Mestrado em Engenharia de Telecomunicações e Informática

Sistema de Controlo e Monitorização de Semáforos com Limitação de Velocidade

Trabalho Prático

Autores: André Ribeiro,
Alexandre Marques,
Bruno Saraiva,
Luís Candelária

Números: 97793,
97711,
97094,
97758

Professor: Nuno Souto

Junho 2021

Lisboa

Índice

Índice de Ilustrações

Capítulo I - Especificação de Requisitos de Software

1. Introdução

A constante evolução das tecnologias, bem como a progressão do nosso conhecimento, permite-nos realizar trabalhos cada vez mais objetivos. Neste projeto foi-nos solicitado a implementação de um protótipo cujo objetivo é simular um sistema que controla quatro semáforos num cruzamento de quatros vias.

Os semáforos ou também conhecidos por sinais luminosos são objetos/instrumentos a que todos nós habituamos a ver no nosso cotidiano. Com o aumento populacional, estes objetos têm vindo a ter uma importância extrema para a segurança quer dos peões, quer dos condutores, uma vez que estes instrumentos têm como objetivo organizar e sincronizar o tráfego de veículos e peões existente numa determinada zona.

Por outro lado, os semáforos permitem monitorizar a velocidade de um determinado veículo. No projeto desenvolvido implementámos mecanismos que detetam se um veículo está acima do limite de velocidade e, se esse for o caso, o semáforo muda de cor, de forma a abrandar a velocidade do condutor.

1.1 Finalidade

Este documento tem como principal objetivo o facto de especificar os requisitos do sistema de controlo e monitorização de semáforos com limitação de velocidade, fornecendo aos utilizadores todas as informações necessárias sobre o projeto e a sua implementação.

A especificação de requisitos de software assegura a identificação das características do sistema a que estamos a desenvolver, bem como se essas mesmas características correspondem aos objetivos que nos foram propostos. O sistema desenvolvido será posteriormente testado, de modo a verificar se satisfaz, ou não, as características previamente identificadas.

1.2 Materiais utilizados

Para a implementação do protótipo em miniatura do sistema de controlo e monitorização de semáforos com limitação de velocidades serão utilizadas algumas peças de hardware, tais como:

- 1 *Arduino + USB*
- 2 *Xbee+Shield+Xbee Explorer+USB*
- 1 *Breadboard*
- Cabos
- 5 Sensores magnéticos (*Hall*) de presença
- 1 Botão de pressão
- 4 *LEDs RGB + 12 R330*
- 1 *Buzzer* Acústico
- Materiais fornecidos pelos vários elementos do grupo, como por exemplo:
    - Cabos adicionais
    - *Breadboard* adicional

Em termos de Software, optámos por utilizar o *Arduino IDE* para programar todo o código necessário para a implementação do nosso trabalho. A interface gráfica foi programada em *JAVA* com a aplicação *Processing*.

1.3 Descrição do projeto

Foi-nos solicitado a implementação de 4 semáforos "inteligentes" posicionados em ruas distintas que se intercetam num cruzamento, cujo objetivo é assegurar a segurança rodoviária. Para a elaboração deste trabalho tivemos a disponibilização de material para a construção de uma miniatura onde será simulada a situação explicada anteriormente.

O primeiro passo para a garantia de proteção dos condutores é a sincronização precisa de todos os semáforos em funcionamento. Desta forma, apenas um dos semáforos poderá encontrar-se com a luminosidade verde, enquanto os restantes encontrar-se-ão no estado vermelho. A sinalização mudará conforme a situação que está a ocorrer, como será detalhadamente explicada nos próximos parágrafos.

As cores dos semáforos poderão alterar a cor se o utilizador do sistema acionar o botão de pressão que consta na nossa miniatura. Esta situação simula a vontade do peão atravessar, em segurança, a rua onde o sinal está verde para os condutores. Este botão de travessia de peões executa uma interrupção no tráfego com o objetivo de fluir o trânsito de outro lado do cruzamento, ou seja, quando o botão é acionado o sinal altera para amarelo e, só depois para vermelho, de forma que os condutores sejam previamente avisados que haverá a obrigatoriedade de paragem em breves segundos. Tal como foi descrito anteriormente, quando o estado deste sinal alterar, haverá um dos outros três semáforos que também irá alterar a sua cor para verde.

A alteração das cores dos semáforos também pode ser efetuada com base na quantidade de veículos que transitam numa determinada rua, de modo a coordenar da melhor forma possível o tráfego existente no cruzamento. Considerando esta situação, tivemos que incorporar um mecanismo de contagem de veículos que cada rua possui para alterar o estado do sinal daquela que tem o maior congestionamento. O correto funcionamento deste mecanismo só foi possível devido aos sensores que nos foram disponibilizados para utilizarmos na miniatura de simulação.

