Universidade do Minho
Escola de Engenharia

Guilherme Gonçalves Nogueira Machado

Sistema de comunicação BLE para sondas neuronais

Dissertação de Mestrado
Mestrado em Engenharia Eletrónica Industrial e de Computadores

Trabalho efetuado sob a orientação do
Doutor João Ribeiro

Outubro de 2018

DECLARAÇÃO

Nome: Guilherme Gonçalves Nogueira Machado

Endereço eletrónico: guillhermemachado@outlook.com

Telefone: 969555106

Cartão do Cidadão: 14050125

Título da dissertação: Sistema de comunicação BLE para sondas neuronais

Orientador:

Doutor João Ribeiro

Ano de conclusão: 2018

Mestrado em Engenharia Eletrónica Industrial e de Computadores

DE ACORDO COM A LEGISLAÇÃO EM VIGOR, NÃO É PERMITIDA A REPRODUÇÃO DE QUALQUER PARTE DESTA TESE/TRABALHO.

Universidade do Minho, _____/_____/_____

Assinatura:

AGRADECIMENTOS

O processo de desenvolvimento e escrita desta dissertação foi, sem dúvida, a etapa mais difícil e desafiante no meu percurso académico. Ao longo desta viagem, foram vários os problemas e desafios que tive de ultrapassar, o que me fez crescer tanto a nível académico como pessoal. Aproveito assim para deixar o meu agradecimento a múltiplas pessoas que tornaram toda este desafio possível.

Ao meu orientador, doutor João Ribeiro, por toda a orientação prestada, disponibilidade, por toda a ajuda ao longo da dissertação e me ter ajudado sempre que precisei de ver as coisas de outro ponto de vista.

À minha namorada e acima de tudo melhor amiga, Mariana, por todo o apoio, toda a ajuda, por sempre me ajudar a ultrapassar todos os problemas e por se manter sempre ao meu lado ao longo destes anos.

Aos meus amigos, que me acompanharam nesta viagem que é a Universidade, por todos os momentos de diversão e por toda a ajuda nos estudos, a todos um muito obrigado.

Às pessoas mais importantes da minha vida, os meus pais, irmãos e avós, obrigado por toda a ajuda e por terem sempre acreditado em mim, sem eles nada disto seria possível.

Por último, quero dedicar esta dissertação, a minha avó Maria, ao meu avó João e ao meu avó Agostinho, todos eles me viram iniciar esta caminhada, mas infelizmente nenhum deles se encontra presente para me ver a completá-la. Acredito que, onde quer que estejam, estão muito orgulhosos de mim.

A todos o meu sincero obrigado, por toda a ajuda e encorajamento. Sem vocês, esta dissertação não seria possível.

RESUMO

O cérebro é um órgão do corpo humano bastante complexo, que contém cerca de 100 biliões de neurónios que comunicam entre si na partilha de informação. Todo este sistema de comunicação permite ao cérebro comunicar com outras partes do corpo, influenciando assim todos os sistemas nele existente, tornando este órgão essencial para o nosso funcionamento. Os neurocientistas estudam a atividade cerebral, o seu comportamento em determinados ambientes e os seus processos de funcionamento. Os estudos são frequentemente efetuados em ratos, num ambiente controlado e com recurso a sondas neuronais invasivas. Com o intuito de potenciar o estudo do cérebro e melhorar o seu aproveitamento é essencial que o animal tenha a maior liberdade de movimentos possível, ou seja, que não existam cabos a atrapalhar e que todo o sistema seja leve e compacto.

O objetivo desta dissertação é desenvolver um sistema de comunicação sem fios Bluetooth Low Energy (BLE) projetado para pequenos animais, que lê os sinais das sondas neuronais em tempo real e os transmite para uma interface gráfica desenvolvida em C#. O sistema de comunicação será bidirecional para permitir através da interface gráfica a ativação da estimulação optogenética.

O sistema é composto pela *headstage* da Intan RHS2116 que trata da aquisição dos sinais eletrofisiológicos, da sua amplificação e digitalização. O envio dos sinais convertidos será efetuado através do módulo Programmable System-On-Chip (PSoC) 4 BLE que os irá receber, converter e enviar para o PC onde serão mostrados numa interface gráfica e gravados num ficheiro de texto. Para além do processamento dos sinais, a *headstage* da Intan RHS2116 é utilizada para atuar um díodo emissor de luz (LED) para a estimulação optogenética.

A validação do sistema foi realizada utilizando um sinal de um gerador de sinais na gama dos µV (gama dos sinais eletrofisiológicos). O sistema desenvolvido atinge as 550 *Samples/s* (550 Hz) com um canal ativo e 41 *Samples/s* (41 Hz) por canal com os 16 canais ativos. A estimulação optogenética (atuação do LED) funciona com dois modos, dando liberdade de escolha ao utilizador e grava o sinal no mesmo ficheiro de texto que os sinais eletrofisiológicos para facilitar a análise dos resultados. Teoricamente, o sistema consume 18.3 mA, por isso, com uma pilha de 120 mAh, o sistema pode operar durante ≈ 6.5 horas e funciona até 15 metros de distância.

Palavras-Chave: Bluetooth Low Energy; Cérebro; Comunicação sem fios; Liberdade de movimentos; Optogenética; Sondas neuronais.

ABSTRACT

The brain is a complex human body organ that contains about 100 billion neurons, sharing information and communicating with each other. All this communication system allows the brain to communicate with other parts of the body, influencing all the systems existing in it and making the brain essential to our functioning. Neuroscientists study brain activity, his behaviour in certain environments and his operating processes. These studies are often performed in rats with a controlled environment and using neural invasive probes. In order to empower the brain study and improve its accuracy, it's essential that the animal has freedom of movement, meaning, no cables to disturb and a light and compact system.

The objective of this dissertation is to develop a Bluetooth Low Energy (BLE) communication system designed for small animals, which reads the signals from the neural probes in real time and transmits them to a graphical interface developed in C#. The communication system will be bidirectional to allow the optogenetic stimulation.

The system contains an Intan RHS2116 headstage that deals with the acquisition, amplification and digitization of the electrophysiological signals. Converted signals will be send by the PSoC 4 BLE module that will receive the converted signals and send them to the PC where they will be shown in a graphical interface and recorded in a text file. In addition to signal processing, the Intan RHS2116 headstage is used to operate a light emitting diode (LED) for optogenetic stimulation.

The system validation was achieved using a sign form generator in conjunction with a voltage divider to have a signal in the μV range (electrophysiological signals range). The developed system reaches the 550 Samples/s (550 Hz) with one active channel and 41 Samples/s (41 Hz) per channel with the 16 active channels. Optogenetic stimulation (LED actuation) operates in two modes, giving the user freedom of choice and recording the signal in the same text file as the electrophysiological signals to facilitate the results analysis. Theoretically, the system consumes 18.3 mA and powered by 3.6 V. When powering the system with a lithium-ion battery, model LiR2450 that has a capacity of 120 mAh, theoretically, the system will have a duration of ≈ 6.5 hours and works up to 15 meter distance.

Keywords: Bluetooth Low-Energy; Brain; Wireless Communication; Freedom of Movement; Optogenetic; Neural Probes

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

Lista de acrónimos	Português	Inglês
ADC	Conversor analógico digital	Analog Digital Converter
ATT	Protocolo de atributo	Attribute Protocol
BLE	-	Bluetooth Low Energy
CMOS	Semicondutor de metal-óxido complementar	Complementary Metal-Oxide-Semiconductor
CPU	Unidade de processamento Central	Central Process Unit
CS	Seleção de chip	Chip Select
ECO	Oscilador de cristal externo	External crystal oscillator
ECoG	Eletrocorticografia	Electrocorticography
EEG	Eletroencefalografia	Eletroencephalography
FCD	Função de Coordenação Distribuída	-
FCP	Função de Coordenação de Pontos	-
fMRI	Ressonância Magnética Funcional	Functional magnetic resonance imaging
GAP	-	Generic Access Profile
GATT	-	Generic Attribute Profile
IC	Circuito Integrado	Integrated Circuit
IDE	Ambiente de Desenvolvimento Integrado	Integrated development environment
ILO	Oscilador interno de baixa velocidade	Internal low-speed oscillator
IMO	Oscilador principal interno	Internal main oscillator
IoT	Internet das coisas	Internet of Things
ISM	Industrial, Científica, Médica	Industrial, scientific and medical

LED	Díodo Emissor de Luz	Light-emitting diode
LVDS	Sinalização diferencial de baixa tensão	Low Voltage Differential Signaling
L2CAP	-	Logical Link Control and Adaptation Protocol
MCU	Microcontrolador	Microcontroller unit
MEG	Magneto Encefalografia	Magnetoencephalography
MISO	-	Master In Slave Out
MOSI	-	Master Out Slave In
PCB	Placa de circuito impresso	Printed circuit board
PDA	Assistente pessoal digital	Personal digital assistant
PET	Tomografia por Emissão de Positrões	Positron-emission tomography
PSoC	-	Programmable System-On-Chip
PWM	Modulação por largura de pulso	Pulse Width Modulation
RF	Radiofrequência	Radio frequency
SCLK	-	Serial Data Clock
SIG	Grupo Especial da Indústria	Special Interest Group
SMP	-	Security Manager Protocol
SPI	-	Serial Peripheral Interface
UART	-	Universal Asynchronous Receiver Transmitter
WCO	-	Watch crystal oscillator

INTRODUÇÃO

Neste capítulo serão apresentados os motivos para a realização desta dissertação, bem como os objetivos a serem atingidos e os contornos da organização estrutural do documento. São também introduzidos conceitos teóricos de neurónios, métodos de gravação neuronal, sondas neuronais e optogenética.

Esta dissertação tem como objetivo o desenvolvimento de um sistema de comunicação sem fios BLE para sondas neuronais invasivas, estando inserida no projeto de investigação *Brain Lighting*.

Motivação

Ao longo dos anos e com o avançar da tecnologia o Homem conseguiu descobrir mais sobre o seu corpo e o seu funcionamento enquanto um todo. Mas apesar de toda a evolução, houve um órgão cujo conhecimento permaneceu vago ao longo do tempo, o cérebro. Descobrir os princípios que governam as operações do cérebro não só proporcionará enormes benefícios no campo da medicina, mas também poderá explicar o que nos torna humanos [1]. Assim, de forma a aumentar o conhecimento existente e descobrir os seus mistérios escondidos foi criado o projeto *Brain Lighting*, uma parceria entre a Universidade do Minho, a Fundação Champalimaud e a Inova+.

O projeto *Brain Lighting* pretende desenvolver uma solução tecnológica que tem como objetivo contribuir para a evolução da neurociência, aumentando o conhecimento sobre a atividade cerebral e a análise de comportamentos em função da reação a estímulos externos. A solução tecnológica visa o desenvolvimento de um dispositivo optogenético para estimulação profunda do cérebro utilizando sondas neuronais 3D dotadas de estimulação ótica e gravação elétrica. A optogenética foi um termo utilizado pela primeira vez por um grupo de investigação da Universidade de Stanford em 2005 para designar a técnica que combina ótica com a genética, possibilitando assim o controlo de eventos bem definidos dentro das células-alvo [2].

A presente dissertação tem como objetivo o desenvolvimento de um módulo de comunicação sem fios, utilizando a tecnologia BLE e um *software* de visualização de sinais obtidos através de uma sonda neuronal em desenvolvimento no projeto de investigação *Brain Lighting*. Um dos maiores desafios deste projeto centra-se em enviar a informação para o *software* em tempo real sem a necessidade de fios, eliminando assim diversos problemas

inerentes à utilização dos mesmos para a comunicação entre as sondas neuronais e o *software*. Assim, sem a utilização de cabos, os animais testados podem andar livremente, sem restrições de movimento. No caso dos sistemas com fios, a locomoção dos animais pode ficar restringida e por vezes acabam mesmo por entrelaçar os fios, desconectando-os e fazendo com que sejam perdidos dados de várias horas/dias de investigação [3].

Objetivos e tarefas

Na figura 1 é mostrado um diagrama de um sistema típico de gravação neural sem fios. Os sensores neurais capturam os sinais dos neurónios. De seguida é amplificado e digitalizado o sinal ao mesmo tempo que rejeita o ruído eletrónico. O sinal digitalizado é processado e transmitido para o computador sendo visualizado posteriormente numa interface gráfica. O módulo sem-fios para além de ser responsável pela transmissão dos sinais para a interface gráfica, também é o responsável pela receção de comandos que ativam ou desativam a estimulação optogenética. Esta dissertação centra-se no desenvolvimento de todo o sistema de gravação/visualização dos sinais eletrofisiológicos.

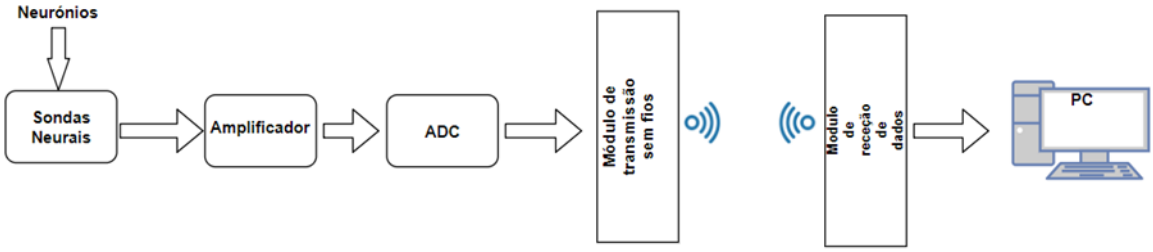


Figura 1-Diagrama típico de um sistema de comunicação sem fios para gravação neuronal.

De seguida encontram-se os pontos a serem explorados ao longo do projeto:

- Estudo da bibliografia e dos materiais a serem utilizados;
- Desenvolvimento do sistema bidirecional de comunicação BLE usando o módulo PSoC 4 BLE;
- Integração do sistema de comunicação BLE com a *headstage* da Intan RHS2116;
- Elaboração de uma interface gráfica para a aquisição/gravação dos dados;
- Escrita da dissertação.

Neurónios

O neurónio é um tipo de célula especial pois é capaz de ser excitada eletricamente processando informação de sinais elétricos e químicos. Existem cerca de 10 biliões de neurónios num cérebro humano [5]. Um típico neurónio incluí o corpo celular, dendrites e o axônio, como mostrado na figura 2. As dendrites podem ter imensos "galhos" com centenas de micrómetros de comprimentos, enquanto um axônio de um neurónio humano chega até a um metro de comprido. Um canal de iões é formado nas membranas da célula neuronal com iões como sódio, potássio e cálcio.

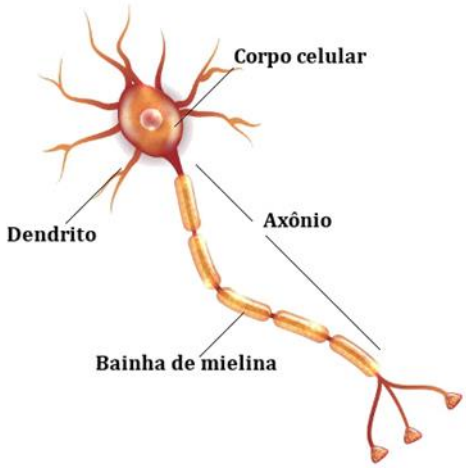


Figura 2-Neurónio [6].

Métodos de gravação neuronal

Geralmente, existem dois métodos de medição de sinais neurais: um método não invasivo e um método evasivo. Métodos não invasivos são a Eletroencefalografia (EEG), Magneto Encefalografia (MEG), Ressonância Magnética Funcional (fMRI) e Tomografia por Emissão de Positrões (PET) [8].

O EEG grava as mudanças no campo elétrico dos neurónios colocando elétrodos no couro cabeludo [8]. Possui um tempo de resolução elevado, na ordem dos milissegundos, detetando assim eventos de baixa magnitude, permitindo-lhe assim ser a técnica principal de monitoramento para a maior parte dos processos neurofisiológicos. Contudo, a maior limitação desta tecnologia é a baixa resolução espacial. Na figura 3 é apresentado um exemplo de uma touca para realização de EEG [7].

O MEG possui o mesmo processo neurofisiológico que o EEG (ver exemplo de realização figura 3) [8]. A diferença é que o MEG deteta as alterações nos campo magnéticos.

Comparado com o EEG, o MEG tem uma melhor resolução espacial, mas transmite menos informação.

O fMRI e o PET monitorizam a atividade neuronal através do fluxo sanguíneo [8]. Estas duas tecnologias quando comparadas com o EEG possuem melhor resolução espacial, por outro lado têm limitações no que respeita ao tempo de resolução e devido ao facto de o sujeito de teste não possuir liberdade de movimentos.

Figura 3-Exemplo de touca para realização de EEG [7].

Os métodos invasivos de gravação neuronal gravam a atividade neuronal diretamente em contactos com o tecido neuronal, sob o crânio. Conseguem ao mesmo tempo possuir um tempo de resolução temporal e espacial elevado quando comparados com os métodos não invasivos, o que é bastante importante para perceber a conectividade entre a atividade neuronal e intracraniana [8]. Nestes métodos estão incluídos a Eletrocorticografia (ECoG) e o EEG Intracraniano.

O ECoG coloca o micro elétrodo na superfície do córtex enquanto o intracraniano EEG o coloca dentro do córtex [8]. O ECoG não consegue gravar sinais neuronais de células individuais enquanto o intracraniano EEG consegue. Para além de conseguir gravar individualmente, possui resolução espacial e temporal elevadas permitindo descodificar sinais neurais e controlar membros artificiais [8].

Sondas neuronais

De todos os sistemas do corpo humano, o cérebro é o menos compreendido e consequentemente o mais difícil de ser tratado [9]. Em meados do século 18 foi utilizado, pela primeira vez, corrente elétrica na tentativa de restaurar o movimento de membros paralisados

[9]. Desde 1950, com o passar dos anos e com o desenvolvimento da tecnologia, as sondas neuronais, tornaram-se uma peça chave na gravação de atividades cerebrais [10] e na estimulação de zonas do cérebro específicas [11].

As sondas neuronais são microestruturas que fazem a conexão entre o tecido neuronal e os dispositivos eletrónicos. Devem possuir o mínimo tamanho possível pois só assim conseguem minimizar o dano nos tecidos cerebrais, facilitando também a sua inserção no cérebro. Tendo estas características, tem melhor relação de sinal-ruído da atividade gravada ou especificar uma região a ser estimulada [11].

Em 1950 começaram a ser utilizados microeléctrodos de metal com fio único [11]. Estes elétrodos consistem num fio de metal isolado excetuando a ponta do fio que representa o local de gravação [11]. Passados 20 anos de desenvolvimento, Wise *et al.* [12], aproveitando o desenvolvimento dos circuitos integrados criaram as primeiras sondas planares baseadas em silício (ver figura 4) [10]. O uso do processo fabrico de fotolitografia permite controlar o tamanho da sonda de acordo com o pretendido, bem como a sua forma de acordo com o local a ser gravado. Existe a possibilidade de aumentar o número de gravações num menor volume, o que não é possível utilizando sondas de metal [13].

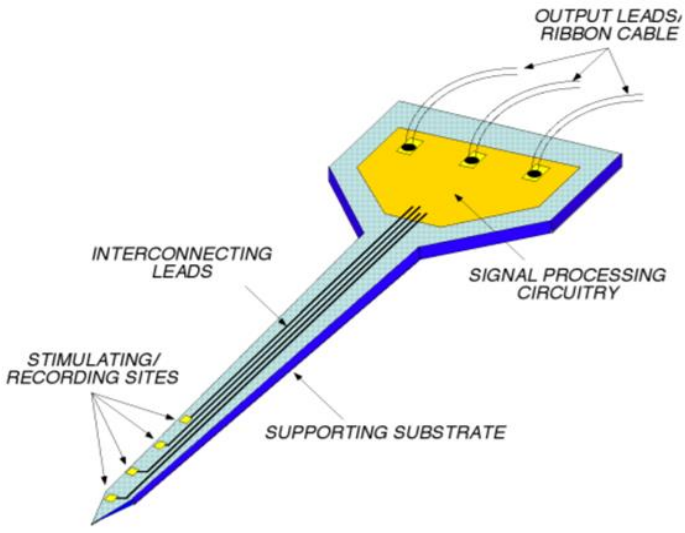


Figura 4-Sonda planar de silício [10].

Na década de 90 e com os avanços dos processos de micro fabricação surgiram as sondas tridimensionais. Em Michigan foram desenvolvidas sondas composta por agulhas onde os elétrodos são posicionados nas suas pontas permitindo medição mais profunda da atividade cerebral [14]. Outra sonda tridimensional desenvolvida na Universidade de Utah consiste em 100 agulhas de silício revestida por platina (ver figura 5) [15].

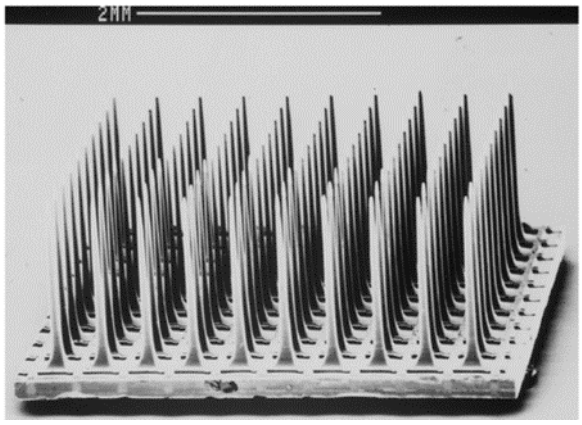


Figura 5-Matriz de Utah [15].

Optogenética

A optogenética é a combinação da genética e de métodos óticos para alcançar eventos definidos em células específicas do cérebro [16]. A optogenética é capaz através da luz de controlar um grupo específico de células. Começou em 1979, quando Francis Crick apontou que o maior desafio da neurociência seria a necessidade de controlar um tipo de células do cérebro não alterando as outras [16]. Crick tinha a ideia que a luz possuía propriedades que serviriam para controlo das células, podendo estas ser ativadas ou desativadas através da sua exposição, mas na altura os neurocientistas não possuíam nenhuma informação em como fazer as células responder à luz [16].

Anos antes, Crick Stoeckenius e Oesterhedl descobriram que a proteína Bacteriorhospin era capaz de atuar como impulsionadora de iões sendo controlada por fotões de luz visível [17]. Outras proteínas capazes de efeitos semelhantes foram descobertas, como por exemplo, em 1977 foi Masuno-Yagi e Mukohata, que descobriram a Halorhospsins e em 2002 Nagel, descobriu a Channelrhodopsins [17]. Todas elas conseguiam controlar o fluxo de iões em resposta à luz [17].

Apesar de toda a evolução na área, foi apenas em 2005 que os neurocientistas conseguiram juntar ambos os conceitos ao introduzir um gene da opsina microbial, nos neurónios de um mamífero alcançando o controlo sustentado ao milissegundo dos potenciais de ação [17]. Desde então, muitas investigações têm sido conduzidas utilizando as 3 opsinas microbiais referidas anteriormente e todas ficaram provadas como tendo funções optogenéticas no controlo dos neurónios [17]. Na figura 6 é apresentado um exemplo de funcionamento da optogenética.

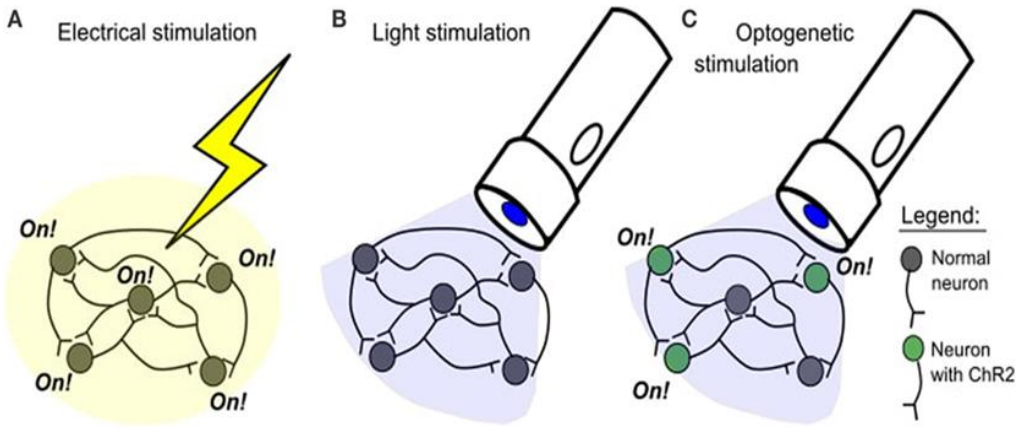


Figura 6-Exemplo de funcionamento da Optogenética (imagem adaptada do site https://www.frontiersing.org): A-Estimulação elétrica; B-Estimulação a luz; C-Estimulação Optogenética.

Organização da dissertação

A presente dissertação divide-se em 7 capítulos. No primeiro capítulo encontra-se uma breve introdução adjacente á realização da dissertação, bem como alguns conceitos utilizados durante a dissertação. Também no primeiro capítulo se encontra a sua motivação e os objetivos a atingir com o projeto.

No segundo capítulo aborda-se o estado de arte onde se encontra informação sobre os vários tipos de comunicação sem-fios, existindo um enfase entre as diferenças do BLE, Bluetooth e WiFi. São apresentados alguns sistemas desenvolvidos por outras universidades assim como alguns modelos comerciais já existentes de sistemas de comunicação sem fios para sondas neuronais.

O terceiro capítulo refere quais as metodologias adotadas na realização da dissertação bem como a arquitetura do sistema. Durante este capítulo serão explicados os materiais utilizados na dissertação e a sua função no sistema desenvolvido.

No quarto capítulo é explicado o *design* do *software* desenvolvido, assim como são explicados vários fluxogramas sobre o código desenvolvido.

No quinto capítulo são explicadas algumas funções implementadas e o protocolo de comunicação implementado.

O sexto capítulo destina-se aos testes e resultados obtidos. Serão apresentados vários testes, assim como as características da interface gráfica.

No sétimo capítulo aborda-se as conclusões acerca do trabalho realizado durante o desenvolvimento do projeto, bem como considerações que possam vir a ser tomadas num futuro com o intuito de melhorar o projeto já existente.

No anexo 1 é apresentado um protocolo para a inicialização e funcionamento do sistema.

ESTADO DE ARTE

Tecnologias de comunicação sem fios

A comunicação é parte integrante do nosso dia-a-dia. A necessidade de comunicar levou ao desenvolvimento de novos métodos que facilitam a troca de dados entre dois dispositivos. Com o objetivo de dinamizar, o ser humano recorreu à tecnologia sem fios de modo a transferir dados sem o uso de cabos, tornando a comunicação mais prática e cómoda.

A comunicação sem fios não se trata de uma inovação recente, mas sim de uma evolução contínua que tem vindo a ser desenvolvida durante vários anos. São apontados como tipos de comunicação sem fios: redes de comunicação móvel via satélite, redes de acesso fixo sem fios, redes de telemóveis, redes de área local sem fios e, por fim, redes de área pessoal sem fios. Para existir uma transmissão entre dois dispositivos são necessários um emissor, recetor e um meio de propagação de forma a difundir a mensagem pretendida.

Redes convencionais vs redes sem fios

A comunicação sem-fios agrega, naturalmente, um conjunto elevado de vantagens e desvantagens relativamente às redes com fios convencionais. Apresentam um maior grau de mobilidade durante a comunicação. Atendem a um número de utilizadores claramente superior às redes que utilizam cabos, sem necessitar de investir abundantemente na quantidade de equipamento, reduzindo consecutivamente o custo de manutenção do sistema, permitindo ainda o acesso à rede sem fios através de qualquer local desde que tenha sinal [18].

Por outro lado, existem aspetos negativos a ter em consideração: a velocidade de transmissão acaba por decair quando comparada com redes com fios, encontrando-se também o sinal de transmissão sujeito a interferências que estão aquém do controlo do sistema da rede (metais, água, sinais da rede elétrica poderão colocar em causa a correta transmissão de dados). Acrescentam-se ainda a vulnerabilidade na segurança onde é necessária a implementação de encriptações de modo a proteger as informações transmitidas. O alcance do sinal é apontado também como um fator negativo, podendo não ser suficiente tendo em conta a área que se pretende cobrir [18].

WiFi

É descrita como uma comunicação sem fios de baixa potência empregue por diversos dispositivos eletrónicos como computadores portáteis ou Smartphones. No WiFi, um *router* remete a uma central de comunicação. Esta tecnologia rege-se por padrões estabelecidos pelo IEEE 802.11, sendo possível a utilização de pontos de acesso ou repetidores de forma a expandir o alcance do sinal oferecendo uma maior área de cobertura [20].

O WiFi utiliza as ondas de rádio para efetuar a comunicação entre dispositivos, tendo uma área de alcance que pode ir desde os 4.5 até 15 metros [20]. Numa fase inicial, para começar a comunicação é necessário que os dispositivos que se desejam conectar passem por um processo de reconhecimento e emparelhamento. Geralmente recorre-se à utilização de senhas de forma a evitar que qualquer pessoa se ligue à rede. O WiFi é baseado numa arquitetura celular, onde cada célula é chamada de serviço básico, sendo um serviço básico um conjunto de estações de WiFi móveis ou fixas. O acesso ao meio de transmissão é controlado por um conjunto de regras chamados de função de coordenação. O WiFi define uma Função de Coordenação Distribuída (FCD) e uma Função de Coordenação de Pontos (FCP), sendo esta última opcional [20].

A tecnologia WiFi apresenta um baixo consumo e custo relativamente às interfaces concorrentes no uso de dispositivos pessoais. Um dos pontos fortes desta tecnologia é o facto do emissor e o recetor não necessitarem de estar alinhados para se efetuar a troca de dados [21].

ZigBee

Esta tecnologia visa aplicações que representam um baixo consumo de energia associados a um baixo custo. Contrariamente às redes conhecidas, esta foi gerada com a finalidade de saciar necessidades inerentes a aplicações de monotorização e controlo onde os dados são recolhidos do meio ambiente através de sensores. Os dispositivos utilizados são, por exemplo, microcontroladores e transcetores que representam uma baixa complexidade e elevada autonomia. Esta tecnologia providencia uma taxa de transmissão de 250 kbps [22]. O ZigBee utiliza canais de 0 a 10 que se encontram definidos na banda sub-1 GHz enquanto na banda industrial, científica, médica (ISM) de 2.4 GHz os canais encontram-se numerados de 11 a 26 [22].

Bluetooth

O desenvolvimento da indústria de Bluetooth *standard* começou no final de 1998, quando a Ericsson, IBM, Intel, Nokia e Toshiba formaram o Grupo Especial da Indústria Bluetooth (SIG) para desenvolver e promover uma solução global para comunicação sem fio de curta distância operando na não licenciada banda de 2.4 GHz ISM [23].

O Bluetooth é um sistema de comunicação sem fios projetado para dispositivos de comunicação baratos e de curto alcance, adequado para substituir cabos de impressoras, fax, ratos, teclados etc. Quando um dispositivo Bluetooth é ligado começa por procurar dispositivos *master* e responde aos seus anúncios. Nesta fase o dispositivo *master* fica a saber o endereço do dispositivo *slave*. Sabendo o endereço do *slave*, o *master* tenta abrir uma conexão [20]. Estando a conexão estabelecida, os dispositivos começam a comunicar após a autenticação estar concluída. O protocolo de comunicação responsável por encontrar outros dispositivos e anunciar a presença de um dispositivo *master* é constituído pela camada física e pela camada de ligação.

O *Link Manager* que faz parte do protocolo de comunicação é parte essencial na arquitetura Bluetooth, sendo utilizado para configurar, autenticar e manipular as conexões entre dispositivos Bluetooth, para além de operar como gestor de energia. Para além do *Link Manager*, fazem igualmente parte do protocolo de comunicação o *Control*, *L2CAP*, *Baseband* e *Physical Radio* [20].

Bluetooth low energy (BLE)

O BLE é uma tecnologia de comunicação sem-fios, que consome menos energia quando comparando com outros modos de Bluetooth. Foi desenvolvida pelo SIG, com o objetivo de comunicação a curto alcance. Estudos apontam que consegue comunicar até dois anos utilizando apenas uma pilha de lítio [25]. Este foi um dos principais motivos para a utilização do BLE como a tecnologia de comunicação sem-fios para a realização desta dissertação de mestrado.

