Politecnico di Torino

Polytechnic university of Turin

Master's degree course in Electronic engineering

A.y. 2022/2023

Degree session November/December 2022

Development of an embedded system based on an original bidirectional transducer

For the implementation of haptic human-machine interfaces in safety-critical automotive applications

Supervisors:
prof. Fabrizio Riente
prof.ssa Giovanna Turvani
ing. Flavio Cerruti

Candidate:
Fulvio Pace

Abstract

The goal that this master's thesis set out to achieve was the innovation of the standard automotive Human-Machine Interface (HMI) by implementing a prototype for a steering wheel based on an original transducer developed by TRAMA ENGINEERING, the *Niceclick*. Thanks to an *ad hoc* MCU-based PCB, the *Niceclick* evaluation board, it is capable of both sensing an applied force and providing an haptic feedback in response to any kind of contact, on the basis of a programmable force threshold. It is therefore perfectly capable of reproducing the sensations returned by the various mechanical controls found on the steering wheel or on the dashboard of a vehicle, especially of a car, and, as such, replace them, but without presenting the problems that plague other competing technologies, leaving room for further development too.

The thesis focused on the comprehensive development of the original *Niceclick* system, made of support board and up to four transducers, and went through all the phases of the electronic design process. Indeed, both hardware and firmware were flawed by issues or, at least, optimizations were needed and so, after the understanding of the system working principles, each problem was addressed and resolved by means of a systematic approach: the overview of the situation was obtained, then an algorithmic solution was designed relying on flow charts and finally implemented in hardware or in firmware (in C), thanks to PSoC Creator. Whenever necessary, simulations were carried out in MATLAB, Simulink or LTspice XVII and evaluations or acquisitions were accomplished, resorting to several hardware and software, depending on the purpose. In particular, data acquisitions were done thanks to IPEmotion, taking advantage of the CAN bus communication protocol.

The tests proved that the resulting embedded system is characterized by an enhanced force sensing capability thanks to a different measuring method. Then, a series of models was developed in order to reproduce the behavior of the *Niceclick* in different scenarios. So, a database, recording force sensing acquisitions, was build for further development. Finally, the intrinsic set of problems, related to the system working principles and its applied implementation, was overcome, improving the haptic effect generation capability.

The *Niceclick* transducer turned out to be equivalent, at least, to a mechanical push button, but, in fact, it proves even superior, because it offers customizable functionalities and it is open to further development, therefore, there are no limits to the possibilities of integration in any kind of application.

Contents

List of Figures

Introduction

Currently cars are characterized by a Human-Machine Interface (HMI), which is responsible for the management of the interaction between human and machine, consisting of a set of controls scattered all over the cockpit, especially on steering wheel and dashboard. Historically, such devices were fully mechanical, like push buttons, switches, knobs, wheels and levers, but more recently, their electronic versions or variations, like touch screens, panels and pads, are supplanting them.

The purpose of this master's dissertation was quite to modernize such interface thanks to the realization of a prototype of an automotive steering wheel. Indeed, it deals with the substitution of all the mechanical controls with an original electronic replacement, the *Niceclick* system. Necessarily, such device has to guarantee the preservation of the functionality, but also of the feeling returned by what it is intended to replace. It is important to note that the subject matter of the research relies on the sense of touch only: nevertheless, other prospective solutions could be visual warnings or acoustic warnings, but for such application they were not taken into account, because they do not properly fulfill the task of a mechanical control and, again, do not rely on the sense of touch.

Before presenting it, it is necessary to describe the context in which such substitution has to take place, to comprehend both the requirements and the problems, whose combination imposes the conditions that the potential replacement has to meet in order to be considered a solution for all intents and purposes, but also to take into account other possible options too, in order to have an inclusive overview of the reasons behind this work.

In automotive field, standards are restrictive, especially for what concerns safety, because it is a critical point obviously, but not from every point of view. Supposing to refer to an automotive steering wheel, equipped with a series of controls and, as such, capable of handling a set of the car functionalities, the reason states that such devices should not be a source of distraction for the driver and, theoretically, should be capable to deal, as much as possible, with human error, like an accidental contact which may cause an involuntary activation or deactivation, again, for a matter of safety. Therefore, an ideal Human-Machine Interface to be integrated in the steering wheel of a car should be functional, compliant, while still being user friendly and visually appealing maybe, but, above all, safe.

There are not many competing technologies for such critical assignment, especially relying on the sense of touch: a solution could be replacing the mechanical controls with devices exploiting touch technology. Such solution works from several perspectives, but it is not the best for the purposes of this thesis, because it suffers from non-negligible safety issues. In fact, touch sensing is no more a pioneering technology, it is the main building block at the foundation of the HMI in a lot of different applications and, as such, it has no problems related to reliability for sure. Moreover, for what concerns functionality, almost nothing would change with respect to mechanical controls. Among other things, touch screens can gather and manage many car features, instead of having controls scattered all over its interior and, finally, touch switches

can be placed directly on the steering wheel also, preventing the driver from taking the hands off it in order to take advantage of the functionalities. Basically, such HMI would exploit the principle of operation of a common smartphone and would be characterized by the same usability. However, touch technology is troubled by problems too: like smartphones, touch switches and touch screens require constant visual attention from the driver in order to be used and the question of potentially harmful inadvertent contacts is there also. Therefore, without deeply relying on the sense of sight, they do not make the driver aware of what is happening in the car, because such devices are passive, they do not provide any sort of feedback, so the resulting cause-effect relationship is not evident. Driving a car is a highly visual task already and they are a source of distraction [1]: it is a risk in terms of safety. So, touch does not suffice and can not be considered an efficient solution.

Being the non-visual feedback the key factor, a solution could be haptic technology. Haptic perception is achieved through the active exploration of surfaces and objects by a moving subject, as opposed to passive contact by a static subject during tactile perception. So, haptic and touch are equated by the exploitation of the sense of touch, but, especially with reference to this application, they differ much for what concerns the sensory information they provide. An haptic feedback is a tactile stimulus generated by a surface in response to a contact by a user. Therefore, the driver does not need anymore to considerably rely upon the vision in order to have the overview of the situation, because it is the control itself which returns such information, at most a glance suffices. So, haptic technology allows the feeling provided by a mechanical switch to be reproduced accurately. Nevertheless, the matter of accidental touches persists, making haptic not properly suitable as it is.

Since the fact that both touch and haptic technology prove not to be equipped to handle the situation, they still inspired the solution at the foundation of such work, the *Niceclick*. It is a transducer, developed by TRAMA ENGINEERING of Alba (CN), capable of both sensing a force applied to it and generating an haptic feedback detectable by a user, making it both a force sensor and an haptic actuator. Its duality is possible thanks to a combination of hardware and firmware which, on the base of a programmable force threshold, allows the transition from a mode of operation to the other, hardening it to inadvertent contacts. So, this solution fulfills all the requirements for such application.

This thesis is experimental by nature. Indeed, starting from the original embedded system, a lot of changes were made in order to evaluate the feasibility of potential improvements and a series of analysis was carried out to verify the effectiveness of such solutions, by means of both simulations and on-field tests.

Chapter 1

Technological overview

In order to understand the potentialities of the *Niceclick* system, an introduction of the main features of its original transducer is needed. The most important one is the capability of both behaving as a force sensor and an haptic actuator and, to stress the importance of such characteristic, a description of the state of the art is necessary.

1.1 Force sensing technology

[2] Many types of force sensors exist, each of which exploits a different operating principle to sense the magnitude of a force. The most common types of force sensors are load cells, strain gauges, and force sensing resistors.

1.1.1 Load cell

Load cells are a type of force sensor that measures forces such as compressive ones, most commonly weight. Load cells can exploit different working principles and so different types of load cells are available:

- Pneumatic load cell;
- Hydraulic load cell;
- Piezoelectric crystal load cell;
- Inductive load cell;
- Capacitive load cell;
- Magnetostrictive load cell;
- Strain gauge load cell;

Pneumatic load cell

Pneumatic load cells consist of a source of pressurized air or gas that is fed through a pressure regulator to a chamber inside the load cell itself. A flexible diaphragm is compressed when a force is applied to the plate on the top of the sensor. A pressure gauge measures the pressure resulting from the application of the weight. The amount of pressure needed to balance out such weight can be used to measure the weight itself and can be converted to an electrical signal.

Hydraulic load cell

Hydraulic load cells are similar to pneumatic load cells and use a pressurized liquid such as oil or water to balance out the applied load. A flexible diaphragm inside it is mounted beneath a piston attached to a plate, called the load platform. Applying a load, its weight moves the piston which flexes the diaphragm and compresses the fluid in the chamber, causing an increase in pressure. A pressure gauge monitors such change, whose value is directly proportional to force. After proper calibration, the pressure can be converted into weight and read directly from an analog gauge or may be converted into an electrical output signal. Hydraulic load cells, as well as the pneumatic ones, do not directly rely on the use of current and so can be used in environments where there is a risk of a potential explosive hazard.

Piezoelectric crystal load cell

Piezoelectric crystal load cells are based on the piezoelectric effect. Inside there is a set of crystal elements with an electrode between them. In the absence of mechanical excitation, the crystals are unstressed and exhibit a balanced charge. When subjected to a load, the crystal distorts itself causing a change in the center of symmetry of the charges producing a proportional change in the overall charge, which can be measured. Measuring the amount of charge provides information concerning the magnitude of the weight or force. A charge amplifier can be employed in order to convert the magnitude of the charge to an analog or digital output signal.

Inductive load cell

Inductive load cells feature a ferromagnetic core within the coil of a solenoid. Applying a load to the sensor, the force changes the position of the core within the coil causing a change in inductance. Such variation can be used to quantify the movement and can be converted to a measurement of the applied force. A reluctance load cell operates similarly, exhibiting a change in reluctance of an air gap in response to the application of a force.

Capacitive load cell

Capacitive load cells exploit the variation in capacitance of the sensor to measure the magnitude of the applied load. Inside there is a set of parallel plates that are charged by a current until a stable charge state is reached, implementing a capacitor. The amount of charge is directly proportional to the plate area and inversely proportional to the gap between the plates. As a load is applied, the force changes the distance between the plates, thereby changing the capacitance. Such change can be measured and converted into an indication of the weight.

Magnetostrictive load cell

Magnetostrictive load cells are based on the principle of magnetostriction, wherein the magnetic permeability of ferromagnetic materials changes if subjected to mechanical stress. When such materials are magnetized, they are subjected to strain, changing their length or inducing a torque. This physical principle is reciprocal: mechanical stress applied to the material results in a change to its magnetic properties. So magnetostrictive force sensors sense the amount of deflection that has occurred as a result of the application of a load. The deflection can be translated into a value representing the weight of the object or the magnitude of the applied force.

Strain gauge load cell

Strain gauge load cells exploit a strain gauge as sensing element.

Advantages

- Very accurate (< 0.1 % of full scale);
- Readily available;
- Calibrated by manufacturer;

Disadvantages

- Bulky in size and rigid construction;
- Costly signal conditioning electronics;
- Not a good solution for OEM/design-in applications;

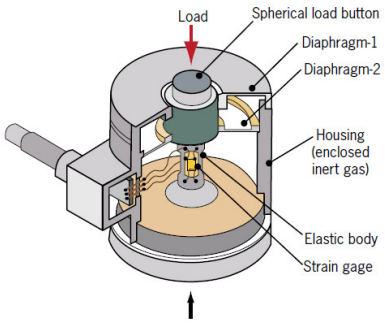

Figure 1.1: Structure of a strain gauge load cell

1.1.2 Strain gauge

Strain gauges are a type of sensor whose electrical resistance varies depending on the applied force. A strain gauge consists of an insulating substrate onto which a conductive metallic foil has been deposited in a zig-zag pattern. When subjected to a force, it either compress or elongate itself depending on the direction of the force. The elongation or compression distorts the metallic foil on the substrate, which changes its electrical resistance. Such change can be used to measure the applied force. A Wheatstone bridge circuit is used to convert the change in electrical resistance to a voltage one.

1.1.3 Force Sensing Resistor (FSR)

Force sensing resistors, also known as printed force sensors or force-sensitive resistors, are a type of piezoresistive sensor that consists of a semi-conductive material or ink which is sandwiched between two substrates that are separated by a spacer. When a force is applied, a conductive film is deformed and comes into contact with the conductive ink printed on the substrate. As more of the conductive film comes into contact with the printed conductive layer, the electrical resistance decreases. With no force, the sensor exhibits a very high resistance. The electrical resistance is inversely proportional to the applied force. FSRs exhibit a linear increase in conductance with force. Force sensing resistors can be used in different types of force sensing use cases:

- To detect the rate of change of an applied force;
- To detect a relative change in the applied force;
- To detect contact or touch;
- To detect that a force has exceeded a set threshold;

FSRs can be used for point source detection or in an array for detecting force distribution over an area.

Advantages

- Very thin (0.2 mm) and flexible construction allows for unobtrusive measurement;
- Readily available;
- Inexpensive signal conditioning electronics;
- Great solution for OEM/design-in applications;
- Customizable;
- Ideal for innovative product design: thin, light weight, low power requirements;

Disadvantages

- Less accurate ($\pm$ 5 % full scale) than typical load cell;
- Calibrated by user;

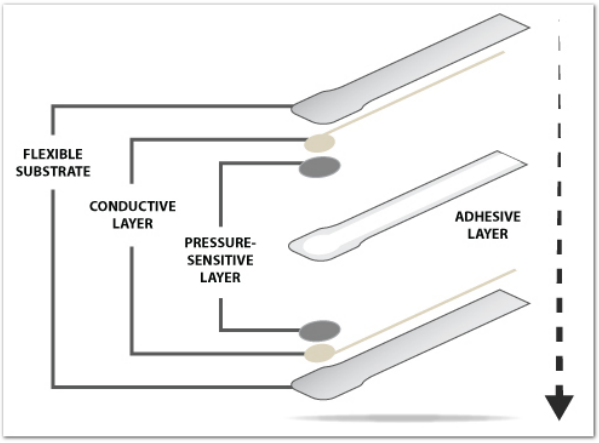

Figure 1.2: Structure of a force sensing resistor

1.1.4 Other types of force sensors

Other force sensor types are optical and ultrasonic ones. Optical force sensors employ an optical fiber into which has been inscribed a Fiber Bragg Grating (FBG) at specific intervals along its length. If the cable is subjected to stress or strain, the distance of the grating changes. By monitoring the reflections of light as it passes through the cable, the degree of deformation (elongation or compression) can be established and used to quantify the force applied. Ultrasonic force sensors emit high-frequency sound waves from an ultrasonic transducer and sense any changes in received pulses due to the application of a force.

1.2 Haptic technology

The most common haptic actuators, based on different haptic technologies, are Eccentric Rotating Mass (ERM) motors, Linear Resonant Actuators (LRA) and piezoelectric actuators.

1.2.1 Inertial haptic actuator

Eccentric Rotating Mass (ERM) vibration motor or pager motor

[3][4] The Eccentric Rotating Mass vibration motor, or ERM, also known as pager motor is a DC motor with an offset mass attached to the shaft. As the ERM rotates, the centripetal force is asymmetric, resulting in a net centrifugal force, and this causes a displacement of the motor. So the motor is constantly being displaced and moved by these asymmetric forces. It is this repeated displacement that is perceived as a vibration. However, the source of vibration from rotating machinery should be minimised because it generates noise and causes excessive machine wear and fatigue.

Miniature DC vibration motors have the benefit of being easy to implement, due to their small size and low power requirements, and are cheap. Moreover, integrating pager motors into a design is not hard and a wide range of simple motor drive circuits can be implemented. ERMs were made available in a wide range of form factors to suit all kinds of applications. For example, although coin vibration motors look dramatically different externally, they still rotate an internal eccentric mass to create an unbalanced force. Their design makes them very low profile and the

eccentric mass is protected, however, this means their vibration amplitude is limited. However, there are design trade-offs for each form factor:

- Coin/Pancake;
- BrushLess (BLDC);
- Encapsulated/Enclosed;
- PCB mounted;

Advantages

- Low cost;
- Widely available;
- Mature technology;
- Wide range of specifications;
- Simple design and use;

Disadvantages

- Vibration signal amplitude relies on driving frequency;
- High power requirement;
- Slow response (starting and stopping);

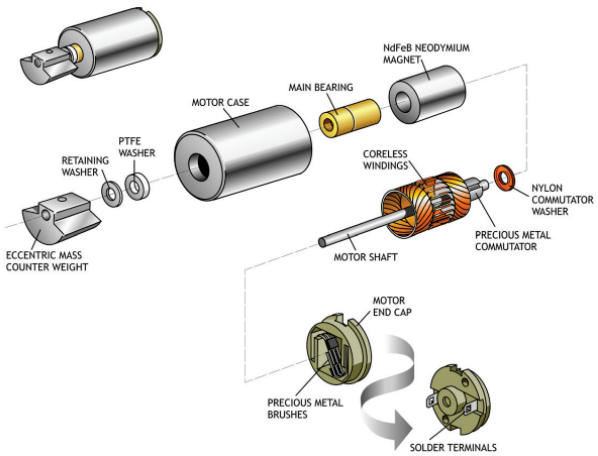

Figure 1.3: Structure of an ERM (courtesy of Precision Microdrives)

Linear Resonant Actuator (LRA) or linear vibrator

[5][6] Linear resonant actuators are an alternative to eccentric rotating mass vibration motors and have several distinct advantages. For example, they have better haptic performance characteristics and are more efficient (at resonance). Therefore they are used in many handheld and touchscreen devices, amongst other applications.

As with Eccentric Rotating Mass vibration motors (ERMs), the vibrations created by an LRA are based upon the movement of a mass which causes repeated displacement. Firstly, the ERM has an off-centre load and so, when it rotates, the centripetal force causes the motor to move. The rotation is obtained by applying a current to the armature windings attached to the motor shaft. As these are inside a magnetic field created by the permanent magnets on the inside of the motor, a force is generated causing the shaft to rotate. To ensure that the rotation continues in the same direction, the current in the windings is reversed. This is achieved by using static metal brushes at the motor terminals, which connect to a commutator that rotates with the shaft and windings. The different segments of the commutator connect to the brushes during rotation and, as a consequence, the current is reversed, maintaining the direction of rotation.

Similarly, LRAs use magnetic fields and electrical currents to generate a force. A major difference is that the voice coil (the equivalent of the armature windings) remains stationary and the magnetic mass moves instead. The mass is attached to a spring, which helps it going back to its resting position. Driving the magnetic mass up and down causes the displacement of the LRA and thereby the vibration. As the voice coil remains stationary and the direction of the force on the magnet has to be switched, the direction of the current in the voice coil has to be switched too. This means that LRAs require an AC drive signal to operate correctly. It is preferable that this signal is a sinewave at the resonant frequency. The driving waveform can, of course, be amplitude modulated to generate more advanced haptic feedback effects.

The LRA works analogously to a loudspeaker. In such device, the speaker cone is used to produce audio waves through displacement. However, a loudspeaker is designed to operate over a range of frequencies, whereas an LRA is tuned to its resonant frequency only, because the combination of spring stiffness, mass and magnet/coil size will cause the linear vibrator to have a natural resonant frequency, and, instead of a cone that produces sound pressure waves, there is a mass that generates vibrations.

Advantages

- More acceleration for the same size compared to ERM;
- Easy amplitude modulation;
- LRA consumes less power compared to ERM;
- Quicker response;
- Smaller in size;
- High power efficiency due to high Q;
- Reliable;

Disadvantages

- Only vibrates in one axis;
- Resonant frequency varies between each LRAs, due to manufacturing variations;
- Lower vibration amplitude outside resonant frequency due to narrow operating bandwidth;

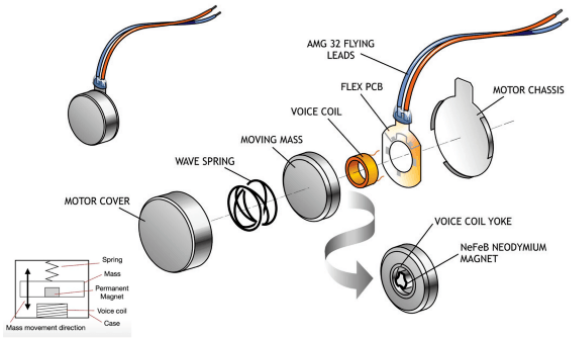

Figure 1.4: Structure of a LRA (courtesy of Precision Microdrives)

1.2.2 Piezoelectric actuator or ceramic actuator

[7] The ERM is perfect for vibration alerts. However, exploiting an ERM for other haptic applications quickly runs out the battery. The ERM is inertial and needs overdrive to spin faster. Start-up time is in the range from 50 to 100 ms. Braking or stopping the motor requires a similar time frame. For a simple haptic event like a click, the overhead ranges from 100 to 200 ms. If the application demands repeated haptic events, the latency associated with motor-based haptics may be undesirable. Another aspect of the ERM is the buzzing or audible noise associated with the spinning. This is less of a concern if the haptic feedback is combined with audio one. The ERM also has few perceptible haptic effects that can be generated and the vibration frequency and amplitude are tied to a single control voltage.

