93.576	93.636	93.636	93.654	93.711	93.698
94.6	-2476900	94.407	94.472	94.471	94.486	94.543	94.529
95.6	-2501800	95.413	95.483	95.480	95.492	95.548	95.532
96.6	-2527800	96.462	96.539	96.533	96.543	96.594	96.577
98.43	-2573600	98.311	98.400	98.389	98.391	98.431	98.414
99.43	-2598750	99.326	99.422	99.407	99.405	99.436	99.420
100.43	-2623800	100.338	100.439	100.422	100.415	100.434	100.420
101.43	-2648900	101.351	101.459	101.439	101.426	101.430	101.420
103.26	-2695900	103.248	103.370	103.342	103.317	103.285	103.286
104.26	-2721900	104.298	104.427	104.395	104.362	104.304	104.314
105.26	-2745900	105.267	105.403	105.367	105.326	105.241	105.262
106.26	-2769800	106.232	106.374	106.335	106.285	106.170	106.203

Diferença de erro entre os vários polinómios

Valores Conhecidos (kg)	Valor medido pelo HX711	Diferença de ERRO	Linear	2ª Ordem	3ª Ordem	4ª Ordem	5ª Ordem	6ª Ordem
0	-135200							
0.145	-139200		0.76578	0.292501	0.1545	0.329526	0.083996	0.203744
0.25	-141500		0.492747	0.220435	0.141605	0.24074	0.009128	0.056981
0.5	-148050		0.217524	0.084475	0.046767	0.093005	0.011401	0.03988
0.75	-154275		0.143277	0.056535	0.032464	0.061231	0.001437	0.017691
1	-160550		0.104134	0.040551	0.023293	0.043352	0.001386	0.008858
1.25	-166880		0.078873	0.029187	0.016015	0.030869	0.001194	0.005525
1.5	-173330		0.058802	0.018395	0.007949	0.01934	0.002219	0.006643
1.75	-179580		0.04908	0.01527	0.006749	0.015715	0.000244	0.003122
2	-185750		0.043403	0.014527	0.007442	0.014615	0.002735	0.000954
2.25	-191900		0.039346	0.014302	0.008328	0.014125	0.005331	0.004358
2.5	-198400		0.030449	0.0085	0.003426	0.008107	0.001796	0.00146
2.75	-204700		0.026105	0.006669	0.002319	0.006115	0.001724	0.001859
3	-210600		0.027869	0.010493	0.006729	0.009825	0.006902	0.007376
3.25	-217000		0.02315	0.00755	0.004295	0.006779	0.005133	0.005894
3.5	-223320		0.020028	0.005943	0.003119	0.005093	0.004479	0.005458
3.75	-229550		0.018291	0.005512	0.003056	0.004602	0.004827	0.005969
4	-236050		0.014046	0.002422	0.000293	0.00146	0.002411	0.003685
4.25	-242250		0.01315	0.002528	0.000678	0.001531	0.003053	0.004417
4.5	-248850		0.008766	0.000948	0.002544	0.001978	5.44E-05	0.00149
4.75	-255050		0.008242	0.000679	0.002058	0.001728	0.000695	0.002173
5	-261000		0.00979	0.001569	0.000378	0.00051	0.003239	0.004737
5.25	-267170		0.009498	0.001919	0.000901	0.000853	0.003848	0.005357
5.5	-273600		0.007324	0.000338	0.000521	0.000734	0.002494	0.004003
5.75	-279950		0.005901	0.000549	0.001265	0.001622	0.00179	0.00329
6.025	-287150		0.003032	0.002874	0.003445	0.003945	0.000364	0.001115
6.275	-293250		0.003507	0.001962	0.00242	0.003025	0.000656	0.002107
6.525	-299850		0.000853	0.004198	0.004547	0.005254	0.00148	6.16E-05
6.775	-306000		0.001076	0.003602	0.003857	0.004647	0.000821	0.00056
7.045	-312750		0.00068	0.003624	0.003785	0.004654	0.000791	0.000544
7.295	-318700		0.002	0.001991	0.002077	0.003006	0.000867	0.002157
7.545	-325250		2.15E-05	0.003662	0.003672	0.004662	0.000782	0.000461
7.795	-331450		1.7E-05	0.003424	0.003367	0.004406	0.000539	0.000653
8.075	-339010		0.003137	0.00624	0.006109	0.007202	0.003356	0.002222

8.325	-345350		0.003757	0.006616	0.006429	0.00756	0.00375	0.002669
8.575	-351650		0.004153	0.006785	0.006545	0.007708	0.003945	0.002917
8.825	-357980		0.004663	0.007081	0.006793	0.007983	0.004274	0.003301
9.07	-364200		0.00521	0.00743	0.007098	0.008311	0.004662	0.003742
9.32	-370400		0.005102	0.007135	0.006763	0.007994	0.004412	0.003546
9.585	-376950		0.004901	0.006747	0.006336	0.007583	0.004078	0.003269
9.835	-383100		0.004601	0.006282	0.005836	0.007095	0.003668	0.002912
10	-386400		0.001348	0.002938	0.002477	0.003736	0.000364	0.000363
10.5	-398900		0.001724	0.003016	0.002496	0.003768	0.000564	5.78E-05
11	-411200		0.001332	0.00236	0.00179	0.003065	3.95E-05	0.000483
11.5	-423500		0.000975	0.001763	0.001151	0.002422	0.000419	0.000845
12	-436200		0.001993	0.002558	0.001908	0.003169	0.000522	0.00019
12.5	-448300		0.000991	0.001363	0.000683	0.001928	0.000529	0.000774
13	-460800		0.001309	0.001498	0.000792	0.002017	0.000245	0.000406
13.5	-472800		0.000108	0.000138	0.000589	0.000611	0.00146	0.001546
14	-485500		0.001012	0.000886	0.00014	0.001314	0.000563	0.000573
14.5	-497400		0.000375	0.000633	0.001392	0.000249	0.001941	0.001885
15	-509900		5.37E-05	0.00044	0.001211	0.000101	0.001606	0.001485
15.5	-522100		0.000535	0.001036	0.001816	0.000739	0.002065	0.001886
16	-534900		0.000528	8.53E-05	0.000872	0.000168	0.000976	0.000741
16.5	-547200		0.000303	0.000409	0.0012	0.000196	0.001169	0.000885
17	-559950		0.00116	0.000354	0.000441	0.000524	0.000278	5.23E-05
17.5	-572400		0.001276	0.000385	0.000411	0.000515	0.000125	0.000246
18	-585050		0.001834	0.000862	6.8E-05	0.000954	0.000472	0.000879
18.5	-597400		0.001707	0.000664	0.000128	0.000718	0.000385	0.000823
19	-610390		0.002947	0.001833	0.001044	0.001849	0.001665	0.002132
19.5	-622720		0.002757	0.001581	0.000798	0.001562	0.001514	0.002005
20	-635340		0.003162	0.001928	0.00115	0.001874	0.001958	0.002469
21	-658400		0.000278	0.001602	0.002363	0.001716	0.001409	0.000872
22	-683100		0.000395	0.001805	0.002549	0.00198	0.001458	0.000905
23	-707650		0.000765	0.002248	0.002971	0.002479	0.001768	0.00121
24	-732200		0.001105	0.002649	0.00335	0.002932	0.002057	0.001503
25	-758300		0.001086	0.000518	0.001194	0.000851	0.000176	0.000715
26	-783300		0.0014	0.000248	0.000898	0.000625	0.000522	0.001039
27	-808400		0.001841	0.000156	0.000467	0.000262	0.000984	0.001472