Como no Bluetooth clássico o protocolo da *stack* do BLE é composto por duas camadas baixas, a camada física e a camada de ligação (ver figura 7) [24]. A camada física faz a transmissão e receção dos bits. Da camada de ligação faz parte a *Logical Link Control and Adaptation Protocol* (L2CAP), *Attribute Protocol* (ATT), *Generic Attribute Profile* (GATT),

Security Manager Protocol (SMP) e *Generic Access Profile* (GAP) [24]. Ao nível mais alto da *stack*, através do GAP podem ser definidas funções para o dispositivo. Um dispositivo pode ser *Broadcaster, Observer, Peripheral* e *Central*. Um *Broadcaster* apenas transmite data, não suportando comunicação bidirecional. O *Observer* é o contrário do *Broadcaster* sendo que tem o propósito de receber a informação transmitida por outros dispositivos. O papel de um dispositivo *Central* trata-se de iniciar e coordenar várias conexões, ao mesmo tempo um dispositivo *Peripheral* é como que o *slave*, tendo função de se conectar ao *Central*. Um dispositivo pode suportar várias funções, mas só pode desempenhar uma função num determinado espaço de tempo [24].

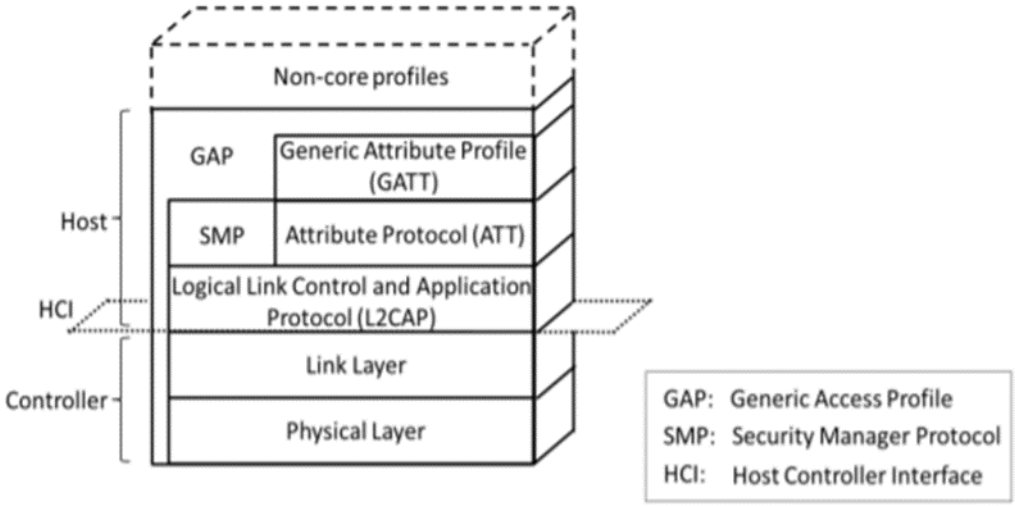


Figura 7-Stack BLE [24].

O BLE opera na banda dos 2.4 GHz. Existem 2 tipos de canais Radiofrequência (RF), canais de dados e de *advertising*. Os canais de *advertising* são utilizados na descoberta de dispositivos, estabelecer a conexão com o dispositivos e na transmissão de dados. Quando a transmissão de dados é bidirecional são utilizados os canais de dados [25]. Três canais estão definidos para trabalharam como canais de *advertising*, sendo o canal 1, 6 e 11 os mais habitualmente utilizados em vários países. Para um melhor aproveitamento dos canais é utilizado um mecanismo de *hopping*. Mecanismo este que seleciona um dos 37 canais disponíveis de dados para comunicar durante um determinado intervalo de tempo [25].

No BLE existem 4 métodos de transferência de informação [24]:

- A escrita, onde o dispositivo definido como *Central* envia uma mensagem para o dispositivo definido como *Peripheral*;

- A leitura, onde o *Central* pede ao dispositivo *Peripheral* para enviar a resposta lendo depois a resposta;
- Os últimos dois métodos de transferência de dados, são as notificações e as indicações, onde o dispositivo *Central* ativa as notificações ou as indicações consoante o que pretende utilizar, e de seguida o *Peripheral* envia a informação para que está programado, até receber do *Central* "ordens" para a desativação do envio da informação.

A única diferença entre estes dois últimos métodos é que as notificações não enviam confirmação de receção da informação ao contrário das indicações que por cada pacote enviado, envia de volta um *acknowledge* [24].

A comunicação bidirecional entre dois dispositivos requer a existência de uma conexão entre os dois [24]. A conexão ao ser criada entre os dois dispositivos será um procedimento assimétrico onde o anunciante anuncia a sua presença através dos canais de *advertising*, dando conta que está pronto a ser conectado enquanto o outro dispositivo ouve a presença do anunciante. Após encontrar a presença do anunciante, o dispositivo envia um pedido de conexão para o anunciante, criando assim uma conexão entre os dois dispositivos.

Para a comunicação entre dois dispositivos BLE existem dois papéis, o *master* e o *slave.* O *master* consegue conectar a vários *slaves*, mas o *slave* apenas se consegue conectar a um único *master* [24]. O BLE destaca-se dos outros métodos de comunicação devido ao menor gasto de energia [24]. Um dos motivos para isso acontecer é graças ao modo de comunicação entre os seus dispositivos. Na figura 8 encontra-se é apresentado o funcionamento da transmissão de dados [26].

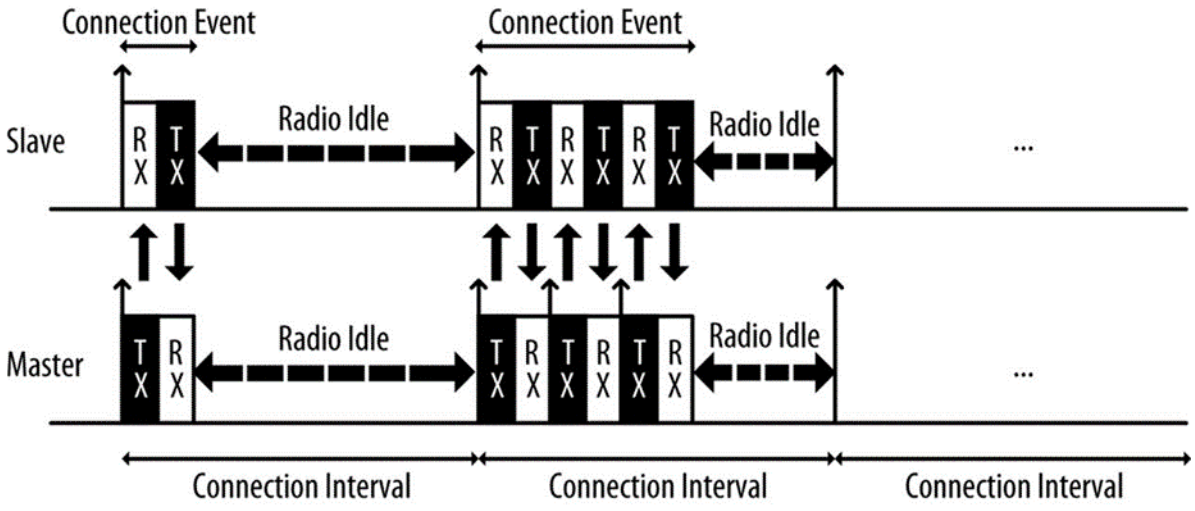


Figura 8-Exemplo de um intervalo de conexão BLE [26].

No início de cada conexão, o *master* determina o intervalo de conexão, isto é, o tempo que o *slave* tem de sair do modo *sleep*, para ouvir o *master* e receber os pacotes de dados. Durante um intervalo de conexão são enviados pacotes com a informação, quando não há mais informação a ser enviada o *slave* volta ao modo *sleep*, poupando assim energia. O tempo entre cada intervalo de conexão pode ser definido em múltiplos de 1.25 ms, sendo o tempo total entre os 7.5 ms e os 4 s [24]. Como podemos ver na figura 9, num intervalo de conexão podem ser enviados vários pacotes de dados, dependendo do tamanho do pacote e do intervalo de conexão. Outro parâmetro importante na comunicação e que pode ser utilizado para poupar mais energia é a latência do *slave*, parâmetro que pode variar entre 0 e 499. Define o número de eventos de conexões consecutivas que podem ser saltadas pelo *slave*, não precisando de ouvir o *master*, logo não precisando de enviar pacotes. Na figura 9 podemos ver um intervalo de conexão com latência do *slave* de 3 [24].

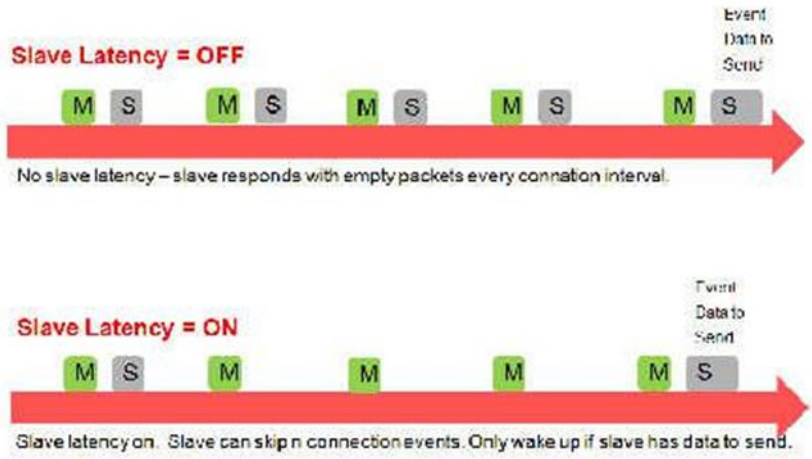


Figura 9-Intervalo de conexão com latência ativa [27].

Bluetooth vs bluetooth low energy vs WiFi

Sendo o Bluetooth, o BLE e o WiFi as tecnologias de comunicação sem fios mais utilizadas nos mais variados sistemas, de seguida apresenta-se a tabela 1 que compara as três tecnologias. Compara-se especificações como a distância, a taxa de transmissão e os seus consumos de energia.

Tabela 1-Tabela comparativa entre Bluetooth, BLE e WiFi [28, 29].

Especificações Técnicas	**Bluetooth Clássico**	**Bluetooth Low Energy**	**WiFi**
Network Standard	IEEE 802.15.1	IEEE 802.15.1	IEEE 802.11
Frequência de Rádio	2.4-2.5 GHz	2.4-2.5 GHz	2.4/5 GHz
Distância	100 m	50 m	70-250 m
Taxa de transmissão no ar	2.1 Mbps	1 Mbps	200 Mbps -1.73 Gbps
Taxa de transmissão	0.7-2.1 Mbps	0.27 Mbps	Dependente de vários fatores
Consumo de energia	1 Wh	0.01-0.50 Wh (Dependendo para o que está a ser utilizado)	1.60 Wh (Receção) 10 Wh (Transmissão)
Corrente de pico	< 30 mA	< 15 mA (Leitura e transmissão)	~ 50 mA (Receção) ~200 mA (Transmissão)
Latência	< 100 ms	< 3 ms	< 1 ms
Utilizado frequentemente em	Telemóveis, periféricos computadores	Pulseiras *fitness*, Relógios, dispositivos de segurança	Computadores, impressoras, telemóveis, TV's

Como podemos verificar pela tabela 1 existem bastantes diferenças entre estes sistemas de comunicação sem fios, quer em termos de consumo de energia, de distância ou de taxa de transmissão. Começando pela frequência de banda que utilizam, todas funcionam nos 2.4 GHz, sendo que a nova geração de WiFi pode funcionar também nos 5 GHz.

Em termos de distância verifica-se uma melhor performance por parte do WiFi, conseguindo até 250 metros de cobertura, estando dependente da força do sinal e dos obstáculos. Comparando com o Bluetooth a diferença pode ir até aos 150 metros. Como seria de esperar, maior diferença existe quando comparado com o BLE, que possui uma cobertura de apenas 50 metros.

Igualmente, a taxa de transmissão do WiFi é superior às outras, mas esta está dependente de fatores como a força do sinal, a distância e os obstáculos entre o recetor e o transmissor. A

diferença entre o Bluetooth e o BLE é significativa sendo que o Bluetooth pode atingir os 2.1 Mbps e o BLE consegue apenas 0.27 Mbps.

O consumo de energia tem uma diferença significativa, sendo o BLE a tecnologia que consome menos energia comparativamente com as outras duas. No máximo tem um gasto de 0.50 W, quando tem uma utilização elevada ou estão a ser enviados pacotes de grande dimensão. O Bluetooth por seu lado tem um gasto de energia mais constante, mas é aproximadamente o dobro do BLE. O WiFi destaca-se bastante no consumo de energia, principalmente aquando da transmissão de dados, chegando aos 10 Wh, por ter um consumo de energia tão elevado consegue taxas de transmissão bastante superiores relativamente às outras tecnologias de comunicação sem fios. A corrente de pico está ligada invariavelmente ao consumo de energia, quanto menos consumo, menor vai ser a corrente de pico.

As especificações de cada tecnologia são o que as levam a ser escolhidas para as mais variadas aplicações. O BLE tem a vantagem de ter um consumo muito reduzido, mas por outro lado possui uma taxa de transmissão baixa. Por isso, é mais utilizado em pequenos aparelhos como bandas de *fitness*, sensores de proximidade e equipamentos de saúde como a medição de batimentos cardíacos. O Bluetooth é utilizado em aparelhos um pouco mais exigentes que o BLE, pois possui uma maior taxa de transmissão e por consequência um consumo maior. Tem utilização em computadores, telemóveis, periféricos para computadores como ratos, teclados ou auriculares. São equipamentos que precisam de uma maior taxa de transmissão e por isso utilizam Bluetooth e não BLE. Devido ao seu maior poder de transmissão o WiFi é utilizado em computadores para o acesso à internet, impressoras, colunas sem fios, *smartphones*, servidores ou televisões.

Literatura

Várias experiências na área da neurociência requerem ou seriam imensamente beneficiadas com sistemas de gravação neural sem fios. Contudo, todos os sistemas existentes para venda comercial são caros ou não se adaptam às exigências pretendidas. Com o intuito de ultrapassar esses obstáculos são várias as universidades e centros de investigação que desenvolvem os seus próprios sistemas [30].

Cada sistema de gravação neuronal sem fios possui um *design* diferente sendo que o objetivo final de todos os sistemas é o mesmo. Atingir um sistema com um baixo consumo de energia, alta frequência de amostragem e pequenas dimensões, com alta performance

independente do número de canais a serem adquiridos. Seguidamente serão apresentados alguns sistemas desenvolvidos por diversos grupos de investigação.

Na Universidade Wake Forest, foi desenvolvido, no departamento de fisiologia e farmacologia, um sistema sem fios que utiliza o Bluetooth como tecnologia para a transmissão de atividade neural em animais que se movem livremente [31]. Sistema este que conta com a gravação de atividade neural de ratos que se mexem livremente, em qualquer laboratório ou outro contexto. A gravação continua independentemente do contexto experimental inserido, o formato dos dados transmitidos não requer pós processamentos e pode ser utilizado para interagir *online* com a performance do animal em transmissões anteriores. O sistema consiste em três partes principais, uma *headstage* que amplifica e filtra os sinais adquiridos, uma mochila que é colocada nas costas do animal e contem uma Placa de circuito impresso (PCB), com um amplificador, um processador para a conversão dos sinais e o transmissor Bluetooth, e por último o recetor Bluetooth 2.0.

A *headstage* amplifica os sinais 100 vezes e possui um *multiplexer* 16:4 controlado pelo processador [31]. Cada um dos quatro *multiplexers* é ligado a um filtro passa alto com um ganho de 40 vezes e largura de banda entre 250 Hz e 6 kHz. A conversão é feita quatro canais de cada vez, com uma frequência de 25 kHz e 8 bits de resolução. O processamento de cada canal tem uma duração aproximada de 1.5 ms o que faz um total de 25 ms para os 16 canais. A conexão com o módulo Bluetooth é feita através de *Universal Asynchronous Receiver Transmitter* (UART) sendo que de seguida o Bluetooth envia para a USB Dongle os sinais já processados e digitalizados. O peso total do sistema é 60.3 g, com bateria incluída. A bateria aguenta mais de cinco horas contínuas de operação [31]. Na figura 10 é mostrado o sistema montado nas costas de um rato.

Figura 10-Sistema da Universidade Wake Forest implementado num rato adulto [31].

A Universidade de Zhejiang também desenvolveu um sistema, pelo departamento de Engenharia Biomédica [32]. Este sistema difere dos outros pois os sinais não são enviados para um computador, mas sim para um Assistente digital pessoal(PDA), sendo utilizada a tecnologia Bluetooth para a comunicação. As características diferenciadoras deste sistema é a capacidade para estimular até quatro posições diferentes do cérebro e adquirir sinais de dois locais diferentes simultaneamente. Para além do referido em cima, também se diferencia ao receber os dados num PDA e ser capaz de estimulação elétrica sendo que o sistema da Universidade Wake Forest não possuía essa funcionalidade.

O sistema é constituído por três componentes principais tal como o anterior, a *headstage*, o PDA portátil e uma mochila [32]. Tal como no sistema anterior também a comunicação entre PDA e Bluetooth é feita através de UART. A *headstage* possui quatro canais independentes para a estimulação e dois canais de gravação separados, com uma largura de banda entre 1 Hz e 4.5 kHz, sendo que possuem ganho de amplificadores diferentes. Um amplificador tem capacidade para um potencial de entrada de +- 0.5 mV e amplifica até 50 vezes, e o segundo aguenta tensões entre +- 80 µV e amplifica até 2 vezes. A mochila possui a placa principal que trata da comunicação bidirecional com o PDA, e o módulo de Bluetooth. O módulo de Bluetooth consegue transmitir informação até uma distância de 10 metros, e uma velocidade de 70 kb/s, mais do que suficiente para mostrar dois canais em tempo real. No PDA foi desenvolvido um *software* que para além da visualização dos canais, também configura os diferentes parâmetros da estimulação.

Como resultados finais foram atingidos os 12 *kSamples/s* por canal, uma bateria capaz de aguentar duas horas com um peso total de 40 g [32]. Em baixo encontra-se figuras que mostram a placa (ver figura 11 a)), e o sistema implementado no rato (ver figura 11 b)).

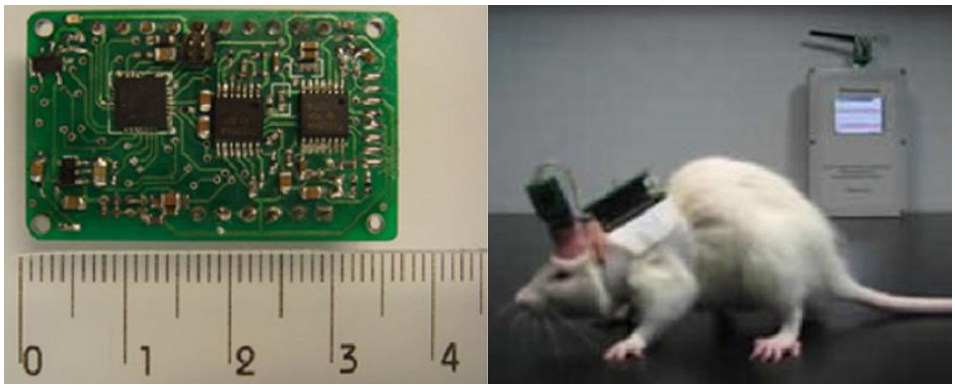


Figura 11- a) Placa principal do sistema da Universidade de Zheijang; b) Sistema da Universidade de Zheijang implementado num rato adulto [32].

Por último vai ser apresentado um sistema desenvolvido pela Universidade de Nova York e parecido com o que foi desenvolvido nesta dissertação devido à utilização da mesma tecnologia de comunicação, o BLE [33]. Neste sistema a prioridade foi minimizar o seu tamanho e possuir uma bateria mínima para duas horas. Como parte principal do sistema o chip ADS128 da Texas Instruments providência um ruído consoante a taxa de amostragem utilizada, conseguindo, o chip ,alcançar uma frequência de 32 *kSamples/s* com 8 canais e 24 bits de resolução. Para a comunicação sem fios é utilizado o PAN1721, que é um módulo sem fios pré-fabricado que inclui o chip CC2541 e uma antena. Módulo este que possui uma distância de transmissão entre 10 a 25 metros.

O sistema (ver figura 12) tem um diâmetro de 25 mm e altura de 9 mm, sendo que atinge os 250 *Samples/s* e utilizando um ganho de 12, a bateria dura aproximadamente entre 30 a 60 minutos, alimentado com um pilha de botão com 3 V. O problema da utilização do BLE é que a taxa de amostragem nunca será a pretendida pois o BLE não é talhado para a transmissão de dados a alta velocidade [33].

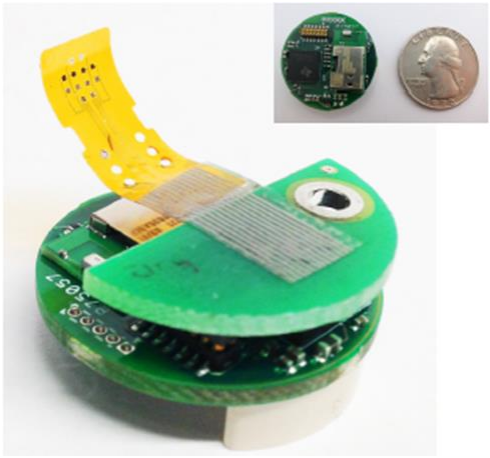


Figura 12-Placa PCB do sistema da Universidade de Nova York [33].

Soluções comerciais

Devido ao aumento da procura por dispositivos de gravação sem fios de sinais eletrofisiológicos são cada vez mais o número de empresas que possuem soluções comercias para o efeito. Cada empresa tem o seu sistema, com as suas especificações e as suas especificidades. Existem empresas que vendem o sistema elétrico e o *software* em separado, outras que vendem tudo em conjunto, tanto *hardware* como o *software* para analisar os dados numa aplicação de computador.

A seguir são apresentados um conjunto de soluções comerciais existentes no mercado de sistemas de gravação sem fios de sinais eletrofisiológicos provenientes de empresas especializadas e que possuem as suas próprias soluções de *hardware* e de *software*.

Jaga16

O Jaga16 é um dispositivo portátil de gravação eletrofisiológica da empresa Jinga-hi [34], uma empresa especialista no desenvolvimento de sistemas compactos sem-fios de gravação neuronal. Jaga16 é um dispositivo independente que inclui amplificador, digitalizador, processador de sinal e transmissor. Possui 16 canais e apesar de primeiramente ter sido pensado para a gravação de sinais eletrofisiológicos em pequenos animais, as suas especificações também permitem a sua utilização para sistemas de EEG-BCI [34]. A figura 13 mostra o dispositivo e as especificações são apresentadas na tabela 2.

Além do Jaga16, a empresa Jinga-hi possui uma variada diversidade de dispositivos de gravação eletrofisiológica, como o Jaga_Penny, este mais destinado a EEG's [34]. O *hardware* é vendido em separado do *software*, existindo um *software* diferente para o Jaga_Penny e para o Jaga16 [34].

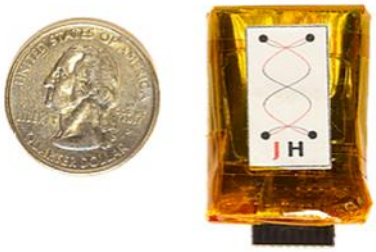


Figura 13-Hardware Jiga16 [35].

RatLog64

Deuteron Technologies Ltd é um fornecedor de equipamento de gravação neuronal sem fios, destinado para investigação em animais [36]. A sua especialidade é a troca de dispositivos pesados de gravação eletrofisiológica por sistemas miniaturizados para promover a investigação em animais sem a utilização de fios [36]. Os seus equipamentos são vendidos como um conjunto. Um único conjunto inclui duas estações base e equipamento para até quatro animais.

Um conjunto típico contém [36]:

4 registos de dados neuronais;

2 emissores/recetores rádio;

Software para computador Windows;

Todos os acessórios necessários.

O RatLog-64 é o sistema de gravação neuronal que se evidencia, é um sistema modular, leve e sem fios para gravação de dados em pequenos animais, como ratos [36]. Este sistema diferencia-se dos outros pois possui cartão de memória para a gravação dos dados em caso de algum problema na transmissão sem fios. Foto do sistema na figura 14 e especificações na tabela 2.

Figura 14-Hardware RatLog64 [37].

W2100-HS16

Multi Channel Systems MCS GmbH foi fundada em 1996 e é uma divisão da Harvard Bioscience, INC [38]. O principal objetivo da empresa é o desenvolvimento de equipamento na área da eletrofisiologia para ser utilizado por universidades e grupos farmacêuticos nas suas investigações. Fornecem soluções para gravação extracelular e com matrizes de microeléctrodos *in-vitro* e *in-vivo*, bem como estimulação elétrica. Têm como objetivo o fornecimento de equipamentos com alta qualidade e performance [38].

W2100-HS16 é uma solução multifuncional para amplificar, gravar e analisar dados em tempo real de 16 canais simultaneamente [38]. A empresa possui outros sistemas com diferentes quantidades de canais, mas para apresentação das especificações foi escolhida esta versão. O pequeno dispositivo que contém um Conversor analógico digital(ADC), um recetor, baterias e um *software* de computador para a leitura dos dados. O sistema utiliza a comunicação rádio com um range de 5 metros sendo os dados posteriormente enviados para o computador através do recetor [38]. Foto na figura 15 e restantes especificações encontram-se na tabela 2.

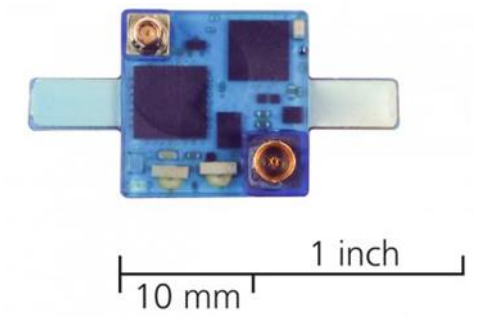


Figura 15-Hardware W2100-HS16 [39].

FreeLynx

FreeLynx é mais um sistema de gravação neuronal sem fios, desenvolvido pela empresa Neuralynx [40]. Desde a sua criação, a Neurolynx tem sido uma empresa pioneira e líder em ferramentas para investigação na neurociência, destacando-se em sistemas de *hardware* costumáveis e *softwares* de aquisição de dados usados para medição de sinais neurais [40].

O FreeLynx utiliza uma interface *Analog Front End* personalizável que se adapta facilmente dependendo das configurações escolhidas [41]. Permite otimizar a distribuição do peso, gravar a partir de vários pontos do cérebro ou número de canais [41]. Foto na figura 16 e especificações na tabela 2.

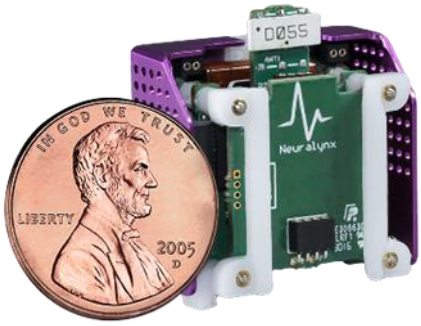


Figura 16-Hardware FreeLynx [41].

IW-Series

IW-Series é um sistema de gravação neural desenvolvido pela empresa Triangle BioSystems International [42]. Empresa focada no desenvolvimento de soluções de *hardware* e *software* para aplicação em investigações de neurociência [42]. O sistema IW-Series foi desenvolvido para a gravação de 15 sinais eletrofisiológicos sem parar durante 90 dias, 24 horas por semana, com o contra de a transmissão cobrir apenas 1 metro de distância. O sistema completo possui uma *headstage* com comunicação sem fios alimentada indutivamente, um

recetor de sinais RF, uma fonte de alimentação e um conjunto de conexão de elétrodos. O sistema apresentado não possui estimulação, mas pode ser utilizado em simultâneo com o sistema de estimulação que a empresa possui [43]. Informações mais detalhadas referentes às suas características encontram-se na tabela 2 e fotos do sistema apresentadas na figura 17.

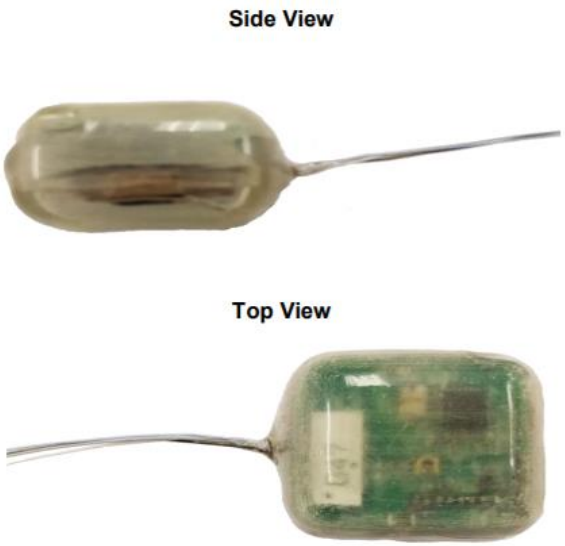


Figura 17-Hardware IW-Series [44].

De seguida, na tabela 2 são apresentadas as especificações dos sistemas acima descritos. Especificações como o número de canais, a tensão de entrada, a resolução, a frequência de amostragem ou a dimensão do sistema. Todos os sistemas são muito semelhantes apresentando ligeiras alterações entre cada um.

Tabela 2-Tabela comparativa de várias soluções comerciais [35, 37, 39, 41, 44].

Especificações	**JAGA16**	**Rata Log64**	**W2100-HS16**	**FreeLynx**	**IW-Series**
Nº de canais	16	64	16	64	15
Tensão de entrada	± 5 mV	± 10 mV	±12.4 mV	± 5 mV	± 5m V
Resolução	0.15 µV/bit	0.2 µV/bit	16 bit	16 bit	-
Frequência de corte baixa	0.1 - 300 Hz	0.2 - 500 Hz	1 Hz - 400 Hz	0.1 Hz - -	8 Hz - -
Frequência de corte alta	300 Hz - 6.75 kHz	300 Hz - 10 kHz	400 Hz – 5 kHz	- - 8 Khz	- -7 KHz
Ruído	2.7 µVrms	2.4 µVrms	< 1.9 µVrms	<2.5 µVrms	4 µVrms
Frequência de amostragem	20(14) kHz para 8(16) canais	32 kHz/canal	25kHz/canal com 16 ou 8 canais/ 40kHz (4 canais selecionados)	30 kHz/canal	50 kHz/canal
Dimensão	23 mm x 33 mm x 6 mm	15 mm x 15 mm	15.5 mm x 15.5 mm x 5.2 mm	30 mm x 30 mm x 30 mm	29 mm x 23 mm x 12 mm
Peso	6.2 g (antena incluída, sem bateria)	4.8 g	2.9 g (sem bateria) g	11.5 g	5.7 g
Bateria	Recarregável, (5-21 g) por 1-6 horas	2 h	-	30 m - 24h dependendo da bateria escolhida	90 dias
Tecnologia de comunicação	WiFi	RadioLink	Rádio	WiFi	Rádio

Como se pode verificar pela tabela 2, existem bastantes semelhanças entre os sistemas, mas cada um tem a sua particularidade e o seu destaque. Começando pelo número de canais, o JAGA16 e o W2100-HS16 possuem 16 canais, o IW-Series 15 canais, sendo que as outras duas soluções têm capacidade para leitura de 64 canais. Na tensão de entrada, a resolução e os filtros passa alto e passa baixo não têm diferenças sendo que todos possuem especificações

semelhantes. Em relação ao ruído, o sistema da Multi Channel Systems MCS GmbH tem um comportamento melhor pois consegue ter um ruído inferior a 1.9 µVrms, todos os outros possuem ruído superior a 2.4 µVrms, sendo que o IW-Series destaca-se pela negativa neste aspeto pois tem um ruído de 4 µVrms.

A frequência de amostragem, uma das especificações mais importantes, destaca-se o sistema da Triangle BioSystems International pois consegue atingir os 50 kHz de frequência de amostragem por cada canal. Os sistemas que possuem 64 canais mesmo tendo mais canais apresentam uma maior frequência de amostragem em relação os outros dois sistemas de 16 canais. O W2100-HS16 apresenta uma maior taxa de amostragem, mas apenas quando são selecionados 4 canais, o que é pouco em relação aos 15 canais apresentados pelo sistema IW-Series e mantendo os 50 kHz por canal mesmo assim.

Em relação ao peso e dimensão, o sistema W2100-HS16 volta-se a destacar dos demais sendo o sistema mais leve com 2.9 g, sem incluir a bateria, e também o mais pequeno em conjunto com o RatLog64. Por outro lado, com 11.5 g e 30 mm x 30 mm x 300 mm de dimensão o FreeLynx é o maior e mais pesado dos sistemas apresentados.