The LRA is used in smartphones for haptics and vibration alerts. The LRA must be driven at a narrow resonant frequency. It also tends to have a slightly better start-up time with respect to the ERM. Depending on the manufacturer, start-up time varies from 40 to 60 ms. This offers a slight improvement over the ERM start-up time which is between 50 and 100 ms. By modulating the resonance-carrier amplitude, it is possible to produce a variety of different haptic effects.

When a differential voltage is applied across both the ends of a piezo actuator, it bends or deforms, generating a vibration. Piezo actuators need high voltage to deform. Depending on the manufacturer, voltage can vary from 50 to 150 Vpp. At higher voltages, the number of required piezo layers decreases; so at 150 Vpp the piezo actuator has approximately 4 layers, whereas at 50 Vpp there may be from 16 to 24 layers. At higher voltages, due to the reduced number of layers, the piezo actuator's capacitance is lower. In other words, less current is needed to drive lower-capacitance haptic actuators. Piezo actuators are available as disks or as rectangular strips, also called benders. Piezo disks deform vertically. Piezo benders can be mounted directly to a "floating" touch screen to vibrate only the screen. Piezo benders can also be mass mounted in a small module that can be mounted to a device's case or PCB to provide vibration for the

whole device. Piezo modules have become popular because mechanical integration is easy. Four key elements distinguish piezo actuators from inertial actuators:

- Faster start-up time: due to inherent mechanical properties of piezo actuators, start-up time is very fast, typically less than 15 ms, which is three to four times faster than ERMs one. Compared to ERMs, the duration of the haptic event may be shortened by 70 ms.
- Higher bandwidth: the higher available bandwidth of piezo actuators offers a more detailed haptic palette with a greater number of effects.
- Lower audible noise: unlike ERMs, piezo actuators have no spinning mass creating mechanical noise.
- Stronger vibration: piezo modules tend to generate stronger vibrations. These stronger vibrations imply that piezo modules are a more suitable candidate for bigger- screen smartphones and tablets.

Even though piezo actuators need higher voltage with respect to inertial actuators, the true current consumption is lower than that of ERMs and on a par with that of LRAs.

Piezo actuators offer better performances and lower costs compared to inertial actuators. Their faster start-up time helps create sharper and crisper clicks for keyboard applications. Their higher bandwidth helps to create more perceivable haptic effects, which are critical for certain applications. The stronger vibration provided by piezo actuators can be used to generate haptic feedbacks for bigger con- sumer devices like tablets and e-readers. Overall, piezo haptics offer compelling features to enhance the tactile feedback experience and helps to improve the overall user experience of mobile devices.

Advantages

- Non-magnetic;
- Holding power not required;
- Quick rise and fall time;
- Higher resonance frequency;
- Multiple vibration patterns;
- Low power consumption;
- Smaller size;
- High precision;

Disadvantages

- Higher voltage;
- Nonlinear;
- Sensitive to electrical overdrive;

1.3 *Niceclick* technology

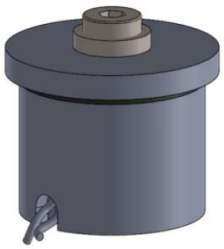

Figure 1.5: *Niceclick* transducer (courtesy of TRAMA ENGINEERING)

[8] The NC-C1012-12V is a linear actuator with integrated force sensor designed for moving a touch surface with a predefined vibration. The user touch action on the surface is detected and measured by the integrated force sensor giving to a control system the needed information to trigger a specific haptic sensation. The typical use of NC-C1012-12V is to reproduce the haptic sensation of a standard tactile switch over a touch surface or a touch screen. A more general application is the use of the NC-C1012-12V for creating user specific haptic effects driven by a microcontroller. A *Niceclick* haptic actuator is an electromechanical device composed of three coaxial main mechanical parts:

- Stator;
- Elastic element;
- Cursor;

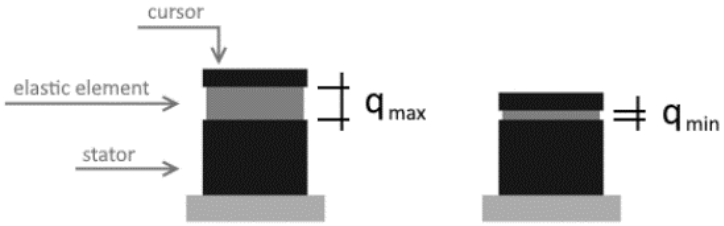

Figure 1.6: Structure of a *Niceclick transducer* (courtesy of TRAMA ENGINEERING)

The *Niceclick* actuator is a small cylinder which can change its vertical dimension when electrically powered. When an electrical current flows in it, the stator will attract the cursor so compressing the elastic element: the vertical dimension will decrease. When the electrical current stops, the elastic element will pull the cursor up until the device will come back to the original dimension. The standard installation of a *Niceclick* device is to attach one side (stator or cursor) to the haptic surface and the other side to the haptic device base or to a mass. When the device is powered, it will change its dimension so moving the haptic surface towards the base or the mass. The displacement of the surface, i.e. a vibration, will be recognized by a human finger as an haptic effect. As the human fingers can recognize specific profiles of vibrations, modulating

the electrical power inside the *Niceclick* will result in creating a controlled surface movement. If this movement is a replica of the movement of a physical tactile switch, the perceived haptic feedback will be the same of the tactile switch. The distance q between the stator and the cursor is a very important parameter as it is needed to create a specific vibration profile. For quality reasons, especially for wide haptic surfaces, the displacement of the cursor has to be limited to a very low value. Thanks to its characteristics, there are a lot of fields of application: automotive, industrial, consumer household appliances, general purpose.

Chapter 2

Niceclick system overview

The functionalities offered by the *Niceclick* are the product of a complex system, based on well-defined working principles and on the interaction between hardware and firmware, which have to be introduced.

2.1 Working principles

The working principles at the foundation of the system come from the magnetic properties and the arrangement of the constitutive elements which characterize the transducer structure:

- Stator and cursor, made of soft iron;
- Coil, made of copper;
- Air gap, occupied by an elastic element;

Soft iron physical properties are particularly important for the purposes of the study: it is a ferromagnetic material, characterized by a non-constant magnetic permeability μ_r, which exhibits magnetic phenomena like hysteresis and saturation, but also by marked affordability and workability. In order to carry out a basic analysis, a series of assumptions, for the sake of simplicity, has to be made:

- Air gap is dimensionally negligible with respect to other elements of the structure;
- Soft iron and air are characterized by the same cross-section;
- Air is considered as vacuum;
- Coil is considered as an infinite continuous solenoid;

For what concerns magnetism, in vacuum:

$$B = \mu_0 H$$

Where:

- B is the magnetic flux density in tesla;
- μ_0 is the vacuum magnetic permeability in henry per meter;

- H is the magnetic field strength in ampere per meter;

In particular, in case of a solenoid:

$$H = \frac{NI}{l}$$

Where:

- N is the number of turns of the solenoid;
- I is the current in the solenoid in ampere;
- l is the length of the solenoid in meters;

And so, due to Ampère-Maxwell's law:

$$B = \frac{\mu_0 NI}{l}$$

However, in the transducer, the solenoid is not in vacuum, but it is winded up to a part integral with stator. Due to the magnetic nature of the stator itself, the formula has to be modified, obtaining:

$$B = \frac{\mu_0 \mu_r NI}{l}$$

But:

$$\mu = \mu_0 \mu_r$$

And so:

$$B = \frac{\mu NI}{l}$$

Therefore, with equal currents, the magnetic induction of a soft iron-core solenoid is multiplied by a factor μ_r with respect to that of an air-core one. In case of a uniform magnetic circuit as assumed at the beginning of such analysis:

$$\mathcal{R} = \frac{l}{\mu A}$$

Where:

- $\mathcal{R}$ is the magnetic reluctance in inverse henry;
- A is the cross-section area in square meters;

After the introduction of permeability and reluctance concepts, a sort of parallelism between electric and magnetic circuits can be realized: in the first, if voltage and resistance values are determined, a current is induced; in the latter, it is possible to substitute:

- Voltage V with magnetic field strength H;
- Resistance R with magnetic reluctance $\mathcal{R}$;
- Current I with magnetic induction B;

So, in the same way, if magnetic field strength and reluctance are established, a magnetic flux density is induced, but, in the case of the *Niceclick* transducer, total reluctance is due to two different contributions: soft iron and air ones. Therefore, this is electrically equivalent to a series of two resistors. Moreover, in case of an ideal non-air-core solenoid, it is possible to define magnetic flux as:

$$\Phi = \frac{\mu N I A}{l}$$

But resorting to reluctance:

$$\Phi = \frac{NI}{R}$$

Which can be put in relation with inductance in henry:

$$L = \frac{N\Phi}{I}$$

Obtaining:

$$L = \frac{N^2}{R}$$

Therefore, permeability, reluctance and inductance are, in broad terms, proportional and depend on the magnetic properties and/or the geometry of a material. In such case, the geometry of the *Niceclick* transducer is not constant due to the presence of the air gap (occupied by an elastic element), whose thickness changes on the basis of the force applied to it. As such, it causes an alteration to reluctance and so a variation of inductance, because, being inversely proportional, they both depend on the relative position between cursor and stator. So, if it would be possible to evaluate the inductance, the displacement would be obtained also.

Applying a current to the coil, due to the Ampère-Maxwell's law, a magnetic field, characterized by well-defined flux density and strength, is generated. If the current varies, its strength exhibits a variation and, as a consequence, the induction too. The *Niceclick* transducer implements, for all intents and purposes, an electromagnet, characterized by north and south poles and, as such, a current-depending Lorentz force is applied to surrounding objects exhibiting magnetic properties. If such force is high enough, it can cause the attraction of the cursor by the stator, decreasing the displacement and so compressing the elastic element. The evolution of the generated force, which depends on current and decreases with displacement, defined as q, is depicted in figure 2.1. Similarly, also inductance, at a given frequency, varies with displacement as shown in figure 2.2, highlighting a relation between the two physical quantities.

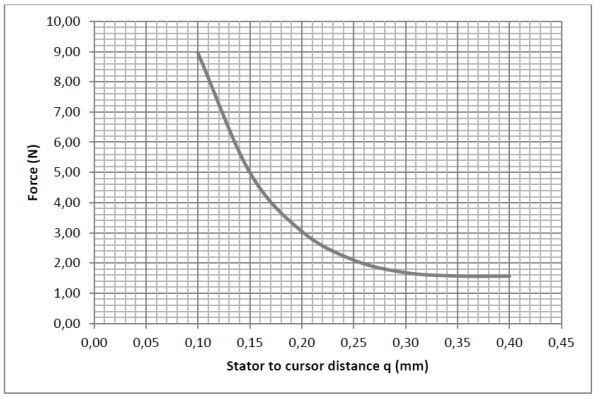

Figure 2.1: Generated force versus displacement (courtesy of TRAMA ENGINEERING)

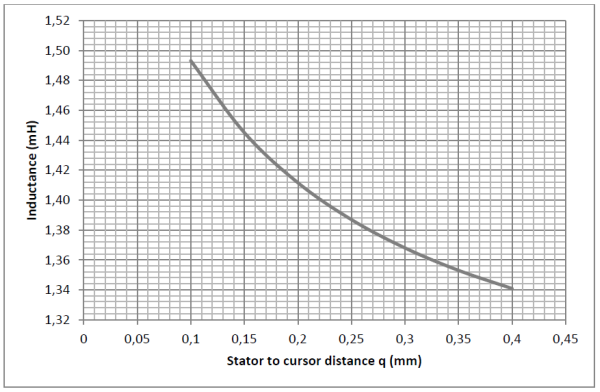

Figure 2.2: Inductance versus displacement (courtesy of TRAMA ENGINEERING)

In fact, the *Niceclick* transducer can be modeled as mass-spring system, with the cursor being the mass and the elastic element being the spring and, as such, can be described by a system of equations, but, for sake of simplicity, a series of assumptions has to be made:

- Elastic element is considered as perfectly linear
- Magnetic force is linearized in the neighborhood of the operating point
- Inductance is linearized in the neighborhood of the operating point

So, in broad terms:

$$\begin{cases} F_e = -k_e q \\ F_m \propto \frac{I}{q} \\ L \propto \frac{1}{q} \end{cases}$$

Where k_e is the stiffness of the elastic element in newton per meter. When a force is applied to the cursor, the elastic element is compressed and the displacement decreases. In order to quantify

such decrease, the inductance, which conversely exhibits an increase, is evaluated. Thanks to the stiffness of the elastic element and to the displacement, the applied force is quantified resorting to the Hooke's law. Moreover, knowing the relative position between cursor and stator allows the evaluation of the generated magnetic force to be evaluated also, if the current is determined.

Force sensing is made possible thanks to the presence of the elastic element and, indeed, it is obtained indirectly: in itself and as the system of equations confirms, the *Niceclick* transducer senses proximity (therefore position) only. So, particular attention has to be paid to the inductance evaluation, because it allows the estimation of the displacement. Since the fact that a solenoid is characterized by an inductance too, it is possible to model the coil as an ideal inductor. Referring to its constitutive equation, in time or in frequency domain, it is possible to quantify the inductance by imposing voltage, frequency and current. Therefore, in frequency domain:

$$V = j\omega LI$$

And inverting for inductance:

$$L = \frac{V}{j\omega I}$$

So, in order to quantify inductance, a current characterized by a non-null frequency has to be applied.

2.2 Hardware

In the *Niceclick* system, in order to apply a current, the coil is connected between DC supply voltage and ground thanks to a low-side switch and the current in it is modulated by applying a PWM signal to the switch itself. A low-DC PWM involves a weak current, the *Niceclick* system senses the displacement only, so, indirectly, the applied force, and the magnetic force generated is not sufficient in order to provoke the attraction of the cursor by the stator, due to the presence of the elastic element which opposes it. Conversely, a high-DC PWM involves a strong current, the generated magnetic force manages to overcome the opposition of the elastic element, causing the displacement to decrease. This is at the base of the haptic effect generation capability. So, the *Niceclick* transducer is both a sensor and an actuator, depending on the duty cycle of the PWM signal applied to the switch.

In general, there is a set of different solutions which allows an inductance measurement to be performed, but it reduces in case of an embedded system due to its limited resources and so capabilities. Therefore, an *ad hoc* circuit is needed. In particular, in the *Niceclick* system, a resistor and a capacitor are added to the inductor-switch pair, implementing a second order system indeed and, as such, characterized by a certain resonant frequency, expressed in hertz and defined as:

$$f = \frac{1}{2\pi\sqrt{LC}}$$

Where C is the capacitance in farad. So, the application of a PWM signal to the low-side switch determines the response of the circuit: it is a second order step response, like in figure 2.3.

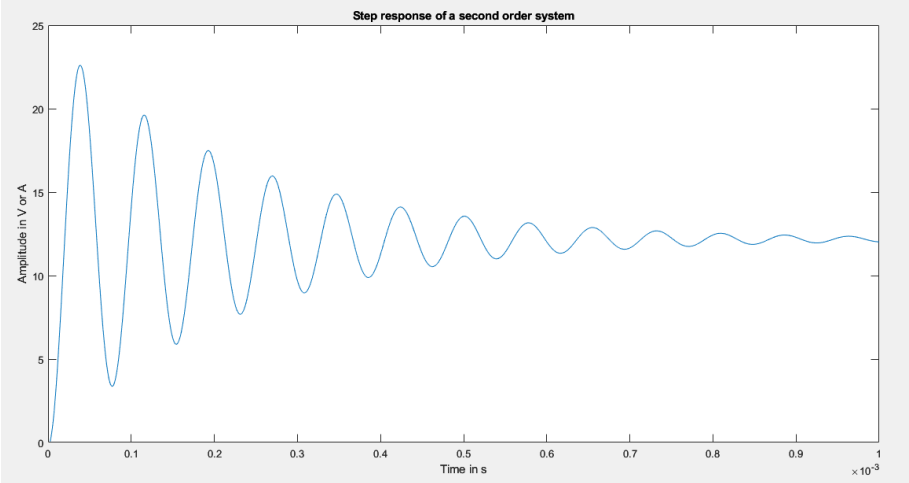

Figure 2.3: Step response of a second order system

The waveform, expressly designed to be underdamped and away from the human hearing range for what concerns its spectrum, intersects the steady state voltage, which corresponds to the DC supply one, in different time instants depending on the resonant frequency, therefore, if the inductance varies due to the displacement, the resonant frequency varies as a consequence and so the intersections. In particular, when the relative position between cursor and stator decreases, the inductance increases, causing the resonant frequency to decrease likewise and vice versa.

For the evaluation of the resonant frequency, the response of the RLC parallel resonator is compared against the steady state voltage thanks to a non-inverting analog comparator of the MCU, but only after a proper rescaling of the voltage levels, resorting to a resisitive voltage divider, to ensure analog-to-digital compatibility. As a consequence, a square wave, characterized by a displacement-dependent frequency, is present at the output of the comparator.

During the generation of an haptic effect, a high-DC PWM signal is applied to the switch, causing both inductor and capacitor to store energy inside the generated magnetic and electric fields, respectively. However, inductance and capacitance differ in several orders of magnitude, being the first higher with respect to the latter. This makes the current to charge the capacitor before the inductor and a change in the current direction which flows in the inductor instead of the capacitor occurs, resulting in a disturbed generation of the haptic feedback. In order to cope with this problem, a flyback diode, managed through suitable circuitry, is put in parallel to the inductor and is enabled in such occasion only, preventing also the damage of the switch due to the transient negative voltage drop caused by the energy stored in the inductive load and which is dissipated across the switch itself otherwise.

So, the *Niceclick* system is implemented as a double-sided MCU-based PCB, capable of handling up to four *Niceclick* transducer and communicating with other devices thanks to the CAN bus serial communication protocol. The original *Niceclick* system circuit, without the diode, is reported in figure 2.4.

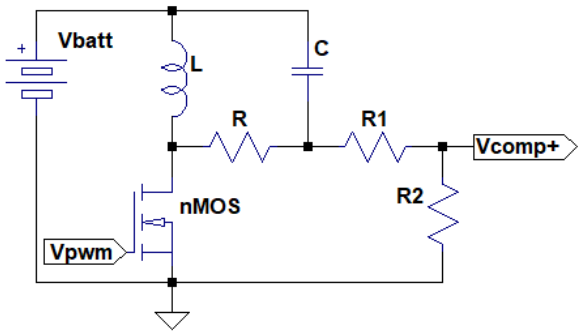

Figure 2.4: Schematic of the original *Niceclick* system circuit

2.3 Firmware

In order to manage such hardware, an *ad hoc* firmware was developed. Being an embedded system, the choice of the programming language fell on C and, due to its complexity, the project was structured in a hierarchical way and made of several modules.