O estado do semáforo será alterado se o radar detetar que há um veículo com velocidade média superior ao limite pré-estabelecido em cada estrada. Os sensores posicionados, estrategicamente, nas quatro ruas registam os valores necessários para que o mecanismo seja capaz de analisar a legalidade da velocidade que um determinado condutor possui. Posto esta análise, o mecanismo terá de verificar se o sinal encontra-se com luminosidade verde ou vermelha. Se estiver verde e, se o condutor estiver com a velocidade abaixo do limite, então o sinal continua verde. Porém, se o condutor infringir o limite de velocidade e o sinal encontrar-se com a luz verde, então haverá uma alteração para amarelo e depois para vermelho. Obviamente, se o sinal se encontrar com a luminosidade vermelha não haverá qualquer alteração de cor, uma vez que o condutor é obrigado a efetuar uma paragem.

1.4 Acrónimos

- **LED-** *Light-Emmiting Diod*
- **IDE-** *Integrated Development Environment*
- **USB-** *Universal Serial Bus*
- **RGB-** *Red Blue Green*

2 Requisitos

Este capítulo tem o objetivo de detalhar todos os requisitos funcionais (seção 2.1) e não funcionais (seção 2.2) que incluímos no nosso trabalho, de forma individual e elucidativa.

O conjunto de todos os requisitos funcionais resultam na funcionalidade final do projeto. A sua especificação também tem um papel bastante importante, uma vez que só assim é possível documentar detalhadamente todas as funcionalidades que o projeto possui.

Relativamente aos requisitos não-funcionais, tal como o nome indica, trata-se de algo que não seja uma funcionalidade do sistema, mas que seja necessário para que o software atenda o seu propósito. Deste modo, documentámos todos os requisitos não-funcionais de forma que o nosso trabalho atue de forma totalmente correta.

2.1 Requisitos Funcionais

2.1.1 RF001 - Ativação dos sinais

Quando iniciado, o sistema deverá proceder à ativação dos 4 sinais luminosos. Apenas um deverá possuir a luz verde e os restantes deverão encontrar-se com a luz vermelha. Quando o sistema é compilado, o sinal que possui a luz verde é calculado de forma aleatória, para que posteriormente possamos passar à execução correta do sistema e alterar as cores dos sinais consoante a situação que está a ocorrer.

2.1.2 RF002 - Executar a transição de um semáforo de verde para amarelo e posteriormente vermelho

A transição do sinal da cor verde para a cor vermelha será intermediada pela cor amarela, isto é, o sinal passará primeiro pela cor amarela e, só posteriormente é que ficará com a cor vermelha. A sinalização amarela possui uma importância bastante grande, uma vez que avisa os condutores que haverá uma paragem dentro de breves momentos. Este sinal proíbe os condutores a transitarem na zona posterior ao sinal, salvo em casos onde o condutor não possui condições de parar.

2.1.3 RF003 - Gerir a gestão de tráfego com base nos números de carros

O sistema terá a capacidade de gerir o tráfego consoante o movimento que cada rua possui. Para que este requisito funcione corretamente tivemos que utilizar quatro sensores na nossa miniatura física. Assim, é possível coordenar de forma eficaz o trânsito, uma vez que a rua que possui o maior número de carros ficará com a luz verde enquanto as restantes terão a sinalização vermelha.

2.1.4 RF004 - Contabilizar o número de automóveis que passam numa determinada estrada

O sistema deverá conseguir contabilizar o número de automóveis que transitam em cada estrada. Este requisito só é conseguido por meio de sensores que adicionámos nas 4 ruas. Relativamente à parte de *software*, recorremos à utilização de um contador cujo objetivo é

verificar o número de carros que passam por cada sensor. Admitimos que se existir mais do que 4 veículos numa rua, valor este que pode ser alterado, a mesma encontra-se com congestionamento. Caso ocorra um congestionamento numa determinada rua, o sistema irá transitar os sinais das outras ruas que estejam a verdes para amarelo, e depois para vermelho, com o objetivo de estar livre o cruzamento para ser feita a passagem do sinal da rua congestionada para verde.

2.1.5 RF005 - Detetar o acionamento do botão de pressão

O sistema deverá ser capaz de responder corretamente à pressão exercida sobre o botão. Quando este requisito é executado, o sinal luminoso verde, passará para amarelo, e posteriormente para vermelho. É importante referir que procurámos aproximar-nos o máximo da realidade e, por isto, efetuámos um *delay* de 4 segundos na transição entre os sinais verde e amarelo.