Em termos de tecnologias dois sistemas utilizam WiFi, o Jiga16 e o FreeLynx. O W2100-HS16 e o IW-Series utiliza comunicação rádio e o RatLog64 usa *RadioLink* como sistema de comunicação sem fios.

METODOLOGIAS E ARQUITETURA DO SISTEMA

No presente capítulo abordam-se os métodos utilizados no desenvolvimento do sistema de comunicação BLE, assim como a sua arquitetura e materiais utilizados. Por fim é apresentada a interface gráfica, desenvolvida para a apresentação dos sinais eletrofisiológicos e para a estimulação optogenética.

Requisitos do sistema

Os requisitos do sistema de comunicação BLE são definidos de acordo com a sua finalidade. Sendo o sistema utilizado para a gravação de atividade cerebral e de estimulação optogenética de pequenos animais, como ratos, tem de ser um sistema pequeno, leve e sem fios [3]. Não ter fios é um requisito essencial deste sistema pois só assim os animais se podem mexer livremente tornando o estudo mais eficiente. Muitas vezes durante uma sessão de gravação, os animais desenfreados tentam remover e morder os cabos, causando danos dispendiosos e perda de informação importante [4].

Uma bateria com longa duração é outra característica importante. A capacidade de realizar a gravação ininterrupta durante o máximo período de tempo é muito atraente para os neurocientistas porque a população neuronal sob análise muda muitas vezes ao longo do tempo [3]. A gravação durante a noite é necessária para assim controlar os neurónios durante um período de tempo mais prolongado para estudos de aprendizagem ou para a combinação de ensaios experimentais com duração de vários dias [3].

Os dados eletrofisiológicos depois de adquiridos são processados, sendo amplificados e digitalizados, por isso o ruído eletrónico é uma questão importante no sistema. Com um ruído elevado serão apresentados sinais destorcidos e diferentes dos adquiridos inicialmente. Com isto, o baixo ruído é outra exigência necessária na dissertação para assim os dados apresentados serem os mais idênticos possíveis aos dados adquiridos não causando confusão aquando da sua interpretação. Potenciais fontes de ruído podem vir do movimento de cabos suspensos [4], o que não acontece em sistemas sem fios.

Para os dados serem apresentados em tempo real e com leitura instantânea é importante que a taxa de transmissão seja elevada. Por isso a tecnologia de comunicação sem fios tem de conseguir enviar aproximadamente 15 *kSamples/s* (3.8 Mbps) [45], valor normalmente utilizado nos sistemas de gravação neural. A tecnologia para além de taxa de transmissão

elevada deverá permitir que o sistema consiga comunicar bidireccionalmente para assim, além de receber os dados na interface gráfica, também ser possível através da interface gráfica enviar comandos ativando ou desativando a estimulação optogenética.

O sistema de comunicação sem fios deverá reunir um conjunto de requisitos de forma a manter viáveis as funções. Com base nos pontos acima referidos os requisitos são os seguintes:

I. Sem fios;
II. Leve e compacto;
III. Bateria duradoura;
IV. Baixo ruído;
V. Elevada taxa de transmissão;
VI. Bidirecional;

Arquitetura do sistema

O sistema é divido em três partes. Primeiramente temos a parte eletrónica. Nesta etapa, o sinal elétrico é recebido, amplificado e digitalizado. Posteriormente é enviado para o módulo de BLE PSoC 4 que contem o chip CY8C4247LQI-BL43 [50]. A *headstage* da Intan com o chip RHS2116 é o responsável por toda a parte eletrónica, ficando encarregue da receção dos sinais eletrofisiológicos, da sua amplificação, digitalização e envio para o módulo BLE indicado em cima [50]. Nesta etapa também está incluída a estimulação optogenética, onde um díodo emissor de luz (LED) é ligado ou desligado consoante a escolha do utilizador.

Após o sinal ser digitalizado, é enviado para o módulo PSoC 4 BLE. Aqui temos a segunda parte do sistema, o sistema de comunicação. Para a transmissão de dados foi proposto o BLE como tecnologia de comunicação sem fios. Foi proposta esta tecnologia devido ao menor consumo de energia quando comparada com outras tecnologias de comunicação sem fios. Esta etapa é responsável pela aquisição do sinal digitalizado e pela sua transmissão sem fios para o módulo Bluetooth CySmart USB 5670 Dongle, módulo que está ligado ao computador.

Na terceira parte temos a interface gráfica, um *software* desenvolvido em C# no Ambiente de Desenvolvimento Integrado (IDE) Microsoft Visual Studio. Através da interface gráfica será possível analisar os dados eletrofisiológicos, e ativar a estimulação optogenética. A interface gráfica possui outras características que serão apresentadas mais à frente neste documento. Na figura 18 apresenta-se o diagrama do sistema de comunicação BLE implementado.

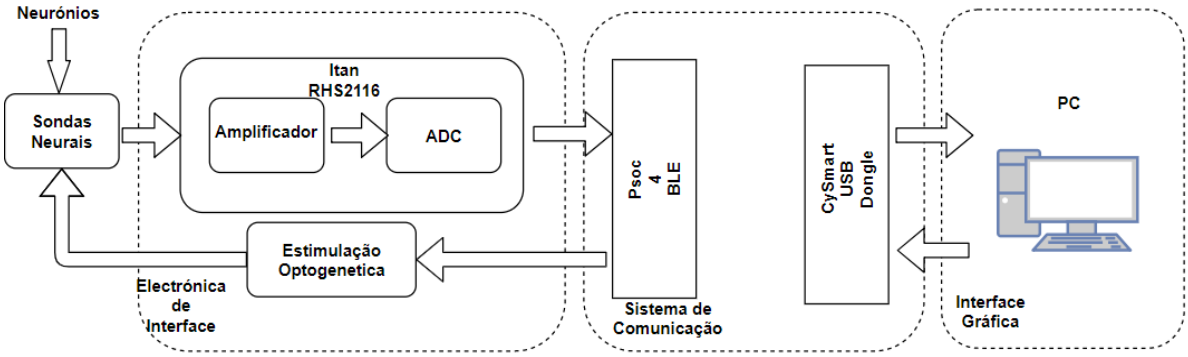


Figura 18-Diagrama do sistema de comunicação BLE.

Eletrónica de interface

A eletrónica de interface trata da aquisição e processamento dos sinais das sonas neurais e envia para o módulo sem fios, para além disso também faz parte o LED utilizado para a estimulação optogenética. Como referido anteriormente, a *headstage* da Intan que contém o chip Intan RHS2116, será o responsável pela parte de amplificação e digitalização dos sinais recebidos através das sondas neurais.

Intan RHS2116

A *headstage* de 16 canais estimulação/gravação da Intan Tecnhologies foi a placa escolhida para esta dissertação, utiliza o chip Intan RHS2116, e é o responsável pela parte eletrónica, fazendo a aquisição dos sinais eletrofisiológicos vindos do sensor neuronal. A RHS2116 inclui um sistema completo de interface bidirecional eletrofisiológico, sendo fabricado pela Intan Technologies. Possui uma arquitetura inovadora combinando estimuladores, amplificadores, filtros analógicos e digitais e um *multiplexer* de 16 bits ligado a um ADC [45]. Nos requisitos do sistema foi referido que uma das exigências seria o baixo ruído do sistema. O chip RHS2116 providencia um ruído de 2.4 µVrms [45] cumprindo assim um dos requisitos. Para além disso, a utilização do RHS2116 substitui qualquer uso de outro componente eletrónico, fazendo tudo o necessário. O ADC do chip suporta uma amostragem de 40 *kSamples/s* [45] em cada canal, mais do que o normalmente utilizado em sistemas de gravação neural.

A frequência de corte superior dos amplificadores é ajustável desde 100 Hz até 20 kHz. A frequência de corte inferior é ajustável desde 0.1 Hz até 1 kHz [45]. Para além de filtros analógicos, também um filtro digital pode ser utilizado. O chip da Intan inclui um módulo

digital que possibilita a utilização de um filtro passa alto em cada canal [45]. Tem utilização na remoção de tensões contínuas residuais associadas aos amplificadores AC. A programação do chip é feita através de comandos Serial Peripheral Interface (SPI). Na figura 19 podemos ver o esquemático da *headstage* utilizada na dissertação.

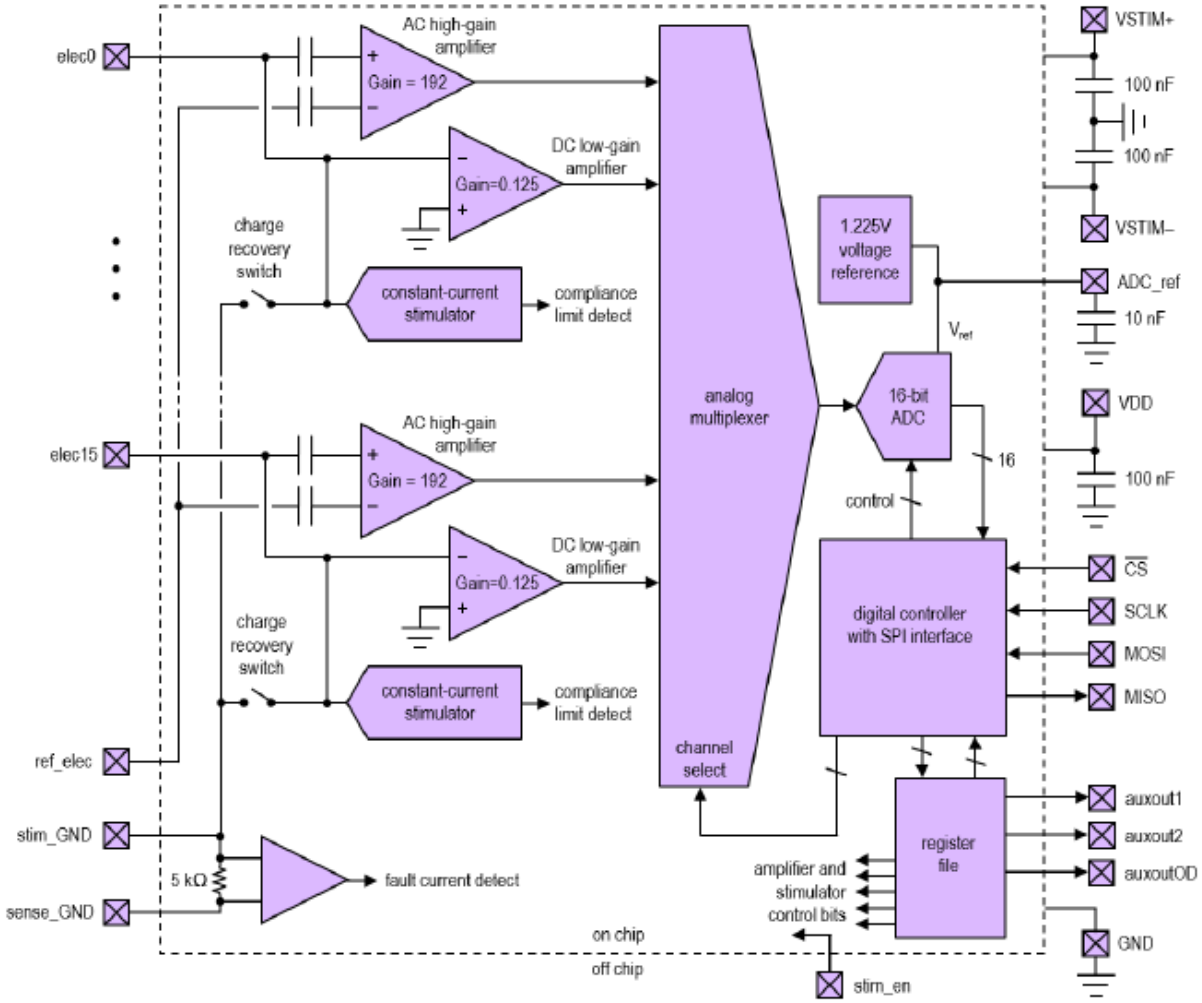


Figura 19-Esquemático da headstage Intan RHS2116 [45].

O RHS2116 contém 16 canais aptos para estimulação e amplificação, sendo que nesta dissertação a estimulação será optogenética logo não será necessário a estimulação elétrica individual de cada canal. Cada canal possui dois amplificadores, um amplificador AC de baixo ruído, ganho elevado e com largura de banda programável, para a observação de pequenos sinais (gama dos microvolts) eletrofisiológicos, suportando tensões de entrada entre ±5 mV e um ruído típico de 2.4 µVrms e um amplificador DC com ganho baixo para monitorizar o potencial dos elétrodos após a sua estimulação elétrica [45]. Estimulação esta como referido em cima, não utilizada, logo o amplificador também não é utilizado neste sistema.

Como podemos ver pela figura 20 e foi mencionado em cima, cada elétrodo está ligado a dois amplificadores diferentes, cada um com a sua função. Os amplificadores estão ligados a um *multiplexer* de 16 bits, onde os comandos recebidos por SPI irão ditar qual o sinal do elétrodo que será convertido em digital. Os resultados do ADC são convertidos utilizando a seguinte equação: $V_{ELE} = 0.195\ \mu V\ *\ (\text{ADC result}\ -32768)$ [45].

Sendo a estimulação utilizada neste sistema, a estimulação optogenética, também para isso o RHS2116 apresenta solução. Como podemos observar na figura 20 existem 3 saídas auxiliares, podendo ser ativadas através de comandos SPI. A saída auxiliar 1 vai ser utilizada para ligar e desligar o LED utilizado para a estimulação optogenética, que como foi referido no estado de arte, utiliza a luz dos LEDs para estimular os neurónios. Verifica-se também a existência dos pinos VSTIM+ e VSTIM- que são a fonte de alimentação para a estimulação elétrica em cada canal, mas não é necessária neste sistema.

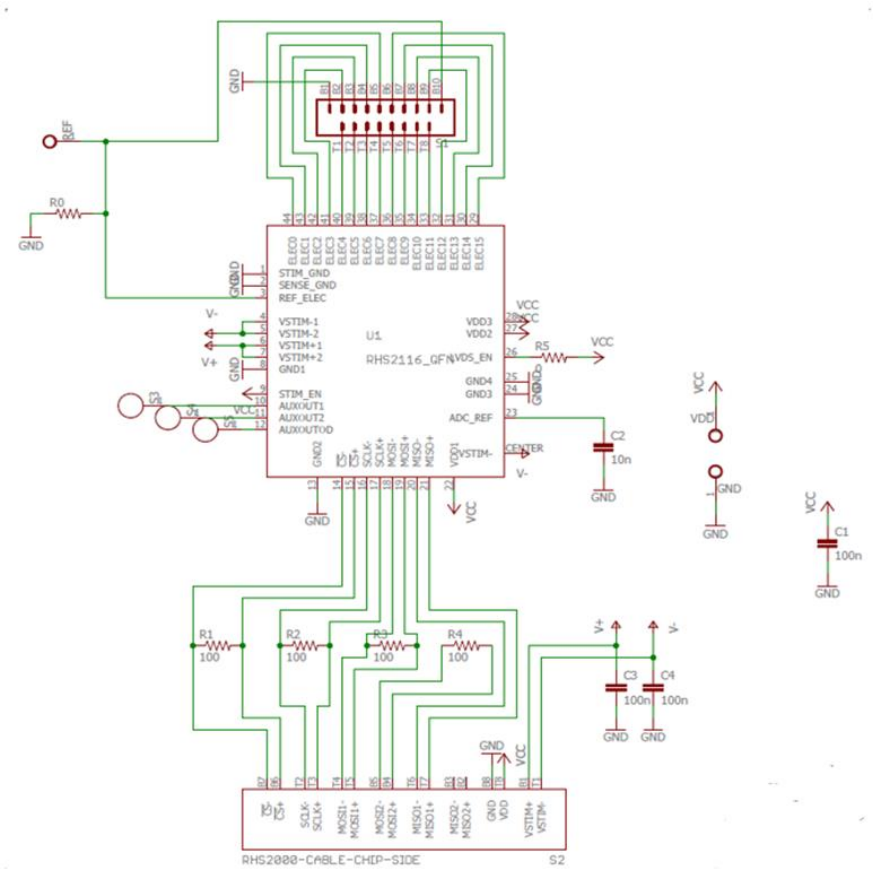


Figura 20-Diagrama simplificado do funcionamento do RHS2116 [46].

A comunicação entre a placa e o módulo BLE é conseguida utilizando o protocolo de comunicação *Serial Peripheral Interface* (SPI). O protocolo SPI funciona através de quatro sinais: o *chip select* (CS), *serial data clock* (SCLK), o *Master Out Slave In* (MOSI), e o *Master In Slave Out* (MISO) [38]. Na figura 21 encontra-se o diagrama de funcionamento do SPI.

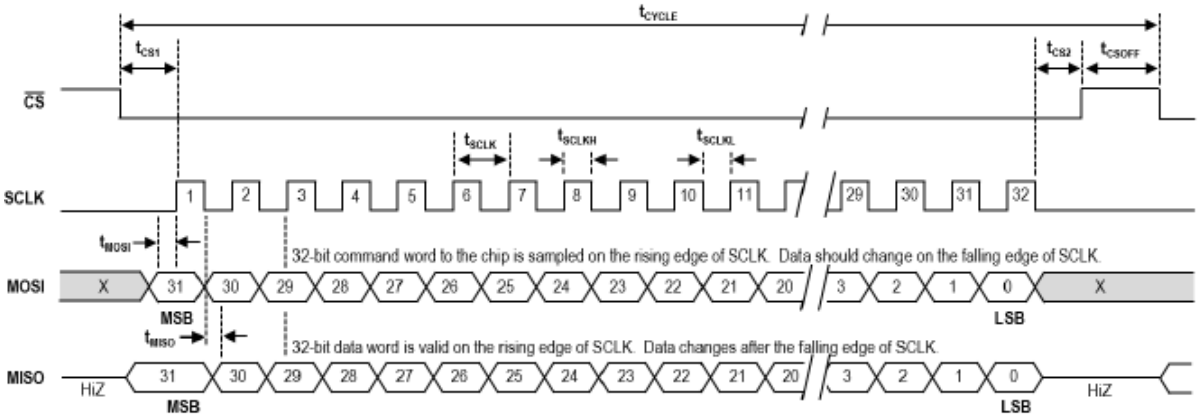


Figura 21-Funcionamento SPI [45].

O sinal *chip select* é o responsável por quando os 32 bits de informação são enviados. Quando o *chip select* é "0", 32 bits de data são enviados em cada direção, do *slave* para o *master*, através do MISO e do *master* para o *slave*, através do MOSI. O SCLK dita quando cada bit individualmente é enviado. Quando o sinal do SLCK se encontra na fase de subida, um bit de data é enviado.

Os níveis de tensão na transmissão dos sinais digitais podem assumir duas formas, complementary metal-oxide-semiconductor (CMOS) ou *low voltage differential signaling* (LVDS). Os sinais CMOS transmitem "0" ou "1", trocando o valor da tensão de saída do fio, entre o VDD e a massa. A corrente consumida é praticamente nula até a mudança de estado, neste ponto um pico de corrente carrega ou descarrega o condensador. O sinal LVDS utiliza um par de fios como por exemplo, MISO+ e MISO-, para a transmissão de sinais. Ligando a terminação de um fio ao outro, encontra-se uma resistência de 100 Ω. Com um valor medio de 1.25 V e corrente de 3.5 mA utilizada pra excitar a resistência, criando assim -350 mV ou +350 mV de diferença potencial na resistência, sinalizando assim o "0" ou "1" [45]. Na figura 22 a) e b) podemos verificar as tensões e correntes utilizadas em LVDS e CMOS.

O LVDS oferece várias vantagens quando comparando com o CMOS, o uso de fios com terminais bifurcados, reduz drasticamente as reflexões, mantendo igualmente o sinal em longos fios e em casos de grande velocidade de transmissão [45]. O uso de pequenas diferenças de potencial reduz a interferência com os fios das proximidades, especialmente num pacote de cabos. Interferências eletromagnéticas são igualmente minimizadas utilizando os sinais LVDS. A corrente ser constante é outro ponto a favor em relação à utilização do LVDS, não introduzindo mais ruído [45]. Por tudo isto, os sinais LVDS são aconselhados em aplicações de baixo ruído, especialmente num chip que contem componentes digitais e analógicos.

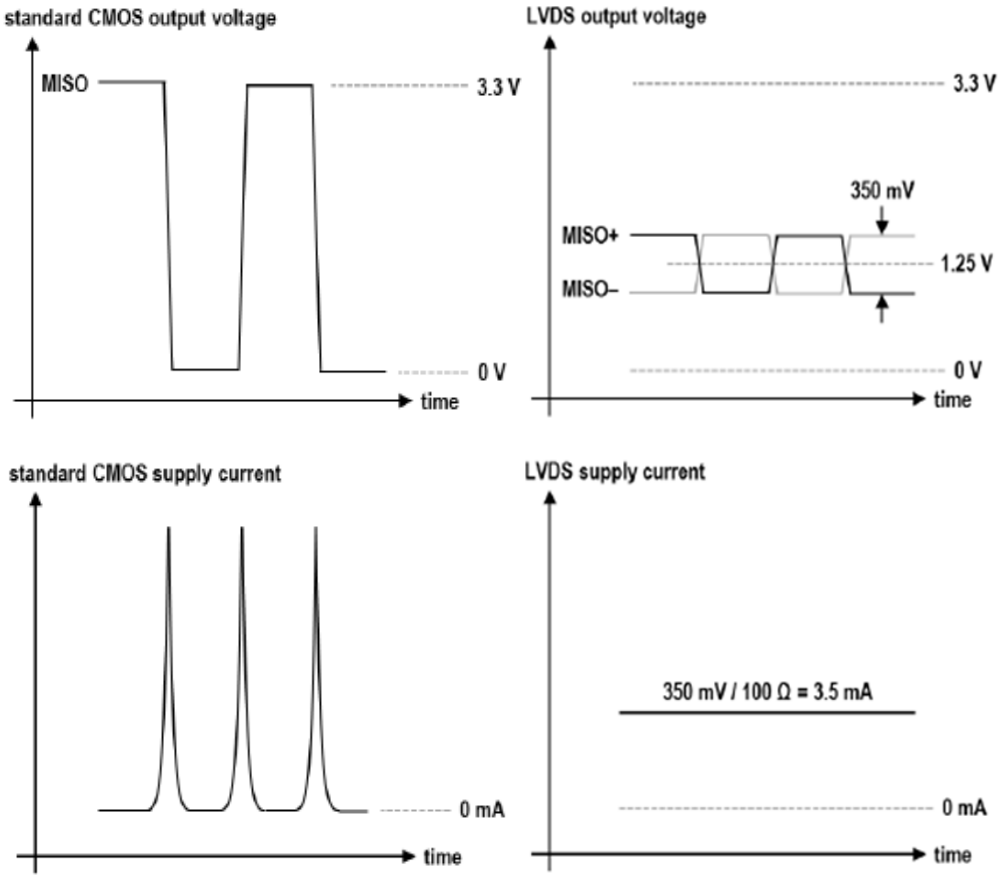


Figura 22-Niveis de tensão e corrente: a) CMOS; b) LVDS [45].

Em termos de consumos, o VSTIM com alimentação de 3.6 V, consome aproximadamente 0.98 mA [45]. O chip RHS2116 necessita de 3.2 mA para alimentar as várias tensões de referências, circuito ADC e gerador de corrente *bias*. Cada amplificador AC tem um consumo aproximado de 9.2 µA/kHz. A montagem ADC/*Multiplexer* consome cerca de 5.2 µA/(*kSamples/s*), a utilização do LVDS consome uns adicionais 5.7 mA [45].

Como referido anteriormente para programar esta placa são utilizados comandos SPI. O RHS2116 é controlado através de quatro comandos. É possível, escrever num registo RAM, ler de um registo RAM ou ROM, converter um sinal de um determinado canal ou efetuar a calibração do ADC. Cada comando é constituído por 32 bits, sendo que a sua composição varia de acordo com o comando a ser enviado. Nestes 32 bits existem 4 bits refentes a *flags* que controlam várias funções. A *flag* U, a mais importante, quando programada para 1 ativa todos os *triggered registers* anteriormente programados, mas não ativados. A *flag* M, que quando ativa limpa o registo 40. Registo responsável pela deteção de voltagem excessiva na estimulação elétrica. A *flag* D é responsável pela ativação da conversão do amplificador DC, não utilizado neste sistema. Por último, a *flag* H, quando programada para 1, o filtro digital associado com o canal é colocado a 0 [45]. A comunicação com o chip possui o seguinte funcionamento como se verifica na figura 23.

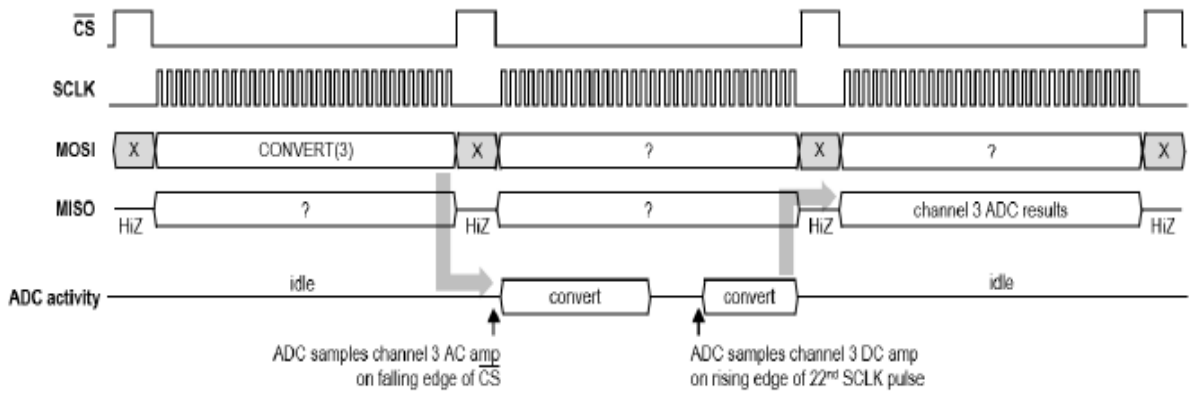


Figura 23-Funcionamento comandos SPI [45].

Na figura 23 é utilizado o exemplo de um comando de conversão do canal 3. Analisando a figura 23 percebe-se que a resposta a um comando é enviada posteriormente a serem enviados mais dois comandos. Ou seja, após ser enviado o primeiro comando, tem de ser enviado outro. Durante o segundo comando ocorre a conversão do sinal para digital, valor esse enviado pelo MISO aquando o envio do terceiro comando. Na figura 24 encontra-se um exemplo de um comando de escrita enviado com a *flag* U ativa, ativando consequentemente todos os *triggered registers* anteriores. *Triggered registers*, são registos do chip que possuem *buffers*, programados com o comando de escrita, mas são apenas atualizados com esses novos valores, quando a *flag* U é 1 [45].

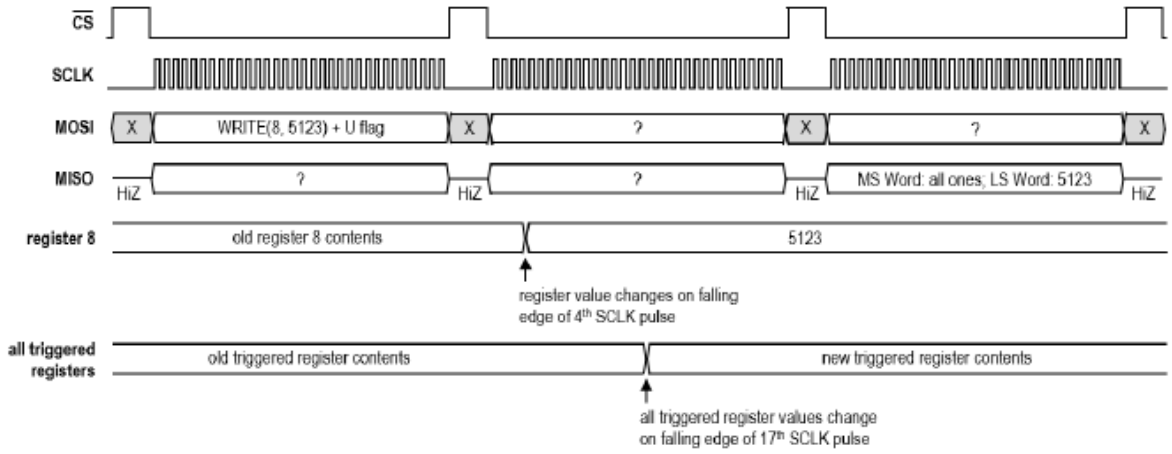


Figura 24-Funcionamento do comando para a ativação triggered registers [45].

Na figura 25 demonstra-se os bits a enviar quando se precisa de enviar o comando para a conversão de um canal. Os 16 bits menos significativos estão todos a 0. Os 16 mais significativos são divididos, estando os últimos 6 bits destinados para qual o canal a ser convertido. O valor dos bits correspondente às *flags* dependem do objetivo final. No pacote de resposta, os 16 bits mais significativos estão destinados para o valor convertido do amplificador

AC. Os 10 bits menos significativos correspondem ao valor do amplificador DC, se a *flag* D tiver sido enviada a 1 [45].

MSB 31	30	29	28	27	26	25 – 22	21 – 16	LSB 15 – 0
0	0	U	M	D	H	0000	C[5:0]	00000000 00000000

Result:

MSB 31 – 16	15 – 10	LSB 9 – 0
A[15:0]	000000	W[9:0]

Figura 25-Comando de conversão [45].

Os bits a enviar no comando de escrita num registo, são de acordo a figura 26. O comando de escrita num registo, escreve num registo R, a *data* D. Sendo D, o valor que o utilizador quer colocar no registo R. Na resposta serão recebidos nos 16 bits menos significativos a *data* D enviada no comando de escrita. Os 16 bits mais significativos são todos 1.

MSB 31	30	29	28	27 – 24	23 – 16	LSB 15 – 0
1	0	U	M	0000	R[7:0]	D[15:0]

Result:

MSB 31 – 16	LSB 15 – 0
11111111 11111111	D[15:0]

Figura 26-Comando de escrita num registo [45].

O comando de leitura é semelhante ao de escrita, figura 27, sendo que no bit 30, o valor é 1 ao invés de 0. Os 16 bits menos significativos são todos 0, pois não são enviados dados para o registo. Na resposta os 16 bits menos significativos são compostos pela informação que contem o registo R enviado pelo comando.

MSB 31	30	29	28	27 – 24	23 – 16	LSB 15 – 0
1	1	U	M	0000	R[7:0]	00000000 00000000

Result:

MSB 31 – 16	LSB 15 – 0
00000000 00000000	D[15:0]

Figura 27-Comando de leitura de um registo [45].

Por último, o comando de calibração (figura 28) é utilizado para inicializar o ADC. Deve ser executado logo que o chip é ligado para assim maximizar a precisão do ADC. A resposta a este comando consiste em tudo 0, como se verifica na figura 28, à exceção do bit mais significativo. Bit esse que será 0 se o bit relacionado com o modo complementar do registo 1 estiver ativo, senão será 1.

MSB 31	30	29	28	27	26	25	24	23 – 0 LSB
0	1	1	0	1	0	1	0	00000000 00000000 00000000

Result:

MSB 31	30 – 0 LSB
*	0000000 00000000 00000000 00000000

Figura 28-Comando de calibração [45].

Se algum comando enviado for desconhecido para o chip, como por exemplo qualquer comando começado por '01' que não seja o de calibração, a resposta do chip consistirá em tudo 0, à exceção do bit mais significativo. Como na resposta ao comando de calibração, depende aqui também do bit relacionado com o modo complementar do registo 1.

Para que a *headstage* seja bem inicializada, aquando da sua ligação, deve ser primeiramente programada com alguns comandos, de forma a melhorar o seu funcionamento. É sempre boa prática ser inicializado com o envio de um comando de leitura de um registo da memória ROM. De seguida o registo 38 deve ser totalmente ativado, pois devido a um erro de *hardware*, pode estar desativo e nesse caso consome adicionalmente 30.9 mA por cada amplificador DC desligado. Os registos 32 e 33 refentes à estimulação elétrica devem ser programados para 0 para assim assegurar que a estimulação elétrica se encontra desabilitada. O comando calibração deve ser enviado antes de qualquer comando de conversão para assim otimizar o ADC. Todos os outros registos devem ser inicializados consoante o objetivo do utilizador.

LVDS adapter board

Devido à utilização da placa da Intan que contém o chip RHS2116, os níveis de tensão na comunicação de SPI, são níveis LVDS. Como se pode ver na figura 20 o pino relativo à ativação ou desativação do LVDS está ligado ao VCC, o que não nos permite alterar para níveis de tensão CMOS.

