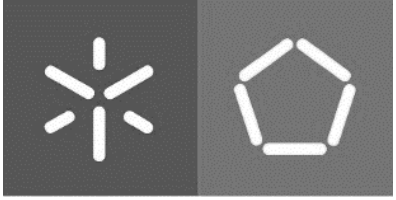

Universidade do Minho
Escola de Engenharia

Filipe Joel da Costa Rodrigues

Controlo de Esforço no Ciclismo com Recurso a Bicicletas Elétricas e Smartphones

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de Mestre em Engenharia de Comunicações

Trabalho efetuado sob a orientação do
Professor Doutor José Augusto Afonso

outubro de 2014

DECLARAÇÃO

Nome: Filipe Joel da Costa Rodrigues

Endereço eletrónico: fjoelcr@gmail.com

Telemóvel: 939899436

Número do Bilhete de Identidade: 12985455

Título dissertação: Controlo de Esforço no Ciclismo com recurso a Bicicletas Elétricas e Smartphones

Orientador: Professor Doutor José Augusto Afonso

Ano de conclusão: 2014

Designação do Mestrado: Ciclo de Estudos Integrados conducentes ao Grau de Mestre em Engenharia de Comunicações

Área de especialização: Engenharia de Comunicações

Escola: Escola de Engenharia da Universidade do Minho

Universidade do Minho, ___/___/_____

Assinatura: _____

Agradecimentos

Passados cinco anos após ingressar na Universidade do Minho, completo a minha etapa académica com a certeza que evoluí as minhas competências profissionais e pessoais. Esta evolução foi progressiva e estruturada, graças, não só, a um enorme empenho em fazer mais e melhor, mas sobretudo pela entreajuda e camaradagem partilhada por todos aqueles que tive o prazer de conhecer.

Quero agradecer ao Professor Doutor José Afonso, por toda a sua ajuda e disponibilidade incansável em orientar todos os trabalhos realizados na dissertação.

Ao Delfim Pedrosa, do grupo de eletrónica de potência, pela ajuda em questões relacionadas com a bicicleta elétrica.

Ao meu pai Manuel e à minha mãe Maria da Conceição, agradeço-vos toda a ajuda, incentivo e coragem que sempre me transmitiram durante todo o percurso académico. Sem vocês isto não seria possível.

Expressar o meu profundo e sentido agradecimento à minha namorada Sandra, pela paciência ímpar que teve nos momentos em que estive mais atarefado e pela força transmitida em ultrapassar todas as barreiras ao longo destes anos. Sem ti, e sem o teu apoio, seria tudo muito mais complicado.

À minha irmã Cristiana e ao meu cunhado Rodrigo, agradeço-vos pela admiração e disponibilidade demonstrada ao longo de todo o meu percurso académico.

Ao meu sobrinho e afilhado Filipe pelo orgulho e inspiração natural que me transmite todos os dias.

Agradecer e expressar a minha gratidão a todos os meus amigos que tive o prazer de conhecer no seio académico, com especial destaque ao António Maio, ao Diogo Matos, ao João Vilela e ao Paulo Alves, companheiros de estudo e camaradas para a vida.

Resumo

O principal objetivo desta dissertação visa o estudo, desenvolvimento e teste de um sistema de monitorização e controlo da resistência e do esforço no ciclismo com recurso a bicicletas elétricas e smartphones, em tempo real, com base em tecnologia sem fios e direcionado para transporte, desporto e lazer.

Através desta monitorização e controlo de esforço torna-se possível disponibilizar as vantagens de controlo do esforço, que é característico de bicicletas estáticas, com o benefício da atividade física ao ar livre fornecido pelas bicicletas convencionais.

Para a concretização deste sistema, foram instalados diversos módulos de *hardware* na bicicleta elétrica, integrados no sistema central de processamento da bicicleta, entre os quais sensores de torque e cadência e um módulo para comunicação sem fios Bluetooth. Utilizou-se ainda um smartphone Android para comunicação com a bicicleta elétrica. Conjuntamente, foram também desenvolvidos vários módulos de *software* para o sistema central de processamento da bicicleta, para o módulo Bluetooth e também para o smartphone. A aplicação desenvolvida para o smartphone implementa um algoritmo de controlo de esforço capaz de receber e interpretar os valores dos sensores, relativos ao desempenho físico do ciclista, e decidir qual o melhor nível de ajuda do motor, em tempo real, com base em perfis pré-definidos com diferentes tolerâncias de esforço admitidas pelo ciclista.

Para que fosse possível verificar o correto comportamento de todo o sistema, foram realizados diversos testes, assentes em diferentes cenários. Os resultados obtidos indicam que o sistema desenvolvido permite reduzir as variações no esforço efetuado pelo ciclista ao longo do tempo seja por influência da cadência ou do torque, desde que a cadência tenha uma boa resolução relativamente ao comportamento que o ciclista está a realizar sobre a pedaleira e salvaguardando sempre que o período de amostragem dos valores dos sensores seja superior ao

tempo necessário para aquisição dos valores dos sensores, processamento de dados e transmissão das tramas por parte da bicicleta elétrica para o smartphone, via Bluetooth, somado ao tempo para que este interprete os valores recebidos, processe qual o melhor nível de ajuda a aplicar ao motor elétrico e transmita a resposta para a bicicleta.

Abstract

The main goal of this thesis is to study, develop and test a system for monitoring and control of both the resistance and effort on cycling, in real-time, using electric bikes, smartphones and wireless technologies, for application in the areas of transportation, sports and leisure.

Through this effort monitoring and control system, it is possible to combine the advantages of effort control which are characteristic of static bikes with the benefits of outdoor physical activity provided by conventional bikes.

To implement this system, several hardware modules were installed in the electrical bike, integrated in the central processing system, including torque and cadence sensors and a Bluetooth wireless communication module. An Android smartphone was also used to communicate with the electrical bike. Several software components were also developed, for the bike's central processing system, the Bluetooth module and the smartphone. The application developed for the smartphone implements an effort control algorithm which is able to receive and interpret the sensors values, regarding the cyclist's performance, and decide which is the most adequate level of motor assistance, in real-time, based on pre-defined profiles with different effort tolerances allowed by the cyclist.

In order to verify the correct behavior of the whole system, several tests were carried, based on different scenarios. The results show that the system allows to reduce the variations of the effort exerted by the cyclist effort over the time, either by influence of the cadence or the torque, as long as the cadence has a good resolution and that the sensors sampling period is higher than the time needed to acquire the sensors values, process them and transmit the data from the electric bike to the smartphone, through Bluetooth, added to the time needed for the

smartphone to process the values received, in order to determine the best level of assistance to apply to the electrical motor, and to transmit the response to the bike.

Índice de conteúdos

Lista de figuras

Lista de tabelas

Lista de abreviaturas

8DPSK	8-ary Differential Phase Shift Keying
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AP	Access Point
BAN	Body Area Network
BLDC	Brushless DC
BLE	Bluetooth Low Energy
BPM	Batimentos Por Minuto
CCDF	Complementary Cumulative Distribution Function
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
ECG	Electrocardiography
EDR	Enhanced Data Rate
FFD	Full Function Device
FHSS	Frequency Hopping Spread Spectrum
GFSK	Gaussian Frequency Shift Keying
GPRS	General Packet Radio Service
GPS	Global Positioning System
GPU	Graphics Processing Unit
I/O	Input/Output

IEEE	Institute of Electrical and Electronics Engineers
IrDA	Infrared Data Association
ISM	Industrial Scientific and Medical
ISTAT	National Institute of Statistics
KERS	Kinetic Energy Recovery System
LOS	Line Of Sight
MAC	Medium Access Control
Nm	Newton metro
Pedelec	Pedal Electric Cycle
PHY	Physical Layer
PRESTO	Promoting Cycling for Everyone as a Daily Transport Mode
PSK	Phase Shift Keying
RAM	Random Access Memory
RFCOMM	Radio Frequency Communications
RFD	Reduce Function Device
RPM	Rotações Por Minuto
SCI	Serial Communication Interface
SGBD	Sistema de Gestão de Base de Dados
SIG	Bluetooth Special Interest Group
SPI	Serial Peripheral Interface
SpO2	Saturation of Peripheral Oxygen
SPP	Serial Port Profile
SQL	Structured Query Language
SSL	Secure Sockets Layer

UMTS	Universal Mobile Telecommunication System
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

1. Introdução

1.1 Enquadramento e motivação

A integração de sensores e atuadores nos mais diversos equipamentos utilizados no quotidiano humano - por forma a melhorar a sua qualidade de vida - é uma realidade bem vincada. A esta realidade podemos juntar-lhe, como forte "aliado", as redes de área pessoal sem fios (WPAN – Wireless Personal Area Network), dado permitirem que informação possa ser disseminada numa tecnologia de rádio de curto alcance, com um baixo consumo de energia e com um custo reduzido.

O constante desenvolvimento na miniaturização e posterior integração de sensores em produtos populares (e.g. smartphones) potenciou que várias aplicações pudessem ser pensadas e desenvolvidas num âmbito inimaginável até então, criando um novo conceito denominado de sensorização móvel, que consiste na recolha de dados sensoriais centrada nas pessoas [1] através dos seus smartphones.

No caso particular do desenvolvimento no âmbito das bicicletas elétricas e na proliferação de smartphones, é proporcionada uma versatilidade de aplicações com grande benefício humano. Neste sentido, as bicicletas elétricas são adotadas cada vez mais como um meio sustentável de locomoção, desporto e lazer [1]. A sua integração com dispositivos móveis (smartphones) permite um controlo de esforço (em tempo real) característico das bicicletas estáticas, com o benefício para a atividade física ao ar livre proporcionada pelas bicicletas convencionais [2], de uma forma igualmente automática.

Inevitavelmente, o sistema a desenvolver no âmbito desta dissertação trará outros benefícios, como exemplo: proporcionar uma maior facilidade de deslocação em zonas com subidas e descidas acentuadas (locais tipicamente com fraca adesão a este tipo de meio de transporte) sem que o utilizador tenha de se preocupar em ajustar manualmente o nível de ajuda do motor. Para esta automação e consequente controlo de esforço, implementou-se um algoritmo no smartphone que

controla o nível de ajuda do motor elétrico com base em perfis pré-definidos pelo utilizador. Nestes perfis, os valores limites (*thresholds*) de controlo sobre o ciclista podem ser direcionado sob diferentes ações, para delimitar os valores da potência mecânica (esforço em Watts) que este admite suportar ou, por outro lado, controlar a resistência máxima (torque) que este está disposto a tolerar. Assim, os limites impostos funcionam como referência para alteração do nível de ajuda do motor, onde a aquisição das grandezas físicas é feita com base num sensor instalado na bicicleta, responsável por traduzir em sinais elétricos os valores da cadência e do torque. Este sensor, denominado X-Cell RT digital [3], foi colocado estrategicamente no rolamento interno da pedaleira, dado que lhe confere a vantagem de ficar invisível, garantindo uma perfeita proteção contra fatores externos ao sistema como o óleo, água, poeira e vibrações do piso, dispensando assim grandes necessidades de manutenção. Adicionalmente, o facto de estar no rolamento interno da pedaleira atribui-lhe uma global compatibilidade com os modelos de bicicletas existentes e evita a necessidade de calibração de cada vez que a bicicleta é desmontada para transporte.

A informação fornecida pelo sensor é processada em tempo real pelo módulo LaunchXL-F28027, da Texas Instruments, abordado na secção 3.2.1. Ao LaunchXL-F28027 foi integrado o módulo Bluetooth WRL-12580, para que fosse possível a sua comunicação com o smartphone recorrendo à tecnologia de comunicação sem fios Bluetooth. Por sua vez, o smartphone disponibiliza ao ciclista informação que passa pelo esforço máximo e mínimo a que o ciclista está sujeito, o nível de bateria da bicicleta elétrica e o torque, juntamente com a cadência, que o ciclista está a exercer sobre a pedaleira.

Dado que se trata de um sistema em tempo real, é de extrema importância garantir que este seja capaz de responder coerentemente e em tempo útil às exigências, sob o ponto de visto do esforço, de cada tipo de percurso, por forma a garantir a qualidade de serviço. Para isto é necessário garantir que o esforço seja controlado quando houver alterações em qualquer um dos fatores constituintes do esforço (cadência e torque) e que o atraso no envio e receção de informação entre a

bicicleta elétrica e o smartphone seja menor que o tempo necessário entre a recolha de cada amostra de esforço, ou seja, inferior ao período de amostragem.

1.2 Objetivos

Pretende-se com esta dissertação desenvolver um sistema capaz de monitorizar e controlar o nível de esforço de um ciclista numa bicicleta elétrica através de um smartphone e de um módulo implantado na própria bicicleta, que servirá para interpretar as instruções do smartphone para a bicicleta elétrica (alteração do nível de ajuda do motor), enviar a aquisição dos valores dos sensores da bicicleta elétrica para o smartphone e desta forma proporcionar um esforço constante e sem interferência por parte do ciclista. Para isto, o sistema recorrerá a sensores e atuadores que, através de um microcontrolador e um módulo Bluetooth, comunicam com o smartphone. Desta forma, os objetivos principais para a concretização desta dissertação são:

- Descrever e desenvolver o módulo responsável pela interação entre o smartphone e a bicicleta;
- Estabelecer o protocolo de comunicação entre o smartphone e a bicicleta;
- Integrar e realizar testes com os sensores para medição da resistência e potência mecânica (esforço) exercida pelo ciclista;
- Desenvolver uma aplicação eficiente e respetivo algoritmo de controlo de esforço no smartphone Android;
- Realização de testes de desempenho para garantir o correto funcionamento do sistema.

1.3 Estrutura da dissertação

O capítulo 1 apresenta uma introdução inicial ao tema da dissertação. São apresentadas as principais motivações para o desenvolvimento do trabalho e

respetivos objetivos a atingir. Por fim apresenta-se a estrutura escolhida para a dissertação.

No capítulo 2 é apresentado o estado da arte no que diz respeito às redes de área pessoal sem fios com base em tecnologia Bluetooth. É abordado também o modo de operação de dois tipos de sensores responsáveis por mensurar dados fisiológicos, bem como o estado de desenvolvimento atual das bicicletas elétricas existentes. Este capítulo termina com uma análise aos dispositivos móveis, focando o sistema operativo móvel Android, seguida da descrição de outros projetos que possuem características semelhantes ao trabalho desenvolvido nesta dissertação.

O capítulo 3 apresenta um estudo introdutório sobre as várias componentes do sistema implementado, que aborda a componente referente ao *hardware* utilizado na dissertação, incluindo o microcontrolador, o tipo de dispositivo utilizado para comunicação sem fios e o sensor escolhido para recolha dos valores de torque e cadência exercidos pelo ciclista.

No capítulo 4 é descrita a implementação de todo o sistema de uma forma pormenorizada. É explicado o processo de aquisição e interpretação dos dados sensoriais e a forma como estes são enviados da bicicleta para o smartphone, tendo em consideração o protocolo de comunicação implementado. Explica-se ainda a estratégia escolhida para implementação do algoritmo de controlo, tendo em consideração o atraso de comunicação necessário entre a bicicleta e o smartphone.

No capítulo 5 são apresentados e discutidos os resultados práticos no que diz respeito ao controlo da resistência (torque) e ao controlo de esforço com base nos valores da cadência e do torque. Comparam-se os resultados de testes realizados nas mesmas condições, onde nalguns casos a ajuda do motor está desabilitada e noutros casos está habilitada. Termina-se com a verificação do tempo mínimo que terá de ser estabelecido para o intervalo de envio de tramas entre a bicicleta e o smartphone.

Por fim, o capítulo 6 apresenta as conclusões obtidas com esta dissertação, em que é possível verificar quais os objetivos que foram cumpridos. São apresentadas

algumas propostas de modificações a ter em consideração, com realce para possibilidades de alterações em trabalho futuro.

2. Estado da arte

Ao longo deste capítulo são abordados os principais temas relacionados com o trabalho realizado na dissertação. Inicialmente é efetuado um estudo acerca das redes de área pessoal sem fios, com especial destaque para a tecnologia de comunicação sem fio Bluetooth. Segue-se uma abordagem no âmbito de sensores relacionados com o setor da saúde cuja análise passa por um sensor cardíaco e um sensor de torque e cadência. Faz-se também uma breve análise sobre bicicletas elétricas e, mais concretamente, uma análise mais detalhada da bicicleta utilizada nos trabalhos da dissertação no que diz respeito às suas especificações. Conjuntamente com uma introdução aos dispositivos móveis, é também analisada a cota de mercado dos principais sistemas operativos móveis, particularizando-se a arquitetura protocolar do Android e suas especificidades técnicas. Finaliza-se este capítulo com um estudo acerca de trabalhos desenvolvidos por terceiros que tenham direta relação com o que é implementado nesta dissertação.

2.1 Redes de área pessoal sem fios

As redes de área pessoal sem fios (WPAN – Wireless Personal Area Network) são hoje em dia um recurso enraizado no quotidiano de cada um de nós. De uma forma consciente ou inconsciente deparamo-nos com tarefas comuns do dia-a-dia que apenas são possíveis de realizar com recurso a este tipo de redes, onde cada vez mais temos objetos comuns que integram este tipo de tecnologia. Como exemplo temos relógios que estão sincronizados com telemóveis, *headphones* sem fios, máquina fotográficas que descarregam os dados sem qualquer tipo de conexão física ao computador, etc. Com esta imensa evolução, deduz-se que os equipamentos de uso pessoal que antes comunicavam com base em tecnologias suportadas por comunicações elétricas cabladas tornam-se hoje mais práticos e apelativos para o

uso comum recorrendo a tecnologia de comunicação sem fios. Assim, foi necessário proceder à criação de *standards* que permitissem a interoperabilidade e coexistência entre os vários tipos de equipamentos que recorrem a este tipo de comunicação.

2.1.1 Bluetooth

O Bluetooth Special Interest Group (Bluetooth SIG), fundado em Setembro de 1998 por empresas lideres em telefonia e computação como a Ericsson, IBM, Intel, Nokia e Toshiba, é o responsável pela especificação do *standard* Bluetooth. Posteriormente, o IEEE (Institute of Electrical and Electronics Engineers), responsável pelo grupo de trabalho WLAN (Wireless Local Area Network) IEEE 802.11, formou um novo grupo de trabalho para as WPAN, dado que o *standard* IEEE 802.11 já existente não era considerado apropriado para as WPANs, pois estas últimas possuem requisitos diferentes relativamente ao raio de abrangência, custo e aplicação relativamente ao IEEE 802.11. Assim surgiu o grupo de trabalho IEEE 802.15, que daria resposta a este tipo de redes de área pessoal sem fios.

A cooperação no desenvolvimento do Bluetooth entre o Bluetooth SIG e o IEEE atingiu o seu máximo quando em 2002 o IEEE lançou o *standard* IEEE 802.15.1 WPAN/Bluetooth baseado na especificação Bluetooth v1.1 do Bluetooth SIG. Posteriormente a esta colaboração inicial conjunta entre o Bluetooth SIG e o IEEE foram tomados rumos estratégicos diferentes entre estas duas entidades. No caso do Bluetooth SIG, lançou o Bluetooth v2.0 em 2004, que garantia a compatibilidade com a versão anterior, e introduziu o mecanismo EDR (Enhanced Data Rate) que proporcionou o aumento da taxa de transmissão de dados no *payload* de 1 Mbit/s para 3 Mbit/s. Em 2009 o grande melhoramento consistiu no aumento das velocidades de transmissão com o Bluetooth v3.0, que passou dos 3 Mbits/s, suportados pela versão 2.0, para os 24 Mbits/s alcançados pela versão 3.0, devido à substituição da ligação Bluetooth por uma ligação IEEE 802.11 durante a transmissão de dados. Com taxas de transmissão bastante satisfatórias, o desenvolvimento seguinte assentou na redução do consumo de energia por parte dos equipamentos,

e assim, em 2010, surgiu o Bluetooth v4.0 que inclui o Bluetooth v3.0 juntamente com o novo protocolo BLE (Bluetooth Low Energy).

Os fabricantes de dispositivos telefónicos móveis perceberam desde cedo que o Bluetooth seria uma tecnologia com um potencial bastante elevado no que diz respeito à troca de informação entre dispositivos [4]. A Ericsson foi a primeira empresa de telecomunicações a incorporar Bluetooth num telemóvel, denominado Ericsson T36 e lançado para o mercado no ano 2000 [4]. Desde então, o crescimento de incorporação de interface Bluetooth nos telemóveis cresceu exponencialmente.

2.1.1.1 Especificação

Um dos requisitos do Bluetooth é a garantia de interoperabilidade entre componentes Bluetooth de diferentes fabricantes. Na Figura 2.1 podemos visualizar a pilha protocolar do Bluetooth que todos os dispositivos Bluetooth desenvolvidos devem respeitar de forma que seja possível um correto funcionamento global.

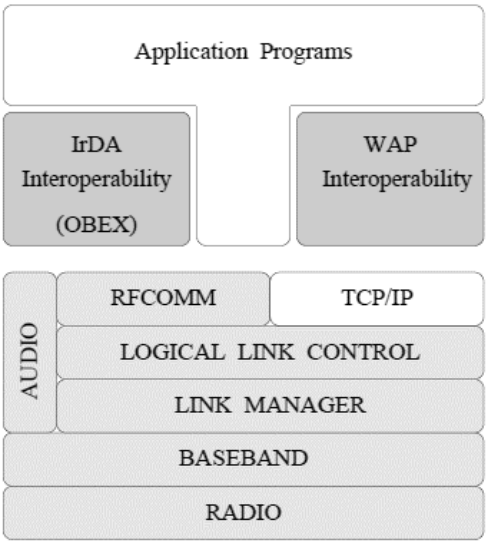

Figura 2.1. Pilha protocolar Bluetooth [8].

A camada Radio trata do envio e receção do fluxo de bits modulados na interface rádio. A camada Baseband controla as ligações físicas no acesso rádio, constrói tramas a entregar nas camadas acima e controla os saltos em frequência. A camada Link Manager tem como responsabilidade a definição e gestão do estado da conexão

com outros dispositivos, nomeadamente ao nível da segurança e autenticação, e gerir a energia nas ligações. A camada Logical Link Control realiza a multiplexação de pacotes para as camadas acima e realiza tarefas de descoberta de dispositivos. A camada Audio torna possível a transferência de ficheiros áudio entre um ou mais dispositivos Bluetooth apenas através da abertura de uma ligação áudio. A camada Radio Frequency Communications (RFCOMM) é responsável por fornecer emulação da interface RS-232. O resto da pilha protocolar trata do acordo de interoperabilidade com outras redes sem fios a que a tecnologia Bluetooth se compromete. Por exemplo, de forma a ser possível comunicação com redes IEEE 802.11/WiFi temos a camada TCP/IP que serve para fornecer uma ponte para a Internet. Por outro lado, a camada IrDA possibilita que dispositivos Bluetooth possam comunicar com dispositivos com IrDA.

2.1.1.2 Camada Física

As redes Bluetooth têm as suas frequências de operação na banda ISM (Industrial Scientific and Medical) dos 2.4 GHz e desta forma operam sem necessidade de licença. Observando a Tabela 2.1 podemos verificar que a banda de frequência utilizada não é comum para todo o mundo e que existem países em que a banda de alocação é menor.

Tabela 2.1. Frequência de operação do Bluetooth.

Área geográfica	Banda regulada	Canais RF
USA e Europa	2.400-2.4835 GHz	F=2402+k MHz, k=0,…,78
França	2.4465-2.4835 GHz	F=2454+k MHz, k=0,…,22

O *standard* Bluetooth especifica 23 ou 79 canais espaçados de 1 MHz para evitar interferências entre os canais. Utiliza também a técnica de espelhamento espectral por saltos em frequência (FHSS – Frequency Hopping Spread Spectrum) para evitar interferências externas. Desta forma, esta técnica faz com que a frequência da portadora esteja constantemente em alteração no final de cada transmissão devido ao uso de uma sequência pseudoaleatória partilhada pelo emissor e recetor.

Originalmente a modulação do sinal Bluetooth utiliza a técnica GFSK (Gaussian Frequency Shift Keying) cujo débito bruto atinge 1 Mbps. Contudo existem outros tipos de modulações que permitem débitos maiores, como por exemplo, a modulação PSK (Phase Shift Keying) cujo débito máximo é 2 Mbps e a 8DPSK (8-ary Diferential Phase Shift Keying) que atinge até um débito máximo de 3 Mbps.

2.1.1.3 Camada MAC

Nas redes Bluetooth estão definidos dois tipos de dispositivos: o mestre, que é responsável por iniciar a conexão, e o escravo. Um mestre pode conectar-se até 7 escravos ao mesmo tempo, e quando estas conexões ocorrem forma-se uma rede Bluetooth denominada *piconet*. É também possível que um dispositivo pertença a mais de que uma *piconet*, e quando assim é, forma-se uma *scatternet*. Na Figura 2.2 podemos observar três tipos de redes Bluetooth: à esquerda temos uma *piconet* com apenas um mestre e um escravo, ao centro outra *piconet* com um mestre e dois escravos e à direita uma *scatternet* composta por três *piconets*.

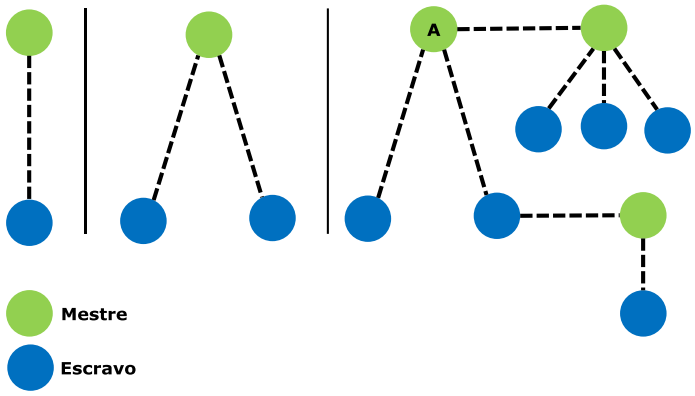

Figura 2.2. Topologias de redes Bluetooth.

A comunicação é sempre entre mestre-escravo ou escravo-mestre, não sendo admitida comunicação direta entre escravos. Como a comunicação dentro da *piconet* recorre a um meio partilhado, é necessário controlar o acesso ao meio a fim de evitar a perda de informação devido a colisões ou interferências. Esse controlo é feito pelo mestre e recorre a um protocolo de *polling*. Cada dispositivo Bluetooth tem a particularidade de possuir um endereço identificador único de 48 bits que é

utilizado como variável no cálculo da sequência de saltos, o que permite tornar esta sequência também ela única para cada dispositivo que assuma as funções de mestre da *piconet*.

Como no caso do dispositivo A da Figura 2.2, é possível que um dispositivo de uma *piconet* se junte a uma outra *piconet* obtendo o estatuto de escravo nesta segunda (mesmo sendo mestre na primeira). Assim, temos o dispositivo A como responsável por garantir a comunicação das duas *piconets*, tomando este dispositivo o nome de *gateway*. De realçar que um *gateway* apenas pode ser mestre numa das *piconets* pois caso assim não fosse iriam existir duas *piconets* com a mesma sequência de saltos, o que provocaria interferências dado a proximidade entre as redes.

2.2 Sensores

Os permanentes avanços no desenvolvimento de novos tipos de sensores, que possibilitam mensurar grandezas físicas e convertê-las em grandezas elétricas com a finalidade de serem tratadas eletronicamente, têm possibilitado um enorme desenvolvimento em diversas áreas. Como exemplos, temos a área militar, de saúde, domótica, ambiental, veicular, etc. De forma sucinta, um sensor é um dispositivo que muda o seu estado conforme a grandeza física recolhida. Através deste princípio novas grandezas físicas tomam um significado em termos eletrónicos e possibilitam que automações ao nível da monitorização, controlo de processos ou proteção sejam realizadas mediante valores registados pelos sensores.

A título exemplar vejamos de seguida dois exemplos de sensores que podem ser aplicados à monitorização de comportamentos fisiológicos de um indivíduo comum.

2.2.1 Sensor cardíaco

Uma das vertentes da recolha de dados sensoriais é no setor da saúde. Desta forma, inúmeros sistemas têm sido desenvolvidos por forma a possibilitarem a monitorização do batimento cardíaco [11] e posteriormente disponibilizá-lo para

consulta externa através de um relógio ou smartphone. A empresa Zephyr tem um vasto leque de dispositivos que têm como base a medição de dados fisiológicos. Um desses dispositivos é o HxM BT [9] e tem como vantagem a disponibilização do seu SDK para Android para que programadores possam construir a sua própria aplicação sem qualquer subscrição associada. Através deste dispositivo, e do seu SDK, é possível ter acesso à frequência cardíaca, velocidade, distância percorrida e número de passos realizados por um utilizador.

A comunicação entre o HxM BT e o smartphone é realizada com recurso à tecnologia sem fios Bluetooth no modo SPP (Serial Port Profile), sendo possível conectar apenas um dispositivo de cada vez. Após a conexão estar estabelecida, o HxM BT tem definido um período de 1 segundo de intervalo entre o envio de cada pacote de dados. Em termos de outras especificações relevantes podemos realçar os limites dos batimentos cardíacos entre os 25 e 240 bpm (batimentos por minuto), uma autonomia até 26 horas e um raio máximo de funcionamento até 10 metros.

O modo de funcionamento do dispositivo HxM BT não está preparado para processar nenhuma informação vinda do smartphone. Assim, o HxM BT apenas está definido para enviar mensagens com o formato que está ilustrado na Figura 2.3.

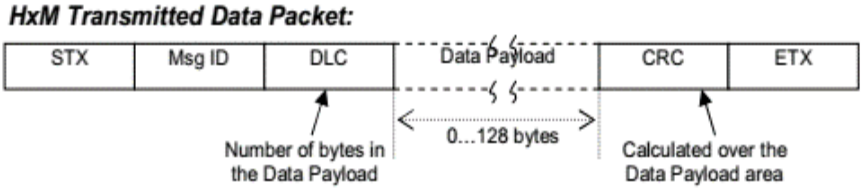

Figura 2.3. Formato das mensagens do HxM BT [9].

As mensagens são enviadas com uma frequência de 1 Hz, e do *payload máximo admissível* (128 bytes) apenas 55 bytes são preenchidos. Entre esses 55 bytes, temos a frequência cardíaca, a distância percorrida e o número de passos realizados pelo utilizador.

2.2.2 Sensor de torque e cadência

O módulo do sensor de torque e cadência (X-Cell RT digital), analisado com maior detalhe posteriormente na secção 3.2.3, ao contrário do sensor analisado anteriormente na secção 2.2.1, não tem funções de processamento de dados e respetiva construção de mensagens a serem consumidas por uma aplicação. Este módulo inclui somente um sensor que está desenhado para ser colocado na pedaleira da bicicleta e fornecer sinais em cada um das suas saídas (fios elétricos). Na Tabela 2.2 podemos visualizar quais os fios que este possui e respetivo modo de funcionamento característico de cada um.

Tabela 2.2. Características dos fios do X-Cell RT digital.

Cor do fio	Descrição	Sinal	Observação
Branco	Alimentação	+7 ... 16 V DC	-
Preto	Massa	0 V	-
Azul	Saída	Coseno	16 impulsos/rotação
Castanho	Saída	Seno	16 impulsos/rotação
Cinzento	Saída	Torque	+/- 10 mV/Nm

O sinal fornecido pelo fio castanho é o resultado da rotação da pedaleira da bicicleta. Esse sinal comporta-se como um seno digital (onda quadrada) e sempre que existe uma rotação completa da pedaleira este fio fornece 16 impulsos. Desta forma é possível calcular qual a velocidade de rotação da pedaleira em rpm (rotações por minuto) ou rad/s, conforme explicação na secção 4.3. O fio azul (cosseno) tem o mesmo comportamento, sendo a única diferença o desfasamento de 90 graus com o sinal do fio castanho.

A saída do fio cinzento representa o torque que o ciclista está a exercer sobre a pedaleira. Este torque provoca uma variação da voltagem conforme a força que está a ser aplicada na pedaleira pelo ciclista. A voltagem obtida pode ser convertida em Nm (Newton metro) como é também explicado na secção 4.3.

2.3 Bicicletas elétricas

É cada vez mais comum a utilização de bicicletas como meio de transporte ou para fins desportivos. Mesmo que a sua utilização seja simplesmente como meio de transporte, contrariamente aos meios de transporte pessoais como o carro ou motociclo, esta introduz uma componente desportiva. A partir da década de 90, através da intensificação de utilização das bicicletas por parte da população, e com a aposta no seu desenvolvimento por parte das empresas e universidade ligadas ao ramo eletrónico, proporcionaram-se condições para o desenvolvimento de motores, sensores e baterias capazes de permitir que se evoluísse no contexto das bicicletas elétricas. Com esta aposta foi possível a produção massiva de bicicletas elétricas, atraindo assim mais pessoas para este tipo de transporte.

A bicicleta elétrica utilizada nesta dissertação [21] já se encontra desenvolvida, assim, o desenvolvimento do algoritmo de controlo de esforço e a integração dos componentes físicos necessários têm de respeitar de forma coerente, acertada e compatível as características já existentes da própria bicicleta.

A bicicleta utilizada já tem em consideração as normas aplicadas às bicicletas elétricas. Segundo o artigo 112º do código da estrada, as bicicletas elétricas são veículos de tração humana (velocípedes) que estão equipadas com um motor elétrico que pode ser utilizado pelo ciclista como auxiliar de potência. Esta ajuda só é permitida após o arranque da bicicleta com recurso exclusivo à propulsão humana. A potência máxima que este pode debitar é de 250 W e a sua alimentação terá de ser reduzida linearmente com o aumento da velocidade e desligada quando a velocidade atingir os 25 km/h ou se o ciclista parar de pedalar.

Em análise à bicicleta elétrica do sistema podemos verificar que esta utiliza um motor BLDC (Brushless DC) que assenta na filosofia dos motores síncronos. Este tipo de motores tem a particularidade de combinar a rotação do estator e do campo magnético gerado pelo rotor na mesma frequência de funcionamento. O motor está preparado originalmente para operar em 3 níveis de ajuda, podendo posteriormente ser alterado para funcionar em mais níveis.

O sistema de controlo é também uma componente que requer alguma atenção, pois é aqui que estão salvaguardados as normas, referidas no segundo parágrafo, aplicadas às bicicletas elétricas. É parte constituinte deste sistema de controlo 4 tipos de sensores:

- Sensor de corrente que permite verificar qual a corrente existente no controlador do motor BLDC e garantir que aquando de um aumento da corrente no circuito este possa ser desativado a fim de não danificar o módulo;
- Sensor de tensão que possibilita saber qual o valor instantâneo da tensão nos terminais da bateria, permitindo saber qual o valor da carga da bateria a fim de antecipar o carregamento e verificar qual o nível de descarga que esta realizou;
- Sensor magnético para realizar a aquisição da velocidade a que a bicicleta circula, dado que, como já referido anteriormente, a legislação portuguesa em vigor impõe que ao serem atingidos os 25 km/h a alimentação do motor tenha de ser interrompida. Este sensor é do tipo *reed switch*, que confere um modo de funcionamento muito semelhante a um interruptor. Como podemos visualizar no esquema da Figura 2.4 o sensor magnético encontra-se em circuito aberto e assim que um elemento ferromagnético se aproxime do sensor o circuito elétrico fecha-se. Deste princípio de funcionamento resulta um sinal digital que é de 5 V quando este está em aberto e de 0 V quando este está fechado;

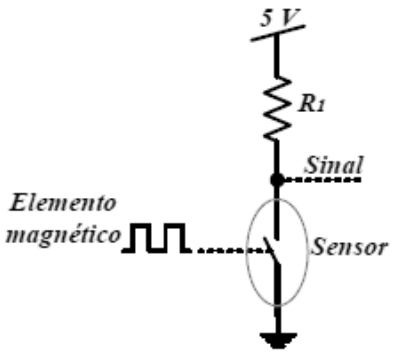

Figura 2.4. Esquema elétrico do sensor magnético.

- Sensor de posição, que torna possível a aquisição da informação relativa ao sentido da rotação da pedaleira, para que esta possa ser tratada ao nível do processamento de controlo. Assim é possível saber se o ciclista está a pedalar e em que sentido o está a fazer. Torna-se também possível saber se este parou de pedalar para que a ajuda do motor seja também desabilitada.

Todos estes sensores estão combinados de forma a que a bicicleta tenha um comportamento bastante seguro e fiável no que diz respeito à integridade do circuito de controlo, bem como ao cumprimento da legislação para as bicicletas elétricas.

2.4 Dispositivos móveis

O conceito de dispositivo móvel pode hoje ser facilmente confundido com o de smartphone dado os inúmeros serviços que estes últimos disponibilizam ao utilizador. Para além de funcionalidade básica de voz, dados e serviços típicos de imagem/som, têm sofrido uma enorme evolução no que diz respeito à sua capacidade de processamento a ponto de estar equiparável a um PC. Com recurso a este poder de processamento, cada vez mais aplicabilidade se lhe confere no que diz respeito ao tipo de tarefas exigidas. Hoje em dia têm processamento fotográfico de alta resolução, áudio hi-fi, e uma GPU (Graphics Processing Unit) para tratar dedicadamente todo o processamento relativo à componente gráfica, inclusive 3D, fazendo com que a CPU (Central Processing Unit) fique livre para tratar de outros processos.