2.1.6 RF006 - Conceber um radar de estrada

O sistema deverá dispor de um radar de estrada. Será implementado um mecanismo que verifica numa determinada distância fixa qual a duração em que a mesma será percorrida por um certo veículo. Estes valores serão conseguidos através de sensores que a nossa miniatura possuirá. Após o registo dos valores, será possível efetuarmos o cálculo da velocidade média e compará-la ao limite pré-estabelecido que os veículos poderão transitar naquela zona.

2.1.7 RF007 - Gerir o estado do semáforo consoante a velocidade do condutor

O sistema deverá estar apto a gerir o estado do semáforo consoante a velocidade do condutor. Tal como documentado no *RF006*, cada estrada terá um radar onde será possível analisar a velocidade média que um determinado veículo possua. Caso a velocidade média seja superior ao limite pré-estabelecido de circulação naquela zona, o semáforo que se encontra verde, deverá passar para o estado amarelo e, depois para o estado vermelho. Este requisito tem como principal objetivo assegurar a segurança de todos os condutores que transitam no cruzamento.

2.1.8 RF008 - Buzz acústico emitir som quando for ativado

O sistema deverá emitir um som proveniente do *Buzz* acústico quando um determinado utilizador aciona o botão de pressão. O som ficará audível durante 8 segundos, de forma a auxiliar o peão a atravessar a estrada em segurança.

2.1.9 RF009 - Visualização da potência do sinal

O sistema deve permitir a transmissão da potência de sinal em tempo real para uma plataforma gráfica, onde um operador possa visualizar e averiguar a existência de problemas de comunicação.

2.1.10 RF010 – Consumo energético reduzido

O sistema deve permitir a transição para um estado de energia reduzida durante o seu funcionamento normal.

2.1.11 RF 011 – Segurança da Rede

O sistema deve fazer as comunicações via rádio entre o terminal remoto (*i.e.*, computador ou similar), e um terminal móvel (*i.e.*, *Arduino*) usando algum método de criptografia, aplicacional ou nos módulos de comunicação, que permita uma transmissão de forma segura sem possibilidade de escuta ou alteração por terceiros.

2.2 Requisitos Não-Funcionais

2.2.1 RNF001- Aplicação

As interfaces com o usuário devem seguir os padrões de interfaces estabelecidos pelo docente da unidade curricular de SE.

2.2.2 RNF002- O sistema deverá ter alta disponibilidade

O sistema deverá possuir uma disponibilidade bastante elevada, já que este retrata a simulação de 4 semáforos num cruzamento. Visto que a segurança dos utilizadores está dependente do nosso sistema, o tempo médio entre falhas deverá ser, de igual forma, elevado, de forma a minimizar os danos que poderão ocorrer em caso de paragem abrupta do sistema.

2.2.3 RNF003- O sistema necessitará de uma instalação

A simulação do nosso sistema ficará ainda mais completa com a visualização da interface gráfica. Esta será corrida no programa "*Processing IDE*" que facilitará não só a apresentação do nosso sistema, como também os diversos testes que iremos realizar.

2.2.4 RNF004- O sistema precisará da realização de testes

A exigência do nosso sistema terá de ser alta, sendo que para isto, necessitaremos da realização de vários testes, com o objetivo de verificar se tudo está a decorrer nas melhores condições depois da compilação do código. Estes testes serão realizados maioritariamente através da interface gráfica e do hardware concebido.

2.2.5 RNF005- Será proibido ao utilizador modificar o Software

Será expressamente proibido ao utilizador modificar, adaptar, desmontar ou por qualquer meio decodificar algum componente do nosso sistema. Este é um aspeto fulcral, já que o *software* disponibilizado será entregue com testes de segurança realizados.

3 Prototipagem

A implementação do protótipo do sistema foi o passo inicial para a correta elaboração do trabalho proposto. O principal objetivo deste capítulo é aumentar a qualidade do documento de requisitos, já que para além de garantir um maior envolvimento e consentimento entre todos os elementos do grupo, garante ainda um maior aperfeiçoamento em solucionar quais os requisitos funcionais e não funcionais que teremos de acrescentar ao sistema.

Construímos a prototipagem dos quatro principais componentes do nosso trabalho: a interface gráfica, a integração dos sensores, dos *LEDs* e do *Buzzer/LED* da passagem de peões. O primeiro foi efetuado de forma bastante simples, utilizando o programa *Paint*, enquanto os restantes foram realizados em *Tinkercad*.

