POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Title

Tesi di Laurea Magistrale in
Xxxxxxx Engineering - Ingegneria Xxxxxxx

Author: **Matteo Regge and Manuel Stoppiello**

Student ID: 10619213
Advisor: Prof. Maurizio Magarini
Co-advisors: Antonio Coviello
Academic Year: 2023-24

Abstract

Here goes the Abstract in English of your thesis followed by a list of keywords. The Abstract is a concise summary of the content of the thesis (single page of text) and a guide to the most important contributions included in your thesis. The Abstract is the very last thing you write. It should be a self-contained text and should be clear to someone who hasn't (yet) read the whole manuscript. The Abstract should contain the answers to the main scientific questions that have been addressed in your thesis. It needs to summarize the adopted motivations and the adopted methodological approach as well as the findings of your work and their relevance and impact. The Abstract is the part appearing in the record of your thesis inside POLITesi, the Digital Archive of PhD and Master Theses (Laurea Magistrale) of Politecnico di Milano. The Abstract will be followed by a list of four to six keywords. Keywords are a tool to help indexers and search engines to find relevant documents. To be relevant and effective, keywords must be chosen carefully. They should represent the content of your work and be specific to your field or sub-field. Keywords may be a single word or two to four words.

Keywords: here, the keywords, of your thesis

Abstract in lingua italiana

Qui va l'Abstract in lingua italiana della tesi seguito dalla lista di parole chiave.

Parole chiave: qui, vanno, le parole chiave, della tesi

Contents

1 | Introduction

1.1. PNRelay project

Peripheral nerves injuries (PNI) are a significant cause of morbidity and disability today, they affect almost 20 milions in the USA alone, and deeply damage both the physical and psychological well-being of those who suffer from these injuries. [**?**] The loss of functionality is caused by the damage of the nervous systems pathways that connect the brain to peripheral parts of the body. Moreover PNI's receive a poor prognosis since the regenerative capabilities of the nervous systems are limited and slow.

Recently, though there have been promising works of reserach in neuroscience that aim at improving the well-being of people that suffer from PNI by developing implanted devices that can connect to the Pheripheral Nervous System (PNS). In particular, bi-directional interfaces have the goal to restore the communication between the brain and the peripheral area whose sensorimotor feedback has been lost, effectively restoring the nervous pathways that have been damaged. This kind of interfaces can be realized with nerve electrodes that are installed chirurgically and can measure the voltage of a certain nerve and assess its signal or stimulate them directly. In this deeply interesting and thriving area of research it has emerged the PNRelay project which has been presented as a collaboration effort between the Politecnico di Milano and the Politecnico di Torino and has the objective to target PNI related issues by developing a new peripheral nerve interface, capable of conveying information between brain and body peripherical areas. In particular, this interface has the goal to register the nerve electrical signal, send it to a computer so that it can be interpreted and classified via a deep learning model and then stimulate the right nerves in order to elicit the corresponding movement. Although muscle stimulation has not yet been incorporated, it remains a pivotal area for future exploration and development. The project is targeted for preclinical trials on rats and in particular, injuries to the sciatic nerve, but it nontheless has great potential for future clinical applications. The recorded signal thanks to the electrode is known as Electroneurography (ENG) signal which has the potential to provide a valuable control signal for closed-loop neuroprostheses [5] (from davide) "'davide"' Other biological signals have been used with the aim of restoring PNI.

Electromyography, (EMG) signals from muscle activity and Electroencephalogram (EEG) signals reflecting brain activity are used in restoring limb function. EMG signals decode control commands for powered upper-limb prostheses, facilitating dexterity and mobility, but the acquisition and use of such myosignals are cumbersome and complicated [12]–[14]. EEG signals can be used to decipher user intent, driving neuroprosthetic devices to restore lost limb functionality. In recent studies, these signals are used to examine the functions and movements behavior of humans [15], [16]. "'davide"'

The device that has been engineered as part of the PNRelay is made of two main subparts: one that is implanted and one that is external.
The internal implant is composed of different parts: the cuff electrodes to acquire the signal, the Senseback ASIC chip, wrapped in a biocompatible capsuel, that is required for the processing of the singal and a transdermal port for the powering of the wireless device.

Extra-neural cuff electrodes, positioned externally on the nerve's surface, are chosen for their lower invasiveness in acquiring ENG data [20]. The data transmission module, which is responsible for data signal processing, is currently being developed for full-body integration. The external component hosts the classifier, which by performing an online operation, allows the different stimuli to be recognized. The signal is then sent to the stimulator, which proceeds to close the loop. The overall scheme for animal testing is shown in Figure 1.1. During initial experimental phases, the classifier may operate on a computer connected to the external circuit, with the ultimate objective being its integration into the implanted circuitry. Due to the typically substan- tial variations in neurological data among individuals, classification algorithms in this domain are often tailored to specific subjects [3]. This further compounds the challenge of gathering a sufficient amount of biological data on same subject. Challenges persist in decoding neural signal information due to limitations imposed by acquisition invasiveness. While extra-neural cuff electrodes represent a less invasive option for chronic implantation [21], the classification of sensory stimuli recorded through them remains complex due to a limited SNR. Various techniques have been explored to address this challenge [22]–[24], but the optimal classification approach remains an open area for further research and exploration.

1.1.1. PNRelay Setup

1.1.2. PNRelay functionality (chiara objectives)

1.1.3. Ethical concerns

1.1.4. Thesis outline

Our master's thesis endeavors to provide a comprehensive understanding of the research project's scope and objectives, aiming to delve into various facets of the subject matter. In Chapter 2, we lay the foundation by presenting the necessary background information essential for contextualizing our study. Moving forward, Chapter 3 offers a detailed exploration of the current state-of-the-art of the topics incorporated in our work, offering insights into the existing research landscape.

In Chapter 4 we delineate our methodology, detailing the approach undertaken to conduct our research. Here, we expound upon the materials utilized and outline the practical implementations of our project. Moreover, this chapter serves as a platform to showcase our contribution to the PNRelay project, elucidating the innovative approaches and methodologies we have introduced.

Transitioning to Chapter 5, we present the results derived from our efforts, providing a comprehensive analysis and discussion of the outcomes obtained. Through an in-depth examination, we elucidate the implications and significance of our findings, shedding light on the potential benefits and applications of our endeavors. This chapter also serves as a repository of the data that underpins our conclusions, offering insights into future directions and avenues for further exploration within the realm of PNRelay.

Finally, Chapter 6 marks the culmination of our thesis, wherein we encapsulate the essence of our research journey. Here, we provide a reflective summary of our contribution, acknowledging both the strengths and limitations of our work. Furthermore, we offer insights into potential areas of improvement and future iterations, paving the way for continued advancements in the field.

2 | Background

In this section, we will present an outline of the information transmission process within the nervous system, particularly emphasizing the conveyance of sensory data. The objective is to gain insight into the operational flow of our project and to grasp the attributes of the signal requiring transmission. Additionally, we will delve into the distinctive features of ENG signals and the significance of extra-neural cuff electrodes, as they play pivotal roles in data acquisition for transmission purposes.

2.1. Nervous Systems

The nervous system orchestrates an organism's activities and ensures seamless communication throughout the body. In vertebrates, it is divided into two primary components: the central nervous system (CNS) and the peripheral nervous system (PNS). The CNS, consisting of the brain and spinal cord, acts as the main processing hub, whereas the PNS includes nerves that branch out from the CNS to various parts of the body.

On a cellular scale, the nervous system is distinguished by its specialized nerve cells, known as neurons. These neurons are equipped with unique features that facilitate the rapid and precise transmission of signals to other cells. These signals travel as electrochemical waves along slender fibers called axons, leading to the release of neurotransmitters at synapses—the contact points between neurons. In addition to neurons, the nervous system encompasses glial cells, or glia, which offer structural and metabolic support. Recent studies indicate that glia may also have important roles in signaling processes. "'need to rewrite""" "'add neurons picture"'

2.1.1. Central nervous system

The central nervous system (CNS) is essential for processing sensory information from the peripheral nervous system (PNS) and integrating it to form appropriate responses. It generates motor responses, coordinates voluntary movements, and manages reflex actions. The CNS also governs higher cognitive functions and emotions, maintains homeostasis, fa-

cilitates learning and memory, processes sensory inputs, and controls autonomic functions such as heart rate and digestion. Through these roles, the CNS ensures the organism's ability to perceive, respond, and adapt to its environment while maintaining internal stability. It consist of two main parts: the brain and the spinal cord. The brain is housed inside the cranium and is responsible for all the main functions of the brain, while the spinal cord can be found inside the vertebral column and is crucial in linking the CNS to the PNS, allowing the transmission of motor commands from the brain to the rest of the body and the sensory information back to the brain. The spinal cord is also responsible for reflexes, unconcious responses to quick or unexpected stimuli.

2.1.2. Peripheral nervous system

2.2. Implanted medical devices normative and regulation

In order to have a medical implanted device, such as the one taken into consideration in this thesis work, it should respect the premarket standards for the specific location where the device is intended to be approved, various organizations co-operate to guarantee the best standards [] 1 . In particular, the responsible authorities are Food and Drugs Administration in the United State (USA) and European Commission (CE) in the European Union (EU). The normative given by these authorities are necessary to guarantee safeness of the patient and a lower grade of risk for the patient who will use the specific medical device, considering for each one the trade off between risks and benefits, together with the ensure of efficiency of medical devices.

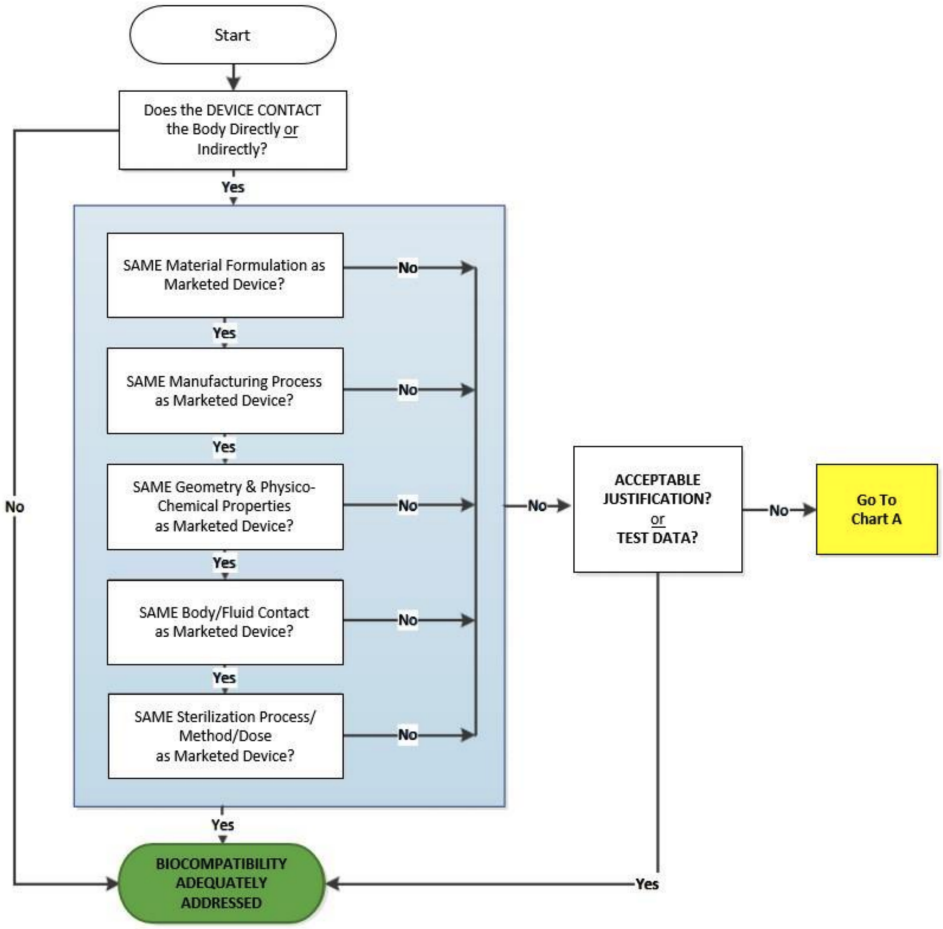
2.2.1. FDA

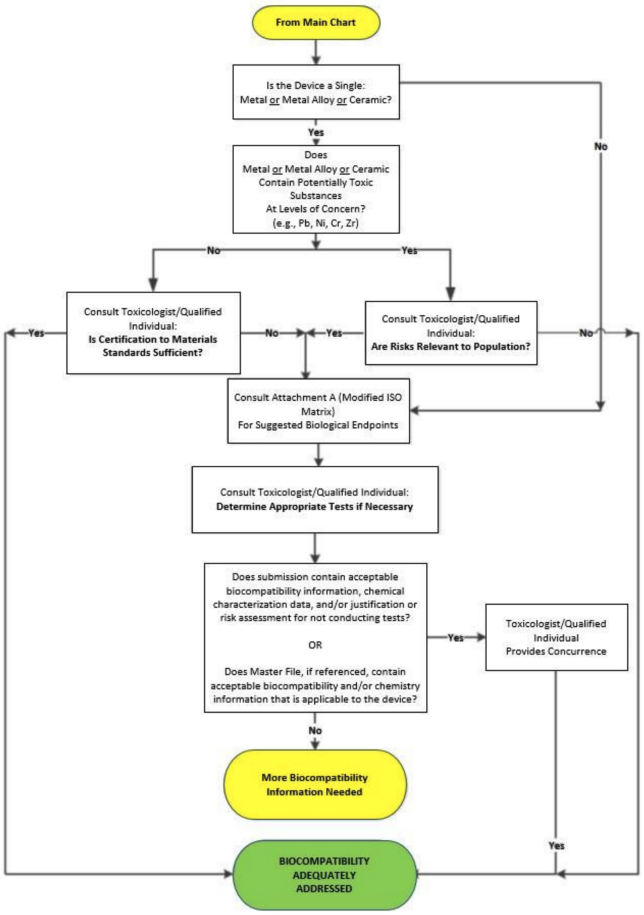
FDA administration subdivides the medical devices that should be evaluated and approved into three classes of risks, based on the specific application. Up to now, more than 1700 different devices have been classified and divided into the specific class. [**?**] The classes of risk that FDA utilizes and their relative needs to be approved are the following ones:

- Class I (General Controls): This class can be subdivided into two subclasses: with exe t4mptions and without exemptions.
- Class II (General Controls and Special Controls): this class can also be subdivided into the two subclasses with and without exemptions.

- Class III (General Controls and Premarket Approval)

The class is assigned to a specific device basing on a risk based approach, the more the device can be dangerous for the patient, the highest would be the class of risk and as a consequence, the more selective would be the approval practice. For what concern Class I and Class II a 510k is required. Class III is the only one which is subject to premarket approval (PMA), unless the device were on the market before the medical amendments in 1976, in which case a 510k could be sufficient. To determine the classification of the device for FDA it is possible to consult the device specific device panel [**?**] which best describes the product in analysis. For Senseback, the specific device panel is the one related to neurology. Specifically, the one which best fits the intended use is the 882.5870 "Implanted peripheral nerve stimulator for pain relief" [**?**] which makes Senseback falls into the Class of risk II. In order to understand how FDA values the biocompatibility of the devices with the body, standard iso 10993, in particular 10093-1 [**?**] should be taken into account. Recently, on September 8 2023, FDA has published a guidance [**?**] which explicit how to use this standard for a biological evaluation during the design of medical devices. ISO 10993-1 and FDA guidance ISO is an independent, non-governmental international organization [**?**]. It provides standards to guarantee to the user that products are safe as much as possible, providing guidance for each step of design and premarket of the device in analysis. The standard ISO 10993-1 focuses on the categorization of medical devices basing on the duration and the nature of contact, both for implanted and external devices. The flow charts reported in Figure 1 and in Figure 2 reports how to proceed for a biocompatibility evaluation.