Através desta aposta evolutiva, a massificação de smartphone*s* em relação a PCs domésticos, PCs portáteis e até mesmo *tablets*, por parte da população, cresceu abruptamente, sendo que a IDC (International Data Corporation) prevê que até 2017 os smartphone*s* tenham uma cota de mercado na ordem 70.5% [22] em relação aos equipamentos referidos anteriormente.

Devido a esta excelente aceitação do mercado, foi possível o amadurecimento dos smartphone*s* como plataforma de computação. Para além deste crescimento em termos de processamento, foram integrados um conjunto de sensores, como um acelerômetro, bússola digital, giroscópio, GPS, sensores de luminosidade e proximidade, entre outros. Juntando esta capacidade sensorial, com a massificação por parte da população e a facilidade disponibilizada para desenvolver aplicações, estamos perante um dispositivo com enorme capacidade para realizar tarefas de recolha de dados sensoriais para áreas já referidas anteriormente como a saúde [11][15], transporte [14][20], entre outros.

Para além das características já mencionadas, os tipos de comunicação que permitem têm vindo a acompanhar o desenvolvimento das comunicações sem fios. Wi-Fi, Bluetooth e NFC (Near Field Communication) são alguns dos tipos de tecnologias de comunicação sem fios mais importantes que suportam, podendo também servir de interface para interligação com outros tipos de sensores externos ao smartphone.

Para complementar toda a capacidade de processamento acima descrita, o sistema operativo é o principal responsável pelo desempenho que o smartphone vai conseguir atingir. Nos dias de hoje existem vários sistemas operativos disponíveis para esta plataforma, cada um com as suas políticas de funcionamento, gestão de memória, interface com o utilizador, permissões de tarefas, etc.

Na Tabela 2.3 podemos visualizar e comparar a cota de mercado do top 5 dos sistemas operativos no que dizem respeito aos smartphones, relativamente às unidades vendidas e respetiva cota do mercado do quarto trimestre de 2012 e quarto trimestre de 2013.

Tabela 2.3. Volume de vendas no quarto trimestre de 2012 e 2013 (unidades em milhões) e quota de mercado dos sistemas operativos móveis [23].

Sistema Operativo	Volume de vendas (4T13)	Quota de mercado (4T13)	Volume de vendas (4T12)	Quota de mercado (4T12)
Android	226.1	78.1%	161.1	70.3%
IOS	51.0	17.6%	47.8	20.9%
Windows Phone	8.8	3.0%	6.0	2.6%
BlackBerry	1.7	0.6%	7.4	3.2%
Outros	2.0	0.7%	6.7	2.9%
Total	**289.6**	**100.0%**	**229.0**	**100.0%**

2.4.1 Android

Inicialmente, a Android Inc. era uma pequena empresa sediada no estado Norte Americano da Califórnia que apostava no desenvolvimento de plataformas para dispositivos móveis. O principal objetivo dos seus fabricantes consistiu em desenvolver uma plataforma de código aberto que fosse facilmente adaptável a diversos tipos de *hardware*. No ano de 2005 a empresa Google adquiriu a Android Inc., e desde então tem desenvolvido a capacidade do Android no desempenho de tarefas de sistema de operação central.

O crescimento do Android como sistema operativo foi exponencial, e está patente na Tabela 2.3 a diferença existente para os concorrentes atuais. Um dos pontos que potenciam estes números consiste na estratégia traçada de fazer com que o Android seja uma plataforma transversal ao *hardware* e não dependa de nenhum fabricante, fazendo com que diversos fabricantes adotem o sistema operativo Android para os seus smartphones.

Através da sua política de código aberto e através de um conjunto de ferramentas para a criação e desenvolvimento de aplicações fornecidas pelo Android SDK (Software Development Kit), as aplicações desenvolvidas por terceiros para Android têm de respeitar o paradigma da linguagem de programação Java.

2.4.1.1 Arquitetura do sistema operativo

O sistema operativo Android pode ser dividido em cinco camadas. Como podemos visualizar na Figura 2.5, e de cima para baixo, existe a camada Aplicacional, a Framework de Desenvolvimento, o Android Runtime, as bibliotecas nativas do Android e o Kernel.

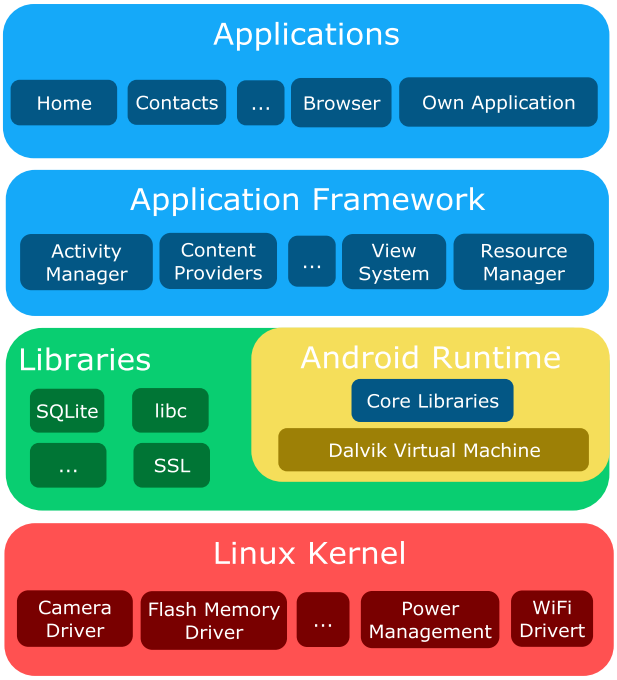

Figura 2.5. Arquitetura do Sistema Operativo Android.

No topo da hierarquia encontra-se a camada aplicacional, que contém todas as aplicações visíveis ao utilizador. Por exemplo, temos o navegador de Internet, os contactos armazenados, calendário, etc. De realçar que é através desta camada que podemos ter acesso a aplicações pessoais desenvolvidas de raiz ou a aplicações adquiridas na loja da Google (chamada Play Store). Todas as aplicações executadas nesta camada são escritas em Java e recorrem à máquina virtual Dalvik para realizar a sua execução. Na camada abaixo encontra-se a camada Framework de Desenvolvimento, que disponibiliza nativamente toda a API, em linguagem Java, necessária para serem desenvolvidas aplicações para a plataforma Android. É também disponibilizada a API para interação com todos os componentes de *hardware* do smartphone (ex.: ativação de Bluetooth, Wi-Fi, colunas, câmara, etc.)

sem que seja necessário conhecer tecnicamente esses componentes. De um ponto de vista mais visual fornece um gestor de atividades (*Activity Manager*) que consiste na tela com a qual os utilizadores podem interagir para realizar operações (e.g., visualizar um mapa, enviar um email, etc.). Normalmente cada *activity* preenche a totalidade do ecrã do smartphone e marca a interface das funcionalidades do *software* com o utilizador.

Descendo um pouco mais, encontramos a camada Android Runtime, que é composta pela máquina virtual Dalvik, onde são executadas as aplicações desenvolvidas, e por uma API de desenvolvimento Java onde, tal e qual como na camada acima, o programador tem acesso a várias bibliotecas do sistema e do *hardware*, sem que para isso tenha de ter um conhecimento técnico ao nível da sua constituição. A camada das bibliotecas nativas do Android tem as suas funções escritas em linguagem C/C++. Nela está definida, e otimizada para dispositivos móveis, a biblioteca *standard* da linguagem C (libc), um SGBD (Sistema de Gestão de Base de Dados) chamado SQLite, o protocolo SSL (Secure Sockets Layer) para confidencialidade das ligações HTTP, entre outros componentes.

Por fim, a última camada traduz a base do Android, que consiste numa versão modificada do *kernel* do Linux 2.6 com o objetivo de garantir serviços ao nível da segurança, bem como gerir a memória, processos e questões relativas à rede. Como base de operação fundamental, esta fornece uma abstração do *hardware* para as camadas de *software* acima descritas.

2.5 Trabalho relacionado

Nesta secção são apresentados trabalhos relacionados com o sistema implementado, sendo que, para além de estarem referenciados projetos em que as bicicletas elétricas são o principal foco, existem também projetos que têm por base demonstrar as potencialidades de uma rede de sensores, com recurso a tecnologias de comunicação sem fios como o Bluetooth, IEEE 802.15.4/ZigBee ou Wi-Fi, em áreas como a saúde e aeronáutica.

2.5.1 Redes de sensores sem fios

Como referido na secção 2.2 o desenvolvimento de sensores tem vindo a permitir que novas aplicações sejam criadas para áreas até então pouco exploradas no que diz respeito ao controlo automático baseado em variáveis físicas inerentes a um determinado sistema.

O setor da saúde tira bastante proveito destas progressivas evoluções dado que vários projetos de investigação procuram explorar novas formas que garantam um melhor serviço prestado aos doentes. Um desses projetos é o MOHLL [11] que tem como principal propósito fornecer aos médicos uma visualização em tempo real dos parâmetros vitais dos seus pacientes através de um PDA conectado à Internet. Os parâmetros vitais disponibilizados são a SpO2 (oxigenação do sangue), temperatura corporal e o ECG (eletrocardiograma). A maior vantagem da evolução deste projeto consiste na possibilidade dos médicos conseguirem acompanhar o estado dos seus doentes sem que para isso tenha de estar ao lado deles visualizando os dados vitais num determinado equipamento estático. Como podemos ver na Figura 2.6, todo o processo é iniciado através da aquisição dos dados vitais por parte de sensores com base na tecnologia IEEE 802.15.4/ZigBee. Após esta recolha, os dados são enviados através de uma rede de sensores sem fios (WSN, Wireless Sensor Network) do hospital para o Servidor de Dados. Neste percurso os dados dos sensores são encaminhados através da utilização de vários dispositivos de rede (nós) intermédios. Estes dispositivos de rede têm a capacidade de se auto-organizar, admitem encaminhamento *multi-hop* e estão preparados para dar resposta a modificações topológicas originadas não só pela modificação da disposição dos nós intermediários mas também pela falha de funcionamento de um qualquer nó intermédio. Existe um nó coordenador da rede que é responsável por receber e verificar a integridade dos dados, para que estes sejam enviados para um computador *gateway* (ZigBee-to-Wi-Fi), que os reencaminha, via HTTP (Hypertext Transfer Protocol), para o Servidor de Dados. Desta forma é possível que a informação relativa aos pacientes esteja acessível, em tempo real, através da Internet ou de uma rede Wi-Fi do hospital.

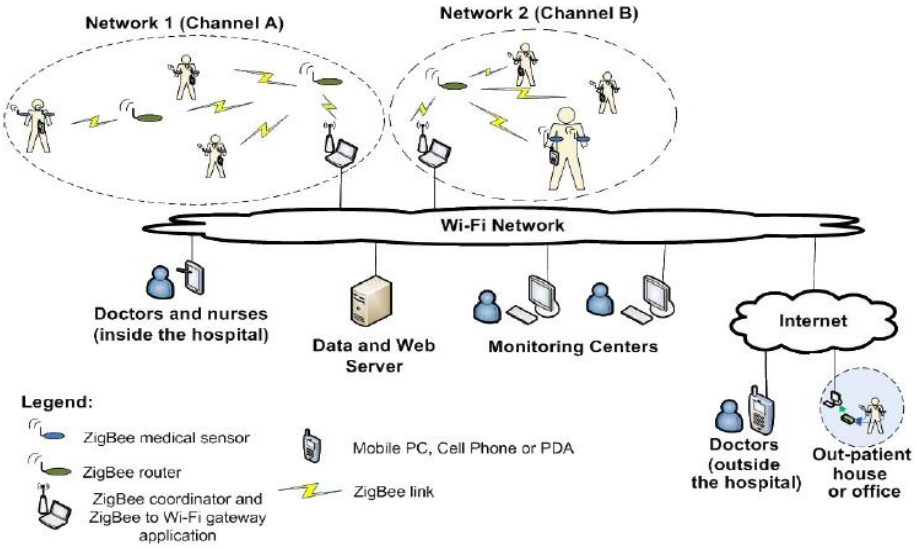

Figura 2.6. Arquitetura do projeto MOHLL [11].

Em [12] os autores do projeto implementaram uma rede de sensores com aquisição e controlo de dados em tempo real baseada na tecnologia de comunicação Bluetooth. O projeto desenvolvido foi usado num veículo aéreo não tripulado (VANT) que possuí uma estação mestre ligada ao controlador de voo e até um máximo de 7 estações escravas distribuídas por toda a estrutura da aeronave e conectadas a vários sensores e atuadores. O microcontrolador utilizado foi o MSP430 da Texas Instruments que confere um baixo consumo de bateria (cerca de 280 µA a 1 MHz e 2.2 VDC) e disponibiliza programação por interrupções, periféricos para comunicação em série, ADC (Analog-to-Digital Converter) de 12 bits, etc. Os módulos Bluetooth usados diferem de mestre para escravo, sendo que para os primeiros foram usados os connectBlue OEMSPA33i e para os segundos o connectBlue OEMSPA13i. A Figura 2.7 apresenta a disposição dos nós Bluetooth sobre a aeronave.

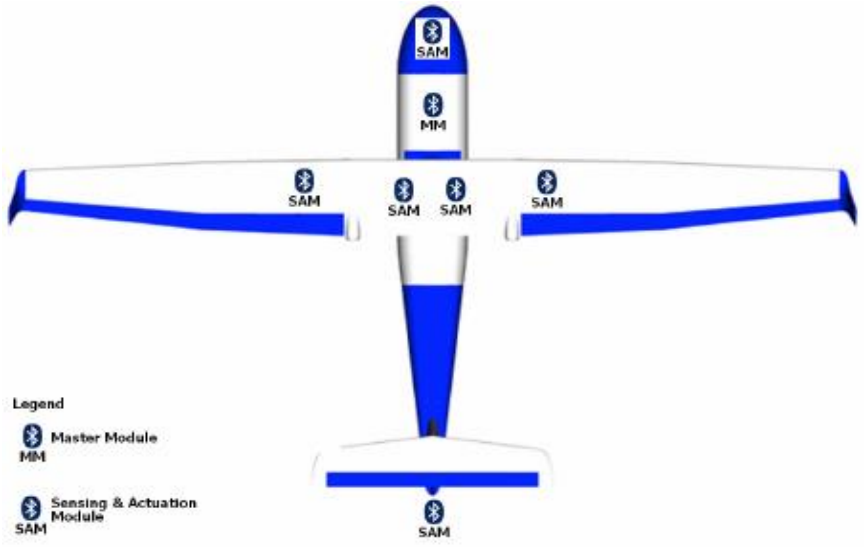

Figura 2.7. Distribuição dos nós Bluetooth na estrutura da aeronave.

Como podemos observar, o módulo mestre (MM) está colocado na fuselagem e opera como controlador do voo, armazena internamente um registo de voo e controla as comunicações com a estação terrestre. Os outros seis nós (Sensing and Actuation Modules - SAM) são escravos na rede. Estes estão distribuídos de forma lógica, sendo que em cada asa existe um nó SAM que é responsável por controlar a velocidade do motor de propulsão elétrica, regular o ângulo formado pelo aileron e monitorizar a temperatura. Na cauda temos um outro nó SAM que controla o leme e o elevador traseiro. No corpo de fuselagem existem mais 2 nós SAM onde um realiza funções de GPS (Global Positioning System) e o outro de giroscópio para que seja possível a realização de tratamento de dados referente à navegação. Por fim, o último nó SAM está localizado no nariz da aeronave e tem integrado seis sensores de pressão que disponibilizam informações relativas ao controlo do voo e uma sonda ultrassónica responsável pelo sistema automático de descolagem e aterragem.

2.5.2 Bicicletas elétricas

Um outro tipo de aplicação já explorada consiste em dotar as bicicletas elétricas com sensores a fim de monitorizar uma determinada grandeza física para que esta seja processada e posteriormente despoletar um determinado comportamento pré-configurado. As características ecológicas e económicas das bicicletas conferiram-lhe

uma forte adesão de utilização por parte da população mundial. O projeto PRESTO (Promoting Cycling for Everyone as a Daily Transport Mode) [13] da *EU's Intelligent Energy,* através da Tabela 2.4, traduz em números o crescimento que se fez sentir na aquisição de bicicletas elétricas até 2010 e uma previsão, que se verificou, até 2012.

Tabela 2.4. Número de bicicletas elétricas vendidas.

	2007	2008	2009	2010	2011	2012
China	21.000.000	22.000.000	21.000.000	22.000.000	23.000.000	25.000.000
India	85.000	20.000	7.500	10.000	15.000	17.500
Japan	300.000	300.000	300.000	325.000	350.000	350.000
EU	250.000	500.000	750.000	1.000.000	1.350.000	2.200.000
Taiwan	10.000	10.000	11.000	12.000	14.000	15.000
SE Asia	200.000	500.000	400.000	600.000	800.000	1.000.000
USA	120.000	170.000	150.000	300.000	400.000	500.000
Total	**21.965.000**	**23.500.000**	**22.618.500**	**24.247.000**	**25.929.000**	**29.082.500**

Com isto, universidades e empresas de todo o mundo também vislumbraram enormes potencialidades de aplicações capazes de melhorar o funcionamento das bicicletas elétricas e até mesmo conferirem-lhe novas aplicações de utilização com base em dados recolhidos de sensores implantados na própria bicicleta.

Como bom exemplo disto, temos um projeto proveniente da academia Samsung Maestros, o projeto Samsung Smart Bike [14] apresentado na *Milan Design Week* no decorrer do presente ano de 2014. Este realçou, como principal preocupação a colmatar a atribuição de veículo mais inseguro do mundo às bicicletas por parte da ISTAT (National Institute of Statistics) em Itália. Assim, a Samsung aliou à sua componente de comercialização de smartphones a integração com esta nova temática relacionada com as bicicletas. Os principais componentes inovadores que esta bicicleta integra na sua estrutura são: uma bateria, um Arduino, um módulo Bluetooth, Wi-Fi, 4 projetores laser e uma camara digital. A camara digital tem como função enviar para o smartphone, em tempo real, a visão traseira, permitindo que o ciclista veja sempre o que se passa no sentido contrário à sua direção. Os projetores laser efetuam a projeção para o chão de linhas guias em torno da bicicleta para que os ciclistas sejam vistos mais facilmente e para que os condutores tenham uma referência para a distância de segurança a respeitar. Esta ativação pode ser feita

manualmente numa aplicação no smartphone ou através de um sensor de luminosidade que os ativa quando estiver abaixo de um determinado nível de luz. Como possibilidade de armazenamento de rotas realizadas pelo ciclista, a bicicleta tem integrado um sistema GPS.

Um outro projeto chamado Mobile Health Monitoring Systems [15], à semelhança de [11], não foi exclusivamente dimensionado a pensar nas bicicletas elétricas, mas também para a área da saúde. Este demonstra bem a potencialidade da recolha de dados de sensores e formas de os aplicar com benefício humano. Neste caso em concreto, a principal preocupação dos seus criadores é disponibilizarem os valores obtidos da leitura de sensores para que possam ser acedidos remotamente e em tempo útil, para garantir o atempado auxílio de pacientes por parte de médicos e enfermeiros. Assim, e em caso de uma emergência, a unidade de socorro que for requerida para se deslocar ao local de um acidente pode antecipar a triagem, que só seria realizada aquando da chegada ao local de emergência, preparando-se atempadamente para um melhor processo de ajuda durante o caminho até ao hospital. Contudo, e contrariando o princípio exclusivo de aplicação de [11], uma das aplicabilidades pela qual este sistema foi pensado e dimensionado é no âmbito das bicicletas elétricas, e neste caso em concreto é necessário que cada individuo possua um nó sensor sem fios (apelidado de BioTE pelos seus desenvolvedores). Como é visível na Figura 2.8 o BioTE é constituído por um microcontrolador MSP430 [16] e um *transceiver* CC2420 RF [17], ambos da Texas Instruments.

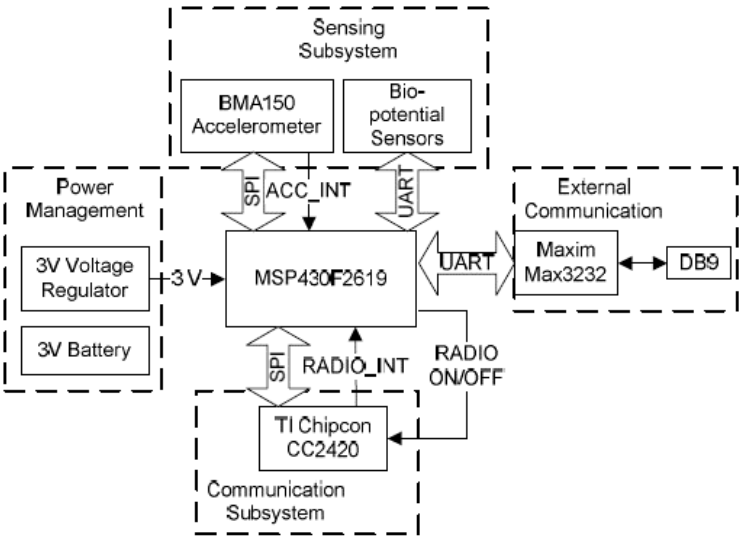

Figura 2.8. Arquitetura BioTE [15].

O MSP430 possui periféricos para comunicação em série que permitem a comunicação com vários tipos de sensores digitais como um medidor de pulsação, um sensor de temperatura corporal, um sensor de temperatura ambiente e um sensor para cálculo da velocidade da bicicleta. Após a aquisição destes valores, estes são enviados para um coordenador central através um módulo de comunicação IEEE 802.15.4/ZigBee, que por sua vez se encarrega de enviar os valores para um dispositivo móvel com recurso a comunicação Bluetooth. Por sua vez o dispositivo móvel realiza o *upload* desses dados via rede celular UMTS (Universal Mobile Telecommunication System) para um servidor remoto, que os armazena numa base de dados SQL (Structured Query Language), a qual será acedida via HTTP por uma aplicação cliente através de um qualquer *browser* Web.

Num contexto mais comercial, os investigadores do projeto BikeNet [18] verificaram que os típicos sistemas comercializados que cooperavam com as bicicletas e com os seus utilizadores tinham como base as mesmas preocupações. Consistiam acima de tudo na monitorização da velocidade de deslocação, distância percorrida e calorias queimadas. Desta forma este projeto visa dar resposta a outras componentes relevantes para os utilizadores de bicicletas. Constataram que os ciclistas não fazem exercício num ambiente isolado e que cada vez mais procuram rotas seguras e silenciosas para praticar exercício, e decidiram que o conhecimento

relativo à exposição de poluição atmosférica e sonora, bem como o perigo devido à densidade automóvel, seriam fatores preponderantes de avaliação do trajeto realizado. Neste sentido foi utilizada uma plataforma já desenvolvida chamada Tmote Invent [19], que já possui sensores integrados, bem como um microcontrolador nativo com TinyOS. Esta plataforma fornece um acelerómetro com dois eixos, um termístor e um microfone. Por outro lado, integra externamente um dispositivo móvel chamado N80 que oferece o serviço de câmara e um outro microfone. Para que todos os requisitos do projeto fossem cumpridos foi necessário acrescentar sensores que comunicam pelas interfaces de comunicação série do microcontrolador MSP430 existente no Tmote Invent para manipulação conjunta dos valores dos sensores. Como podemos verificar na Figura 2.9, para além dos sensores já existentes integraram um magnetómetro externo para deteção de automóveis na proximidade. Para medir a direção e o desvio efetuado relativamente ao campo magnético terrestre foi adicionado o sensor magnético-indutivo Honeywell HMC1052L e um sensor para medição do CO_2 e da temperatura chamado Telaire 7001; ambos são ligados aos ADCs disponíveis no microcontrolador MSP430 do Tmote Invent. Foi também conectado a um periférico de comunicação série do MSP430 um dispositivo de registo horário e GPS chamado Garmin Etrex. A toda esta rede de sensores foi atribuído o nome de BAN (Bicycle Area Network) a qual comunica com o smartphone através de um *gateaway* IEEE 802.15.4/Bluetooth.

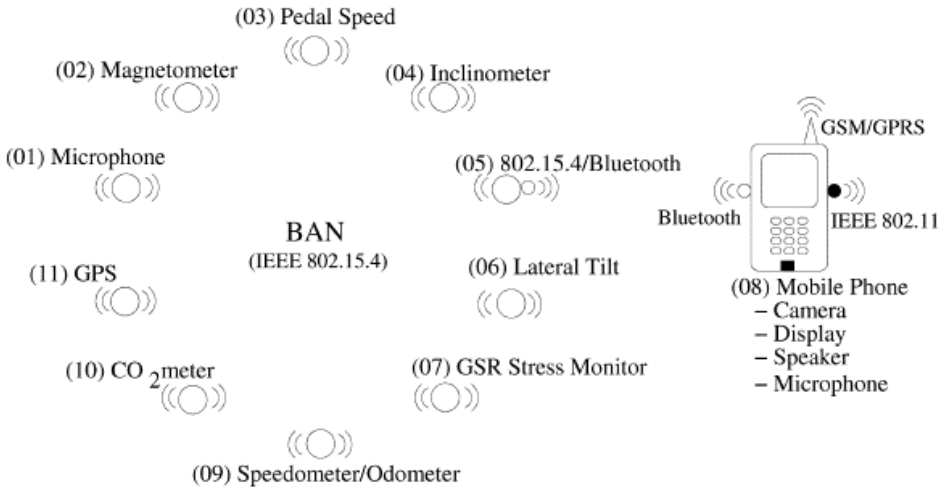

Figura 2.9. Sensores do projeto BikeNet [18].

Por sua vez o smartphone comunica para um servidor remoto através de rede móvel GPRS (General Packet Radio Service), no caso de existir conexão ao pacote de dados da operadora, ou através de um AP (Access Point) Wi-Fi. Como em [15] no que diz respeito à consulta *online*, uma aplicação Web consome os dados enviados para o servidor e apresenta-os aos clientes registados para possível consulta de todos os dados recolhidos num determinado percurso.

O projeto Copenhagen Wheel [20] procura que qualquer pessoa possa transformar, de uma forma rápida e fácil, uma simples bicicleta comum numa bicicleta elétrica sem que para isso seja necessário conhecimentos eletromecânicos. O principal foco deste projeto visa explorar e promover as bicicletas elétricas nas cidades, sendo a sua capacidade de adaptação possível dada a facilidade com que foi pensado o projeto. Trata-se de uma roda traseira facilmente adaptável (*plug-and-play*) a uma bicicleta comum e que integra vários tipos de funcionalidades. Esta roda possui:

- Um módulo de comunicação Bluetooth para emparelhamento com um smartphone (neste caso em concreto um iPhone) para proceder à recolha de informação proveniente dos sensores descritos posteriormente;
- Uma sistema designado por *Braking* que aciona um travão eletrónico que reduz a velocidade se pedalarmos para trás;
- Uma tecnologia de regeneração semelhante à que revolucionou a Fórmula 1 chamado KERS (Kinetic Energy Recovery System), onde, aquando da travagem, a energia cinética é recuperada pelo motor elétrico que a armazena nas baterias contidas na roda;
- 3 velocidades de operação do motor;
- Mecanismo antirroubo que, em caso de movimento da bicicleta, verifica se o smartphone do ciclista está próximo e, no caso de não estar, dispara um alarme, reporta via GPRS a posição onde a bicicleta se encontra e bloqueia a roda traseira;
- Um módulo GPS e um conjunto de sensores ambientais que medem o nível de monóxido de carbono (CO), monóxido de nitrogênio (NOx), temperatura,

ruido (dB) e a humidade. Esta recolha é realizada a cada 2 segundos e enviada para o smartphone a fim de processar esses dados e determinar qual o nível de qualidade do meio ambiente para o ciclista.

3. Estudo do sistema

Este capítulo tem como principal objetivo abordar, de uma forma sintetizada, todos os componentes inerentes ao trabalho realizado na dissertação, por forma a demonstrar teoricamente a estratégia escolhida para a implementação do sistema desenvolvido.

3.1 Visão geral do sistema

O sistema de controlo de esforço é composto por dois componentes subdivididos em quatro blocos lógicos que cooperam conjuntamente. Como podemos ver na Figura 3.1, incluído no componente smartphone temos o bloco de decisão; por outro lado, os blocos *gateway*, de aquisição e de controlo são parte integrante do componente bicicleta.

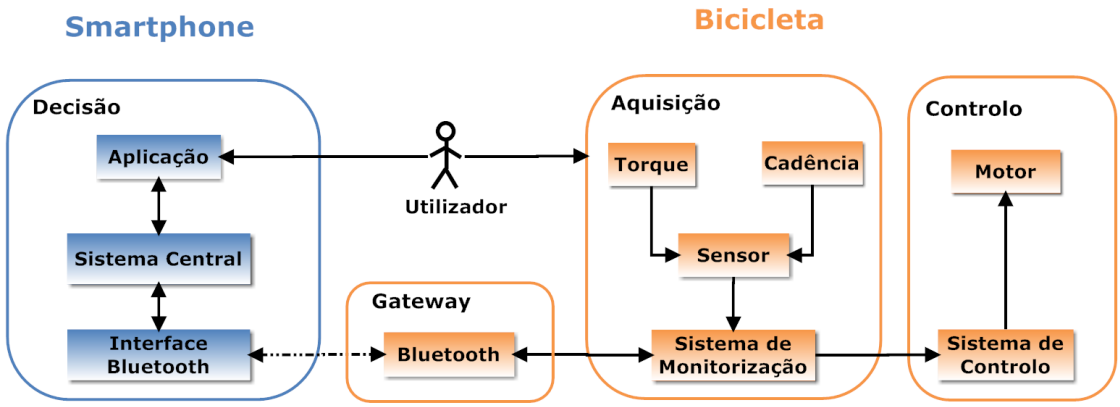

Figura 3.1. Arquitetura do sistema de controlo de esforço.

O utilizador tem contacto com uma aplicação dedicada ao funcionamento do sistema através do smartphone, e por outro lado utiliza a bicicleta. O smartphone é o responsável pela interface com o utilizador através da aplicação desenvolvida. Nesta é possível realizar uma personalização do modo de uso da bicicleta elétrica, como por exemplo, definir qual o esforço máximo e mínimo que o utilizador deseja tolerar

na utilização da bicicleta. Este componente recorre a tecnologia Bluetooth para comunicar com a bicicleta.

O bloco de aquisição é responsável por proceder à leitura periódica, por parte do sistema de monitorização, dos valores mensurados pelo sensor de torque e cadência. Após esta recolha, a informação é prontamente enviada para a aplicação do smartphone através do bloco *gateway* que recorre ao uso de interface rádio Bluetooth. Esta comunicação pode ser realizada nos dois sentidos pois a aplicação do smartphone é também responsável por executar um algoritmo de controlo de esforço que decidirá qual o nível de ajuda que o motor deverá executar. A ação desta operação de ajuda ocorre no bloco de controlo, onde o sistema de controlo receberá o valor a aplicar ao nível de ajuda do motor.

3.2 Hardware

3.2.1 Microcontrolador

O módulo utilizado nos blocos de aquisição e de controlo foi o LaunchXL-F28027, da Texas Instruments, que integra o microcontrolador Piccolo F28027 [24][25]. Embora estes blocos realizem tarefas bem distintas, cada um é constituído por um microcontrolador deste tipo, onde a principal razão por terem sido adotados dois microcontroladores está justificada na secção 4.5.1. O microcontrolador do bloco de aquisição tem como principal função ler os valores do torque, cadência e nível de bateria da bicicleta para transmiti-los ao smartphone. Por outro lado o bloco de controlo é o responsável pela receção do nível de ajuda a aplicar ao motor, chegando esta informação do smartphone por intermédio do microcontrolador do bloco de aquisição. Mais detalhes são apresentados na secção 4.3.

Na Figura 3.2 podemos ver o módulo LaunchXL-F28027 da Texas Instruments, que se trata de uma plataforma especialmente projetada para realizar tarefas de controlo em tempo real.

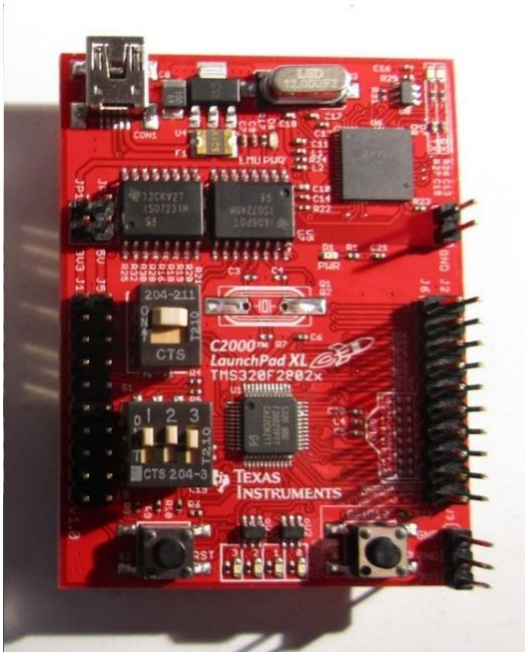

Figura 3.2. Módulo LaunchXL-F28027, da Texas Instruments.

O microcontrolador F28027 tem alimentação máxima suportada até 3.3 V, uma arquitetura de 32 bits e funciona a uma frequência de 60 MHz. Possui 32 kB de memória Flash e 6 kB de memória RAM (Random Access Memory). Tem disponíveis 40 pinos de I/O, entre os quais uma interface de comunicação série (SCI - Serial Communication Interface) assíncrona com dois sinais (SCIRX, SCITX), normalmente conhecida como UART, uma interface síncrona SPI (Serial Peripheral Interface), 18 pinos de I/O digitais, um pino de alimentação a 3.3 V e outro de 5 V. Possui também 13 canais ADC de 12 bits cada um, o que lhe confere uma resolução de 4096 níveis em 3.3 V, ou seja, uma mudança de nível por cada 805 µV de variação.

Relativamente a interrupções, estas podem ser acionadas externamente ou com recurso a temporizadores. No caso do módulo Texas LaunchXL-F28027 este possui 3 pinos para interrupções externas e tem disponíveis 3 *timers* (0, 1, 2) de 32 bits, estando o *timer* 2 reservado exclusivamente para funcionamento do próprio microcontrolador.

3.2.2 Módulo Bluetooth

O *gateway* usado pelo bloco de aquisição para enviar e receber informação do smartphone foi o módulo Bluetooth WRL-12580 [26]. Este permite a substituição de uma solução baseada em comunicação cablada, pela comunicação sem fios Bluetooth. Como podemos observar na Figura 3.3 o módulo Bluetooth disponibiliza uma interface protocolada RS-232/UART, que foi fisicamente ligada ao módulo LaunchXL-F28027 do sistema de monitorização do bloco de aquisição, e uma interface rádio Bluetooth que comunica com o smartphone.

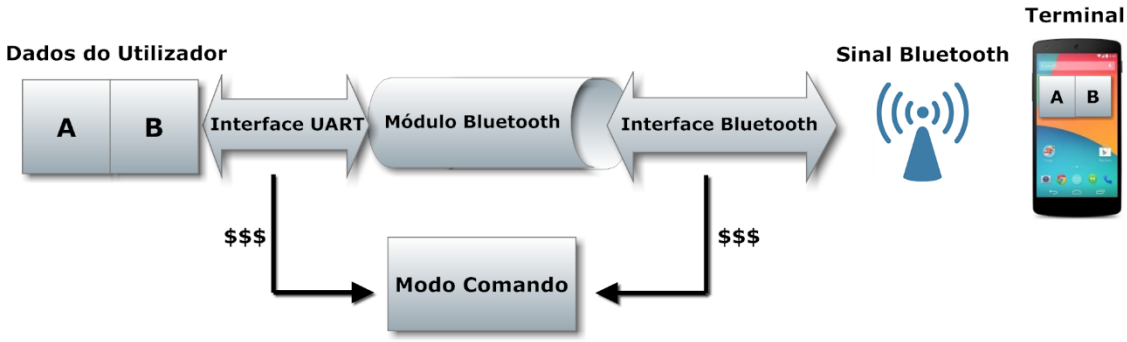

Figura 3.3. Modos de operação do Bluetooth WRL-12580 (adaptado de [26]).

A informação que o microcontrolador, do módulo LaunchXL-F28027, envia para a interface série, onde está ligado o módulo Bluetooth, é prontamente enviada via comunicação rádio Bluetooth para o smartphone. Por outro lado, quando o smartphone envia informação via Bluetooth esta é entregue ao microcontrolador do bloco de aquisição via interface série.

Como é também possível ver na Figura 3.3 o módulo Bluetooth pode operar de duas formas: no modo de dados (por omissão) ou no modo comando. No modo de dados este funciona como um túnel, ou seja, quando recebe dados pela interface Bluetooth, é realizada a separação dos cabeçalhos intrínsecos ao protocolo Bluetooth e são apenas passados os dados do utilizador para a interface UART, que os entrega ao microcontrolador. No caso de ser procedido o envio de dados de utilizador pela interface UART o módulo Bluetooth constrói o pacote Bluetooth e envia-o através da ligação sem fios Bluetooth para o smartphone.

O modo de comando é acionado sempre que a interface UART ou Bluetooth recebe a sequência de caracteres '$$$'. Na secção 4.4 está detalhadamente explicada esta funcionalidade e a forma como foi configurada para que o sistema funcione corretamente.