The key elements of the hierarchy are drivers, layers, tasks and the *Niceclick* library. A driver is a part of the firmware which is specifically designed in order to manage hardware at bottom level. In particular, there is a driver which acts as an interface between hardware and firmware for each peripheral. In the *Niceclick* system, there are drivers for the implementation of the CAN bus communication protocol and the top-level system timer. A layer is a part of the firmware which operates as an Hardware Abstraction Layer (HAL), so as an intermediate level of abstraction between hardware and software. They are of particular importance for code reusability and compatibility. A task is a part of the firmware which is scheduled periodically in order to carry out the top-level functions. The timer, along with the tasks, is the key element for the implementation of a Real-Time System (RTS): a series of timer events, each characterized by a different period, is scheduled in order to execute, on time, the operations which allow the *Niceclick* system to function, by triggering an interrupt when the time is elapsed. The library is a collection of Application Programming Interfaces (API) which allow a user to interact with the application in order to program its behavior. Apart from this, the most important elements, which firmware manages, are:

- Input capture;
- Force threshold;
- Pulse-Width Modulator;
- Flyback diode;

For the evaluation of the inductance, only the two intersections of the first oscillation are taken into account, because this suffices for the determination of the resonant frequency. In order to accurately measure such inductance-dependent time instants, interrupt-based input capture is exploited and the interval in between is evaluated, obtaining the period. Then, the cursor-to-stator distance is calculated and finally the applied force is computed. Switching between *Niceclick* system modes of operation occurs thanks to the excess of a programmable force threshold. Programmable PWM properties, period and especially duty-cycle, define the second order step

response of the RLC resonator and clearly distinguish one modality from the other. Moreover, firmware allows the enabling/disabling of the flyback diode for protection purposes. Finally, the most important operations, like the management of the input capture and the generation of the haptic effects, are accomplished resorting to a series of Moore's Finite State Machines (FSM). The flow chart describing the top-level algorithm at the base of the *Niceclick* system operation is reported in figure 2.5

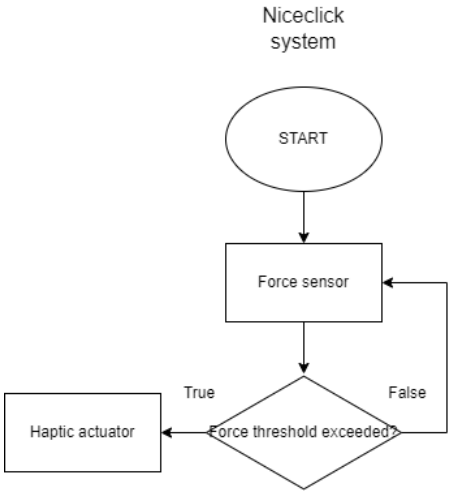

Figure 2.5: Flow chart of the *Niceclick* system

Chapter 3

Force sensor development

The chapter focus on the development of the *Niceclick* capability of indirectly sensing force, which is the key factor at the base of the operation of the embedded system.

First, it presents the limits and the problems which the original measuring system exhibited, focusing, in particular, on the limited resolution and the noise which affected the measurements obtained through input capture. Then, it moves to the theoretic description of the adopted solution, which is no more based on input capture, but on an ADC. Such solution is then analyzed more in detail: both the hardware and the firmware are presented. Finally, the obtained results are investigated out in order to remark that the original problems are solved and the system is tested out as a whole, especially, the measurements provided by the system are compared against the ones provided by a reference, a laser, an input-output characteristic of the system is obtained thanks to a dynamometer and a brief timing analysis is carried out also.

It is important to note that, in the following, the *Niceclick* system is considered made up of a board and a single transducer and directly sensing position instead of force (indirectly obtained). Moreover, the transition from sensor to actuator mode is disabled.

3.1 Problems of the original force sensor

The original *Niceclick* system was operational already, but its functioning was afflicted by several issues, primarily related to the applied implementation of the sensor mode of operation and to electronic noise.

As already said, sensing works in input capture mode, which involves the use of a digital comparator which support digital voltage and current levels only. However, in a lot of fields supply voltages are not compatible with digital ones, they may be higher and may be characterized by a high variability: for example, in the automotive field, nominal supply voltage, provided by a battery, is 12 V, but an automotive-compliant device has to work correctly within a higher range. The *Niceclick* system will be considered in an automotive context, so the nominal supply voltage will be 12 V. Then, considering its operating principle, based on the response of a underdamped second order system, such voltage levels at the digital comparator input would damage it irreparably. In order to prevent such situation, a couple of passive voltage dividers was applied, but being resistive, they cause an increase in noise.

Moreover, a digital comparator is not an ideal device: apart from its contribution to noise, it is characterized by hysteresis and offset problems which affect the result of the comparison. Such uncertainty, along with the low resolution due to the low number of bits employed, becomes

particularly critical when remembering how the measurements take place: in particular, only the two first intersections of the second order response are obtained through input capture.

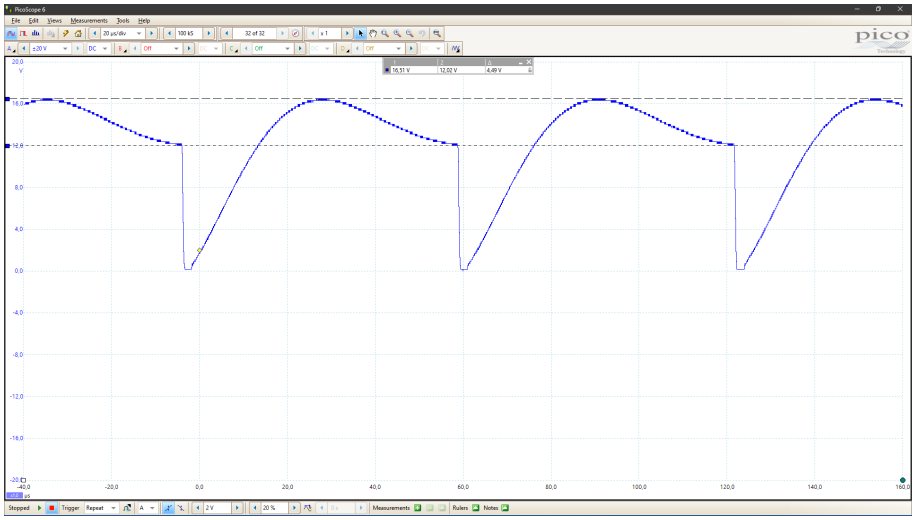

Figure 3.1: Periodical step response of the *Niceclick* system when cursor is pressed

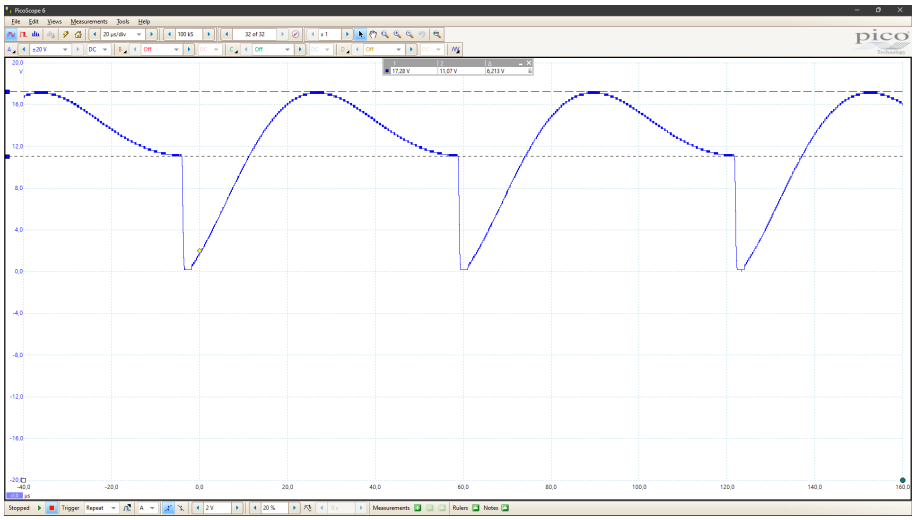

Figure 3.2: Periodical step response of the *Niceclick* system when cursor is released

As can be seen in figures 3.1 and 3.2, which report the change in the waveform caused by the application or not of a force to the cursor, supposing to perform input capture around 12 V on a single period, in the first intersection the response is almost vertical, so the derivative is positive and has a high value, while in the second the response is almost horizontal, so the derivative is negative, almost equal to zero. That said, it is possible to evince that due to noise and other uncertainty contributions, the recognition of the second point is much more critical. Moreover, due to its frequency behaviour, being the capacitor connected directly to supply voltage causes high frequency noise to affect the measurements.

So, due to noise and low resolution, considering that the time that elapses between two consecutive intersections has a variation in the order of few microseconds, the resulting system it is not so accurate.

3.2 Solution overview

In order to solve such problems, a series of actions took place: the most important is the radical change of the measurement method, but several other optimizations were carried out too.

First, the capacitor was connected to ground, instead of supply voltage, preventing the step response of the second order system from being disturbed by high-frequency noise without changing the order of the system.

Second, the resistive network, aimed at rescaling analog voltage levels in order to be compliant with the digital ones of the comparator and critical from the point of view of noise, was substituted by a passive high-pass CR filter.

Third, the digital comparator working in input capture mode was substituted by an ADC, characterized by an higher number of bits and whose conversions are hardware triggered, in order to improve both resolution and timing accuracy. Whenever a conversion is ended, a interrupt is triggered.

3.2.1 Hardware

The charging of the coil happens when the nMOS is on and so when the PWM signal, applied to it, is in its high state. Instead, the discharge of the coil happens exactly in the complimentary way. The discharging characterizes the sensor mode of the *Niceclick* system operation, because it actually causes the generation of the variable second order response.

While the original was made of few elements, the circuit is more complex and it is made of three different stages, which are going to be presented individually for sake of clarity.

First stage

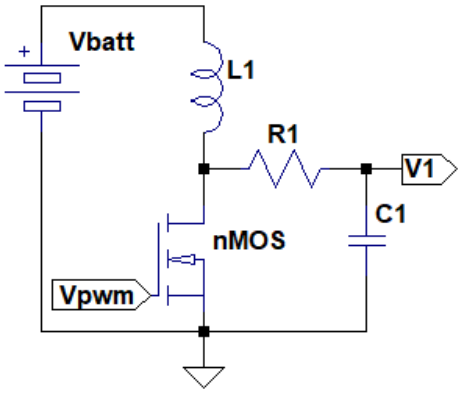

Figure 3.3: First stage of the *Niceclick* circuit

The first stage is reported in figure 3.3. As before, this circuit is made of supply voltage, inductor, along with its parasitic resistance, representing the *Niceclick* transducer, a resistor

and a capacitor, but the latter is connected to ground instead of supply voltage. The resistor is employed in order to limit the current flowing through the inductor and the capacitor. The capacitor gets charged thanks to the current coming from the supply voltage and from the magnetic field of the inductor and exhibits a voltage across it. Such combination of elements is a well-known circuit: a series RLC resonator. So, driving the inductor with a nMOS, controlled by a proper PWM signal, it is possible to obtain, out of such stage, a second order step response. The original flyback diode for protection, not reported, is still present.

Second stage

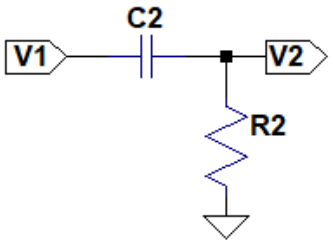

Figure 3.4: Second stage of the *Niceclick* circuit

The second stage is reported in figure 3.4. It is an analog passive first order high pass CR filter which has to delete the DC component of the signal coming from the previous stage and so it is characterized by a low cutoff frequency.

Third stage

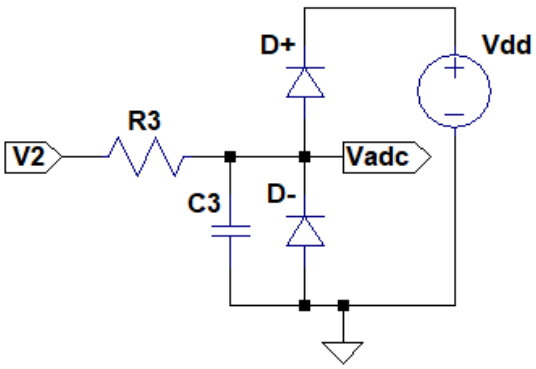

Figure 3.5: Third stage of the *Niceclick* circuit

The third stage is reported in figure 3.5. It is an analog passive first order low pass RC filter. This stage is equipped with a conditioning circuitry, especially a pair of Schottky diodes, because it has to interact with a SAR ADC, which is the device actually performing the measurement, characterized by a limited input range. Such diodes saturate the input voltage of the ADC

whenever necessary in order to guarantee that the input dynamic of the ADC is not exceeded, preventing it from being damaged by undervoltage or overvoltage.

3.2.2 Firmware

Such hardware had to be handled by a proper firmware in order to work, but, according to the principle of code reusability, as much original code as possible was kept.

The process consisted of different steps. To correctly exploit the ADC, a proper algorithm was designed and, as a consequence, the schematic, feature of Infineon Technologies' PSoC Creator, offering an high-level overview of the system as a set of logic blocks, was edited. The code managing input capture was disabled by commenting it out.

In particular, the algorithm, describing the fundamental steps necessary in order to perform sensing, can be represented by means of a flow chart, as reported in figure 3.6.

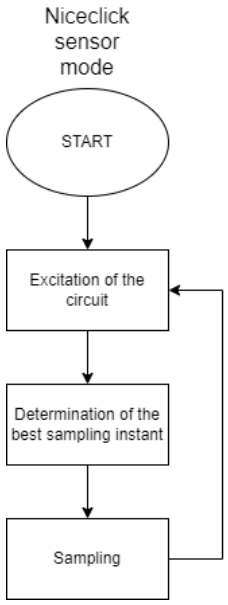

Figure 3.6: Flow chart of *Niceclick* sensor mode of operation

The first and the third steps are trivial, but the second is not: with input capture, sensing took the form of differential information, a difference in time, so two instants which could be used to characterize the system response to position-depending inductance variation. With ADC instead, only a single point of the waveform is taken into account and so it has to be meaningful. The triggering of the conversion of such point has to be accurate for what concerns the timing because, depending on inductance, the waveform changes. In fact, this point should be taken where the waveform exhibits the maximum variation with respect to inductance and, therefore, to the applied force, also to maximize the ADC input dynamic range and so, SNRq. The flow chart turns automatically to the PSoC Creator schematic reported in figure 3.7.

The excitation of the second order circuit is, again, performed resorting to a PWM block, which is actually a counter. It is responsible for the generation of the periodical step response in the sense that its "Period" value, the duty cycle denominator, fixes its frequency, so it acts on the horizontal time axis, and its "Compare" value, the duty cycle numerator, establishes its amplitude, so it acts on the vertical voltage axis. In the schematic, this block is then connected

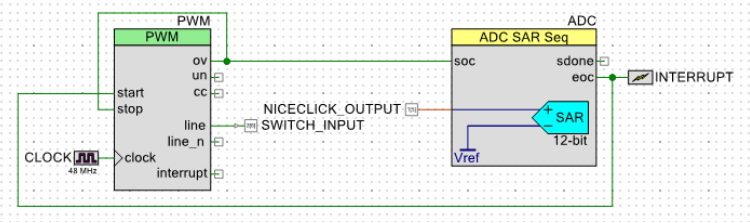

Figure 3.7: Schematic of the *Niceclick* sensor mode of operation

directly to a SAR ADC block (which, as a consequence, can be defined as "hardware triggered"), in particular, the PWM "Overflow" output is connected to the ADC "Start of conversion" input (and to its "Stop" input also, stopping the PWM signal generation and introducing a dependence on the timing of the ADC). Therefore, the PWM block determines the exact time instant in which a conversion has to be triggered. When the counter overflows and stops, a single conversion is started, after acquisition and conversion times have elapsed, the PWM signal generation is started again, but the result of the conversion has to be retrieved from the ADC's output register before being overwritten. In order to do so, for the sake of efficiency, an interrupt is triggered and an ISR is executed, summoning a callback.

The callback performs the reading of the converted voltage value by exploiting a proper API of the ADC and stores it, as a variable, in a specific field of the data structure of the *Niceclick* library, after the clearance of the pending interrupt flag. Such callback is fundamental because it is responsible for the collection of the position-dependent inductance measurement, which is at the base of the force sensing, the main task of the *Niceclick* system and whose value allows the change of its mode of operation.

3.3 Solution evaluation

Once the solution was integrated, a series of tests was carried out on the system in order to evaluate its effectiveness as a sensor. In particular, three tests took place:

- Comparison of position sensing capability between the *Niceclick* system and a reference;
- Identification of sensor transcharacteristic (true force versus displacement) with a dynamometer;
- Interrupt callback effect on CPU load;

3.3.1 Evaluation of position sensing capability

In order to verify the operation of the *Niceclick* transducer as a position sensor, a test consisting of a qualitative comparison against a reference was arranged. In order to carry out such test three elements were needed:

- The Device Under Test (DUT), so the *Niceclick* transducer itself;
- A suitable reference;
- An appropriate data acquisition system;

The choice of the reference had to fall necessarily on a standard position sensor, which had to be sufficiently accurate in order to properly fulfill the task. A Single Mode Fiber red laser (655 nm) by KEYENCE, an IL-S025 with an IL-1000 amplifier, acting as a position sensor, was selected.

The data acquisition system for such experience consisted of a combination of both hardware and software. For what concerns the hardware, the acquisition of data provided by the laser sensor was implemented thanks to a M-SENS 8plus by IPETRONIK, a module performing analog measurements with sensor excitation capability. The data coming from the transducer was collected by its evaluation board instead. Such information had then to be transmitted to PC in order to be diplayed in a human-readable format, but both the module and the *Niceclick* board are capable to communicate only through CAN bus and not USB, so an additional module, always provided by IPETRONIK, the IPEcan FD Pro was exploited to perform the conversion. This device is a 2-channel USB2CAN FD/LIN interface which converts data from/to USB. In order to support both the sensors, an *ad hoc* test bench was built. In particular, the laser sensor is hanged on and aligned with the *Niceclick* transducer, with the beam perpendicular to its cursor and so parallel to its sensing direction. Pressing or releasing the test bench element, hinged on a side and placed between the sensors, causes an increase or decrease in the relative position between cursor and stator and so of the value sensed by the DUT, provoking, simulataneously, a variation of the distance from the laser to the test bench element. The arrangement for such test is depicted in figure 3.8.

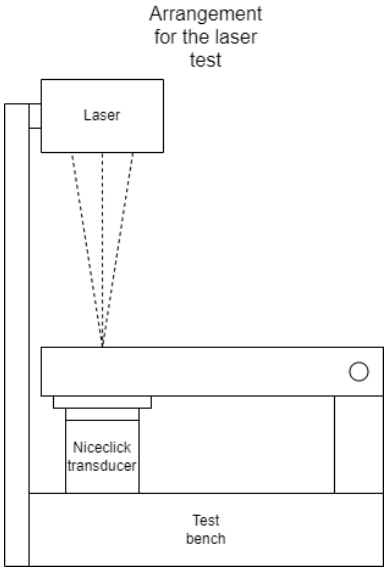

Figure 3.8: Approximate arrangement for the position sensing capability test

Set the hardware, for what concerns the software, the IPETRONIK IPEmotion, a DAQ software, was adopted. Before starting the acquisition, IPEmotion has to be configured on the base of the characteristics of the sensors, specifying, in particular, their input-output characteristic and the frequency for the sampling of the data they provide.

Starting the acquisition and the real-time visualization allows the observation of the evolution in time of the sensors' output, in particular, as can be seen in figure 3.9, pressing on the test bench element, it is possible to note an increase in the distance sensed by the laser, in parallel, the displacement decreases, causing an increase in inductance and resulting in a lower resonant frequency. In time domain, the duration of the first oscillation increases, so at the sampling time

instant, voltage is higher and, therefore, the value at the ADC output, provoking the *Niceclick* system curve to go up. Trivially, the opposite happens releasing it.

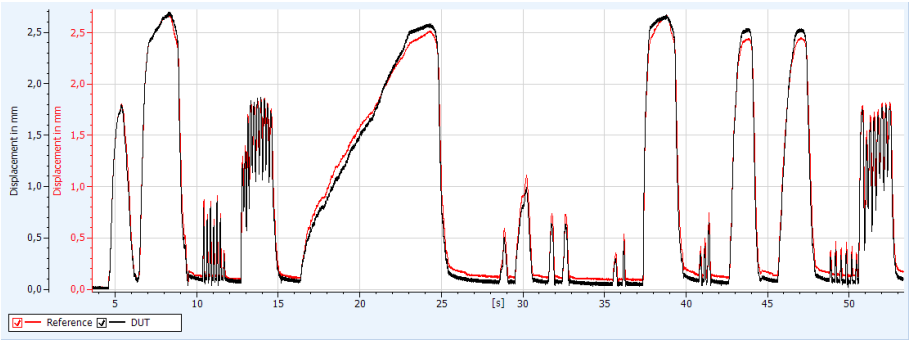

Figure 3.9: Comparison of position sensing capability between laser and *Niceclick* system

A series of qualitative observations can be expressed: the DUT trace, which indicates the relative position of the cursor with respect to the stator, follows the reference one close up, both in x-axis and y-axis. The approximately overlapping of the curves is simply achieved by applying linear compensations like offset and gain to the values supplied by the transducer. In broad terms, the measuring system appears to be good in terms of responsiveness, but also in terms of linearity and it is evidently comparable to a commercial laser sensor, but there is a difference which has to be highlighted: the cost and so the cost-quality ratio favours slightly the DUT. Such analysis allowed the traits of the *Niceclick* system and the validity of the ideas at its base to be half-seen.