28	-833920		0.002855	0.001139	0.000545	0.000686	0.002011	0.002463
29	-857500		0.001099	0.000635	0.001201	0.001117	0.000259	0.000673
30	-883000		0.002044	0.000291	0.000245	0.000219	0.001197	0.001568
31	-907500		0.001625	0.000139	0.000645	0.000671	0.000765	0.001091
32	-932000		0.001233	0.000537	0.001014	0.001088	0.000352	0.000631
33	-955450		0.00042	0.00219	0.002638	0.002755	0.001327	0.001092
34	-980300		0.000314	0.002083	0.002501	0.00266	0.001254	0.001068
35	-1005600		0.000306	0.001461	0.001848	0.002047	0.000676	0.000538
36	-1030270		0.000184	0.001574	0.001932	0.002166	0.000841	0.00075
37	-1054780		0.000105	0.001853	0.002182	0.002446	0.001178	0.001131
38	-1079375		0.000289	0.002023	0.002324	0.002616	0.001412	0.001407
39	-1104750		0.000344	0.001376	0.001647	0.001964	0.000835	0.000871
40	-1129300		0.000112	0.001589	0.001832	0.00217	0.00112	0.001194
41	-1154900		0.000926	0.000755	0.00097	0.001328	0.000365	0.000474
42	-1179450		0.000692	0.000967	0.001155	0.001527	0.000654	0.000794
43	-1204970		0.001379	0.000256	0.000417	0.000801	2.4E-05	0.000192
44	-1230000		0.001586	2.37E-05	0.000158	0.000552	0.000128	6.41E-05
45	-1255850		0.002519	0.000937	0.000828	0.000428	0.001006	0.000793
46	-1280400		0.00227	0.000717	0.000633	0.00023	0.00071	0.000481
47	-1305000		0.002075	0.000553	0.000492	8.82E-05	0.000471	0.00023
48	-1329400		0.00172	0.00023	0.000192	0.00021	7.55E-05	0.000174
49	-1354850		0.002245	0.000787	0.000772	0.000374	0.000561	0.000307
50	-1379500		0.002102	0.000679	0.000685	0.000294	0.000387	0.000133
51	-1404150		0.001966	0.000578	0.000605	0.000222	0.000224	2.78E-05
52	-1428850		0.001873	0.000522	0.000568	0.000196	0.000111	0.000135
53	-1454620		0.002599	0.001285	0.001351	0.000992	0.00082	0.000584
54	-1479300		0.002483	0.001208	0.001292	0.000947	0.000697	0.000472
54.7	-1496300		0.0022	0.000952	0.001048	0.000714	0.000413	0.000198

55.7	-1520770		0.001943	0.000734	0.000847	0.00053	0.00016	3.88E-05
56.7	-1545400		0.001808	0.00064	0.000768	0.000469	3.59E-05	0.000144
57.7	-1570000		0.001658	0.000531	0.000673	0.000394	9.7E-05	0.000256
58.7	-1595335		0.002017	0.000933	0.001089	0.000832	0.000288	0.000152
59.7	-1620050		0.001946	0.000904	0.001073	0.000838	0.00025	0.000137
60.7	-1644600		0.001767	0.000768	0.000949	0.000737	0.000111	2.26E-05
61.7	-1669100		0.001561	0.000606	0.000798	0.000609	4.7E-05	0.00011
62.7	-1693000		0.000976	6.42E-05	0.000265	0.0001	0.000578	0.000617
63.7	-1717360		0.0007	0.000167	4.29E-05	9.76E-05	0.000792	0.000806
64.7	-1742800		0.001107	0.000286	0.000504	0.00039	0.000315	0.000302
65.7	-1767560		0.001083	0.000309	0.000534	0.000445	0.000261	0.000224
66.7	-1792050		0.000897	0.000169	0.0004	0.000336	0.000364	0.000303
67.7	-1816650		0.000782	0.0001	0.000337	0.000299	0.00039	0.000306
68.7	-1843050		0.001728	0.001096	0.001337	0.001326	0.000658	0.000765
69.7	-1867700		0.001633	0.001049	0.001293	0.001307	0.000665	0.000791
70.7	-1892300		0.001513	0.000977	0.001222	0.001262	0.000652	0.000795
71.7	-1916900		0.001396	0.000908	0.001154	0.001218	0.000647	0.000805
72.7	-1942500		0.001837	0.0014	0.001646	0.001734	0.00121	0.00138
73.7	-1967450		0.00191	0.001523	0.001768	0.001879	0.001405	0.001586
74.8	-1993500		0.001236	0.000901	0.001143	0.001277	0.000862	0.00105
75.9	-2020000		0.00082	0.000539	0.000778	0.000934	0.000583	0.000774
77.02	-2046200		8.67E-07	0.000229	5.7E-06	0.000182	0.000101	9.04E-05
78.16	-2072500		0.001002	0.001176	0.000948	0.000753	0.000964	0.000776
79.25	-2098400		0.001549	0.001669	0.001448	0.001237	0.001375	0.001193
80.35	-2124470		0.002119	0.002186	0.001973	0.001747	0.001809	0.001638
82.1	-2167800		0.002083	0.00206	0.001863	0.001617	0.001552	0.001405
83.55	-2202500		0.002636	0.002539	0.002358	0.0021	0.001934	0.001813
84.55	-2228600		0.00197	0.001818	0.00165	0.001386	0.001146	0.001047
85.55	-2254600		0.001367	0.00116	0.001005	0.000738	0.000428	0.000354
86.55	-2279200		0.001431	0.001171	0.001031	0.000764	0.000391	0.000342
87.55	-2304350		0.001239	0.000926	0.000802	0.000537	0.000105	8.26E-05
88.64	-2332100		0.000883	0.00051	0.000404	0.000145	0.000345	0.000336
89.64	-2357800		0.000454	2.58E-05	6.17E-05	0.000312	0.000848	0.00081
90.64	-2382200		0.000614	0.000133	6.43E-05	0.000174	0.000745	0.00068
91.64	-2408200		6.61E-05	0.000472	0.00052	0.000742	0.00134	0.001247
92.64	-2432900		9.62E-05	0.000496	0.000522	0.000726	0.001338	0.00122
93.6	-2456300		0.000259	0.000384	0.00039	0.000572	0.001185	0.001047
94.6	-2476900		0.002036	0.001349	0.001363	0.001202	0.000599	0.000752
95.6	-2501800		0.00196	0.001219	0.001256	0.001125	0.000546	0.000715
96.6	-2527800		0.001426	0.000627	0.000691	0.000595	5.87E-05	0.000237
98.43	-2573600		0.001207	0.000306	0.00042	0.000396	1.33E-05	0.000163
99.43	-2598750		0.001041	8.43E-05	0.000227	0.000249	6.09E-05	0.000101
100.43	-2623800		0.000918	9.43E-05	7.81E-05	0.000152	3.63E-05	0.000101
101.43	-2648900		0.000778	0.000291	8.72E-05	4.2E-05	2.21E-06	0.000102
103.26	-2695900		0.000112	0.001063	0.000799	0.000551	0.000239	0.000249
104.26	-2721900		0.000365	0.001599	0.001299	0.000979	0.000426	0.000523
105.26	-2745900		6.61E-05	0.001354	0.00102	0.000627	0.000178	1.68E-05
106.26	-2769800		0.000265	0.001076	0.000707	0.000237	0.000851	0.000538
		Media dos ERROS (%)	1.663	0.712	0.462	0.777	0.192	0.357

ANEXO 6 – CÓDIGO DE PROGRAMAÇÃO DO ARDUÍNO

```
#include <Arduino.h>
#include "HX711.h"
#include <Wire.h>
#include <UTFT.h> // Biblioteca do TFT LCD.