Start
Does the DEVICE CONTACT the Body Directly or Indirectly?
Yes
SAME Material Formulation as Marketed Device?
No
Yes
SAME Manufacturing Process as Marketed Device?
No
Yes
SAME Geometry & Physico-Chemical Properties as Marketed Device?
No
Yes
SAME Body/Fluid Contact as Marketed Device?
No
Yes
SAME Sterilization Process/ Method/Dose as Marketed Device?
No
Yes
No
ACCEPTABLE JUSTIFICATION? or TEST DATA?
No
Go To Chart A
No
Yes
BIOCOMPATIBILITY ADEQUATELY ADDRESSED



As it is possible to see from these flowcharts, biocompatibility evaluation is not necessary only if the device taken into consideration has not any direct or indirect contact with the body, or if it has the same properties of a similar well known device present in the market. Moreover, devices are subdivided into three categories based on the nature of contact with the body: surface-contacting medical devices, which only have an external contact with the body such as the one in contact with the skin, externally communicating devices and implanted medical devices, which are again subdivided into their tissue of contact.

Medical device categorization by			Biological effect												
Nature of Body Contact		Contact Duration	Cytotoxicity	Sensitization	Irritation or Intracutaneous Reactivity	Acute Systemic Toxicity	Material-Mediated Pyrogenicity	Subacute/Subchronic Toxicity	Genotoxicity	Implantation	Hemocompatibility	Chronic Toxicity	Carcinogenicity	Reproductive/Developmental Toxicity#	Degradation@
Category	Contact	A – limited (≤24 h) B – prolonged (>24 h to 30 d) C – long term (> 30 d)													
Surface device	Intact skin	A	X	X	X										
		B	X	X	X										
		C	X	X	X										
	Mucosal membrane	A	X	X	X										
		B	X	X	X	X	O	X		X					
		C	X	X	X	X	O	X	X	X		X			
	Breached or compromised surface	A	X	X	X	X	X								
		B	X	X	X	X	X	X		X					
		C	X	X	X	X	X	X	X	X		X	X		
External communicating device	Blood path, indirect	A	X	X	X	X	X				X				
		B	X	X	X	X	X	X			X				
		C	X	X	X	X	X	X	X	X	X	X	X		
	Tissue+/bone/ dentin	A	X	X	X	X	X								
		B	X	X	X	X	X	X	X	X					
		C	X	X	X	X	X	X	X	X		X	X		
	Circulating blood	A	X	X	X	X	X		X^		X				
		B	X	X	X	X	X	X	X	X	X				

Figure 3: Table A part 1 in FDA guidance [**?**], biocompatibility evaluation endpoints.

Medical device categorization by			Biological effect												
Nature of Body Contact		Contact Duration	Cytotoxicity	Sensitization	Irritation or Intracutaneous Reactivity	Acute Systemic Toxicity	Material-Mediated Pyrogenicity	Subacute/Subchronic Toxicity	Genotoxicity	Implantation	Hemocompatibility	Chronic Toxicity	Carcinogenicity	Reproductive/Developmental Toxicity#	Degradation@
Category	Contact	A – limited (≤24 h) B – prolonged (>24 h to 30 d) C – long term (> 30 d)													
		C	X	X	X	X	X	X	X	X	X	X	X		
Implant device	Tissue+/bone	A	X	X	X	X	X								
		B	X	X	X	X	X	X	X	X					
		C	X	X	X	X	X	X	X	X		X	X		
	Blood	A	X	X	X	X	X		X	X	X				
		B	X	X	X	X	X	X	X	X	X				
		C	X	X	X	X	X	X	X	X	X	X	X		

Figure 4: Table A part 2 in FDA guidance [**?**], biocompatibility evaluation endpoints.

The table in Figure 3 and in Figure 4 shows, bases on the category of the product and the tissue and duration of contact, which could the possible biological effect that should be considered for the approval of the specific device. Senseback could fall into the external communicating device category, with a long-term contact and tissue surface of contact, with relative biological effects reported in the table. FDA suggests that this table should not be threated as a checklist for testing. Instead, some specific medical devices the reported effects could not be enough and other endpoints should be considered. For what concern cytotoxicity, the related ISO standard is 10993-5:2009 [**?**], a recent study [**?**]

has been conducted in order to verify whether this standard is clear enough to guarantee that devices development which have followed this standard are cytotoxicity safe. The conclusions report that different that this standard should be revised, because cytotoxicity tests performed by different companies with same standard cannot guarantee reliable and comparable results. This standard was last reviewed and confirmed in 2022. Therefore, this version remains current. UE regulations In UE the normative that regulate the medical devices approval are the 2017/745/UE [**?**] which is relative to medical devices and 2017/746/UE [**?**], which is relative to in vitro diagnostic. The first one is applied since 26, May 2021, the second one since 26, May 2022 [**?**]. In these regulations, as before, devices are divided into classes of risks from the lowest, which is Class I to Class II-a and Class II-b to Class III, which is the highest one. The device classification follows similar step as in FDA evaluation, in fact some parameters are evaluated such as duration of contact, invasiveness, specific medical purpose and anatomical location [] mcdg , it is up to the manufacturer to correctly classify the device according to the appropriate class of risk. As reported in the regulations, all implanted medical devices and long-term surgically invasive devices are classified as class IIb []mcdg . However, since Senseback is an active device, which means that needs a source of energy which is different from the body itself, it could fall into class III, the highest one. With the new normative UE 2023/607 [**?**] the deadline for the manufacturers to adequate to 2017/745/UE and 2017/746/UE has been postponed based on the class of risk of the devices as shown in Figure 5

Figure 5: new date of application of normative (EU) 2017/745 and (EU) 2017/746 according to (EU) 2023/607 [**?**]

2.2.2. Medical software regulation

According to the regulations EU 2017/745 [**?**] and EU 2017/746 [**?**], a medical software that is intended to be used alone or in combination with a hardware which fall under the definition of medical device should be considered as a medical device software (MDSW). In order to understand whether or not the software could be considered as a MDWS, the steps are reported in.

Figure 6: steps to verify whether or not the software falls into the MDSW definition according to European Commission [] b865

First step is to verify that the software under analysis fall under the definition of software in the regulations, as reported in [] untitled , which states that “software” is defined as a set of instructions that processes input data and creates output data. Second step is to

determine whether or not the software influences or drives the hardware device, if it does it could be classified as an accessory for the hardware medical device and as a consequence, be considered as a MDSW. If not, next steps need to verify which is the function for which the software is intended to be used, including whether or not the software provides an effective benefit to the patient. If it does and the software falls under the definition of MDSW in [] untitled, which states that a MDSW is a software that is intended to be used, alone or in combination, for a purpose as specified in the definition of a "medical device" in the medical devices regulation or in vitro diagnostic medical devices regulation, than the software should be treated as a medical device and it is covered by the new MDR regulations. To sum up, a software is classified as a medical device if it satisfies the following characteristics: it should be independent, so it should have its own medical purpose, it can influence or drive the hardware medical device and it may be intended to be used by medical professionals. Note that a software can be classified ad a MDSW no matter the location intended for it.

The rules to classify a MDSW are reported in Annex VII [**?**]. As it is reported in chapter II, a software which is stand alone and has not influence on the hardware medical device should be classified separately. If it is not, however, it should be classified considering the most critical specified use. I the software and the hardware have two different classes of risks, as a consequence, they both fall into the highest class.

Senseback software, then, falls into the class of risk III.

2.2.3. Normatives for wireless devices

In European Union, normative EU 2017/745 [**?**] treat a wide range of medical devices, including the ones with a radio frequency communication such as Bluetooth low energy like the case in analysis. It is reported that manufacturers should ensure that devices are designed in a way so that external interferences such as external environment or radio signal interferences are removed or at least reduced as much as possible, so it is for possible undesired interaction with the environment which is thought for the device to be used with. In the United States, instead, FDA has provided a regulation [**?**] which regards specifically radio frequency wireless medical devices, this is thought for all the devices which use at least one functionality with a wireless feature such as Bluetooth, WI-FI and so on. In this normative it is reported that wireless communication could be a benefit for the patient in terms of mobility, due to the fact that wires are not present anymore, moreover also a remote monitoring of the real-time health of the patient is possible. However, possible risks in patient daily life should be considered, and it is up

to the manufacturer to inform the patient about them. Firstly, a possible issue regards the fact that airways are shared, and the device could be influenced by other devices operating the near range of work of the medical device which is intended to be used. The increasing usage of these kinds of products, including the non-medical one, could condition the medical device performance. The suggested aspects that the manufacturer should take into consideration are the following ones then: Selection of the wireless technology

- Quality of service
- Coexistence
- Security
- Electromagnetic Compatibility (EMC)

For these, many standards are directly suggested by FDA itself, such as the directives AAMI TIR 69 [**?**] or ANSI C63-27/D1.0 [**?**] for what concern the Wireless coexistence or IEC 60601-1-2 [**?**] for electromagnetic compatibility (EMC). For the security field FDA issued a guidance [**?**] on September 2023 which regards specifically the cybersecurity in Medical devices, this document substitute the previous one [**?**] issued on October 2014.

2.2.4. FDA Medical devices cybersecurity

According to FDA guidance [**?**], manufacturers have the responsibility of the identification of possible cybersecurity risks associated to their devices, including both the ones related to the device itself and the ones related to the environment in which the product operates, for example the ones introduced by device reliance on hospital networks. FDA also recommends the submitting of a detailed documentation about the security features of the product which is intended to be approved by them. They also recommend that this information should take the form of views in order to effectively prove that the developed architectures are effective and safe. Manufacturers should demonstrate that security features are effectively been implemented and tested. The necessary requirements asked by FDA are the following ones:

- Authentication
- Authorization
- Cryptography
- Code, Data and Execution Integrity
- Confidentiality

- Event Detection and Logging
- Resiliency and recovery
- Updatability and Patchability

Note that these aspects are just the suggested ones, but the necessary features may vary depending on the specific intended usage of the device. These aspects are explained in detail in Appendix 1

Authentication

Authentication is divided into two controls:

- Authentication of information: this means that it is possible to prove that data are generated from a trusted and verified source and they have not been altered during the transmission to the endpoint.
- Authentication of entities: it is possible to prove the identity of an endpoint from which is the one which provides information.

The device then, needs to verify that information received from an external source are reliable and so are the ones generated from the device itself. Authenticity can then be evaluated for:

- Information at rest (stored)
- Information in transit (transmitted)
- Entity authentication of communication endpoints
- Software binaries
- Integrity of the execution state of currently running software
- Any other appropriate parts of the medical device system where manufacturer's threat model and/or risk analyses reveal the need for it

The effective strength of the authentication implemented is evaluated on the difficulty that an external unauthorised source would need to identify the decomposition of authentication scheme. A cryptographic algorithm in general should be preferred, this is because non-cryptographic ones are generally weak. As a consequence, an attacker could easily emulate the behaviour of an authorized user. Some of the other recommendations regards the usage of a proper authentication method such as the multi-factors ones, strong passwords choices, authentication requirement before possible software updates,

anti-replay measure in communications which can result harmful and the avoidance of cyclic redundant checks as security control. Furthermore, manufacturer should consider how the device reacts to a possible authentication failure.

Authorization

Authorization is required in order to prevent the access to sensitive information or resources. In a well-designed system, only an authorized which has fully permissions can have the access to specific functions of the medical device, depending of the least privilege principle. This means that, in case of a hacker attack, they should not be able to access to possible functionality reserved to the manufacturer in case they gained the credential associated to patient privilege. The recommendation about authorization regards the access to the device, which should be limited to authorized users, the usage of timer in order to terminate sessions and "deny by default" principle, which means that device should reject unauthorized connections by default.

Cryptography

About cryptography, this is suggested to be implemented because of its higher level of security. It is underlined that this should be properly implemented to avoid undesired vulnerabilities. The recommendation which regards this field suggest the usage of industry-standard cryptographic algorithms, in particular the one suggested in the current NIST standard for cryptography, the algorithm should also allow the device to use the highest level of security possible, unless otherwise necessary. Recommendation also suggests avoiding a situation in which the fully revealing of the key for a specific device could implicate the revealing of keys for other devices. Eventually, it is suggested avoiding downgrades in security level unless they are strictly necessary for the health of the patient.

Code, Data and Execution Integrity

Integrity is subdivided into three field: Code, Data and Execution. In the code field it is suggested, in addition to previous suggestions, to prefer, when possible, solution which are hardware-based, to disable the access to unauthorized ports such as the UART and to ensure that device is physically integer by employing the usage of tamper seals. For data integrity, the suggestions regard the verification of the received data coming from an external source, which should be validated and not modified during the transit and the protection of the data which are relevant for the safeness of the device. Finally, for

the execution integrity purposes it is suggested to use industry-accepted best practices to maintain the code integrity during the execution process and the design and review all code that handles external data.

Confidentiality

Even if authentication ad authorization suggestions provided in previous points should be enough to ensure confidentiality, manufacturers should verify if for the specific implementation it could be necessary to implement additional features. A loss of confidential could result in a potential harmful effect on the patient.

Event detection and logging

The suggestions regarding this point suggest that each security relevant event, in particular suspicious behaviours should be detected and logged promptly, including possible software changes or malfunctions. Manufacturers should also implement a log file in which these tracked events are stored. It is also suggested to design devices which are able to integrate antivirus/anti-malware protections.

Resiliency and recovery

Resiliency and recovery are the capabilities of the device to face with a safety margin a possible incident scenario and maintain availability. Manufacturers should then design devices which are resilient to possible incidents or noises, specifying the level of resilience that any component of the medical device have. Th design should also include methods for recovery default configurations and protections for critical functionality or data.

Firmware and software updates

Last recommendations regard software and firmware updates. These should anticipate the future cybersecurity vulnerabilities, should be reliable even in case of interruption or failure of the process, and the cybersecurity related ones should be separated from regular feature update cycles. Moreover, it is indicated that updates should be easily verified, validated and distributed. Manufacturers should also implement all the necessary tools and processes to ensure that updates are applied in a safely ad timely manner. Lastly, third party licenses should be maintained for the whole life of the device.

Cybersecurity regulations for medical devices in EU

Regarding cybersecurity, the European Union has recently issued the (EU) 2022/2555 [**?**] normative on Security of Networks and Information systems (NIS2), which is entered into force in January 2023. The normative is not just focused on medical device as the FDA guidance [**?**], but it also discusses about them. The first article reports that the purpose of this normative is to establish rules in terms of cybersecurity risk management for the topics which are considered "critical" in normative 2022/2557 [] publicationsoffice or the ones reported in the Annex I or Annex II of (EU) 2022/2555, such as point 5.a of Annex II which is referred to the fabrication of medical devices and diagnostic medical devices in vitro. In NIS2 a distinction had been made between important entities and essential ones, allowing to have in this way a fair trade-off between risks-based requirements and relative obligations and administrative burden stemming. As it is reported in Article 3, entities in Annex II could be classified as essential depending on whether or not they are classified like that by a Member State. One of the main differences in terms of regulations between important entities and essential entities is the fact that the essential ones are proactively supervised by the authorities, while important ones are subject just to a light ex post supervisory regime, which can be triggered in cases in which evidence of a possible infringement of the directives are brought to the authorities. In the article 21 point 2 of the NIS2 directive cybersecurity management measures are threated, which both essential and important entities should follow. It is reported that companies have to manage their own security risks and take adequate measures to manage them. The recommendations about the measures to take are reported below. They need to include:

- Policies on risk analysis and information system security
- Incident handling
- Business continuity and crisis management
- Supply chain security
- Security in network and system acquisition
- Policies and procedures to assess the effectiveness of cybersecurity risk management measures
- Cryptography and encryption, and multi-factor authentication
- Cybersecurity training and basic cyber hygiene practices

2.3. Ethical concerns

Even if scientific and biomedical research is essential for the development of our society, it helps people to recover from medical diseases which were chronical before for example, it is important to wonder which is the limit that medical research should reach. Ethical implications should then be considered, especially in some setups like the one considered in this Master Thesis, which purpose is to have in close future an effective usage in vivo. In all of the steps, from the premarket ones and relative experimentations to the post market ones and relative surveillance, ethic should be taken as a monitor, in order to guarantee that not only the device in use is safe and effective, but also the development of it has been made respecting environment, the work ethic of people involved in, and the eventual animals used for the experimentation phase. To do that, research should be the as transparent as possible, by clearly explaining the goals and the intended benefits of the project, the methods and design of the study and which could be the possible risks. When available the results of the study should be published, possibly even the experimental ones. This would contribute to the check of the results by other researchers, promoting the re-doability and consenting a critical evaluation of the work. In case of possible criticisms or questions coming from the scientific community, these should be answered, and corrective actions should be taken when necessary. The practicality of sharing the information on the project is essential in an ethical and responsible research. Some of the principal ethical topics that should be considered are treated in detail below.