O módulo Bluetooth WRL-12580 foi projetado para ter um baixo consumo (média de 25mA) e contém reguladores de tensão integrados pelo que pode ser alimentado entre 3.3 V – 6 V. A comunicação série disponibilizada pode ser configurada para um *bit rate* entre os 2400 e 115200 bps, atingindo um alcance de transmissão máxima de 100 metros (Classe 1).

3.2.3 Sensor de torque e cadência

O sensor escolhido para disponibilizar ao bloco de aquisição os valores de torque e cadência exercidos pelo ciclista foi o X-Cell RT digital [27]. Este modelo tem inúmeras vantagens em relação a outros sensores, dado que foi alvo de estudo pelos seus criadores para ser colocado na bicicleta no sítio mais vantajoso. Vários locais eram passíveis de escolha, mas como podemos ver pelo formato do sensor na Figura 3.4, o local escolhido foi o rolamento interno da pedaleira, que lhe confere a vantagem de ficar invisível.

Figura 3.4. Sensor X-Cell RT digital.

Este local garante também uma perfeita proteção contra fatores externos ao sistema como o óleo, água, poeira, vibrações do piso, etc. Confere-lhe ainda uma baixa necessidade de manutenção, uma global compatibilidade com os modelos de bicicletas existentes e a não necessidade de calibração de cada vez que a bicicleta é desmontada para transporte.

Na Figura 3.5 podemos observar o comportamento elétrico do sensor para o fornecimento do torque e cadência.

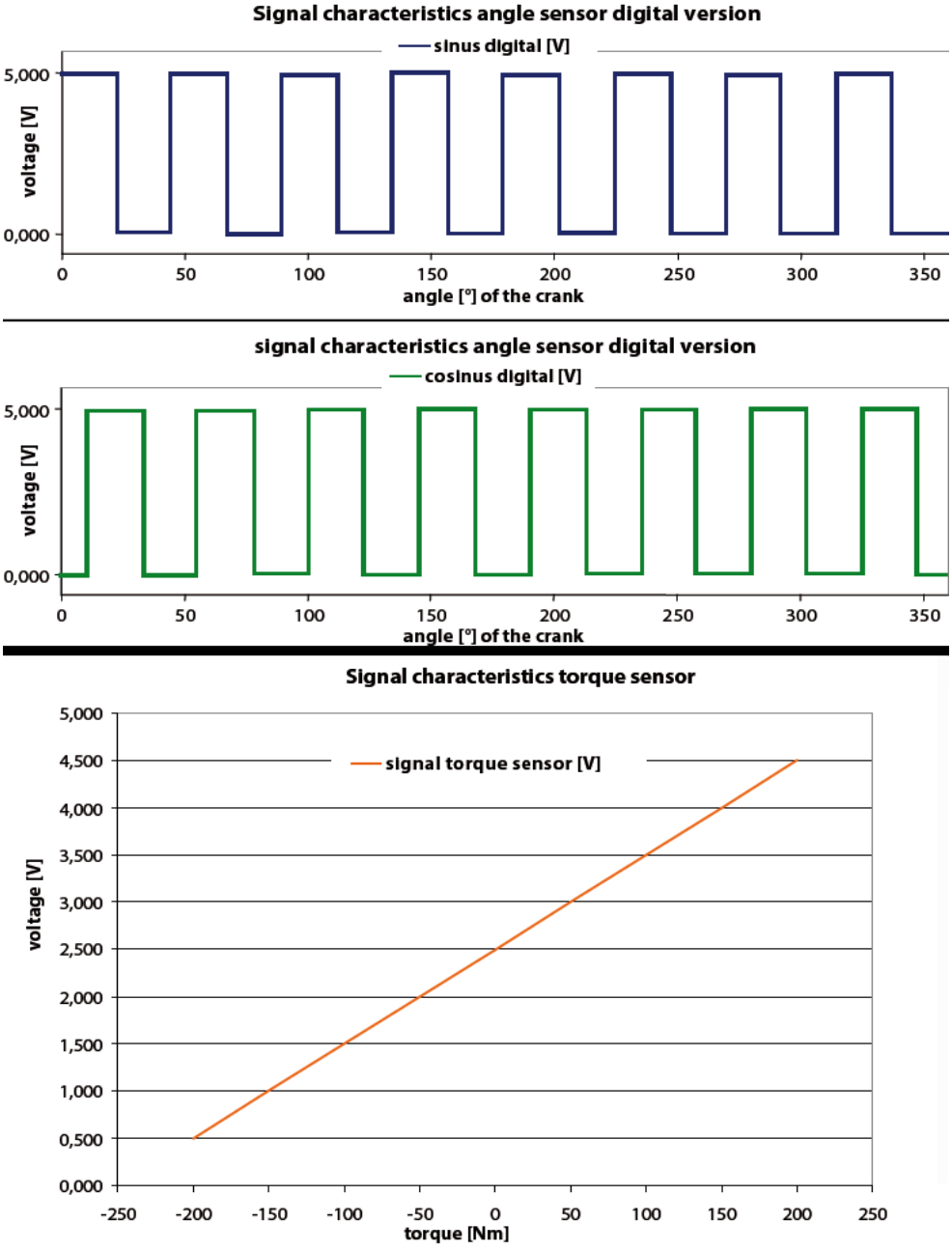

Figura 3.5. Comportamento dos sinais de cadência e torque fornecidos pelo X-Cell RT [27].

Como já foi referido anteriormente na secção 2.2.2, os fios castanho e azul fornecem sinais relativos à cadência, sendo o castanho correspondente a um seno e o azul a um cosseno, ambos digitais. Na parte superior da Figura 3.5 temos o comportamento elétrico desses dois sinais, sendo o sinal de cima correspondente ao seno e o de baixo ao cosseno. Sempre que o ciclista realiza uma pedalada completa, para que seja possível determinar a velocidade, são fornecidos pelo sensor 16 impulsos de seno e cosseno.

Na parte de baixo da Figura 3.5 temos o comportamento do sinal de torque. Como podemos observar, este tem um comportamento linear. Quando não existe torque na pedaleira, o fio cinzento fornece 2.5 V o que significa que temos um torque de 0 Nm. Como exemplo, podemos também verificar que assim que o ciclista começa a pedalar e a voltagem do torque aplicado na pedaleira corresponda a 3 V, o torque correspondente passa a ser de 50 Nm. Desta forma é possível criar uma relação entre a voltagem fornecida e o torque que está a ser aplicado na pedaleira.

Em termos mais específicos, temos na secção 4.3 a especificação da implementação do trabalho realizado para a correta correspondência aos requisitos do sistema.

4. Implementação do sistema

Este capítulo começa por apresentar a implementação inicial do sistema com recurso à plataforma Arduino. Posteriormente é explicada a implementação do sistema final, bem como a forma utilizada para a aquisição e tratamento dos valores de torque, cadência e respetivo esforço. Relativamente à componente física, é planificada a forma como o módulo Bluetooth está interligado com o módulo LaunchXL-F28027 e é também detalhada a configuração do seu *hardware* e *software*. É igualmente abordado o protocolo de comunicação utilizado entre a bicicleta e o smartphone bem como o atraso fim-a-fim.

Antes de se introduzir o sistema operativo Android, é apresentado um *software* utilizado no início do desenvolvimento do projeto, implementado em linguagem C e a operar num PC. Posto isto, segue-se com uma demonstração dos vários menus opcionais desenvolvidos para o *software* Android que determina o método de utilização da bicicleta, como se procedeu para a aquisição dos dados dos sensores e a forma que está implementada a solução relativa ao algoritmo de controlo de esforço.

4.1 Implementação inicial

O sistema foi inicialmente pensado e projetado para ter como base do bloco de aquisição a plataforma Arduino. Foi desenvolvida grande parte da solução tendo em consideração o paradigma desta plataforma, inclusive a sua API de desenvolvimento, chamada Wired, baseada na linguagem de programação C. Através da Figura 4.1 e do Anexo I podemos consultar uma parte do trabalho realizado, desde a especificação e configuração da plataforma Arduino Duemilanove até a apresentação correta de como integrar-lhe a comunicação Bluetooth.

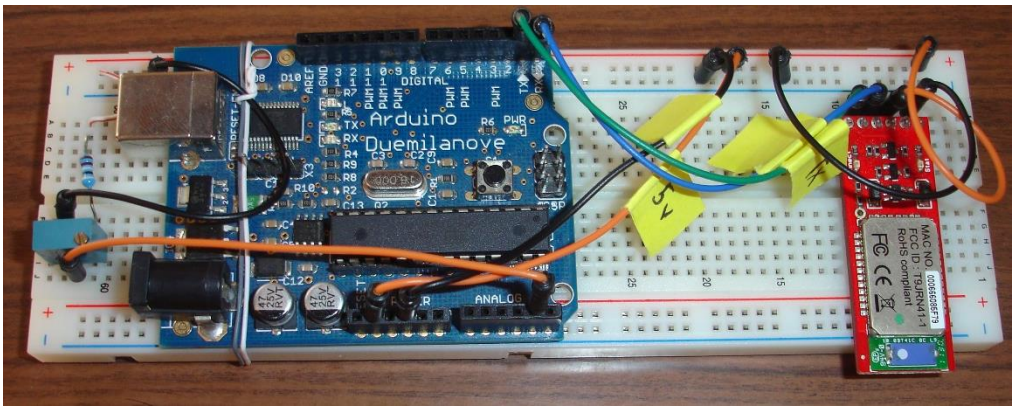

Figura 4.1. Montagem inicial entre o Arduino Duemilanove e o módulo Bluetooth.

Dado que o trabalho da dissertação foi integrado com um outro trabalho já desenvolvido anteriormente, no âmbito de uma dissertação sobre o desenvolvimento de uma bicicleta elétrica, foi necessário proceder à migração do trabalho implementado na plataforma Arduino Duemilanove para a plataforma LaunchXL-F28027 da Texas Instruments, pois a bicicleta elétrica já tinha esse módulo integrado.

4.2 Diagrama do sistema

Através da Figura 4.2 podemos visualizar mais detalhadamente como estão divididos os componentes utilizados e a forma como estes interagem. O bloco de aquisição, que integra um dos módulos LaunchXL-F28027 da Texas Instruments, utiliza dois canais do ADC, um para ler a tensão fornecida pelo torque e outro para calcular qual o nível da bateria que a bicicleta tem disponível. O GPIO está configurado para funcionar como contador de impulsos do sinal da cadência; esta contagem recorre à definição de uma interrupção sempre que nesse pino exista um impulso. Estes valores são posteriormente processados pela aplicação desenvolvida e instalada no microprocessador, com a finalidade de serem agregados numa trama de dados. Por sua vez o módulo LaunchXL-F28027 tem ligado um módulo Bluetooth à sua interface SCI, que procede à entrega da trama de dados, via Bluetooth, à aplicação desenvolvida especificamente para o efeito existente no smartphone Android. Esta aplicação executa um algoritmo de decisão para determinar qual o

valor a aplicar ao motor elétrico e responde via Bluetooth para o microcontrolador do sistema de monitorização do bloco de aquisição. De seguida é comunicado ao bloco de controlo, através da comunicação série síncrona SPI, o valor que deve ser aplicado ao motor elétrico.

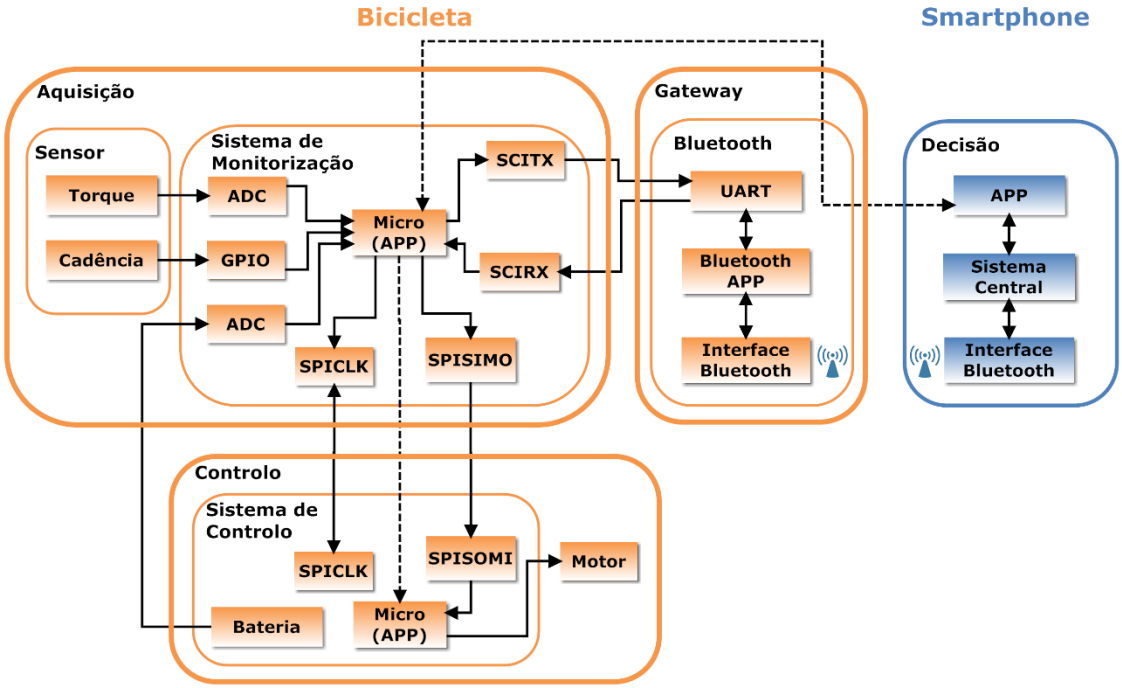

Figura 4.2. Diagrama de interação entre componentes.

Nas secções que se seguem são apresentados cada um destes componentes e procede-se à explicação técnica da forma específica como cada um realiza as suas tarefas.

4.3 Aquisição de dados sensoriais

Como é possível visualizar na Figura 4.3, o sensor X-Cell RT foi colocado inicialmente no rolamento interno de um módulo parcial de uma bicicleta numa bancada de teste.

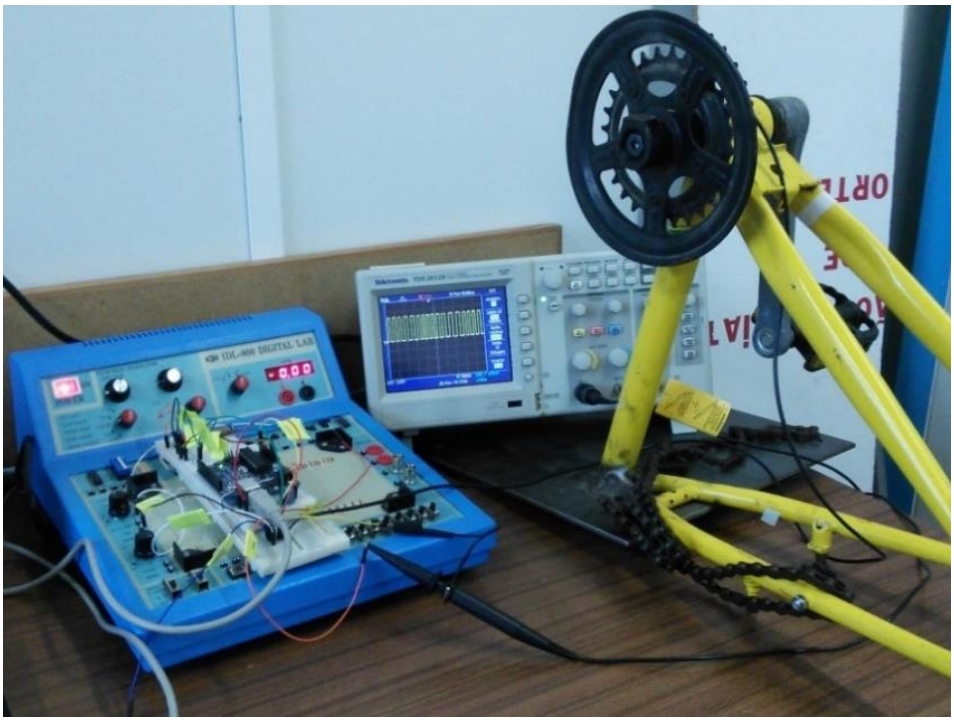

Figura 4.3. Bancada de teste com digital lab, osciloscópio e módulo parcial da bicicleta.

Após o estudo teórico do comportamento do sensor, e recorrendo a estes testes, foi possível estudar o comportamento físico real do sensor, a fim de o adaptar ao modo de funcionamento suportado pelo módulo Texas LaunchXL-F28027, como pode ser consultado nas secções 4.3.1 e 4.3.2.

Após serem realizados estes aspetos relativos ao estudo do comportamento físico do sensor X-Cell RT, foi realizada a sua integração na bicicleta elétrica, para confirmação do seu correto funcionamento num contexto real, como pode ser visualizado na Figura 4.4.

Figura 4.4. Bicicleta elétrica do sistema juntamente com o suporte traseiro.

Como também é possível verificar na Figura 4.4, existe um suporte traseiro para efeitos de teste laboratorial. Com recurso a este suporte é possível simular em laboratório o comportamento realizado pela bicicleta no exterior, dado que existe uma resistência regulável sobre a roda traseira que pode ser alterada para simulação de variações na inclinação do terreno. Desta forma, conseguimos exercer uma força de torque maior ou menor na pedaleira.

4.3.1 Torque

A aquisição do torque é realizada pelo canal ADCINA4 do módulo LaunchXL-F28027. Através deste, é possível converter a sua tensão de entrada num valor digital entre 0 e 4095. Inicialmente, em termos de análise comportamental do canal do ADC, foi testada uma variação de tensão à entrada recorrendo a um potenciómetro e sempre que existia uma alteração na sua resistência interna ocorria uma alteração do valor lido pelo ADC. Através deste método foi possível verificar a

variação comportamental por parte do ADC e projetar um cenário futuro onde se utilizaria o sensor real.

Posteriormente foi integrado o sensor real, e teve de ser considerada a hipótese de este fornecer tensões entre 0 V e 5 V, como foi analisado na Figura 3.5. Procedeu-se assim a um ajuste de tensão máxima do sinal de torque para 3.3 V, como é visível na Figura 4.5, através de um divisor resistivo, de forma a limitar o valor máximo suportado pelo canal do ADC.

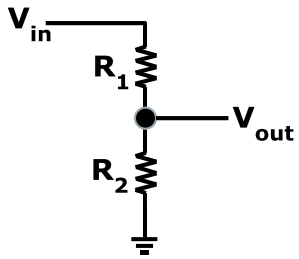

Figura 4.5. Divisor resistivo para o sinal de torque.

Dado que, pelas características de funcionamento do sensor, a tensão de entrada (V_{in}) máxima é de 5 V e a tensão de saída (V_{out}) pretendida é de 3.3 V, e tendo em consideração as propriedades do divisor resistivo, foi obtida a fórmula:

$$R_1 = 0.5152 \times R_2 \qquad (4.1)$$

Atribuindo a R_2 o valor de 22 kΩ, obtemos para R_1 o valor correspondente de 11.3 kΩ, conseguindo assim um V_{out} máximo de 3.3 V e respeitando o princípio de funcionamento do ADC do microcontrolador.

Quando foi testado o sensor na bancada de teste, este estava a ser alimentado por um Digital Lab IDL-800, o que lhe conferia uma alimentação sem ruídos e isolada de qualquer outro circuito externo (o sinal de saída do torque era uma tensão constante e sem interferências ao longo do tempo). Por outro lado, a bicicleta elétrica já possuía uma bateria de alimentação e um circuito de controlo do motor elétrico. Como podemos ver na Figura 4.6, aquando da ligação do sensor na bicicleta elétrica, foi detetado que o sinal de torque à saída do sensor estava afetado por um

ruído de alta frequência que provocava um comportamento altamente prejudicial à leitura do valor real do torque exercido pelo ciclista na pedaleira.

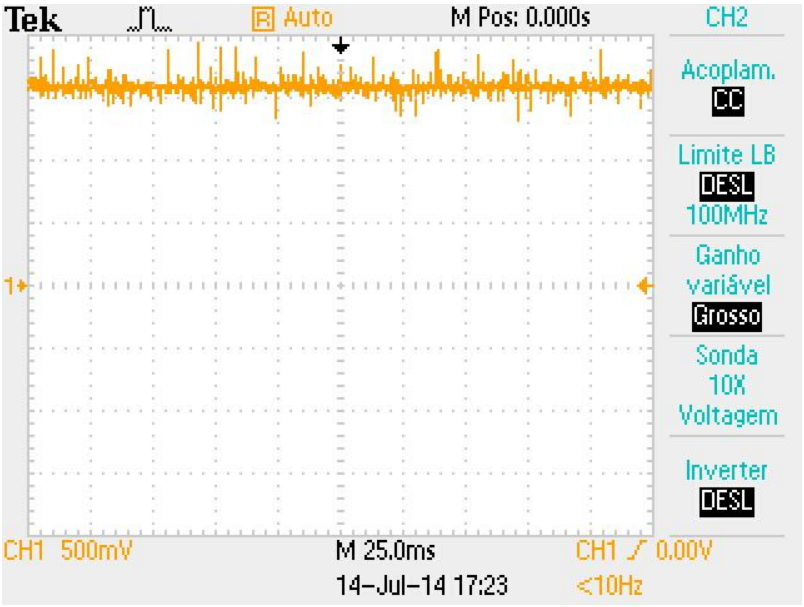

Figura 4.6. Ruído no sinal do torque em repouso.

Para atenuar este ruído de alta frequência e ser possível um funcionamento controlado por parte do sensor, pensou-se inicialmente em colocar um filtro ativo (com amplificador operacional) para que através da sua impedância de entrada infinita proporcionasse que o divisor resistivo e o filtro passa baixo ficassem isolados e, assim, não fosse alterada a frequência de corte dimensionada. Isso não foi possível dado que não existem disponíveis na bicicleta voltagens de -5 V ou -12 V para alimentação negativa do amplificador operacional. Assim, decidiu-se colocar um filtro passa baixo de 1ª ordem com uma resistência (R) muito mais alta (no mínimo 10 vezes) que a resistência equivalente do paralelo de R_1 e R_2, conforme mostra a Figura 4.7, à saída do sinal de torque proveniente do divisor resistivo, fazendo assim com que fosse respeitada, de forma aproximada, a frequência de corte pretendida.

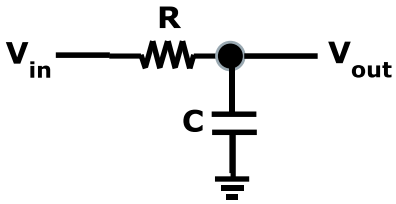

Figura 4.7. Filtro passa baixo de 1ª ordem.

Com recurso a este tipo de filtro é possível definir uma frequência de corte para que sinais/ruídos com frequências acima de um determinado valor sejam filtrados. A frequência de corte deste tipo de filtro é dada pela seguinte fórmula:

$$fc = \frac{1}{2\pi \text{ x } R \text{ x } C} \qquad (4.2)$$

Foi definida uma frequência de corte (f_c) de 30 Hz (dado ser mais que suficiente para filtrar frequências altas e garantir a integridade total do sinal de torque) e estabeleceu-se um valor do condensador (C) de 10 nF. Realizados os cálculos encontrou-se uma resistência (R) de 530 kΩ. Como podemos ver na Figura 4.8**,** obteve-se um sinal de torque cuja componente de alta frequência foi atenuada por forma a ser possível recolher um valor de torque que corresponde à realidade de execução.

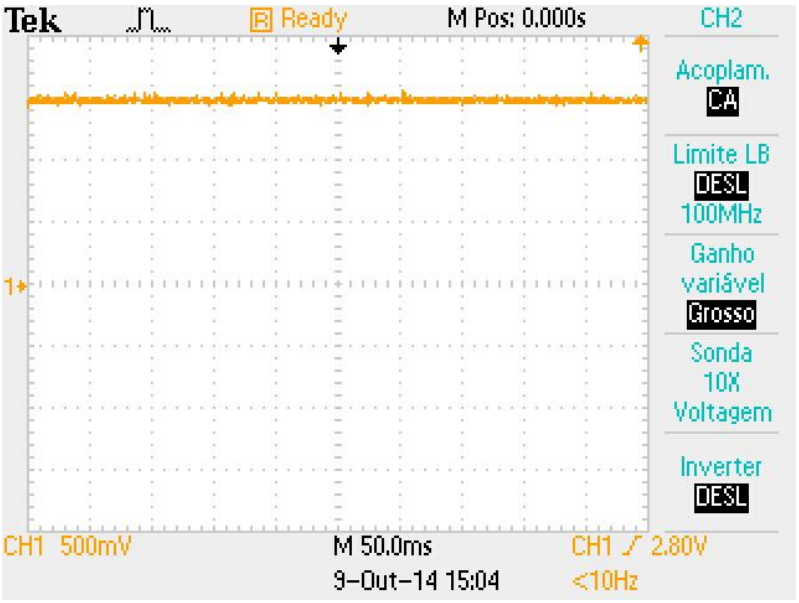

Figura 4.8. Sinal de torque em repouso com filtro passa baixo.

Com este ajuste da tensão máxima e após a filtragem da componente de alta frequência introduzida pela alimentação do circuito de controlo, foi possível ligar a saída do filtro passa baixo de 1ª ordem diretamente ao canal ADCINA4 e ler valores corretos do torque.

Para que o microcontrolador converta o sinal lido do ADC referente ao torque num valor em Nm, é necessário proceder à definição de uma fórmula de conversão. A Figura 4.9 permite visualizar a conversão do valor lido pelo ADC do microcontrolador para o valor da tensão correspondente e posteriormente desse valor de tensão para o valor do torque em Nm.

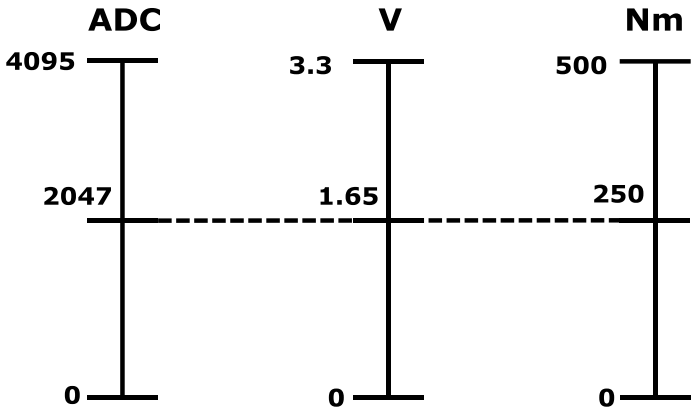

Figura 4.9. Relação entre as diferentes unidades do valor do torque.

A conversão do valor do ADC para tensão (V_1) é conseguida através da fórmula:

$$V_1 = \frac{ADC \text{ x } 3.3}{4095} \tag{4.3}$$

Por outro lado a conversão de tensão para o torque (τ) em Nm é realizada com base na fórmula:

$$\tau = \frac{V_1 \text{ x } 500}{3.3} \tag{4.4}$$

Procedendo à substituição de variáveis obtemos a fórmula geral final:

$$\tau = 0.122 \text{ x } ADC \tag{4.5}$$

Como forma de exemplificação do comportamento do sinal de torque, na Figura 4.10 é visível o sinal de torque para uma situação em que se simulou um pedalar normal num terreno sem inclinação.

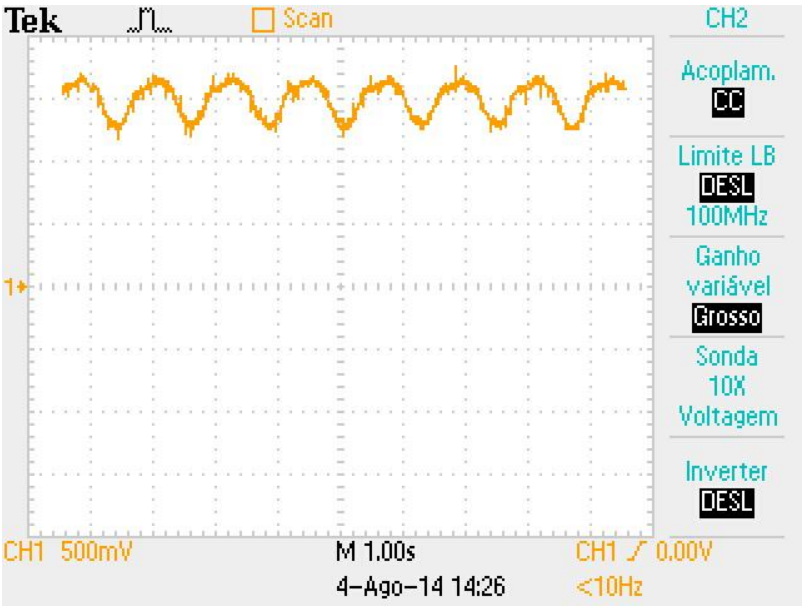

Figura 4.10. Sinal do torque (com resistência e cadência).

Verificando a Figura 4.10**,** constatamos que o comportamento do sinal de torque assemelha-se a uma sinusoide. Este comportamento terá de ser levado em consideração aquando da recolha dos dados e respetiva decisão no algoritmo de controlo de esforço, como veremos na secção 4.8.2**.**

4.3.2 Cadência

Tal como no sinal do torque, o sinal de cadência foi alvo de ajuste da tensão máxima para 3.3 V, dado que a tensão do sensor atinge um valor máximo de 5 V. O procedimento realizado para a cadência foi o mesmo que para o sinal de torque, visível na Figura 4.5.

O comportamento do sinal fornecido por este sensor já foi introduzido, em termos teóricos, na secção 3.2, com ajuda da Figura 3.5. Em termos práticos, podemos visualizar o funcionamento do sensor com auxílio de um osciloscópio, através da Figura 4.11.

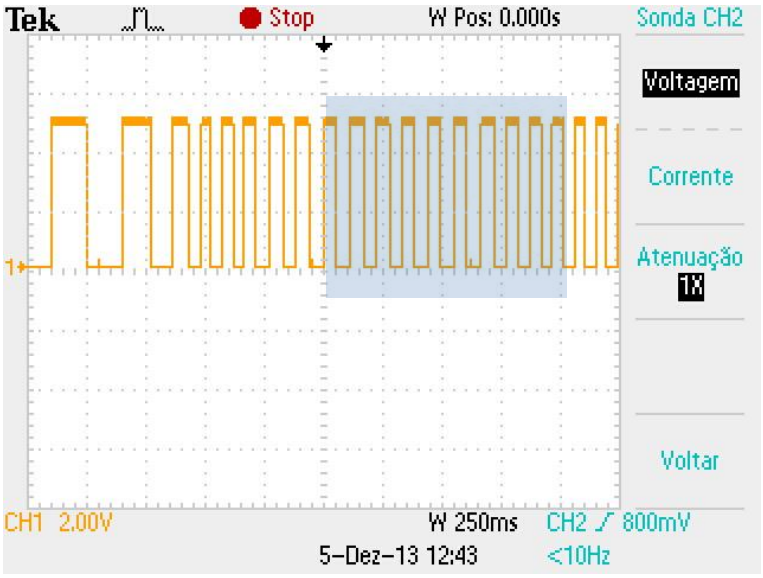

Figura 4.11. Sinal de cadência.

Como se trata de um sinal digital, este tem de ser captado pelo microcontrolador através de um pino de I/O digital. Neste caso em concreto usou-se o pino GPIO12, ao qual foi associada uma rotina de interrupção para que, cada vez que ocorra uma transição positiva no sinal de cadência, esta interrupção seja acionada e assim sejam contabilizados quantos impulsos positivos ocorreram. Como exemplo, na parte sombreada do sinal da Figura 4.11, temos 10 impulsos positivos a serem contabilizados pela rotina de interrupção do microcontrolador durante o período de 1 segundo. Após a contabilização do número de impulsos ($N_{impulsos}$) é necessário proceder ao cálculo com a seguinte fórmula, para realizar a conversão para rotações por minuto (rpm):

$$N_{rpm} = \frac{N_{impulsos} \text{ x } 60}{8 \text{ x } T} \qquad (4.6)$$

De realçar que o cálculo do número de rotações por minuto (N_{rpm}) é obtido com base no número de impulsos contados pelo GPIO12 multiplicado por 60 segundos e dividido pela multiplicação entre 8 e o período (T) durante o qual foi contabilizado o número de pulsos.

Este procedimento foi usado inicialmente quando o período de amostragem da cadência utilizado era superior a 1 segundo. Com o aprimoramento do projeto, foi necessário utilizar períodos de amostragem muito inferiores a 1 segundo. Temos o

exemplo da utilização para períodos de 200 ms como é visível na secção 5.1. Assim, facilmente se percebe que nestes casos a solução acima descrita não é adequada sobre o ponto de vista de resolução da leitura da cadência.

Para resolução deste problema, procedeu-se à substituição da contagem de impulsos durante o período de amostragem pela contabilização do tempo que decorre entre cada impulso de cadência. Com este procedimento, a interrupção originada por uma transição positiva no pino GPIO12 inicia uma contagem de tempo (*t0*), recorrendo a um temporizador do microcontrolador, que se prolonga até existir nova interrupção por transição positiva (*t1*). Para melhor entendermos este procedimento, vejamos a Figura 4.12.

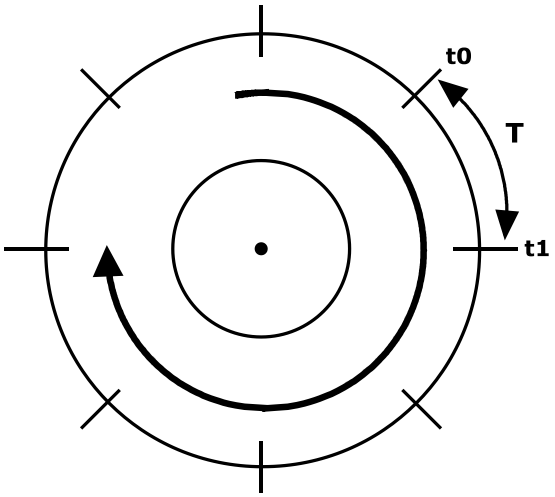

Figura 4.12. Determinação da cadência com recurso ao tempo entre impulsos.

Após a contabilização do tempo de início e tempo de fim, procede-se à subtração entre o tempo final e o tempo inicial de forma a calcular qual o tempo (*T*) gasto entre os dois impulsos. No final, converte-se o valor de *T* segundos para rpm através da fórmula:

$$N_{rpm} = \frac{60}{8 \text{ x } T} \qquad (4.7)$$

Desta forma confere-se maior resolução à leitura da cadência efetuada pelo ciclista, como podemos constatar na secção 5.1.

4.3.3 Esforço

No smartphone temos implementado o algoritmo de controlo. Este controlo pode ser direcionado para o esforço ou para a resistência a que o ciclista está a ser sujeito e tem como objetivo indicar ao sistema qual o nível de ajuda a ser aplicado ao motor. Para que esta resposta seja fornecida, é necessário proceder ao cálculo da potência mecânica (esforço) ou resistência (torque) que o ciclista está a realizar num momento específico. Consegue-se obter esta informação recorrendo aos dados dos sensores que são recolhidos em tempo real pelo bloco de aquisição. O torque e a cadência são sinais que podem ser utilizados de forma independente ou serem conjugados por forma a traduzirem um terceiro sinal (potência). Neste caso específico, podemos obter o esforço do ciclista em Watts através da seguinte fórmula:

$$P = \frac{2\pi \text{ x } C \text{ x } \tau}{60} \tag{4.8}$$

De uma forma sucinta, a potência em Watts (*P*) corresponde à multiplicação do torque (τ) pela cadência em rad/s ($\frac{2\pi}{60}$ x C), sendo *C* a cadência em rpm.

Para que este valor de esforço possa ser utilizado coerentemente pelo algoritmo de controlo de esforço, foi necessário aplicar um mecanismo para suavizar as variações abruptas de torque (ver secção 5.2) detetadas pelo sistema para não ser precipitadamente aplicado um nível de ajuda do motor que não se coadune com o exigido pelo ciclista. Inicialmente, equacionou-se a utilização de uma média móvel simples para que a leitura da última amostra de esforço fosse influenciada por um número de amostras anteriores, de modo a suavizar as transições abruptas de esforço. Rapidamente se percebeu que esta solução não seria a mais acertada, pois dependentemente do tamanho da janela para o cálculo da média, esta terá de ser armazenada num *buffer*, o qual requer utilização de memória interna do sistema. Tratando-se de um sistema em tempo real, ainda mais problemático é o facto de esta solução atribuir igual peso a todos os valores contidos na janela definida e, por outro lado, ir também excluindo os valores mais antigos.

Para contornar estas limitações, estabeleceu-se como solução a utilização da técnica denominada *exponential smoothing*. Esta técnica tem a particularidade de atribuir um maior peso aos últimos valores utilizados (com os valores passados a diminuírem exponencialmente de importância ao longo do tempo) e integrar todo o passado recolhido no cálculo do valor atual, armazenando-o num único valor. Esta técnica utiliza as seguintes fórmulas:

$$S_k = X_k,\ k = 0 \tag{4.9}$$

$$S_k = \alpha X_k + (1-\alpha) S_{k-1},\ k > 0 \tag{4.10}$$

O valor do esforço com recurso ao *exponential smoothing* (S_k) utiliza um fator ($0 \leq \alpha \leq 1$) que determina qual o peso a atribuir à amostra atual (X_k) em desfavorecimento do valor acumulado anteriormente calculado (S_{k-1}). Tem como domínio (k) o conjunto dos números naturais, sendo que para $k = 0$ este toma somente o valor da primeira amostra (X_0).