#define MAX_VALORES 50 // Tamanho do array.

UTFT myGLCD(ITDB32S,38,39,40,41); // Inicializa o objeto UTFT com os pinos correspondentes ao ITDB02.
extern uint8_t SmallFont[]; // Chama a função "SmallFont" da biblioteca UTFT.
extern uint8_t BigFont[]; // Chama a função "BigFont" da biblioteca UTFT.
extern unsigned short icon_logo_dee[]; // Chama o array onde está guardada a imagem do logotipo do Departamento de Engenharia Eletrotécnica.
extern unsigned short Logo_LEE[]; // Chama o array onde está guardada a imagem do logotipo do curso de Engenharia Eletrotécnica.

float ValoresP[MAX_VALORES]; // Array para armazenar os vários valores do peso para depois se fazer uma média dos mesmos.
int numValores = 0; // Número atual de valores no array.
const int ARRAY_SIZE = 5; // Tamanho do array para a tendência.
float values[] = {0 , 0, 0, 0, 0}; // Array da tendência.
int numValor=0;
// Variáveis do tipo float.
float peso, tara, soma_pesos, mediaTotal, mediaFinal, readValues, diff,soma=0, sum=0, PesoMax=0.0 ,PesoMin=150.0, somaTotal=0;
// Variáveis do tipo int.
int LimiteSupDef, LimiteInfDef, LEDMAX, LEDMIN, LEDTEND ,LimiteSup=35, LimiteInf=25, contador=0, contaValores=0;
// Flags
bool CZero = false; // Indica se o peso atual é zero.
bool aumentar = true, diminuir = true; // Flags para tendência.

// Ligação do circuito HX711.
const int LOADCELL_DOUT_PIN = 8;
const int LOADCELL_SCK_PIN = 9;
HX711 scale;
```

```
void setup() {
  // Pinos LED's
  pinMode(11, OUTPUT); // LED tendência (amarelo).
  pinMode(12, OUTPUT); // LED sinal máximo (vermelho).
  pinMode(13, OUTPUT); // LED sinal mínimo (azul).

  // Inicialização da balança e do monitor série.
  Serial.begin(115200);
  Serial1.begin(115200);
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN); // Inicialização do circuito HX711.
  myGLCD.InitLCD(); // Inicializa o ecrã.
  myGLCD.clrScr(); // Limpa a tela.
  myGLCD.drawBitmap(0, 0, 125, 62, icon_logo_dee); // Desenha a imagem no LCD.
  myGLCD.drawBitmap(195, 0, 125, 62, Logo_LEE); // Desenha a imagem no LCD.
  myGLCD.setFont(BigFont); // Define o tamanho de letra para "grande" (BigFont).
  myGLCD.setColor(255, 255, 255); // Define a cor do texto para branco.
  myGLCD.print("Sistema de Pesagem", CENTER, 100); // Exibe a mensagem no centro da tela.
  myGLCD.setFont(SmallFont); // Define o tamanho da letra para "pequeno" (SmallFont).
  myGLCD.print("Projeto realizado por:", 135, 180); // Exibe a mensagem.
  myGLCD.print("Joao Figueiredo 18111", 135, 200); // Exibe a mensagem.
  myGLCD.print("Pedro Amaral 18149", 135, 215); // Exibe a mensagem.
  delay(5000); // Aguarda 5 segundos.
  myGLCD.clrScr(); // Limpa novamente a tela.

// Definição da tara inicial.
  float reading = scale.read();
// Polinómio de 3ª Ordem.
  peso = 3.73634e-20*pow(reading,3)+ 2.56745e-13*pow(reading,2) -3.99307e-05*reading - 5.44063;
  tara=peso;
// Define o zero da balança.
  peso=peso-tara;
}

void loop() {
  float reading = scale.read();
// Polinómio de 3ª Ordem.
  peso = 3.73634e-20*pow(reading,3)+ 2.56745e-13*pow(reading,2) -3.99307e-05*reading - 5.44063;
```

```
 if(peso < tara+0.05){ // Caso varie até 50 gramas ele define como um novo 0.
 tara=peso;
 }
 peso=peso-tara;
 // Tratamento dos dados do "peso" para encontrar uma média dos pesos através do array.
 if (peso > 0.1 && numValores < MAX_VALORES) {
  // Se o peso é maior que 0 e o array ainda não estiver cheio.
  ValoresP[numValores] = peso; // Adiciona o valor ao array.
  numValores++; // Incrementa o contador de valores.