2.3.1. Animal welfare

The first aspect that should be considered for this purpose is animal welfare. The World Organisation for Animal Health (OIE), defined animal welfare as follows: An animal is in a good state of welfare, if it is healthy, comfortable, well-nourished, safe, able to express innate behaviour, and if it is not suffering from unpleasant states such as pain, fear and distress [**?**]. Animal welfare, in this kind of research, implies the careful and responsible consideration of the management, the treatment and health conditions of the animals involved in the research. Animals are still necessary to test pharmaceutical or medical products before putting them into the market, because the experimental phase made on animals guarantees that each product on the market is effectively safe and which could be eventual collateral effects that patients should be warned about. However, the treatment of the animals involved in the experiments must be ethically correct. This means that this phase should respect current rules and animal welfare regulations, and this also means that animal experimentation is made only when necessary and the least number of animals

is involved. The Animal Welfare Act (AWA) []comps was the first policy responsible for guaranteeing the standards for animal welfare in the USA. It included rules relative to transportation, treatment during tests and research, teaching and dealings by animal dealers. This standard was issued in 1966, but it only covered some warm-blooded animals such as dogs, cats or rabbits, and some other animals like birds or rats, which are the ones mostly used in scientific experiments, were excluded from the act. The Act was amended eight times until now, however it is just in 2002 with the publication of Farm Security and Rural Investment Act []plaw that the excluded animals, including rats used in medical research, has been included into the definition of "animals" of the Animal Welfare Act. The European Union also issued some policies to define the minimum standards necessary for animal welfare, which are some of the world's highest animal welfare standards [**?**]. The current normative responsible for setting this kind of standards in the EU is the 2010/63/EU [**?**], published in 2010 and effective 1 January 2013. It is strictly related to animals used for scientific research and treats all the related steps, from development to manufacture to tests. The ultimate goal would be to have fully non-animal methods. By following these policies and by implementing the right measurements to guarantee animals well-being, it is possible to reduce the negative impact on the life of the animals involved in the study. Some of the terms that implies a sufficient welfare of the animal are: Implant procedure. The chirurgical procedure necessary for the implanting of the device should be painless and as less invasive as possible. It is important to reduce at the minimum the pain of the patient during this operation and guarantee that the patient would have sufficient time for recovery after that. Health monitoring. The health of the patient should be monitored for the whole time of the experiment. This includes the monitoring of vital functions, the behaviour of the animal and physiologic indicators. Duration of the experiment. The experiment duration should have the strictly necessary duration to get the necessary data for the experiment. The minimum number possible of involved animals should also be considered. Usage of positive and negative controls. They should be implemented in order to check if the observed effects are due to the intervention or external environment. Environment in which animals live. This should guarantee that animals' physiological needs are respected. These include water and food access other than movement range and social iteration. Biological behaviour of the animals. The natural behaviour of the animal involved in the experiment should be respected, including the possible negative impacts on their daily life, which should be minimised. In those cases in which euthanasia is necessary, it should be performed in agreement with the current normative of the corresponding country. This process should be conducted in a way which is ethical and respectful towards animals, making sure that this is conducted with attention on animal welfare. When possible, animal usage should be avoided in favour of

cultured cells and computer-based models. Transparency. The documentation of all the precautions taken during the study underlines the ethical diligence of the research.

2.3.2. Need of research

When approaching biomedical research, in which living beings are involved, it should be considered whether or not the intended research is necessary and whether or not the purpose of the research could be reached without animal usage. It is important for this purpose to make a trade-off between the possible benefits that people or animals could get from the research and the eventual damages caused to animals. The concept of "Need" for research refers to whether or not there is proper justification and needs to conduct the intended research. This means that developers should wonder whether or not the research faces a significant demand in the scientific development and in practical applications, these could justify resources usage and experimentations conducted on the animals. In order to make sure that ethical implications have correctly been monitored in the intended experiment, it is suggested to submit it to a careful revision from an ethical committee or institutional ones. To sum up, needs of the research means carefully wondering about the scientific and ethical value of the project, making sure that these justify animal involvement. These precautions make sure that research is conducted in a responsible way and it is in line with the highest ethical standards.

2.3.3. Safety of the patient

The concept of safety of the patient is fundamental for guaranteeing that the iteration between the device implanted in the animal's body and the external device happens in a safe way and it does not involve any risks for the animal's health. One of the first aspects that should be considered for this purpose, is the fact that materials should be safe and biocompatible. The utilised components should minimise risks and adverse reactions in surrounding tissues. A careful planning and a correct post-operation management could contribute to reducing the minimum uneasiness and collateral effects. Moreover, during the implantation procedure, protocols should be adopted in order to prevent possible risks of infections. Hygiene is essential to prevent issues that could intact the safety of the animal. Furthermore, considering that Senseback communicates with the external device wirelessly, the right procedures for assuring that the protocol of communication, in this case for Bluetooth Low Energy (BLE), is secure and cryptographed in order to protect sensitive data. This is essential for the protection of the privacy of the patient. Lastly, it is important to consider the procedure to follow in those cases in which the experiment presents issues or complications. This includes the planning of emergency

procedures and the possibility to interrupt the experiment if some warnings related to the safety of the patient come up.

2.3.4. Inclusivity

The concepts of inclusivity and heterogeneity are linked to ethical and social considerations related to equal access, representation and to the individual differences in the process of research and in the application of the results. Firstly, the project should promote equality in the access of the benefits coming from the research. This includes the consideration on how the application of the technology can be made available to different ethnic groups, avoiding discriminations and differences in the therapeutic approach. Talking about this specific project, this is intended to be experimented on animals in next few years, however in a distant future this should be made available on humans. For this reason, the differences between the group of patients should be taken into consideration, to make the project more general and inclusive. An example of this is a study, [**?**] proposed by Institute of Physic and Engineering in Medicine (IPEM), in which a whole-body voxel model for the average Japanese man and woman has been developed. This study shows how different are some physical effects such as SAR on the body. A comparison with previous studies, in which only Caucasian men were studied, has been performed in order to underline the differences. This study should be taken as a monitor to take into account that different bodies of men and women of different ethnicities could have some substantial differences. Moreover, the project should also respect individual differences in response to technology in order to maximise efficiency and minimise collateral effect. Some genetic, physiologic or lifestyle related variations or possible disabilities can influence the response of the patient to the treatment. The project should then be relatable for the higher range of individuals possible, including the ones with special needs. Furthermore, the manufacturer should also consider how the related project could impact society in terms of inclusion and diversity and which could be possible positive or negative effects on social dynamics and equity of health. Inclusivity could also mean the respect for different cultural sensitivities, which should be taken into account in the design of the project. Medical practices could vary by varying the country and these differences must be considered. Research could work in strict contact with the related community, this could help to better understand the needs and the expectation of the people involved. Also the informed agreement in case of experimentation on humans should be sensitive to culture and should be understandable for all the participants. Information contained in that should respect differences in terms of language and culture. All in all, the concepts of inclusivity and diversity focused on the obtaining of an equal approach, representative

and respectful of the differences both in the research phase and in the usage one. The attention to these considerations would contribute to guarantee that the project has a positive impact on a larger range of individuals and communities.

2.3.5. Informed agreement

Informed agreement is crucial in those projects in which humans are involved in experimental research or in the application of new medical technologies. The informed agreement is an ethical process which guarantees that all the participants have a complete view of the intended purposes, the risks and the possible benefits of the research or the medical intervention in which they have given their voluntary consent to participate. Before submitting anyone to experimental treatments or to procedures linked to the related technology, it is necessary to get an informed agreement. In the agreement it is important to furnish to the participants all the necessary details about the procedure, including the goals of the study, the kind of intervention that will be made and possible issues or risks. All of the available options should be illustrated, and information about possible alternative or standard treatments. This allows participants to make clear decisions about their participation. The agreement should also assure that anyone who is not purported to participate anymore could retire themselves from the study without any negative consequence. This enforces the principle of voluntary consent and respect about the autonomous decision of the participant. The operator should explain in detail and in a clear manner how technology will work in the context of treatment or research. The participants should have the chance to ask questions and receive clarifications about the utilised technology. For what concerns animals, since they cannot give a proper informed agreement, it becomes more and more important to have a consent from an ethical committee and from authorities. This will assure that animal welfare is respected. Ultimately, researchers should have a clear documentation of the informed agreement, including the details related to the explanation given, the comprehension from the participants and their signs. This documentation is fundamental for transparency and ethical conformity. To conclude, assuring that an informed agreement is obtained in a complete and ethical manner is fundamental for the respect of rights and dignity of the human or animals participants involved in the project.

2.3.6. Privacy and surveillance

Considering that in the Senseback project a wireless communication (BLE) is used, the concepts of surveillance and privacy are fundamentals and should be treated. It is essential that, in the communication, security regarding the transmission of data is guaranteed. To

do that, all the necessary robust precautions, such as cryptography, should be taken, in order to protect the communications from possible non authorised interceptions. It is also important to focus on the nature of the transmitted data, only necessary information should be shared, and the communication should be limited to the finalities related to the project. Due to the sensitive nature of the biological or medical information, the privacy of the generated data should also be protected. To do that, measures to guarantee that data are treated in accordance with privacy-related laws and ethical standards becomes crucial. Moreover, collected data should be conserved and stored in a secure way, by limiting the access just to necessary cases and by implementing security protocols to avoid losses, manipulations and unauthorised accesses. Another precautions to guarantee the privacy of the patient is the anonymity of the data when possible, especially in those cases in which data are shared or aggregated. This would protect the identity of the patient involved in the project. Other than communication, also the device in usage should be securely designed. This includes protections against unauthorised access to the device itself and preventions against possible risks linked to the implant. In addition, during the informed agreement, human participants should be warned about data management and privacy. They should have a clear explanation about how their personal information will be treated and an explicit agreement about storage and usage of data should be furnished. In particular, it is important to follow the norms related to the privacy of the patient in the related geographical context, in order to guarantee that the project is ethically and legally acceptable. In EU privacy is regulated by the General Data Protection Regulation (GDPR) [**?**], whose publication has provided a clear regulation for personal data treatment. In the USA, privacy regulation is not uniform and it is present just in some states like in California with the California Consumer Privacy Act (CCPA) []california or in Virginia with the Virginia Consumer Data Protection Act (VCDP) [**?**]. Normative however, should be continuously monitored, since dynamics related to privacy can evolve and the adaptation to new normative or to the recent ethical worries is essential. To sum up, the concepts of surveillance and privacy regard the responsible management of sensitive information, while it is guaranteed that secure measurements are in line with the ethical and legal normative applicable. This is fundamental to respect the rights and dignity of the involved people and to maintain trust in the research.

2.4. Signal Acquisition

3 State of the Art

3.1. Introduction to Bluetooth Low Energy

Firstly, we need to define that "Bluetooth low energy is a brand new technology that has been designed as both a complementary technology to classic Bluetooth as well as the lowest possible power wireless technology that can be designed and built. Although it uses the Bluetooth brand and borrows a lot of technology from its parent, Bluetooth low energy should be considered a different technology, addressing different design goals". [?] ""chiara"" The main advantage of the BLE over the standard Bluetooth is that it keeps the radio off as much as possible when no data has to be sent [47]. It is optimized for low power consumption and short-range communication; thus, it enables devices to operate on a small energy budget, making Bluetooth technology is based on a master-slave concept, where one device acts as the master and controls one or more slave devices. The master initiates and maintains the communication, while the slave devices respond to the master's commands. For this there are four roles that a BLE device can implement. A device can be a peripheral (slave), it advertises their presence and respond to central devices' requests. A central (master), instead, scans for these advertising packets and it could, if the advertisement packet allows it, connect to that device. Moreover, there are other two roles possible: the broadcaster and the observer. A broadcaster sends out advertising packets without allowing any connections, while an observer discovers peripherals and broadcasters, but without the capability of accepting connections from a central. ""chiara""

3.2. Communication system

3.3. Previous code implementation

As show in the introduction, Senseback is an implanted device which communicates with an external unit through Bluetooth Low Energy. The starting point for the firmware code of this communication was previously developed [Chiara Master Thesis]. This master

thesis purposes improvement and additional features on the previous code, which presents some limitations.

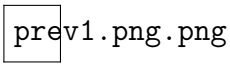


Figure 1: BLE communication between Senseback and external device The general architecture for the previous code is shown in Figure 2.

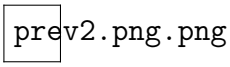


Figure 2: Previous implementation, setup of the experiment. As is it possible to see, two main devices facilitate communication: the transmitter and the receiver. For the experiment, two identical nRF52 Development Kits (DKs) [**?**] were used as the transmitter and receiver. These boards were thought to be employed just for testing, the final implementation, in fact, will replace the transmitter board with the Senseback device, and the external unit will serve as the receiver. Both boards were powered and controlled through the external unit, which was a PC connected via serial ports. External unit allowed the user to manage the communication by sending "START," "STOP," and "RESET" commands. This was done thanks to the usage of a Matlab script. These commands were transmitted through the serial port and received by the receiver using Universal Asynchronous Receiver/Transmitter (UART). UART is a common serial communication protocol used for connecting microcontrollers and other devices, often facilitated by USB-UART converters [**?**]. For both the transmitter and receiver, separate codes were implemented to manage communication. The first code set, designated as "Offline," was designed to test communication with a fixed number of packets. Specifically, 32,400 packets of one byte each were transmitted. The test began when the user entered the "START\n" command through MATLAB, giving the signal to the transmitter to start sending packets. This test ended when all packets were sent. The "Online" communication, on the other hand, simulated real-time behaviour with no predefined packet count. This test also begun once the user digited the "START\n" command, but it required also a "STOP\n" command to end. Lastly, the "RESET\n" command was just used in those cases in which it was necessary to restart the communication from the beginning. Upon receiving the appropriate command in MATLAB, the external unit would send it to the receiver via UART, which then relayed the command to the transmitter through BLE. The transmitter would start sending data to the receiver upon receiving the "START\n" command and stop when it received the "STOP\n" command or all packets had been sent, depending on the test

mode. After receiving the packets, the receiver printed the data to the serial port, allowing the external unit to read it through MATLAB. Last element of the previous setup is the sniffer. The purpose of this was to detect the packets exchanged from transmitter and receiver, including possible retransmissions during the communication. The chosen software to play this role is Wireshark (v.4.0.6) with the nRF Sniffer Tool (v.4.1.1), provided by Nordic Semiconductor©. All the codes used for this setup are deeply analysed below.

3.3.1. Matlab scripts

To achieve the two main objectives of communication control and data verification through a serial port, two separate MATLAB scripts were created. The first script is designed for offline testing, the second one for online. Here's how they work: Both scripts start by creating a vector that simulates the incoming ADC samples from a Senseback device. These samples are in uint16 format. Each value is then split into two uint8 values using the MATLAB function typecast, ensuring that the data is converted without losing information. The next step is to set up the serial port with a baud rate of 115200 bps, whose limit is imposed by the UART communication. After setting the baud rate, the serial port is flushed to ensure it's clear before communication begins. The scripts, instead, differ in the following ways: Commands: In the offline test script, only one command is required from the user, while the online test script requires two commands, the start and the stop of the communication. Data Reception: The online script does not include any data reception functionality due to memory limitations and the slower baud rate. In contrast, the offline script does receive data from the serial port. The online test script is simpler, focusing on generating the data and ensuring the serial port is set up properly. There is no data reception because of the limitations mentioned earlier. The offline test script uses a callback function to handle data reception. It uses the MATLAB configureCallback function to call the readData function each time new data is received from the selected serial port. The purpose of this function is to store the incoming data in a vector and keep track of the total data received. The callback function is turned off when the last two values are both 255 (indicating the end of a data sequence). Once all data is received, it can be converted back to uint16 using typecast and organized into a matrix for plotting and further analysis. The offline test script can then compare the received data with the data originally generated in MATLAB to ensure its accuracy and reliability.