A utilização de níveis LVDS levou a um problema pois a placa BLE utilizada nesta dissertação não está preparada para a comunicação por SPI com sinais de nível LVDS, levando assim à utilização de uma placa de conversão dos sinais de LVDS em níveis CMOS. A Intan LVDS Adapter board traduz *standard* CMOS em LVDS utilizados nas placas da serie RHS200, RHD200 ou CLAMP.

A placa possui um integrado SN65LVDT41 que faz a conversor de CMOS para LVDS. A figura 29 mostra um esquemático simplificado dessa interface, dos seus drivers e recetores na Adapter board. Para um valor logico baixo "0", a tensão tem de variar entre 0 V e 0.8 V enquanto para um valor lógico alto, a tensão poderá variar entre 2 V e 5 V, sendo posteriormente convertidos para sinais LVDS [47]. O valor recebido pelo MISO vindo da placa de gravação de sinais eletrofisiológicos, é convertido em nível CMOS de 3.3 V [47].

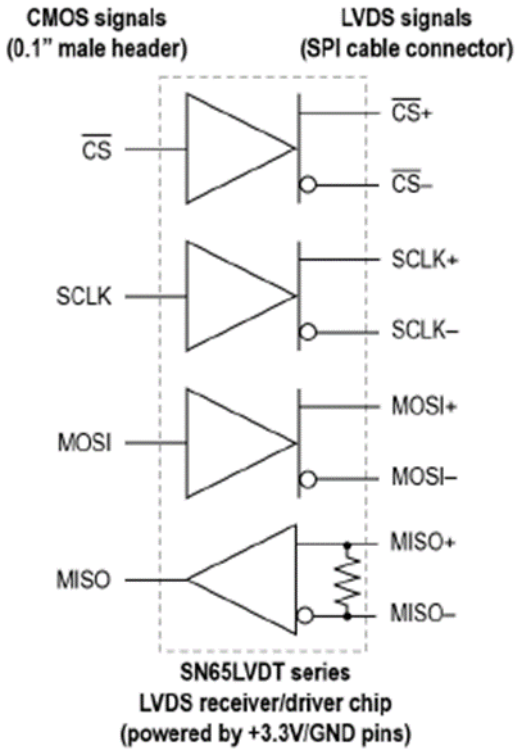


Figura 29-Esquematico simplificado LVDS Adapter board [47].

Estimulação

O objetivo do sistema, para além de mostrar os sinais neurais, é mostrar como a estimulação optogenética influencia o cérebro dos animais testados. Por isso o sistema tem de ser capaz de atuar um LED. O LED utilizado na sonda do projeto *Brain-Lighting* é o ELC-470-37 da empresa Roithner [48].

O LED será ativado usando a saída auxiliar da *headstage* da Intan. Possui uma corrente direta de 20 mA e um pico de corrente de 100 mA para um tempo menor que 50 µs [48]. Como se pode ver pela figura 30, o LED é constituído por um N pad, um P pad, um substrato em safira e uma camada de grafiti GaN.

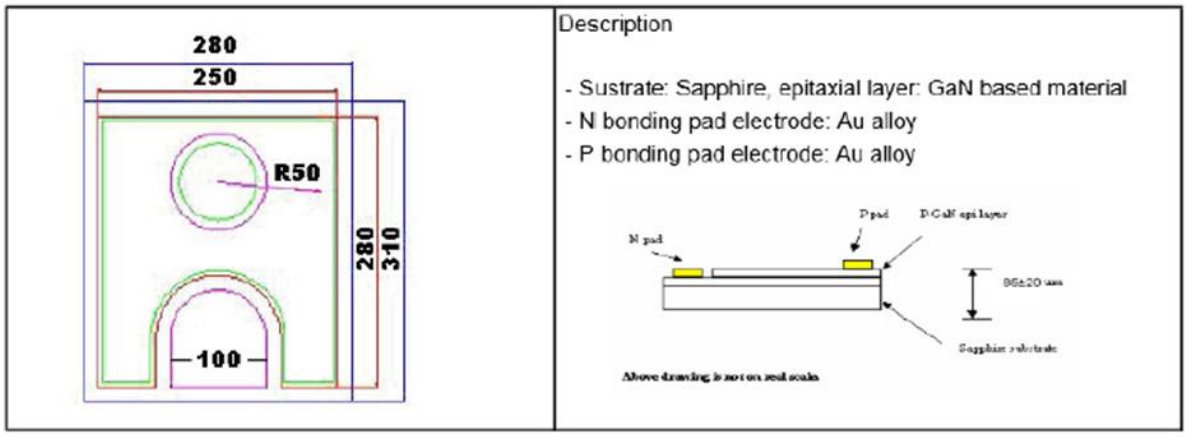


Figura 30-Descrição do LED ELC 470-37 [48].

Sistema de comunicação

Na presente dissertação o sistema de comunicação sem fios será realizado usando o BLE. O menor consumo em relação a outra tecnologia de comunicação sem fios foi predominante para a sua escolha, em detrimento de uma maior taxa de transmissão. Os módulos de BLE escolhidos são da empresa Cypress Semicondutor. É uma empresa que tem como alvo mercados em crescimento rápido como a indústria automóvel, industrial e eletrónica de consumo. São especialistas em tecnologias sem fios, assim como em Microcontroladores (MCUs), controladores USB, e circuitos integrados (ICs) analógicos, tendo assim vantagem competitiva no mercado da *internet of things* (IoT) [49]. O Módulo BLE utilizado pertence à família PSoC 4 BLE. PSoC significa *Programmable System on Chip* e é uma família de microcontroladores da Cypress. O seu circuito integrado é composto por uma unidade central de processamento (CPU), blocos analógicos e digitais configuráveis e programáveis. Os blocos configuráveis é o que diferencia o PSoC dos outros microcontroladores [49]. O IDE utilizado para programar o microcontrolador é o IDE da Cypress, o PSoC Creator.

Um dos módulos é uma Dongle para ligar ao PC e receber os dados, e o outro módulo estará conectado à placa da Intan, referida anteriormente, para enviar os sinais eletrofisiológicos digitais para o computador.

PSoC 4 BLE

O PSoC 4 BLE oferece várias características como sistemas analógicos inteligentes, recursos digitais programáveis em adição a um vasto número de funções como ADCs de alta performance, *timers*, *counters* e Pulse width modulation (PWM). Possui protocolos de comunicação série como o I2C, UART e SPI, tudo através da sua arquitetura programável, que é o grande ponto de diferenciação deste módulo, integrado com o chip CY84C4247LQI-BL483. A figura 31 representa o diagrama de blocos do módulo PSoC 4.

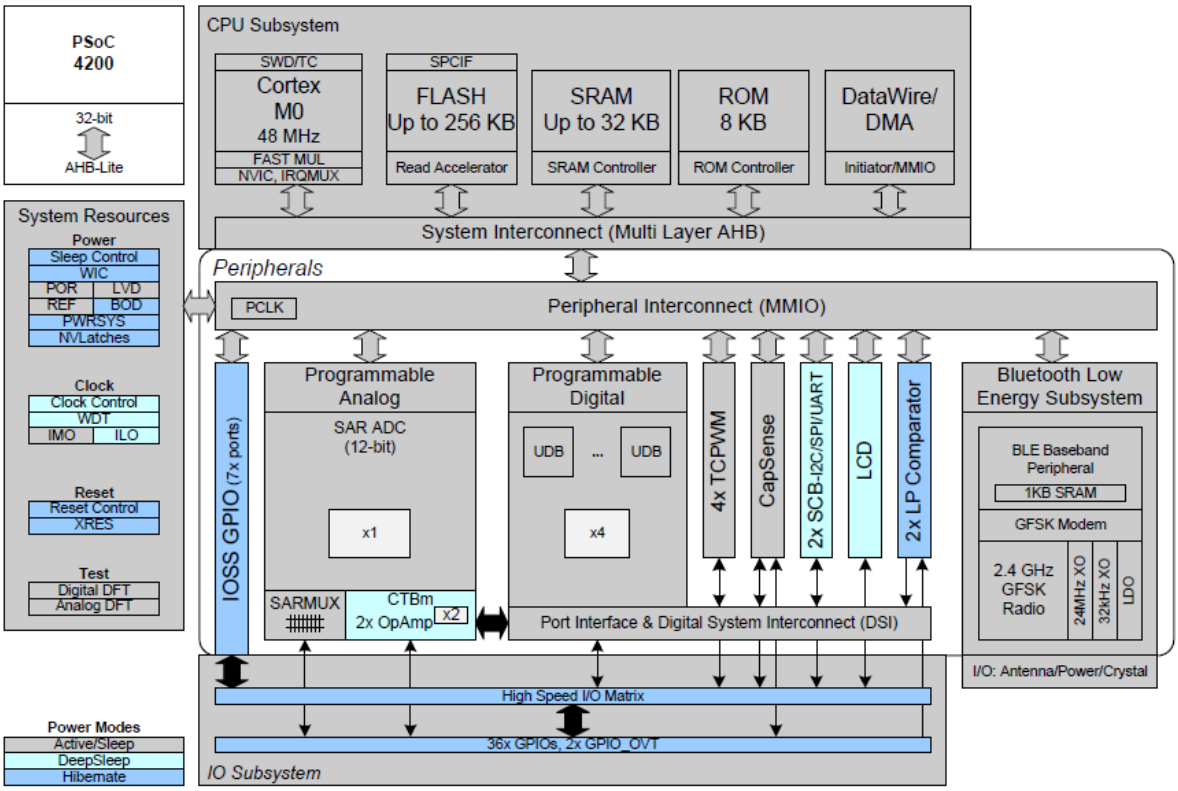


Figura 31-Diagrama de blocos PSoC4 [50].

O subsistema CPU possui um processador ARM Cortex M0 com 32 bits e uma velocidade de 48 MHz, processador este otimizado para operações de baixa energia. Funciona com tensão entre 1.71 V e 5.5 V sem transmissão de rádio frequência e com rádio frequência funciona entre 1.9 V e 5.5 V. O dispositivo tem diferentes modos de funcionamento, desde *Sleep, Deep Sleep, Hibernate* e *Stop-low power*. Como podemos ver pela figura 33, faz parte também do subsistema CPU, a memória do microcontrolador. A memória flash possui 256 kB, está junta com o processador para assim ser mais rapidamente acessível. Possui um tempo de *reset* e de reprogramação de 20 ms por cada 256 bytes. A memória SRAM é a memória retida

durante a hibernação e tem capacidade de guardar até 32 kB. A memória SROM possui 8 kB de memória, armazenando funções executáveis.

O sistema de *clock* é responsável por assegurar o funcionamento de todos os *clocks* e por trocar de *clock*, sem que ocorra nenhuma falha, dependendo do subsistema a ser utilizado. O *clock* consiste no *Internal main oscillator* (IMO), *Internal low-speed oscillator* (ILO), 24 MHz *External crystal oscillator* (ECO) e num 32 kHz *Watch crystal oscillator* (WCO). Em adição a estes *clocks* ainda é suportada a ligação de um *clock* externo, ligado através de um pin. O IMO tem um valor predefinido de 24 MHz e pode ser ajustado entre 3 a 48 MHz com incrementos de 1 MHz. O ILO é o *clock* geralmente utilizado quando o dispositivo se encontra no modo *Deep Sleep*, O W*atch crystal oscillator* é utilizado como um *sleep clock* para que o subsistema BLE atinja os objetivos de precisão do *clock* de +- 500 ppm. Igual função tem o ECO, mas como *clock* de ativação [50]. Na figura 32 encontra-se a arquitetura do *clock* do PSoC 4200_BL.

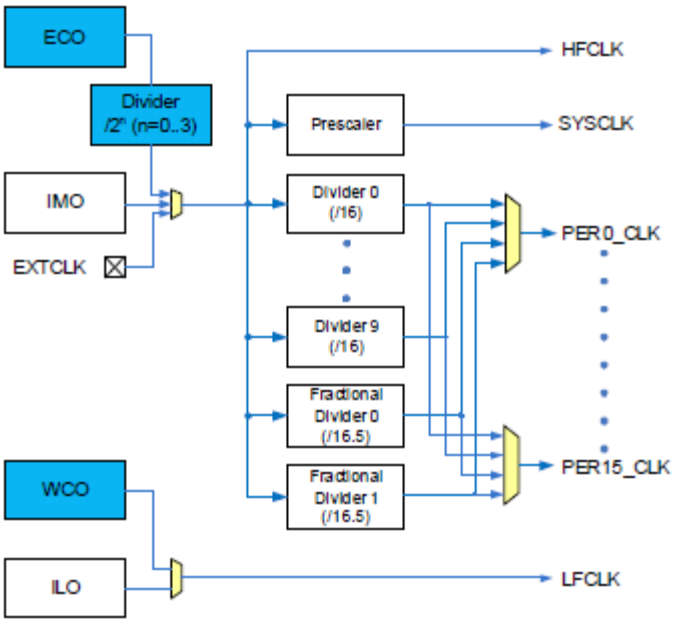


Figura 32-Arquitetura do clock PSoC 4 BLE [50].

Os periféricos existentes neste módulo são vários, um ADC com um *clock* de 18 MHz e 12 bits de conversão, 4 blocos digitais de programação e funcionalidades digitais como *counter*, PWMs, *timers* e comunicação serie como o I2C, modo UART e SPI, a utilizada nesta dissertação.

O subsistema BLE contem a camada física e de ligação, funcionando como explicado no capítulo do estado de arte sobre o BLE. Funciona na banda dos 2.4 GHz e com Bluetooth 4.2. A grande diferença entre o Bluetooth 4.2 e o 4.1 é o tamanho admitido nos pacotes de

transmissão de dados, ver figura 33, sendo que no Bluetooth 4.2 consegue enviar até 256 bytes de dados, ao contrário do Bluetooth 4.1 que consegue apenas 26 bytes [51]. O PSoC 4 BLE possui uma taxa de transmissão teórica de 784 kbps [51]. Visto que cada sample possui 16 bits de resolução, na teoria conseguimos apresentar uma taxa de amostragem de 3 *kSamples/s*, isto sem contar com os obstáculos na transmissão e o tempo de conversão dos canais, o que vai reduzir ainda mais a taxa de amostragem do sistema. Segundo o *datasheet*, com um intervalo de conexão de 1 segundo, o consumo durante a transmissão é de 18.4 µA [51].

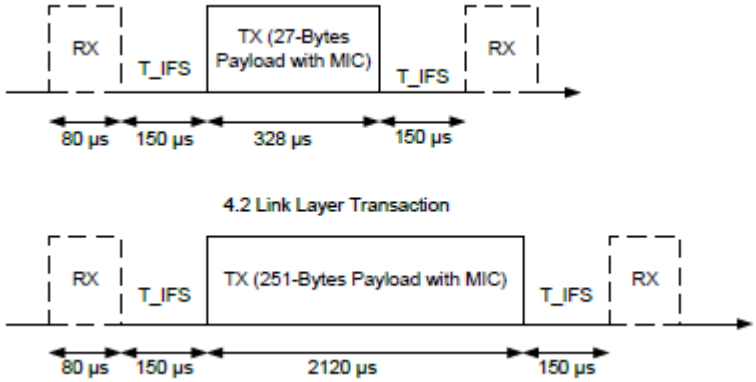


Figura 33-Diferença entre Bluetooth 4.1 e 4.2 no pacote de transmissão [51].

No subsistema de I/O, este módulo possui 36 GPIOs, cada pino pode possuir 8 modos:

- Entrada analógica;
- Entrada apenas;
- *Pull-up* fraco com *pull-down* forte;
- *Pull-up* forte com *pull-down* fraco;
- *Open drain* com *pull-down* forte;
- *Open drain* com *pull-up* forte;
- *Pull-up* forte com *pull-down* fraco;
- *Pull-up* fraco com *pull-down* fraco.

Para além dos 8 modos diferentes, em cada pino é possível selecionar a tensão de entrada (CMOS OU LVTTL). Os pinos 0 e 1 da porta 5 suportam sobretensão, controlo individual do *buffer* de entrada e saída em adição do controlo individual do modo de cada pino. Cada pino pode gerar a sua própria interrupção e cada porta tem uma solicitação de interrupção [50]. Na figura 34 encontra-se a conexão dos pinos do módulo BLE utilizado.

A placa de desenvolvimento PSoC 5LP, através do IDE PSoC Creator, foi utilizada para a programação do módulo PSoC 4.

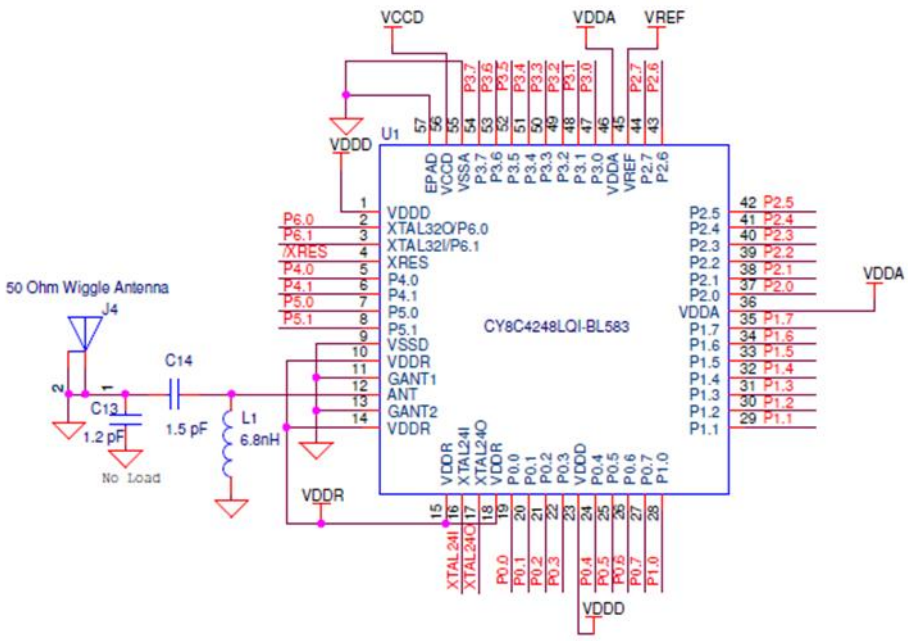


Figura 34-Esquemático módulo PSoC 4 BLE [52].

CY5671 Dongle

A CY5671 CySmart Bluetooth USB Dongle é um conversor de BLE para USB utilizada no desenvolvimento e depuração em aplicações BLE. Possui dois chips, um PRoC BLE (CYBL10162-56LQXI) e um PSoC 5LP (CY8C5868LTI-LP039).

O PRoC BLE é um chip com um processador, 48 MHz ARM Cortex m0, idêntico ao do módulo explicado em cima, possui rádio BLE, ADC de 12 bits e 4 *timers/counter*/PWM. O PSoC 5 LP utiliza um ARM Cortex M3 sendo o chip destinado para a programação e depuração [53]. Na figura 35 encontra-se a descrição do *pinout* da USB Dongle utilizada, como se pode ver para além dos chips falados em cima também possui dois botões, um botão que pode ser programado pelo utilizador e um botão de *reset*.

O CySmart USB Dongle vem programado de fábrica para funcionar como um dispositivo Central, permitindo assim a depuração de programas implementados utilizando como por exemplo o módulo PSoC 4 BLE. Para tal utilização conta com um *software* providenciado pela Cypress chamado de Cypress *CySmart Windows*. Através deste *software* é possível aceder, configurar e descobrir as informações Bluetooth. No sistema desenvolvido nesta dissertação, esse *software* foi substituído por um *software* desenvolvido em C#.

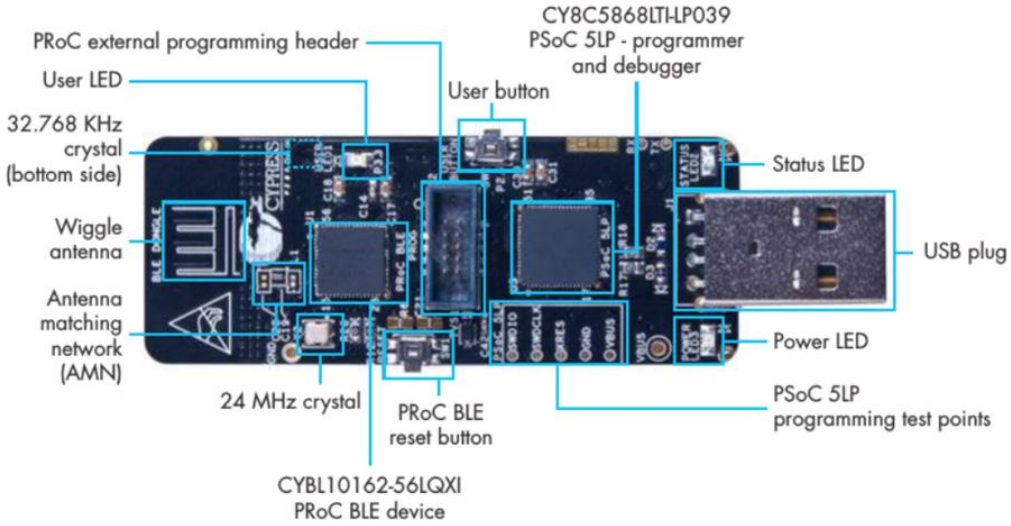


Figura 35-Descrição pinout CySmart USB Dongle [53].

Interface gráfica

Na base desta dissertação encontra-se a interface gráfica, após a aquisição dos sinais, do seu processamento e da sua transmissão, precisávamos de um *software* para visualizar os sinais eletrofisiológicos em tempo real. Por isso foi desenvolvido um *software* em C#, no IDE *Microsoft Visual Studio*. Através deste *software* para além de podermos observar os dados também queremos ser capazes de comunicar com a placa Bluetooth para a ativação da estimulação optogenética.

Como requisitos principais, a aplicação tem de ter uma interface gráfica"amigável" para o utilizador alvo (neurocientistas) conseguirem utilizar sem problemas. Precisa obviamente de ser capaz de se conectar ao módulo PSoC 4 BLE. Para além de mostrar os dados em gráficos, deve ser também capaz de os gravar num ficheiro de texto para assim serem posteriormente analisados. Outro requisito é a capacidade para ativar a estimulação optogenética e poder observar um gráfico com a tensão do LED, para assim saber quando estão ativos e como essa estimulação influenciou o animal nesse intervalo de tempo. Na figura 36 encontra-se a imagem frontal da interface gráfica desenvolvida para este sistema.

O *design* da aplicação teve em conta o facto de serem apresentados 16 canais em simultâneo (figura 36). Optou-se por mais de metade da aplicação ser para a apresentação dos 16 canais. O resto da aplicação trata das várias características do sistema. Do lado direito para além de ser possível observar os gráficos com os dados dos elétrodos neuronais, também é possível ativar/desativar um gráfico. Ao selecionar com o botão direito do rato no gráfico a ser

desativado, o botão de ativação/desativação no separador *Channels* é ativado e ao clicar nesse botão, é possível ativar/desativar o gráfico selecionado.

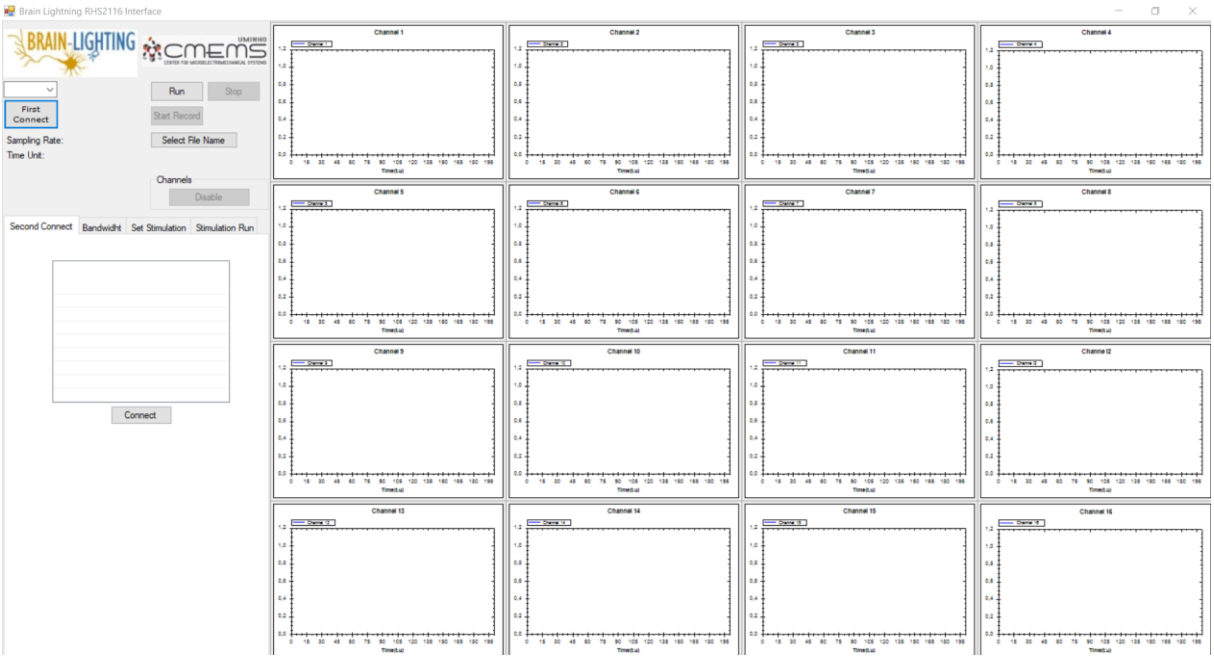


Figura 36-Imagem frontal da interface gráfica desenvolvida.

No lado esquerdo da aplicação, como já foi referido, é possível controlar vários aspetos do sistema. É possível começar ou parar o sistema com o com o botão *Run* ou *Stop*, sendo que o botão *Stop* só se encontra ativo quando o sistema está a funcionar, e o botão *Run* encontra-se ativo quando o sistema se encontra parado.

Através da aplicação é possível ver a taxa de amostragem e o *time unit*. Taxa de amostragem é o número de amostras que estão a ser recebidas por cada canal, o *Time unit* é o tempo que passa entre cada ponto do gráfico, para assim ser possível ver o tempo que passou entre cada amostra recebida. Quanto menos gráficos estão ativos maior é a taxa de amostragem e menor é o *Time unit*. Esses valores também são guardados no ficheiro de texto. Na figura 37 encontra-se uma imagem mais pormenorizada da parte superior esquerda da aplicação.

Para ativar a gravação, primeiramente é necessário escolher o caminho e o nome do ficheiro através do botão *Select File Name*, botão que abre uma janela (ver figura 38) para o utilizador escolher o melhor local e o nome para o ficheiro. Ao escolher o nome e o caminho, o botão *Start Record* fica ativo, sendo necessário clicar nesse botão para ativar a gravação. Serão apenas guardados os dados dos gráficos ativos, reduzindo assim o ficheiro de texto guardado apenas aos dados necessários, evitando tamanhos de ficheiro demasiado grandes.

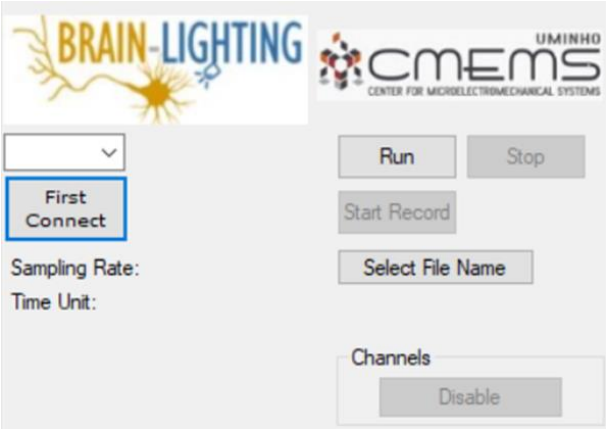


Figura 37-Imagem pormenorizada da parte superior esquerda da aplicação.

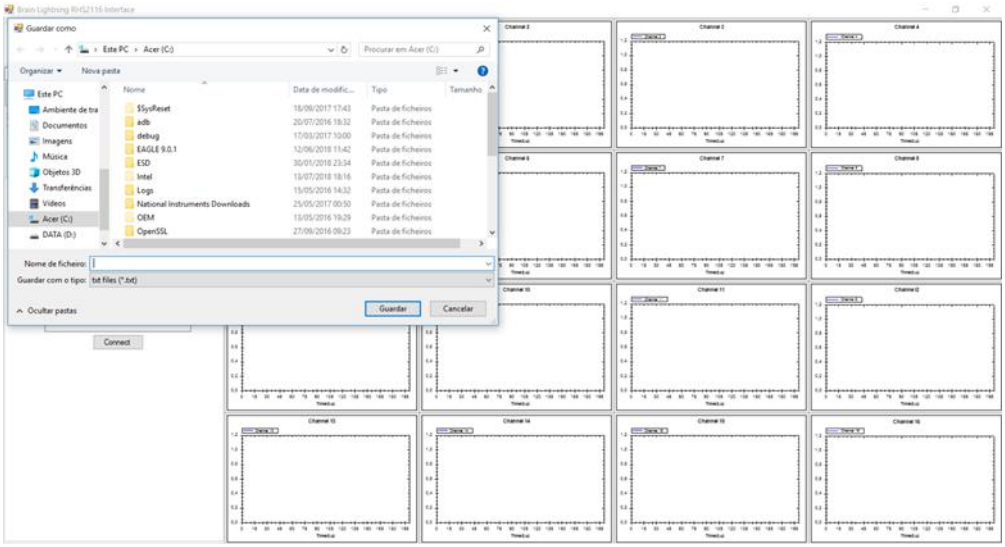


Figura 38-Janela aberta para criar e guardar ficheiro com os dados da interface gráfica.

Na parte de baixo da aplicação temos 4 separadores, como se pode ver pela figura 39. O mais à esquerda é para a conexão ao módulo Bluetooth, na seguinte permite alterar as definições dos filtros tanto passa-alto como passa-baixo, estando predefinidos para 100 Hz e 1 Hz.

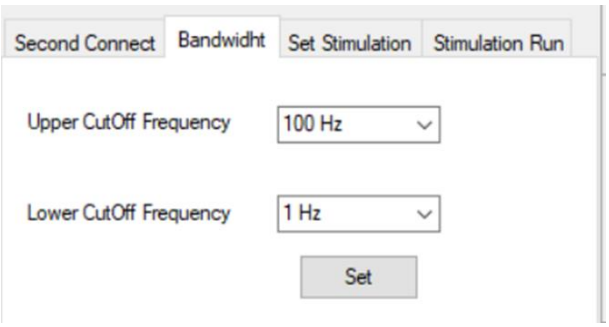


Figura 39-Separador para modificar largura de banda.

Os dois separadores seguintes tratam da estimulação optogenética. No separador *Set Stimulation* (ver figura 40 a)) existem dois modos para a ativação da estimulação optogenética. Ao clicar no botão *Mode 0*, podemos definir o tempo *On*, tempo *Off* e o número de períodos. Esses valores servem para definir os tempos em que o LED vai estar ligado, desligado e quantas vezes se irá repetir. No *Mode 1* é possível definir até 5 períodos diferentes, com tempos a *On* e *Off* diferentes e depois o número de ciclos que esses períodos se repetem. Após selecionado o modo, e inseridos os tempos a *On*, *Off* e o número de ciclos, no separador a seguir chamado *Stimulation Run* ao clicar no botão *Enable* ativamos a estimulação optogenética. Nesse separador (figura 40 b)) para além da ativação da estimulação optogenética, é apresentado um gráfico com a tensão do LED.

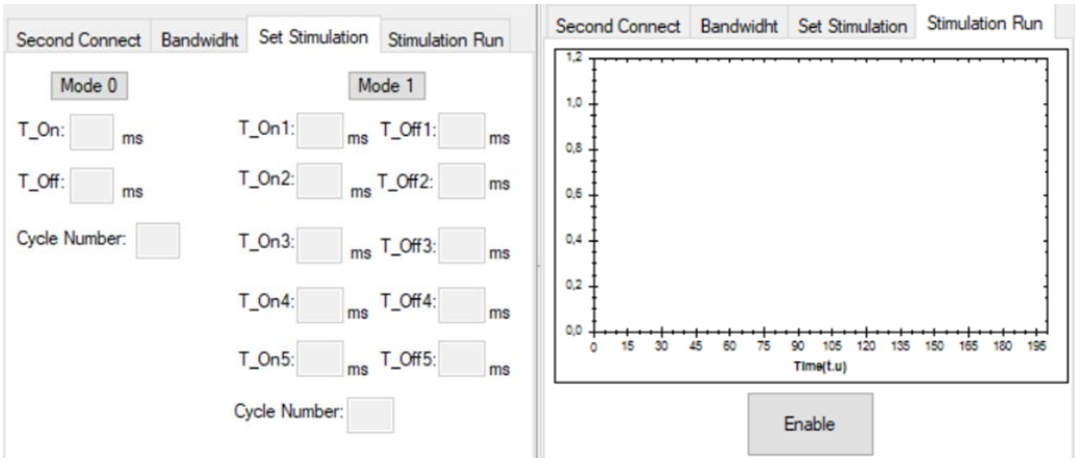


Figura 40- a) Separador de preparação da estimulação; b) Gráfico para visualização de tensão do LED de estimulação optogenética.