4.4 Módulo Bluetooth

Através dos pinos do sistema de monitorização do bloco de aquisição referentes à interface de comunicação série SCI é possível estabelecer uma comunicação sem fios, desde que se efetue a ligação física com a interface UART do módulo Bluetooth WRL-12580. Observando a Figura 4.13, podemos conhecer o módulo e a constituição da sua interface UART.

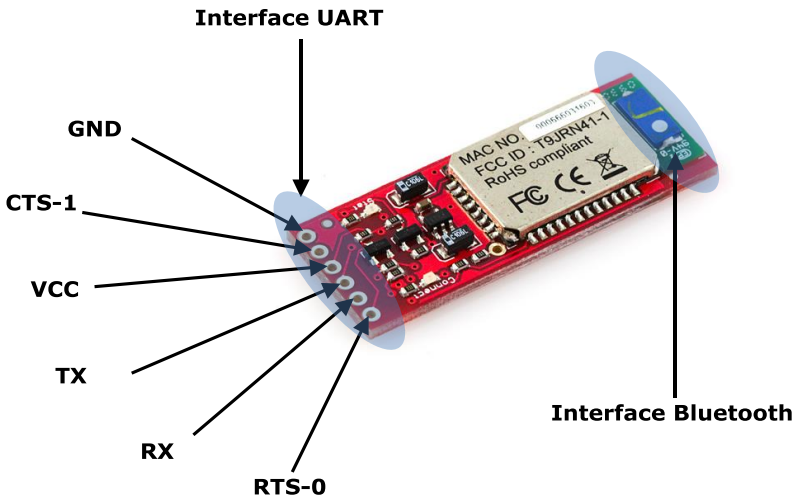

Figura 4.13. Interface UART do módulo Bluetooth.

A interface UART do módulo Bluetooth contém dois pinos responsáveis pela alimentação (GND e VCC), que operam sob os valores especificados na secção 3.2.2, dois pinos responsáveis por tarefas de controlo de fluxo (CTS-1 e RTS-0) e dois pinos que garantem a ligação física à porta de comunicação série SCI do microcontrolador do sistema de monitorização (TX e RX) para transmissão de dados. Ligou-se o pino TX do módulo Bluetooth ao pino RX da interface SCI (GPIO28) e o pino RX ao pino TX (GPIO29) da mesma interface SCI. Assim, conseguiu-se fazer com que os dados que sejam enviados pelo microcontrolador para a interface SCI sejam entregues ao módulo Bluetooth WRL-12580 (para que este, por sua vez, os envie via interface rádio Bluetooth), e vice-versa.

No que diz respeito à interface Bluetooth, esta não tem de ser tida em consideração para estabelecimento de ligações físicas, dado que a sua configuração é essencialmente realizada por intermédio do acesso em modo de comando. Completando a explicação da secção 3.2.2, quando é necessário aceder a este modo de comando, basta enviarmos para o módulo Bluetooth a sequência de caracteres "$$$", o que origina uma resposta de confirmação de acesso a este modo de operação (CMD). Posteriormente é necessário escrever o comando a utilizar para proceder a uma determinada alteração pretendida na configuração interna, e após a sua correta introdução o módulo Bluetooth envia uma confirmação (AOK) de aceitação do comando. Por fim, para salvar as alterações realizadas na configuração

interna e sair do modo de comando envia-se "- - -". A Tabela 4.1 contém alguma das configurações utilizadas na configuração do módulo Bluetooth.

Tabela 4.1. Configuração módulo Bluetooth.

Comando	Descrição
$$$	Ativar modo de comando
D	Consultar configurações
SF, 1	Repor configurações de fábrica
SN, <string>	Nome dispositivo
SP, <string>	Código de acesso
SU, <value>	Baud Rate
SL, <E,O,N>	Paridade (E = Par, O = Impar, N = Desativada)
SM, <0,1>	Modo (0 = Escravo, 1 = Mestre)
---	Guardar e Sair

Utilizando a Tabela 4.1, podemos realizar uma consulta das configurações do módulo através do comando D. Como podemos ver na Figura 4.14, para este caso específico foi definido Mobi-eBike como nome do dispositivo Bluetooth, um *baud rate* de 115 kbps, paridade desativada, "1234" como código de acesso e funcionamento em modo escravo.

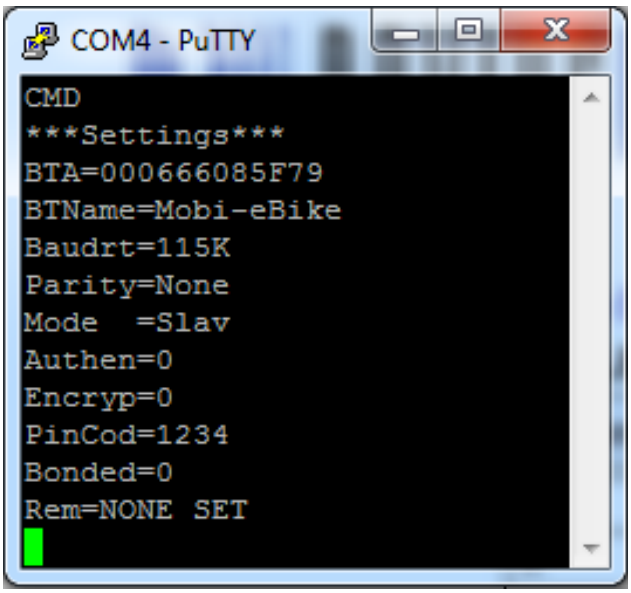

Figura 4.14. Configuração módulo Bluetooth.

4.5 Microcontroladores

4.5.1 Hardware

Como já foi referido na secção 3.2, e como é possível ver na Figura 4.15, foram usados dois módulos LaunchXL-F28027 da Texas Instruments, um para o bloco de aquisição e outro para o bloco de controlo do motor elétrico. Em ambos os casos foram usados reguladores lineares LD1117v33 para conversão do valor da tensão fornecida pela bateria da bicicleta para 3.3 V.

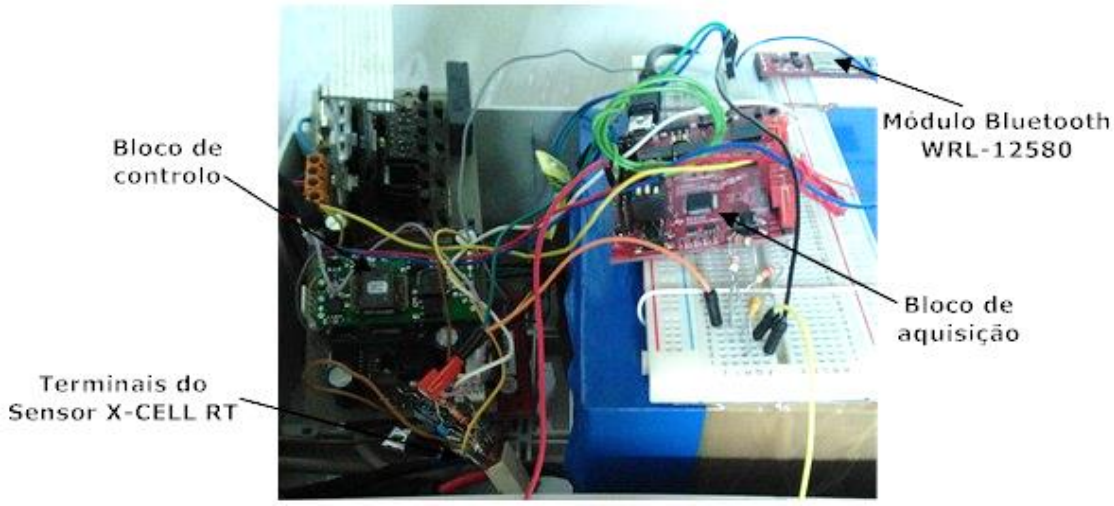

Figura 4.15. Hardware do sistema na bicicleta.

No que diz respeito ao bloco de aquisição, este é responsável pela aquisição dos valores mensurados pelos sensores integrados na bicicleta, onde foram usados dois canais do ADC, para aquisição do valor do torque (ADCINA4) e nível de bateria da bicicleta (ADCINA1). Para a cadência, foi utilizada forçosamente outra abordagem, dado tratar-se de um sinal digital, recorreu-se a uma rotina de interrupção externa disponibilizada pelo microcontrolador através do pino GPIO12, que está configurado para ocorrer sempre que exista uma transição positiva (*PIE_ExtIntPolarity_RisingEdge*) no sinal da cadência.

Foi também utilizado o *timer* 0 para realizar o controlo da interrupção sempre que seja excedido o período definido para o intervalo de tempo entre o envio de

cada trama. Para a disponibilização de interface Bluetooth com o smartphone foi utilizado o módulo Bluetooth WRL-12580 onde a sua interface UART liga aos pinos da interface de comunicação em série SCI do microcontrolador disponibilizado pelos pinos GPIO28 (SCIRX) e GPIO29 (SCITX).

O bloco de controlo é responsável por analisar e controlar a corrente que está a ser usada para alimentar o motor e caso esta seja superior a 7 A é prontamente desligada a alimentação ao motor para que nada seja danificado. No caso de ser acionado o travão, a ajuda do motor é igualmente terminada. Estas variantes de controlo já estavam implementadas na bicicleta elétrica no âmbito da dissertação anterior [21]. Para este caso em concreto a alteração corresponde à forma como é acionado o nível de ajuda do motor, ou seja, anteriormente este era acionado por dois botões físicos cujas funções eram de aumento ou diminuição do nível de ajuda do motor, e neste caso foi necessário fazer com que a instrução chegasse ao microcontrolador da bicicleta por intermédio da comunicação rádio Bluetooth do smartphone.

Inicialmente o sistema de monitorização e o sistema de controlo foram testados no mesmo microcontrolador, contudo esta abordagem foi prontamente descartada, dado que o sistema de controlo já existente requer um tempo de processamento totalmente dedicado ao controlo do motor elétrico, pelo que as interrupções necessárias ao sistema de monitorização referentes à contagem do tempo entre pulsos de cadência, do *timer* 0 para contagem do intervalo entre envio de tramas e da comunicação com recurso ao protocolo SCI inviabilizam essa integração conjunta num único microcontrolador. Desta forma separaram-se os sistemas, estando o bloco de aquisição configurado para comunicar com o smartphone e, aquando da receção do nível de ajuda do motor, comunicá-la ao bloco de controlo via comunicação série síncrona SPI. Para configuração física desta comunicação, recorreu-se ao pino GPIO18 (modo SPICLK) para sincronização da comunicação entre ambos os sistemas, e aos pinos GPIO16 (modo Mestre - SPISIMO) do sistema de monitorização e GPIO17 (modo Escravo - SPISOMI) do sistema de controlo para configuração do canal de comunicação entre ambos.

A Figura 4.16 apresenta o esquema das configurações físicas acima referidas, em conjunto com a especificação da secção 4.4 referente ao módulo Bluetooth WRL-12580, com a secção 2.2 relativa ao sensor X-Cell RT e com recurso ao *datasheet* do módulo Texas LaunchXL-F28027 [25] e dos reguladores lineares da série LD1117 [28].

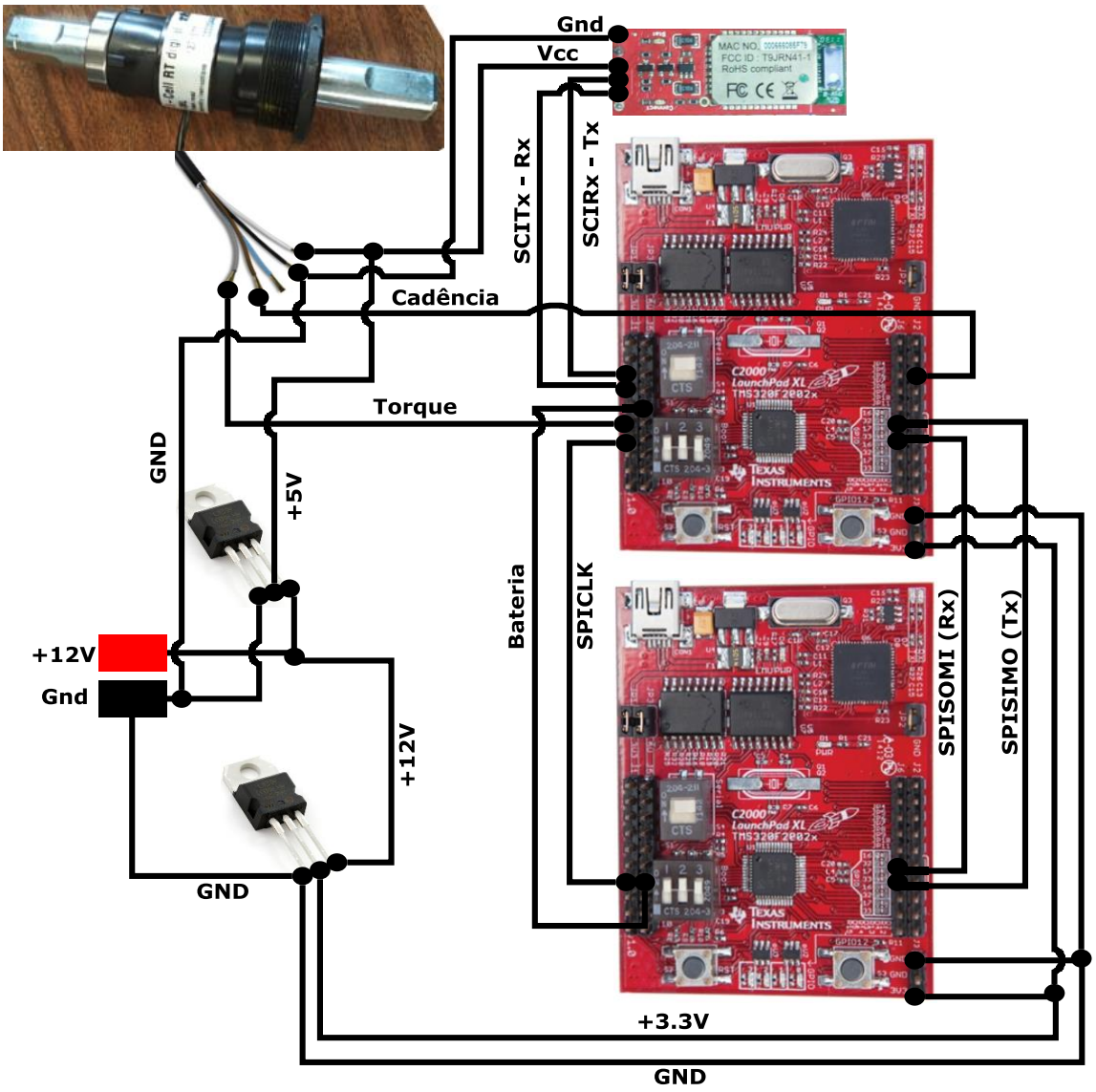

Figura 4.16. Esquema geral do sistema na bicicleta.

Relativamente ao bloco de controlo, este inicialmente apenas disponibilizava 3 níveis de ajuda; porém, cedo se percebeu que seriam necessários mais níveis, pelo que se realizou uma adaptação para 10 níveis de ajuda em que 0 corresponde ao nível de ajuda mais baixo e 10 ao maior. Contudo, esta adaptação acarretou consequência ao nível de aquecimento do equipamento pelo que foi colocado um

dissipador de calor nos MOSFETs existentes no bloco de controlo, como é visível na Figura 4.17.

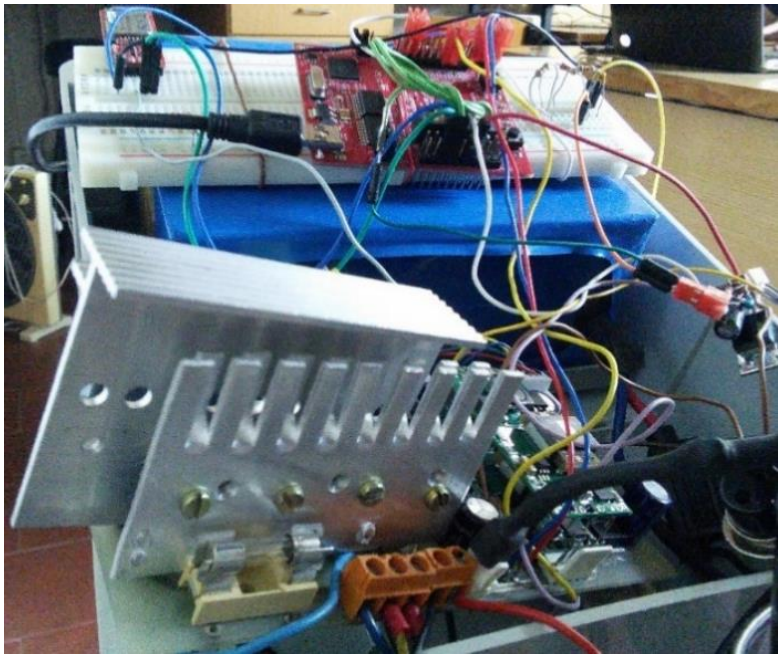

Figura 4.17. Hardware com o dissipador de calor.

4.5.2 Software

A configuração física dos componentes de nada faz sentido sem uma programação coerente e eficiente. A linguagem de programação que o microcontrolador do módulo LaunchXL-F28027 executa é a linguagem C, e como forma de cortesia existe um pacote disponibilizado pela própria Texas Instruments, chamado ControlSUITE, que contém exemplos de código básico para utilização livre. Após definição de tarefas e objetivos a atingir, foi desenhada parte da solução baseada no fluxograma da Figura 4.18. De realçar que inicialmente o microcontrolador habilita a comunicação assíncrona via interface série SCI (que por sua vez tem ligado o módulo Bluetooth) e após o smartphone estabelecer ligação com o módulo Bluetooth ligado à sua interface SCI, o microcontrolador está configurado para autenticar a aplicação do smartphone com recurso a uma senha, ficando o microcontrolador à escuta de uma combinação que seja igual à que tem definida em *hard code*. Após a receção correta da senha de autenticação, o

microcontrolador aguarda por receber do smartphone o modo de funcionamento pretendido pelo utilizador. O microcontrolador tem como opção de escolha o modo automático e o modo manual. Na secção 4.6 está explicado detalhadamente o funcionamento de cada modo.

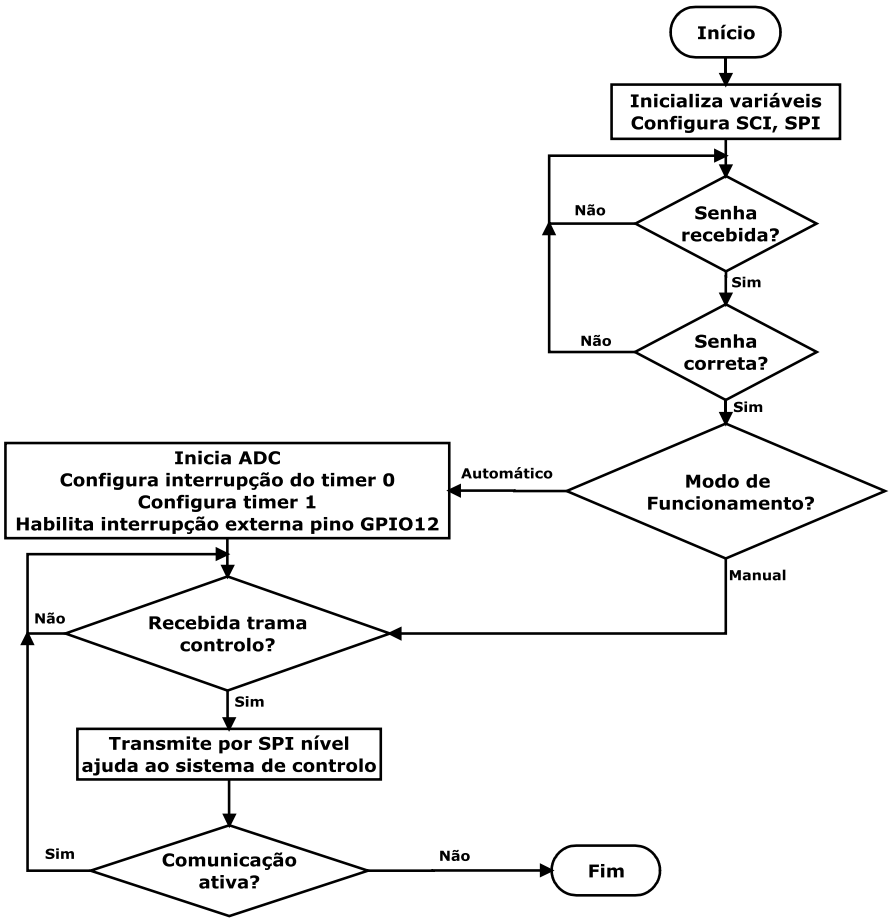

Figura 4.18. Fluxograma de inicialização do sistema, escolha de modo de funcionamento e receção de controlo.

Se a escolha recair para o modo automático, são habilitados o canal ADCINA1 para leitura do nível de bateria e o canal ADCINA4 para aquisição do valor do torque. É configurada uma interrupção externa relativa ao GPIO12 (que configura também o *timer* 1), que será acionada sempre que existir uma transição positiva nesse pino e uma outra interrupção despoletada pelo *timer* 0. Após estas configurações, o microcontrolador espera pela receção de uma trama de controlo, e assim que esta chega, envia-a prontamente ao bloco de controlo via comunicação síncrona SPI. O teste de verificação se a comunicação está ativa serve essencialmente para terminar

o encadeamento do processamento e iniciar um novo sempre que é terminada a ligação Bluetooth com o smartphone.

O modo manual não requer esta configuração de ADC, *timers* e interrupções, dado não requerer um tratamento de controlo de esforço pois é o utilizador que define qual o nível de ajuda que quer aplicar ao motor. Este modo está igualmente detalhado na secção 4.6**.**

A Figura 4.19 apresenta o fluxograma que demonstra como se processa a rotina de interrupção externa configurada para o GPIO12. De realçar que a variável *InicioPulso* está definida globalmente a 1.

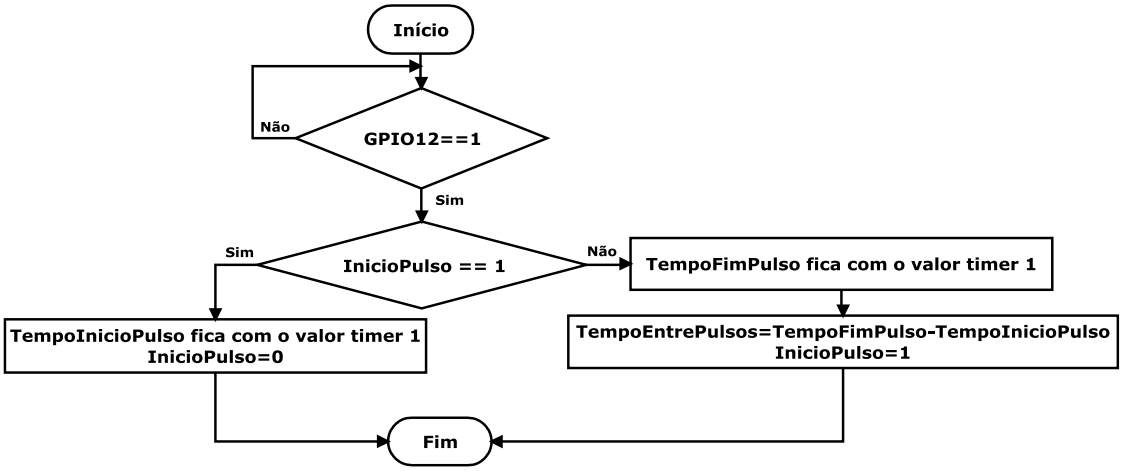

Figura 4.19. Fluxograma de interrupção externa do pino GPIO12.

Como podemos verificar, se uma transição positiva for detetada no GPIO12, a rotina de interrupção acionada confere se realmente ocorreu essa transição, e em caso positivo marca esse ponto como sendo o início do pulso e guarda na variável *TempoInicioPulso* o valor do *timer* 1, passando para 0 o valor de *InicioPulso.* Quando houver nova interrupção, será guardado na variável *TempoFimPulso* o valor do *timer* 1 e realizada a diferença entre estes tempos para ser possível calcular qual a velocidade com que o ciclista está a pedalar, conforme retrata a Figura 4.20.

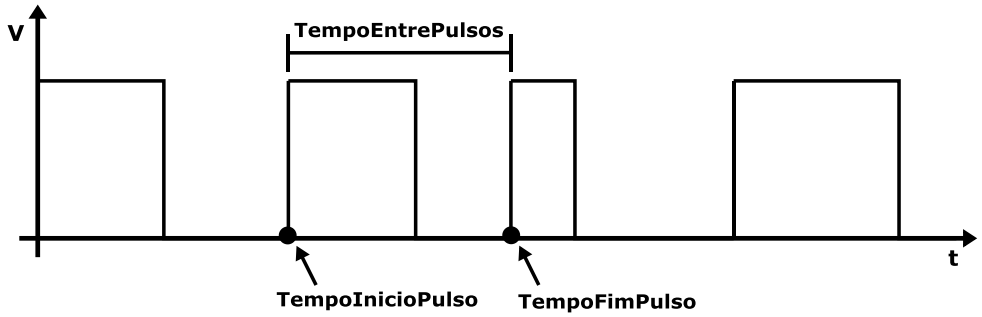

Figura 4.20. Tempo entre pulsos de cadência.

Consultando a Figura 4.21 podemos analisar a rotina anteriormente referida de atendimento à interrupção acionada pelo *timer* 0. Esta regula a periodicidade com que são enviadas as tramas e tem como tarefas guardar o último valor calculado do tempo entre os pulsos de cadência, recolher no momento o valor do canal ADCINA1 referente à bateria e o valor do canal ADCINA4 do valor de torque. Recolhidos estes dados dos sensores, constrói a trama de dados (especificada na secção 4.6) e envia-a para o smartphone através da interface de comunicação série assíncrona SCI.

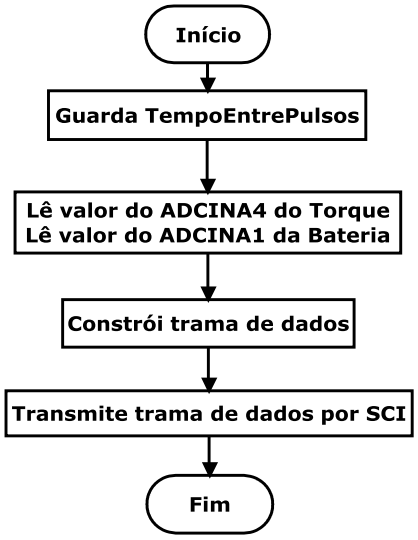

Figura 4.21. Fluxograma da rotina de interrupção do *timer* 0.

4.6 Protocolo de comunicação

Para explicação da forma como é processada a comunicação do sistema como um todo, é necessário desdobrá-lo em componentes independentes e identificar

quais as mensagens trocadas entre cada parte. Na Figura 4.22 podemos identificar cada um dos componentes, bem como o fluxo da troca de informação e respetivo conteúdo de cada uma.

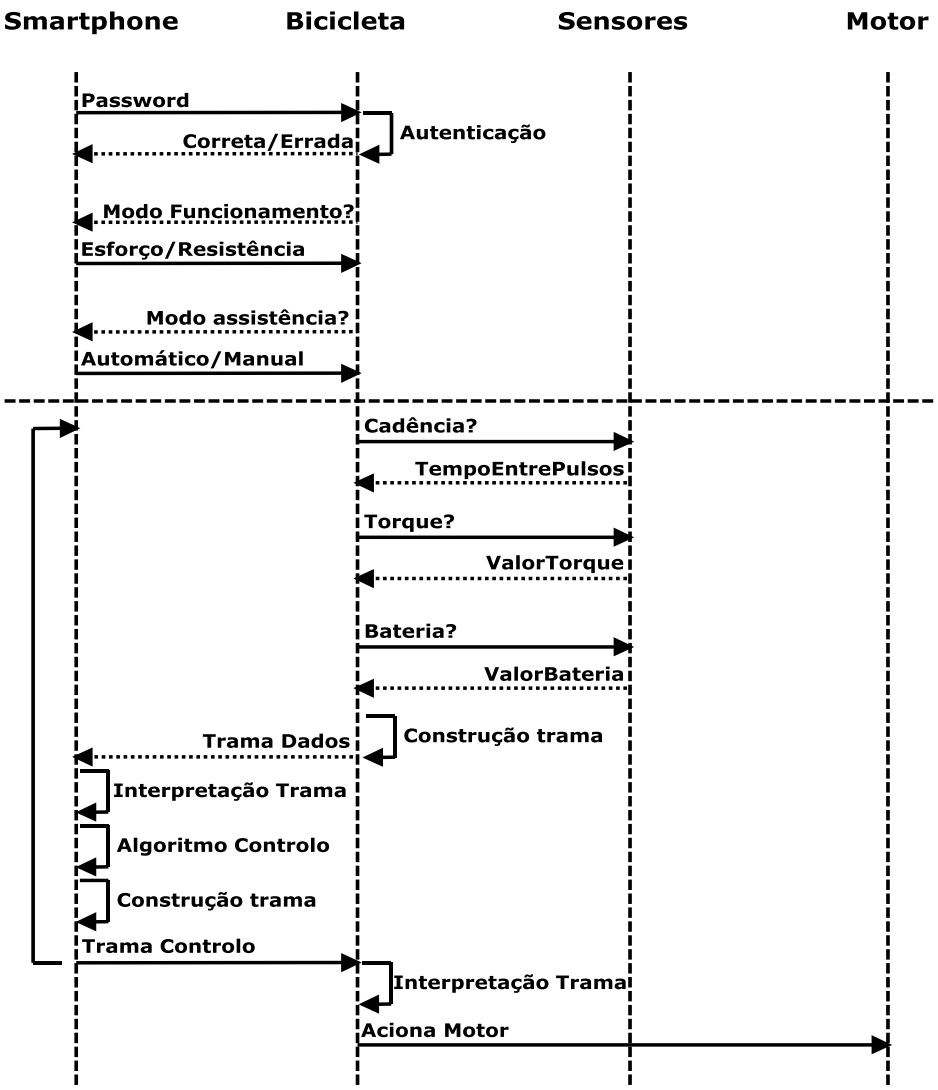

Figura 4.22. Diagrama do fluxo da informação.

Inicialmente é realizada a autenticação da aplicação do smartphone e de seguida é definido qual o modo de funcionamento que queremos utilizar na bicicleta. No caso de escolhermos o modo resistência, somente o valor do torque é tido em consideração no algoritmo de controlo para decisão do nível de ajuda a aplicar no motor, no caso de escolhermos o modo esforço, o algoritmo de controlo irá considerar como fator de decisão de escolha do nível de ajuda tanto a cadência calculada como o torque. Posteriormente, e para que fossemos ao encontro do

trabalho já realizado na anterior dissertação relativa à bicicleta, é permitido escolher qual o modo de assistência que pretendemos utilizar, ou seja, o modo manual, que consistia na única forma como a bicicleta era usada (embora agora os botões de aumento e diminuição da ajuda sejam implementados recorrendo ao smartphone), ou em modo automático, que no fundo é a principal transição da anterior versão da bicicleta e que lhe confere um controlo de forma automática.

Posteriormente a esta definição inicial de utilização, segue-se o fluxo da informação quando é escolhido o modo automático, que é realizado sistematicamente até existir um término da ligação entre o smartphone e a bicicleta. O microcontrolador do bloco de aquisição está encarregue de calcular e registar o tempo entre pulsos, o valor do torque e o nível de bateria da bicicleta e depois de construir a trama de dados envia-a prontamente para a aplicação do smartphone. Este, por sua vez, tem como função interpretar a trama de dados recebida, e após calcular o valor da cadência da bicicleta em rad/s, o valor do torque em Nm e recolher o valor da bateria, executa um algoritmo de controlo que retorna o valor do nível de ajuda a aplicar pelo motor da bicicleta. De seguida, a aplicação do smartphone constrói a trama de controlo e envia-a ao bloco de aquisição da bicicleta, onde este interpreta o valor do nível de ajuda do motor e comunica-o ao sistema de controlo para que seja aplicado ao motor elétrico.

Existem dois tipos de tramas trocadas entre a bicicleta e o smartphone. De uma forma sucinta, a bicicleta envia uma trama de dados para o smartphone e este por sua vez responde-lhe com uma trama de controlo. A Figura 4.23 apresenta como é constituída a trama de dados enviada pela bicicleta elétrica.

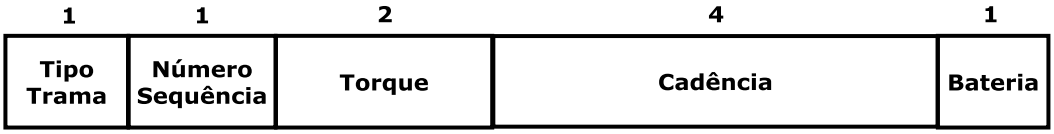

Figura 4.23. Constituição da trama de dados.

A trama de dados tem um tamanho de 9 bytes, sendo constituída pelos seguintes campos:

- **Tipo Trama (1 byte)**: serve essencialmente para identificar que tipo de trama se trata. Para estas tramas de dados o identificador escolhido foi o caracter '&'. Útil para o caso de querermos introduzir novas tramas com constituição de campos diferentes, como foi o caso de envio de tramas com atraso fim-a-fim (tratado na secção 5.4);
- **Número Sequência (1 byte)**: usado essencialmente para identificar qual o número da trama que estamos a receber para realização de despiste quanto à perda de pacotes;
- **Torque (2 bytes)**: pelo facto do ADC do microcontrolador usado ser de 12 bits, foram definidos 2 bytes para acomodar o valor;
- **Cadência (4 bytes)**: contém o valor do tempo entre pulsos conforme especificado na Figura 4.19. Possui 4 bytes por forma a conseguir representar qualquer valor de tempo, já que o registo do *timer* 1 é de 32 bits;
- **Bateria (1 byte)**: disponibiliza o nível da bateria da bicicleta, já calculado pelo microcontrolador em percentagem (valor entre 0 e 100).

A trama de controlo que chega ao microcontrolador vinda do smartphone requer menos bytes para acomodar a informação, conforme podemos constatar na Figura 4.24.

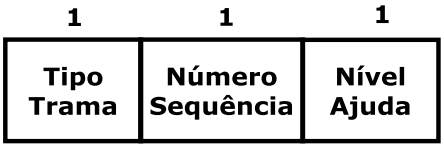

Figura 4.24. Constituição da trama de controlo.

Para este tipo de trama, temos apenas 3 campos:

- **Tipo Trama (1 byte)**: como forma de identificação do tipo de trama, este campo foi preenchido, tal e qual como na trama de dados, com o valor '&'. Possibilita assim a inclusão de outras tramas com constituições diferentes desta;

- **Número Sequência (1 byte)**: tem como função a identificação do número da trama de controlo e possibilita detetar se alguma trama foi perdida;
- **Nível Ajuda (1 byte)**: este campo contém o valor fornecido pelo algoritmo de controlo de esforço a aplicar no motor da bicicleta elétrica.

4.7 Atraso fim-a-fim

O atraso fim-a-fim do sistema foi calculado com base na forma como é obtido o round-trip time (RTT), ou seja, é contabilizado o tempo necessário desde o envio da trama com os dados dos sensores por parte da bicicleta para o smartphone até que seja recebida uma resposta a essa mesma trama. Juntamente a este tempo é acrescentado o tempo de processamento de casa módulo (T_{proc}), o tempo de transmissão (T_t), o tempo de propagação inerente a cada um dos sistemas (T_{prop}) e o tempo de acesso ao meio do protocolo Bluetooth (T_{acesso}), como é possível ver na Figura 4.25.

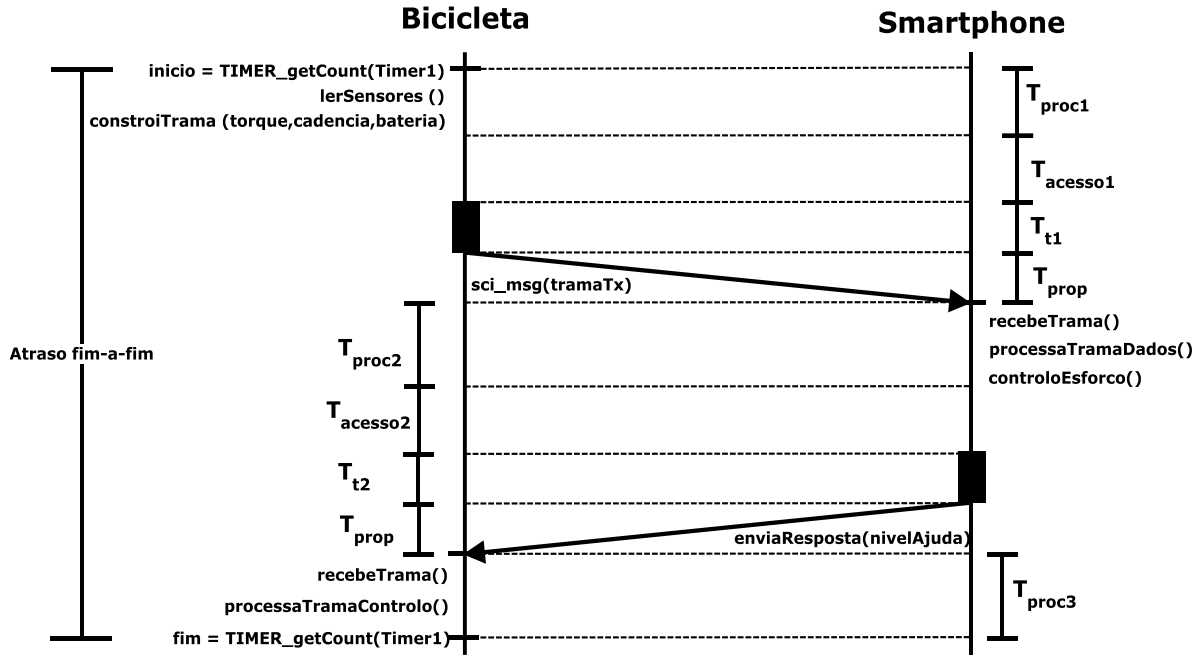

Figura 4.25. Atraso fim-a-fim bicicleta-smartphone-bicicleta.