3.3.2 Evaluation of input-output characteristic

In order to study more in depth the sensor and obtain quantitative information about its properties, another evaluation was carried out. Such test was aimed at identifying the input-output characteristic of the *Niceclick* transducer. In detail, a force was applied to it and measured thanks to a dynamometer, putting it in relation with the value provided by the *Niceclick* system.

Also in this case, the data acquisition system was made of both hardware and software. The hardware taken advantage of for this attempt consisted of a digital dynamometer, a Sauter FK50, and, again, a M-SENS 8plus, a CAN-USB adapter and a PC for the collection of information coming from the combination of board and transducer, implementing the DUT. For what concerns the software, IPEmotion for the acquisition of the data and Microsoft Excel for their processing were exploited. For such experience, the already available test bench was employed, but in place of the laser sensor, the dynamometer was hanged on and aligned with the DUT thanks to a support. The overall setup is reported in figure 3.10.

Acting on the various knobs of the support, it is possible to control accurately the position of the dynamometer and so the direction along which the force is measured. Since the fact that the sensor is bounded to the test bench, it would be the dynamometer that physically applies the force and, in order to do so, a long and stiff tip was attached to it. The support was placed so that the dynamometer is almost perpendicular to the test bench element acting on the cursor of the transducer, so the angle caused by the particular arrangement was considered negligible, being such analysis only qualitative. Therefore, by properly operating the knobs, it is possible to move up and down the instrument, respectively decreasing or increasing the force applied by the tip to the transducer.

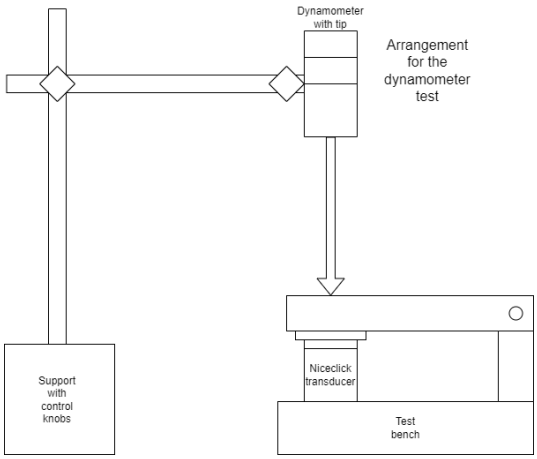

Figure 3.10: Approximate arrangement for the dynamometer test

The test consisted of the application of an increasing force from 0 N to 10 N and followed by a decreasing one from 10 N to 0 N, coupling every force value measured by the dynamometer and read on its display to the corresponding value provided by the *Niceclick* system and transmitted, thanks to the CAN-USB adapter, to the PC in order to be reported by IPEmotion in a human-readable format. So, the data acquisition system it is not fully automatic. In order to achieve more relevance from a statistical point of view, five full force spans were performed, in order to simulate the multiple readings method.

The data, collected by hand, completed a table of a spreadsheet and they were exploited in order to plot two graphs, for the increasing and decreasing force ranges. They are reported in figures 3.11 and 3.12.

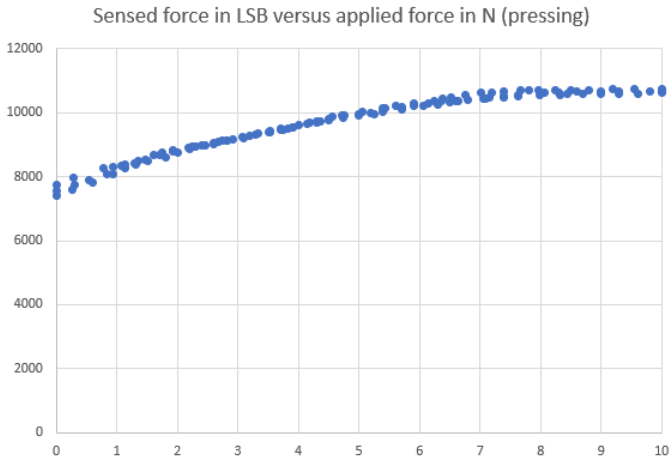

Figure 3.11: Input-output characteristic when an increasing force is applied

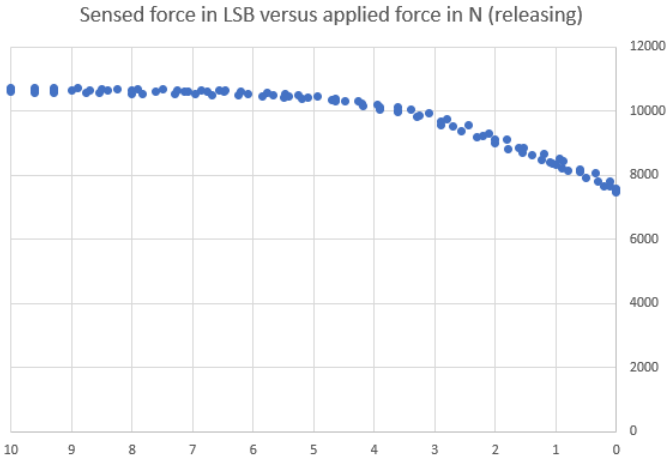

Figure 3.12: Input-output characteristic when a decreasing force is applied

At least three observations come to mind: first, even though only five series of readings were carried out, it is evident that the measurements seem to be, in broad terms, quite reproducible. Second, the transcharacteristic appears quite linear, especially in certain areas. Indeed, both graphs can be divided in two parts: a linear and a non-linear ones. In the first plot, it is evident that the value provided by the *Niceclick* system increases linearly until 6 N, then a sort of saturation takes place and, increasing the applied force, the sensed value does not increase anymore. In the opposite way, something similar characterizes also the other graph. A possible explanation could be that the elastic element of the sensor introduces a non-linearity, a saturation, due to the fact that the elastic force is linear only in correspondence of a limited displacement. Further studies have to be carried out, but, again, there are clues that the system is promising.

3.3.3 Evaluation of interrupt callback effect on CPU load

After the integration of the ADC-based measurement method and the verification of its effectiveness, the effect on the system was investigated. In particular, the contribution of the callback code (and not the overhead of the whole ISR) to the CPU load was the goal of the test. Such analysis was carried out in terms of time, especially the time interval that the CPU was spending executing the callback of the interrupt with respect to the cycle time of the system. In order to evaluate these periods, there are different methods so, supposing to measure the effect of a task, it is possible to:

- Use a pin of the microcontroller and a DSO;
- Use a timer;
- Use a free running counter;

The first method, pin-based/DSO-based, has different implementations. Using a single pin, it is possible to work in two modes: differential or absolute. In differential mode, the pin is complemented only once at each cycle, first, with the task that has to be measured disabled and, second, with the task enabled. In both cases, applying a DSO probe to the pin, a square wave is displayed, but characterized by a different period: shorter in the first case and longer in the second. The effect of the task, in terms of time, on CPU load is the difference between the two. In absolute mode, the pin is complemented twice at each cycle, once just before the start of the task and once just after the end of the task. In that case, applying a DSO probe to the pin, the displayed waveform duty cycle is not equal to 50 % and so a square wave, but less. The CPU load is exactly the duty cycle of the waveform. However, more than one pin can be used also.

Pin-based methods are characterized by ease of comprehension thanks to the use of the DSO and are quite accurate, but there are also other ways, taking advantage of timers and counters which perform better from an accuracy point of view, but the data obtained by such methods can not be displayed directly like in case of a DSO. Moreover, such methods provide data in a format on which it is possible to carry out statistical analysis. Using a single free running counter, counting at its maximum possible frequency, it is possible to read the counter value before and after the task. By taking the difference and through other mathematical manipulations, it is possible to obtain the CPU load.

So far, CPU load was measured resorting to direct methods, measuring the percentage of time that CPU spends executing the task, but, conversely, the percentage of time spent by the CPU not executing such task can be evaluated too. The test was carried out resorting to CW-Analyzer, a software for the decoding and visualization on a PC of the data coming from the *Niceclick* board through CAN-USB adapter, and Excel, exploiting the DSO only for the first method. Adopting the first method, a CPU load equal to 9,02 % was obtained, as it is possible to see in figure 3.13.

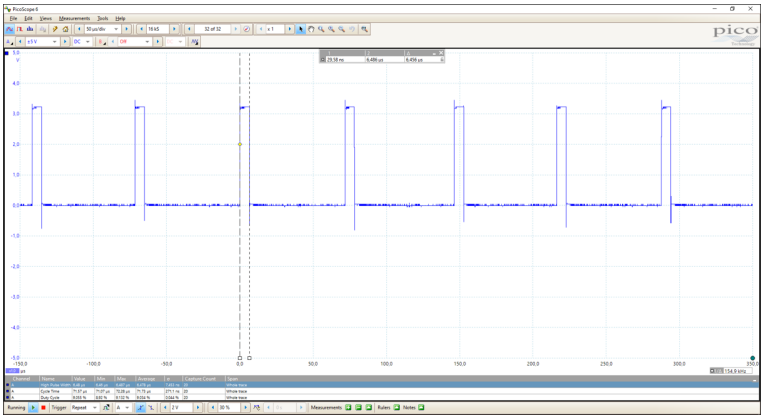

Figure 3.13: CPU load toggling a single pin

A similar result, ranging from 7,76 % and 9,19 % was obtained thanks to a free running counter through proper mathematical manipulations.

Chapter 4

Niceclick system model

In order to carry out an in-depth analysis of the system, a series of models was built. Such models have different secondary goals, but the primary is to study its behavior when the components which make up the core of the circuit are not ideal, so in case of manufacturing tolerances, influence quantities, like temperature and humidity or, more in general, when a different set of parameters is exploited. Moreover, these factors differently affect the system, so it is important to recognize the ones which influence it the most.

So, a proper model allows the behavior of the system to be predicted or reproduced and tests to be carried out, without acting on the circuit directly. First, a model was built up in LTspice to study the transient response of the system and to perform Monte Carlo simulations, based on different sets of parameters. Then, a model was realized in Simulink, and so in MATLAB, to verify the correctness of the mathematical model which was set up to describe the system. Finally, another mathematical model was built up in MATLAB in order to further improve the *Niceclick* system model. Supposing to split the circuit in three independent stages, particular attention has to be paid to the first one, which is the key element of the circuit and the overall system, because it is responsible for the generation of the periodical second order step response shown in figure 4.1.

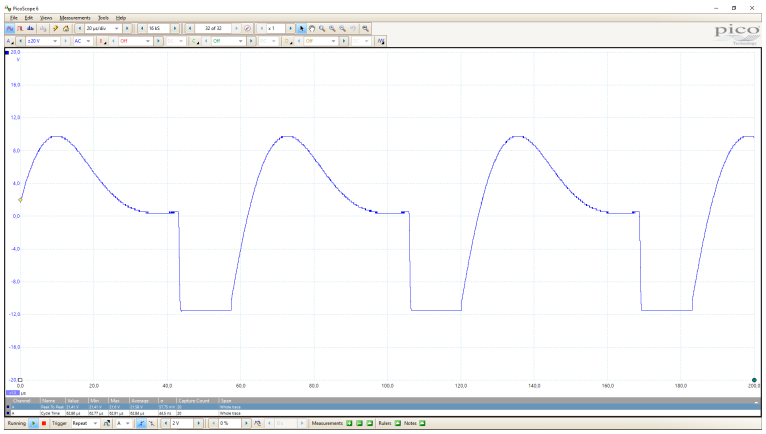

Figure 4.1: AC-coupled second order step response at the output of the first stage

4.1 LTspice XVII circuital model

Resorting to the schematics of the *Niceclick* system it was possible to create a circuital model, made of components with nominal values. The *Niceclick* transducer was modeled as an inductor with a series resistor, as reported in its datasheet 4.2.

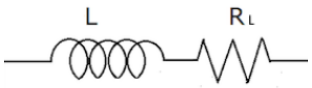

Figure 4.2: *Niceclick* transducer model (courtesy of TRAMA ENGINEERING)

Both the low-side switch and Schottky diodes models were obtained from their respective manufacturer's website. For convenience, the model was divided in its three fundamental stages, as reported in figure 4.3.

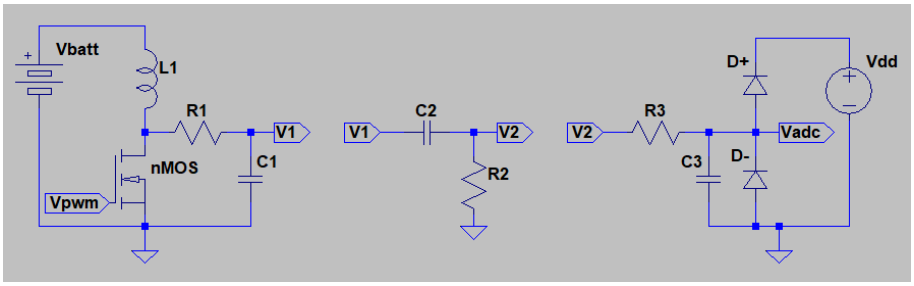

Figure 4.3: Circuital model of the *Niceclick* system

Transient simulations were carried out in order to verify the model behavior in time domain with respect to the true one, which was investigated thanks to a DSO. In particular, referring to the output voltage of the first stage, as in 4.1 and 4.4, it is possible to note how the two

waveforms, consisting of the first oscillation of the step response of a second order system, a series resonator, are similar, but not identical, mainly due to components tolerances or models inaccuracies, especially concerning the transducer one.

Therefore, in order to evaluate the effect of tolerances, a series of Monte Carlo transient simulations was executed. In particular, the nominal tolerances, retrieved from the schematics, of resistor, inductor, capacitor and supply voltage were set as parameters for the analysis. As shown in figures 4.4, 4.5, 4.6, 4.7 and 4.8 at last, the effect of the tolerances of each component type and the overall effect on the first stage output voltage were evaluated, highlighting the fact that the variability characterizing supply voltage and reactive elements has an higher one with respect to passive elements and so particular attention has to be paid to them.

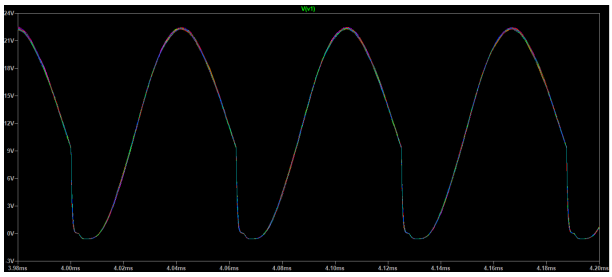

Figure 4.4: Monte Carlo simulation with 10 % tolerance on resistor: it makes almost no difference with respect to the ideal case

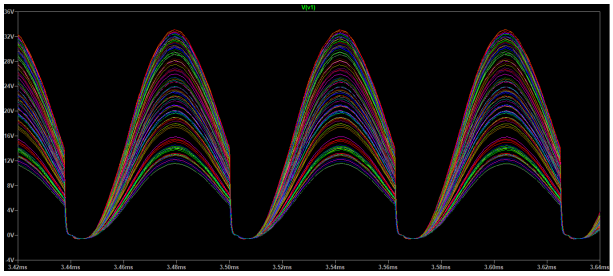

Figure 4.5: Monte Carlo simulation with 50 % tolerance on supply voltage

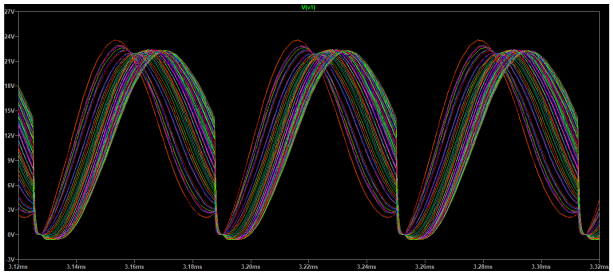

Figure 4.6: Monte Carlo simulation with 20 % tolerance on capacitor

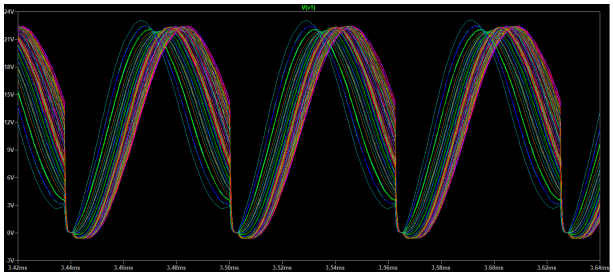

Figure 4.7: Monte Carlo simulation with 20 % tolerance on inductor

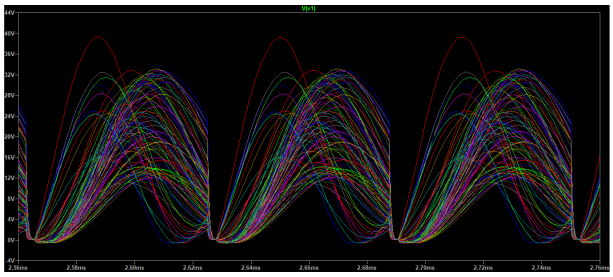

Figure 4.8: Monte Carlo simulation with tolerances on all elements

4.2 Simulink mathematical model

The circuital model realized in LTspice is useful in order to obtain an high-level representation of the system behavior and allows various types of analysis to be done, but it is not ideal for the achievement of an extensive comprehension of the underlying working principles. So, the schematics were studied carefully and, thanks to circuit theory regarding first and second order systems, a mathematical model, focused on the first stage only, was obtained and implemented in Simulink [9].

It is based on the fact that the first stage can be interpreted as a time-varying and order-varying circuit, because due to the presence of the low side switch (supposed ideal in the following), driven by a proper PWM signal, the circuit behaves as a series RLC resonator when opened (open circuit) or as two independent RL and RC circuits when closed (short circuit).

Therefore, for both switch configurations, mathematical formulas are readily available in order to approximate the time response of the system. In particular, when the switch is off (opened), the series RLC resonator behavior can be described mathematically by a homogeneus second order differential equation with constant coefficients, whose canonical form is:

$$\frac{d^2x}{dt^2} + 2\alpha\frac{dx}{dt} + \omega_0^2 x = 0$$

Where:

$$x = i$$
$$\alpha = \frac{R}{2L}$$
$$\omega_0 = \frac{1}{\sqrt{LC}}$$

And:

- α is the damping constant in inverse seconds;
- ω_0 is the resonance pulsation in radians per second;

Then, it is possible to refer to the characteristic equation in the s variable:

$$s^2 + 2\alpha s + \omega_0^2 = 0$$

Where the roots, the natural frequencies $s1$ and $s2$, are:

$$s_1 = -\alpha + \sqrt{\alpha^2 - \omega_0^2}$$
$$s_2 = -\alpha - \sqrt{\alpha^2 - \omega_0^2}$$

So, it is possible to distinguish among three cases:

- Overdamped circuit if $\alpha > \omega_0$;
- Critically-damped circuit if $\alpha = \omega_0$;
- Underdamped circuit if $\alpha < \omega_0$;
- Undamped circuit if $\alpha = \omega_0$;

Calculating the values for the damping constant and resonant frequency with the values retrieved from the schematics, it is possible to state that the circuit is underdamped, so the natural frequencies are complex conjugated. So, defining:

$$\beta = \sqrt{\omega_0^2 - \alpha^2}$$

The solution is:

$$x(t) = e^{-}\alpha t[A_1 cos\beta t + A_2 sin\beta t]$$

Where $A1$ and $A2$ have to be determined by imposing the initial conditions $x(0)$ and $\frac{dx}{dt(0)}$.