DESIGN DO *SOFTWARE*

O desenvolvimento de todo o sistema de comunicação sem fios compreende uma série de passos principais e intermédios até ao produto final estar completo. No presente capítulo é explicado o *design* do *software* desenvolvido nesta dissertação, são abordadas as interações entre as várias placas utilizadas e a interface gráfica. São também apresentados fluxogramas referentes a várias funções do *software* desenvolvido no microcontrolador PSoC 4 e na interface gráfica.

Primeiramente foi estudado e testado o módulo Bluetooth, o seu funcionamento e quais as funções existentes. Após descobrir como trocar informação entre o computador e o módulo BLE, e testado esse funcionamento através da aplicação *CySmart Windows*, foi dado início ao estudo da *headstage*. Começou por ser estudado as suas características, seguido do estudo do funcionamento da comunicação com o módulo Bluetooth, os vários comandos existentes e o estudo de cada registo no sentido em como estes alteravam o funcionamento da placa. Após perceber o seu funcionamento e como a comunicação funcionava, começou-se a desenvolver a comunicação SPI entre o módulo BLE e a *headstage*.

Depois de conseguir comunicar com a *headstage*, alterar o valor dos registos, ativar a conversão de canais e ler as respostas aos comandos enviados, começou a ser desenvolvida a aplicação em C# no Visual Studio. Primeiramente, foi "construída" a conexão entre computador, CY5671 Dongle e PSoC 4 BLE, após todas as conexões estarem conseguidas, começaram a ser desenvolvidas tentativas de comunicação entre a interface gráfica e a *headstage.* A partir deste ponto tanto a interface gráfica como o módulo PSoC foram sendo programados em paralelo, quando uma funcionalidade era acrescentada na interface gráfica, o mesmo tinha de acontecer no módulo BLE para fazer a ponte com a *headstage*.

Arquitetura do *software*

Na figura 41 podemos verificar as interações que ocorrem entre o módulo BLE, PSoC 4, a *headstage* da Intan RHS2116 e a interface gráfica. A interface gráfica conecta-se à CY5671 USB Dongle, e comunica com o módulo PSoC 4 BLE. Este por sua vez, para além de comunicar com o módulo BLE também comunica com a *headstage* através de SPI.

Como já foi referido anteriormente, a placa PSoC 4 BLE serve de ponte entre a interface gráfica e a *headstage*. Isto é, por exemplo, quando o utilizador pretende ativar a conversão dos canais, clica no botão destinado a essa ação, de seguida é enviado para o PSoC 4 o pedido para

que as notificações sejam ativas, ao receber este pedido serão enviados automaticamente os comandos SPI necessários para a conversão dos canais. Os sinais ao serem digitalizados, são enviados de volta através de SPI para o PSoC 4 que ao detetar presença de dados no *buffer* de receção do SPI, processa esses dados para serem enviados de volta para a interface gráfica. O mesmo procedimento ocorre quando o objetivo é alterar a largura de banda dos filtros ou ativar/desativar algum canal, sendo que, o que é enviado de volta nestes casos é a confirmação da alteração.

Posteriormente serão detalhados, através de diagramas e fluxogramas o comportamento interno da placa PSoC 4 BLE e da interface gráfica. Serão explicadas as classes e as suas funções através de diagramas UML. Após serem explicadas as classes, as suas interações e as funções presentes em cada classe, será apresentado um fluxograma sobre o funcionamento mais detalhado do código. A *headstage* como responde ao que lhe é pedido através de comandos SPI, comandos esses já explicados anteriormente, não será explicado o seu funcionamento interno.

Por último será apresentada a implementação das partes mais importantes do código da interface gráfica e do módulo BLE assim como será apresentado e explicado o protocolo de comunicação entre a interface gráfica e o módulo BLE. Protocolo este que foi criado, para que quando a interface gráfica precisar de enviar informação para a *headstage*, o módulo BLE seja capaz de "entender" a informação enviada e consiga acionar as funções corretas para o objetivo definido pelo utilizador.

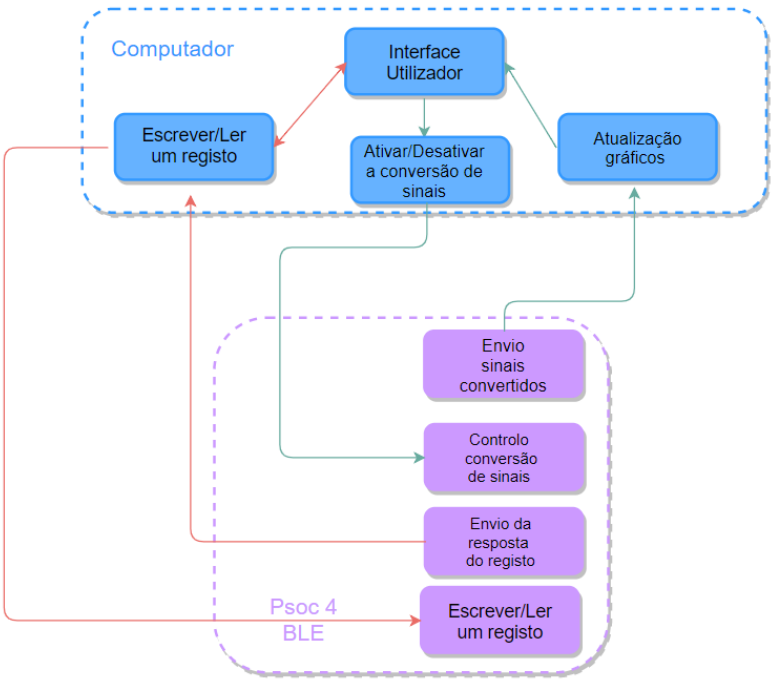


Figura 41-Diagrama da arquitetura do software.

Arquitetura PSoC 4 BLE

Agora será explicado detalhadamente o *software* desenvolvido no PSoC 4 BLE, na figura 42 encontra-se o diagrama dos ficheiros utilizados. Primeiramente foi criado um perfil para o módulo BLE onde o PSoC 4 BLE ficou definido como funcionando no modo *Peripheral* e a *dongle* no modo *Central*. Seguidamente foi criado um serviço com o nome *Communication,* com duas características. A característica *Interface_Communication* que trata da comunicação bidirecional com a interface gráfica. E a característica *ADC_Data* que envia por notificação os valores dos canais. Na programação do módulo BLE decidiu-se dividir o código em quatro ficheiros, *main.c*, *ble.h*, *rhs2116.h* e *características.h*. Primeiramente, o ficheiro *main.c* que é o ficheiro principal, encontra-se em contacto com o resto dos ficheiros e mantem o funcionamento do sistema através de um ciclo infinito. O ficheiro *ble.h* destina-se a todas as funções que tratem de enviar ou receber informação vinda da interface gráfica. O ficheiro *rhs2116.h* possui todas as funções que tratam de comunicar entre o PSoC 4 BLE e a *headstage* da Intan. Por último, o ficheiro *características.h* trata de todas as funções que tenham como objetivo alterar o comportamento do sistema, como por exemplo alterar a largura de banda do filtro ou ativar a conversão dos ADCs.

O ficheiro *main.c* é o ficheiro principal, trata de iniciar o sistema, com a função *initSystem(),*que ativa as interrupções, o SPI e os eventos da *stack* BLE, para além disso mantém um ciclo que verifica a *flag ble_notification* para saber quando as notificações estão ativas e pode ser chamada a função *características_sendADCData.*

O ficheiro *ble.h* tem como objetivo tratar os dados recebidos pela interface gráfica, analisá-los e decidir o que fazer com eles, reencaminhando-os para o ficheiro *rhs2116.h* ou *características.h* consoante o seu objetivo final. Para além disso também recebe dados das classes *rhs2116.h* e *características.h*, enviando-os por BLE para a interface gráfica. Primeiramente dentro do ficheiro *ble.h* existe a função *ble_init(),* função esta que é chamada pela *main.c,* ativando as interrupções e os eventos da *stack* BLE. Esta função é chamada através da função *initSystem()*. No que a receção de dados vindos da interface gráfica diz respeito a função *eventHandler(uint32,void*)* é a principal, sendo que esta função recebe qualquer acontecimento vindo da interface gráfica, quer seja a ativação das notificações ou a receção de comandos. Para além disto também negoceia, no início da conexão, o tamanho dos pacotes a serem trocados. Ao receber a ativação das notificações, a função *eventHandler(uint32,void*)* vai ativar a *flag ble_notification.* Se o que for enviado é um comando, será colocado num array e chamada a função *ble_handleTxData(uint8*),* função esta que trata de perceber consoante o

primeiro byte qual o objetivo do comando que foi enviado. As funções *ble_sendData(uint8*)* e *ble_sendADCData_overNotification()* tratam de enviar dados para a interface gráfica. A função *ble_sendData(uint8*)* envia para a interface gráfica, informação quando é chamada, normalmente como resposta a comandos para a verificação de receção. A função *ble_sendADCData_overNotification()*, envia para a interface gráfica o valor dos sinais dos ADCs convertidos, somente quando as notificações se encontram ativas.

Em relação ao ficheiro *rhs2116.h*, pode ser dividido em duas partes. As funções referentes ao envio de comandos SPI e a função que recebe a reposta aos comandos SPI. Para o envio de comandos temos, a função *rhs_setRegData(uint8,uint8,uint8,uint8)* que envia o comando de escrita num registo, a função *rhs_getRegData(uint8,uint8)* lê de um registo, a função *rhs_Convert(uint8)* que faz a conversão de um canal, e a função *rhs_clear()* que tem como objetivo enviar o comando referente à calibração do ADC. Para a receção das respostas dos comandos, temos a função *rhs_handleRxData()*, que recebe do buffer SPI a resposta aos comandos e chama a função *ble_sendData()* para essa resposta ser enviada.

Por último, o ficheiro *características.h* como já foi referido inclui as funções capazes de alterar o sistema, a função *características_readADCData()*, é a função que faz a conversão de todos os canais ativos, a função *características_powerUpSystem()* trata do envio de todos os comandos necessários sempre que o sistema é ligado, comandos esses já mencionados anteriormente. Para alterar a largura de banda do filtro temos a função *características_setBandwidth(uint8,uint8)*. Na ativação e desativação de um canal, a função *características_turnOffChannel(uint8)* é a chamada à ação assim como a função *características_chandeLed()* ativa e desativa o LED da estimulação optogenética.

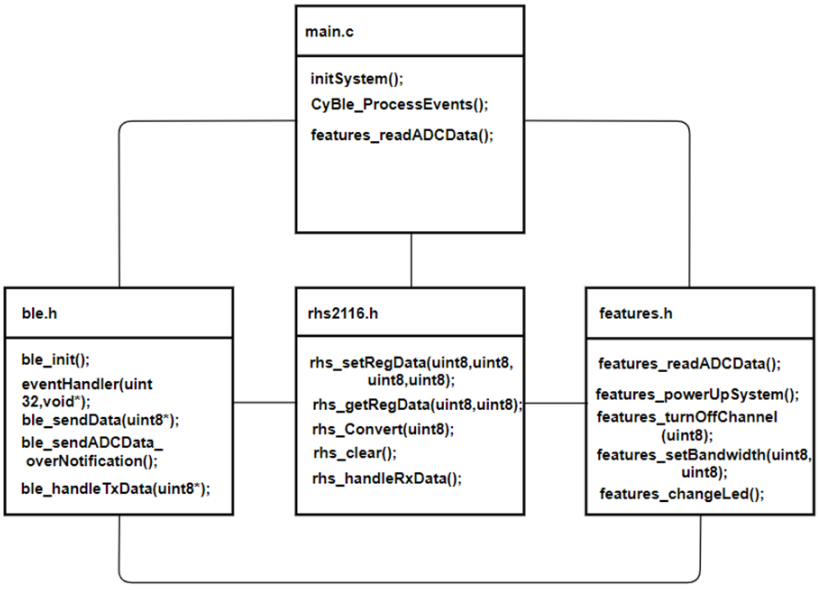


Figura 42-Diagrama dos ficheiros do código desenvolvido no módulo PSoC 4 BLE.

Arquitetura da interface gráfica

De seguida será explicado com mais detalhe a arquitetura da interface gráfica (ver figura 43), as suas classes, ficheiros e funções correspondentes. Como podemos ver na figura 43, o *software* foi divido em 4 ficheiros. O ficheiro principal e referente ao *design* da *interface*, *Interface.c*s, o ficheiro *BLE.c, Stimulation.cs* e *Graphics.cs.* Cada ficheiro contém uma ou mais classes.

O ficheiro principal, *Interface.c*, possui a classe parcial *Interface*, esta classe possui todos os métodos referentes ao *design* da interface gráfica, como por exemplo, qualquer botão, ou gráfico criado, será criado dentro desta classe e não acessível a mais nenhuma, por isso esta classe encontra-se presente em outros ficheiros. Dentro desta classe e neste ficheiro encontra-se o construtor, *Interface()*, função inicializada quando a aplicação é ligada, e que inicializa várias variáveis. Inclui igualmente funções referentes à ativação e desativação das notificações, a função referente ao botão utilizado para inicializar a gravação dos dados e para a escolha do nome e local do ficheiro, a função que efetua a gravação, o botão para a alteração da largura de banda e o *timer* 2, utilizado para o cálculo do valor da frequência de amostragem.

No ficheiro *BLE.cs* encontram-se as funções desenvolvidas para a conexão entre a interface gráfica, a CySmart USB Dongle a o PSoC 4. Funções essas divididas em três classes, a classe parcial *Interface*, a classe *GattClientCb* e a classe *BleMgrCb*. As classes *GattClientCb* e *BleMgrCb* tornam a interface gráfica no *Gap Client*, sendo que a classe *GattClientCb* possui as funções necessárias para a verificação se algo foi escrito ou escrever informação nas características, como por exemplo, verificar se foi recebido algum comando do PSoC 4, ou de ativar e desativar o valor das notificações. E a classe *BleMgrCb* verifica o estado de conexão ao dispositivo BLE. A classe parcial *Interface* possui neste ficheiro as funções referentes a conexão com os módulos BLE. Primeiramente a função *COMPort_DropDown(object, EventArgs)* utilizada para descobrir as portas USB a serem utilizadas, ao descobrir a COMX referente à CySmart USB Donlge e clicar nessa COMX, a função *DongleConnect_Click(object, EventArgs)* é chamada para fazer a conexão entre a interface gráfica e a Dongle. Após ser feita a conexão com a Dongle, a função *SetupScanHandler()* trata de apresentar os resultados referentes ao scan por dispositivos Bluetooth. Para a colocação dos dispositivos organizados numa lista é utilizada a função *FindName(int32,byte[])*. Após todos os dispositivos terem sido encontrados e apresentados na lista, o utilizador pode selecionar o que pretende e clicar no botão *Connect*, ao clicar nesse botão a função *ConnectButton_Click(object, EventArgs)* faz a conexão entre a interface gráfica e o módulo PSoC 4 BLE.

O ficheiro *Graphics.c* possui duas classes, a class *MyGraph* e a classe parcial *Interface*. A classe *MyGraph* foi desenvolvida com a biblioteca *ZedGraph* que contem api's necessárias para a implementação de gráficos com a receção de dados em tempo real. Nessa classe foram implementas as funções *MyGraph(ZedGraphControl,string)*, função construtora, a função *initGraph(string)* que inicializa o gráfico, e as funções *DrawGraph(List<double>)* e *Update()*, que tratam da escrita dos valores nos gráficos. A classe parcial *Interface* dentro deste ficheiro contem as funções que tratam da ativação e desativação dos gráficos. Ao se clicar com o rato num dos gráficos, a função *ZedGraphControlX_MouseClick(object,MouseEventArgs)*, onde X é o gráfico clicado com o rato, vai ativar o botão *disableChannel*, ao clicar nesse botão a função *disableChannel_Click(objects,EventsArgs)* vai ativar/desativar o canal consoante o seu estado anterior.

Por último, o ficheiro *Stimulation.cs* inclui apenas a classe parcial *Interface* com as funções referentes à estimulação. As funções *mode0_Click(object,EventArgs)* e *mode1_Click(object,EventArgs)*, ativam consoante o botão selecionado, os modos 0 e 1 para a estimulação, sendo que apenas um se pode encontrar ativo de cada vez. Para a ativação da estimulação, ao clicar no botão *Enable* debaixo do gráfico referente à estimulação, a função *enableStimulation_Click(object,EventArgs),* vai ativar o *timer1EventProcessor(object,EventArgs). Timer 1* é o responsável por enviar para o PSoC 4 o comando para a troca do valor do LED consoante os valores atribuídos para a estimulação.

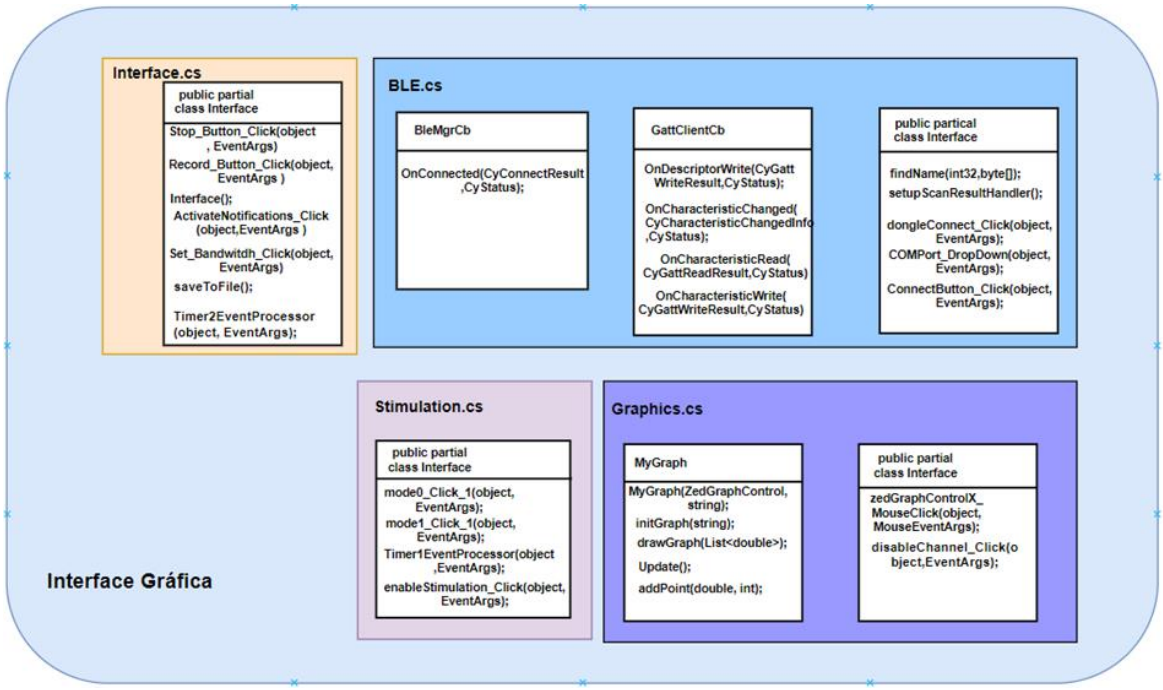


Figura 43-Ficheiros do código da interface gráfica.

Fluxogramas PSoC 4 BLE

Após a descrição da interação entre ficheiros e funções do *software* desenvolvido no PSoC 4 BLE ir-se-á explicar mais detalhadamente o funcionamento do código com recurso a fluxogramas. Serão apresentados diferentes fluxogramas para as diferentes etapas do código.

Inicialização do sistema

A figura 44 apresenta o fluxograma de inicialização do sistema. É possível observar a inicialização da *stack* do BLE, as interrupções e a inicialização do SPI, método de comunicação utilizado para a comunicação com a *headstage*. Após a inicialização do SPI, o sistema encontra-se pronto para efetuar a conexão com a CySmart USB Dongle. Enquanto não existe conexão, o PSoC fica no modo *advertising*. Ao ser feita a conexão, é chamada a função *características_powerUpSystem()* que programa por SPI os registos da *headstage* necessários aquando da sua ligação. Ao estarem todos os registos programados, é enviado para a interface gráfica a resposta ao último comando enviado pela função *características_powerUpSystem().* Após esse envio, o sistema entra num ciclo infinito a testar se as notificações se encontram ativas.

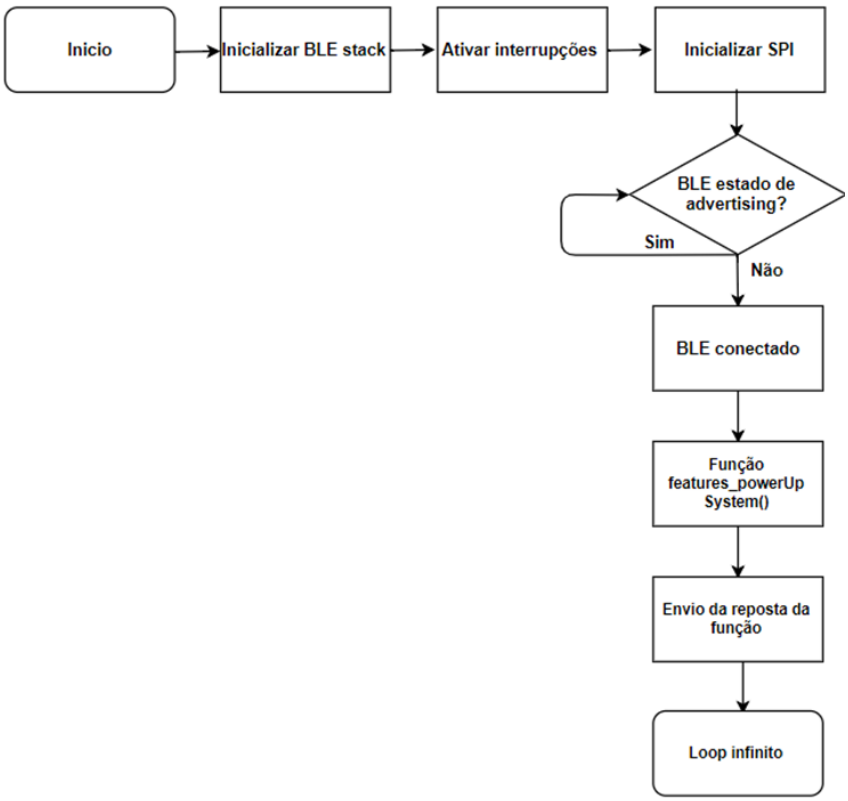


Figura 44-Fluxograma da inicialização do sistema.

Envio de sinais eletrofisiológicos

O fluxograma da figura 45 representa o processo para o envio dos sinais eletrofisiológicos. Após a inicialização do sistema, este encontra-se num ciclo infinito em que é testado o valor da *flag ble_notification* e, sempre que esse valor for 1, a função *características_readADCData()*. Após entrar nessa função vai encontrar um *for loop*, nesse *loop* por cada incremento será testado se o canal referente ao número se encontra ativo. Se estiver ativo será chamada a função *rhs_Convert(uin8)*, função essa que envia o comando de conversão do canal, espera pela resposta e coloca essa resposta num *array* e retorna esse *array* como resposta. Após todos os canais terem sido testados, para verificar os que se encontram ativos, é chamada a função *ble_sendADCData_overNotification()*, que irá enviar o *array* que contém os resultados, por notificação. Este processo repete-se sempre que as notificações se encontraram ativas.

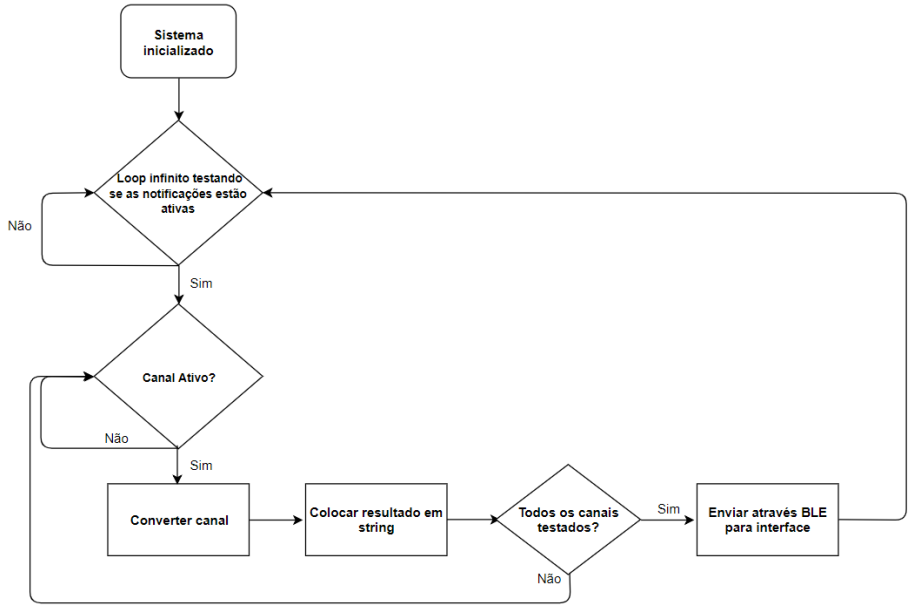


Figura 45-Fluxograma do envio dos sinais eletrofisiológicos.

Envio da resposta de comandos

Por cada envio de um comando para a *headstage*, esse comando vai enviar uma reposta, o comando de conversão. A sua resposta é lida dentro da função *rhs_Convert(uint8)* e enviada através de notificação. Para os outros comandos, a resposta é enviada através do BLE após a receção da resposta. Na figura 46 encontra-se o fluxograma referente ao processo de envio dessa reposta para a interface gráfica. Quando um comando é enviado e a sua reposta é necessária, é chamada a função *rhs_handleRxData()*, função essa que lê do *buffer* SPI a

resposta, coloca essa resposta num *array* e chama a função *ble_sendData(uint8*)* para o envio da resposta para a interface gráfica.

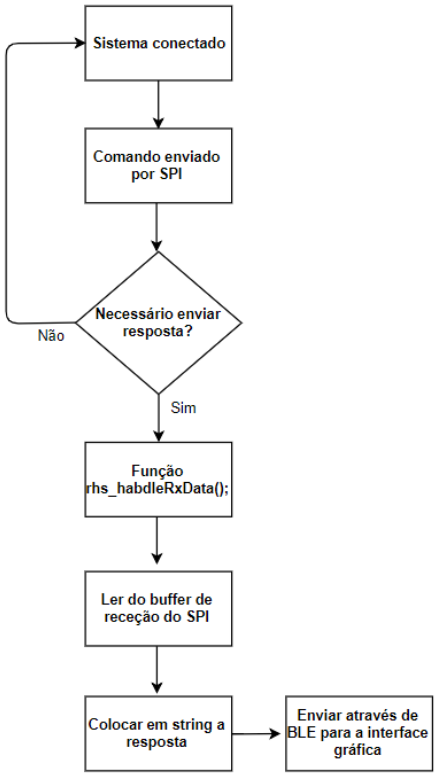


Figura 46-Fluxograma envio da resposta a um comando.

Eventos da stack

Nas figuras 47 a) e b) encontram-se dois fluxogramas referentes aos dois eventos mais importantes da *stack* BLE. Na figura 47 b) encontra-se o evento da *stack* que ocorre após a conexão entre a CySmart USB Dongle e o móudlo PSoC 4 BLE.Na figura 47 a) está descrito o fluxograma mais importante sobre os eventos da *stack*, pois é o evento que trata da receção de informação enviada pela CySmart USB Dongle. Ao entrar nesse evento existem duas possibilidades, ou a informação é escrito na característica das notificações ou é escrita na característica de receção de comandos. Quando a informação é a ativação/desativação das notificações, a *flag ble_notification* é ativa/desativa. Quando se trata da escrita na característica de receção de comandos, trata-se de verificar qual o objetivo do comando enviado e, consoante esse objetivo, chama-se uma das seguintes funções:

- *rhs_getRegData(uint8,uint8)*
- *rhs_setRegData(uint8, uint8,uint8,uint8)*
- *características_powerUpSystem()*
- *características_Bandwidth(uint8,uint8)*

- *características_turnOffChannel(uint8)*
- *características_changeLed()*

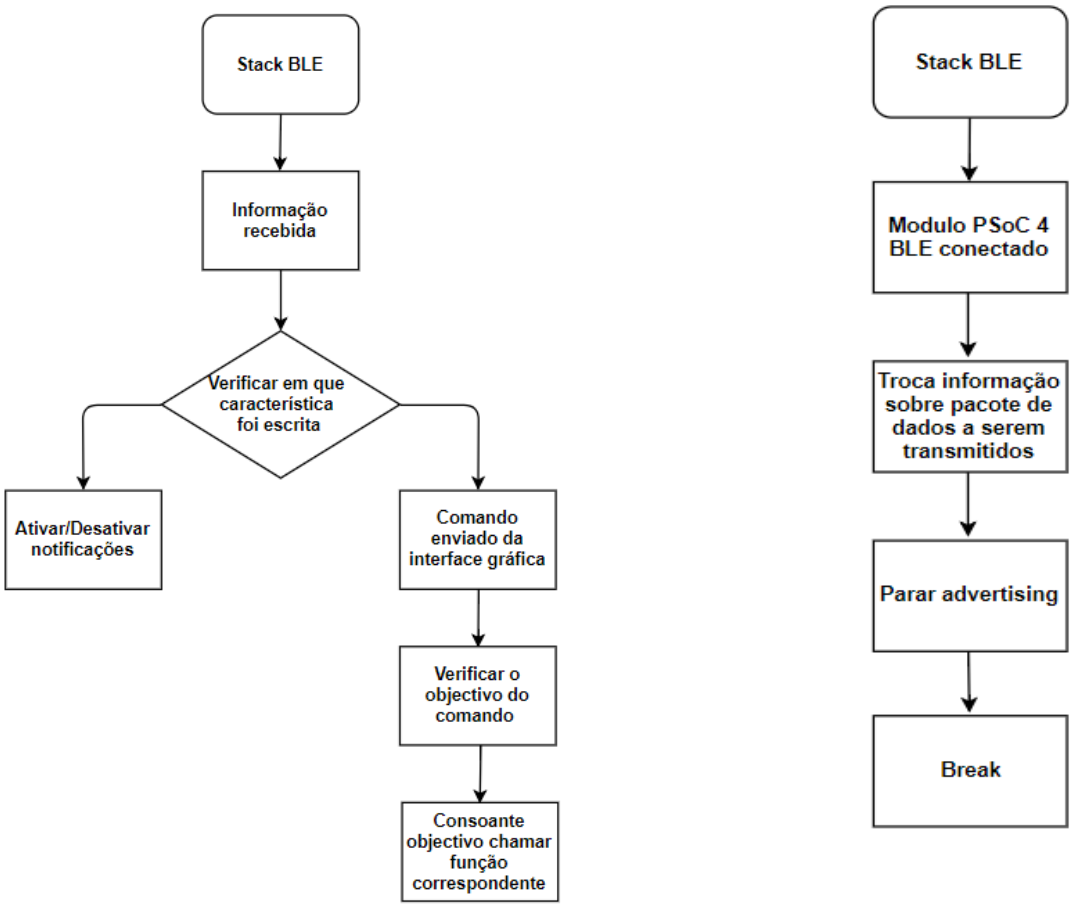


Figura 47-Fluxograma: a) conexão da Dongle ao módulo PSoC 4 BLE; b) informação escrita no serviço Communication.

Fluxograma interface gráfica

Nesta secção serão apresentados, com recurso a fluxogramas, vários esquemas sobre o funcionamento das várias características da interface gráfica.

Inicialização da interface

Na figura 48 a) é apresentado o fluxograma do *software* quando a interface gráfica é inicializada. Primeiramente ocorre a inicialização de cada gráfico, de cada lista para guardar os sinais convertidos e são inicializados os timers necessários e também um conjunto de variáveis.