3.3.2. Offline transmitter code

Figure 3 illustrates a flow chart of the offline code for the transmitter. Each block represents a distinct function. Below is a brief explanation of each one. Log init: This function initializes the log module. This module provides logging capabilities for the intended embedded application [**?**]. It has different usages such as debugging, which means to provide diagnosis for possible issues, info, which implies to check whether or not the application works as expected, warning, so indications about something unexpected which is happened, and error in those cases in which the software is not able to execute one of the functions. This function initializes the timers. For the offline code of the transmitter, two timers have been used. The first one is the timer responsible for data generation, and the second one is responsible for monitoring the total time required for data generation, data pre-processing, and sending to the central device. The two timers are both equally initialized, with the frequency set to 2 MHz for both.

The main difference between these two timers is given by the functions `nrfx_timer_us_to_ticks` and `nrfx_timer_ms_to_ticks` for the first and the second timer, respectively. These functions create an interrupt each time the timer reaches the specified value, which is 12.5 μs for the first one and 1 ms for the second. The value of 12.5 μs has been chosen to simulate the arrival of samples coming from the ADC in the actual application; in fact, 12.5 μs represents a frequency of data generation of 80 kHz.

This function is responsible for data generation. Firstly, it checks whether the transmission is active, i.e., whether the command `START` has been received from the central device. If it is active, a new data `CNT` is generated. Data generation is done as a step, so each new value is equal to the previous one plus one. `CNT` is a `uint16_t` variable. This choice is made because the ADC samples have 10 bits size, and the `uint16_t` variables are the smallest ones which guarantee that the samples are saved without wasting information, which would occur if a `uint8_t` variable were chosen.

Once the new `uint16_t` data is generated, it is put into the ring buffer `uartRx`. `Ringbuf` is a data structure that uses a single, fixed-size buffer as if it were connected end-to-end. This structure is useful for buffering data streams, where data needs to be overwritten as new data comes in, particularly when the buffer is full. It's commonly used in scenarios where the producer and consumer operate at different rates, as in this case. The size of `uartRx` has been set to 16384 `uint16_t` variables, representing the maximum size imposed by the RAM storage limits.

Once `CNT` has been put into the ring buffer, the function `data_arrived`, which is responsible for counting the number of data generated, is incremented by one. Then, it

is checked whether the new value generated is the first one using the boolean variable `first`. If this is still true, as initialized, it means that the new data generated is the first one, so `timer2` starts counting from a time elapsed equal to zero milliseconds. When the condition `first=true` is triggered, the boolean variable `first` is set to `false`, thus avoiding entering the previous condition in the next cycle and starting `timer2` again.

The last step for this function is to check whether the data generation process has arrived at the end. This is done by checking the value that the variable `data_arrived` has reached. If this value is greater than the value imposed by $\left(\frac{\texttt{DATA\_TO\_SEND}-2}{2}\right) - 1$, the condition is triggered. For the offline test, the `DATA_TO_SEND` variable is equal to 32400 and represents the number of bytes that should be sent to the central device to complete an offline test. Consequently, the condition is triggered when the variable `data_arrived` has reached a value of 16198, which also represents the number of samples arrived from the ADC. The choice for this condition triggering will be explained in detail when discussing the `spibuff` process function, which is responsible for data preprocessing before sending.

Once the condition is triggered, the `data_arrived`, `first`, and `CNT` values are reset, and the last 16-bit value is put into the ring buffer: 65535. This value represents the highest value in binary that can be represented using 16 bits. It is, in fact, a sequence of 16 bits all set to 1. Then, the boolean variable `transmission` is set to `false`, indicating that data generation is finished.

Timer2 handler: `Timer2` is the timer responsible for the computation of the total time elapsed during the data generation, data preprocessing, and data sending processes. This function increments the variable `total_time_elapsed_ms` by one each time the interrupt is triggered, so each millisecond. In this way, `total_time_elapsed_ms` provides a timestamp of the current total time elapsed in milliseconds.

Power management init: this function initializes the power management module, which handles power management features. [**?**] Ble stack init: This function is responsible for the initialization of the SoftDevice and BLE event interrupts. A SoftDevice is a precompiled and linked binary software implementing a wireless protocol developed by Nordic Semiconductor. [**?**] In this function, it is first requested to enable the SoftDevice using the `Nrf_sdh_enable_request` function. Once the SoftDevice has been enabled, it is configured through the function `Nrf_sdh_ble_default_cfg_set` by using the default settings specified in the SoftDevice handler BLE configuration. Finally, the functions `Nrf_sdh_ble_enable` and `Nrf_sdh_ble_observer` respectively enable the BLE stack and register a handler for BLE events.

Ble evt handler: This function manages the different BLE events. It first creates a pointer

to the specific GAP events, then a switch gives the instructions for each different event.

- **First case:** the device is connected. If this case is triggered, a log and the usage of LEDs, enabled through the function `bsp_indication_set`, indicate to the user that the connection has been correctly performed. The connection handle is then stored in the variable `m_conn_handle`. The connection handle is a value provided by the BLE stack that uniquely identifies a connection with another BLE device [**?**]. Then, the function `nrf_ble_qwr_conn_handle_assign` assigns a connection handle to the given instance of the Queued Writes module [**?**]. Ultimately, the data rate (PHY) is set to 2 Mbps for both the TX and RX BLE characteristics.
- **Second case:** the device is disconnected. In this case, it is logged to the user that a disconnection occurred, and the connection handle is set to invalid.
- **Third case:** PHY update request. In this case, the data rate is automatically changed to respond to a PHY update request.
- **Fourth case:** security parameters request. In this implementation, security features are not implemented, which means that the reply to the security parameters request is that pairing is not supported.
- **Fifth case:** system attribute access pending, such as bonding information [**?**]. Again, in this case, no system attribute is stored.
- **Next cases:** timeout events. If these are triggered, the connection is terminated.
- **Last case:** transmission has been completed. If this happens, the function `on_tx_complete` is called.

On Tx complete: This function manages the packet transmission queue mechanism. If the `BleTxBusy` variable is true, indicating that a packet has not been sent due to the transmission channel being busy, the variable is reset to false. Then, the `notification_send` function is called to attempt retransmission. This ensures that no packets are lost, as transmission attempts continue until successful completion.

Notification Send

This function oversees the data transmission process and calculates the transmitter's throughput. Initially, a `uint32_t` variable `err_code` is set to `NRF_SUCCESS`, indicating no errors. Before sending data to the receiver, the function checks two conditions:

- **Error Check**: It verifies that `err_code` remains `NRF_SUCCESS`, ensuring no trans-

mission errors have occurred.

- **Data Availability Check**: It ensures that the `uint16_t` variable `txRdPtr` (index tracking the last number of bytes sent) is less than `txSize` (total number of available data bytes). This ensures sufficient new data is pre-processed and ready for transmission.

If both conditions are satisfied, the length of the new packet to be sent, stored in the `length` variable, is updated to `txSize - txRdPtr`. This data of this length is then sent via the `ble_nus_data_send` function and `err_code` is updated accordingly. If `ble_nus_data_send` completes successfully without errors, `txRdPtr` is incremented by the size of `length`, and the `data_sent` variable, which tracks the total bytes sent to the central device, is also increased by `length`.

Next, the function checks if the transmission has reached the end by verifying if the last two bytes sent are both 255. Since 255 is the maximum value representable with 8 bits, this check ensures the last `uint16_t` variable in the ring buffer `uartRx` (set to 65535 in `timer1_handler` function) has been sent. If true, the total elapsed time is calculated by converting the timer ticks of `timer2` to milliseconds. The throughput is then calculated using the formula: throughput=number of bits senttotal time elapsed

$$\text{throughput} = \frac{\text{data\_sent} \times 8}{\text{total\_elapsed\_time}}$$

All relevant values, including the number of waits, are logged. If `ble_nus_data_send` function returns an error, indicating an unsuccessful transmission, `bleTxBusy` is set to true, and the wait counter is incremented. Finally, if `txRdPtr` is greater than or equal to `txSize`, the `txActive` variable is set to false, indicating that another packet can be pre-processed. Gatt init: This function initializes the Generic Attribute Profile (GATT) library. It also sets the Attribute Protocol (ATT) Maximum Transmission Unit (MTU). This parameter defines the maximum amount of data that can be exchanged by a single BLE packet, specifically for the peripheral side. For this application, the MTU has been set to `NRF_SDH_BLE_GATT_MAX_MTU_SIZE`, which corresponds to a value of 247 bytes, the maximum achievable with BLE version 5.2. A high value for `GATT_MTU` generally implies higher efficiency for data exchange and consequently, a higher throughput. In fact, with a single exchange, it is possible to send a larger amount of data, minimizing the wasted time due to latency.

GATT Event Handler

This function handles events coming from the GATT library. It first checks whether the event triggering is caused by the current connection by looking at the connection handle stored in the variable `m_conn_handle`. In this implementation, the condition will always be triggered since there is only one connection, which is point-to-point. It also checks, together with the previous condition, whether the event that occurred is an update of the GATT ATT MTU. If both these conditions are satisfied, then the `m_ble_nus_max_data_len` variable is updated to the effective ATT Payload for the connection, which is equal to ATT Data minus the two headers shown in Figure 4, specifically, the Op Code and the Attribute Handle.

Once this value is updated, the desired values for ATT MTU of the central and peripheral devices are logged to allow the user to check whether they are the same, ensuring that both devices agree on the maximum achievable value.

Services Initialization

This function is used to initialize all the services which will be used in the application. For the intended purposes, the services that are initialized are two: Queued Write Module (qwr) and NUS module. The qwr module handles prepare write, execute write, and cancel write commands. It also manages memory requests related to these operations (link). The NUS module is a custom Bluetooth Low Energy (BLE) service designed to emulate a UART (Universal Asynchronous Receiver/Transmitter) over BLE. This allows devices to communicate wirelessly in a manner similar to serial communication over a wired connection (link).

NUS Data Handler

NRF QWR Error Handler

This function handles Queued Write Module (qwr) errors. A pointer to this function will be passed to each service which may need to inform the application about an error.

Advertising init: This function is used to initialize the advertising functionality of the peripheral device. When a BLE device broadcasts its presence, it sends out advertising packets that can include various types of information, such as device name, service UUIDs, manufacturer-specific data, and appearance. For the current application, the advertising

packets include the device's full name but not the appearance data. Appearance data is a predefined value that indicates the type or category of the device. This value is used to give a general idea of the device's functionality without requiring a full connection. It helps other BLE devices and applications quickly identify what kind of device they are encountering. The choice of not including the appearance value in the advertising packets could be made both for privacy reasons and for saving space for other information. Another included parameter in the advertising packets is the advertising flag field, which in this case is set to `BLE_GAP_ADV_FLAGS_LE_ONLY_LIMITED_DISC_MODE`, meaning that the device only supports BLE and not classic Bluetooth, and that the device will be in a discoverable state for a limited time.

Furthermore, UUIDs should be defined. A UUID is a unique identifier for services offered by the BLE device. The function `advertising_init` first computes the number of UUIDs that are present in the `m_adv_uuids` array, then it sets `uuid_cnt` to that value. Then, `init.srdata.uuids_complete.p_uuids` sets the pointer to the array of UUIDs, which tells the program where to find the list of UUIDs that describe the services the BLE device offers. Finally, the advertising settings are set. In particular, fast advertising mode is enabled, meaning that the device will advertise itself frequently, making it quicker to be discovered.

The last two parameters for advertising are set as follows:

- `APP_ADV_INTERVAL`, which defines how often the device will send out advertising packets when in fast advertising mode. In this case, the `APP_ADV_INTERVAL` parameter is set to 64 in units of 0.625 ms, which corresponds to 40 ms.
- `APP_ADV_DURATION`, which defines how long the device will stay in fast advertising mode before stopping or switching to another mode. In this case, this value is equal to 18000 in units of 10 ms, which corresponds to 180 s.

Ultimately, the event handler function for advertising events is set, which will handle various events related to advertising, such as starting, stopping, or timing out. Once this configuration has been completed, the advertising module is initialized using the function `ble_advertising_init`, and `ble_advertising_conn_cfg_tag_set` sets the connection configuration settings tag for the advertising instance.

On adv evt: This function is responsible for handling the advertising events. Two possible cases are considered:

- **Fast Advertising Event:** If this event occurs, LED1 starts blinking to signal that the device is in the advertising phase.

- **Advertising Idle Event:** If this event occurs, the `sleep_mode_enter` function is called.

Sleep Mode Enter

This function puts the chip into sleep mode to save battery when the device is not in use. To signal the state of the board, the behavior of the LEDs is changed. Specifically, for the idle case, all LEDs are turned off. The `bsp_btn_ble_sleep_mode_prepare` function is used to prepare the wakeup buttons before going into sleep mode [**?**]. Lastly, the system is put into sleep mode using the `sd_power_system_off` function.

Connection Parameters Initialization

This function initializes the connection parameters module that will be used when establishing and maintaining a connection with a central device. The pointer to the initial connection parameters is set to NULL, meaning no initial parameters are provided. The delay before the first update of the connection parameters and the following parameters update delay are set to 5000 ms and 30000 ms respectively. These parameters describe the delay before the first update of the connection parameters is attempted and the delay between subsequent attempts if the first attempt fails. The maximum number of attempts to update the connection parameters before giving up the connection parameter negotiation is set to three. If the connection parameter update procedure fails, disconnection does not happen since `cp_init.disconnect_on_fail` is set to false. The `cp_init.start_on_notify_cccd_handle` sets the handle of the Client Characteristic Configuration Descriptor (CCCD), which triggers the connection parameter update procedure. `BLE_GATT_HANDLE_INVALID` indicates that the connection parameter update procedure is not started by any specific CCCD. The functions `on_conn_params_evt` and `conn_params_error_handle` are assigned to manage possible changes or updates to connection parameters and handle errors during the connection parameters update process. All the settings chosen inside the `conn_params_init` function become effective through the `ble_conn_params_init` function.

On Connection Parameters Event

This function manages possible changes or updates to connection parameters. If a connection parameters event fails, the devices are disconnected.

Connection Parameters Error Handler

This function handles possible errors coming from the Connection Parameters module. The `uint32_t` variable `nrf_error` contains information about what went wrong.

TX Power Set

This function sets the transmission power level for a Bluetooth Low Energy (BLE) advertising role. Specifically, `BLE_GAP_TX_POWER_ROLE_ADV` specifies the role for which the transmission power is being set, in this case, the role is advertising. The variables used to set transmission power are `m_advertising.adv_handle`, which holds the handle of the advertising set, and `TX_POWER_LEVEL`, a predefined constant or variable representing the transmission power level in dBm. For this implementation, this value is set to 0.

Advertising Start

This function starts the advertising process through the `ble_advertising_start` function, setting the advertising mode to fast.

Idle State Handle

This function handles the idle state of the embedded system. It first calls `spiBuff_process`, responsible for data preprocessing. Then, it calls the `NRF_LOG_PROCESS` macro and uses the `UNUSED_RETURN_VALUE` to ignore its return value. Finally, it calls the `nrf_pwr_mgmt_run` function, which handles power management tasks, such as putting the microcontroller into a low-power state if there is nothing else to do, ensuring the system conserves power when idle.