3.1 Interface Gráfica

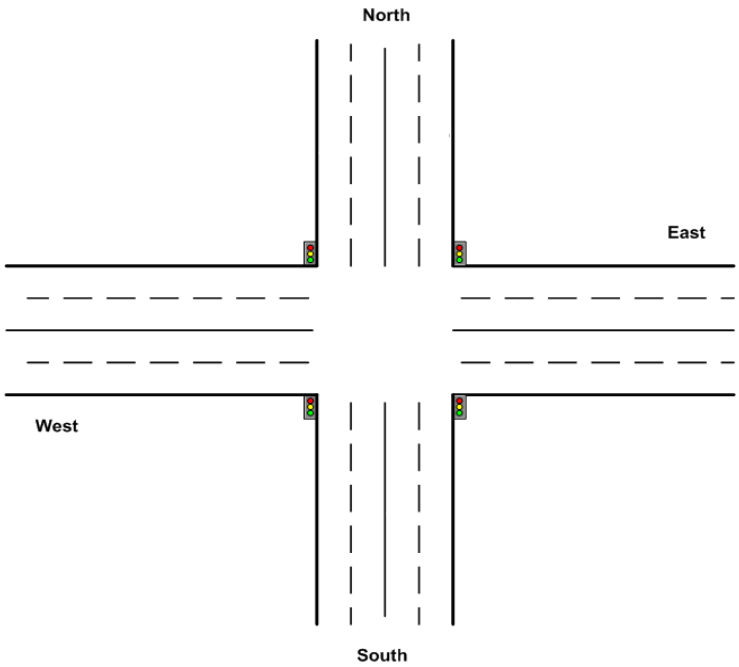

Figura 1 - Esquema Inicial do Cruzamento

Relativamente à prototipagem da interface gráfica, procurámos desenhar um modelo simples, mas realista, de um cruzamento entre quatro estradas com duas vias (*North*, *South*, *East* e *West*). Em cada estrada encontrar-se-à um sinal luminoso, cuja construção provém da aplicação *Processing*, que deverá ser respeitado pelo condutor da respetiva via.

Comparativamente ao resultado final, adicionámos alguns componentes essenciais à interface gráfica do nosso sistema, como por exemplo: o número de carros que transitaram numa determinada via (em tempo real), a potência do sinal recebido do *XBEE*, os respetivos sinais luminosos da passagem

de peões, com indicação de atividade do *Buzzer* e a velocidade média dos carros que circulam na estrada *East*. Temos também uma indicação de quando o botão da passagem de peões é acionado.