Conexão aos módulos BLE

Depois de inicializada a interface gráfica, tem de ser efetuada a conexão entre a interface gráfica e os módulos BLE. Como se pode verificar através do fluxograma da figura 48 b), começa-se por verificar as COMs existentes para efetuar a conexão, escolhe-se a COM associada à CySmart USB Dongle e clica-se no botão *First Connect*. Ao estar finalizada a conexão entre a interface gráfica e a Dongle, aparecem no separador *Second connect*, a lista de todos os dispositivos Bluetooth disponíveis para conexão. Escolhe-se o módulo PSoC 4 BLE da lista e ao clicar no botão *Connect*, é efetuada a conexão entre a Dongle e o PSoC 4. Ao clicar no botão *Connect* e após a conexão estar efetuada, é enviado o comando para a entrada na função *características_powerUpSystem()*, que programa os registos aquando da ligação da *headstage*.

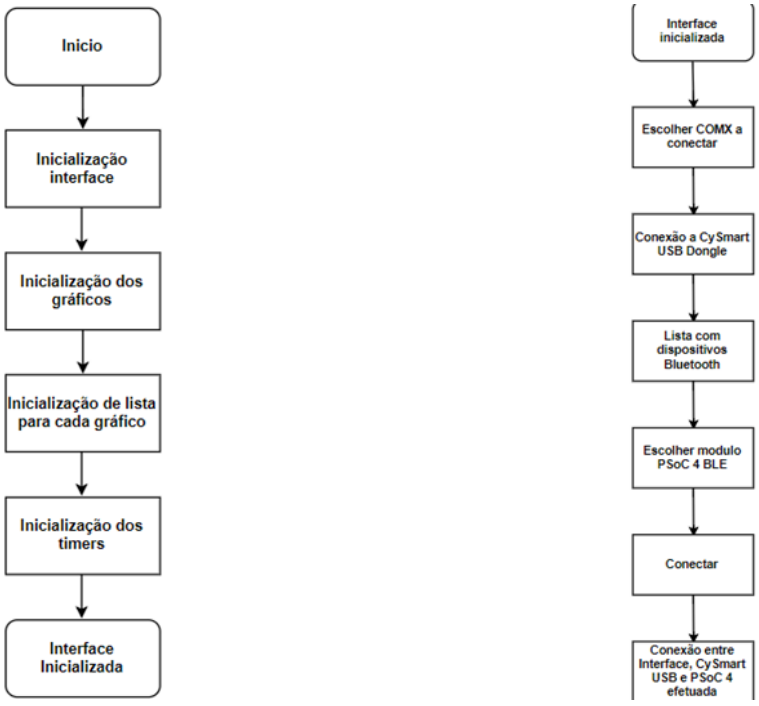


Figura 48-Fluxograma: a) inicialização da interface gráfica; b) conexão entre a Dongle e o BLE na interface gráfica.

Receção dos sinais eletrofisiológicos

Para se iniciar a receção dos sinais eletrofisiológicos vindos da *headstage*, a conexão entre os módulos BLE e a interface gráfica tem de estar concluída. Na figura 49 encontra-se explicado o funcionamento do código para a receção dos sinais. O botão *Run* é o responsável por iniciar essa receção, sendo que, quando clicado, são ativadas as notificações, o *timer 2*, utilizado para o cálculo da taxa de amostragem e do *time unit* e chamada a função *readADCData()*.

Enquanto as notificações estiverem ativas, o código continua num ciclo. Dentro desse ciclo, começa por ser testado se o canal se encontra ativo, se não, é testado o canal seguinte, se sim vai ser colocado num *array* o valor do canal. De seguida esse valor é convertido para mV, adicionado à lista e desenhado no gráfico referente a esse canal. Após ser adicionado no gráfico, se ainda faltarem canais para serem verificados, volta-se a testar o canal seguinte, se já todos os canais tiverem sido testados, adiciona-se no gráfico da estimulação o seu valor, recebido também através das notificações. Quando esse valor for escrito no gráfico referente à estimulação, vai ser testada a *flag Recording* e se estiver ativa, guardam-se os valores num ficheiro definido pelo utilizado e volta-se para o início do ciclo. Se estiver desativa volta a ser testado se as notificações continuam ativas.

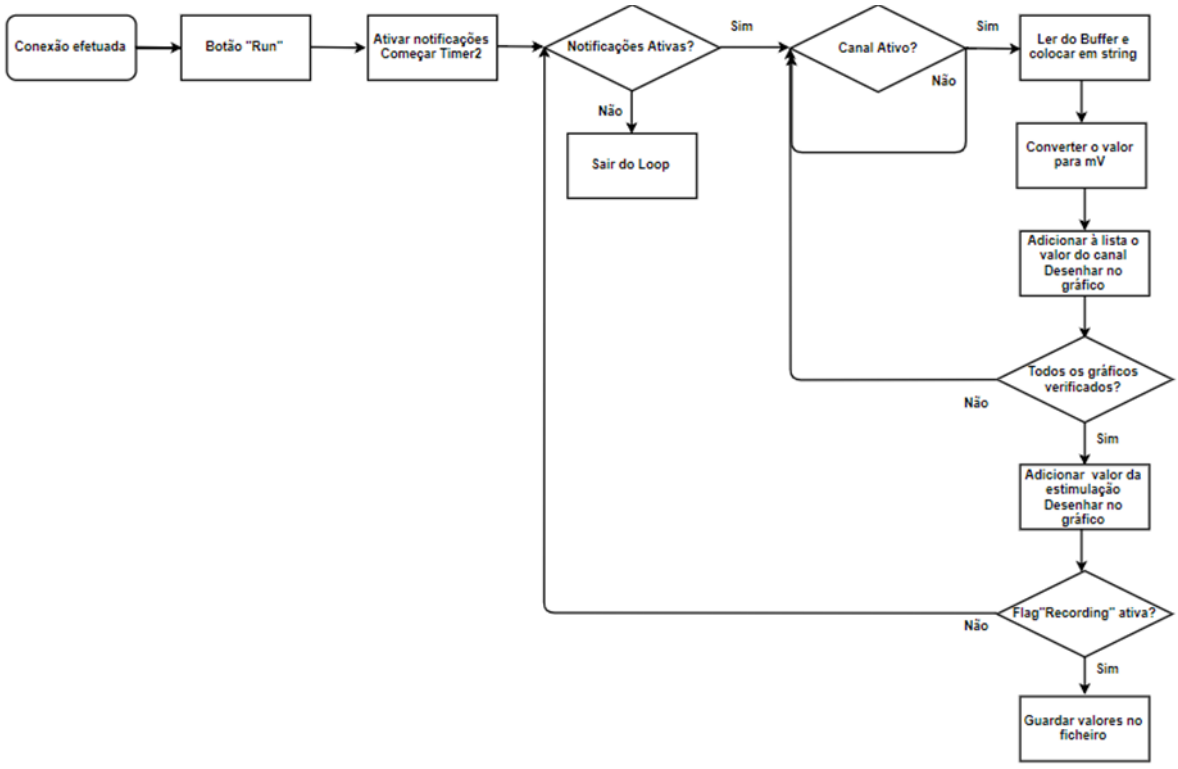


Figura 49-Fluxograma da receção dos sinais eletrofisiológicos enviados pelo BLE.

Guardar em ficheiro

A informação é guardada num ficheiro depois de escolher o local do ficheiro e o nome. Para tal é preciso clicar no botão *Select File Name* e ao ser clicado é aberta uma janela onde o utilizador escolhe o nome do ficheiro e o diretório para guardar esse ficheiro (ver figura 50 a)). O botão *Start Record* encontra-se ativo, assim o utilizador não pode começar a gravar sem antes escolher o ficheiro. Ao clicar no botão *Start Record*, a flag *Recording* é ativada.

Alterar a largura de banda

O fluxograma da figura 50 b) traduz o desenvolvimento do código referente à alteração da largura de banda. A alteração, depois de o sistema estar conectado, faz-se abrindo o separador *Bandwidht* e escolher os novos valores para os filtros. Ao escolher os novos valores, clica-se no botão *Set* e é enviado para o PSoC 4 o comando para a entrada na função *características_setBandwidht(uint8,uint8)* e consequente a alteração da largura de banda de acordo com os valores definidos.

Ativação/desativação de gráficos

Uma das características da interface gráfica é a possibilidade da desativação dos canais, para assim se obter mais taxa de amostragem e ficheiros com menor tamanho. Na figura 50 c) está expresso o fluxograma sobre como ocorre essa desativação ou ativação conforme o estado do gráfico. Para isso, clica-se no gráfico que se pretende ativar/desativar e o botão *Enable/Disable* ativa-se consoante o estado do gráfico. Ao clicar nesse botão, é enviado o comando para o PSoC 4 para entrar na função *características_turnOffChannel()*.

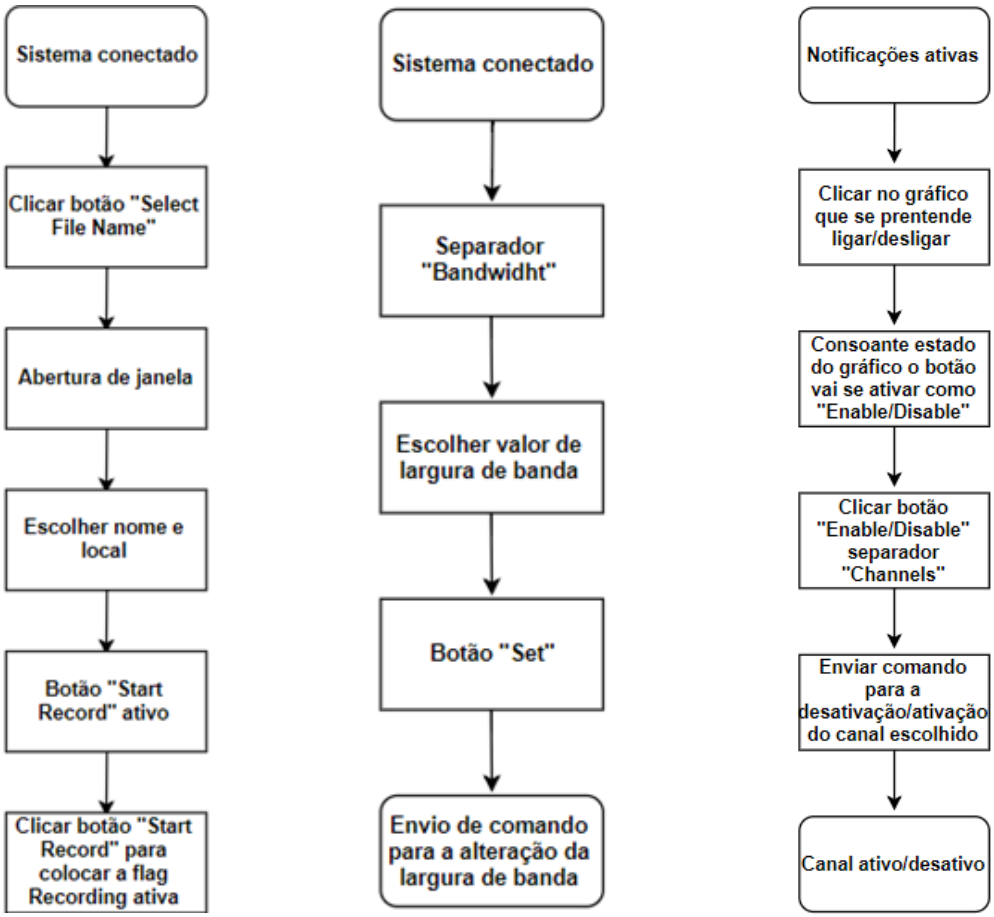


Figura 50-Fluxograma: a) gravação de um ficheiro; b) alteração da largura de banda; c) ativação/desativação de gráficos.

Estimulação

Na figura 51 a) está evidenciado o fluxograma sobre a ativação da estimulação. Para a estimulação ser ativa, primeiramente é necessário escolher o modo de operação, no separador *Set Stimulation*, sendo possível escolher entre o *Mode 0* ou *Mode 1*. Após o utilizador escolher qual o modo, as caixas de entrada desse modo são ativadas e o utilizador pode inserir os valores que pretende. De seguida no separador *Stimulation Run*, onde se encontra o gráfico referente ao valor do LED, é possível ativar a estimulação. Ao clicar no botão *Enable*, por baixo do gráfico, o *timer 1* é ativo. Na função referente ao *timer 1* são enviados comandos para a alteração do valor do LED, sendo que esse tempo são os valores escolhidos pelo utilizador para os tempos a *On* e *Off*. Quando o número de alterações for a correta, o *timer 1* é desativado.

Taxa de amostragem e time unit

Como já foi referido, uma característica da interface gráfica é a capacidade para mostrar a taxa de amostragem e o *time unit*. A taxa de amostragem é o número de valores recebidos por canal num segundo e o *time unit* é o tempo passado entre a receção de cada valor. Para isso é utilizado o *timer 2*, como se pode ver na figura 51 b). Ao serem ativas as notificações, também o *timer 2* é ativado com um intervalo de 2 segundos. Ou seja, de dois em dois segundo, a função do *timer 2* é chamada e é feito o cálculo, consoante o valor da variável *channelReceived*.

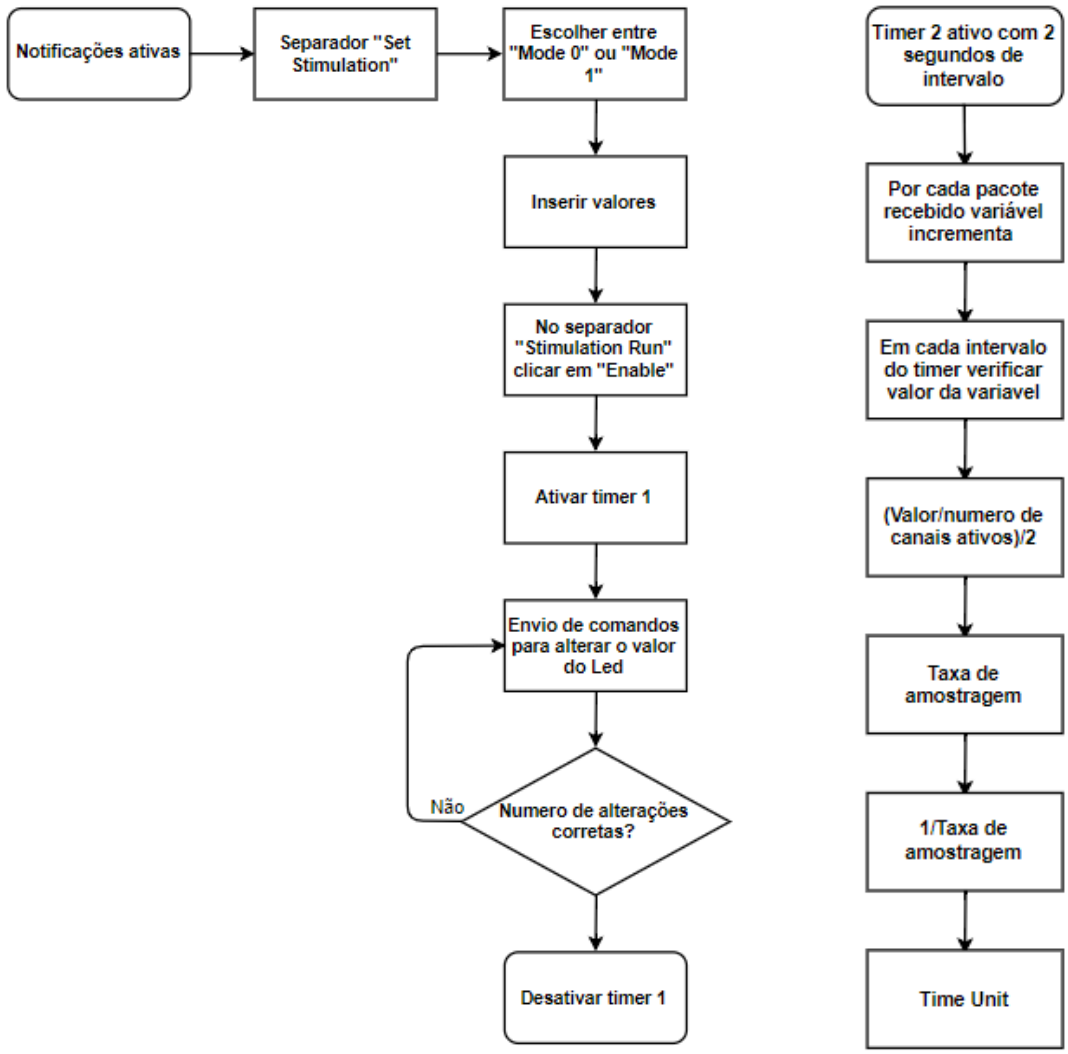


Figura 51-Fluxograma: a) estimulação optogenética; b) cálculo da taxa de amostragem e time unit

FASE DE IMPLEMENTAÇÃO

Neste capítulo será apresentada a fase de implementação das partes mais importantes do código da interface gráfica e do módulo BLE assim como será apresentado e explicado o protocolo de comunicação entre a interface gráfica e o módulo BLE. Protocolo este que foi criado, para que quando a interface gráfica precisar de enviar informação para a *headstage*, o módulo BLE seja capaz de "entender" a informação enviada e consiga acionar as funções corretas para o objetivo definido pelo utilizador.

Protocolo de comunicação

Quando a interface gráfica pretende enviar alguma instrução parao módulo BLE, sem ser a ativação ou desativação das notificações, vai fazê-lo escrevendo no serviço *Interface_Communication*, da característica *Communication.* Como já foi explicado no fluxograma referente aos eventos da *stack* do BLE, ao ser detetada informação escrita neste serviço, o módulo BLE, através do primeiro byte de informação deteta qual o objetivo do comando enviado pela interface gráfica e sabe assim qual a função a ser chamada. A tabela 3 mostra as instruções implementadas no sistema. Se for necessário acrescentar alguma instrução ao código, basta definir um primeiro byte para o módulo BLE perceber o seu objetivo.

Tabela 3-Protocolo de comunicação.

Primeiro byte do comando enviado pela Interface gráfica	**Objetivo**
0x52	*rhs_getRegData(uint8,uint8)*
0x57	*rhs_setRegData(uint8, uint8,uint8,uint8)*
0x60	*características_powerUpSystem()*
0x70	*características_Bandwidth(uint8,uint8)*
0x80	*características_turnOffChannel(uint8)*
0x81	*características_ChangeLed();*

Implementação PSoC 4 BLE

Função main

Na figura 52 está presente a função *main*. Nessa função são realizadas as inicializações e onde o sistema entra num ciclo testando se as notificações se encontram ativas. Antes de tudo, é chamada a função *InitSystem()*, através desta função são inicializadas as interrupções, a *stack* do BLE e o SPI. Após essas inicializações, o PSoC mantém-se num *while loop* enquanto o estado do BLE for um estado de *advertising*. Quando acontece a conexão com a Dongle, prossegue-se para o ciclo infinito onde é testado se a *flag ble_notification*.

```
60  int main(){
61
62      InitSystem();
63      CyDelay(5);
64
65      while(CyBle_GetState()==CYBLE_STATE_ADVERTISING){
66      CyBle_ProcessEvents();
67      }
68
69      for(;;)
70      {
71
72          CyBle_ProcessEvents();
73
74          if(ble_notification==1){
75
76              features_readADCData();
77          }
78      }
79  }
```

Figura 52-Função main().

Envio de sinais eletrofisiológicos

O envio dos sinais eletrofisiológicos para a interface gráfica foi desenvolvido em três funções. A função *características_readADCData()*, a função *rhs_Convert(uint8)* e a função que envia o *array* com os valores, *ble_sendADCData_OverNotification(uint8)*.

Como já foi referido anteriormente, a conversão dos sinais começa apenas quando as notificações forem ativadas. Quando se encontram ativas, a função *características_readADCData()* é chamada num ciclo infinito, até as notificações serem desativadas novamente. Na figura 53 encontra-se essa mesma função. Como se pode verificar, para conversão dos canais é chamada dentro de um *for loop*, *loop* esse que testa todos os 16 canais para verificar se se encontram ativos ou não. Após todos os canais serem testados, é

concatenado o *array* com valor da estimulação, sendo de seguida chamada a função *ble_sendADCData_OverNotification(uint8*)*.

Dentro do *for loop*, o que acontece depois da conversão, é a copia do *array* recebido para o *array* final que contem todos os valores dos canais convertidos, se for o primeiro canal a ser convertido. Após isso, é feita a concatenação do *array* com o valor convertido, com o *array* que contem os valores de todos os canais convertidos.

```
167 void features_readADCData(){ //read from adc
168 
169         firsttime=0;
170     for(uint8 i=0;i<16;i++){
171 
172         if(ActiveChannel[i]==true){
173 
174             *ADC_DATA2 = rhs_Convert(i);
175 
176             if(firsttime==0){
177             strcpy((char*)ADC_FinalResult,(char*)ADC_DATA2);
178             firsttime=1;
179             }
180              else if(firsttime==1) {
181                 strcat((char*)ADC_FinalResult,(char*)ADC_DATA2);
182             }
183             }
184 
185     }
186 
187         firsttime=0;
188 
189    strcat((char*) ADC_FinalResult,(char*) stimulationValue);
190    ble_sendADCData_OverNotification(ADC_FinalResult);
191 
192 }
```

Figura 53-Função características_readADCData().

Para a conversão de um canal ser efetuada, é preciso enviar para a *headstage* o comando referente à conversão, com o valor do canal que pretende ser convertido. Com esse objetivo, foi desenvolvida a função representada na figura 54. Função essa que retorna o valor convertido. O comando que é necessário enviar é armazenado num *array*. De seguida, esse *array* é colocado na API, *SPIM_PutArray(const uint8[], uint8)*, que envia para a *headstage* o valor desse *array*. Como já foi referido anteriormente, só ao terceiro comando enviado é recebido o resultado do primeiro comando. Por isso é repetido mais duas vezes este passo, sendo que após o envio pela terceira vez, é lido através da API, *SPIM_ReadRxData()*, o valor convertido e colocado no *array ADC_RESULT*. Este sistema de enviar três vezes o comando por SPI é repetido para os outros comandos, como a leitura de um registo, a escrita num registo e a calibração do ADC.

```
173 uint8* rhs_Convert(uint8 ADC_Channel){ //co
174 txData[0]=0x00;
175 txData[1]=ADC_Channel;
176 txData[2]=0x00;
177 txData[3]=0x00;
178
179         SPIM_PutArray(txData,4);
180         CyDelayUs(250);
181
182         SPIM_PutArray(txData,4);
183         CyDelayUs(250);
184
185         SPIM_ClearRxBuffer();
186
187         SPIM_PutArray(txData,4);
188         CyDelayUs(250);
189
190      if(SPIM_GetRxBufferSize()!=0){
191
192         for(uint8 i=0;i<2;i++){
193
194         ADC_RESULT[i] = SPIM_ReadRxData();
195
196         }
197     }
198
199     return ADC_RESULT;
200 }
```

Figura 54-Função rhs_Convert(uint8).

Para o envio dos canais convertidos para a interface gráfica através de notificações, foi implementada a função da figura 55. Função essa que envia a *string* com os valores dos canais convertidos. O tamanho do pacote enviado é dependente do número de canais que se encontram ativos. Por cada canal desativado são menos 2 bytes enviados.

```
165 void ble_sendADCData_OverNotification(uint8* ADCData){
166
167 /* Update notification handle with ADC data */
168
169         ADCnotificationHandle.attrHandle = CYBLE_COMMUNICATION_ADC_DATA_CHAR_HANDLE;
170         ADCnotificationHandle.value.val = ADCData;
171         ADCnotificationHandle.value.len =(33 - turnOffChannels);
172
173         /* Send the updated handle as part of attribute for notifications */
174         CyBle_GattsNotification(cyBle_connHandle, &ADCnotificationHandle);
175
176 }
```

Figura 55-Função ble_sendADCData_OverNotification(uint8).*

Eventos da stack

Na *stack* do PSoC existem vários eventos que podem acontecer, por isso a função desenvolvida para os acontecimentos da *stack* contem um *switch case* que dependendo do evento desenvolve o código para determinada função. Em termos de código será apenas

apresentado o código referente quando alguma informação é recebida no serviço da comunicação. Como já foi explicado anteriormente, esse serviço possui duas características, a característica *Interface_Communication* e *ADC_Data*.

Na figura 56 encontra-se a parte do código da função *eventHandler(uint32, void*)* referente ao caso de escrita no serviço *Interface_Communication*. Quando é detetada informação recebida nesse serviço, é colocado num *array* os valores recebidos. Depois de todos os valores serem colocados no *array*, é chamada a função *ble_handleTxData(uint8*).*

```
107         case CYBLE_EVT_GATTS_WRITE_REQ: //Write with response
108         case CYBLE_EVT_GATTS_WRITE_CMD_REQ: //Write without response
109         {
110                 wrReqParam = (CYBLE_GATTS_WRITE_REQ_PARAM_T *) eventParam;
111
112             if(wrReqParam->handleValPair.attrHandle == CYBLE_COMMUNICATION_INTERFACE_COMMUNICATION_CHAR_HANDLE ){
113
114                     while(i<wrReqParam->handleValPair.value.len)
115                   {
116                     data_receivedPC[i] = wrReqParam->handleValPair.value.val[i];
117                     i++;
118                   }
119
120                 ble_handleTxData(data_receivedPC);
121         }
```

Figura 56-Parte da função eventHandler(uint32,void).*

A função *ble_handleTxData(uint8*)*, representada na figura 57, consoante o primeiro byte do *array*, que contem a informação recebida da interface gráfica, determina qual o objetivo do comando enviado pela interface gráfica.

```
177 void ble_handleTxData(uint8 *ble_dataReceivedPC ){    // Receive information from Pc and decide what function enter
178     switch(ble_dataReceivedPC[0]) {
179         case(0x52): //52=R
180
181             rhs_getRegData(ble_dataReceivedPC[1],0x00);
182
183         break;
184
185         case(0x57):   //57=W
186
187             rhs_setRegData(ble_dataReceivedPC[1],ble_dataReceivedPC[2],ble_dataReceivedPC[3],0x00);
188
189         break;
190
191         case(0x60):
192
193             features_powerUpSystem();
194
195         break;
196
197         case(0x70):
198
199             features_Bandwidth(ble_dataReceivedPC[1],ble_dataReceivedPC[2]);
200
201         break;
202         case(0x80):
203
204             features_turnOffChannel(ble_dataReceivedPC[1]);
205
206         break;
207         case(0x81):
208
209             features_ChangeLed();
210
211         break;
212     }
213 }
```

Figura 57-Função ble_handleTxData(uint8).*

A escrita no outro serviço, o *ADC_Data*, serve para ativar ou desativar as notificações, consoante o recebido. O valor da *flag ble_notification* será alterado (ver figura 58).

```
            if(wrReqParam->handleValPair.attrHandle == CYBLE_COMMUNICATION_ADC_DATA_CLIENT_CHARACTERISTIC_CONFIGURATION_DESC_HANDLE)
                {
                    if(0==wrReqParam->handleValPair.value.val [CYBLE_COMMUNICATION_ADC_DATA_CLIENT_CHARACTERISTIC_CONFIGURATION_DESC_INDEX]) {
                    ble_notification=0;
        }
                    if(1==wrReqParam->handleValPair.value.val [CYBLE_COMMUNICATION_ADC_DATA_CLIENT_CHARACTERISTIC_CONFIGURATION_DESC_INDEX]) {
                    ble_notification=0x01;

        }
    }
```

Figura 58-Parte da função eventHandler(uint32,void).*

Função inicialização RHS2116

A função *características_powerUpSystem()* (ver figura 59) como já foi referido anteriormente, é a função que trata de programar os registos com os valores necessários quando se liga o sistema. Todos os registos são programados de acordo com o objetivo do sistema. Esta função é chamada após a interface gráfica se conectar com o módulo BLE. Após todos os registos serem programados, é chamada a função *rhs_handleRxData()* para enviar por BLE para a interface gráfica a resposta ao último canal. Se o valor for o correto, o sistema encontra-se conectado.

```
void features_powerUpSystem(){
    rhs_getRegData(0xFF,0x00); //read from register 255
    rhs_setRegData(32,0x00,0x00,0x00);    //stimulation disabled
    rhs_setRegData(33,0x00,0x00,0x00);
    rhs_setRegData(38,0xFF,0XFF,0x00); //POWER UP ALL DC low gain amplifiers /nedded due to hardware
    rhs_Clear();
    rhs_setRegData(0,0x08,0x28,0x00);    // frequncy samples<120kS/s ADC BUFFER 32 MUX BIAS-40
    rhs_setRegData(1,0x05,0x12,0x00);//high impdance and enable dsp filter to 1/0X11
    rhs_setRegData(2,0x00,0x40,0x00); //disable impedance check
    rhs_setRegData(3,0x00,0x80,0x00);
   // set BANDWIDTH
    rhs_setRegData(0x04,0x00,0x16,0x00);//set Fh to 7.5khz
    rhs_setRegData(0x05,0x00,0x17,0x00);
    rhs_setRegData(0x06,0x00,0xA8,0x00);//SET FL to 5 Hz
    rhs_setRegData(0x07,0x00,0x0F,0x00); //fl B
    rhs_setRegData(0x08,0xFF,0xFF,0x00);//enable all channels
    rhs_setRegData(10,0x00,0x00,0x01);//disable fast settle
    rhs_setRegData(12,0xFF,0xFF,0x01);//set FL to register 6
   //stimulation
    rhs_setRegData(34,0x00,0xE2,0x00); //stimulation step size 1uA
    rhs_setRegData(35,0x00,0xAA,0x00);//stimulation bias voltage for 1uA
    rhs_setRegData(36,0x00,0x80,0x00);//current limit charge to 0
    rhs_setRegData(37,0x4F,0x00,0x00);//charge recovery current limit to 1nA
    rhs_setRegData(42,0x00,0x00,0x01);//SET all stimulators off
    rhs_setRegData(44,0x00,0x00,0x01);//stimulators to negative polarity
    rhs_setRegData(46,0x00,0x00,0x01);//open all charge recovery switches
    rhs_setRegData(48,0x00,0x00,0x01);//disable all current limit charge recovery circuits
//  //  set negative stimulation current magnitude to 0
    rhs_setRegData(64,0x80,0x00,0x01);
    rhs_setRegData(65,0x80,0x00,0x01);
    rhs_setRegData(66,0x80,0x00,0x01);
    rhs_setRegData(67,0x80,0x00,0x01);
    rhs_setRegData(68,0x80,0x00,0x01);
    rhs_setRegData(69,0x80,0x00,0x01);
    rhs_setRegData(70,0x80,0x00,0x01);
    rhs_setRegData(71,0x80,0x00,0x01);
    rhs_setRegData(72,0x80,0x00,0x01);
    rhs_setRegData(73,0x80,0x00,0x01);
    rhs_setRegData(74,0x80,0x00,0x01);
    rhs_setRegData(75,0x80,0x00,0x01);
    rhs_setRegData(76,0x80,0x00,0x01);
    rhs_setRegData(77,0x80,0x00,0x01);
    rhs_setRegData(78,0x80,0x00,0x01);
    rhs_setRegData(79,0x80,0x00,0x01);
    //set positive stimulation current magnitude to 0
    rhs_setRegData(96,0x80,0x00,0x01);
    rhs_setRegData(97,0x80,0x00,0x01);
    rhs_setRegData(98,0x80,0x00,0x01);
    rhs_setRegData(99,0x80,0x00,0x01);
    rhs_setRegData(100,0x80,0x00,0x01);
    rhs_setRegData(101,0x80,0x00,0x01);
    rhs_setRegData(102,0x80,0x00,0x01);
    rhs_setRegData(103,0x80,0x00,0x01);
    rhs_setRegData(104,0x80,0x00,0x01);
    rhs_setRegData(105,0x80,0x00,0x01);
    rhs_setRegData(106,0x80,0x00,0x01);
    rhs_setRegData(107,0x80,0x00,0x01);
    rhs_setRegData(108,0x80,0x00,0x01);
    rhs_setRegData(109,0x80,0x00,0x01);
    rhs_setRegData(110,0x80,0x00,0x01);
    rhs_setRegData(111,0x80,0x00,0x01);

    rhs_getRegData(0xFF,0x01);//READ and set M flag

    rhs_handleRxData();
}
```

Figura 59-a) e b) Partes da função características_powerUpSystem().