 }
 // Calcula a média dos valores no array.
 if(peso<0.1 && numValores>0){
  for (int i = 0; i < numValores; i++) {
   sum += ValoresP[i];
  }
  mediaTotal = sum / numValores;

  // Calcula a média apenas dos valores acima de 80% da média do array, excluindo os
valores de subida e descida da balança.
  for(int j=0 ; j < numValores; j++){
   if(ValoresP[j]>mediaTotal*0.8){
    soma += ValoresP[j];
    contaValores++;
   }
  }
  mediaFinal =  soma / contaValores;
  numValor++;
 }

 // Somatório das pesagem efetuadas.
 somaTotal+=mediaFinal;

 // Atualiza os valores máximo e mínimo.
 if (mediaFinal > PesoMax) {
  PesoMax = mediaFinal;
 }
 if (mediaFinal < PesoMin && mediaFinal > 0.1) {
  PesoMin = mediaFinal;
 }
```

```
 // Contador de pesagens efetuadas.
 if (CZero==true){
   if (peso>0.05){
     contador++;
     CZero = false;
   }
 }
 else if(peso <=0.05){
   CZero = true;
 }

 // Sinal de tendência.
 if(numValor==1 && mediaFinal!=0){
   values[0] = mediaFinal;
 }else if(numValor==2 && mediaFinal!=0){
   values[1] = mediaFinal;
 }else if(numValor==3 && mediaFinal!=0){
   values[2] = mediaFinal;
 }else if(numValor==4 && mediaFinal!=0){
   values[3] = mediaFinal;
 }else if(numValor==5 && mediaFinal!=0){
   values[4] = mediaFinal;
 }

  if(values[4]!=0 && mediaFinal!=0){ //Quando o array está cheio faz o processo de
identificar a tendência.
 diff = abs(values[4] - values[0]);
   for (int i = 0; i < ARRAY_SIZE-1; i++) {
     if (values[i+1] >= values[i]) {
       diminuir = false;
       if(abs(values[i+1]-values[i])<=0.3){
         diminuir = true;
       }
     }
     if (values[i+1] <= values[i]) {
       aumentar = false;
       if(abs(values[i+1]-values[i])<=0.3){
         aumentar = true;
       }
     }
   }
```

```
  }

  if(values[4]!=0 && mediaFinal!=0){ // Quando o array estiver cheio elimina o primeiro valor do array, passando todos os valores para a posição anterior e regista o novo valor.
    memmove(values, values + 1, (ARRAY_SIZE-1) * sizeof(float));
      values[ARRAY_SIZE-1] = mediaFinal;
  }
  if (numValor>=5){
    if (diff >= 0.5 && (aumentar || diminuir)) {
      LEDTEND = 1;
    } else{
      LEDTEND = 0;
    }
  }

  // Envio do estado dos sinais luminosos/sonoro para a base de dados.
  if (mediaFinal>=LimiteSup){
    LEDMAX = 1;
  }else{
    LEDMAX = 0;
  }
  if (mediaFinal<=LimiteInf && mediaFinal > 0.1){
    LEDMIN = 1;
  }else{
    LEDMIN = 0;
  }

  // Envio de dados monitor série para o ESP8266-01 através dos pinos TX e RX.
  if(peso<=0.05 && mediaFinal>0.05){
    String data =  "ç" + String(mediaFinal, 3) + "," + String(PesoMax, 3) + "," + String(PesoMin, 3) + "," + String(contador) + "," + String(LimiteSup) + "," + String(LimiteInf) + "," + String(LEDMAX) + "," + String(LEDMIN) + "," + String(somaTotal, 3) + "," + String(LimiteSupDef) + "," +String(LimiteInfDef) + "," + String(LEDTEND) +"ç";
    Serial1.println(data);
    Serial.println(data);
    delay(500);
    // Reset das variáveis para uma nova pesagem.
    mediaFinal=0;
    mediaTotal=0;
      for(int i = 0; i < numValores; i++) {
```

```
      ValoresP[i] = 0;
    }
  contaValores = 0;
  numValores = 0;
  soma=0;
  sum = 0;
 }

 // Envio de dados para o LCD
  myGLCD.setFont(BigFont); // Define o tamanho da letra para “grande” (BigFont).
  myGLCD.print("Sistema de Pesagem", CENTER, 30); // Exibe a mensagem no centro da tela na posição Y=30.
  myGLCD.print("Contador: ", 0, 80); // Exibe a mensagem à esquerda da tela na posição Y=80.
  myGLCD.printNumI(contador, 150, 80); // Exibe o valor do contador.
  myGLCD.print("Peso: ", 0, 110); // Exibe a mensagem à esquerda da tela na posição Y=110.
  myGLCD.printNumF(peso, 3, 90, 110); // Exibe o valor da pesagem com três casas decimais.
  myGLCD.print("Peso Maximo: ", 0, 140); // Exibe a mensagem à esquerda da tela na posição Y=140.
  myGLCD.printNumF(PesoMax, 3, 200, 140); // Exibe o valor do peso máximo com três casas decimais.
  myGLCD.print("Peso Minimo: ", 0, 170); // Exibe a mensagem à esquerda da tela na posição Y=170.
  myGLCD.printNumF(PesoMin, 3, 200, 170); // Exibe o valor da peso mínimo com três casas decimais.
  myGLCD.print("Producao: ", 0, 200); // Exibe a mensagem à esquerda da tela na posição Y=200.
  myGLCD.printNumF(somaTotal, 3, 150, 200); // Exibe o valor da peso mínimo com três casas decimais.

 // Sinais luminosos/sonoro.
 if (peso>=LimiteSup){
   // Se o peso for maior que o limite superior o LED vermelho irá ligar.
  digitalWrite(12,HIGH);
 }else{
  // Se o peso for menor que o limite superior o LED vermelho irá desligar.
  digitalWrite(12,LOW);
 }
```

```
 if (peso<=LimiteInf & peso >0.1){
  // Se o peso for menor que o limite inferior o LED azul irá ligar.
  digitalWrite(13,HIGH);
 }else{
   // Se o peso for maior que o limite inferior o LED azul irá desligar.
  digitalWrite(13,LOW);
 }

 if (LEDTEND==1){
  // O LED amarelo da tendência irá ligar.
  digitalWrite(11,HIGH);
 }else{
  // O LED amarelo da tendência irá desligar.
  digitalWrite(11,LOW);
 }

 // Definição do novo “LimiteSup” e “LimiteInf” através dos dados enviados da base de dados. Exemplo de mensagem "<xxx,xxx>".
 if (Serial1.available()) {
  // Copia a mensagem que o ESP8266-01 envia para a porta série.
  String inputData = Serial1.readString();
  Serial.println(inputData);
  if(inputData.indexOf("<") != -1){  // Deteta se a mensagem começa por o caracter "<".
    String dataLimites = inputData.substring(inputData.indexOf("<")+1, inputData.indexOf(">"));
    Serial.println(dataLimites);  // Retira os caracteres "<" e ">" da mensagem.
    // Separa as variáveis das vírgulas.