SPI Buffer Process

This function preprocesses data coming from the `timer1_handler` function. It checks that transmission is not active through the boolean `txActive`. If transmission is not active, it starts the process. The `ringbuf_elements` function returns the number of `uint16_t` data still present in the `uartRx` ring buffer, stored in the `queueLength` variable. If `queueLength` is at least half of the `SENSEBACK_MTU` divided by two (at least 122 bytes), a full-size BLE packet can be prepared since there are enough data to fill it. Data from the Senseback ADC are 10 bits in size and are stored in `uint16_t` variables. Each value from the ring

buffer is converted into bytes for sending. A cycle executed 122 times prepares each BLE packet. Each cycle iteration gets a new value from the ring buffer using `ringbuf_get`, saved into a temporary `uint16_t` variable `element`. Two positions in the `nusTx` array, which will contain the final data sent to the central device, are filled. Specifically, position `i*2` (where `i` is the cycle index) is filled with the 8 most significant bits of `element`, while position `i*2+1` is filled with the 8 least significant bits of `element`. This ensures no information is wasted, but the inefficiency and memory waste are generated since 6 bits of each 16-bit variable are empty. Once a packet is fully prepared, `txActive` is set to true, enabling data sending to the central device. The `txSize` value is set to `SENSEBACK_MTU`, representing the prepared packet size, and `txRdPtr` is reset to 0. Once this process is completed, the `notification_send` function can be called to send processed data. If there are elements in the ring buffer but not enough to fill a BLE packet, the last packet of the communication is reached. Data are preprocessed as before, but the cycle completes `queueLength` times, and `txSize` is set to `queueLength*2`.

Offline Receiver Code

Some functions in the receiver code are common with those in the transmitter, such as the `log_init` function, used in both receiver and transmitter in the same way. The `timer_init` function is also similar, differing in the number of timers initialized (one in the receiver). The receiver's timer computes the time elapsed in the data receiving process, equivalent to the transmitter's timer2. Functions `power_management_init` and `ble_stack_init` are identical for both. Some functions are similar but differ due to the different central/peripheral roles, particularly:

- **GATT Init:** `nrf_ble_gatt_att_mtu_periph_set` is replaced by `nrf_ble_gatt_att_mtu_central_`
- **GATT Event Handler:** The connection handle check is not present in the receiver code since the peripheral can be connected to more than one central device.

Other functions are different or not present in the transmitter code and are explained below.

UART Initialization

This function initializes the UART, which lets the central device communicate with the external unit. First, the communication parameters are defined, setting the pin number for the UART receive (RX), UART transmit (TX), UART Ready To Send (RTS), and UART Clear To Send (CTS) lines. In this case, they are 8 and 6 for RX and TX, and

5 and 7 for RTS and CTS, respectively. Hardware flow control is disabled through the `APP_UART_FLOW_CONTROL_DISABLED` macro, and parity checking is disabled, meaning no parity bit will be used for error checking. The baud rate is set to 115200 bits per second. Finally, the `APP_UART_FIFO_INIT` function makes the parameters effective. This setup ensures the UART is correctly configured and ready to handle data transmission and reception with FIFO buffers and appropriate event handling.

UART event handler: This function is responsible for receiving commands from the external unit. Within this function, a switch statement manages different cases. The first case handles the reception of new data through UART. Commands from the external unit are received as sequences of characters in the UART_event_handler function. Upon receiving a new character via `app_uart_get`, it is stored in `data_array` at position `index`, which is then incremented. Once a character is saved, it checks if the character is equal to '\n' and if `index` exceeds `m_ble_nus_max_data_len`. If true, it indicates the command is fully received, and the string is sent to the peripheral device via the function `ble_nus_c_string_send`. `m_ble_nus_max_data_len` represents the maximum data length in bytes that can be transmitted by the Nordic UART service module, specifically BLE_GATT_ATT_MTU_DEFAULT minus the two headers OP Code and Handle Length, which is 23 minus 3 bytes. After sending the string, `index` is reset to zero. The other cases handled in this function are `APP_UART_COMMUNICATION_ERROR` and `APP_UART_FIFO_ERROR`, which report communication and FIFO module errors to the user through log messages.

Buttons LEDs Initialization

This function initializes buttons and LEDs using two functions: `bsp_init` for LEDs and `bsp_btn_ble_init` for buttons.

BSP Event Handler

Handles events from the Board Support Package (BSP) module using a switch-case statement. The sleep event calls `nrf_pwr_mgmt_shutdown` with `NRF_PWR_MGMT_SHUTDOWN_GOTO_SYSOFF` to put the system into a low-power system-off state. The disconnect event calls `sd_ble_gap_disconne` to terminate the Bluetooth connection, providing the connection handle and reason for disconnection.

Database Discovery Initialization

Initializes the Database Discovery (DB) module using `ble_db_discovery_init`, which handles service discovery on GATT servers.

DB Discovery Handler

A callback function to handle events from the database discovery module, forwarding events to respective services based on discovered UUIDs.

BLE Event Handler

Handles BLE events with differences between transmitter and central code. For connection events, changes LED behavior and assigns the connection handle using `ble_nus_c_handles_assign`. Additional tasks include starting service discovery with `ble_db_discovery_start`. PHY settings are not set in central code as it responds to PHY update requests from the peripheral. Disconnect events log disconnection without changing the connection handle. Common cases include security parameter requests, PHY update requests, and GATTC/GATTS event timeouts. Unique cases not present in the transmitter's `ble_evt_handler` are BLE_GAP_EVT_TIMEOUT for GAP operation timeouts and BLE_GAP_EVT_CONN_PARAM_ for peripheral-initiated connection parameter updates.

NUS Central Initialization

Initializes the Nordic UART Service (NUS) for the central role using specific functions.

BLE NUS Central Event Handler

Handles data reception from the peripheral through NUS. Uses a switch statement to manage cases:

- Completes NUS discovery, assigns connection handle, and enables notifications for data reception.
- Manages arrival of data packets from peripheral. Increments `received_bytes` and computes throughput based on the time elapsed since the first packet.
- Saves packet data into `senseback_buff`. Computes throughput when the last two bytes of a packet are 255.

- Updates `senseback_buff_length` and checks for buffer overwrite.
- Handles disconnection events by calling `scan_start` to initiate a new connection.

Throughput is computed using the formula: $\text{throughput} = \frac{\text{number of bits received from peripheral}}{\text{time elapsed in data receiving process}}$. This throughput computation differs from that in the transmitter code by focusing solely on data receiving time, making it more reliable for communication speed analysis.

Scan start: This function initiates the scan process by performing two actions. Firstly, it starts the board scanning using `nrf_ble_scan_start`. Secondly, it sets the LEDs to indicate that the board is in the scanning phase, with `LED1` specifically configured to start blinking.

Scan Init

This function serves two main purposes: setting scanning parameters and enabling scan filters. It sets two scanning parameters:

- `connect_if_match` is set to `true`, allowing the scanning process to automatically attempt to establish a connection.
- `conn_cfg_tag` is set to `APP_BLE_CONN_CFG_TAG`, which refers to the BLE stack configuration tag. In this case, the value is set to 1, indicating the configuration with handle one.

After initializing BLE scanning, filters are configured and enabled. The scanning module filter type is `SCAN_UUID_FILTER`, ensuring that only advertisements from devices offering the NUS service are reported.

Scan Event Handler

This function handles events from the Scanning Module, addressing three specific events:

- An error occurring during the connection process retrieves the error code from the event parameters.
- Successful establishment of a connection retrieves the connected event parameters and logs the address of the connected peer.
- Timeout event during the scan process logs a message indicating the scan has timed out and calls `scan_start` to restart the scanning process.

BLE NUS Chars Received Parameters Print

This function is continuously called in the main program until there is no data overwrite in the circular buffer `senseback_buff`. It handles sending data to the external unit via UART. It takes as inputs each newly stored row in `senseback_buff` and the length of the new packet. Data are sent via UART using `app_uart_put` until all data are sent or a memory resource error occurs. Error checking mechanisms are implemented to notify the user of the position in the packet where an error occurred. Additionally, the function allows for flagging `ECHOBACK_BLE_UART_DATA`. In this case, data sent via UART to the external unit are also sent back to the peripheral, providing an additional transmission verification.

Idle State Handle

This function

3.3.3. Online transmitter code

Figure 6: flow chart implant code online - previous implementation The online transmitter code is quite similar to the offline one, it just presents some small differences to guarantee the previously described different intended behaviour. The flow chart of this code is shown in Figure 6. The differences between the two codes inside the functions will be underlined below. Timer1 handler: This function remains responsible for data generation but differs from the offline case in several aspects. Firstly, although the generated `uint16_t` value `CNT` still increments step-wise, an upper limit of 65530 is set. Upon reaching this value, `CNT` resets and data generation resumes from zero. In offline tests, `CNT` would not exceed 16198 to communicate 32400 bytes, whereas online tests allow `CNT` to potentially grow endlessly. Secondly, in the online code, `timer1_handler` is not responsible for ending communication; instead, transmission concludes upon receiving the STOP command from the user.

Notification Send

NUS Data Handler

This function manages reception of commands from the external unit, particularly handling the STOP command differently. Upon receiving this command, `timer1` is disabled

to halt the data transmission process. Additionally, the value 65535 is placed into the ring buffer, and `transmission` is set to false. These steps are placed differently from the offline code to accommodate behavior variations between tests. In offline tests, these actions were inside `timer1_handler` to end communication upon data generation completion. In online tests, communication ends only upon user command.

SpiBuff Process

This function processes data similarly to the offline case until reaching the last packet. In the online code, it does not check if the remaining number of data to be sent is less than 244. Instead, it verifies if there is still data in the ring buffer and if transmission is false, indicating receipt of the STOP command. If true, it processes the final stop as before.

Additional Functions

Two functions added to the online receiver code that are not in the offline version:

- `buttons_leds_init` and `uart_init`, previously explained in the offline receiver code section. These are not strictly necessary for the online transmitter code's purposes.

Online Receiver Code

For the online receiver code, the flow chart mirrors Figure 5. However, some functions have minor differences to accommodate the online test's different behavior:

- `Ble nus c evt handler`: Similar to the offline case, with an additional condition to check the end of transmission. Besides checking the last two values being 255, it verifies `end_transmission` is true. This ensures the receiver recognizes the last two 255 values after the user enters the STOP command, concluding data reception.

Sniffer Code

The sniffer provides real-time monitoring of transmission parameters such as PHY, ATT MTU, and protocol, and also tracks the number of retransmissions during tests. To utilize it:

- A software tool is needed, available for the nRF52840 dongle when in debug mode.

Specifically, `sniffer_nrf52840dongle_nrf52840_4.1.1.hex` is used.

- Cross-platform development software like nRF Connect for Desktop is required, with the "Programmer" app used to execute the code on the sniffer.
- Once the sniffer is connected to the PC in debug mode, the code can be executed, enabling it to function with Wireshark for analysis.

3.4. Communication speed

3.5. ENGNet: advantages and shortcomings

Given the structure of the project there is the need of a fast, computationally efficient model that can boast good performance on classification of ENG signals. In particular we need this model to classify the electroneurography signal sent from the board to the computer in different classes that correspond to different movements. Because of this, as we said, there is a strong need for both speed of classification, since we don't want excessive delay between the firing of the neurons and the transmission to the area of stimulation, but we also need accuracy since a movement different from the one that has been transmitted by the firing of the neurons would compromise the result of the project as a whole.

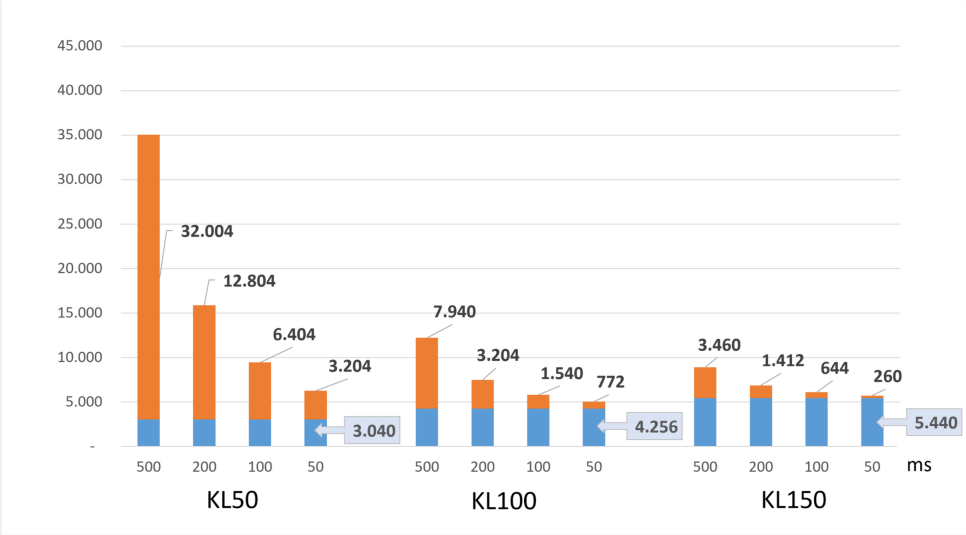
A first promising model called EEGNet was introduced by [**?**] for the classification of EEG singals, it presents a tailored variant of CNNs aimed at Brain-Computer Interface (BCI) applications. Its primary advantage lies in its compact architecture, boasting fewer parameters compared to traditional CNNs. This efficiency renders EEGNet well-suited for the processing of EEG signals, facilitating improved performance in BCI tasks while simultaneously reducing computational complexity. [tesi di davide] Starting from this model previous work by Davide Gottardello in the PRelay project developed a newer model in order to classify ENG Data called ENGNet. In this section we will describe this model and outline the most relevant features along with the advantages and shortcomings. As we will show in Chapter 4, ENGNet was the model we used to verify the performance of our channel selection algorithms and as such, understanding its functioning and performance was a key aspect of our work. In particular, at the beginning of our work of research, we spent some time setting up on our machines the same testing conditions such as the same dataset and code, this not only allowed us to gain familiarity with the architecture of the model and previous code but also gave us the possibility of verifying all previous results and benchmarks in order to acquire data to compare our own work

to.

3.5.1. ENGNet architecture

The principal parameter to modify in EEGNet architecture to create an ENGNet is the kernel length, which plays a crucial role in shaping the network's feature extraction capabilities. [Davide] A kernel length of 100 samples, which translates to a sequence of 20 ms at a 5 kHz sampling rate, could be a compromise between kernel length and the ability to capture ENG signal characteristics effectively. Hence, kernel lengths of 10, 20, and 30 ms are investigated in this study which correspond to different values for the kernel length (denoted as kernLength) which means 50, 100 and 150 samples [Davide]. "re write " All the layers after the first one are therefore scaled to maintain the proportions proposed in the EEGNet model. [Davide 27]

The architecture involves two convolutional steps, employing 2D convolutional filters to capture EEG signals at different band-pass frequencies and depthwise convolutions to learn spatial filters for each temporal filter. The approach also utilizes separable convolutions to reduce the number of parameters and efficiently combine feature maps. In the classification block, softmax classification is directly applied to the extracted features without a dense layer, reducing the number of parameters in the model.



The number of parameters in a neural network is strictly related to the performance metrics of the network. Given too many parameters a model will inevitably overfit to the training test and will therefore poorly generalize on the validation and test portion of the data. Conversely, too few parameters will make the model unable to learn effectively

the classification required for the task at hand. There is plenty of tools to overcome these kind of issues when training model, given that is one of the most prominent issues when dealing with neural networks performances such as tweaking the learning rate or adding dropouts layers in the architecture of the model in order to discard some of the parameters. Clearly, reducing the number of parameters not only reduces the chances of overfitting but also decreases training and inference time. Let us now delve deeper into the actual architecture. The configuration for block 1 is as follows:

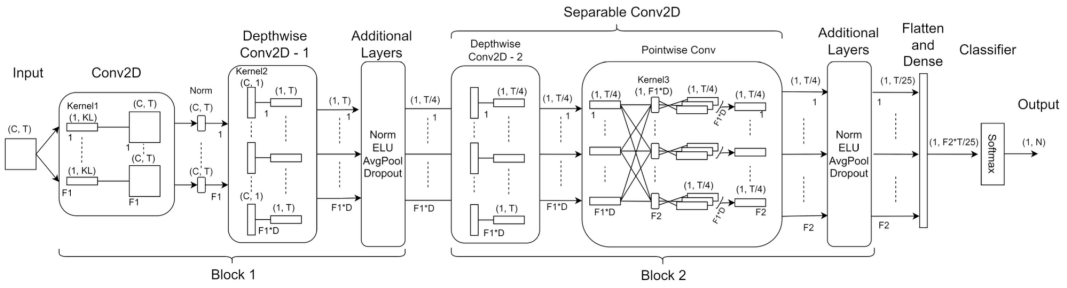
- Conv2D layer: Conv2D(, (1, kernLength), input shape=(Chans, Samples, 1))
- AveragePooling2D layer: AveragePooling2D((1, kernLength//10))

Here, F1 represents the number of 2D convolutional filters (F1 = 16), Chans is the number of channels in the EEG signals (Chans = 16), and Samples is the length of the ENG samples. The Conv2D layer uses the kernel length to create a 2D convolutional filter with height 1 and width equal to kernLength. The AveragePooling2D layer reduces the spatial dimensions of the feature maps by performing average pooling with a pool size of (1, kernLength//10).