Implementação da interface gráfica

Receção sinais eletrofisiológicos

A figura 60 mostra o algoritmo desenvolvido para a receção dos sinais eletrofisiológicos recebidos através de notificação, enviados pelo PSoC 4 BLE. Algoritmo esse já explicado anteriormente, na seção dos fluxogramas. Para além do algoritmo já explicado, existem algumas condições que não foram incluídas na explicação do algoritmo principal. A primeira é a condição da linha 468, esta condição *if*, só permite a entrada no *for loop* para o teste da ativação dos canais, se o número de canais ativos for o mesmo que o número de dados enviados pelo módulo BLE. A segunda condição *if*, na linha 489, é para quando um canal estiver desativado, colocar 0 na lista e adicionar ao gráfico, para assim o gráfico quando for ativo novamente não se encontra "atrás" dos outros.

```
442         private void receiveADCData()
443         {
444             AutoResetEvent sync = new AutoResetEvent(false);
445             CyApiErr err = CyApiErr.OK;
446 
447             GATTClientEventCallback.DescriptorWriteHandler = (CyConnectResult, status) =>
448             {
449                 if (status != CyStatus.BLE_STATUS_OK)
450                     err = new CyApiErr("Failed to start/stop notification monitoring. Reason: " + status.ToString());
451 
452                 sync.Set();
453             };
454 
455             if (!NotificationEnabled)
456             {
457                 // Initiate write descriptor request to the CCCD
458                 err = GattClient.WriteDescriptor(new CyGattWriteInfo(0x0017, BitConverter.GetBytes(0x0100)));
459                 timer2.Start();
460                 enableStimulation.Enabled = true;
461 
462                 if (err.IsOK)
463                 {
464                     GATTClientEventCallback.CharacteristicChangedHandler = (info) =>     //receive data from notifications!!!!
465                     {
466                         Invoke((MethodInvoker)(() =>
467                         {
468                             if (Channel_number * 2 == info.Value.Length)
469                             {
470                                 pointCounter++;
471                                 for (int i = 0; i < 16; i++)
472                                 {
473                                     if (Canais_ativos[i] == true)
474                                     {
475                                         channelReceived++;
476                                         for (int j = 0; j < 2; j++)
477                                         {
478                                             adcValues[x] = info.Value[x];
479                                             x++;
480                                         }
```

```
482                                         adcValuesNew[i] = adcValues[x - 2] << 8 | (adcValues[x - 1] & 0xFF);
483                                         adcRealValue[i] = 0.000195 * (adcValuesNew[i] - 32768);
484 
485                                         GraphicValues[i].Add(adcRealValue[i]);
486                                         Channels[i].DrawGraph(GraphicValues[i], null, false);
487 
488                                     }
489                                     else if (Canais_ativos[i] == false) GraphicValues[i].Add(0);
490                                 }
491 
492                                 stimulationListValues.Add(info.Value[x]);
493                                 stimulationGraphic.DrawGraph(stimulationListValues, null, false);
494                             }
495                             x = 0;
496                         }));
497 
498                         if (Recording == true) {
499                             saveToFile();
500                         }
501                     };
502                 }
```

Figura 60-a) e b) Partes da função receiveADCData().

Estimulação

O algoritmo para a ativação da estimulação optogenética foi divido em dois. No modo 0 e modo 1. O modo 0 é o mais simples, o utilizador apenas pode configurar um período e quando vezes esse período se repete. O modo 1 é mais complexo, pois já permite ao utilizador configurar até 5 períodos, e quantas vezes esses períodos se podem repetir. Seguidamente serão explicados os algoritmos na implementação desses dois modos.

Após o modo 0 ser ativo, através do botão *Mode 0*, o timer 1 é ativo com um intervalo de 1 ms, o que leva à sua ativação imediata. Ao entrar na função referente à interrupção do *timer 1*, e estando o modo 0 ativo, é testada a condição da linha 71 do código representado na figura 61, que compara o tamanho de duas variáveis, a variável *stimulationTimes* e o valor introduzido pelo utilizador para o número de repetições do período. Enquanto a variável *stimulationTimes* for menor, é testado através da condição *if* se a variável *stimualtionOn* é 1 ou 0. Se for 1, vai ser enviado o comando para a alteração do valor do LED, e colocado como valor para a interrupção do timer 1, o tempo a *Off*. Se for 0 é igualmente enviado o comando para a alteração do LED, mas o valor da interrupção do *timer* 1 vai ser o valor do tempo a *On*.

Quando a variável *stimulationTimes* for maior que o valor introduzido pelo utilizador para o número de repetições do período, significa que já foram cumpridas as ações definidas pelo utilizador e a variável *stimulationTimes* volta a 0 para ser utilizada uma próxima vez que for ativo o modo 0.

```
69              if (mode0 == 1)
70              {
71                  if (stimulationTimes < (byte.Parse(cycleNumber.Text) * 2))
72                  {
73                      if (stimulationOn == 1)
74                      {
75
76                          err = GattClient.WriteCharacteristic(new CyGattWriteInfo(0x12, 0x81));
77                          timer1.Interval = Convert.ToInt32(offTime.Text);
78                          stimulationOn = 0;
79                          stimulationTimes++;
80                          timer1.Start();
81                      }
82
83                      else
84                      {
85
86                          err = GattClient.WriteCharacteristic(new CyGattWriteInfo(0x12, 0x81));
87                          timer1.Interval = Convert.ToInt32(onTime.Text);
88                          stimulationOn = 1;
89                          stimulationTimes++;
90                          timer1.Start();
91                      }
92                  }
93                  else
94                  {
95                      stimulationTimes = 0;
96                  }
97              }
```

Figura 61-Parte da função TimerEventProcessor(object, EventArgs).

O algoritmo para o modo 1 é mais complexo que o modo 0 pois permite mais funcionalidades. Ao se encontrar o modo 1 ativo, o código da interrupção do timer 1 entra nesta parte da função. A primeira condição testada é na linha 102 do código representado na figura 62, onde são comparadas duas variáveis. A variável *counter*, que é incrementada consoante o número de vezes que o LED é alterado e a variável *stimulationTimes2*, o valor desta variável é o período multiplicado por dois, e o resultado desse valor multiplicado pelo número de ciclos definidos pelo utilizador. O valor do período depende do número de tempos a *On* e *Off* configurados pelo utilizador. Esta condição serve para verificar se o número total de alterações já foi cumprido. A condição seguinte compara a variável *stimulationTimes* com o número de períodos multiplicado por dois. Condição esta que verifica se já foi percorrido um ciclo de estimulação. Se ainda não foi, o código irá passar para um *switch case* dependendo do número de estimulações já ocorridas. Quando um ciclo é ultrapassado, a variável *stimulationTimes* passa a 0 e é repetido novamente até o número de ciclos estar completo.

```
99              if (mode1 == 1)
100             {
101
102                 if (counter < stimulationTimes2)
103                 {
104
105                     if (stimulationTimes == (periodNumber * 2))
106                     {
107
108                         stimulationTimes = 0;
109
110                     }
111
112                     switch (stimulationTimes)
113                     {
114                         case (0):
115
116                             err = GattClient.WriteCharacteristic(new CyGattWriteInfo(0x12, 0x81));
117                             timer1.Interval = Convert.ToInt32(onTime1.Text);
118
119                             stimulationTimes++;
120                             counter++;
121                             timer1.Start();
122
123                             break;
124
125                         case (1):
126
127                             err = GattClient.WriteCharacteristic(new CyGattWriteInfo(0x12, 0x81));
128                             timer1.Interval = Convert.ToInt32(offTime1.Text);
129
130                             stimulationTimes++;
131                             counter++;
132                             timer1.Start();
133
134                             break;
```

Figura 62-Parte da função TimerEventProcessor(object, EventArgs).

Ativar/desativar gráfico

Na figura 63, encontra-se uma parte do código desenvolvido para a ativação/desativação de um gráfico, neste caso do gráfico 1. Esta parte do código faz parte da função *disableChannel_Click(object, EventArgs)*. Ao entrar nesta função, significa que o botão no separador *Channels* foi clicado. Nesta função é testado se a variável *Enable[X]*, sendo X o número do gráfico, é ativa ou não. Variável esta que representa se o gráfico X foi selecionado para ser ativado/desativado. Se o gráfico foi selecionado, a variável *Enable[X]* encontra-se no estado *true* e a condição é por isso verdadeira. Se for verdadeira, é enviado para o módulo BLE o comando para a entrada na função *características_turnOffChannel(uint8)*, que irá cancelar ou ativar a conversão do canal consoante o estado anterior. De seguida é testado se o gráfico está ativo ou não na interface interface, se estiver ativo, vai se desativar o gráfico, e subtrair 1 à variável *Channel_number*. Se o gráfico estiver desativo, irá se proceder à sua ativação e incrementado em 1 o valor da variável *Channel_number*. Este processo repete-se para os outros gráficos de igual modo.

```
513             if (Enable[0] == true)
514             {
515                 err = GattClient.WriteCharacteristic(new CyGattWriteInfo(0x12, 0x80, 0x01));
516                 if (err.IsOK)
517                 {
518                     if (activeChannel[0] == true)
519                     {
520                         activeChannel[0] = false;
521                         Channel_number = Channel_number - 1;
522 
523                         enable2.Text = "Enable Channel";
524                         enable2.Enabled = false;
525                         Enable[0] = false;
526 
527                         zedGraphControl1.GraphPane.CurveList.Clear();
528                         zedGraphControl1.AxisChange();
529                         zedGraphControl1.Invalidate();
530                     }
531                     else
532                     {
533 
534                         Channel_number = Channel_number + 1;
535                         activeChannel[0] = true;
536                         enable2.Text = "Disable Channel";
537                         Enable[0] = false;
538 
539                     }
540                 }
541             }
```

Figura 63-Parte da função disableChannel_Click(object, EventArgs).

TESTES E RESULTADOS

Neste capítulo serão demonstrados e analisados os resultados obtidos no sistema desenvolvido nesta dissertação. Ao longo do desenvolvimento do sistema foram realizados vários testes intermédios às várias características e ao funcionamento do sistema. Neste capítulo serão apresentados os resultados referentes a um teste final realizado, mostrando o funcionamento das funcionalidades implementadas.

Equipamento utilizado para o teste

Para a demonstração dos resultados alcançados no desenvolvimento desta dissertação foi necessário recorrer a um gerador de sinais para a simulação de um sinal eletrofisiológico. Através de uma ponta de prova ligamos o sinal do gerador de sinais a um divisor de tensão, para assim alcançar sinais na gama dos µV (gama dos sinais eletrofisiológicos). Esse sinal é depois ligado aos vários canais da *headstage*, simulado assim os sinais provenientes de uma sonda neuronal.

Foram realizados testes com vários sinais de diferentes frequências para assim verificar as diferentes respostas da aplicação consoante a variação da frequência do sinal de entrada. Os testes foram realizados com sinais de frequências entre os 2.5 Hz e os 10 Hz devido à baixa taxa de amostragem que o sistema possui (41 Samples/s). A frequência de amostragem deve ser pelo menos duas vezes a frequência máxima do sinal original para não ocorrer *aliasing* (teorema de *Nysquit*) [54].

A alimentação do sistema, como se pode verificar pela figura 64, é realizada através de um Digital Lab. O Digital Lab alimenta o módulo PSoC 4 BLE com 3.6 V e a LVDS Adapter board com ± 3.6 V (garante a conversão da comunicação SPI entre o nível LVDS e CMOS).

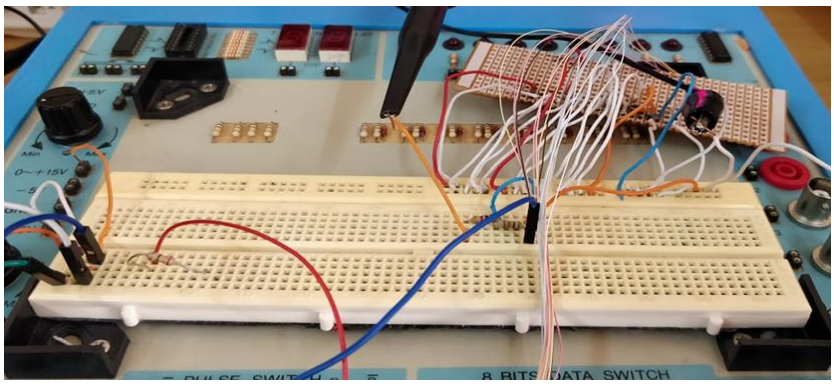


Figura 64-Equipamente utilizado para testes do sistema.

Sistema

Na figura 65 pode-se observar todo o sistema que trata da conversão dos sinais eletrofisiológicos e do seu envio. Observa-se as três placas conectadas entre si, a *headstage* conectada através de um cabo *Stim SPI* com a placa *LVDS Adapter board* que consequentemente se encontra ligada ao módulo PSoC 4 BLE. Os pinos do módulo PSoC 4 BLE ligados à LVDS Adapter board encontram-se descritos na tabela 4.

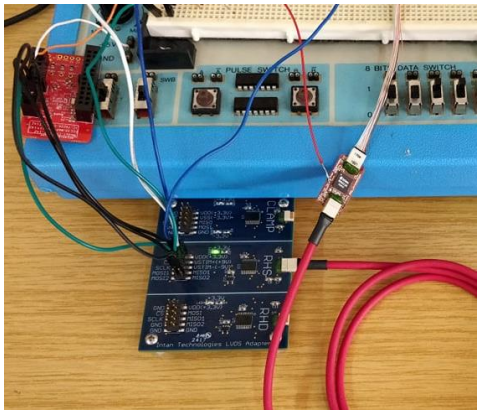


Figura 65-Sistema conectado.

Tabela 4-Pinos de ligação SPI.

Pinos SPI	Pinos PSoC 4 BLE
CS	P0.2
SCLK	P0.3
MOSI	P0.0
MISO	P0.1

Resultados

Seguidamente serão apresentados os resultados obtidos com a configuração apresentada acima. Será apresentada a interface gráfica, o funcionamento das várias características, bem como a taxa de amostragem conseguida e o consumo de energia estimado.

Sinais eletrofisiológicos

Os níveis de tensão dos sinais eletrofisiológicos foram conseguidos através de um gerador de sinais em conjunto com um divisor de tensão. Como primeiro resultado será apresentado a resposta da aplicação a um sinal sinusoidal de 2.5 Hz, menor valor conseguido pelo gerador de sinais. Na figura 66 temos a aplicação a receber todos os 16 sinais mostrando

o sinal sinusoidal enviado. Para a validação da onda recebida, fazendo os cálculos do período da onda apresentada na figura 66, verifica-se que um período demora aproximadamente 16 unidades sendo que cada unidade equivale a 24.4 ms, ou seja, um período demora aproximadamente 390.4 ms, o que que dá 2.5 Hz de frequência, frequência da onda de entrada. A amplitude é 1mV de amplitude de pico. Em alguns canais a escala alterou-se devido à momentânea receção de picos de tensão mais elevado.

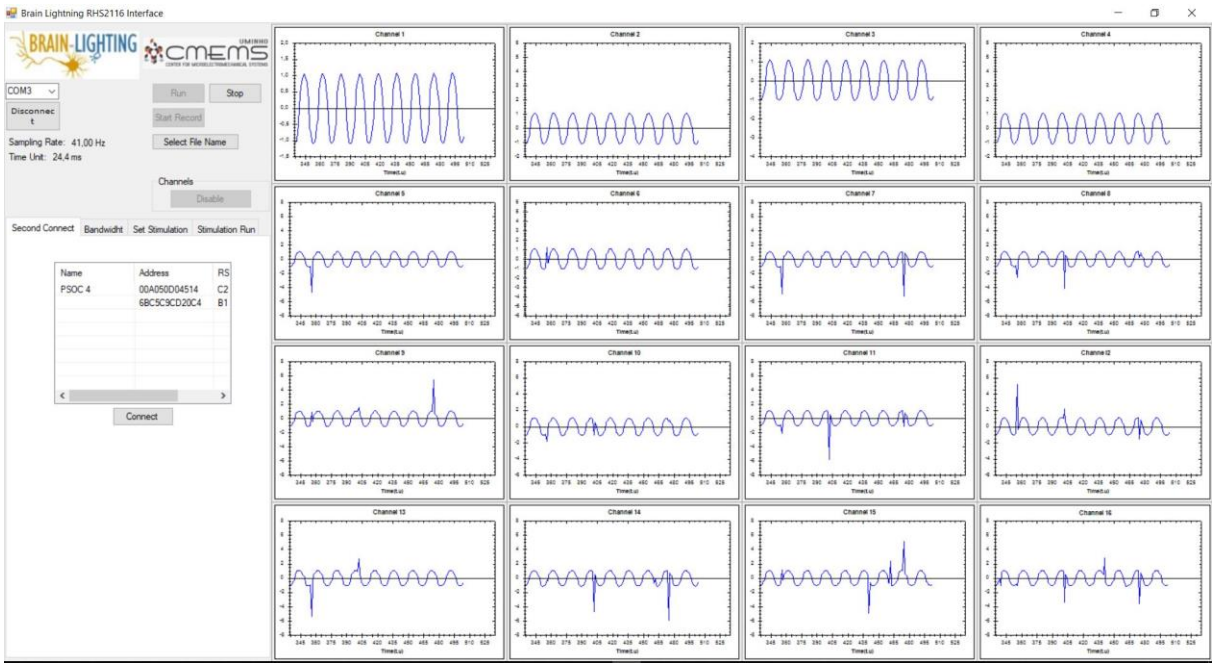


Figura 66-Sinal sinusoidal de 2.5 Hz com 16 canais ativos.

A taxa de amostragem alcançada foi de 41 *Samples/s* por canal, sendo que à medida que se desligam canais, a taxa de amostragem aumenta, como se pode verificar na figura 67.

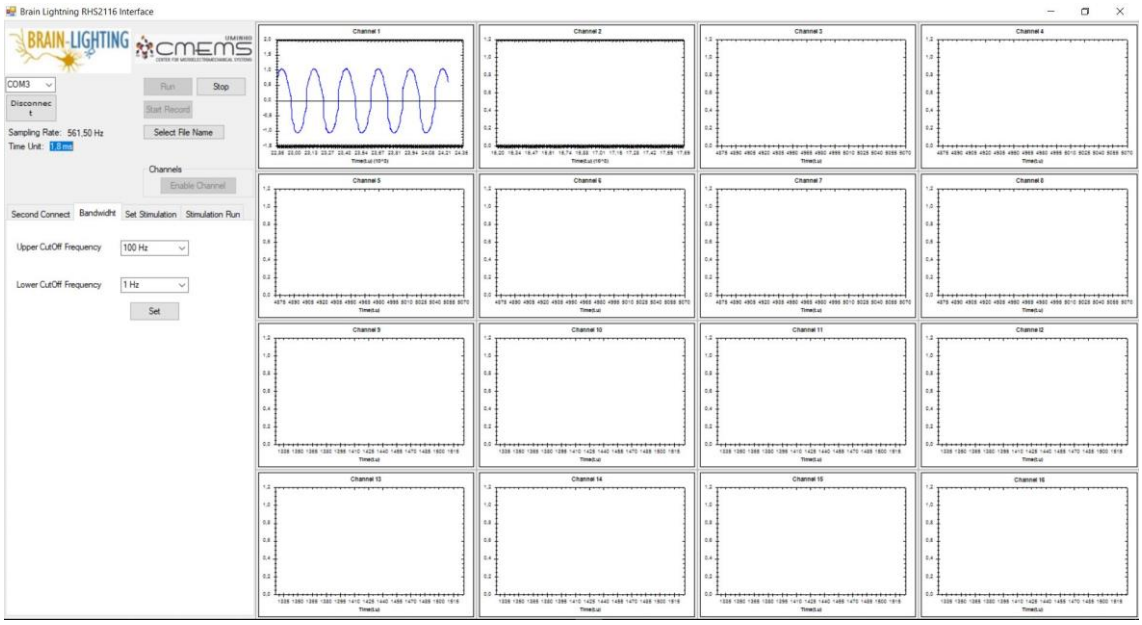


Figura 67-Sinal sinusoidal de 2.5 Hz com 1 canal ativo.

Com apenas um canal verifica-se uma taxa de amostragem de 561 *Samples/s* por canal, taxa essa que aumenta com cada canal que é desligado pois é menos um canal que tem de ser convertido e menos dois bytes que são enviados. Com maior taxa de amostragem, recebe-se mais pontos da onda e esta apresenta-se com maior resolução. Devido à maior velocidade com apenas um canal, a escala do eixo dos XX do gráfico ajusta-se automaticamente para melhor visualização do utilizador. De seguida serão apresentados os resultados quando utilizado um sinal sinusoidal de 4.1 Hz e de 5.1 Hz (figura 68 a) e b)).

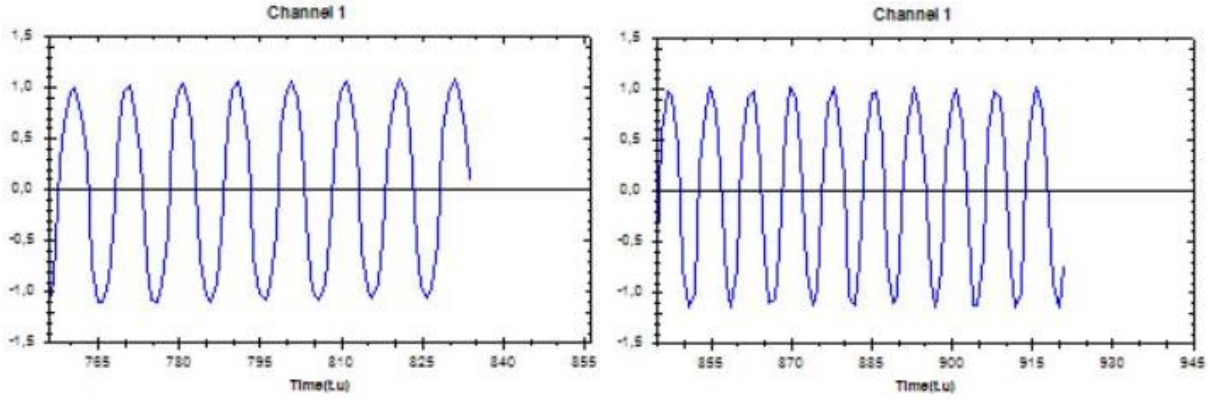


Figura 68-Sinal sinusoidal com 16 canais ativos: a) 4.1 Hz; b) 5.1 Hz.

Através da figura 68 a) e b) verifica-se diferenças nas ondas apresentadas, sendo que é apresentado um melhor resultado com a onda sinusoidal de 4.1 Hz (frequência dez vezes inferior à taxa de amostragem), pois a onda apresentada é de melhor qualidade em relação à onda de 5.1 Hz (frequência oito vezes inferior à taxa de amostragem). A onda sinusoidal de 5.1 Hz (figura 68 b)) não tendo pontos suficientes para construir a onda. Assim o nosso sistema garante o seu melhor funcionamento com frequências menores ou iguais a 4.1 Hz. As figuras 69 a) e b), apresentam as diferenças quando são desligados gráficos e aumentada a taxa de amostragem.

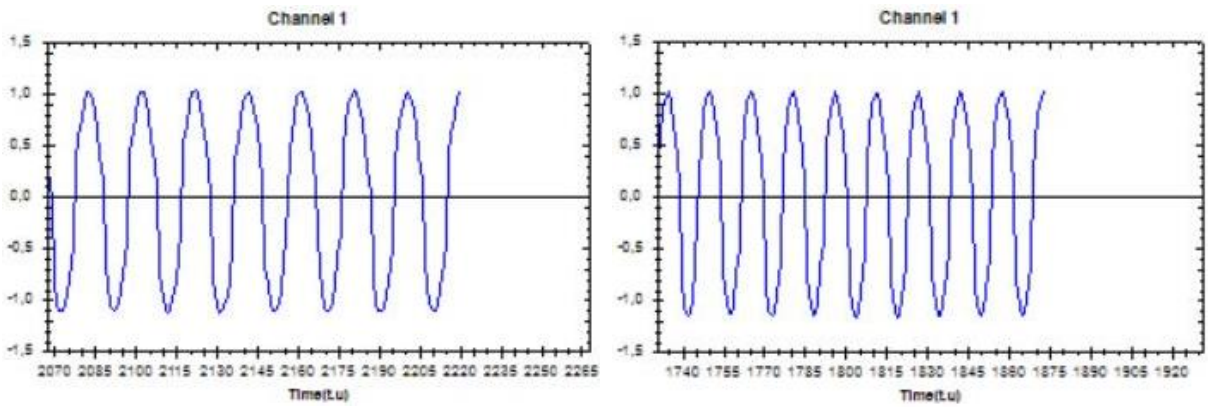


Figura 69-Onda sinusoidal com 8 canais ativos: a) 4.1 Hz; b) 5.1 Hz.

A figura 69 a) é bastante semelhante ao resultado apresentado na figura 68 a), sendo que na figura 69 b) notam-se melhorias em relação à onda apresentada na figura 68 b). Não se verifica os "cortes" das ondas no pico, pois já são recebidos mais pontos da onda.

Através dos resultados apresentados em cima, conclui-se que a frequência do sinal deve ser pelo menos dez vezes inferior à frequência de amostragem. Sendo que o nosso sistema garante o melhor funcionamento com ondas iguais ou inferiores à frequência de 4.1 Hz. Contudo, à medida que temos maior taxa de amostragem, devido ao desativar dos gráficos, conseguimos apresentar uma maior frequência do sinal de entrada. Na tabela 5 é apresentada a frequência máxima aconselhada para o correto funcionamento do sistema com diferente número de canais ativos.

Tabela 5-Frequencia máxima aconselhada com diferente número de canais ativos.

Número de canais ativos	Frequência de amostragem do sistema (Hz)	Frequência máxima do sinal a transmitir (Hz) (dez vezes menor que a frequência de amostragem)
1	550	55
2	300	30
3	205	20.5
4	155	15.5
5	125.5	12.5
6	105.5	10.5
7	91	9.1
8	79.5	7.9
9	71	7.1
10	63	6.3
11	57	5.7
12	53	5.3
13	49.5	4.9
14	45.5	4.5
15	43	4.3
16	41	4.1

Alteração da largura de banda

Na figura 70 verifica-se o sinal recebido após a alteração da largura de banda. Ao ser alterada a largura de banda entre valores superiores ao da frequência do sinal, o sinal convertido passa para zero, como se pode verificar.

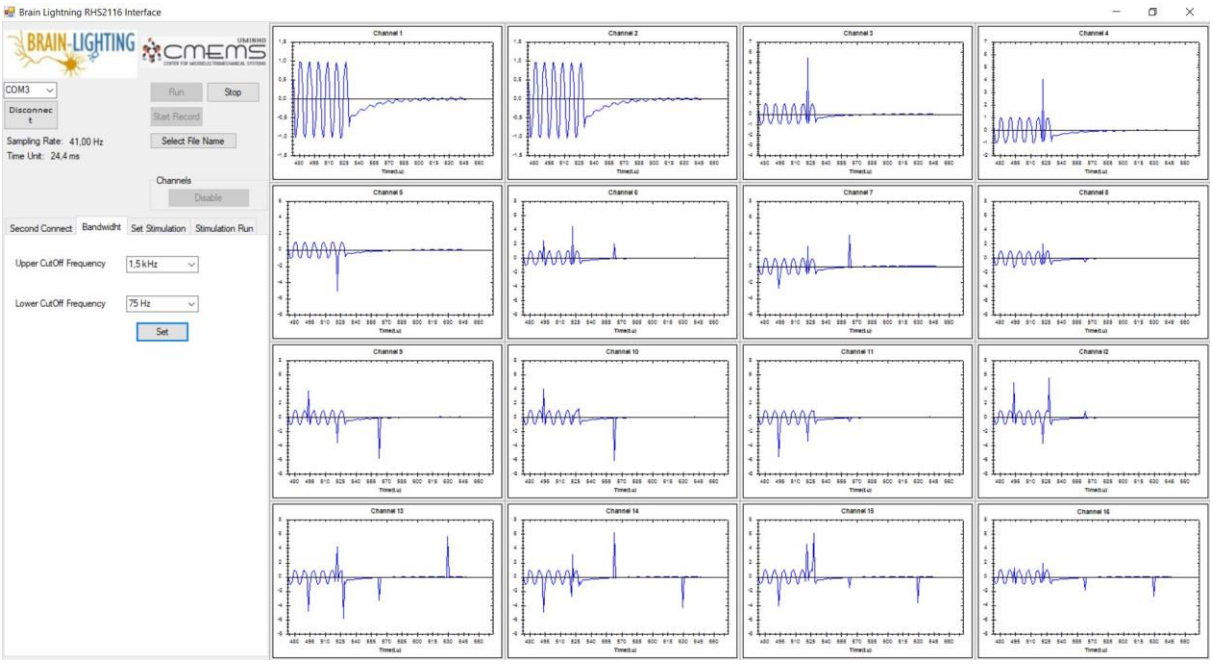


Figura 70-Sinal após alteração da largura de banda.

Estimulação

Para a simulação da estimulação optogenética, foi ligada à *headstage* da Intan a sonda neuronal do projeto *BrainLighting* com o LED referenciado anteriormente (ver figura 71 a)). Sendo a estimulação configurada no modo 0 com 1 segundo de tempo On e 1 segundo de tempo Off, o resultado encontra-se na figura 71 b). O LED está ligado e no gráfico referente à estimulação verifica-se que já foi ligado uma vez e se encontra ligado outra vez, pois o número de ciclos configurado foi de 2.

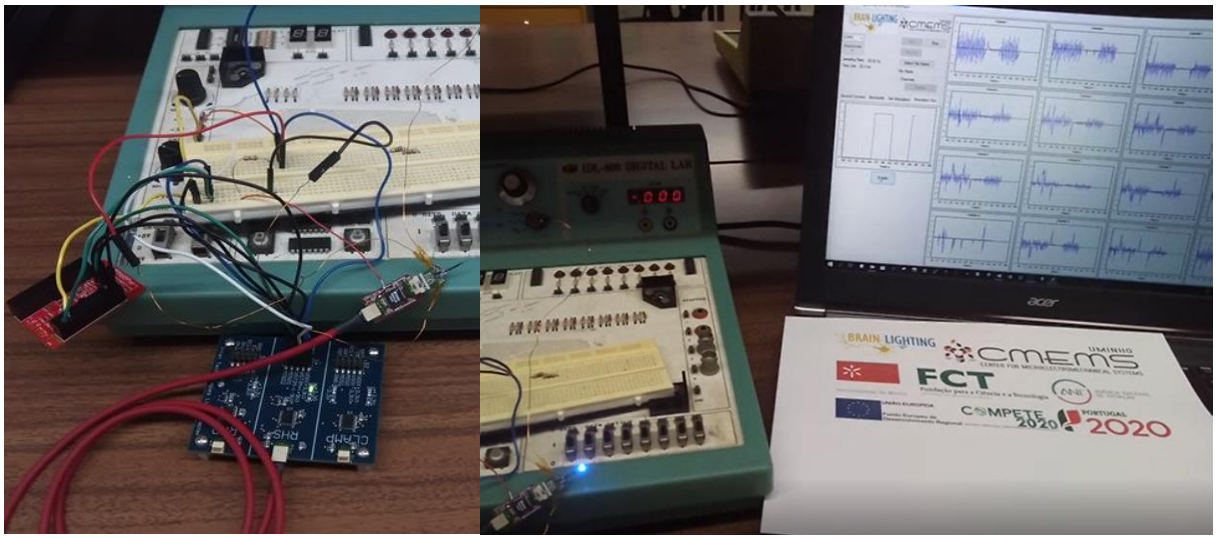


Figura 71-Sistema utilizado para a simulação da estimulação optogenética: a) LED desligado; b) LED ligado.

Guardar ficheiro

Na figura 72 pode-se verificar um exemplo de um ficheiro gravado com os dados da interface gráfica. O ficheiro grava os valores da taxa de amostragem, do *time unit*, o valor dos canais ativos, dos filtros da largura de banda e por fim, apesar de não ser percetível na figura 72, o valor da estimulação.