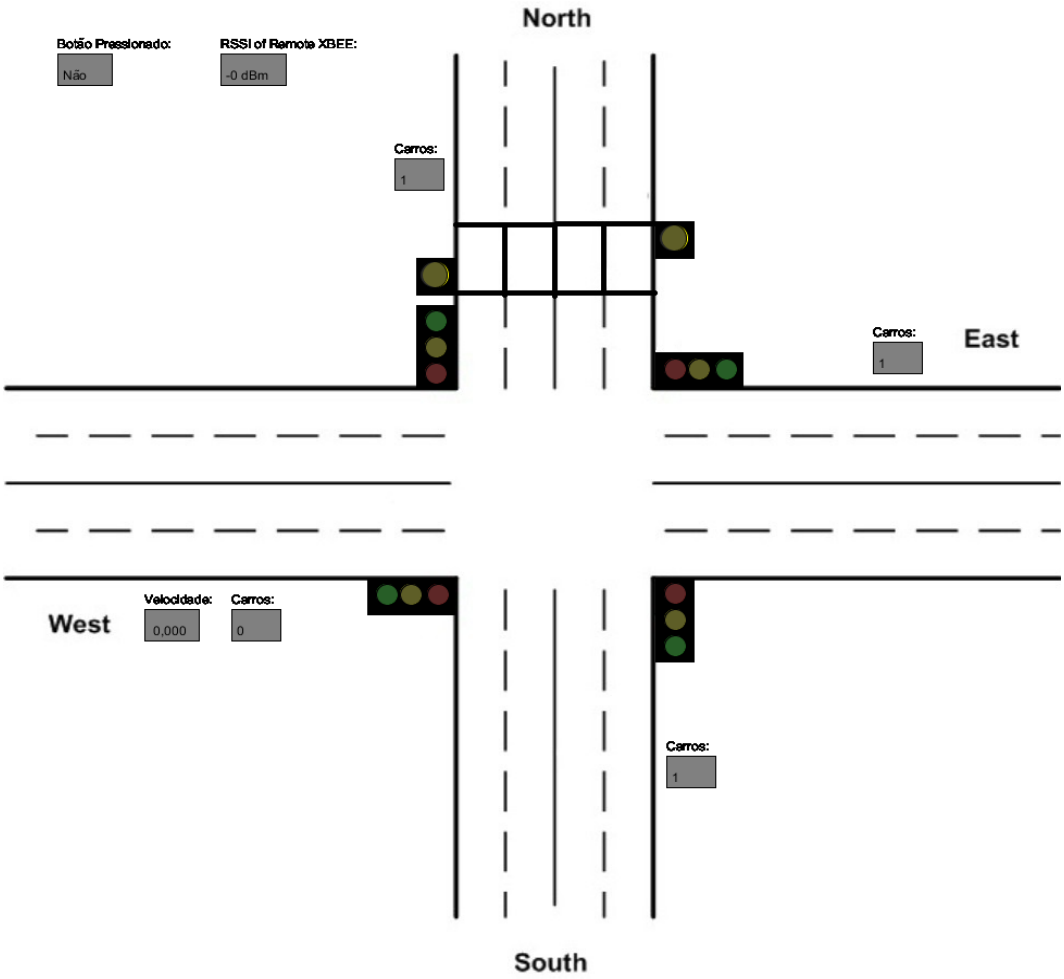

Figura 2 - Esquema Final do Cruzamento

Esta será a forma final do nosso interface gráfico, notando que neste momento os sinais luminosos encontram-se apagados.

3.2 Integração dos Sensores

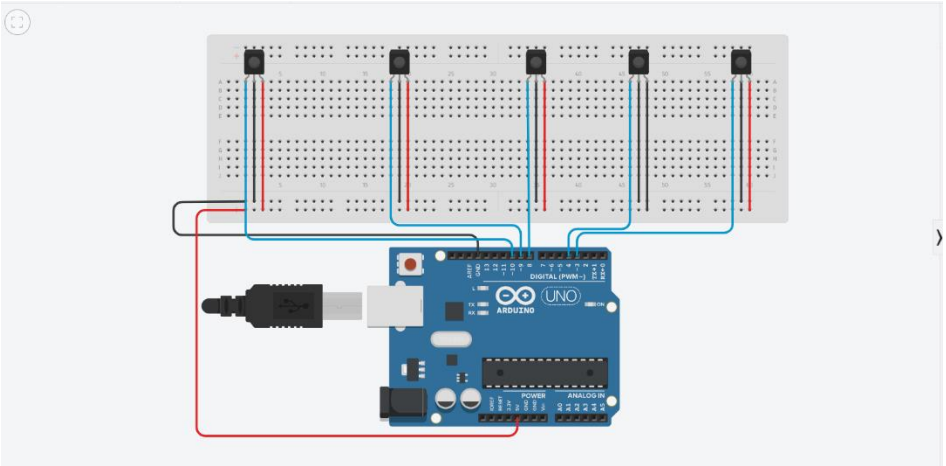

Figura 3 - Esquema Montagem dos Sensores

Utilizamos o programa *Tinkercad* para simular o que iremos implementar em termos de *hardware*. A imagem acima representa o protótipo da integração dos cinco sensores do tipo “hall”.

Esta foi a forma mais organizada que encontramos para o correto funcionamento deste componente: usufruímos de sete portas do A*rduino*, sendo que duas delas aplicam-se à energia do sinal (portas *5V* e *GND*), e cinco á deteção de movimento magnético.

Comparativamente ao resultado final, nada mais foi acrescentado, já que a ideia inicial funcionava sem problemas adicionais.