Os tempos de processamento são referente aos tempos utilizados para realizar tarefas internas do *software* em linguagem C desenvolvido para o microcontrolador da bicicleta e do *software* em linguagem Java desenvolvido para o smartphone.

No que diz respeito ao tempo de transmissão, este considera o tempo que demora até que toda a trama seja disseminada na interface rádio Bluetooth pela interface de comunicação utilizada em ambas as partes. Sendo que o tamanho das tramas enviadas pelo smartphone e pelo microcontrolador da bicicleta não são iguais, consideremos L_1 como sendo o tamanho da trama enviada pelo smartphone e L_2 o tamanho da trama enviada pelo microcontrolador da bicicleta. Neste caso pode-se calcular estes tempos através das seguintes fórmulas:

$$T_{t_i} = \frac{L_i}{BR} \tag{4.11}$$

Relativamente ao cálculo do tempo de propagação, este pode ser calculado através da fórmula:

$$T_{prop} = \frac{d}{V} \tag{4.12}$$

O tempo de propagação (T_{prop}) depende, assim, da distância entre a bicicleta e o smartphone (d), que neste caso é cerca de 1 metro, e da velocidade de propagação do sinal (V), que corresponde a 3 x 10^8 m/s.

O tempo de acesso ao meio por parte do protocolo Bluetooth (T_{acesso}) e o tempo de processamento (T_{proc}) não são determinísticos, pois o primeiro depende da implementação do protocolo de controlo de acesso ao meio e o segundo da disponibilidade do processador e respetivo escalonador do sistema operativo do smartphone bem como do tratamento ao nível das interrupções do microcontrolador do módulo LaunchXL-F28027, daí o tempo de atraso não ter sempre o mesmo valor.

Analisadas estas fórmulas e com base na Figura 4.25, podemos concluir que o atraso fim-a-fim pode ser calculado com base na fórmula:

$$Atraso = \sum_{i=1}^{3} T_{proc_i} + 2T_{prop} + \sum_{i=1}^{2} T_{t_i} + \sum_{i=1}^{2} T_{acesso_i} \tag{4.13}$$

Dado que os tempos de processamento (T_{proc}) e os tempos de acesso ao meio (T_{acesso}) não são conhecidos matematicamente, recorreu-se ao uso de *timers* para a contabilização do atraso fim-a-fim (*Atraso*).

Neste caso foi utilizado o *timer* 1 de 32 bits do microcontrolador do módulo LaunchXL-F28027 para contabilizar este atraso. Após recolha do tempo de início e fim podemos calculá-lo recorrendo à fórmula:

$$Atraso = T_{fim} - T_{inicio} \quad (4.14)$$

As amostras de atraso foram enviadas uma a uma no final de cada trama de dados com base no tipo de trama representado na Figura 4.26.

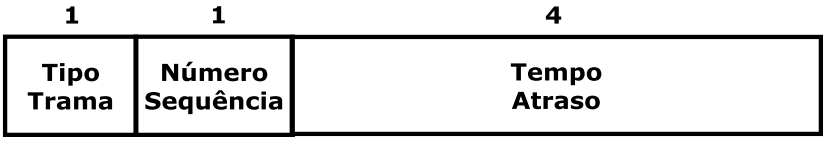

Figura 4.26. Constituição da trama de tempo de atraso.

Este tipo de trama tem apenas 3 campos de dados que são os seguintes:

- **Tipo Trama (1 byte)**: este campo de identificação do tipo de trama foi preenchido com o valor '#' para que fosse identificada como sendo uma trama correspondente ao tempo de atraso;
- **Número Sequência (1 byte)**: identifica o número da trama de atraso e permite saber se alguma trama deste tipo foi perdida;
- **Tempo Atraso (4 bytes)**: campo responsável por armazenar o tempo de atraso contabilizado pelo *timer* 1.

Esta trama, enviada pelo microcontrolador da bicicleta, tem como destino o smartphone. Este, por sua vez, armazena o valor do atraso num ficheiro de texto para que seja possível concluir, pelo uso do *software* Matlab e da função de distribuição acumulada (CDF – Cumulative Distribution Function), qual a probabilidade de o atraso fim-a-fim ser inferior a um determinado valor. Após análise das probabilidades, definiu-se o período mínimo de envio de tramas do microcontrolador da bicicleta para o smartphone para que não haja perda de tramas

por ausência de resposta dentro desse período. O teste englobou a recolha de 3000 amostras de atrasos fim-a-fim com a finalidade de definir qual esse período mínimo, e pode ser consultado na secção 5.4**.**

4.8 Integração Android

O smartphone com o sistema operativo Android foi introduzido numa fase mais avançada do trabalho, dado que inicialmente foi necessário aprimorar o correto funcionamento do microcontrolador do módulo LaunchXL-F28027. Para realizar os testes necessários até este ter um comportamento estável, foi desenvolvida uma aplicação de teste toda escrita em linguagem C para ser executada no PC. Com recurso a esta aplicação foi possível estabelecer uma comunicação coerente com o microcontrolador e respetiva construção de tramas de envio de dados. A Figura 4.27 apresenta o conteúdo de uma trama recebida do microcontrolador, já construída com base no protocolo de comunicação definido na secção 4.6 e a trama processada pela aplicação em C, conforme explicado na secção 4.8.1. Mais abaixo, temos o valor do torque e da cadência calculados pelo programa, bem como a informação do nível da bateria. De realçar que o ADC usado nesta fase da aplicação era ainda o do Arduino (com 10 bits) e a cadência ainda estava a ser calculada com base no número de pulsos e não no tempo entre pulsos.

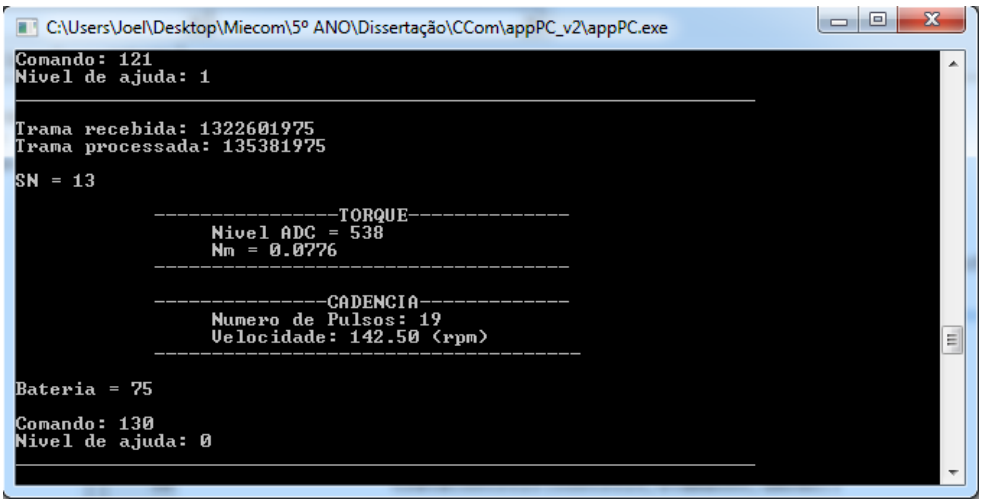

Figura 4.27. Programa para PC em linguagem C.

Posteriormente a esta consolidação do funcionamento do microcontrolador da bicicleta com base no programa para PC, foi iniciada a integração com o sistema operativo móvel, utilizando para isso um smartphone HTC Sensation com a versão 4.0.3 do Android. Através da Tabela 4.2 podemos ver as especificações técnicas deste smartphone.

Tabela 4.2. Características do smartphone HTC Sensation.

Componente	Descrição
Bateria	Li-ion 1520 mAh
Sensores	Proximidade, acelerómetro, luz e giroscópio
GPU	Qualcomm Adreno 220
CPU	1.2 GHz Dual-Core Qualcomm Scorpion
SoC	Qualcomm Snapdragon S3 MSM8260
Tipo de ecrã	S-LCD Capacitivo
USB	MicroUSB v2.0
Armazenamento	4 GB (apenas 1 GB acessível ao utilizador)
Memória RAM	768 MB
Memória externa	Slot microSD
WLAN	Wi-Fi 802.11 b/g/n
WPAN	Bluetooth 3.0
Redes móveis	GSM e UMTS

O desenvolvimento da aplicação para o sistema operativo Android foi conseguido com recurso ao ambiente de desenvolvimento (IDE – Integrated Development Environment) Eclipse. Para que o IDE ficasse dotado com as APIs necessárias para desenvolvimento Android instalou-se um *plugin* chamado ADT (Android Developer Tools).

A aplicação (Mobi.eBike) foi desenvolvida para versões do Android superiores à 4.0 (ICS – Ice Cream Sandwich) embora a sua versão alvo seja a 4.4 (KitKat). Através da Figura 4.28 podemos ver quais as funcionalidades que estão disponíveis na aplicação para o utilizador e quais as etapas necessárias até que seja ativado o controlo de esforço ou resistência.

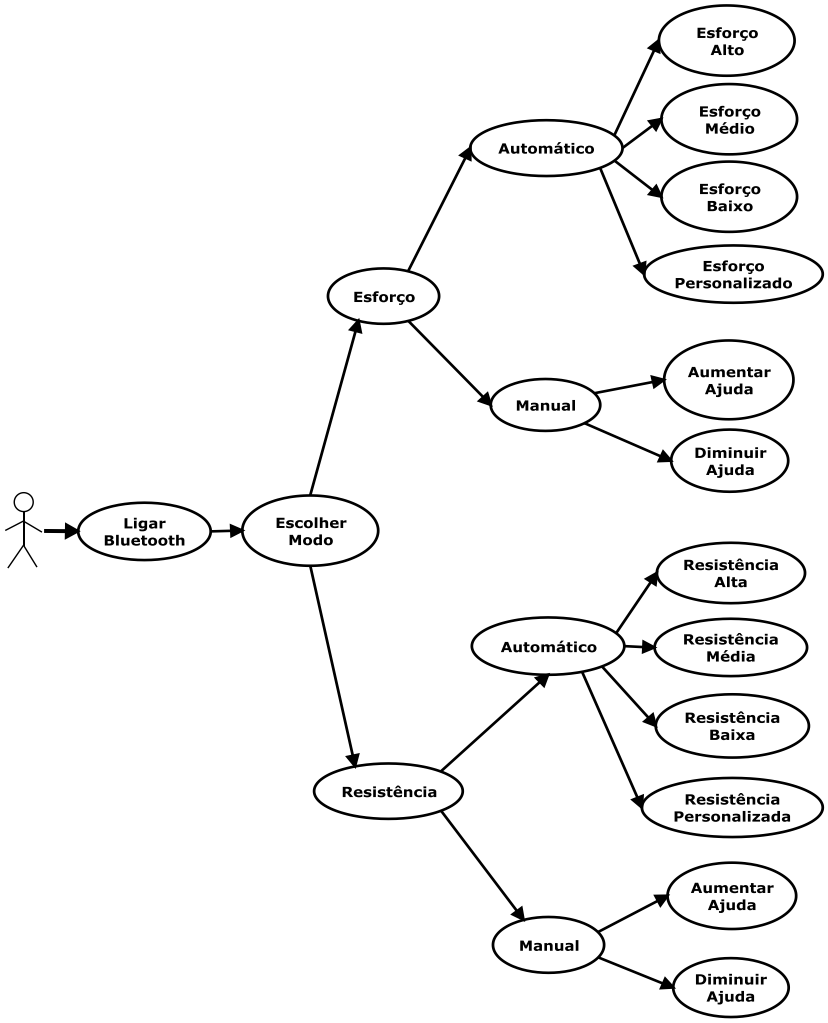

Figura 4.28. Funcionalidades da aplicação Android.

A combinação das diferentes opções de escolha ao longo da aplicação tira partido do conceito de *activity* do Android, onde é utilizada uma janela para mostrar ao utilizador quais as funcionalidades oferecidas pela própria aplicação. Por forma a poder integrar mais funcionalidades numa única *activity* de forma simples, utilizou-se o conceito de *fragment* que no fundo permite que múltiplos *fragments* (cada um com a sua funcionalidade) sejam combinados numa única *activity*.

Para ativar o Bluetooth do smartphone através da aplicação Android usa-se a API *android.bluetooth.BluetoothAdapter*, onde inicialmente verifica-se se o smartphone possui dispositivo Bluetooth, através do método *BluetoothAdapter.getDefaultAdapter()*. Após esta verificação procede-se à ativação do dispositivo utilizando *startActivityForResult(turnOnIntent, REQUEST_ENABLE_BT)*,

seguido de uma pesquisa dos dispositivos com Bluetooth que estejam próximos executando o método *myBluetoothAdapter.isDiscovering()*. Em termos de interface com o utilizador podemos ver na Figura 4.29 a forma como estão construídos os *layouts* das *activities* relativamente a este componente de ativação e pesquisa Bluetooth.

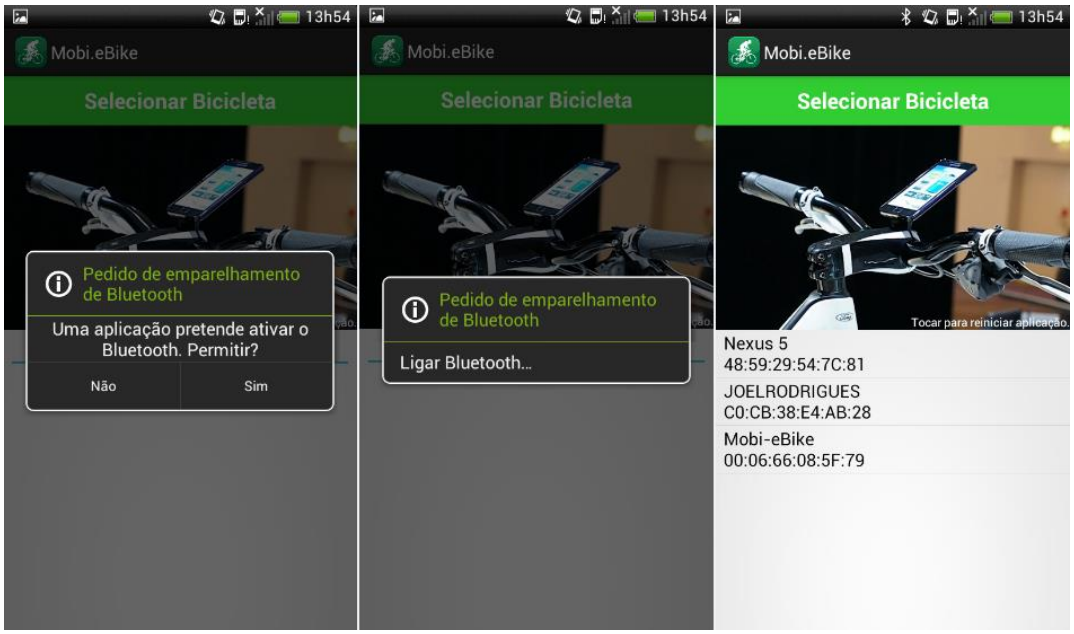

Figura 4.29. Ativação e pesquisa Bluetooth.

A ligação do smartphone com a bicicleta através de Bluetooth é um requisito do sistema, e foi realizada com recurso à uma API do Android chamada *android.bluetooth.BluetoothSocket*. Esta API permite uma ligação entre dispositivos Bluetooth através de um *socket* que se assemelha à forma como funcionam os *sockets* TCP, utilizando assim o paradigma cliente/servidor. Para este caso, o módulo Bluetooth WRL-12580 funciona como servidor e a aplicação Mobi.eBike do smartphone como cliente. Após a escolha do módulo Bluetooth ao qual a aplicação Android se deseja conectar (identificada pelo endereço MAC), usou-se o método *BluetoothDevice.createRfcommSocketToServiceRecord()* para criar o *socket* de comunicação entre ambos e de seguida o método *socket.connect()* para realizar o emparelhamento. Foi ainda necessário criar dois descritores para comunicação entre o Bluetooth da bicicleta e a aplicação Android. O *socket.getInputStream()*, responsável por receber os dados vindos da bicicleta, e o *socket.getOutputStream()*,

para realização do envio de dados para a bicicleta. Por fim, sempre que queremos enviar dados de controlo para a bicicleta usa-se o *out.write(data)* (o parâmetro *data* corresponde ao que queremos enviar pelo descritor de saída) e para ler os dados vindos da bicicleta usa-se *in.read(buffer)* (a variável *buffer* armazena os dados lidos do descritor de entrada).

Após escolha do dispositivo com o qual queremos estabelecer ligação, neste caso o Mobi.eBike, temos de escolher se queremos usar a bicicleta em modo de controlo de resistência ou esforço, conforme o *layout* da aplicação visível na Figura 4.30.

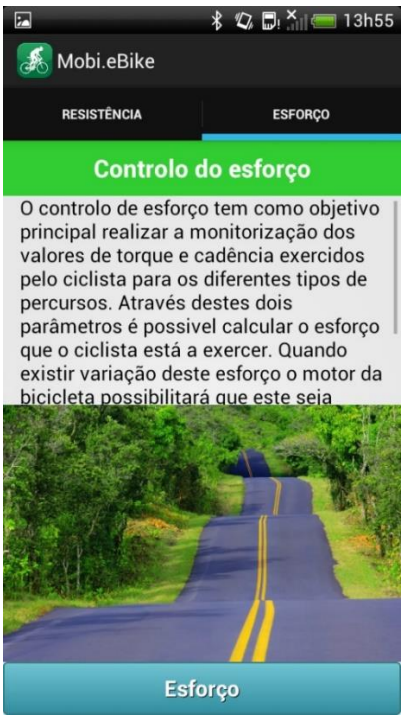

Figura 4.30. Menu de escolha do modo de operação.

Como já referido, estes *layouts* de escolha de resistência ou esforço são *fragments* apresentados sobre uma mesma *activity*. Neste caso, se escolhermos a opção esforço ou resistência, será apresentado de seguida o mesmo conjunto de opções relativos ao tipo de controlo que queremos que seja estabelecido para o funcionamento da bicicleta. Na Figura 4.31 temos as duas opções possíveis de utilização da bicicleta elétrica.

Figura 4.31. Modo de funcionamento do controlo da bicicleta elétrica.

Como podemos verificar na Figura 4.31, à esquerda temos a opção automática, em que o controlo do motor elétrico é realizado com recurso ao algoritmo de controlo incluído na aplicação Android. Já à direita temos a opção manual, onde é o ciclista que define qual o nível do motor que quer aplicar na bicicleta. Em ambas as opções é realizada a verificação se existe cadência na pedaleira por forma a ativar a possibilidade de ajuda do motor conforme legislação rodoviária referida anteriormente na secção 2.3. A Figura 4.32 demonstra a interface para a opção de escolha manual.

Figura 4.32. Interface da opção de controlo manual.

Como é possível ver na Figura 4.32, à esquerda temos um aviso para que a marcha seja iniciada. Este permanecerá até que o ciclista realize um movimento de marcha na pedaleira. Após esse início de marcha, à direita na Figura 4.32 temos a disponibilização de um botão de aumento e outro de diminuição do nível de ajuda do motor, que podem ser livremente utilizados. Contudo, existe um controlo de verificação de nível do motor para que este possa ser utilizado somente entre os intervalos de valores de 0 a 10.

No caso de ser realizada a escolha de funcionamento em modo automático, o ciclista pode escolher qual o esforço que tolerará. Conforme a Figura 4.33, temos 4 níveis de esforço admitidos: alto, médio, baixo e personalizado. Os *thresholds* de esforço, em Watts, são diferentes para cada um dos níveis e estão referidos na secção 4.8.2.

Figura 4.33. Definição do nível de esforço.

Após escolha do nível de esforço, o tratamento que é dado relativamente à requisição de cadência antes do início de auxílio é igual ao exigido no modo manual anteriormente referido. Na Figura 4.34 temos os *layouts* que o utilizador irá visualizar assim que escolher o nível de esforço que deseja tolerar.

Figura 4.34. Interface da opção de controlo automático.

De realçar que o botão "*stop*", existente nos *layouts* onde já está a ser executado o controlo da ajuda do motor, permite que o motor seja prontamente desligado, servindo essencialmente para situações de emergência.

4.8.1 Aquisição e tratamento de dados

A receção, por parte da aplicação Mobi.eBike do smartphone, das tramas de dados enviadas pelo microcontrolador da bicicleta via Bluetooth, estão sujeitas a um tratamento por forma a serem corretamente interpretadas de acordo com os valores reais de envio. Como foi visto na Figura 4.23, temos 9 bytes de *payload* em cada trama enviada por Bluetooth. Estes 9 bytes que constituem a trama de dados agregam 5 campos distintos, e aquando da receção serão entregues à função *processaTramaDados()*, indicada na Figura 4.25, que é responsável pela correta divisão e armazenamento dos dados das tramas em variáveis que são usadas posteriormente ao longo da aplicação.

Os campos que são constituídos por apenas 1 byte (TipoTrama, NumeroSequencia e Bateria) são diretamente armazenados nas variáveis correspondentes. No que diz respeito aos campos com mais de 1 byte (Torque e Cadência), antes de serem guardados nas respetivas variáveis, são processados de forma a serem atribuídos os valores corretos aos seus diferentes bytes e, por fim, realiza-se a soma conjunta dos bytes para serem armazenados nas variáveis correspondentes.

O torque é derivado da leitura do canal do ADC de 12 bits do microcontrolador, pelo que 2 bytes são suficientes para o acondicionar. Desta forma quando a aplicação do smartphone recebe a trama, é necessário ler a posição 0 do campo Torque e multiplicá-lo por 256 e de seguida somar o resultado com a posição 1 do campo Torque. Obtendo assim o valor numérico correto correspondente ao torque.

Os dados relativos ao cálculo da cadência que são recebidos na aplicação do smartphone correspondem ao tempo entre pulsos fornecidos pelo sensor X-Cell RT. Esta contabilização temporal é conseguida com recurso ao *timer* 1 de 32 bits do

microcontrolador, pelo que foram definidos 4 bytes para condicionar corretamente o tempo obtido, o qual vem expresso em microssegundos. Assim é necessário proceder ao cálculo do tempo da cadência (*C*) através da seguinte fórmula:

$$T_{entrePulsos} = (16777216 \text{ x } C[0]) + (65536 \text{ x } C[1]) + (256 \text{ x } C[2]) + C[3] \quad (4.15)$$

4.8.2 Algoritmo de controlo de esforço

Como referido na seção 4.3.3, após os dados serem recolhidos e tratados, possibilitam a realização do cálculo do esforço em Watts. O esforço em Watts é conseguido através da multiplicação entre o torque, em Nm, e a cadência, em rad/s. Este tratamento é realizado no smartphone HTC Sensation pois possui maior capacidade e velocidade de processamento comparativamente com o microcontrolador do módulo LaunchXL- F28027 da Texas Instruments.

Através da Figura 4.35, podemos visualizar o comportamento pretendido para que tenhamos um controlo do esforço adequado. É exigido que o esforço realizado pelo ciclista esteja entre um valor máximo admitido (*thresholdMAX*) e um valor mínimo (*thresholdMIN*) de forma a aproximá-lo de um comportamento constante.

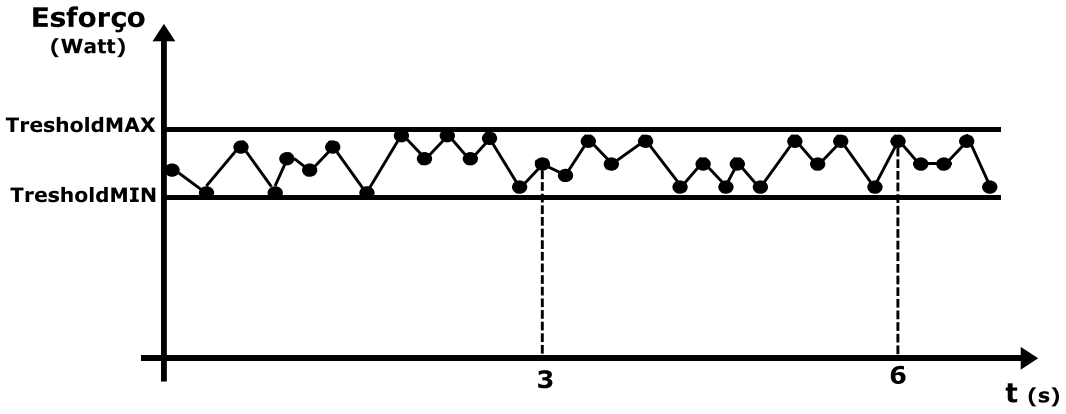

Figura 4.35. Comportamento ideal do gráfico de esforço.

Como é igualmente referido na seção 4.3.3, o valor do esforço utilizado no algoritmo de controlo é o resultado da aplicação do *exponential smoothing*, por forma a controlar variações abruptas e evitar assim a aplicação de ajuda desajustada. O fluxograma da Figura 4.36 ilustra como se realiza o controlo de esforço por parte

do algoritmo implementado na aplicação Mobi.eBike do smartphone. Através deste verifica-se que após a receção no smartphone dos valores de torque (*Torque*) e cadência (*Cadencia*), o smartphone procede ao cálculo do esforço (*Esforco*) que o ciclista está a realizar naquele instante e aplica-lhe a fórmula do *exponential smoothing* (da forma que foi especificado na secção 4.3.3). Segue-se a verificação se o valor de esforço obtido pelo cálculo do *exponential smoothing* (*Esforco_ES*) excedeu o *threshold* máximo (*thresholdMAX*) definido; em caso afirmativo, é incrementada a variável *aumentaAjuda* (variável de contenção do aumento precipitado do nível de ajuda). De seguida é realizada a verificação conjunta se o nível de ajuda do motor (*nivelAjuda*) é menor ou igual ao nível máximo de ajuda do motor elétrico (10) e se a variável de contenção do aumento de ajuda do ciclista é igual a 4, ou seja, se o esforço ultrapassou 4 vezes o *threshold* máximo. No caso negativo, ou seja, se alguma destas condições falhar, calcula-se novo valor de esforço com base em valores de torque e cadência atualizados; em caso afirmativo realiza-se a limpeza da variável de contenção do aumento de ajuda e incrementa-se o nível de ajuda do motor elétrico da bicicleta (*nivelAjuda*).

Para o caso do esforço não ter ultrapassado o *threshold* máximo definido verifica-se se este é inferior ao *threshold* mínimo (*thresholdMIN*); em caso negativo calcula-se novo valor de esforço com base em valores de torque e cadência atualizados; em caso positivo incrementa-se uma variável de contenção (*diminuiAjuda*) para evitar a diminuição precipitada do nível de ajuda. Para este caso segue-se também a verificação conjunta se o nível de ajuda do motor (*nivelAjuda*) é superior ao nível mínimo de ajuda do motor (0) e se a variável de contenção da diminuição de ajuda do motor elétrico é igual a 4, ou seja, se o esforço ultrapassou 4 vezes o *threshold* mínimo. No caso de ambas as condições não serem satisfeitas, volta-se a recalcular novo esforço com base em novos valores de torque e cadência; no caso de ambas as condições serem satisfeitas a variável de contenção de diminuição de ajuda é limpa e decrementa-se o valor atual da variável referente ao nível de ajuda do motor da bicicleta elétrica.

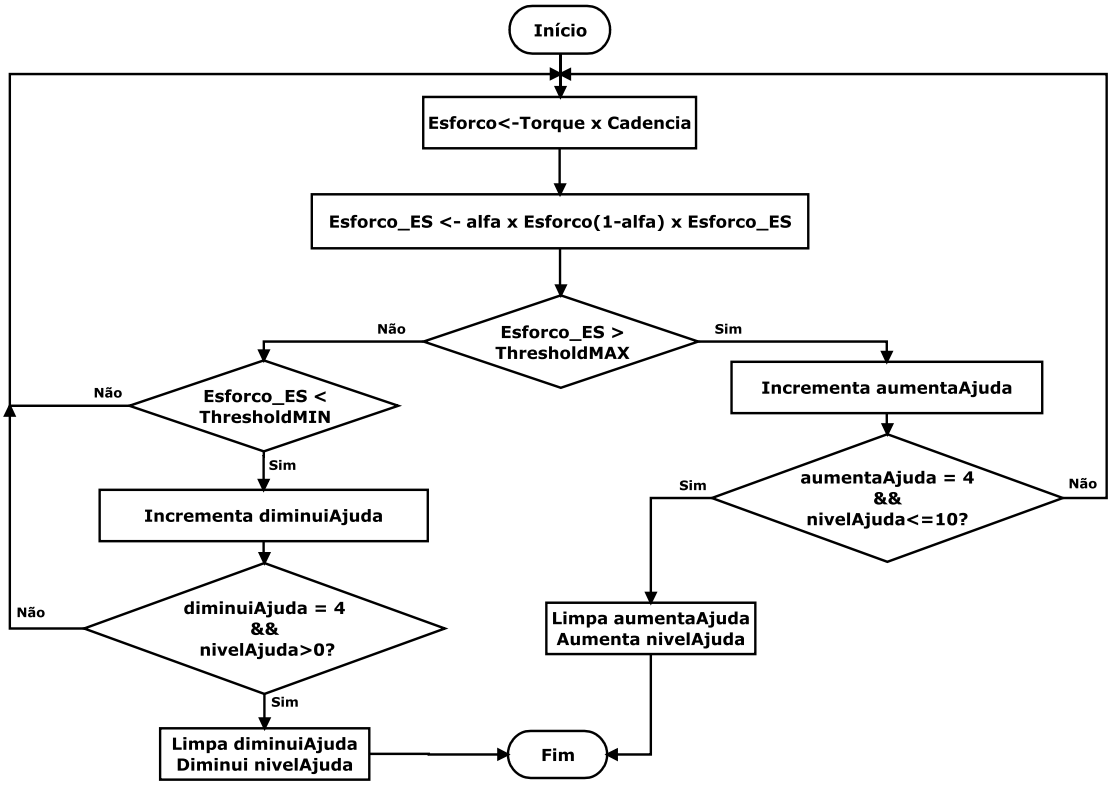

Figura 4.36. Fluxograma de controlo de esforço.

De realçar que o valor de *alfa* está pré-definido na aplicação com o valor de 0.1. Os valores dos *thresholds* admitidos diferem conforme o modo de funcionamento pretendido. Caso tenhamos escolhido na aplicação um nível de esforço alto, o valor do *ThresholdMAX* é de 36 W e o valor do *ThresholdMIN* é de 31 W. No caso de a escolha ter sido um nível de esforço médio, temos um *ThresholdMAX* de 31 W e um *ThresholdMIN* de 26 W. Se for escolhido um nível de esforço baixo, então temos um *ThresholdMAX* de 26 W e um *ThresholdMIN* de 21 W. Por fim, no caso de o ciclista pretender definir manualmente o valor do *ThresholdMAX* e *ThresholdMIN*, estes ficarão com os valores introduzidos.

A Figura 4.35 representa um cenário ideal, em que o esforço do ciclista é constante e não há necessidade de ativação do motor. Contudo, o fluxograma referente ao algoritmo implementado na aplicação do smartphone atua em situações em que o esforço comportamental não é constante e é necessário proceder à alteração dos níveis de ajuda, por parte do algoritmo, para que o esforço do ciclista retorne a um comportamento constante e se assemelhe ao gráfico da

Figura 4.35. De realçar que o período de envio de cada trama por parte do microcontrolador da bicicleta está definido em 200 ms (a justificação da escolha deste período está presente na secção 5.4), e que o nível de ajuda do motor só é aumentado quando o valor do esforço excede em 5 amostras consecutivas o valor do *thresholdMAX* e decrementado quando o esforço é inferior ao *thresholdMIN* igualmente durante 5 amostras consecutivas. Esta condição serve essencialmente para o nível de ajuda do motor não estar a ser precipitadamente alterado sempre que pontualmente existir uma pequena oscilação no esforço do ciclista. Como forma de demonstração teórica do impacto do algoritmo de controlo em termos de controlo do esforço, vejamos o gráfico da Figura 4.37.

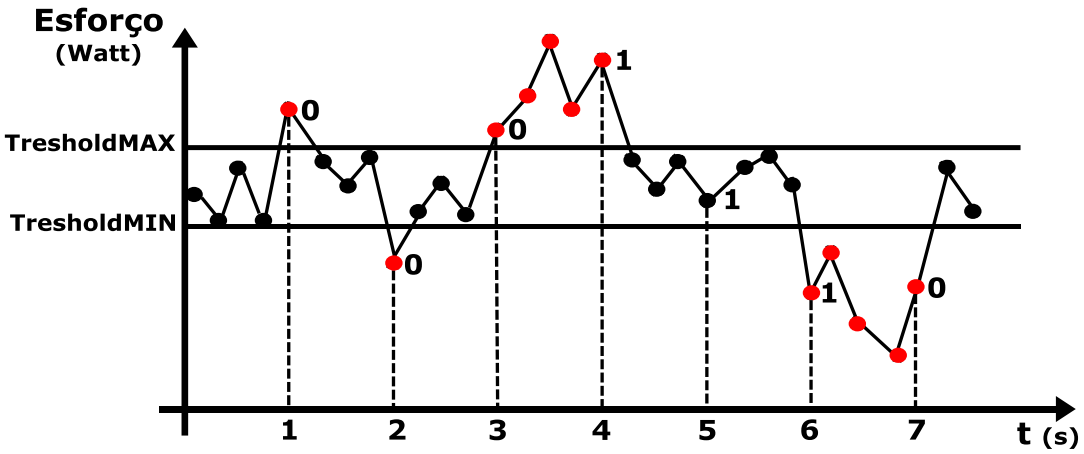

Figura 4.37. Gráfico de variação de esforço.

Nesse gráfico temos representada cada amostra de esforço, bem como o nível de ajuda do motor. A vermelho são indicadas as amostras que ultrapassaram os *thresholds* definidos. Analisando o gráfico de acordo com o fluxograma do algoritmo de controlo de esforço da Figura 4.36, podemos verificar que ao 1 segundo o valor do esforço excedeu o *thresholdMAX* tolerado; contudo, o nível de ajuda do motor manteve-se a 0, dado que *threshold* foi excedido apenas em uma amostra. Como podemos verificar nas amostras seguintes até aos 2 segundos o esforço voltou a valores ideais sem qualquer aumento do nível de ajuda do motor. A mesma situação ocorreu no intervalo de 2 a 3 segundos com a diferença que o valor do esforço teve uma amostra inferior ao *thresholdMIN.* No intervalo de 3 a 4 segundos podemos constatar que o valor de *thresholdMAX* foi excedido em 5 amostras, originando

assim que o nível de ajuda do motor fosse aumentado em 1 nível. Constata-se no intervalo de 4 a 5 segundos que este aumento foi suficiente, dado que o valor do esforço diminuiu para valores de esforço pretendidos, devido ao aumento do nível de ajuda do motor. No intervalo de 6 a 7 segundos pode-se verificar que o nível 1 de ajuda passa a ser exagerado, pois foram recolhidas 5 amostras de esforço inferiores ao *thresholdMIN*, pelo que prontamente é realizada a diminuição do nível de ajuda do motor para 0 por parte do algoritmo de controlo de esforço, de forma a repor o nível de esforço definido inicialmente.

5. Resultados e discussão

Este capítulo apresenta os resultados relevantes para a compreensão e validação do funcionamento prático do trabalho implementado. São apresentados todos os resultados obtidos relativamente às abordagens adotadas para o cálculo da cadência realizada pelo ciclista, juntamente com os testes relativos à resistência a que este está sujeito.

O esforço realizado pelo ciclista é também alvo de teste. Verifica-se a influência que a ajuda do motor provoca no controlo do esforço quando este é alterado, seja por influência da variação da cadência ou pela variação da resistência. O atraso fim-a-fim é igualmente analisado como forma de estabelecimento do valor mínimo para o período de envio de tramas por parte do microcontrolador.