    LimiteSupDef= dataLimites.substring(0,dataLimites.indexOf(",")).toInt();
    LimiteInfDef= dataLimites.substring(dataLimites.indexOf(",")+1).toInt();
    Serial.println(LimiteSupDef);
    Serial.println(LimiteInfDef);
  }
 }
 // Faz a atualização dos limites superior e inferior se estes forem diferentes de 0 e diferentes dos limites superior e inferior já existente.
 if(LimiteSupDef!=0 && LimiteInfDef !=0){
  if(LimiteSup != LimiteSupDef || LimiteInf != LimiteInfDef){
    LimiteSup = LimiteSupDef;
    LimiteInf = LimiteInfDef;
```

```
      myGLCD.clrScr(); // Limpa a tela.
      myGLCD.print("Sistema de Pesagem", CENTER, 30); // Exibe a mensagem no centro da tela.
      myGLCD.print("Limite Superior: ", 0, 130); // Exibe a mensagem à esquerda da tela na posição Y=130.
      myGLCD.printNumI(LimiteSup, 270, 130); // Exibe o valor do novo limite superior.
      myGLCD.print("Limite Inferior: ", 0, 190); // Exibe a mensagem à esquerda da tela na posição Y=190.
      myGLCD.printNumI(LimiteInf, 270, 190); // Exibe o valor do novo limite inferior.
      delay(2000); // Espera 2 segundos.
      myGLCD.clrScr(); // Limpa a tela.
    }
  }
  delay(500); // Tempo para apresentar o próximo valor da pesagem.
}
```

ANEXO 7 – CÓDIGO DE PROGRAMAÇÃO DO ESP8266-01

```
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
// Biblioteca Thread, permite que varias funções sejam executadas simultaneamente.
#include <Thread.h>
#include <ThreadController.h>

WiFiClient client;
const char* ssid = "labelect"; // Nome da internet.
const char* password = "elect2016"; // Password da internet.
// Link para aceder ao ficheiro PHP que irá encaminhar os dados para a base de dados.
const char* serverName = "http://sistemadepesagem2023.000webhostapp.com/post-
data.php";
// Chave para que quando o ficheiro PHP receber dados, só envia se este tiver recebido a
chave no início da mensagem.
String apiKeyValue = "#54321";

float peso, PesoMax, PesoMin, somaTotal, peso_dup;
String DataLimiteDup; // Variável para verificar se os dados que se recebe da base de dados
são iguais aos já recebidos (previne que duplicações sejam enviadas para o Arduíno).

int LEDMAX, LEDMIN, LEDTEND, MAX, MIN, contador=0;
int LimiteSupDef=0, LimiteInfDef=0;

ThreadController controll = ThreadController();
// Cria as threads.
Thread thread1 = Thread();
Thread thread2 = Thread();

void setup() {
  Serial.begin(115200); // Inicia o monitor série.
  delay(5000); // Tempo para passar a primeira imagem do LCD.
  WiFi.begin(ssid, password); // Função que inicia o módulo WiFi.
  Serial.println("Connecting to the Wifi Network");
  // Aguarda até ligar ao WiFi.
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
```

```
    Serial.print(".");
  }
  Serial.println("");
  // Mensagem a confirmar a ligação do módulo WiFi.
  Serial.print("WiFi is Connected at this IP Address : ");
  Serial.println(WiFi.localIP());
  // A função "SendData" vai estar sempre a ser executada.
  thread1.onRun(SendData);
  thread1.setInterval(0);
  // A função "ReceiveData" vai ser executada de 5 em 5 segundos.
  thread2.onRun(ReceiveData);
  thread2.setInterval(5000);
  controll.add(&thread1);
  controll.add(&thread2);
}

void loop() {
  controll.run();
}

void SendData(){
  // Verifica o estado da ligação WiFi.
  if(WiFi.status()== WL_CONNECTED){
    if (Serial.available()) {
      // Copia a mensagem que o Arduíno envia para o monitor série.
      String datafull = Serial.readString();
      // Verifica se a mensagem começa e acaba com o caracter "ç". Por exemplo:
"ç83.895,89.332,77.217,7,80,50,1,0,327.239,80,50,0ç".
      if(datafull.startsWith("ç") || datafull.indexOf("ç")) {
        int idxinicio = datafull.indexOf("ç");
        int idxfim = datafull.indexOf("ç",idxinicio+2);
        String data = datafull.substring(idxinicio+2,idxfim); // Retira o caracter "ç" da
mensagem.
        // Separa os valores das vírgulas e guarda os valores em cada variável.
        int idx1 = data.indexOf(',');
        int idx2 = data.indexOf(',', idx1 + 1);
        int idx3 = data.indexOf(',', idx2 + 1);
        int idx4 = data.indexOf(',', idx3 + 1);
        int idx5 = data.indexOf(',', idx4 + 1);
        int idx6 = data.indexOf(',', idx5 + 1);
        int idx7 = data.indexOf(',', idx6 + 1);
```

```
      int idx8 = data.indexOf(',', idx7 + 1);
      int idx9 = data.indexOf(',', idx8 + 1);
      int idx10 = data.indexOf(',', idx9 + 1);
      int idx11 = data.indexOf(',', idx10 + 1);
      peso = data.substring(0,idx1).toFloat();
      PesoMax = data.substring(idx1 + 1, idx2).toFloat();
      PesoMin = data.substring(idx2 + 1, idx3).toFloat();
      contador = data.substring(idx3 + 1, idx4).toInt();
      MAX = data.substring(idx4 + 1, idx5).toInt();
      MIN = data.substring(idx5 + 1, idx6).toInt();
      LEDMAX = data.substring(idx6 + 1,idx7).toInt();
      LEDMIN = data.substring(idx7 + 1,idx8).toInt();
      somaTotal = data.substring(idx8 + 1,idx9).toFloat();
      LimiteSupDef = data.substring(idx9 + 1,idx10).toInt();
      LimiteSupDef = data.substring(idx10 + 1,idx11).toInt();
      LEDTEND = data.substring(idx11 + 1).toInt();
    }
  }
  // Envia os dados somente se o peso for maior que 0 e estes não tivessem sido enviados anteriormente.
  if(peso>0.00 && peso!=peso_dup){
  // Cria a mensagem com os valores das variáveis.
  String httpRequestData = "api_key=" + apiKeyValue + "&LimiteSupDef=" + String(LimiteSupDef) + "&LimiteInfDef="+ String(LimiteInfDef) + "&Peso=" + String(peso, 3) + "&PesoMax=" + String(PesoMax, 3) + "&PesoMin=" + String(PesoMin, 3)
                        + "&Contagem=" + String(contador) + "&LimiteSup=" + String(MAX) + "&LimiteInf=" + String(MIN) + "&somaTotal=" + String(somaTotal, 3) + "&PesoMaxSinal=" + String(LEDMAX) + "&PesoMinSinal=" + String(LEDMIN) + "&TendenciaSinal=" + String(LEDTEND) + "";
  HTTPClient http;
  http.begin(client,serverName);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  int httpResponseCode = http.POST(httpRequestData); // Envia os dados para a base de dados.

  if (httpResponseCode>0) { // Verifica se os dados foram enviados com sucesso.