3.3 Integração das LEDs

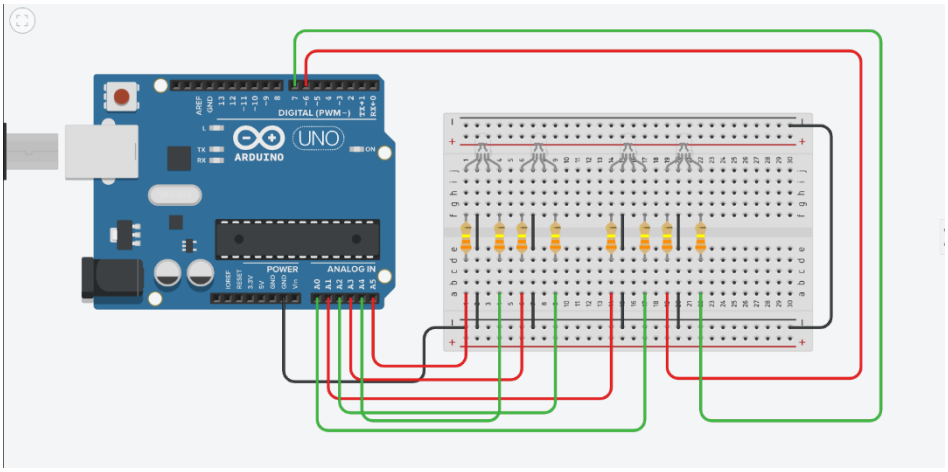

Figura 4 - Esquema Montagem dos LED's

Na integração dos LEDs tivemos que utilizar uma maior quantidade de cabos, bem como oito resistências. Os cabos de cor preta representam o *ground* do sinal, enquanto os restantes aplicam-se à informação dos LEDs. Sendo estes do tipo RGB, os cabos de cor vermelha, obviamente representam o transporte da informação quando os LEDs emitem a cor vermelha e, os cabos verdes representam o transporte da informação quando os LEDs transmitem a cor verde. A cor amarela é conseguida através do transporte de sinais nestes dois tipos de cabos.

3.4 Integração do Buzzer Acústico

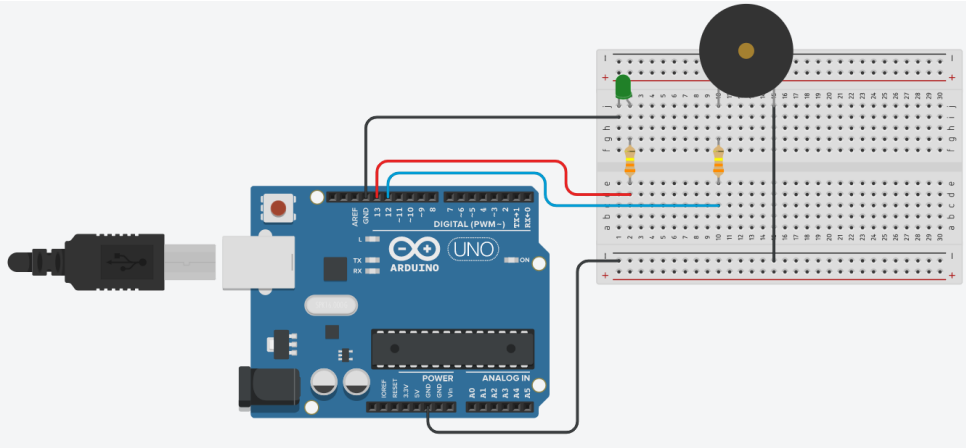

Figura 5 - Esquema Montagem Buzzer

Na implementação do protótipo deste componente utilizamos duas resistências para limitar a energia transmitida para o LED e para o Buzz, de forma a que estes materiais funcionem sem problemas adicionais, com uma vida útil mais elevada.

4 Arquitetura

4.1 Diagrama de Blocos Geral

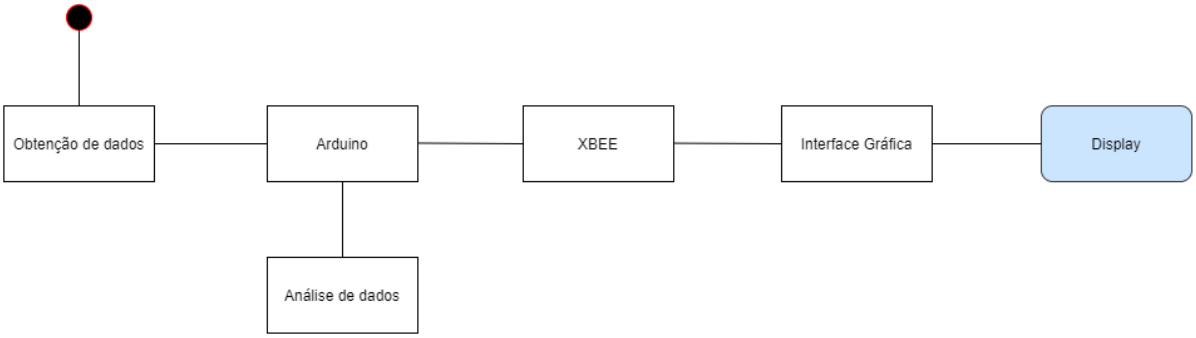

Figura 6 - Diagrama de Blocos Geral

A arquitetura do nosso sistema é apresentada na imagem acima. Os sensores ou botão de pressão existentes no hardware concebido recebem os dados necessários para o funcionamento do trabalho. Posto isto, os dados são enviados para o *Arduino* que, por sua vez, serão analisados de forma a realizar as devidas verificações de velocidade, número de carros existentes em cada via ou ainda, a cor dos sinais luminosos. O *XBEE* possibilita o envio de informação em tempo real dos dados processados que constam no *Arduino* para a interface gráfica via *wireless*. O display é conseguido através da visualização da interface gráfica, isto é, a compilação do nosso sistema.