5.1 Cadência

Como já foi discutido na secção 4.3.2**,** a recolha da cadência foi alvo de um melhoramento em prol do melhor funcionamento possível do sistema de controlo de esforço. Assim e aplicada a equação 4.6 obteve-se valores com uma resolução de cadência bastante baixa, dado que, durante o período de recolha de cada amostra (200 ms para este exemplo), o número de impulsos ocorridos nesse curto espaço de tempo não proporcionava uma resolução aceitável para tradução da cadência real. Para melhor entendermos esta falta de resolução, vejamos a Figura 5.1, na qual temos uma espécie de patamares, em que até aos 30 segundos temos maioritariamente 37.5 rpm, obtidos pela contagem de 1 impulso durante os 200 ms de período de amostragem, enquanto dos 30 aos 60 segundos existe uma variação de velocidade entre os 75 rpm e os 112.5 rpm, resultado da contagem de 2 e 3 impulsos, respetivamente. Posteriormente, após 60 segundos, voltamos a ter maioritariamente 37.5 rpm, apesar de termos várias ocasiões que não houve

nenhum impulso ocorrido durante o período de recolha da amostra e consequentemente observação de 0 rpm.

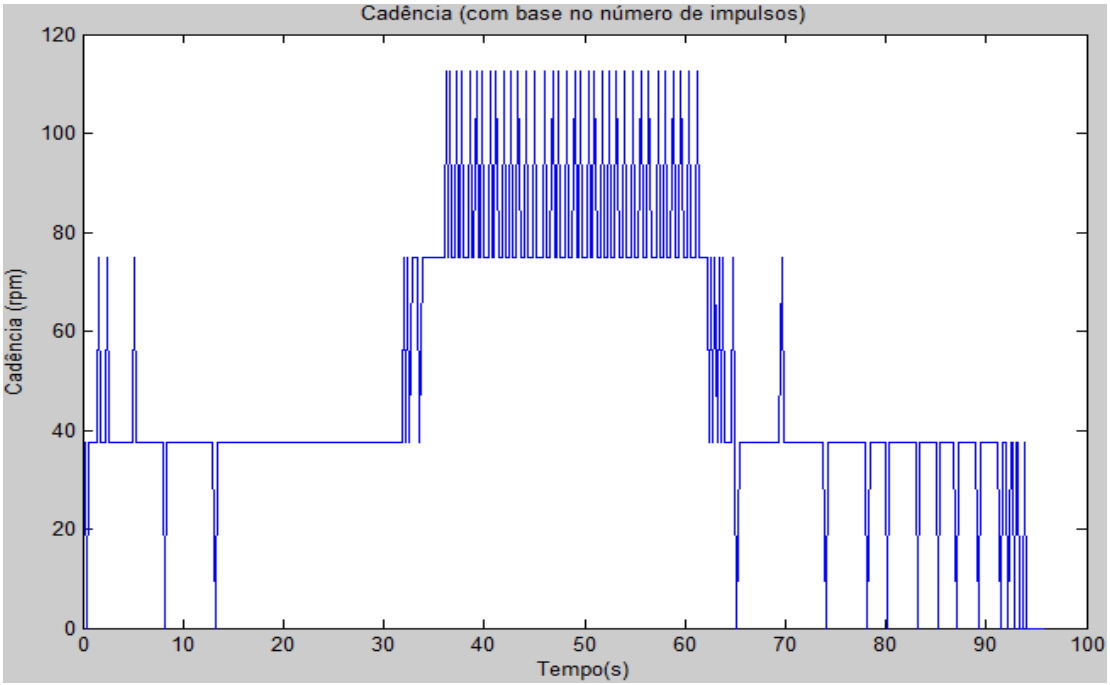

Figura 5.1. Cadência com base no número de impulsos.

Como referido na secção 4.3.2, para colmatar esta limitada resolução procedeu-se à determinação do tempo que ocorre entre cada impulso fornecido pelo sensor correspondente à cadência. Assim como podemos ver na Figura 5.2, para o mesmo comportamento realizado na Figura 5.1, consegue-se uma resolução bastante superior, o que nos permite uma tradução mais aproximada da velocidade real exercida pela ciclista na pedaleira.

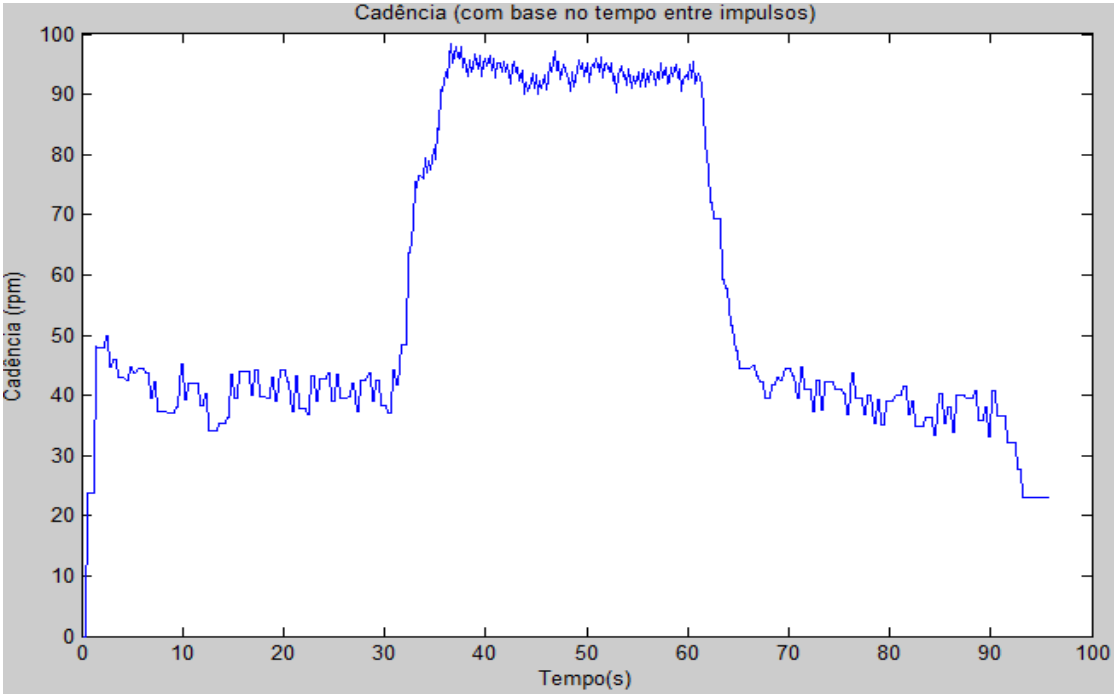

Figura 5.2. Cadência com base no tempo entre impulsos.

5.2 Resistência

Um dos focos desta dissertação passa pela possibilidade de controlar a resistência imposta pela bicicleta ao ciclista. No fundo, a resistência consiste no torque que é necessário aplicar na pedaleira. Dado que os testes foram realizados em laboratório, para simulação de uma maior resistência recorreu-se ao suporte implantado na roda traseira da bicicleta, que permite regular a resistência aplicada à roda. Na Figura 5.3 temos um teste realizado apenas para a monitorização da resistência sentida pelo ciclista. Nesta figura está representado a azul o torque recolhido diretamente do sensor (Torque Instantâneo), a vermelho o torque obtido pela aplicação da técnica de *exponential smoothing* (Torque Exponential Smoothing), e com linhas horizontais os valores dos *thresholds* estabelecidos para este modo de operação.

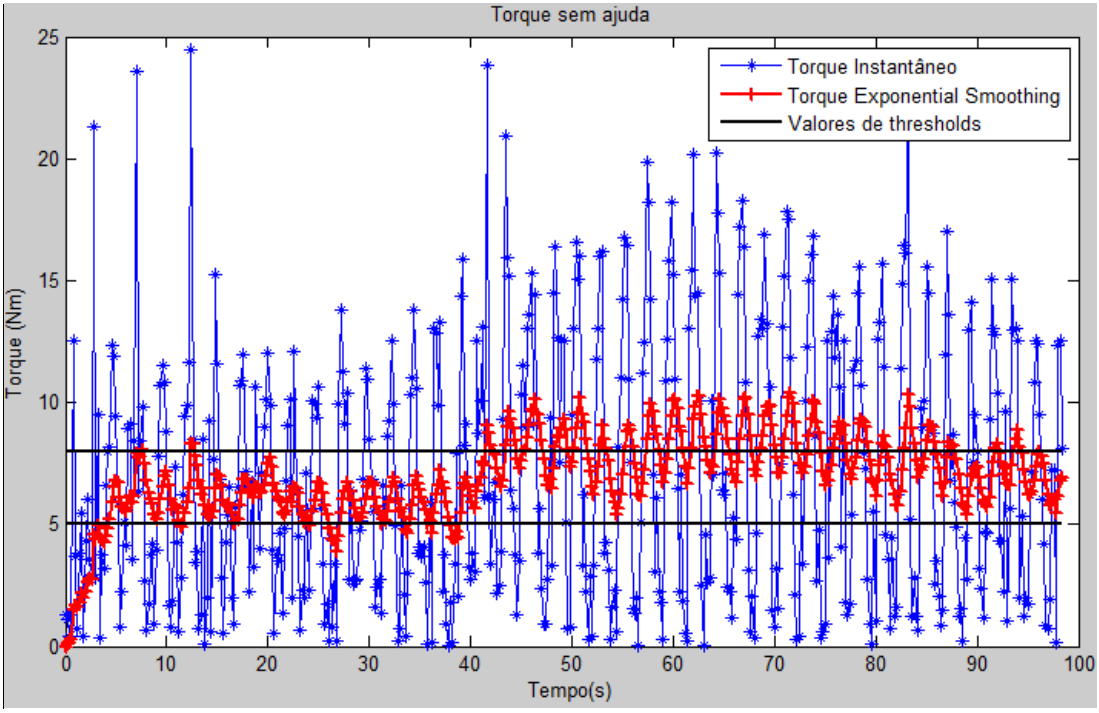

Figura 5.3. Resistência sem ajuda do motor.

Para este teste em concreto, até aos 30 segundos a resistência aplicada pelo suporte foi colocada no valor mínimo. Dos 30 segundos aos 60 segundos a resistência do suporte foi aumentada até atingir o valor máximo, que foi mantido até aos 70 segundos. Após os 70 segundos a resistência foi progressivamente diminuída até atingir novamente o valor mínimo perto dos 100 segundos. Como podemos verificar no gráfico, as linhas horizontais, correspondentes aos valores dos *thresholds*, são repetidamente ultrapassadas pelo torque com *exponential smoothing*, o que permite concluir que não existe qualquer controlo da resistência para este caso.

Após a ativação do motor e do algoritmo de controlo de resistência, é esperado que a resistência sentida pelo ciclista seja atenuada e passe a ter um comportamento mais estável. Realizou-se um novo teste nas mesmas condições que as descritas na Figura 5.3, mas com o motor ligado e em cooperação com o algoritmo de controlo de resistência. Na Figura 5.4 podemos verificar a diferença no comportamento da resistência exercida pela ciclista na pedaleira ao longo do tempo,

em sincronismo com a ajuda exercida pelo motor que foi determinada pelo algoritmo de controlo da resistência.

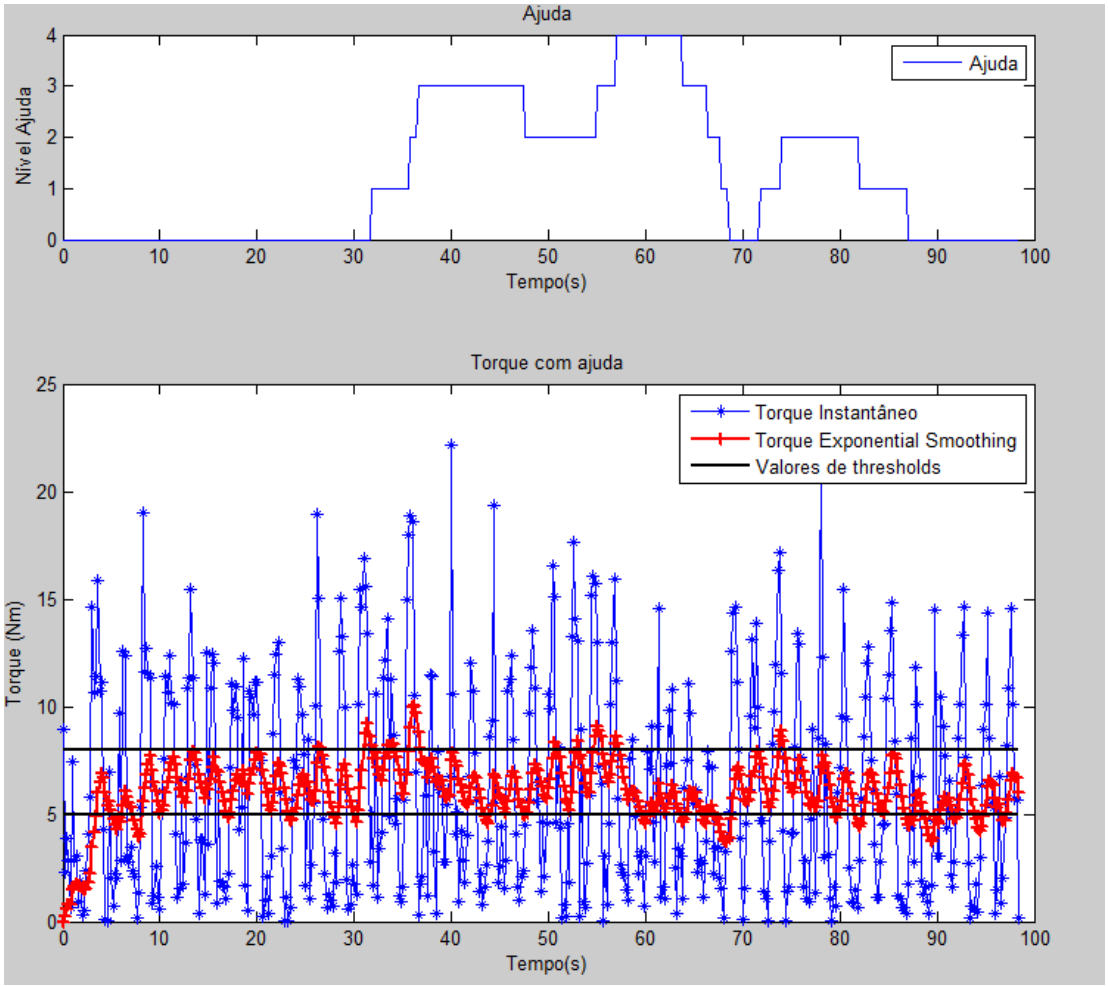

Figura 5.4. Resistência e respetiva ajuda do motor.

Após os 30 segundos, a resistência exercida na roda traseira pelo suporte foi aumentada, o que provocou a ultrapassagem dos valores limites admitidos pelo algoritmo de controlo da resistência, resultando na necessidade de aumento de ajuda do motor para colmatar esse excesso. O nível de ajuda do motor foi ao nível máximo admitido aos 60 segundos, cujo tempo corresponde à resistência máxima aplicada pelo suporte. Como resultado do funcionamento do algoritmo de controlo, proporciona-se ao ciclista uma resistência aproximadamente constante, independentemente da resistência aplicada pelo suporte, devido à ajuda complementar fornecida pelo motor elétrico da bicicleta.

5.3 Esforço

Como explicado na secção 4.3.3, o esforço do ciclista (potência mecânica em Watts) é o resultado da multiplicação do torque em Nm pela cadência em rad/s. Como o esforço está dependente de duas variáveis, o seu controlo tem de ter em consideração a variação da resistência (torque) e da cadência de forma independente. São apresentados de seguida os testes relativos a situações de variação da resistência e cadência, por forma a visualizarmos o comportamento quando não é realizado o controlo de esforço, e posteriormente na situação em que o motor é colocado a funcionar em conjunto com o algoritmo de controlo de esforço, com os respetivos valores de *thresholds* devidamente configurados.

5.3.1 Variação da cadência

Uma situação possível de alteração do valor do esforço realizado pelo ciclista é o aumento da cadência com que este pedala. Iniciada a marcha para este teste, procurou-se manter o valor da cadência nas 30 rpm no intervalo de 5 a 25 segundos. Entre os 25 segundos e os 45 segundos, tentou-se estabilizar a cadência nas 40 rpm. Após os 45 segundos, o valor da cadência foi reduzido novamente para os 30 rpm iniciais. Com este comportamento, é possível testar a influência da cadência na variação do esforço. Podemos ver na Figura 5.5 o comportamento do torque e da cadência para este cenário, sem qualquer influência auxiliar provocada pelo motor elétrico.

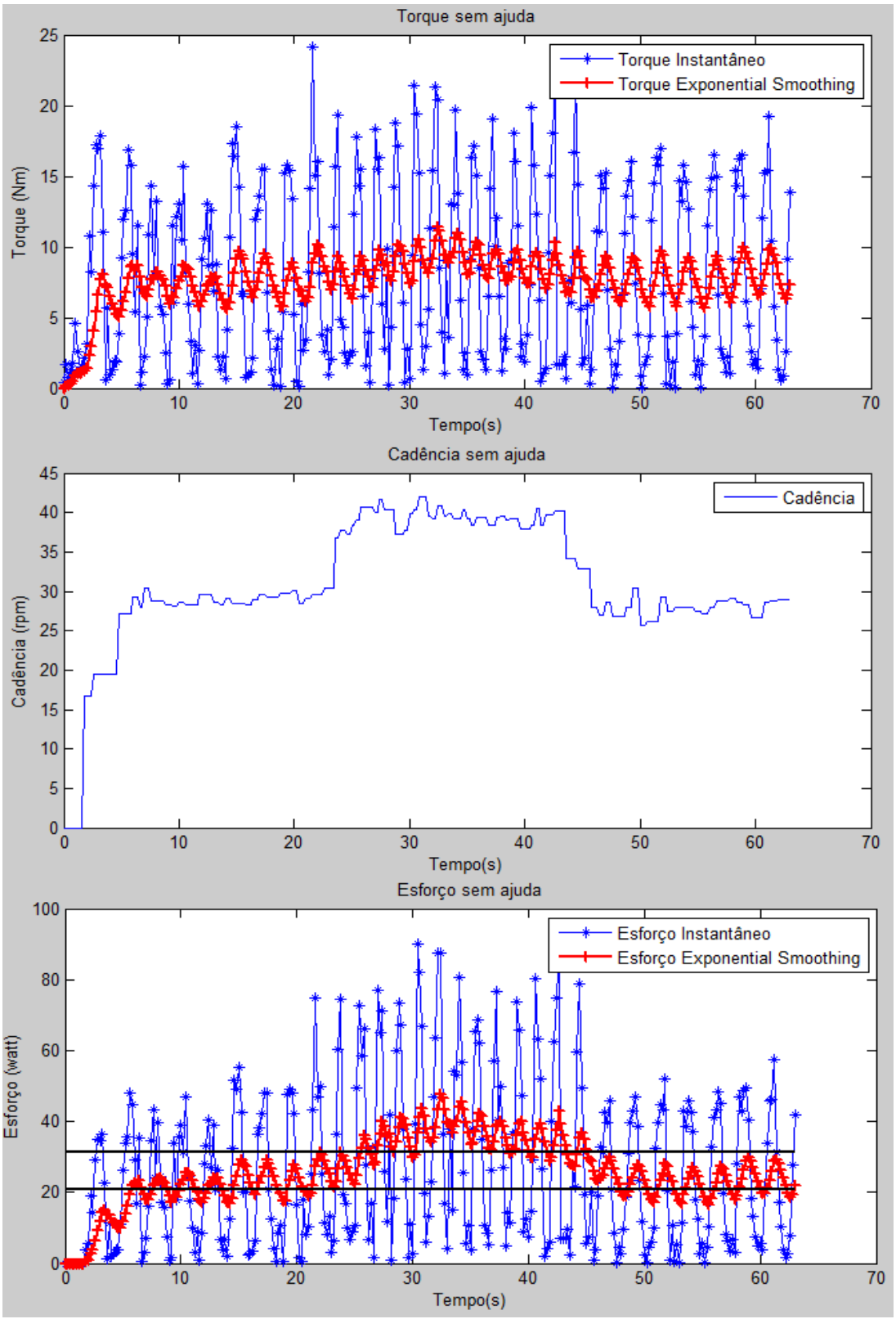

Figura 5.5. Torque, cadência e esforço provocados pela variação da cadência sem ajuda do motor.

É facilmente percetível visualizar que a cadência segue o comportamento descrito acima, e que o torque não sofre uma variação significativa. Com a multiplicação destas duas unidades obtemos o esforço realizado pelo ciclista como é igualmente possível observar na Figura 5.5. Os valores dos *thresholds* representados estão definidos entre os 21 e os 31 Watts e, como podemos observar, não existe nenhum tipo de comportamento controlado relativamente a estes valores limites.

No cenário relativo à inclusão da ajuda do motor elétrico para controlo de esforço, apresentado na Figura 5.6, a cadência realizada pelo ciclista é a mesma que foi descrita acima. Neste caso, o valor do torque passa a ser influenciado pela ajuda do motor elétrico e apresenta um comportamento mais regular do que no exemplo da Figura 5.5.

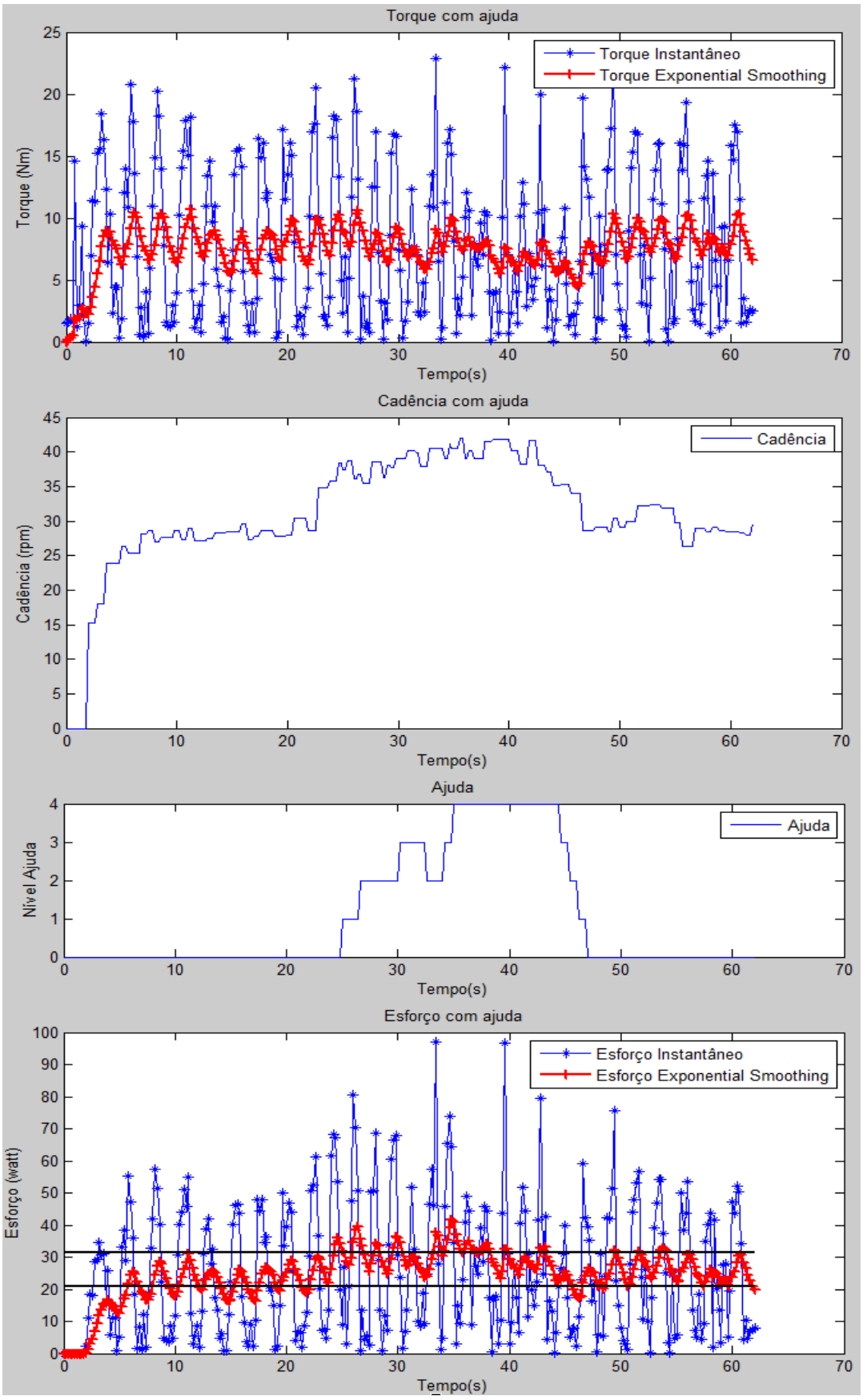

Figura 5.6. Torque, cadência, esforço e nível de ajuda provocados pela variação da cadência.

A multiplicação dos parâmetros de torque e cadência origina o gráfico referente ao esforço, igualmente visível na Figura 5.6. Podemos também observar os níveis de ajuda do motor, acionados pelo algoritmo por forma a atenuar o aumento do esforço do ciclista, provocando assim um comportamento relativo ao esforço mais estável.

5.3.2 Variação da resistência

Uma outra forma de variação de esforço é em situações em que a resistência (torque) varia devido à inclinação do terreno. O teste que se segue traduz um cenário em que após iniciada a marcha, temos uma resistência mínima até aos 25 segundos. De seguida começou-se a aumentar progressivamente a resistência do suporte traseiro até que aos 45 segundos, onde foi atingido o seu máximo permitido. Houve um período de 10 segundos (entre os 45 e 55 segundos) em que foi mantida a resistência no máximo, e de seguida diminuiu-se progressivamente a resistência até esta atingir o valor mínimo, o que ocorreu aproximadamente aos 75 segundos. Ao longo de todo o teste, tentou-se manter uma cadência de 30 rpm. A Figura 5.7 demonstra os valores do torque, da cadência e do esforço para o cenário acima descrito. Relativamente ao esforço, este apresenta-se sem controlo dado que é visível uma insistente ultrapassagem dos valores limites definidos.

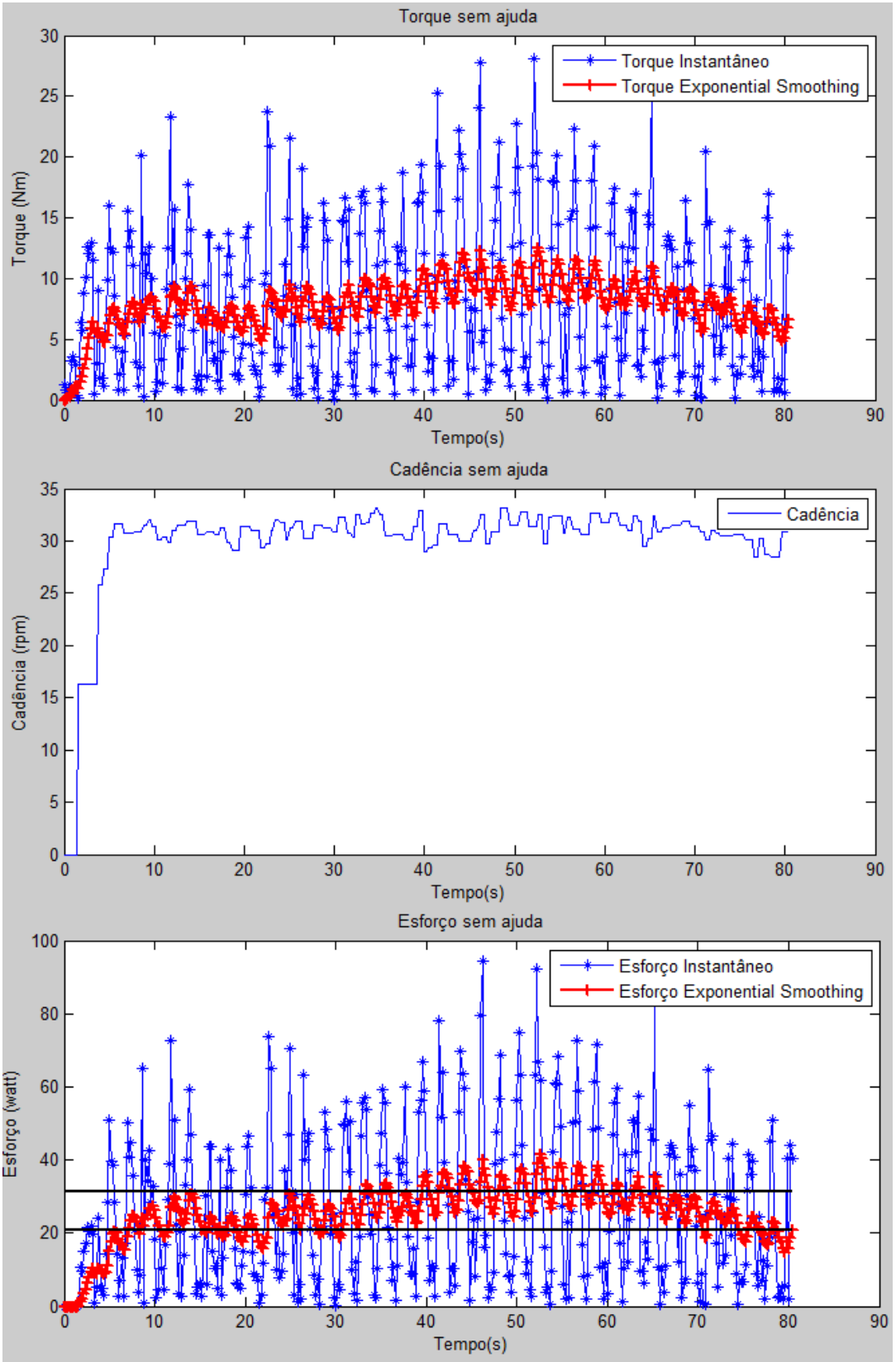

Figura 5.7. Torque, cadência e esforço provocados pela variação da resistência sem ajuda do motor.

Realizando novo teste com os parâmetros iguais aos referidos acima e com ativação do algoritmo de controlo de esforço em cooperação com o motor elétrico, podemos verificar na Figura 5.8 os novos valores mensurados relativamente ao torque e à cadência. É visível uma atenuação do torque entre os 45 e 55 segundos, provocada pela ajuda do motor, que será analisada a seguir.

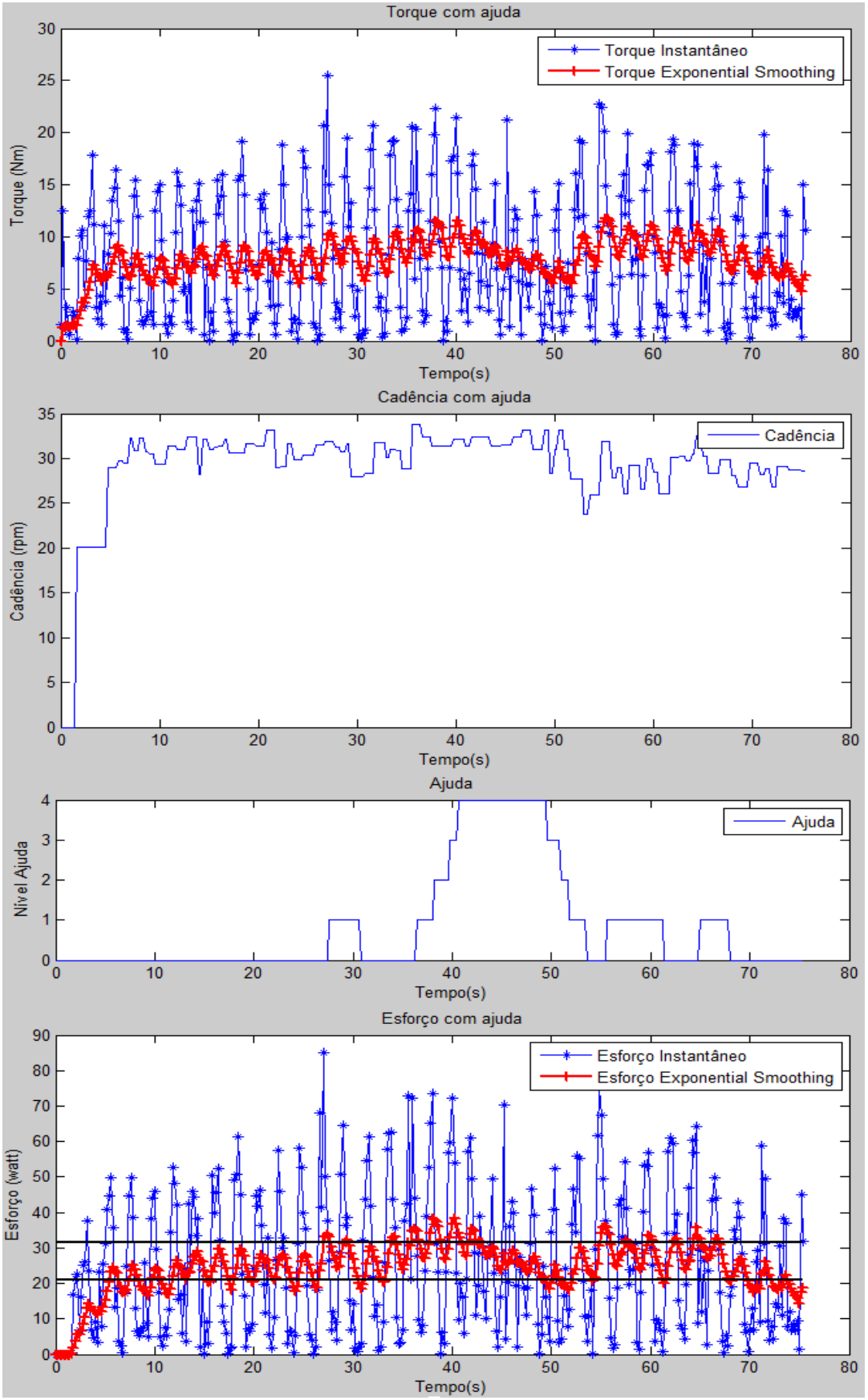

Figura 5.8. Torque e cadência provocados pela variação da resistência com ajuda do motor.

O controlo de esforço provocado pela ajuda do motor elétrico está também ele representado na Figura 5.8. Pode ser constatado que assim que o esforço excede os valores limites definidos, ocorre um pronto aumento do nível de ajuda do motor com a finalidade de atenuar o excesso de esforço. Podemos verificar também que é precisamente entre os 40 e 55 segundos que o motor atinge o maior nível de ajuda, dado que é nesse intervalo que a resistência provocada pelo suporte traseiro é maior, fazendo com que o esforço retorne para os limites de valores definidos. Contudo, aproximadamente entre os 35 e 42 segundos o esforço está acima do valor de limite superior. Neste caso em concreto constata-se que neste período o suporte traseiro foi colocado a exercer uma resistência que os níveis inferiores de ajuda do motor elétrico não foram capazes de contrariar rapidamente. Apenas quando o motor atinge o maior nível de ajuda disponível é que é conseguido realmente contrariar a resistência que está a ser aplicada pelo suporte traseiro. Neste caso, e em semelhança do que se passa nos testes das secções 5.2 e 5.3.1, seria uma mais valia corrigir o problema que leva ao sobreaquecimento dos MOSFETs do bloco de controlo da bicicleta elétrica (referido na secção 4.5.1) e, assim, ser possível aumentar a potência dos níveis de ajuda mais baixos do motor elétrico com a finalidade de ser proporcionada uma maior ajuda e consequentemente uma diminuição mais rápida do esforço que o ciclista está a ser sujeito.

5.4 Atraso fim-a-fim

Como forma de definição do valor a atribuir ao período temporal com que o microcontrolador envia as tramas para o smartphone recorreu-se à contabilização do atraso caracterizado na secção 4.7. Após a recolha de 3000 amostras e com recurso ao *software* Matlab, foi possível agregar esses valores e traduzir na forma de um gráfico qual a probabilidade de ocorrer um determinado atraso. Para isto ser possível, utilizou-se a função de distribuição acumulada (CDF), que originou o gráfico da Figura 5.9, onde é possível verificar a probabilidade do atraso ser maior que o valor correspondente no eixo do x (atraso fim-a-fim).

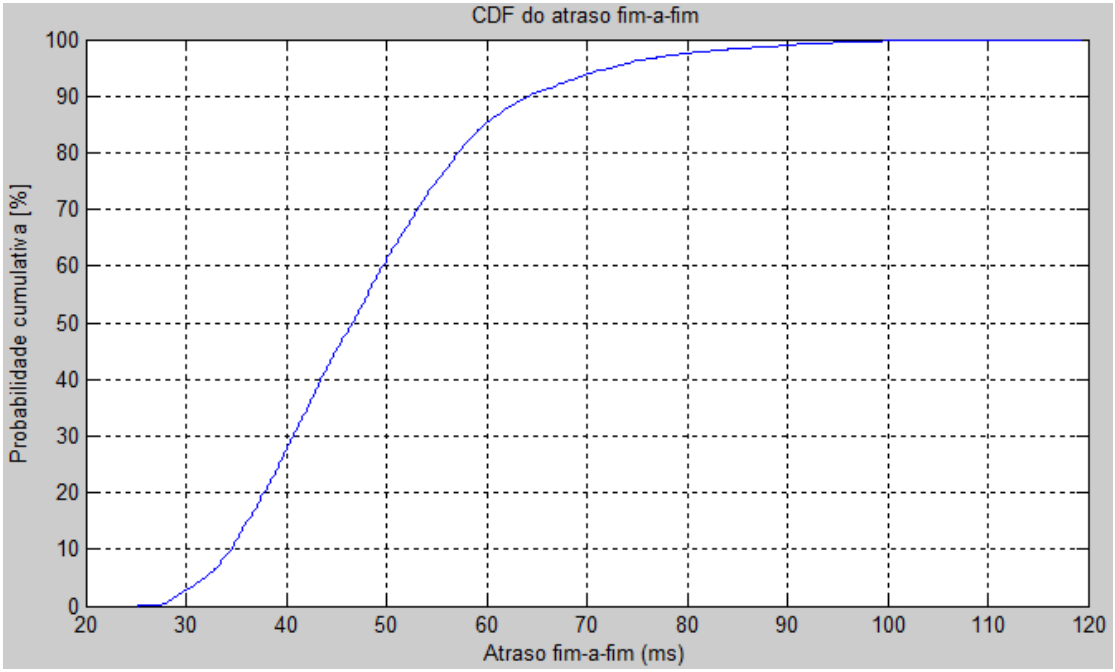

Figura 5.9. Função de distribuição acumulada do atraso fim-a-fim.