When the switch is on (closed), the circuit can be divided ideally in two independent subcircuits due to the presence of the implicit short circuit. So, a part is a first order RL circuit and the other is a first order RC circuit, both describable thanks to well-defined mathematical formulas. Both are characterized by a specific time constant and, in case of a constant generator, their evolution in time is approximated thanks to a non-homogeneous first order linear differential equation with constant coefficients, whose canonical form is:

$$\frac{dx(t)}{dt} + \frac{1}{\tau}x(t) = \frac{x_p}{\tau}$$

Where:

- RC circuit with $\tau = RC$ and $x_p = V_s$;
- RL circuit with $\tau = \frac{L}{R}$ and $x_p = I_s$;

Whose solution is:

$$x(t) = [x(0) - x(\infty)]e^{-t/\tau} + x(\infty)]$$

Where:

- $x(0)$ is the initial condition in volt/ampere;
- $x(\infty)$ is the final condition in volt/ampere;

Thanks to the mathematical formulas concerning the series resonator and the RC circuit, the time response of the current at the output of the first stage is obtained, but remembering the *Niceclick* system operation, based on the conversion from an analog voltage value to a digital one by an ADC, voltage is more relevant for the purposes of the study indeed. The output of the first stage is the voltage drop across the capacitor, which is characterized by a well-known constitutive relation correlating current and voltage:

$$I_C = C\frac{dV_C}{dt}$$

And inverting:

$$V_C = \frac{1}{C}\int I_C$$

Therefore, by integrating the current and dividing by the capacitance, the voltage at the output of the first stage, caused by the application of a PWM signal at its input and depicted in figure 4.9, is achieved. However, also in this case such waveform differs from the true one in 4.1, again, due to non-idealities and inaccuracies. The Simulink model, based on the components values retrieved from the schematics, is reported in figure 4.10.

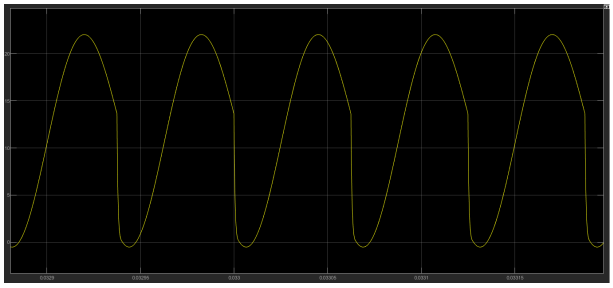

Figure 4.9: Voltage at the output of the Simulink model

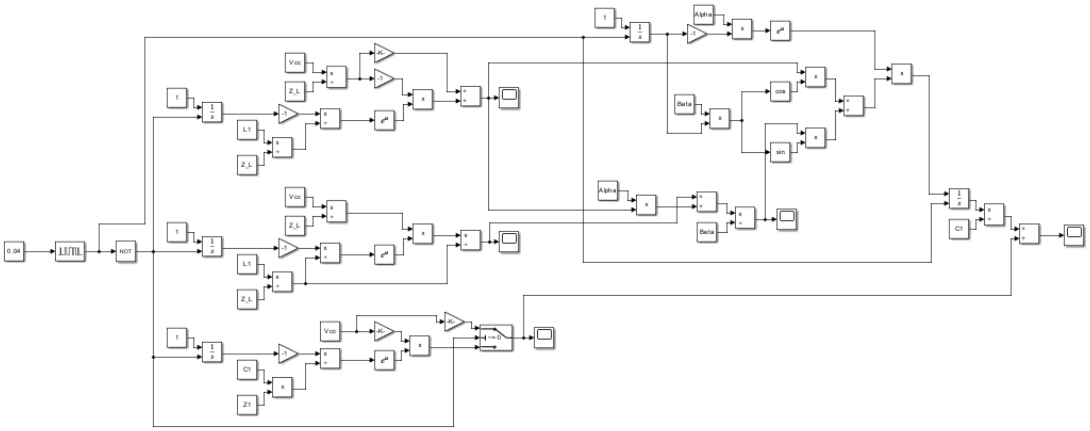

Figure 4.10: Mathematical model of the first stage of the *Niceclick* system

4.3 MATLAB mathematical model

As already pointed out, the Simulink one allows a mathematical model, which only approximates the behavior of the *Niceclick* system to be obtained, and the differences are noticeable by inspecting the circuit with a DSO. Moreover, it models the time response of the circuit, the first stage only, as the output of an high-level entity which does not specify so much about its true input-output relation, at least in terms of equations, and does not take into account the other stages and, in particular, both the timing and the output of the ADC, whose value depends on the duty cycle of the PWM signal.

So, an in-depth analysis to fill these gaps was carried out through the realization of a series of more and more inclusive models, starting from the Simulink one. However, in spite of adjustments and approximations, made in order to try to take into consideration all that is needed for the achievement of a high level of accuracy, the output of the model, reported in figure 4.11 and 4.12, still does not follow exactly the one in 4.1 for the reasons as above.

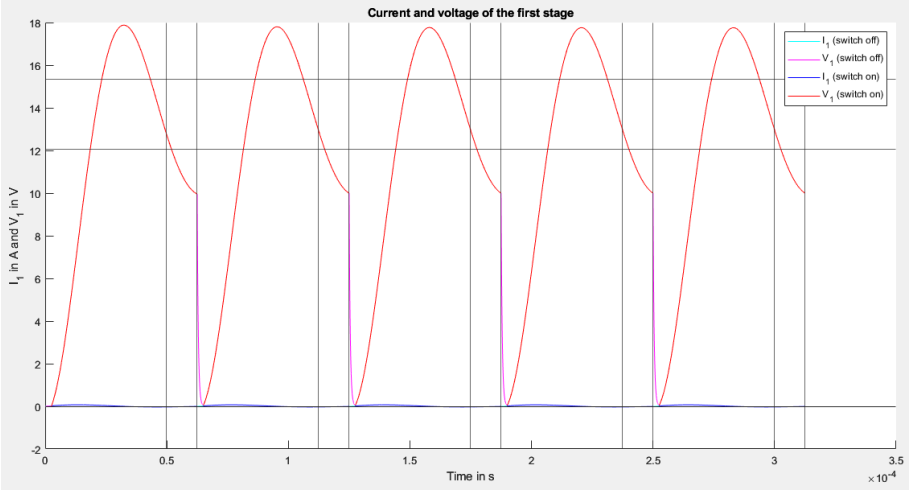

Figure 4.11: Voltage and current at the output of the MATLAB model with inductance saturation: vertical lines indicate start and end of conversion for each period and horizontal ones the conditioned input dynamic of ADC

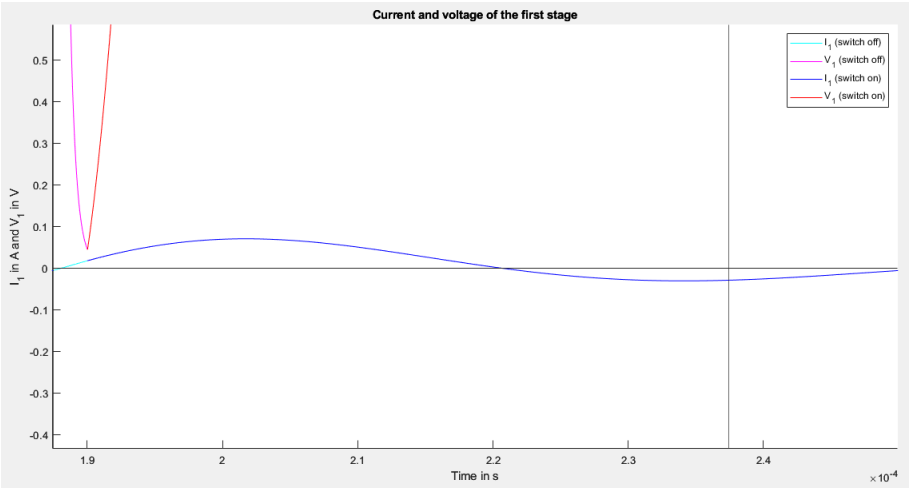

Figure 4.12: Detail of the current at the output of the MATLAB model with inductance saturation

Chapter 5

Data acquisition and processing

The chapter focus on the collection of information coming from the *Niceclick* system, in compliance with predefined conditions, and the processing of the latter in order to deepen the knowledge of the system and to pave the way for further developments.

The first part deals with the completion of a series of data acquisitions ("force patterns" in the following, because they are the curves traced by the time-varying quantity under test, which is the sensed force) to allow a detailed study of the features of the forces applied by a human being to the sensor and try to find out similarities with well-known mathematical concepts in order to ease their characterization.

Then, for statistical relevance, such analysis was carried out on a wider population, the TRAMA ENGINEERING employees, in order to build up a database collecting different force patterns for each person.

Finally, an algorithm for pulse detection, operating on the database, was designed. For "pulse" the curve traced by the sensed force as a result of a single pressing followed by a releasing of the cursor is intended. Indeed, a force pattern is a sequence of the latter.

5.1 Preliminary analysis of force patterns

In order to analyze the interaction between a human being and the sensor in different scenarios, a series of force patterns was applied to the *Niceclick* transducer and the force values out of the system were collected for processing.

First of all, a proper data acquisition system was set up. The DUT, connected to its support board, was bounded to the well-known test bench and the board itself was connected to a PC, through a CAN-USB adapter. To make possible the acquisition of information, IPEmotion was exploited. Then, the sensor was subjected to:

- A periodical force applied with a short period;
- A periodical force applied with a medium period;
- A periodical force applied with a long period;

The force values provided by the *Niceclick* board were displayed in real-time, as shown in figure 5.1 and finally exported in MATLAB in order to allow processing to take place.

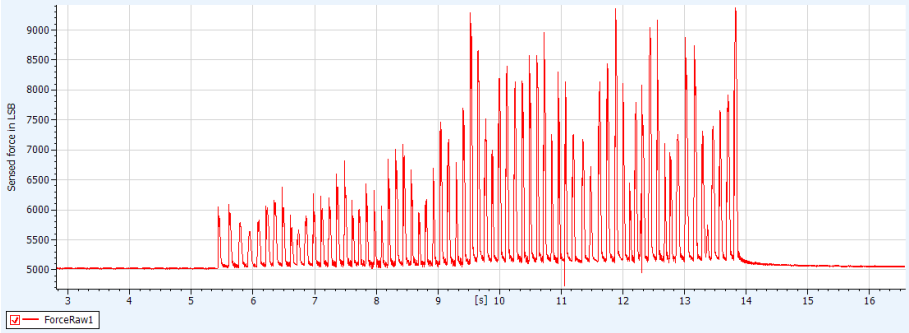

Figure 5.1: Example of sensed force acquisition

The idea at the base of the processing is to find out a set of features which allows a single pulse to be compared against each other making up a force pattern and so characterize the force pattern itself somehow. Such need arises from the fact that a human being could apply infinite force patterns, each different from the other, and also the pulses of a force pattern may differ a lot.

So, in order to handle this variability, a statistical characterization was carried out in order to find a sort of average pulse. However, for the purposes of finding a proper algorithm to do so, it is important to note that force patterns, are, technically, sequences of digital values: only the human eye could identify such data as a series of pulses. Therefore, in order to treat a force pattern as a sequence of pulses, a criterion for correctly recognising them has to be found. The flow chart, representing the algorithm, is reported in figure 5.2.

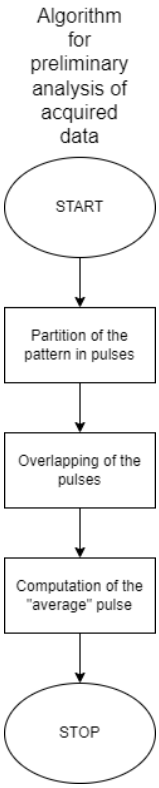

Figure 5.2: Flow chart of the algorithm for the analysis of a force pattern

In order to accomplish a trivial analysis in MATLAB, the local minima were searched and taken as the two ends of a pulse. Then, for the calculation of the average one, these pulses have to be arranged due to the fact that their duration varies. Again, for the sake of simplicity, each pulse was overlapped with respect to a common axis of symmetry which is placed in the half of each pulse, performing a sort of centering somehow. So, the mean pulse can be computed effortlessly as reported in figure 5.3.

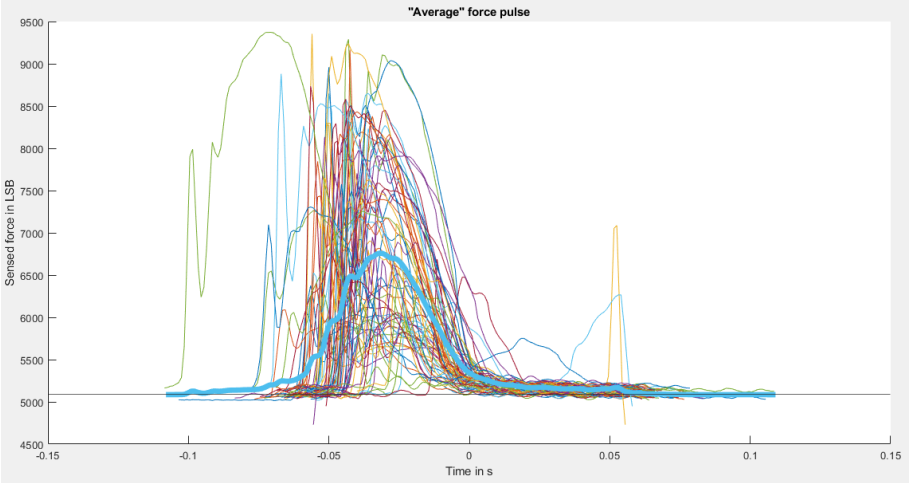

Figure 5.3: "Average" force pulse

It is important to observe that, in spite of the ease of the algorithm, the average pulse resembles a statistical concept: the gaussian function. If this statement holds true, it would be a point in favour, because its properties are well-known and it can be accurately described by a limited set of parameters, mean μ and standard deviation σ, definitely simplifying the study. In order to test the validity of such conclusion, a comparison in MATLAB was implemented. Starting from the equation of the gaussian function:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

Both the values of $\frac{1}{\sigma\sqrt{2\pi}}$ and μ are retrieved from the properties of the mean pulse, especially its amplitude and central value respectively, and extracting the value of σ from the first, it is possible to obtain a gaussian function which almost overlap the average pulse, as in figure 5.4, confirming, in broad terms, the correctness of the observation as above.

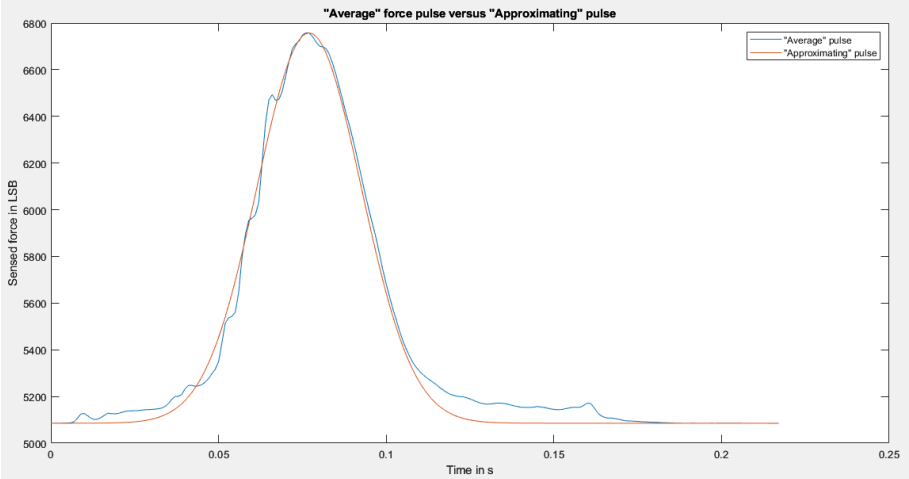

Figure 5.4: Gaussian approximation of the "average" pulse

5.2 *Niceclick* database

In order to collect information to allow further development, the need of a database showed up. Such database should contain a collection of measurements supplied by the *Niceclick* system to perform statistical analysis or any other kind of post-processing based on these force patterns. To reach the goal three main steps were taken:

- Definition of the specifications;
- Setup of the data acquisition system;
- Post-processing of information;

For what concerns specifications, the building of a database, collecting the data of TRAMA ENGINEERING employees had to be done in order to carry out a statistical survey on the sequences of pulses obtained through the application of forces to the *Niceclick* sensor, in order to analyze in-depth force patterns and constitute a point of departure for the development of algorithms. To do so, the database has to record pulses characterized by well-defined properties, therefore observing certain criteria. Such database should theoretically take into account the variability which inherently affects the force pulses applied by human beings. So, it should contain, at least, for each employee:

- A sequence of pulses applied spontaneously;
- A sequence of pulses applied quickly;
- A sequence of pulses applied slowly;

However, a non-trivial consideration has to be made: how a user applies the force may differ slightly among the three cases. In particular, a user, performing a sequence of pulses may or may not leave its finger on the transducer between two consecutive ones, so preventing or not

the cursor from reaching its resting position. The sequences have to be made applying the force with a single finger, the index.

After the definition of the specifications, a proper test bench, integral part of the data acquisition system, had to be built. Due to its versatility, the already available test bench was employed. In particular, such test bench was already equipped with a force sensor, a load cell based on piezoresistive strain gauge, by TE Connectivity. On such device, the *Niceclick* transducer was placed by means of double-sided tape and, on top of it, a mechanical push button switch was put, again, thanks to double-sided tape. Finally, in order to definitely reproduce the perception provided by a common mechanical button, a plastic element was glued to the switch. So such system implemented a stack of three different sensing elements plus the plastic one on top of it. Such stack lacked stability and so, the test bench was modified with the addition of a metallic frame in order to better support the plastic button. Moreover, being the *Niceclick* transducer not characterized by a monolithic structure, particular attention had to be paid also to the cables, whose arrangement can condition the measurements by exerting accidental forces on the sensing elements. The sensing stack is reported in figure 5.5.

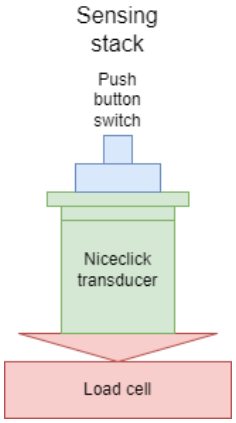

Figure 5.5: Three-element sensing stack

Therefore, the system exploits a elementary operating principle: whenever a force is applied to the plastic element, it is absorbed by three elements mainly:

- The push button switch;
- The elastic element between the cursor and the stator of the *Niceclick* sensor;
- The load cell;

The push button is a standard 3 N switch, so when the force exceeds this first threshold, which is actually higher, in absolute terms, thanks to the dispersion of the force due to the other elements of the stack, it toggles from its resting position to its pressed position, producing a noticeable haptic effect (defined “Click” in the following). Being the switch connected between supply voltage thanks to a pull-up resistor and ground, once pressed, its output is driven to ground. Then, when the force drops till a second threshold, lower with respect to the first, from its pressed position the switch goes back to its resting position, another haptic effect is produced (defined “Clack” in the following) and the output of the switch is again pulled up to supply voltage. The toggling of its output allows the instants in which the two haptic feedbacks are

generated by the switch itself and the time interval in-between to be identified quite accurately, stating how long the pressed state lasts and vice versa. The *Niceclick*, which is limited to operate in sensor mode, merely senses the displacement, which although is proportional to the applied force. Instead, the load cell measures it indeed.

Once the test bench was built, the rest of data acquisition system, depicted in figure 5.6, had to be set up. Again, the accumulation of the data coming from the DUT relies on IPEmotion along with the M-SENS 8plus. In particular, information coming from force sensor and switch are directly acquired by the latter and transmitted through CAN bus to a PC, while the data provided by the transducer are processed by its board and only then transmitted to the PC, taking advantage of CAN bus too. In order to make the PC capable to communicate on two different CAN buses simultaneously, an IPEcan FD Pro is exploited for the conversion from/to USB. Again, IPEmotion had to be correctly set in order to acquire data. In particular, attention had to be paid to the setting of the input-output characteristic of the sensors and the frequency at which their output is sampled.