4.2 Diagrama de Blocos do *Hardware*

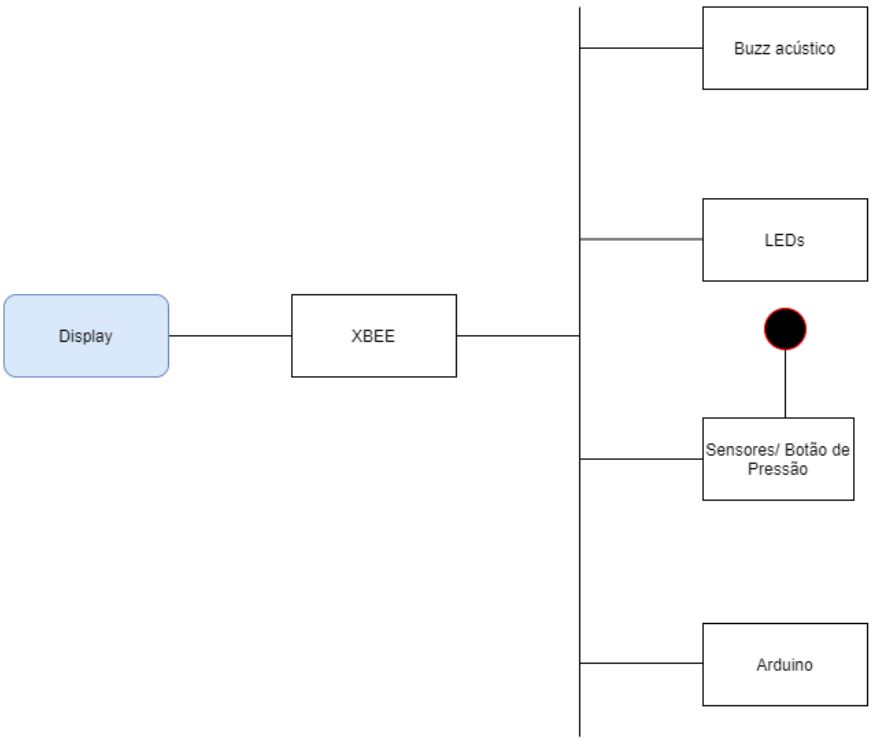

Figura 7 - Diagrama de Blocos de Hardware

O diagrama de blocos de hardware apresentado na imagem acima representa a organização básica do sistema implementado. Aqui é possível observar que o ponto inicial do seu funcionamento é nos sensores ou no botão de pressão, uma vez que estes são os componentes chave para obtenção de dados em tempo real. Os dados são posteriormente analisados pelo software implementado que subsequentemente enviará informação para os restantes componentes de hardware, como por exemplo para as *LEDs* e *Buzzer* acústico.

4.3 Diagrama de Blocos de *Software*

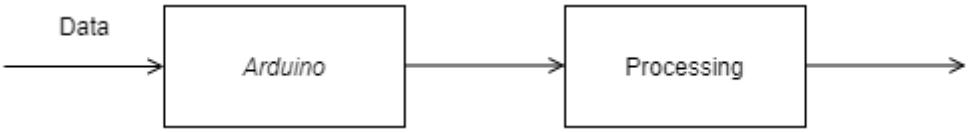

Figura 8 - Diagrama de Blocos de Software

Relativamente ao diagrama de *software* apenas apresentamos estes dois componentes, já que a implementação do sistema é maioritariamente incidida na parte de *hardware*/ integração de módulos. Contudo, o software do nosso sistema é composto por duas partes: *Arduino* e *Processing*, sendo que o primeiro é utilizado para análise de dados e implementação de funções que compõem o objetivo do trabalho e, o segundo é utilizado para a interação entre sistema-utilizador.

O *Arduino* recebe todos dados provenientes do modelo *hardware*, como por exemplo: dados dos sensores e do botão de pressão. Numa fase posterior este componente envia todos os dados processados necessários para que o *Processing* obtenha a maior funcionalidade possível quando interage com o utilizador.

5 Integração do sistema

A integração do sistema foi uma das partes mais importantes do trabalho, pois optámos por implementar os vários componentes em separado e, numa fase posterior, interligá-los de forma correta. É também importante referir que cada componente foi sujeito a vários testes, para termos a confiança de que nenhum problema adicional acontecesse na fase seguinte.