Esta abordagem não confere grande resolução para definição do valor mínimo que pode ser definido como período de envio de tramas por parte do microcontrolador de forma a não ocorrer perda de pacotes devido à ultrapassagem do tempo definido para o período do envio de tramas por parte do atraso fim-a-fim. Desta forma, procedeu-se a uma nova abordagem para o tratamento dos valores de atraso recolhidos com base na função de distribuição cumulativa complementar (CCDF – Complementary Cumulative Distribution Function) expressa em escala logarítmica, como está visível na Figura 5.10.

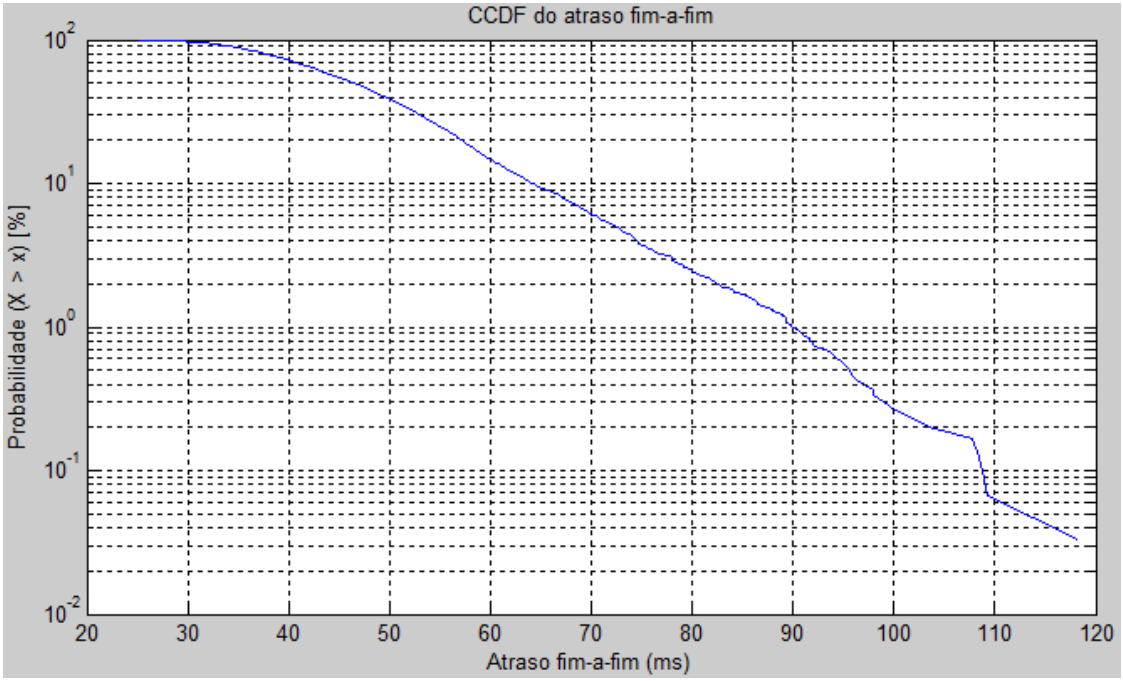

Figura 5.10. Função de distribuição cumulativa complementar do atraso fim-a-fim.

Através da CCDF torna-se possível visualizar com maior detalhe a probabilidade do atraso ser maior que o valor correspondente no eixo do x (atraso fim-a-fim) e determinar qual o valor ideal a atribuir ao período de envio de tramas por parte do microcontrolador. Neste caso, podemos verificar que qualquer valor acima dos 120 ms poderá ser usado para que o período de envio de tramas seja maior que o atraso fim-a-fim.

No caso de pretendermos usar um valor abaixo dos 120 ms sem que haja perda de pacotes, uma possibilidade consiste em migrar o algoritmo de controlo de esforço que foi implementado do smartphone para o microcontrolador. Esta estratégia não faria que a comunicação Bluetooth fosse descartada, pois é sempre necessário comunicação entre o smartphone e o microcontrolador para definição de parâmetros do algoritmo de controlo bem como a aquisição, visualização e armazenamento dos dados. Contudo, nesta dissertação foi estabelecida a estratégia de implementar o algoritmo de controlo de esforço no smartphone, devido à sua grande capacidade de processamento e vasta disponibilidade de recursos comparativamente ao microcontrolador.

6. Conclusões

A combinação de redes de área pessoal sem fios com sensores e atuadores permite que novas aplicações sejam desenvolvidas no âmbito da saúde e entretenimento. Através do rápido desenvolvimento de novos sensores e atuadores, e da sua constante miniaturização, permite-se que novas formas de aplicabilidade sejam possíveis de executar. Os *standards* de redes de sensores sem fios como o Bluetooth e o IEEE 802.15.4 fornecem qualidade de serviço e interoperabilidade na comunicação entre componentes.

Produtos populares como os smartphones são um alvo bastante apetecível de utilização para recolha de dados sensoriais centrada no desempenho físico das pessoas, dado que possuem sensores integrados, interface Bluetooth para comunicação com dispositivos externos, e disponibilizam uma API para que sejam desenvolvidos programas a operar sobre o sistema operativo do smartphone, para interpretação dos dados dos sensores.

Atualmente a venda de bicicletas elétricas em todo o mundo superou os 30 milhões de unidades vendidas, pelo que universidades e empresas de todo o mundo vislumbraram enormes potencialidades de aplicações capazes de melhorar o funcionamento destas, e até mesmo conferirem-lhe novas aplicações de utilização com base em smartphones e dados recolhidos de sensores implantados na própria bicicleta.

Nesta dissertação foi implementado um módulo numa bicicleta elétrica capaz de monitorizar e controlar o esforço a que o ciclista está a ser sujeito, através de um sensor de torque e cadência, que comunica com um módulo que contém um microcontrolador integrado com um módulo de interface Bluetooth passível de emparelhar e comunicar com um smartphone. O sensor foi implementado no rolamento interno da pedaleira, o que lhe confere a vantagem de ficar invisível e imune a adversidades naturais como a poeira e a chuva. A programação do microcontrolador foi realizada de raiz em linguagem de programação C, e a

programação da aplicação do smartphone em Java, sendo que o Bluetooth foi o tipo de comunicação adotado.

O smartphone contém uma aplicação com um algoritmo de interpretação do esforço do ciclista e, com base nisso, decide qual o nível de ajuda a aplicar no motor da bicicleta elétrica para que o esforço do ciclista seja constante. Como podemos verificar nos resultados obtidos na Figura 5.6 e Figura 5.8 da secção 5.3, não se verificou um esforço sempre constante, sendo que em determinadas alturas este ultrapassou durante cerca de 7 segundos o valor máximo de esforço definido pelo ciclista na aplicação do smartphone. Este comportamento deve-se ao facto de terem sido utilizados apenas os 4 níveis de ajuda mais baixos dos 10 inicialmente previstos. Estes níveis de ajuda inferiores do motor não possuem grande potência e, consequentemente, não foram capazes de proporcionar uma ajuda conivente com a necessária para atenuar o esforço do ciclista. Uma possível solução passa por resolver a questão de sobreaquecimento dos MOSFETs do bloco de controlo da bicicleta elétrica, o que permitirá aumentar a potência de ajuda dos níveis mais baixos do motor elétrico, sem que haja perigo de danificar todo o bloco de controlo.

De forma que o valor de esforço fosse utilizado coerentemente pelo algoritmo de controlo, aplicou-se um mecanismo para suavizar as variações abruptas de esforço detetadas pelo sistema. Através da técnica *exponential smoothing* contorna-se a ativação precipitada de um nível de ajuda do motor que não se adeque ao exigido pelo ciclista, dado que esta técnica tem a particularidade de atribuir um maior peso aos últimos valores utilizados e integrar todo o passado recolhido no cálculo do valor atual, armazenando-o num único valor. A escolha de colocar o algoritmo de controlo no smartphone passa pela sua grande capacidade de processamento e vasta disponibilidade de recursos comparativamente ao microcontrolador do módulo LaunchXL-F28027 da Texas Instruments. De realçar que o algoritmo implementado é de fácil alteração no caso de ser pretendido adaptar o sistema a outros contextos de aplicação tendo por base a bicicleta elétrica.

Como todo o sistema está implementado ao nível da arquitetura aplicacional e das comunicações de todas as partes, é necessário a definição de um período

mínimo de 120 ms entre o envio de tramas do módulo da bicicleta para o smartphone. No caso de necessitarmos de um valor inferior a estes 120 ms terá de ser excluída a hipótese de ser o smartphone a executar o algoritmo de controlo de esforço e implementar esse controlo no microcontrolador. Esta estratégia não faria que a comunicação Bluetooth fosse descartada, pois é sempre necessário comunicação entre o smartphone e o microcontrolador para definição de parâmetros do algoritmo de controlo bem como a aquisição, visualização e armazenamento dos dados.

A utilização da bicicleta em ambiente exterior (sem suporte traseiro) não foi realizada dado que, aquando dos testes em ambiente laboratorial (com suporte traseiro), verificou-se uma limitação em termos de capacidade de ajuda do motor elétrico ao aplicar-se uma grande resistência na roda traseira por parte do suporte. Esta limitação consistiu no já referido aquecimento excessivo do bloco de controlo da bicicleta elétrica, o que provoca um término abrupto do funcionamento de todo o bloco de controlo. Concluiu-se assim que não estavam reunidas as condições necessárias para a passagem dos testes para ambiente exterior, e optou-se por refinar ao máximo todo o funcionamento da bicicleta em ambiente laboratorial para que futuramente, aquando da sua possível utilização em ambiente exterior, estivesse o mais desenvolvido possível para funcionar neste tipo de cenário.

Como trabalho futuro propõe-se uma nova funcionalidade para o controlo de esforço. Em vez de termos um esforço sempre constante, proporcionar-se-ia a hipótese de escolha de um modo em que o esforço seria alterado, mas de uma forma controlada. Por exemplo, permitir-se-ia que o esforço passasse por três patamares diferentes de esforço, que fossem ativados de forma gradual e cíclica. É também sugerido que se integrasse a monitorização conjunta do controlo de esforço e do controlo cardíaco. Assim, o parâmetro de monitorização que ultrapassasse primeiro os limites estabelecidos para o normal controlo da atividade física do ciclista acionaria o controlo para alteração do motor elétrico, com o objetivo de repor os valores que ultrapassem os limites definidos.

Um outro ponto fundamental é o aquecimento excessivo dos componentes da bicicleta elétrica, que limitou significativamente o tempo de uso contínuo e impediu a utilização dos níveis superiores de ajuda do motor elétrico nos testes para o controlo de esforço em laboratório, inviabilizando também os testes em ambiente exterior. Desta forma, sugere-se, uma vez mais, que melhoramentos sejam realizados relativamente ao sobreaquecimento dos MOSFETs do bloco de controlo. Propõe-se também que a solução já encontrada e implementada seja integrada num único microcontrolador para o controlo e monitorização da bicicleta elétrica. Por último, mas não menos importante, funcionalidades de segurança ao nível do roubo da bicicleta podem ser pensadas. Por exemplo, aquando de um afastamento abrupto entre o smartphone e a bicicleta, provocar um bloqueio da roda traseira por forma a impedir que alguém utilize a bicicleta indevidamente.

Referências

[1] A. T. Campbell, N. D. Lane, E. Miluzzo, R. A. Peterson, & S. B. Eisenman, "The Rise of People-Centric Sensing.": *IEEE Internet Computing*, vol. 12, no. 4, pp. 12-21, Jul. 2008.

[2] P. Ferreira, J. A. Afonso & H. Fernandez-lopez, "Sistema de Sensorização Móvel e Controlo baseado em ZigBee para Bicicletas Elétricas", 2012.

[3] Thun: X-CELL RT: http://www.thun.de/en/products/bb-cartridges/x-cell-rt/ (2013-12-03).

[4] H. Arfwedson and R. Sneddon, "Ericsson's Bluetooth modules", 2002.

[5] IEEE, 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low Rate Wireless Personal Area Networks (LR-WPANs), IEEE Std., 2003. [Online]. Disponível em: http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf.

[6] J. Zheng & M. J. Lee, "A Comprehensive Performance Study of IEEE 802.15.4", 2004.

[7] FIPS Pub 197, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, US Department of Commerce/N.I.S.T, Springfield, Virginia, November 26, 2001, (http://csrc.nist.gov/).

[8] J. Haartsen, M. Naghshineh, J. Inouye & W. Allen, "Bluetooth: Vision, Goals, and Architecture", October 1998, pp. 1–8.

[9] "HxM Bluetooth API Guide", Zephyr technology, 2010. [Online] Disponível em: http://zephyranywhere.com/media/pdf/HXM1_API_P-Bluetooth-HXM-API-Guide_20100722_V01.pdf.

[10] "Specification of the Bluetooth System", pp. 19 [Online]. Disponível em: http://grouper.ieee.org/groups/802/15/Bluetooth/core_10_b.pdf.

[11] B. Fernades, J. A. Afonso, R. Simões, “Vital Signs Monitoring and Management using Mobile Devices”, 6ª Conferência Ibérica de Sistemas de Tecnologias de Informação (CISTI 2011), Chaves, Portugal, June 2011.

[12] E. T. Coelho, P. Carvalhal, M. J. Ferreira, L. F. Silva, H. Almeida, C. Santos, J. A. Afonso, “A Bluetooth-based wireless distributed data acquisition and control system”, Proceedings of IEEE International Conference on Robotics and Biomimetics - ROBIO 2006, Kunming, China, December 2006.

[13] A. Roetynck, “Promoting Cycling for Everyone as a Daily Transport Mode”, Belgium, February 2010.

[14] Samsung Maestros Academy: “Samsung Smart Bike”, Samsung Electronics, Itália, 2014. [Online] Disponível em: http://www.maestrosacademy.it/progetto-sbike.

[15] W. Walker, L. P. Aroul, & D. Bhatia, “Mobile health monitoring systems.”: Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Conference, 2009.

[16] Texas Instruments Inc., “16-bit ultra low power mixed signal microcontroller” [Online] Disponível em: http://www.ti.com/msp430.

[17] Texas Instruments Inc., “2.4 GHz IEEE 802.15.4 RF Transceiver” [Online] Disponível em: http://www.ti.com/product/CC2420?keyMatch=cc2420.

[18] S. B. Eisenman et al., "The BikeNet mobile sensing system for cyclist experience mapping," *5th International Conference on Embedded Networked Sensor Systems*, Sydney, Austrália, 2007.

[19] Tmote Invent: “Wireless Sensing System”, [Online] Disponível em: http://www.tempsensornews.com/generic-temp-sensors/tmote-inventtm-wireless-sensing-system.

[20] C. Outram et al., "The Copenhagen Wheel: An innovative electric bicycle system that harnesses the power of real-time information and crowd sourcing", *EVER’2010*, Monaco, March de 2010.

[21] R. Araújo, “Desenvolvimento de uma bicicleta elétrica”, DEI – Universidade do Minho, 2012.

[22] “Market share by product category”, [Online] Disponível em: http://www.idc.com/getdoc.jsp?containerId=prUS24314413.

[23] “Smartphone Operating System”, [Online] Disponível em: http://www.idc.com/getdoc.jsp?containerId=prUS24676414.

[24] Texas Instruments Incorporated, “LAUNCHXL-F28027 C2000 Piccolo LaunchPad Experimenter Kit”, Literature Number: SPRUHH2A, January 2014.

[25] Texas Instruments Incorporated, “Piccolo Microcontrollers”, Literature Number: SPRS523J, October 2013.

[26] R. Networks, “Bluetooth Data Module Command Reference & Advanced Information User Guide”, 2013, pp. 1-83.

[27] M. Van. Ditten, “Torque Sensing for E-bike Applications”, Department of Precision and Microsystems Engineering – Technische Universiteit Delft, Netherlands, June 2011.

[28] STMicroelectronics Group, “Low drop fixed and adjustable positive voltage regulators”, 2005, pp. 2.

Anexo I

I. Tutorial Arduino Duemilanove e módulo Bluetooth WRL-12580

O foco deste tutorial consiste numa análise da plataforma Arduino Duemilanove, do módulo Bluetooth WRL-12580, da forma como estes se instalam num sistema operativo Windows e do modo como estes podem ser colocados a operar conjuntamente.

I.1. Arduino Duemilanove

A plataforma de desenvolvimento usada neste tutorial é o Arduino Duemilanove, que tem integrado um microcontrolador ATmega328P. Recorde-se que um microcontrolador é composto por um processador, periféricos de I/O e memória. A título de exemplo, com recurso a sensores ligados aos seus pinos de entrada (analógicos ou digitais, explicados mais à frente) é possível a transformação de grandezas físicas em grandezas elétricas onde, posteriormente a esta conversão, o microprocessador processa esta informação com base no *software* que tem programado no momento e possibilita uma resposta/ação dinâmica e personalizada por parte de um componente externo que esteja ligado a um pino de saída.

A Figura A.1 apresenta uma breve descrição do Arduino Duemilanove onde é necessário analisar cada uma das componentes em destaque por forma a melhor compreender o restante tutorial.

Como principais componentes temos:

- **Pinos digitais**: O Arduino possui 14 pinos digitais que operam a 5 V. Especial destaque para os pinos digitais: número 0 (RX) e número 1 (TX)

usados para receber (RX) e transmitir (TX) dados em série TTL (0 V ou +5 V) com base no protocolo UART (Universal Asynchronous Receiver/Transmitter);

- **FTDI**: Módulo responsável pela conversão de UART para sinais USB, fornecendo assim uma porta COM virtual. Os *drivers* referentes a este módulo (existentes no software do Arduino) podem ser descarregados em http://www.ftdichip.com/Drivers/VCP.htm;
- **USB**: Conector USB com duas funcionalidades: uma permite que o microcontrolador seja reprogramado e outra que sirva de alimentação para todo o módulo;
- **Alimentação**: A particularidade deste componente consiste na possibilidade de alimentação da plataforma sem que tenhamos uma interface USB. Assim é possível alimentá-la de duas formas: através de uma fonte externa ligada a uma tomada com um conector de 2,1 mm (centro positivo) até um máximo de 12 V DC ou através de uma bateria externa conectada aos pinos V_{in} (tensão) e GND (terra);
- **Pinos de alimentação**: Pinos que servem essencialmente para fornecer alimentação a elementos externos à placa. Estes pinos (5 V e o 3.3 V) são alimentados pela ligação da interface USB;
- **Pinos analógicos:** Existem 6 entradas analógicas, onde cada uma delas está ligada a um ADC (Analog-to-Digital Converter) de 10 bits ($2^{10} = 1024$) o que possibilita conversões de tensões entre 0 V e 5 V em um valor de 0 a 1023;
- **Microcontrolador:** O microcontrolador é de 8 bits e tem como arquitetura base a RISC (Reduced Instruction Set Computer). Em termos de memória, tem disponível 32 kB de memória Flash onde são armazenados os programas, 1 kB de memória EEPROM (Electrically-Erasable Programmable Read-Only Memory) acessível através da biblioteca EEPROM onde os dados são guardados permanentemente e 2 kB de memória RAM (Random Access Memory) onde são alocadas todas as variáveis até o circuito ser desligado. Para além dessas especificações

podemos realçar a existência de três *timers* e, como referido anteriormente, 6 canais ADC de 10 bits;

- ***Reset*:** Botão que permite fazer com que o processamento do microcontrolador volte ao estado inicial. O Arduino Duemilanove possui a capacidade de realizar o *reset* através de *software*, ou seja, sem a ação física do botão, para que seja possível realizar o *upload* de um novo programa para o microcontrolador carregando somente no respetivo botão do IDE (Integrated Development Environment);
- **Cristal:** Permite uma frequência de oscilação de 16 MHz.

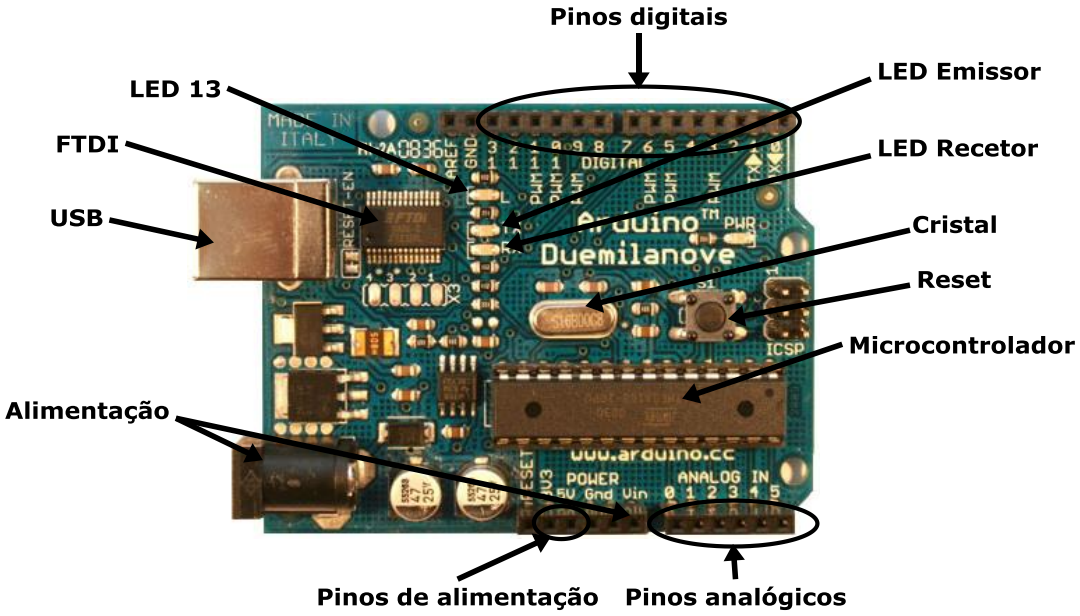

Figura A.1. Descrição dos componentes principais do Arduino Duemilanove.

A Tabela A.1 sintetiza a informação relevante acerca da plataforma Arduino Duemilanove adotada.

Tabela A.1. Característica de funcionamento dos componentes principais do Arduino Duemilanove.

Componentes	Características
Microcontrolador	ATmega328
Pinos de I/O digitais	14
Pinos de entrada analógica	6
Tensão de operação	5 V
Tensão limite de alimentação	6 V - 20 V
Tensão recomendada de alimentação	7 V - 12 V
Memória Flash	32 kB
RAM	2 kB
EEPROM	1 kB
Frequência do clock	16 MHz

I.1.1. Instalação da plataforma Arduino Duemilanove

A instalação da plataforma para o sistema operativo Windows passa por várias etapas.

1) **Fazer *download* do Arduino IDE e instalá-lo:**

Para fazermos o *download* do software do Arduino temos de aceder a http://arduino.googlecode.com/files/arduino-1.0.5-windows.exe. Este ficheiro de *download*, para além do Arduino IDE, contém também os *drivers* para correta instalação e configuração do módulo FDTI USB.

Depois de executarmos o ficheiro de instalação, e de acordo com a Figura A.2, começamos por ler o acordo de licença para procedermos posteriormente à aceitação deste carregando no botão “I Agree”.

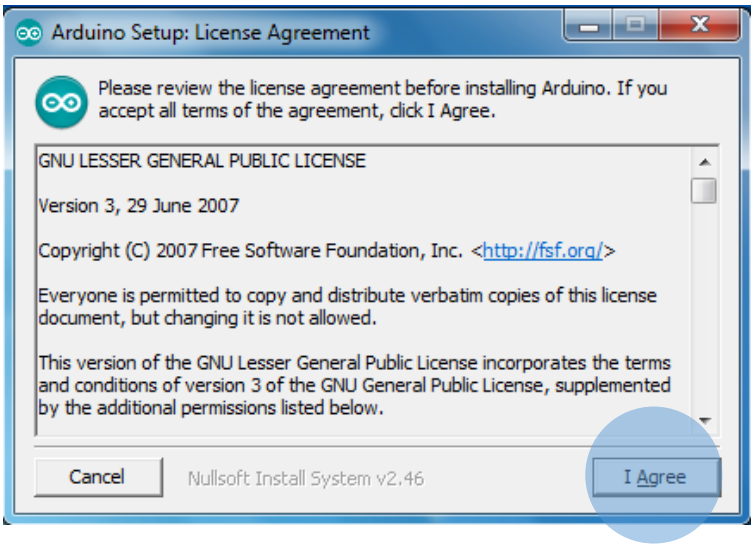

Figura A.2. Primeiro passo da instalação do Arduino Duemilanove.

De seguida temos a possibilidade de escolher quais os componentes que desejamos instalar. É aconselhável que seja selecionada a instalação por defeito do *software* Arduino e a instalação dos *drivers* USB. A criação de atalhos para acesso ao *software* é facultativa.

Posto isto podemos seguir na instalação carregando no botão "Next" visível na Figura A.3.

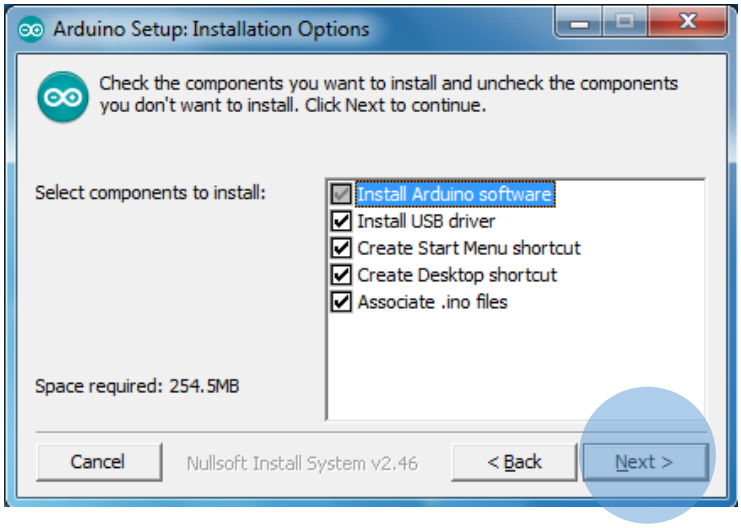

Figura A.3. Segundo passo da instalação do Arduino Duemilanove.

A especificação do caminho onde será instalado o *software* é de seguida requerida ao utilizador, e para tal aconselha-se que este fique guardado no caminho que estiver especificado, conforme mostra a Figura A.4.

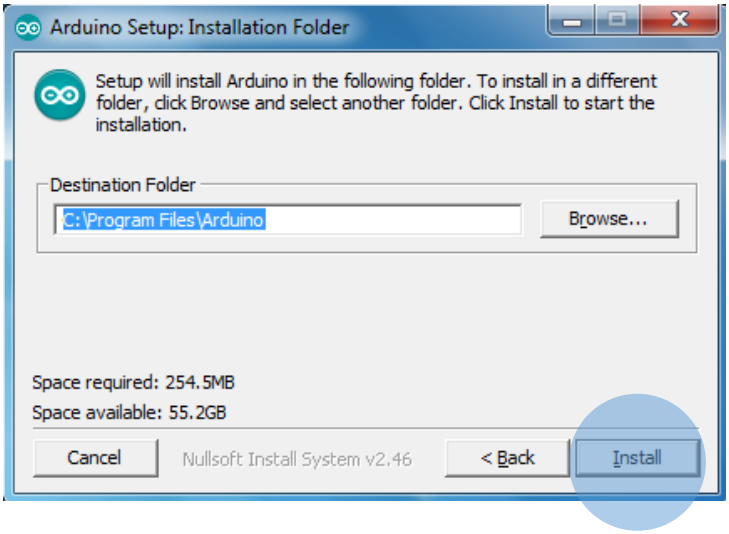

Figura A.4. Terceiro passo da instalação do Arduino Duemilanove.

Feita a verificação do caminho onde será instalado e do espaço livre em disco suficiente para a instalação, procede-se à instrução de instalação através do botão "Install". O tempo do processo de instalação varia conforme as especificidades de cada computador pelo que se exige um tempo de espera variável.

Terminada a instalação é tempo para proceder ao encerramento deste ponto e passar para o ponto 2).

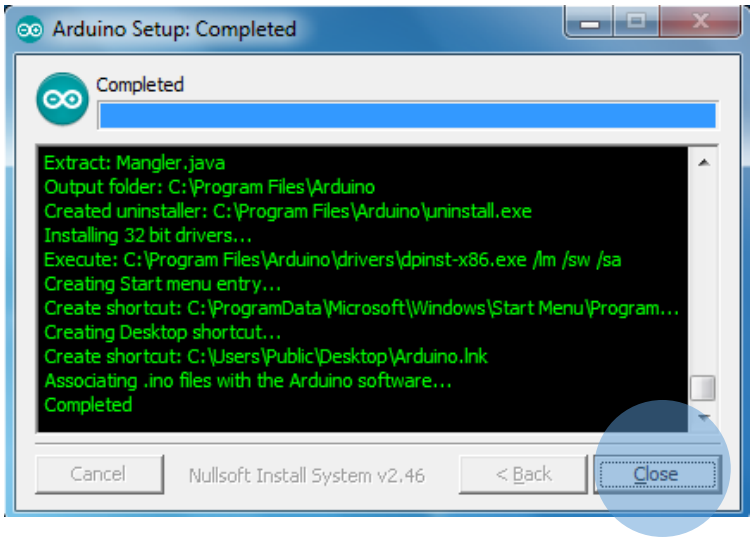

Figura A.5. Quarto passo da instalação do Arduino Duemilanove.

2) Ligar a placa ao computador via USB:

Após ligar a placa Arduino ao computador, este irá proceder à verificação do *hardware* e a instalação não será realizada com êxito. Assim que a verificação terminar observe apenas o aviso que aparecerá no canto inferior direito do monitor que irá dar conta da falha de instalação do *hardware* e siga para o ponto 3).

No caso de não aparecer nenhum aviso no monitor – possível desativação das notificações por parte da configuração existente no Windows – avance igualmente para o ponto 3).

3) Ir a gestor de dispositivos:

A partir daqui, conforme a Figura A.6, iremos aceder ao gestor de dispositivos do Windows para que possamos ver em que ponto de situação está a instalação do nosso *hardware*.

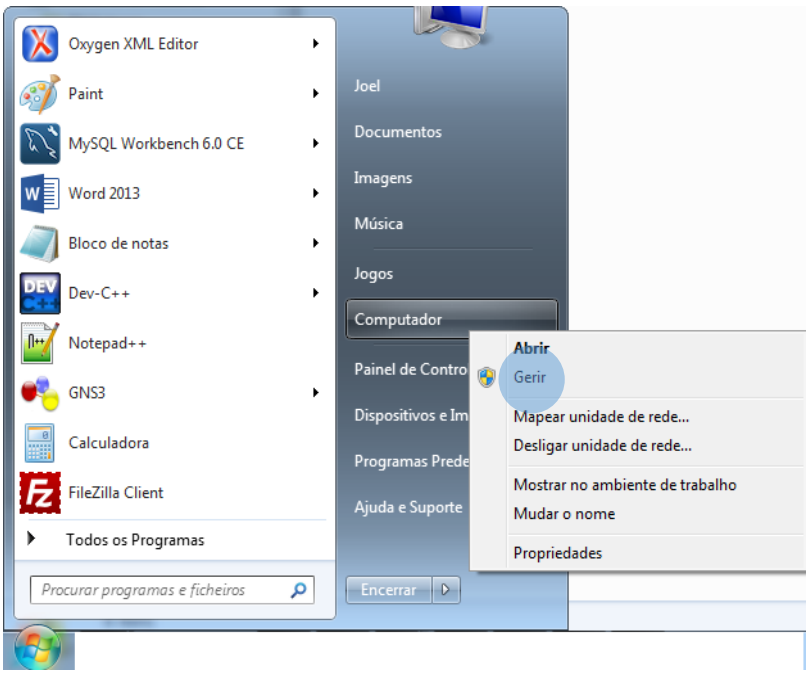

Figura A.6. Acesso ao gestor de dispositivos.

Sem nenhuma surpresa estará incorretamente instalado dada a não associação direta com os *drivers* FTDI USB da pasta do *software* Arduino. Após acedermos ao gestor de dispositivos passemos para o ponto 4).

4) Atualizar controlador de Software:

Já na janela de gestão de dispositivos é fácil verificar que a instalação não foi efetivamente realizada com sucesso por parte do sistema operativo e é necessário proceder manualmente à sua correta instalação. Com a ajuda visual da Figura A.7 passemos à atualização do controlador de Software.

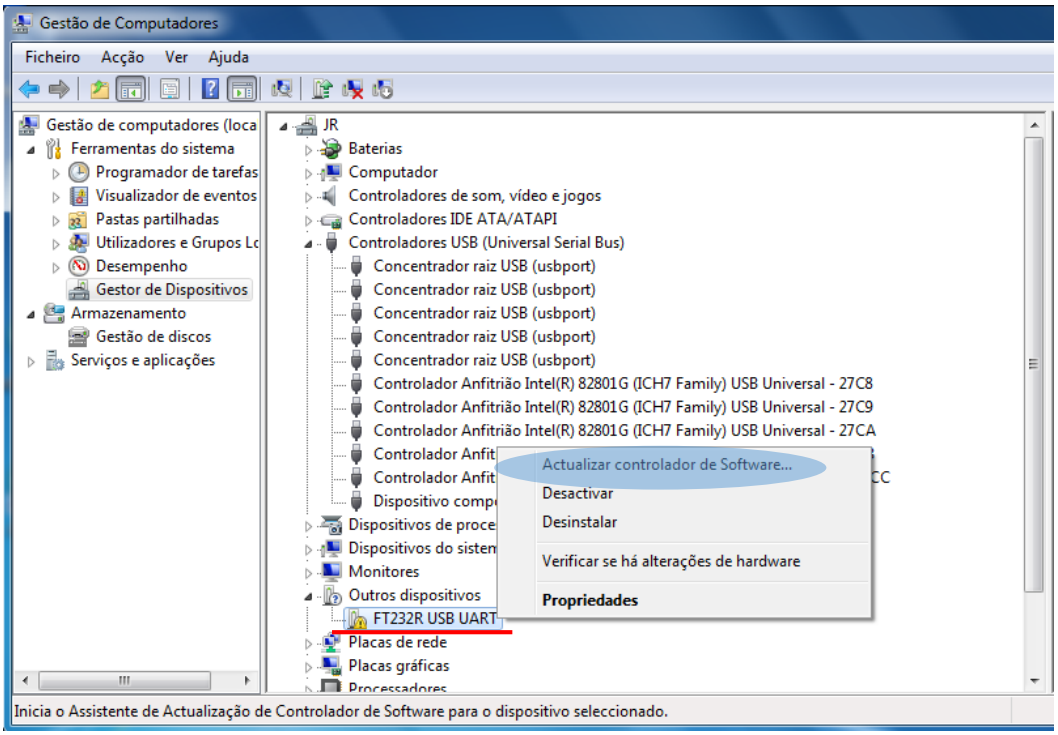

Figura A.7. Atualização do controlador para comunicação com o Arduino.

Posteriormente a isto temos de escolher a opção de "Procurar *software* do controlador no computador" para podermos escolher a pasta local onde temos os *drivers* a instalar.

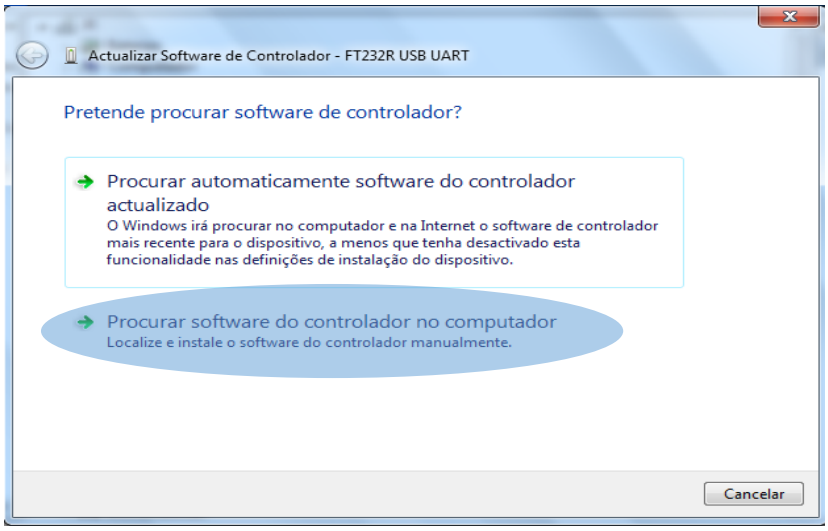

Figura A.8. Procurar software do controlador a instalar.