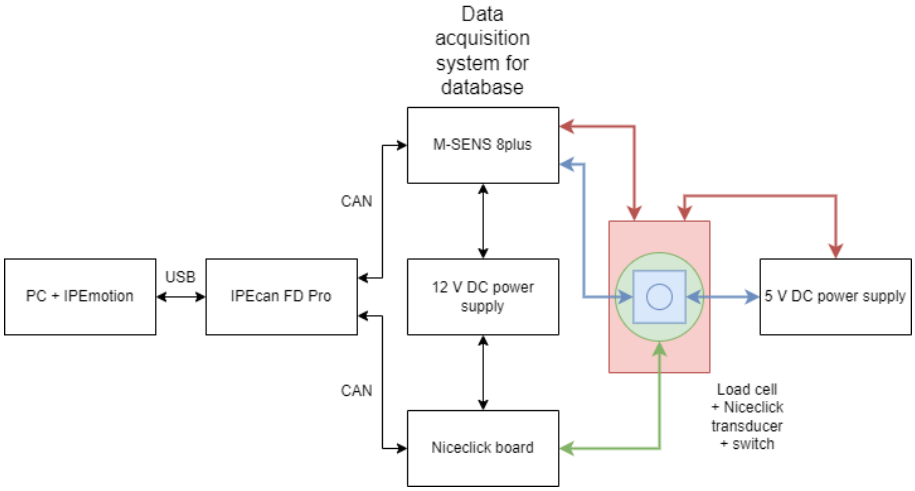

Figure 5.6: Data acquisition system for the building of the *Niceclick* database

Finally, one by one, each TRAMA ENGINEERING employee did his part by applying:

- A spontaneous sequence of about ten pushes, taking care of lifting the finger from the button;
- A spontaneous sequence of about ten pushes, taking care of not lifting the finger from the button;
- A fast sequence of about ten pushes;
- A slow sequence of about five pushes;

Examples of such data acquisitions are reported in figures 5.7 and 5.8, in which it is possible to note the difference, in terms of sensed force, between the first two cases as above. In particular, in 5.8, there is a sort of noticeable residual force between two consecutive pulses which is not present in 5.7.

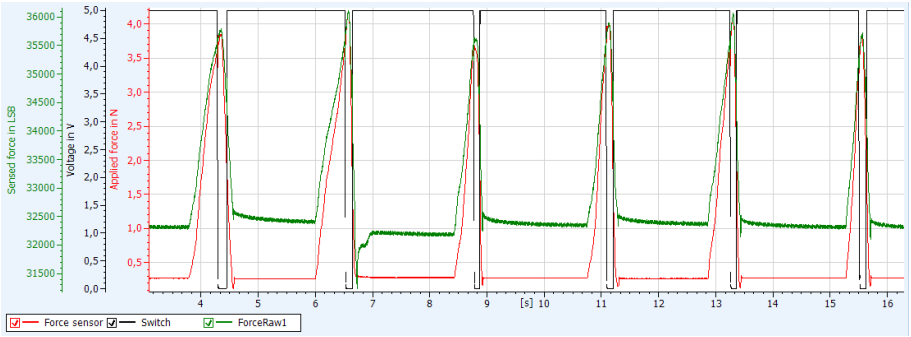

Figure 5.7: Example of data acquisition in which the finger was lifted between pulses

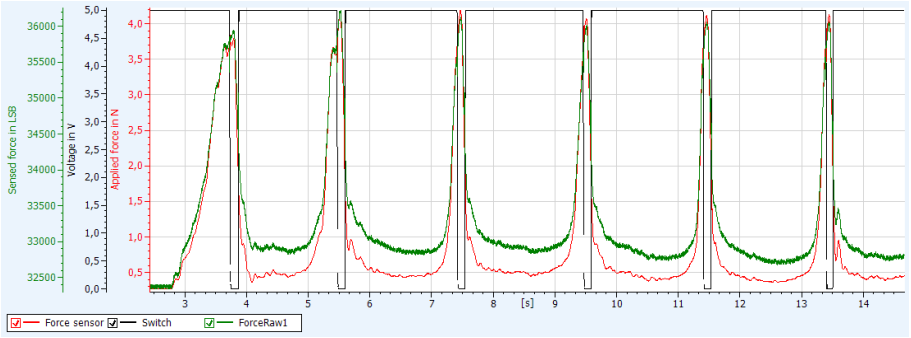

Figure 5.8: Example of data acquisition in which the finger was not lifted between pulses

5.3 Pulse detection algorithm

After the acquisition of the data was accomplished, the post-processing of such information took place. The purpose of manipulating the data was to find an algorithm, suitable for an embedded system, capable of analyzing the force patterns and performing a sort of pulse detection. This task makes it similar to face detection algorithms. These algorithms find a series of reference points which allows a face to be distinguished among everything else. Also in this case, in order to recognize a pulse in the sequence of points which makes up a force pattern, a set of conditions had to be met. These conditions were determined by observing the patterns of the *Niceclick* database.

In particular, a single ideal force pulse, as in figure 5.9, can be defined as such if certain points of reference can be identified:

- A time interval from the starting point of the pulse to the “Click” instant;
- The “Click” time instant;
- A time interval from the “Click” instant to the “Clack” instant;
- The “Clack” time instant;
- A time interval from the “Clack” instant to the ending point of the pulse;

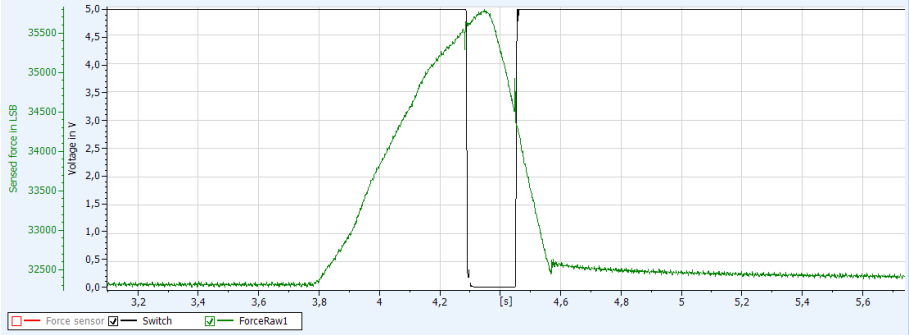

Figure 5.9: Detail of an acquired force pulse

It is important to remember that both the "Click" and "Clack" haptic feedbacks are provided by the push button switch and not by the *Niceclick* system itself.

During the first time interval, the force value moves from its resting position, whatever it is, and increases or decreases with a slope which depends on the applied force: theoretically, the average derivative should be positive. Whenever the "Click" threshold is exceeded, at the "Click" time instant, the "Click" perception is delivered to the user. Then, the time interval from the "Click" to the "Clack" occurs. Again, the force value can exhibit almost any kind of variation and, whenever the "Clack" threshold is exceeded, at the "Clack" time instant, the "Clack" perception is delivered. Finally, during the last time interval, the force value goes back to its resting position: theoretically, the average derivative should be negative.

However, a single ideal pulse is not so relevant for the purposes of the analysis and the algorithm should operate on a series of pulses, like the force patterns that were acquired for the database. So, in this case, an additional time interval, which is not really part of any pulse, but rather separates two consecutive ones, can be pinpointed. Moreover, two situations can take place: thanks to the non-negligible mechanical time constant of the sensing stack, due to the switch, the elastic element, the load cell and other mechanical non-idealities, the time interval between two consecutive pulses, if the applied force stops, may be or not sufficient to let the system reach its resting position and so the sensed force value, after the "Clack". Therefore, the starting force value of the next pulse can be the resting one or a higher one, like if a residual force is still applied, as already showed in 5.8.

In order to obtain a serviceable algorithm, it is important to take care of another aspect: in order to accurately find points of reference on the force pattern, noise has to be taken into account because it may lead to misbehavior. So, for the purposes of coping with noise and its consequences, first of all, the force pattern has to be filtered. The introduction of a filter eases the recognition of the reference points, but has a drawback: it causes the introduction of a delay which may be or not negligible and require a sort of compensation because it could condition the timing of the system.

Then, the core of the algorithm, so the capability of detecting the pulses making up a force pattern, has to be determined. Since the fact that force measuring system is affected by noise and force patterns applied by human beings are anything but ideal, where, for "ideal", sequences of pulses made of monotone segments only are meant, it is difficult to design an algorithm which works locally. Moreover, remembering that the *Niceclick* system is embedded and, as such, can rely on limited hardware resources and has to operate in real-time, this analysis can not be done during the post-processing and has to be completed quickly enough in order to ensure the real-time constraint and so the responsiveness of the system. In order to meet the requirements

imposed by the context, firstly, the force pattern, made up of a finite sequence of samples, is subdivided in equal parts and each segment is averaged. Then, the mean of a segment is compared against the one of the previous: if it is higher, the force pattern slope within these two segments is strictly positive, strictly negative if lower or null if equal. As a consequence, local differential information, a sort of unrefined derivative, is obtained. By associating a suitable numeric value to the result of each comparison, it is possible to turn the force pattern from a sequence of points, varying from a minimum to a maximum and quantifying the force, into a sequence of points which can assume three values only and which roughly indicates the evolution in time of the derivative, as in figure 5.10. Therefore, a sort of discretization takes place.

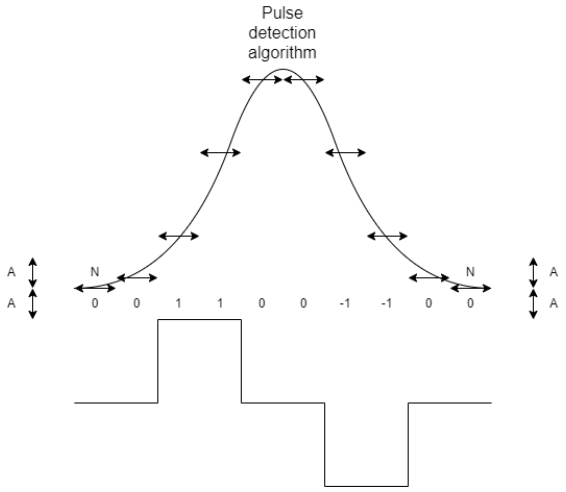

Figure 5.10: Concept at the base of the pulse detection algorithm

Thereafter, a pulse would consist of, in the best possible case, a segment in which the derivative is positive and a segment in which the derivative is negative. As a consequence, the shape of the pulse would be triangular, but it was verified experimentally that a trapezoidal shape would be more realistic.

The effectiveness of the pulse detection algorithm obviously depends on the length N of the equal segments in which the force pattern was subdivided and so it has to be considered as an unknown parameter that has to be determined. Moreover, local non-monotonicities can introduce a sort of noise into the sequence of segments making up a pulse. Such noise has a different contribution depending on N and can potentially compromise the functioning of the algorithm. To avoid this, a noise threshold A, was introduced as another unknown parameter to be determined in order to prevent multiple toggling of the sign of the derivative regarding a single pulse, implementing a sort of hysteresis indeed. The choice of N and A, whose relation is explained in figure 5.10, is at the base of the algorithm, because it has to guarantee the detection of all pulses making up all force patterns, theoretically. Moreover, these values need to be optimum somehow: they have to be chosen accurately, high enough to allow the correct operation of the algorithm, but low enough to permit the feasibility of itself, in the sense that, reminding the embedded system context, too high ones may compromise the responsiveness, violating the real-time constraint and making the algorithm useless actually. The identification of the couple of values can be carried out in at least two ways: manually, by trial and error, but testing all the possible combinations is evidently time-consuming or automatically, letting the algorithm itself determining the best ones by performing a statistical analysis on the *Niceclick*

database, at the expense of an increased complexity, which is obviously the best option.

The flow chart of the algorithm, simplified through the imposition of $N = A$, is reported in figure 5.11. As in 5.12, its operation was tested out in MATLAB. However, in spite of analyzing the force patterns of the database, universal values for N and A were not found.

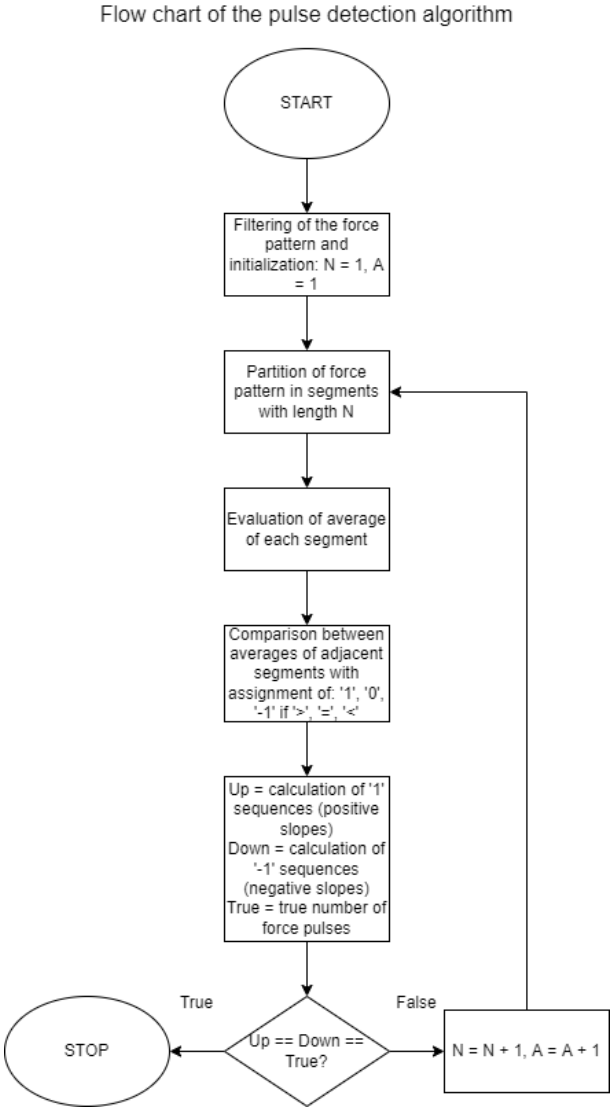

Figure 5.11: Flow chart of the pulse detection algorithm

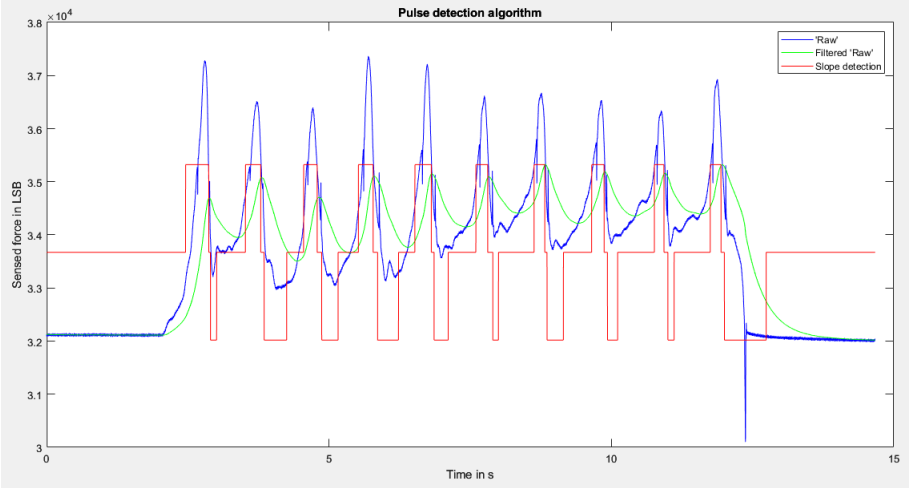

Figure 5.12: Example of operation of the pulse detection algorithm on a force pattern

Chapter 6

Haptic actuator development

The chapter focus on the development of the *Niceclick* transducer as an haptic actuator and on the issues which originally made difficult its proper functioning.

The first part introduces concepts of electromagnetism which are necessary to comprehend the set of problems relating haptic effect generation and their causes. Moreover, the original attempt to mitigate such issues, the demagnetization pulse, is presented along with its drawbacks. The second part is about a possible solution to these problems, implemented in hardware, based on a full H-bridge driving of the transducer. The third part presents an algorithm aimed at overcoming such issues, implemented in firmware.

It is important to note that, in the following, the *Niceclick* system is considered made up of a board and a single transducer and directly sensing position instead of force (indirectly obtained). Moreover, the transition from sensor to actuator mode is enabled.

6.1 Problems of the original haptic actuator

The *Niceclick* system working principles are based on concepts of electromagnetism. By construction, the transducer exhibits a ferromagnetic nature, so, in case of a magnetic field, it is possible to refer to the $B-H$ (or $M-H$) curve, the so-called "hysteresis loop" depicted in figure 6.1, in order to achieve a proper comprehension of what such property entails.

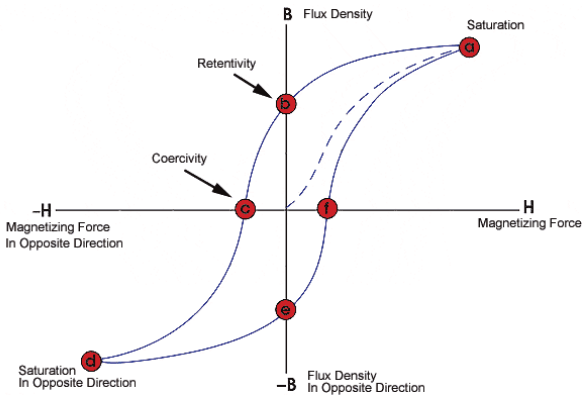

Figure 6.1: Hysteresis loop (courtesy of Iowa State University)

After the initial magnetization curve, which states that the magnetization moves from a null value to a saturating one thanks to the application of an increasing magnetic field, magnetization and demagnetization are not the same curve anymore, there is a sort of memory, an hysteresis, and the magnetization and demagnetization curves trace loops in which magnetization can assume any value, inside the main one. This means that the initial value of magnetization will almost never be equal to the final one as a result of a variation in the magnetic field (as a consequence of a variation in the current), being its initial and final values equal, because the alignment of the magnetic domains changes randomly. Such magnetic concepts, saturation and hysteresis, inherently characterize the *Niceclick* system and are responsible for part of its original misbehavior, in particular, when the transducer operates as an actuator for the generation of haptic effects.

In order to highlight such phenomena, a data acquisition system was arranged in order to collect information from the sensing equipment when the *Niceclick* is enabled to operate in both modes. Its evaluation board, powered by a DC power supply, was connected to a PC and the data, transmitted thanks to a CAN-USB adapter, were processed by CW-Analyzer in order to be displayed in real-time. When both operating modes are enabled, the *Niceclick* transducer works, normally, as a force sensor and turns into an haptic actuator if and only if a predefined force threshold is exceeded. So, an acquisition of the sensed force was carried out as reported in figure 6.2.

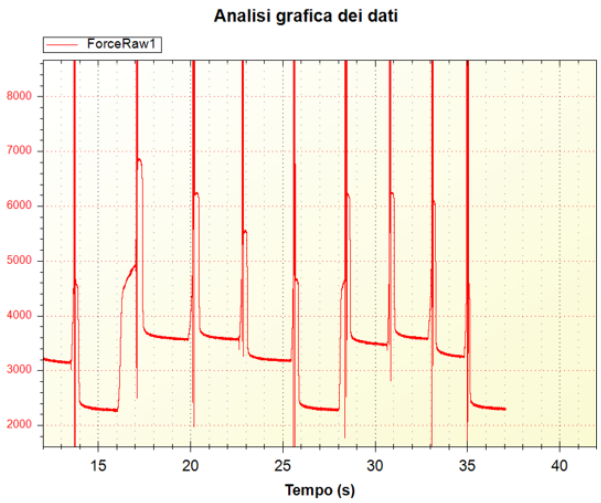

Figure 6.2: Data acquisition with haptic feedback generation enabled

The sections in which the curve, portraying the sensed force, is almost vertical represent the time intervals during which an haptic effect is generated: reminding the working principles of the *Niceclick* system, if the force overcomes the set threshold, toggling between sensor and actuator modes occurs, the switch responsible for the charging of the coil is driven by a high DC PWM signal, instead of a low DC one. As a consequence, the induced magnetic field is strong enough to attract the cursor to the stator, compressing the elastic element, the inductance increases, resonant frequency does the opposite and the voltage, to be converted by the ADC, saturates its input dynamic; conversely, the sections in which the curve is almost horizontal, rather than increasing or decreasing, identify the time intervals during which only the force sensing is carried out. That said, it is evident that the value measured after the generation of an haptic effect is

not necessarily equal to the one before, being applied force equal, but such variability can be also seen switching on and off the DC power supply and observing the resting position of the curve without applying any force. All that experimentally proves the effects on the *Niceclick* system of the magnetic concepts as above.