Começámos por implementar os protótipos, apresentados anteriormente no documento, para facilitar a construção do *hardware*. Depois de tudo testado, iniciámos a implementação do software necessário para o sistema. Nesta fase tivemos que interligar o *hardware* com o *software* através do *USB*, pois ainda não tínhamos o módulo *XBEE* ativo. A última parte a ser integrada foi a interface gráfica, que recebe valores do *Arduino* para o *Processing*. A integração desta última fase foi conseguida através da correta implementação do *XBEE*. A interface gráfica foi implementada por módulos idênticos aos que foram demonstrados pelo docente nas aulas da unidade curricular. Na fase final, bastou-nos interligar todos os módulos e obtivemos uma interface com funcionamento correto.

5.1 Comandos Recebidos no Processing

Para passarmos informação do que está a acontecer no *Arduino* em tempo real para o *Processing*, decidimos optar por uma estratégia em que cada ação no *Arduino* corresponde a um comando no *Processing*. Assim, sempre que é necessário enviar uma ação ou dado do *Arduino*, este é enviado na forma de um código base X, com um dado valor Y, na forma de YX.

Os comandos são transmitidos através do *XBee* no *Arduino*, de forma encriptada e recebidos no terminal final, onde se encontra o utilizador que irá supervisionar o sistema. Assim, apresentamos a seguinte tabela onde, de forma resumida, indicamos o comando a ser enviado e a ação a que esse corresponde:

Comando	**Descrição**
YX	**Dado Y - Comando Base X**
110	Estrada *North* com sinal vermelho
210	Estrada *North* com sinal amarelo
310	Estrada *North* com sinal verde
120	Estrada *South* com sinal vermelho
220	Estrada *South* com sinal amarelo
220	Estrada *South* com sinal verde
130	Estrada *East* com sinal vermelho
230	Estrada *East* com sinal amarelo
330	Estrada *East* com sinal verde
140	Estrada *West* com sinal vermelho

240	Estrada *West* com sinal amarelo
340	Estrada *West* com sinal verde
150	Sinal de Passagem de Peões Ativo
250	Sinal de Passagem de Peões Inativo
160	Contador da passagem de carros na estrada *North*
260	Contador da passagem de carros na estrada *South*
360	Contador da passagem de carros na estrada *East*
460	Contador da passagem de carros na estrada *West*
170	Peão pressionou o botão
270	Peão não pressionou o botão
X80	Velocidade média do veículo, em que X é o valor transmitido do *Arduino*

Capítulo II - Sumário do Projeto

1. Elaboração do Projeto

De uma forma geral, começamos por planear o projeto através da recolha de requisitos. Nesta fase contactámos com o docente da unidade curricular várias vezes durante as aulas para termos uma ideia geral do tema proposto.

Elaborámos vários esquemas para termos uma noção do que teríamos de construir e implementar numa fase futura. Após várias reuniões de grupo e, depois de concebido os vários protótipos de cada módulo, iniciámos a fase de execução do projeto. Cada componente do sistema foi implementado de forma separada e, depois de vários testes, iniciamos a fase de integração. Esta fase foi fulcral para o correto funcionamento do sistema, já que existiam certos módulos em *hardware* e outros em *software*.

Ao longo das várias fases ocorreram alterações na montagem dos componentes, de modo a simplificar o esquema final. Por fim, após estes passos todos restava-nos implementar a interface gráfica, que apenas foi conseguida através da interligação dos referidos módulos.

Em termos do que foi atingido ao nível dos requisitos funcionais mencionados, podemos afirmar que foram todos cumpridos com a exceção do requisito funcional RF010 - Consumo energético reduzido. Assim todos os objetivos principais e um dos secundários, estão implementados com sucesso.

Relativamente à colaboração de cada elemento do grupo, apresentamos a seguinte tabela:

Nome	Contribuição
Alexandre Marques	40%
André Ribeiro	20%
Bruno Saraiva	20%
Luís Candelária	20%
Total	100%

2. Implementação do Projeto

Em termos de implementação, o projeto foi desenvolvido de raiz pelos elementos do grupo, sendo que aproveitámos algum material disponível com características semelhantes em várias fontes.

Desses, e a título de exemplo, temos a definição dos sinais em *Arrays*, o uso de *Interrupts* para o acionamento de certos eventos no projeto, e o uso das funções de comunicação sem fios através dos módulos *XBee*, disponibilizadas pelo docente desta Unidade Curricular nas aulas práticas.

Este último ponto foi bastante relevante a nós, e assim, adaptada ao nosso projeto, sendo que foi idealizado uma solução com o uso de comandos dependendo das ações a serem realizadas em tempo real no *Arduino*. O facto de puder enviar um número simples que representa uma instrução foi bastante bem recebido pelos elementos do grupo como uma solução eficaz e simples. Alguns elementos gráficos foram também baseados nos materiais das aulas práticas.