For block 2, the configuration is as follows:

- SeparableConv2D layer: SeparableConv2D(F2, (1, kernLength//4))
- AveragePooling2D layer: AveragePooling2D((1, kernLength//25))

In block 2, the SeparableConv2D layer is used with a kernel size of (1, kernLength//4) and F2 = 32 to conduct depthwise separable convolution. This layer effectively reduces the number of parameters while preserving feature extraction capabilities. The parameters F1 and F2 are doubled compared to what was explained in chapter 3.5.1 and in [27] due to the larger frequency and consequently greater number of samples, while D remains equal to 2. This adjustment has been validated as a superior solution in preliminary tests.



ENGNet layers

We will now briefly explain the main layers present in the ENGNet model to outline what each layer does.

Block	Layer	# filters	Size	# params	Output	Activation	Options
1	Input				(C, T, 1)		
	Conv2D	F_1	(1, K_L)	K_L*F_1	(C, T, F_1)	Linear	Mode = same
	BatchNorm			4*F_1	(C, T, F_1)		
	DepthwiseConv2D	D*F_1	(C, 1)	C*D*F_1	(1, T, D*F_1)	Linear	Mode = valid, depth = D, max norm = 1
	BatchNorm			4*D*F_1	(1, T, D*F_1)		
	Activation				(1, T, D*F_1)	ELU	α =1
	AveragePool2D		(1, K_L/10)		(1, T/4, D*F_1)		
	Dropout				(1, T/4, D*F_1)		p = 0.2
2	SeparableConv2D	F_2	(1, K_L/4)	K_L/4*D*F_1+F_2*D*F_1	(1, T/4, F_2)	Linear	Mode = same
	BatchNorm			4*F_2	(1, T/4, F_2)		
	Activation				(1, T/4, F_2)	ELU	α =1
	AveragePool2D		(1, K_L/25)		(1, T/25, F_2)		
	Dropout				(1, T/25, F_2)		p = 0.5
	Flatten				(F_2*T/25)		
Classifier	Dense			N*(F_2*T/25)	N	Softmax	

3.5.2. Classification performance metrics

Accuracy This measures the proportion of correctly classified examples in the dataset. It is calculated as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \quad (3.21) \qquad (3.1)$$

F1-Score It combines precision and recall to offer a balanced evaluation of a model's performance. Precision (or Positive Predicted Value (PPV)) is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall (or sensitivity, or True Positive Rate (TPR)), is the ratio of correctly predicted positive observations to the total actual positive observations. F1-Score is computed as:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.22) \qquad (3.2)$$

where:

$$\text{Precision (PPV)} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (3.23) \qquad (3.3)$$

$$\text{Recall (TPR)} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (3.24) \qquad (3.4)$$

F1-Score combines precision and recall exploiting the harmonic mean to penalize the extreme values. Both Accuracy and the F1-score are important for a classification task:

- Accuracy is preferred when correctly identifying True Positives and True Negatives is the primary concern.
- F1-score is favored when minimizing the impact of False Negatives and False Positives is crucial.

Moreover, when dealing with imbalanced class distributions, where some classes have significantly fewer instances than others, the F1-score tends to be a more suitable metric for model evaluation.

Confusion Matrix This matrix visualizes a model's performance by showing the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). A general construction of confusion matrix is given in Table 3.1. It is particularly helpful for understanding the model's ability to correctly classify each class and to investigate the errors in predict phase.

Table 3.1: Confusion Matrix for two classes (Positive and Negative) or outcomes to evaluate. The word "Predicted" refers to the output label from the classifier, while the "Actual" label is the one associated to the data.

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

In a confusion matrix with multiple classes, it is possible to compute precision, recall, and the F1-score for each individual class. Precision for a class is calculated as the ratio of true positives for that class to the total instances predicted as that class (both true positive and false positives). It is calculated as the ratio of true positives to the sum of true positives and false positives for that class.

$$\text{Precision (PPV)} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (3.23) \qquad (3.5)$$

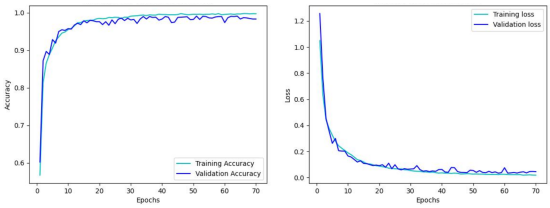
$$\text{Recall (TPR)} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (3.24) \qquad (3.6)$$

The F1-score is the harmonic of precision and recall. It provides a balanced measure that considers both false positives and false negatives. It is calculated using the following formula:

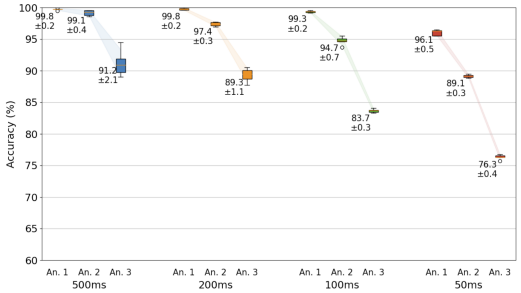
$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.22) \tag{3.7}$$

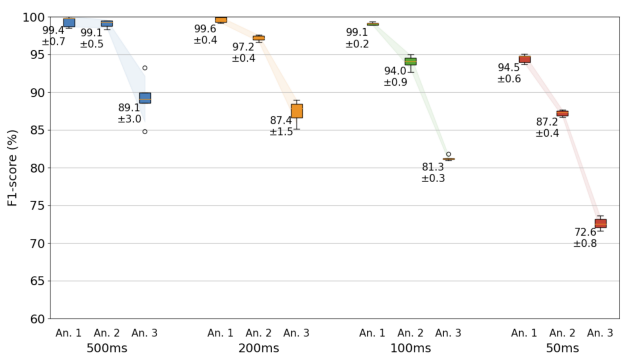
By calculating these metrics for each class, one gains insights into the model's performance for individual categories, allowing a more nuanced evaluation of classification effectiveness.

3.5.3. ENGNet results



Analysis of weights in the first convolutional layer indicate the network's capability to recognize and accentuate characteristic ENG signal frequencies, and these are relevant to improves classification performance. However, the network's ability to identify these frequencies depends on the quality of the signal used during training. The dependence of the power spectrum analysis on the data suggests the importance of carrying out specific training per subject, given that even within data taken from the same animal there is a variation in the frequencies emphasized by ENGNet,





Talk about limitation

Comparison between validation and test re- sults showed that ENGNet can generalize well to new data. This can be further improved by increasing the dropout probability

'foto dei results'

3.6. nrf5280 Board pros and cons

3.7. Power Supplier

3.8. Memory management

The memory architecture of the Nordic board is intricately designed to optimize performance and flexibility. The system comprises various memory regions and buses interconnected to facilitate efficient data transfer and processing. Below is a detailed explanation of each component illustrated in the provided diagram:

1. Central Processing Unit (CPU) ARM Cortex-M4: The CPU is based on the ARM Cortex-M4 architecture, which is a highly efficient processor designed for high-performance, low-power applications. It includes a rich set of features such as floating-point unit (FPU), digital signal processing (DSP) extensions, and low-latency interrupt handling. 2. Memory Organization The memory is divided into several sections, each serving a specific purpose and connected via different buses to optimize performance.

RAM Sections (AHB Slaves) The system contains multiple RAM sections, each interfaced through the Advanced High-performance Bus (AHB). These sections are labeled from RAM0 to RAM8, with each section further divided into smaller subsections. The sections are organized as follows:

RAM0 to RAM8: These RAM sections are accessible via the System bus, DCODE, and ICODE buses, providing flexible access paths for different types of data operations. Each RAM section is split into two subsections (Section 0 and Section 1), and their memory addresses are specified for both data RAM (System) and code RAM (ICODE/DCODE).

Memory Addresses: The RAM sections have designated address ranges:

Flash Memory Flash memory is used for non-volatile storage, providing persistent data retention even when the system is powered down. The flash memory is divided into pages and accessed via the ICODE and DCODE buses.

Page Addresses: Specific addresses for pages include: Subsequent pages follow this addressing pattern up to Page 255. 3. Bus Architecture The system utilizes a multilayer AHB interconnect to manage data flow between different components efficiently.

Advanced High-performance Bus (AHB) The AHB multilayer interconnect ensures high-speed data transfer between the CPU, RAM sections, flash memory, and peripherals. It allows multiple masters (such as the CPU and DMA controllers) to access shared resources without significant performance bottlenecks.

System Bus, DCODE, and ICODE These buses provide specific pathways for different types of data access:

System Bus: Used for general data transfers. DCODE Bus: Dedicated to instruction fetches from flash memory. ICODE Bus: Used for instruction fetches from both RAM and flash memory, enabling efficient code execution. 4. Peripherals and EasyDMA Peripherals: The system includes various peripherals connected via the AHB2APB bridge, allowing integration with different peripheral devices. EasyDMA: Direct Memory Access (DMA) controllers facilitate direct data transfers between peripherals and memory, bypassing the CPU to enhance efficiency. EasyDMA controllers are connected via DMA buses to the peripheral modules. 5. Address Mapping The memory map clearly defines the address ranges for each memory section and peripheral, ensuring well-organized access to system resources. RAM Sections: Each RAM section has a specific address range, both for system data and code execution, ensuring organized and conflict-free memory access. 6. Interconnect and Bus Matrix The AHB multilayer interconnect/bus matrix allows concurrent access to different memory sections and peripherals, reducing latency and improving overall system throughput.

The Nordic board's memory architecture is a sophisticated system designed for efficient data handling and high performance. By leveraging multiple RAM sections, a well-organized flash memory layout, and an advanced bus interconnect system, the architecture supports the demanding requirements of modern embedded applications. The integration of DMA controllers further enhances data transfer efficiency, making the Nordic board suitable for a wide range of applications requiring reliable and high-speed memory access.

3.9. Introduction

The memory architecture of the Nordic board is intricately designed to optimize performance and flexibility. The system comprises various memory regions and buses interconnected to facilitate efficient data transfer and processing. This document provides a

detailed analysis suitable for a thesis.

3.10. Overview of the Address Map

The address map is divided into two main sections: the System Address Map and the Address Map for peripherals. The System Address Map shows the hierarchical arrangement of different memory regions and device spaces, while the Address Map specifies the address ranges for peripherals and other key components.

3.11. System Address Map

3.11.1. Device Space (0xE0000000 - 0xFFFFFFFF)

This region is reserved for device-specific addresses, typically used for system control and private peripheral bus (PPB) mappings. The PPB includes critical control registers for system-level operations such as interrupts and debugging.

3.11.2. RAM (0x80000000 - 0xBFFFFFFF)

This region is designated for RAM, divided into two main blocks:

- **0x80000000 - 0x9FFFFFFF**: General RAM space.
- **0xA0000000 - 0xBFFFFFFF**: Additional RAM or special-purpose RAM.

3.11.3. Peripheral Space (0x60000000 - 0x7FFFFFFF)

This area is allocated for peripheral devices, allowing the CPU to interact with various hardware modules via memory-mapped I/O.

3.11.4. SRAM (0x40000000 - 0x5FFFFFFF)

This region is dedicated to SRAM, providing fast, volatile memory storage for critical data that requires quick access times.

3.11.5. Code Space (0x00000000 - 0x1FFFFFFF)

This segment includes different types of memory used for storing executable code:

- **0x00000000 - 0x007FFFFF**: Flash memory for non-volatile code storage.

- **0x00800000 - 0x00FFFFFF**: Code RAM for executing code from RAM.
- **0x10000000 - 0x10000FFF**: FICR (Factory Information Configuration Registers) containing device-specific configuration data.
- **0x12000000 - 0x12000FFF**: UICR (User Information Configuration Registers) for user-specific configuration.
- **0x19FFFFFF - 0x20000000**: XIP (Execute In Place) memory, allowing execution directly from external flash.

3.12. Peripheral Address Map

3.12.1. Private Peripheral Bus (0xE0000000)

The PPB includes system control space (SCS), NVIC (Nested Vectored Interrupt Controller), and other essential control registers.

3.12.2. AHB Peripherals (0x50000000)

This range is used for high-performance peripherals connected via the Advanced High-performance Bus (AHB), providing efficient data transfer capabilities.

3.12.3. APB Peripherals (0x40000000)

The Advanced Peripheral Bus (APB) space is designated for slower peripherals that do not require the high throughput provided by the AHB.

3.13. Detailed Address Ranges and Usage

3.13.1. Flash Memory (0x00000000 - 0x007FFFFF)

Used for storing the firmware and application code. Flash memory is non-volatile, retaining data even when power is off.

3.13.2. Code RAM (0x00800000 - 0x00FFFFFF)

Provides a space for loading and executing code from RAM, useful for applications requiring high-speed code execution.

3.13.3. FICR (0x10000000 - 0x10000FFF)

Contains factory-programmed data specific to the device, such as calibration values and unique identifiers.

3.13.4. UICR (0x12000000 - 0x12000FFF)

Stores user-programmable configuration data, which can include bootloader settings and memory protection configurations.

3.13.5. XIP (0x19FFFFFF - 0x20000000)

Allows the system to execute code directly from external flash memory, saving internal RAM for other uses.

3.14. Summary

The Nordic board's memory architecture is designed to efficiently manage various types of memory and peripheral access. The system address map and peripheral address map clearly define regions for devices, RAM, peripherals, and code execution. This structured organization allows for optimized performance, flexibility in code execution, and efficient interaction with peripherals.

By understanding this memory layout, developers can make informed decisions about memory usage, peripheral integration, and system optimization, ultimately enhancing the performance and functionality of applications running on the Nordic board.

Based on the detailed analysis of the Nordic board's memory architecture, several key observations can be made to optimize code and effectively manage memory. These recommendations will help developers enhance the performance and efficiency of their applications.

3.15. Observations and Recommendations

3.15.1. Utilizing Flash Memory for Code Storage

Observation:

- Flash memory (0x00000000 - 0x007FFFFF) is non-volatile and ideal for storing the firmware and application code.

Recommendation:

- Store all static code and infrequently updated data in flash memory to ensure data persistence across reboots and power cycles.
- Optimize the code to minimize the frequency of flash writes, as flash memory has a limited number of write cycles.

3.15.2. Efficient Use of Code RAM

Observation:

- Code RAM (0x00800000 - 0x00FFFFFF) allows for high-speed code execution.

Recommendation:

- Load performance-critical code segments and frequently accessed functions into Code RAM to benefit from faster execution times.
- Use this space judiciously due to its limited size, prioritizing code that requires low-latency access.

3.15.3. Managing Volatile Data with SRAM

Observation:

- SRAM (0x40000000 - 0x5FFFFFFF) provides fast, volatile memory for temporary data storage.

Recommendation:

- Use SRAM for variables, stack, and heap allocation that require quick read/write access.
- Regularly monitor SRAM usage to prevent stack overflows and ensure efficient memory allocation.

3.15.4. Leveraging XIP for External Code Execution

Observation:

- XIP (0x19FFFFFF - 0x20000000) enables execution directly from external flash memory.

Recommendation:

- Utilize XIP for large codebases or rarely executed code to free up internal RAM and Flash memory.
- Ensure that the execution speed from external memory meets the application's performance requirements.

3.15.5. Using FICR and UICR for Configuration Data

Observation:

- FICR (0x10000000 - 0x10000FFF) and UICR (0x12000000 - 0x12000FFF) store factory and user-specific configuration data, respectively.