Após seguir à risca os passos anteriores, a localização a indicar para procura do *software* a instalar será o mesmo que está na Figura A.9. De qualquer forma será mais assertivo proceder à procura manual da localização de instalação do software Arduino e ir descendo na árvore de diretorias até chegar à pasta "FDTI USB Drivers".

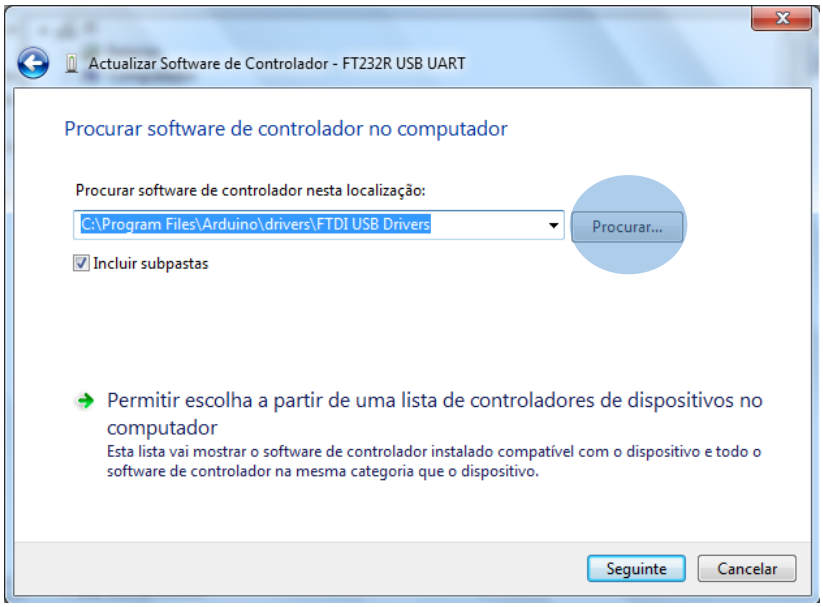

Figura A.9. Escolha do caminho onde os *drivers* estão guardados.

Realizada a instalação recebemos um aviso que esta foi realizada com êxito e posteriormente é possível verificar que no gestor de dispositivos foi associado um

novo controlador USB juntamente com uma USB porta série virtual com a descrição COM3.

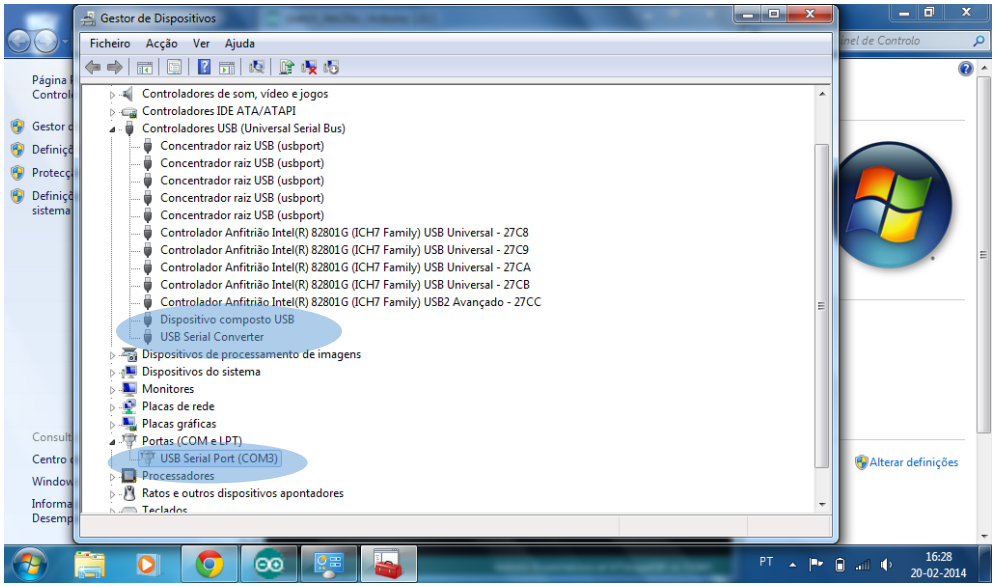

Figura A.10. Verificação da instalação do Arduino.

I.2. Ligação entre Arduino e módulo Bluetooth

Para que as ligações sejam bem realizadas existem requisitos que têm de ser cumpridos. De maneira a facilitar a visualização como estas devem ser feitas, tenhamos em consideração o esquema da Figura A.11.

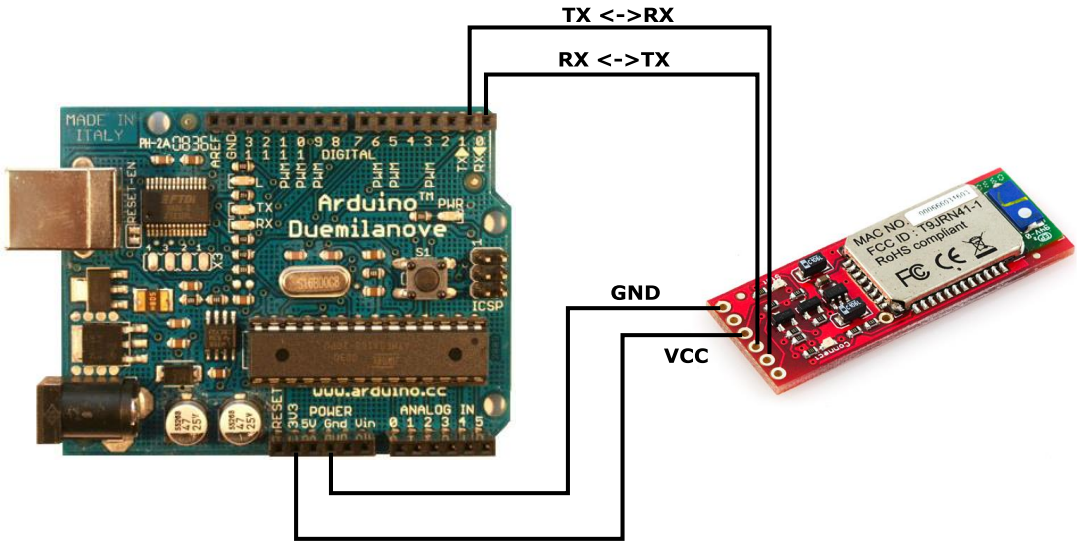

Figura A.11. Esquema de ligação entre o Arduino e o módulo WRL-12580.

I.2.1. Instalação do módulo Bluetooth WRL-12580 no Windows

Neste ponto já temos o Arduino ligado fisicamente ao módulo Bluetooth e podemos proceder à sua instalação deste último no computador. Começamos por ligar o cabo USB do Arduino ao computador e verificamos se o módulo Bluetooth tem a luz vermelha do LED *status* intermitente. Em caso afirmativo constatamos que temos as ligações bem realizadas, em caso negativo devemos rever novamente as ligações especificadas anteriormente na Figura A.11.

Posto isto estamos prontos para procedermos à instalação do módulo Bluetooth. Como primeira tarefa devemos verificar se temos o *software* do módulo Bluetooth do próprio computador bem instalado. Neste caso específico, está bem instalado e já se encontra em execução no canto inferior direito do Windows, mais propriamente na área de notificações. Conforme a Figura A.12, acedendo ao respetivo ícone, podemos proceder à deteção de dispositivos com interface Bluetooth em redor do nosso computador pressionando o botão direito sobre o ícone e escolhendo a opção "Adicionar um Dispositivo".

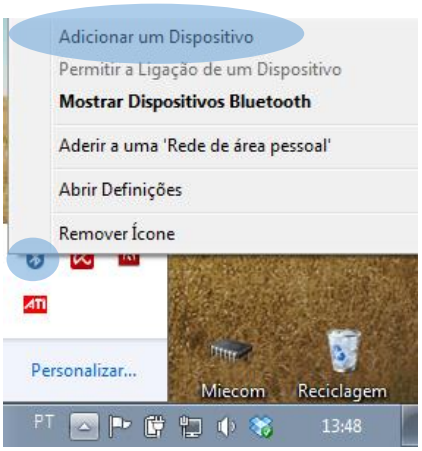

Figura A.12. Adicionar um dispositivo Bluetooth ao computador.

De seguida é iniciada a procura de dispositivos Bluetooth na área de abrangência. Verificados quais os dispositivos detetados temos de escolher o que tiver o nome "FireFly-XXXX" (onde XXXX varia de dispositivo para dispositivo e representa os últimos quatro *nibbles* do endereço MAC do dispositivo Bluetooth) e carregar em "Seguinte".

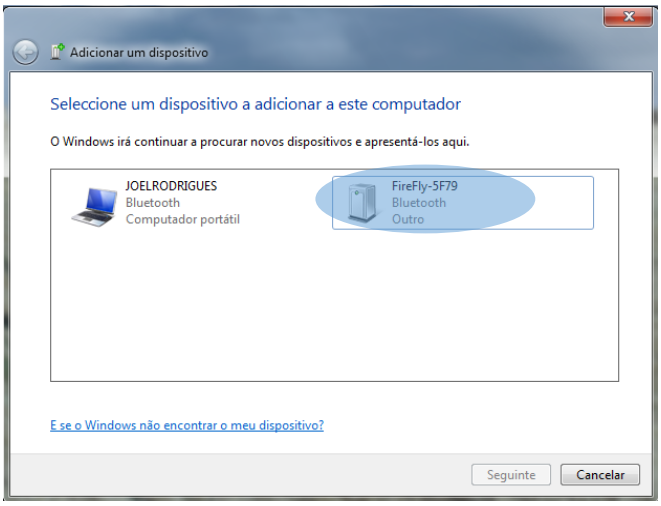

Figura A.13. Escolha do dispositivo Bluetooth a adicionar ao computador.

Temos agora de escolher qual a opção de emparelhamento que desejamos realizar com o nosso módulo. No caso do módulo Bluetooth WRL-12580, este vem definido com um código de segurança para que seja realizado o emparelhamento. Assim é necessário escolher a opção "Introduzir o código de emparelhamento do dispositivo" para prosseguir na sua correta instalação.

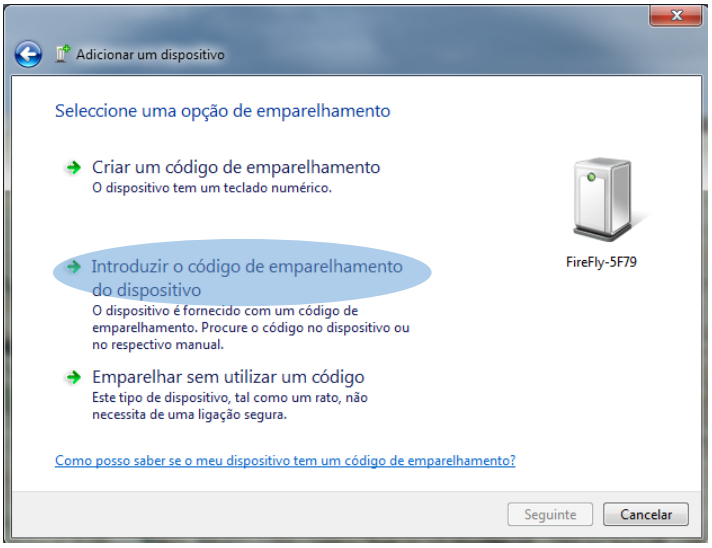

Figura A.14. Emparelhar o dispositivo com base no seu código.

O código de emparelhamento do dispositivo é o "1234" (fornecido pelo fabricante) e à frente veremos como o podemos consultar. Para prosseguir temos de digitar o código e escolher a opção "Seguinte".

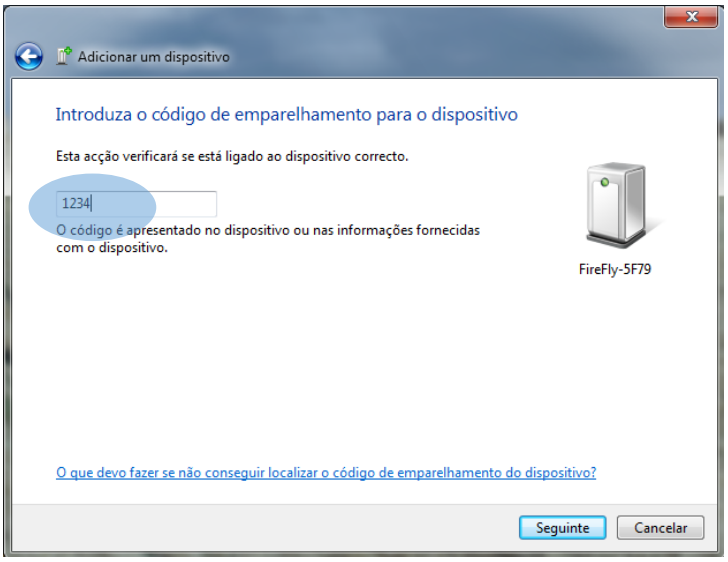

Figura A.15. Introdução do código de emparelhamento.

Se tudo foi bem realizado aparecerá uma mensagem de finalização de configuração que indicará que o dispositivo foi adicionado com êxito ao computador.

Para que este dispositivo funcione corretamente é necessário que fique associado a uma porta série COM virtual. Essa associação é realizada através do SPP (Serial Port Profile) que consiste na emulação da comunicação cablada através de cabo de série em comunicação sem fios Bluetooth. Para realizarmos essa emulação temos de aceder aos dispositivos já instalados no computador.

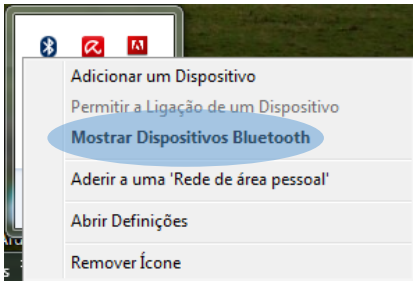

Figura A.16. Aceder aos dispositivos Bluetooth instalados.

De seguida carregamos com o botão direito sobre o dispositivo referente ao módulo Bluetooth (FireFly-5F79) e escolher a opção “Propriedades”.

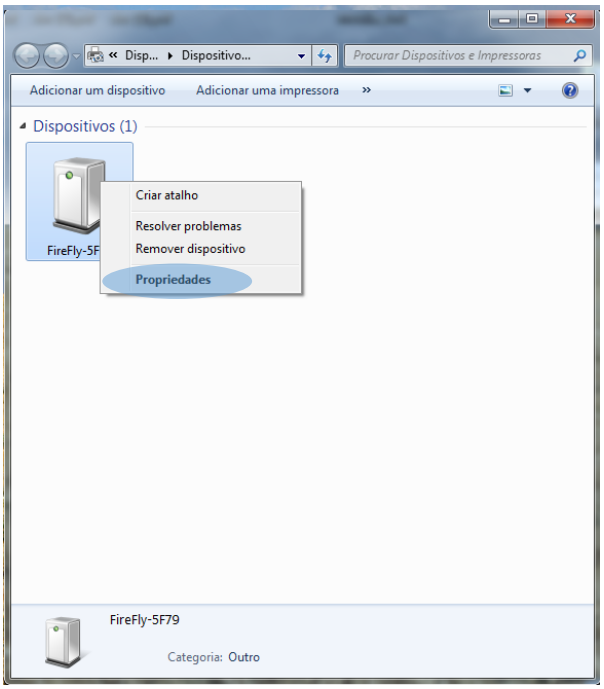

Figura A.17. Acesso às propriedades do dispositivo Bluetooth.

Seguimos para o separador "Serviços" e esperamos até que seja atribuída uma porta série virtual ao dispositivo. Neste caso é bem visível pela Figura A.18 que a porta série atribuída pelo sistema operativo foi a "COM4".

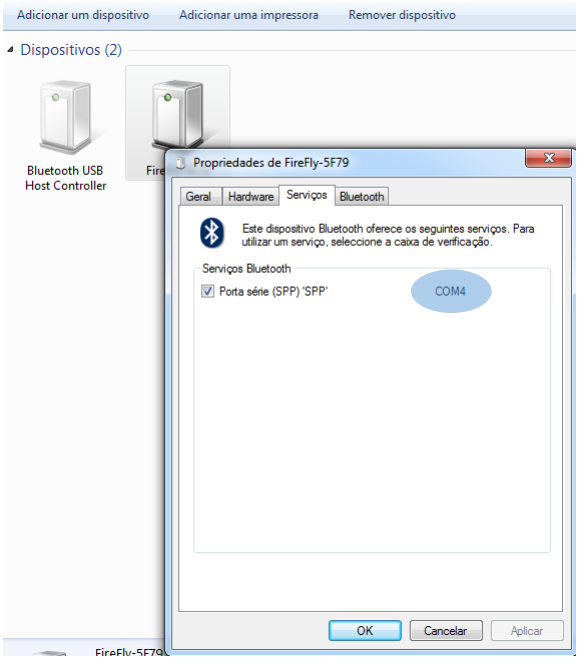

Figura A.18. Associação de uma Porta Série ao Bluetooth.

Para concluir todo o processo e verificar que o módulo foi corretamente instalado, acedemos novamente ao gestor de dispositivos e para além da porta série instalada para o Arduino (COM3), temos de ter agora uma outra referente ao módulo Bluetooth (COM4).

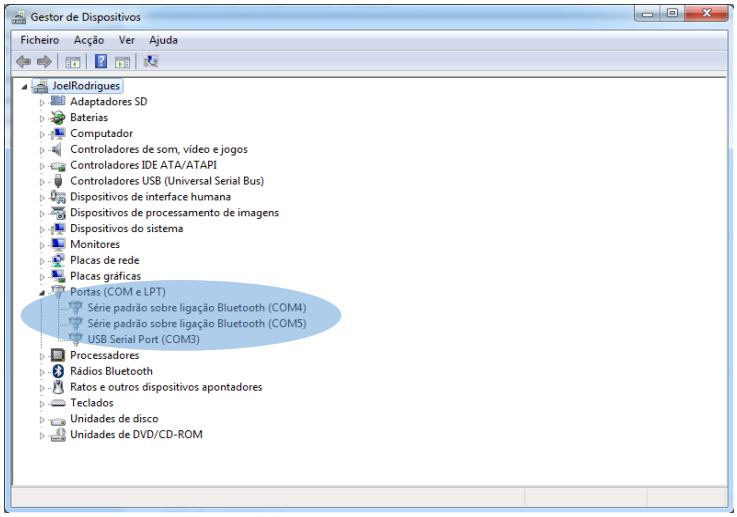

Figura A.19. Portas series USB e Bluetooth

I.3. Arduino e módulo Bluetooth em modo cooperativo

Realizados todos os procedimentos anteriores relativos à configuração do Arduino Duemilanove, do módulo Bluetooth WRL-12580 e das ligações entre ambos, é importante estarmos familiarizados com o ambiente gráfico que iremos utilizar para procedermos à programação do *software* a introduzir no microcontrolador. É apresentado de seguida esse ambiente de desenvolvimento do Arduino.

I.3.1. Arduino IDE

O ambiente de desenvolvimento onde será realizada a programação do Arduino é bastante intuitivo e contém todos os recursos necessários à programação do microcontrolador. Vejamos na Figura A.20 o seu interface gráfico e, de seguida, na Tabela A.2 a descrição de algumas funcionalidades imprescindíveis à inicialização de programação do Arduino.

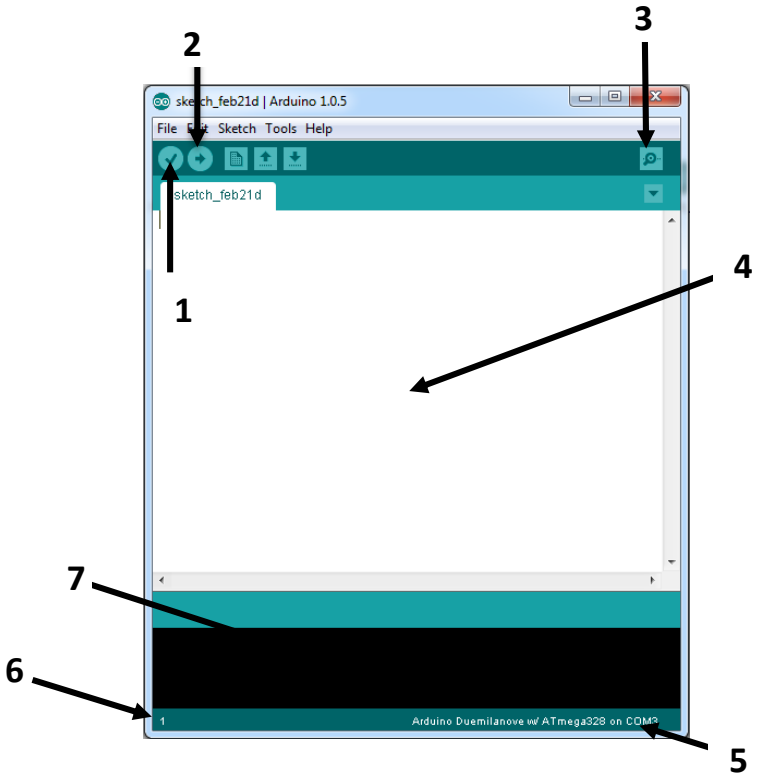

Figura A.20. IDE do Arduino.

Tabela A.2. Descrição relativa à Figura A.20.

Número	Descrição
1	Verify - Verificação da sintaxe do código
2	Upload – Reprogramação do microcontrolador
3	Serial Monitor – Comunicação direta com o módulo Bluetooth
4	Área de escrita do código
5	Plataforma Arduino que está configurada
6	Número da linha atual do código
7	Informação de debug

Após estas indicações, é imprescindível realizar a configuração associativa entre o *software* de programação Arduino e o *hardware* que temos disponível.

Assim, acedendo ao Arduino IDE através da diretoria onde este está instalado - ou de um atalho criado posteriormente à instalação - iremos começar por definir qual a plataforma que vamos usar. Neste caso será o Arduino Duemilanove com um microcontrolador ATmega328.

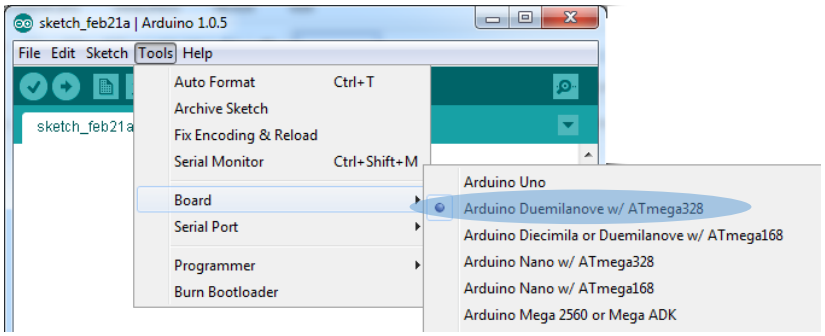

Figura A.21. Associação entre hardware e Arduino IDE.

É importante também saber se o Arduino IDE realizou um correto reconhecimento às portas série virtuais que configuramos anteriormente, para isso aconselha-se a realização da operação da Figura A.22. De realçar apenas que, neste caso, temos três portas série reconhecidas:

- **COM3**: relativa à emulação da ligação USB do Arduino em Porta série;
- **COM4**: corresponde à virtualização da comunicação série em comunicação Bluetooth do módulo WRL-12580;
- **COM5**: corresponde a uma ligação Bluetooth anteriormente configurada (a não ter em consideração para este tutorial).

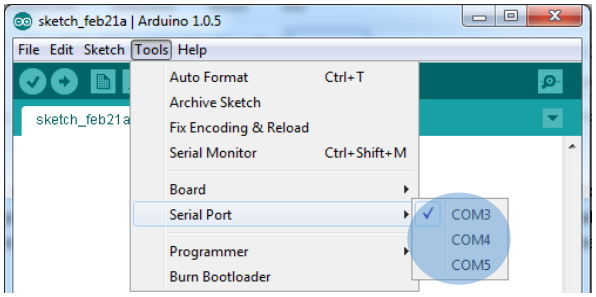

Figura A.22. Reconhecimento das portas série pelo Arduino IDE.

I.3.2. Modo Comando no módulo Bluetooth

Uma das muitas valias que o Arduino IDE proporciona ao utilizador é a disponibilização de uma ferramenta de comunicação direta com o módulo Bluetooth sem que tenhamos de adotar um outro *software* para conseguirmos estabelecer comunicação. Nesta fase será importante realizar uma visualização das

configurações existentes no módulo Bluetooth para que possamos proceder às corretas definições de parâmetros de comunicação no *software* cliente.

Em primeiro lugar é importante proceder à alteração da porta série de comunicação no Arduino IDE para que a comunicação se realize via Bluetooth. Neste caso específico a virtualização do módulo Bluetooth em porta série tem como descrição COM4.

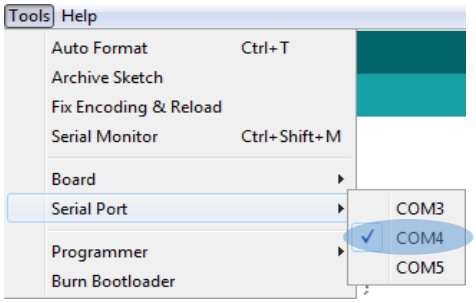

Figura A.23. Definição de comunicação sem fios Bluetooth (COM4).

Acedemos agora à ferramenta "*Serial Monitor*", digitamos "$$$" na caixa de inserção de texto, verificamos se o envio de dados está como "*no line ending*" e carregamos em "*Send*".

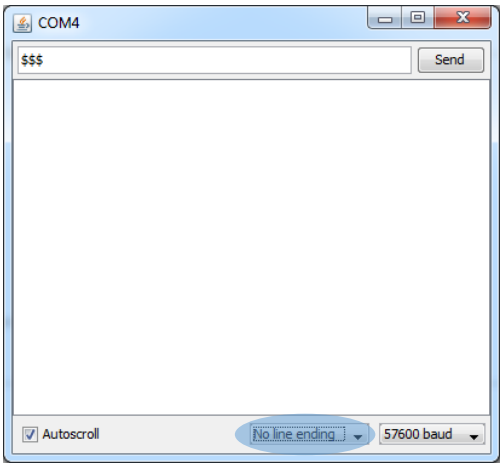

Figura A.24. Instrução de acesso ao Bluetooth em modo Comando.

Após esta operação será visível a receção de uma mensagem "CMD" que informa que estamos acedendo ao módulo Bluetooth no modo de comando. Posteriormente, podemos realizar operações de visualização de configurações internas. Como conselho de utilização sugere-se que se envie o comando "*h*" no

modo "*Newline*" para visualizarmos quais as operações que o módulo suporta e escolhermos a que pretendemos.

No que diz respeito às operações que podemos realizar, estas podem ser de três tipos:

- Definição de comandos;
- Visualização;
- Outros.

Como forma exemplificativa de utilização de uma destas operações, vejamos como estão as configurações básicas do nosso módulo. Para isso, e recorrendo às opções de visualização disponibilizadas pelo comando "h", digitamos o comando "D" (referente às definições básicas do módulo Bluetooth) e enviamos.

A resposta ao pedido está visível na Figura A.25 e destaca-se especial interesse para o valor definido no nome de Bluetooth (*BTName*), no *baud rate* definido (*Baudrt*), no modo de operação (*Mode*) e no código PIN (*Pin Code*) que introduzimos aquando da sua configuração.

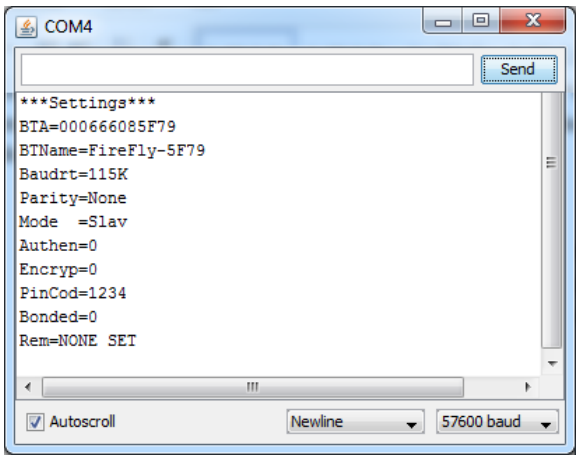

Figura A.25. Configurações atuais do módulo Bluetooth.

No caso de desejarmos alterar qualquer parâmetro é neste modo que o devemos fazer, aconselhando-se a consulta dos métodos de alteração através do menu disponibilizado pela opção "h". Após a alteração do parâmetro pretendido digita-se "---" como instrução para guardar as alterações realizadas.

I.4. Primeira aplicação demonstrativa

Para demonstração de operação conjunta entre todos os componentes analisados anteriormente irá ser explicado como se processa a reconfiguração do *software* que está a ser executado no microcontrolador do Arduino e como se processa a sua comunicação com um software cliente externo com base na tecnologia sem fios Bluetooth.

Para realizarmos esta tarefa necessitaremos de:

- Arduino Duemilanove;
- Módulo Bluetooth WRL-12580;
- Cabo USB 2.0;
- Arduino IDE.

O objetivo final deste primeiro *software* é controlar o LED 13 (ligar e desligar) do Arduino através de uma aplicação cliente via comunicação Bluetooth.

I.4.1. Programar microcontrolador

Para não desviarmos o foco essencial deste tutorial iremos ver de seguida um exemplo de um programa em linguagem Wiring, onde este será explicado de uma forma sucinta.

Para que exista alguma base de como o código está estruturado tenhamos em consideração que todo o código Arduino tem as seguintes funções:

- **setup()**:é executada sempre que o Arduino se liga e serve essencialmente para inicializar variáveis, librarias e pinos. Tem como particularidade ser executada uma vez por cada utilização, até que o módulo seja novamente reiniciado;
- **loop()**: é processada após o setup() e tudo que se encontra dentro desta função é executado de forma cíclica. A sua particularidade é que a sua execução não termina enquanto o Arduino estiver ativo. Podemos pensar nela como sendo uma condição “while(1);”.

Quando temos um *software* já desenvolvido e o queremos passar para o microcontrolador, não é possível fazê-lo através da comunicação Bluetooth (COM4). Para isso temos de proceder ao ajuste da porta série para a COM referente à comunicação direta via USB com o Arduino (COM3).

É estritamente necessário que a ligação TX-RX entre o Arduino e o Bluetooth esteja desligada, procedendo-se novamente à sua ligação somente quando estiver realizada a programação do microcontrolador.

Por fim, procedemos à programação do microcontrolador com recurso ao botão "*Upload*" do IDE do Arduino. Podemos verificar os LEDs de TX e TX do módulo Arduino a piscar e quando este estiver terminado aparecerá uma informação de sucesso na janela de debug do IDE do Arduino.

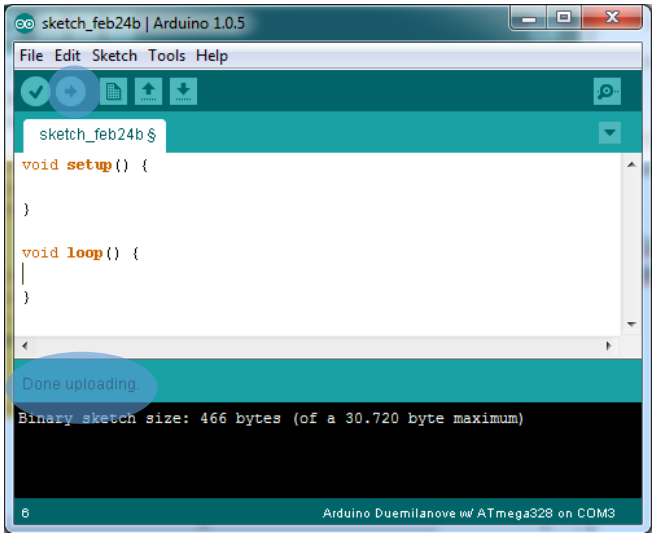

Figura A.26. Forma de programar o microcontrolador pelo Arduino IDE.

I.4.2 Software Arduino

O *software* que estará em execução no Arduino terá como função acender e apagar o LED 13 conforme a instrução que chegue via Bluetooth ao módulo por parte do software cliente. Caso chegue o valor ASCII '0' é realizada a operação de desligar o LED 13 e no caso de chegar o ASCII '1' o LED 13 é ligado.

Para que toda a comunicação seja realizada corretamente e de forma coerente entre o Arduino e o módulo Bluetooth é necessário configurar o microcontrolador

para o mesmo *baud rate* que esteja configurado na EEPROM do módulo Bluetooth. Neste caso irá ser definido a 115200 bps.

Escrito o código da Figura A.27, procede-se à programação do microcontrolador conforme os procedimentos descritos no ponto anterior.

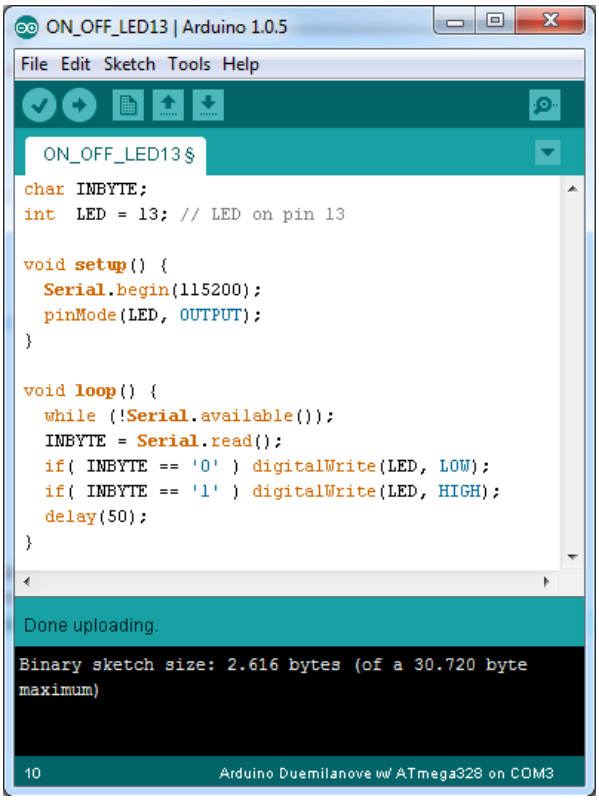

Figura A.27. Código exemplo para programar o microcontrolador Arduino.

I.4.3. Software Cliente

O *software* cliente terá de seguir igualmente as mesmas configurações que estão estabelecidas no módulo Bluetooth. O *baud rate* terá de ser o mesmo sob risco de – no caso de não terem as mesmas configurações – a comunicação não conseguir ser realizada corretamente. Para garantir que a comunicação seja bem efetuada é aconselhável que o *software* cliente cumpra as configurações que estão estabelecidas na Figura A.28.

Para além do módulo da main.c temos os módulos CCom.cpp e CCom.h que não estão especificados aqui. Contudo, é possível usar bibliotecas alternativas a esta para proceder ao envio e receção via porta série desde que as variáveis de dados sejam

do mesmo tipo. Todavia, aconselham-se estas bibliotecas de comunicação que estão disponíveis para uso livre em https://mega.co.nz/#!Es4XwaLR!nUqdVSRv2n5mkP7m-iWh6tLE0k04gfVDYFwaF7fttOQ.

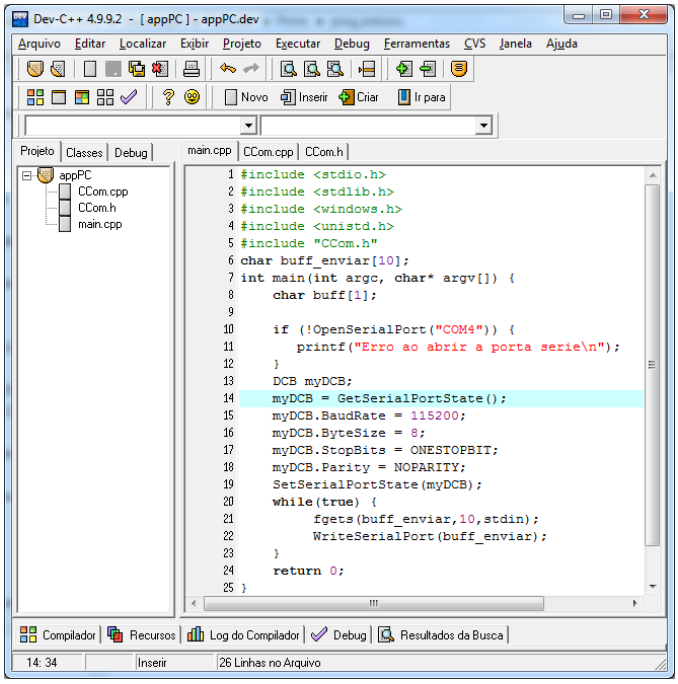

Figura A.28. Software cliente para comunicação com Arduino.

Colocando este *software* em execução e adaptando as configurações da porta série ao nosso hardware Bluetooth, o LED de conexão do módulo Bluetooth deve ficar verde (informação que está bem conectado ao software cliente). Além disso, se enviarmos o caracter '0' através do *software* cliente o LED 13 deve apagar e se enviarmos '1' o LED 13 deve acender. No caso de não estar a funcionar deste modo, alguma coisa estará mal configurada, pelo que será necessário proceder à sua análise e respetiva correção.