Such phenomena can not be neglected: intuitively, equal applied forces should imply the same sensed value, but this is not the case if an haptic feedback is provided in between. Such consideration leads to the problem which makes difficult the handling of the haptic feedback generation and which is caused, as such, by the implicit working principles of the system. This requires a detailed study of the actuating equipment, focused on the threshold management. The *Niceclick* system toggles between sensor and actuator modes of operation thanks to a programmable force threshold, which is not absolute, but rather relative. Especially, in order to determine if it is time or not to generate an haptic effect, two elements are needed:

- A sensed force value;
- A reference force value;

The first, defined as "Raw", is the force value measured by the system. The latter, called "Baseline", acts as a force value to be compared against. In particular, the comparison takes the shape of a difference between these two, the so-called "Delta". Toggling between modes takes place on the basis of a comparison between the "Delta", converted in firmware to a true force, and the threshold. The timing, given by the interaction between PWM and ADC, establishes the refreshing rate of the "Raw" value, while the "Baseline" one is updated thanks to a low pass filter, implemented in firmware and characterized by a high time constant, which performs a sort of average on the "Raw" and introduces a delay also, so its value takes time to experience a variation, but this allows the functioning of the mechanism: the "Baseline" does not react promptly to variations in the applied force, so it is not able to follow the evolution of the "Raw", causing the "Delta" to be not null.

For what concerns the reproduction of the mechanical push button effect through the *Niceclick* system, it is important to note that there are two thresholds in fact, one for each haptic feedback to be generated in order to fulfill this task: again, the "Click" and the "Clack" thresholds, with the first higher with respect to the latter, as in 5.9. Therefore, when a force, exceeding the first one, is applied to the *Niceclick*, the "Click" is released, but the haptic effect generation causes a step in the sensed force after it. Considering the time response of the "Baseline", with respect to the "Raw" one, this compromises the operation of the Moore's finite state machine devoted to the management of the haptic feedback generation, causing erroneously fast transitions among states, resulting in poor quality of the perceived effect or deadlocking in a state in which the exit condition can not be met physically. Such considerations find confirmation in experimental data: a series of acquisitions was carried out and the occurrence of these situations was revealed, like in figure 6.3.

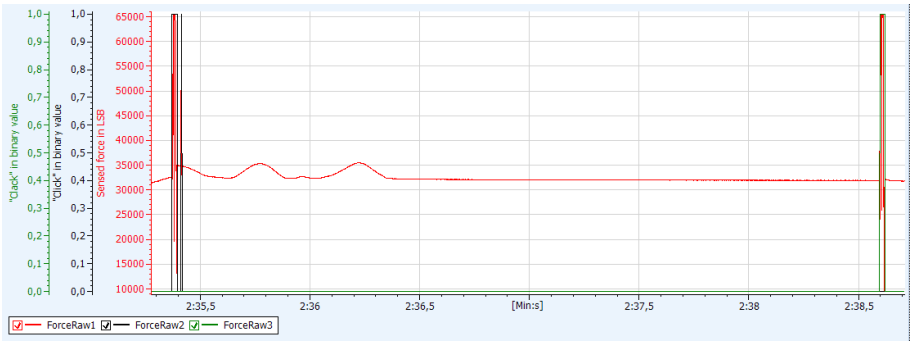

Figure 6.3: Erroneous generation of “Clack” due to deadlocking

In order to avoid such misbehavior, originally, the so-called “demagnetization pulse” was exploited. During the transition from actuator to sensor mode, a short-lasting high-DC PWM signal drives the switch. Theoretically, this current spike should force residual magnetization to decrease by resetting magnetic domains somehow. The results of such expedient are visible in figure 6.4 and can be compared against 6.2.

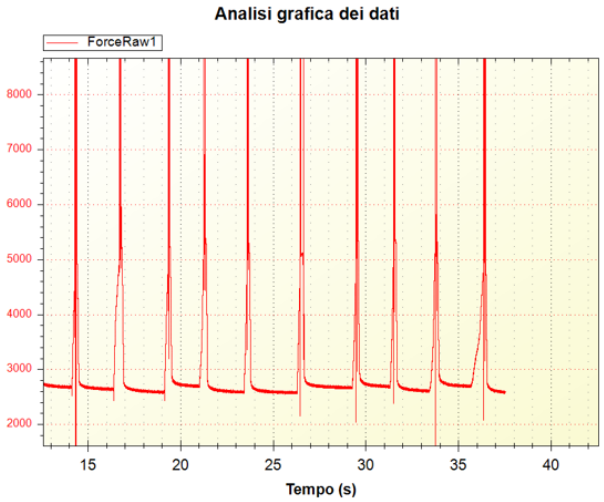

Figure 6.4: Data acquisition with haptic feedback generation enabled and demagnetization pulse

Although, the solution is characterized by some drawbacks:

- It causes the introduction of electronic noise;
- It can be detected by a user;

So, in order to cope with these problems, further solutions were investigated, but, before going on, it is important to point out that by employing pure iron, in place of soft, these phenomena would have a decisively lower overall effect on the system. However, this type of iron is difficult to work and, moreover, it is expensive, so such option was discarded.

6.2 Hardware solution: full H-bridge

To try to counteract the effects of these innate phenomena, but with acceptable consequences, an hardware solution was explored. So far, the *Niceclick* transducer was driven thanks to a circuit which implements an half-bridge actually (taking into account also the high-side switch that implicitly connects coil and supply voltage). In both low-side switch configurations and operating modes, the current in the inductor, which models the *Niceclick* transducer, has always the same direction and magnetic hysteresis and saturation occur.

In order to attempt to cope with such problems, the driving circuit was changed and, in particular, other high-side and low-side switches were added, obtaining a full H-bridge driving configuration. The schematic is reported in figure 6.5.

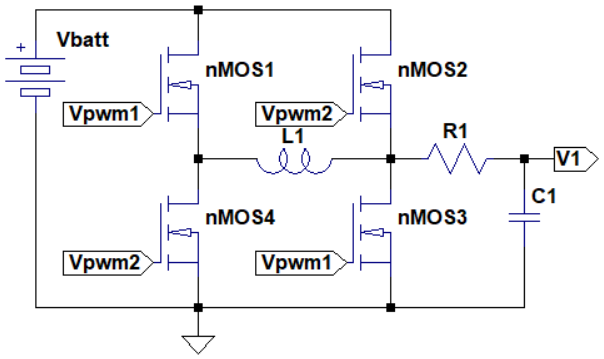

Figure 6.5: Schematic of the full H-bridge

Such architecture permits the control of the direction of the current in the coil. This allows the change in part of the operating principle of the *Niceclick* system, in particular, toggling the operating mode, the current direction is toggled too by managing the branches of the bridge. Thus, hysteresis should be somehow compensated theoretically due to the fact that the two modes of operation are characterized by an opposite direction of the current in the inductor.

Firmware also has to be altered in order to manage the switches for the successful implementation of such driving scheme, especially, to avoid short circuits from supply voltage to ground which may lead to hardware faults.

However, tests, accomplished thanks to a standard data acquisition system, made of a PC, a CAN-USB adapter and CW-Analyzer for the real-time visualization of data, revealed that such solution was not working properly and magnetic hysteresis was still not negligible, so it was discarded.

6.3 Firmware solution: algorithm for hysteresis compensation

Carrying out the development of the *Niceclick* as an haptic actuator, a set of problems came to light. In particular, they became clearly visible accomplishing several data acquisitions. An in-depth analysis of the latter allowed the identification of the cause of such issues: misbehavior of the original finite state machine responsible for the managing of the haptic feedback generation due to the effect of magnetic hysteresis. This is present by default, due to the combination of

magnetic properties of the material and working principles of the system, and so has to be taken into account somehow.

Since the fact that both solutions tried up to now, the demagnetization pulse and the full H-bridge driving, solve such problems in part only or do not solve them at all, an algorithmic solution was explored. It is important to note that it does not get rid of magnetic hysteresis, but it only performs a sort of compensation. The idea at the base of this algorithm comes from the detailed study of the acquisition of a true force pulse, as depicted in figure 6.6.

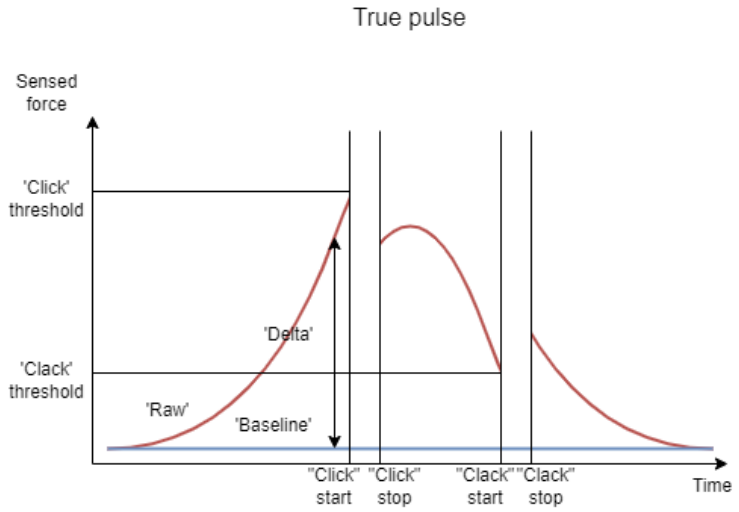

Figure 6.6: True pulse sensed by the Niceclick system

Supposing that the “Baseline” value, the reference with respect to which the force variation, the “Delta”, is calculated, coincides with the “Raw” one, in sensor mode, the latter changes as the applied force. When the first higher force threshold is reached, the *Niceclick* system transitions from sensor to actuator mode and the “Click” is provided to the user. This operation lasts for a certain amount of time and then, the system goes back to sensor mode, but the current peak, applied for the generation of the haptic effect, has permanently changed the magnetization state, causing the “Raw” to exhibit a step, up or down. The latter should then decrease up to the second lower force threshold, before the generation of the “Clack”. However, the “Baseline” has not got enough time to change and, as for the “Click” one, the “Clack” threshold also was predefined. This static nature makes the system unable to cope with a dynamic event like an hysteresis step. In fact, it normally induces misbehavior of the FSM, because such steps can cause the accidental meeting of the two thresholds or, conversely, can make them impossible to be reached. After the “Clack”, another hysteresis step takes place, possibly causing problems with the generation of the next “Click” and so on, making the force pulse split in three sections actually, as in 6.6. So, in order to cope with such problems a proper managing of the force thresholds and/or the “Baseline” is mandatory. This need implies the modification of the related FSM also.

In spite of magnetic hysteresis, it is possible to generate both haptic feedbacks at the right time thanks to the manipulation of such values, “Raw”, “Baseline” and so “Delta” through trivial operations, like extrapolation and offset. In broad terms, by doing an extrapolation, based on the derivative of the “Raw” value, during the generation of the “Click” and so in actuator mode, the last “Delta” one, before going back to sensor mode, can be estimated. Then, the first “Delta” immediately after is evaluated and by applying an offset to the “Baseline”, given by the difference

between these last two, the “Clack” can be correctly provided finally, as shown in figure 6.7.

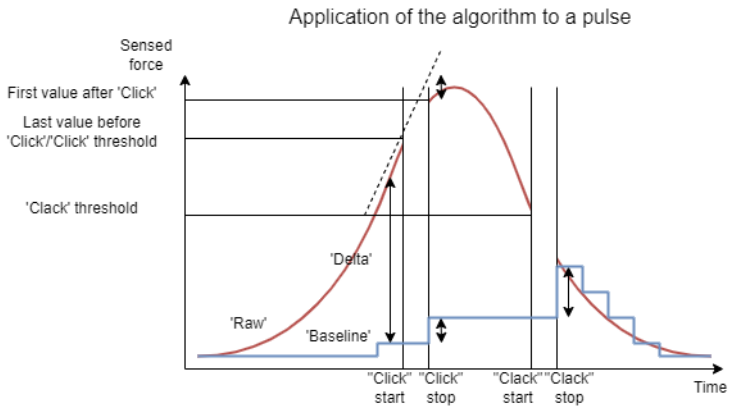

Figure 6.7: Example of operation of the algorithm for hysteresis compensation

The flow chart of the algorithm for hysteresis compensation is reported in figure 6.8.

6.3.1 Hysteresis-insensitive Moore’s FSM

For what concerns the implementation, an alternative FSM, needed for the management of the haptic effect generation, was designed and such algorithm was integrated in the latter. It is made of four states which make up a loop. The states are OFF, CLICK, ON, CLACK and the corresponding state diagram is depicted in figure 6.9.

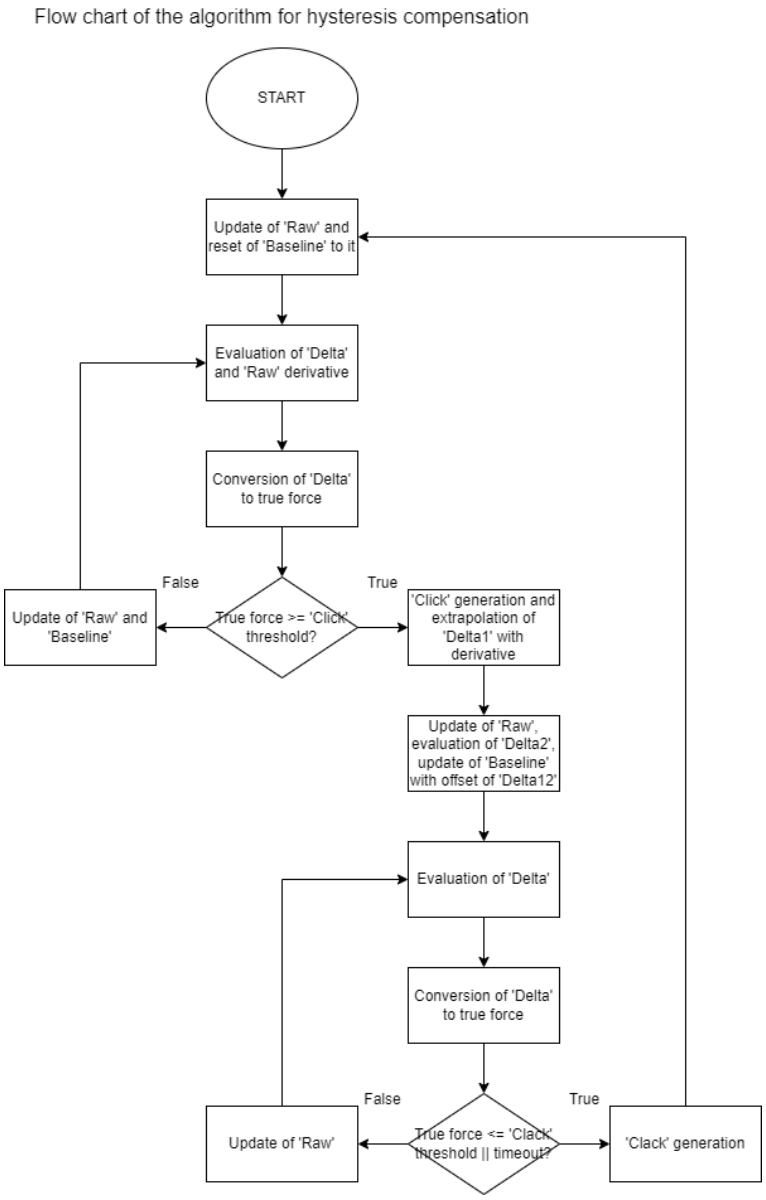

Figure 6.8: Flow chart of the algorithm for hysteresis compensation

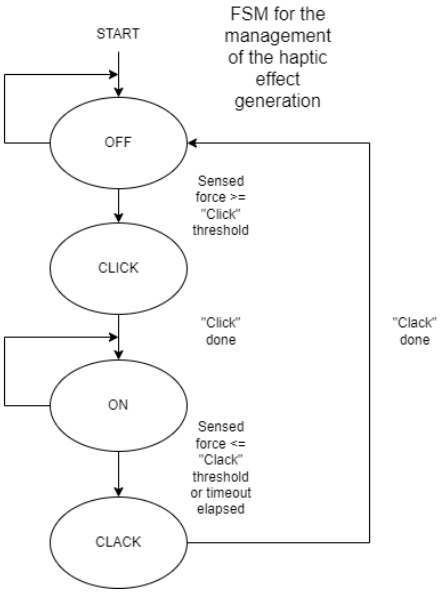

Figure 6.9: State diagram of the hysteresis-insensitive FSM

OFF state

The OFF one is the state at the starting of the FSM and immediately after the CLACK. The operations that have to be carried out are:

1. “Baseline” reset (and update);
2. “Delta” calculation;
3. “Raw” derivative calculation

First, the “Baseline” value has to be reset in order to be equal to the sensed force one, the “Raw”, both at startup and transition from CLACK state, because the reference value require an initialization or re-initialization after the magnetic hysteresis step to guarantee the consistency of the (static) “Click” threshold. Moreover, as long as the state does not change, its value should be updated accordingly, not so fast, in order to allow the detection of an applied force, but not so slow too, in order to compensate the effects of influence quantities.

Second, the “Delta”, the difference between “Raw” and “Baseline”, has to be computed because the state exit condition is the passing of the “Click” threshold of such value, converted to a true force.

Third, the “Raw” derivative has to be calculated as the difference between two “Raw” values, but particular attention has to be paid to noise, because it can significantly condition the computation. The calculation of the derivative is critical also because the latter is used for the extrapolation. In order to exit the state and transition to the CLICK one, the “Click” threshold has to be met.

CLICK state

The CLICK one is the state immediately after the OFF. The operations that have to be carried out are:

1. Flyback diode enabling;
2. "Click" generation with "Delta" extrapolation;
3. Flyback diode disabling;

First, the flyback diode has to be enabled in order to allow proper generation of the haptic effect and protection of the hardware.

Second, a high-DC PWM signal has to drive the low-side switch, causing a high current in the coil such as to attract the cursor from its resting position to the stator.

Third, starting from the last available "Delta" value before the CLICK state, the one before the ON state has to be extrapolated resorting to the last available "Raw" derivative. The justification of such operation is verifiable experimentally: in fact, observing the data acquisitions of the *Niceclick* database, it is possible to note that the sensed force after the "Click" tends to be higher with respect to the one before it.

Fourth, the flyback diode has to be disabled. After the generation of the haptic feedback, the transition to the ON state takes place.

ON state

The ON one is the state after the CLICK. The operations that have to be carried out are:

1. "Baseline" reset;
2. "Delta" calculation;

First, the "Delta" value after the CLICK state has to be subtracted from the extrapolated one. Such difference consists of an offset which has to be added to the "Baseline" value in order to compensate the step in the sensed force caused by magnetic hysteresis. This guarantees the consistency of the "Clack" threshold, in particular, it ensures that the "Delta", converted to a true force and then compared against the static "Clack" threshold, is calculated with respect to a proper "Baseline", because if the "Raw" exhibits such phenomenon, also the "Baseline" has to be updated accordingly in order to obtain a coherent "Delta" in the following.

Second, the "Delta" value has to be computed (and converted to a true force) in order to check if the exit condition, which is the excess of the "Clack" threshold, is met. Moreover, in order to be sure not to run into a deadlock, a timeout has to be inserted. In order to exit the state and transition to the CLACK one, the "Clack" threshold or the timeout has to be met.

CLACK state

The CLACK one is the state after the ON. The operations that have to be carried out are:

1. Flyback diode enabling;
2. "Clack" generation;
3. Flyback diode disabling;

First, the flyback diode is enabled in order to allow proper generation of the haptic effect and protection of the hardware.

Second, a high-DC PWM signal has to drive the low-side switch, causing a high current in the coil such as to attract the cursor from its resting position to the stator. Third, the flyback diode has to be disabled. After the generation of the haptic effect, the transition to the OFF state takes place.

FSM applied implementation

In order to move from the theoretical algorithm to the applied implementation, some details have to be specified. However, it is important to note that for the sake of simplicity and in agreement with the principle of code reusability, as much as possible of the original code was kept.

For what concerns the OFF state, particular attention has to be paid to the managing of the "Baseline". Its value is obtained thanks to a filter implemented in software. The specifications, regarding the different behavior of the "Baseline" with respect to applied force or influence quantities, suggest that its parameters should be changed on the fly or two different filters should be used, because two different time constants are needed. In particular, a low pass filter, characterized by a high time constant and, as such, insensitive to fast changes, is employed to allow the detection of the applied force and, instead, in order to reproduce the behavior of a low (actually zero) time constant one, so sensitive to fast changes, the "Baseline" value is kept equal to the "Raw" one. Since the fact that, in such state, the derivative is calculated continuously, its sign is exploited to choose the right filter to update the "Baseline" value, because, theoretically, if the "Raw" derivative is positive, a force is beginning to be applied, so, in order to allow the force threshold mechanism to work, the "Baseline" has not to vary or to slowly vary at most, so the need of a high time constant filter. Conversely, if the derivative is negative, a force is ending to be applied, so the "Baseline" value can be kept equal to the "Raw" one and no filter is needed. In a case, the "Click" has to be provided to the user, in the other, the "Clack" was just generated. It is important to note that the evaluation of the derivative takes place in the ADC callback instead of the FSM, because the first is characterized by a lower period with respect to the latter, so it is computed on a shorter time interval and refreshed more frequently as a consequence. However, this causes noise to have a higher effect on the calculation, therefore "Raw" derivative is given as the difference between the last and first values of a finite-length circular buffer, handled as a FIFO queue, and then further filtered thanks to a moving average filter. Nevertheless, for what concerns the latter, particular attention has to be paid because of fixed point operation of the embedded system. Especially, reminding how such a filter works, there are two complementary coefficients stating the percentages of the two inputs which contribute to the output. In particular:

$$Output = New \cdot Input + Old \cdot Output$$

Where $New + Old = 1$. So, if coefficients differ much, one of the two inputs does not have any effect on the output, because one of the two addends is null due to numerical errors caused by fixed point operation indeed. In order to avoid such situation in all probability, the moving average filter has to be implemented differently: a finite-length buffer is filled with samples and its output is the sum of the actual samples divided by the number of such samples.