Recommendation:

- Store device-specific and user-defined configuration settings in FICR and UICR.
- Use these regions to keep calibration data, unique identifiers, and bootloader settings.
- These registers should be written infrequently to avoid wear.

3.15.6. Optimizing Peripheral Access

Observation:

- AHB (0x50000000) and APB (0x40000000) peripheral spaces are designated for high and low throughput peripherals.

Recommendation:

- Map high-speed peripherals that require frequent access to the AHB space for efficient data transfers.
- Use the APB space for peripherals with lower bandwidth requirements.
- Ensure that peripheral access patterns do not create bottlenecks, particularly in real-time applications.

3.15.7. Balancing RAM and Peripheral Usage

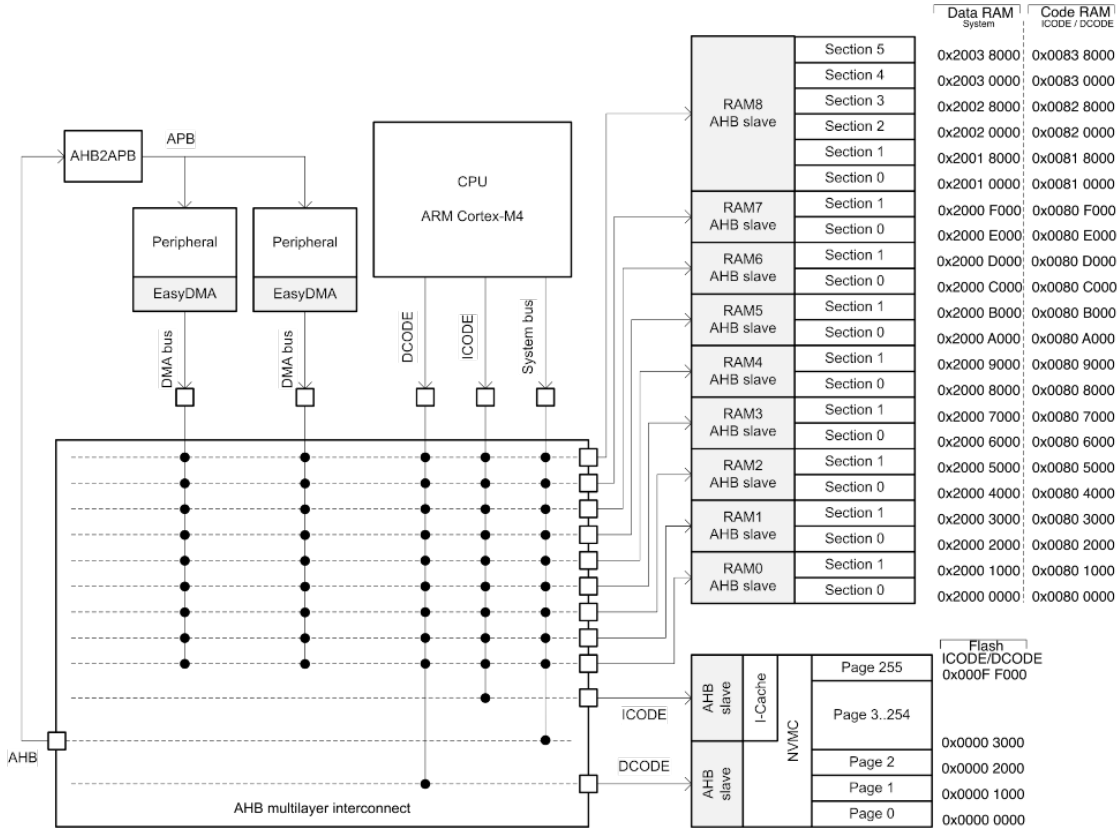
Observation:

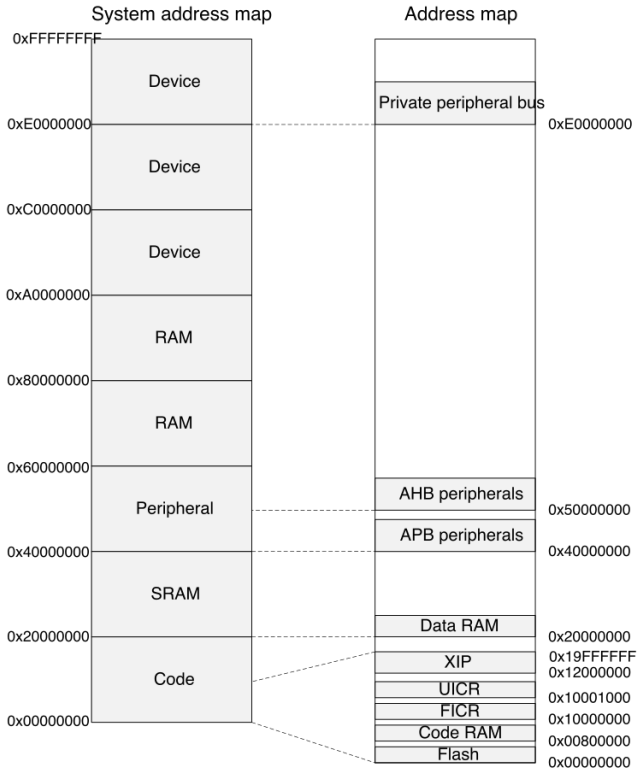
- The board has multiple RAM sections and peripheral regions connected via AHB and APB buses.

Recommendation:

- Distribute memory usage across different RAM sections to avoid contention and ensure balanced load.
- Efficiently manage DMA transfers to minimize CPU involvement in data movement between peripherals and memory, thus improving overall system performance.

Our boards, the nRF52840 contain 1024 kB of flash memory and 256 kB of RAM, they can be used for both code and data storage. [https://infocenter.nordicsemi.com/index.jsp?topic=





Structure of the board memory, many images to describe ram. No cache Why no cache is a problem and why it is ok. What devices have caches maybe. Show a couple methods to avoid programmer control over memory managment from paper. We could include metrics later if we have time, like space occupied and shit and if we can reduce that.

3.16. Radiation absorbtion

3.17. Channel selection

We have already described in previous sections the project at hand and why every possible improvement can ease the development of a functional implementation. One of those crucial aspects is the amount of information sent between the devices, in the particular the channels that get transmitted. As previously shown, the dataset at hand was registered using cuff electrodes chirurgically implanted on the animals' peripheral nerves; each of these cuff electrodes has 16 sensors that are able to measure the voltage of the nerve they cover. Each of these sensors, being in a different position, will register different data, furthermore one or more of these sensors may be damaged during the surgery or may record data with high noise. Because of this, not only it would be useful to limit the

channel we share between devices to help with the communication, but we could also help the classifier by removing channels with faulty or noisy sensors. In short, the three benefits of channel selection are : reducing the computational complexity of any process on the dataset, reduce the amount of overfitting of the models and reudce the times to setup the application. [**?**] There is a great amount of literature that deals with the issues of channel selection of EEG channels, since it is a well known area of research and proves to be very helpful in reaching better results. In particular our area of interest focused on channel selection applied to motion-imagery EEG Signals (MI-EEG). We will now present an overview of the most relevant methods for channel selection.

3.17.1. Common spatial patterns methods

Common spatial patterns (CSP) method was firstly suggested for classification of multi-channel EEG during imagined hand movements by H. Ramoser [**?**]. The core concept involves employing a linear transformation to map multi-channel EEG data into a reduced-dimensional spatial realm using a projection matrix. Each row of this matrix represents channel weights. This process aims to optimize the variance of signal matrices for two classes. The CSP method relies on concurrently diagonalizing the covariance matrices of both classes. [**?**]

Let X ϵ R^{MxN} denote a matrix of EEG, where the channel number is M and the samples are denoted by N. The classic CSP problem is stated as follows:

$$\max_{W \epsilon R^M} = \frac{W^T C_1 W}{W^T C_2 W} \tag{3.8}$$

where W is a spatial filter coefficient, C_i(i=1,2) indicates a single-class covariance matrix. The generalised eigenvalue decomposition (EVD) generally can handle this problem. [**?**]

Sparse CSP

Due to classic CSP inadequacies, several researchers aim to integrate sparsing behavior with conventional CSP to discover and eliminate highly noisy or interfering channels. Given w's sparsity k, i.e., the number of nonzero items in w. The sparse CSP problem is stated as follows:

Regularized CSP

$$\tilde{C}_C = (1-\gamma)\hat{C}_C + \gamma I \tag{3.9}$$

$$\hat{C}_C = (1-\beta)s_C C_C + \beta G_C \tag{3.10}$$

Another approach to obtain RCSP algorithms consists in regularizing the CSP objective function itself (1). More precisely, such a method consists in adding a regularization term to the CSP objective function in order to penalize solutions (i.e., resulting spatial filters) that do not satisfy a given prior. Formally, the objective function becomes:

$$J_{P_1}(w) = \frac{w^T C_1 w}{w^T C_2 w + \alpha P(w)} \tag{3.11}$$

3.17.2. Correlation-based methods

These methods help identifying which subsets of channels is the most relevant by defining a correlation metric and ranking higher those channels that show stronger similarities between them according to the metric. There are various ways in which we can score the correlation between signals of different channels and here we will briefly present three.

Spectral entropy dunno first citation

Spectral entropy is a generic measure of disorganization of signal, as expressed in [**?**] with the following formulas:

$$H(E) = -\sum_{i=1}^{N} p\left(E_i\right) \log_{10} p\left(E_i\right), \tag{3.12}$$

where $\mathcal{E} = \{\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n\}$ is the signal in the time domain. $p(\mathcal{E}_i)$ is the probability of $\mathcal{E}_i$. It is usually estimated by Burg's algorithm. [**?**]
Spectral entropy separates the background noise from the organized signal, the motor imagery in this study.

The 'interested class versus the rest' strategy is adopted for the channel selection using the correlation coefficient method. EEG data are split into two groups, s_1 and s_2, where s_1 is the group containing the class of interest.

Spectral entropy H_1 and H_2 are identified corresponding to s_1 and s_2. Correlation of the spectral entropy from the two groups, s_1 and s_2, is an indicator of how closely these two groups have a linear relationship, written as:

$$\rho\left(H_{1,j}, H_{2,j}\right) = \frac{\operatorname{cov}\left(H_{1,j}, H_{2,j}\right)}{\sigma_{H_1,j}\sigma_{H_2,j}}, \tag{3.13}$$

where σ_{H_j} is the standard deviation of the respective spectral entropy. j is the index of the channel.
For the purpose of channel selection, we consider the spectral entropy of each channel across all frequency ranges by taking the sum of the squared correlation coefficient,

$$P\left(H_{1,j}, H_{2,j}\right) = \sum_{i=1}^{N} \rho^2\left(H_{1,j}, H_{2,j}\right), \tag{3.14}$$

where $i = 1$ - N is the number of frequency bins during estimation of the spectral entropy.

Pearson's Correlation Coefficient

Pearson correlation coefficient is a statistical association or linear dependence between two or more random variables and, in our case, it can measure the linear correlation between two channels. [**?**]

$$\rho(X, Y) = \frac{1}{n-1}\sum_{i}^{N}(\frac{X_i - \bar{X}}{\sigma_X})(\frac{Y_i - \bar{Y}}{\sigma_Y}) \tag{3.15}$$

Considering the two variables, denoted as X and Y, with n representing the number of observations, $\bar{X}$ and $\bar{Y}$ as their respective means, and σ_X and σ_Y as the default deviations between them, the correlation coefficient $\rho(X, Y)$ ranges from 0 to 1. This range indicates the strength of the relationship, varying from low to high. In this context, the correlation coefficient is computed for each pair of EEG channels.

Cross-Correlation Based Discriminant Criterion (XCDC)

XCDC is another correlation method whose fundamental idea is that if a channel is informative and useful for the classification task at hand, then it should be highly correlated

Given two finite discrete time series $x(i)$ and $y(j)$ (i ϵ $[0, 1, ..., n-1]$, j ϵ $[0, 1, ..., m-1]$, $m \geq n$), their cross-correlation series is given as follows:

$$r_{xy}(k) = \sum_{i=0}^{n-1} x(i)y(i+k), k = 0, 1, ..., m-n+1 \tag{3.16}$$

where x and y are EEG signals of length T (m=n=T).
Zero-padding is performed at both sides of y so that the length of the cross-correlation series is the same of the original signals.
We apply the cross-correlation formula to the signals x and y, but we must firstly apply z-score normalization to account for the effect of signal amplitudes on cross-correlation.

$$\tilde{x_i} = \frac{x_i - \bar{x}_i}{\sigma_{x_i}} \tag{3.17}$$

Then, the similarity between x and y is measured with the following formula:

$$S(x, y) = \max(r_{x\hat{y}(k)}), k = 0, 1, ..., T-1 \tag{3.18}$$

with $\hat{y}$ being y with zero-padding. where x_i is the signal of the i-th trial collected from our channel of interest. $\bar{x_i}$ and σx_i are the mean and standard deviation of x_i respectively. Having defined the function S we now define the within-class similarity R_w and betweenclass dissimilarity R_b which are obtained as follows:

$$R_w = \mathbf{mean}(S(\tilde{x_i}\tilde{x_j}|c_i = c_j)) \tag{3.19}$$

$$R_b = -\mathbf{mean}(S(\tilde{x_i}\tilde{x_j}|c_i \neq c_j)) \tag{3.20}$$

where c_i is the class label of the i-th trial. The discriminant score D is then defined as follows:

$$D = \lambda R_w + (1-\lambda)R_b \tag{3.21}$$

in which λ is a weighting hyperparameter to be tuned empirically. Channels are ranked in a descending order according to their discriminant score D after obtaining D for every channel using.

Table 3.1: Accuracy for correlation based methods

Methods	Methods' stategy	Classifier	Accuracy	No. of Selected Chan-nels/Total No. of Channels	Dataset
Afghanistan	AF	AFG	004	004	004
Afghanistan	AF	AFG	004	004	004

3.17.3. Sequential Based methods

These methods implement a sequential approach where top features are identified iteratively. The most common example of Sequential method is the Sequential Forward Selection that was first used for variable selection by Whitney (1971).
It begins the research with an empty set of features, all the features are evaluated according to their impact on the evaluation metric and then, the variable with the best effect gets added to the subset of features.
This step gets re-iterated untill all the variables are progressively included in the set for comparisons purposes. SFS has what is defined as the "nesting property" which means that once a feature gets included in the subset, that feature stays in the subset even if removing it would improve the results. [**?**] A different version is the Sequential Backward Search which functions identically but starts from a full set of variables and removes one them one by one according to the worst impact on the evaluation metrics. [**?**]
Both of these methods' complexity is O(n^2). [**?**]

Sequential Floating Forward Selection (SFFS)

The sequential forward floating selection has in many instances demonstrated to be a superior algorithm.[**?**] [**?**] In our case in particular by addressing the nesting issue we better account for two features that highly impact the evaluation metrics but being highly correlated effectively carry the same information and may be not be both relevant to the final subset of features. It returns a better subset of feature at the cost of a higher complexity of O(2^n). [**?**]
The core approach utilized in Sequential Floating Forward Selection (SFFS) to address nesting issues involves initiating a backtracking phase after the addition of each variable. During this phase, variables are systematically excluded until superior subsets of corre-

sponding sizes are identified. This process continues until no better subset is discovered, prompting a return to the initial step to include the best previously excluded variable. Subsequently, another backtracking phase ensues. [?] [?]
In the original work of Pudil (1994) it is also proposed a floating version of the SBS called SBFS that implements the same idea but once again starting with a full set of features and removing them iteratively. Its complexity is $O(2^n)$.

Generalized Sequential Forward Selection (GSFS)

Generalized Sequential Forward Selection (GSFS) is an extension of the Sequential Forward Selection (SFS) algorithm.
In GSFS, similar to SFS, the algorithm starts with an empty set of features and iteratively adds features to the model based on some evaluation criterion, such as minimizing error or maximizing predictive performance. However, GSFS introduces additional flexibility by allowing the inclusion of multiple features at each step instead of just one. This can potentially speed up the feature selection process and lead to better performance by considering feature combinations more efficiently.