    Serial.print("HTTP Response code: ");
    Serial.println(httpResponseCode);
    peso_dup = peso; // Iguala a variável “peso_dup” a “peso”, para que não sejam enviadas duplicações para a base de dados.
  }
```

```
    else {
// Se ocorrer um erro ao enviar os dados para a base de dados, escreve o erro.
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }

    http.end();
  }
  }
  else {
    Serial.println("WiFi Disconnected");
  }
}

void ReceiveData(){

 if (WiFi.status() == WL_CONNECTED) {
  HTTPClient http;
  http.begin(client, serverName);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  int httpResponseCode = http.GET();
    if (httpResponseCode == HTTP_CODE_OK) {
      String DataLimite = http.getString();
      if(DataLimite!=DataLimiteDup){
        delay(200);
        Serial.println(DataLimite);
        DataLimiteDup=DataLimite; // Iguala a variável “DataLimiteDup” a “DataLimite”, para
que não sejam enviadas duplicações para o Arduíno.
        delay(200);
      }
    }
  }
}
```

ANEXO 8 – CÓDIGO DE PROGRAMAÇÃO DO FICHEIRO "*UPDATE_VALUES*"

```
<?php
// Dados necessários para a ligação com a base de dados.
$servername = "localhost";
$dbname = "id20576975_balancainfo";
$username = "id20576975_esp_board";
$password = "Balancainfo2@23";

// Ligação à base de dados .
$com = new mysqli($servername, $username, $password, $dbname);
// Verifica se houve algum erro na ligação.
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
// Definição das variáveis e valores que foram inseridos pelo utilizador.
$LimiteInfDef = $_POST['LimiteInfDef'];
$LimiteSupDef = $_POST['LimiteSupDef'];
// Faz a atualização as variáveis "LimiteInfDef" e "LimiteSupDef" com os novos valores
inseridos pelo utilizador.
$sql = "UPDATE BalancaInfo SET LimiteInfDef = '$LimiteInfDef', LimiteSupDef
='$LimiteSupDef' ORDER BY id DESC LIMIT 1";
mysqli_query($conn, $sql);
// Volta para a interface ("index").
header("Location: index.php");
?>
```

ANEXO 9 – CÓDIGO DE PROGRAMAÇÃO DO FICHEIRO "*POST-DATA*"

```
<?php
// Dados necessários para a ligação com a base de dados.
$servername = "localhost";
$dbname = "id20576975_balancainfo";
$username = "id20576975_esp_board";
$password = "Balancainfo2@23";

$api_key_value = "#54321";
$api_key= $LimiteSupDef = $LimiteInfDef = $Peso = $PesoMax = $PesoMin = $Contagem =
$LimiteSup = $LimiteInf = $PesoMaxSinal = $PesoMinSinal = $TendenciaSinal = "";

// Envio dos dados do Arduíno para a base de dados.
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   // Verifica se a mensagem que recebeu é do tipo "POST".
   $api_key = test_input($_POST["api_key"]);
   if($api_key == $api_key_value) {
      // Compara se a chave que recebeu do Arduíno é igual a "#54321".
      $LimiteSupDef = test_input($_POST["LimiteSupDef"]);
            $LimiteInfDef = test_input($_POST["LimiteInfDef"]);
            $Peso = test_input($_POST["Peso"]);
      $PesoMax = test_input($_POST["PesoMax"]);
      $PesoMin = test_input($_POST["PesoMin"]);
      $Contagem = test_input($_POST["Contagem"]);
      $LimiteSup = test_input($_POST["LimiteSup"]);
            $LimiteInf = test_input($_POST["LimiteInf"]);
            $somaTotal = test_input($_POST["somaTotal"]);
            $PesoMaxSinal = test_input($_POST["PesoMaxSinal"]);
            $PesoMinSinal = test_input($_POST["PesoMinSinal"]);
            $TendenciaSinal = test_input($_POST["TendenciaSinal"]);
      // Ligação à base de dados.
      $conn = new mysqli($servername, $username, $password, $dbname);
            // Verifica se houve algum erro na ligação.
      if ($conn->connect_error) {
         die("Connection failed: " . $conn->connect_error);
      }
```

```
        // Insere os dados recebidos do Arduíno na tabela "BalancaInfo" da base de dados.
        $sql = "INSERT INTO BalancaInfo (LimiteSupDef,LimiteInfDef,Peso, PesoMax, PesoMin, Contagem, LimiteSup, LimiteInf, somaTotal, PesoMaxSinal, PesoMinSinal, TendenciaSinal)
        VALUES ('" . $LimiteSupDef . "', '" . $LimiteInfDef . "', '" . $Peso . "', '" . $PesoMax . "', '" . $PesoMin . "', '" . $Contagem . "', '" . $LimiteSup . "', '" . $LimiteInf . "', '" . $somaTotal . "', '" . $PesoMaxSinal . "', '" . $PesoMinSinal . "', '" . $TendenciaSinal . "')";

        if ($conn->query($sql) === TRUE) { // Verifica se criou uma nova linha com os dados na tabela.
            echo "New record created successfully";
        }
        else {
            echo "Error: " . $sql . "<br>" . $conn->error;
        }
        $conn->close(); // Fecha a ligação com a base de dados.
    }
    else {
        echo "Wrong API Key";
    }
}
else {
    echo "No data posted HTTP POST.";
}

// Envio dos dados da base de dados para o Arduíno.
$conn = new mysqli($servername, $username, $password, $dbname);
// Verifica se houve algum erro na ligação.
if ($conn->connect_error) {
    die("Conexão falhou: " . $conn->connect_error);
}
// Seleciona todos os dados da tabela "BalancaInfo" da base de dados.
$sql = "SELECT * FROM BalancaInfo ORDER BY id DESC LIMIT 1";
$result = "";
$result = $conn->query($sql);
// Verifica se há algum resultado.
if ($result->num_rows > 0) {
    // Converte o resultado em um array associativo.
    $row = $result->fetch_assoc();
        // Cria uma mensagem com os valores dos campos.
        $data = $row["LimiteSupDef"] . chr(44) . $row["LimiteInfDef"];
```

```
        // Imprime a mensagem na tela.
        echo ("<" . $data . ">");
}else {
        echo "0 resultados";
}
// Fecha a ligação com a base de dados.
$conn->close();
// Função para limpar e validar dados.
function test_input($data) {
   $data = trim($data); // Remove espaços em branco do início e do final da mensagem.
   $data = stripslashes($data); // Remove barras invertidas adicionadas a caracteres especiais.