In the CLICK state, the last sensed force before transitioning to the ON state is estimated thanks to an extrapolation, based on the last derivative. Such extrapolation lasts for the duration of the haptic feedback generation, determined by the parameters which characterize the haptic effect itself and, in fact, can be customized. In particular, an haptic effect consists of a variable sequence, made of PWM values applied with a predefined period, which can be repeated multiple

times with a waiting time in-between. Therefore, the extrapolation of the "Delta" value takes the form of:

$$Delta_{new} = dRaw \cdot (Period_{PWM} \cdot N_{PWM} \cdot N_s + Time_{waiting} \cdot (N_s - 1)) + Delta_{old}$$

Where:

- $Delta_{new}$ is the final "Delta" after extrapolation;
- $dRaw$ is the "Raw" derivative;
- $Period_{PWM}$ is the period at which the PWM values are applied;
- N_{PWM} is the number of PWM values used;
- N_s is the number of PWM sequences used;
- $Time_{waiting}$ is the waiting time between two PWM sequences;
- $Delta_{old}$ is the initial "Delta" before extrapolation;

In the ON state, the "Baseline" value has to be updated as a result of the step in sensed force caused by magnetic hysteresis. Especially, an offset, calculated as the difference between the first "Delta" value in the ON state and the last one extrapolated in the CLICK state, is applied to it in order to guarantee the consistency between "Raw" and "Baseline", so of the "Delta". The operation of the algorithm for hysteresis compensation is depicted in figures 6.11 and 6.10.

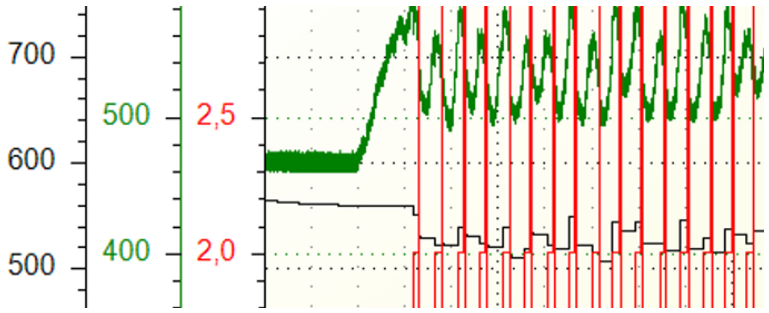

Figure 6.10: Data acquisition exhibiting the operation of hysteresis-insensitive FSM

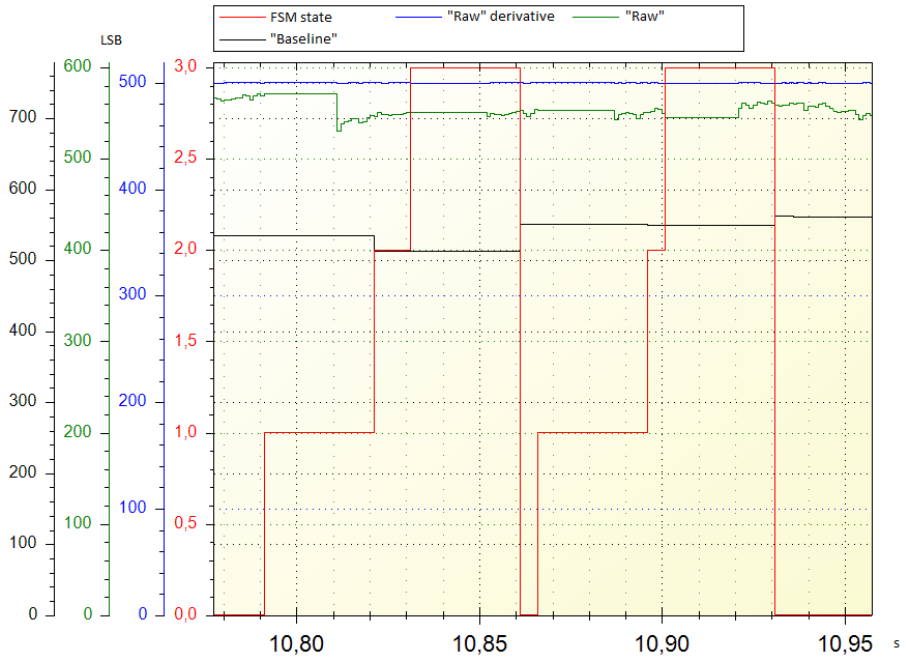

Figure 6.11: Detail of the same data acquisition

Chapter 7

Conclusion

The resulting *Niceclick* system is characterized by an improved force sensing capability thanks to a different management of both hardware and firmware which led to a better exploitation of the microcontroller functionalities. A series of tests revealed or confirmed such characteristic. A group of models intended to describe or predict its behavior, with a given set of parameters, is available to carry out simulations even without interacting with it directly. A database, recording force sensing acquisitions, constitutes the point of departure for data analysis and algorithm design, like the pulse detection one, in order to allow its development. An algorithm, implemented in firmware, solves the problems, limiting its haptic feedback generation capability, caused by its electromagnetic working principles, magnetic hysteresis first and foremost, even without modifying the hardware. However, even though the *Niceclick* system is operational, it is still a work-in-progress also and there is room for improvement indeed: the capability of handling multiple transducers has to be developed. Attempts were made, but the managing of the interaction between the PWM and the SAR ADC turned out to be a non-trivial problem, due to limited hardware resources, timing constraints and components tolerances.

Moreover, its models proved to be accurate only to a certain point, so an enhancement has to be done, especially for what concerns the *Niceclick* transducer model. Therefore, the thesis discussed the comprehensive development of the *Niceclick* system. It encompassed the development of both hardware and firmware, but also their testing and verification. It demonstrated the validity of the project thanks to its experimental nature: different solutions were applied in order to cope with problems related to the youthfulness and novelty of the basic idea behind it. Nevertheless, it proved to be a promising technology, solving the most important issues of other competing ones. Finally, as anticipated, the *Niceclick* system was integrated into an automotive steering wheel as a demonstration.

Appendix

MATLAB code

Niceclick system model

```
% Initialization
% Timing
% Due to system architecture:
% Duration of a cycle = timing of PWM counter (Ton: switch on, Toff: switch
% off) + timing of SAR ADC (Tacq: acquisition, Tconv: conversion)
Ton = (Prescaler + 1) * (Compare + 1) / f_clock;
Tcycle = (Prescaler + 1) * (Period + 1) / f_clock + Tacq + Tconv;
T_PWM = Tcycle - Tacq - Tconv;
Toff = T_PWM - Ton;
t_inf = Tcycle - Ton;
Npoint = 1e3;
t = linspace(0, t_inf, Npoint);
% I_L1 and V_C1 with initial and final conditions
% Number of cycles to simulate
Ncycle = 5;
L1_0 = 1.0 * L1;
% Saturation current
Is = 0.09;
I_L1_0test = zeros(1, Npoint);
dI_L1_0test = zeros(1, Npoint);
V_C1_0test = zeros(1, Npoint);
I_L1 = zeros(1, Npoint);
dI_L1 = zeros(1, Npoint);
V_C1 = zeros(1, Npoint);
Rtester = 49.8;
Req = Rtester + Zsw;
ReqL = Rtester + Zsw;
ReqC = (1 / Rtester + 1 / Zsw) ^ -1 + Z1;
% Saturation coefficient
a = 1;
% Time interval from 0 s to Ton s
t_test = linspace(0, Ton, Npoint);
figure(1)
hold on
for i = 0 : Ncycle - 1
% Switch on
for j = 1 : Npoint
    % Inductance saturation taken into account with tanh (non-linear inductor)
    if j == 1
        L1 = L1_0 * Is * tanh((I_L1(end) / Is) ^ a) ^ (1 / a) / I_L1(end);
    else
        L1 = L1_0 * Is * tanh((I_L1_0test(j - 1) / Is) ^ a) ^ (1 / a) / I_L1_0test(j - 1);
    end
    if isnan(L1)
        L1 = L1_0;
    end
    % Evolution in time of first order circuits
    % (initial conditions for second order circuit are final conditions
    % for first order circuits and vice versa)
    I_L1_0test(j) = (I_L1(end) - Vcc / Req) * exp(-t_test(j) / (L1 / ReqL)) + Vcc / Req;
    dI_L1_0test(j) = -(I_L1(end) - Vcc / Req) * exp(-t_test(j) / (L1 / ReqL)) / (L1 / ReqL);
    V_C1_0test(j) = (V_C1(end) - Vcc * Zsw / Req) * exp(-t_test(j) / (ReqC * C1)) + Vcc * Zsw / Req;
end
Acid1 = I_L1(end);
Acid2 = L1;
% Switch off
for j = 1 : Npoint
if j == 1
    % Inductance saturation taken into account with tanh (non-linear inductor)
    L1 = L1_0 * Is * tanh((I_L1_0test(end) / Is) ^ a) ^ (1 / a) / I_L1_0test(end);
else
    L1 = L1_0 * Is * tanh((I_L1(j - 1) / Is) ^ a) ^ (1 / a) / I_L1(j - 1);
end
if isnan(L1)
    L1 = L1_0;
end
% Time interval from Ton s to Ton s
```

```
% Evolution in time of second order circuit
% (initial conditions for second order circuit are final conditions
% for first order circuits and vice versa)
I_L1inf = 0;
A1 = I_L1_0test(end);
Ztot = 1.15 * (Rtester + Z1);
Alpha = Ztot / (2 * L1);
w0 = 1 / sqrt(L1 * C1);
Beta = sqrt(w0 ^ 2 - Alpha ^ 2);
A2 = (dI_L1_0test(end) + Alpha * A1) / Beta;
A = sqrt(A1 ^ 2 + A2 ^ 2);
Phi = -atan(A2 / A1);
I_L1(j) = exp(-Alpha * t(j)) .* (A1 * cos(Beta * t(j)) + A2 * sin(Beta * t(j))) + I_L1inf;
dI_L1(j) = -Alpha * exp(-Alpha * t(j)) .* (A1 * cos(Beta * t(j)) ...
    + A2 * sin(Beta * t(j)))+ Beta * exp(-Alpha * t(j)) .* (-A1 * sin(Beta * t(j)) + A2 * cos(Beta * t(j)));
V_C1(j) = (A * exp(-Alpha * t(j)) .* (Beta * sin(Beta * t(j) + Phi) ...
    - Alpha * cos(Beta * t(j) + Phi))) / (Alpha ^ 2 + Beta ^ 2) / C1 + V_C1_0test(end) + Vcc;
% Maximum of V_C1
t_max = (pi / 2 - Phi) / Beta;
t_max_cycle = t_max + Ton;
t_min_cycle = t_max_cycle + pi / Beta;
end
V_C1zeroing = V_C1(1);
V_C1 = V_C1 - V_C1zeroing + V_C1_0test(end);
% Plot of I_L1 and V_C1
plot(t_test + i * Tcycle, I_L1_0test, 'c', t_test + i * Tcycle, V_C1_0test, 'm', ...
    t + Ton + i * Tcycle, I_L1, 'b', t + Ton + i * Tcycle, V_C1, 'r')
xline(T_PWM + i * Tcycle)
xline(Tcycle + i * Tcycle)
yline(0)
title('Current and voltage of the first stage')
xlabel('Time in s')
ylabel('I_1 in A and V_1 in V')
end
yline(Vcc + V_C1_0test(end))
yline(Vcc + V_C1_0test(end) + Vf_r)
legend('I_1 (switch off)', 'V_1 (switch off)', 'I_1 (switch on)', 'V_1 (switch on)')
% Voltage at ADC input (before S/H) after high-pass filter
I_L1(end) = Acid1;
V_ADCtest = (A * exp(-Alpha * Toff) .* (Beta * sin(Beta * Toff + Phi) ...
    - Alpha * cos(Beta * Toff + Phi))) / (Alpha ^ 2 + Beta ^ 2) / C1 - V_C1zeroing + V_C1_0test(end);
V_ADC = (sqrt(((I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) + Vcc / Req) ^ 2 ...
    + ((-(I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) / (Acid2 / ReqL) ...
    + Ztot / (2 * L1) * ((I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) ...
    + Vcc / Req)) / Beta) ^ 2) * exp(-Alpha * Toff) * (Beta * sin(Beta * Toff ...
    - atan(((-(I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) / (Acid2 / ReqL) ...
    + Ztot / (2 * L1) * ((I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) ...
    + Vcc / Req)) / Beta) / ((I_L1(end) - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) ...
    + Vcc / Req))) - Alpha * cos(Beta * Toff - atan(((-(I_L1(end) - Vcc / Req) ...
    * exp(-Ton / (Acid2 / ReqL)) / (Acid2 / ReqL) + Ztot / (2 * L1) * ((I_L1(end) ...
    - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) + Vcc / Req)) / Beta) / ((I_L1(end) ...
    - Vcc / Req) * exp(-Ton / (Acid2 / ReqL)) + Vcc / Req))))) / (C1 * (Alpha ^ 2 + Beta ^ 2));
V_ADC = V_ADC - V_C1zeroing + V_C1_0test(end);
```

Pulse detection algorithm

```
% Enter database directory
cd NC_measurement_database\
% List all elements
Files = dir;
Files = Files(3 : end);
Nelements = numel(Files);
% Coefficient of SW LPF
K = 0.003;
N = zeros(1, Nelements);
A = zeros(1, Nelements);
for x = 1 : Nelements
    % Enter database subdirectory
    cd(Files(x).name)
    % Load acquisition file data
    load(dir('DataAcquisitionFile.mat').name, 'ForceRaw1');
    Npoints = length(ForceRaw1);
    Filtered_ForceRaw1 = zeros(1, Npoints);
    % Starting value of SW LPF
    Filtered_ForceRaw1(1) = ForceRaw1(1);
    for i = 2 : Npoints
        % SW LPF in action
        Filtered_ForceRaw1(i) = K * ForceRaw1(i) + (1 - K) * Filtered_ForceRaw1(i - 1);
    end
    % Search for maxima in filtered data
    Npulses_up = sum(islocalmax(Filtered_ForceRaw1, 'MinProminence', 100));
    % Number of maxima is supposed equal to number of minima, which are more difficult to detect
    Npulses_down = Npulses_up;
    % Starting value of number of samples to average
    Ntest = 0;
    % Simplification: A = N
    Atest = Ntest;
    Nsquare_pulses_up = 0;
    Nsquare_pulses_down = 0;
    % While value of Ntest < 1000 and number of positive or negative pulses calculated thanks to Ntest
    % being different from the number of maxima or minima previously calculated
    while Ntest < 1000 && (Nsquare_pulses_up ~= Npulses_up || Nsquare_pulses_down ~= Npulses_down)
        Ntest = Ntest + 1;
        Atest = Ntest;
        Samples = zeros(1, Ntest);
```

```
        Mean = zeros(1, Npoints);
        Ternary = zeros(1, Npoints);
        j = 1;
        k = 1;
        for i = 1 : Npoints
            % Loading of samples to average
            Samples(j) = ForceRaw1(i);
            j = j + 1;
            % If there are Ntest samples, calculate mean
            if j > Ntest
                j = 1;
                Mean(k : k + Ntest - 1) = mean(Samples);
                % If there are at least 2 averages, compare calculated average to previous one
                if k > Ntest
                    if Mean(k + Ntest - 1) > Mean(k - 1) + Atest
                        Ternary(k : k + Ntest - 1) = 1;
                    elseif Mean(k + Ntest - 1) < Mean(k - 1) - Atest
                        Ternary(k : k + Ntest - 1) = -1;
                    else
                        Ternary(k : k + Ntest - 1) = 0;
                    end
                end
                k = k + Ntest;
            end
        end
        % Initialization to compensate last unwanted increment
        Nsquare_pulses_up = -1;
        y = 1;
        while y <= Npoints
            % While value is different from 1 it isn't a positive pulse
            while y <= Npoints && Ternary(y) ~= 1
                y = y + 1;
            end
            % While value is equal to 1 it is a positive pulse
            while y <= Npoints && Ternary(y) == 1
                y = y + 1;
            end
            % End of positive pulse
            Nsquare_pulses_up = Nsquare_pulses_up + 1;
        end
        % Initialization to compensate last unwanted increment
        Nsquare_pulses_down = -1;
        y = 1;
        while y <= Npoints
            % While value is different from -1 it isn't a negative pulse
            while y <= Npoints && Ternary(y) ~= -1
                y = y + 1;
            end
            % While value is equal to -1 it is a negative pulse
            while y <= Npoints && Ternary(y) == -1
                y = y + 1;
            end
            % End of negative pulse
            Nsquare_pulses_down = Nsquare_pulses_down + 1;
        end
    end
    % Store value of Ntest which exited while
    N(x) = Ntest;
    A(x) = N(x);
    % Exit subdirectory
    cd ..\
end
% Exit database directory
cd ..\
% Print mean and standard deviation of N/A
Average_N_A = mean(N)
Standard_deviation_N_A = std(N)
```

Bibliography

[1] S. J. Breitschaft, S. Clarke, C. C. Carbon, "A Theoretical Framework of Haptic Processing in Automotive User Interfaces and Its Implications on Design and Engineering," *Frontiers in Psychol.*, vol. 10, no. 1470, Jul. 2019, doi: 10.3389/fpsyg.2019.01470. [Online]. Available: `https://www.frontiersin.org/articles/10.3389/fpsyg.2019.01470/full` (cit. on p. 8)

[2] Thomas. "All About Force Sensors." thomasnet.com. `https://www.thomasnet.com/articles/instruments-controls/all-about-force-sensors/` (cit. on pp. 9-13)

[3] Precision Microdrives. "Eccentric Rotating Mass Vibration Motors – ERMs." precisionmicrodrives.com. `https://www.precisionmicrodrives.com/eccentric-rotating-mass-vibration-motors-erms` (cit. on pp. 13-14)

[4] Precision Microdrives. "AB-004: Understanding ERM Vibration Motor Characteristics." precisionmicrodrives.com. `https://www.precisionmicrodrives.com/ab-004` (cit. on pp. 13-14)

[5] Precision Microdrives. "Linear Resonant Actuators – LRAs." precisionmicrodrives.com. `https://www.precisionmicrodrives.com/linear-resonant-actuators-lras` (cit. on p. 15)

[6] Precision Microdrives. "AB-020: Understanding Linear Resonant Actuator Characteristics." precisionmicrodrives.com. `https://www.precisionmicrodrives.com/ab-020` (cit. on p. 15)

[7] S. Rao, "High-Definition Haptics: Feel the Difference!," *Analog Appl. J.*, 3Q 2012. [Online]. Available: `https://www.ti.com/lit/pdf/slyt483` (cit. on pp. 16-17)

[8] *NC-C1012-12V Datasheet*, 2nd ed., Trama Eng., Alba, CN, Italy, 2021. (cit. on pp. 18-19)

[9] R. Perfetti, *Circuiti Elettrici*, 1st ed., Italy: Zanichelli, 2003, ch. 7-8. (cit. on pp. 44-48)

Acknowledgements

Heartfelt thanks go to my family, especially my parents and grandparents to which such work is dedicated, for their endless support and the opportunities they offered me.
I would like to thank my supervisors for their willingness and, in particular, TRAMA ENGINEERING that trusted and hosted me, giving a big chance to grow, because they made possible all that.
Last, but not least, I would like to express my gratitude to the people which I met and the experiences which I lived during my journey at the Polytechnic university of Turin: they made me who I am and laid the foundations for who I will be.