The key idea behind GSFS is to explore feature subsets in a more generalized manner, considering not only individual features but also combinations of features at each step of the selection process. This can be particularly useful in situations where interactions between features are important for modeling complex relationships in the data.

Overall, GSFS offers a more flexible and comprehensive approach to feature selection compared to traditional forward selection methods like SFS, making it suitable for a wide range of applications in machine learning and data analysis. [?] GSFS and the backward implementation GSBS, both have a complexity that depend of the number of features they can add per iteration, if we define g as the number of features added per iteration, then their complexity is $O(n^{g+1})$.
With g = 1 GSFS is equal to SFS.

3.17.4. Binary Particle Swarm Optimization Based methods

In 1995, the PSO algorithm was presented by Kennedy and Eberhart. [?]
The idea behind it is inspired by the social behavior of birds flocking or fish schooling. [?] In standard PSO, particles move around in a continuous solution space, adjusting their positions and velocities based on their own best-known position and the global best-known position found by any particle in the swarm.
This collective learning process guides the particles toward better regions of the search

space, converging to the optimal solution.
The balance between exploration (searching for new solutions) and exploitation (refining known solutions) is controlled by various parameters such as inertia weight, cognitive factor, and social factor.
Example update functions for position ($x_i(t+1)$) and speed (v^i_{k+1}) are as follows from Abdullah[**?**]:

$$v^i_{k+1} = wv^i_k + c_1 r_1 \left(p^i_k + x^i_k\right) + c_2 r_2 \left(gbest - x^i_k\right) \tag{3.22}$$

$$x_i(t+1) = x_i t + v_i(t+1) \tag{3.23}$$

where i represents the i-th particle. c_1 is the cognitive coefficient and c_2 is the social one. They regulate how fast the particle is going and are both constant. w is the inertial weight which limits the speed. r_1,r_2 are random values between 0 and 1. k represents the iteration of the update function. t represents the time. p^i_k represents the best solution found so far by the i-th particle at k-th iteration. gbest is the best position found by the swarm so far. Just below we present a visualization of particle movement in a 3D space.

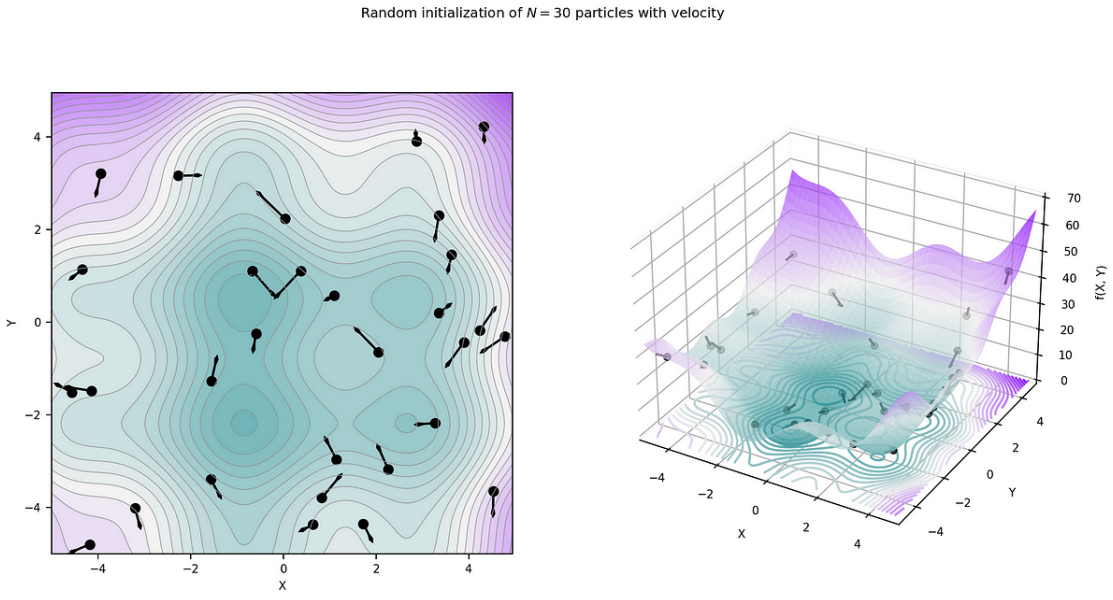


Figure 3.1: Illustration of particles moving in a 3D space

PSO has also a binary implementation called Binary Particle Swarm Optimization (BPSO). The basic principles of BPSO are similar to those of standard PSO, with particles searching for the optimal solution through iterative updates of their positions and velocities.

However, in BPSO, the binary representation of the particle's position is manipulated using binary operators such as bitwise AND, OR, and NOT, instead of arithmetic operations used in continuous PSO. [?]

The binary version suits feature selection since the binary values represent whether a feature is present or not in the subest of the most relvevant features.

We present here the updated equations for BPSO, the main difference being that here the position of the particle can either be 1 or 0. [?]

$$S(v) = \left(1 + e^{-v}\right)^{-1} \tag{3.24}$$

$$x_{k+1}^{i} = 1 \qquad \text{if } \tau < S\left(v_{k+1}^{i}\right) \tag{3.25}$$

$$x_{k+1}^{i} = 0 \qquad \text{if } \tau > S\left(v_{k+1}^{i}\right), \tag{3.26}$$

With v defined as previously in PSO, and S being a logistic function that varies from 0 to 1. We decide if the position of the i-th particle is 1 or 0 depending on the τ. There are also multi-objective versions for both PSO and BPSO called Multi-Objective Particle Swarm Optimization (MOPSO) and Binary Multi-Objective Particle Swarm Optimization respectively (BMOPSO). [?]

Table 3.2: Accuracy for particle swarm optimization methods

Methods	Methods' stategy	Classifier	Accuracy	No. of Selected Channels/Total No. of Channels	Dataset
Afghanistan	AF	AFG	004	004	004
Albania	AL	ALB	008	004	004
Algeria	asdasdsssssssssssasd	DZA	012	004	004
Andorra	AD	AND	020	004	004
Angola	AO	AGO	024	004	004

3.17.5. Other methods

We will now present a few other methods that are relvant for the idea behind them, the performance on the evaluation metrics or are particularly interesting with regard to future

possible developments.

Min. Redundancy Max. Relevancy (mRMR)

Rayleigh coefficient maximization Based Genetic Algorithm

Fisher's criterion

Automatic Channel Selection with Sparse Squeeze and Excitation Blocks (ACS-SE)

Auto-encoder

Dataset and results

We wanted to find algorithm that not only performed well on the usual Accuracy-F1Score metrics while reducing the numbers of channels, but also would fit well with our models. Given the architecture of our ENGNet model, we needed an algorithm to perform channel selectiom that would work with a Neural Network model. "' expand test case scenario and explain why "' In addition, given future test case scenarios we also needed an algorithm that could be performed in an "on-line" way, or as close to it as possible. To summarize we needed:

- -online
- Cnn ready
- good performance on f1-Accuracy
- works well with few channels

There are many algorithms that reduce noise... [**?**] but those are already inside the ENGNet since spatial features ... Talk about why ASR didn't improve, basically same reason. Ideally we would select channels, by iterating over all possible subsets of features but too much time. Particle swarm optimization no, beacuse too much time. Therefore correlation algorithms, promising and tested with CNN. Since we already had CNN why not implement a new algorithm that hasn't been well tested that requires to add one layer. Critical issue could be that generally black but we will discsuss it later.

4 | Materials and Methods

This chapter delves deeper into the materials and methods we used to both write the firmware code for the boards of the PNRelay project and to work on the algorithms for the channel selection. Given the limitations at hand, the focus of our work was to try and solve a list of different issues that came up during the development of the project. In particular:

- Optimization of firmware communication parameters
- Safe transmission via BLE
- Channel selection
- Calculation of dissipated power
- Memory management
- Reversible transmission

We will divide this chapter into two main parts: one that is related to the firmware and the hardware aspects, one that is related to the channel selection algorithms. These two parts have been carried on for the majority of our work indipendently and despite the common goal share little to no similarities as far as materials and methods go.

4.1. Channel selection

4.1.1. Newcastle Dataset

The Newcastle Dataset presents the recording of ENG signals of three different Sprague Dawley rats in response to mechanical sensory stimulation. To each of these rats a 16-contact cuff electrode was chrirurgically inserted and placed around the sciatic nerve in a distal position, close to where the nerve branches into its peroneal, sural, and tibial components. The data include ten different kinds of stimulation grouped into the three main categories of stimulation: Nociception, Proprioception and Touch.

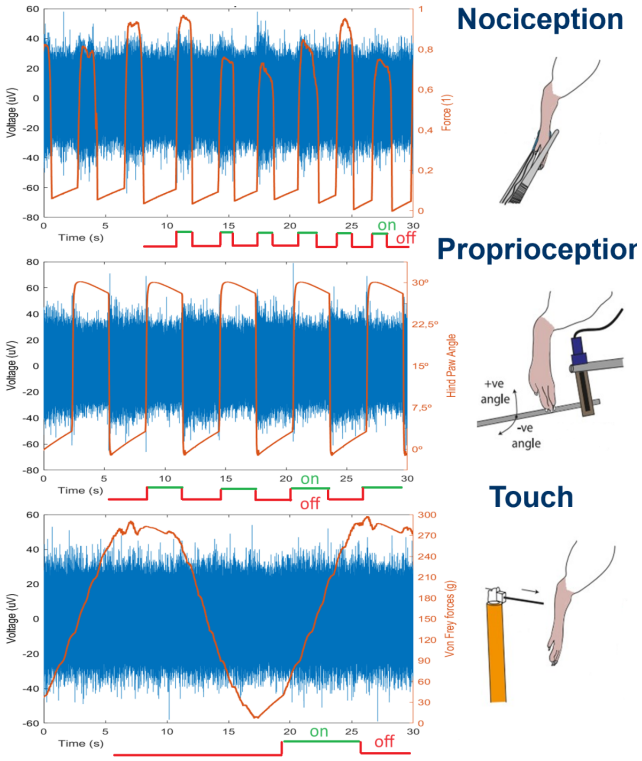
Nociception This class represents the pinching at two different points in the foot: the heel and the outer toe. These two nociceptive stimuli were elicited through manually operated tweezers with a Force Sensitive Resistor sensor connected to them in order to standardize the force applied, the duration and the synchronization in order to replicate the stimulus with precision.

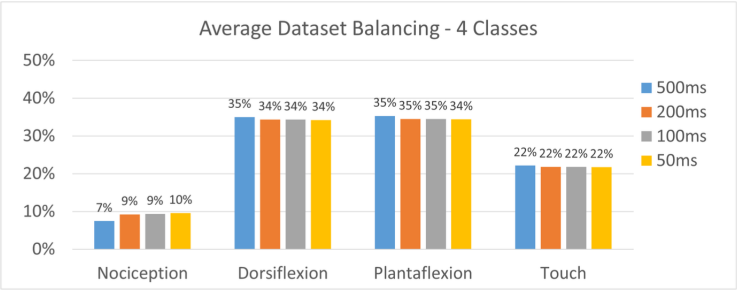
Proprioception This class of stimulus includes six different kinds of applied stimulation, each of them correspond to a different angle of ankle flexion, considered either dorsiflexion or plantarflexion. Ther is three different angles for dorsiflexion with positive degrees (-10°, -20°, -30°) and three different angles for plantarflexion with positive degrees (+10°, +20°, +30°). The angle measurement is relative to the neutral position of the ankle. This movements were obtained by pushing the hind paw of the rats with a bar, with a flex sensor to detect precisely the movement in order to standardize the procedure. Each movement was performed and then the position at the specific angle was kept for three seconds, then the neutral position was restored.

Touch This class represents two different stimuli based on the force of the touch: 0.1 Kg and 0.3 Kg. These two stimuli were elicited using Von Frey(VF) fibers, once again with a flex sensor in order to synchronize the procedure and standardize it by unfolding as the motor pushed the VF fiber to touch the hind paw.

While these ENG recordings were taken the rats were anesthetized in order to eliminate any possible EMG voluntary activity. The signals were captured at a frequency of 30 kHz and then digitally flitered between 0.25 and 5kHz. Each of the different stimuli previously presented was applied 50 times, producing 500 sampples in total for each rat. [davide]

Preprocessing





4.1.2. Channel selection algorithms

As stated previously, the focus of the channel selection algorithm is to reduce the number of channels that need to be transmitted in order to perform the classification of the signals properly. In particular as we will see better in the results chapter, we consider as an acceptable loss anything better than a 5 percentage points drop in F1-Score which is our core metric for the evaluation of the classification.

Test-case conditions

Poeple + setup +time + how signals come ....

Our dataset as explained in the previous section comprises 16 channels. Once condition that needs to be specified is that in order to simulate real test-case condtions on people, the channels may not come in batches of 16 channels, but as 8 channels + 8 channels. Therefore we performed our algorithms of evaluation not only on the whole 16 channels, but also on 8 channels, then other 8 channels and then cross-checked the results. In addition we have also tried a different approach we will explain later that due to its characteristic of being implemented at training time could not be applied to both 16 channels and 8 + 8 channels but requiered the 16 channels

Correlation based algorithms

These methods help identifying which subsets of channels is the most relevant by defining a correlation metric and ranking higher those channels that show stronger similarities between them according to the metric. As shown in Chapter 3 there are more than a few examples of correlation based algorithms for channel selection. We focus here on the ones we used extensively during our work: Cross-Correlation Based Discriminant Criterion (XCDC) and our own iteration. We will now present again the structure of the algorithms explaining the steps and our overall pipeline for the channel selection.

First of all, we take 48 samples for each one of the four classes we described in the dataset section.

- 48 Nociception samples
- 48 Plantaflexion samples
- 48 Dorsiflexion samples
- 48 Touch samples

Given two finite discrete time series $x(i)$ and $y(j)$ (i ϵ $[0, 1, ..., n-1]$, j ϵ $[0, 1, ..., m-1]$, $m \geq n$), their cross-correlation series is given as follows:

$$r_{xy}(k) = \sum_{i=0}^{n-1} x(i)y(i+k), k = 0, 1, ..., m-n+1 \tag{4.1}$$

where x and y are EEG signals of length T (m=n=T).
Zero-padding is performed at both sides of y so that the length of the cross-correlation series is the same of the original signals.
We apply the cross-correlation formula to the signals x and y, but we must firstly apply z-score normalization to account for the effect of signal amplitudes on cross-correlation.

$$\tilde{x_i} = \frac{x_i - \bar{x}_i}{\sigma_{x_i}} \tag{4.2}$$

Then, the similarity between x and y is measured with the following formula:

$$S(x, y) = \max(r_{x\hat{y}(k)}), k = 0, 1, ..., T - 1 \tag{4.3}$$

with $\hat{y}$ being y with zero-padding. where x_i is the signal of the i-th trial collected from our channel of interest. $\bar{x}_i$ and σx_i are the mean and standard deviation of x_i respectively. Having defined the function S we now define the within-class similarity R_w and betweenclass dissimilarity R_b which are obtained as follows:

$$R_w = \mathbf{mean}(S(\tilde{x_i}\tilde{x_j}|c_i = c_j)) \tag{4.4}$$

$$R_b = -\mathbf{mean}(S(\tilde{x_i}\tilde{x_j}|c_i \neq c_j)) \tag{4.5}$$

where c_i is the class label of the i-th trial. The discriminant score D is then defined as follows:

$$D = \lambda R_w + (1 - \lambda)R_b \tag{4.6}$$

in which λ is a weighting hyperparameter to be tuned empirically. Channels are ranked in a descending order according to their discriminant score D after obtaining D for every channel using.

A | Appendix A

If you need to include an appendix to support the research in your thesis, you can place it at the end of the manuscript. An appendix contains supplementary material (figures, tables, data, codes, mathematical proofs, surveys, ...) which supplement the main results contained in the previous chapters.

B | Appendix B

It may be necessary to include another appendix to better organize the presentation of supplementary material.

List of Figures

List of Tables

List of Symbols

Variable	Description	SI unit
$\boldsymbol{u}$	solid displacement	m
$\boldsymbol{u}_f$	fluid displacement	m

Acknowledgements

Here you might want to .