   $data = htmlspecialchars($data); // Converte caracteres especiais em entidades HTML.
   return $data;
}
```

ANEXO 10 – CÓDIGO DE PROGRAMAÇÃO DO FICHEIRO "*INDEX*"

```
<!DOCTYPE html>

<html>
<head>
  <title>Sistema de Pesagem</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.3/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    <style>
        #logo1 {
          width: 100px; /* Define o tamanho da imagem. */
          height: auto;
          float: left; /* Coloca a imagem à esquerda. */
        }

        #logo2 {
          width: 200px; /* Define o tamanho da imagem. */
          height: auto;
          float: right; /* Coloca a imagem à direita. */
        }
        #header {
          text-align: center; /* Coloca a mensagem ao centro. */
          font-size: 30px; /* Define o tamanho de letra. */
          font-weight: bold; /* Define a letra como negrito. */
          padding: 20px 0;
          margin: 0 auto; /* Centraliza horizontalmente */
          margin-top: 20px; /* Adiciona um espaço superior */
        }
    </style>
</head>
<body>
<div class="container-fluid">
<div class="row">
```

```
<div class="col-sm-12">
<img id="logo1" src="https://keystoneacademic-res.cloudinary.com/image/upload/element/93/93574_thumb.jpg" alt="Logo1"> /* Adiciona uma imagem à esquerda e no topo da tela. */
<img id="logo2" src="https://www.estgv.ipv.pt/userfiles/Logotipos%20das%20Licenciaturas/EE.jpg" alt="Logo2"> /* Adiciona uma imagem à direita e no topo da tela. */
<div id="header">SISTEMA DE PESAGEM</div> /* Adiciona uma mensagem no centro da tela. */
</div>
</div>
<div class="row">
<div class="col-sm-10">
<?php
    // Dados necessários para a ligação com a base de dados.
    $servername = "localhost";
    $dbname = "id20576975_balancainfo";
    $username = "id20576975_esp_board";
    $password = "Balancainfo2@23";
    // Ligação à base de dados.
    $conn = new mysqli($servername, $username, $password, $dbname);
    // Verifica se houve algum erro na ligação.
    if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
    }
    // Seleciona todos os dados da tabela "BalancaInfo" da base de dados.
    $sql = "SELECT * FROM BalancaInfo";
    // Cria os títulos de cada coluna da tabela a apresentar no website.
    echo '<table style="text-align:center" class="table">
    <thead>
    <tr>
    <th style="text-align:center">Contagem</th>
    <th style="text-align:center">Peso</th>
    <th style="text-align:center">Peso Máximo</th>
    <th style="text-align:center">Peso Mínimo</th>
    <th style="text-align:center">Soma Total de Pesagens</th>
    <th style="text-align:center">Sinal Peso Máximo</th>
    <th style="text-align:center">Sinal Peso Mínimo</th>
    <th style="text-align:center">Sinal de Tendência</th>
    <th style="text-align:center">Data</th>
```

```
        </tr>
        </thead>';
        // Verifica se conseguiu ligar a base de dados.
        if ($result = $conn->query($sql)) {
                // Copia todos os valores das variáveis da base de dados.
                while ($row = $result->fetch_assoc()) {
                $row_ID = $row["ID"];
                $row_Peso = $row["Peso"];
                $row_PesoMax = $row["PesoMax"];
                $row_PesoMin = $row["PesoMin"];
                $row_Contagem = $row["Contagem"];
                $row_somaTotal = $row["somaTotal"];
                $row_PesoMaxSinal = $row["PesoMaxSinal"];
                $row_PesoMinSinal = $row["PesoMinSinal"];
                $row_TendenciaSinal = $row["TendenciaSinal"];
                $row_Data = $row["Data"];
                // Apresenta todos os valores copiados da base de dados no website.
                echo '<tr>
                <td>' . $row_Contagem . '</td>
                <td>' . $row_Peso . '</td>
                <td>' . $row_PesoMax . '</td>
                <td>' . $row_PesoMin . '</td>
                <td>' . $row_somaTotal . '</td>
                <td><img id="PesoMaxSinal' . $row_ID . '" src="" width="25"
height="25"></td>
                <td><img id="PesoMinSinal' . $row_ID . '" src="" width="25"
height="25"></td>
                <td><img id="TendenciaSinal' . $row_ID . '" src="" width="25"
height="25"></td>
                <td>' . $row_Data . '</td>
                </tr>
                <script>
                var pesoMaxSinal = ' . $row_PesoMaxSinal . ';
                var pesoMinSinal = ' . $row_PesoMinSinal . ';
                var tendenciaSinal = ' . $row_TendenciaSinal . ';

                if (pesoMaxSinal == 1) {
                document.getElementById("PesoMaxSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/ybxlO.jpg";
                } else if(pesoMaxSinal == 0) {
```

```
                document.getElementById("PesoMaxSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/b983w.jpg";
                }
                if (pesoMinSinal == 1) {
                document.getElementById("PesoMinSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/ybxlO.jpg";
                } else if(pesoMinSinal == 0) {
                document.getElementById("PesoMinSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/b983w.jpg";
                }
                if (tendenciaSinal == 1) {
                document.getElementById("TendenciaSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/ybxlO.jpg";
                } else if(tendenciaSinal == 0) {
                document.getElementById("TendenciaSinal' . $row_ID . '").src =
"https://i.stack.imgur.com/b983w.jpg";
                }
                </script>';
                }
        $result->free(); // Liberta a memória alocada da variável “result”.
        }
        $conn->close(); // Fecha a ligação com a base de dados.

// A instrução abaixo é o que possibilita a inserção de valores pelo utilizador para um novo
limite superior e inferior.
?>
</table>
</div>
        <div class="col-sm-2">
                <form action= update_values.php method="post">
                <div class="form-group">
                        <label for="input">Limite Superior</label>
                        <input type="text" class="form-control" name="LimiteSupDef"
placeholder="Valor">
                </div>
                <div class="form-group">
                        <label for="input">Limite Inferior</label>
                        <input type="text" class="form-control" name="LimiteInfDef"
placeholder="Valor">
                </div>
                        <button type="submit" class="btn btn-primary">Enviar</button>
```

```
            </form>
        </div>
  </div>
</div>
</body>
</html>
```

ANEXO 11 – CÓDIGO DE PROGRAMAÇÃO DO FICHEIRO *"DOWNLOAD_TABELA"*

```
<?php
// Dados necessários para a ligação com a base de dados.
$servername = "localhost";
$dbname = "id20576975_balancainfo";
$username = "id20576975_esp_board";
$password = "Balancainfo2@23";

while (true) { // Executa infinitamente.
// Definição do horário de cada turno.
$turno1 = "05:00"; // Turno da noite "05:00".
$turno2 = "13:00"; // Turno da manha "13:00".
$turno3 = "21:00"; // Turno da tarde "21:00".
        // Verifica a hora atual.
        $now = date("H:i");
        if($now == $turno1){ // Se a hora atual for igual à do turno 1.
                // Efetua a ligação à base de dados.