TESTE.txt - Bloco de notas

Ficheiro Editar Formatar Ver Ajuda

SamplingRate	TimeUnit	Time	UpperCutOff	LowerCutOff	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
41,50 Hz	23,0 ms	22.80 ms	100 Hz	1 Hz	0,29 mV	0,32 mV	0,33 mV	0,34 mV	0,35 mV	0,28 mV	0,39 mV
41,50 Hz	23,0 ms	45,98 ms	100 Hz	1 Hz	1,00 mV	1,03 mV	1,01 mV	1,03 mV	1,05 mV	1.05 mV	1,05 mV
41,50 Hz	23,0 ms	68,97 ms	100 Hz	1 Hz	0.75 mV	0.74 mV	0.78 mV	0.79 mV	0.80 mV	0.74 mV	0.75 mV
43,50 Hz	23,0 ms	114,94 ms	100 Hz	1 Hz	0,32 mV	0,31 mV	0,33 mV	0,28 mV	0,27 mV	0,24 mV	0,25 mV
43,50 Hz	23,0 ms	137,93 ms	100 Hz	1 Hz	0,07 mV	0,08 mV	0,06 mV	0,04 mV	0,02 mV	0,02 mV	-0,02 mV
43,50 Hz	23,0 ms	160,92 ms	100 Hz	1 Hz	-0,15 mV	-0.15 mV	-0,16 mV	-0,16 mV	-0,19 mV	-0,20 mV	-0,23 mV
43,50 Hz	23,0 ms	183,91 ms	100 Hz	1 Hz	-0,42 mV	-0,42 mV	-0,43 mV	-0,44 mV	-0,43 mV	-0,45 mV	-0,48 mV
43,50 Hz	23,0 ms	206,90 ms	100 Hz	1 Hz	-0.65 mV	-0.65 mV	-0.65 mV	-0.65 mV	-0.65 mV	-0.65 mV	-0.64 mV
43,50 Hz	23,0 ms	229,89 ms	100 Hz	1 Hz	-1,08 mV	-1,09 mV	-1,06 mV	-1,06 mV	-1,03 mV	1,02 mV	-2,92 mV
43,50 Hz	23,0 ms	252,87 ms	100 Hz	1 Hz	-0,52 mV	-0,37 mV	-0,48 mV	-0,48 mV	-0,46 mV	-0,45 mV	-0,43 mV
43,50 Hz	23,0 ms	275,86 ms	100 Hz	1 Hz	-0,13 mV	-0,16 mV	-0,10 mV	-0,09 mV	-0,10 mV	-0,07 mV	-0,04 mV
43,50 Hz	23,0 ms	298,85 ms	100 Hz	1 Hz	0,20 mV	0,21 mV	0,12 mV	0,24 mV	0,27 mV	0,27 mV	0,27 mV
43,50 Hz	23,0 ms	321,84 ms	100 Hz	1 Hz	0,45 mV	0,44 mV	0,43 mV	0,45 mV	0,45 mV	0,45 mV	0,45 mV
43,50 Hz	23,0 ms	344,83 ms	100 Hz	1 Hz	0,93 mV	0,95 mV	0,45 mV	0,97 mV	0,97 mV	0,99 mV	1,00 mV
43,50 Hz	23,0 ms	367,82 ms	100 Hz	1 Hz	0,50 mV	0,53 mV	0,76 mV	0,52 mV	0,53 mV	0,56 mV	0,52 mV
43,50 Hz	23,0 ms	390,80 ms	100 Hz	1 Hz	0,44 mV	0,43 mV	0,45 mV	0,40 mV	0,38 mV	0,34 mV	0,33 mV
43,50 Hz	23,0 ms	413,79 ms	100 Hz	1 Hz	0,18 mV	0,18 mV	0,17 mV	0,14 mV	0,13 mV	0,12 mV	0,09 mV
43,50 Hz	23,0 ms	436,78 ms	100 Hz	1 Hz	0.05 mV	0.05 mV	0.05 mV	0.05 mV	0.05 mV	0.05 mV	0.05 mV
43,50 Hz	23,0 ms	459,77 ms	100 Hz	1 Hz	-0,04 mV	-0,06 mV	-0,07 mV	-0,09 mV	-0,11 mV	-0,12 mV	-0,15 mV
43,50 Hz	23,0 ms	482,76 ms	100 Hz	1 Hz	-0,34 mV	-0,35 mV	-0,37 mV	-0,37 mV	-0,39 mV	-0,40 mV	-0,41 mV
43,50 Hz	23,0 ms	505,75 ms	100 Hz	1 Hz	-0,45 mV	-0,45 mV	-1,06 mV	-0,45 mV	-1,06 mV	-0,46 mV	-1,08 mV
43,50 Hz	23,0 ms	528,74 ms	100 Hz	1 Hz	-1,01 mV	-1,00 mV	-0,54 mV	-0,97 mV	-0,53 mV	-0,97 mV	-0,52 mV
43,50 Hz	23,0 ms	551,72 ms	100 Hz	1 Hz	-0,40 mV	-0,38 mV	-0,20 mV	-0,36 mV	-0,18 mV	-0,33 mV	-0,13 mV
43,50 Hz	23,0 ms	574,71 ms	100 Hz	1 Hz	-0.10 mV	-0.10 mV	-0.10 mV	-0.10 mV	-0.10 mV	-0.10 mV	-0.10 mV
43,50 Hz	23,0 ms	597,70 ms	100 Hz	1 Hz	-0,03 mV	-0,01 mV	0,00 mV	0,01 mV	0.03 mV	0,05 mV	0,17 mV
43,50 Hz	23,0 ms	620,69 ms	100 Hz	1 Hz	0,34 mV	0,35 mV	0,38 mV	0,39 mV	0,40 mV	0,91 mV	0,96 mV
43,50 Hz	23,0 ms	643,68 ms	100 Hz	1 Hz	1,03 mV	1,05 mV	1,04 mV	1,05 mV	1,05 mV	0,51 mV	0,52 mV
43,50 Hz	23,0 ms	666,67 ms	100 Hz	1 Hz	0,55 mV	0,54 mV	0,54 mV	0,53 mV	0,52 mV	0,44 mV	0,44 mV
43,50 Hz	23,0 ms	689,66 ms	100 Hz	1 Hz	0,28 mV	0,28 mV	0,25 mV	0,28 mV	2,27 mV	6,37 mV	6,37 mV
43,50 Hz	23,0 ms	712,64 ms	100 Hz	1 Hz	0,12 mV	0,13 mV	0,10 mV	0,14 mV	0,15 mV	0,12 mV	0,13 mV
43,50 Hz	23,0 ms	735,63 ms	100 Hz	1 Hz	0,04 mV	0,01 mV	0,01 mV	0,02 mV	-0,03 mV	-0,02 mV	-0,03 mV
43,50 Hz	23,0 ms	758,62 ms	100 Hz	1 Hz	-0,18 mV	0,18 mV	-0,26 mV	-0,23 mV	-0,30 mV	-0,27 mV	-0,34 mV
43,50 Hz	23,0 ms	781,61 ms	100 Hz	1 Hz	-0,43 mV	-0,44 mV	-0,41 mV	-0,45 mV	-0,43 mV	-0,47 mV	-0,44 mV
43,50 Hz	23,0 ms	804,60 ms	100 Hz	1 Hz	-1,07 mV	-1,06 mV	-1,03 mV	-1,04 mV	-1,02 mV	-1,05 mV	-1,01 mV
43,50 Hz	23,0 ms	827,59 ms	100 Hz	1 Hz	-0,49 mV	-0,48 mV	-0,46 mV	-0,44 mV	-0,42 mV	-0,43 mV	-0,40 mV
44,50 Hz	22,5 ms	850,06 ms	100 Hz	1 Hz	-0.21 mV	-0.20 mV	-0.26 mV	-0.25 mV	-0.20 mV	-0.22 mV	-0.24 mV
44,50 Hz	22,5 ms	872,53 ms	100 Hz	1 Hz	-0,10 mV	-0,09 mV	-0,07 mV	-0,05 mV	-0,02 mV	-0,02 mV	-0,02 mV
44,50 Hz	22,5 ms	895,00 ms	100 Hz	1 Hz	0,23 mV	0,25 mV	0,23 mV	0,28 mV	0.25 mV	0,29 mV	0,29 mV
44,50 Hz	22,5 ms	917,47 ms	100 Hz	1 Hz	0,96 mV	0,98 mV	1,03 mV	1,07 mV	1,03 mV	1,85 mV	2,28 mV
44,50 Hz	22,5 ms	939,95 ms	100 Hz	1 Hz	0,52 mV	0,51 mV	0,50 mV	0,51 mV	0.54 mV	0,52 mV	0,56 mV
44,50 Hz	22,5 ms	962,42 ms	100 Hz	1 Hz	0,41 mV	0,41 mV	0,39 mV	0,35 mV	0.34 mV	0,36 mV	0,30 mV
44,50 Hz	22,5 ms	984,89 ms	100 Hz	1 Hz	0,09 mV	0,14 mV	0,13 mV	0,10 mV	0,09 mV	0,08 mV	0,04 mV

Figura 72-Exemplo de um ficheiro gravado.

Corrente consumida

A corrente consumida é uma característica importante do sistema e deve ser o menor possível. Os consumos da *headstage* e da PSoC 4 já foram referidos anteriormente, sendo que serão apresentados agora os resultados finais, pois esses consumos dependem do sistema implementado. A *headstage*, com uma alimentação de 3.6 V o VSTIM consume 0.98 mA e o resto da *headstage* consome cerca de 9 mA. O total é cerca de 9.8 mA.

O módulo PSoC 4 BLE alimentado com 3.6 V, tem um consumo de energia de proximamente 8.5 mA, utilizando um intervalo de conexão de 12.5 ms e contanto que o sistema se encontra sempre a enviar informação para a interface gráfica.

No total temos um sistema que consome aproximadamente 18.3 mA. Para a alimentação do sistema, pode ser utilizado, por exemplo, uma pilha de botão, recarregável de ião-lítio, modelo LiR2450 que possui uma tensão nominal de 3.6 V e uma capacidade para 120 mAh [55]. Como a corrente do sistema é de ≈ 18.3 mA, teoricamente teremos uma duração de ≈ 6.5 horas do sistema, com esta pilha.

Sistema futuro

No futuro, o objetivo é juntar o sistema todo numa placa PCB para assim ser possível fazer os testes num rato. Atualmente o sistema é demasiado grande para o rato. Na figura 73 encontra-se o sistema projetado, sendo necessário apenas uma única placa PCB que contenha os chips RHS2116 e CYC4247LQI-BL483, a antena Bluetooth e a ficha de entrada para os sinais da sonda neural. Para além destes componentes serão necessárias também resistências e condensadores. Do lado de trás deverá ter um *socket* de bateria para a alimentação do sistema.

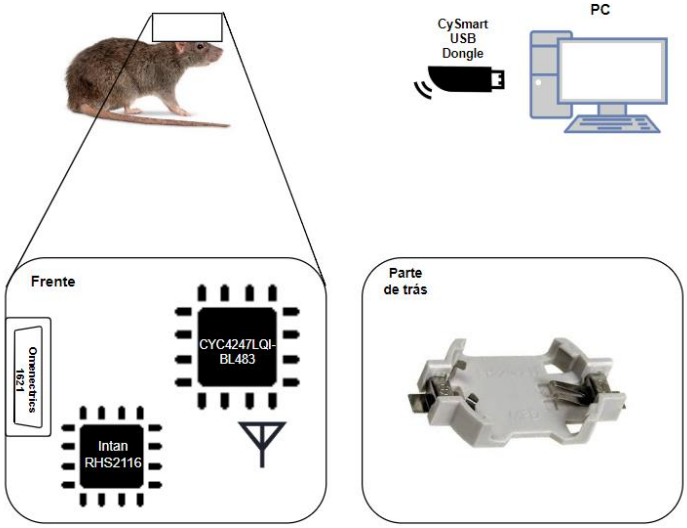


Figura 73-Sistema futuro numa só PCB.

CONCLUSÕES E TRABALHO FUTURO

Conclusões

O objetivo desta dissertação centrou-se na implementação de um sistema de comunicação sem fios com recurso à tecnologia BLE, para sondas neuronais. Sistema esse com capacidade de visualizar numa interface gráfica sinais eletrofisiológicos de pequenos animais e capacidade para estimulação optogenética. A peculiaridade deste sistema incide na não utilização de fios, para assim permitir ao animal testado andar livremente.

O sistema encontra-se divido em três partes, a parte eletrónica, o sistema de comunicação e a interface gráfica. A *headstage* da Intan RHS2116 é responsável pela parte eletrónica tratando pela aquisição dos sinais do sensor neuronal, pela sua amplificação e digitalização. Para além disso também faz parte o LED utilizado na estimulação optogenética. O sistema de comunicação é constituído por dois módulos BLE, um módulo ligado à parte eletrónica e que envia os sinais convertidos para o outro módulo BLE, que se encontra ligado ao PC. O *software*, desenvolvida em C#, é capaz de configurar a estimulação optogenética e tratar da receção dos sinais digitalizados conseguindo o utilizador observá-los em gráficos, onde cada canal possui um gráfico correspondente.

O desenvolvimento do sistema compreendeu uma série de passos, até ao sistema final se encontrar completo. Começou-se pelo estudo dos módulos BLE e a implementação do seu diálogo. Após a implementação da comunicação entre o computador e o módulo BLE, foi estudada a *headstage*, o seu funcionamento e como se programava. Só depois a interface gráfica começou a ser pensada e implementada. Sempre que alguma funcionalidade era implementada na interface gráfica, tinha de se implementar no módulo BLE pois trabalham em simultâneo e ambos têm de se "entender" um ao outro. O código foi desenvolvido de acordo com o pretendido no sistema, sendo otimizado com o objetivo de aumentar ao máximo a taxa de amostragem, que por si só já é baixa devido à utilização do BLE.

Através da interface gráfica somos capazes de controlar o sistema na totalidade. O utilizador pode escolher quais os canais que se encontram ativos, podendo desativar ou ativar a qualquer momento, qualquer canal, dependendo do objetivo do utilizador. Pode alterar a largura de banda do sinal que pretende adquirir a qualquer momento. Também a funcionalidade de gravação se encontra implementada, permitindo gravar os dados num ficheiro de texto, sendo

que fica a cargo do utilizador escolher o local de gravação e o nome do ficheiro, sem esses dados não é possível ativar a gravação.

Para além das funcionalidades explicadas em cima, é possível ativar a estimulação optogenética. Existem dois modos, o utilizador conforme o modo escolhido pode introduzir os dados para a ativação do LED. Para se verificar quando o LED se encontra ligado ou não, existe um gráfico com esse prepósito.

Os testes efetuados ao sistema ditaram uma taxa de amostragem de aproximadamente 41 Samples/s por canal com os 16 canais ativos, aumentando para 550 Samples/s com apenas um canal ativo. O sistema garante o seu funcionamento para frequências do sinal de entrada iguais ou menores a 4.1 Hz até uma distância de pelo menos 15 metros. Em termos de consumo, temos cerca de 18.4 mA de corrente e 65.8 mW de potência, alimentado com 3.6 V. Sendo que teoricamente é possível o sistema funcionar durante ≈ 6.5 horas seguidas (valor teórico). O valor calculado encontra-se dentro do alcançado por outros sistemas desenvolvidos em Universidades. Sendo um dos objetivos um sistema com um baixo consumo, esse objetivo foi atingido, mas as custas de uma baixa taxa de amostragem.

Trabalho futuro

Como trabalho futuro, o próximo passo deste sistema passa pelo desenvolvimento da PCB apresentada no capítulo 6.4, para assim o sistema BLE se encontrar completo a nível de *hardware*. Ao desenharmos a nossa própria PCB, o sistema terá dimensões menores e assim poderá ser utilizado em pequenos animais. Depois de desenvolvida a PCB, os testes *in-vivo* são fundamentais para a validação do sistema.

Para além da PCB, para o sistema ter uma maior eficiência na gravação de sinais neurais, em vez de BLE, a tecnologia de comunicação sem fios utilizada deverá ser outra, como por exemplo WiFi ou Comunicação Rádio. Através destas duas tecnologias de comunicação sem fios, o objetivo de alcançar uma taxa de amostragem que os sistemas de gravação neural pretendem seria atingido, superiores a 15 *kSamples/s*, pois ambas possuem uma maior taxa de transmissão. Pelo lado negativo, o consumo seria muito mais elevado do que o apresentado neste sistema.

BIBLIOGRAFIA

[1] M. Yin, “A Multi-Channel Wireless Implantable Neural Recording System,” 2009.

[2] K. Deisseroth, G. Feng, A. K. Majewska, G. Miesenbock, A. Ting, and M. J. Schnitzer, “Next-Generation Optical Technologies for Illuminating Genetically Targeted Brain Circuits,” *J. Neurosci.*, vol. 26, no. 41, pp. 10380–10386, 2006.

[3] T. A. Szuts *et al.*, “A wireless multi-channel neural amplifier for freely moving animals,” in *Nature Neuroscience*, 2011.

[4] H. Miranda, V. Gilja, C. A. Chestek, K. V. Shenoy, and T. H. Meng, “HermesD: A high-rate long-range wireless transmission system for simultaneous multichannel neural recording applications,” in *IEEE Transactions on Biomedical Circuits and Systems*, 2010.

[5] W. RW and H. K. “The control of neuron number” *Annual Review of Neuroscience*, vol. 11, pp 423-453,1988.

[6] “O que é neurônio? - Brasil Escola.” [Online]. Available: https://brasilescola.uol.com.br/o-que-e/biologia/o-que-e-neuronio.htm. [Accessed: 04-Sep-2018].

[7] “Toucas de EEG MCScap-E Classic | Kandel.” [Online]. Available: https://kandel.com.br/acessorios/eeg/toucas-de-eeg-mcscap-e-classic/. [Accessed: 04-Sep-2018].

[8] M. A. L. Nicolelis, “Actions from thoughts,” *Nature*. 2001.

[9] J. F. R. S. Pringle, B. From, E. Franklin, F. R. S. Author, and B. Franklin, “An Account of the Effects of Electricity in Paralytic Cases. In a Letter to,” *Source Philos. Trans.*, vol. 50, pp. 481–483, 1683.

[10] R. C. Gesteland, J. Y. Lettvin, B. Howland, B. Howland, and W. H. Pitts, “Comments on Microelectrodes,” *Proc. IRE*, vol. 47, no. 11, pp. 1856–1862, 1959.

[11] M. HajjHassan, V. Chodavarapu, and S. Musallam, “NeuroMEMS: Neural probe microtechnologies,” *Sensors*, vol. 8, no. 10. pp. 6704–6726, 2008.

[12] K. D. Wise, A. M. Sodagar, Y. Yao, M. N. Gulari, G. E. Perlin, and K. Najafi, “Microelectrodes, microelectronics, and implantable neural microsystems,” *Proc. IEEE*, vol. 96, no. 7, pp. 1184–1202, 2008.

[13] J. Muthuswamy, M. Okandan, T. Jain, and A. Gilletti, “Electrostatic microactuators for precise positioning of neural microelectrodes,” *IEEE Trans. Biomed. Eng.*, vol. 52, no. 10, pp. 1748–1755, 2005.

[14] S. Myllymaa, K. Myllymaa, and R. Lappalaine, “Flexible Implantable Thin Film Neural Electrodes,” in *Recent Advances in Biomedical Engineering*, InTech, 2009.

[15] R. A. Normann, E. M. Maynard, P. J. Rousche, and D. J. Warren, “A neural interface for a cortical vision prosthesis,” *Vision Res.*, vol. 39, no. 15, pp. 2577–2587, 1999

[16] K. Deisseroth, “Optogenetics,” *Nature Methods*, vol. 8, no. 1. pp. 26–29, 2011.

[17] O. Yizhar, L. E. Fenno, T. J. Davidson, M. Mogri, and K. Deisseroth, “Optogenetics in Neural Systems,” *Neuron*, vol. 71, no. 1. pp. 9–34, 2011.

[18] D. Tse and V. Pramod, *Fundamentals of wireless communication*, vol. 9780521845274. 2005.

[19] W. C. Brown, “The History of Power Transmission by Radio Waves,” *IEEE Trans. Microw. Theory Tech.*, vol. 32, no. 9, pp. 1230–1242, 1984.

[20] E. Ferro and F. Potortì, “Bluetooth and Wi-Fi wireless protocols: A survey and a comparison,” *IEEE Wireless Communications*. 2005.

[21] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, “Device-to-device communications with WiFi direct: Overview and experimentation,” *IEEE Wirel. Commun.*, vol. 20, no. 3, pp. 96–104, 2013.

[22] M. Banzi, “Zigbee Wireless Networking,” *Chem. …*, p. 130, 2011.

[23] C. Bisdikian, “An overview of the Bluetooth wireless technology,” *IEEE Commun. Mag.*, 2001.

[24] C. Gomez, J. Oller, and J. Paradells, “Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology,” *Sensors*, vol. 12, no. 12, pp. 11734–11753, 2012.

[25] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen, “How low energy is bluetooth low energy? Comparative measurements with ZigBee/802.15.4,” in *2012 IEEE Wireless Communications and Networking Conference Workshops, WCNCW 2012*, 2012.

[26] “A brief introduction to Bluetooth Low Energy – Alessandro Pagiaro.” [Online]. Available: https://apagiaro.it/bluetooth-low-energy-presentation/.

[27] “Generic Access Profile (GAP) — BLE-Stack User’s Guide for Bluetooth 4.2 3.01.00.05 documentation.”[Online].Available:http://dev.ti.com/tirex/content/simplelink_cc2640r2_sdk_1_40_00_45/docs/blestack/ble_user_guide/html/ble-stack-3.x/gap.html.

[28] “Gi-Fi (New Era of Wireless Technology) – S. DHEERAJ & S. GOPICHAND.” [Online]. Available: http://www.yuvaengineers.com/gi-finew-era-of-eireless-technology-s-dheeraj-s-gopichand/

[29] "SSR LLC – Quick Thoughts: For Payments, Bluetooth Puts a Beat Down on NFC and WiFi." [Online]. Available: http://ssrllc.com/quick-thoughts-for-payments-bluetooth-puts-a-beat-down-on-nfc-and-wifi/.

[30] R. Bashirullah, J. G. Harris, J. C. Sanchez, T. Nishida, and J. C. Principe, "Florida Wireless Implantable Recording Electrodes (FWIRE) for Brain Machine Interfaces," 2007 IEEE Int. Symp. Circuits Syst., 2007.

[31] R. E. Hampson, V. Collins, and S. A. Deadwyler, "A wireless recording system that utilizes Bluetooth technology to transmit neural activity in freely moving animals," J. Neurosci. Methods, 2009.

[32] X. Ye et al., "A portable telemetry system for brain stimulation and neuronal activity recording in freely behaving small animals," J. Neurosci. Methods, 2008.

[33] A. Ghomashchi, Z. Zheng, N. Majaj, M. Trumpis, L. Kiorpes, and J. Viventi, "A low-cost, open-source, wireless electrophysiology system," in 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014, 2014.

[34] "About." [Online]. Available: https://www.jinga-hi.com/about.

[35] "Hardware." [Online]. Available: https://www.jinga-hi.com/hardware.

[36] "About." [Online]. Available: https://www.jinga-hi.com/about.

[37] "RatLog-64 - Deuteron Technologies Electronics for Neuroscience." [Online]. Available: http://deuterontech.com/32-64-channel-loggers/.

[38] "RatLog-64 - Deuteron Technologies Electronics for Neuroscience." [Online]. Available: http://deuterontech.com/32-64-channel-loggers/.

[39] "W2100-HS16 | www.multichannelsystems.com." [Online]. Available: https://www.multichannelsystems.com/products/w2100-hs16.

[40] "About Neuralynx, Inc." [Online]. Available: https://neuralynx.com/home/about.

[41] "FreeLynx (formerly Cube2). Neuralynx." [Online]. Available: https://neuralynx.com/hardware/freelynx.

[42] "Company." [Online]. Available: http://www.trianglebiosystems.com/company.html.

[43] "IW-Series." [Online]. Available: http://www.trianglebiosystems.com/iw-series-systems.html.

[44] "TBSI W16 System Manual Version 1.0" [Online] Available: http://www.trianglebiosystems.com/assets/tbsi_iw16_systemmanual2016.pdf.

[45] "RHS2116 Digital Electrophysiology Stimulator / Amplifier Chip," 2016. [Online] Available: http://intantech.com/files/Intan_RHS2116_datasheet.pdf.

[46] “RHS2116 Headstage-16 Board,” 2016. [Online] Available: http://intantech.com/downloads.html

[47] “Intan LVDS Adapter Board” 2017 [Online] Available: http://intantech.com/files/Intan_LVDS_adapter_board.pdf

[48] “Datasheet ELC-470-34” 2016. [Online] Available: http://www.roithner-laser.com/datasheets/led_chip/elc-470-34.pdf.

[49] “About Cypress.” [Online]. Available: http://www.cypress.com/about-cypress.

[50] “Programmable System-on-Chip (PSoC),” 2017. [Online] Available: http://www.cypress.com/file/416486/download

[51] “PSoC ® 4 BLE and PRoC™ BLE: Bluetooth LE 4.2 Features Associated Part Family: CYBL11X7X and CY8C4XX8-BL5XX Related Application” [Online] Available: http://www.cypress.com/file/224826/download

[52] “CYPRESS SEMICONDUCTOR © 2016 CY8CKIT-142 PSoC 4 BLE 256KB” [Online] Available: http://www.cypress.com/documentation/development-kitsboards/cy8ckit-042-ble-bluetooth-low-energy-ble-pioneer-kit

[53] “CY5670 CYSMART USB Dongle” 2015. [Online] Available: http://www.cypress.com/file/140721/download

[54] J. W. Leis, Digital Signal Processing Using MATLAB for Students and Researchers. 2011.

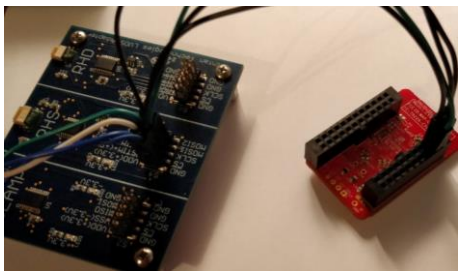
[55] J. Wang, Y. Yamada, K. Sodeyama, C. H. Chiang, Y. Tateyama, and A. Yamada, “Lithium-Ion Battery,” Nat. Commun., vol. 7, no. May, pp. 1–9, 2016.

ANEXO I

Protocolo para inicialização e funcionamento do sistema

Preparação do sistema

1. Ligação PSoC 4 BLE – LVDS Adapter board:



CS – P0.2.

SCLK – P0.3.

MISO – P0.1.

MOSI – P0.0.

2. Alimentação LVDS Adapter board:

VDD – 3.6 V.

GND – 0 V.

VSTIM+- +3.6 V.

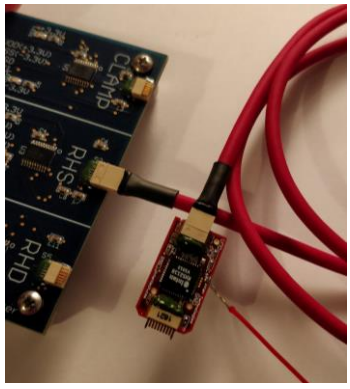
VSTIM- - -3.6V.

3. Alimentação PSoC 4 BLE:

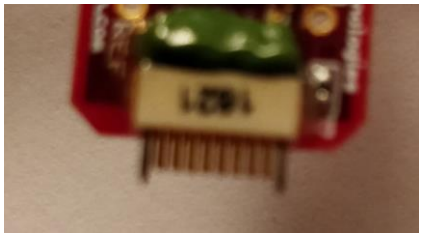
VDDD- 3.6 V.

GND- 0 V.

4. Ligar *headstage* RHS2116 a LVDS Adapter board através do cabo SPI.



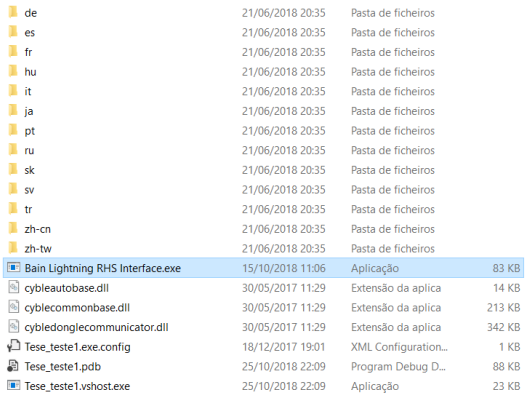
5.Ligar ao conector Omnectrics 1621 os sinais que se pretendem analisar.



6.Ligar ao computador a CySmart USB Dongle.

Inicialização da Interface Gráfica1

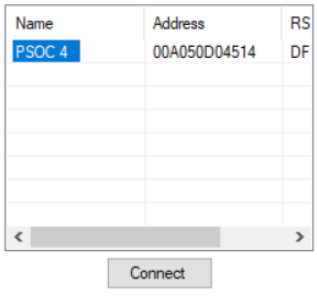
1.Abrir o ficheiro com o nome “Brain Lightning RHS Interface.exe”.



2. Escolher a COMX correspondente à CySmart USB Dongle.

3. Clicar no botão *First Connect.*

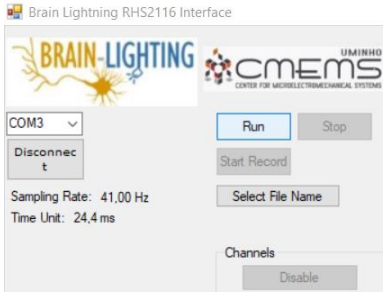
4.No separador *Second Connect* escolher o módulo PSoC 4 BLE.



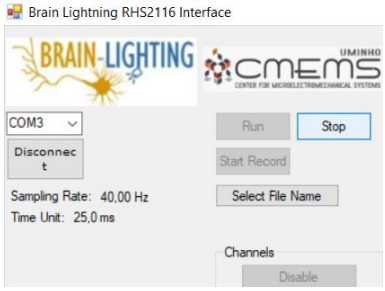
5.Clicar no botão *Connect.*

Conversão dos sinais eletrofisiológicos

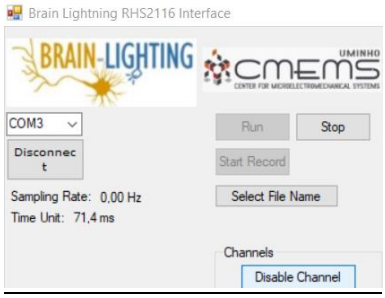
1.Clicar botão *Run* para ativar a conversão dos sinais.



2.Clicar botão *Stop* para parar a conversão.



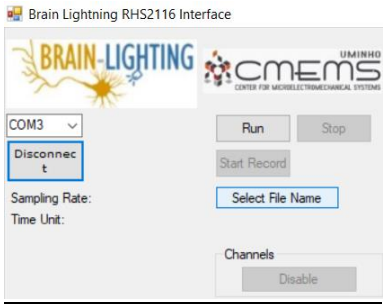
3.Para desativar um canal, clicar no canal que pretende desativar, e de seguida clicar no botão *Disable Channel*, no separador *Channels*, ao lado dos gráficos.



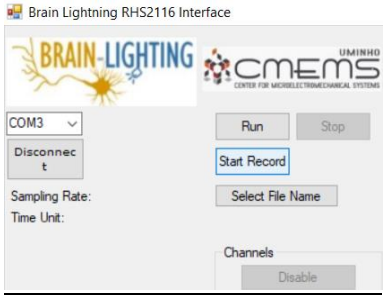
4. O mesmo procedimento para voltar a ativar.

<u>Gravação de dados</u>

1.Clicar botão *Select File Name*, de seguida escolher o local e o nome do ficheiro.

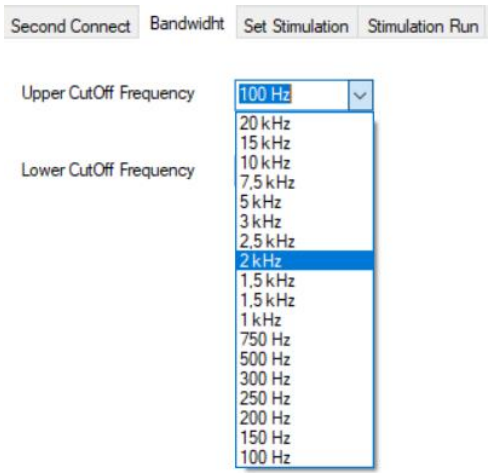


2.Clicar no botão *Start Recording* para ativar a gravação.



<u>Alteração largura de banda</u>

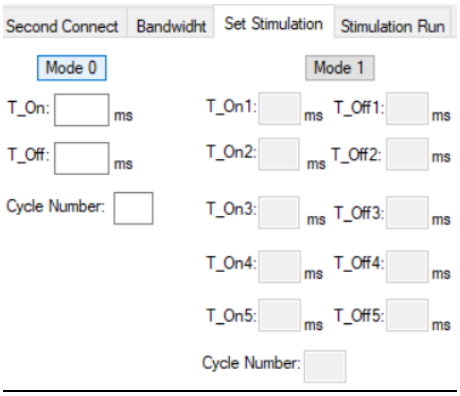
1.Escolher os novos valores para a frequência de corte baixa e frequência de corte alta



2.Clicar no botão *Set* para a alteração.

<u>Estimulação</u>

1.Escolher o modo de utilização pretendido, no separador *Set Stimulation.*



2.Configurar os tempo a *On, Off* e o número de ciclos pretendido.

3.No separador *Stimulation Run*, clicar no botão *Enable*, para a ativação do LED.