                $conn = new mysqli($servername, $username, $password, $dbname);
                // Verifica se a ligação foi bem sucedida.
                if($conn->connect_error){
                        die("Erro ao conectar ao banco de dados: " . $conn->connect_error);
                }
                // Seleciona os dados da tabela "BalancaInfo" da base de dados.
                $sql = "SELECT Peso, PesoMax, PesoMin, Contagem, somaTotal, PesoMaxSinal,
PesoMinSinal, TendenciaSinal , Data FROM BalancaInfo ORDER BY ID ASC";
                $result = $conn->query($sql);
                // Se a tabela tiver mais que uma linha de dados faz o seguinte código.
                if ($result->num_rows > 0) {
                        // Abre um arquivo CSV (Excel) para guardar os dados.
                        $filenameT1 = 'T1_' . date('Y-m-d') . '.csv';
                        $fileT1 = fopen($filenameT1, 'w');
                        // Define o cabeçalho do ficheiro CSV.
                        $header = array(
                                'Contagem',
                                'Peso',
                                'Peso Maximo',
                                'Peso Minimo',
```

```
                    'Soma Total de Pesagens',
                    'Sinal do Peso Maximo',
                    'Sinal do Peso Minimo',
                    'Sinal de Tendencia',
                    'Data'
                );
                fputcsv($fileT1, $header, ';');
                // Escreve os dados da tabela no ficheiro CSV.
                while ($row = $result->fetch_assoc()) {
                    $data = array(
                        $row['Contagem'],
                        $row['Peso'],
                        $row['PesoMax'],
                        $row['PesoMin'],
                        $row['somaTotal'],
                        $row['PesoMaxSinal'],
                        $row['PesoMinSinal'],
                        $row['TendenciaSinal'],
                        $row['Data']
                    );
                    fputcsv($fileT1, $data, ';');
                }
                // Fecha o ficheiro CSV.
                fclose($fileT1);
                // Elimina os registos da tabela.
  $delete_query = "DELETE FROM BalancaInfo";
  $conn->query($delete_query);
                // Fecha a ligação com a base de dados.
                $conn->close();
        } else {
                echo "Não foram encontrados registos na tabela.";
        }
}

if($now == $turno2){ // Se a hora atual for igual à do turno 2.
        // Efetua a ligação à base de dados.
        $conn = new mysqli($servername, $username, $password, $dbname);
        // Verifica se a ligação foi bem sucedida.
        if($conn->connect_error){
                die("Erro ao conectar ao banco de dados: " . $conn->connect_error);
        }
```

```
        // Seleciona os dados da tabela “BalancaInfo” da base de dados.
        $sql = "SELECT Peso, PesoMax, PesoMin, Contagem, somaTotal, PesoMaxSinal,
PesoMinSinal, TendenciaSinal , Data FROM BalancaInfo ORDER BY ID ASC";
        $result = $conn->query($sql);
        // Se a tabela tiver mais que uma linha de dados faz o seguinte código.
        if ($result->num_rows > 0) {
            // Abre um ficheiro CSV (Excel) para guardar os dados.
            $filenameT2 = 'T2_' . date('Y-m-d') . '.csv';
            $fileT2 = fopen($filenameT2, 'w');
            // Define o cabeçalho do ficheiro CSV.
            $header = array(
                'Contagem',
                'Peso',
                'Peso Maximo',
                'Peso Minimo',
                'Soma Total de Pesagens',
                'Sinal do Peso Maximo',
                'Sinal do Peso Minimo',
                'Sinal de Tendencia',
                'Data'
            );
            fputcsv($fileT2, $header, ';');
            // Escreve os dados da tabela no ficheiro CSV.
            while ($row = $result->fetch_assoc()) {
                $data = array(
                    $row['Contagem'],
                    $row['Peso'],
                    $row['PesoMax'],
                    $row['PesoMin'],
                    $row['somaTotal'],
                    $row['PesoMaxSinal'],
                    $row['PesoMinSinal'],
                    $row['TendenciaSinal'],
                    $row['Data']
                );
                fputcsv($fileT2, $data, ';');
            }
            // Fecha o ficheiro CSV.
            fclose($fileT2);
            // Elimina os registos da tabela.
    $delete_query = "DELETE FROM BalancaInfo";
```

```
        $conn->query($delete_query);
                // Fecha a ligação com a base de dados.
                $conn->close();
            } else {
                echo "Não foram encontrados registos na tabela.";
            }
        }

        if($now == $turno3){ // Se a hora atual for igual à do turno 3.
            // Efetua a ligação à base de dados.
            $conn = new mysqli($servername, $username, $password, $dbname);
            // Verifica se a ligação foi bem sucedida.
            if($conn->connect_error){
                die("Erro ao conectar ao banco de dados: " . $conn->connect_error);
            }
            // Seleciona os dados da tabela “BalancaInfo” da base de dados.
            $sql = "SELECT Peso, PesoMax, PesoMin, Contagem, somaTotal, PesoMaxSinal,
PesoMinSinal, TendenciaSinal , Data FROM BalancaInfo ORDER BY ID ASC";
            $result = $conn->query($sql);
            // Se a tabela tiver mais que uma linha de dados faz o seguinte código.
            if ($result->num_rows > 0) {
                // Abre um ficheiro CSV (Excel) para guardar os dados.
                $filenameT3 = 'T3_' . date('Y-m-d') . '.csv';
                $fileT3 = fopen($filenameT3, 'w');
                // Define o cabeçalho do ficheiro CSV.
                $header = array(
                    'Contagem',
                    'Peso',
                    'Peso Maximo',
                    'Peso Minimo',
                    'Soma Total de Pesagens',
                    'Sinal do Peso Maximo',
                    'Sinal do Peso Minimo',
                    'Sinal de Tendencia',
                    'Data'
                );
                fputcsv($fileT3, $header, ';');
                // Escreve os dados da tabela no ficheiro CSV.
                while ($row = $result->fetch_assoc()) {
                    $data = array(
                        $row['Contagem'],
```

```
                                $row['Peso'],
                                $row['PesoMax'],
                                $row['PesoMin'],
                                $row['somaTotal'],
                                $row['PesoMaxSinal'],
                                $row['PesoMinSinal'],
                                $row['TendenciaSinal'],
                                $row['Data']
                            );
                            fputcsv($fileT3, $data, ';');
                        }
                        // Fecha o ficheiro CSV.
                        fclose($fileT3);
                        // Elimina os registos da tabela.
        $delete_query = "DELETE FROM BalancaInfo";
        $conn->query($delete_query);
                        // Fecha a ligação com a base de dados.
                        $conn->close();
                    } else {
                        echo "Não foram encontrados registos na tabela.";
                    }
                }

                sleep(60); // Aguarda 1 minuto antes de verificar novamente a hora.
}
?>
```